NAME: ADARSH V **SECTION: A**

SRN: PES2UG23CS026 **UNIT: 1**

Understanding about the Hugging face:

Hugging face is basically like a platform which has many ready to use pre trained models and we can share our models also

Transformers library:

Basically this library helps you to connect and use the models from the hugging face , it provides the API's to communicate with the models

Pipeline ():

It's a tool in the hugging face library which automatically does the processes such as tokenization , pre-processing , Loading the model , Run inferences , Post processing tasks

nltk : Natural Language toolkit library which is used for the text processing

In this handson we are comparing 2 types of models:

Dumb model : These models are faster , cheaper , less memory , follow simple pattern or fixed rules and their outputs are predictable and sometimes out of context and they are trained on less parameters

Ex: distilgpt2

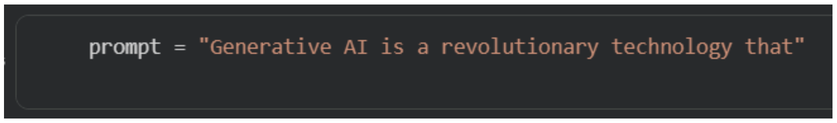
Smart model : These models are trained on more parameters , they are little slower than the dumb model , more memory , understands the context and generates new content

Ex: gpt2

Seed value: It's the initial value given to a random number generator , so that the random results can be same and repeated

1)TRYING DIFFERENT SEED VALUES

Prompt



```
prompt = "Generative AI is a revolutionary technology that"
```

Prompt : “Generative AI is a revolutionary technology that”

SEED VALUE – 42

FAST MODEL (distilgpt2) - Generative AI is a revolutionary technology that can take on the task of finding, learning, and learning in a given environment.

STANDARD MODEL (gpt2) - Generative AI is a revolutionary technology that enables a wide range of intelligent systems to work independently from one another. It introduces a new way of thinking about AI and provides a new paradigm for the development of intelligent AI.

SEED VALUE – 45

FAST MODEL(distilgpt2) **-** Generative AI is a revolutionary technology that can revolutionize and transform the world in the most significant way possible.

Standard model (gpt2) **-** Generative AI is a revolutionary technology that is the future. We are now in the last decade or so of a global AI revolution. It is being used to create artificial intelligence in the fields of medicine and medical research. The emergence of new medicines can be considered as a step toward realizing such a technology.

Inference from both the models:

Fast model (dumb model) distilgpt2 :

This model gives very short responses , which is less detailed and it is more generic phrasing and the response if sometimes repetitive and shallow also it focuses on speed and also consumes less memory

Standard mode (Smart model) gpt2:

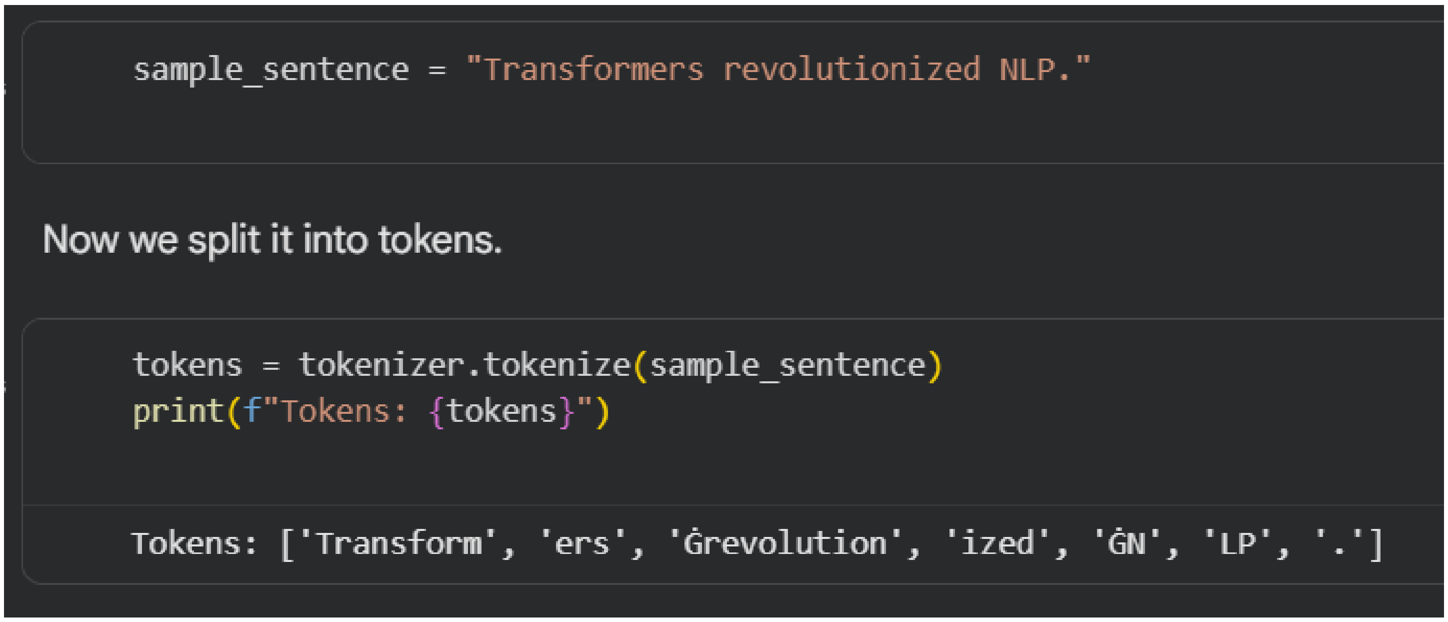
This model gives long responses , with better sentences and it is more context aware and also it consumes more memory and it provides result little slower than the dumb model

Compare the two outputs. Does the standard model stay more on topic? Does the fast model drift into nonsense?

Ans : The standard model (gpt 2) stayed more on the topic with clearer and more meaningful responses where as the fast model (distilgpt2) gives shorter and faster responses but shows loss of depth and the responses are repetitive and slight topic drift , its not fully drift to nonsense but it is weaker

Tokenization : The process of breaking down the text into the set of tokens such as words, sub words , characters etc

Example :



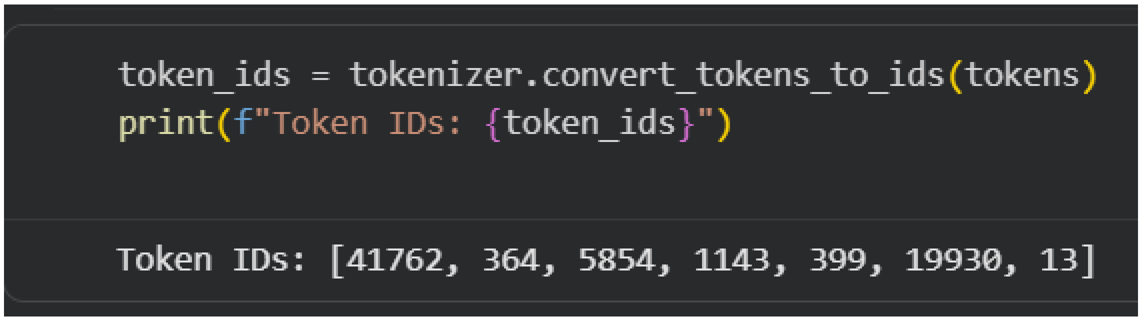
```
sample_sentence = "Transformers revolutionized NLP."
```

Now we split it into tokens.

```
tokens = tokenizer.tokenize(sample_sentence)
print(f"Tokens: {tokens}")
```

```
Tokens: ['Transform', 'ers', 'Ġrevolution', 'ized', 'ĠN', 'LP', '.']
```

