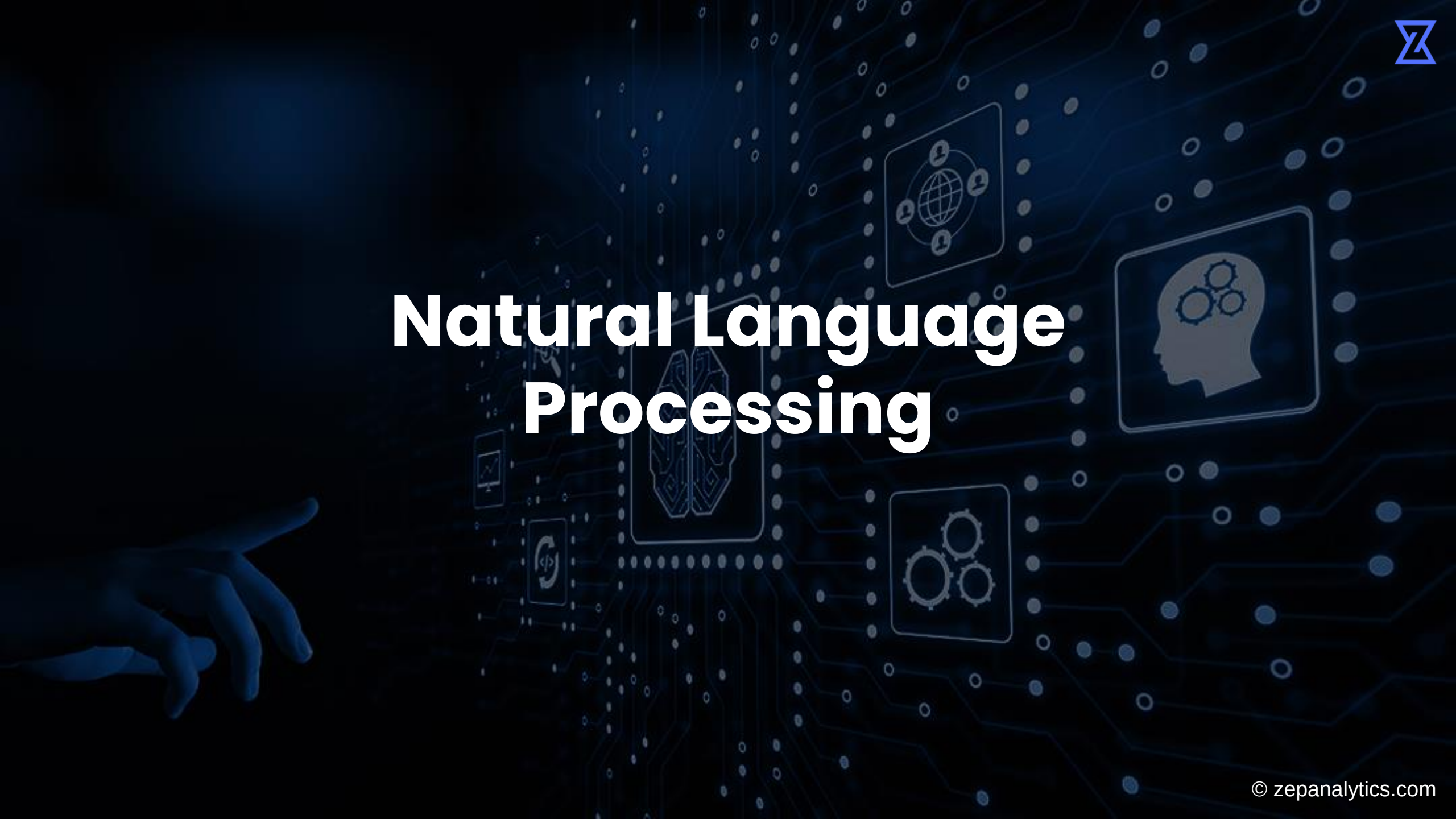


Agenda

- Introduction to NLP
- Linguistic Foundations
- Text Preprocessing & Tokenization
- NLP Libraries & Tools
- Word Embeddings
- Neural Networks for NLP
- Advanced NLP Techniques
- NLP Applications

Introduction to NLP

Understanding the World of Human Language and Computers

Natural Language Processing

Natural Language Processing (NLP) is a field of study that focuses on the interaction between computers and human language. It involves the development of algorithms and models to enable computers to understand, interpret, and generate human language.

Overview of NLP and its Applications

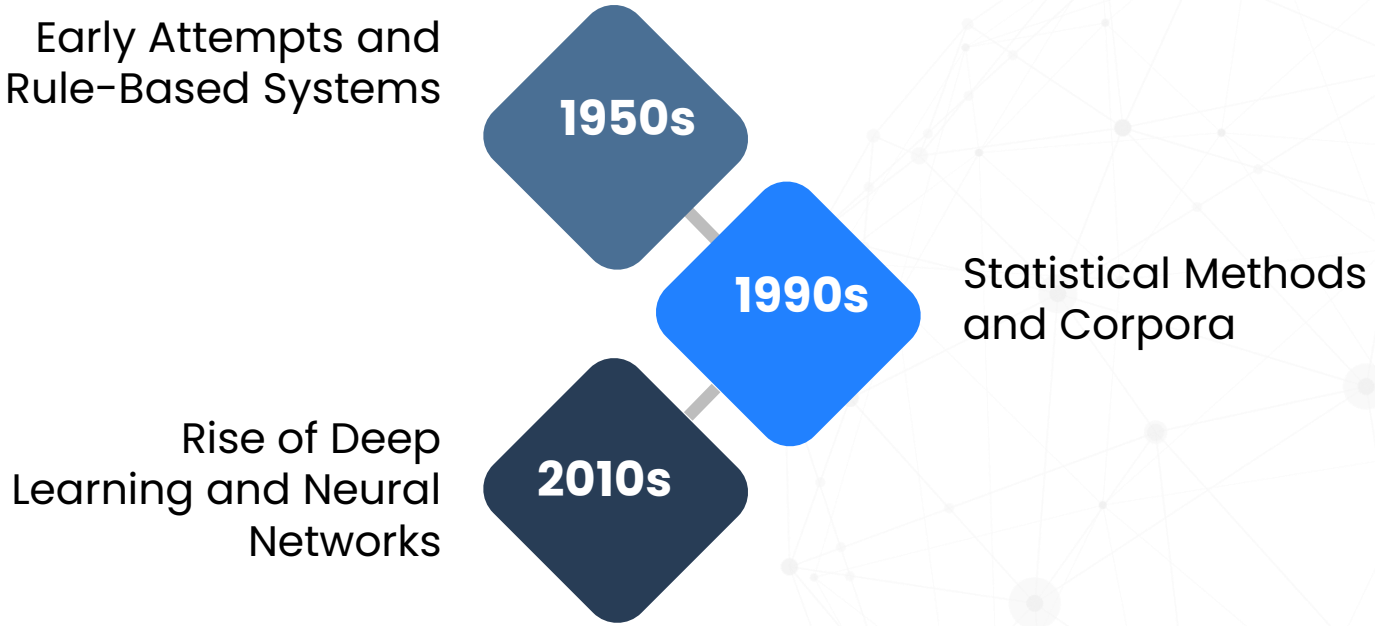
01 **Language Translation -** NLP enables computers to translate text or speech from one language to another, facilitating cross-language communication and accessibility.

02 **Sentiment Analysis -** NLP techniques are used to identify and extract subjective information from text data, helping in understanding public opinion and sentiment.

03 **Chatbots -** NLP powers the development of chatbots that can interact with humans in natural language, transforming customer service and support.

Historical Perspectives

Evolution of NLP



Milestones

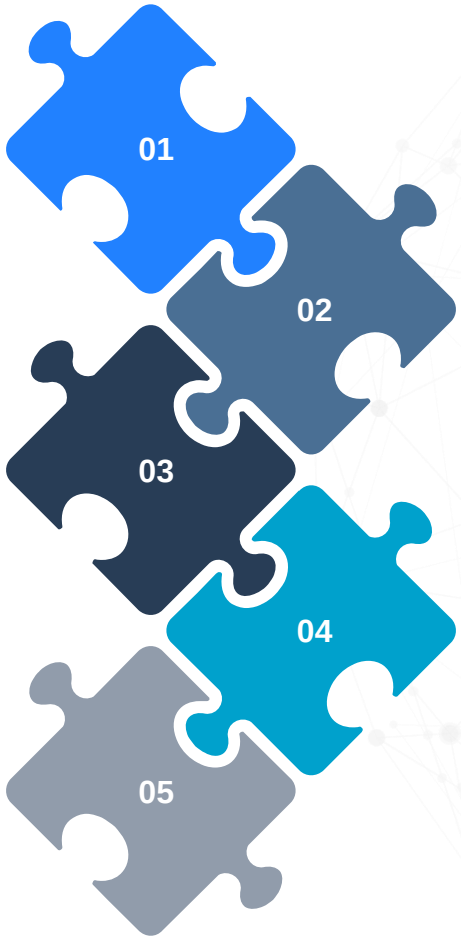
- Turing Test (1950)
- Introduction of HMMs for Speech Recognition (1970s)
- Google's Neural Machine Translation (2016)

Key Challenges in NLP

Ambiguity:
•Multiple Meanings of Words and Phrases

Lack of Standardization:
•Variation in Language Across Regions and Dialects

Ethical Considerations:
•Bias in Models, Privacy Concerns, and Responsible AI



Context Understanding:
•Grasping Contextual Nuances in Language

Data Sparsity:
•Limited Data for Training Models, Especially in Specialized Domains

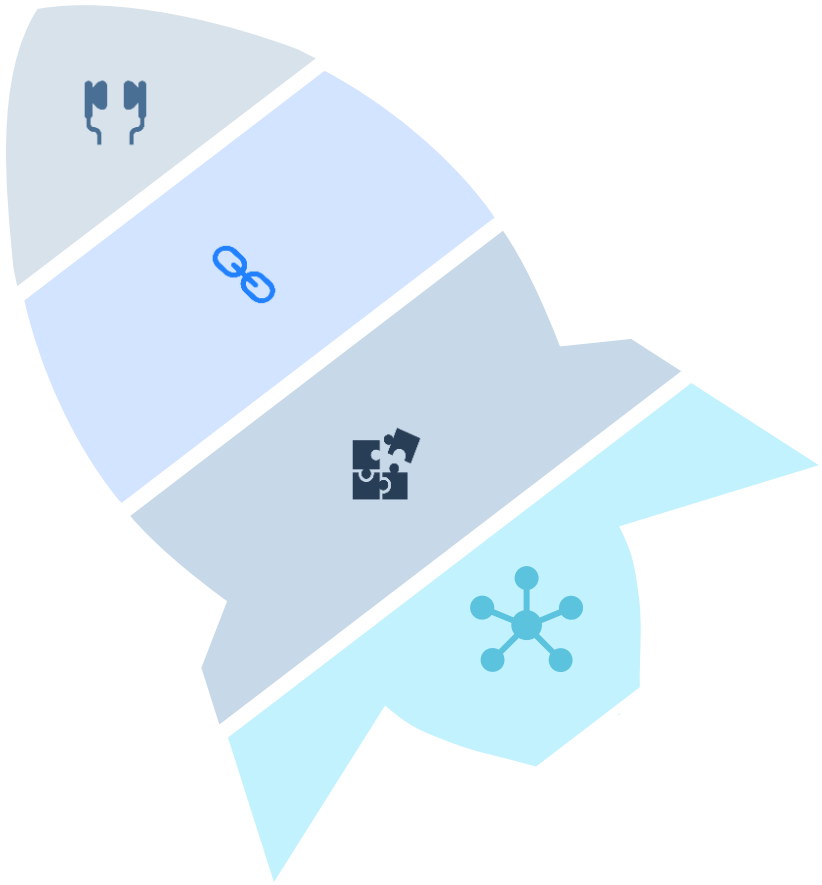
Linguistic Foundations

Understanding the Building Blocks of Human Language

•Agenda:

- Basics of Linguistics for NLP
- Syntax and Semantics
- Morphology and Phonetics

Basics of Linguistics



Phonetics (Sounds)
Investigates the sounds of language and how they are produced and perceived.

Morphology (Word Structure)
Analyzes the internal structure of words and how they form.

Syntax (Sentence Structure)
Examines the rules governing the arrangement of words to form grammatically correct sentences.

Semantics (Meaning)
Studies the meaning and interpretation of words, phrases, and sentences.

Phonetics

- Unlocking the Sound of Language
- Speech Recognition Systems: Transforming spoken words into digital text.
- Pronunciation Variation: Understanding diverse ways words are spoken.

Definition:

Phonetics explores the physical sounds of human speech.

Examples:

Vowels, Consonants, Speech Sounds

Morphology

- Decoding Word Construction
- Word Stemming: Extracting the root form for information retrieval.
- Word Formation: Unraveling the creation of new words in machine translation.

Definition:

Morphology delves into the structure of words and how they are formed.

Examples:

Prefixes, Suffixes, Roots

Syntax

- Building the Framework of Language
- Grammar Checkers: Ensuring correct sentence construction.
- Sentence Parsing: Understanding the grammatical structure in NLP models.

Definition:

Syntax examines sentence structure and the rules governing word order.

Examples:

Subject-Verb-Object (SVO) Structure

Semantics

- Unraveling Layers of Meaning
- Understanding Context: Vital for sentiment analysis.
- Ambiguity Resolution: Disambiguating multiple meanings of words.

Definition:

Semantics explores the meaning within language.

Examples:

Word Sense Disambiguation, Lexical Semantics

Integrating Linguistic Foundations in NLP

NLP Applications:

- Named Entity Recognition (NER)
- Machine Translation
- Sentiment Analysis
- Question Answering Systems

Challenges:

- Ambiguity Handling: Navigating the complexities of words and phrases with multiple meanings.
- Context Understanding: Grasping nuanced meaning in different situations and contexts.
- Multilingual NLP: Overcoming language variations and diversities for global applicability.

Text Preprocessing & Tokenization

Case Folding

Case Insensitivity –
Text data is converted to lowercase to ensure uniformity and eliminate the impact of letter case on downstream processes.

Normalization –
Case folding helps in standardizing the text data, making it easier to process and analyze.

Enhanced Consistency –
It ensures that the same word in different cases is treated as identical, preventing redundancy and inconsistency in the dataset.

Input:
Can We Use NAtural LanguagE prOcEssIng on MARATHI?

Output:
can we use natural language processing on marathi?

Code:
text = text.lower()

Special Character Removal

Eliminating non-alphanumeric characters, aiding in noise reduction and improving text clarity for processing.

Why to Remove this Characters?

Elimination of Noise –
Special characters, such as punctuation marks and symbols, are removed to clean the text data.

