Gen-AI Semester-6

LAB-1 **Date:**23/01/2026

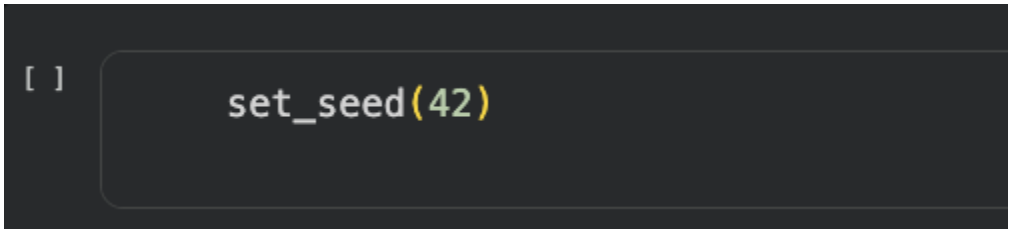
Name: Ankana Mandal **SRN:** PES2UG23CS076 **Section: B**

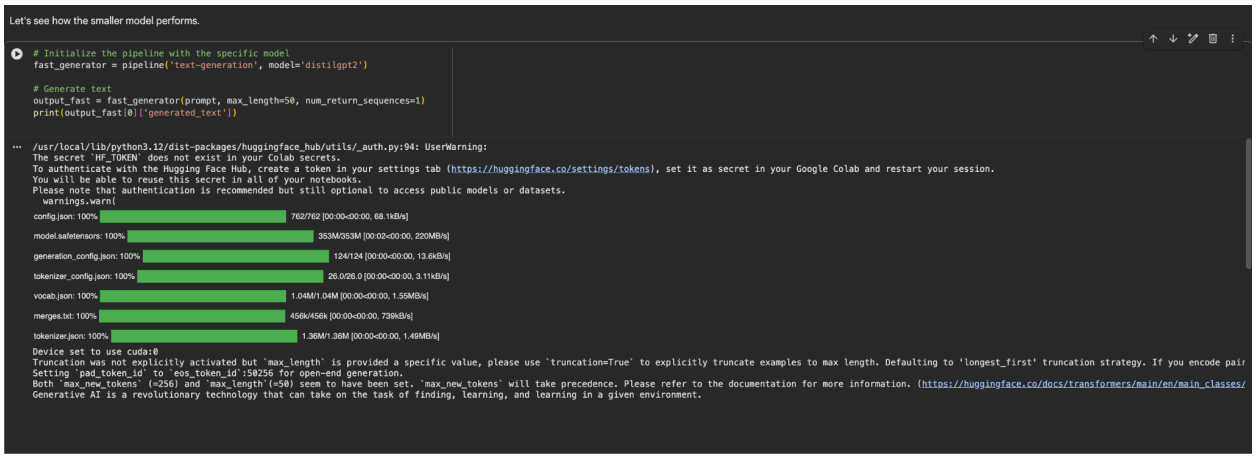
Q) SEED Value

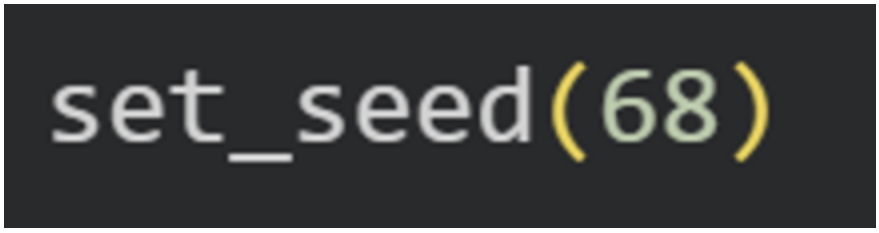
The set_seed() function is used to initialize the pseudo-random number generator. It provides a fixed starting point for randomness in a program.

When the same seed value (for example, set_seed(42)) is used, the sequence of random numbers generated remains the same every time. As a result, operations that rely on randomness—such as text generation in Generative AI models produce identical outputs. This is important for reproducibility in experiments.