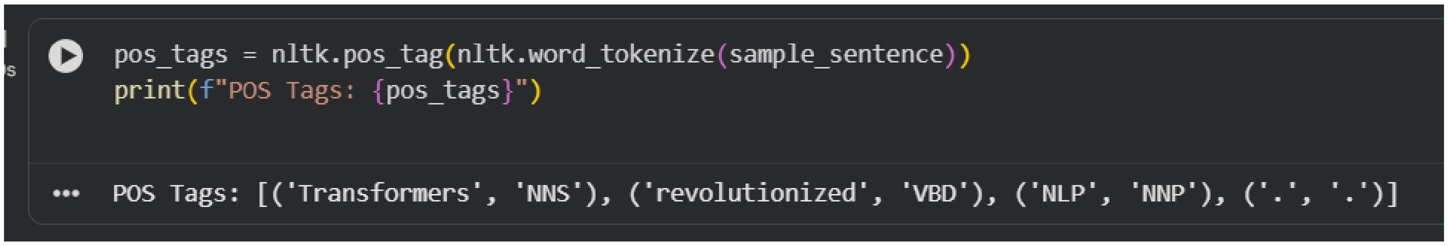
Then every token is assigned to some unique id because the computers cannot read texts



```
token_ids = tokenizer.convert_tokens_to_ids(tokens)
print(f"Token IDs: {token_ids}")
```

```
Token IDs: [41762, 364, 5854, 1143, 399, 19930, 13]
```

POS TAGGING : The process of assigning grammatical labels to the words is known as Parts of speech tagging



```
pos_tags = nltk.pos_tag(nltk.word_tokenize(sample_sentence))
print(f"POS Tags: {pos_tags}")
```

```
POS Tags: [('Transformers', 'NNS'), ('revolutionized', 'VBD'), ('NLP', 'NNP'), ('.', '.')]
```

Example of different parts of speech are:

NNS – Noun (Plural)

NNP – (proper Noun)

VBD – Verb (Past tense)

JJ – Adjective

RB – Adverb

PRP – Pronoun

DT – Determiner

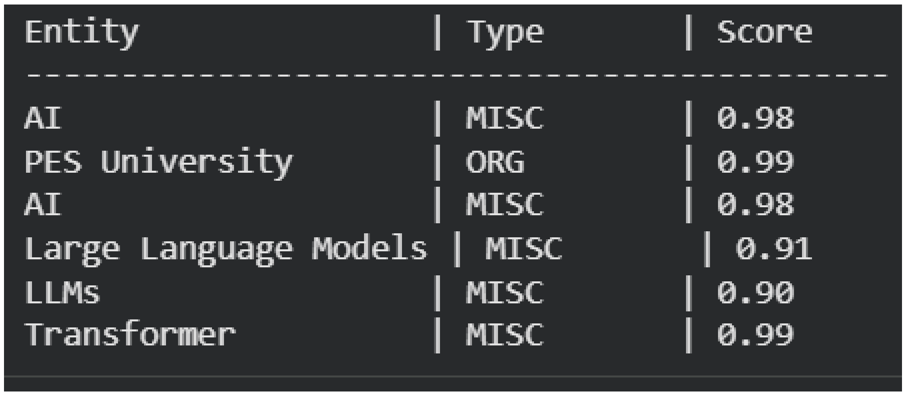
CC – Coordinating conjunction

CD – Cardinal numbers

. – Sentence terminator

And Many more ....

Name Entity recognition: The process of detecting and classifying the named entities in text like the names , organization and dates



```
Entity                | Type        | Score
----------------------------------------------
AI                    | MISC        | 0.98
PES University        | ORG         | 0.99
AI                    | MISC        | 0.98
Large Language Models | MISC        | 0.91
LLMs                  | MISC        | 0.90
Transformer           | MISC        | 0.99
```

Model used for summarization :

2 models are used here fast and Quality summarizer

The introduction of the Transformer architecture in the 2017 paper "Attention is all you need" was a watershed moment in AI. It provided a more effective and scalable way to handle sequential data like text, replacing older, less efficient methods like recurrence (RNNs) and convolutions.

The fundamental innovation of the Transformer is the attention mechanism. This component allows the model to weigh the importance of different words (tokens) in the input sequence when making a prediction. In essence, for each word it processes, the model can "pay attention" to all other words in the input, helping it understand context, resolve ambiguity, and handle long-range dependencies. This is crucial for tasks like translation, summarization, and question answering.

The Transformer architecture consists of an encoder stack (to process the input) and a decoder stack (to generate the output), both of which heavily utilize multi-head attention and feed-forward networks.