Preprocessing for Tokenization –
Remove special characters to prepare text for meaningful tokenization

Code -

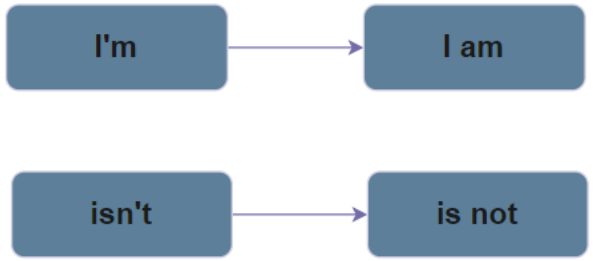
```
def process_text(text):
    # Case Folding
    text = text.lower()

    # Special Character Removal using regular
     expressions
    text = re.sub(r'[^a-zA-Z0-9\s]', '', text)

    return text
```

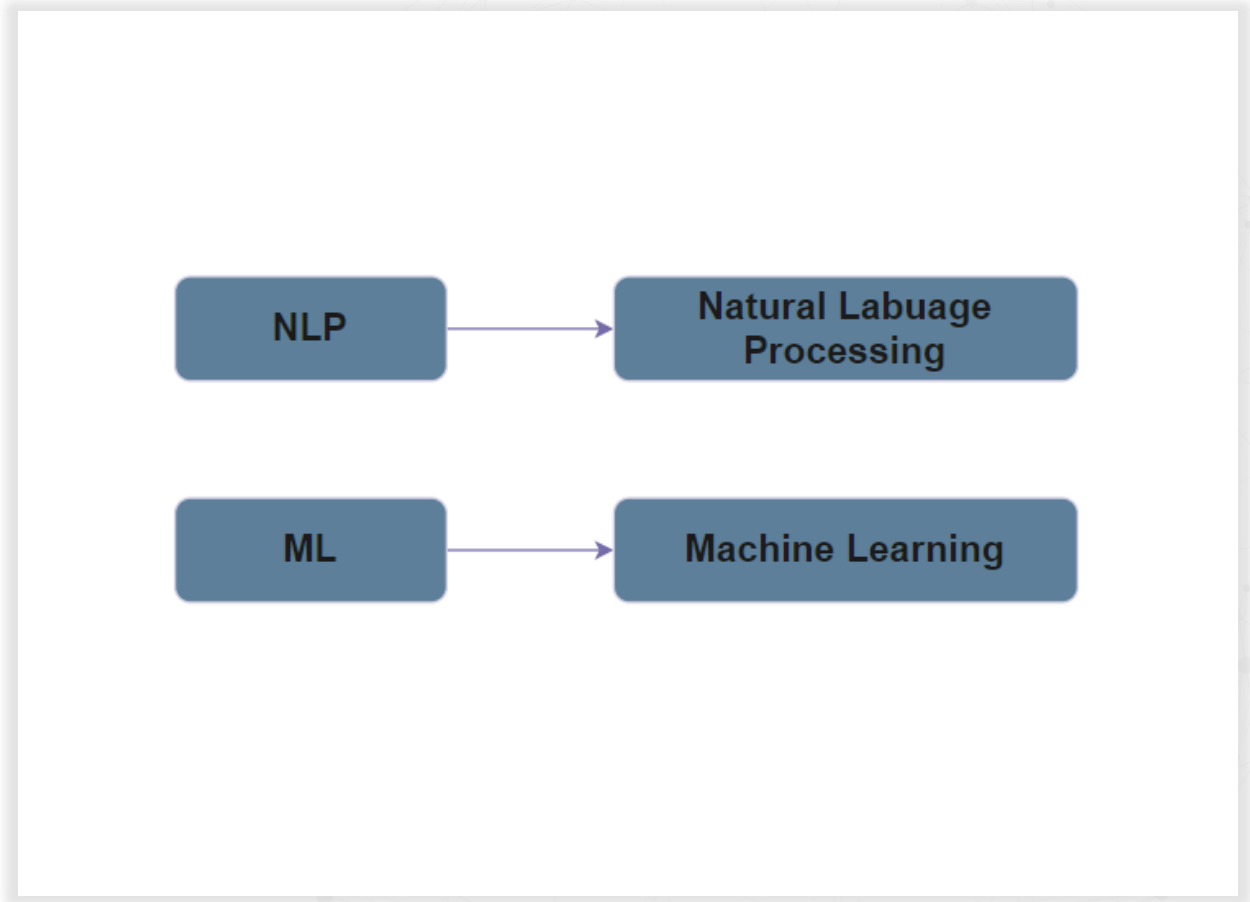
Handling Contractions

Contractions are combinations of words that are shortened by dropping letters and replacing them with apostrophes



Handling Abbreviations

Abbreviations: Resolving abbreviations to their complete representation, enhancing text clarity and context comprehension.



Tokenization (Sentence, Word)

Sentence Tokenization

Dividing text into individual sentences.

Example:

Input -

Tokenization is a fundamental step in NLP. It breaks text into sentences.

Output –

- Tokenization is a fundamental step in NLP.
- It breaks text into sentences and then into words.

Word Tokenization

Segmenting sentences into individual words.

Example:

Input –

Tokenization is a fundamental step in NLP.

Output –

['Tokenization', 'is', 'a', 'fundamental', 'step', 'in', 'NLP', '.']

Stop Words Removal

Eliminating common and non-informative words from the text.

Stop words removal enhances the signal-to-noise ratio in text data, focusing on meaningful content.

Example –

1) When was the First computer invented?

2) How can I complete course on Natural Language Processing?

Stemming And Lemmatization

Stemming and Lemmatizations main task is to convert words to their base forms.

Problem –

- Machine Learning systems must be trained to recognize different words as different variants of one base word.
- One base word has multiple morphological variants.

Solution -
Convert words to their base forms, also known as “lemma”.

Example

Original: "Running, runs, runner"
Stemmed: "Run, run, runner"
Lemmatized: "Run, run, runner"

Stemming

Reducing words to their root or base form (stem).

Reducing words to their root or base form (stem).

Simplify variations of words by truncating prefixes or suffixes.

How it Works:

- Utilizes heuristic algorithms.
- Applies rules to remove common suffixes.

Lemmatization

Reducing words to their base or dictionary form (lemma).

- Transform words to their canonical forms for accurate analysis.
- Providing accurate base forms for more precise retrieval.

How it Works:

- Leverages linguistic rules and morphological analysis.
- Considers part-of-speech information.

Example –
Better becomes Good

N-grams (Unigrams, Bigrams, Trigrams)

N-grams preserve the sequential structure of language, aiding in context understanding and feature extraction.

Types

Unigrams: Single words as individual units.

Bigrams: Pairs of consecutive words.

Trigrams: Triplets of consecutive words.

Examples

Sentence - “I reside in Bengaluru”.

Unigram(n=1) [“I”,”reside”,”in”,”Bengaluru”]

Bigram(n=2) [“I reside”,”reside in”,”in Bengaluru”]

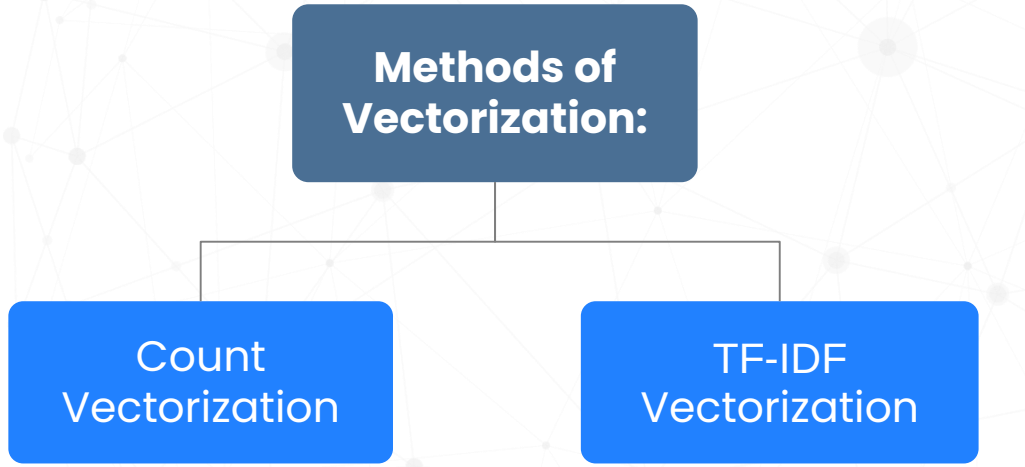
Trigram(n=3) [“I reside in”, “reside in Bengaluru”]

Vectorization of Text Data

Vectorization: Converting text data into numerical vectors for machine learning algorithms.

Machine Learning Compatibility: Vectorization enables the application of numerical algorithms to process and analyze text.

Methods



Word Embeddings

What are Word Embeddings?

- Word embeddings are numerical representations of words that capture semantic relationships and contextual similarities.

Key Concepts:

- Dense Vector Representation: Words are represented as high-dimensional vectors with continuous values.
- Semantic Similarity: Words with similar meanings have similar vector representations.
- Contextual Understanding: Captures relationships and context between words.

Bag-of-Words (BoW)

A representation of text as an unordered set of words, simplifying language for analysis.

Importance in NLP:

Simplified Representation: BoW transforms text into a numerical format, enabling easy application of machine learning algorithms.

Applications:

Text Classification
Sentiment Analysis
Information Retrieval

TF-IDF (Term Frequency-Inverse Document Frequency)

TF-IDF: A numerical statistic that reflects the importance of a term in a document relative to its occurrence across multiple documents. If the value of variance is closer to mean then it's a low variance.

Term Weighting: TF-IDF helps highlight terms that are significant to a document while downplaying common terms shared across documents.

Frequency (TF): Measures how often a term appears in a document.

Inverse Document Frequency (IDF): Measures the rarity of a term across documents.

TF – IDF Formula

Term Frequency (TF):

$$\mathrm{TF}(t, d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

Inverse Document Frequency (IDF):

$$\mathrm{IDF}(t, D) = \log\left(\frac{\text{Total number of documents in the corpus } N}{\text{Number of documents containing term } t}\right)$$

TF-IDF = TF×IDF

NLP Libraries & Tools

Explore NLP Libraries:

NLTK:

- Comprehensive library for NLP tasks.
- Robust tools for tokenization, stemming, tagging, parsing, and more.
- Large collection of datasets and resources.

spaCy:

- Designed for production use, emphasizing speed and efficiency.
- Advanced tokenization, named entity recognition (NER), and part-of-speech tagging.
- Pre-trained models for multiple languages.

TextBlob:

- Simple API for common NLP tasks.
- Built on NLTK and Pattern libraries.
- Provides sentiment analysis, part-of-speech tagging, noun phrase extraction, and more.

Word Embeddings

Introduction

Understanding Word Embeddings: Introduction to a technique that represents words as dense vectors in a continuous vector space.

What are Word Embeddings?

- Word embeddings are numerical representations of words that capture semantic relationships and contextual similarities.

Key Concepts:

- Dense Vector Representation: Words are represented as high-dimensional vectors with continuous values.
- Semantic Similarity: Words with similar meanings have similar vector representations.
- Contextual Understanding: Captures relationships and context between words.

Word2vec

- Understanding the Skip-Gram and Continuous Bag of Words (CBOW) models for word embeddings.
- **Word2vec –** Neural network model that learns word embeddings by predicting context words based on target words

Skip-Gram

- Objective: Predict context words given a target word.
- Training: Adjusts word vectors to maximize the probability of predicting context words.
- successful in capturing semantic relationships between words, such as word analogies (e.g, “king” – “man” + “woman” ≈ “queen”)

Continuous Bag of Words

- Objective: Predict target word given context words.
- Training: Adjusts word vectors to maximize the probability of predicting the target word.
- Example – “She is a great dancer.” is converted to -> ([she, a], is), ([is, great], a) ([a, dancer], great)

GloVe

GloVe - model is mainly based on capturing vector statistics in global context.

Glove captures both global statistics and local statistics for generating the embeddings.

The way GloVe predicts surrounding words is by maximizing the probability of a context word occurring given a centre word by performing a dynamic logistic regression.

Key Concepts:

1. **Global Context:** Considers the entire corpus for word vectorization.

2. **Word Co-occurrence Matrix:** Captures how often words co-occur in the same context.

3. **Vector Arithmetic:** Enables algebraic operations on word vectors to reveal semantic relationships.

Steps in Glove

Build Co-occurrence Matrix:

- Construct a matrix where each cell represents the number of times word *i* appears in the context of word *j* across the entire corpus.

Compute Word Probabilities:

- Normalize the co-occurrence matrix to obtain conditional probabilities of word co-occurrences.

Initialize Word Vectors:

- Initialize word vectors for each word in the vocabulary.

Optimize Word Vectors:

- Minimize the difference between predicted & actual word co-occurrence probabilities.
- Adjust word vectors iteratively to improve the model's fit to the co-occurrence statistics.

Generate Final Word Embeddings:

- Once optimization converges, obtain final word embeddings where each word is represented as a dense vectors

FastText Embeddings

FastText is an extension of Word2Vec that incorporates sub-word information, allowing it to represent words as a combination of character n-grams.

Steps

Construct Character n-grams: Break words into overlapping character n-grams.

Learn Word Embeddings: Train embeddings for words and sub-word units simultaneously.

Combine Embeddings: Aggregate sub-word embeddings to obtain word embeddings.

Key Concepts

1. **Sub-word Information:** Captures morphological and compositional aspects of words.

2. **Character n-grams:** Divides words into smaller units (n-grams) to handle out-of-vocabulary words.

Neural Networks in NLP

Neural Nets & Types

AI-powered systems designed to understand and respond to user queries with precise information.

Why Neural Networks in NLP?

- Neural Networks are actually really good in understanding and learning anything.
- By just considering examples , Neural Networks can perform any kind of task.

Types of Neural Networks

- RNN – Recurrent Neural Networks.
- LSTM – Long Short Term Memory
- GRU – Gated Recurrent Unit
- MLP – Multi-Layer Perceptron

Why Recurrent for NLP?

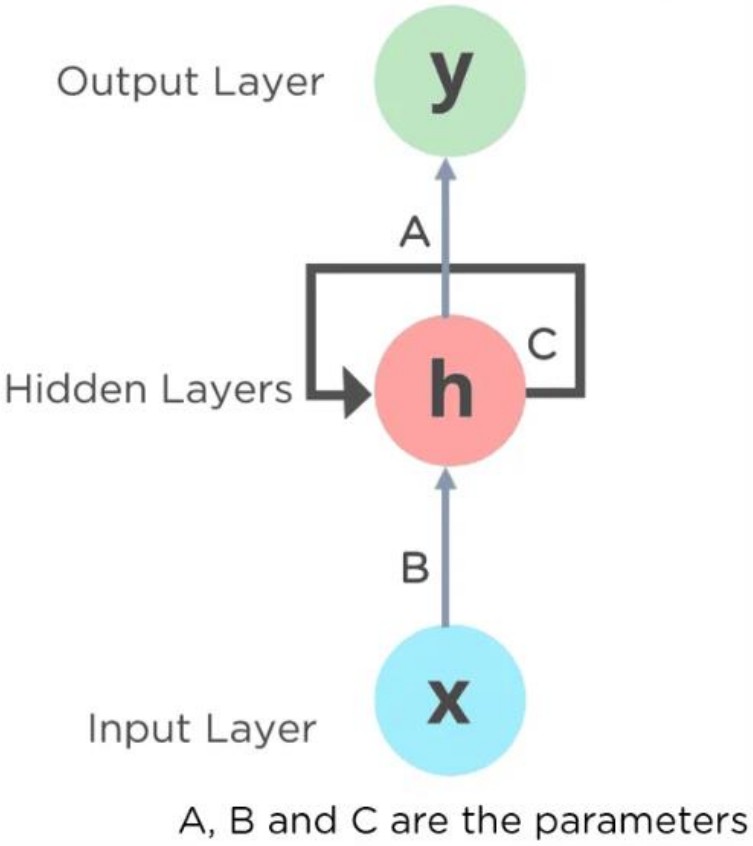
RNN are really useful in NLP tasks like QnA , Text Completion and Classification.

No Memory!

The Major Problem

The central loophole in neural networks is that it does not have memory. Since no memory is associated, it becomes very difficult to work on sequential data like text

Recurrent Neural Networks



RNN

RNN addresses the memory issue by giving a feedback mechanism that looks back to the previous output and serves as a kind of memory.

Why LSTM when we have RNN?

Understanding of previous tokens in LSTM is far better than traditional RNN.

