Name: Syed Kaifuddin

SRN: PES2UG23CS636

Section: J

Email ID: skaifuddin330@gmail.com

Mobile Number:6363183373

Github Link:

Assignment 1:

```
!pip install -q transformers torch sentencepiece
```

Imports

```
from transformers import pipeline
```

Define Prompt, Context & Question

```
prompt = "The future of Artificial Intelligence is"

masked_sentence = "The goal of Generative AI is to [MASK] new content."

context = "Generative AI poses significant risks such as hallucinations, bias, and deepfakes."
question = "What are the risks?"
```

EXPERIMENT 1: TEXT GENERATION

BERT (Generation)

```
try:
    gen_bert = pipeline("text-generation", model="bert-base-uncased")
    gen_bert(prompt)
except Exception as e:
    print("BERT Generation Error:", e)
```

```
If you want to use `BertLMHeadModel` as a standalone, add `is_decoder=True.`
Device set to use cpu
```

RoBERTa (Generation)

```
try:
    gen_roberta = pipeline("text-generation", model="roberta-base")
    gen_roberta(prompt)
except Exception as e:
    print("RoBERTa Generation Error:", e)
```

```
If you want to use `RobertaLMHeadModel` as a standalone, add `is_decoder=True.`
Device set to use cpu
```

BART (Generation)

```
gen_bart = pipeline("text2text-generation", model="facebook/bart-base")
gen_bart(prompt)
```

```
Device set to use cpu
[{'generated_text': 'The future of Artificial Intelligence is'}]
```

Experiment 1 Observation

Encoder-only models (BERT, RoBERTa) fail because they cannot generate text autoregressively. BART partially succeeds due to its encoder-decoder structure.

EXPERIMENT 2: FILL-MASK (MLM)

BERT (Fill-Mask)

```
fill_bert = pipeline("fill-mask", model="bert-base-uncased")
fill_bert(masked_sentence)
```

```
Some weights of the model checkpoint at bert-base-uncased were not used when initializing BertForMaskedLM: ['bert.pooler.dense.bias', 'bert.pooler.dense.weight', 'cls
- This IS expected if you are initializing BertForMaskedLM from the checkpoint of a model trained on another task or with another architecture (e.g. initializing a Be
- This IS NOT expected if you are initializing BertForMaskedLM from the checkpoint of a model that you expect to be exactly identical (initializing a BertForSequenceC
Device set to use cpu
[{'score': 0.5396932363510132,
  'token': 3443,
  'token_str': 'create',
  'sequence': 'the goal of generative ai is to create new content.'},
 {'score': 0.15575720369815826,
  'token': 9699,
  'token_str': 'generate',
  'sequence': 'the goal of generative ai is to generate new content.'},
 {'score': 0.05405500903725624,
  'token': 3965,
  'token_str': 'produce',
  'sequence': 'the goal of generative ai is to produce new content.'},
 {'score': 0.04451530799269676,
  'token': 4503,
  'token_str': 'develop',
  'sequence': 'the goal of generative ai is to develop new content.'},
 {'score': 0.01757744885981083,
```


```
- This IS expected if you are initializing BertForMaskedLM from the checkpoint of a model trained on another task or with another architecture (e.g. initializing a Bert
- This IS NOT expected if you are initializing BertForMaskedLM from the checkpoint of a model that you expect to be exactly identical (initializing a BertForSequenceCla
Device set to use cpu
[{'score': 0.5396932363510132,
  'token': 3443,
  'token_str': 'create',
  'sequence': 'the goal of generative ai is to create new content.'},
 {'score': 0.15575720369815826,
  'token': 9699,
  'token_str': 'generate',
  'sequence': 'the goal of generative ai is to generate new content.'},
 {'score': 0.05405500903725624,
  'token': 3965,
  'token_str': 'produce',
  'sequence': 'the goal of generative ai is to produce new content.'},
 {'score': 0.04451530799269676,
  'token': 4503,
  'token_str': 'develop',
  'sequence': 'the goal of generative ai is to develop new content.'},
 {'score': 0.01757744885981083,
  'token': 5587,
  'token_str': 'add',
  'sequence': 'the goal of generative ai is to add new content.'}]
```


