Abstractive Text **Summarization**

Sky Blues

Team Members

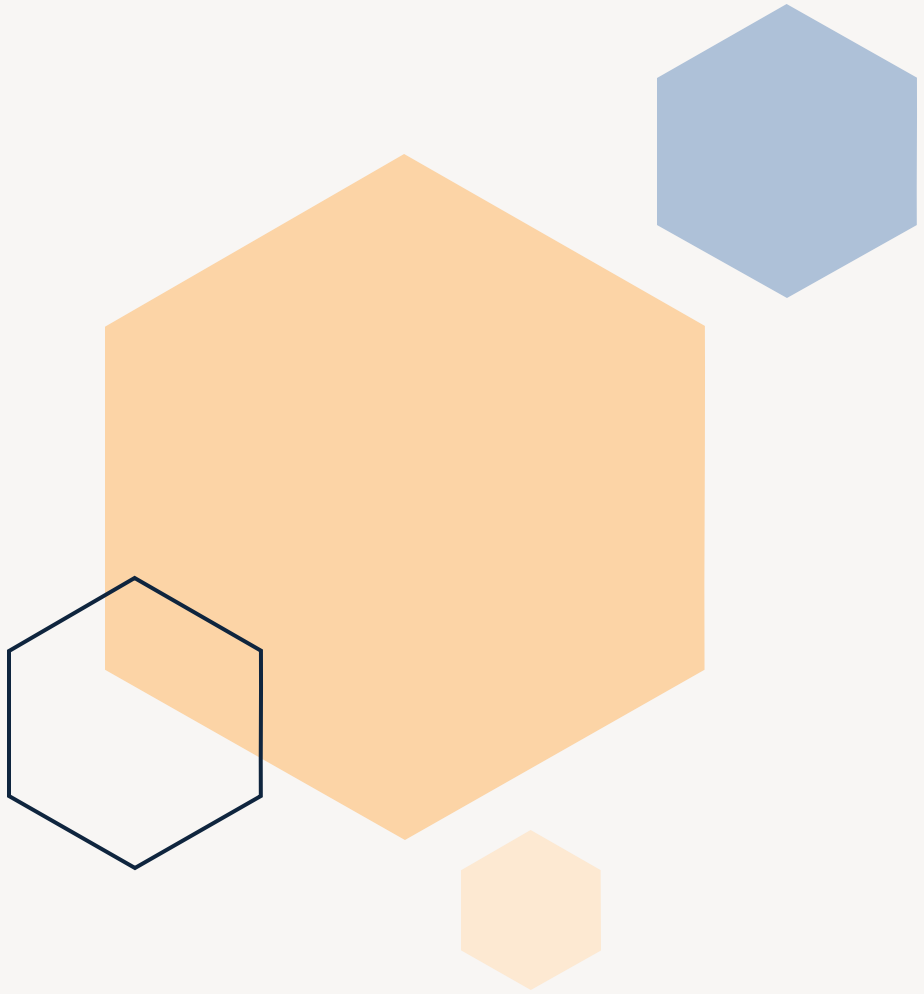
Sky Blues

Ahmed Mohamed	Mostafa Khaled	Sherif Alaa	Mohamed Ahmed	Mohamed Osama
CS	CS	CS	AI	AI

Supervisor : DR/ Sara Swidean

Agenda

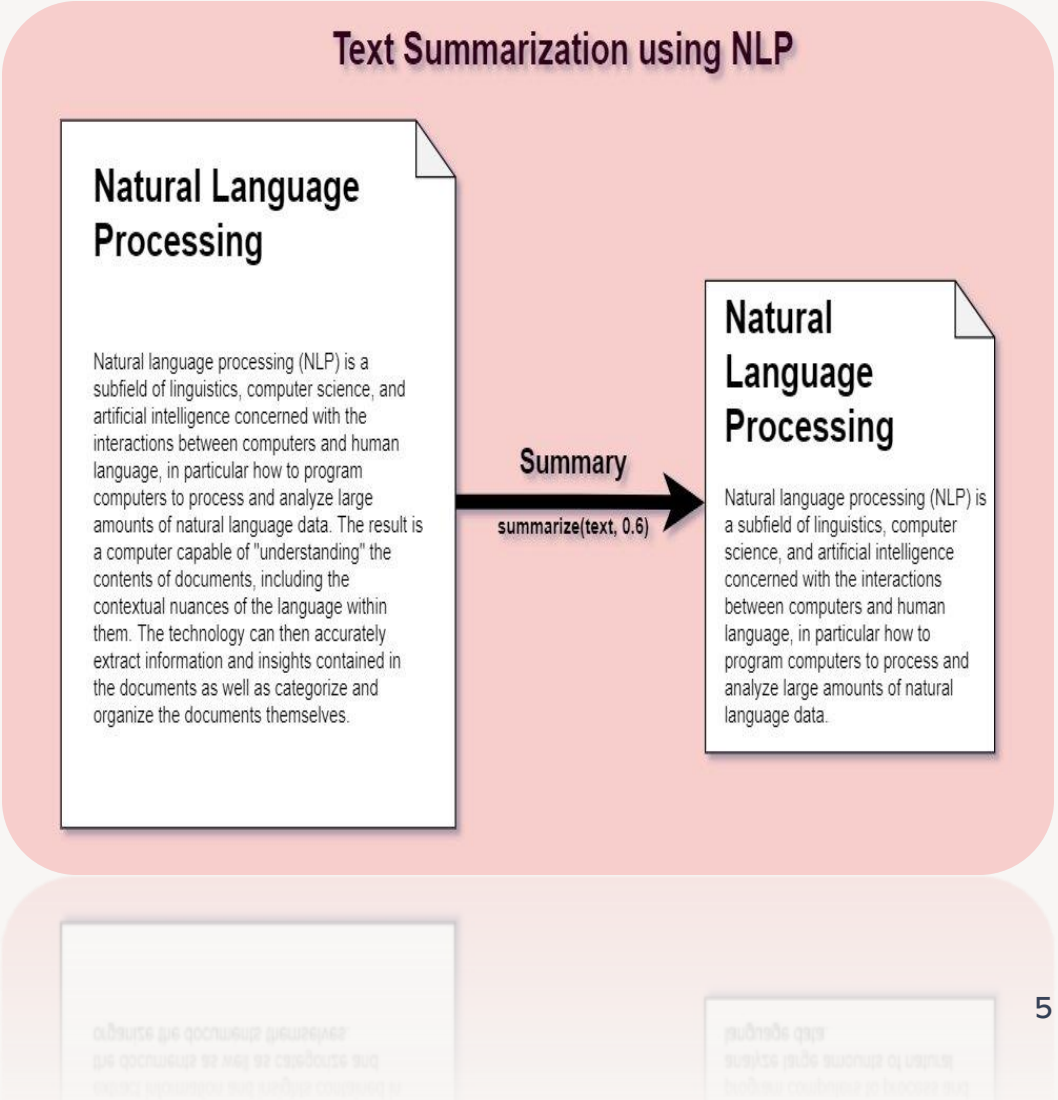

Background and Motivation
Model Architecture
Approach
Training and Evaluation
Results and Analysis



Background and Motivation

Background

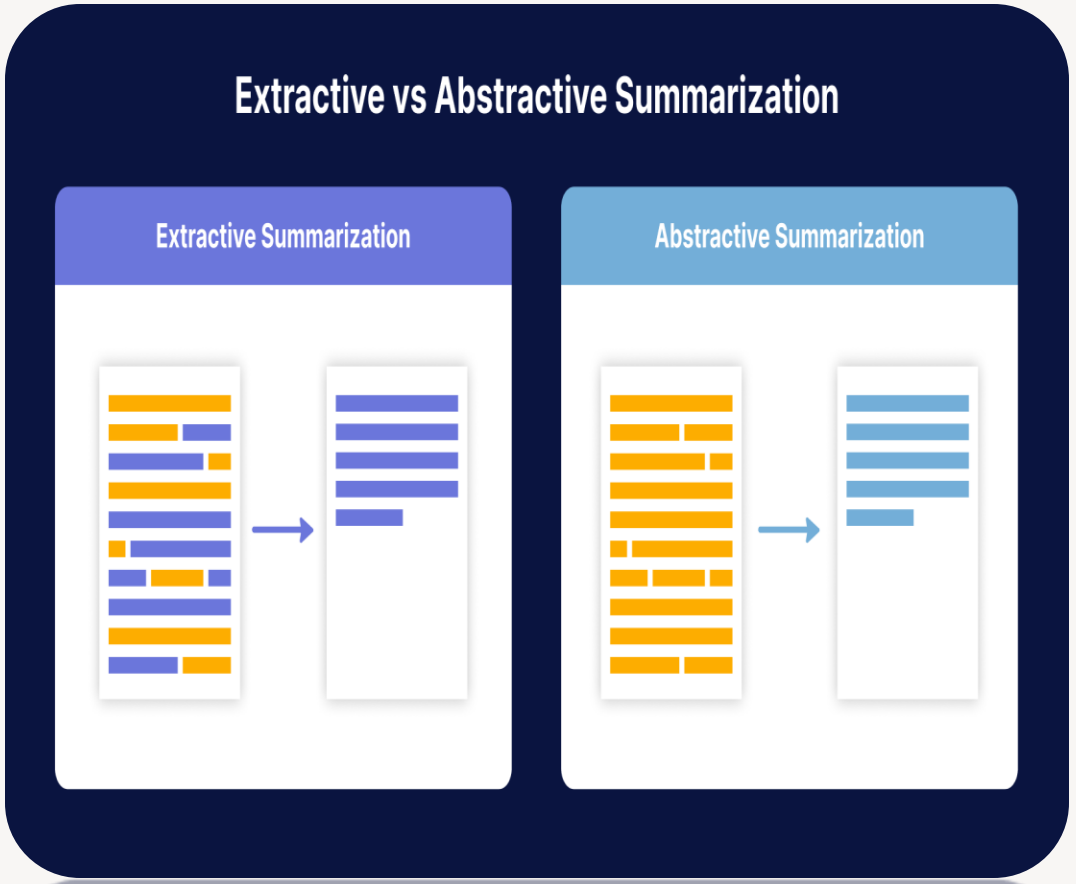
- Text summarization:
  text summarization is the process of automatically creating a shorter version of a longer text by extracting the most important information and preserving the meaning of the original text



Types of Text Summarization

- Extractive, where important sentences are selected from the input text to form a summary. Most summarization approaches today are extractive in nature.

- Abstractive, where the model forms its own phrases and sentences to offer a more coherent summary, like what a human would generate. This approach is more appealing, but much more difficult than extractive summarization.



Advancing abstractive summarization techniques is important for several reasons:

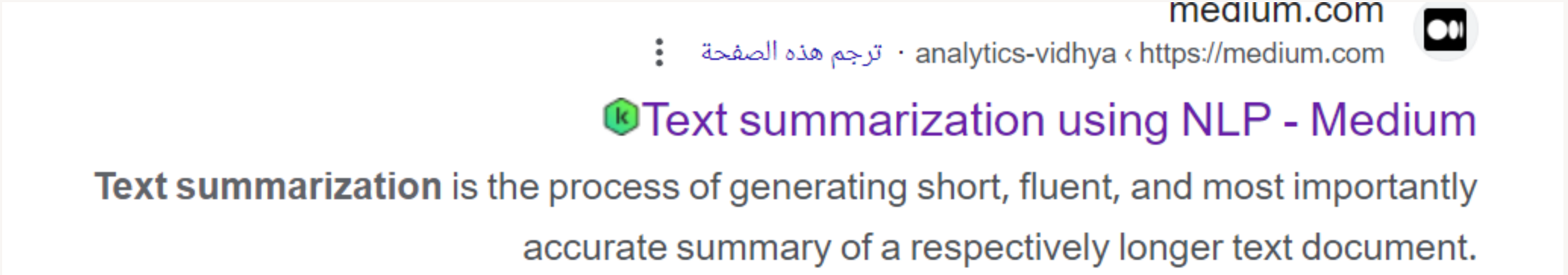
1. Efficiency: Abstractive summarization can help individuals and organizations quickly extract important information from large amounts of text. Advancements in abstractive summarization techniques can improve the efficiency and accuracy of these summaries, saving time and increasing productivity.
2. Accessibility: Abstractive summarization can make information more accessible to people who may not have the time or ability to read through large amounts of text. Advancements in abstractive summarization techniques can help to create more accurate and informative summaries, making information more accessible to a wider audience.
3. Innovation: Advancements in abstractive summarization techniques can lead to new applications and use cases for the technology. For example, more accurate and informative summaries could be used to train machine learning models, or to generate personalized recommendations based on a user's interests.