LSTM

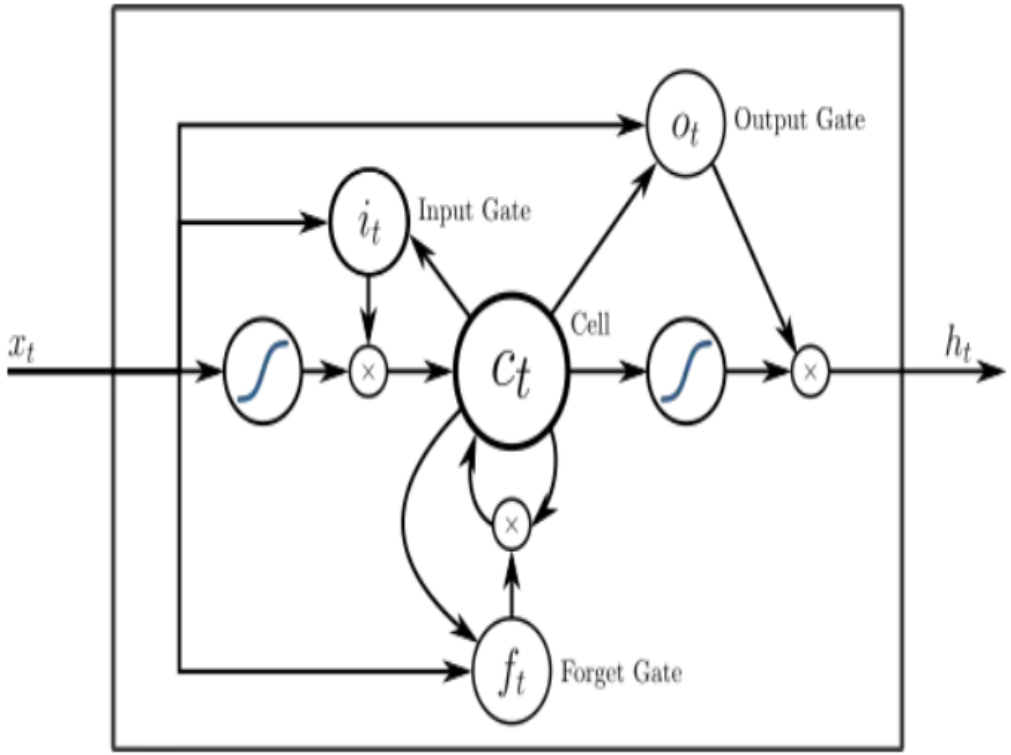
- A sentence or phrase only holds meaning when every word in it is associated with its previous word and the next one.
- LSTM, short for Long Short Term Memory extends it by creating both short-term and long-term memory components to efficiently study and learn sequential data.

LSTM – Long Short Term Memory

- It excels at capturing long-term dependencies, making it ideal for sequence prediction tasks.
- **LSTM** incorporates feedback connections, allowing it to process entire sequences of data, not just individual data points.

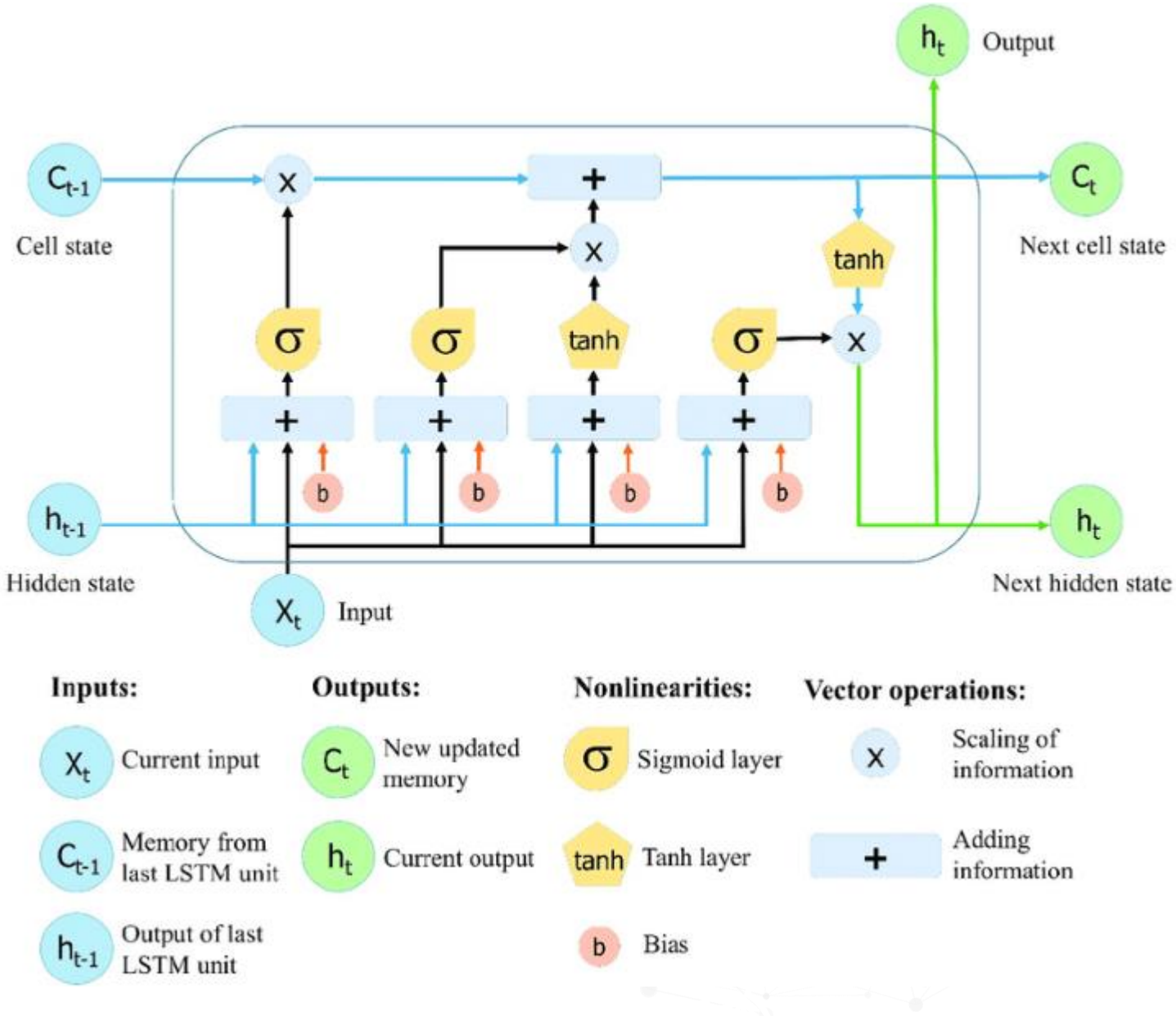
Advantages

- Actively selects and rejects tokens.
- Robust Gated Architecture System
- It resolves the vanishing gradient problem faced by RNN



Architecture

- Input Gate- quantify the importance of the new information
- Forget Gate - responsible for deciding which information is kept
- Output Gate- decides what the next hidden state should be



GRU-Gated Recurrent Unit

GRU are a simpler alternative to Long Short-Term Memory (LSTM) networks.

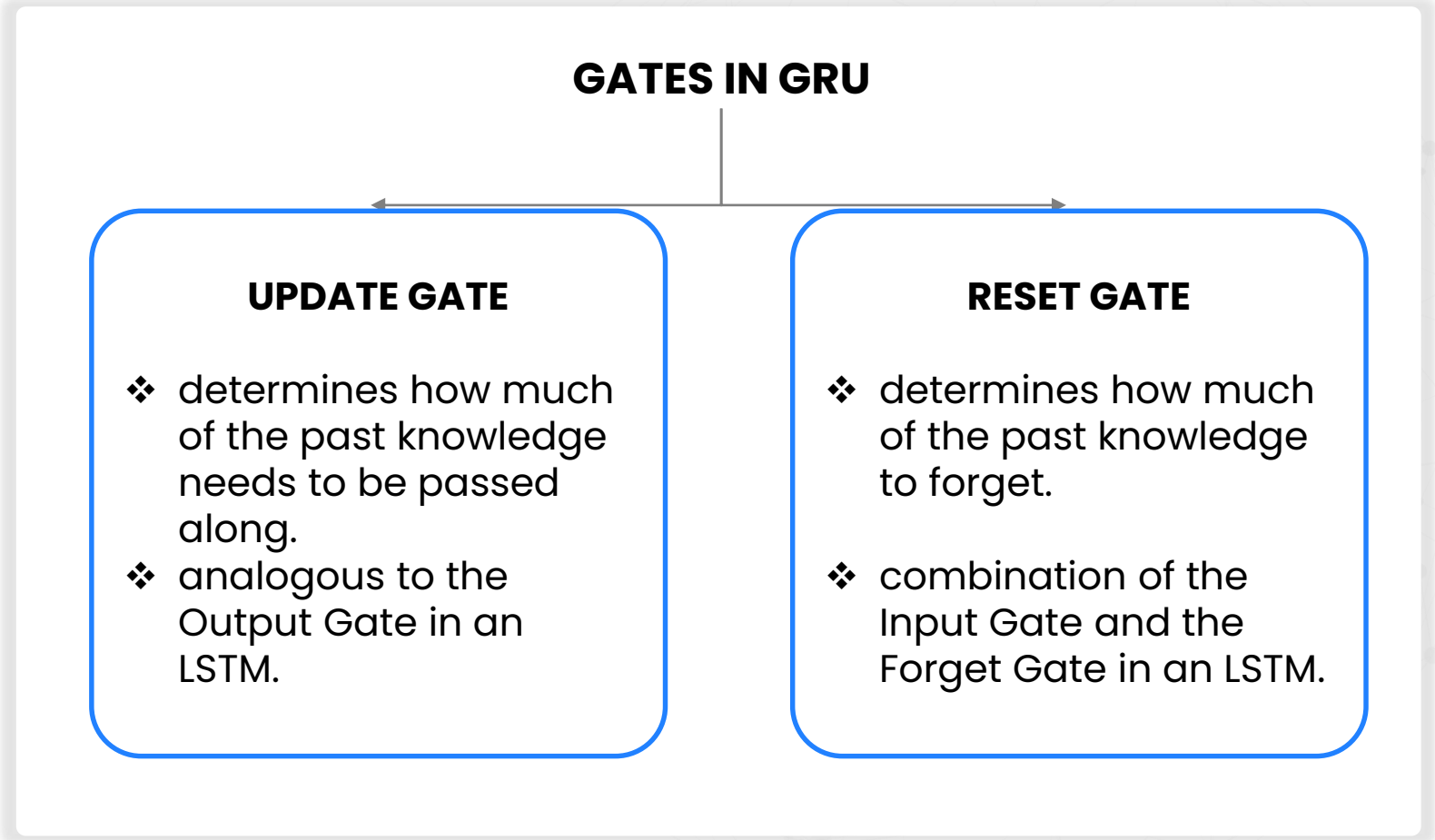
❖ GRU

- ❖ Like LSTM, GRU can process sequential data such as text, speech, and time-series data.
- ❖ The basic idea behind GRU is to use gating mechanisms to selectively update the hidden state of the network at each time step.
- ❖ The gating mechanisms are used to control the flow of information in and out of the network.

GRU vs LSTM

01 GRU have a simpler Architecture and thus have less training time.

02 GRU have only 2 Gate system meanwhile LSTM have 3 of them.



Architecture

Input Layer - takes in sequential data

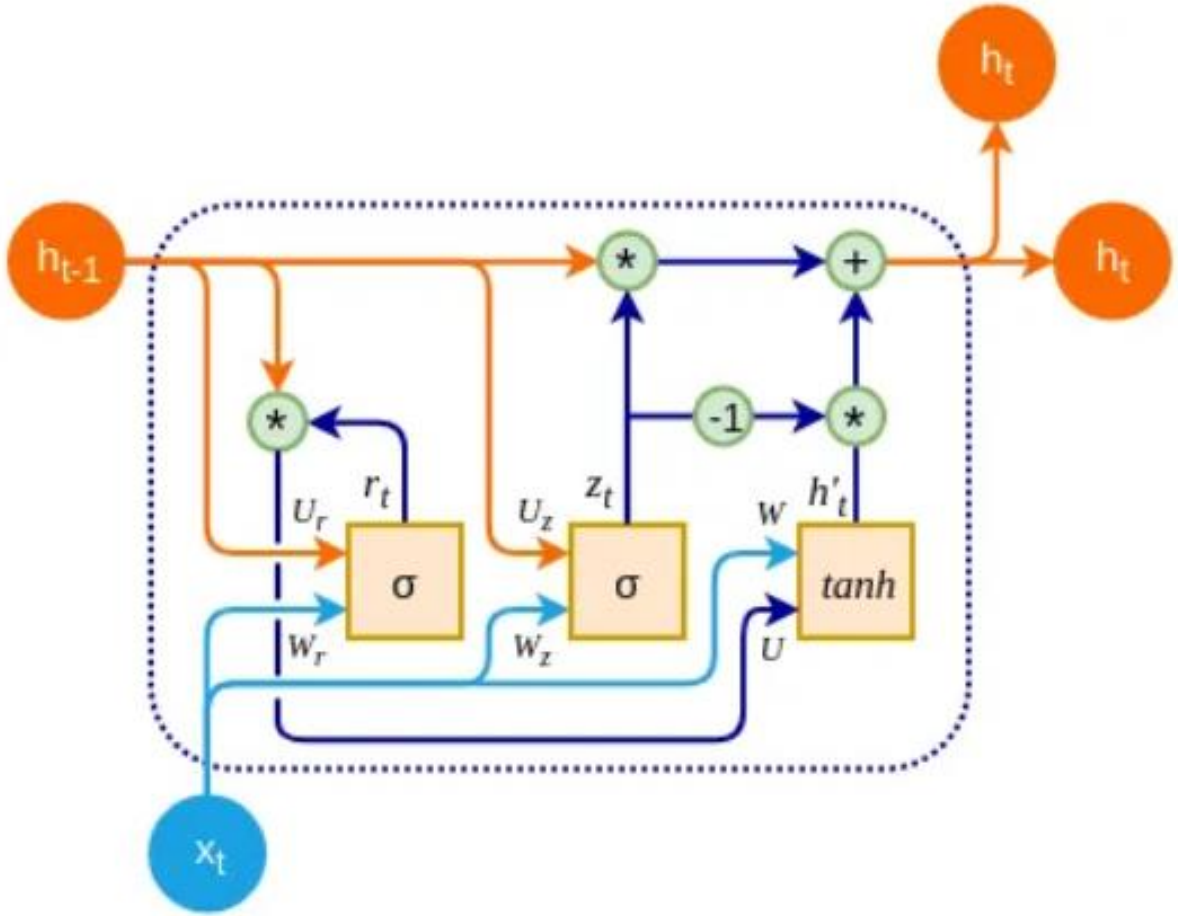
Hidden Layer - where the recurrent computation occurs

Reset Gate - determines how much of the previous hidden state to forget

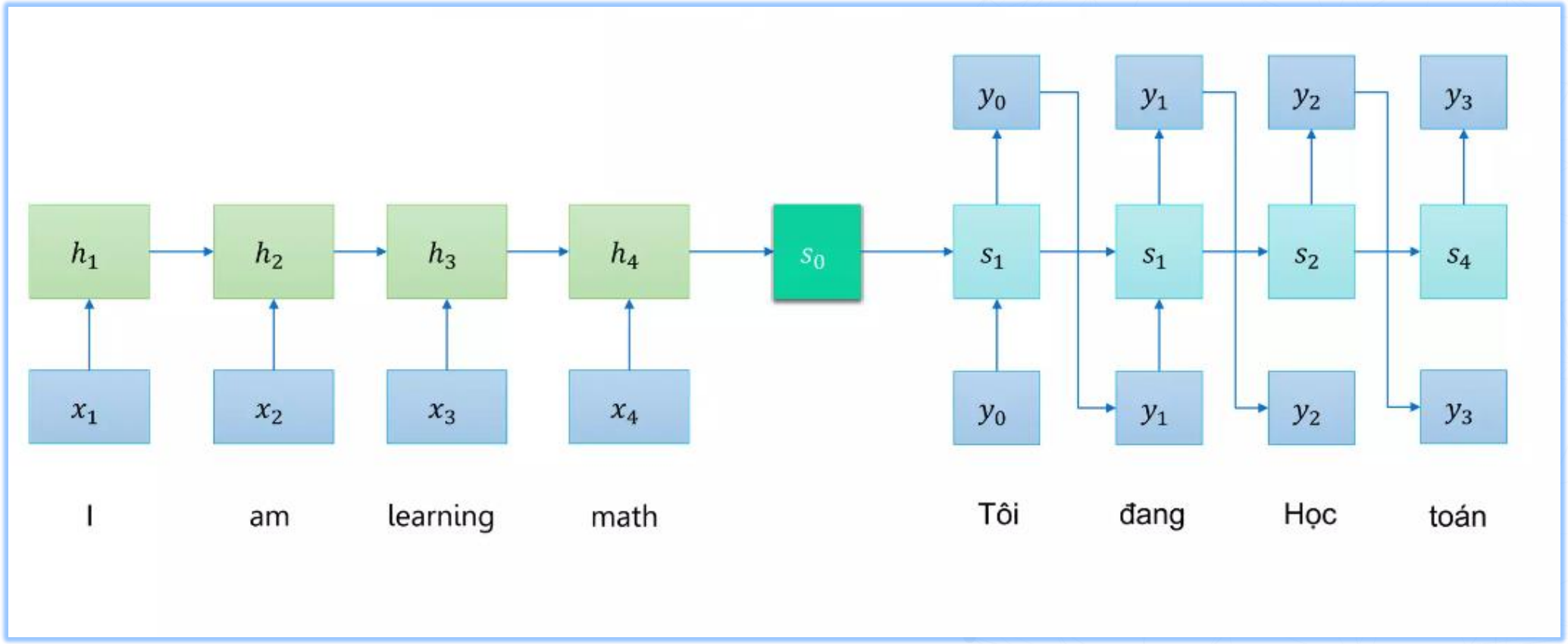
Update Gate - determines how much of the candidate activation vector to incorporate

Candidate Activation Vector - modified version of the previous hidden state

Output Layer - network's output



Traditional Encoder Decoder Model



Problems with the ED Model

The traditional ED Model was not sufficient and betterment was required.

