Natural Language Processing

M1 Data & IA -Nexa Digital School

By: **Naima OUBENALI**

Disclaimer

- This Course is an introduction to Natural Language Processing tools and models. It is by no means exhaustive.
- It contains the essential elements to get you started and ready for the hands-on project.
- To fit everything into a small timeframe, I will be obliged to simplify some aspects.
- I encourage you to read NLP books/ articles, and check out online courses to dig deeper.

COURSE OUTLINE

Introduction to NLP

Definition, Importance, History, Applications

01

Basic concepts of NLP

Text Preprocessing, Tokenization, Stemming, Stop Words, POS tagging, NER.

02

Advanced Machine Learning and Deep Learning for NLP

Models, Embeddings,RNNs, LSTM, CNNs, Attention Mechanisms, Transfer Learning .

03

04

Advanced NLP techniques & Applications

Sentiment Analysis, Text categorization, Machine Translation, QA answering, Chatbots.

05

Conclusion & hands-on Project

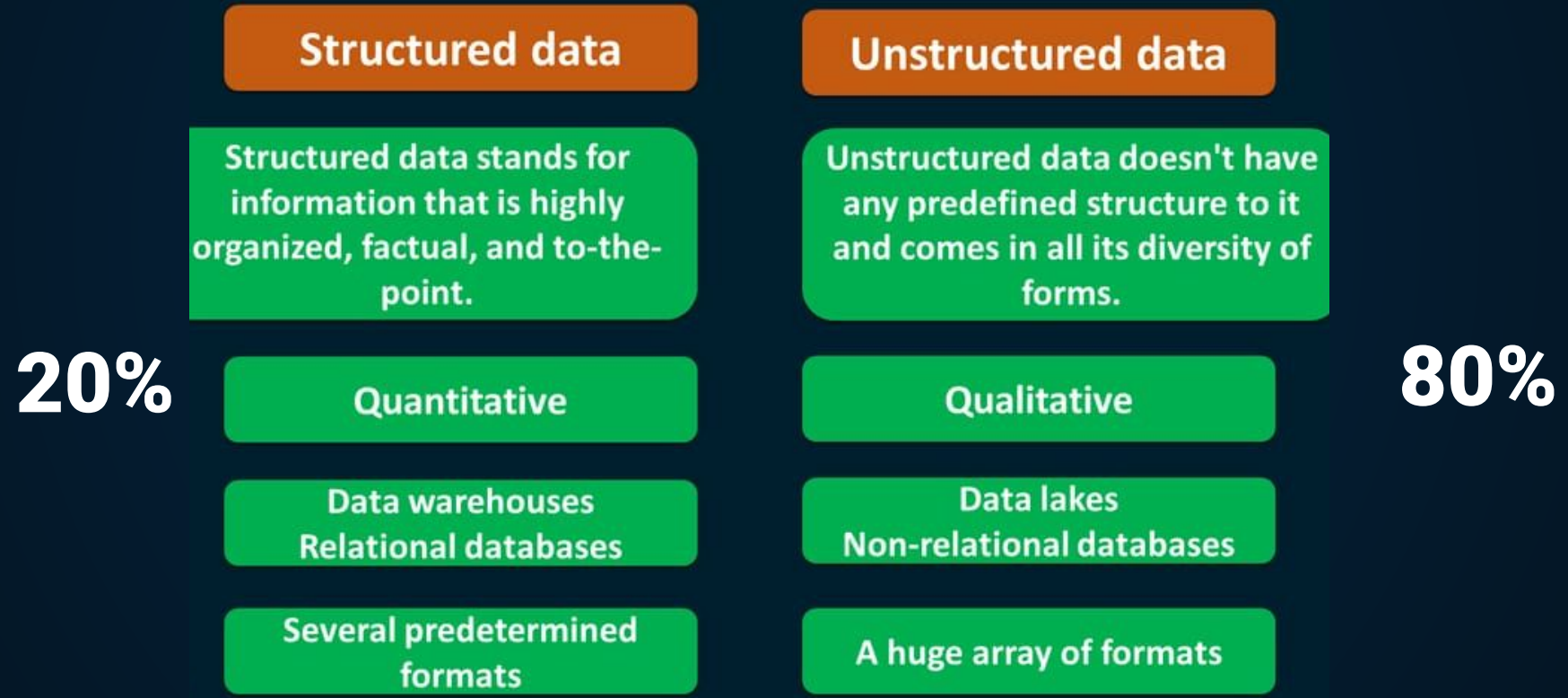
Text Mining

Structured vs Unstructured data

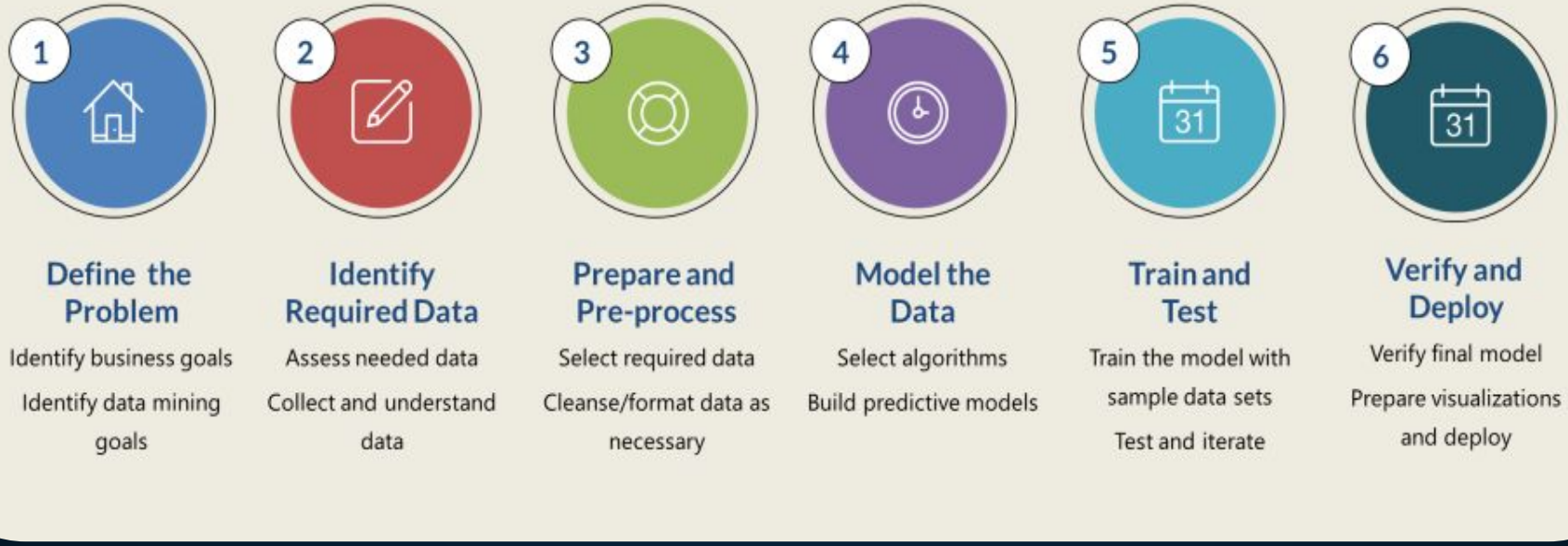
UNSTRUCTURED DATA

STRUCTURED DATA

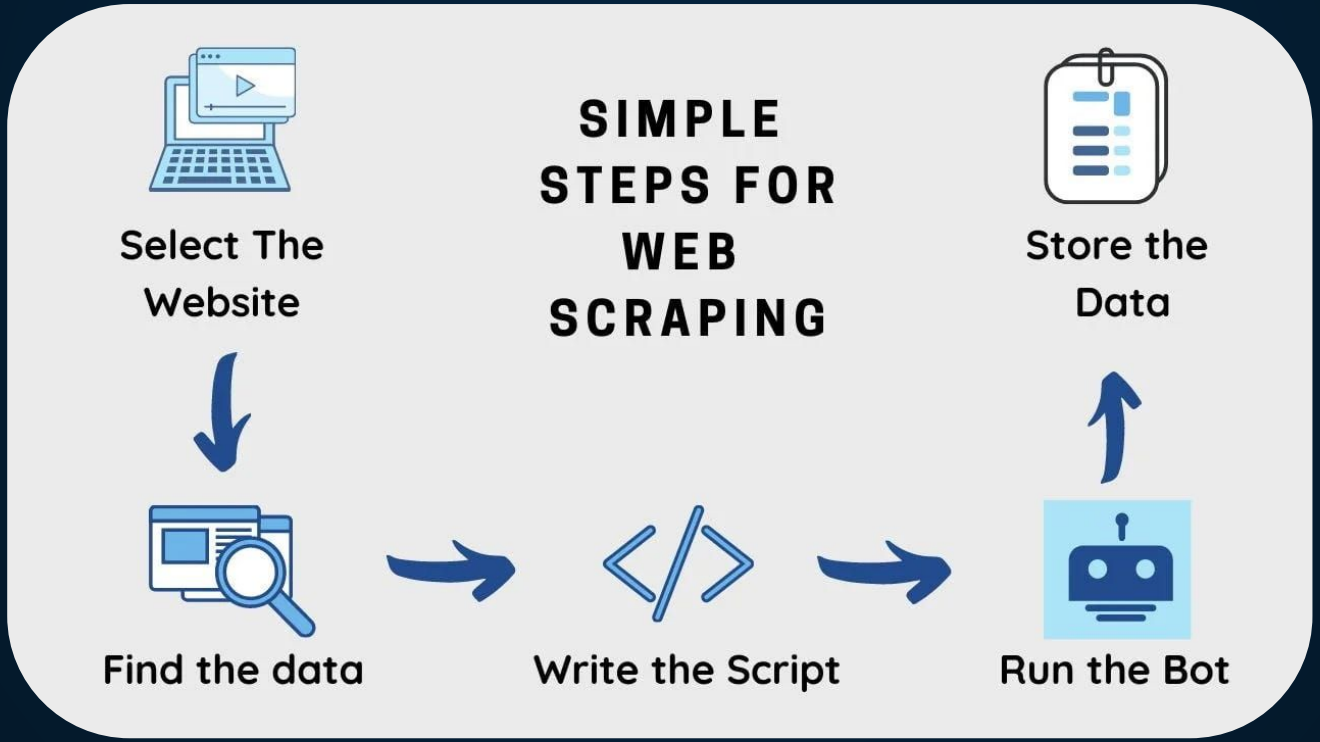
Structured vs Unstructured data



Data Mining


1
Define the Problem
Identify business goals
Identify data mining goals
2
Identify Required Data
Assess needed data
Collect and understand data
3
Prepare and Pre-process
Select required data
Cleanse/format data as necessary
4
Model the Data
Select algorithms
Build predictive models
5
31
Train and Test
Train the model with sample data sets
Test and iterate
6
31
Verify and Deploy
Verify final model
Prepare visualizations and deploy

Data Mining


SIMPLE STEPS FOR WEB SCRAPING
Select The Website
Find the data
Write the Script
Run the Bot
Store the Data

Textual Data Scraping

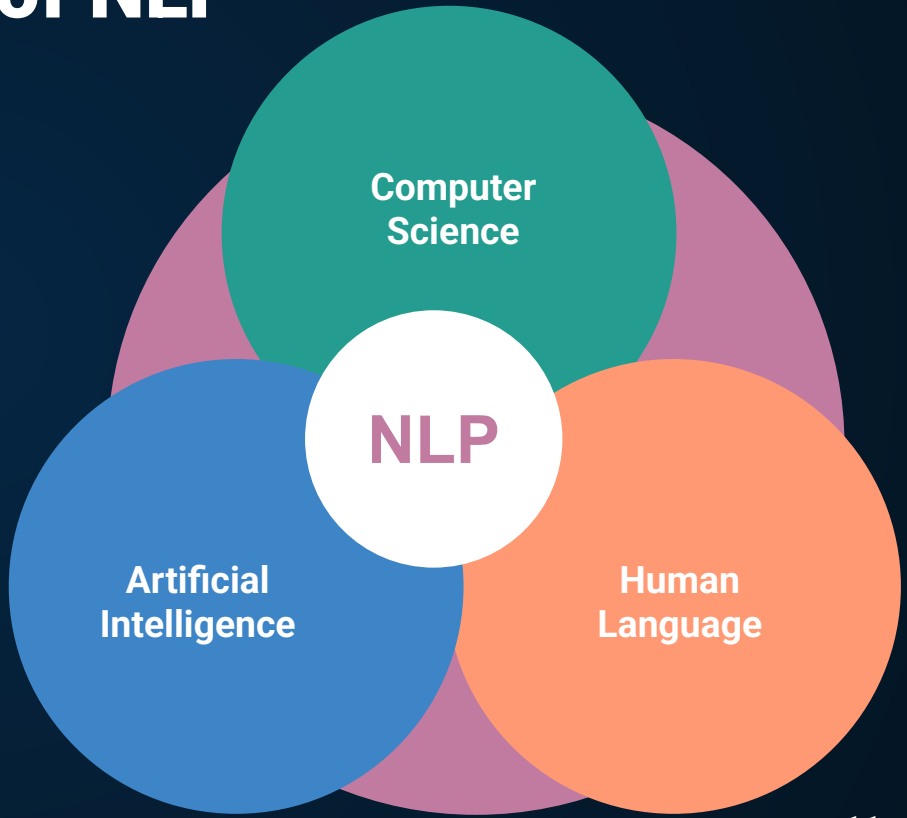
Introduction to Natural Language Processing

Definition of NLP

Natural language processing (NLP) is the interdisciplinary field of computer science and linguistics, using machine learning to achieve the end goal of artificial intelligence.

Human Language is complex to understand and interpret for computers.

The goal of NLP is to make computers understand this complex language structure and retrieve meaningful pieces of information from it.



Natural Language Processing

Natural Language Understanding

Taking some spoken / typed sentence and working out what it means

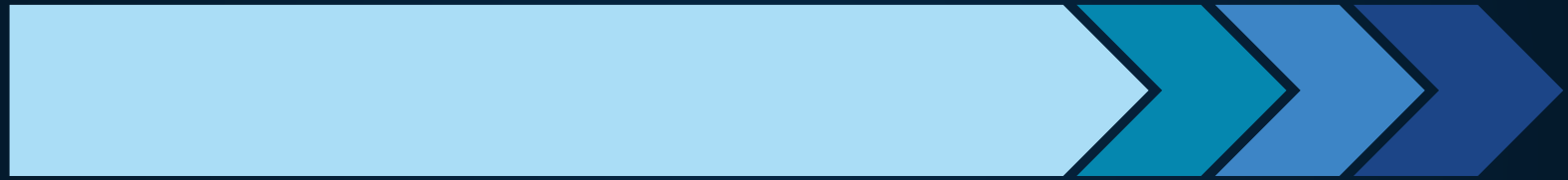
Natural Language Generation

Taking some formal representation of what you want to say and working out a way to express it in a natural human language

Fundamental goal

Deep understanding of broad language, not just string processing or keyword matching

Brief History of NLP



1950s - 1960s
Rule-based methods

The Turing Test
Georgetown-IBM experiment

1970s
Statistical Approaches

SHRDLU
Naive Bayes

1980s - 1990s
Machine Learning Approaches

Dragon Dictate
IBM model 1
Archie

2000s - 2010s
Deep Learning & NNs

RNNs
Word2Vec
Google Neural Machine Translation

2020s
Large Language Models

GPT- 3
Turing NLG
Megatron Turing NLG

Applications

Text categorization

Classify documents/ texts by topics, language, sentiment classification,Spam detection, etc.

Spelling & Grammar corrections

Detect spelling, structure and grammar mistakes in texts

Speech recognition

Siri, Alexa, etc.

Information Retrieval

Retrieving relevant information from documents

Information Extraction

Extracting data from text & converting unstructured text into structured data

Text Generation

Language translation, Chatbots and virtual assistants, Content generation, etc.

Summarization

Summarize long texts such as articles and documents

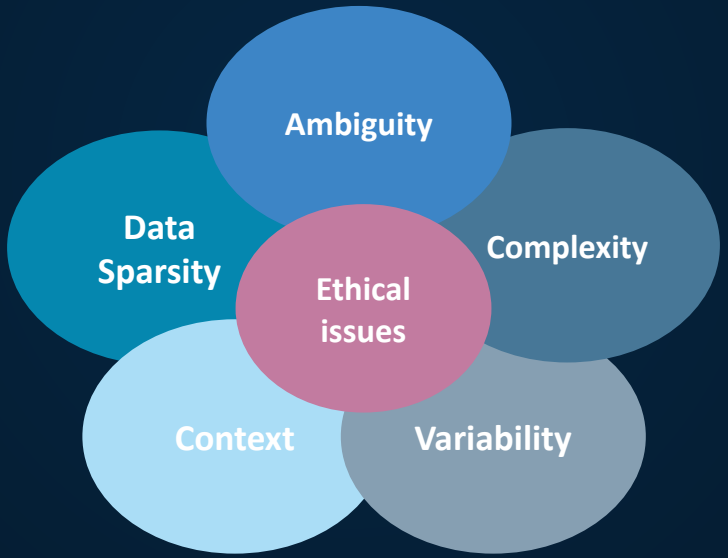
Question Answering

Customer support, Educational applications, Knowledge management, Search Engines, etc.

