

Natural Language Processing : Basics to Advance Concept and Application

Vijay Dwivedi



Natural Language Processing

What is NLP?

By “natural language” we mean a language that is used for everyday communication by humans.

NLP is an Intersection of several fields

- Computer Science
- Artificial Intelligence
- Linguistics
- Statistics

It is basically teaching computers to process human language

Two main components:

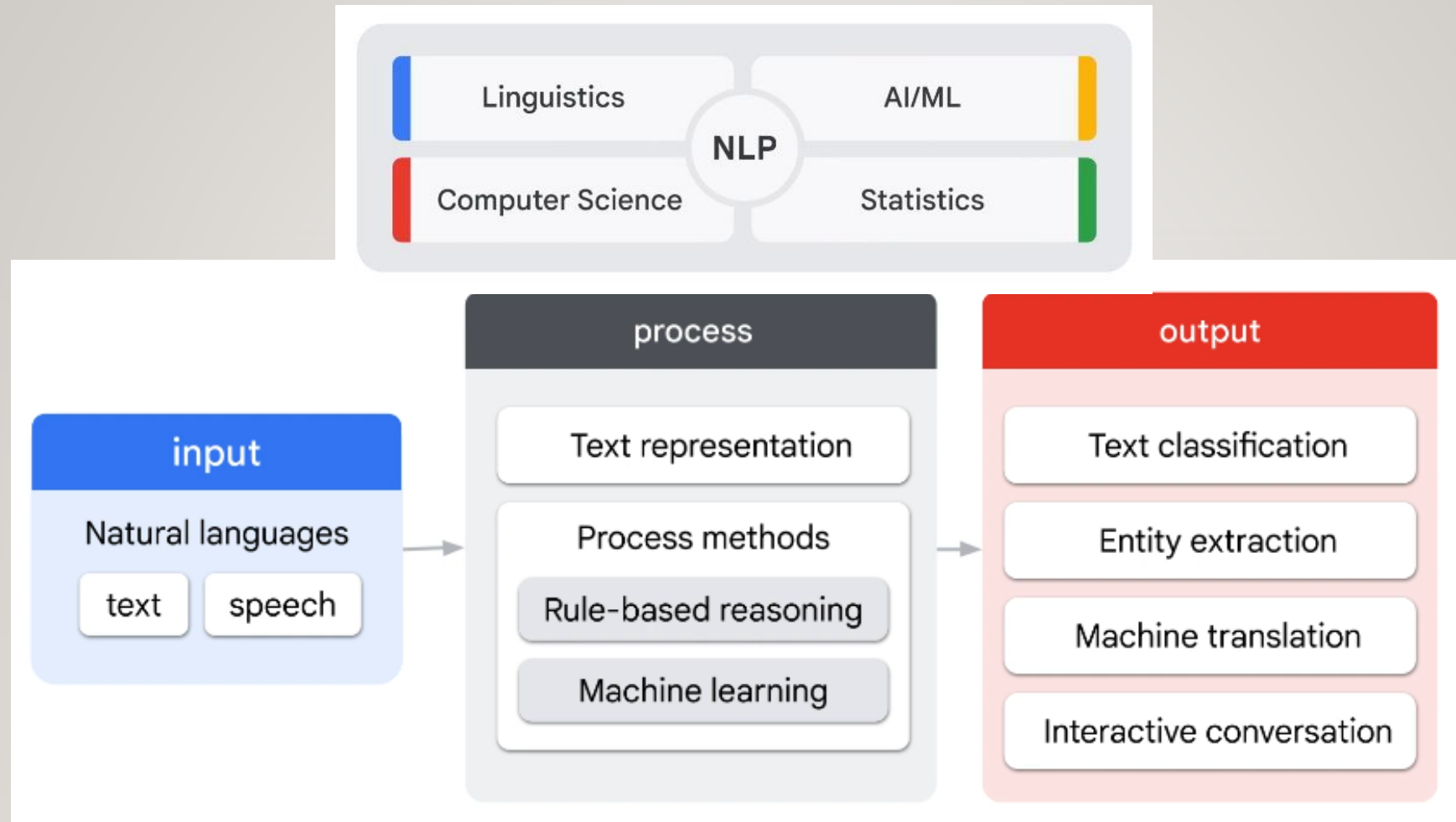
- Natural Language Understanding (NLU)
- Natural Language Generation (NLG)

NLP is AI Complete

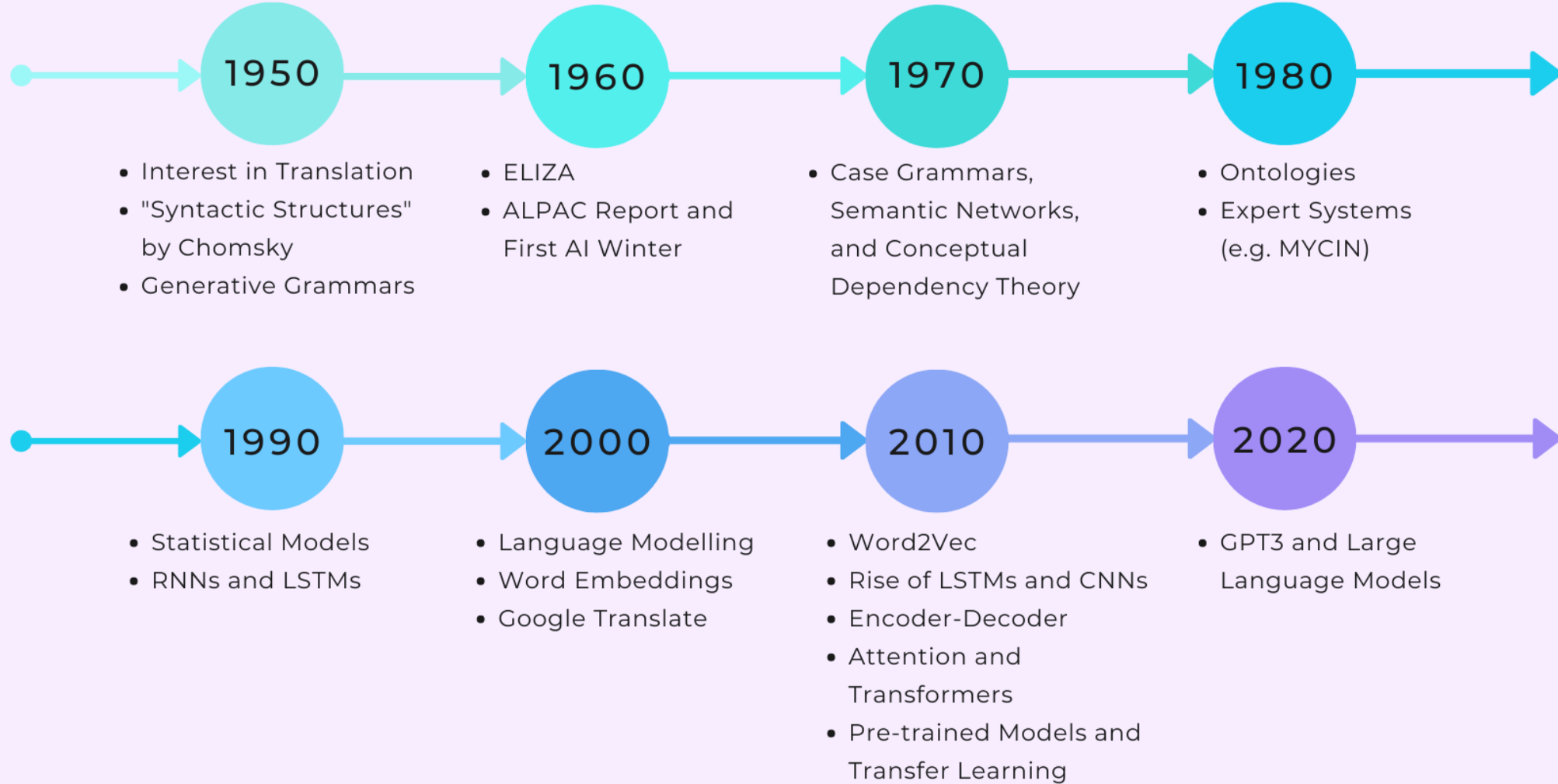
- Requires all types of knowledge humans possess → It's hard!



