Natural Language Processing

Contents

- Introduction to NLP and its Use Cases
- Natural Language Basics
- Text Wrangling
- Feature Engineering

Introduction

- NLP is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.

- In simple terms, a natural language is a language developed and evolved by humans through natural use and communication rather than constructing and creating the language artificially, like a computer programming language.

Some Use Cases of NLP

- Machine Translation
- Question Answering Systems
- Text Summarisation
- Text Categorisation
- Text Analytics

Some Applications

- Spam detection
- News articles categorization
- Social media analysis and monitoring
- Biomedical
- Marketing
- Sentiment analysis
- Chatbots
- Virtual assistants

Natural Languages Background

- Linguistics is the scientific study of language, a special field that deals with various aspects of language.
- The main things to consider in any language are-
  - Syntax
  - Semantics
  - Grammar

Syntax

- **Definition**
  Syntax refers to the set of rules, principles, and processes that govern the structure of sentences in a given language, specifically the order of words and phrases.
- **Example**
  Correct Syntax: "The cat sat on the mat."
  Incorrect Syntax: "Cat the on mat the sat."
- **Key Points**
- Syntax rules help ensure clarity and understandability in language.
- Syntax is crucial for NLP models to parse and interpret sentences correctly.

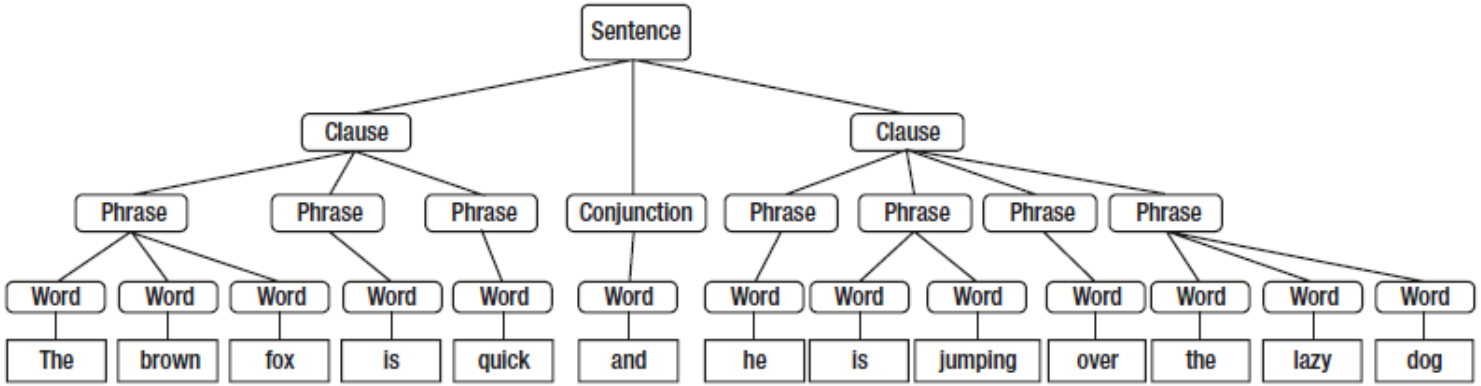
Language Syntax

- Words are combined into phrases. Phrases are combined into clauses and clauses are combined into sentences.
- Shallow parsing (also chunking) is an analysis of a sentence which first identifies constituent parts of sentences (nouns, verbs, adjectives, etc.) and then links them to higher order units that have discrete grammatical meanings (noun groups or phrases, verb groups, etc.).

- Example - The brown fox is quick and he is jumping over the lazy dog.