Machine translation

Translate text from one language to another

Challenges

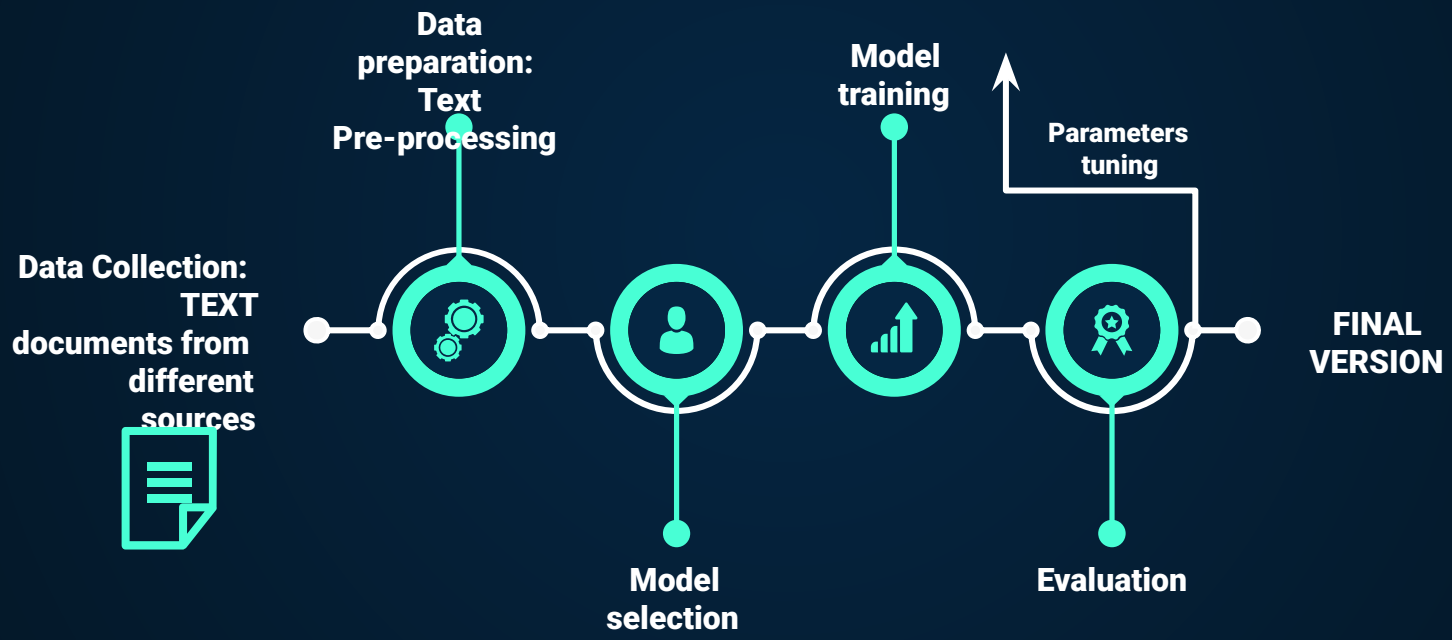

Ambiguity
Data Sparsity
Complexity
Ethical issues
Context
Variability

Basic Concepts of Natural Language Processing

TABLE OF CONTENTS

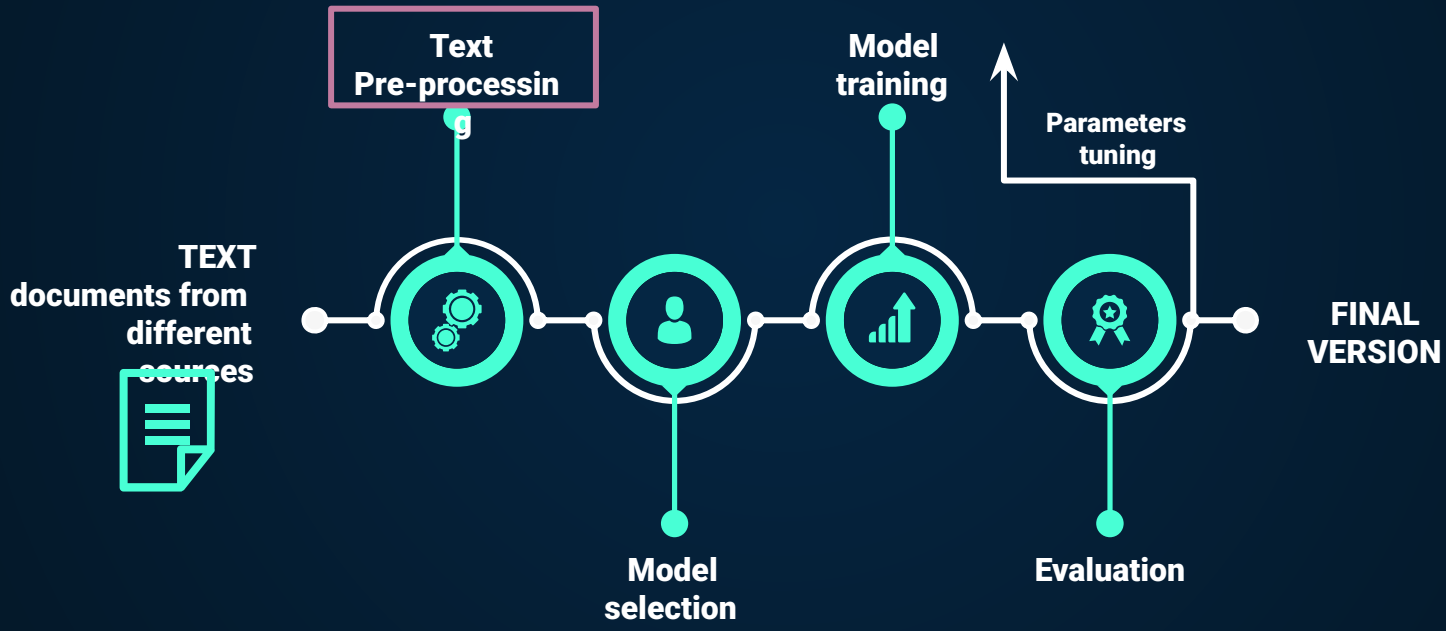
Introduction

“Classical” NLP Pipeline

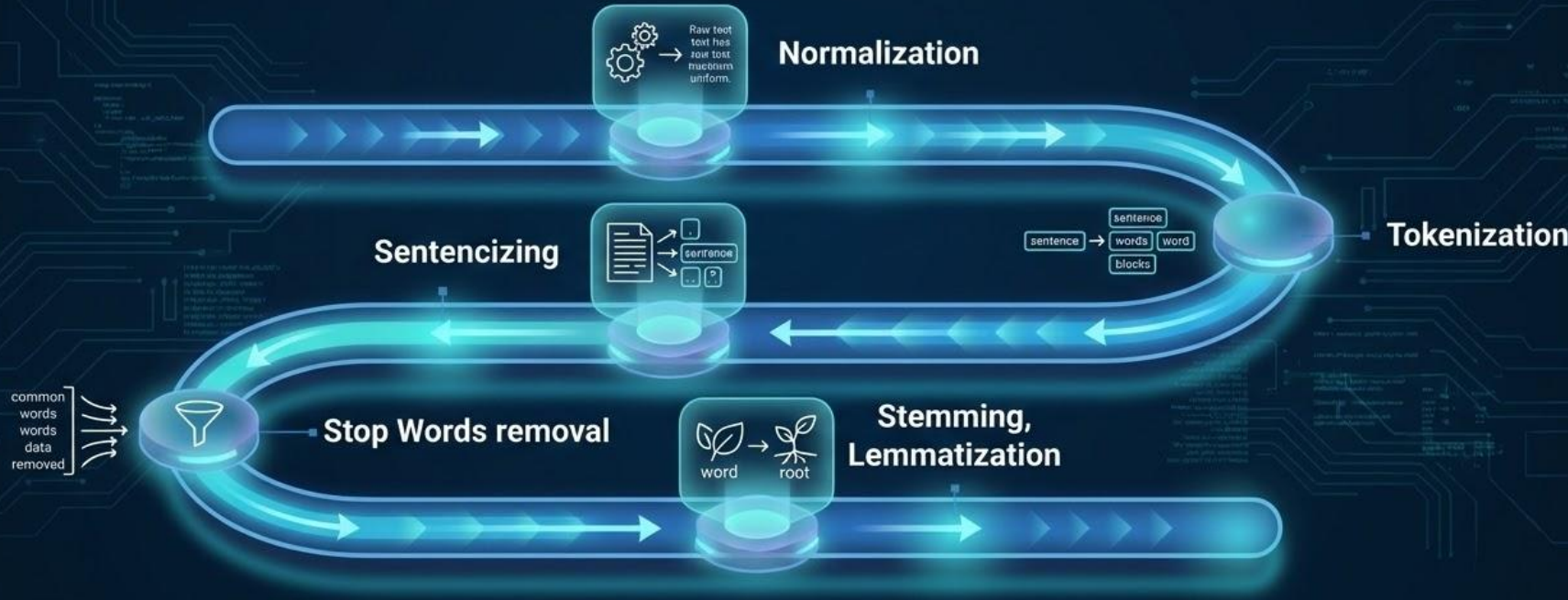


Standard pre-processing steps

Introduction

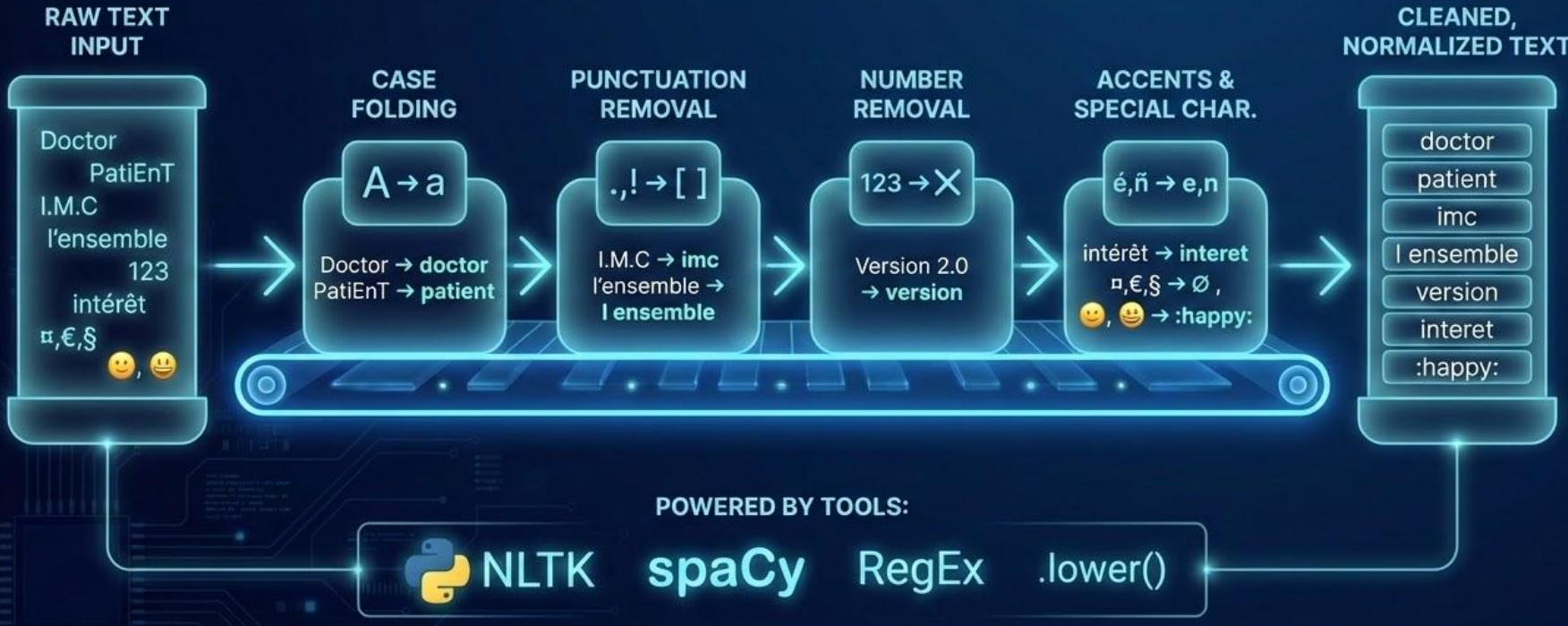


Standard pre-processing steps



Normalization / Cleaning

Normalization is pre-processing the text data so it is homogenous



DEMO

```
import re
```

```
text = """
Cher confrère, madame a été adMise pour une hémorragie cérébrale....
Elle est actuellement aux soins intensifs, et a besoin d'une opération en urgence!!!
() Ce courrier est à transmettre au chef du service Neurochirurgie. Mail du médecin traitant: médecintraitant@gmail.com
$ù*ù$ numéro de téléphone: 04395835035
"""
```

```
# Replace accented characters with unaccented characters
text = text.replace('é', 'e').replace('à', 'a').replace('è', 'e').replace('ù', 'u')
text
```

```
"\nCher confrere, madame a ete adMise pour une hemorragie cerebrale.... \nElle est actuellement aux soins intensifs, et a besoin d'une operation en ur
gence!!! \n() Ce courrier est a transmettre au chef du service Neurochirurgie. Mail du medecin traitant: medecintraitant@gmail.com \n$u*u$ numero de t
elephone: 04395835035\n"
```

```
# Remove unwanted characters and white spaces
text = re.sub('[^A-Za-z0-9\s]+', '', text)
text
```

```
'\nCher confrere madame a ete adMise pour une hemorragie cerebrale \nElle est actuellement aux soins intensifs et a besoin dune operation en urgence
\n Ce courrier est a transmettre au chef du service Neurochirurgie Mail du medecin traitant medecintraitantgmailcom \nuu numero de telephone 043958350
35\n'
```

```
# Convert all characters to lowercase
text = text.lower()
```

```
# Print the cleaned text
print(text)
```

```
cher confrere madame a ete admise pour une hemorragie cerebrale
elle est actuellement aux soins intensifs et a besoin dune operation en urgence
 ce courrier est a transmettre au chef du service neurochirurgie mail du medecin traitant medecintraitantgmailcom
uu numero de telephone 04395835035
```

Tokenization

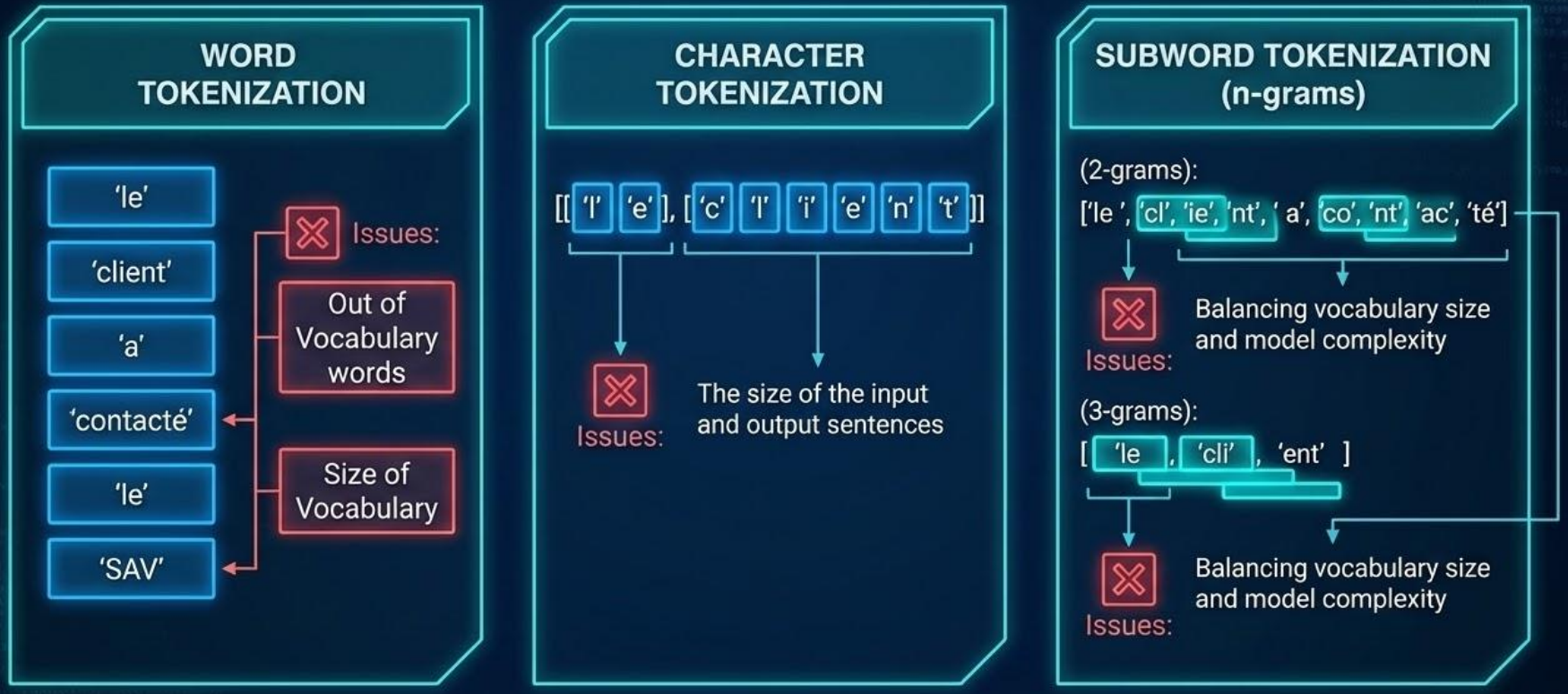
Tokenization is the process of breaking a stream of text into words, symbols, or other meaningful elements called tokens. The aim of the tokenization is the exploration of the words in a sentence. The list of tokens becomes input for further processing such as parsing or text mining. *(Gurusamy and Kannan, 2014)*

e.g.,

Text = ['Le 07/08, le client a contacté le SAV pour une panne dûe à une coupure d'électricité"]

Tokenized_text = ['le', '07/08', ',', 'le', 'client', 'a', 'contacté', 'le', ' SAV', pour', une', 'panne', dûe', à', 'une', 'coupure' 'd"', électricité", '.']

Tokenization: methods



Tokenization

BERT tokenizer

Text = ['le', '07/08', ',', 'le', ‘client’, ‘a’, ‘contacté’, ‘le’, ‘ SAV”, pour’, une’, ‘panne’, dûe’, à’, ‘une’, ‘coupure’ ‘d”, électricité”, '.']

Tokenized_text = ['le', '07/08', ',', 'le’, ‘client’, ‘a’, ‘cont’, ‘#acté’, ‘le’, ‘ SAV”, pour’, une’, ‘panne’, dûe’, à’, ‘une’, ‘coup’, ‘#ure’ ,‘d”, ‘élec’, ‘#tric’, ‘##ité”, '.']

Tokenization: tools

spaCy

RegEx

gensim

K Keras

.split()

Sentencizing

Sentencizing is the process of splitting the text into sentences, based on line breaks, punctuation or other patterns.

Stanza RegEx NLTK spaCy

DEMO

```
[1]: import nltk
```

```
[7]: nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\naima\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

```
[7]: True
```

```
[9]: text = """
Cher confrère,
madame a été admise pour une hémorragie cérébrale. Elle est actuellement aux soins intensifs, et a besoin d'une opération en urgence.
Ce courrier est à transmettre au chef du service Neurochirurgie.
"""
```

```
•[10]: # Sentencizing the text

sentences = nltk.tokenize.sent_tokenize(text)
```

```
[11]: sentences
```

```
[11]: ['\nCher confrère,\nmadame a été admise pour une hémorragie cérébrale.',
 "Elle est actuellement aux soins intensifs, et a besoin d'une opération en urgence.",
 'Ce courrier est à transmettre au chef du service Neurochirurgie.']
```

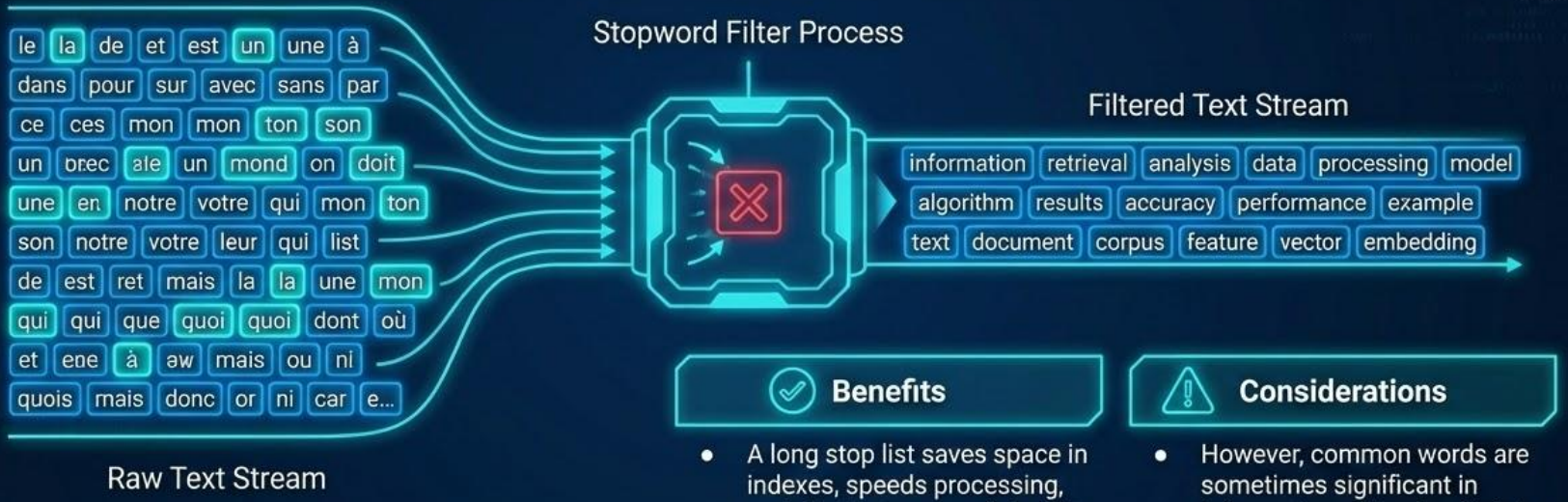
```
•[13]: # Tokenizing each each sentence into words

for sentence in sentences:
    mots = nltk.word_tokenize(sentence)
    print(mots)
```

```
['Cher', 'confrère', ',', 'madame', 'a', 'été', 'admise', 'pour', 'une', 'hémorragie', 'cérébrale', '.']
['Elle', 'est', 'actuellement', 'aux', 'soins', 'intensifs', ',', 'et', 'a', 'besoin', "d'une", 'opération', 'en', 'urgence', '.']
['Ce', 'courrier', 'est', 'à', 'transmettre', 'au', 'chef', 'du', 'service', 'Neurochirurgie', '.']
```

Stopwords removal

Stopwords are very common words, such as of, and, the, are rarely of use in information retrieval. A stop list is a list of such words that are removed during lexical analysis.



Benefits

- A long stop list saves space in indexes, speeds processing, and eliminates many false hits.
- Helps focus on meaningful content.

Considerations

- However, common words are sometimes significant in information retrieval, which is an argument for a short stop list.
- Long stop words list may lower recall.

Stopwords removal

Example

Text = ['Le 07/08:le client a contacté le SAV pour se plaindre d'une panne, ***sans*** *demande de remboursement.']*

nltk_text = ['07/08: client contacté sav se plaindre une panne. demande remboursement.']

spacy_text = ['07/08: client contacté sav pour se plaindre une panne. demande remboursement.]

DEMO

```
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\naima\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
# French stop words
stop_words = set(stopwords.words('french'))
```

```
# Text to be processed
text = """
Cher confrère, madame a été admise pour une hémorragie cérébrale.
Elle est actuellement aux soins intensifs, et a besoin d'une opération en urgence.
Ce courrier est à transmettre au chef du service Neurochirurgie.
"""
```

```
# Tokenize the text
words = word_tokenize(text)
```

```
# Filter out stop words
filtered_words = [word for word in words if word.casefold() not in stop_words]
```

```
# Join the filtered words back into a string
filtered_text = ' '.join(filtered_words)
```

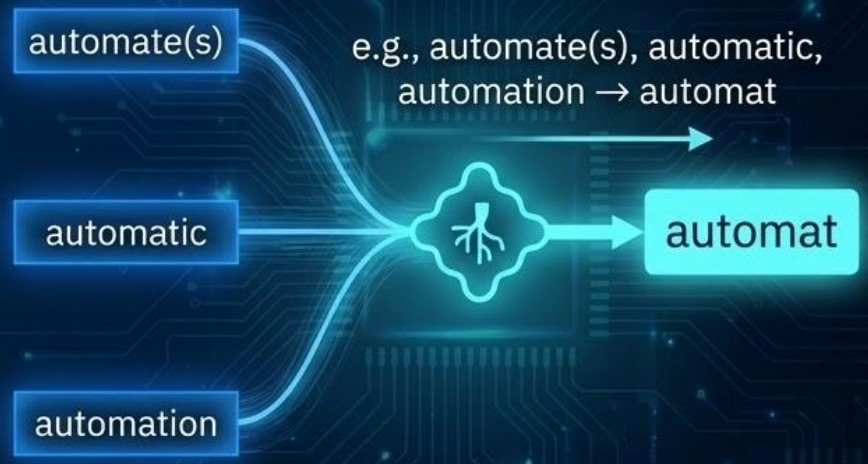
```
# Print the filtered text
print(filtered_text)
```

