```
from transformers import pipeline, set_seed, GPT2Tokenizer
import os
import nltk
```

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

```
file_path = "unit 1.txt"
```

```
try:
  with open(file_path,'r',encoding="utf-8") as f:
    text=f.read()
  print("File loaded successfully!")
except FileNotFoundError:
  print(f"Error: {file_path} not found!")
```

```
File loaded successfully!
```

```
print("--- Data Preview ---")
print(text[:500]+"...")
```

```
--- Data Preview ---
Generative AI and Its Applications: A Foundational Briefing

Executive Summary

This document provides a comprehensive overview of Generative AI, synthesizing foundational concepts, technological underpir
```

```
set_seed(42);
```

```
prompt = "Generative AI is an elective subject"
```

```
fast_generator = pipeline("text-generation", model="distilgpt2")
output_fast = fast_generator(prompt, max_length=50, num_return_sequences=1)
print(output_fast[0]['generated_text'])
```

```
/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
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% 762/762 [00:00<00:00, 53.9kB/s]
model.safetensors: 100% 353M/353M [00:07<00:00, 32.4MB/s]
generation_config.json: 100% 124/124 [00:00<00:00, 1.49kB/s]
tokenizer_config.json: 100% 26.0/26.0 [00:00<00:00, 320B/s]
vocab.json: 100% 1.04M/1.04M [00:00<00:00, 5.13MB/s]
merges.txt: 100% 456k/456k [00:00<00:00, 10.7MB/s]
tokenizer.json: 100% 1.36M/1.36M [00:00<00:00, 18.3MB/s]
Device set to use cpu
Truncation was not explicitly activated but `max_length` is provided a specific value, please use `truncation=True` to expli
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 refe
Generative AI is an elective subject that is currently a major focus in AI research.
```

```
fast_generator = pipeline("text-generation", model="gpt2")
output_fast = fast_generator(prompt, max_length=50, num_return_sequences=1)
print(output_fast[0]['generated_text'])
```

```
config.json: 100%  665/665 [00:00<00:00, 13.8kB/s]
model.safetensors: 100%  548M/548M [00:08<00:00, 46.0MB/s]
generation_config.json: 100%  124/124 [00:00<00:00, 2.69kB/s]
tokenizer_config.json: 100%  26.0/26.0 [00:00<00:00, 561B/s]
vocab.json: 100%  1.04M/1.04M [00:00<00:00, 12.1MB/s]
merges.txt: 100%  456k/456k [00:00<00:00, 9.67MB/s]
tokenizer.json: 100%  1.36M/1.36M [00:00<00:00, 24.2MB/s]
Device set to use cpu
Truncation was not explicitly activated but `max_length` is provided a specific value, please use `truncation=True` to expli
```

```
tokenizer = GPT2Tokenizer.from_pretrained("gpt2")
```

```
Generative AI is an elective subject, i.e. an AI that can learn from a specific situation. Once you have created an agent, y
```

```
sample_sentence = "Transformers have encoders and decoders"
```

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

```
Tokens: ['Transform', 'ers', 'Ġhave', 'Ġenc', 'od', 'ers', 'Ġand', 'Ġdec', 'od', 'ers']
```

Actions are implemented by

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

```
Token IDs: [41762, 364, 423, 2207, 375, 364, 290, 875, 375, 364]
```

```
nltk.download('averaged_perceptron_tagger_eng', quiet=True)
nltk.download('punkt', quiet=True)
nltk.download('punkt_tab')
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
True
```