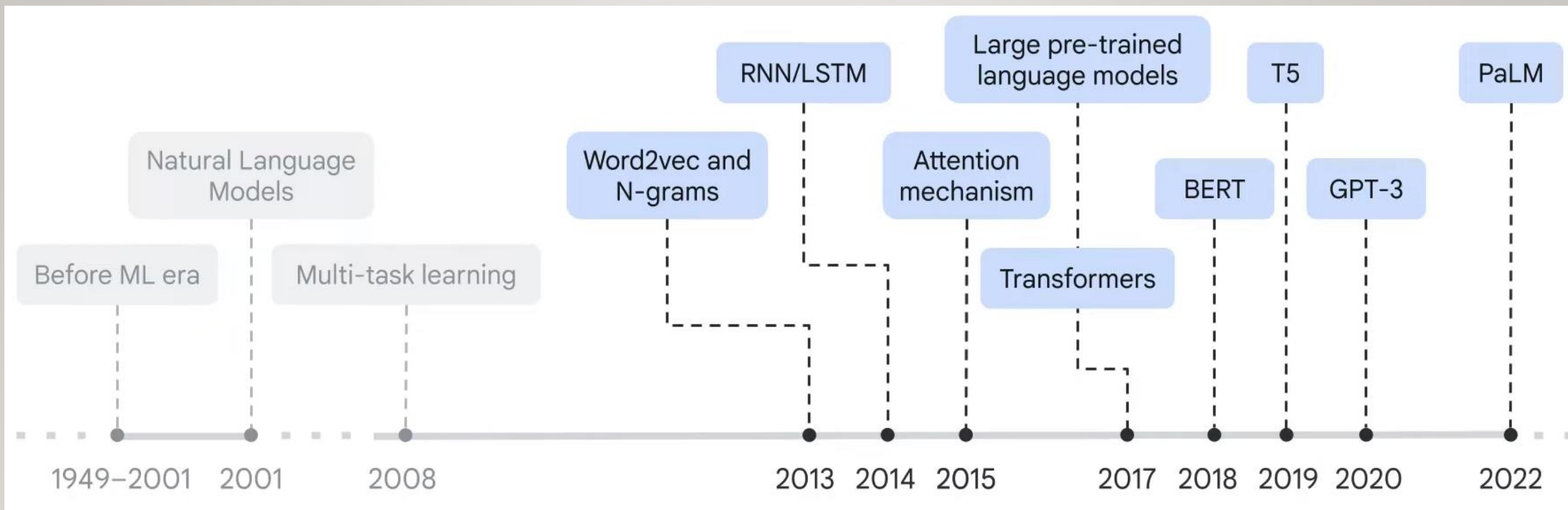
Natural Language Processing



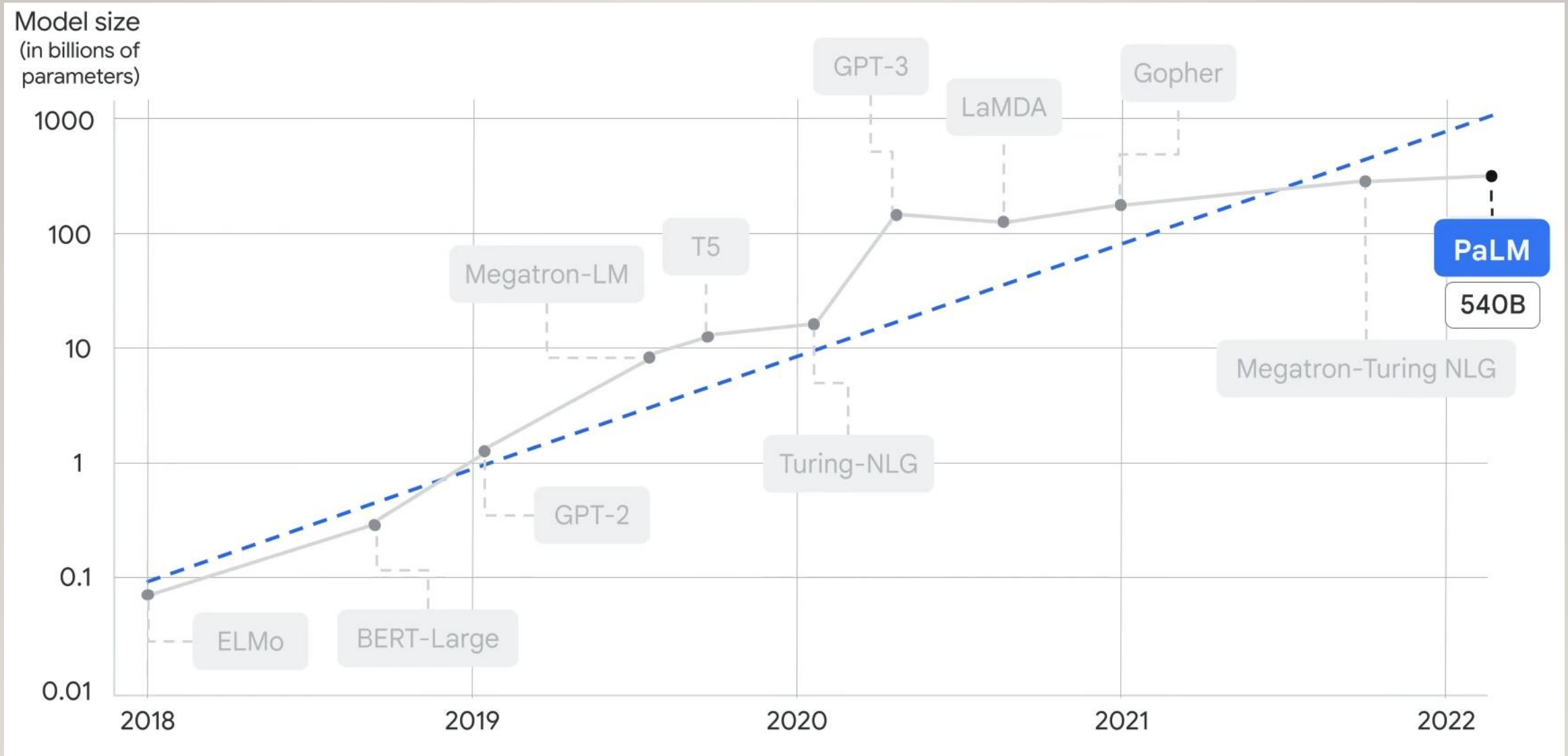
Natural Language Processing : Brief History



Natural Language Processing : Language Model History



Natural Language Processing : Model Development Timeline



Natural Language Processing

NLU Applications

ML on Text (Classification, Regression, Clustering)

Classify emails as spam vs. not spam

Document Recommendation

Recommend news articles based on past articles liked

Sentiment Analysis

Extracting Word/Document Meaning (vectors) and

Relationship Extraction

Topic Modeling

Find documents belonging to a certain topic



Natural Language Processing

NLG Applications



- Image Captioning
- Text Summarization
- Machine Translation
- Question Answering/Chatbots
- Notice NLU is almost a prerequisite for NLG



UNDERSTANDING THE MARKOV PROCESS

- "The future is independent of the past, given the present."
- A Markov process is a stochastic model where the next state depends only on the current state.
- This is called the Markov Property.
- Example:
- If it's raining today, the chance it will rain tomorrow depends only on today's weather, not the weather two days ago.

MARKOV CHAINS — VISUALIZED

- Simple weather model:
- States:  Sunny,  Rainy
 - Sunny \rightarrow Rainy (0.3)
 - Sunny \rightarrow Sunny (0.7)
 - Rainy \rightarrow Sunny (0.4)
 - Rainy \rightarrow Rainy (0.6)
- Transition matrix defines probabilities of moving between states.
- We remember only the current state, not the entire history.

THE MATH BEHIND IT

- Markov Property:
- $P(X_{t+1} \mid X_t, X_{t-1}, \dots, X_0) = P(X_{t+1} \mid X_t)$
- Transition Matrix:
 - $P = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$
- X_t is the state at time t
- P is a probability function

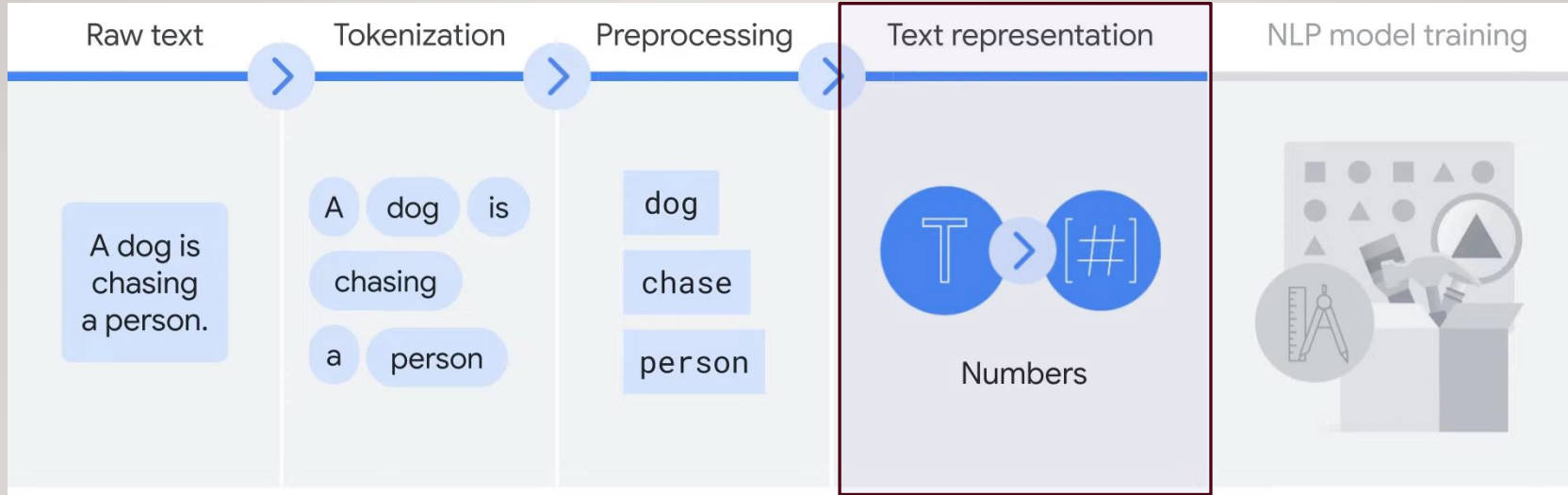
APPLICATIONS OF MARKOV PROCESSES

- Google PageRank: Web surfing modeled as Markov chain
- NLP (n-grams): Predict next word using last few
- Finance: Market states transitions
- Bioinformatics: DNA sequence modeling
- Robotics: Localization and tracking
- Markov models work when current state gives enough context.

WHY GO BEYOND MARKOV?

- Fixed memory: only 1 (or n) steps back
- No learning: transitions are fixed or estimated
- Cannot model long-term dependencies
- This motivates:
 - RNNs: learn from full sequence history
 - LSTMs/GRUs: solve memory problems using gates
 - "Markov models are elegant and fast, but neural nets are powerful."

Text Representation – Feature Engineering



Text Representation:

1. Basic Vectorization :
 - ❖ One hot encoding & Bag of word
2. Word embeddings
 - ❖ Word2vec is family of model architecture and optimizations that can be used to learn word embeddings from large datasets (CBOW, Skip gram)
3. Transfer Learning

Word Embeddings

- Embeddings are a way to encode words capturing the **context** around them.
- Bag of Words -> Sparse one-hot encoded vectors; Count; TF-IDF; Co-occurrence matrix
- Idea of word embeddings:
 - Dimension Reduction
 - Contextual Similarity
 - Semantic Information of words and relations between them

Word Embeddings

- Dimension Reduction
 - If the vocabulary is of 50,000 words, and if the document has a maximum of 500 words, then the document is represented by $50000 * 500$ matrix
 - However, with word embeddings, imagine we assume 20 vector length for a word, then the embedding matrix would be of $50000 * 20$.

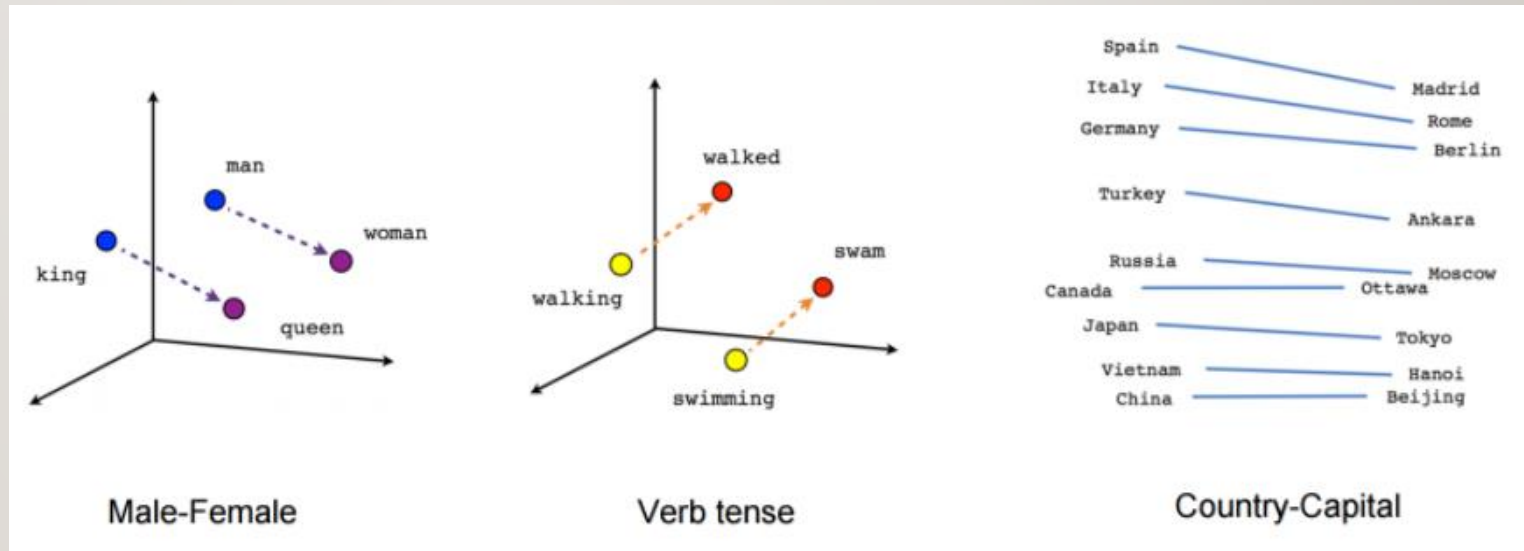
Word Embeddings

- Contextual Similarity – Captures meaning of the word enabling natural language understanding.