```
Cher confrère , madame a admise hémorragie cérébrale . actuellement soins intensifs , a besoin d'une opération urge
nce . courrier transmettre chef service Neurochirurgie .
```

Stemming

Stemming means reducing terms to their “roots” before indexing.

- Helps recall for some queries but harm precision on others.
- Fine distinctions may be lost through stemming.
- Performance of various algorithms is similar.



Lemmatization

Lemmatizing means reducing inflectional/variant forms of a word to its base form.

Original Text
(Inflected Forms)

Dans les formes
peu sévères,
ce traitement n'est
habituellement
pas prescrit

Lemmatization Process
(Base Form Reduction)

Lemmatized Text
(Base Forms)

→ Dans le forme
peu sévère,
ce traitement ne être
habituellement
pas prescrire

e.g., The process converts inflected words to their dictionary form (lemma).

DEMO

```
import nltk
nltk.download('punkt')
nltk.download('omw-1.4')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')

from nltk.stem import SnowballStemmer, WordNetLemmatizer
from nltk.tokenize import word_tokenize
```

```
# English stemmer and lemmatizer
stemmer = SnowballStemmer('english')
lemmatizer = WordNetLemmatizer()
```

```
# Text to be processed
text = """
Dear colleague,
Madam was admitted for a cerebral hemorrhage.
She is currently in intensive care and needs an emergency operation.
This letter is to be transmitted to the head of the Neurosurgery department.
"""
```

```
# Tokenize the text
words = word_tokenize(text)
```

```
# Perform stemming and lemmatization on the words
stemmed_words = [stemmer.stem(word) for word in words]
lemmatized_words = [lemmatizer.lemmatize(word, pos='v') for word in words]
```

```
print("Lemmatized words: ", lemmatized_words)
```

```
Lemmatized words:  ['Dear', 'colleague', ',', 'Madam', 'be', 'admit', 'for', 'a', 'cerebral', 'hemorrhage', '.', 'S
he', 'be', 'currently', 'in', 'intensive', 'care', 'and', 'need', 'an', 'emergency', 'operation', '.', 'This', 'let
ter', 'be', 'to', 'be', 'transmit', 'to', 'the', 'head', 'of', 'the', 'Neurosurgery', 'department', '.']
```

Advanced pre-processing steps

Advanced pre-processing steps

Negation handling

Negation handling tools help detect negated spans in a text.

- **Algorithms & Tools**
  - NegEx algorithm *(Chapman et al, 2001)*, NegFinder *(Mutallk et al. 2001)*, eds.NLP *(APHP)*

French Example

[Le traitement] ne [génère] pas [d'effets indésirables particuliers]

Detects negated concept 'génère' and its scope.

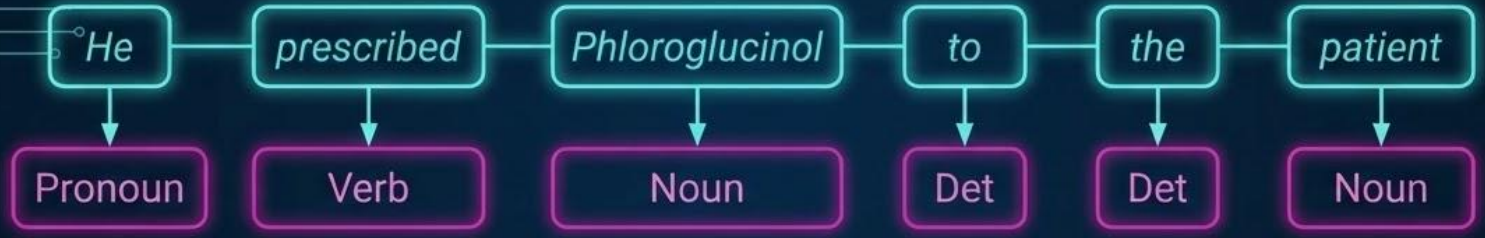
English Example

[The patient does not show any side effects, apart from headache]

Identifies negation cue 'not' and exception 'apart from'.

PoS tagging

Part-of-Speech tagging involves adding a part of speech category to each token within a text. Some common PoS tags are verb, adj, noun, pronoun, etc.



Tools & Libraries

Stanford NLP

NLTK

TextBlob

spaCy

DEMO

```
import nltk
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
from nltk.tokenize import word_tokenize
```

```
# Text to be processed
text = """
Dear colleague,
Madam was admitted for a cerebral hemorrhage.
She is currently in intensive care and needs an emergency operation.
This letter is to be transmitted to the head of the Neurosurgery department.
"""
```

```
# Tokenize the text
words = word_tokenize(text, language='english')
```

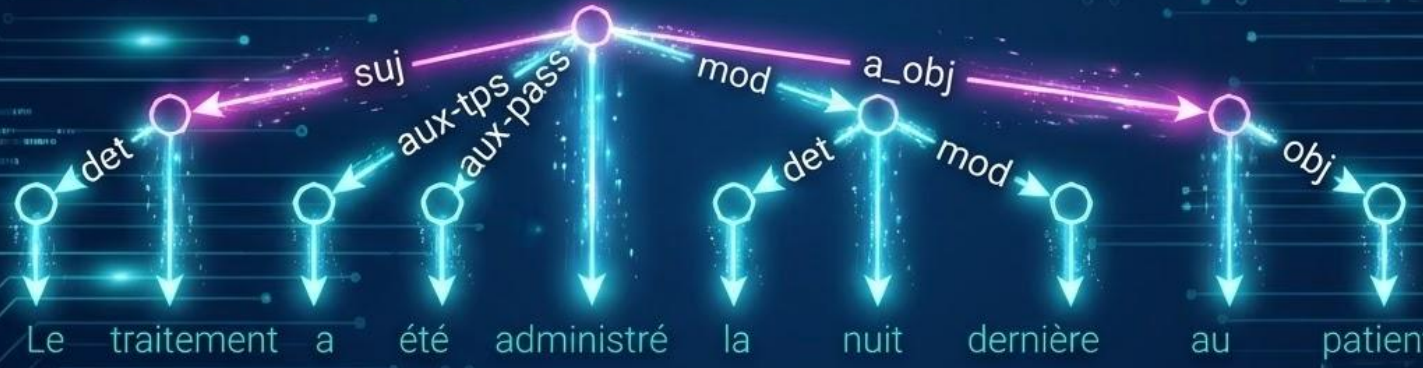
```
# PoS tag the words
pos_tags = nltk.pos_tag(words, lang='eng')
```