If the seed value is changed, the random number generator starts from a different point, producing a different sequence of random numbers. This leads to different outputs for tasks like text generation. The numerical value of the seed itself does not make the output better or worse; it only ensures that the randomness is repeatable and consistent for that specific seed.








```
smart_generator = pipeline('text-generation', model='gpt2')

output_smart = smart_generator(prompt, max_length=50, num_return_sequences=1)
print(output_smart[0]['generated_text'])
```

```
config.json: 100% 665/665 [00:00<00:00, 20.2kB/s]
model.safetensors: 100% 548M/548M [00:04<00:00, 248MB/s]
generation_config.json: 100% 124/124 [00:00<00:00, 6.86kB/s]
tokenizer_config.json: 100% 26.0/26.0 [00:00<00:00, 1.23kB/s]
vocab.json: 100% 1.04M/1.04M [00:00<00:00, 8.38MB/s]
merges.txt: 100% 456k/456k [00:00<00:00, 3.32MB/s]
tokenizer.json: 100% 1.36M/1.36M [00:00<00:00, 10.3MB/s]
Device set to use cuda:0
Truncation was not explicitly activated but `max_length` is provided a specific value, please use `truncation=True` to explicit
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
Both `max_new_tokens` (=256) and `max_length`(=50) seem to have been set. `max_new_tokens` will take precedence. Please refer t
Generative AI is a revolutionary technology that enables a wide range of intelligent systems to work independently from one and

In this article, we will discuss the main features of the new AI platform, and how it can be used to help us create a world tha

1. How Can I Use It?

The concept of AI is not new. It has been used by many people to measure their mental health and health-related behaviors, and

It is based on the premise that AI is a way for humans to move towards a more efficient way of thinking, and therefore, a bette

In this article, we will explain what AI can do.

What does it do

In this article, we will explain how all of our cognitive and emotional systems interact with the AI platform. The main feature

A new way of thinking about AI

A new paradigm for the development of intelligent AI

A new way of thinking about mental health and health-related behaviors
```

Q) Difference between Distilled Model and Plain Model

A plain model is the original large model trained directly on a dataset using ground-truth labels. It has a large number of parameters, which allows it to achieve high accuracy, but it requires more memory and has slower inference speed. Because of its size and resource requirements, deploying a plain model on edge or mobile devices is often difficult.

A distilled model is a smaller and compressed version of a plain model that is trained to mimic the behavior of a larger “teacher” model using a technique called knowledge distillation. Instead of learning only from ground-truth labels, it learns from the soft predictions of the teacher model. Although a distilled model may have slightly lower accuracy than the plain model, it offers much

faster inference, lower memory usage, and is more suitable for deployment on resource-constrained devices.

Q)What is POS Tagging?

POS Tagging (Part-of-Speech Tagging) is an NLP technique that assigns a grammatical label to each word in a sentence based on its role and context.

Word	POS Tag	Full Form	Explanation
Modern	JJ	Adjective	Describes the noun
AI	NNP	Proper Noun	Name of a specific field
systems	NNS	Noun, Plural	Indicates more than one system
analyze	VB	Verb, Base Form	Shows an action being performed
languag e	NN	Noun, Singular	Refers to a single concept

Tag	Description

NN	Singular noun
NNS	Plural noun
NNP	Proper noun
VB	Base form verb
VBD	Past tense verb
JJ	Adjective
RB	Adverb

Tokenization

Tokenization is the process of breaking text into smaller units called *tokens*, which can be words, subwords, or characters. It is usually the first step in Natural Language Processing (NLP). By converting raw text into tokens, models can analyze and understand language more effectively. Tokenization helps in tasks like text generation, sentiment analysis, and machine translation. Different models may use different tokenization strategies depending on efficiency and vocabulary size.