Encoder Decoder Model

- Long Sentence Problem
- Input Output alignment Problem

.

Attention Mechanisms

The attention mechanism has changed the way we work with deep learning algorithms

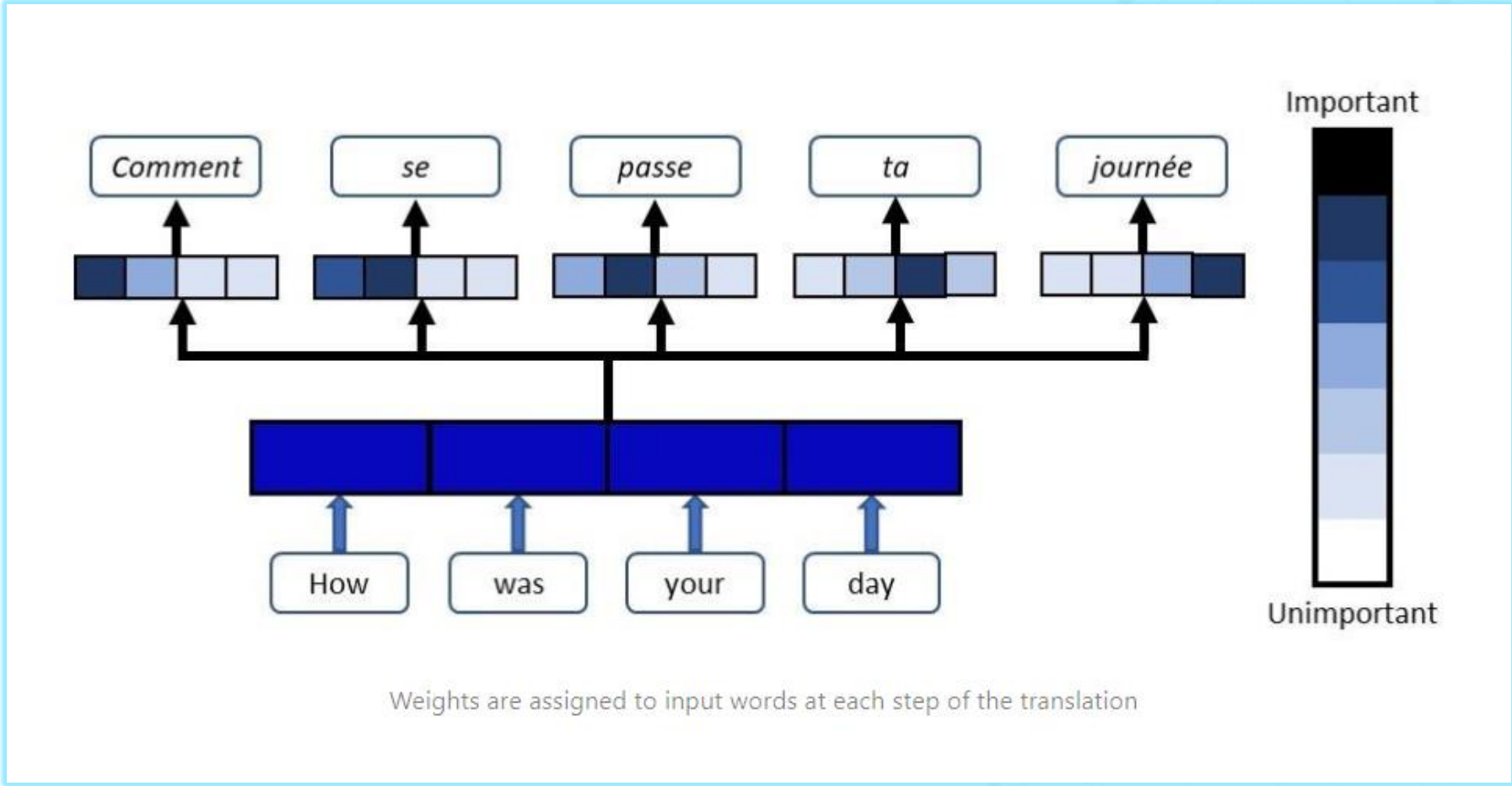
What is Attention?

- Attention is the cognitive process of selectively concentrating on one or a few things while ignoring others.
- Selectively focusing on important input elements , increasing accuracy.

Why is it better?

- Allows the Decoder to access the entire encoded input.
- Prioritizes attention weights to give more weightage to relevant inputs.

Attention Mechanisms Architecture



Weights are assigned to input words at each step of the translation

Transformers

The Transformer architecture follows an encoder-decoder structure but does not rely on recurrence and convolutions in order to generate an output.

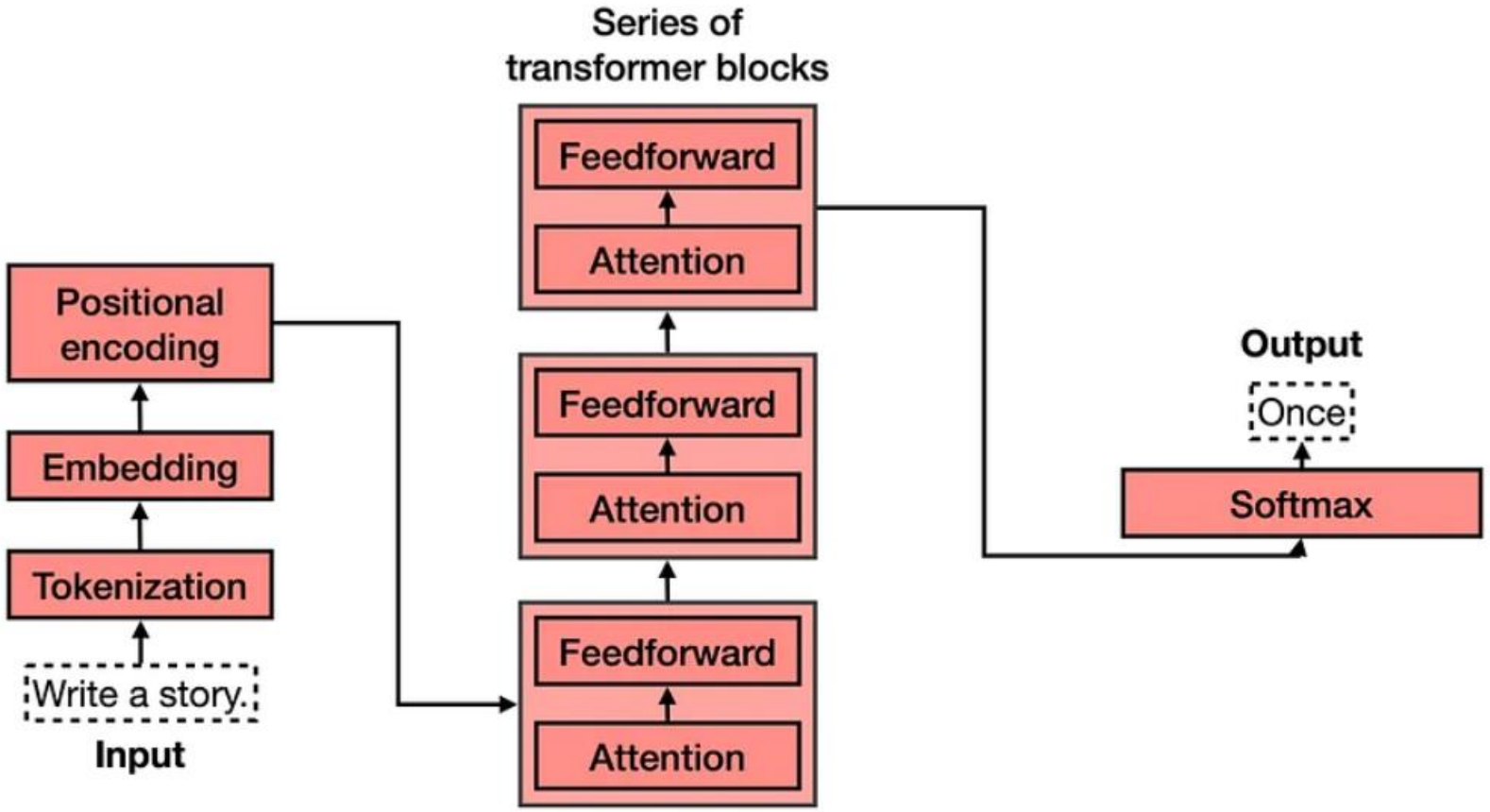
Transformers in a nutshell?

- In a nutshell, the task of the encoder, on the left half of the Transformer architecture, is to map an input sequence to a sequence of continuous representations, which is then fed into a decoder.
- The decoder, on the right half of the architecture, receives the output of the encoder together with the decoder output at the previous time step to generate an output sequence.

Why is it better?

- Self-attention layers were found to be faster than recurrent layers
- understand the relationships among sequential elements that are far from each other.

Transformer Mechanisms Architecture

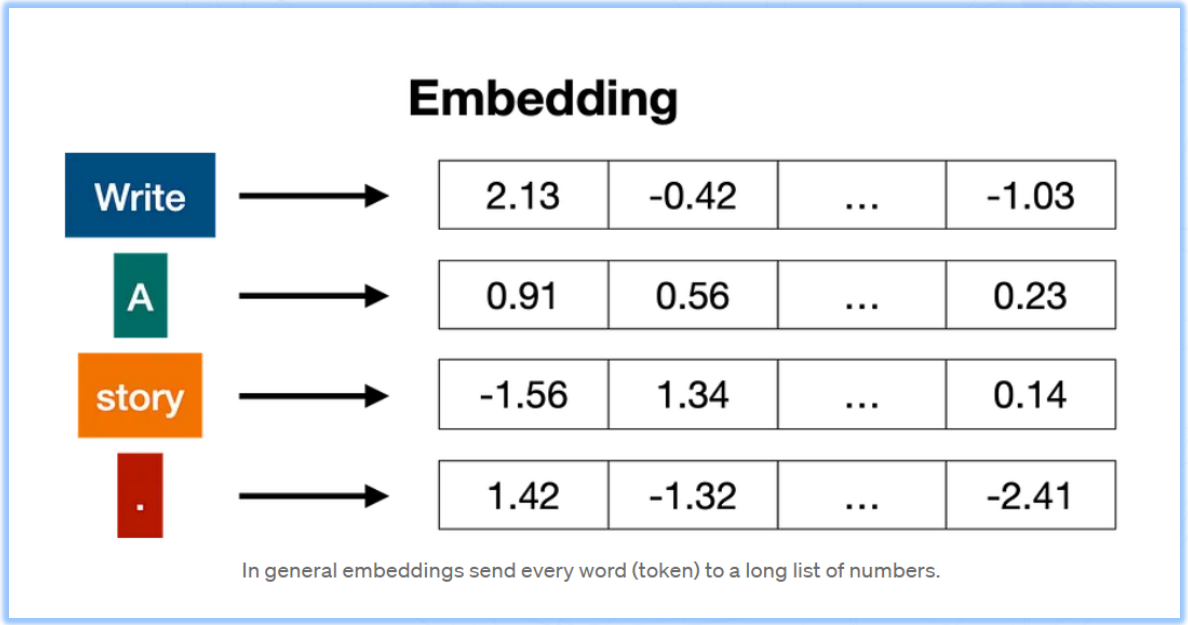


The architecture of a transformer model

Figuring out the Transformer Architecture step by step.

Tokenization - large dataset of tokens, including all the words

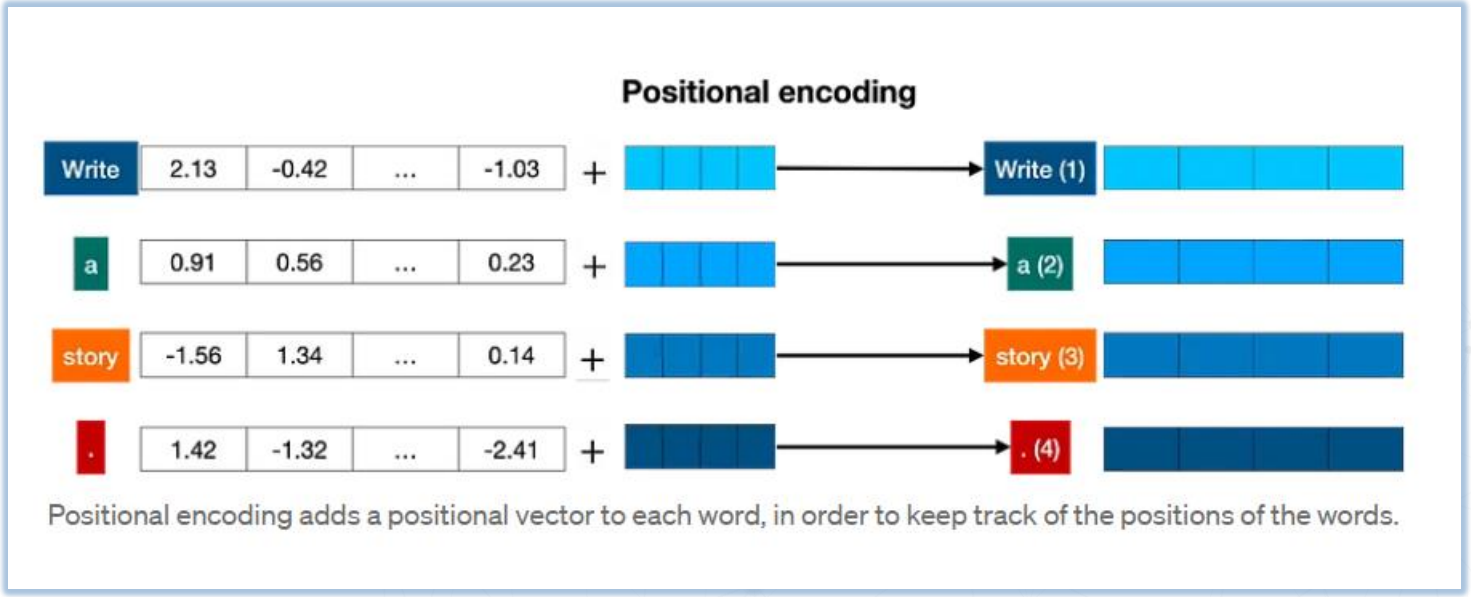
Embeddings - turn words into numbers.