```
# Print the part-of-speech tags
print(pos_tags)
```

```
[('Dear', 'NNP'), ('colleague', 'NN'), (',', ','), ('Madam', 'NNP'), ('was', 'VBD'), ('admitted', 'VBN'), ('for',
'IN'), ('a', 'DT'), ('cerebral', 'JJ'), ('hemorrhage', 'NN'), ('.', '.'), ('She', 'PRP'), ('is', 'VBZ'), ('currentl
y', 'RB'), ('in', 'IN'), ('intensive', 'JJ'), ('care', 'NN'), ('and', 'CC'), ('needs', 'VBZ'), ('an', 'DT'), ('emer
gency', 'NN'), ('operation', 'NN'), ('.', '.'), ('This', 'DT'), ('letter', 'NN'), ('is', 'VBZ'), ('to', 'TO'), ('b
e', 'VB'), ('transmitted', 'VBN'), ('to', 'TO'), ('the', 'DT'), ('head', 'NN'), ('of', 'IN'), ('the', 'DT'), ('Neur
osurgery', 'NNP'), ('department', 'NN'), ('.', '.')]
```

Dependency Parsing

Dependency parsing refers to the way the words in a sentence are connected grammatically.



Project examples

Sentiment analysis on tweets

Necessary pre-processing steps:

- Text normalization
- Tokenization
- Sentencizing
- Expanding abbreviations
- Lemmatization
- Stopwords removal
- Negation handling

De-identification of legal records

Necessary pre-processing steps:

- Text normalization
- Tokenization
- Padding
- Masking
- Sentencizing

TP

Projet: Détection des spams

1. Collecte de données :

- https://www.kaggle.com/datasets/chandramoulinaidu/spam-classification-for-basic-nlp

Détection de spam à partir des données textuelles

2. Prétraitement des données :

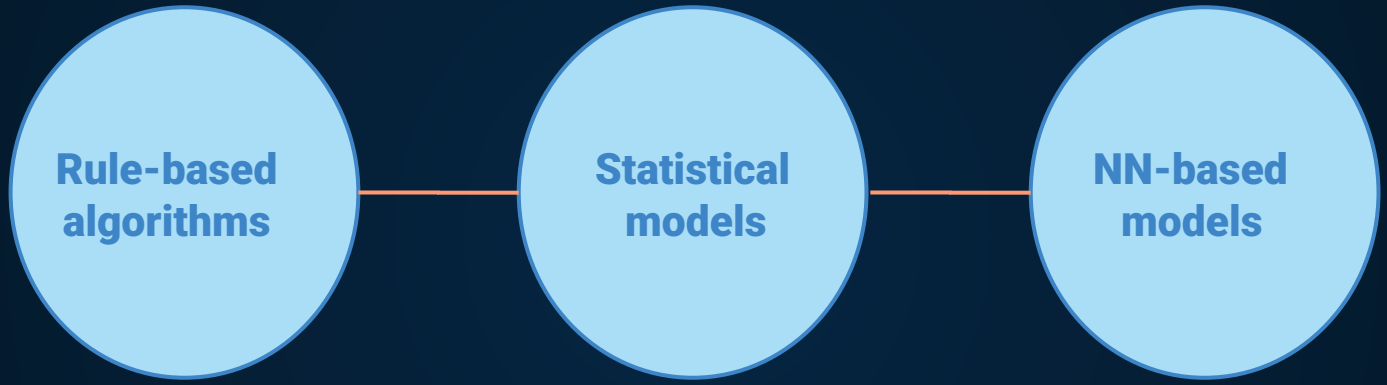
Ecrivez des fonctions en Python pour :

- **la Suppression des balises HTML**
- **La Suppression des caractères spéciaux**
- **La tokenisation du texte**
- **La conversion du texte en minuscules**
- **la suppression de la ponctuation**
- **la suppression des mots blancs**
- **Le Stemming/lemmatization des mots**

Advanced Machine Learning and Deep Learning for NLP

TABLE OF CONTENTS

Introduction



Hard-coded and require humans to manually set parameters for grammatical syntax but ignore pragmatism and semantics

Provide ample real-world data to give computers the ability to predict words / labels and intent based on historical usage

Use deep learning techniques to learn the patterns and relationships between words in a more complex way.

Introduction

Bag of Words

The nurse reported that the patient has been dismissed

A single word is a one-hot encoding vector with the size of the dictionary

Introduction

Problem

- Manually designed features are often over-specified, incomplete and take a lot of time to design and validate
- Often require PhD-level knowledge of the domain
- Researchers spend decades hand-crafting features
- Bag of Words model is very high-dimensional and sparse, it cannot capture semantics or morphology

Maybe Deep Learning could help?

TF-IDF

TF

Local Term Frequency

The frequency of the term in a document

TF-IDF

Global Term Frequency

Weighted by the "significance " of the term in the corpus of training documents

TF-IDF

$$w_{ij} = tf_{ij} \times \log_2 \frac{N}{n}$$, where

w_{ij} = weight of term T_j in document D_i
tf_{ij} = frequency of term T_j in document D_i
N = number of documents in collection
n = number of documents where T_j occurs at least once

Deep Learning for NLP

Core Idea:

represent words as dense vectors → **Embeddings**

~~[0 1 0 0 1 1 1 0 0 0 1 0 1]~~ [0.234 0.538 1.435 0.543]

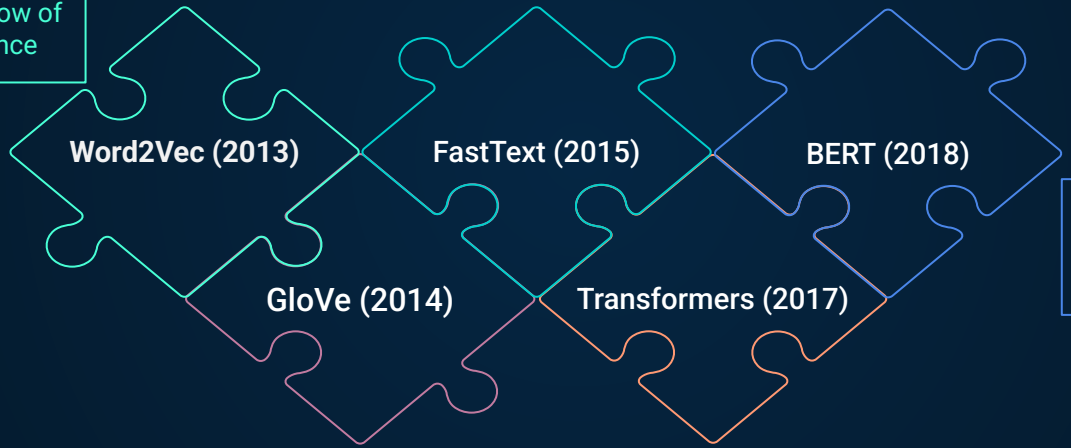
Try to capture semantics and morphologic similarity so that the features for "similar" words are "similar"
e.g., closer in Euclidean space

Natural language is context dependent: **Use context for learning**

Evolution of NLP algorithms

Simple NN to predict the word based on the context window of other words in the sentence

Extension of Word2Vec
Each word is treated as a set of sub-words



Pre-training of Deep Bidirectional Transformers for Language Understanding

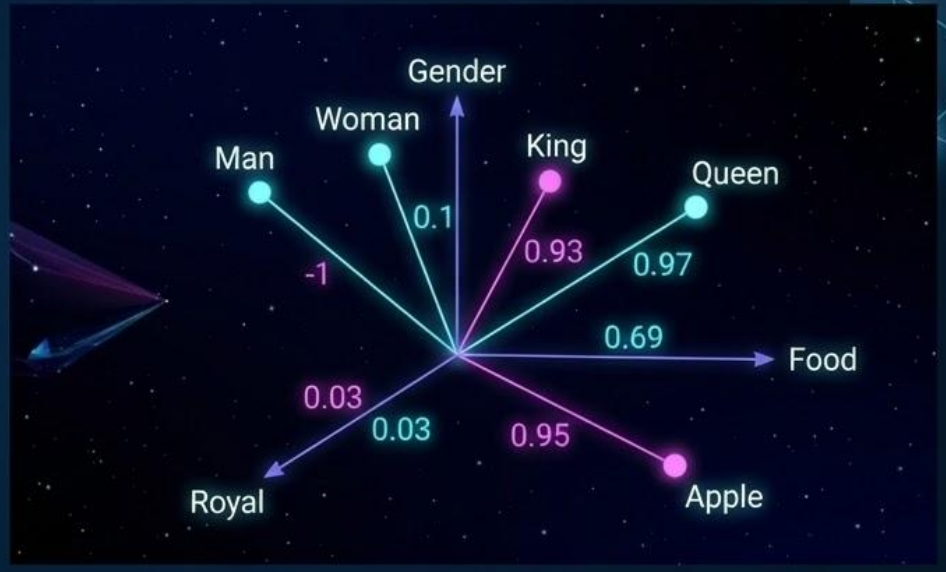
Global Vectors for Word representation
Matrix factorization

Attention is All you need:
Introduction of Attention mechanisms

Word Embeddings

Word embeddings are a technique where individual words of a domain or language are represented as real-valued vectors in a lower dimensional space.

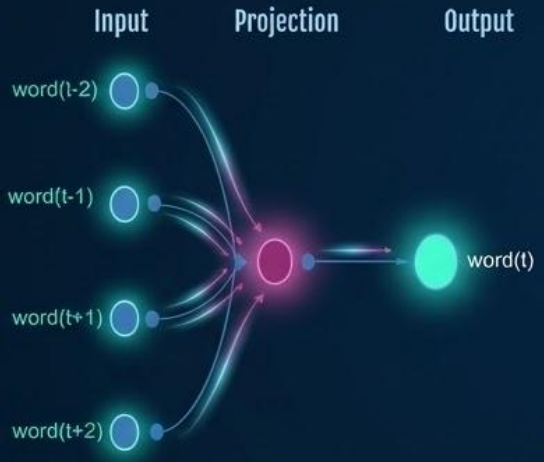
Word Embedding resolves 2 major issues of word representations : curse of dimensionality and absence of relatedness between words.



Source: from the lecture of Sequence Models by Andrew Ng

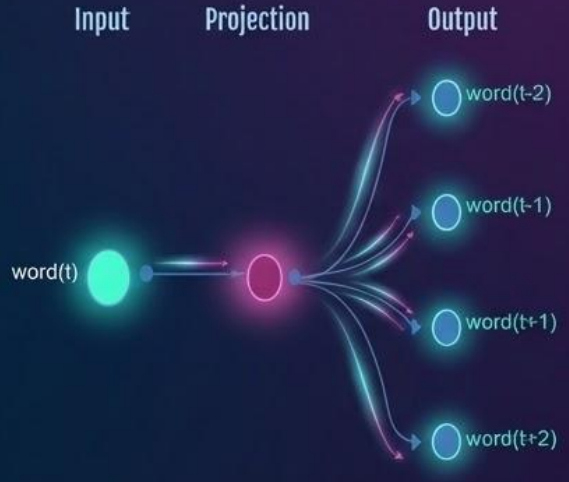
Word2Vec

CBOW (Continuous Bag-of-Words)



Predict the target word from its context.

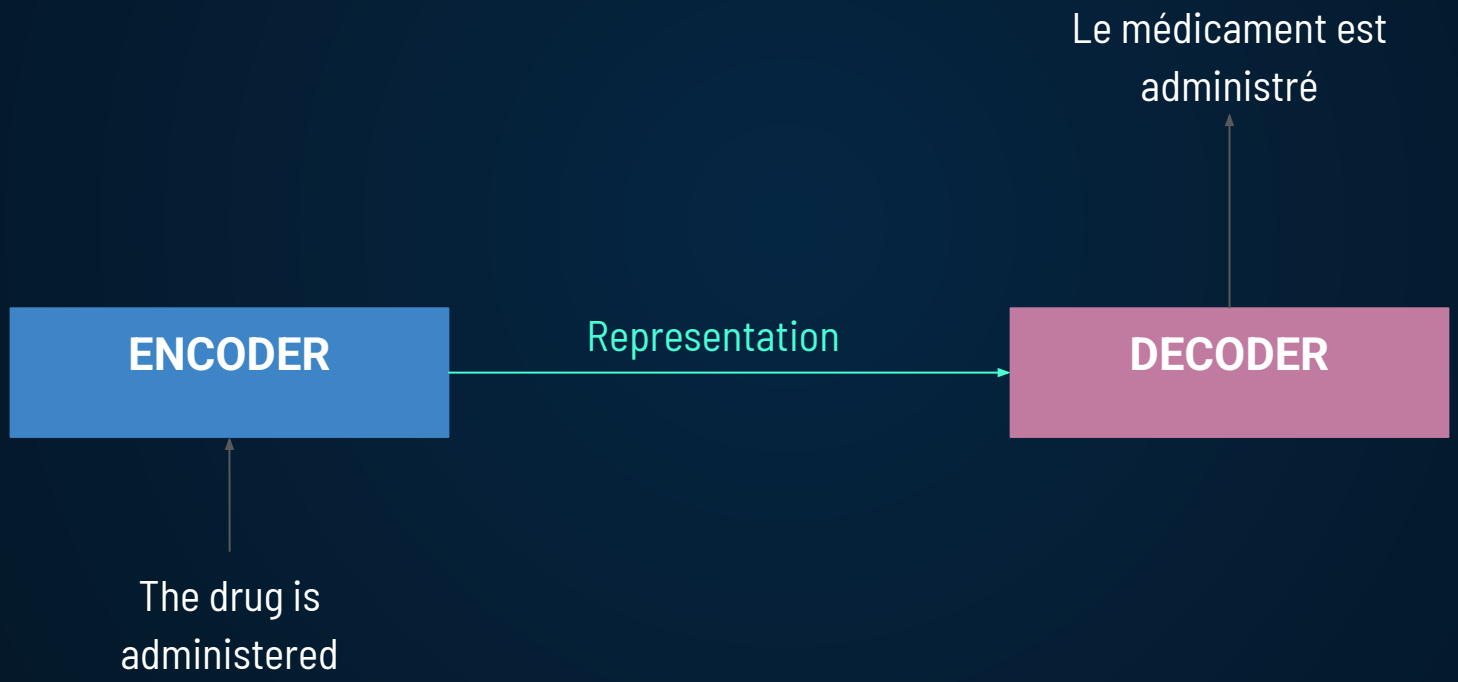
Skip-Gram



Predict the context words from the target word.

Transformers

How do transformers work?



Transformers

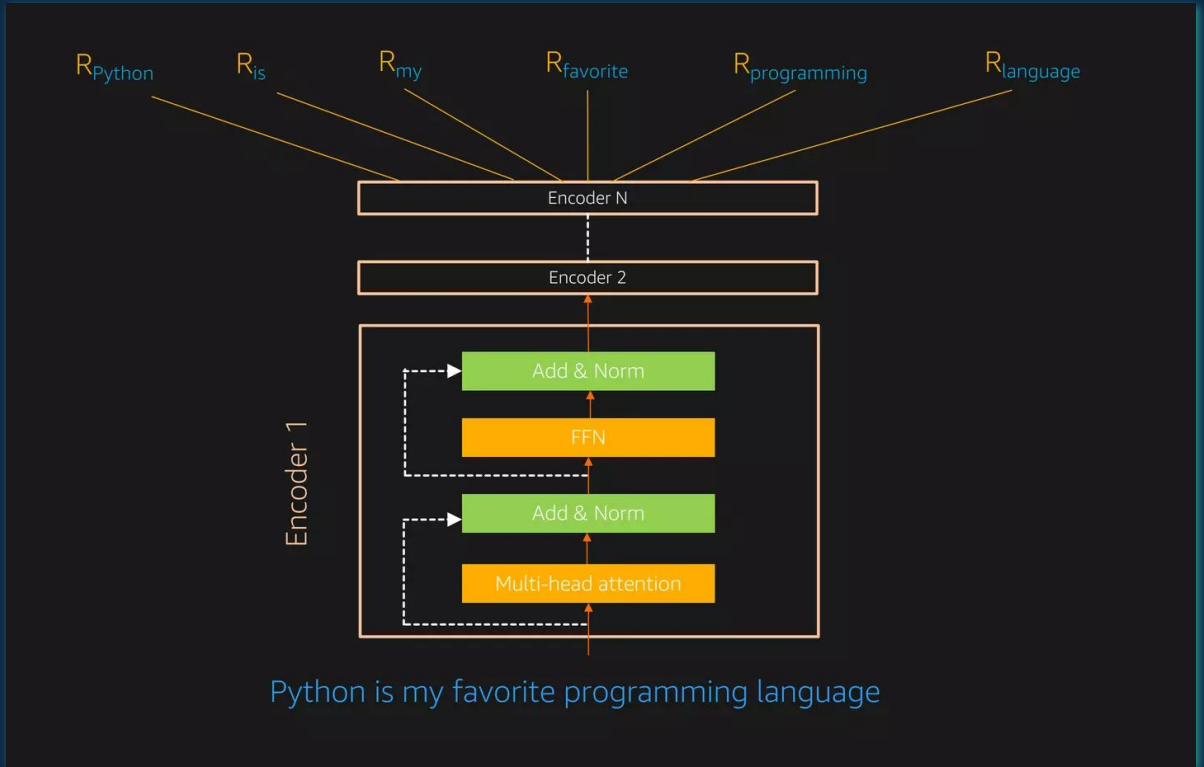
1. Position Encoding: This is a key component in Transformer models, allowing the order of words in a sentence to be taken into account

2. The Attention Mechanism: Allows the model to focus on certain parts of an input sequence when generating an output sequence

3. Self-attention: This is a variant of the attention mechanism, enabling the model to take the whole input sequence into account

BERT

Bidirectional Encoder Representation from Transformer



BERT

How does BERT work?

Masked Language Model (MLM)

Predict the masked words from the surrounding context.

e.g., Lille is a beautiful city. I love Lille

Next Sentence Prediction (NSP)

Determine if Sentence B logically follows Sentence A.

BERT

How does BERT work?

Benefits of Word Embeddings

- Learning features of each word on its own, given a text corpus
- No heavy preprocessing required, just a corpus
- Word vectors can be used as features for lots of supervised learning applications:
  → PoS tagging, NER, chunking.
- Taking into account similarities and linear relationships between word vectors

Word embeddings: Evaluation

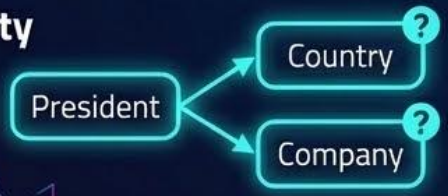
Intrinsic Evaluation:

Assess how well the word embeddings inherently capture the **semantic (meaning)** or **syntactic** (grammar) **relationships** between words.

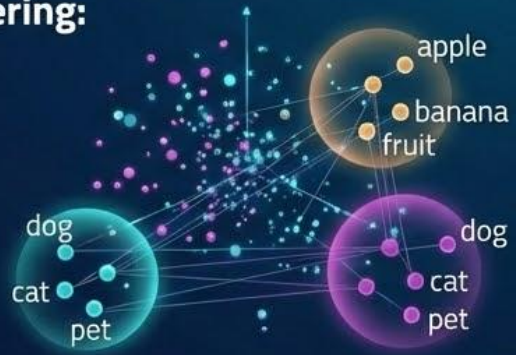
Test on semantic analogies:

Test on syntactic analogies:

Test on Ambiguity



Clustering:

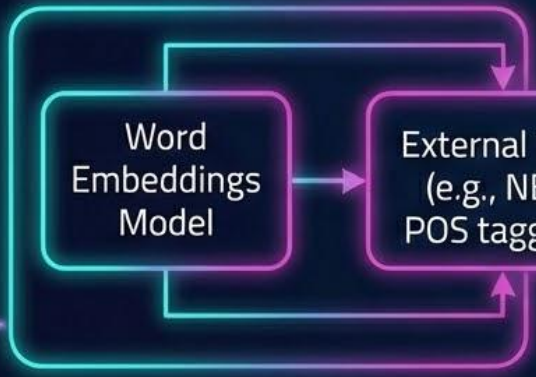


Using a clustering algorithm to group similar word embedding vectors, and determine if the clusters capture related words

Word embeddings: Evaluation

Extrinsic Evaluation:

Test on External Task:



Test the word embeddings to perform an external task, e.g. Named Entity Recognition, POS tagging

Evaluate Classifier:

Evaluate this classifier on the test set with some selected evaluation metric such as: accuracy, precision, recall or F1-Score

Challenges:

The evaluation will be more time-consuming than an intrinsic evaluation and more difficult to troubleshoot

Exploratory Data Analysis

Purpose & Goals:

- Exploring data
- Generating insights
- Testing hypotheses
- Revealing underlying hidden patterns

Key Analysis Techniques:

- Word frequency analysis

- Sentence length analysis

- Average word length analysis

- Frequency of different words or phrases

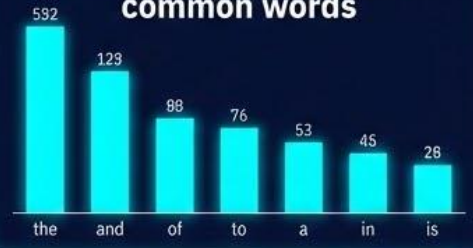
Exploratory Data Analysis

Visualization:

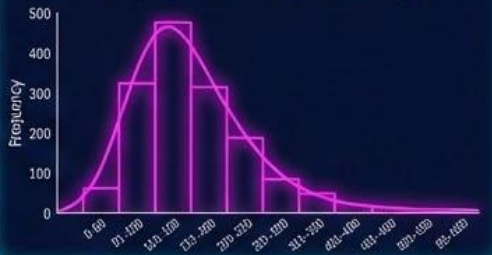
Understand the patterns and trends in the data

Word clouds

Bar plots of the most common words



Histograms of text length



Bar plots of most common n-grams

Topic Modeling

TP

Projet : Classification des spams

3. Word embeddings

- Utilisez les outils des embeddings de mots pour générer une représentation en **bag of words, Word2vec** Embeddings et **TF-IDF** des mails.

4. Analyse exploratoire des données :

- **Nombre moyen de mots par mail**
- **Les mots les plus fréquents**
- **Visualiser les Word clouds**
- **Bar plots des mots les plus communs**
- **Histogrammes de la longueur des mails**

Topic Modeling

Topic Modeling

Technique to identify dominant themes in a document

Assumptions: LDA

- No training required
- Multiple Algorithms for implementation

Core Concepts

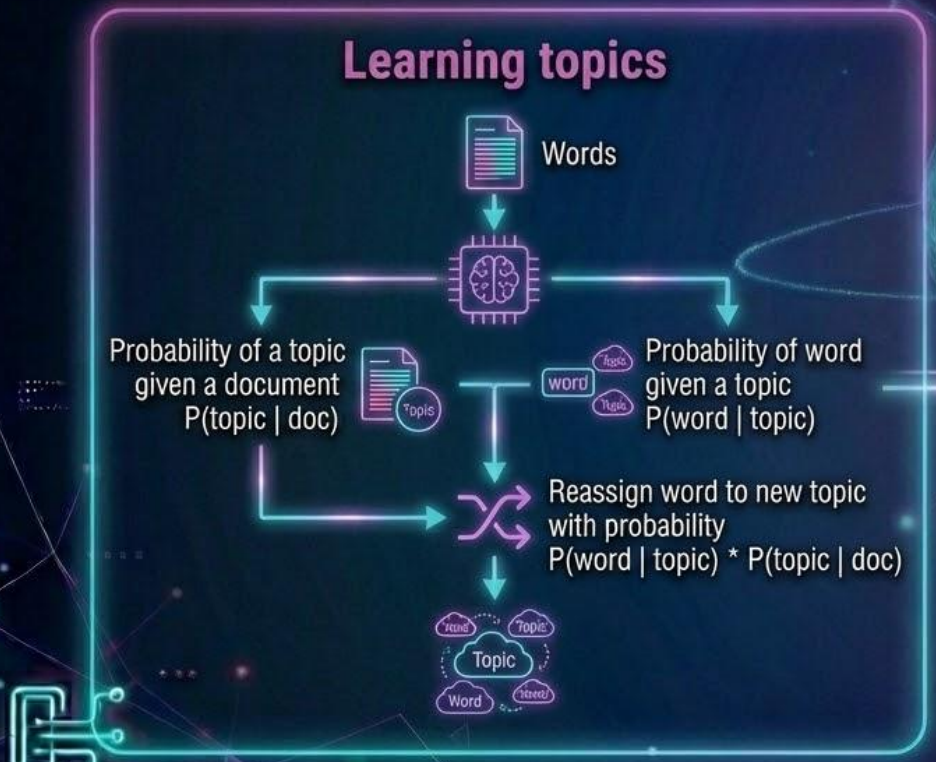
- Each topic is a distribution over words
- Each document is a mixture of corpus-wide topics
- 

Each word is drawn from one of those topics

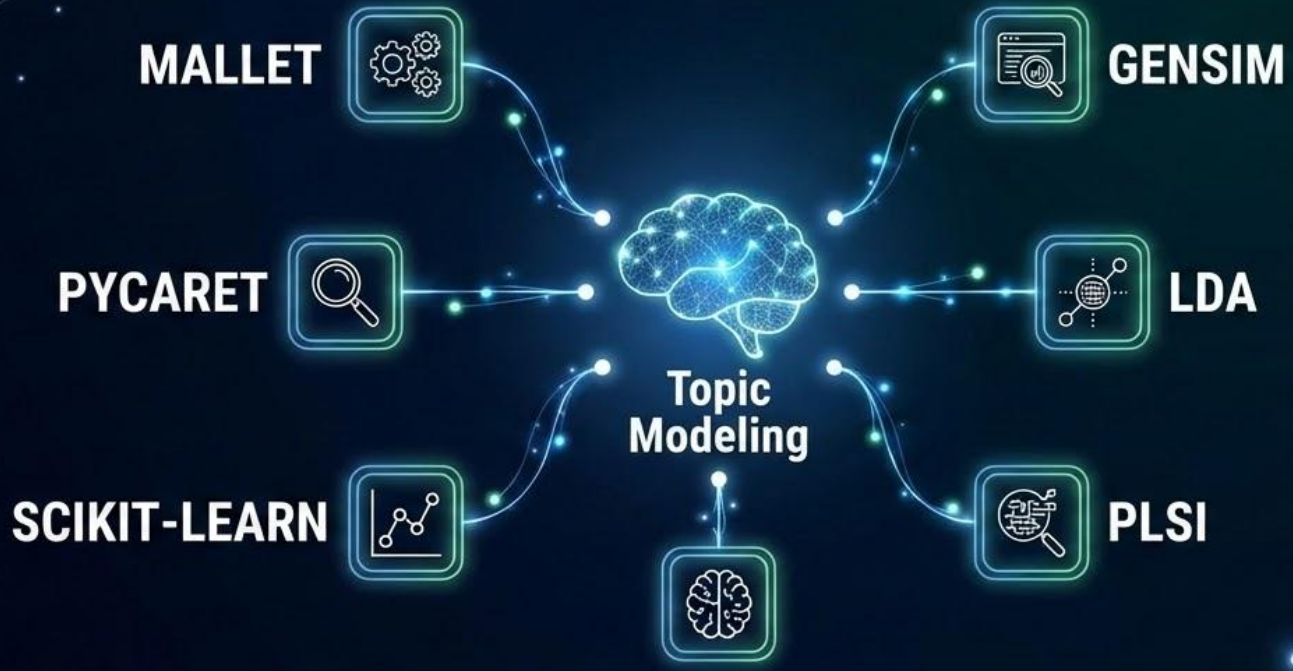
Applications

- Data Exploration in large corpora
- Pre-classification analysis
- Identify dominant themes
- Trendspotting

Topic Modeling



TOPIC MODELING: TOOLS



Topic Modeling on Covid-19 Tweets

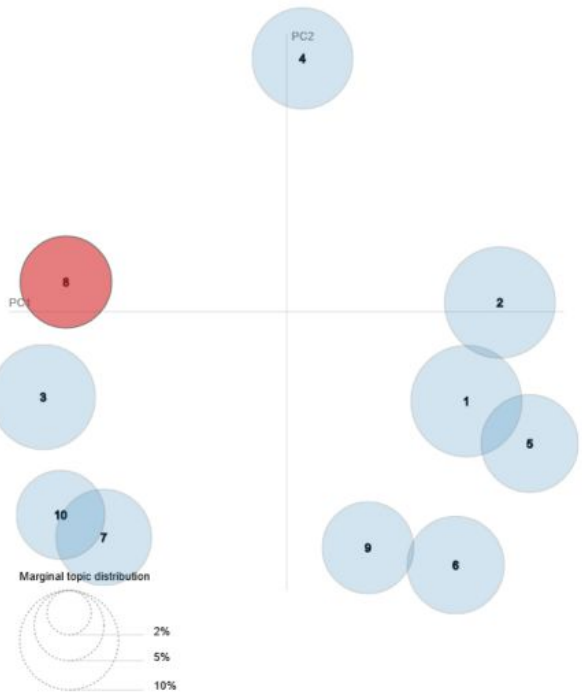


Figure 5: Intertopic distance map between the 10 topics

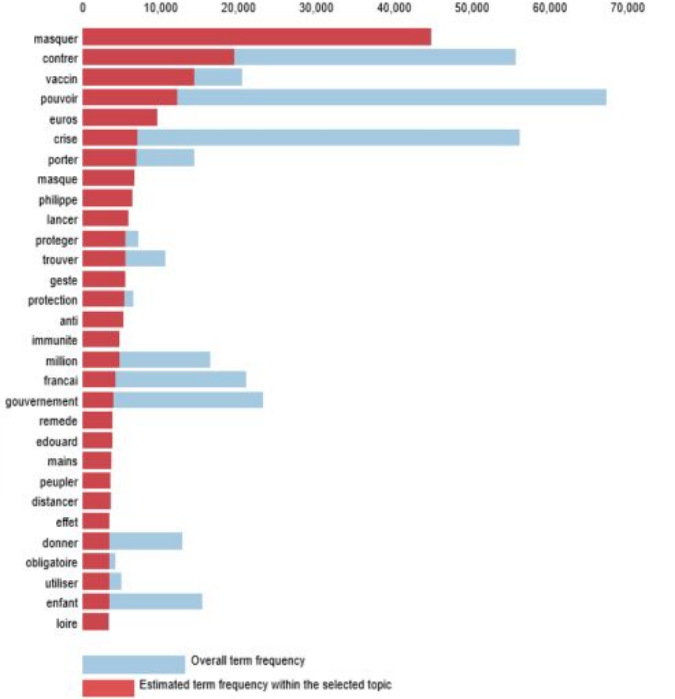


Figure 6: Top 30 most relevant words for topic 8

Projet : Topic modeling

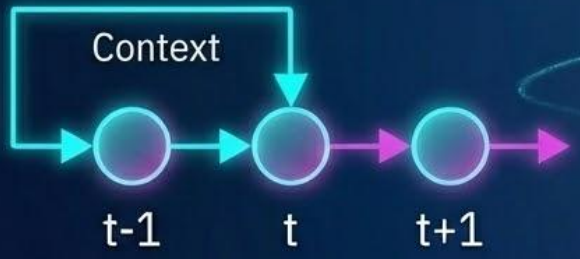
1. **Topic Modeling**
   Faites du topic Modelling sur les emails (LDA)

2. **Visualisation:**
   Visualisez les topics présents dans votre corpus et analysez-les

Neural Network-based Models

RNNs

Memory & Context



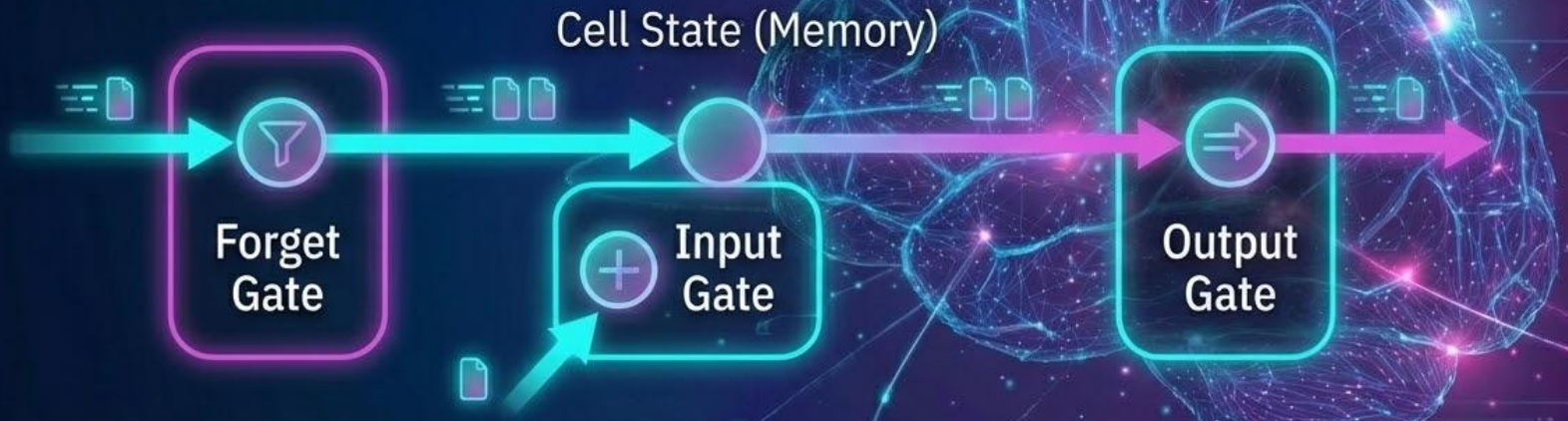
RNNs can use **past information without restricting the size** of the context.

The Long-Term Memory Problem

But in practice, **can't recall information** that came in a long ago.

Long Short Term Memory (LSTM)

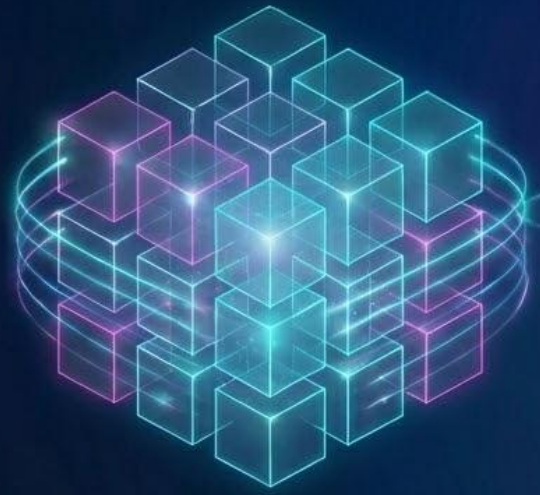
Surprisingly amazing performances at language tasks compared to RNNs



LSTMs contain gates that control **forgetting**, **adding**, **updating** and **outputting** information, effectively managing long-term dependencies.

Convolutional Neural Networks (CNNs)

Multidimensional Arrays / Tensors



CNNs are designed to process data in the form of **multidimensional arrays / tensors**.

CNNs are essentially NNs that use **convolution** in place of general matrix multiplications for the 1st layer.

Advanced Applications in NLP

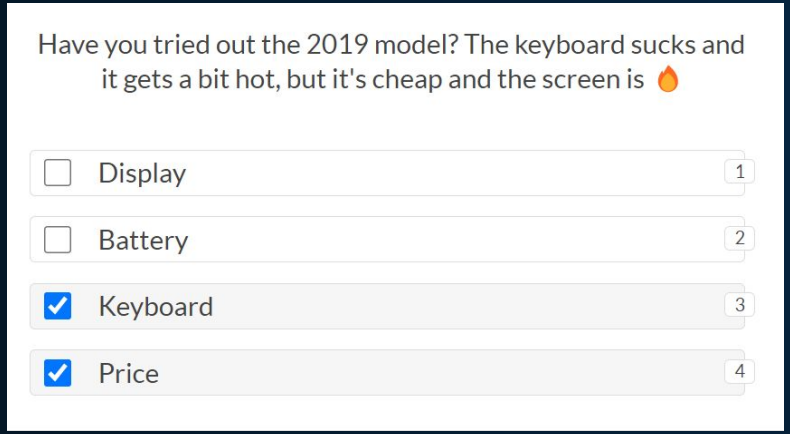
Text Categorization

Text Categorization

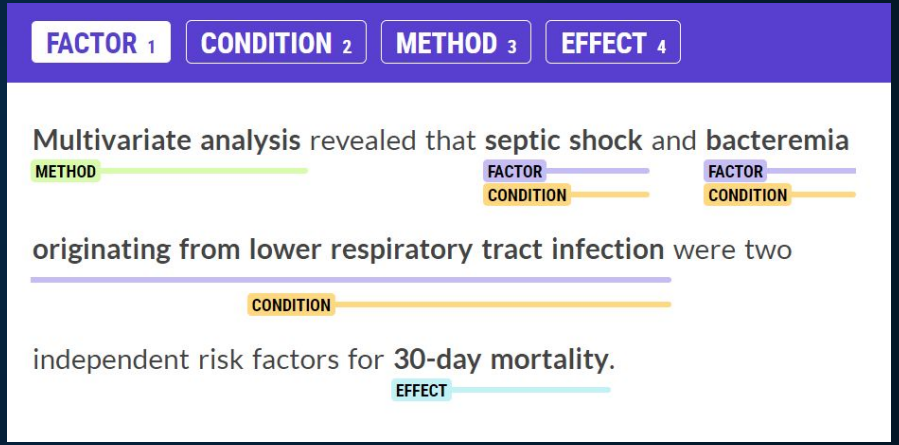
Sentiment Classification / Analysis

Text Categorization

Text Classification

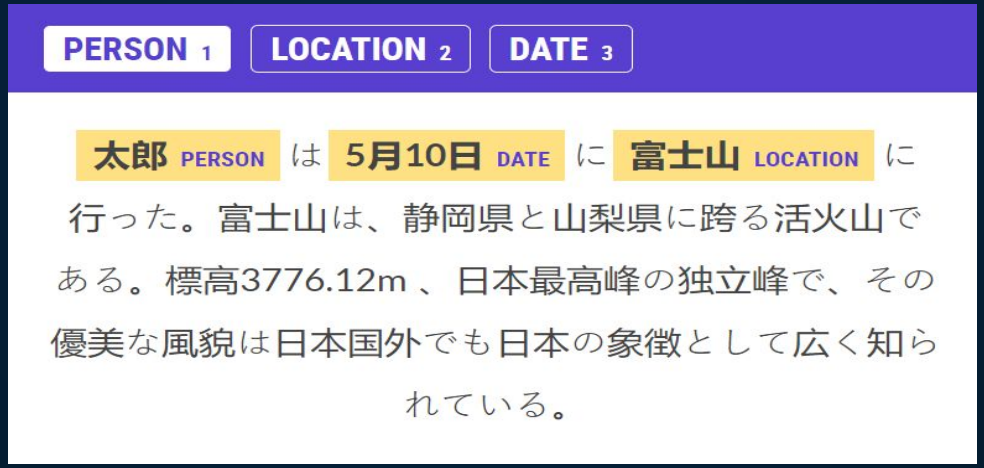


Span Categorization

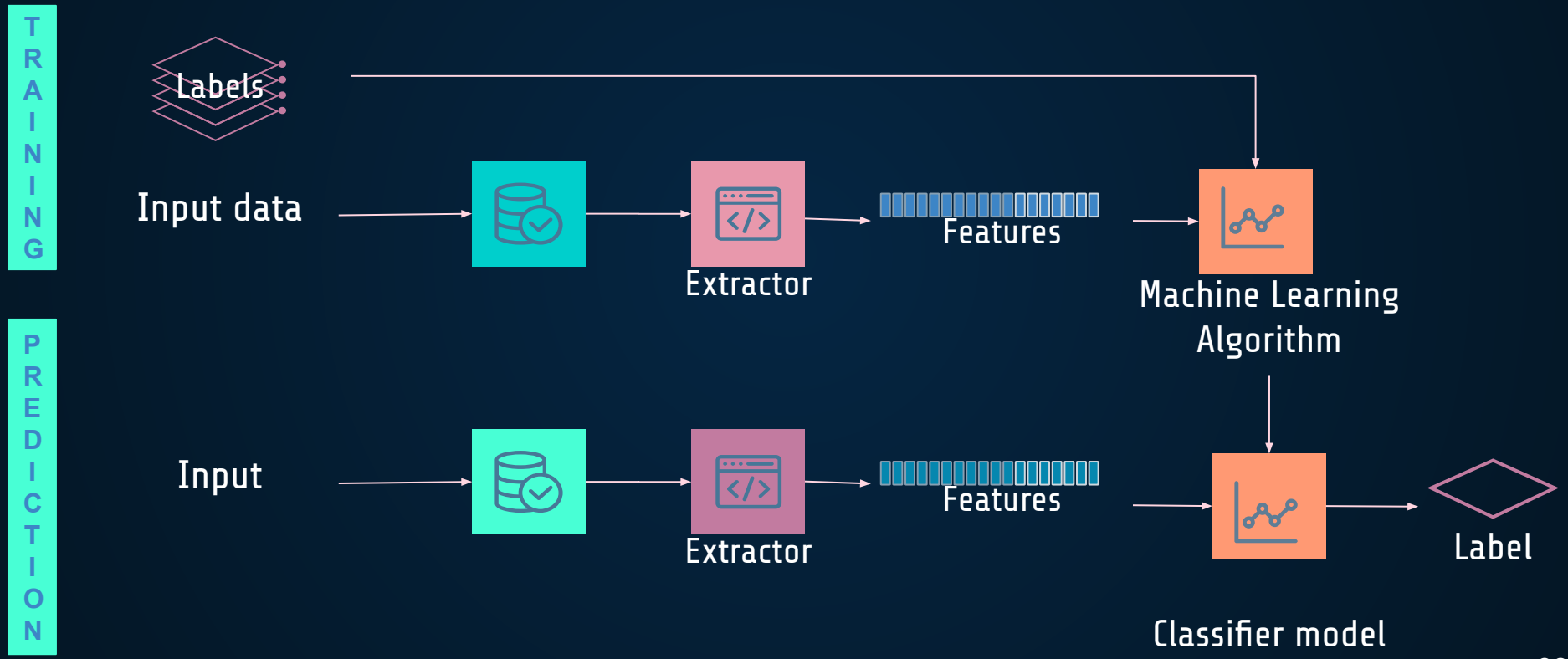


Text Categorization

Named Entity Recognition



Text Categorization



Named Entity Recognition

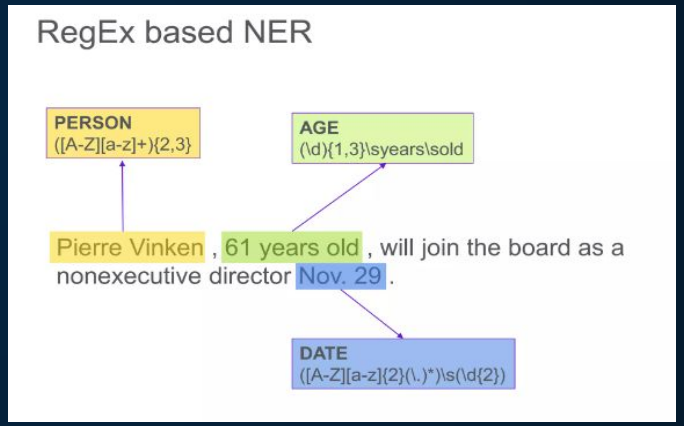
Named Entity Recognition

- Named Entity Recognition (NER) is a common task of Natural Language Processing
- Find and Classify entities in text into predefined categories
- Popular categories like: person, organizations, locations, date, etc.
- Usages: Machine translation, Information retrieval, Question answering, etc.

NER Approaches

Rule based

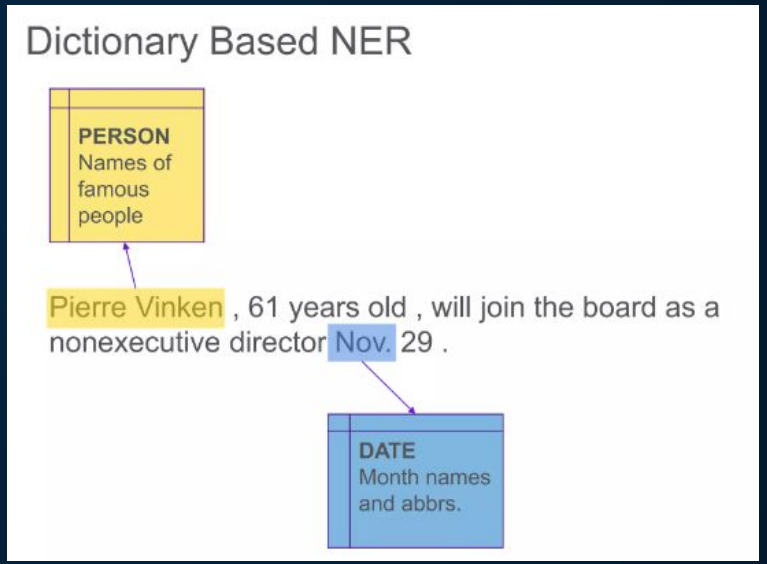
- Regular Expressions
- Language rules



NER Approaches

Dictionary based

- Pre-built vocabulary dictionaries



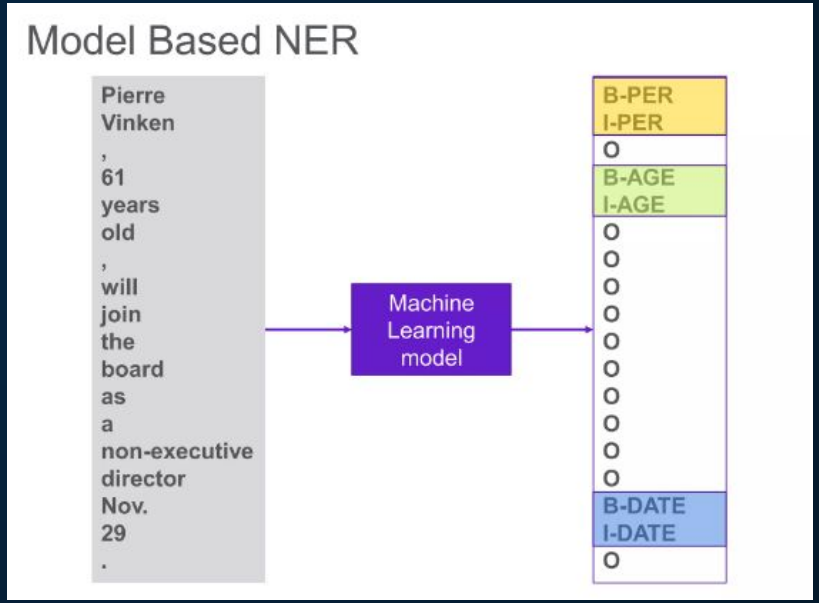
NER Approaches

Model Based

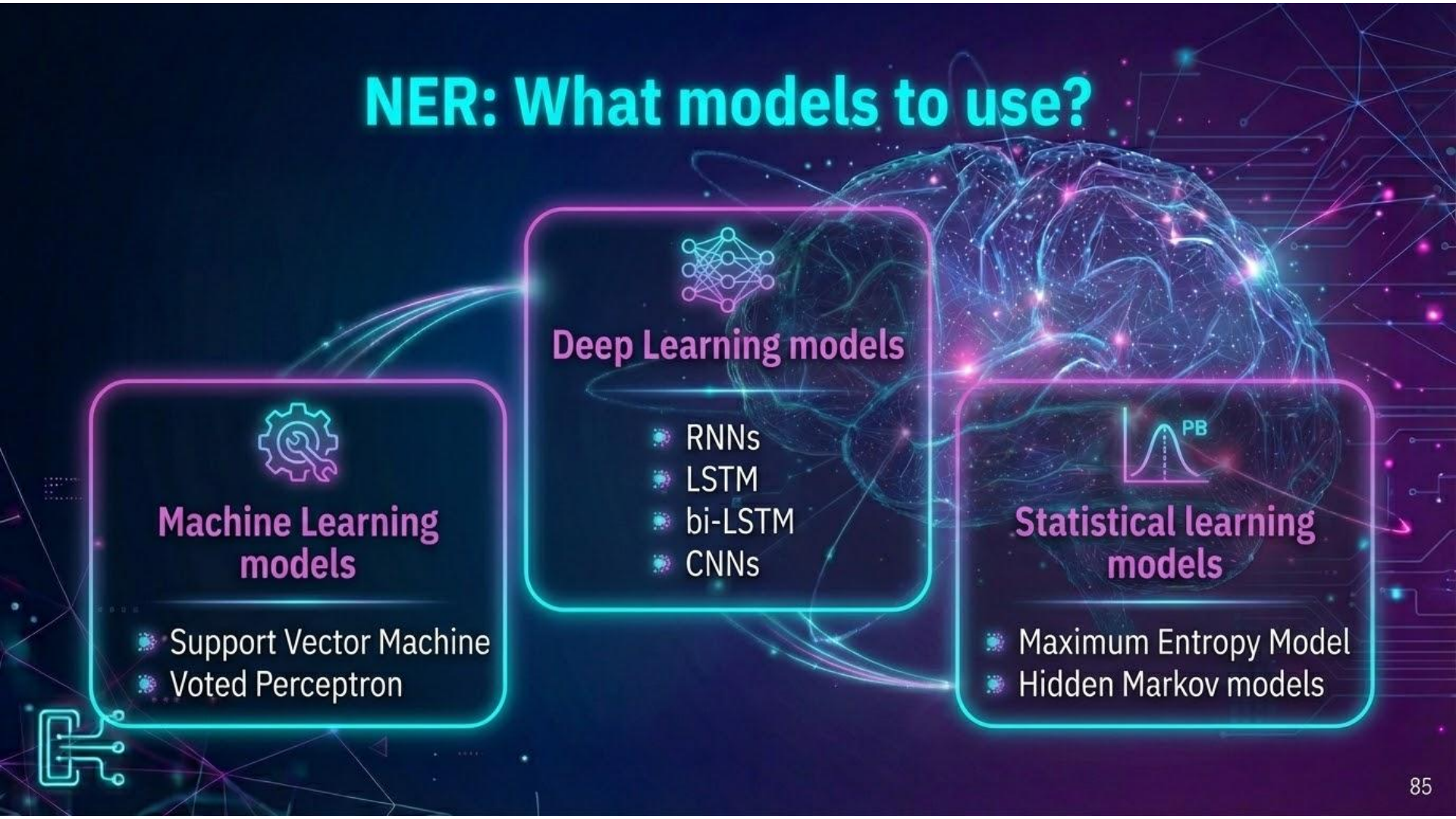
- Statistical models
- ML models
- DL models

Hybrid approaches

- Combining approaches
- Data Programming
- Active Learning



NER: What models to use?



Machine Learning models

- Support Vector Machine
- Voted Perceptron

Deep Learning models

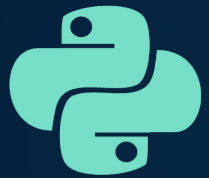
- RNNs
- LSTM
- bi-LSTM
- CNNs

Statistical learning models

- Maximum Entropy Model
- Hidden Markov models

NER: What libraries to use?

spaCy



NLTK

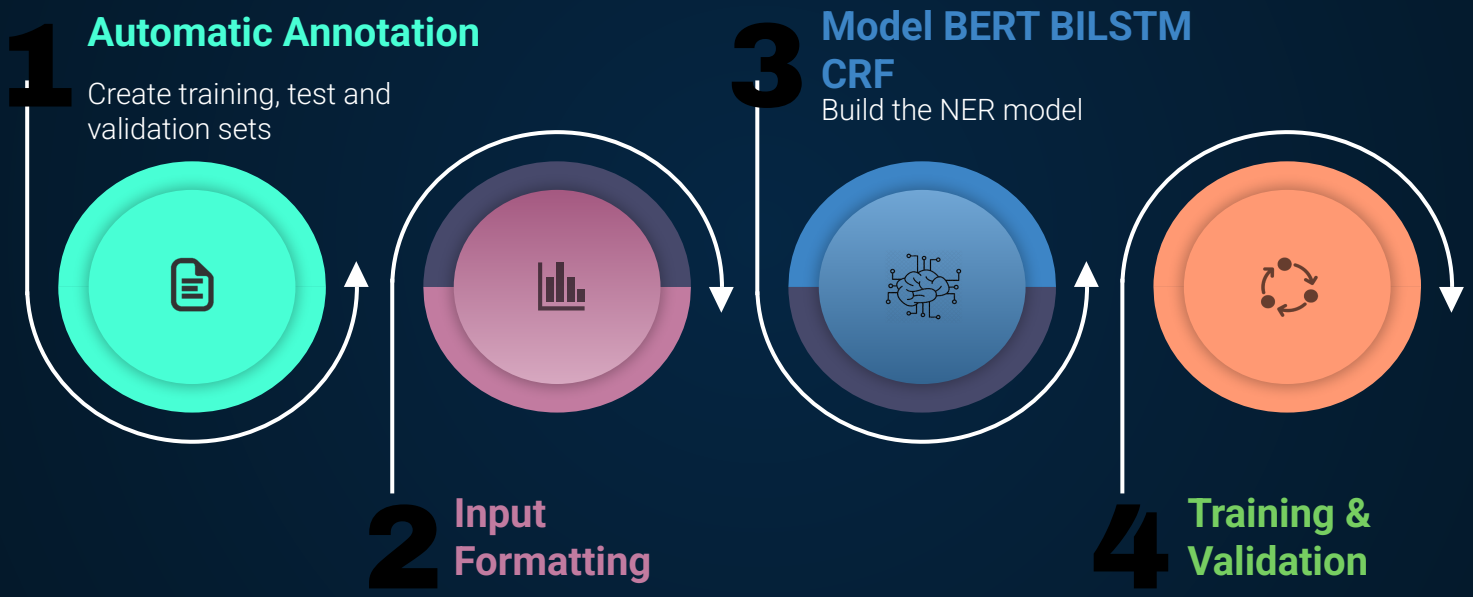
RegEx

gensim

flair

Stanford | NLP

NER: a Pipeline Example



DEMO