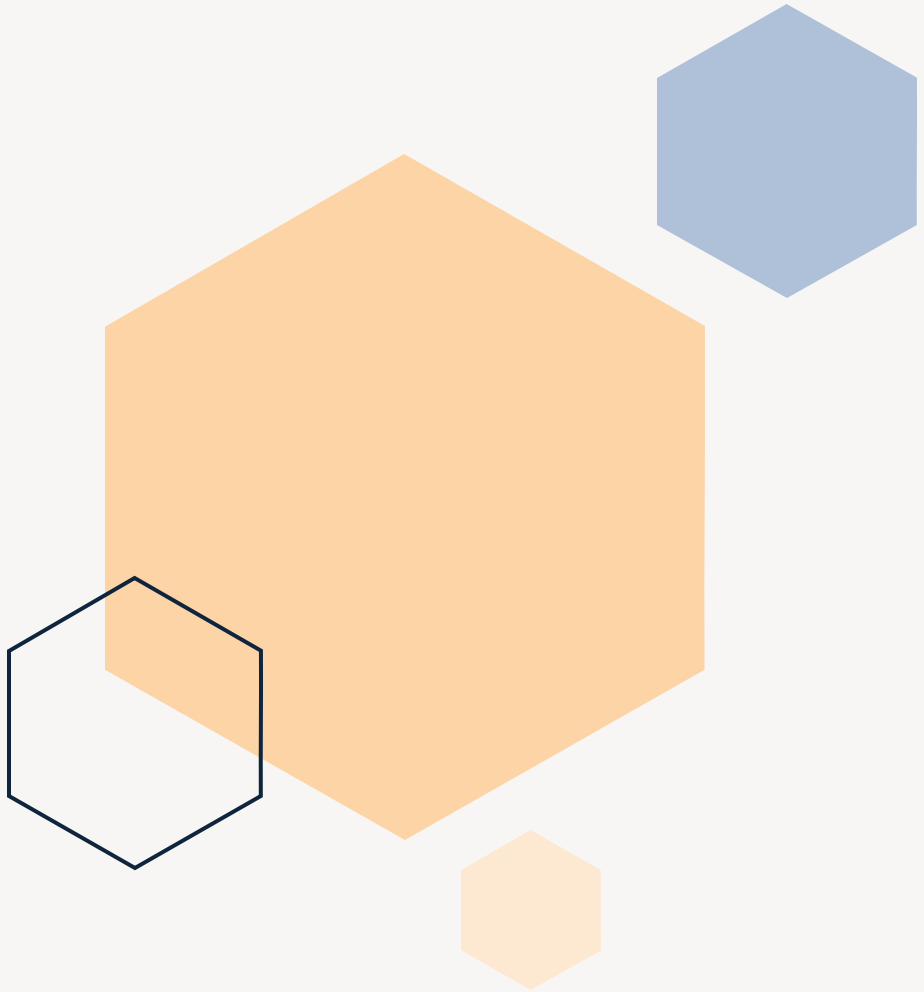
Motivation behind the competition

The competition aims to encourage the development and advancement of abstractive summarization techniques in natural language processing , which are crucial for efficiently processing and understanding vast amounts of information. Advancements in abstractive summarization techniques have significant implications for various domains, including news and document summarization, business intelligence, and social media analysis. By improving the accuracy and efficiency of summarization, these techniques can help individuals and organizations access important information more quickly and easily, improve productivity, and make information more widely available to a broader audience.

Real-world applications:

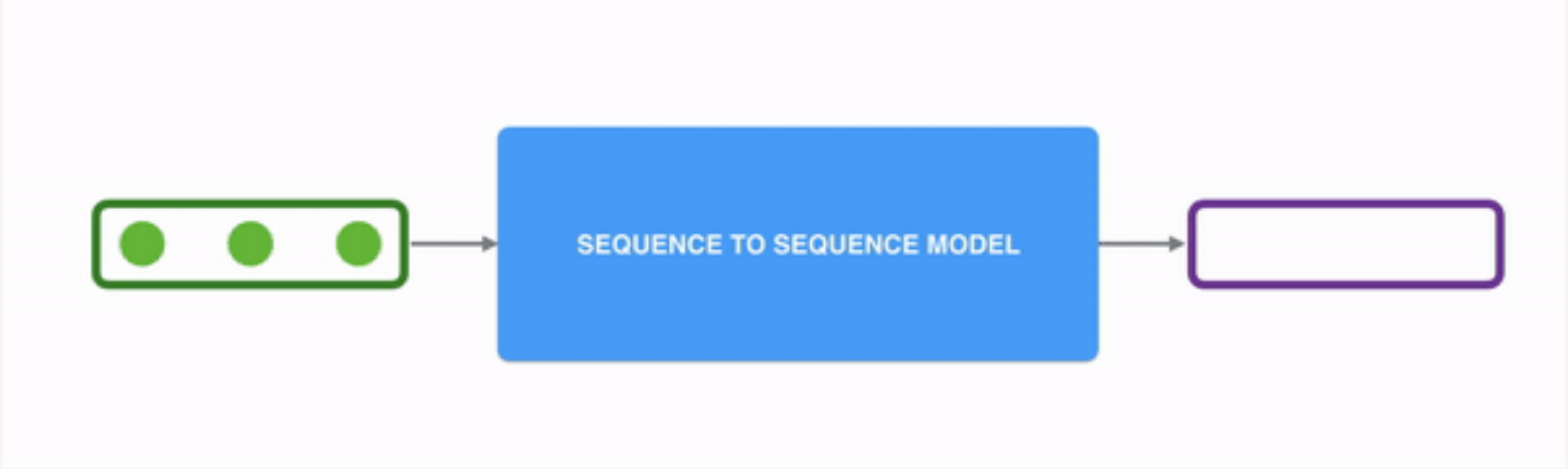
Abstractive summarization techniques have various real-world applications, including news and media, business intelligence, social media, and education. These techniques can save time, increase productivity, improve decision-making, and increase accessibility to information. The potential impact of abstractive summarization techniques is significant and can be felt across a wide range of domains, making them a valuable tool for anyone who needs to process and understand large volumes of text.





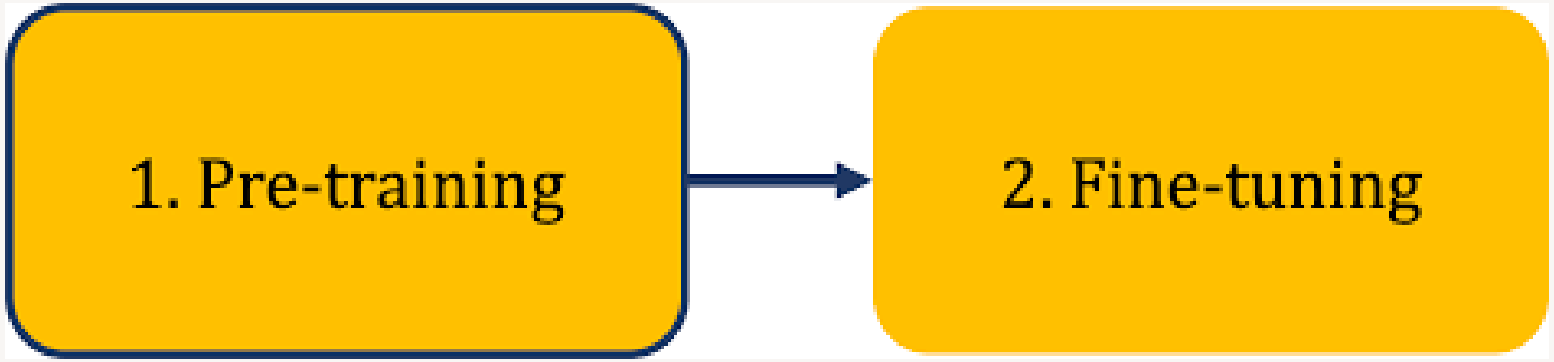
Model Architecture

SEQUENCE TO SEQUENCE MODELS



Architecture of the pre trained models (In general)

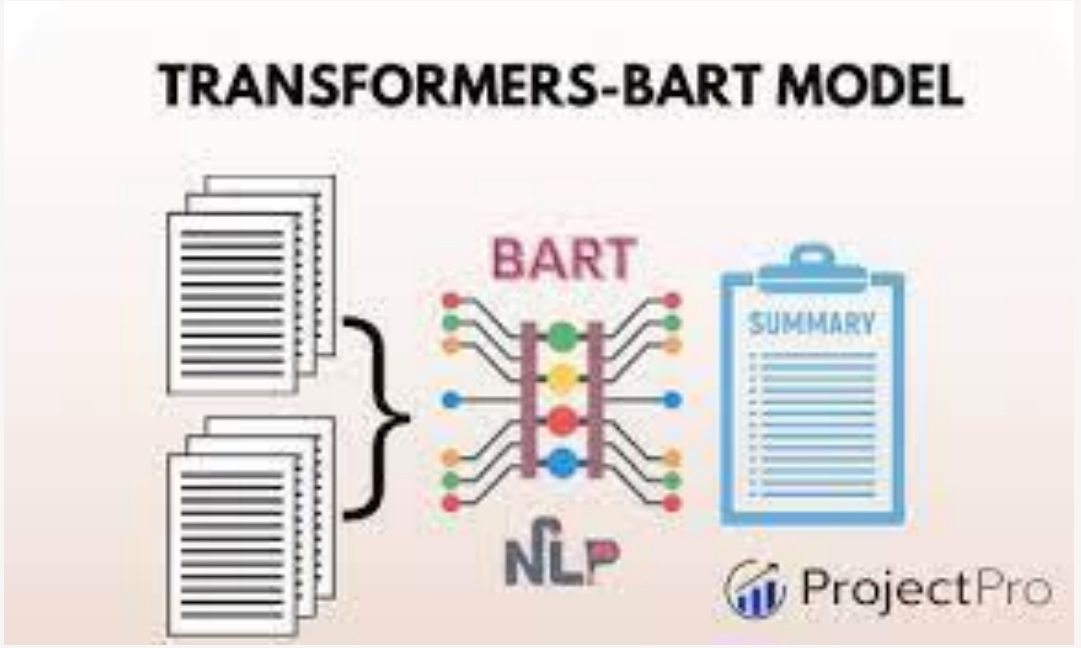
Pretraining and finetuning are two phases involved in training deep learning models, particularly in the context of transfer learning and pre-trained models.



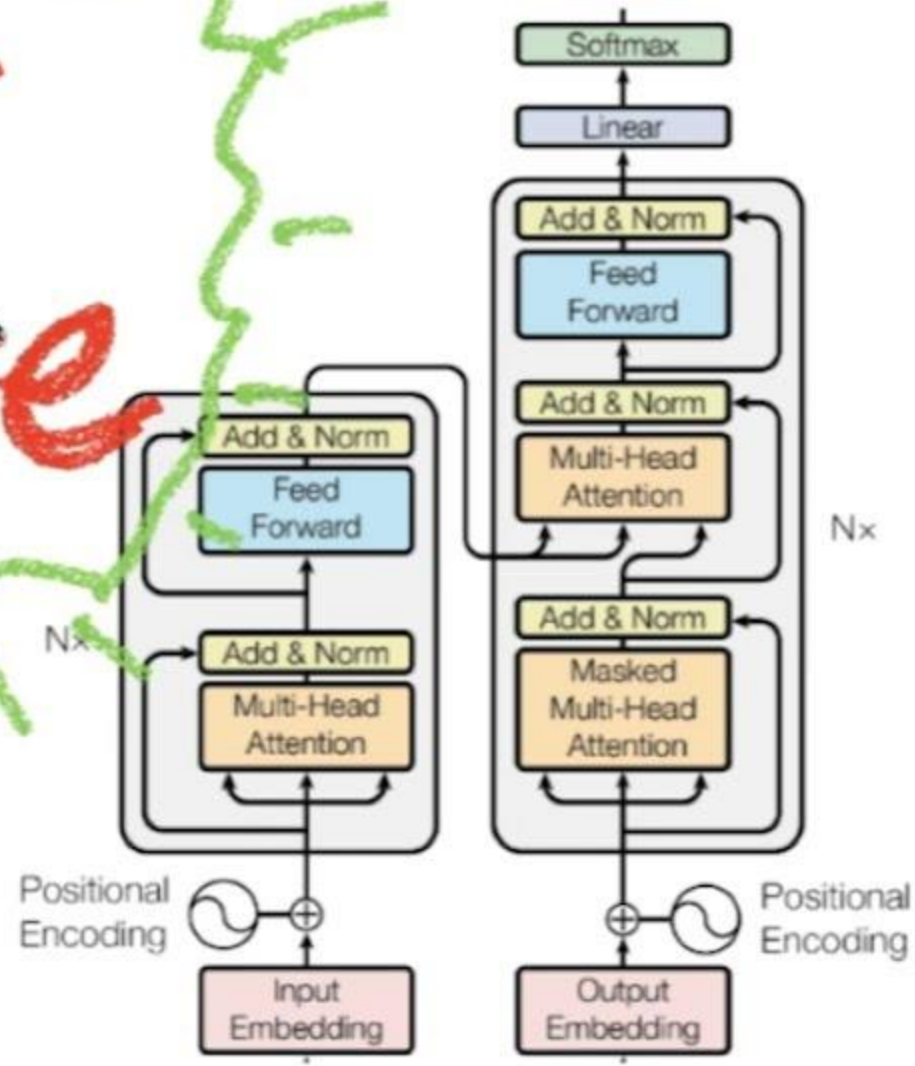
mBART Architecture

In this project we used mBART Large 50 transformer

- The mBART Large 50 model is a pre-trained neural network architecture that can be fine-tuned for abstractive summarization. It is based on the transformer architecture and was developed by Facebook AI Research.
- The model consists of an encoder-decoder framework, which is a common architecture used in sequence-to-sequence models for natural language processing tasks like abstractive summarization.



