



# DEPI GENERATIVE AI: TEXT GENERATION MODEL

## **Group Members:**

Shahd Mohamed

Fady Boktor

Abdalla Ibrahim

UNDER  
SUPERVISION OF:  
ENG MOHAMMED  
AGOOR

# PROJECT DESCRIPTION

We have built a generative AI text generating model which supports users with quick accurate responses using the right prompts. The model has been trained and tested utilizing cleaned and processed using cleaned and pre-processed wiki text data. The code below has been written on google Collab with python.







## PROJECT IMPACT

The model is meant to serve the community in reaching faster to the accurate required information saving time. Quick and accurate information savings has much more impact beyond time savings (e.g. Economical and Financial Impact).

# PROJECT BREAKDOWN:

The section below outlines the steps involved in fine-tuning a pre-trained GPT-2 language model on the WikiText-2 dataset and defines the architecture for a basic text-generating using PyTorch.

# PRODUCT OVERVIEW

ENVIRONMENT SETUP  
AND LIBRARIES  
INSTALLATION



LOADING AND  
PREPROCESSING  
DATASET



SPLITTING DATA AND FINE  
TUNING GPT2 MODEL  
(TRAINING AND TESTING)



DEPLOYMENT AND  
RESULTS

```
{
  "prompt": "Q: What is the capital of France? A: (Only the city name)",
  "max_new_tokens": 20,
  "temperature": 0.3,
  "top_k": 0.95,
  "repetition_penalty": 1.2,
  "stop": ["\n"]
}
```

Execute

Request URL  
http://127.0.0.1:8000/generate

Server response

Code	Details
200	<p>Response body</p> <pre>{   "status": "success",   "generated_text": "Q: What is the capital of France? A: (Only the city name) The capital city of the french nation is Paris, but it is a",   "prompt": "Q: What is the capital of France? A: (Only the city name)",   "max_new_tokens": 20,   "temperature": 0.3,   "top_k": 0.95,   "repetition_penalty": 1.2 }</pre> <p>Response headers</p> <pre>access-control-allow-origin: * content-length: 323 content-type: application/json date: Thu, 26 Apr 2023 03:46:05 GMT server: waitress</pre>

Responses

Code	Description
------	-------------

Links



# ENVIRONMENT SETUP AND LIBRARIES INSTALLATION:

This section of code is responsible for installing and setting up the required libraries for the project. It uses pip, the package installer for Python, and `python -m`, which runs a library module as a script.





## LOADING AND PREPROCESSING DATA:

This section takes a raw text dataset, cleans it, performs NLP preprocessing (tokenization and lemmatization), and then tokenizes it specifically for the GPT-2 model. The final output, tokenized dataset, is ready to be used for training or fine-tuning a GPT-2 model.



# SPLITTING DATA AND FINE TUNING GPT- 2 MODEL:

## SPLITTING DATA:

splitting the pre-processed dataset into two parts: (80%) one for training the model (train\_data) and (20%) one for validating its performance during training (val\_data).

## FINE TUNING:

- Libraries and Setup
- Data Preparation
- Training Arguments
- Creating and Training the trainer
- Saving the Fine-Tuned Model
- Checking Training Results

Epoch	Training Loss	Validation Loss
1	0.350200	0.333600
2	0.340700	0.324433
3	0.324100	0.322543