```
doc_fr_ner = nlp_fr("J'ai mangé à Paris avec Jonas")

print("--- Named Entities ner ---")
for ent in doc_fr_ner.ents:
    print(f"Text: {ent.text}, Label: {ent.label_}")
```

```
--- Named Entities ner ---
Text: Paris, Label: LOC
Text: Jonas, Label: PER
```

```
doc_en_ner = nlp_en("I was recently in New York. I was at Apple Store")

print("--- Named Entities ner ---")
for ent in doc_en_ner.ents:
    print(f"Text: {ent.text}, Label: {ent.label_}")
```

```
--- Named Entities ner ---
Text: New York, Label: GPE
Text: Apple Store, Label: ORG
```

Projet : Reconnaissance d’Entités Nommées

1. **NER**

Utilisez Spacy ou BERT pré-entraîné pour trouver l’ensemble des entités nommées présentes dans votre corpus
Mettez l’ensemble des entités dans une nouvelles colonne entities

2. Analyse des entités:

Analysez les entités qui sont les plus récurrentes dans les mails labellisés comme spam

Text Classification

Training a Text Classifier

Algorithms

Supervised Learning:
SVM
Perceptrons
kNN
SGD Classifier
Naive Bayes

Ensemble methods:
Random forest
AdaBoost

Data

Text data is unstructured

Data
Representation model (embedding model)

Algorithms and modules

Positive examples: From representative corpus

Negative examples:
Randomly selected

Word representations

Word2Vec embeddings
FastText embeddings
BERT embeddings

Text Classification: SVM

Large Margin Classifier

- Commonly used in text classification

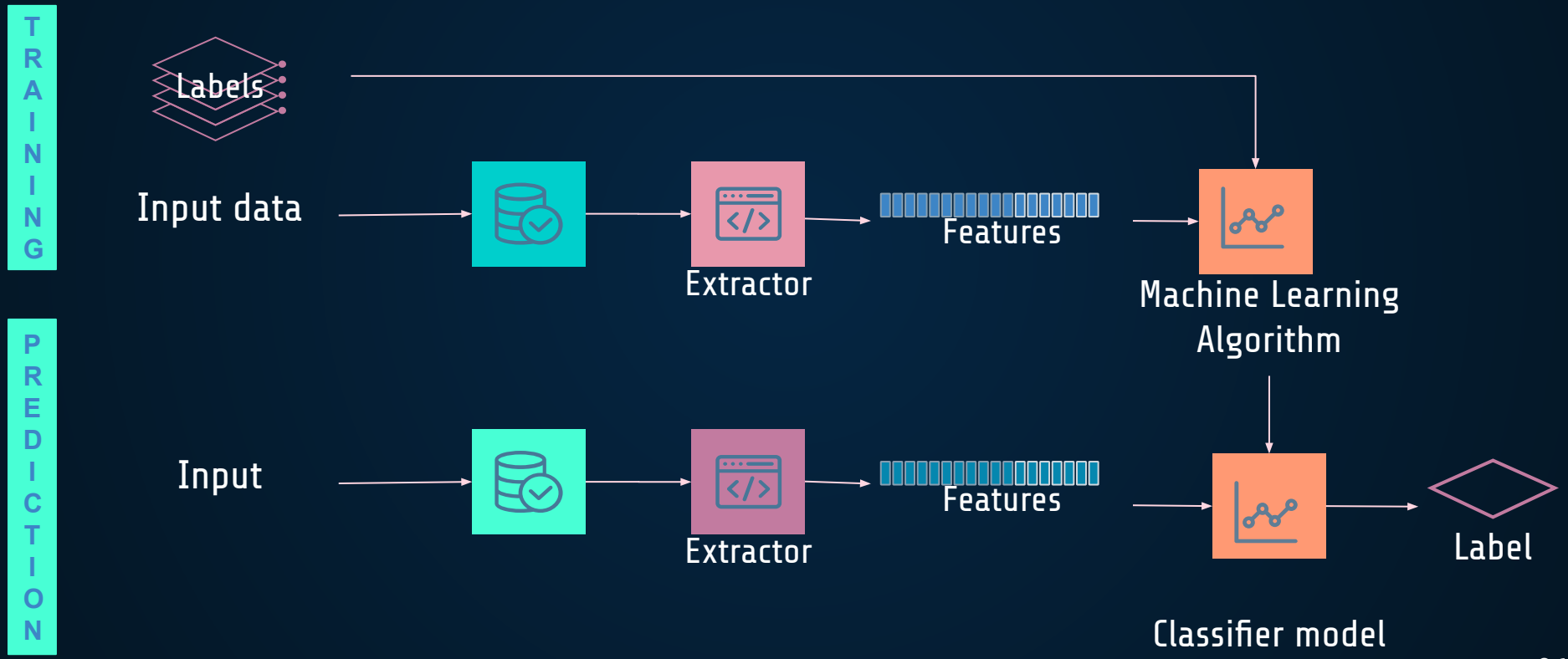
Initial Results

- based on life sciences sentence classifier

Applications

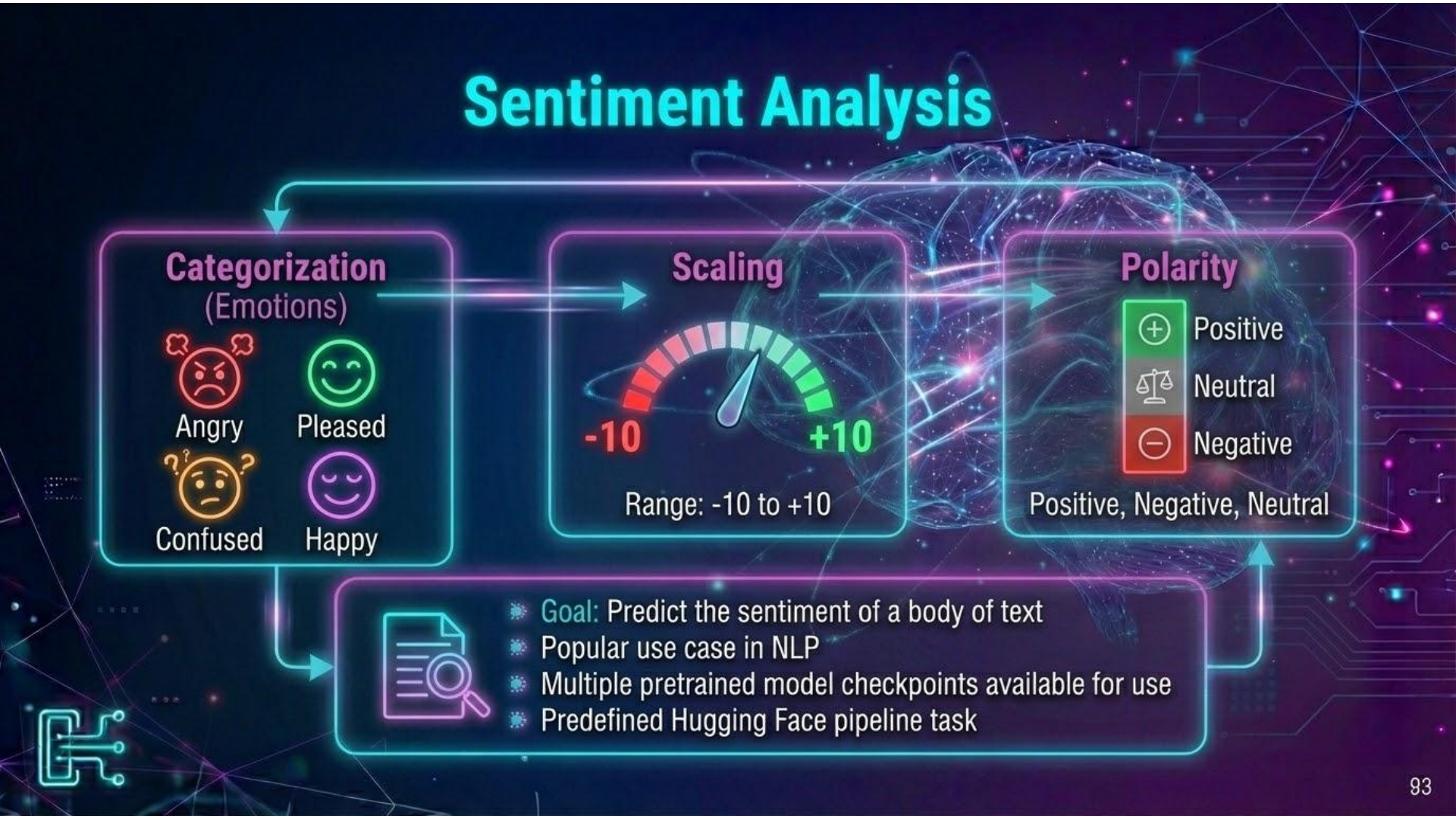
- Text Classification
- Sentiment Analysis
- Spam Detection

Text Classification: Pipeline



Sentiment Analysis

Sentiment Analysis



Sentiment Analysis: what models to use?

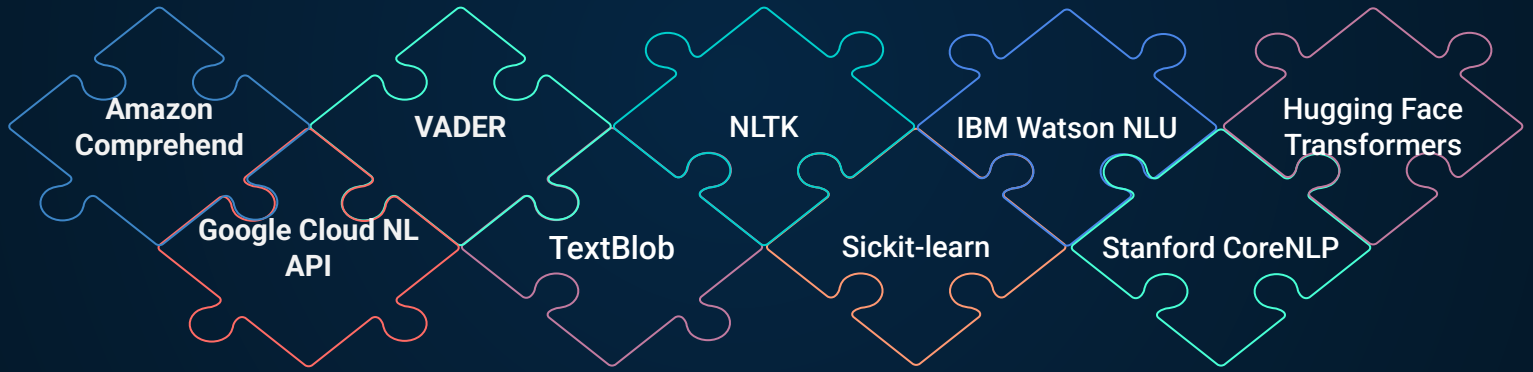
Machine Learning-based methods

- Naive Bayes
- SVM
- Logistic Regression
- Decision Trees

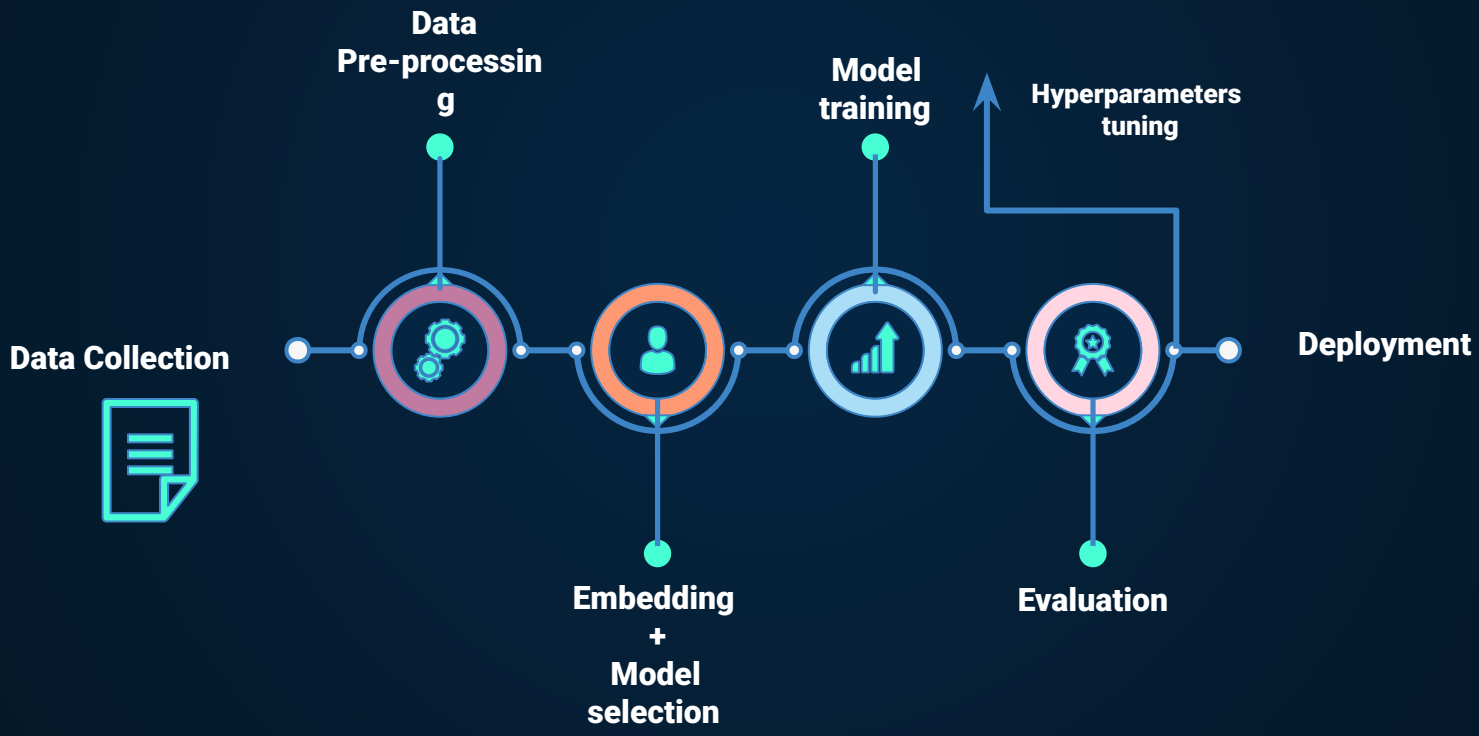
Deep Learning-based methods

- CNNs
- RNNs
- LSTM
- Transformers

Sentiment Analysis: What tools to use?



Sentiment Analysis: Pipeline



Projet : Classification des spams

1. **Machine Learning Supervisé**

Utilisez un algorithme (SVM, Naive Bayes, Regression Logistique, Arbre de Décision**) pour prédire si un mail est un spam ou pas**

2. **Neural Networks :**

Utilisez un réseau neuronal basique ou LSTM pour prédire si un mail est un spam ou non

3. **Evaluation :**

Comparez les deux modèles :

Accuracy
Précision
Recall
F1-Score

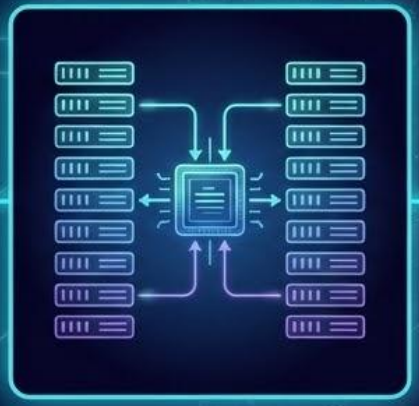
Machine Translation

MACHINE TRANSLATION

The process that uses models to translate text from one natural language to another.
Machine Translation on the web started with Systran offering free translation of small texts (1996).

DICTIONARY-BASED MACHINE TRANSLATION

The words will be translated as a dictionary does – word by word, usually without much correlation of meaning between the words.



EXAMPLE-BASED MACHINE TRANSLATION

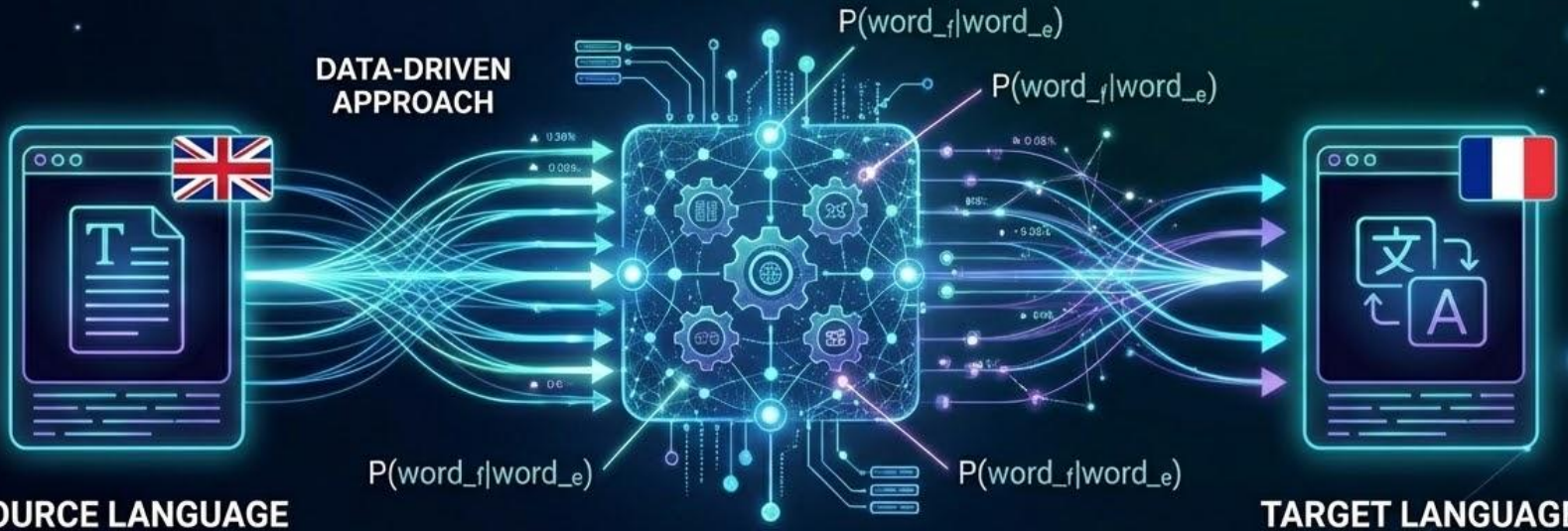
Use of a bilingual corpus with parallel texts as its main knowledge.

RULE-BASED MACHINE TRANSLATION

More information about the linguistics of the source and target languages, using the syntactic rules and semantic analysis of both languages.

STATISTICAL MACHINE TRANSLATION

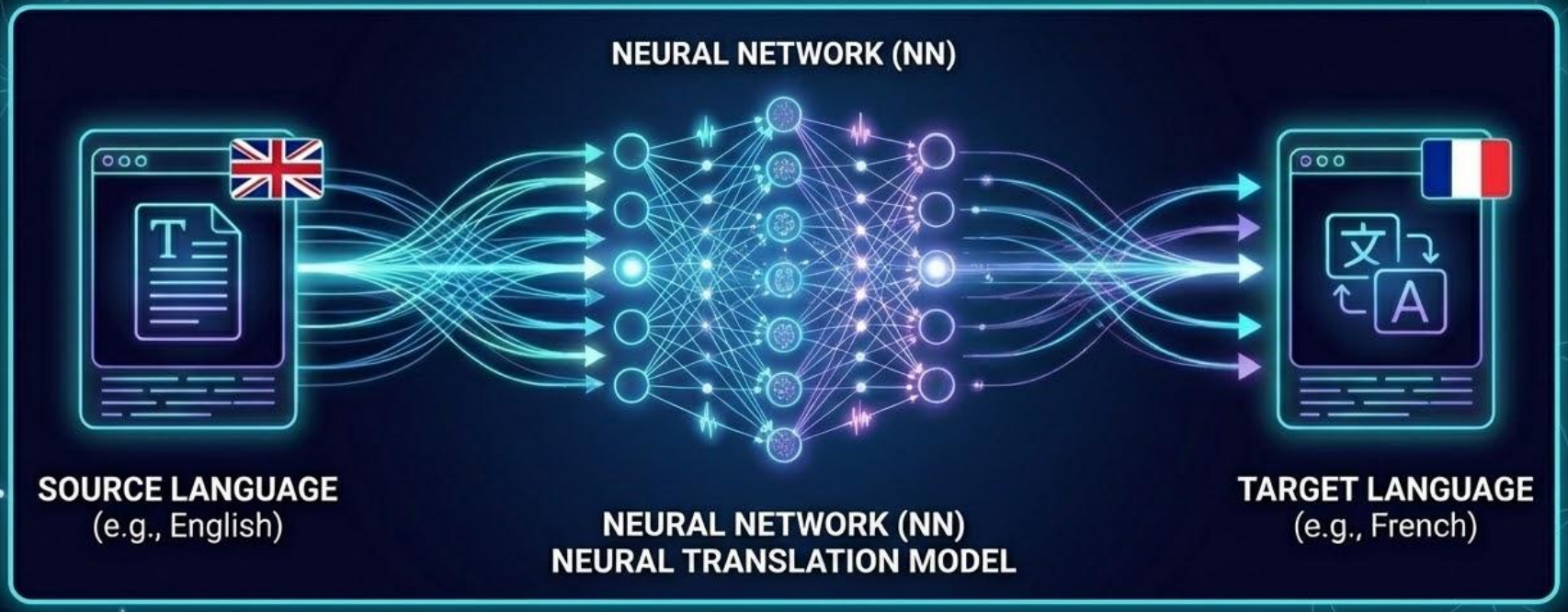
Data-driven approach to estimate translation probabilities between source and target languages using statistical models.



NEURAL MACHINE TRANSLATION

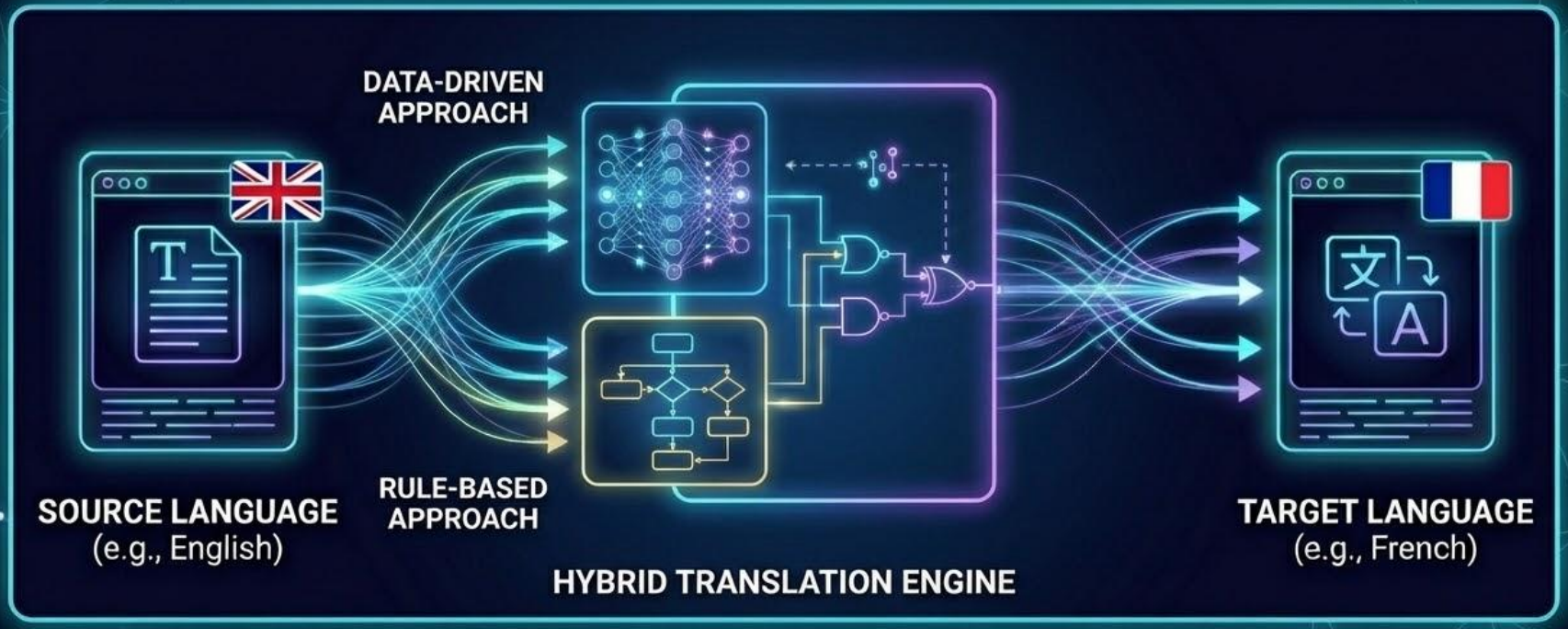
Machine Translation

Data Driven Approach that uses NNs to learn the translation mappings between languages.

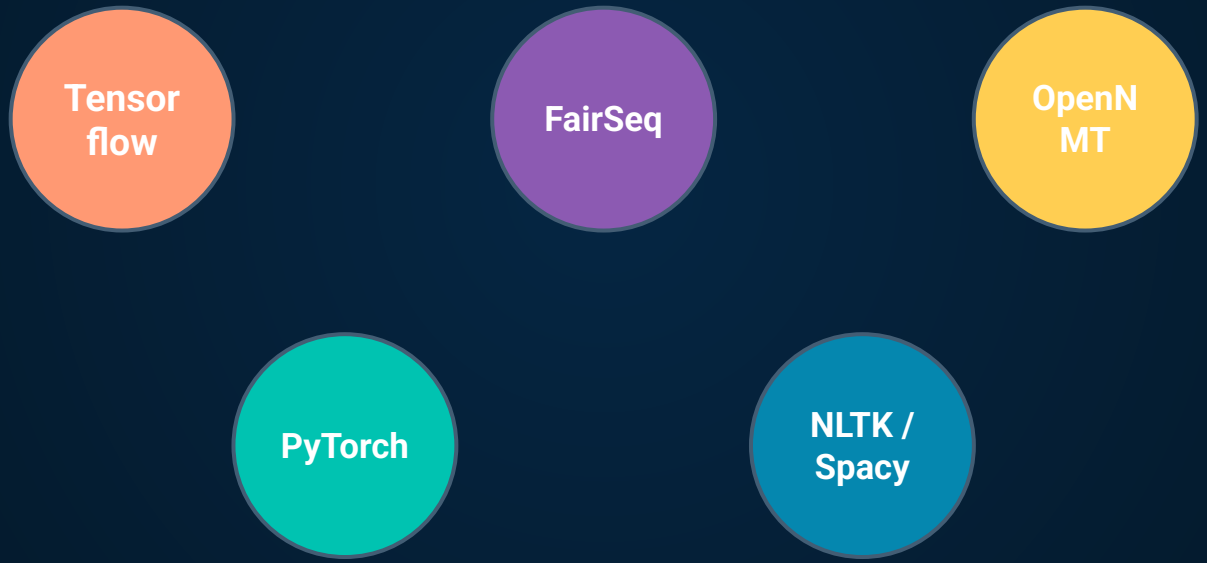


HYBRID MACHINE TRANSLATION

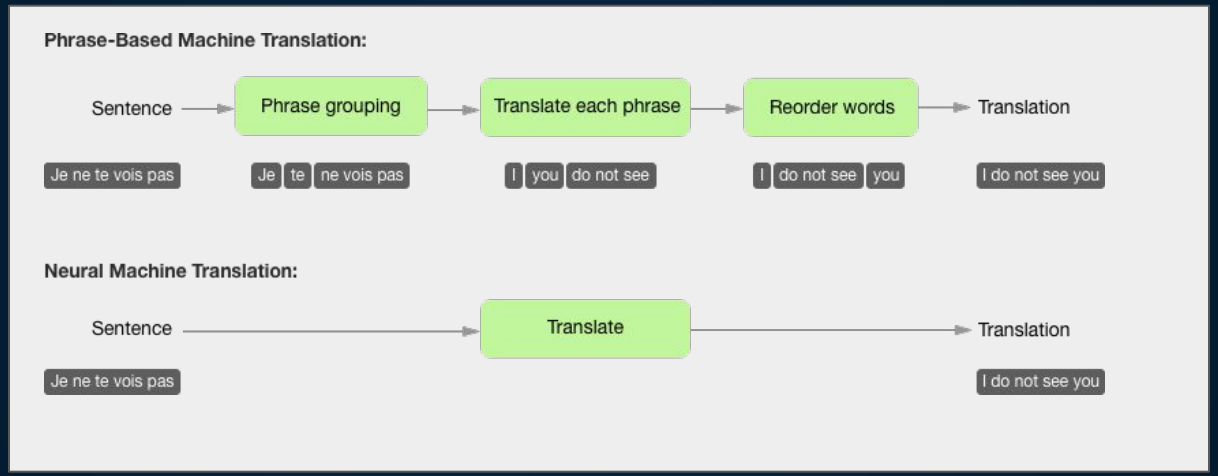
Combination of data-driven and rule-based approaches.



Machine Translation: tools



Machine Translation: Example



Machine Translation: Challenges

- Ambiguity
- Language Complexity
- Vocabulary size
- Domain specificity
- Cultural differences
- Data quality & quantity
- Computing resources

Question-Answering

Question Answering (QA)

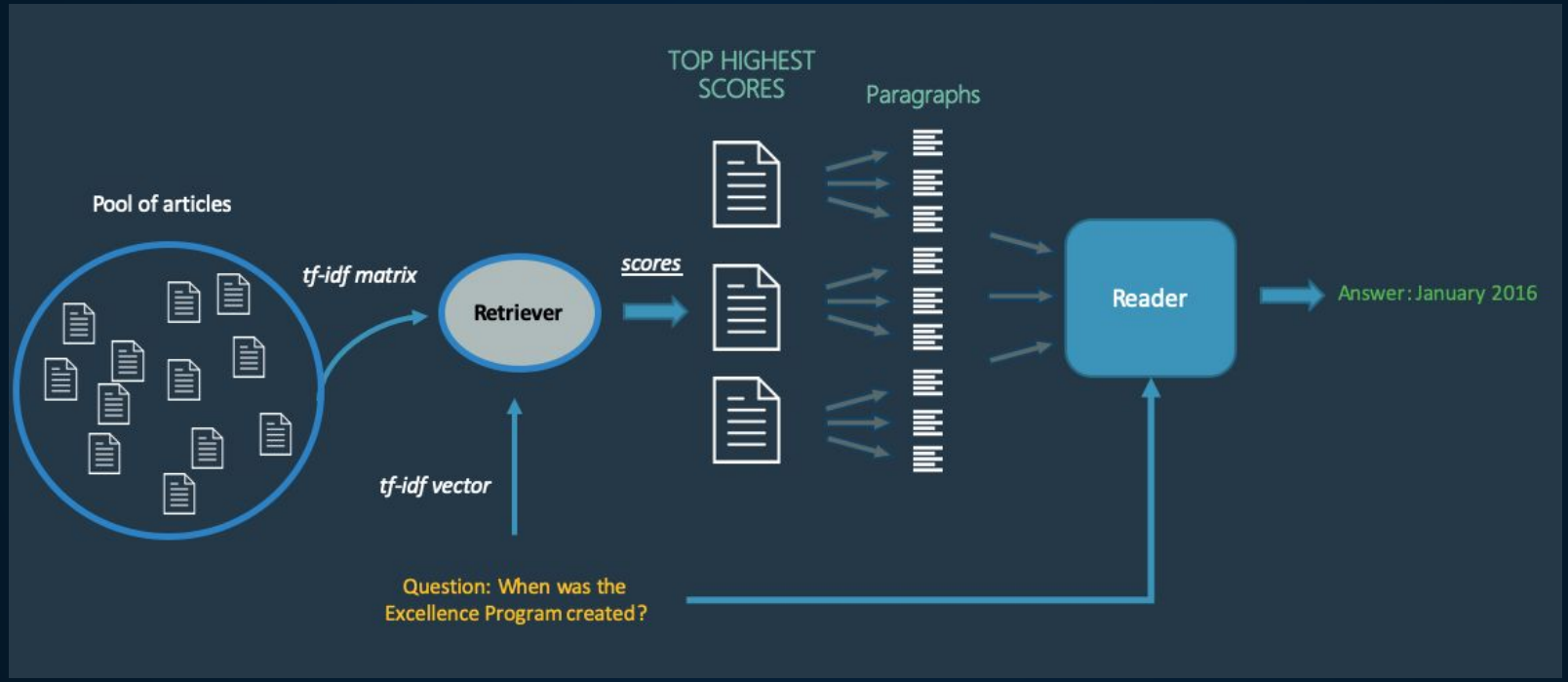
Question-answering (QA) is a subfield of NLP that involves developing algorithms and models that can automatically answer questions posed in natural language.

Question Analysis

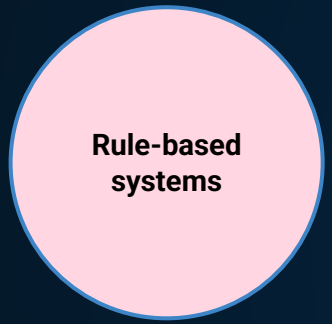
Information Retrieval