	Apple	Orange	Google
Fruit	0.9	0.95	0.1
Company	0.85	0.9	0.9
Classy	0.8	0.2	0.85
Network	0.12	0.8	0.2
Juice	0.87	0.92	0.05

Word Embeddings

- Visualizing the vectors on 2-dimensional space after applying dimension-reduction technique (t-SNE) shows the semantic information about words and their relationships to one another.



Embedding Projector: <http://projector.tensorflow.org/>

GloVe & Word2Vec – CBOW and Skip-gram

- GloVe -> Global Vectors for Word Representation

<https://nlp.stanford.edu/projects/glove/>

- Word2Vec
 - CBOW (Continuous Bag Of Words)
 - Skip-Gram (CBOW in reverse).

See: Embedding.py

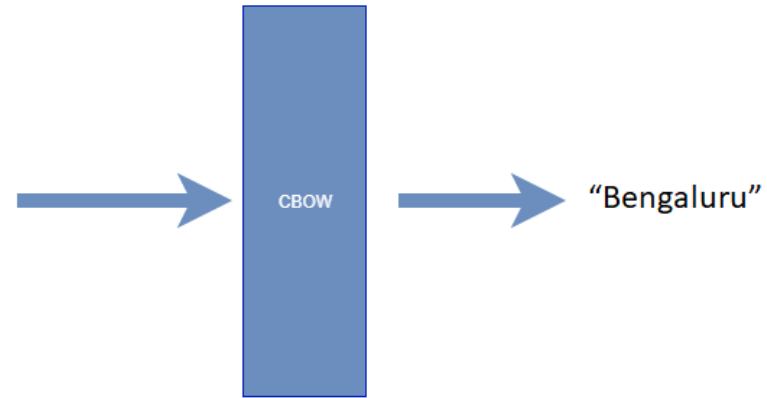
Continuous Bag of Words

- "IT capital of India is _____"

- "Silicon Valley of India is _____"

- "____ is capital of Karnataka"

- "Bannerghatta national park is in _____"



CBOW Implementation

- Given we have vocabulary (set of words in the corpus), and the corpus, from each document we remove any random word and use other words (context) to predict the removed word.
- The context window would include the words before and after the removed word.

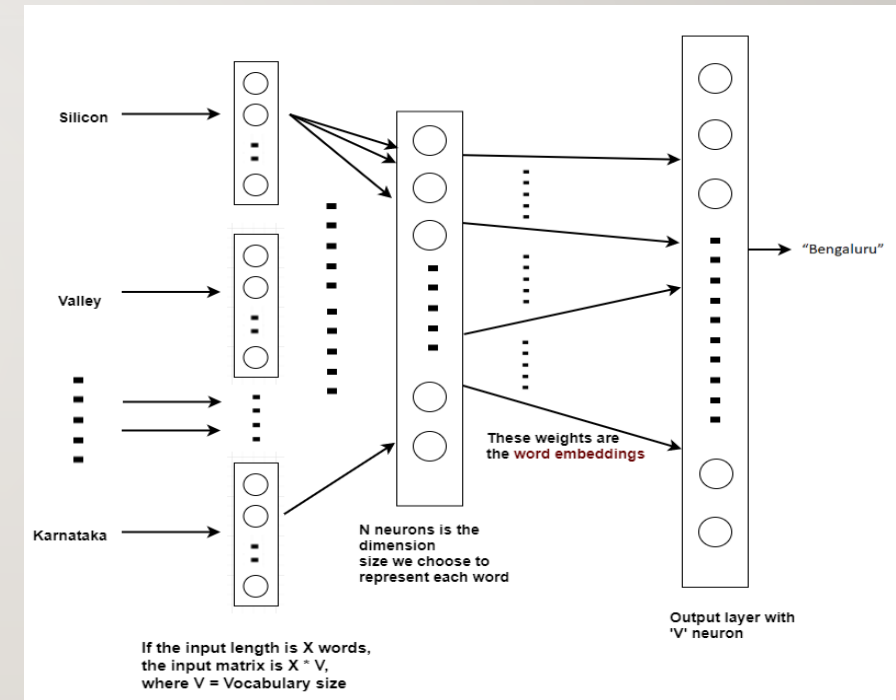
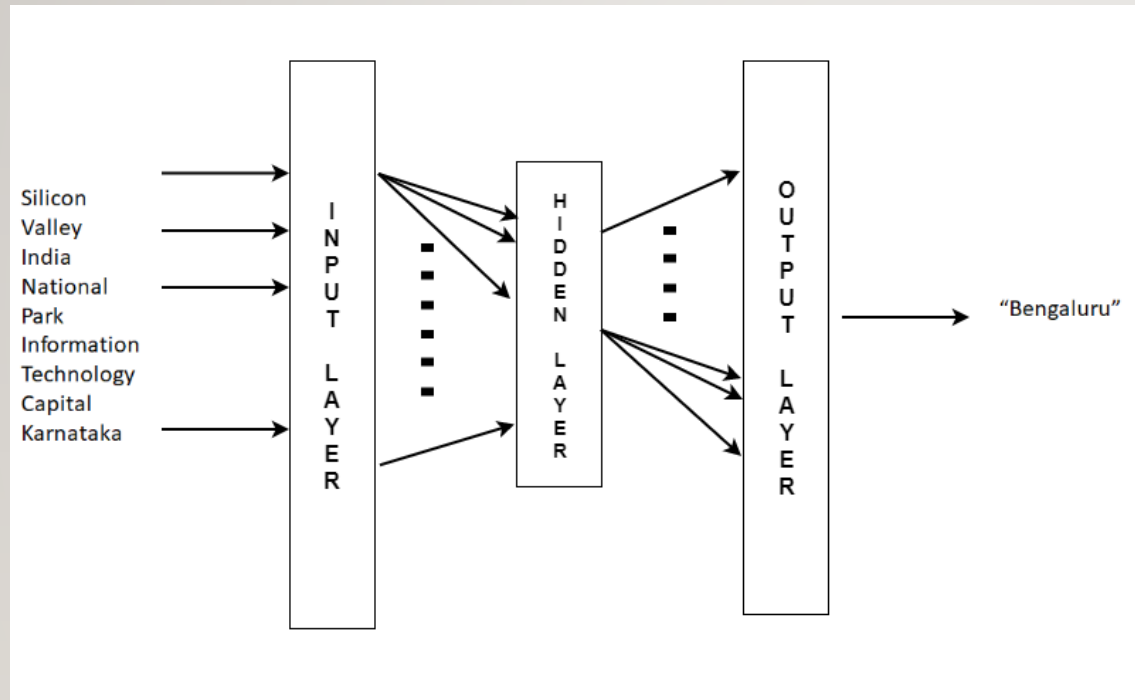
- Example –

“The capital city of Karnataka is _____, the silicon valley and Information Technology capital of India.”

- We have to find out the missing word (W_t) from the vocabulary, whose probability value (given below) is maximum.

$$P(W_t | W_1, W_2, \dots, W_{t-1} \text{ AND } W_{t+1}, W_{t+2}, \dots, W_{t+n})$$

CBOW Neural Network



- Input – Each word is One-hot encoded
- All the layers are linear i.e. no explicit activation function is specified.

Natural Language Processing : Models

Basic Models:

1. ANN (Artificial Neural Network)

A narrow neural network with a single hidden layer

2. DNN (Deep Neural Network):

A deep neural network with multiple hidden layers

3. CNN (Convolution Neural Network)

Best suited for spatial data (e.g., images).

4. RNN (Recurrent Neural Network):

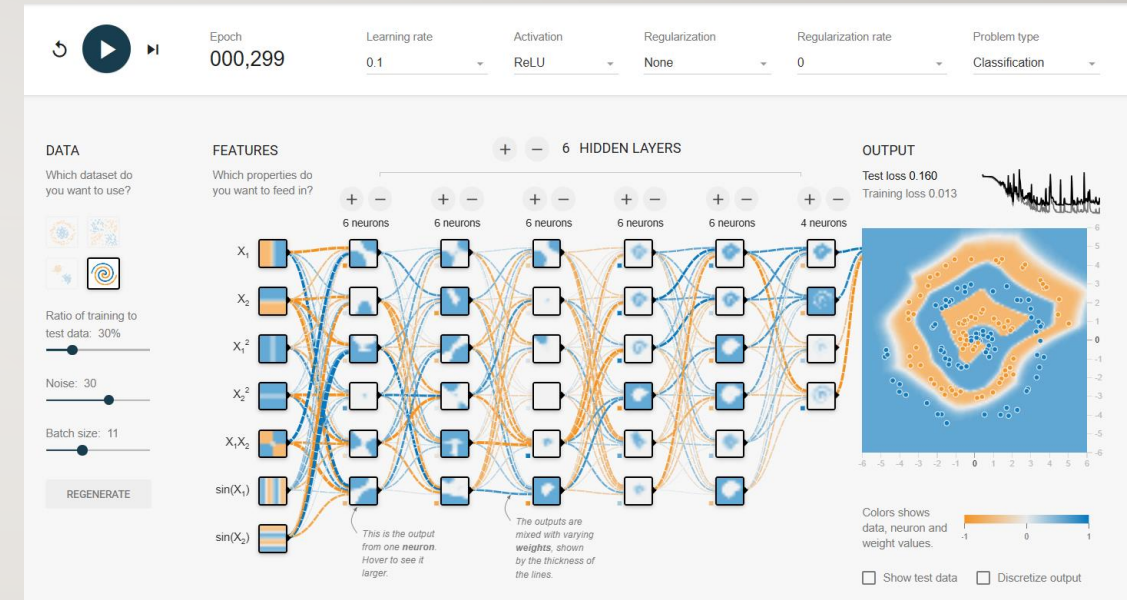
A neural network with short –term memory