Transformer Architecture Explained!



Text summarization

Transformer Components

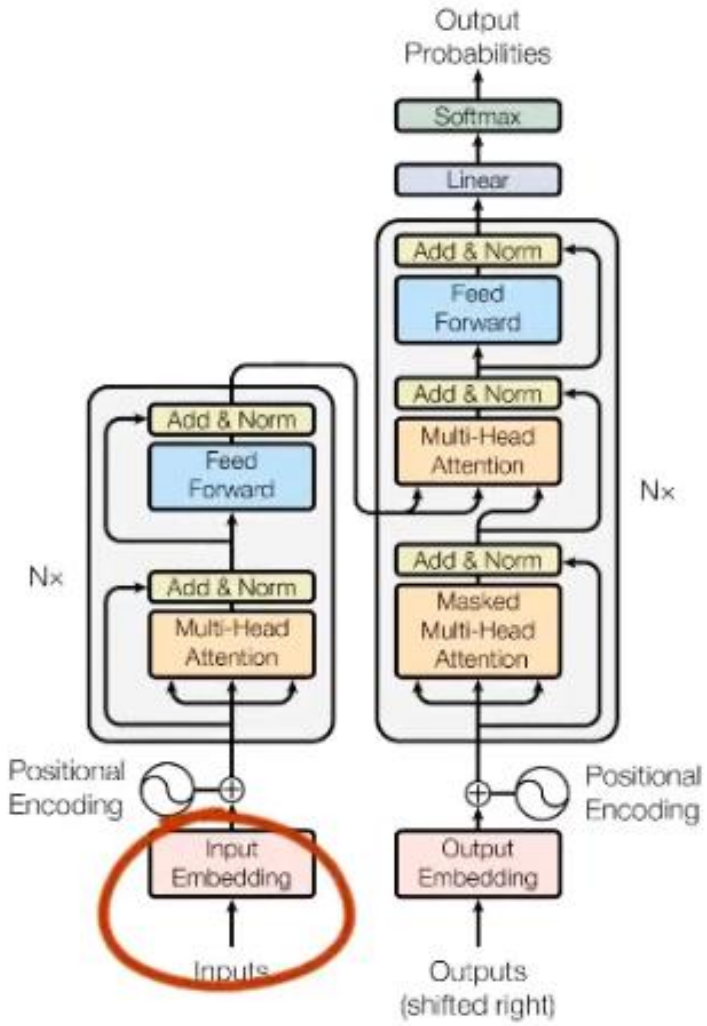
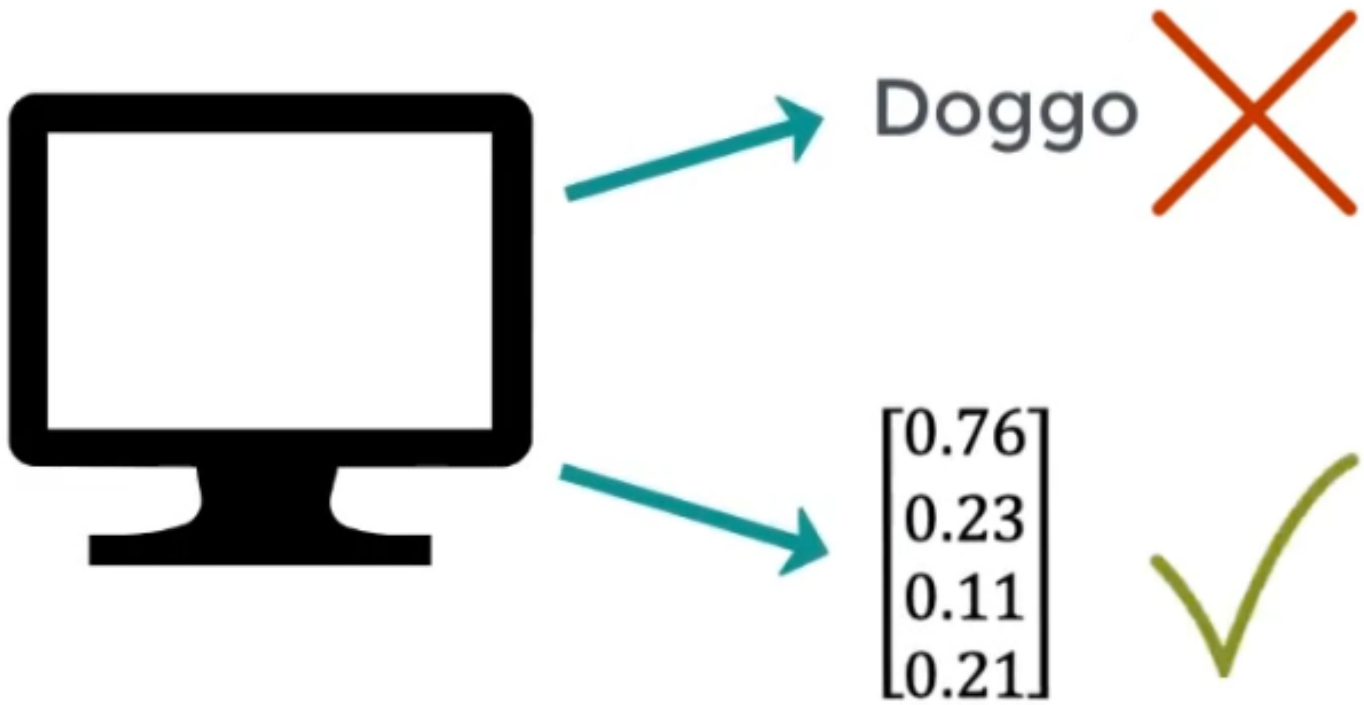


Figure 1: The Transformer - model architecture.

Text summarization

Transformer Components

Input Embedding



Transformer Components

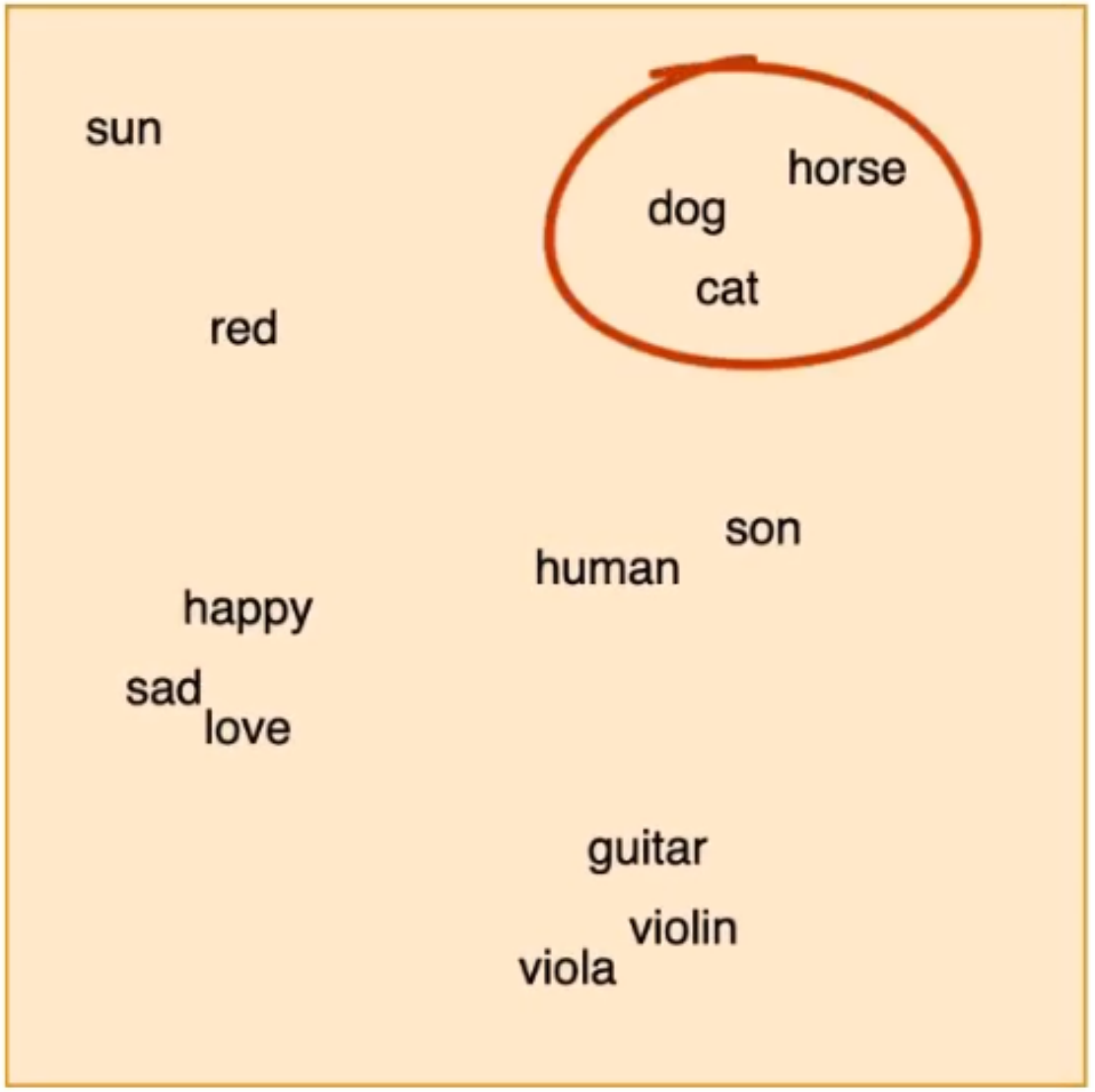
Input Embedding

learns

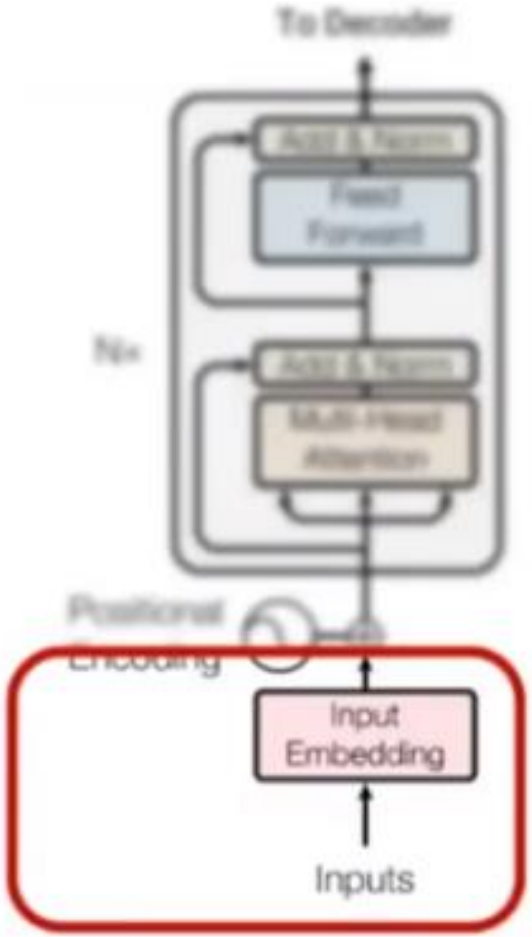
Transformer Components

Input Embedding

learns

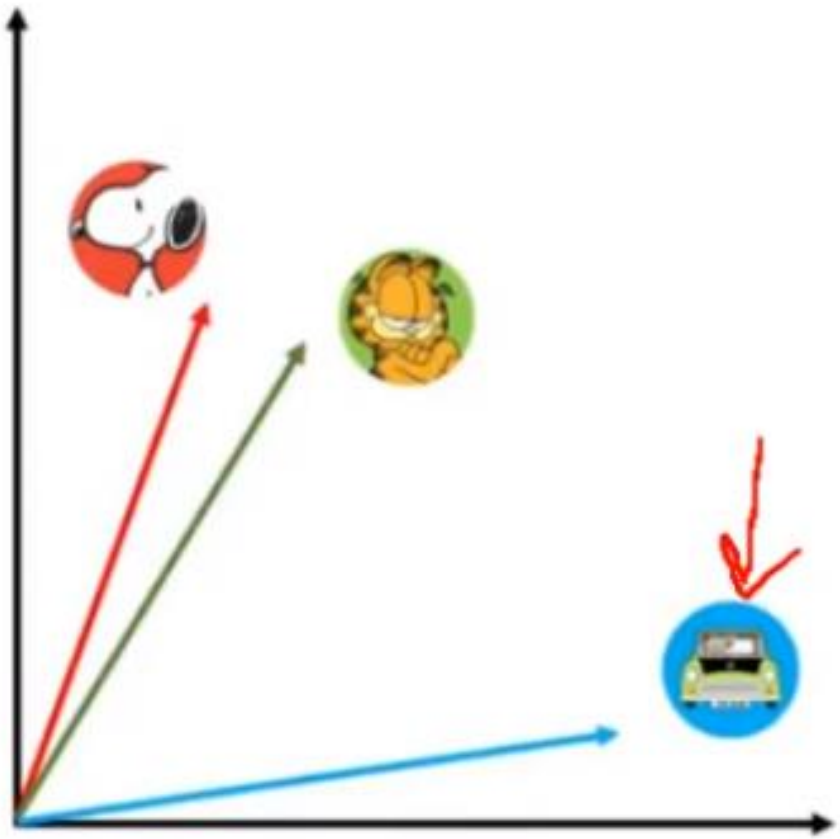


Encoder



Word Embeddings

Key Idea: Similar words should have similar representation vectors.