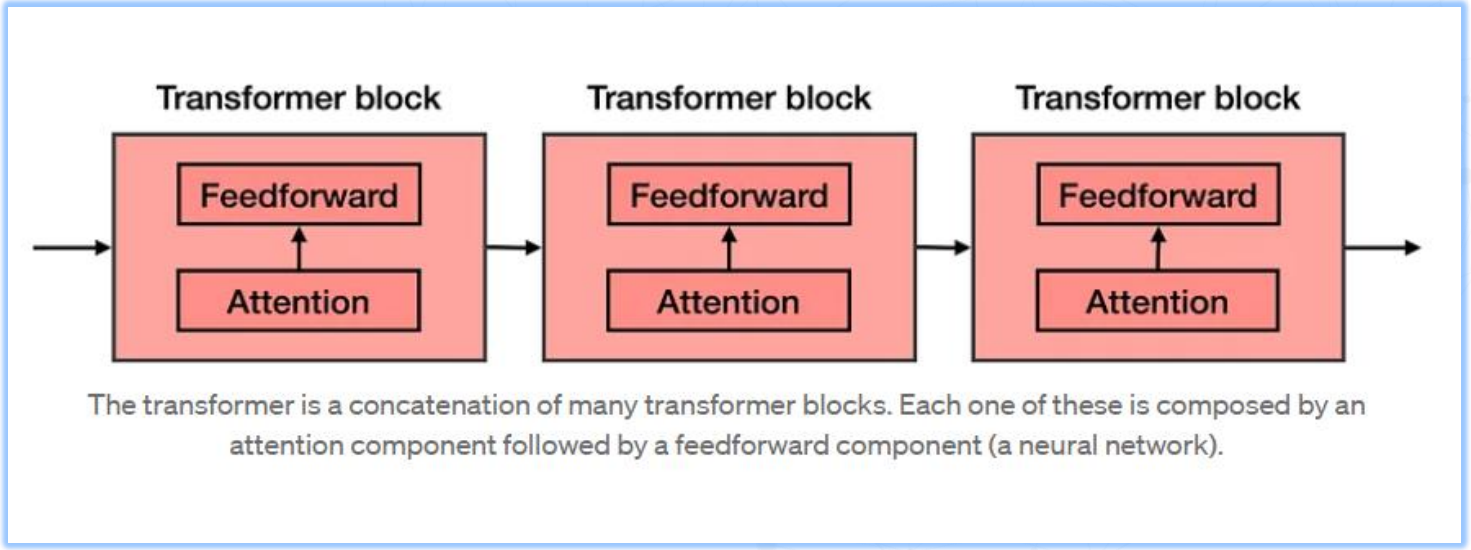
Figuring out the Transformer Architecture step by step.

Positional Encoding – Turning vectors into a process, making a single one from a bunch.

Transformer Block – A very large Neural Network for generation.



Positional encoding adds a positional vector to each word, in order to keep track of the positions of the words.

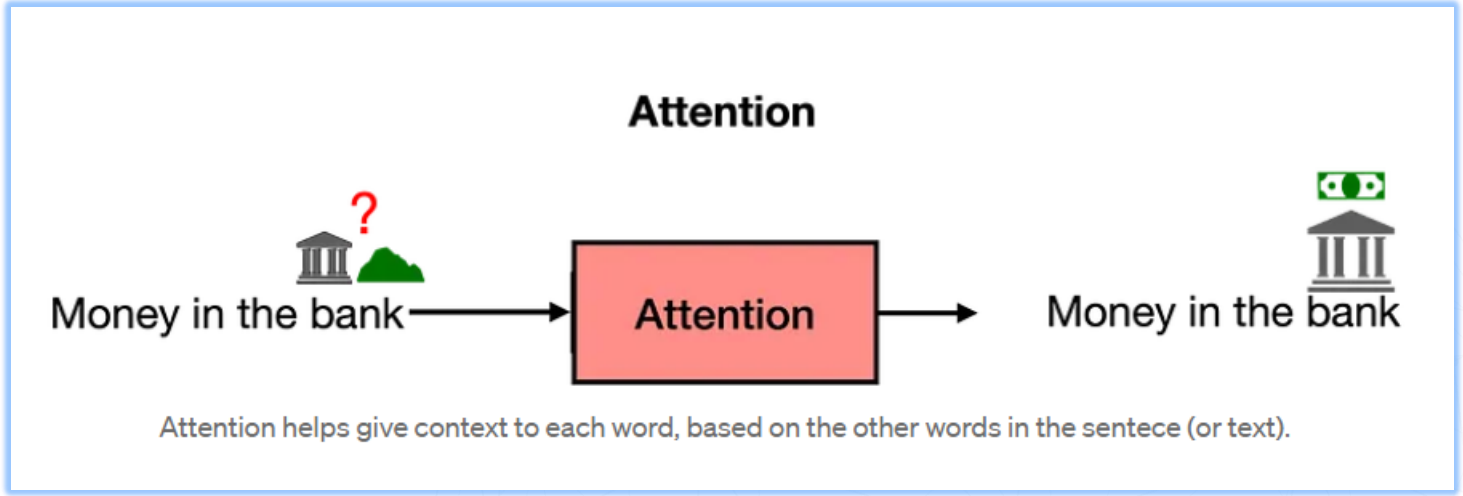


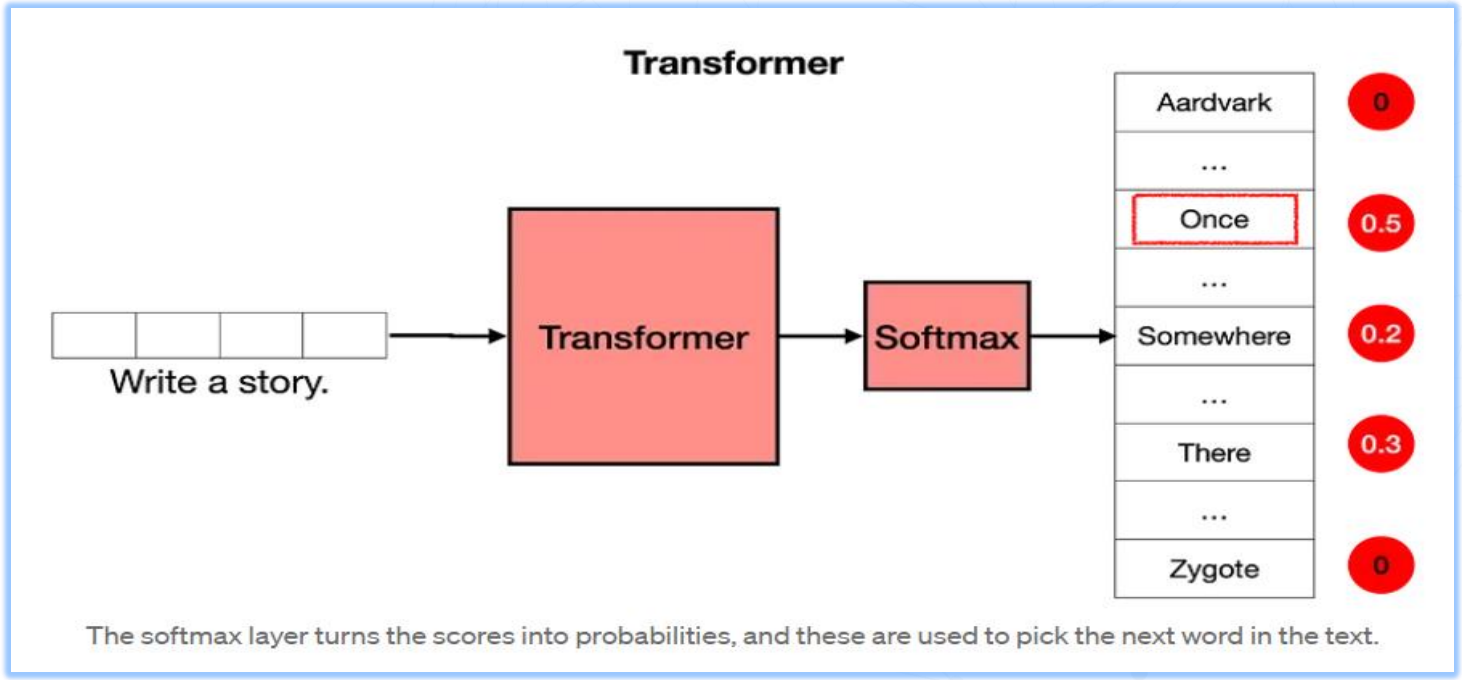
The transformer is a concatenation of many transformer blocks. Each one of these is composed by an attention component followed by a feedforward component (a neural network).

Figuring out the Transformer Architecture step by step.

Attention Mechanism– solving the problem of context.

SoftMax– Turns scores to probabilities.





Bi-Directional RNN

They are artificial neural networks that process input data in both the forward and backward directions.

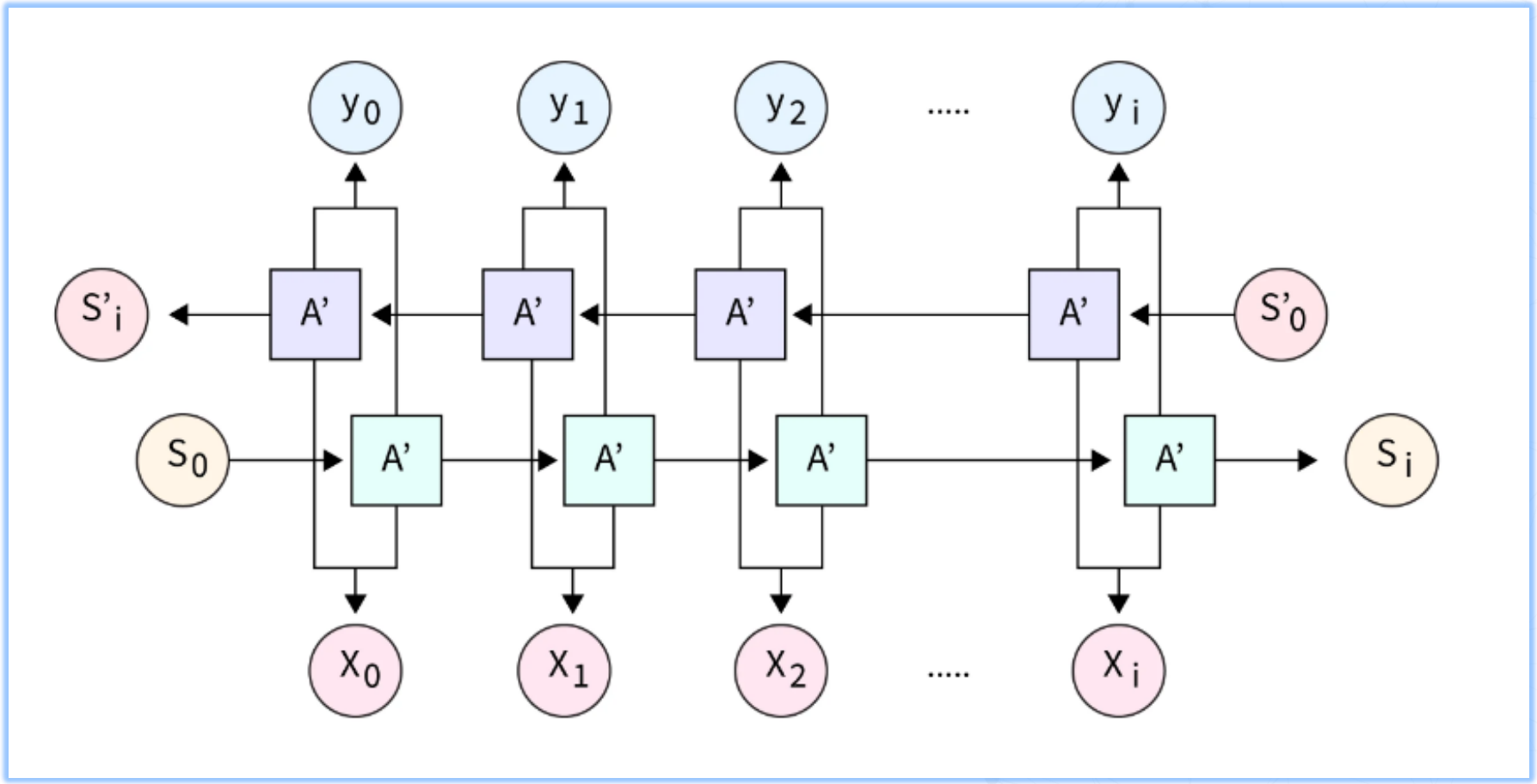
What are they?

- Capture the contextual dependencies in the input data by processing it in both directions
- Uses two separate RNNs: one processes the data in the forward direction, while the other processes it in the reverse direction.

Why is it better?

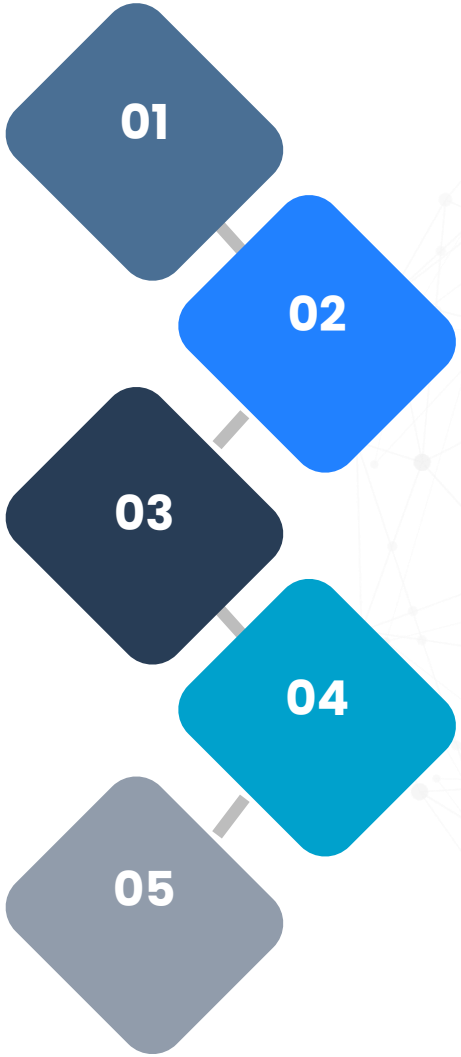
- Better than traditional RNN's which just use previous knowledge.
- To perform well on natural language tasks, the model must be able to process the sequence in both directions.

Bi-Directional RNN Architecture



Merge Modes in Bi-directional RNN

The output of these two RNNs is then combined, or "merged," in some way to produce the final output of the model.



Concatenation: In this mode, the outputs of the forward and backward RNNs are concatenated together, resulting in a single output tensor that is twice as long as the original input.

Sum: In this mode, the outputs of the forward and backward RNNs are added together element-wise, resulting in a single output tensor that has the same shape as the original input.

Average: In this mode, the outputs of the forward and backward RNNs are averaged element-wise, resulting in a single output tensor that has the same shape as the original input.

Maximum: In this mode, the maximum value of the forward and backward outputs is taken at each time step, resulting in a single output tensor with the same shape as the original input.

BERT

BERT language model is an open source machine learning framework for natural language processing (NLP).

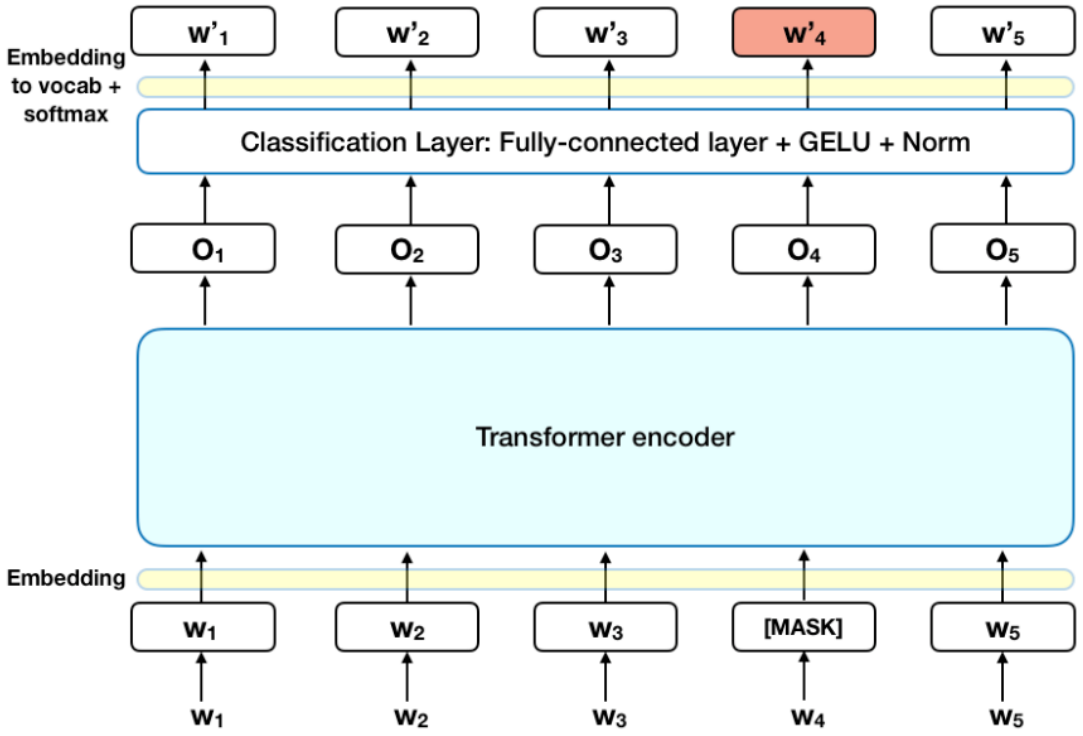
Why BERT?

- BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on transformers
- BERT is designed to help computers understand the meaning of ambiguous language in text by using surrounding text to establish context.

How?

- BERT predicts a word in a blank, for this typically train using a large repository of specialized, labeled training data
- BERT uses an MLM method to keep the word in focus from seeing itself, or having a fixed meaning independent of its context.

BERT Flowchart



Masked Language Modeling

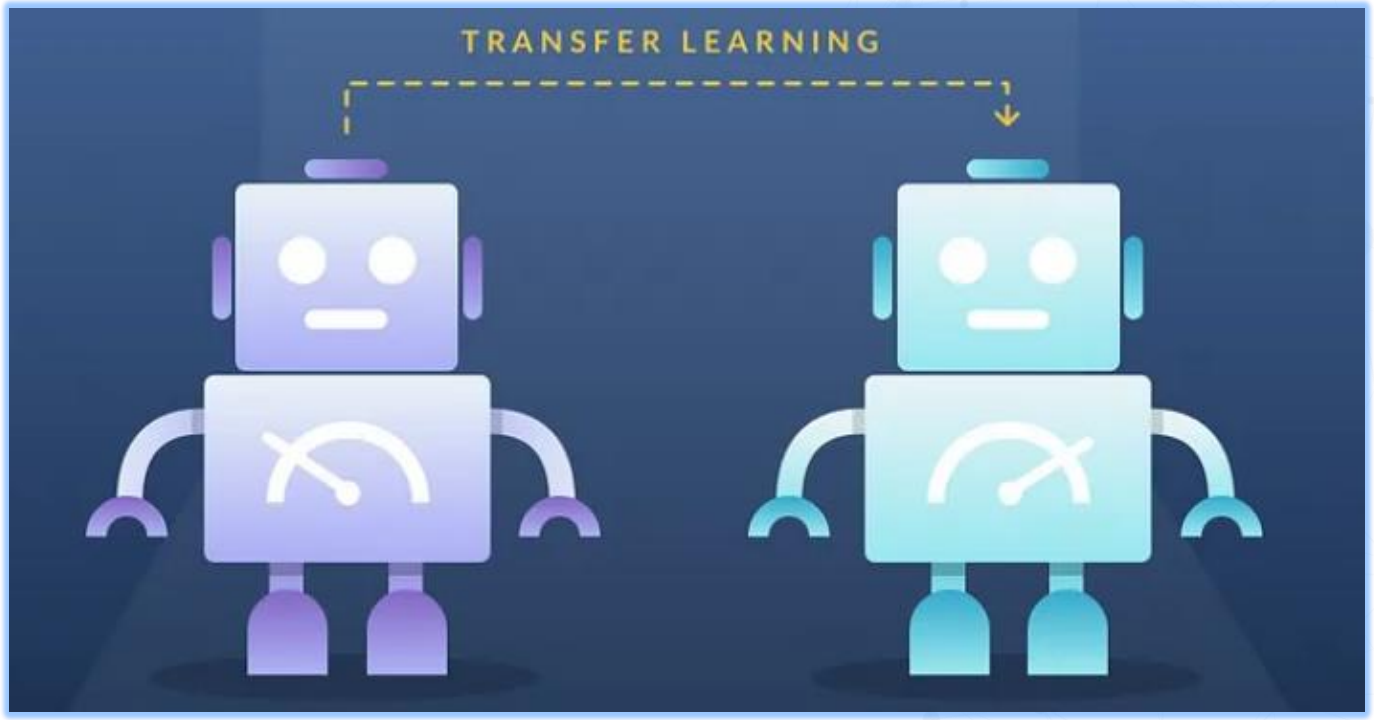
- Before passing the words to BERT the model uses masks on 15% words the model then tries to predict original value of the masked words. [Embedding to vocab+softmax]
- BERT loss function takes into account only the prediction of the masked values and ignores prediction of non masked values

Next Sentence Prediction
In BERT model takes multiple sentences and is trained to predict the next sentence. During training 50% words are first sentence and 50% are second sentences.

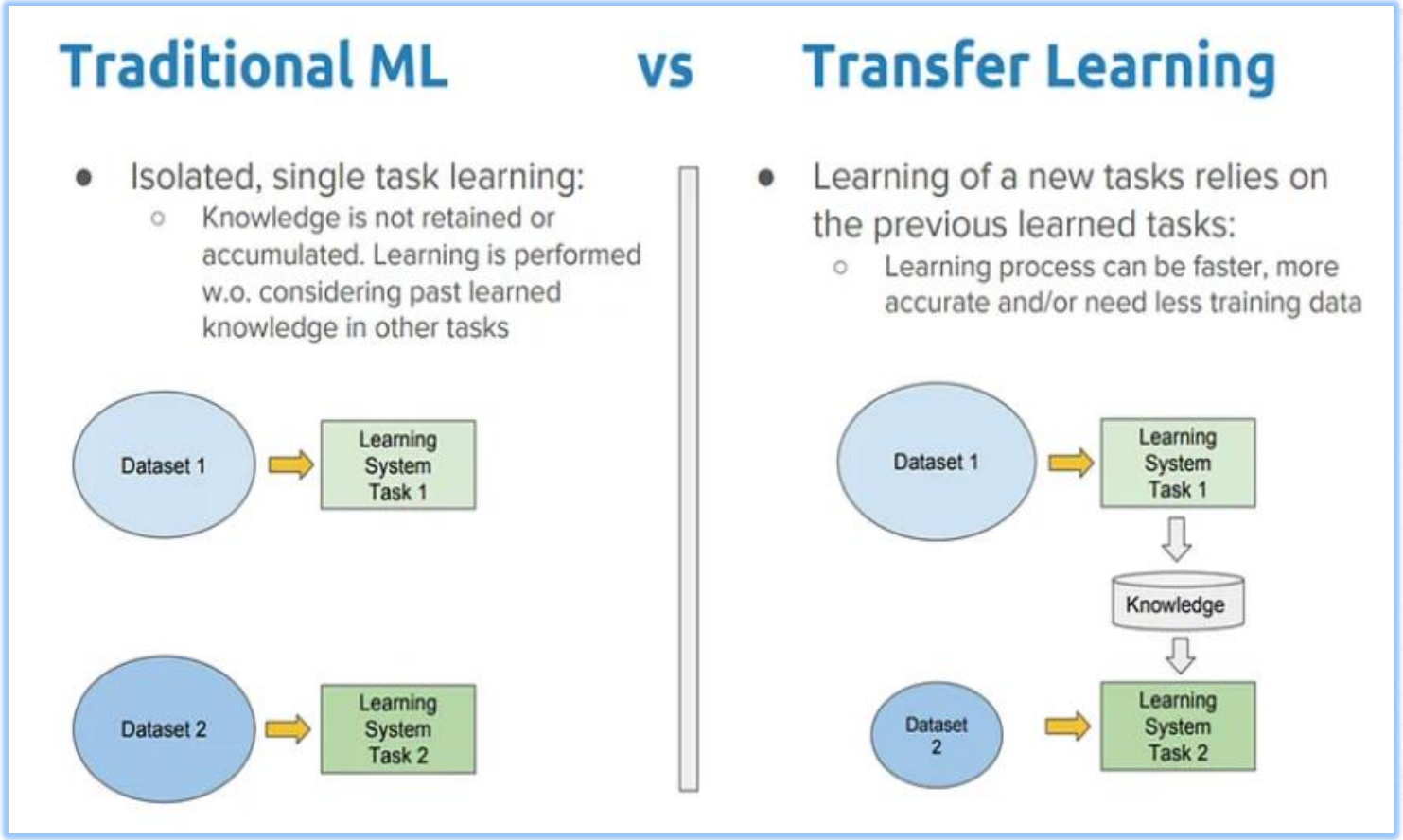
Finetuning –
BERT can be finetuned for multiple applications for example Question Answering , We can provide question and answers and train the model for giving similar outputs

Transfer Learning

Transfer Learning will be the next driver of Machine Learning success. — Andrew Ng



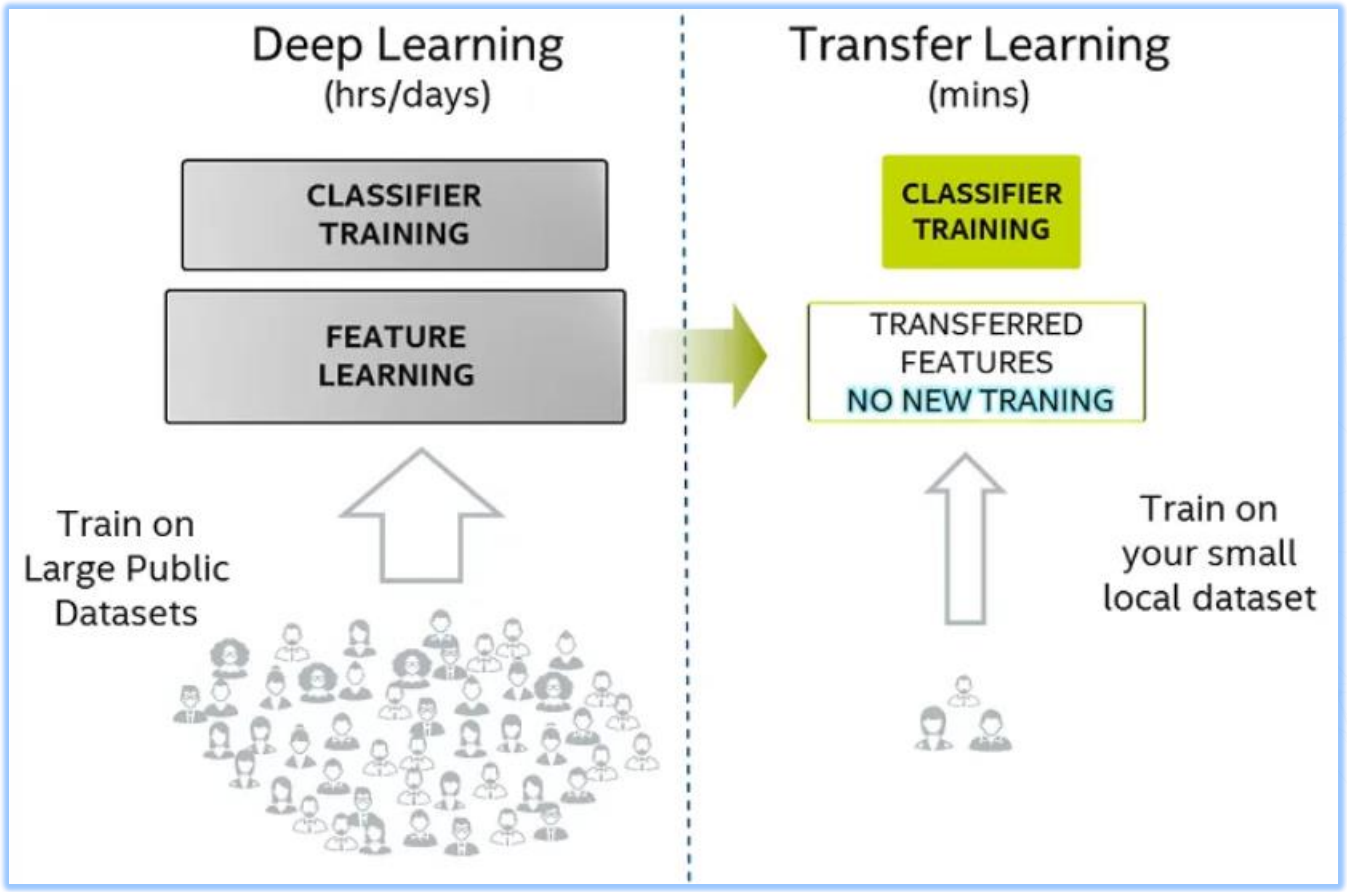
Transfer Learning



What is Transfer Learning?

- Instead of starting the learning process from an (often randomly initialized) **blank sheet**, you start from patterns that have been learned to solve a different task.
- The ability to learn from a **large number of experiences**, and exporting 'knowledge' into new environments is exactly what transfer learning is all about.

Transfer Learning



Why is it better?

- Computation Time is reduced exponentially.
- Caring about the Environment- Model training can take months which increases the carbon footprints.

Probabilistic Language Models

Probabilistic Models are the Key to advanced Natural Language Processing.

What are they?

- They attempt to predict the next word by capturing patterns from a given sequence of words using a statistical approach.
- Key goal is to calculate the probability of a given sequence of words , and assign that probability value to the sequence.

Types of models

- Bi-gram Models & Tri-gram Models used for predicting next 2-3 words.
- N-gram Models -To predict n from n-1 given words

Probabilistic Language Models

$$P(w_1 w_2 \ldots w_n) = \prod_i P(w_i | w_1 w_2 \ldots w_{i-1})$$

P(I love dogs) = P(I)P(love | I)P(dogs | I love)

Now the individual probabilities can be obtained in the following way :

P(I) = Count('I') / Total no. of words

P(love | I) = Count('I love') / Count('I')

P(dogs | I love) = Count('I love dogs') / Count('I love')

How do they work?

- The chain rule is used to compute the probability of a sentence in a language model. Let w1w2...wn be a sentence where w1, w2, wn are the individual words. Then the probability of the sentence occurring is given by the following formula.

Hidden Markov Models (HMMs)

HMM's are the Secret Sauce in Natural Language Processing

What are they?

- A statistical model called a Hidden Markov Model (HMM) is used to describe systems with changing unobservable states over time.
- It is used to predict future observations or classify sequences, based on the underlying hidden process that generates the data.

Why are they better?

- They have high Flexibility and great for forecasting tasks.
- HMMs use a state-based representation that can capture hidden patterns or latent structures within the data.

Architecture

Step 1: Define the state space and observation space.

Step 2: Define the initial state distribution.

Step 3: Define the state transition probabilities.

Step 4: Define the observation likelihoods.

Step 5: Train the model.

Step 6: Decode the most likely sequence of hidden states.

Step 7: Evaluate the model.