5. LSTM (Long Short-term Memory):

A neural network with long-term memory

6. GRU (Gated Recurrent Unit):

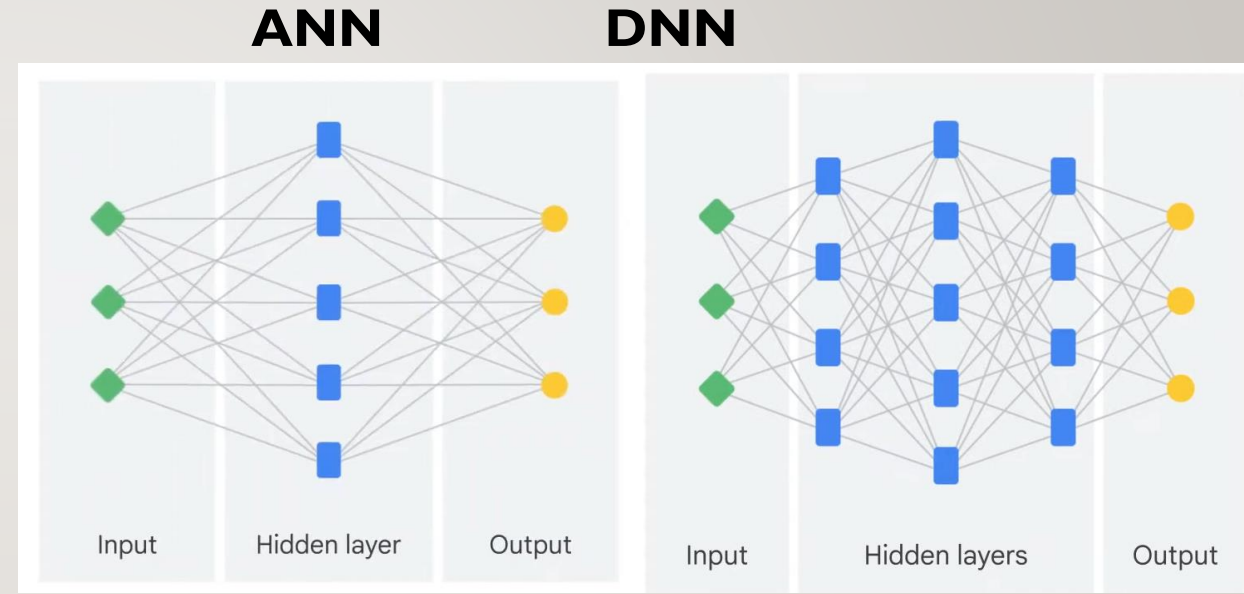
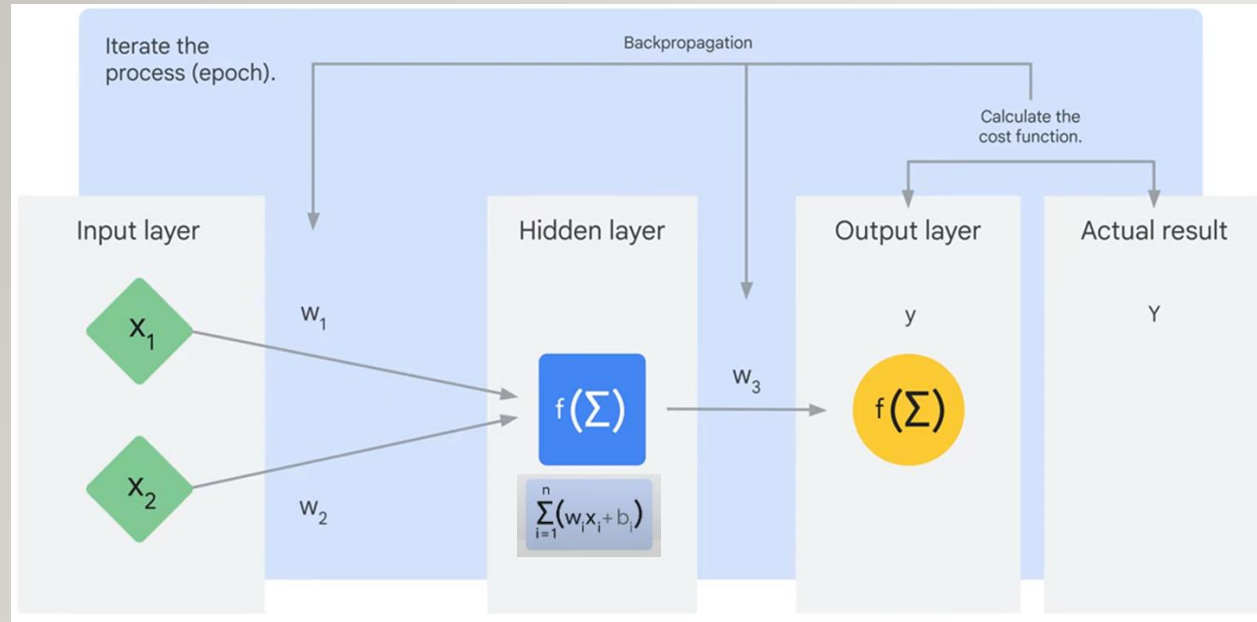
An improved and simplified variant of LSTM



Artificial neural network (ANN)

Deep neural network (DNN)

Definition: ANNs are the simplest form of neural networks, consisting of layers of interconnected neurons that process and transmit information

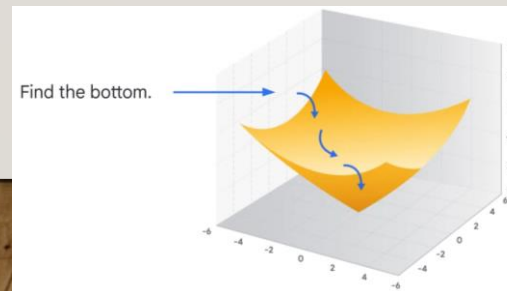


Cost Function

$$MSE = \frac{\sum_{i=1}^n (\hat{Y}_i - Y_i)^2}{n}$$

\hat{Y}_i – Predicted value
 Y_i – Actual value
 n – Size of the training data

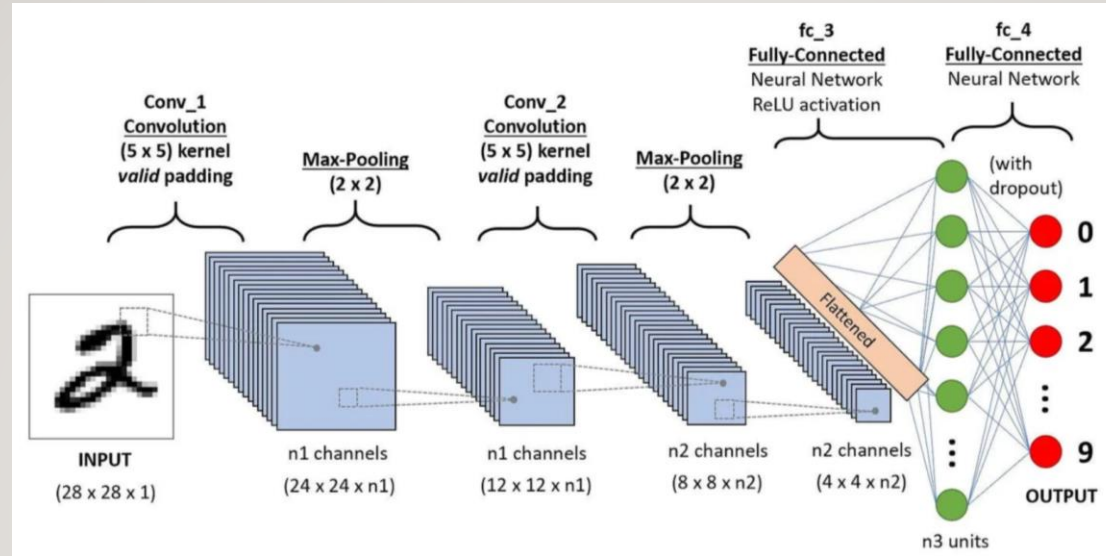
Gradient Descent



Weight Update



CNN : Convolutional Neural Networks



Definition: CNNs are specialized neural networks designed for processing structured grid data like images. They use convolutional layers to detect features.

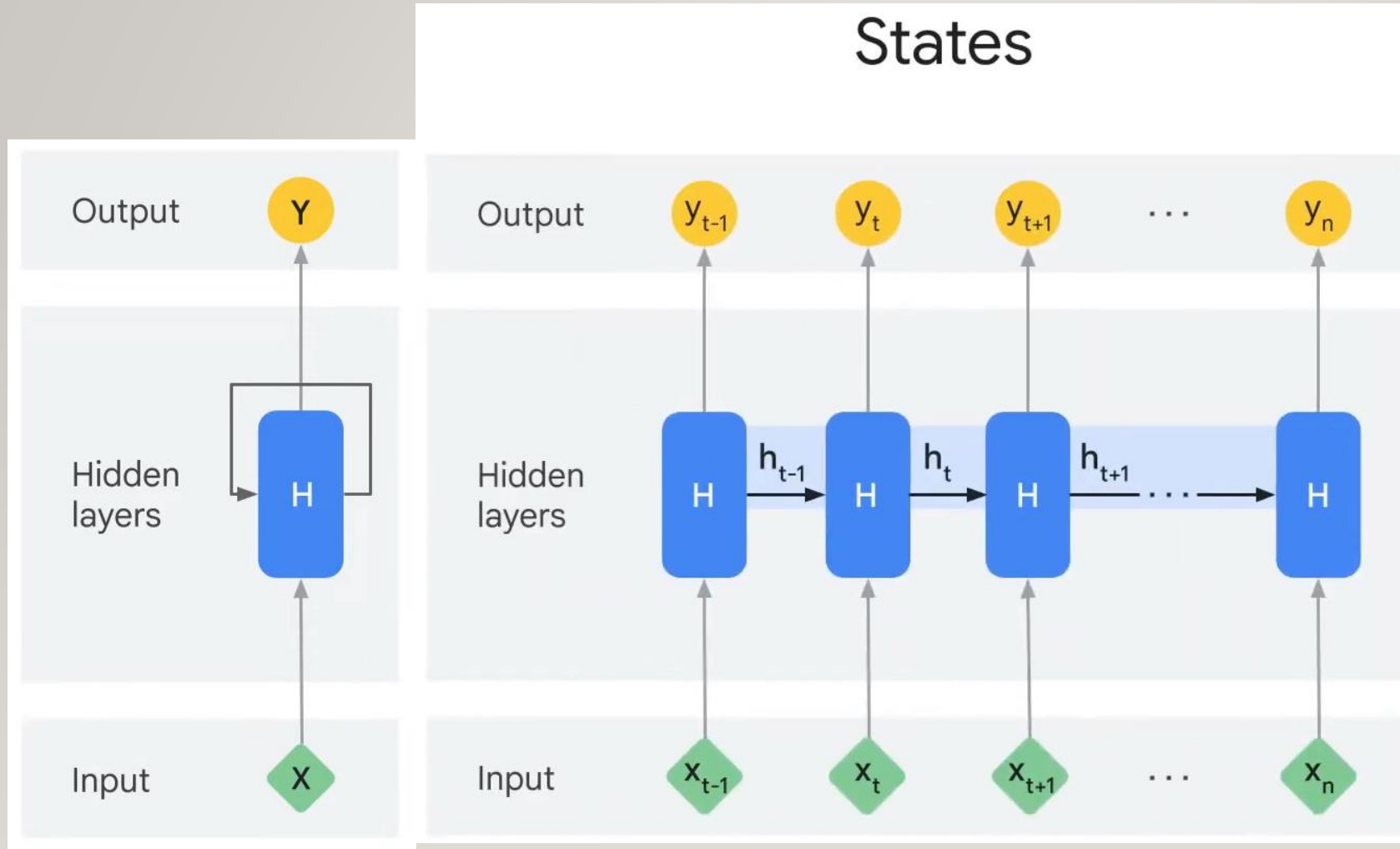
Structure:

Convolutional Layers: Apply filters to the input data to create feature maps.

Pooling Layers: Reduce the dimensionality of feature maps.