dog the over he
lazy jumping is the fox
and is quick brown

A collection of words without any relation or structure



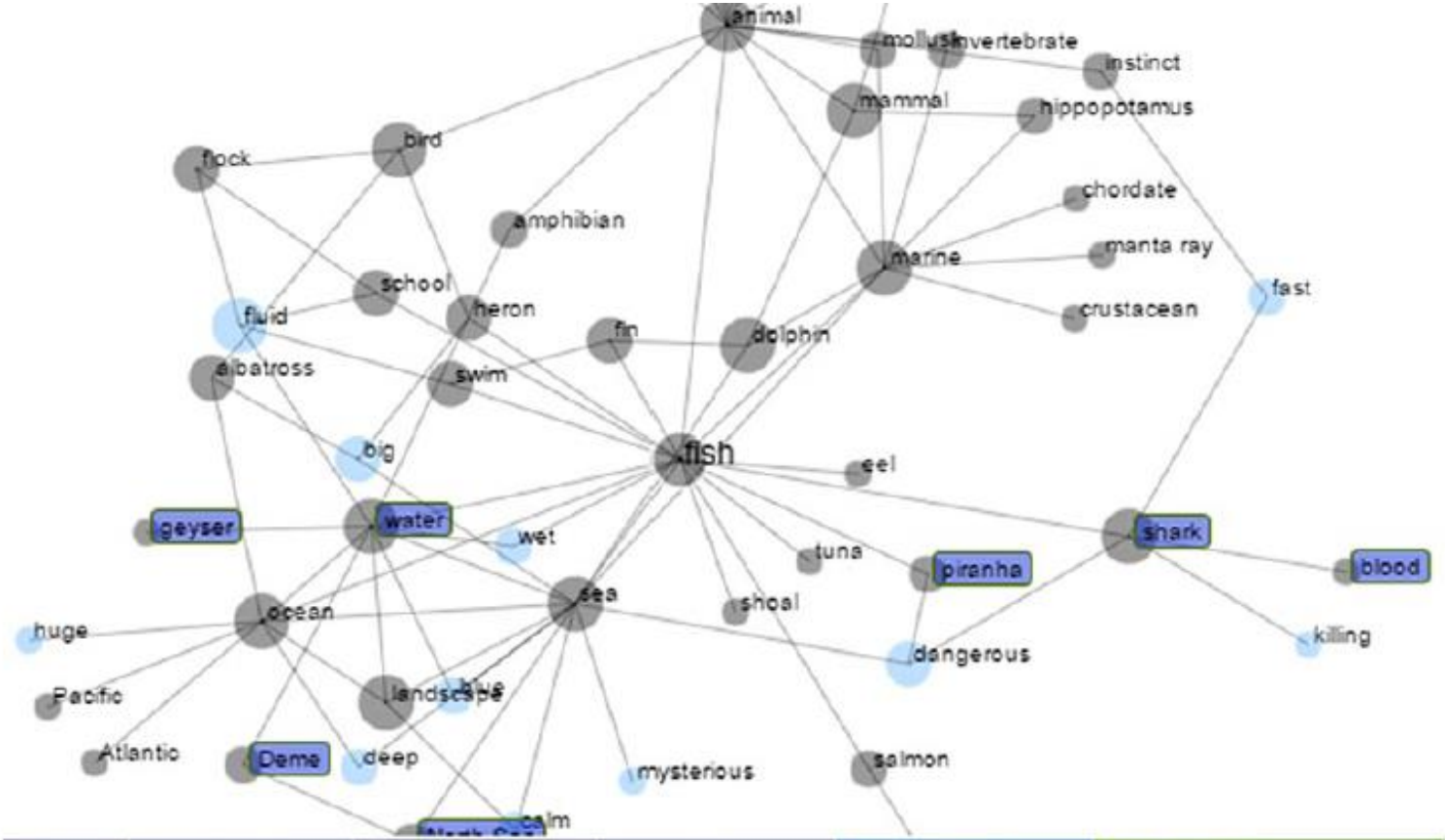
Semantics

- **Definition**
  Semantics is the study of meaning in language, focusing on understanding what words, phrases, and sentences signify and convey in context.
- **Example**
  "The bank is on the river" (meaning: a riverbank)
  "The bank is closed on Sundays" (meaning: a financial institution)
- **Key Points**
- Semantics involves deciphering meaning based on word context.
- It helps NLP models accurately understand and generate contextually appropriate responses.

Semantics

- A Lemma is also known as the canonical or citation form for a set of words.
- The lemma is usually the base form of a set of words, known as a lexeme.

- For example, {eating, ate, eats} is a lexeme and the lemma of these words is the word “eat”.



IS-A	HAS-SPECIFIC	IS-PART-OF	HAS-PART	HAS-PROPERTY	IS-RELATED-TO
animal	eel	ocean	fin	wet	heron
marine	piranha	school			
	salmon	sea			
	shark	water			
	shoal				
	tuna				

Semantic network around the concept of fish

Grammar

- **Definition**
  Grammar is a system of rules that defines the correct use of words, structures, and punctuation in a language.
- **Example**
  Correct Grammar: "She is going to the store."
  Incorrect Grammar: "She are going to store."
- **Key Points**
- Grammar ensures sentences are logically structured and interpretable.
- Proper grammar is essential for NLP tasks like translation, summarization, and chatbots for coherent responses.

