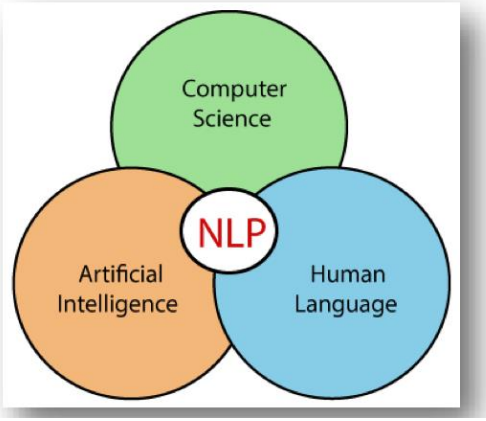
IA E Natural Language Processing

INTELLIGENZA ARTIFICIALE E NATURAL LANGUAGE PROCESSING



- ELABORAZIONE DEL LINGUAGGIO NATURALE
- **DEFINIZIONE**: PROCESSO CHE PERMETTE AD UNA MACCHINA DI COMPRENDERE UN TESTO, PAROLE PRONUNCIATE, IN MODO AUTOMATIZZATO COME SE SI TRATTASSE DI ESSERE UMANI
- INFORMATICA + INTELLIGENZA ARTIFICIALE + LINGUISTICA
- ALGORITMI DI INTELLIGENZA ARTIFICIALE
- OBIETTIVO: LEGGERE, DECIFRARE, COMPRENDERE E DARE UN SENSO AI LINGUAGGI UMANI IN UN MODO VALUTABILE

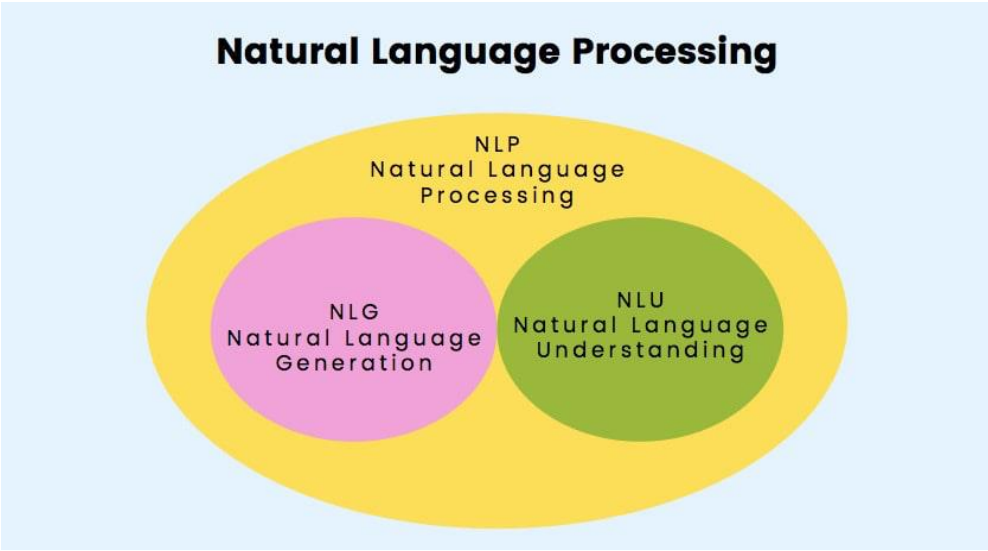
- *Natural Language Processing* (NLP) è la disciplina che permette la realizzazione di macchine in grado di manipolare il linguaggio umano, o i testi prodotti dagli esseri umani, tenendo conto di come essi sono realizzati;

- La disciplina NLP può essere suddivisa in due settori di sviluppo:

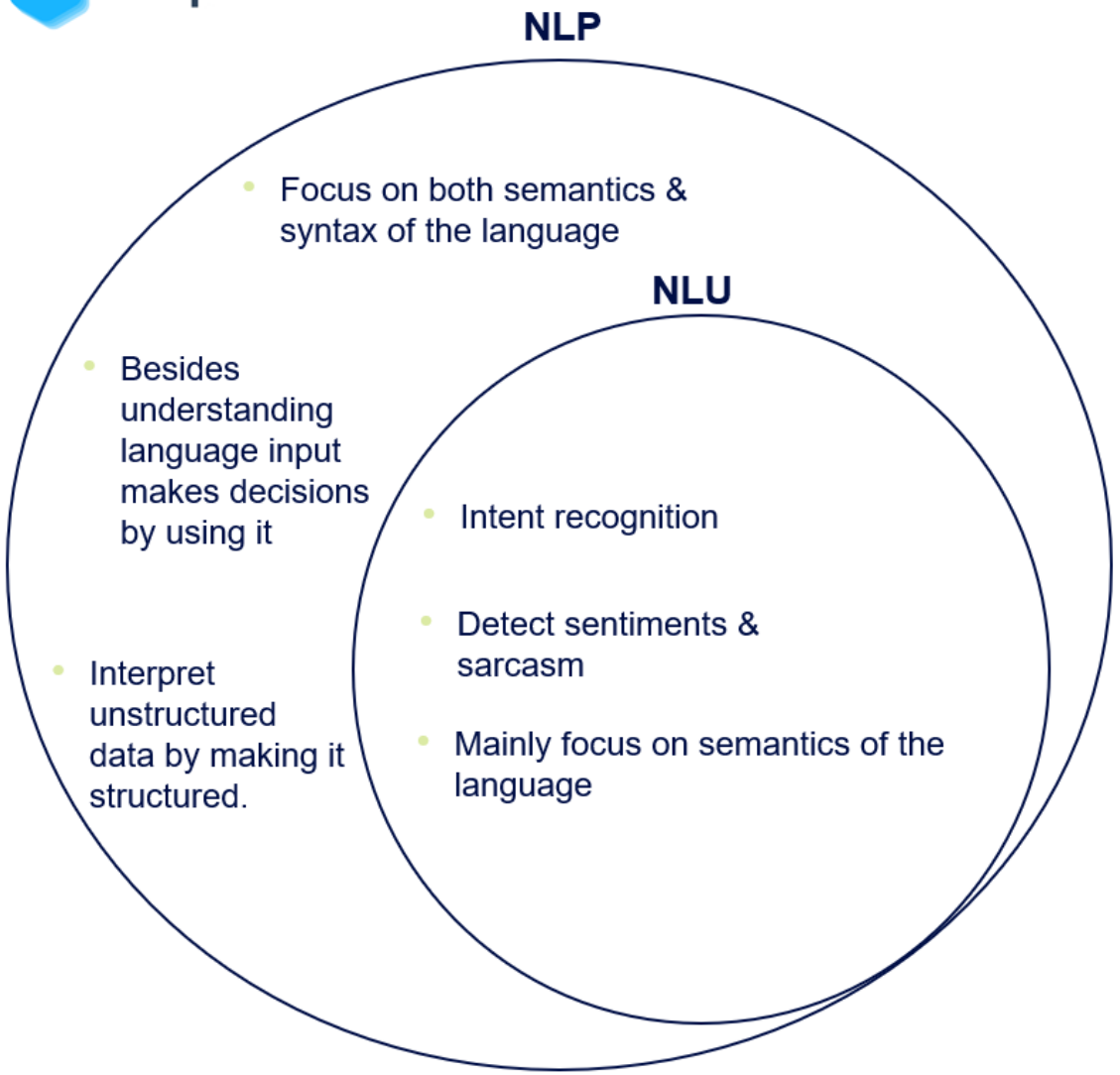
  - *Natural Language Understanding* (NLU), che si focalizza sull'analisi semantica o sull'individuazione del significato contenuto dalle parole;

  - *Natural Language Generation* (NLG),che si focalizza sulla generazione di testi;

  - NLP è separato ma spesso opera insieme alla disciplina della *Speech recognition*, che studia come separare le componenti del parlato in parole , trasformando i suoni in testo e viceversa.



Natural Language Understanding (NLU): si focalizza sull'analisi semantica o sull'individuazione del significato contenuto dalle parole;



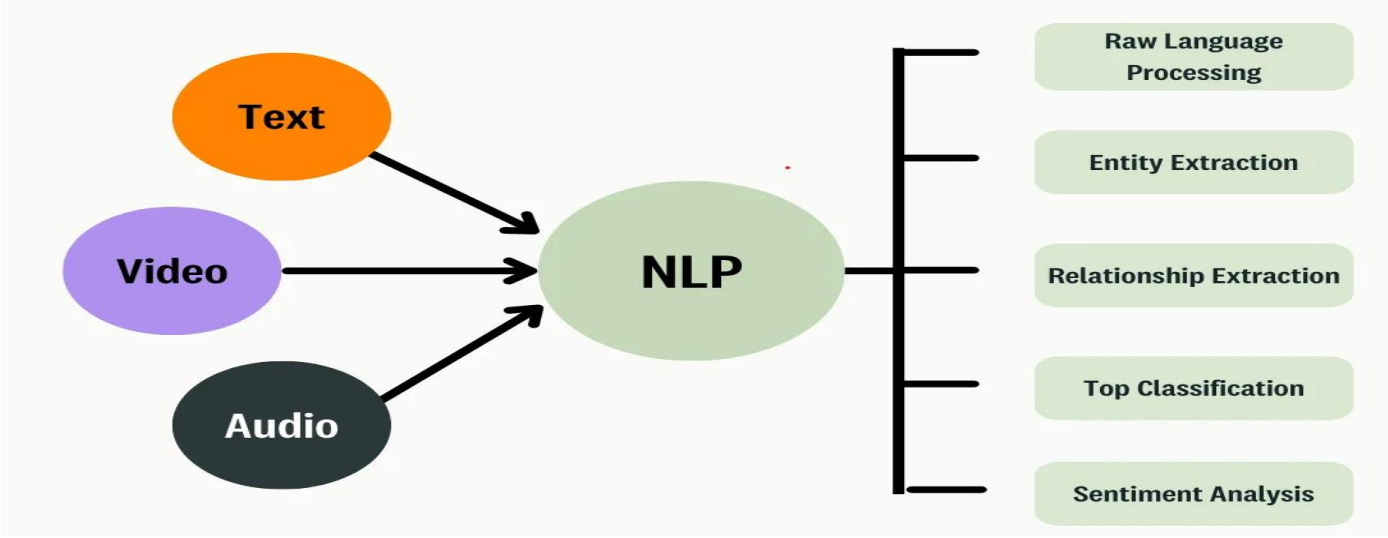
natural-language-processing-graph-exemple.webp

- NLP è utilizzato per un gran numero di compiti inclusi la risposta a domande, la classificazione di testi, la conversazione con utenti;
- NLP è utilizzata per la risoluzione di vari compiti.

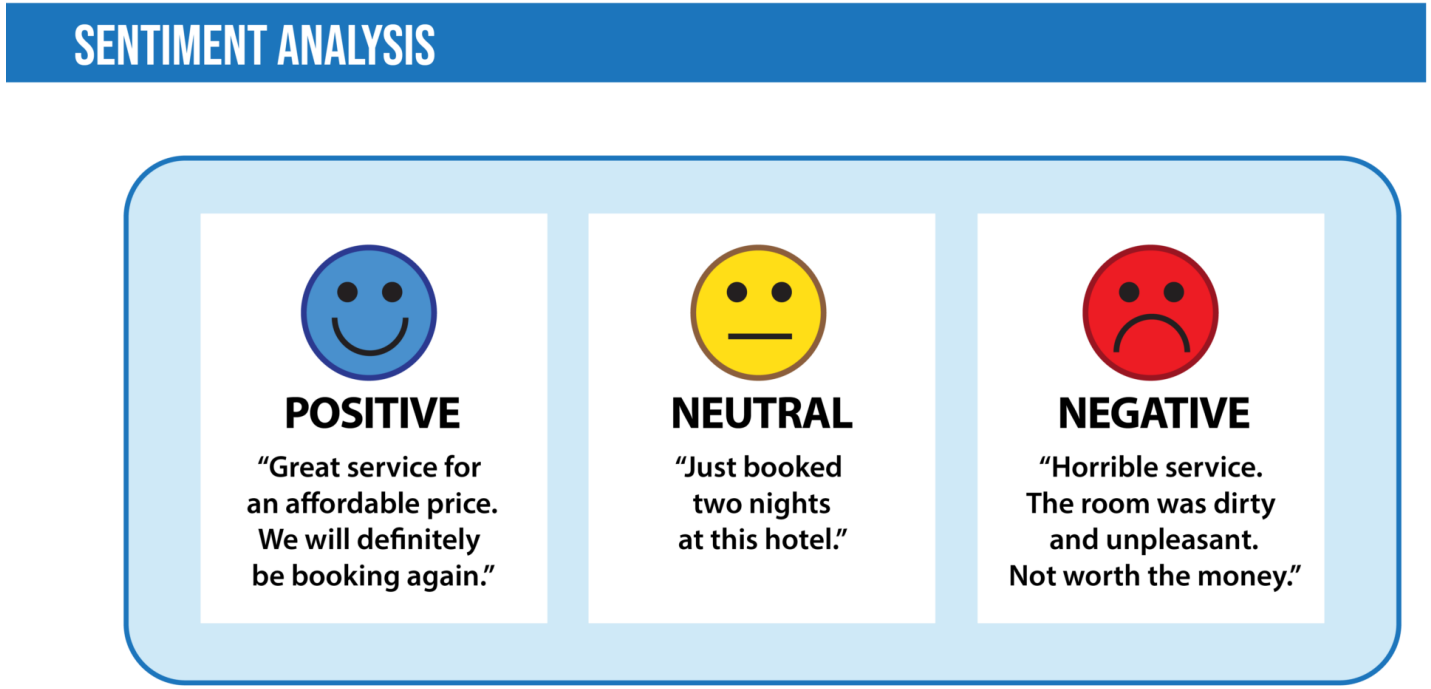
Il **dialogo tra uomo e macchina** coinvolge diversi aspetti, quali fonetica, fonologia, morfologia, sintassi, semantica, pragmatica e il discorso nel suo complesso.

Di conseguenza, sono numerosi i **task di NLP** che automatizzano queste aree, ad esempio compiti semplici come:

- il riconoscimento della lingua;
- la scomposizione della frase in unità elementari;
- l'analisi semantica;
- l'analisi del sentiment.

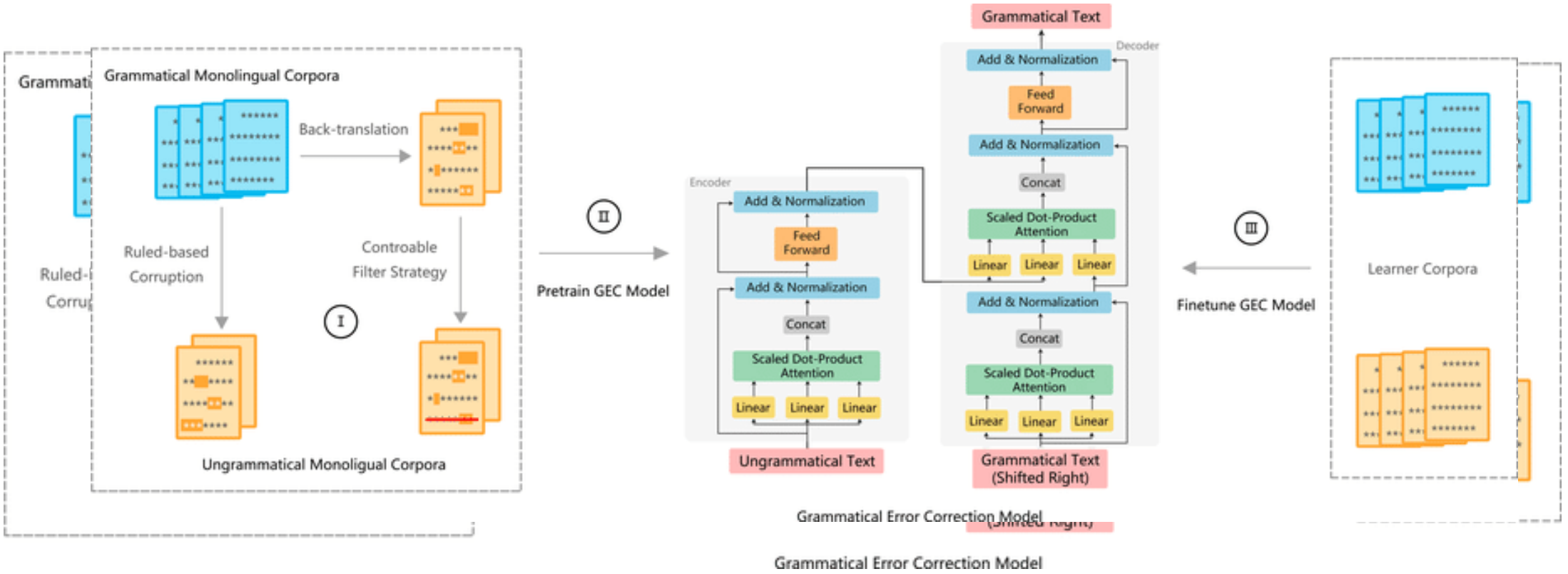


- **sentiment analysis** è il processo finalizzato alla classificazione degli intenti racchiusi nel testo;
- Generalmente gli input di un modello per la classificazione dei sentimenti può essere un pezzo di un testo e l'output invece è la probabilità the il sentimento espresso sia positivo, negativo o neutro nei confronti del soggetto della frase;
- Sentiment analysis può essere utilizzato per catalogare sentimenti all'interno di una frase come segnali di instabilità mentale racchiusi all'interno di dialoghi sulla rete.