Fully Connected Layers: Connect neurons from previous layers to the output layer.

Recurrent Neural Network



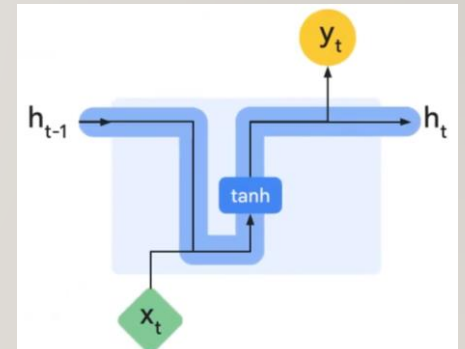
An RNN suffers from short-term memory due to vanishing gradients.

Definition: RNNs are designed for sequential data, where each neuron connects to the next layer and to neurons within the same layer.

Structure:

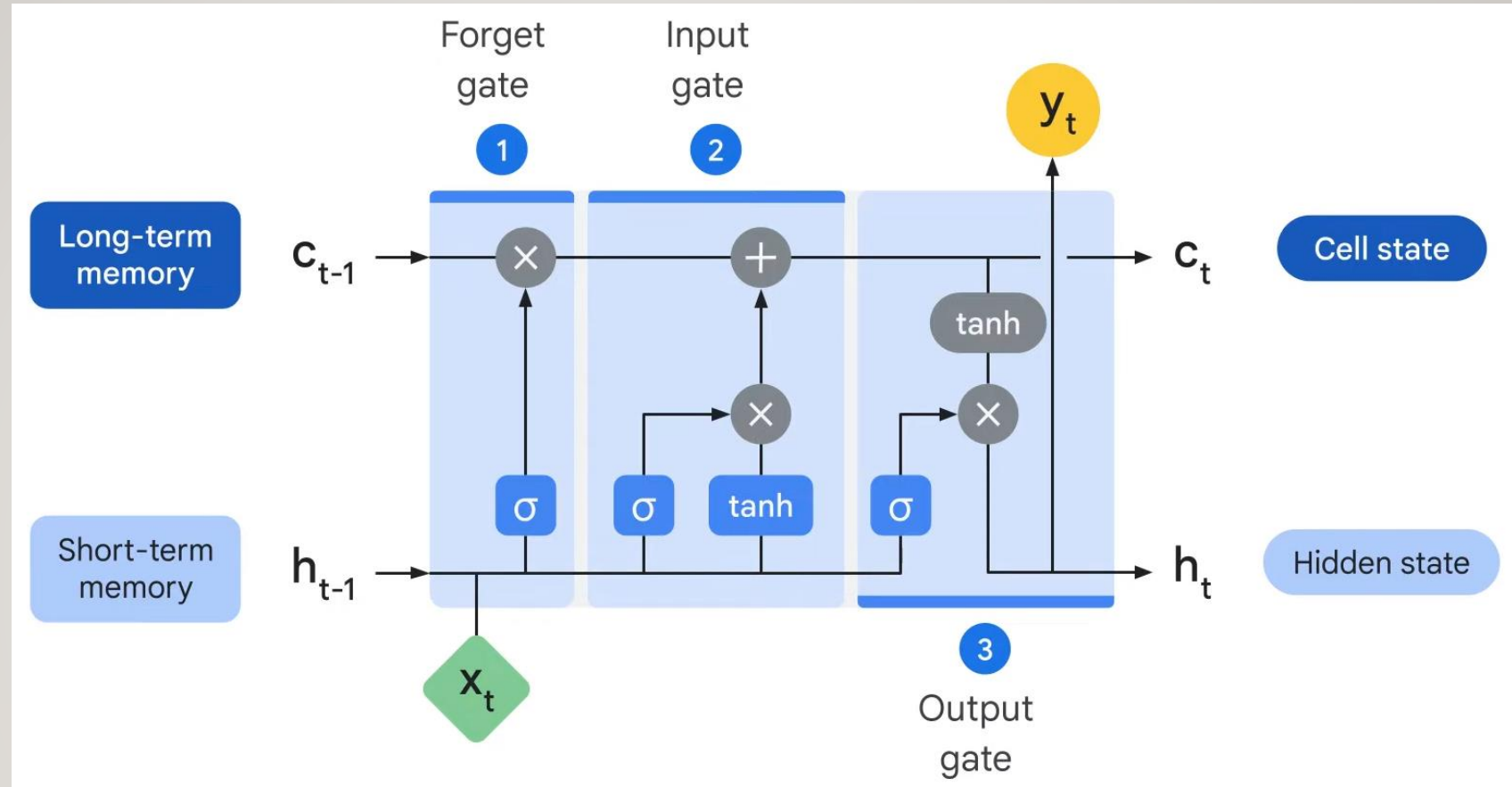
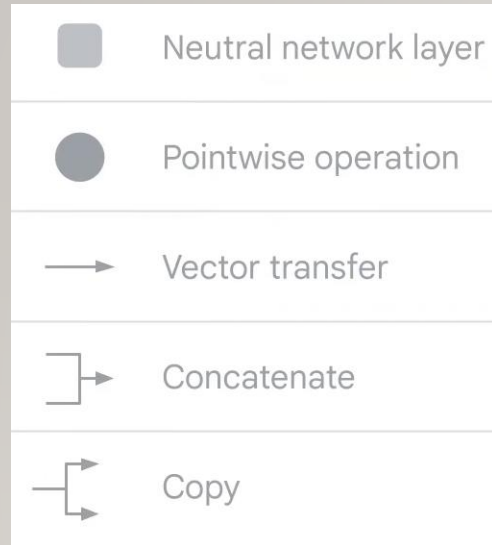
Recurrent Connections: Neurons have connections looping back to themselves, allowing information to persist.

Hidden State: Maintains a memory of previous inputs.



He grew up in the United States of America.

LSTM Cell



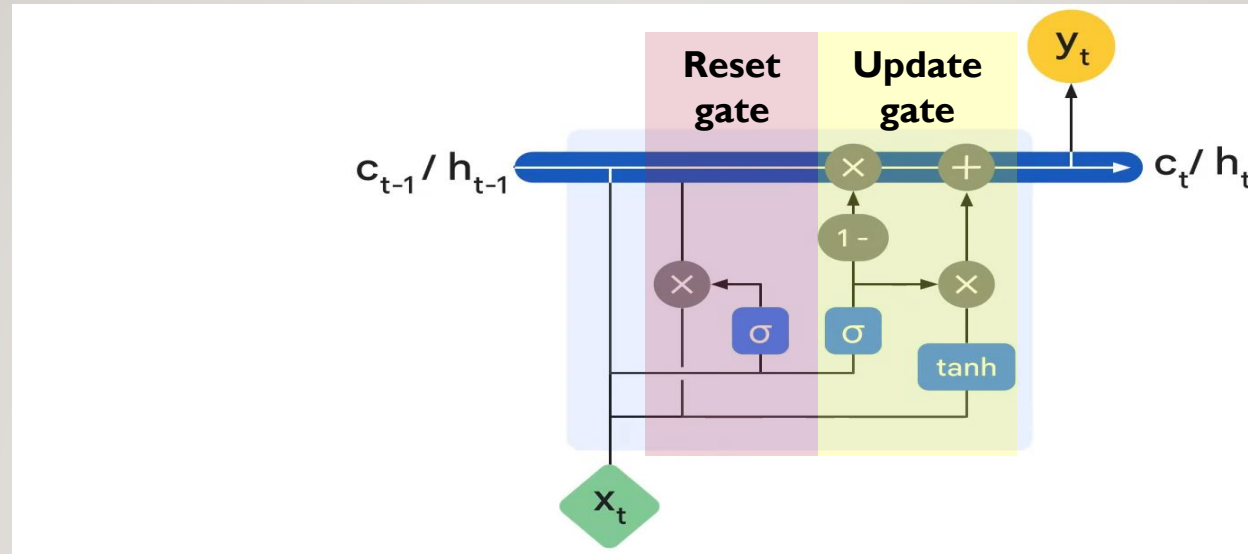
Definition: LSTMs are a type of RNN designed to learn long-term dependencies and retain information over longer sequences.

Structure:

Memory Cell: Maintains information over time.

Gates: Control the flow of information in and out of the cell (input gate, forget gate, output gate).

Gated Recurrent Unit (GRU)



Definition: GRUs are a type of RNN similar to LSTMs but with a simplified architecture, designed to capture dependencies in sequential data.

Structure:

Gates: Includes an update gate and a reset gate, simplifying the control of information flow compared to LSTMs.

Hidden State: Combines the memory cell and hidden state into a single structure.

Popular Libraries of NLP (NLTK/spaCy)

NLTK was originally created in 2001 as part of a computational linguistics course in the Department of Computer and Information Science at the **University of Pennsylvania**.

- Since then, it has been developed and expanded with the help of dozens of contributors. It has now been adopted in courses in dozens of universities, and serves as the basis of many research projects

NLTK is a leading platform for building Python programs to work with human language data.

- It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries.

spaCy is an open-source software library for advanced Natural Language Processing, written in the programming languages Python and Cython.

- The library is published under the MIT license and currently offers statistical neural network models for English, German, Spanish, Portuguese, French, Italian, Dutch and multi-language NER, as well as tokenization for various other languages.



Comparision Between NLTK & spaCy)

Options Vs. Best

NLTK provides a number of algorithms to choose from – good for research. Its provides different stemming libraries, for example, allow you to finely customize your model.

In contrast, spaCy implements a single stemmer, the one that the spaCy developers feel to be best (Opinionated).

Library Implementation Perspective

NLTK is essentially a string processing library, unlike spaCy which uses object-oriented approach.

Research Vs. Production

Unlike NLTK, which is widely used for **teaching and research**, spaCy focuses on providing software for **production** usage (spaCy also supports deep learning workflows that allow connecting statistical models trained by popular machine learning libraries like TensorFlow, Keras, Scikit-learn or PyTorch).



Demo Time

Natural Language Processing :Advance

NLP: LARGE LANGUAGE MODELS

- LLMs are AI systems used to model and process human language. They are called “large” because these types of models are normally made of hundreds of millions or even billions of parameters that define the model's behavior, which are pre-trained using a massive corpus of text data.
- The underlying technology of LLMs is called transformer neural network, simply referred to as a **transformer**. Presented by Google researchers in the famous paper **Attention is All You Need** in 2017, transformers are capable of performing natural language (NLP) tasks with unprecedented accuracy and speed. With its unique capabilities, transformers have provided a significant leap in the capabilities of LLMs. It's fair to say that, without transformers, the current generative AI revolution wouldn't be possible.
- LLM process flow: **Tokenization** → **Embedding** → **Attention via Transformer** → **Prediction** → **Response**

Major Large Language Models (LLMs)

ranked by capabilities, sized by billion parameters used for training

CLICK LEGEND ITEMS TO FILTER

anthropic chinese google meta microsoft mistral openAI other

search...

show only: significant models



David McCandless, Tom Evans, Paul Barton
Informationisbeautiful // Jan 2024

MMLU = benchmark for measuring LLM capabilities
* = parameters undisclosed // source: [LifeArchitect](#) // [data](#)

Natural Language Processing :Transformers History

June 2018: GPT, the first pretrained Transformer model, used for fine-tuning on various NLP tasks and obtained state-of-the-art results (<http://nlp.seas.harvard.edu/2018/04/03/attention.html>)

October 2018: BERT, another large pretrained model, this one designed to produce better summaries of sentences.