Transformer Components

Input Embedding

dog →

→ $\begin{bmatrix} 0.37 \\ 0.99 \\ 0.01 \\ 0.08 \end{bmatrix}$

AJ's dog is a cutie

AJ looks like a dog

Transformer Components

Positional Encoder :vector that gives context based on position of word in sentence

AJ's dog is a cutie → Position 2

AJ looks like a dog → Position 5

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Transformer Components

Positional Encoder :vector that gives context based on position of word in sentence

Embedding of "Dog"

Vector Encoding of position in sentence

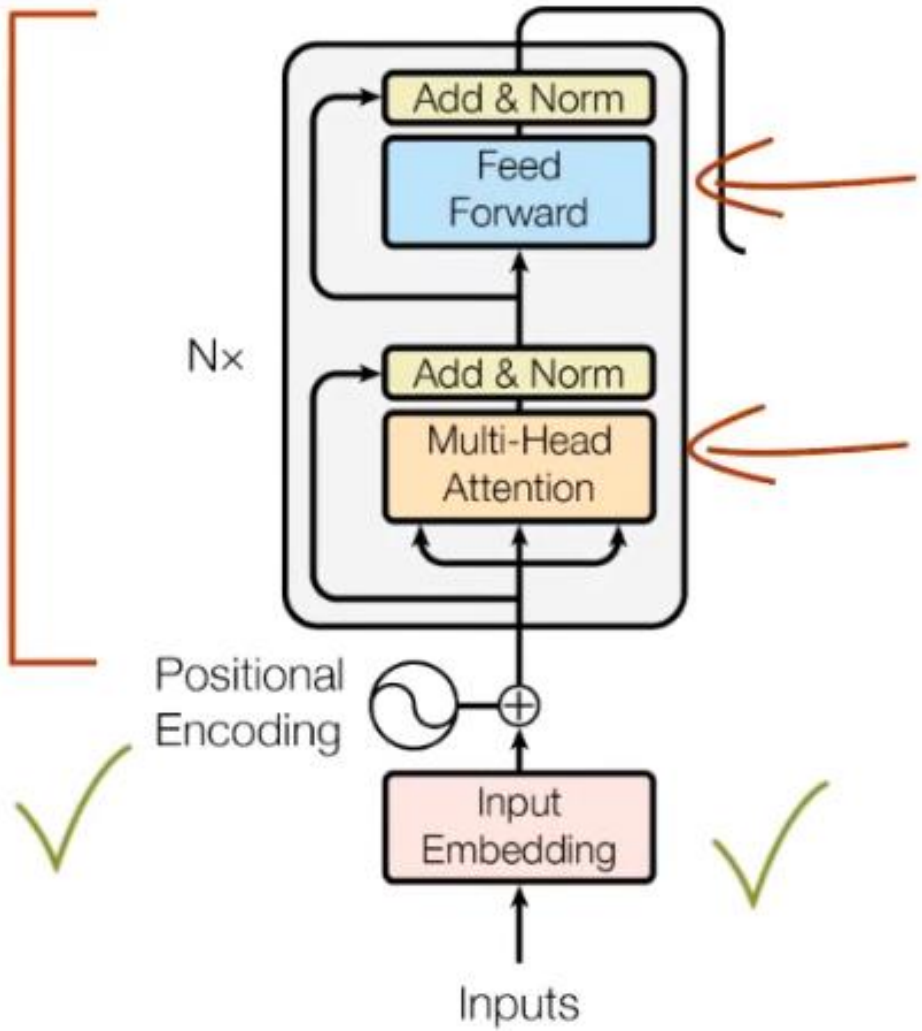
Embedding of Dog (with context info)

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Transformer Components

Encoder Block



Transformer Components

Attention : What part of the input should we focus?

The → The big red dog

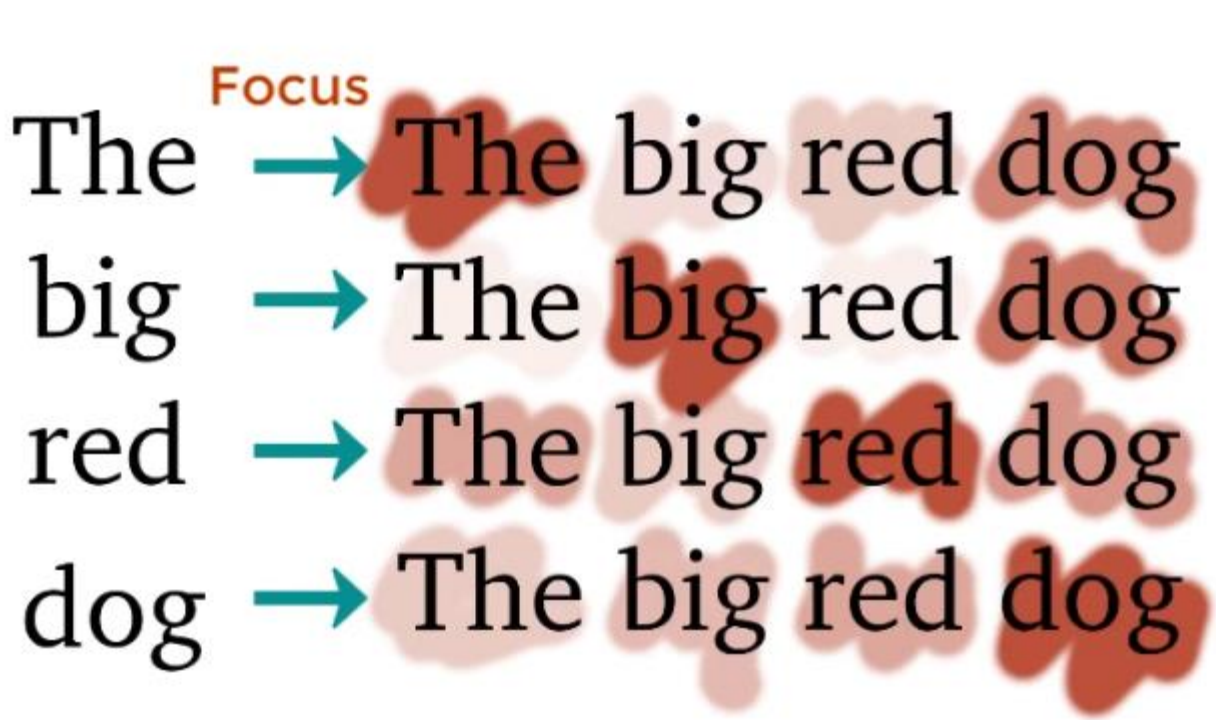
big → The big red dog

red → The big red dog

dog → The big red dog

Transformer Components

Attention : What part of the input should we focus?



Attention Vectors

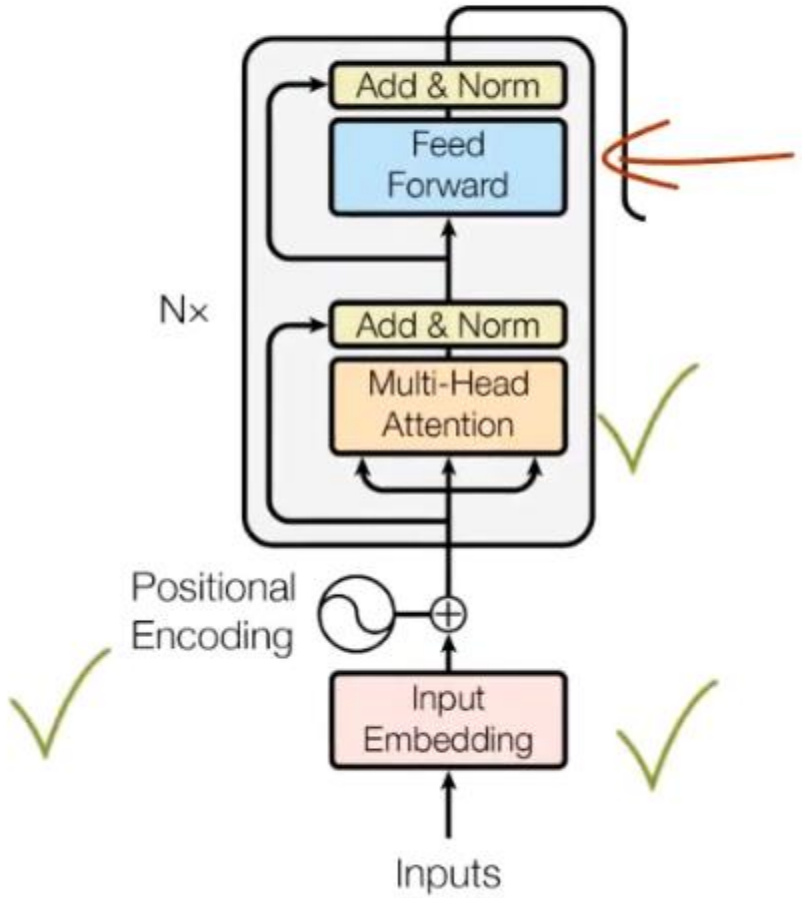
$[0.71 \quad 0.04 \quad 0.07 \quad 0.18]^T$

$[0.01 \quad 0.84 \quad 0.02 \quad 0.13]^T$

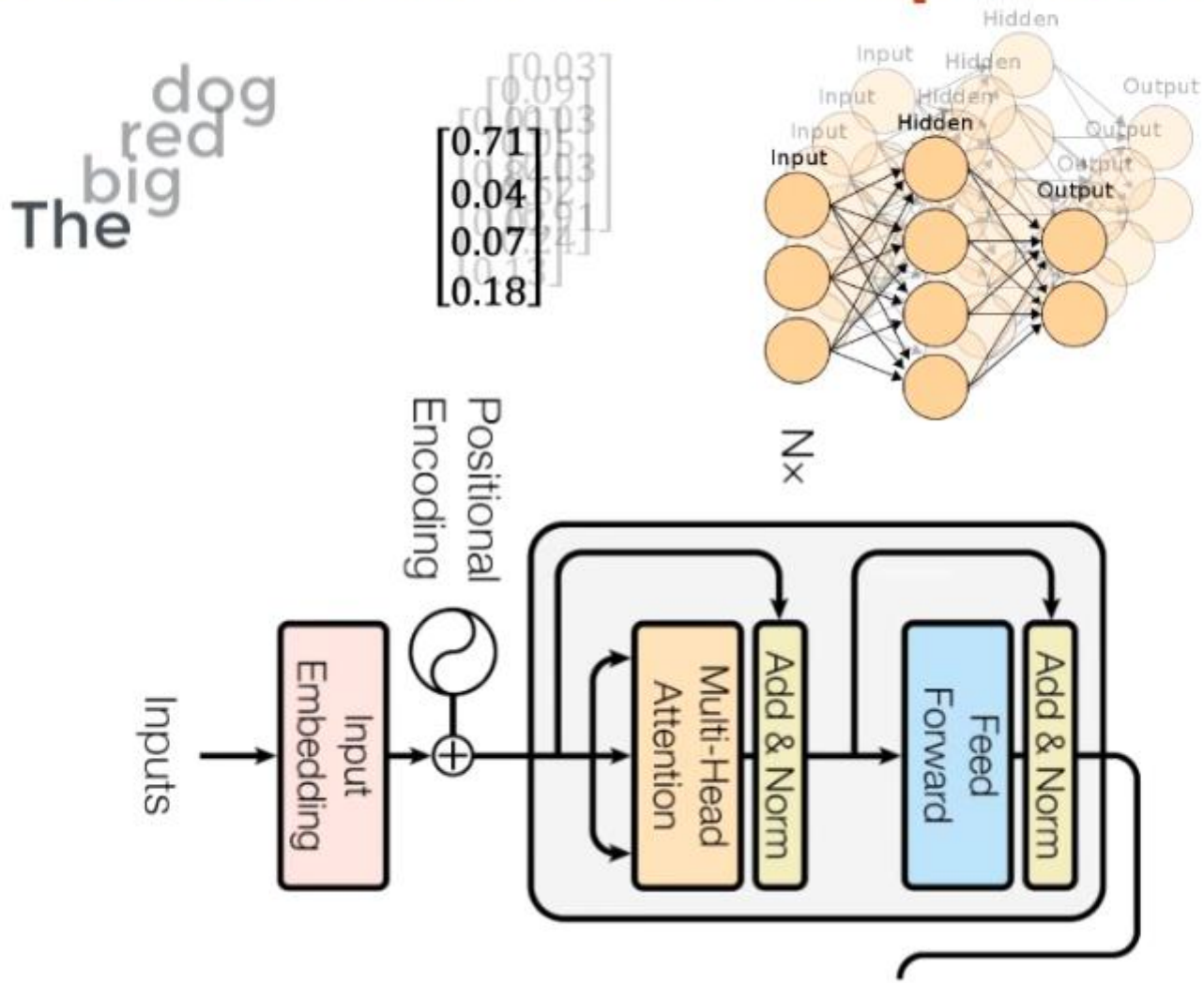
$[0.09 \quad 0.05 \quad 0.62 \quad 0.24]^T$