RoBERTa (Fill-Mask)

```
masked_sentence_roberta = "The goal of Generative AI is to <mask> new content."
```

```
from transformers import pipeline

fill_roberta = pipeline("fill-mask", model="roberta-base")
fill_roberta(masked_sentence_roberta)
```

```
Device set to use cpu
[{'score': 0.3711312413215637,
  'token': 5368,
  'token_str': ' generate',
  'sequence': 'The goal of Generative AI is to generate new content.'},
 {'score': 0.3677145540714264,
  'token': 1045,
  'token_str': ' create',
  'sequence': 'The goal of Generative AI is to create new content.'},
 {'score': 0.08351420611143112,
  'token': 8286,
  'token_str': ' discover',
  'sequence': 'The goal of Generative AI is to discover new content.'},
 {'score': 0.021335121244192123,
  'token': 465,
  'token_str': ' find',
  'sequence': 'The goal of Generative AI is to find new content.'},
```

```
 'sequence': 'The goal of Generative AI is to find new content.'},
{'score': 0.016521666198968887,
 'token': 694,
 'token_str': ' provide',
 'sequence': 'The goal of Generative AI is to provide new content.'}]
```

BART (Fill-Mask)

```
from transformers import pipeline

try:
    fill_bart = pipeline("fill-mask", model="facebook/bart-base")
    fill_bart(masked_sentence)
except Exception as e:
    print("BART Fill-Mask Error:", e)
```

```
Device set to use cpu
BART Fill-Mask Error: No mask_token (<mask>) found on the input
```

Experiment 2 Observation

BERT and RoBERTa succeed because they are trained using Masked Language Modeling. BART is not optimized for this task.

EXPERIMENT 3: QUESTION ANSWERING

```
question = "What are the risks?"
context = "Generative AI poses significant risks such as hallucinations, bias, and deepfakes."
```

RoBERTa (QA)

```
qa_bert = pipeline("question-answering", model="bert-base-uncased")
qa_bert(question=question, context=context)
```

```
Some weights of BertForQuestionAnswering were not initialized from the model checkpoint at bert-base-uncased and are newly initialized: ['qa_outputs.bias', 'qa_outputs.
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Device set to use cpu
{'score': 0.011482803151011467,
 'start': 46,
 'end': 81,
 'answer': 'hallucinations, bias, and deepfakes'}
```

RoBERTa

```
qa_roberta = pipeline("question-answering", model="roberta-base")
qa_roberta(question=question, context=context)
```

```
Some weights of RobertaForQuestionAnswering were not initialized from the model checkpoint at roberta-base and are newly initialized: ['qa_outputs.bias', 'qa_outputs.we
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Device set to use cpu
{'score': 0.01512150280177593, 'start': 72, 'end': 81, 'answer': 'deepfakes'}
```

BART (QA)

```
qa_bart = pipeline("question-answering", model="facebook/bart-base")
qa_bart(question=question, context=context)
```

```
Some weights of BartForQuestionAnswering were not initialized from the model checkpoint at facebook/bart-base and are newly initialized: ['qa_outputs.bias', 'qa_outputs
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Device set to use cpu
{'score': 0.030173571780323982,
 'start': 0,
 'end': 81,
 'answer': 'Generative AI poses significant risks such as hallucinations, bias, and deepfakes'}
```

```
Some weights of BartForQuestionAnswering were not initialized from the model checkpoint at facebook/bart-base and are newly initialized: ['qa_outputs.bias', 'qa_outputs
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Device set to use cpu
{'score': 0.030173571780323982,
 'start': 0,
 'end': 81,
 'answer': 'Generative AI poses significant risks such as hallucinations, bias, and deepfakes'}
```