Answer generation & Answer ranking

Question Answering (QA)

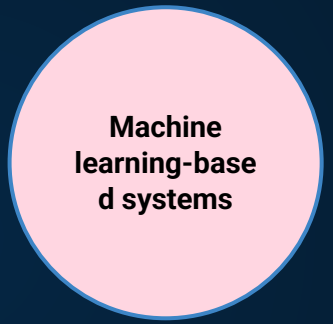


Question Answering: methods

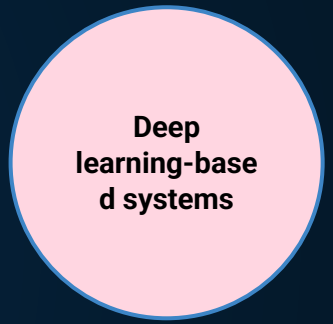


use a set of predefined rules to analyze questions and generate answers

retrieve relevant information from a knowledge base or corpus of text and use it to generate answers

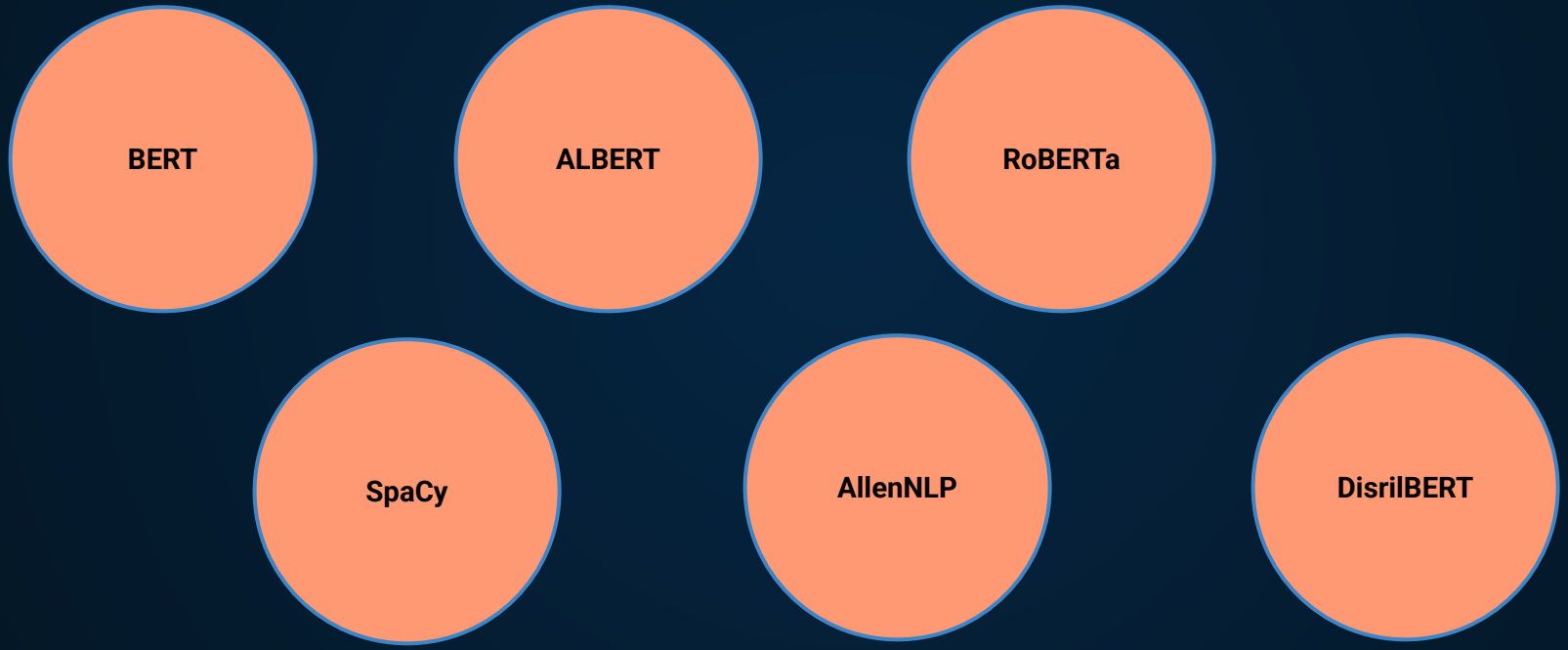


use machine learning algorithms to learn from examples and generate answers



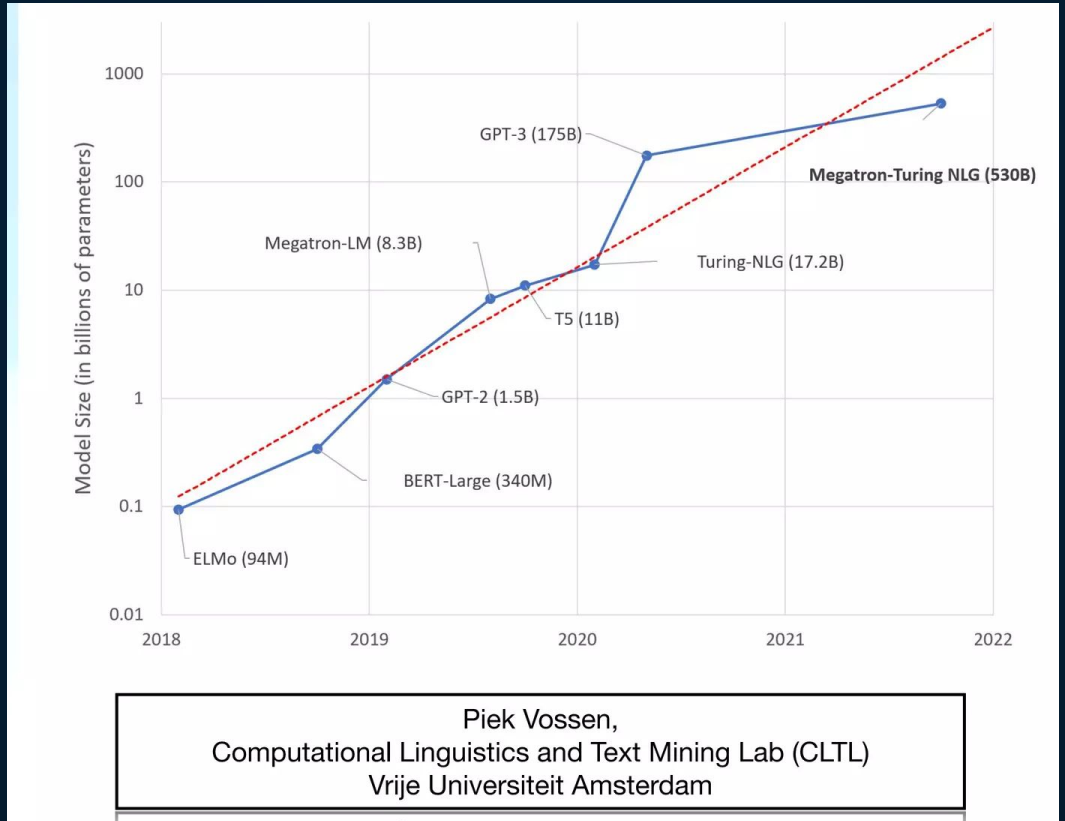
can be highly effective for answering complex questions and can achieve state-of-the-art performance on some benchmarks

Question Answering: tools



Large Language Models

Large Language Models (LLMs)

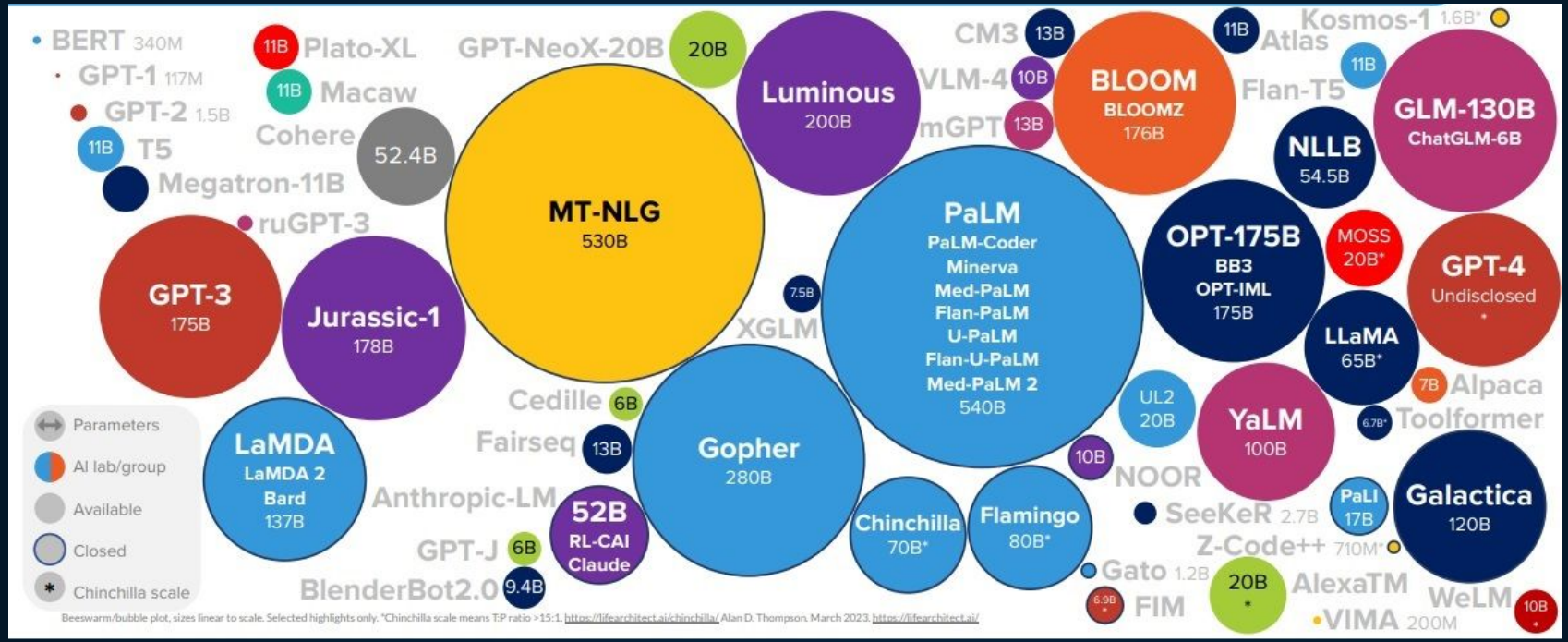


Piek Vossen,
Computational Linguistics and Text Mining Lab (CLTL)
Vrije Universiteit Amsterdam

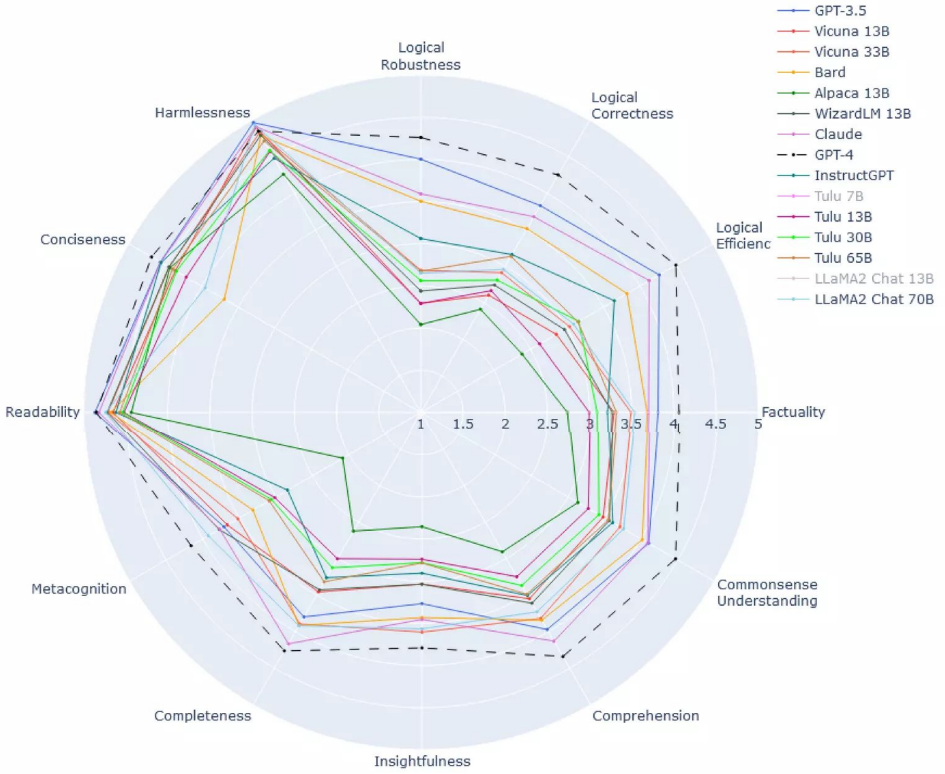
More advanced language models

- PaLM
- GPT4
- LLAMA
- Gemini

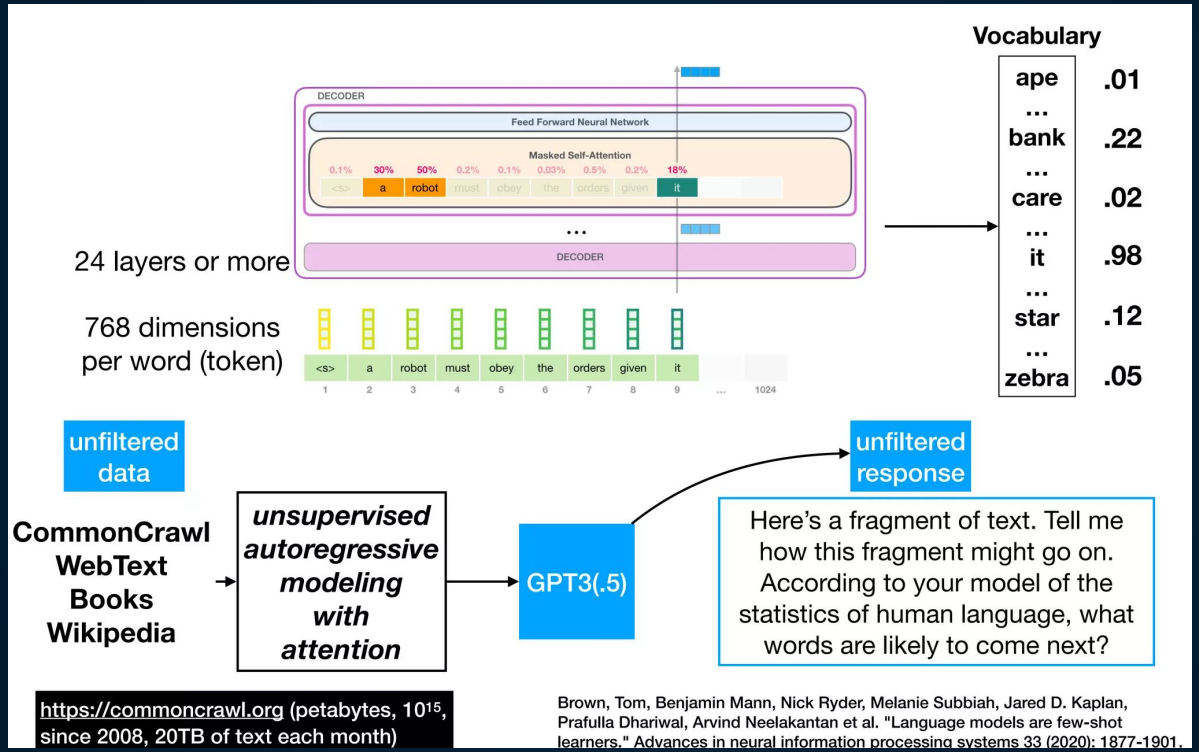
More LLMs


BERT 340M
GPT-1 117M
GPT-2 1.5B
11B T5
Megatron-11B
11B Plato-XL
11B Macaw
Cohere
52.4B
ruGPT-3
GPT-3
175B
Jurassic-1
178B
LaMDA
LaMDA 2
Bard
137B
GPT-NeoX-20B
20B
MT-NLG
530B
Luminous
200B
7.5B
XGLM
Cedille 6B
Fairseq
13B
Anthropic-LM
52B
RL-CAI
Claude
GPT-J 6B
BlenderBot2.0 9.4B
Gopher
280B
Chinchilla
70B*
Flamingo
80B*
CM3 13B
VLM-4 10B
mGPT 13B
PaLM
PaLM-Coder
Minerva
Med-PaLM
Flan-PaLM
U-PaLM
Flan-U-PaLM
Med-PaLM 2
540B
10B
11B Atlas
BLOOM
BLOOMZ
176B
Kosmos-1 1.6B*
11B
Flan-T5
NLLB
54.5B
GLM-130B
ChatGLM-6B
OPT-175B
BB3
OPT-IML
175B
MOSS
20B*
GPT-4
Undisclosed
LLaMA
65B*
7B Alpaca
6.7B* Toolformer
UL2
20B
YaLM
100B
NOOR
SeeKeR 2.7B
PaLI
17B
Galactica
120B
Z-Code++ 710M*
Gato 1.2B
6.9B*
FIM
20B
AlexaTM
WeLM
10B
VIMA 200M
Parameters
AI lab/group
Available
Closed
Chinchilla scale
Beeswarm/bubble plot, sizes linear to scale. Selected highlights only. *Chinchilla scale means T:P ratio >15:1. https://lifearchitect.ai/chinchilla/ Alan D. Thompson. March 2023. https://lifearchitect.ai/

Comparing LLMs



Generative Pre-trained Transformers (GPT)

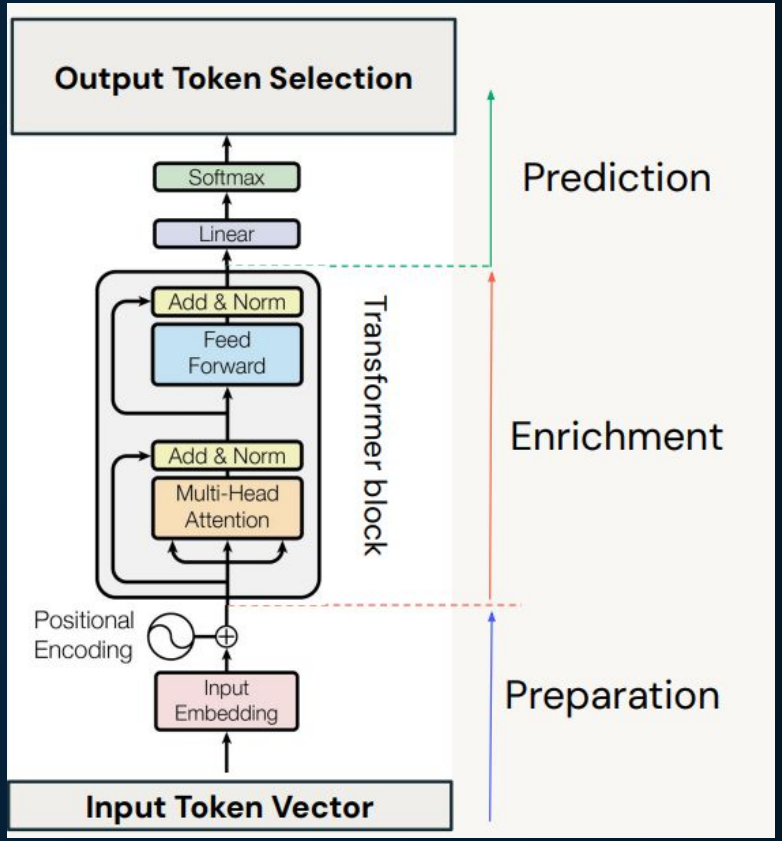


GPT

Architecture:
Transformer architecture
→ Multi-head attention mechanisms
→ Residual connections

Training:
Large datasets of unlabelled text
→ Generation of human-like text

Applications:
Task-specific GPTs (model fine-tuning)



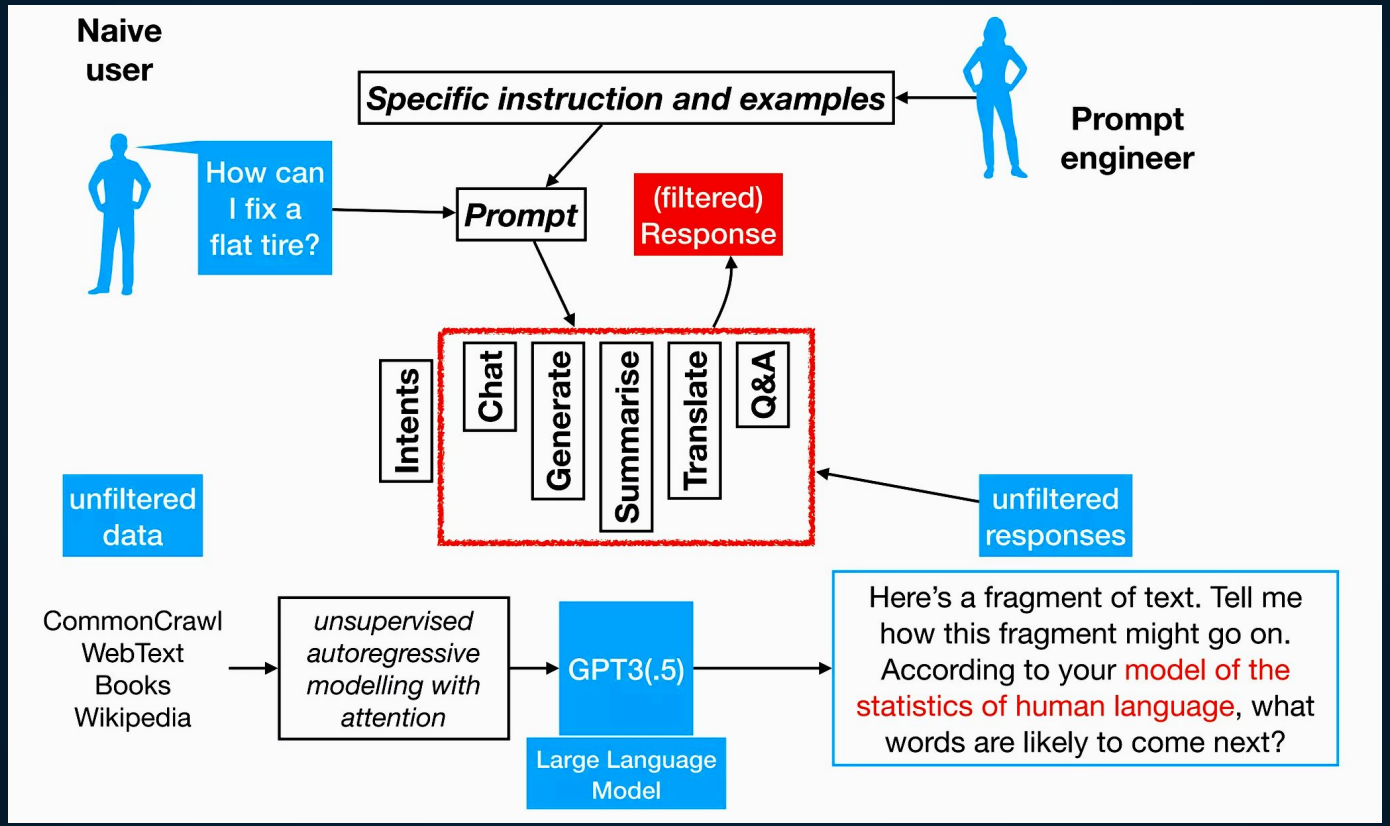
Chat GPT

ChatGPT is an Artificial Intelligence Chatbot developed by OpenAI and launched in November 2022.

It is built on top of OpenAI's GPT-3.5 and GPT-4 families of Large Language Models and has been fine-tuned using both supervised and reinforcement learning techniques.

How does it work?

Chat GPT


Naive user
Specific instruction and examples
Prompt engineer
How can I fix a flat tire?
Prompt
(filtered) Response
Intents
Chat
Generate
Summarise
Translate
Q&A
unfiltered data
unfiltered responses
CommonCrawl
WebText
Books
Wikipedia
unsupervised autoregressive modelling with attention
GPT3(.5)
Large Language Model
Here's a fragment of text. Tell me how this fragment might go on. According to your model of the statistics of human language, what words are likely to come next?

Projet : Machine Translation

1. **GPT**

En utilisant une clé d'API OpenAI, GougleTrans, OU MarianMT, traduisez une partie des emails en Français

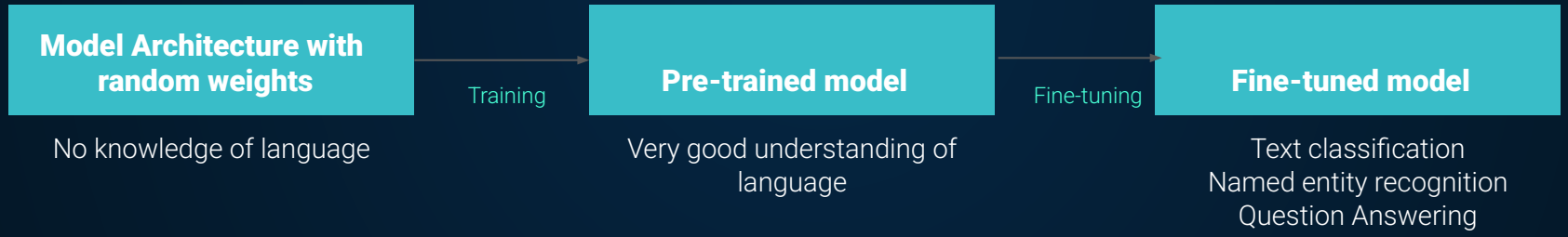
2. **Evaluation :**

Utilisez le Blue score pour évaluer les traductions faites par vos modèles

Transfer Learning

Transfer learning

2 main tasks:
- Pre-training
- Fine tuning



Benefits

- Faster development
- Less data to fine-tune
- Excellent results (computer vision as an example)

Transfer learning

Hugging Face

Hugging Face

Hugging face is an online community and platform that provides community-sourced building blocks for advanced deep learning applications

https://huggingface.co

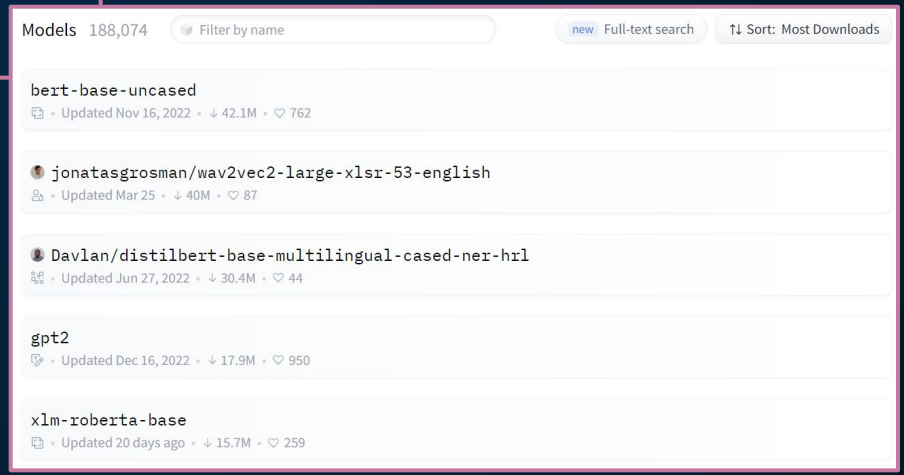
Features:

- Open source community
- Mainly focused on transformer models
- Pretrained and packaged models for popular ML tasks
- Preformatted and labeled datasets
- A library and an API for easy training and inference
- Supports documentation and code for tensorflow
- Additional ML building blocks like tokenizers

Pre-trained models in hugging face

Hosts a huge repository of pretrained model checkpoints
Searchable by task, ML framework, and architecture