Given text, sentiment analysis classifies its emotional quality.

Grammatical error correction models codifica le regole grammaticali per correggere la grammatica all'interno di un testo. Sistemi di correzione automatica del testo, sulla base della grammatica quali Microsoft word hanno integrata questa funzione;



- **Text generation**, natural language generation (NLG) model, produce testo simile a quello generato dall'uomo.
- I modelli possono essere affinati per produrre testi in diversi formati e lingue, includendo. La generazione di testo viene raggiunta attraverso Markov processes, LSTMs, BERT, GPT-2, LaMDA, e altri prodotti quali sistemi di autocompletamento e chatbots:
  - **Autocompletamento** predice la parola che verrà; tali sistemi sono utilizzati, a vari livelli di complessità, in applicazioni quali WhatsApp. Google. Uno dei più famosi sistemi di autocompletamento è GPT-2, che viene utilizzato in per scrivere articoli, canzoni, etc,
  - **Chatbots** sistemi che automatizzano la risposta simulando un essere umano . Abbiamo due tipologie di chatbots:
    - Database query: post di aver un database disponibile di domande e risposte e lo utilizziamo per far porre all'utente domande;
    - Conversation generation: sistemi che sono in grado di gestire una conversazione con esseri umani (Google's LaMDA).

Data preprocessing

Prima di utilizzare un testo per le nostre applicazioni NLP, il testo spesso necessita di essere preprocessato per migliorare le prestazioni del modello stesso o per trasformare le parole e i caratteri in un formato che il modello di NLP possa comprendere.

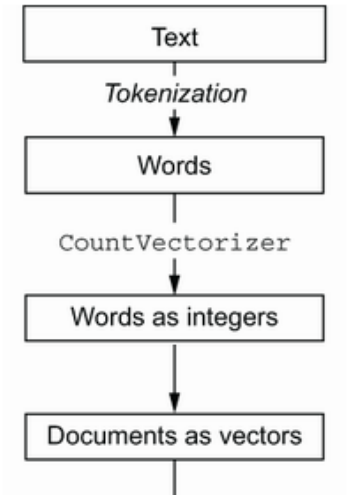
Per fare questo possono essere utilizzate varie tecniche:

- *Tokenizzazione: suddividere il testo in piccoli "frammenti"*
- *Stemming* : riferito ad un processo che "taglia" la parte finale delle parole, eliminando cosi I suffissi;
- *Lemmatization:* tecnica che sfrutta l'analisi morfologica delle parole puntando ad ottenere un formato analogo a quello presente per la stessa parola in un vocabolario, che è conosciuto come lemma;
- Esempio: consideriamo la parola "visto":
  - Utilizzando lo *stemming* potremmo ottenere solo "vi"
  - Utilizzando la *lemmatization* potremmo ottenere sia "vedere" sia "visto": questo risultato può dipendere dal fatto che il termine sia utilizzato come verbo, o come nome o come aggettivo; Le due valutazioni differiscono in quanto spesso nello convergono forme contratte mentre nel lemmatization comunemente convergono diverse inflessioni.

- Tokenizzazione
- Si suddivide il testo in piccoli frammenti (token) secondo un certo set di regole;

Ogni task NLP richiede normalizzazione del testo:

1. Segmentando (“*tokenizzare*”) le paro[illegible]
2. Normalizzando le parole
3. Segmentando le frasi

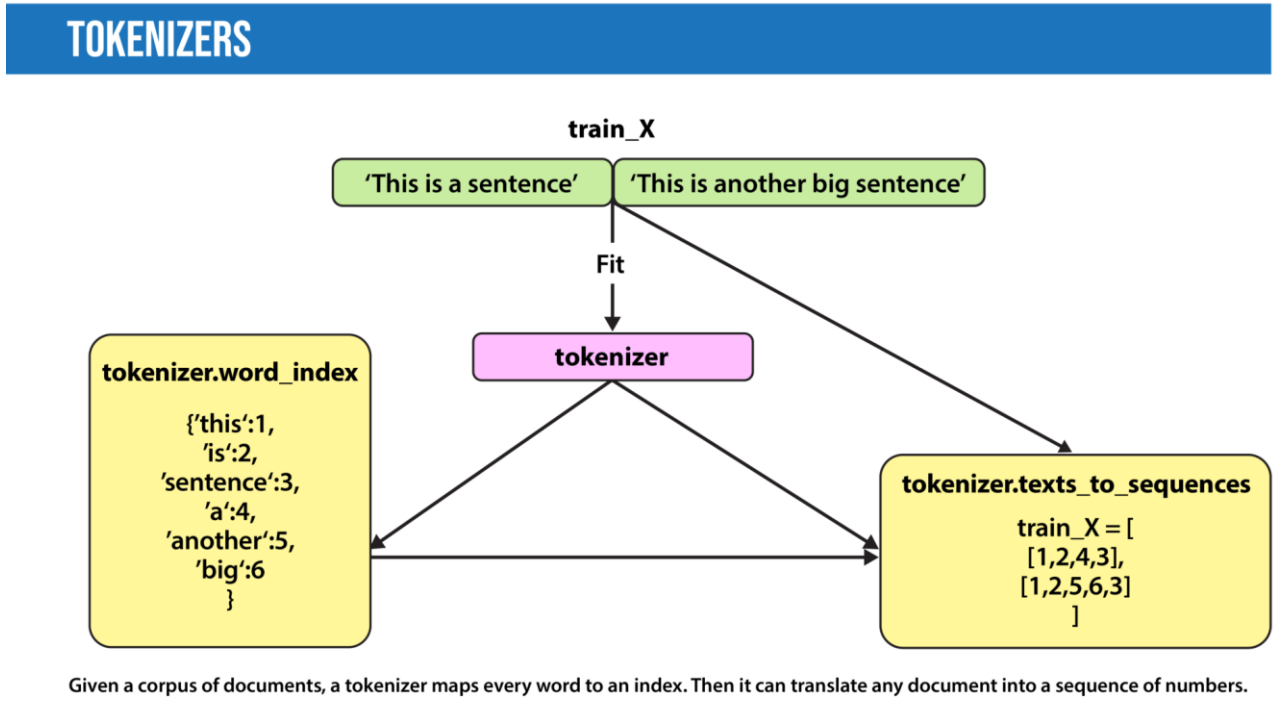


the	cat	sat	on	a	mat
1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1

- cat: 010000
- a: 000010

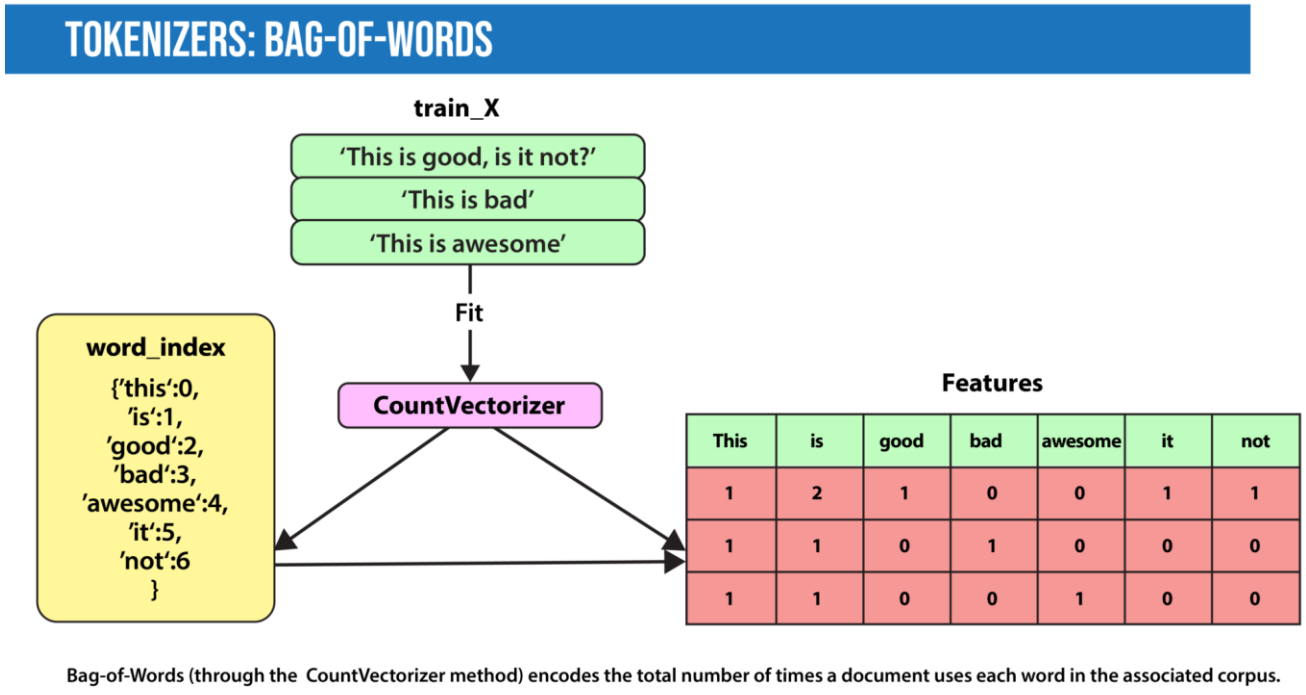
I modelli NLP lavorano trovando relazioni tra le varie parti costituenti un linguaggio , per esempio lettere, parole o frasi trovate in un discorso. Le architetture NLP usano vari metodi per il preprocessing dei dati, l'estrazione delle caratteristiche e la modellizzazione dl linguaggio.

Tokenizzazione: suddivide il testo in singole parole e parti fai parole (congiunzioni). Il risultato generalmente consiste in un indice di parole e un testo "tokenizzato" in cui le parole possono essere rappresentate da un "token" numerico" da utilizzare in applicazioni di deep learning: un metodo che addestri il modello di linguaggio a ignorare token poco importanti migliora l'efficienza.



Given a corpus of documents, a tokenizer maps every word to an index. Then it can translate any document into a sequence of numbers.

Bag-of-Words: Bag-of-Words conta il numero di volte che ciascuna parola o combinazioni di n-parole compaiono in un.Per esempio, nella rappresentazione Bag - of Words crea una rappresentazione **numerica delle frasi considerate** basandosi su quante volte ciascuna parola appare nel documento.



Bag-of-Words (through the CountVectorizer method) encodes the total number of times a document uses each word in the associated corpus.

TF-IDF: in Bag of Words vengono contate le occorrenze di ciascuna parola in un documento. AL contrario con l'algoritmo TF-IDF ciascuna parola viene "pesata" secondo la sua importanza nel documento: per fare questo abbiamo bisogno di conoscere il significato della parola. Per valutare il significato di una parola vengono considerati due aspetti:

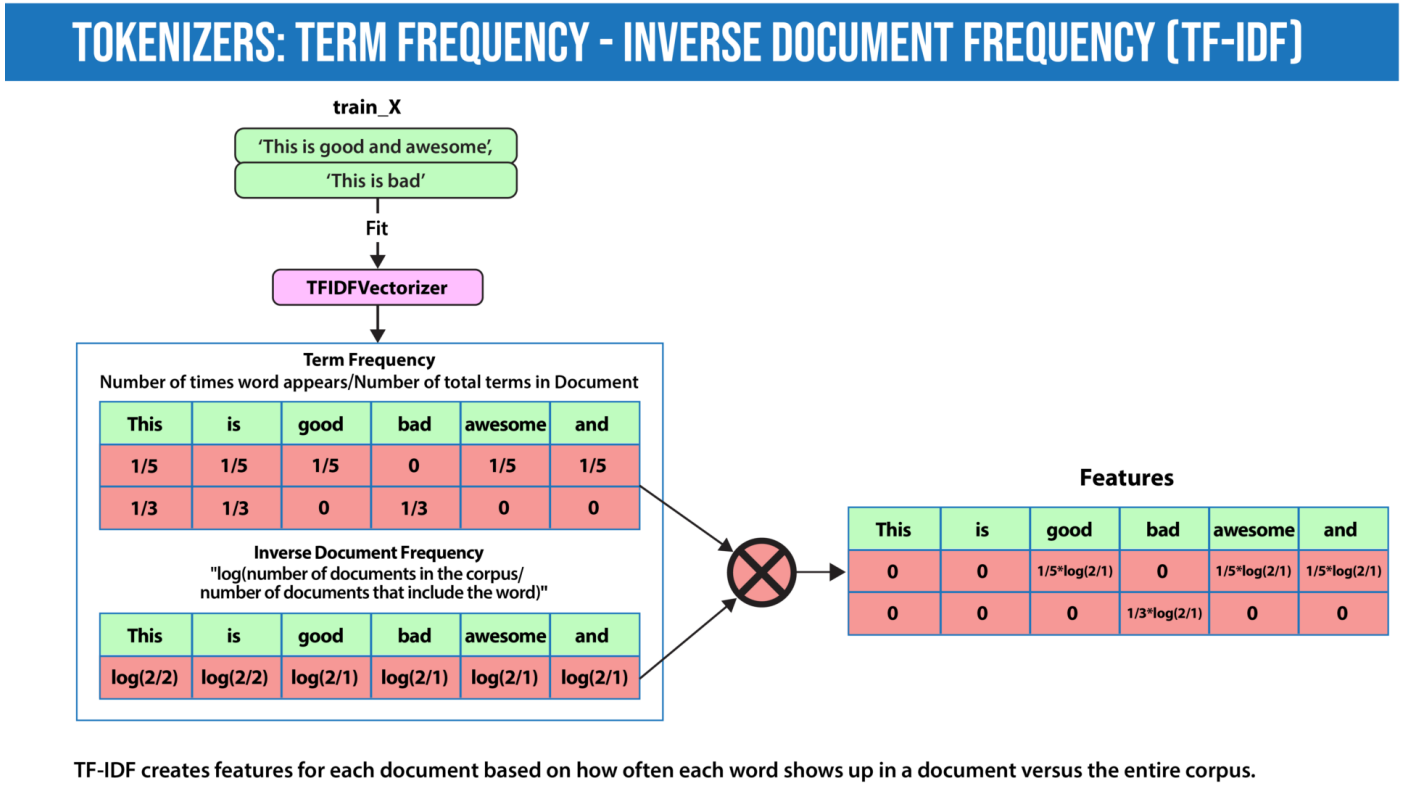
- **Term Frequency(TF) :**quanto importante è la parola nel documento?