Grammar

- It primarily consists of a set of rules that are used to determine how to position words, phrases, and clauses when constructing sentences in a natural language.

S -> NP VP

NP -> [DET][ADJ]N[PP]

VP -> V|MD[VP][NP][PP][ADJP][ADVP]

PP -> PREP[NP]

Grammar

S -> NP VP: This rule defines a sentence as consisting of a **Noun Phrase (NP)** followed by a **Verb Phrase (VP)**.

- **NP (Noun Phrase)**: This is the subject of the sentence, which can include a determiner, adjectives, and a noun (e.g., "The cat").
- **VP (Verb Phrase)**: This is the predicate of the sentence, which includes a verb and can also include objects, adverbs, and additional phrases (e.g., "sat on the mat").

Example

- For the sentence "The cat sat on the mat," the structure would be:
- **NP**: "The cat"
- **VP**: "sat on the mat"
- This rule is fundamental in NLP as it represents the basic sentence structure in English, helping parse and interpret simple sentences by breaking them down into subject (NP) and predicate (VP). This structure underpins many syntactic parsing tasks and is essential for language understanding models.

Grammar

NP -> [DET][ADJ] N [PP]: This production rule indicates the possible structure of a Noun Phrase (NP).

- **DET**: A determiner, which specifies the noun (e.g., "the," "a," "some").
- **ADJ**: An adjective, describing or modifying the noun (e.g., "big," "delicious").
- **N**: The noun, the main element of the phrase (e.g., "dog," "cake").
- **PP**: A Prepositional Phrase, providing additional context or information (e.g., "on the table").

Example

- In a phrase like "the delicious cake on the table," the breakdown could look like this:
- **DET**: "the"
- **ADJ**: "delicious"
- **N**: "cake"
- **PP**: "on the table"
- This structure helps NLP systems identify and analyze noun phrases within sentences, aiding in tasks like entity recognition, syntactic parsing, and language understanding. It shows how components can combine to form meaningful phrases.

Grammar

VP -> V | MD [VP][NP][PP][ADJP][ADVP]: This is a production rule indicating how a Verb Phrase (VP) can be structured.

- **V**: A verb (e.g., "run," "eat").
- **MD**: A modal verb (e.g., "can," "will").
- **VP**: Another Verb Phrase, indicating recursion (a VP can contain another VP).
- **NP**: A Noun Phrase, typically the object or subject of the verb (e.g., "the dog").
- **PP**: A Prepositional Phrase, providing additional information (e.g., "in the park").
- **ADJP**: An Adjective Phrase, which can describe the subject or object (e.g., "very happy").
- **ADVP**: An Adverbial Phrase, adding information about the verb (e.g., "quickly").

Example

- In a sentence like "She can quickly eat the delicious cake in the park," the breakdown might look like this:
- **VP** (Verb Phrase) includes:
  - **MD**: "can"
  - **V**: "eat"
  - **ADVP**: "quickly"
  - **NP**: "the delicious cake"
  - **PP**: "in the park"
- This structure helps NLP systems parse sentences into their grammatical components, which is essential for tasks like syntactic parsing and understanding sentence structure.