$[0.03 \quad 0.03 \quad 0.03 \quad 0.91]^T$

Transformer Components



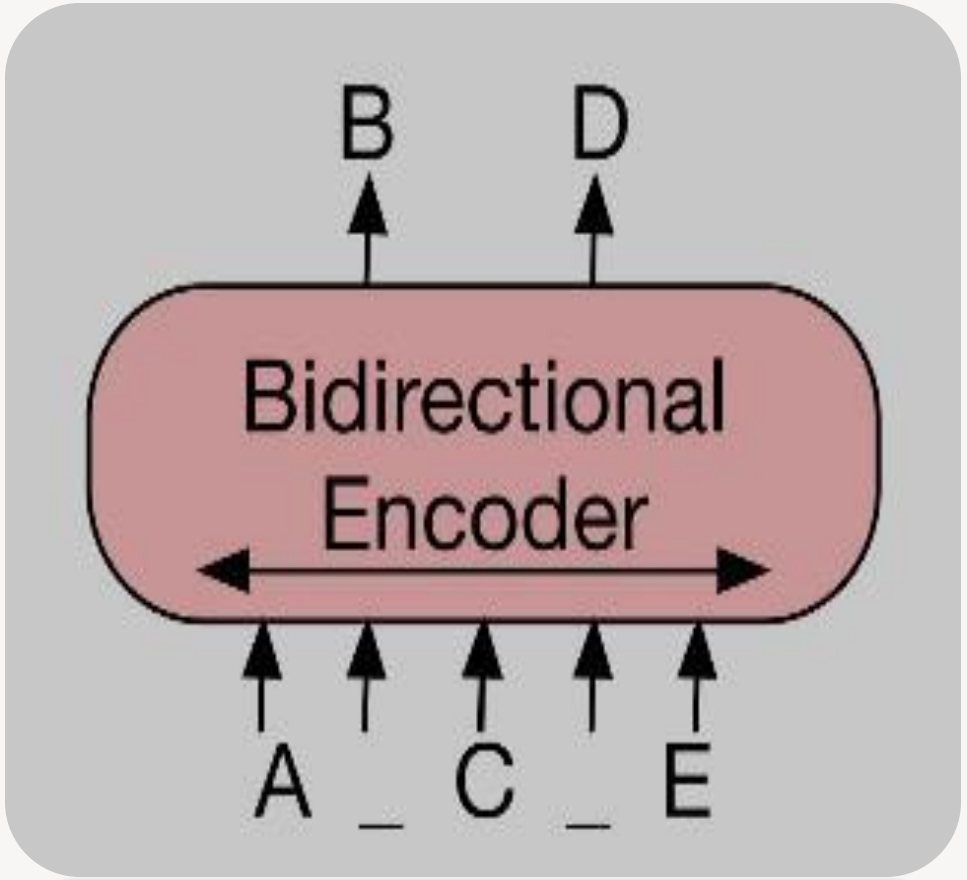
Transformer Components



Text summarization

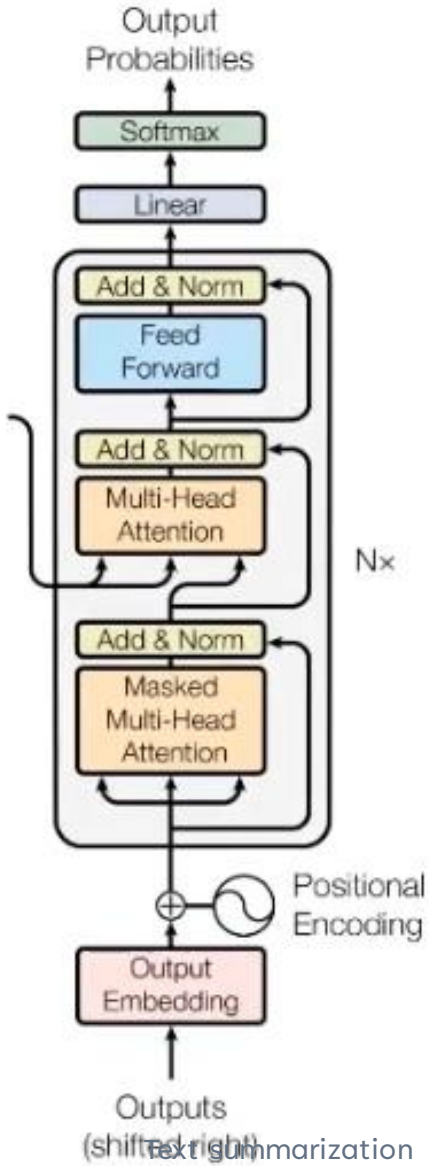
Encoder

- The encoder is responsible for encoding the input text into a set of hidden representations, also known as embeddings. The encoder in the mBART Large 50 model is a multi-layer bidirectional transformer, which means that it processes the input text in both forward and backward directions, allowing it to capture contextual information from both directions.

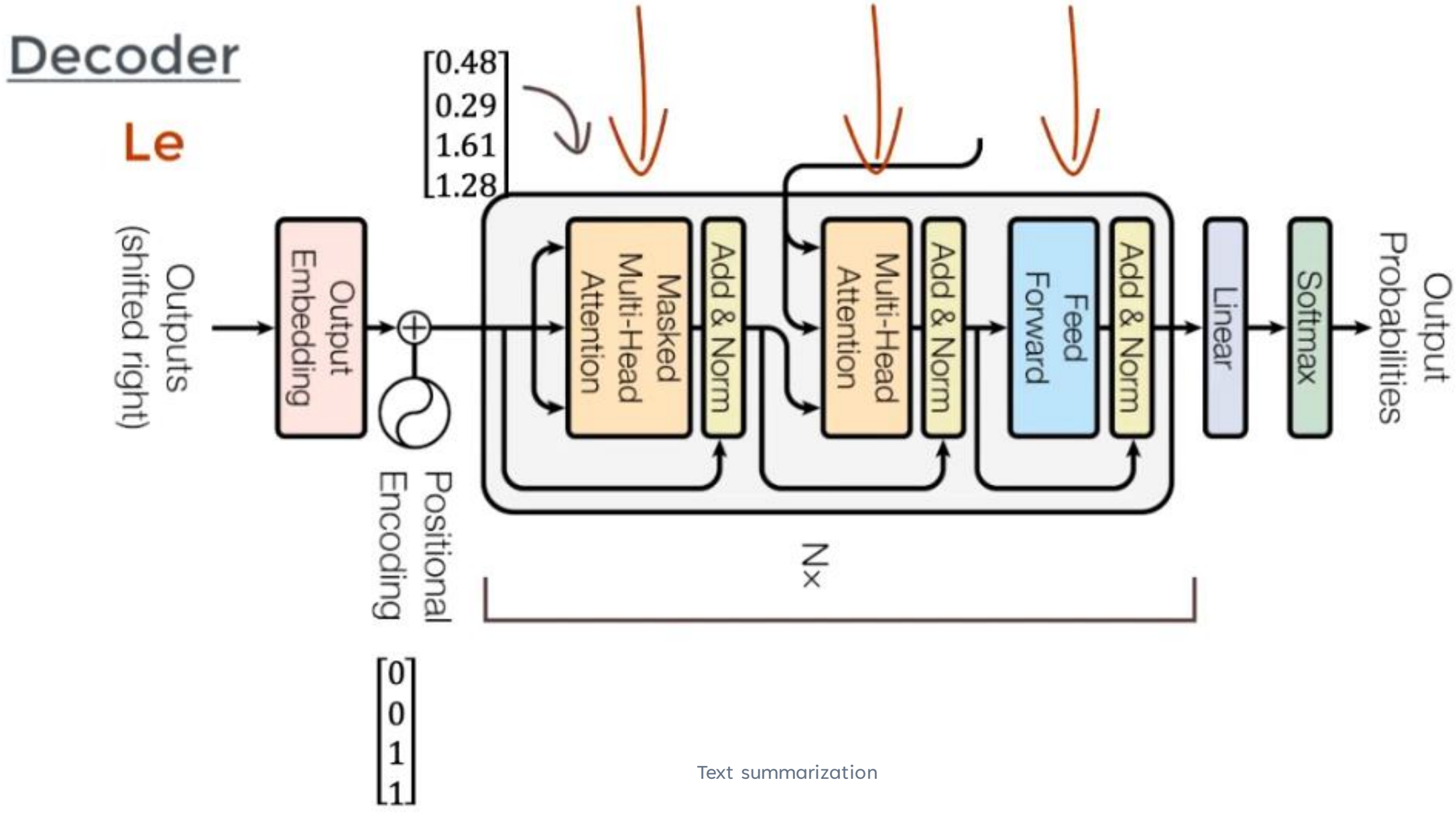


Transformer Components

Decoder



Transformer Components



Decoder

Why the "Mask"?

"You're a wizard, Harry"