TF(di una parola in un documento)= Numero di volte che la parola appare nel documento// Numero di parole nel documento

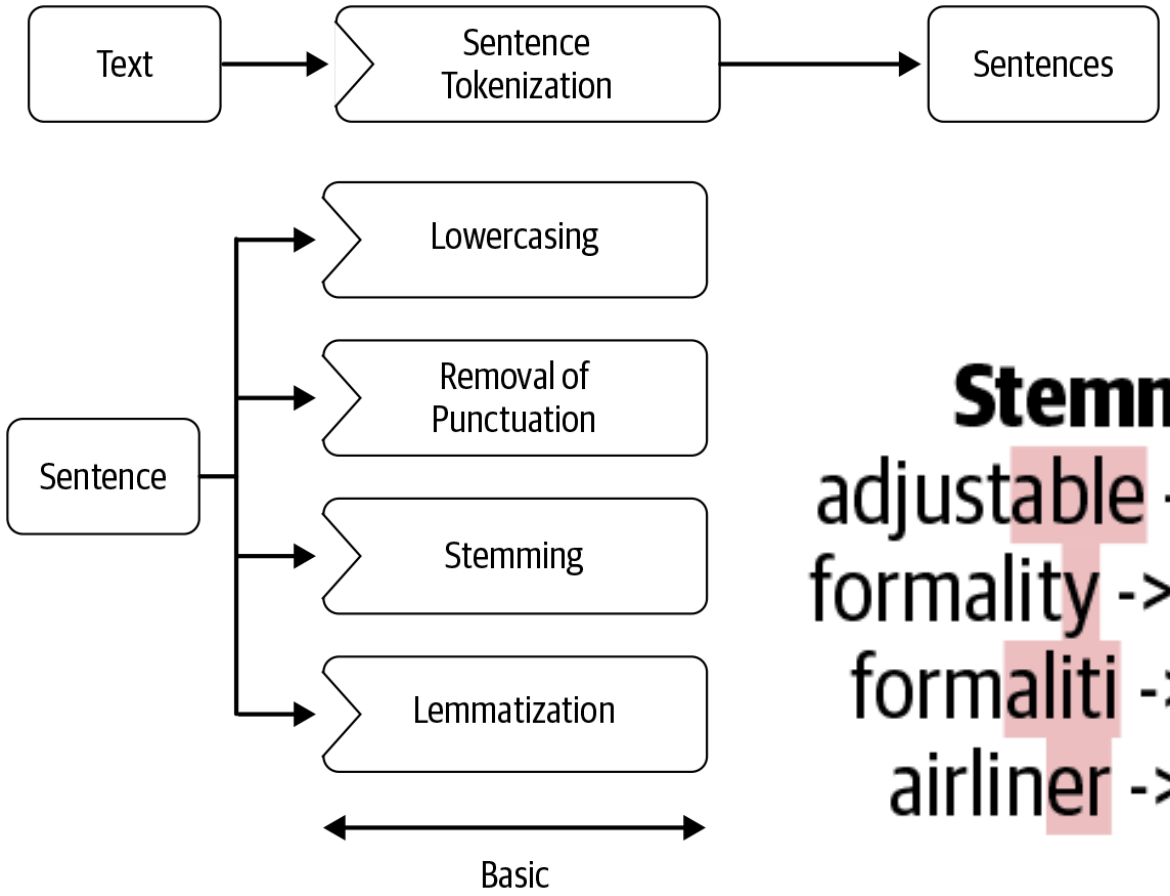
- **Inverse Document Frequency (IDF):** Quanto importante è il termine, la parola all'interno del contesto ?

IDF(parola nel contesto)=log(numero di documenti nel contesto / numero di documenti che includono la parola)

Una parola è importante se appare molte volte nel documento stesso. Ma parole come "a" e "il" , "e" appaiono spesso incrementando quindi il valore di TF. Questo aspetto viene risolto usando appunto IDF che è elevato se la parola è rara alto se la parola è comune all'interno del corpo del testo. Il valore TF-IDF di una parola è il prodotto di TF e IDF.



PREPROCESSING



Stemming

adjustable -> adjust
formality -> formaliti
formaliti -> formal
airliner -> airlin

Lemmatization

was -> (to) be
better -> good
meeting -> meeting

- **<u>L'NLP si occupa principalmente della comprensione di *testi e linguaggio parlato*, intesi come sequenze di parole che in una lingua esprimono uno o più messaggi (es. pagine web, post, tweet, log, informazioni aziendali);</u>**
- L'elaborazione del *parlato* (riconoscimento vocale) è considerato un ambito a sé.

- **Traduzione automatica**
  - Le performances variano drammaticamente a seconda della similarità tra i linguaggi utilizzati
- **Sentiment analysis**
  - Dato un testo, determinare se questo esprime un sentimento negativo o positivo
- **Summarization**
  - Riassumere un testo in maniera da preservare tutte le informazioni salienti
- **ChatBots**
  - Generazione di risposte coerenti ed utili, mantenendo al contempo traccia degli stati precedenti della conversazione

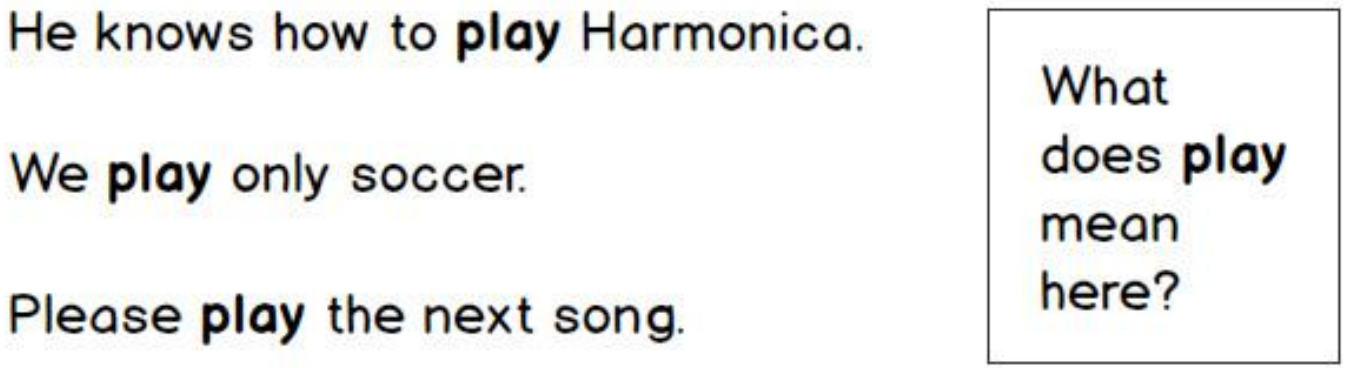
- **Question answering**
  - Può utilizzare modelli generativi o essere basato su una knowledge base
- **Named Entity Recognition**
  - Dato un testo, riconoscere quali parole appartengono alle categorie di interesse e per queste restituire una classificazione
  - Può essere visto come un rilevamento applicato al testo
- **Generazione testo**

Oggi l'**NLP** ci pone di fronte all'analisi di frasi complesse, che per essere interpretate correttamente devono essere scomposte in unità elementari: le parole. E oltre all'**analisi della singola parola,** è necessaria la **comprensione della semantica dell'intera frase**. Da un punto di vista tecnico, per passare dalla dimensione di analisi della singola parola alla comprensione della frase nel suo complesso (***Natural Language Understanding***), principalmente i task da prendere in considerazione sono:

- ***Word Sensing Disambiguation: associare alle parole, nel contesto, i corretti significati (es. nei motori di ricerca online);***

- ***Semantic Parsing: trasformare il testo in una rappresentazione semantica strutturata (nella pratica, rispondere a domande data una specifica frase e una collezione di documenti).***

Word Sensing Disambiguation: associare alle parole, nel contesto, i corretti significati (es. nei motori di ricerca online);



Gli algoritmi più utilizzati

•algoritmo Naive Bayes;

•algoritmo di Lesk;

Word Sensing Disambiguation: associare alle parole, nel contesto, i corretti significati (es. nei motori di ricerca online);

Algoritmo Naive Bayes

Questo algoritmo appartiene alla classe degli algoritmi di apprendimento supervisionato e si basa su tre fattori:

$$w_i : \text{parola}$$
$$s_i : \text{significato}$$
$$f_i : \text{feature}$$

Per semplicità indichiamo come *feature* $\boldsymbol{f_i}$ di una *parola* $\boldsymbol{w_i}$ tutte le altre *parole* $\boldsymbol{w_j}$ che appaiono nella stessa frase; con $\boldsymbol{i \neq j}$.

$$count(x) : \text{numero di occorrenze di } x$$
$$count(s_i, w_j) : \text{numero di volte in cui } s_i \text{ viene attribuito a } w_j$$
$$count(f_j, s_i) : \text{numero di volte in cui } f_j \text{ compare in una frase in cui c'è una parola a cui è attribuito } s_i$$

Avremo quindi:

$$P(s_i) = \frac{count(s_i, w)}{count(w)}$$
$$P(f_j|s_i) = \frac{count(f_j, s_i)}{count(s_i)}$$
$$\hat{s} = \arg\max_{i \in S} P(s_i) \prod_j P(f_j, s_i)$$

Il risultato dell'ultima espressione rappresenta il significato più adatto per la *parola* $\boldsymbol{w}$.

Word Sensing Disambiguation: associare alle parole, nel contesto, i corretti significati (es. nei motori di ricerca online);

Algoritmo di Lesk

L'algoritmo di Lesk utilizza un dizionario per disambiguare le parole: WordNET è il database che serve.

Il concetto alla base dell'algoritmo di Lesk è l'intersezione tra la definizione data dal dizionario e il contesto in cui appare la parola da disambiguare. Più precisamente, il significato attribuito dall'algoritmo è quello nella cui definizione (che può essere estesa con alcune frasi di esempio) compaiono il numero maggiore di parole (feature) del contesto.

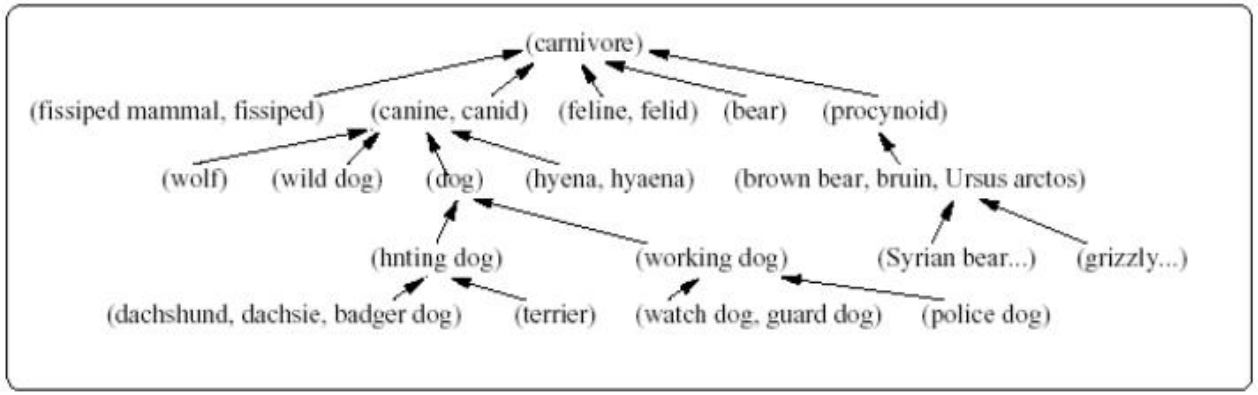
$\vec{d_s}$: parole che compongono la definizione di s

$\vec{e_s}$: parole che compongono gli esempi per s

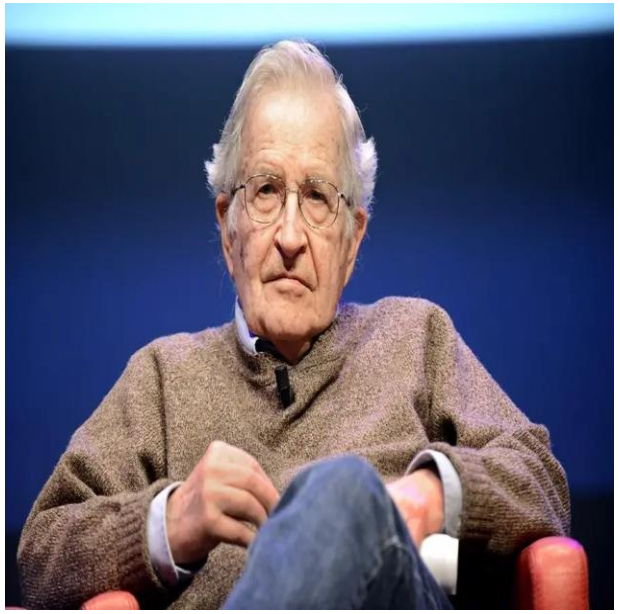
$\vec{f}$: parole che appaiono nel contesto

$$\hat{s} = \arg\max_{s \in S} \left| \left(\vec{d_s} \cup \vec{e_s} \right) \cap \vec{f} \right|$$

WordNet divide il lessico in cinque categorie: sostantivi, verbi, aggettivi, avverbi e parole funzione .
WordNet organizza l'informazione lessicale in termini dei significati delle parole (gerarchie semantiche).

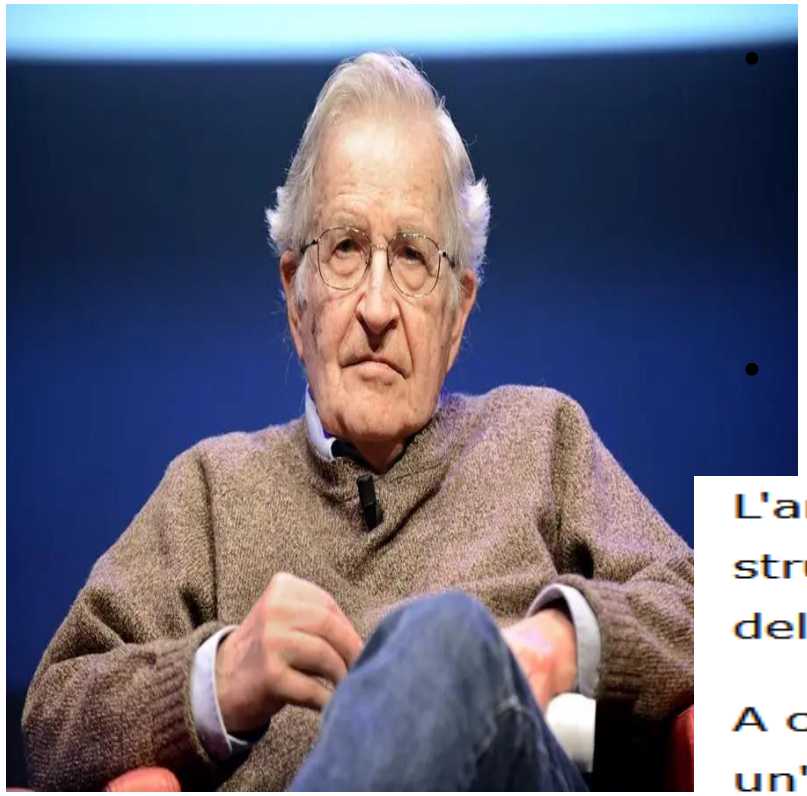


Semantic Parsing: trasformare il testo in una rappresentazione semantica strutturata (nella pratica, poter rispondere a domande data una specifica frase e una collezione di documenti)



L'obiettivo del programma che esegue il parsing, `e di individuare la
struttura sintattica relativa ad una frase, grammaticalmente corretta, vista come una sequenza di parole.