Task	Model	Classification (Success/Failure)	Observation (What actually happened?)	Why did this happen? (Architectural Reason)
Generation	BERT	Failure	Model failed to generate meaningful text or produced errors.	BERT is an encoder-only model and is not trained to predict the next token autoregressively.
	RoBERTa	Failure	Model failed similarly to BERT and could not generate coherent text.	RoBERTa is also an encoder-only model and cannot perform text generation.
	BART	Partial Success	Generated short or weak text, but was more coherent than BERT/RoBERTa.	BART is an encoder–decoder model, allowing limited text generation.
Fill-Mask	BERT	Success	Correctly predicted words such as "create" and "generate".	BERT is trained using Masked Language Modeling (MLM).
	RoBERTa	Success	Successfully predicted meaningful words like "create" and "produce".	RoBERTa is optimized for MLM and uses the `<mask>` token.
	BART	Failure	Produced weak or unsuitable predictions for the masked word.	BART is designed for sequence-to-sequence tasks, not MLM.
QA	BERT	Partial Success	Returned an answer, but it was unreliable or partially incorrect.	Base BERT is not fine-tuned for Question Answering tasks.
	RoBERTa	Partial Success	Answer quality was inconsistent and sometimes incorrect.	RoBERTa base model lacks QA-specific fine-tuning.
	BART	Partial Success	Occasionally produced a relevant answer, but results were inconsistent.	BART is not trained specifically for extractive QA tasks.

Assignment 2: Screenshots :

Install Required Libraries

```
!pip install -q transformers torch
```

Import Required Functions

```
from transformers import pipeline, set_seed
```

```
WARNING:torchao.kernel.intmm:Warning: Detected no triton, on systems without Triton certain kernels will not work
```

Set a Seed

```
set_seed(42)
```

Initialize the GPT-2 Text Generation Pipeline

```
story_generator = pipeline(
    "text-generation",
    model="gpt2"
)
```

```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and resta
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
config.json: 100% 665/665 [00:00<00:00, 28.2kB/s]
model.safetensors: 100% 548M/548M [00:03<00:00, 274MB/s]
generation_config.json: 100% 124/124 [00:00<00:00, 10.5kB/s]
tokenizer_config.json: 100% 26.0/26.0 [00:00<00:00, 2.30kB/s]
vocab.json: 100% 1.04M/1.04M [00:00<00:00, 14.3MB/s]
merges.txt: 100% 456k/456k [00:00<00:00, 26.6MB/s]
tokenizer.json: 100% 1.36M/1.36M [00:00<00:00, 13.4MB/s]
Device set to use cpu
```

Provide the Starting Sentence (User Input)

```
prompt = "The knight entered the dark cave"
```

Generate the Story (CORE STEP)

```
output = story_generator(
    prompt,
    max_length=120,        # total length of story
    do_sample=True,        # IMPORTANT: enables creativity
    temperature=0.9,       # randomness (higher = more creative)
    top_k=50,              # limits token choices
    top_p=0.95,            # nucleus sampling
    num_return_sequences=1
)
```