February 2019: GPT-2, an improved (and bigger) version of GPT that was not immediately publicly released due to ethical concerns

October 2019: T5, A multi-task focused implementation of sequence-to-sequence Transformer architecture.

May 2020, GPT-3, an even bigger version of GPT-2 that is able to perform well on a variety of tasks without the need for fine-tuning (called *zero-shot learning*)

January 2022: InstructGPT, a version of GPT-3 that was trained to follow instructions better This list is far from comprehensive, and is just meant to highlight a few of the different kinds of Transformer models. Broadly, they can be grouped into three categories:



Natural Language Processing : Transformers History

January 2023: [Llama](#), a large language model that is able to generate text in a variety of languages.

March 2023: [Mistral](#), a 7-billion-parameter language model that outperforms Llama 2 13B across all evaluated benchmarks, leveraging grouped-query attention for faster inference and sliding window attention to handle sequences of arbitrary length.

May 2024: [Gemma 2](#), a family of lightweight, state-of-the-art open models ranging from 2B to 27B parameters that incorporate interleaved local-global attentions and group-query attention, with smaller models trained using knowledge distillation to deliver performance competitive with models 2-3 times larger.

November 2024: [SmolLM2](#), a state-of-the-art small language model (135 million to 1.7 billion parameters) that achieves impressive performance despite its compact size, and unlocking new possibilities for mobile and edge devices.



Natural Language Processing : Transformers History

2018

GPT

2019

BERT

XLNet

XLNet

GPT-2

RoBERTa

XLNet

ALBERT

T5

BART

DistilBERT

2020

ELECTRA

Longformer

DeBERTa

GPT-3

2021

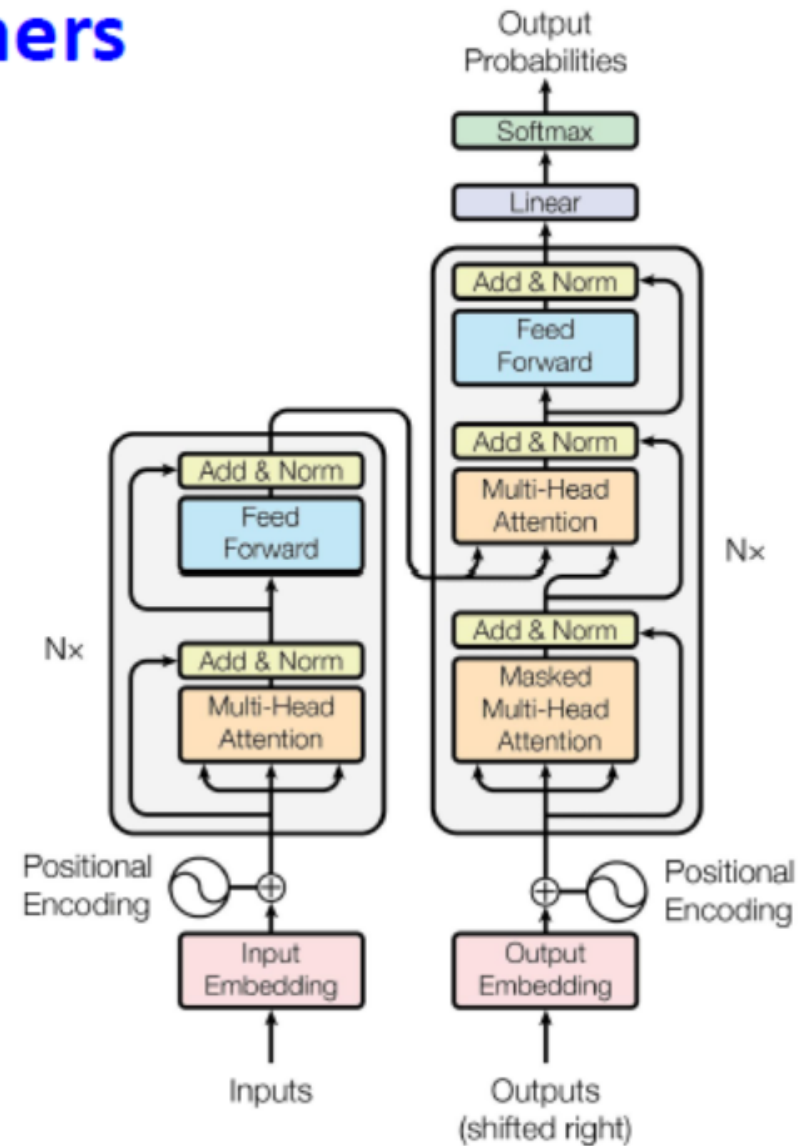
M2M100

LUKE



Transformers

- Tokenization
- Input Embeddings
- Position Encodings
- Query, Key, & Value
- Attention
- Self Attention
- Multi-Head Attention
- Feed Forward
- Add & Norm
- Encoders
- Masked Attention
- Encoder Decoder Attention
- Linear
- Softmax
- Decoders
- Encoder-Decoder Models



Natural Language Processing :Transformers History

2018

GPT

2019

BERT

XLNet

GPT-2

XLNet

RoBERTa

ALBERT

DistilBERT

BART

2020

T5

ELECTRA

Longformer

DeBERTa

2021

GPT-3

M2M100

LUKE

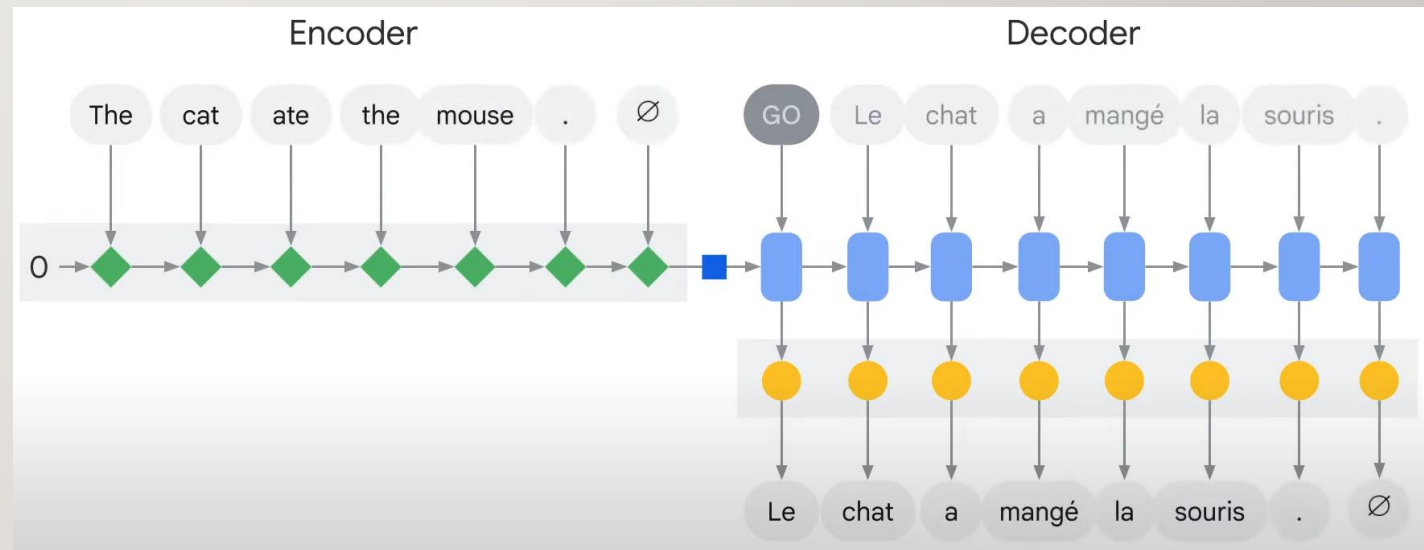


Encoder – Decoder Architecture

Goal: Develop an architecture capable of generating *contextually appropriate, arbitrary length*, output sequences

- **Applications:**

- Machine translation,
- Summarization,
- Question answering,
- Dialogue modeling.



Encoder – Decoder Architecture

The **Encoder–Decoder architecture** is a foundational design pattern in deep learning, particularly effective for tasks involving sequences, such as machine translation, summarization, and text generation. It consists of two main components:

1.Encoder: This part processes the input sequence and compresses it into a fixed-length context vector (or a set of vectors), capturing essential information. Typically, it uses RNNs, GRUs, LSTMs, or Transformers to handle variable-length inputs and maintain sequence order.

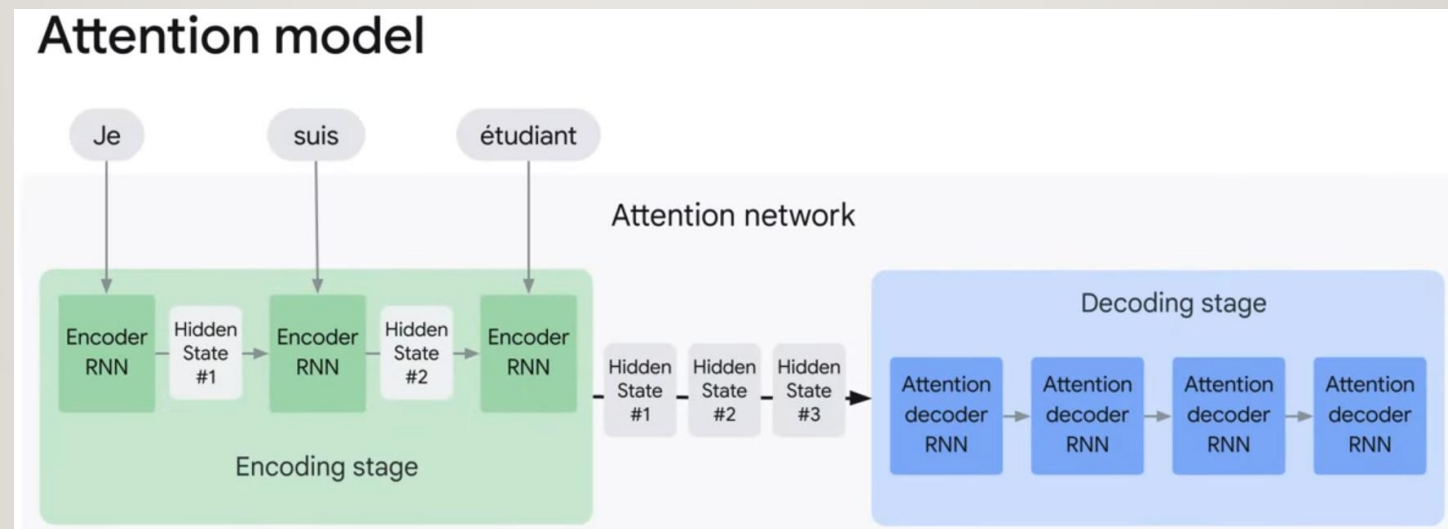
2.Decoder: The decoder takes the encoder's context and generates an output sequence, one element at a time. It is also usually built using sequence models and is trained to predict the next token based on the context and previous outputs.