Chomsky ha proposto il concetto di **phrase-structure grammar** (o grammatiche a struttura sintagmatica generativa)

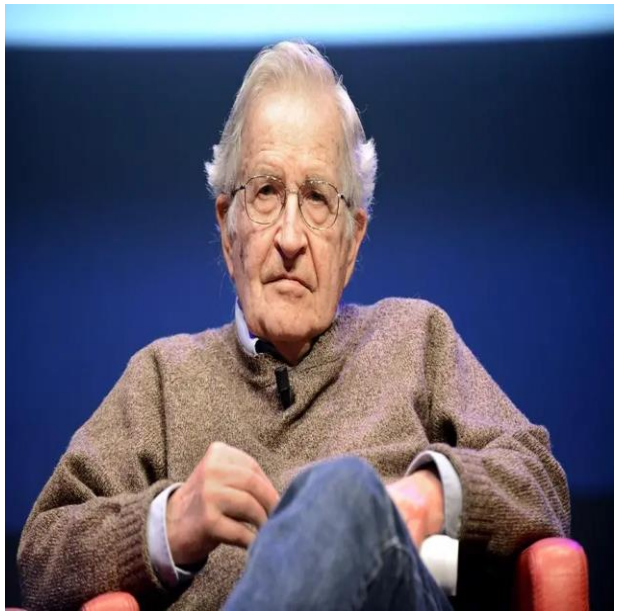


- Una tipologia di grammatica generativa basata su regole che permettono di associare ad ogni frase corretta del linguaggio una struttura ad albero, chiamata **albero sintattico (o parse tree),** che rappresenta i collegamenti sintattici esistenti fra le varie parole della frase.

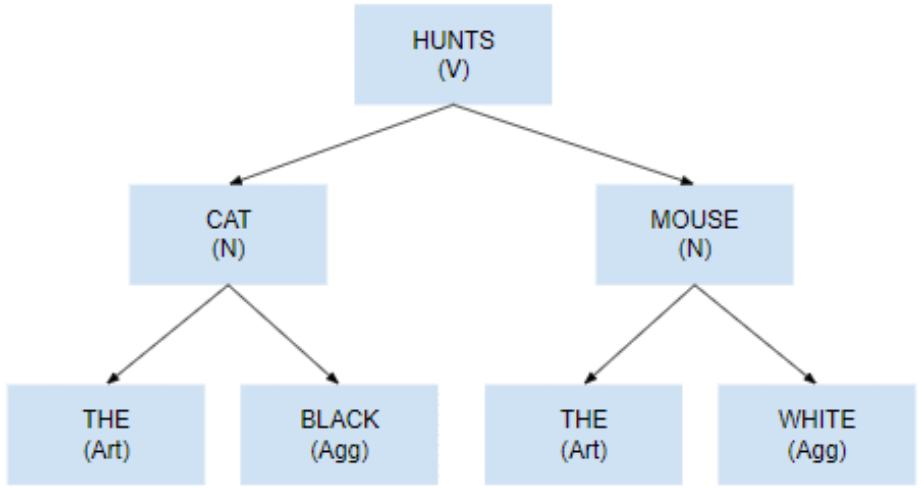
- I collegamenti sono definiti tramite delle **etichette (o label)** che rappresentano le diverse categorie sintattiche.

L'analisi sintattica assegna alla frase in input una struttura che prende in considerazione le singole unità del discorso e le relazioni tra loro.

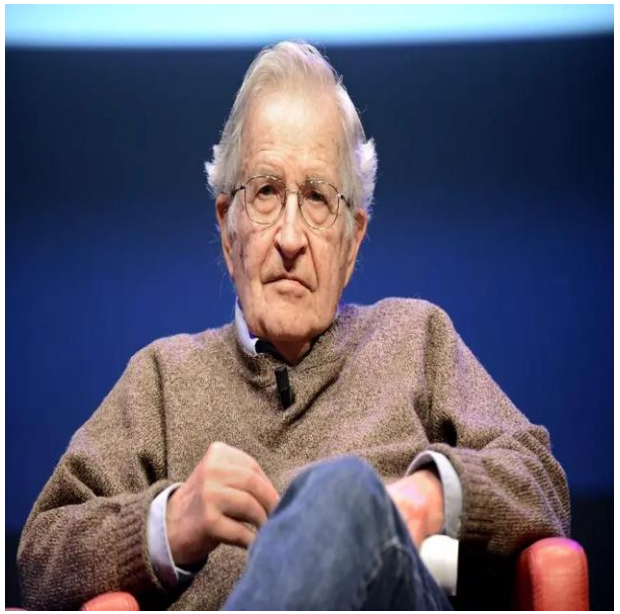
A ciascuna unità linguistica (sintagma) viene assegnata un'etichetta grammaticale.



La struttura sintagmatica può essere rappresentata sotto forma di albero.



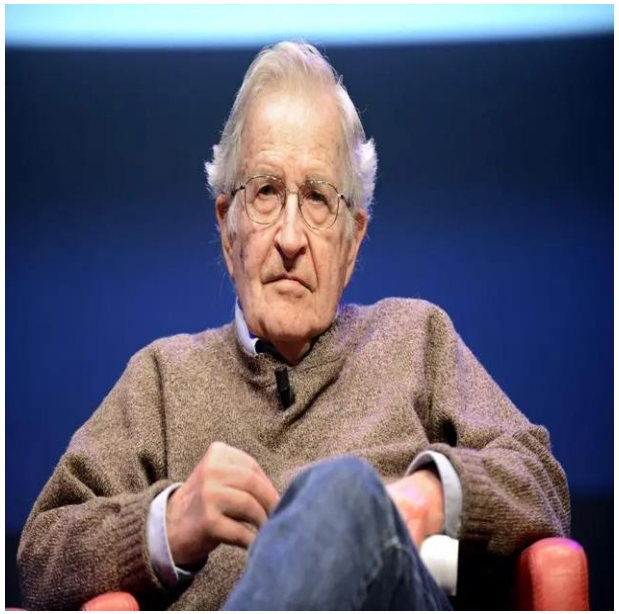
A cosa serve l'analisi sintattica? E' una delle prime fasi nel text mining ed e utile nell'elaborazione del linguaggio naturale (NLP) da parte di un algoritmo.



1] WORD

Nell'analisi sintattica ogni parola (word) è associata a una categoria grammaticale **POS (parts of speech)**.

- **Noun (N)**. E' un nome. In genere indica un oggetto (es. libro) o un'entità vivente (es. cane) o non vivente (es. sicurezza). E' associato al tag N e ci sono diverse sottocategorie.
- **Verb (V)**. E' un verbo. I verbi indicano un'azione (es. camminare) oppure uno stato (es. essere). Possono essere ausiliari, riflessivi, transitivi o intransitivi, ecc. E' associato al tag V
- **Adjective (ADJ)**. E' un aggettivo. Sono parole che descrivono o qualificano altre parole. Ac esempio, una "bella" casa, un "buon" libro, ecc. Nell'analisi POS sono associate al simbolo ADJ.
- **Adverb (ADV)**. E' un avverbio. Sono parole che modificano l'intensità o il senso di altre parole, come sostantivi, aggettivi o verbi. Nell'analisi POS sono associate al simbolo ADV.

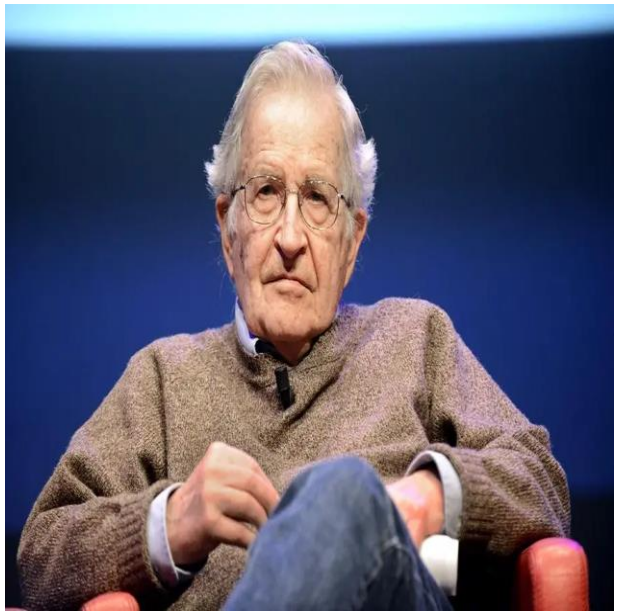


2] PHRASE

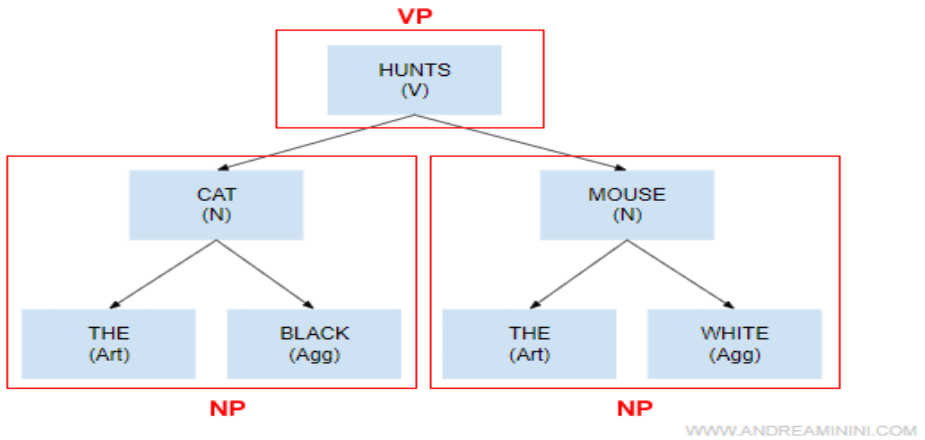
I gruppi di parole possono formare una **phrase**.

Una delle parole delle altre è più importante delle altre (*keyword*).

Cos'è una phrase? Nell'analisi sintattica inglese la phrase è un gruppo di parole senza un soggetto o senza un verbo. Nel discorso è un'unità grammaticale di base. In un albero gerarchico sono i nodi superiori (genitori) alle parole. Ad esempio "*the black cat*" è una phrase.

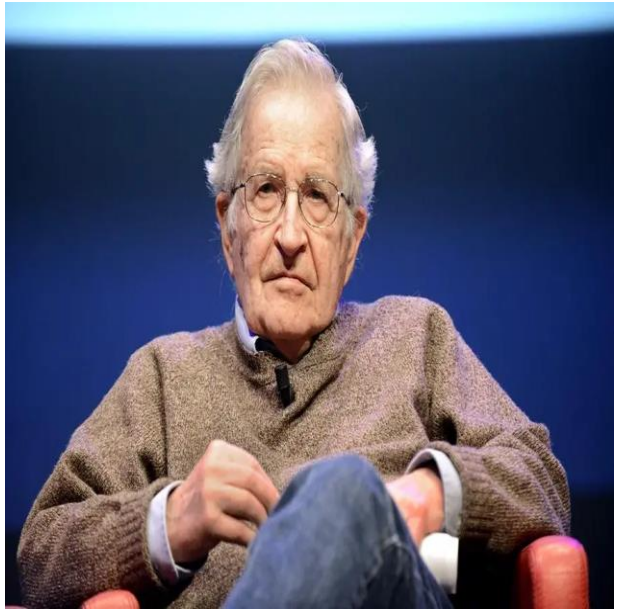


Un esempio pratico

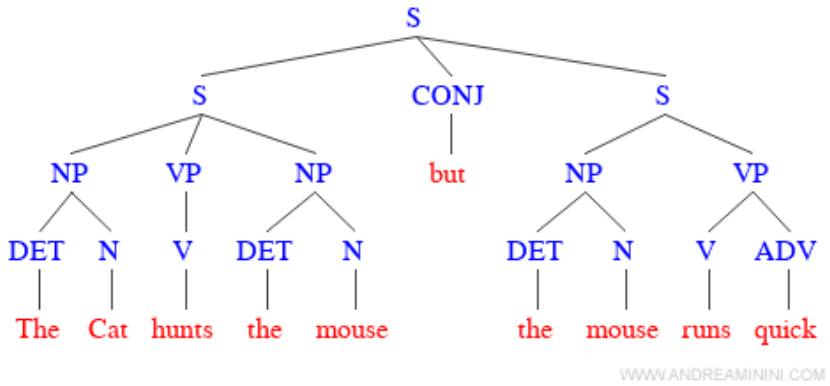


Le principali categorie sintattiche di un gruppo phrase sono le seguenti:

- **Noun Phrase (NP)**. E' un gruppo di parole in cui il sostantivo è la parola chiave. In genere sono il soggetto o l'oggetto di un verbo. Ad esempio "the black cat".
- **Verb Phrase (VP)**. E' un gruppo di parole in cui il verbi è la parola chiave. In genere sono il soggetto o l'oggetto di un verbo. Ad esempio, la parola "hunts" è anche un gruppo lessicale VP.

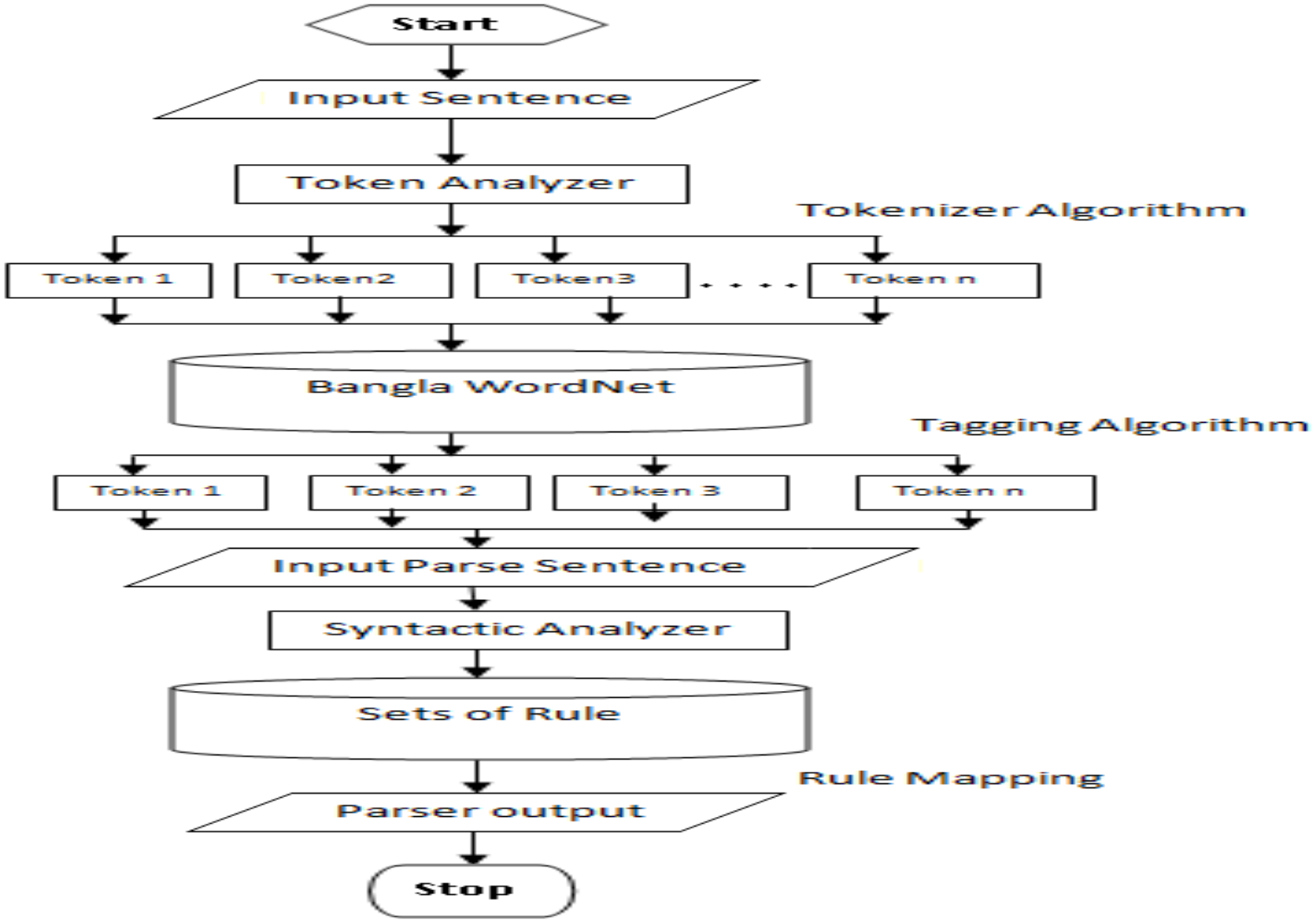


- **Adjective phrase (ADJP)**. E' un gruppo di parole in cui l'aggettivo è la parola chiave. In genere descrive o qualifica un nome o pronome. Ad esempio, nella frase "The train is so fast" il gruppo "so fast" (così veloce) è classificato ADJP.
- **Adverb phrase (ADVP)**. E' un gruppo di parole in cui l'avverbio è la parola chiave. In genere descrive l'azione. Ad esempio, nella frase "You arrived too late" il gruppo "too late" (troppo tardi) è classificato ADVP.
- **Conjunction (CONJ)**. Sono le congiunzioni ossia le parti del discorso che uniscono due o più parole o frasi (S) tra loro.