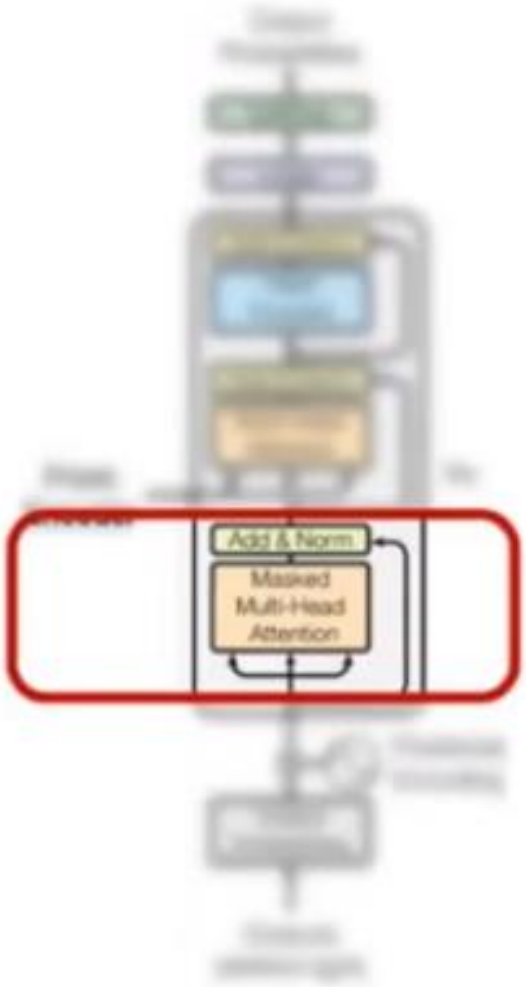
Encoder

Decoder

<s>, Tu, es, un, sorcier, Harry, <eos>

<s>, Tu, es, un, sorcier, Harry, <eos>

Decoder

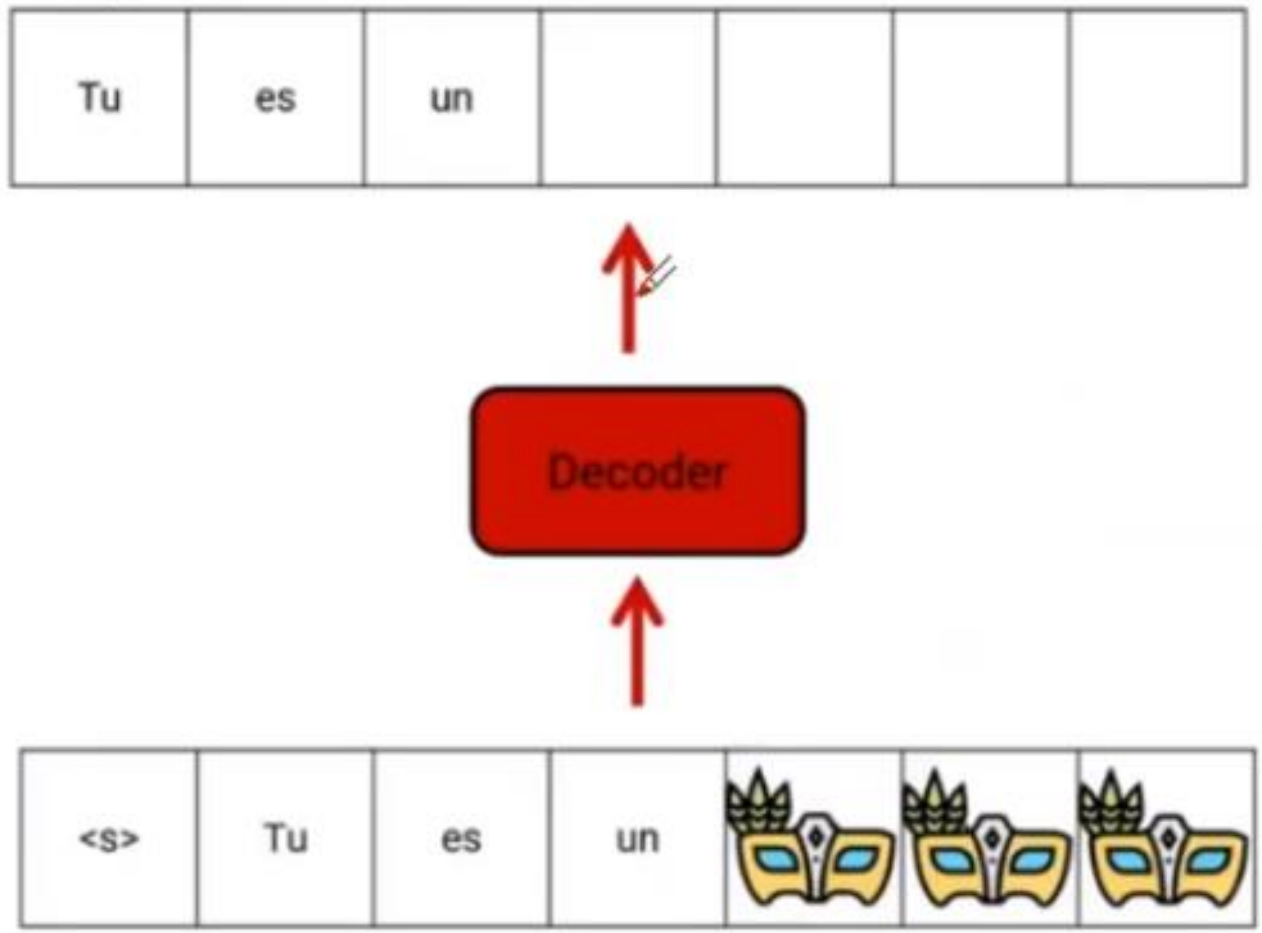


Why the "Mask"?

Decoder

Why the "Mask"?

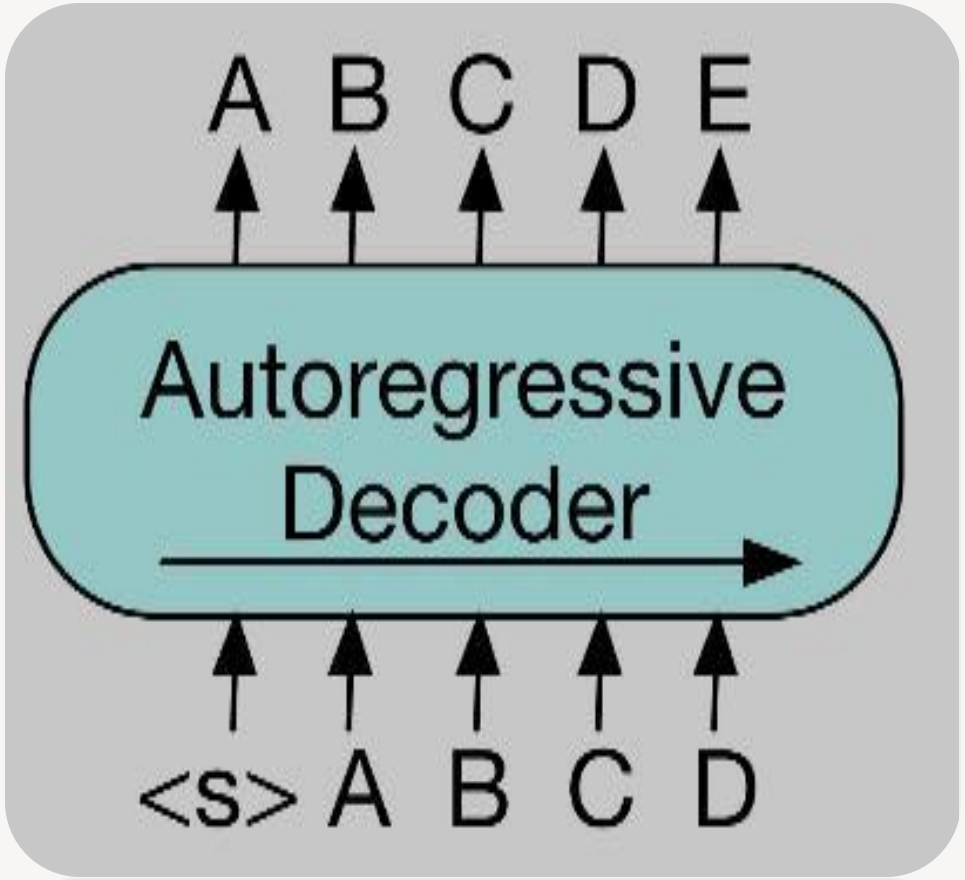
Key Idea: don't let the decoder cheat.



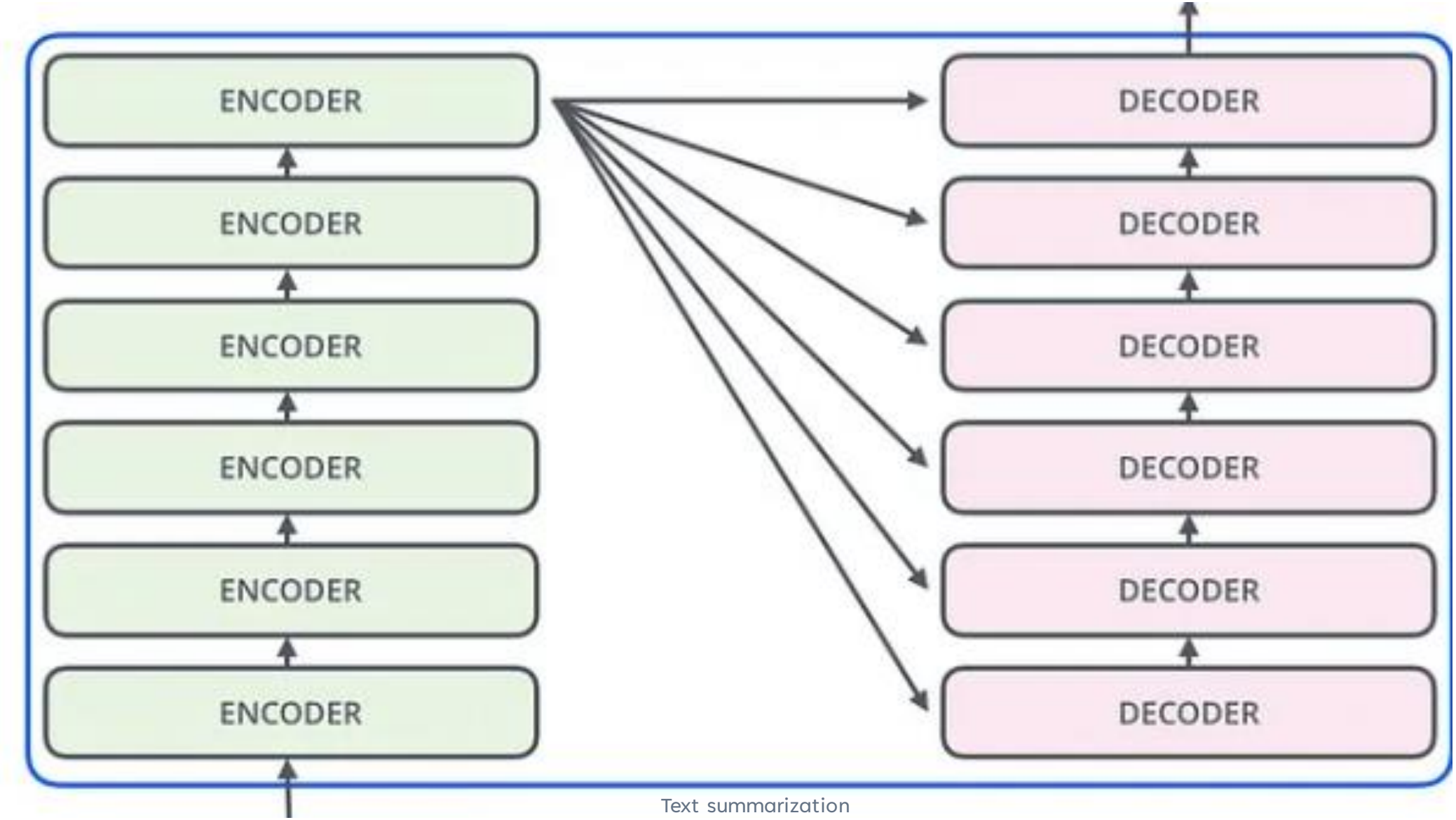
Text summarization

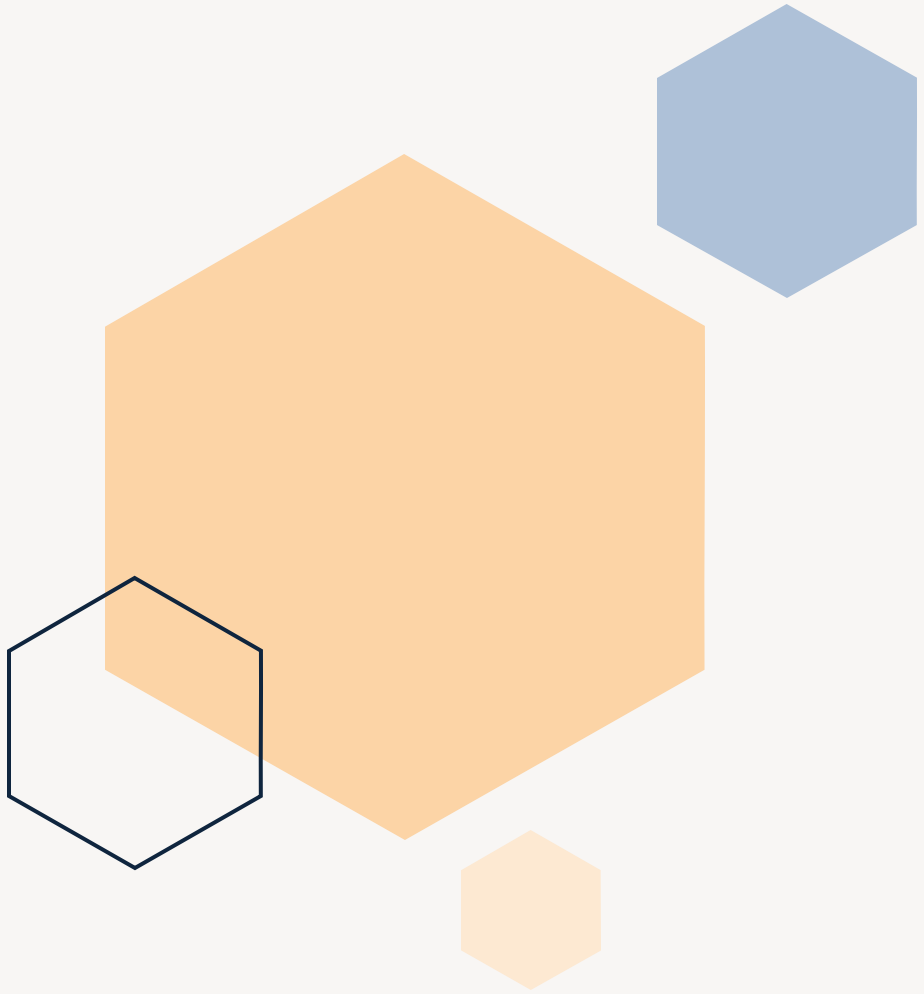
Decoder

- The decoder is responsible for generating the summary based on the hidden representations produced by the encoder. The decoder in the mBART Large 50 model is also a multi-layer transformer, but it is unidirectional, which means that it generates the summary in a single direction. During generation, the decoder attends to the encoder hidden representations and generates the summary word-by-word.



Internally, the Transformer has a similar kind of architecture as the previous models above. But the Transformer consists of six encoders and six decoders.





Approach

Chosen Approach

- The chosen approach is to perform text summarization using the Hugging Face Transformers library.

- We chose the 'facebook/mbart-large 50' model for abstractive summarization ,This model is a pre-trained sequence-to-sequence transformer-based model that has been trained on multilingual data, making it suitable for summarizing text in various languages ,including Arabic.

- The approach involves fine-tuning the base model on the specific summarization task using the provided training data.

- The model is trained to generate concise and coherent summaries given input paragraphs.

- The training process involves optimizing the model's weights based on the evaluation of generated summaries against the reference summaries in the training dataset.