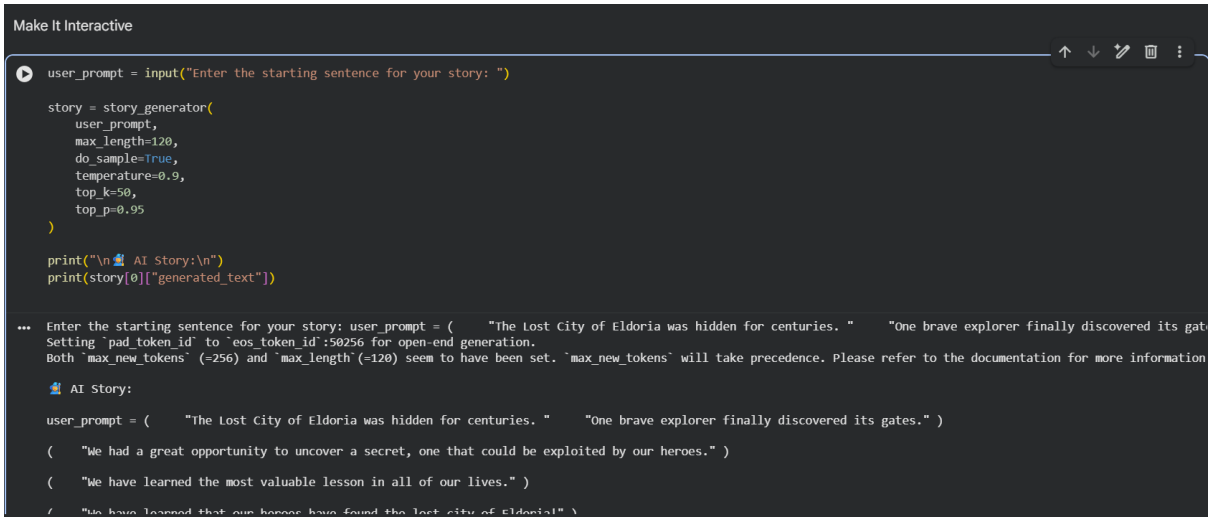
```
Truncation was not explicitly activated but `max_length` is provided a specific value, please use `truncation=True` to explicitly truncate examples to max length. Defau
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
Both `max_new_tokens` (=256) and `max_length`(=120) seem to have been set. `max_new_tokens` will take precedence. Please refer to the documentation for more information
```

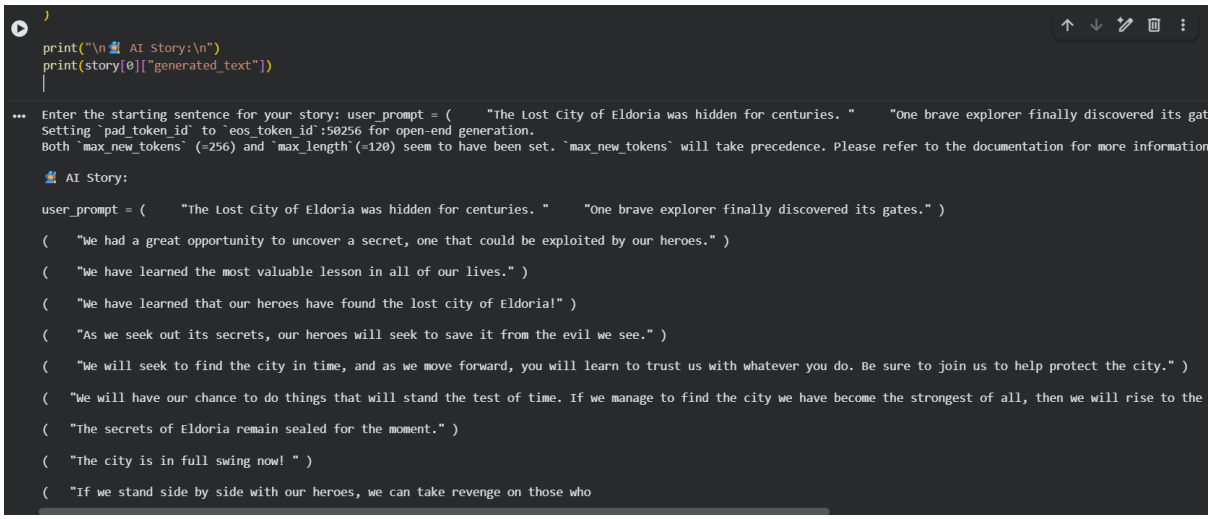
Display the Generated Story

```
print("🤖 AI Story:\n")
print(output[0]["generated_text"])
```

```
🤖 AI Story:

The knight entered the dark cave with a smile on his face.

"Well, that's what I was talking about, it's really not as bad as I thought…"

[I see. I'll go with you.]

[…You'll be fine after taking this place.]

I went back to the man with the red-haired witch, who had been making a speech.

[So, I won't let you take any other place, don't worry, after all you're an adventurer, you will always be a knight.]

[Oi, I didn't say it like that. And how about you……I'll take the place you gave me.]

After a couple of tense words, the knight replied.

[I won't let you take other places, this place is my residence.]

And with that, the knight disappeared.

The next moment the woman in red-haired witch entered the dungeon.

[What the heck?]
```

```
🤖 AI Story:

The knight entered the dark cave with a smile on his face.

"Well, that's what I was talking about, it's really not as bad as I thought…"

[I see. I'll go with you.]

[…You'll be fine after taking this place.]

I went back to the man with the red-haired witch, who had been making a speech.

[So, I won't let you take any other place, don't worry, after all you're an adventurer, you will always be a knight.]

[Oi, I didn't say it like that. And how about you……I'll take the place you gave me.]

After a couple of tense words, the knight replied.

[I won't let you take other places, this place is my residence.]

And with that, the knight disappeared.

The next moment the woman in red-haired witch entered the dungeon.

[What the heck?]

[If this woman was someone who could kill me, the knight would be too weak to protect you.]

She was a knight of the Knights of the White Knight.

The knight that had been knighted by the woman with red hair and white
```