Riassumendo

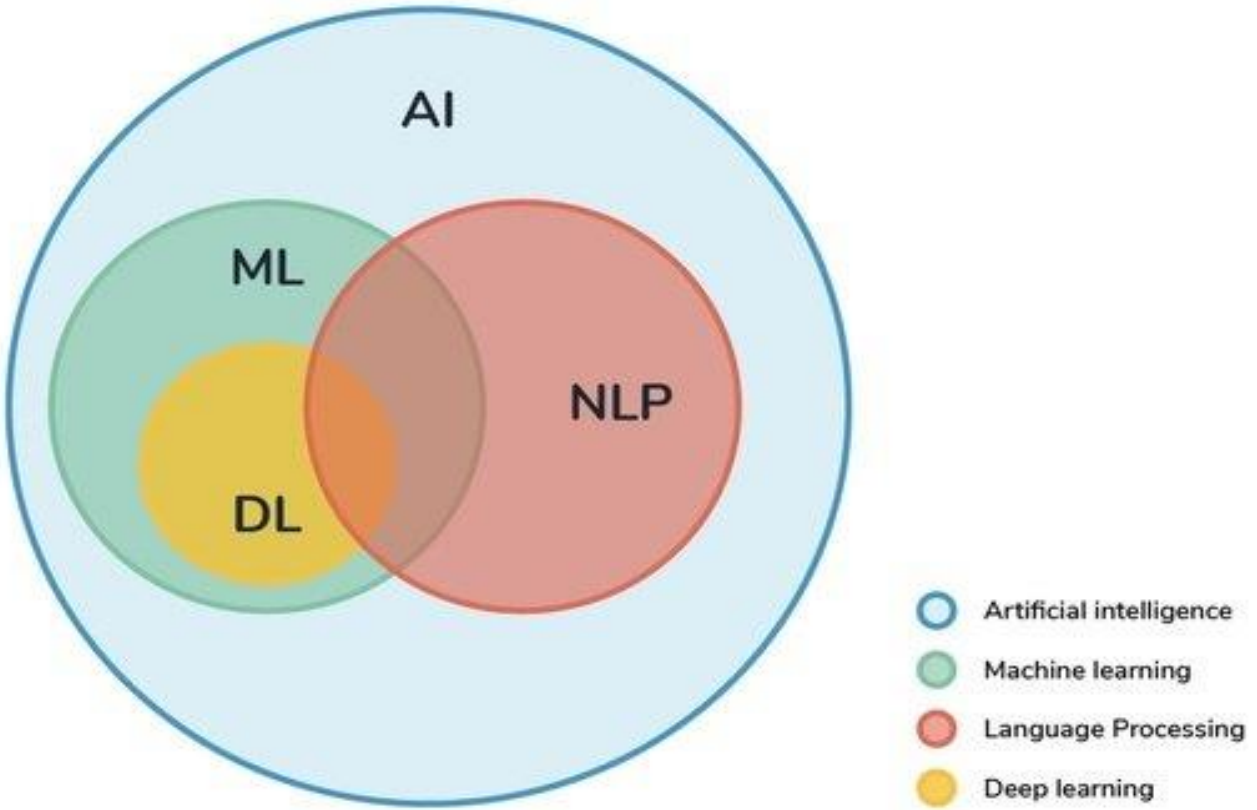
I parser in output producono classi che aiutano a strutturare le risposte del modello di linguaggio.
Ci sono due metodi principali che un parser implementa:
«Get format instructions": un metodo che restituisce una stringa contenente le istruzioni per come l'output di un modello di lingua dovrebbe essere formattato.
"Parse": un metodo che prende in una stringa (presunta) per essere la risposta da un modello di lingua) e analizza in una certa struttura.



UTILIZZI NLP

Utilizzi NLP:

- ***Text Analysis: analisi di un testo e, laddove richiesto, individuazione di elementi chiave (es. argomenti, persone, date);***
- ***Text Classification: interpretazione di un testo per classificarlo in una categoria predefinita (es. spam);***
- ***Sentiment Analysis: rilevamento dell'umore all'interno di un testo (es. recensione positiva/negativa);***
- ***Intent Monitoring: comprensione del testo per prevedere comportamenti futuri (es. la volontà di acquisto da parte di un cliente);***
- ***Smart Search: ricerca, all'interno di archivi, dei documenti che meglio corrispondono ad un'interrogazione posta in linguaggio naturale;***
- ***Text Generation: generazione automatic di un testo;***
- ***Automatic Summarization: produzione di una versione sintetica di uno o più documenti testuali;***
- ***Language Translation: traduzione di testi scegliendo, volta per volta, il significato migliore a seconda del contesto.***


AI
ML
NLP
DL
Artificial intelligence
Machine learning
Language Processing
Deep learning

INTRODUZIONE AL LINGUAGGIO

- Fino all'introduzione del deep learning e della moderna AI, lo studio del linguaggio e del Natural Language Processing (NLP) era marcatamente differente da altri campi multimediali.
- Il linguaggio si compone di signicanti (le parole, oggettive) e significato (idea associata, in maniera non univoca, al significante). Questo tipo di ambiguità (polisemia) sono difficili da gestire per le macchine
- Di converso, alcune parole diverse possono avere significati simili
- Esistono espressioni che cambiano completamente il significato delle parole
- Diverse lingue hanno strutture molto diverse tra loro (e.g. Cina e Giappone hanno linguaggi basati su simboli invece che una scrittura che indichi la pronuncia)
- Si presenta a maggior ragione la dualità sintassi-semantica

COME RAPPRESENTARE LE PAROLE?

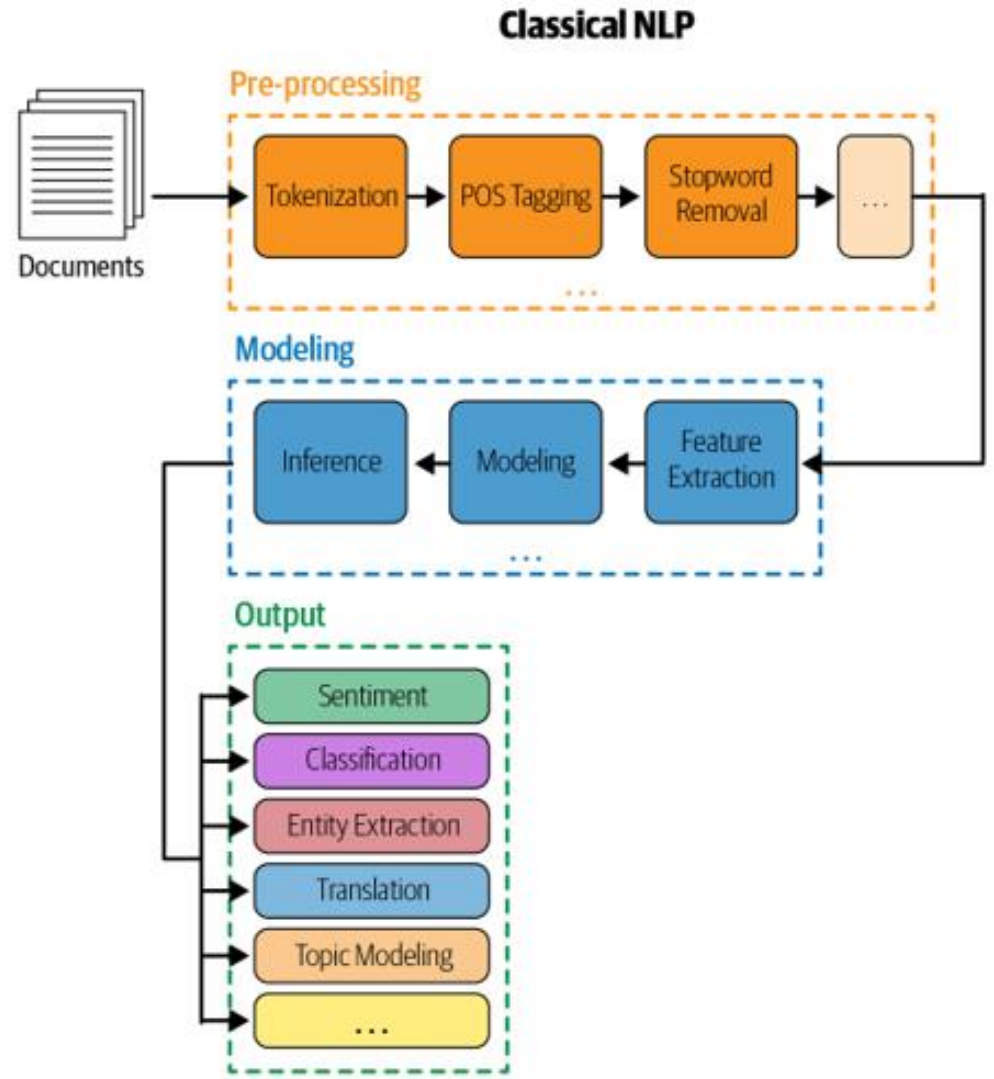
- Hot encoding: assegnare un numero ad ogni parola
  - **Problema**: quando si considerano tutte le possibili varianti di una parola, la dimensione dei vettori diventa ingestibile (e.g. in Italiano si possono stimare milioni di variazioni a partire da qualche migliaio di parole base)
  - **Gestisce inefficacemente la polisemia**: la parola cane e canide sono distanti quanto cane e pianoforte;

- **Idea**: esistono spazi a dimensione più bassa che possano codificare efficacemente
- **Idea ulteriore**: le parole hanno significato in funzione delle altre parole che le circondano

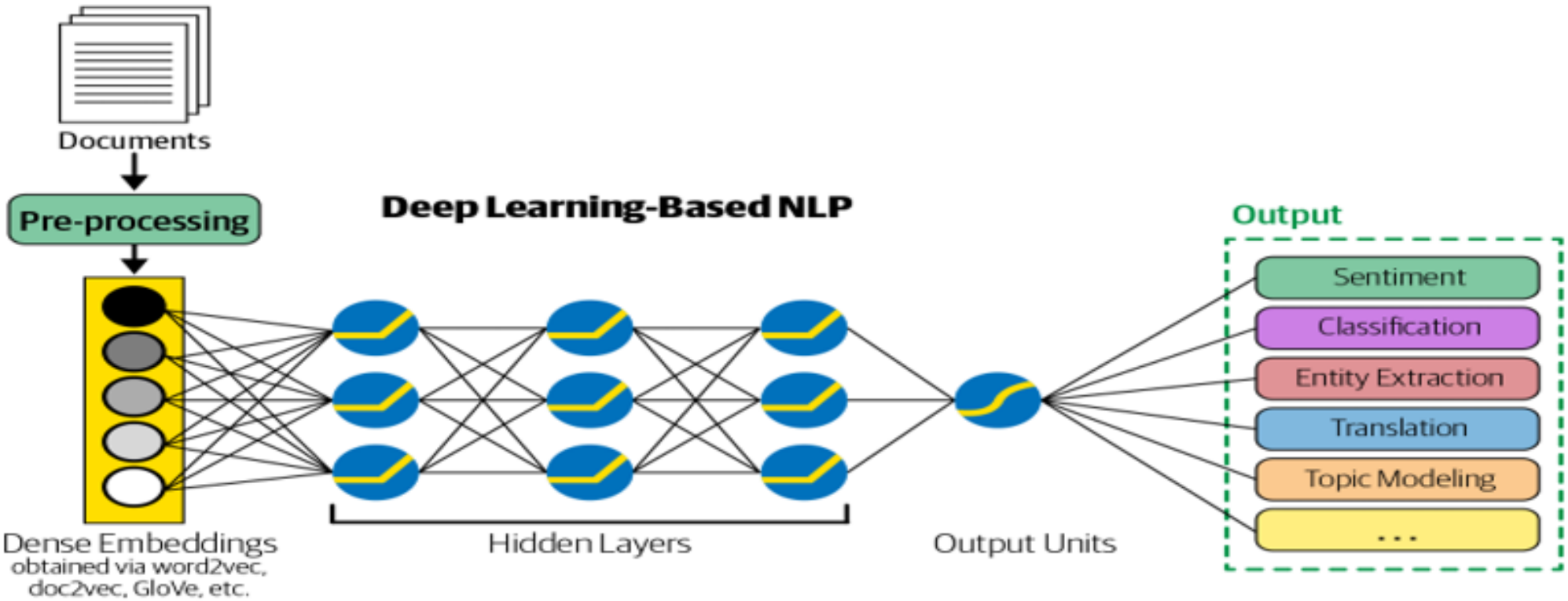
PREPROCESSING

- **Alternativa al Hot Encoding: Tokenizzazione.** Si suddivide il testo in piccoli frammenti (token) secondo un certo set di regole;
- **Tokenizer elementare**: si taglia la frase ogni volta che si incontra uno spazio o un segno di punteggiatura (chiamato anche word tokenizer). Problema: il numero di token complessivo è molto alto (10^6)
- **Character tokenizer**: ogni carattere rappresenta un token. Il numero di token rimane basso, ma il significato portato da ogni token è molto scarso
- **Subword tokenizer**: il migliore ad oggi. Divide il testo in sotto-componenti che occorrono frequentemente. Si può costruire iterativamente a partire da una divisione in caratteri, sulla base di un corpus documentale;
- **Stop word removal: rimuovere le parole che non portano informazione come per esempio “a”, “al”,**