Distilbart (fast summarizer) Output:

The introduction of the Transformer architecture in the 2017 paper "Attention is all you need" was a watershed moment in AI . It provided a more effective and scalable way to handle sequential data like text, replacing older, less efficient methods like recurrence (RNNs) and conv

bart-large-cnn (Quality Summarizer) Output:

The introduction of the Transformer architecture in the 2017 paper "Attention is all you need" was a watershed moment in AI. It provided a more effective and scalable way to handle sequential data like text.

Inference:

Distilbart (fast summarizer)

Produces long summaries with extra details and less polished and it is optimized for the speed and may sacrifice completeness due to length constraint

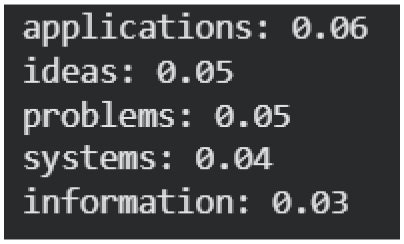
bart-large-cnn (Quality Summarizer) :

Produces shorter and cleaner summary and its well formed and produces the complete sentence and it prioritize for the quality than for the speed

Question and Answering
The extractive QA model correctly selects answers from the given context, but when the exact answer is not clearly stated, it may return a related but less precise span. This highlights a key limitation of extractive question answering compared to generative models

Masked language modeling

Masked Language Modeling enables BERT to understand context by predicting missing words, allowing multiple valid completions rather than a single correct answer



```
applications: 0.06
ideas: 0.05
problems: 0.05
systems: 0.04
information: 0.03
```

Gives the probability of occurrence of the next word

Unit1_benchmark file observation and the inference:

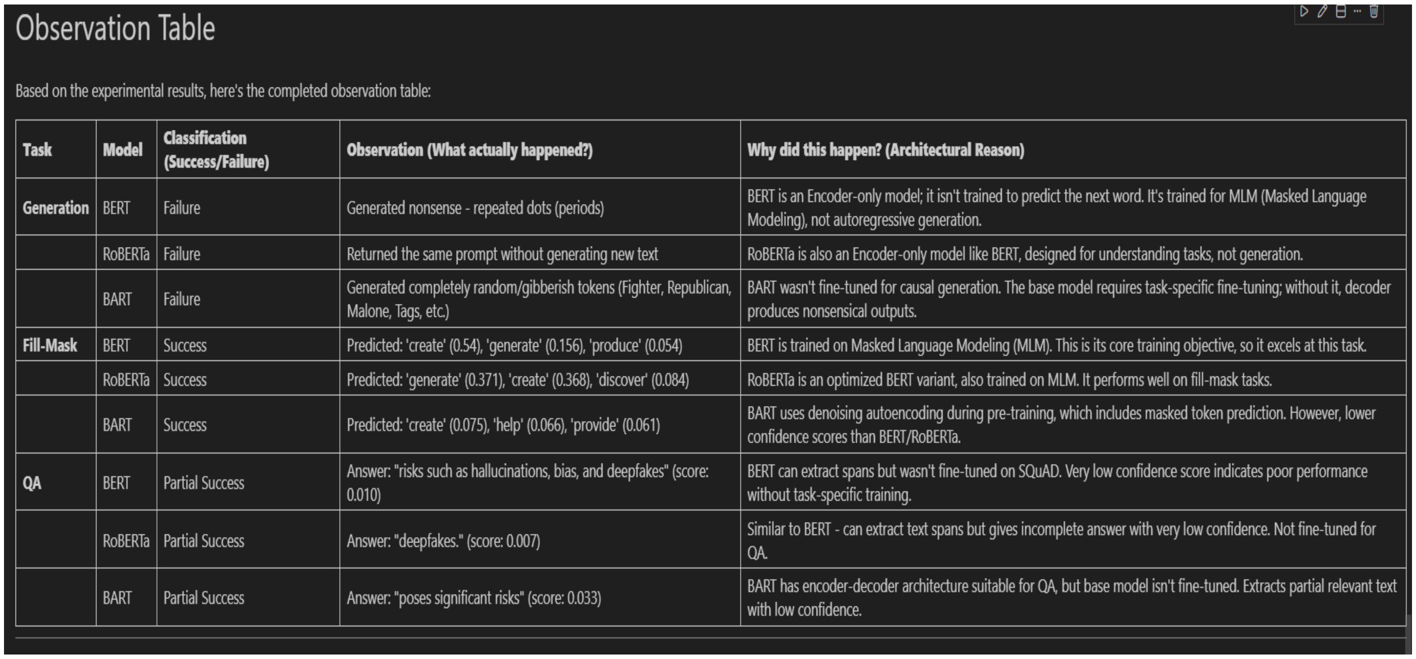
1. Encoder-only models (BERT, RoBERTa): Excel at understanding tasks (MLM, classification) but fail at generation tasks