This architecture allows for **sequence-to-sequence (seq2seq)** learning, where the input and output lengths may differ. Attention mechanisms are often added to enhance performance by allowing the decoder to focus on different parts of the input at each step, improving translation accuracy and contextual understanding.



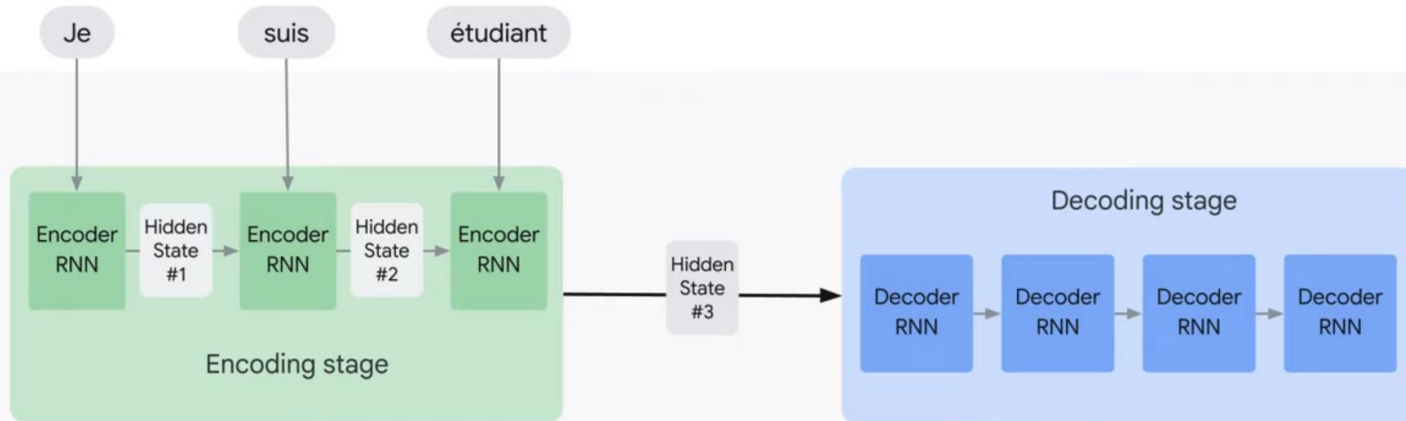
Attention Mechanism

The **attention mechanism** is a powerful enhancement to neural networks, especially in sequence modeling tasks like translation and summarization. It allows models to dynamically focus on different parts of the input sequence when generating each element of the output. Rather than encoding the entire input into a single fixed vector, attention computes a weighted sum of all input representations, giving higher importance to relevant tokens. This improves performance, especially on long sequences, by preserving contextual details. Attention is central to models like Transformers, enabling parallel processing and superior accuracy across NLP tasks.



Attention Mechanism

Traditional RNN encoder-decoder



Attention model differs from a traditional model

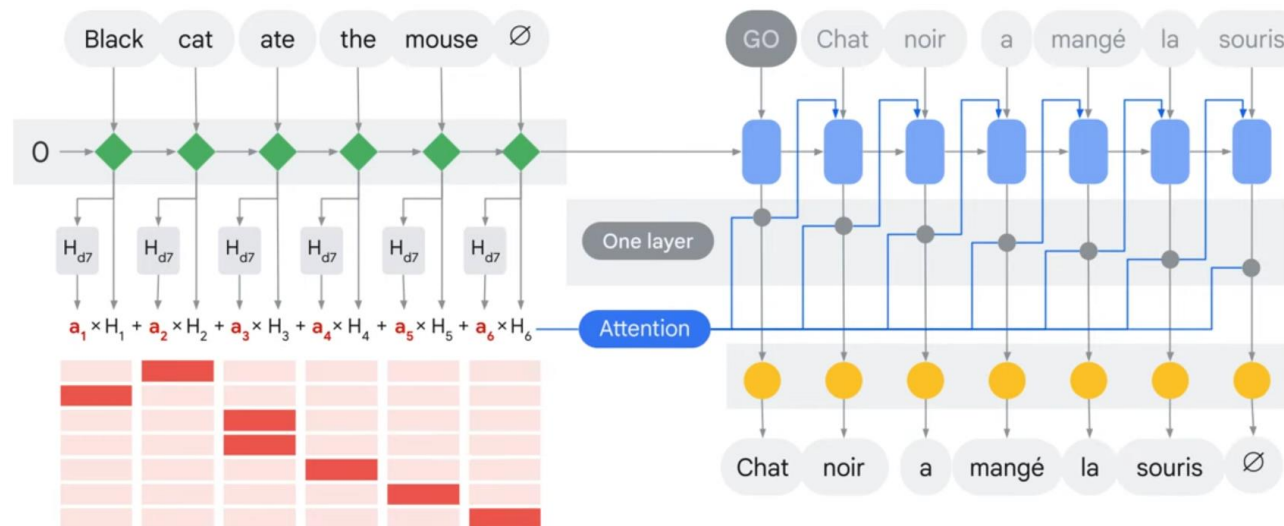
01

Passing more data

02

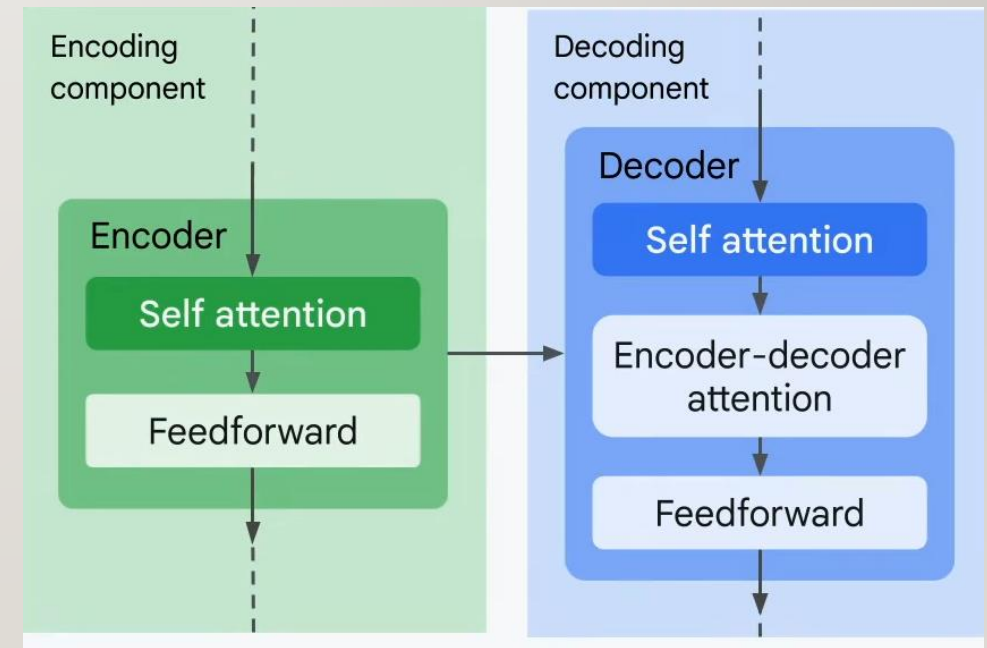
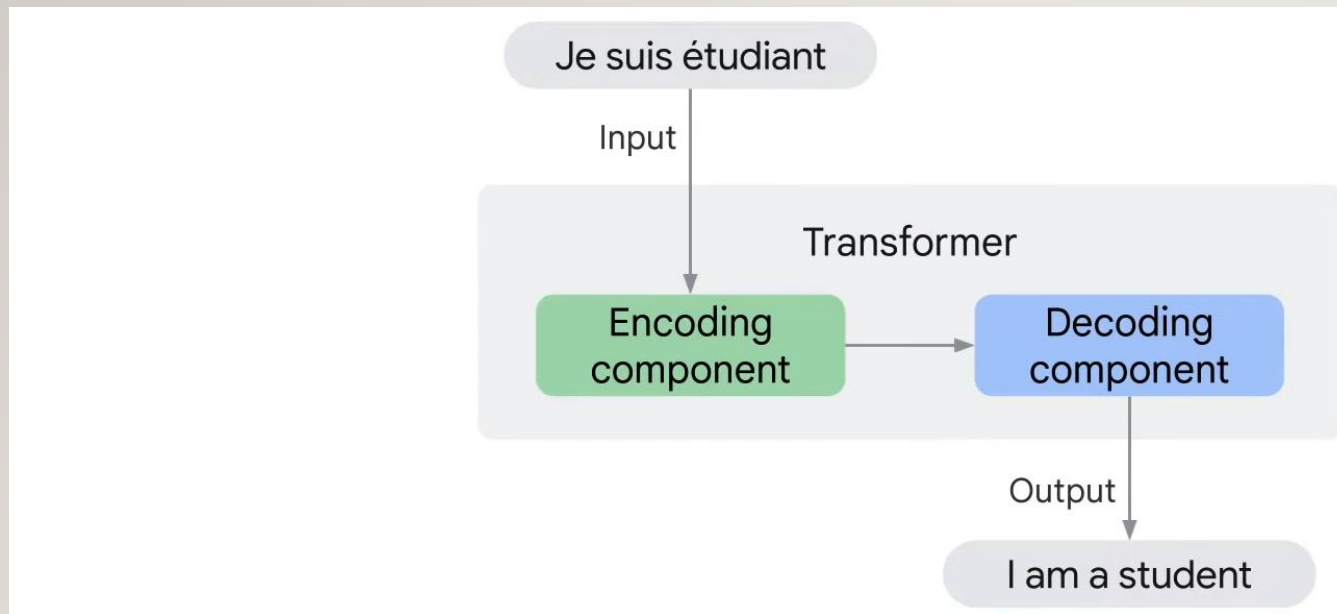
Extra step before
producing its output