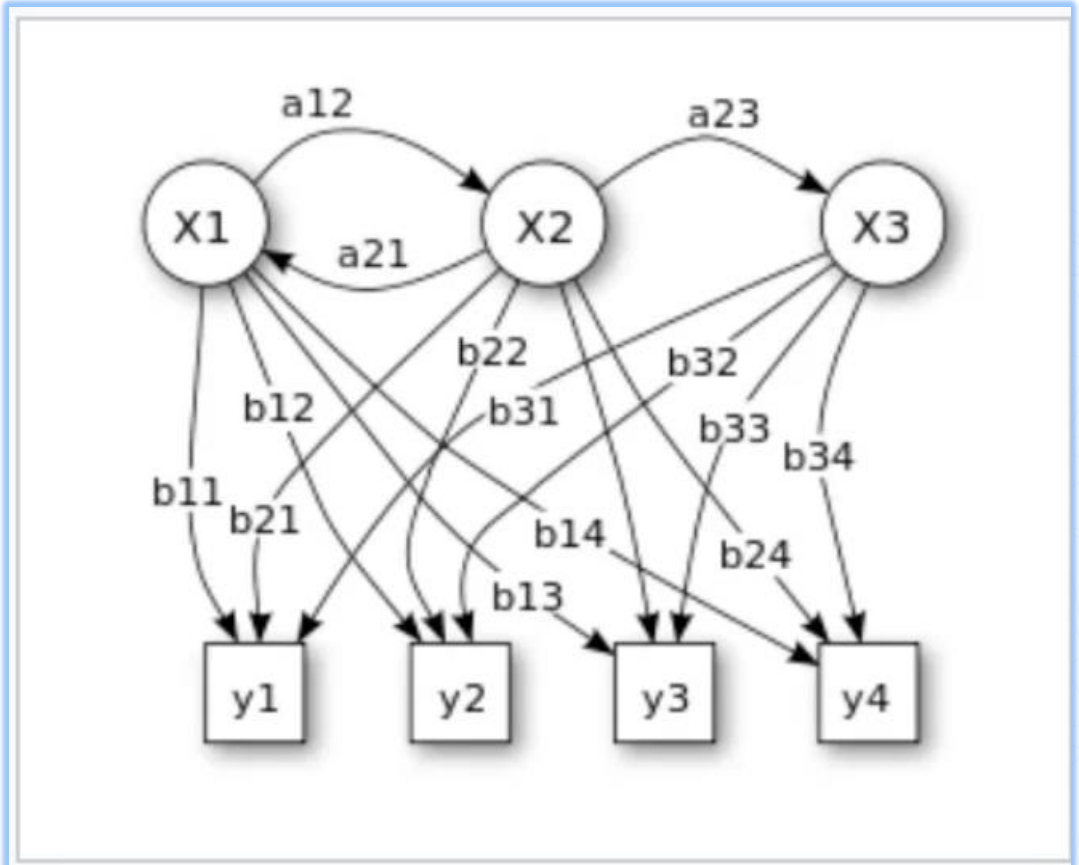


Figure 1. Probabilistic parameters of a hidden Markov model (example)

X — states

y — possible observations

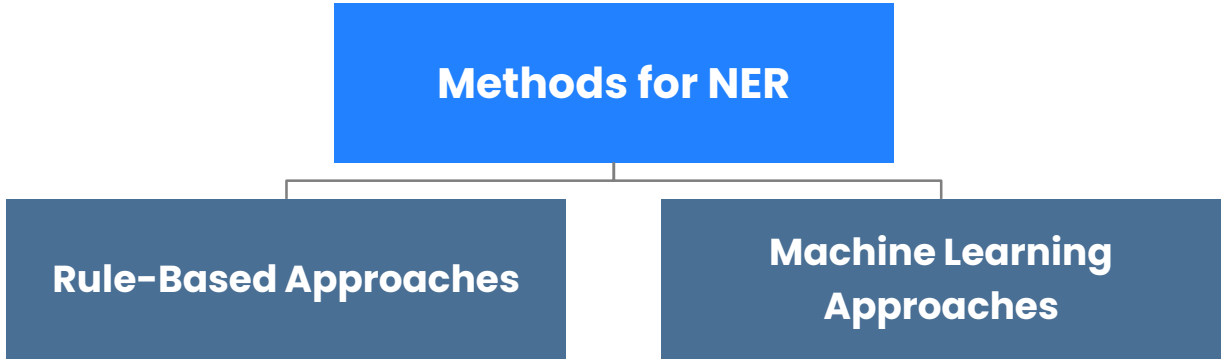
a — state transition probabilities

b — output probabilities

Advanced NLP Techniques

Named Entity Recognition (NER)

Named Entity Recognition (NER) is the process of identifying and classifying entities such as persons, organizations, locations, dates, and more in text.



- Use predefined rules and patterns for entity identification.
- Train models on labeled data to predict entity types.

Key Concepts

1. **Entity Types**: Classification into predefined categories like person, organization, location, date, etc.
2. **Context Awareness**: Recognition of entities based on their context within sentences.

Applications

1. **Information Extraction**: Extracting structured information from unstructured text.
2. **Question Answering Systems**: Enhancing the understanding of queries and generating accurate responses.

NER Example

"Apple Inc. was founded by Steve Jobs, Steve Wozniak, and Ronald Wayne on April 1, 1976, in Cupertino, California. The company is known for its innovative products, including the iPhone, iPad, and MacBook."

Identified Entities

[Apple Inc.] was founded by [Steve Jobs], [Steve Wozniak], and [Ronald Wayne] on [April 1, 1976], in [Cupertino], [California]. The company is known for its innovative products, including the [iPhone], [iPad], and [MacBook].

Names	(Apple Inc.)
Persons	Steve Jobs, Steve Wozniak, Ronald Wayne
dates	April 1, 1976
locations	Cupertino, California

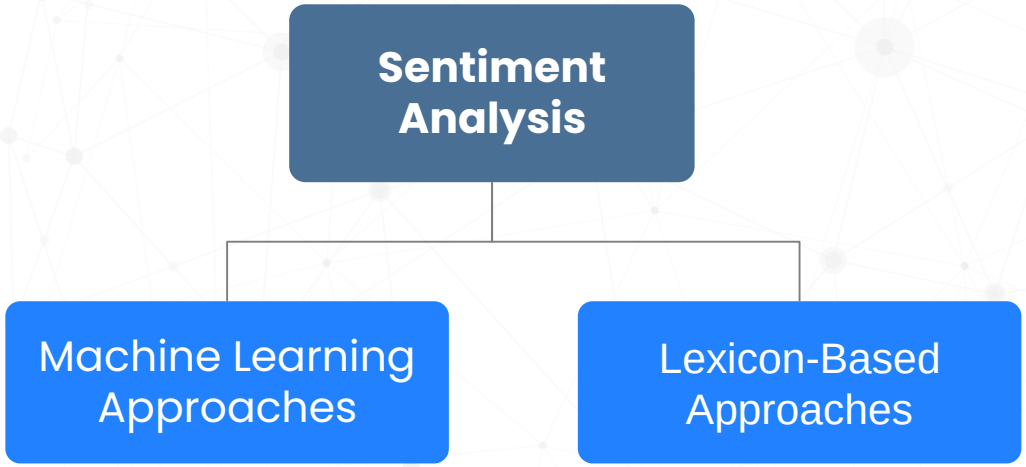
Sentiment Analysis

Sentiment Analysis, is the process of determining the sentiment or emotion expressed in a piece of text, such as positive, negative, or neutral.

Applications:

1. *Social Media Monitoring*
2. *Customer Feedback Analysis*
3. *Brand Monitoring*

Methods



Sentiment Analysis Example

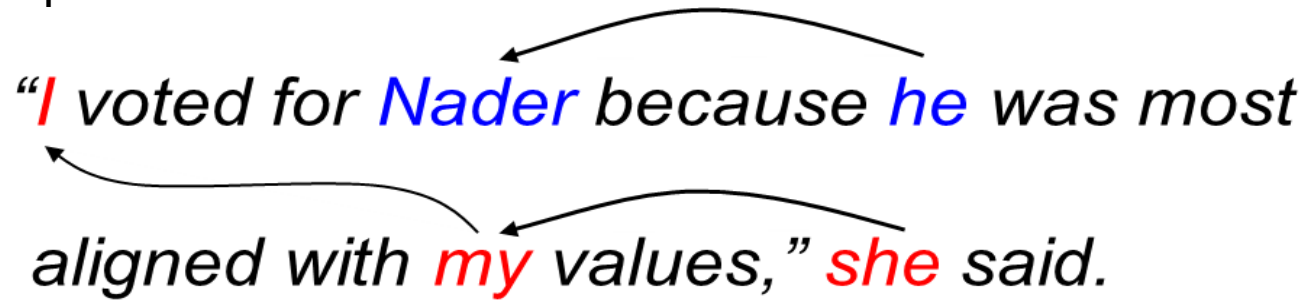
Text	Sentiment
The movie was absolutely fantastic, and I enjoyed every moment of it.	Positive
Really frustrated with the poor performance.	Negative
Overall, it was an okay experience.	Neutral

Coreference Resolution

- **Understanding Coreference Resolution:** Introduction to the NLP task of identifying and connecting expressions that refer to the same entity in text.

- Coreference Resolution is the process of determining when two or more expressions in a text refer to the same entity.
- It resolves references like pronouns (he, she, it) to their corresponding entities.

Example -



Relation Extraction

Relation Extraction is the process of identifying and classifying semantic relationships between entities mentioned in a text.

It aims to discover how entities are connected or associated with each other.

Key Concepts:

1. **Entity Pairs**: Identification of pairs of entities that exhibit a specific relationship.
2. **Relation Types**: Classification of relationships into predefined categories (e.g., "is-a," "works-for," "located-in").

Methods for Relation Extraction:

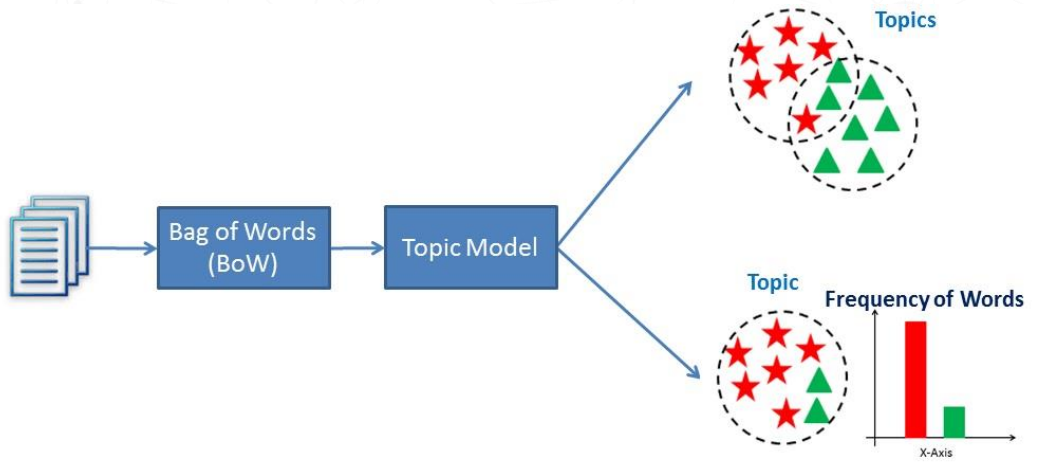
1. **Supervised Learning:** Train models on labeled data to predict relationship types.
2. **Distant Supervision:** Use existing knowledge bases to automatically label data for training.

Topic Modelling

- Topic Modelling is a frequently used approach to detect topics in an unstructured data
- It's a type of unsupervised learning technique that analyses and identifies clusters

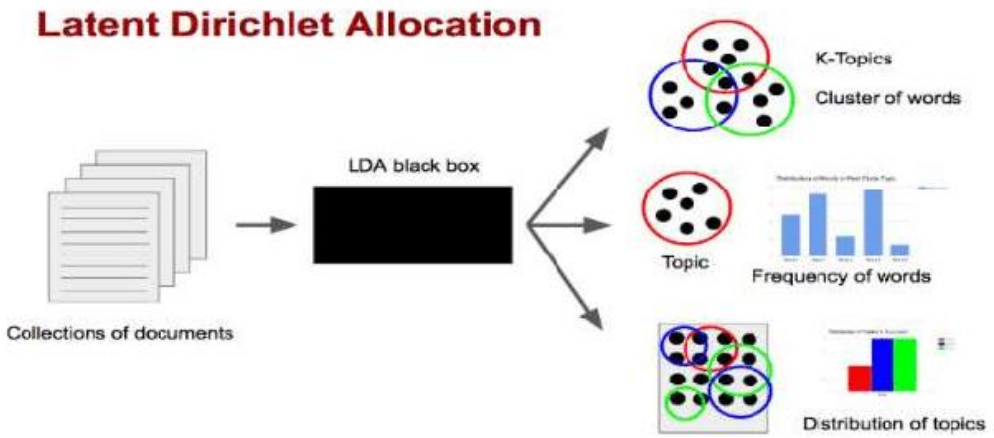
- Documents regarding a specific topic are more likely to produce certain words more frequently.
- Essentially, topic models work by deducing words and grouping similar ones into topics to create topic clusters.

Diagram



Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) is a machine learning algorithm that groups words and documents into topics. LDA is a generative model that can be used for topic extraction.



LDA makes two assumptions:

- Documents are a mixture of topic.
- Topics are a mixture of words

LDA uses these assumptions to categorize the text into documents and words per topic.

LDA uses these assumptions to categorize the text into documents and words per topic.
The LDA model has two parameters:

- Alpha (α): Controls per-document topic distribution
- Beta (β): Controls per topic word distribution

The alpha parameter is Dirichlet prior concentration parameter that represents document-topic density.
With a higher alpha, documents are assumed to be made up of more topics and result in more specific topic distribution per document.

Dependency Parsing - Unraveling the Grammar Web

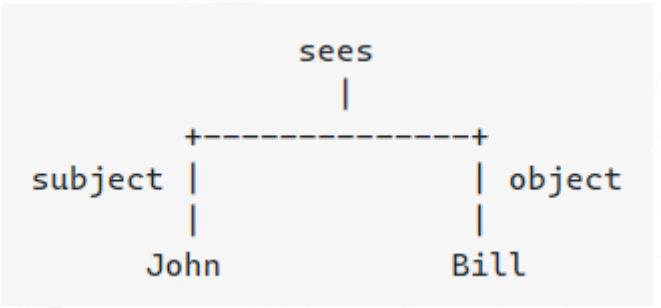
Understanding Sentence Connections: Dependency Parsing helps us figure out how words in a sentence relate to each other.

How Does it Work?

- **Discovering Word Teams:** Think of words as teammates in a sentence, each playing a specific role.
- **Mapping Sentence Relationships:** Dependency Parsing creates a map that shows how words depend on each other to make sentences make sense.

Example

John sees Bill



Discourse Analysis - Understanding Conversational Flow

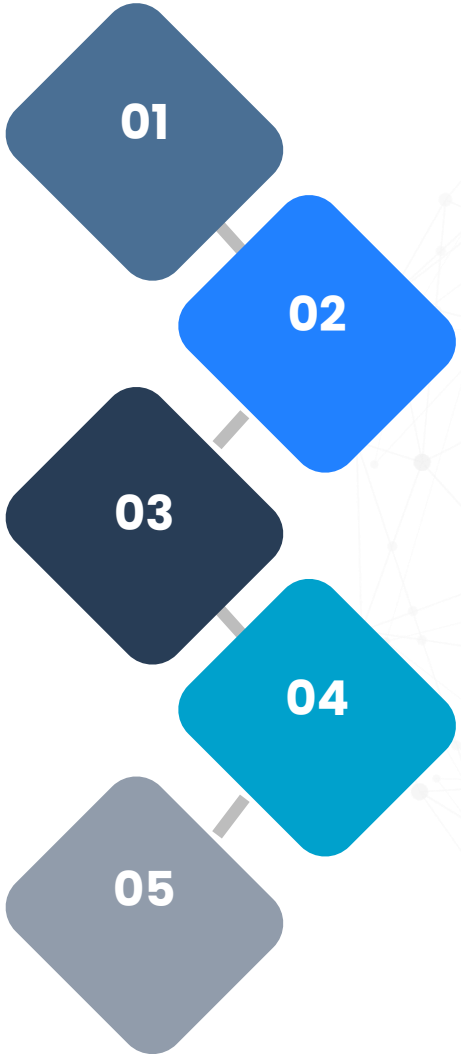
What is Discourse Analysis?

- **Unraveling Conversations:** Discourse Analysis helps us study how conversations flow and how language shapes meaning.
- **Convo Detective:** Think of it as playing detective in conversations, understanding the patterns and dynamics.

How Does Discourse Analysis Work?

- It identifies recurring patterns and structures in spoken or written conversations.
- Like exploring different chapters of a book, Discourse Analysis delves into the context that gives words meaning.

Working of Discourse Analysis



Similar to observing conversations, identifying patterns, and understanding language use.

Discourse Analysis looks at who speaks, what they say, and how it contributes

Discourse Analysis delves into context—cultural, social, and situational factors influencing language use

Discourse Analysis recognizes the linguistic expressions, tones, and styles creating a rhythm in communication.