2. Encoder-decoder models (BART): Designed for generation but require fine-tuning for specific tasks

3. Fine-tuning matters: All base models show poor performance on tasks they weren't specifically trained for

4. Task alignment: Best performance occurs when model architecture matches the task (e.g., BERT on fill-mask)

Observation table:



Observation Table

Based on the experimental results, here's the completed observation table:

Task	Model	Classification (Success/Failure)	Observation (What actually happened?)	Why did this happen? (Architectural Reason)
Generation	BERT	Failure	Generated nonsense - repeated dots (periods)	BERT is an Encoder-only model; it isn't trained to predict the next word. It's trained for MLM (Masked Language Modeling), not autoregressive generation.
	RoBERTa	Failure	Returned the same prompt without generating new text	RoBERTa is also an Encoder-only model like BERT, designed for understanding tasks, not generation.
	BART	Failure	Generated completely random/gibberish tokens (Fighter, Republican, Malone, Tags, etc.)	BART wasn't fine-tuned for causal generation. The base model requires task-specific fine-tuning; without it, decoder produces nonsensical outputs.
Fill-Mask	BERT	Success	Predicted: 'create' (0.54), 'generate' (0.156), 'produce' (0.054)	BERT is trained on Masked Language Modeling (MLM). This is its core training objective, so it excels at this task.
	RoBERTa	Success	Predicted: 'generate' (0.371), 'create' (0.368), 'discover' (0.084)	RoBERTa is an optimized BERT variant, also trained on MLM. It performs well on fill-mask tasks.
	BART	Success	Predicted: 'create' (0.075), 'help' (0.066), 'provide' (0.061)	BART uses denoising autoencoding during pre-training, which includes masked token prediction. However, lower confidence scores than BERT/RoBERTa.
QA	BERT	Partial Success	Answer: "risks such as hallucinations, bias, and deepfakes" (score: 0.010)	BERT can extract spans but wasn't fine-tuned on SQuAD. Very low confidence score indicates poor performance without task-specific training.
	RoBERTa	Partial Success	Answer: "deepfakes." (score: 0.007)	Similar to BERT - can extract text spans but gives incomplete answer with very low confidence. Not fine-tuned for QA.
	BART	Partial Success	Answer: "poses significant risks" (score: 0.033)	BART has encoder-decoder architecture suitable for QA, but base model isn't fine-tuned. Extracts partial relevant text with low confidence.

Ai_story_teller_project

Model Used: distilgpt2 (faster version)

Task: Text generation for creative story telling

Pipeline : Used HUGGING FACE pipeline API

Observation of the output generated

As we are using the gpt2 model the output generated is faster, uses less memory but it losses context , results are repetitive , POS confusion and few of them are incomplete so using a standard model would improve it

Parameters Used

Temperature (0.5, 0.8, 1.0)

- Controls randomness/creativity
- Lower (0.5) → More predictable, coherent output
- Higher (1.0) → More diverse, creative, but potentially less coherent

Top-p (0.95)

- Nucleus sampling - considers top 95% probability mass
- Balances diversity with quality

max_new_tokens (180)

- Generates ~180 new tokens (roughly 1-2 paragraphs)

no_repeat_ngram_size (2)

- Prevents repetitive bigrams, improving output quality

Inference of using different Temperature values

- 0.5 (Low): More repetitive ("dark dark room"), dream-like/philosophical but coherent
- 0.8 (Medium): Better narrative variety, more dynamic action
- (High): Most creative but also most chaotic; unusual descriptions but fragmented logic