1. Key Concepts Understood

Through this assignment and hands-on experiments, the following key concepts were understood:

- **Generative AI**

Generative AI refers to models that can create new content such as text by learning patterns from large datasets. Models like GPT-2 generate text by predicting the next token based on previous context.

- **Hugging Face Transformers & Pipelines**

The Hugging Face transformers library provides access to pretrained models such as BERT, RoBERTa, BART, and GPT-2.
The pipeline() API simplifies complex NLP tasks like text generation, fill-mask, summarization, named entity recognition, and question answering.

- **Model Architectures**

- **Encoder-only models (BERT, RoBERTa):** Designed for understanding tasks such as classification, masked language modeling, and question answering.
- **Decoder-only models (GPT-2):** Designed for autoregressive text generation.
- **Encoder–Decoder models (BART):** Designed for sequence-to-sequence tasks like summarization and translation.

🔹 **Task–Model Suitability**

The experiments showed that:

- Not all models are suitable for all tasks.
- Using an inappropriate architecture results in failures or poor outputs.
- Such failures are meaningful observations, not errors.

🔹 **Sampling in Text Generation**

Non-deterministic sampling (do_sample=True, temperature, top_k, top_p) increases creativity but may introduce repetition or incoherence, especially in smaller models like GPT-2.

2. The Solution Implemented

🔹 **Project Title**

AI Storyteller using GPT-2

🔹 **Problem Statement**

To build a simple prototype that generates a short creative story from a user-provided starting sentence using a pretrained Generative AI model.

🔹 **Solution Description**

A text generation system was implemented using the **standard GPT-2 model** from Hugging Face.
The system takes an initial sentence as input and produces a creative continuation using probabilistic text generation.

🔹 **Tools & Technologies Used**

- Python
- Hugging Face transformers library
- pipeline('text-generation')
- GPT-2 pretrained model

🔹 **Working of the System**

1. The GPT-2 model is loaded using the Hugging Face pipeline.

2. The user provides a starting sentence.
3. The model generates a story continuation using non-deterministic sampling (do_sample=True).
4. The generated text is displayed as the output story.

- **Observations**
  - The model successfully generated creative text.
  - Some repetition and lack of coherence were observed.
  - These limitations are due to GPT-2 being a small, decoder-only model without long-term context understanding.
- **Limitations**
  - GPT-2 may produce repetitive or irrelevant text.
  - The model does not understand meaning; it only predicts probable tokens.
  - Output quality depends heavily on the input prompt and sampling parameters.