Model card provides background and documentation
Hosted inference API - to run a quick example

https://huggingface.co/models

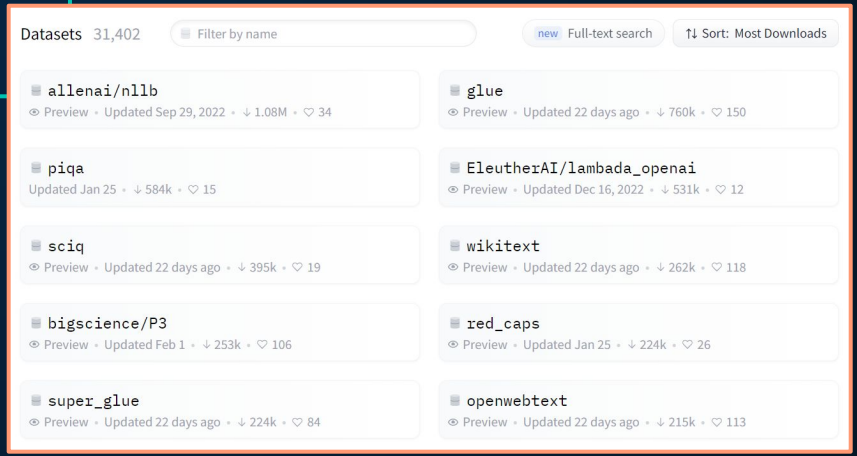


Datasets in Hugging Face

Datasets collected for a variety of tasks and languages
Can be loaded and used with a few lines of code in Apache Arrow format

Datasets package allows easy downloading, caching, and use of datasets

https://huggingface.co/datasets



Evaluation methods & tools

Evaluation methods

BLEU (Bilingual Evaluation Understudy)

ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

NIST (National Institute of Stetards et Technology)

METEOR (Metric for Evaluation of Translation with Explicit Ordering)

WER (Word Error Rate)

Ethics & Challenges in NLP

Ethical issues

Ethical issues of NLP

Fairness and Bias in NLP

Sources of bias in NLP:

- **The data used to train the models**
- **The algorithms and techniques employed**
- **The people who create and use the models**

Fairness and bias in NLP can have a significant impact on the results produced, and it is crucial to address these issues to ensure equitable outcomes.

Ethical issues of NLP

Privacy Concerns in NLP

Sensitive data:

- **Personal information**
- **Medical Records**
- **Financial information**

Risks:

- **Data exposure**
- **Data breaches**

It is essential to establish best practices for data protection and user privacy in NLP applications to ensure the ethical and responsible use of data.

Future Perspectives

Future Perspectives

Growing need in different fields:

- **Healthcare**
- **Finance**
- **Academics**
- **Scientific Research**
- **Marketing**

Opportunities:

- **Improve patients' experience and hospital stays**
- **Automate processes**
- **Gain insights from large amounts of unstructured data**

Don't forget the ethical issues ;)

Any questions?

Useful Resources

BIBLIOGRAPHY

- Anandarajan M, Hill C, Nolan T. Text preprocessing. In Practical Text Analytics 2019 (pp. 45-59). Springer, Cham.
- Chapman WW, Bridewell W, Hanbury P, Cooper GF, Buchanan BG. A simple algorithm for identifying negated findings and diseases in discharge summaries. Journal of biomedical informatics. 2001 Oct 1;34(5):301-10.
- Clément Dalloux, Vincent Claveau, and Natalia Grabar. Détection de la négation : corpus français et apprentissage supervisé. In SIIM 2017 - Symposium sur l'Ingénierie de l'Information Médicale, 1–8. Toulouse, France, November 2017. URL: https://hal.archives-ouvertes.fr/hal-01659637.
- Denny MJ, Spirling A. Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it. Political Analysis. 2018 Apr;26(2):168-89.
- Uysal AK, Gunal S. The impact of preprocessing on text classification. Information processing & management. 2014 Jan 1;50(1):104-12.
- Vijayarani S, Ilamathi MJ, Nithya M. Preprocessing techniques for text mining-an overview. International Journal of Computer Science & Communication Networks. 2015 Feb;5(1):7-16
- Zhang Y, Tiryaki F, Jiang M, Xu H. Parsing clinical text using the state-of-the-art deep learning based parsers: a systematic comparison. BMC medical informatics and decision making. 2019 Apr;19(3):51-8.
- Multi-step Transfer Learning in Natural Language Processing ... - Springer. https://link.springer.com/article/10.1007/s11063-024-11526-y.
- A Review of Recent Work in Transfer Learning and Domain Adaptation for .... https://www.thieme-connect.com/products/ejournals/pdf/10.1055/s-0041-1726522.pdf.
- NLP in Healthcare: Techniques, Applications, Challenges & Future. https://anubrain.com/nlp-in-healthcare/.

OTHER RESOURCES

- https://www.nltk.org/
- https://spacy.io/
- https://radimrehurek.com/gensim/
- https://huggingface.co/blog/bert-101
- https://stanfordnlp.github.io/stanza/
- https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp
- https://monkeylearn.com/natural-language-processing/
- https://www.slideserve.com/shubha/text-preprocessing
- https://aphp.github.io/edsnlp/latest/pipelines/qualifiers/negation/#performance
- Natural Language Processing with Python" by Steven Bird, Ewan Klein, and Edward Loper
- "Speech and Language Processing" by Daniel Jurafsky and James H. Martin
- "Deep Learning for Natural Language Processing" by Palash Goyal, Sumit Pandey, and Karan J
- NLP conferences: ACL, EMNLP, and NAACL
- Online courses and tutorials on platforms: Coursera, Udacity, and edX.