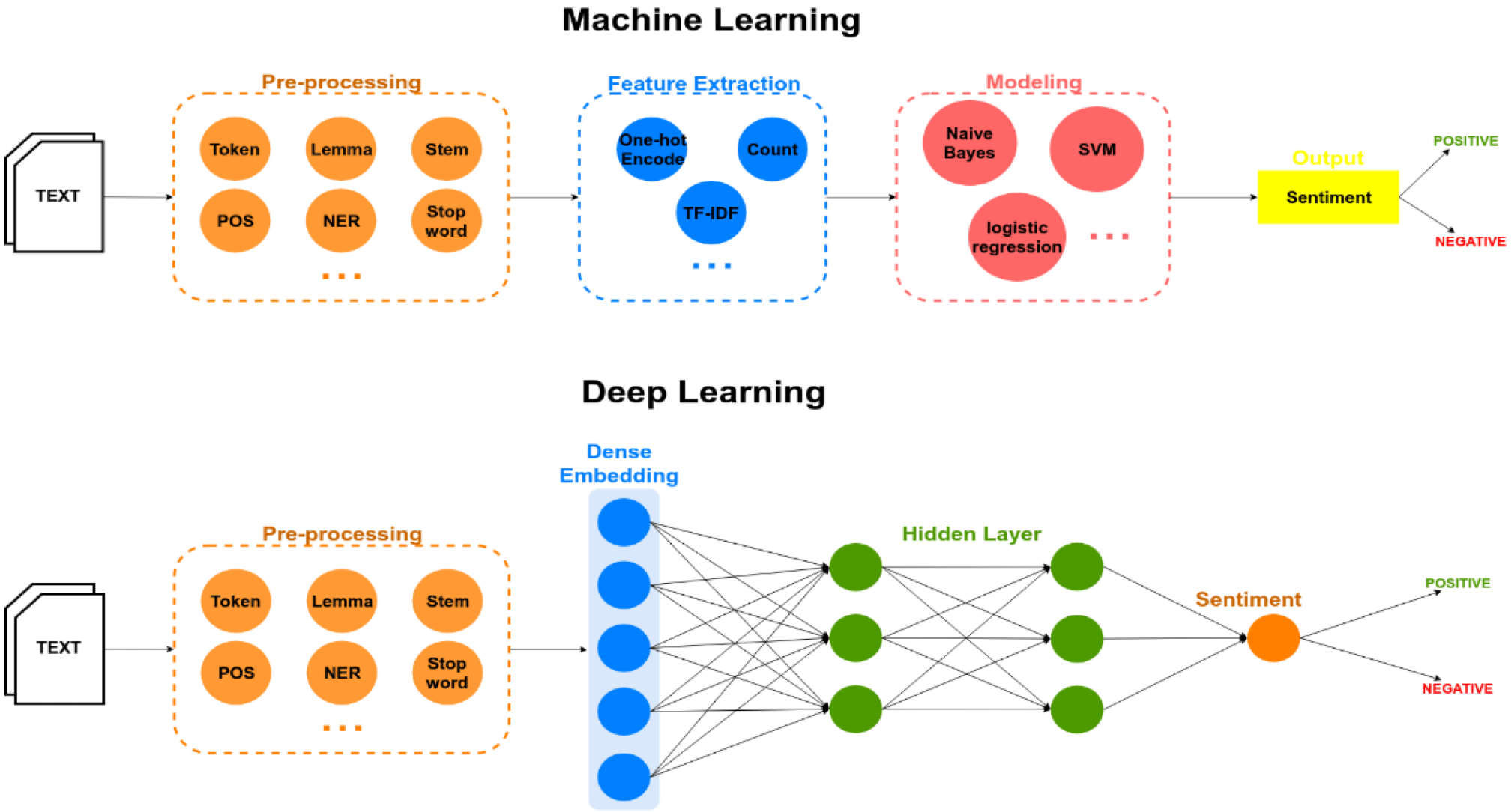
CLASSICAL NLP vs DEEP LEARNING NLP



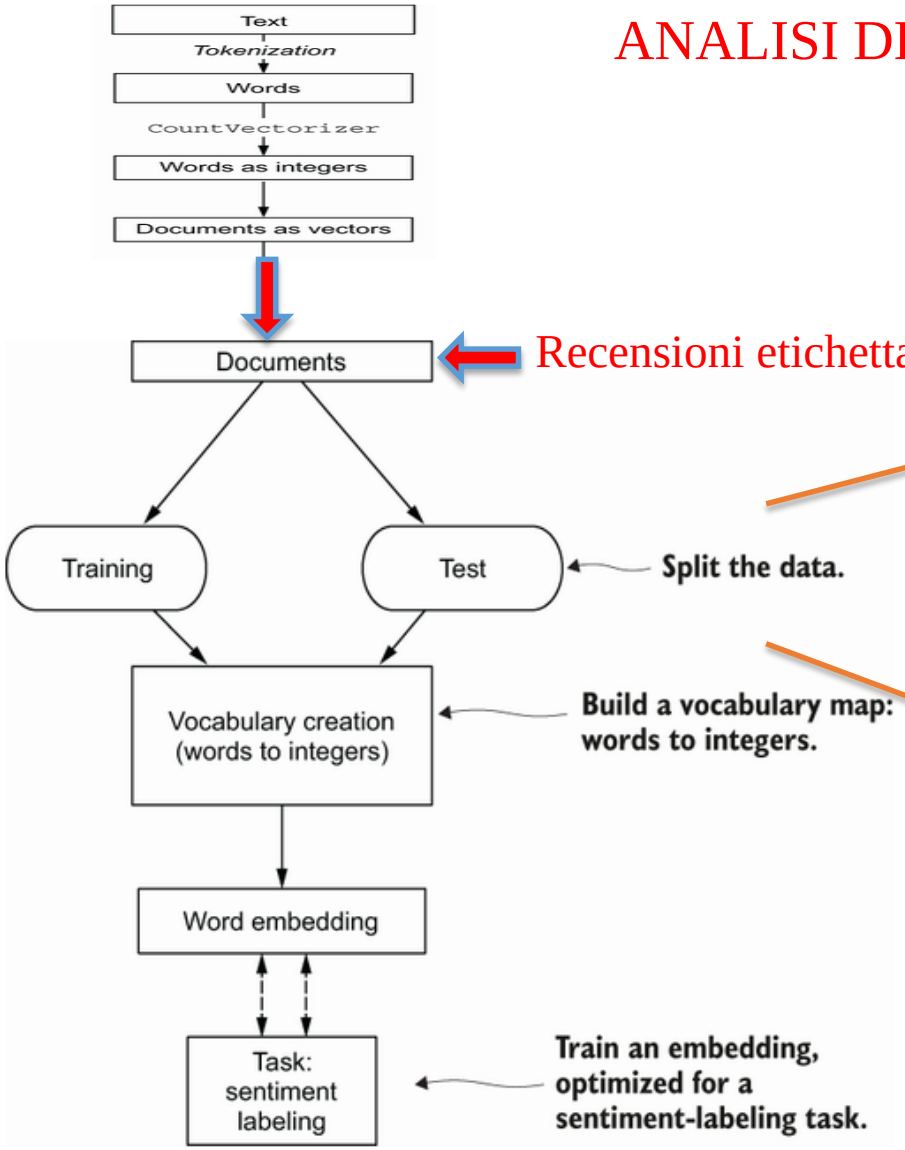
CLASSICAL NLP vs DEEP LEARNING NLP

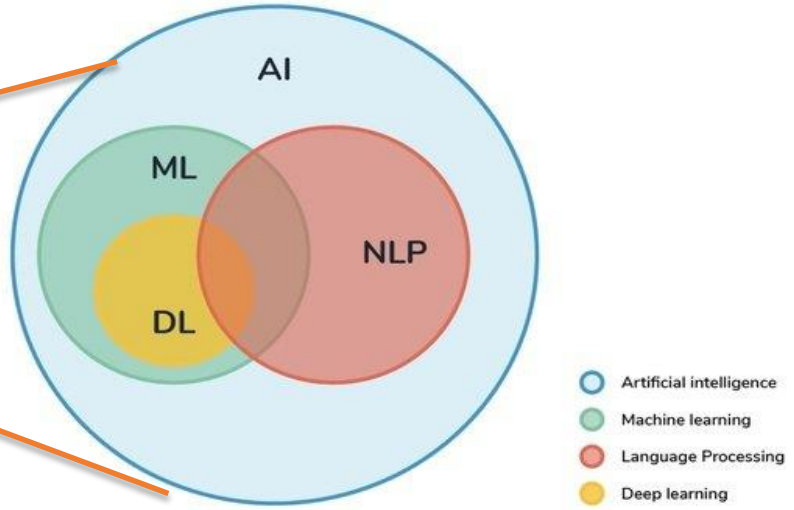


CLASSICAL NLP vs DEEP LEARNING NLP

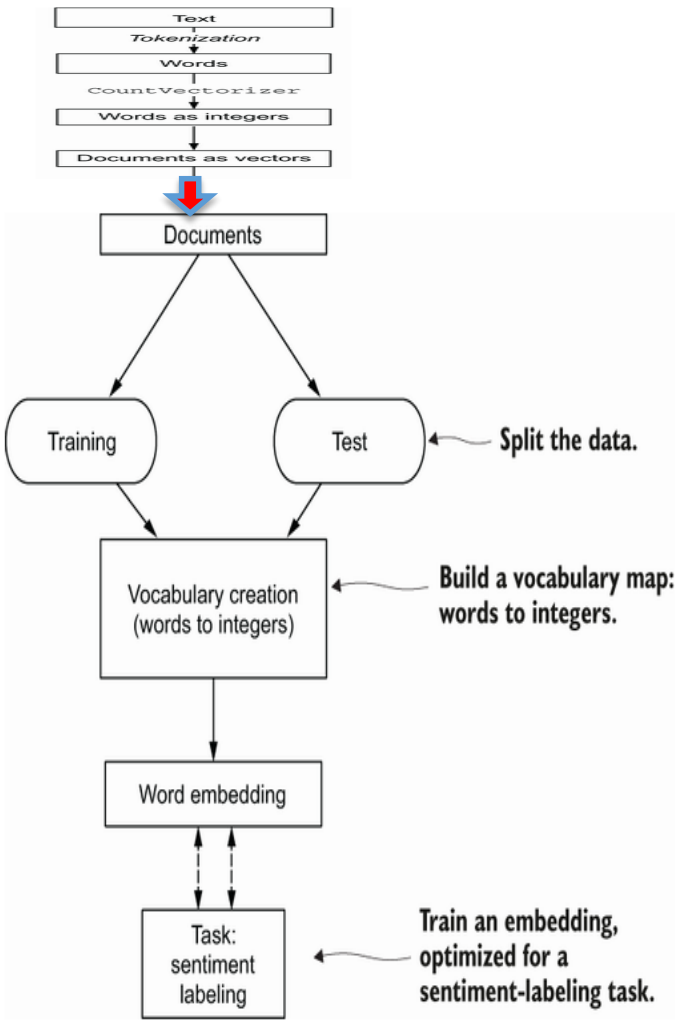


ANALISI DEL «SENTIMENT»



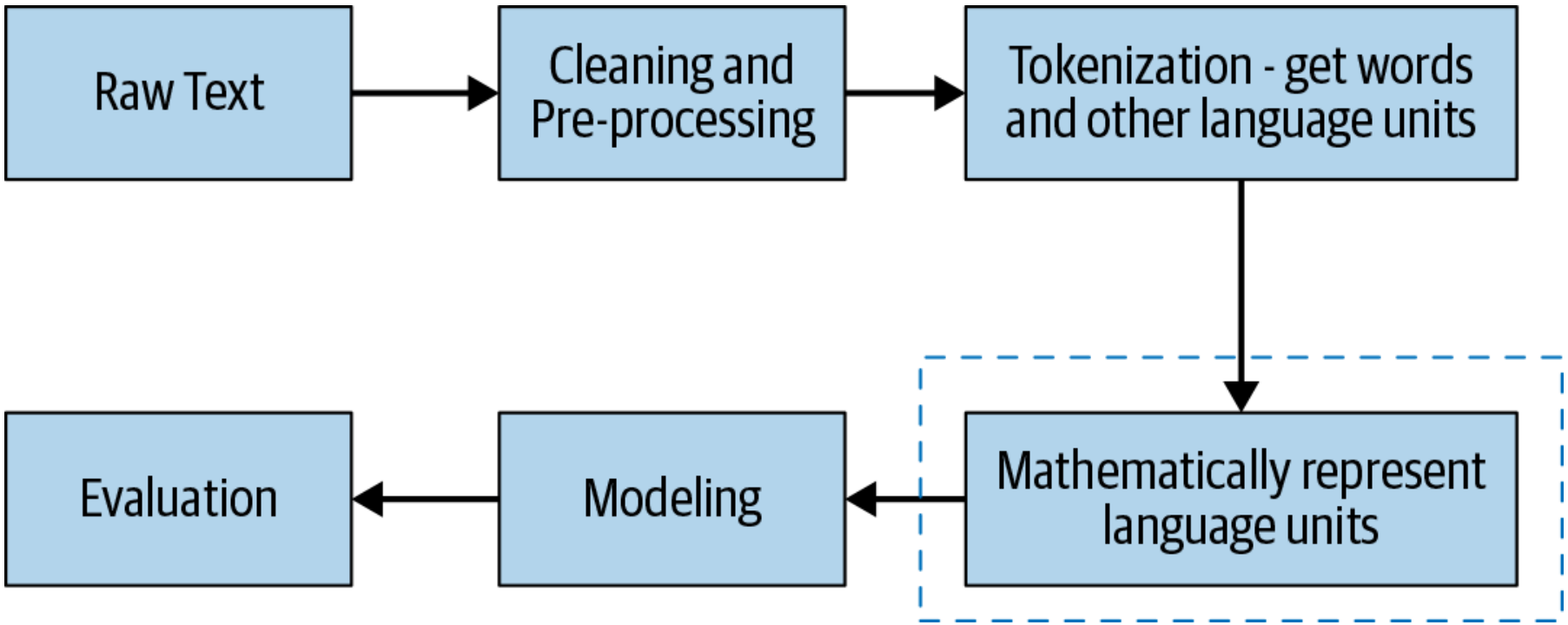


ANALISI DEL «SENTIMENT»

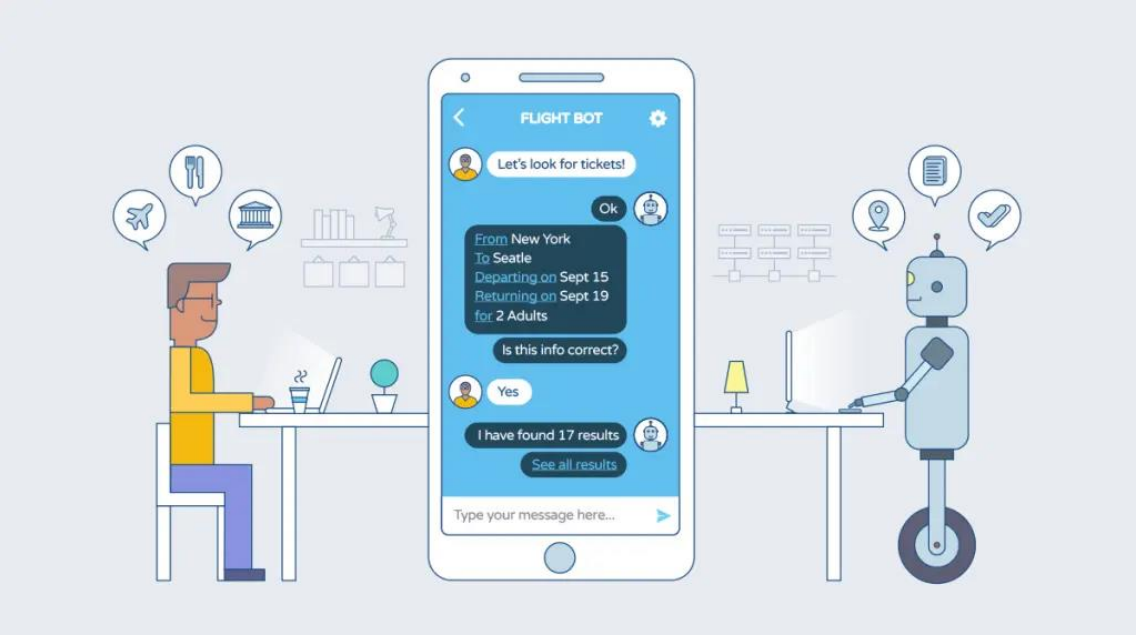


Review	Sentiment score
Wow... Loved this place.	1
Crust is not good.	0
Not tasty and the texture was just nasty.	0
Stopped by during the late May bank holiday off Rick Steve recommenda-tion and loved it.	1
The selection on the menu was great and so were the prices.	1
Now I am getting angry and I want my damn pho.	0
Honestly it didn't taste THAT fresh.	0
The potatoes were like rubber and you could tell they had been made up ahead of time being kept under a warmer.	0
The fries were great too.	1
A great touch.	1
Service was very prompt.	1
Would not go back.	0

Review	Sentiment score
Wow... Loved this place.	1
Crust is not good.	0
Not tasty and the texture was just nasty.	0
Stopped by during the late May bank holiday off Rick Steve recommendation and loved it.	1
The selection on the menu was great and so were the prices.	1
Now I am getting angry and I want my damn pho.	0
Honestly it didn't taste THAT fresh.	0
The potatoes were like rubber and you could tell they had been made up ahead of time being kept under a warmer.	0
The fries were great too.	1
A great touch.	1
Service was very prompt.	1
Would not go back.	0

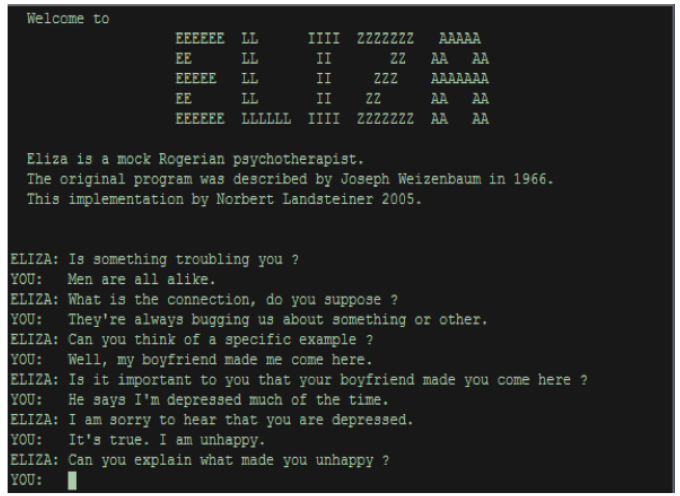

Raw Text
Cleaning and Pre-processing
Tokenization - get words and other language units
Mathematically represent language units
Modeling
Evaluation

- Un chatbot è un software che simula ed elabora le conversazioni umane (scritte o parlate), consentendo agli utenti di interagire con i dispositivi digitali come se stessero comunicando con una persona reale.
- I chatbot possono essere semplici come programmi rudimentali che rispondono a una semplice query con una singola riga oppure sofisticati come gli assistenti digitali.