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

```
POS Tags: [('Transformers', 'NNS'), ('have', 'VBP'), ('encoders', 'NNS'), ('and', 'CC'), ('decoders', 'NNS')]
```

```
ner_pipeline = pipeline("ner", model="dbmdz/bert-large-cased-finetuned-conll03-english")
```

```
config.json: 100%  998/998 [00:00<00:00, 55.4kB/s]
model.safetensors: 100%  1.33G/1.33G [00:16<00:00, 75.5MB/s]
Some weights of the model checkpoint at dbmdz/bert-large-cased-finetuned-conll03-english were not used when initializing Ber
- This IS expected if you are initializing BertForTokenClassification from the checkpoint of a model trained on another task
- This IS NOT expected if you are initializing BertForTokenClassification from the checkpoint of a model that you expect to
tokenizer_config.json: 100%  60.0/60.0 [00:00<00:00, 3.50kB/s]
vocab.txt: 213k/? [00:00<00:00, 11.8MB/s]
Device set to use cpu
```

```
snipper = text[:1000]
entities = ner_pipeline(snipper)

print(f"{'Entity':<20} | {'Type':<10} | {'Score':<5}")
print("-" * 45)

for entity in entities:
    if entity['score'] > 0.9:
        print(f"{entity['word']:<20} | {entity['entity']:<10} | {entity['score']:.2f}")
```

```
Entity               | Type       | Score
---------------------------------------------
AI                   | I-MISC     | 0.98
AI                   | I-MISC     | 0.99
P                    | I-ORG      | 1.00
##ES                 | I-ORG      | 1.00
University           | I-ORG      | 0.97
AI                   | I-MISC     | 0.98
Language             | I-MISC     | 0.98
Models               | I-MISC     | 0.99
LL                   | I-MISC     | 0.95
GP                   | I-MISC     | 0.96
##T                  | I-MISC     | 0.93
```

```
4                   | I-MISC     | 0.97
Trans               | I-MISC     | 0.99
##former            | I-MISC     | 0.99
```

```
transformer_section = """
The introduction of the Transformer architecture in the 2017 paper "Attention is all you need" was a watershed moment in AI
The fundamental innovation of the Transformer is the attention mechanism. This component allows the model to weigh the impo
The Transformer architecture consists of an encoder stack (to process the input) and a decoder stack (to generate the outpu
"""
```

```
fast_sum = pipeline("summarization", model="sshleifer/distilbart-cnn-12-6")
res_fast = fast_sum(transformer_section, max_length=60, min_length=30, do_sample=False)
print(res_fast[0]['summary_text'])
```

```
Device set to use cpu
 The introduction of the Transformer architecture in the 2017 paper "Attention is all you need" was a watershed moment in AI
```

```
smart_sum = pipeline("summarization", model="facebook/bart-large-cnn")
res_smart = smart_sum(transformer_section, max_length=60, min_length=30, do_sample=False)
print(res_smart[0]['summary_text'])
```

```
l you need" was a watershed moment in AI. It provided a more effective and scalable way to handle sequential data like text.
```

```
qa_pipeline = pipeline("question-answering", model="distilbert-base-cased-distilled-squad")
```

```
config.json: 100%                                                 473/473 [00:00<00:00, 33.8kB/s]
model.safetensors: 100%                                           261M/261M [00:06<00:00, 39.2MB/s]
tokenizer_config.json: 100%                                       49.0/49.0 [00:00<00:00, 2.03kB/s]
vocab.txt: 100%                                                   213k/213k [00:00<00:00, 9.73MB/s]
tokenizer.json: 100%                                              436k/436k [00:00<00:00, 21.4MB/s]
Device set to use cpu
```

```
questions = [
    "What is the fundamental innovation of the Transformer?",
    "What are the risks of using Generative AI?"
]

for q in questions:
    res = qa_pipeline(question=q, context=text[:5000])
    print(f"\nQ: {q}")
    print(f"A: {res['answer']}")
```

```
Q: What is the fundamental innovation of the Transformer?
A: to identify hidden patterns, structures, and relationships within the data

Q: What are the risks of using Generative AI?
A: data privacy, intellectual property, and academic integrity
```

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

```
config.json: 100%                                                 570/570 [00:00<00:00, 43.9kB/s]
model.safetensors: 100%                                           440M/440M [00:18<00:00, 21.8MB/s]
Some weights of the model checkpoint at bert-base-uncased were not used when initializing BertForMaskedLM: ['bert.pooler.der
- This IS expected if you are initializing BertForMaskedLM from the checkpoint of a model trained on another task or with ar
- This IS NOT expected if you are initializing BertForMaskedLM from the checkpoint of a model that you expect to be exactly
tokenizer_config.json: 100%                                       48.0/48.0 [00:00<00:00, 3.15kB/s]
vocab.txt: 100%                                                   232k/232k [00:00<00:00, 11.3MB/s]
tokenizer.json: 100%                                              466k/466k [00:00<00:00, 24.8MB/s]
Device set to use cpu
```

```
masked_sentence = "The goal of Generative AI is to create new [MASK]."
preds = mask_filler(masked_sentence)

for p in preds:
    print(f"{p['token_str']}: {p['score']:.2f}")
```

```
applications: 0.06
ideas: 0.05
problems: 0.05
```

```
    systems: 0.04
    information: 0.03
```