Grammar

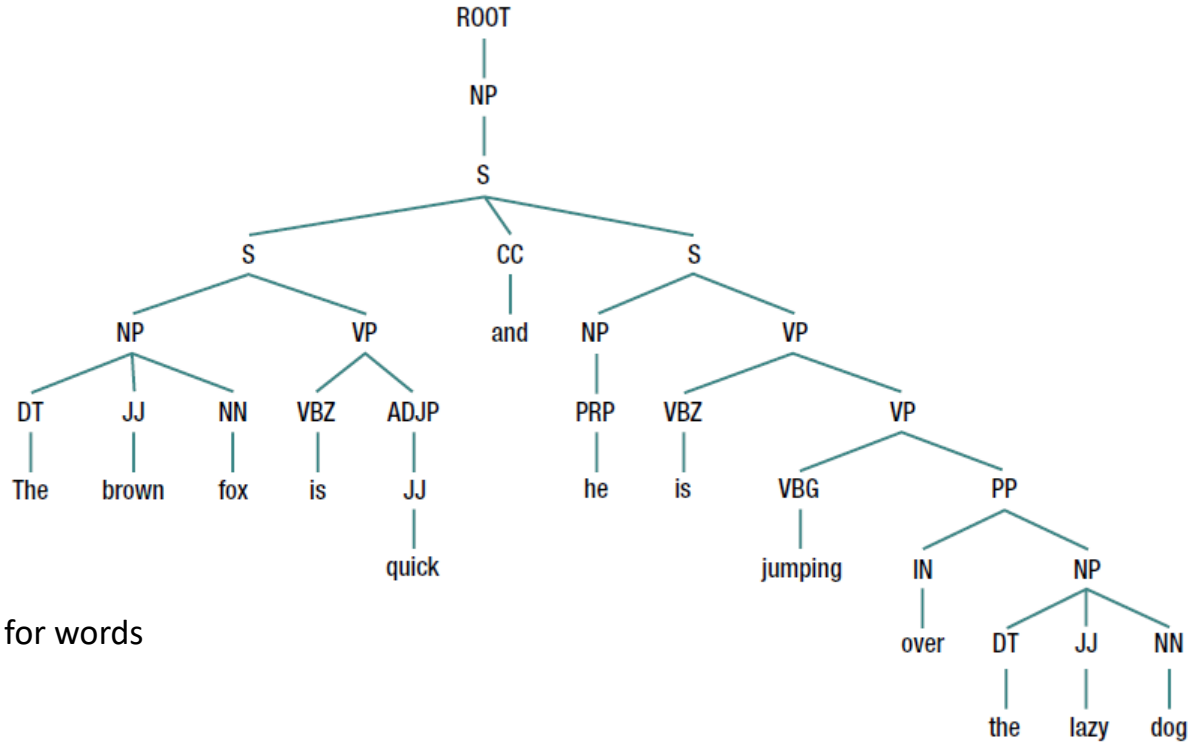
PP -> PREP [NP]: This rule defines a Prepositional Phrase (PP) as consisting of a **Preposition (PREP)** followed by a **Noun Phrase (NP)**.

- **PREP**: A preposition, which establishes a relationship between the noun phrase and other parts of the sentence (e.g., "in," "on," "with").
- **NP (Noun Phrase)**: The object of the preposition, which can include a determiner, adjectives, and a noun (e.g., "the park").

Example

- In a phrase like "in the park," the structure would be:
- **PREP**: "in"
- **NP**: "the park"
- This structure is used in NLP to parse sentences and understand spatial, temporal, and other relationships between entities. Prepositional phrases provide additional context to sentences, making them essential in sentence interpretation tasks.

Constituency tree



POS tags for words

Text Corpora

A Text corpora is the plural form of “text corpus” and can be defined as large and structured collections of texts or textual data.

Some of the most commonly used methods and techniques for annotating text corpora are-

- POS (Part of Speech): Annotating each word with a POS tag indicating the part of speech associated with it.
- Word stems: A stem for a word is a part of the word to which various affixes can be attached.
- Word lemmas: A lemma is the canonical or base form for a set of words and is also known as the head word.
- Grammar based parsing
- Semantic types and roles: The various constituents of sentences, including words and phrases, are annotated with specific semantic types and roles often obtained from an ontology that indicates what they do. These include things like place, person, time, organization, agent, recipient, theme, etc

Why is Analysing Text Data Difficult

- Unstructured data.
- Language is inherently very high dimensional and sparse. There are tons of rare words, words can mean the same thing in different contexts.
- There is no way to define a word in an unambiguous way. 2 = 1 + 1, but how do you define e.g. family = ? love = ? There is no universal definition for most of the concepts we use everyday.
- No readily available features like in numerical data, images or audio data.

Text Pre-Processing

- **Removing HTML tags** – use the library BeautifulSoup or regex
- **Tokenization** – splitting into tokens using delimiters, sentence/word tokenizers available in nltk and spaCy
- **Removing accented characters**(converting é to e) – Unicode.normalize
- **Removing stopwords**(words having littile or no significance, eg - a, an, the… – nltk.corpus.stopwords.words('english')
- **Case conversions** – string functions
- **Removing special characters** – regex

Text Pre-Processing

- **Handling contractions** – create a dictionary of contractions and replace them
- **Correcting spelling errors** – very tricky, libraries like textblob and PyEnchant available
- **Stemming** – jumping/jumped/jumps = jump (removing prefix/suffix)
- **Lemmatization** – ate=eat, fancy=fancier

Feature Engineering

- Vector space model:

A document **D** in a document vector space **VS**.

$$VS = \{W1, W2, \ldots , Wn\}$$

where there are **n** distinct words across all documents.

$$D = \{wD1, wD2, \ldots , wDn\}$$

where wDn denotes the weight for word **n** in document **D**. This weight is a numeric value and can be anything ranging from the frequency of that word in the document, the average frequency of occurrence, embedding weights, or the *TF-IDF weight.*

Bag of Words (BoW) Model

- Definition: BoW is a simple NLP model that represents text as a collection of words without considering grammar or word order, only the presence or frequency of words.
- Example:
- Text: "The cat sat on the mat."
- BoW Vector: {"the": 2, "cat": 1, "sat": 1, "on": 1, "mat": 1}
- Strengths: Simple to implement, useful for small texts or basic sentiment analysis.
- Weaknesses: Ignores grammar and word order, doesn't capture context.