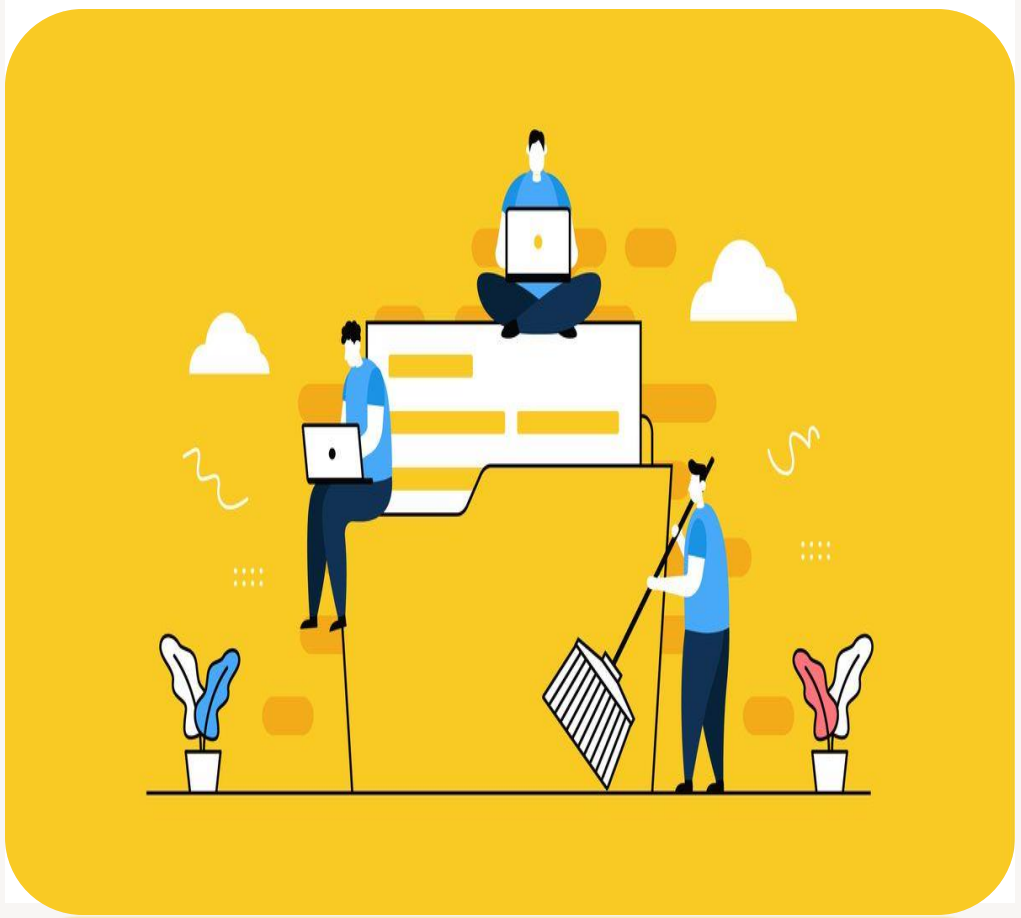
Novel Techniques

- While the code itself does not introduce novel techniques, it leverages the power of the 'facebook/mbart-large-50' model, which is pre-trained on a large amount of multilingual data, This enables the model to capture language patterns and generate high-quality summaries.

- Additionally, the code utilizes the capabilities of the Transformers library, which provides a convenient and efficient framework for fine-tuning and utilizing pre-trained models.

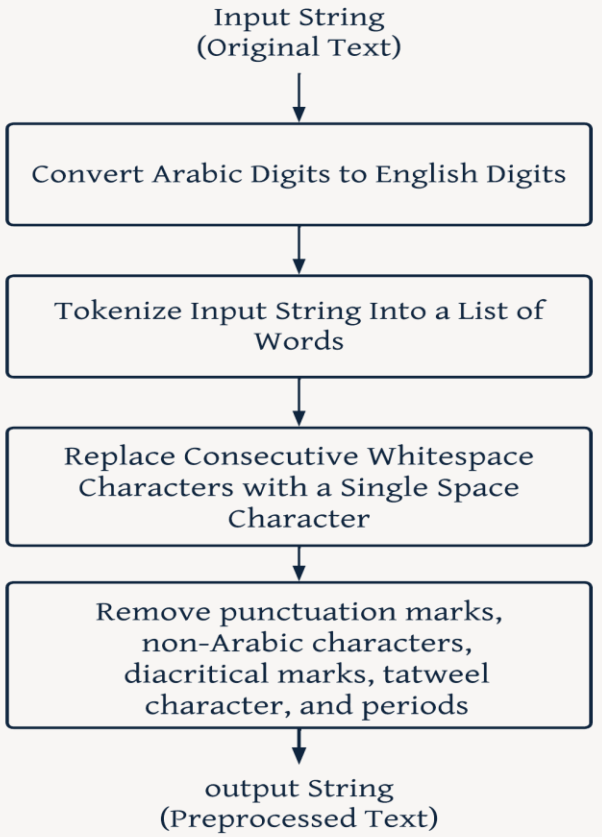
Preprocessing

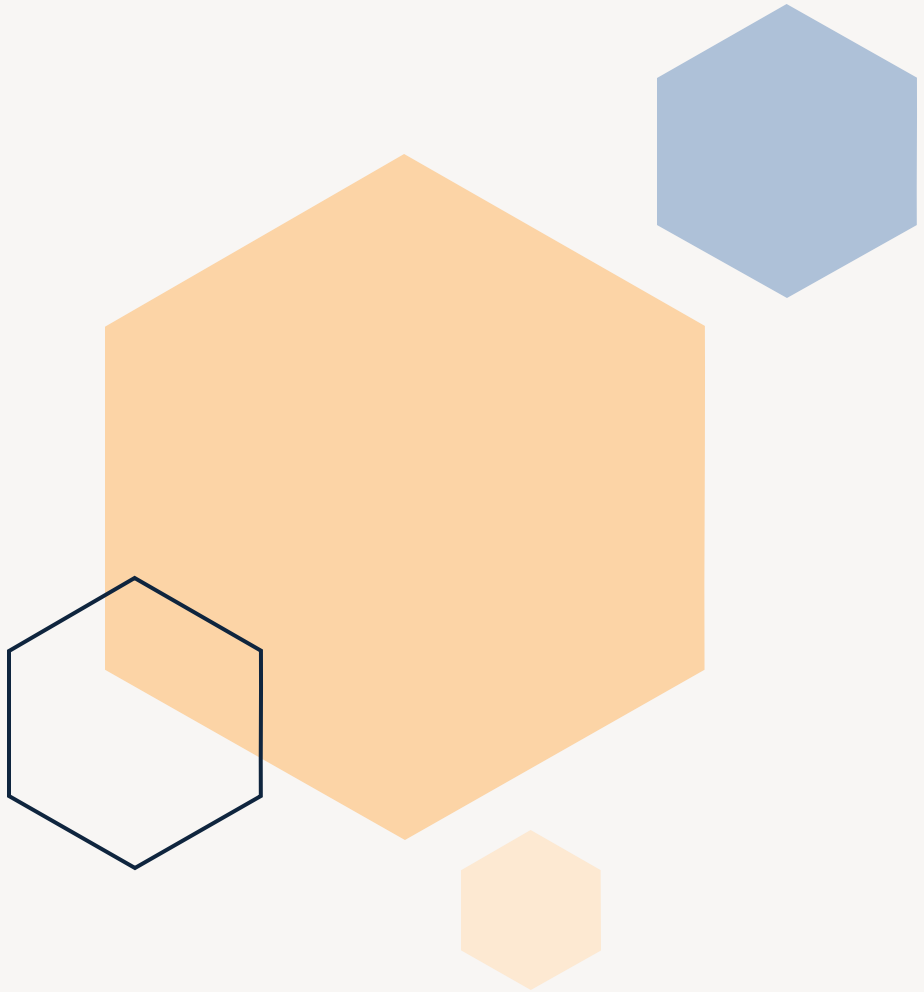
- As we are working in Arabic data the Arabic text has some unique characters and structures that may require special preprocessing techniques to ensure that the model can process the text correctly
- We use translation and cleaning operations to ensure the text is in a suitable format for summarization



Preprocessing Steps:

1. Convert Arabic digits to English digits using the **translate()** function.
2. Tokenize the input string into a list of words.
3. Replace any consecutive whitespace characters in the input string with a single space character to standardize word spacing.
4. Remove all punctuation marks from the list of tokens and join the remaining tokens back into a string separated by spaces.
5. Remove all characters that are not Arabic script letters, digits, whitespace, period, or comma from the input string.
6. Remove all diacritical marks (harakat or tashkeel) from the input string.
7. Remove the tatweel character (-) from the input string. The tatweel character is a long underscore used in Arabic script to extend the length of certain letters or words.
8. Remove all period characters from the input string.

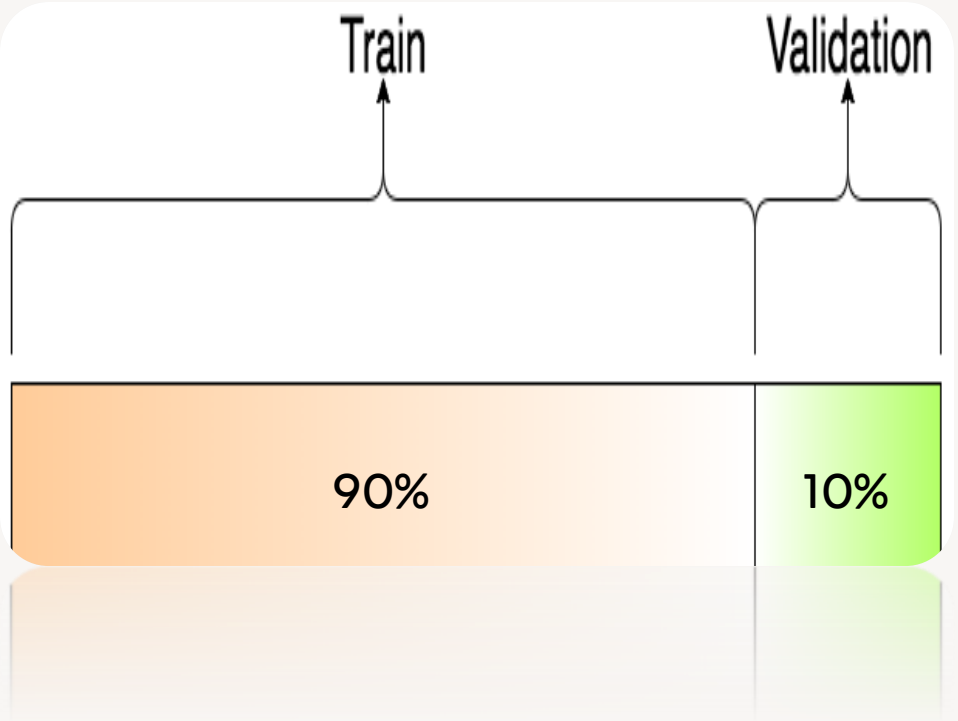




Training and Evaluation

Data preparation

- First, We used labeled validation dataset provided by the competition
- Divided The data into Train and Validation by the percent of :
- 90% For training
- 10% for Validation



Fine-Tuning

- In the fine-tuning process we used Hugging-Face Summarization script
- The following hyperparameters were used during training:
- - learning_rate: 5e-05
- - train_batch_size: 4
- - eval_batch_size: 8
- - seed: 42
- - optimizer: Adam with betas=(0.9,0.999) and epsilon=1e-08
- - lr_scheduler_type: linear
- - num_epochs: 10.0
- --save_strategy "no"
- --load_best_model_at_end True