Improve translation with attention network



Transformer : An encoder-Decoder model with Attention Mechanism

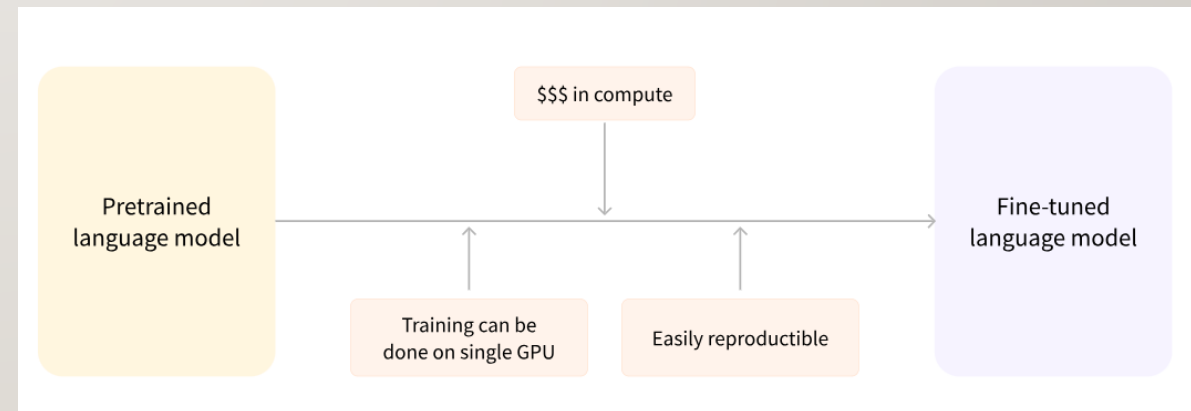
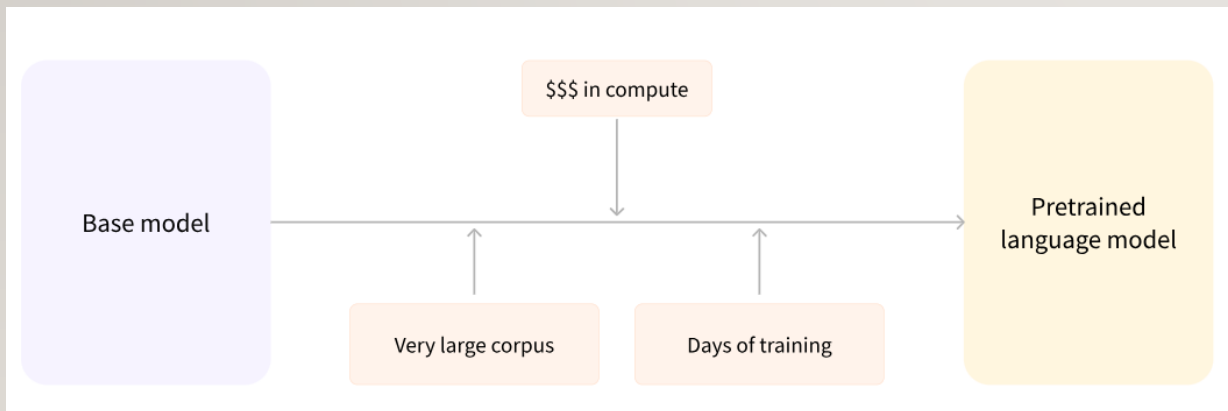
Transformer is a neural network architecture that has fundamentally changed the approach to Artificial Intelligence. Transformer was first introduced in the seminal paper "[Attention is All You Need](#)" in 2017 and has since become the go-to architecture for deep learning models, powering text-generative models like OpenAI's **GPT**, Meta's **Llama**, and Google's **Gemini**. Beyond text, Transformer is also applied in [audio generation](#), [image recognition](#), [protein structure prediction](#), and even [game playing](#), demonstrating its versatility across numerous domains.



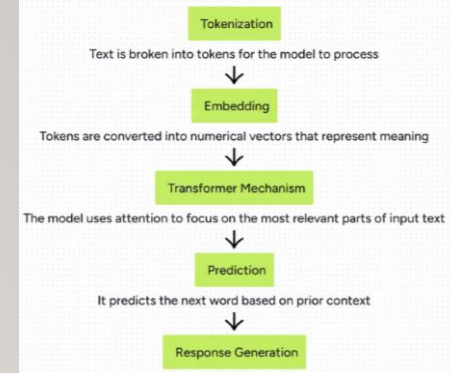
Large Language Model

NLP vs LLM:

- ✓ **NLP (Natural Language Processing)** is the broader field focused on enabling computers to understand, interpret, and generate human language. NLP encompasses many techniques and tasks such as sentiment analysis, named entity recognition, and machine translation.
- ✓ **LLMs (Large Language Models)** are a powerful subset of NLP models characterized by their massive size, extensive training data, and ability to perform a wide range of language tasks with minimal task-specific training. Models like the Llama, GPT, or Claude series are examples of LLMs that have revolutionized what's possible in NLP.



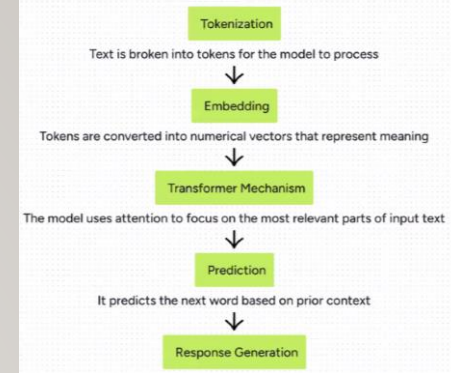
STEP I – TOKENIZATION



- Text is broken into smaller units called tokens
 - 1 token = $\frac{3}{4}$ of words (rough approximation for large corpus)
- Tokens can be words, sub words, or characters
- Helps the model understand and process the input efficiently
- **“ Explain large language models to me like I am five years old”**

Explain large language models to me like I am five years old

STEP 2 – EMBEDDING



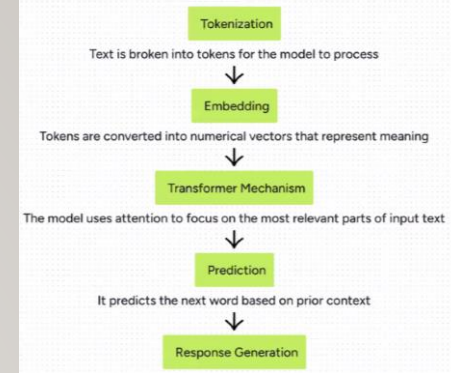
- Tokens are converted into numerical vectors
- These vectors represent the semantic meaning of words
- Embeddings enable mathematical operations and pattern recognition in text
- “ **Explain large language models to me like I am five years old**”

Explain = [0.2, 0.5,, 0.9]

Large = [0.45, 0.2,, 0.7]

- Similar words will take closure distance

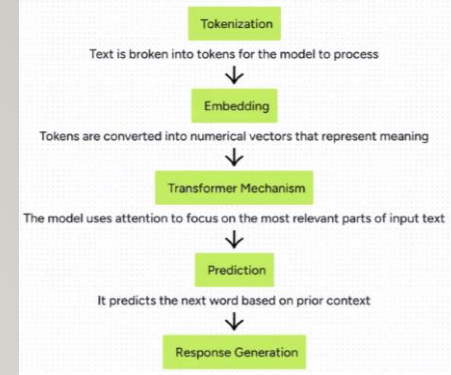
STEP 3 – TRANSFORMER MECHANISM



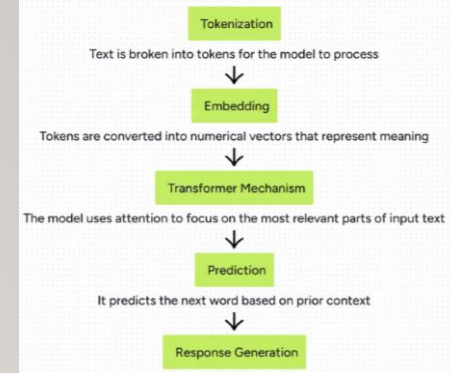
- The core of modern language models
- Uses self-attention to weigh importance of each word relative to others
- Helps the model focus on the most relevant parts of the input text
- “ **Explain large language models to me like I am five years old**”
- “ **large language model**” and “ **five years old**” are important group of words

STEP 4 – PREDICTION

- Based on context, the model predicts the next word or token
- Utilizes previous tokens and attention to ensure logical flow
- This is done for one token at a time until the sequence is complete
- “ **Imagine you have a super-smart robot friend who has read every ____.**”
- What is likely next word ? Book or Bank.

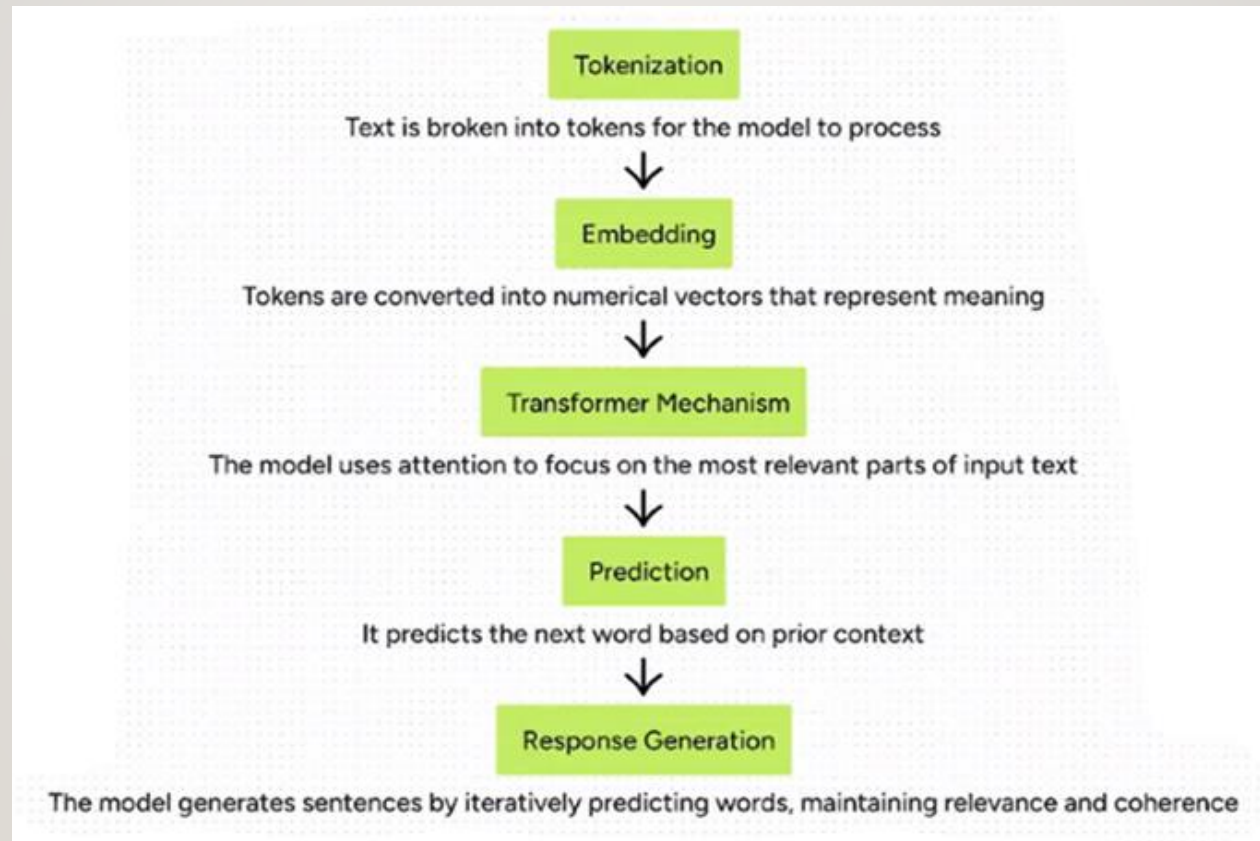


STEP 5 – RESPONSE GENERATION



- The predicted tokens are converted back into human-readable text
- The final result is a coherent and context-aware response
- This response is what the user sees
- **“ Imagine you have a super-smart robot friend who has read every book, story, and conversation ever written.”**
- When one asks a question, it looks like all the words it has learned and find the best answer, just like magic!

PUTTING IT ALL TOGETHER



Natural Language Processing : Project List based on Transformer and BERT

- 1.Sentiment Analysis using Hugging Face
- 2.Text Classification using Hugging Face
- 3.Text Summarizations using Hugging Face
- 4.Text to Text (Translate) using Hugging Face
- 5.Question Answering using Hugging Face
- 6.Text to Image using Hugging Face
- 7.Text to Video Synthesis using Hugging Face