N-gram Model

- Definition: An N-gram is a contiguous sequence of n items (words or characters) from a text, allowing the model to capture some word order information.
- Example:
- Text: "The cat sat"
- - Unigram (1-gram): "The", "cat", "sat"
- - Bigram (2-gram): "The cat", "cat sat"
- - Trigram (3-gram): "The cat sat"
- Strengths: Captures partial context with word sequences, useful for language generation.
- Weaknesses: Doesn’t capture long-range dependencies; sparse for large N.

TF-IDF (Term Frequency-Inverse Document Frequency)

- Definition: A statistical measure that evaluates how important a word is to a document relative to a collection of documents.
- Example:
- Text: "The cat sat on the mat." in a corpus where "cat" is unique but "the" appears often.
- - TF-IDF gives higher weight to "cat" than "the".
- Strengths: Emphasizes rare but relevant words, useful in information retrieval.
- Weaknesses: Doesn’t capture semantic meaning, word order, or context.

TF-IDF Model

$$tfidf = tf \times idf$$

Term frequency in any document vector is denoted by the raw frequency (fractional) value of that term(word) in a particular document.

$$tf(w,D) = f_{w_D}$$

$$idf(w,D) = 1 + \log \frac{N}{1 + df(w)}$$

N represents the total number of documents in our corpus, and *df* (*w*) represents the number of documents in which the term ***w*** is present.

TF-IDF Model

0	The sky is blue and beautiful.
1	Love this blue and beautiful sky!
2	The quick brown fox jumps over the lazy dog.
3	A king's breakfast has sausages, ham, bacon, eggs, toast and beans
4	I love green eggs, ham, sausages and bacon!
5	The brown fox is quick and the blue dog is lazy!
6	The sky is very blue and the sky is very beautiful today
7	The dog is lazy but the brown fox is quick!

$$tf(w,D) = f_{w_D} \qquad idf(w,D) = 1 + \log\frac{N}{1+df(w)} \qquad tfidf = tf \times idf$$

Word Embeddings (e.g., Word2Vec, GloVe)

- Definition: Word embeddings represent words in a dense vector space, capturing semantic relationships. For example GloVe (Global Vectors for Word Representation) is a model for word embedding, which represents words as continuous-valued vectors in a multi-dimensional space.
- Example:
- "King" - "Man" + "Woman" ≈ "Queen" (captures gender-related context in vectors).
- Strengths: Captures semantic meaning and word similarity.
- Weaknesses: Requires a lot of data; fixed embeddings lack flexibility across contexts.

Latent Dirichlet Allocation (LDA)

- Definition: A probabilistic model that discovers topics in a collection of documents by finding word patterns.
- Example:
- From documents about cooking, LDA might identify topics like "ingredients," "methods," and "tools."
- Strengths: Useful for topic modeling and unsupervised classification.
- Weaknesses: Can struggle with ambiguous words and requires tuning.

FastText

- Definition: An extension of Word2Vec that represents words as the sum of character n-grams, allowing it to handle misspellings and out-of-vocabulary words.
- Example:
- Words like "apple" and "apples" are closer in vector space because they share similar character n-grams.
- Strengths: Good for handling rare words, typos, and morphologically rich languages.
- Weaknesses: Still context-independent, lacking adaptability to polysemy (one word with multiple meanings).

Encoder-Decoder Models

- Definition: A sequence-to-sequence (Seq2Seq) architecture where the encoder processes the input sequence, and the decoder generates an output sequence, often used for tasks like translation.
- Example:
- In machine translation, the encoder converts English sentences into context vectors, and the decoder generates equivalent sentences in French.
- Strengths: Effective for translation, summarization, and other sequence transformation tasks.
- Weaknesses: Traditional Seq2Seq models may struggle with very long sequences without attention mechanisms.