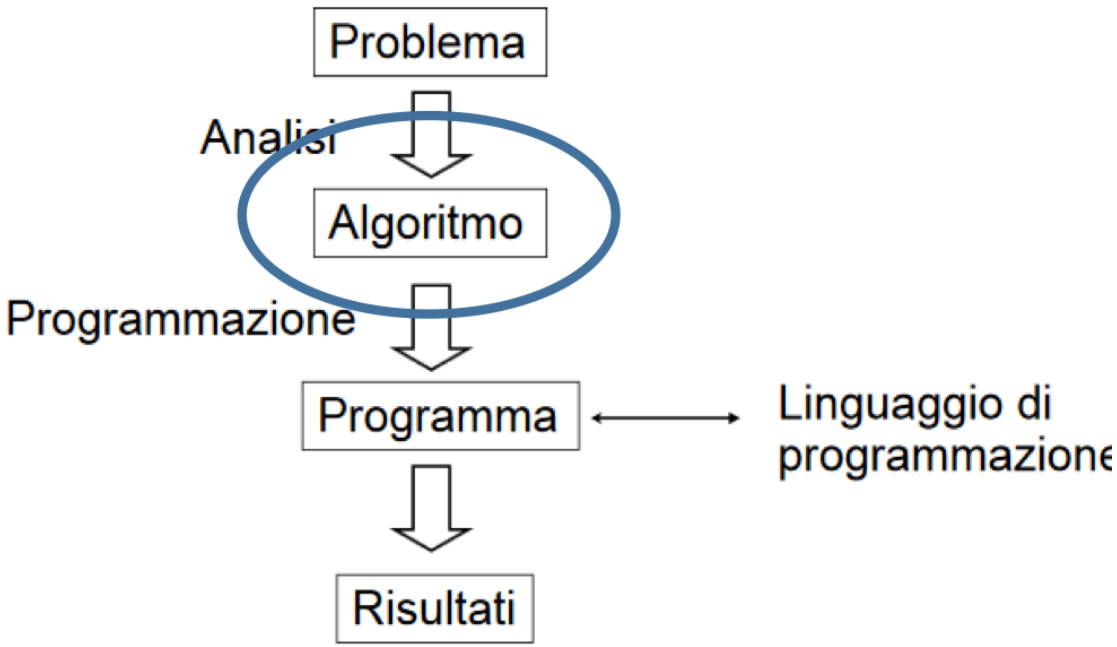


CODIFICARE UNA SEMPLICE *CONVERSATIONAL* BOT

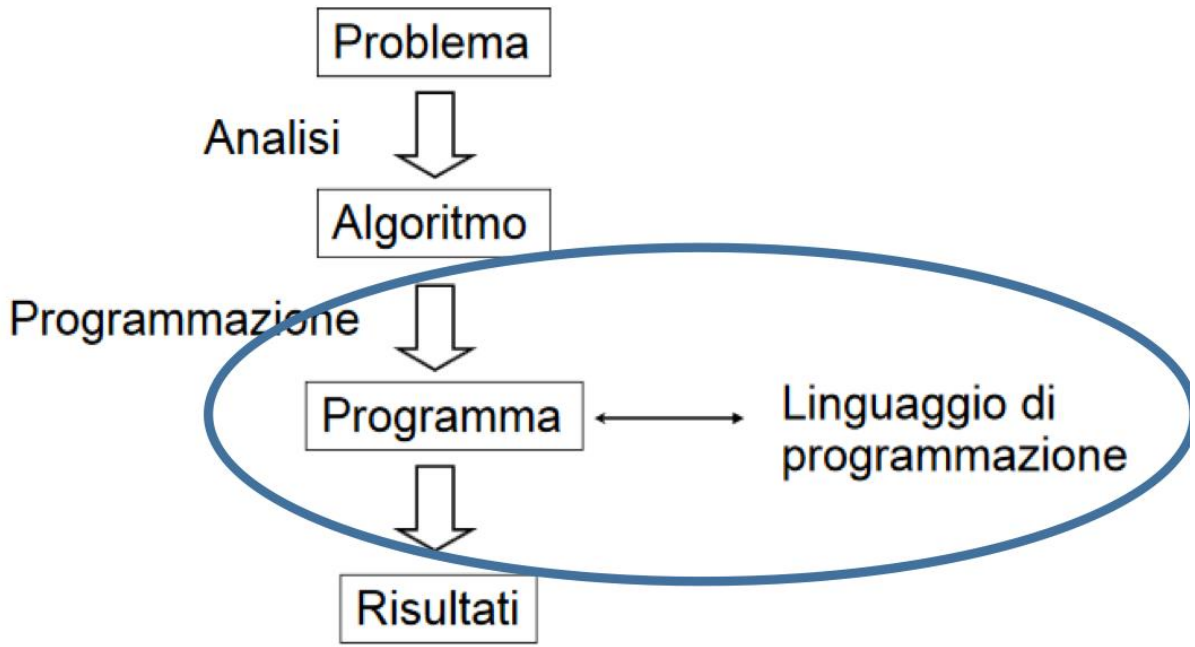
- ELIZA: SOLLECITA L'INPUT DELL'UTENTE E SEMBRA CAPIRE E RISPONDERE IN MODO INTELLIGENTE
- IL NOSTRO BOT NON AVRÀ DIVERSE REGOLE CHE GLI DANNO L'IMPRESSIONE DI AVERE UNA CONVERSAZIONE INTELLIGENTE
- IL NOSTRO BOT AVRÀ UNA SOLA CAPACITÀ, MANTENERE VIVA LA CONVERSAZIONE CON RISPOSTE CASUALI CHE POTREBBERO FUNZIONARE IN QUASI TUTTE LE CONVERSAZIONI BANALI



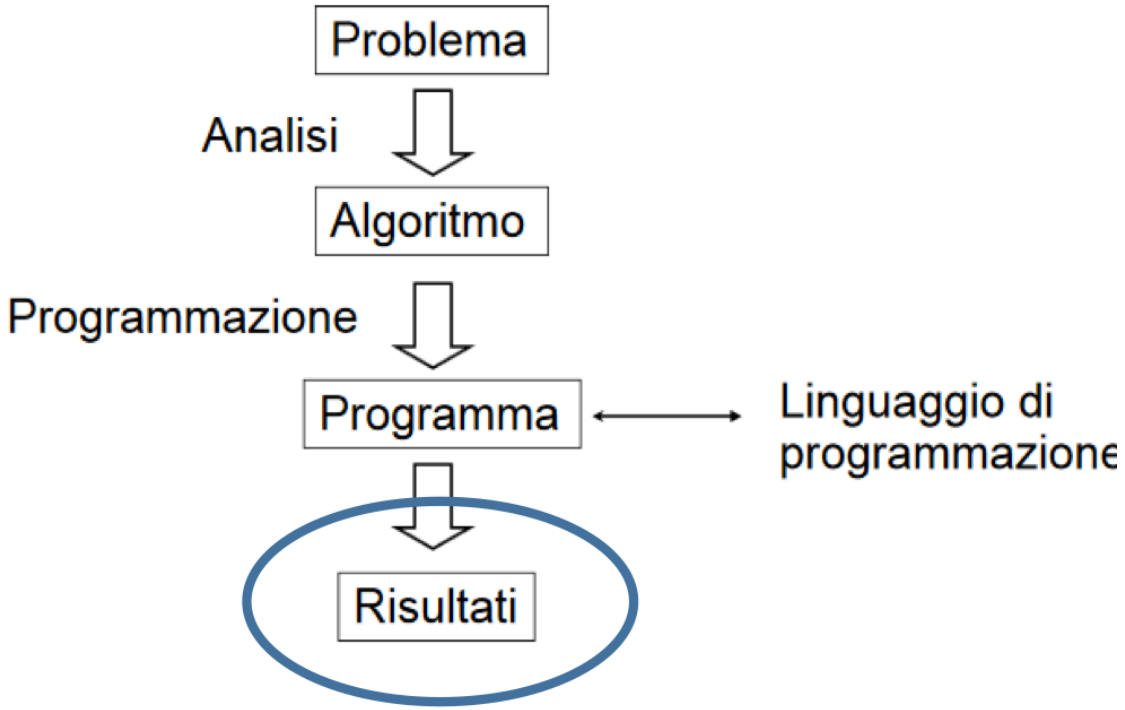
1. STAMPA LE ISTRUZIONI CHE CONSIGLIANO ALL'UTENTE COME INTERAGIRE CON IL BOT
2. INIZIA UN CICLO
    a. ACCETTA L'INPUT DELL'UTENTE
    b. SE L'UTENTE HA CHIESTO DI USCIRE, ALLORA ESCI
    c. ELABORARE L'INPUT DELL'UTENTE E DETERMINARE LA RISPOSTA (IN QUESTO CASO, LA RISPOSTA È UNA SCELTA CASUALE DA UN ELENCO DI POSSIBILI RISPOSTE GENERICHE)
    d. STAMPA RISPOSTA
3. TORNA AL PASSAGGIO 2



```python
import random

# This list contains the random responses (you can add your own or translate them into your own language too)
random_responses = ["That is quite interesting, please tell me more.",
                    "I see. Do go on.",
                    "Why do you say that?",
                    "Funny weather we've been having, isn't it?",
                    "Let's change the subject.",
                    "Did you catch the game last night?"]

print("Hello, I am Marvin, the simple robot.")
print("You can end this conversation at any time by typing 'bye'")
print("After typing each answer, press 'enter'")
print("How are you today?")

while True:
    # wait for the user to enter some text
    user_input = input("> ")
    if user_input.lower() == "bye":
        # if they typed in 'bye' (or even BYE, ByE, byE etc.), break out of the loop
        break
    else:
        response = random.choices(random_responses)[0]
    print(response)

print("It was nice talking to you, goodbye!")
```

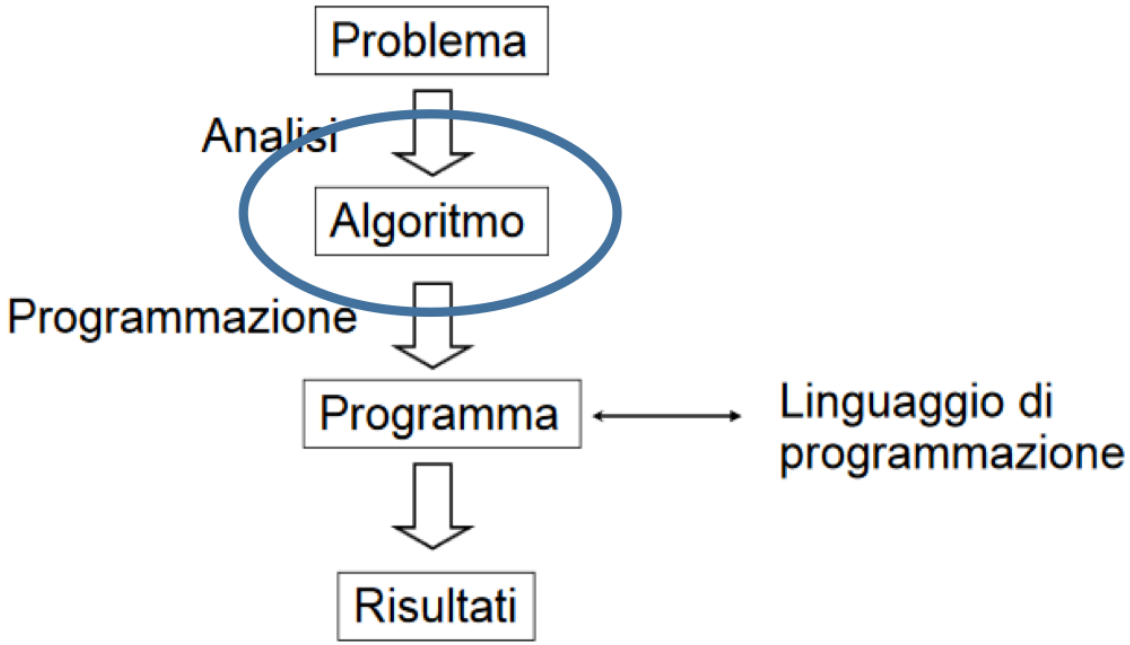


```
Hello, I am Marvin, the simple robot.
You can end this conversation at any time by typing 'bye'
After typing each answer, press 'enter'
How are you today?
> I am good thanks
That is quite interesting, please tell me more.
> today I went for a walk
Did you catch the game last night?
> I did, but my team lost
Funny weather we've been having, isn't it?
> yes but I hope next week is better
Let's change the subject.
> ok, lets talk about music
Why do you say that?
> because I like music!
Why do you say that?
> bye
It was nice talking to you, goodbye!
```