BART Model for Text Summarization

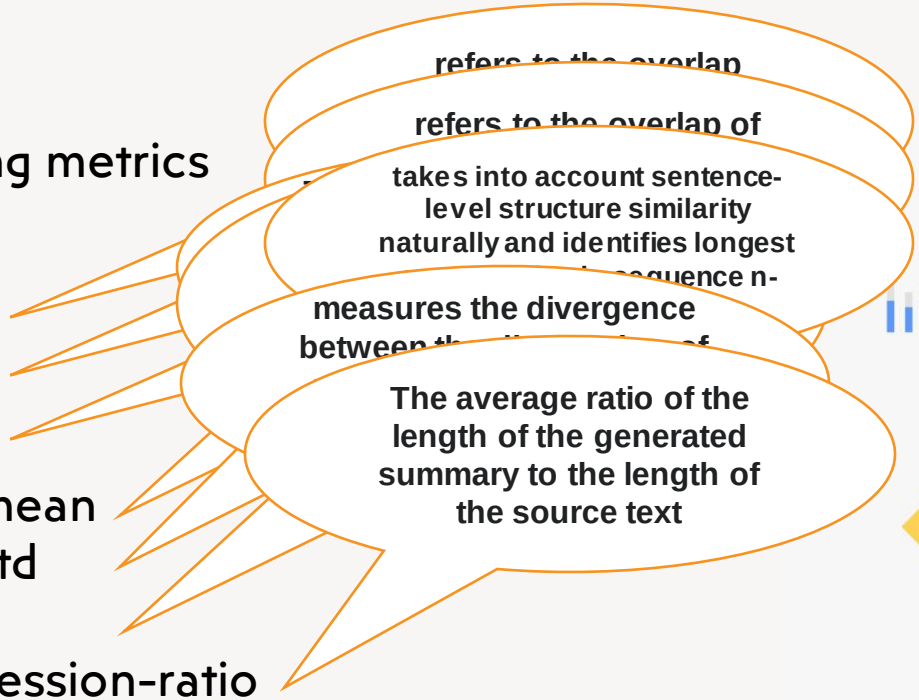
Training Infrastructure

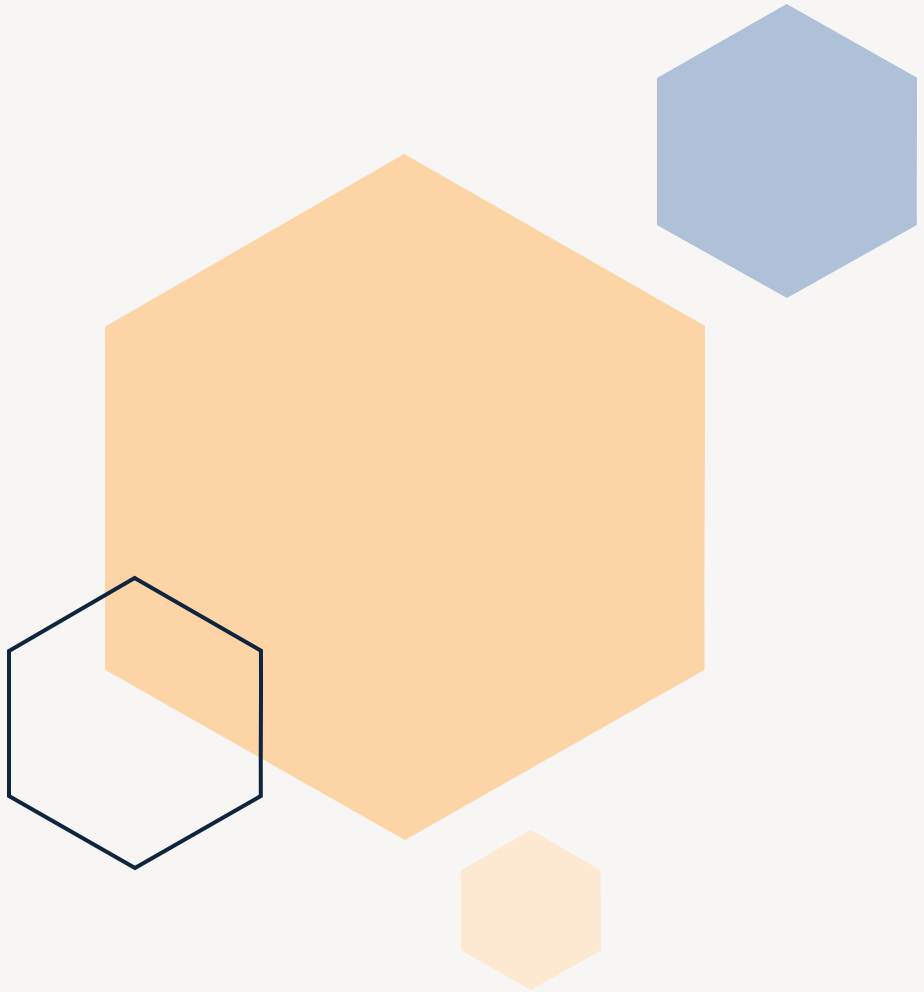
We used both Kaggle and Colab, which are cloud-based platforms that provide free access to GPU and TPU hardware for machine learning tasks

To Assess The Performance

Used The Following metrics

- Training lose
- Eval lose
- Rouge 1
- Rouge 2
- Rouge L
- ss-population-mean
- ss-population-std
- inv-norm-kl-div
- average-compression-ratio

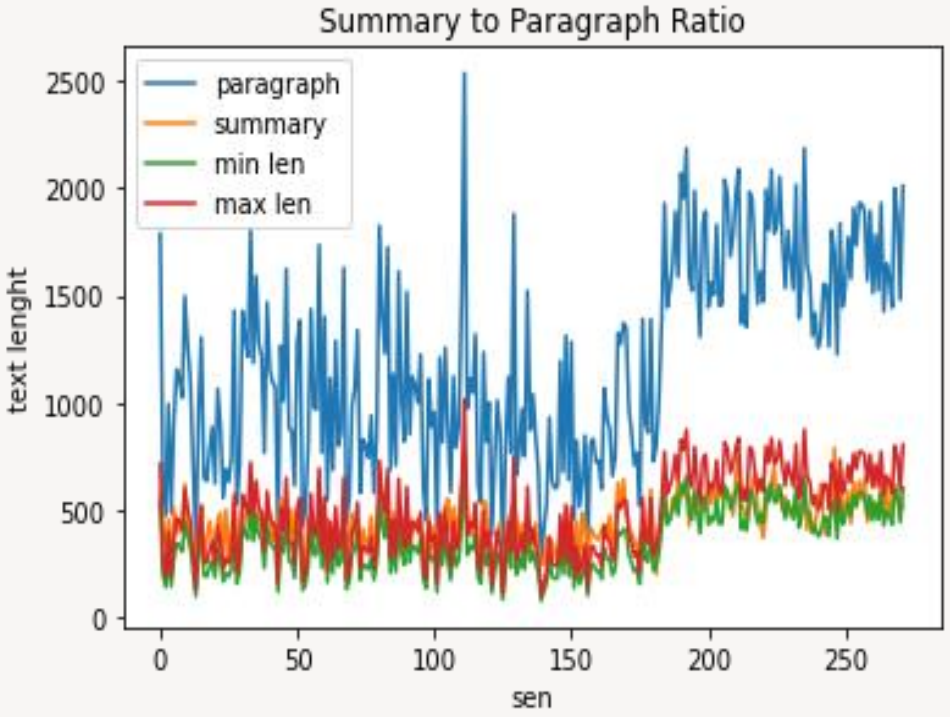




Results and Analysis

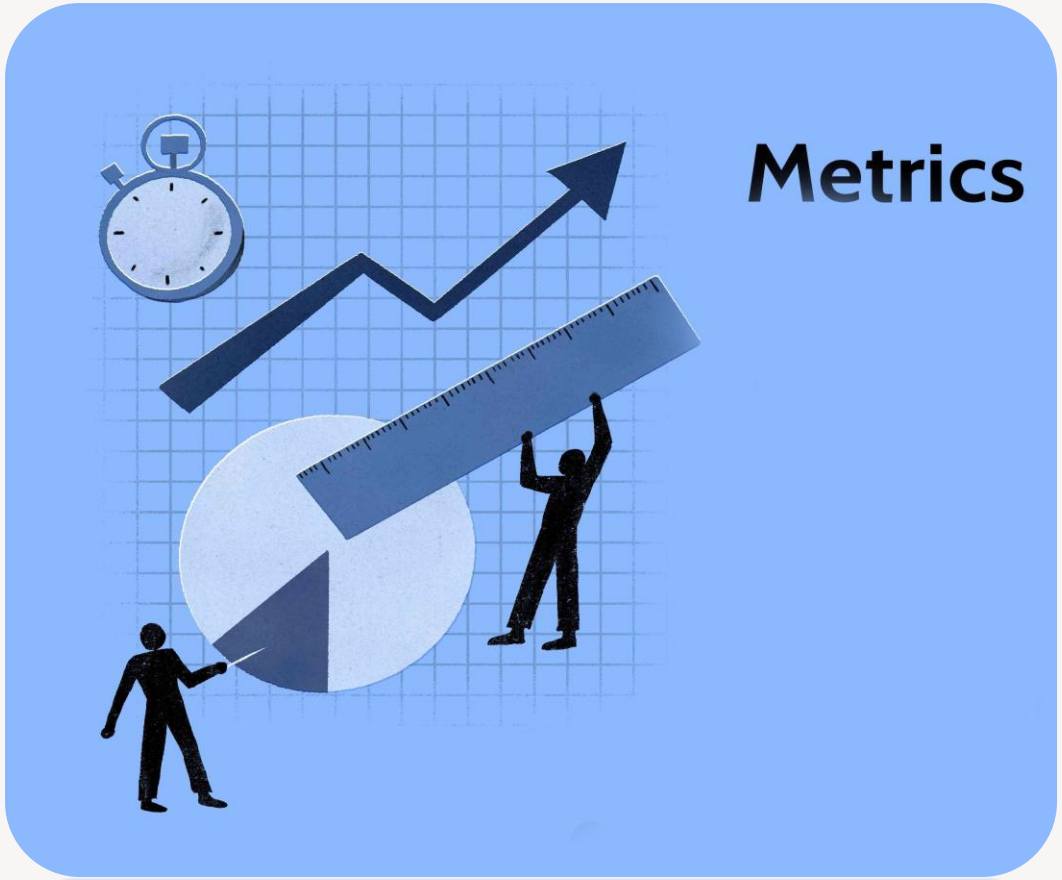
Summary to Paragraph Ratio

- Since we are doing summarization and according to multiple studies and the common sense. The number of words in summary should be between 30%-40% of the words in the original paragraph.
- As a result of setting the min , max length when generating summaries to:
- Max length = 0.4*length of paragraph
- Min length = 0.08*length of paragraph
- We got the following graph



Metrics

- score : 0.836153838
- rouge-l : 0.218532285
- ss-population-mean : 0.821394218
- ss-population-std : 0.087161352
- inv-norm-kl-div : 0.989210544
- average-compression-ratio :0.38710485
- And Finally Rank : 1

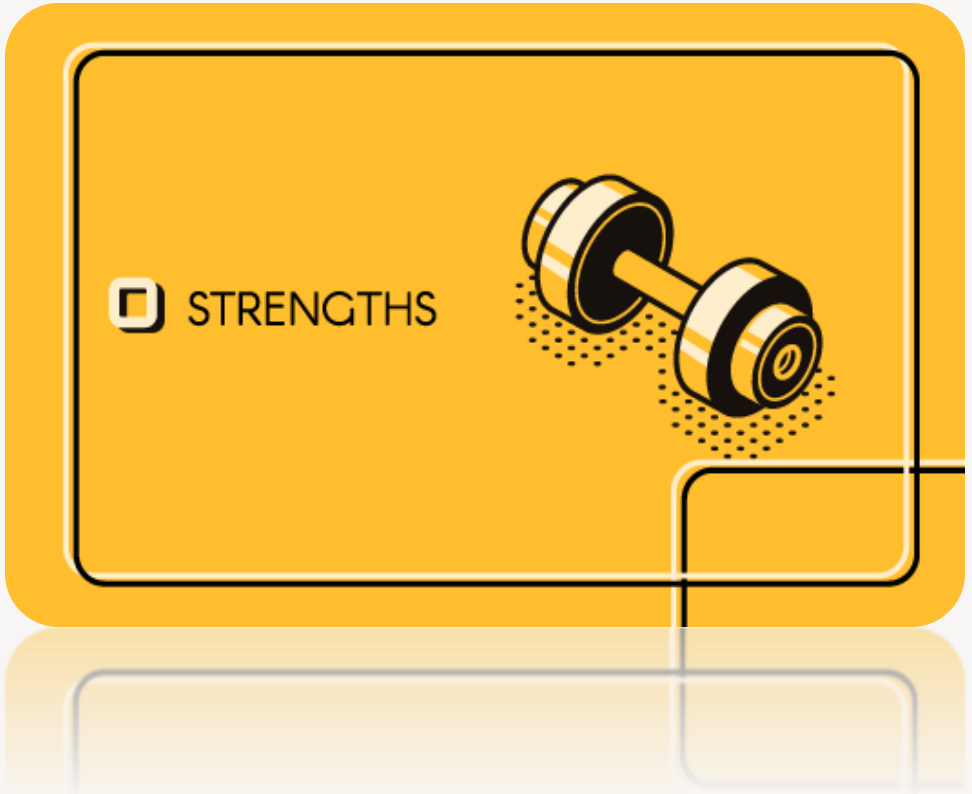


Strengths of the model

- The model is a pre-trained multilingual which gives it the advantage of working with 53 different language including Arabic.

- During training model is capable of understand the meaning of the source text and generate new text that conveys the same meaning but in a shorter form.

- The number of words in summary is between 30%-40% of the words in the original paragraph



Weaknesses of the model

• It needs large amount or computational resources to train and use effectively makes it less accessible to individuals or organizations with limited resources.

• Has maximum input length of 1024 tokens , which may not be sufficient for summarizing longer texts .

- Hight Population STD of Semantic Similarity
- Hight Average Compression Ratio

Insights And interesting findings.

- The rouge-l score of 0.218532285 is particularly relevant, as it is a commonly used metric for evaluating the quality of text summaries. A score of 0.2-0.3 is generally considered to be moderate, while a score of 0.4 or higher is good.

- The average-compression-ratio of 0.38710485 is also particularly relevant, as it suggests that the model can effectively compress information when generating summaries. This is an important feature of a good summarization model, as the goal of summarization is to extract the most important information from a longer piece of text and present it in a more concise form.




Thank you

Sky blues

ahmedmohamed2017e@gmail.com

References

Huggingface

BART Model for Text Summarization

mbart-large-50

Hugging-Face Summarization script