Attention Mechanism

- Definition: A mechanism allowing models to focus on relevant parts of the input when generating each word in the output, critical in complex tasks like translation.
- Example:
- In translation, while translating “I love New York,” the model learns to focus on “New York” as a unit when translating it.
- Strengths: Improves performance in sequence-to-sequence tasks, especially with long sentences.
- Weaknesses: Attention alone doesn't manage context as well as Transformer models.

Recurrent Neural Networks (RNNs)

- Definition: A type of neural network that processes sequential data by maintaining hidden states over time, which allows it to capture temporal dependencies.
- Example:
- Text: "The stock market went up today."
- RNN captures the sequential nature and may retain the "trend" of "up" over time.
- Strengths: Good for time series and sequence prediction.
- Weaknesses: Struggles with long-term dependencies; may suffer from vanishing gradients.

Long Short-Term Memory Networks (LSTM)

- Definition: A variant of RNNs designed to capture long-term dependencies using memory cells to control information retention.
- Example:
- LSTM can process sentences where word relationships are distant, like "She remembered that she wanted to go, but forgot."
- Strengths: Excellent for long sequences; mitigates vanishing gradient problem.
- Weaknesses: Computationally intensive and complex.

ELMo (Embeddings from Language Models)

- Definition: A contextual word embedding model that captures the meaning of a word based on its surrounding words by combining left-to-right and right-to-left LSTMs.
- Example:
- The word “bat” in “He swung the bat” vs. “The bat flew at night” has different embeddings based on sentence context.
- Strengths: Captures contextual meaning, allowing for more accurate representation in polysemous words.
- Weaknesses: Large and slow compared to newer transformer-based embeddings.

Transformers (e.g., BERT, GPT)

- Definition: A deep learning model using self-attention mechanisms to process all words in a sequence simultaneously, capturing context more efficiently.
- Example:
- BERT can process "The bank can be slippery" and infer "bank" as a riverbank or financial institution based on sentence context.
- Strengths: Excels at capturing context and meaning; state-of-the-art in NLP.
- Weaknesses: High computational cost, large training data required.

Convolutional Neural Networks (CNNs) for NLP

- Definition: CNNs, typically used in image processing, are adapted for NLP to detect local patterns in text, such as n-grams, using convolutional layers.
- Example:
- CNNs can capture local word patterns like “good movie” or “bad acting” in a sentence for sentiment analysis.
- Strengths: Effective in capturing short-range dependencies and local context.
- Weaknesses: Not as effective for long-range dependencies compared to RNNs or Transformers.

Conditional Random Fields (CRFs)

- Definition: A probabilistic model used to predict sequences, often for tasks like part-of-speech tagging and named entity recognition.
- Example:
- In named entity recognition, CRF can label "New York" as a location and "Apple" as an organization based on context.
- Strengths: Highly effective for structured prediction in sequence labeling tasks.
- Weaknesses: Limited scalability, often combined with neural networks for deep learning.

Bidirectional Encoder Representations from Transformers (BERT)

- Definition: A Transformer-based model pre-trained on a large corpus, allowing it to understand context from both left-to-right and right-to-left.
- Example:
- BERT can understand nuanced sentences like “The man saw the woman with a telescope” by using context from all directions.
- Strengths: Sets the standard for a wide range of NLP tasks due to its contextual awareness.
- Weaknesses: Computationally demanding; large memory footprint.

GPT (Generative Pre-trained Transformer)

- Definition: An autoregressive Transformer model pre-trained on a large corpus, generating text by predicting the next word in a sequence.
- Example:
- Used for text generation tasks, GPT can complete or expand sentences based on context, like creating dialogue for a chatbot.
- Strengths: High-quality text generation and effective in various conversational applications.
- Weaknesses: Prone to generating incorrect information and bias; limited context window.

XLNet

- Definition: A permutation-based model that captures the benefits of autoregressive and autoencoding language models, outperforming BERT in certain NLP tasks.
- Example:
- In question answering, XLNet uses permutations to capture both directions, allowing it to understand nuanced questions.
- Strengths: Strong performance in tasks where bidirectional context is critical.
- Weaknesses: Complex training process, high resource demand.

T5 (Text-To-Text Transfer Transformer)

- Definition: A transformer model that treats all NLP tasks as text-to-text problems, making it versatile for translation, summarization, and more.
- Example:
- In summarization, T5 converts input text to a brief summary by treating it as a sequence-to-sequence task.
- Strengths: Uniform framework for various tasks; flexible and powerful.
- Weaknesses: Requires fine-tuning for specific tasks, large memory requirements.