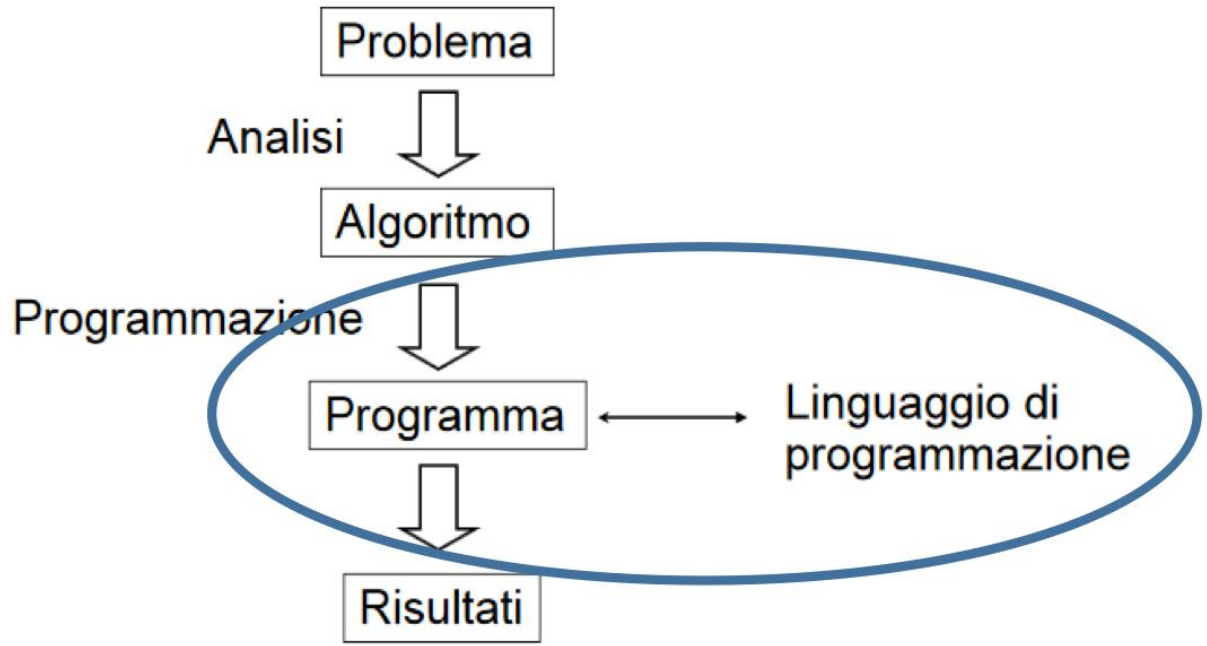


MIGLIORIAMO LA CONVERSATIONAL BOT CON NLP

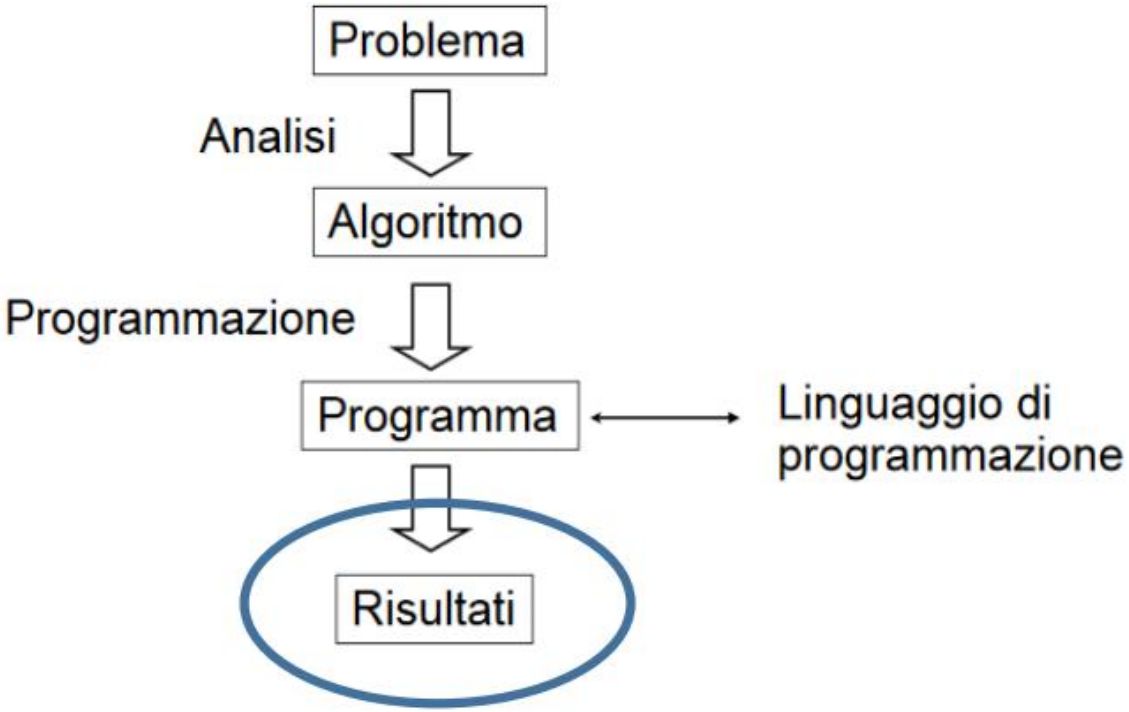
1. STAMPA LE ISTRUZIONI CHE CONSIGLIANO ALL'UTENTE COME INTERAGIRE CON IL BOT
2. INIZIA UN CICLO
    a. ACCETTA L'INPUT DELL'UTENTE
    b. SE L'UTENTE HA CHIESTO DI USCIRE, ALLORA ESCI
    c. ELABORARE L'INPUT DELL'UTENTE E **DETERMINARE LA RISPOSTA IN RELAZIONE AL GRADIMENTO (SENTIMENT)**
    d. SE VIENE RILEVATA UNA FRASE CHE ESPRIME UN SENTIMENT, CHIEDI ULTERIORI INPUT SU QUELL'ARGOMENTO
    e. STAMPA RISPOSTA
3. TORNA AL PASSAGGIO 2



```python
import random
from textblob import TextBlob
from textblob.np_extractors import ConllExtractor
extractor = ConllExtractor()

def main():
    print("Hello, I am Marvin, the friendly robot.")
    print("You can end this conversation at any time by typing 'bye'")
    print("After typing each answer, press 'enter'")
    print("How are you today?")

    while True:
        # wait for the user to enter some text
        user_input = input("> ")

        if user_input.lower() == "bye":
            # if they typed in 'bye' (or even BYE, ByE, byE etc.), break out of the loop
            break
        else:
            # Create a TextBlob based on the user input. Then extract the noun phrases
            user_input_blob = TextBlob(user_input, np_extractor=extractor)
            np = user_input_blob.noun_phrases
            response = ""
            if user_input_blob.polarity <= -0.5:
                response = "Oh dear, that sounds bad. "
            elif user_input_blob.polarity <= 0:
                response = "Hmm, that's not great. "
            elif user_input_blob.polarity <= 0.5:
                response = "Well, that sounds positive. "
            elif user_input_blob.polarity <= 1:
                response = "Wow, that sounds great. "

            if len(np) != 0:
                # There was at least one noun phrase detected, so ask about that and pluralise it
                # e.g. cat -> cats or mouse -> mice
                response = response + "Can you tell me more about " + np[0].pluralize() + "?"
            else:
                response = response + "Can you tell me more?"
            print(response)

    print("It was nice talking to you, goodbye!")

# Start the program
main()
```



```
Hello, I am Marvin, the friendly robot.
You can end this conversation at any time by typing 'bye'
After typing each answer, press 'enter'
How are you today?
> I am ok
Well, that sounds positive. Can you tell me more?
> I went for a walk and saw a lovely cat
Well, that sounds positive. Can you tell me more about lovely cats?
> cats are the best. But I also have a cool dog
Wow, that sounds great. Can you tell me more about cool dogs?
> I have an old hounddog but he is sick
Hmm, that's not great. Can you tell me more about old hounddogs?
> bye
It was nice talking to you, goodbye!
```

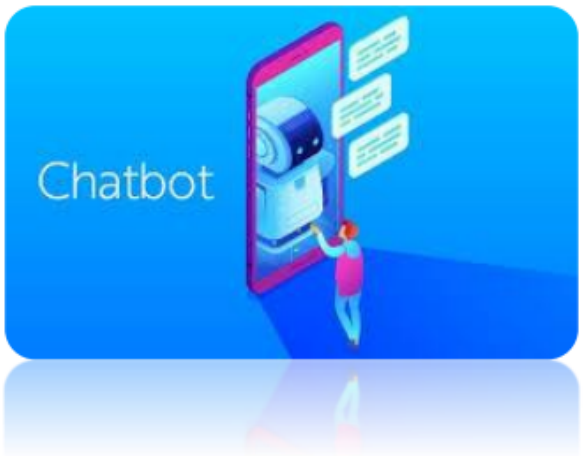


STEP 1: SETUP DELL'AMBIENTE DI SVILUPPO

```
pip install chatterbot
pip install chatterbot_corpus

#se hai python3
pip3 install chatterbot
pip3 install chatterbot_corpus
```

STEP 2: CREAZIONE DEL BOT

```
my_bot = ChatBot(name='Il_NostroPrimo_BOT', read_only=True,
                 logic_adapters=
['chatterbot.logic.MathematicalEvaluation',
                                 'chatterbot.logic.BestMatch'])
```



STEP 3: ADDESTRAMENTO DEL BOT

STEP 3: ADDESTRAMENTO DEL BOT

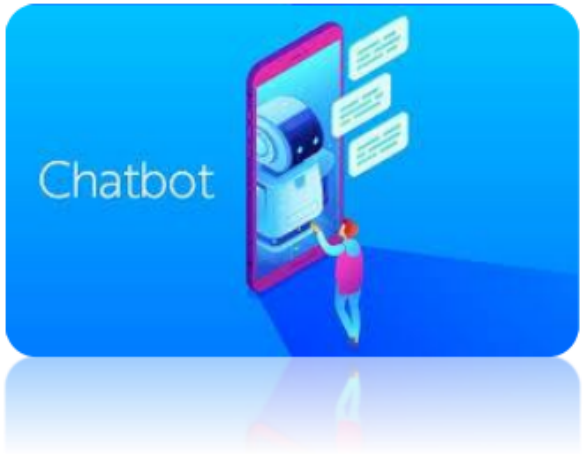
```
small_talk = ['Ciao!',
              'Ciao Umano!',
              'Come stai?',
              'Come va?',
              'Sono un robot sto benissimo.',
              'Meglio di un robot chi può stare, tu come stai?',
              'sto bene',
              'felice di sentirlo.',
              'mi sento benissimo',
              'eccellente, felice di sentirlo.',
              'non così bene',
              'mi dispiace sentirlo.',
              'come ti chiami?',
              'Sono pybot. fammi una domanda di matematica, per
favore.']

math_talk_1 = ['teorema di pitagora',
               'a al quadrato più b al quadrato uguale a c al
quadrato.']

math_talk_2 = ['legge dei coseni',
               'c**2 = a**2 + b**2 - 2 * a * b * cos(gamma)']
```

```
list_trainer = ListTrainer(my_bot)
for item in (small_talk, math_talk_1, math_talk_2):
    list_trainer.train(item)
```

STEP 4: COMUNICARE CON IL BOT

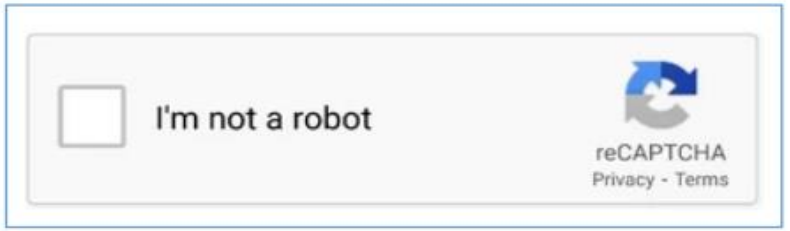
```
a = input()
while a != "esci":
    print(my_bot.get_response(a))
    a = input()
```

```
In [9]: a = input()
        while a != "esci":
            print(my_bot.get_response(a))
            a = input()

        Ciao
        Ciao Umano!
        Come ti chiami
        Sono pybot. fammi una domanda di matematica, per favore.
        Teorema di pitgora
        sto bene
        teorema di pitagora
        a al quadrato più b al quadrato uguale a c al quadrato.
        esci
```



UN BOT È «SEMPLICEMENTE» UN PROGRAMMA SOFTWARE AUTOMATIZZATO CHE PUÒ ESSERE UTILIZZATO PER SCOPI LEGITTIMI O DANNOSI

VS



- Chat GPT è un chatbot AI online progettato per supportare in una vasta gamma di attività.

- Chat GPT utilizza l'elaborazione del linguaggio naturale (NLP) per comprendere le domande e fornire risposte pertinenti.

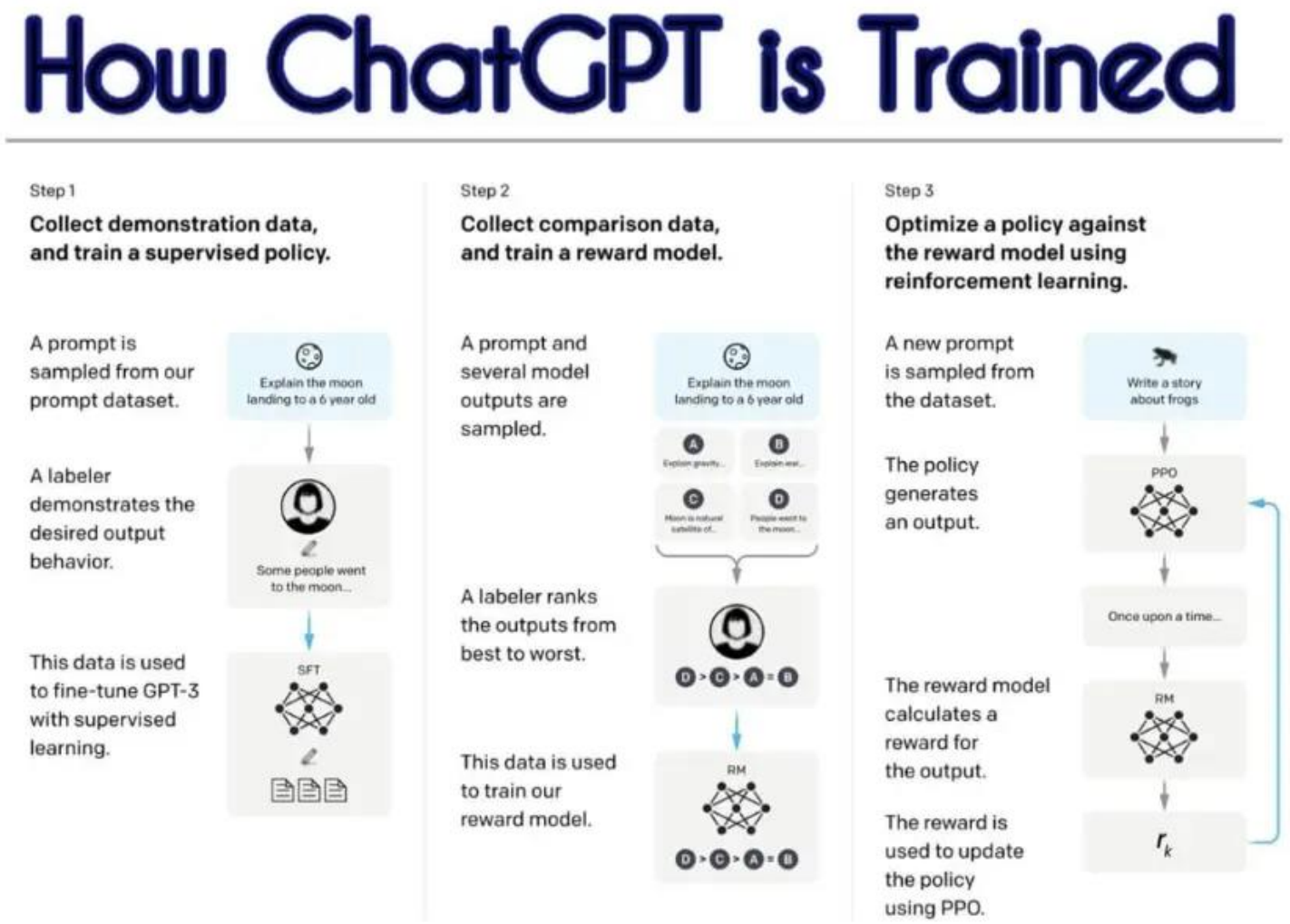
- Chat GPT è stato creato da OpenAI e lanciato a novembre 2022.

- OpenAI è stata fondata nel 2015 da un gruppo di influenti imprenditori e investitori tecnologici, tra cui Elon Musk, Sam Altman, Greg Brockman e Ilya Sutskever.

- Utilizza il modello di linguaggio GPT-3, che alimenta Chat GPT.

- Generative Pre-trained Transformer 3 (GPT-3) è una combinazione di algoritmi avanzati di deep learning e un enorme set di dati di testo, che consente di generare testo in linguaggio naturale di alta qualità e comprendere e rispondere all'utente input in modo colloquiale.

GPT (***Generative Pretrained Transformer)***. Strumento di elaborazione del linguaggio naturale, un'applicazione di intelligenza artificiale che utilizza algoritmi di machine learning.



- L’addestramento di un GPT è concettualmente molto semplice: si chiede al modello di predire il prossimo token a partire dai precedenti N, in maniera sequenziale

Il gatto è seduto il tavolo della cucina

Il gatto è seduto <?> Il gatto è seduto sopra <?>. Il gatto è seduto sopra il <?>

- Oltre al successo di chatGPT, GPT ha esibito proprietà interessanti sin dalle prime iterazioni:
  - Capacità di generalizzazione zero-shot: GPT (dalla versione 2) ha dimostrato capacità di performare alcuni task senza alcun tipo di addestramento specifico
  - Capacità di generalizzazione few-shot: capacità di raggiungere performances prossime allo stato dell’arte con pochi esempi

- GPT-2 ha rappresentato un passo in avanti sostanziale, tanto che fu inizialmente mantenuto non disponibile al pubblico per paura degli effetti sociali che avrebbe potuto avere
- Rappresenta uno smart auto-complete
  - Generazione di contenuto (propaganda)
  - Generazione di codice
- Tuttavia GPT-2 spesso generava risposte senza senso, grottescamente inaccurate e molto biased
- GPT-2, con i suoi 1.5 miliardi di parametri (circa 6 GB) rappresenta l'ultimo modello linguistico (prima degli sviluppi moderni) con cui è possibile fare inferenza (non addestramento) su hardware desktop