Documenting the Benchmark Assignment

The objective of the benchmark assignment was to study the behavior of different transformer architectures by forcing them to perform tasks they are not equally suited for. This experiment helps in understanding why model architecture matters in Generative AI.
Three transformer models were used for the benchmark:
• BERT (bert-base-uncased) – an encoder-only model

• RoBERTa (roberta-base) – an optimized encoder-only model
• BART (facebook/bart-base) – an encoder-decoder model
All three models were evaluated on the same set of tasks using Hugging Face pipelines.

Tasks Performed

1. **Text Generation**

The models were asked to generate text for a given prompt.
It was observed that BERT and RoBERTa failed to generate meaningful text because encoder-only models are not designed for next-token generation. BART, having a decoder component, was able to generate text, though the output quality was limited.

2. **Fill-Mask (Masked Language Modeling)**

In this task, a sentence containing a masked token was provided.
BERT and RoBERTa performed well, correctly predicting suitable words due to their training on masked language modeling. BART showed weaker performance as it is not primarily optimized for this task.

3. **Question Answering**

The models were asked to answer a question based on a given context.
The outputs were inconsistent because the base versions of these models are not fine-tuned specifically for question answering. However, partial or approximate answers were sometimes produced.

Observations

The benchmark clearly demonstrated that:
• Encoder-only models such as BERT and RoBERTa are best suited for understanding tasks like fill-mask.
• Encoder-decoder models like BART are more flexible for generation-related tasks.
• Using a model for a task it was not designed for leads to poor or failed outputs, which is an expected and valid observation.

Conclusion

This benchmark assignment highlights the importance of transformer architecture in Generative AI. It shows that model success or failure depends on how well the task aligns with the model's design, reinforcing key concepts covered in Unit 1.

Task	Model	Classification (Success/Failure)	Observation (What actually happened?)	Why did this happen? (Architectural Reason)
Generation	BERT	*Failure*	*Example: Generated nonsense or random symbols.*	*BERT is an Encoder; it isn't trained to predict the next word.*
	RoBERTa	*Failure*	*didn't generate new text, just returned the input prompt*	*RoBERTa is also an encoder only model so it isnt designed to predict the next word*
	BART	*Failure*	*generated text but the text was meaningless and rubbish*	*Even though BART is an encoder-decoder model and can handle this task, we used the base model which is trained on a lot of raw data, and isnt finetuned to understand and handle grammar and logic*
Fill-Mask	BERT	*Success*	*Predicted 'create', 'generate', highest confidence score(0.5397).*	*BERT is trained on Masked Language Modeling (MLM).*
	RoBERTa	*Sucesss*	*Predicted 'generate', 'create', more consistent scores for synonymns*	*more robust training data than BERT.*
	BART	*Success*	*Predicted 'create', 'help', had lower confidence*	*flexible architecture but is designeed for seq2seq.*
QA	BERT	*partial success*	*Extracted 'hallucinations, bias' (Score: 0.017).*	*BERT had NSP(next sentence prediction) in its training and that helped but since the heads are randomly initialised the outputs are almost random*
	RoBERTa	*Failure*	*Extracted only "deepfakes." (Score: 0.012)*	*removal of NSP in pretraining reduced its ability to link QA and it had low confidence*
	BART	*partial failure*	*Extracted "Generative AI poses significant" (Score: 0.064)*	*the decoder helped it handle longer sequence queries but the head being randomly initialised meant the output was random*

Project Components:

- Libraries and Models: It utilizes the Hugging Face transformers library to implement a pre-trained question-answering pipeline (qa_pipeline).
- Dataset/Context: The system uses a provided study_text which contains educational information about Generative AI, Large Language Models (LLMs), and their architectures.
- Task Implementation: The project demonstrates the model's ability to extract answers from the context by processing specific queries.

Results of the Question-Answering Tasks:

The notebook includes the following results from the executed pipeline:

- Question: "What is Generative AI?"
    - Answer: "a branch of artificial intelligence"
- Question: "What architecture do LLMs use?"
    - Answer: The notebook output for this specific question is truncated in the provided source, but it is configured to process this query using the qa_pipeline against the same study_text.