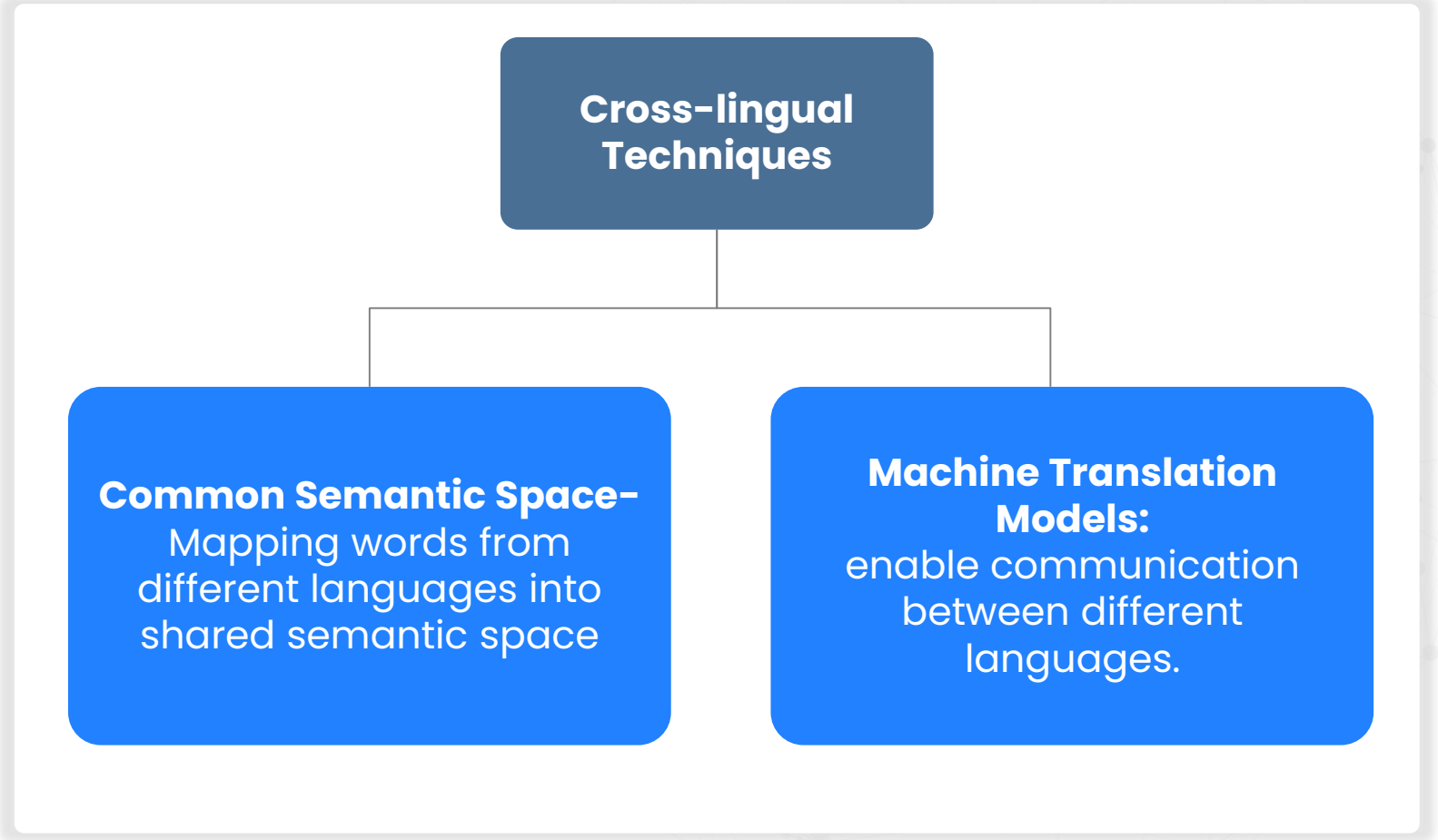
Discourse Analysis helps uncover hidden messages and cultural nuances in conversations.

Cross-lingual and Multilingual NLP Techniques

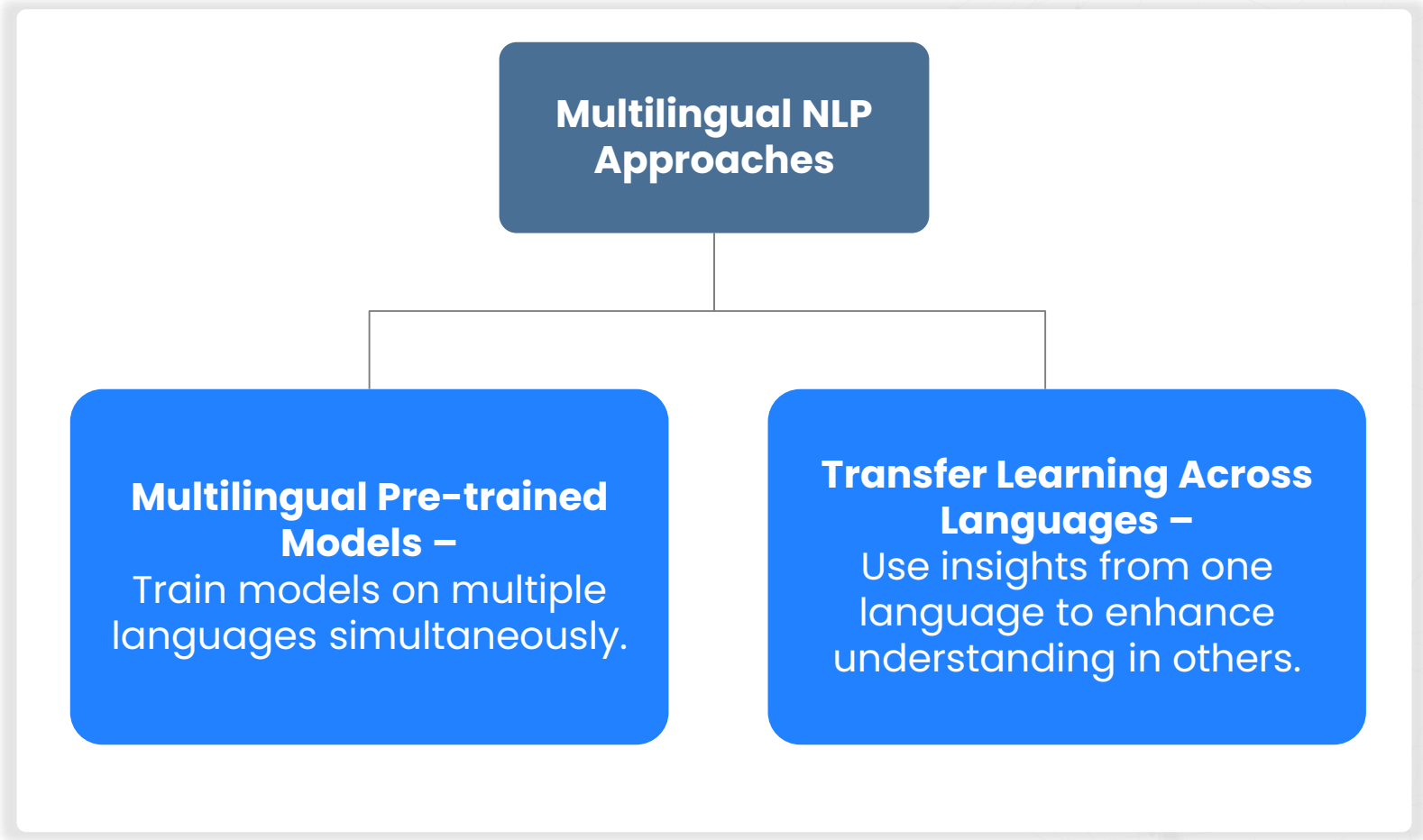
Challenges in Language Variety

01 **Diverse Linguistic Landscapes**
Navigating through various linguistic landscapes, including diverse dialects and language variations.

02 **Crossing Linguistic Borders -**
Addressing differences in syntax, morphology, and semantics across multiple languages.



Cross-lingual and Multilingual NLP Techniques



Sequence Labeling 🏷️

Understanding Sequential Patterns –

- ❖ Identifying and classifying elements in a sequence.
- ❖ Assigning labels to each element for pattern recognition.

Let's Break it Down!

- ❖ Think of it like giving roles to each word in a sentence.
- ❖ Imagine labeling each word for its specific function or entity.

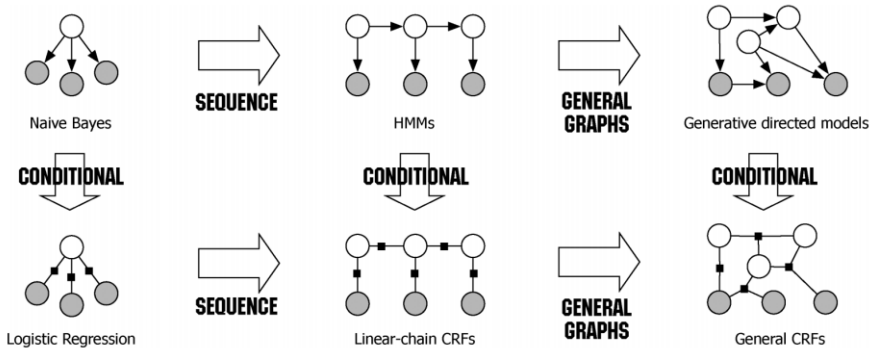
Example -

- *Input: "John visited New York."*
- *Labels: Person O Location*

Techniques for Sequence Labeling

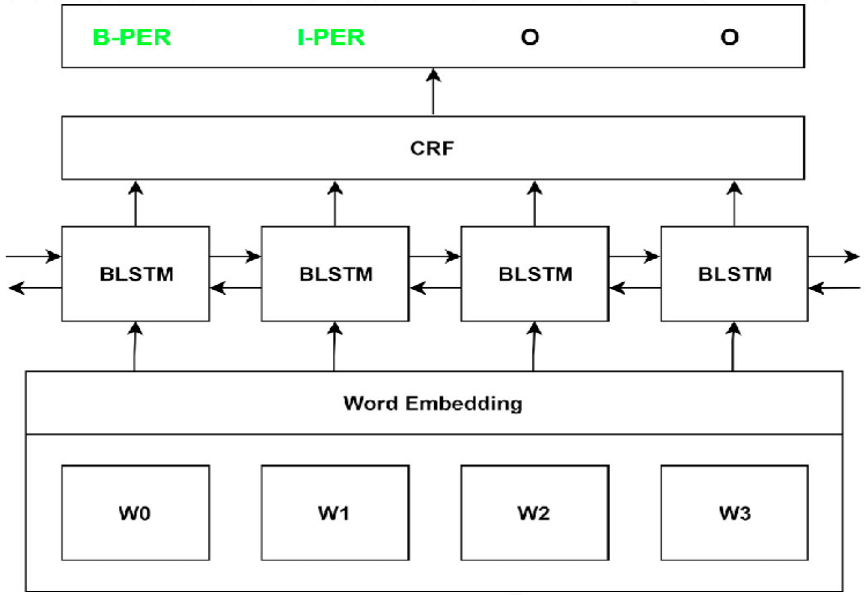
Conditional Random Fields (CRFs):

- Probabilistic models that consider context.
- Helpful for labeling based on neighboring elements.



BiLSTM-CRF:

- Neural network-based approach.
- Utilizes bidirectional LSTM for context and CRF for labeling.



Unveiling Part-of-Speech Tagging

What is Part-of-Speech Tagging?
Assigning grammatical categories (tags) to words in a sentence.

Each tag represents the word's syntactic function.

Techniques for Part-of-Speech Tagging –

- Rule-Based Approaches
- Statistical Models (e.g., Hidden Markov Models)

Let's Dive In!

Common Part-of-Speech Tags:

Noun (N): Names a person, place, thing, or idea.
Verb (V): Describes an action or state of being.
Adjective (Adj): Modifies or describes a noun.
Article (Art): Indicates a noun is coming.
Preposition (Prep): Shows a relationship between a noun (or pronoun) and another word.

Example:

Sentence	The	quick	brown	fox	Jumps.
Tags	Article	Adjective	Adjective	Noun	Verb

Chunking

01 Grouping words in a sentence into meaningful, syntactically correlated chunks.

02 Identifying phrases, such as noun phrases or verb phrases, to unveil structure.

Example-
Sentence: "Microsoft announced the acquisition of LinkedIn."
Chunks: [Microsoft] [announced] [the acquisition of] [LinkedIn]

Techniques for Chunking

Rule-Based Approaches:

- Use predefined grammatical rules to identify chunks.
- Effective for languages with clear structural patterns.

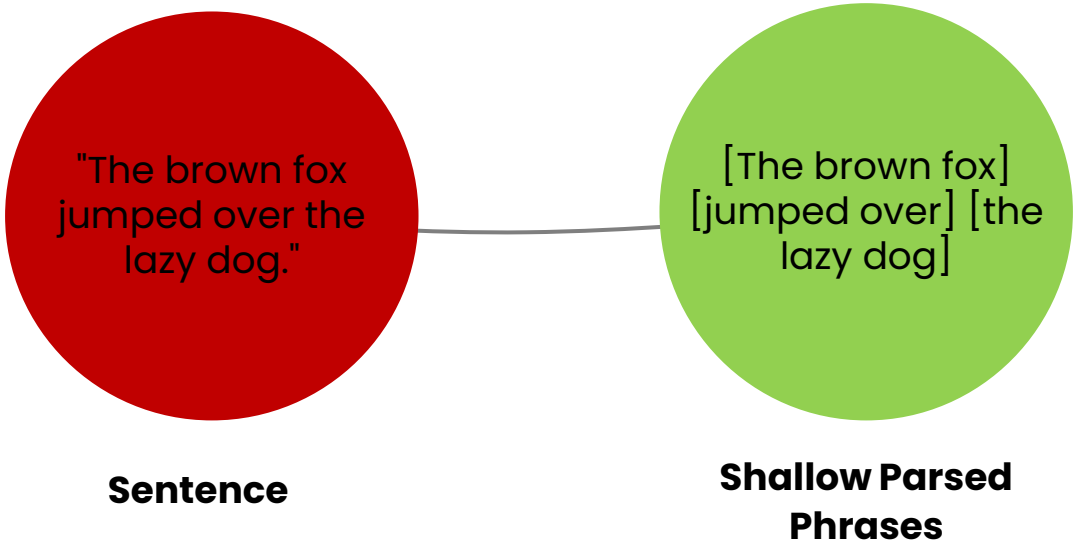
Statistical Models:

- Learn from annotated data to predict chunks.
- Adapt well to diverse linguistic patterns.

Shallow Parsing in NLP

Analyzing the structure of a sentence at a basic level.

Identifying and categorizing specific phrases without a deep syntactic analysis.



NLP Applications

Machine Translation

The use of automated systems to translate text or speech from one language to another.

Why Machine Translation?

- Picture a world where language is no longer a barrier.
- Machine Translation aims to break down language barriers for seamless communication.

Types of Machine Translation

Rule-Based Translation:
- Relies on linguistic rules and dictionaries.

Statistical Machine Translation (SMT):
- Learns translation patterns from large bilingual corpora.

Neural Machine Translation (NMT):
- Utilizes neural networks for end-to-end translation.

Chatbots and Conversational Agents 💬

Enable interactions with users through natural language, providing information or performing tasks.

Why Chatbots?

- Imagine having a virtual assistant available 24/7.
- Chatbots offer instant responses, efficiency, and accessibility for various applications.

Types of Chatbots

Rule-Based Chatbots:

- Follow predefined rules and patterns.
- Suitable for straightforward tasks and structured interactions.

AI-Powered Chatbots:

- Utilize machine learning and natural language processing.
- Evolve through interactions, adapting to user needs.

Question Answering Systems

AI-powered systems designed to understand and respond to user queries with precise information.

Why Chatbots?

- Imagine effortlessly obtaining specific information without searching through texts.
- Question Answering Systems aim to provide instant and relevant responses.

Types of Chatbots

Retrieval-Based Systems:

- Retrieve pre-existing information from a knowledge base.
- Suitable for factual and well-defined queries.

Generative Systems:

- Generate answers based on understanding the query and available knowledge.
- Effective for open-ended and complex questions.

Thank you

94