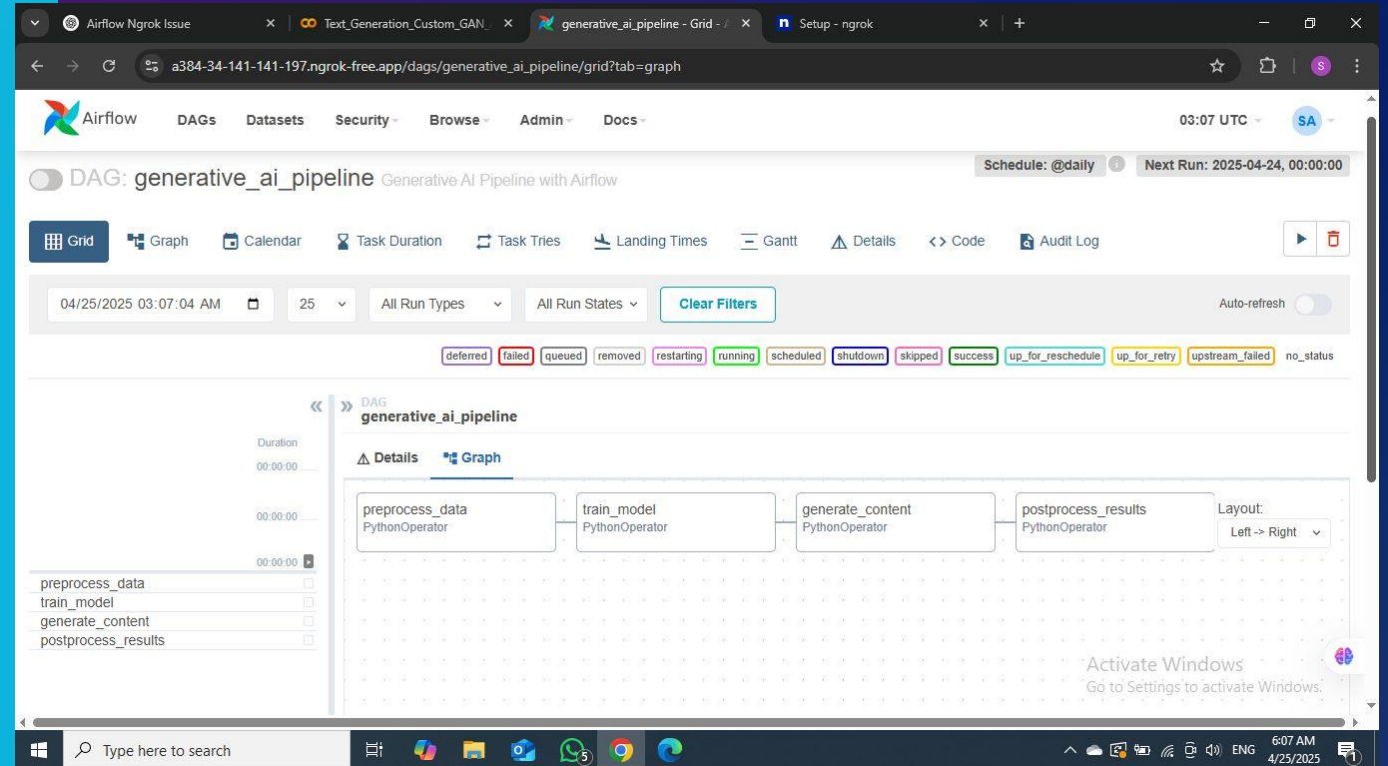


# APACHE AIRFLOW CONTENT GENERATION:

Apache Airflow automates and manages workflows in the content generation pipeline, ensuring tasks run in the right order and on time.

- Automates Task Scheduling: Runs tasks like data scraping, preprocessing, and model training at scheduled intervals.
- Workflow Orchestration: Defines task dependencies as Directed Acyclic Graphs (DAGs) to ensure smooth execution
- Monitoring: Tracks task status and provides real-time error handling.
- Scalable and Flexible: Easily scales to handle larger datasets and more tasks.
- Seamless ML Integration: Automates model retraining and content generation.

8/06/20XX



# DEPLOYMENT AND RESULTS

- We have deployed a text generation model using FastAPI, enabling us to generate text from a prompt through an API.

```
generatee.py
1  from transformers import GPT2LMHeadModel, GPT2Tokenizer
2  import torch
3
4  # Load tokenizer and model once
5  model_path = "./saved_model/final model"
6  tokenizer = GPT2Tokenizer.from_pretrained(model_path, padding_side="left")
7  tokenizer.pad_token = tokenizer.eos_token # Set padding token to eos
8  model = GPT2LMHeadModel.from_pretrained(model_path)
9  model.eval()
10
11 def generate_text(
12     prompt,
13     max_new_tokens=100, # Control how much new text to generate
14     temperature=0.9,
15     top_k=50,
16     top_p=0.95,
17     repetition_penalty=1.5,
18     do_sample=True,
19     num_beams=1
20 ):
21     # Tokenize the input prompt, allowing truncation and padding to the left
22     inputs = tokenizer(prompt.strip(), return_tensors="pt", padding=True, truncation=True, max_length=512)
```

```
EXPLORER
...
TEXT GENERATION PROJECT
  app
    > __pycache__
    > __init__.py
    > generate.py
    > main.py
    > model.py
    > notebook
    > saved_model
    > static
    > training
    > gpt2_finetune.py
    > wandb_config.py
    > venv
    > requirements.txt

main.py
app > main.py
1  from fastapi import FastAPI, HTTPException, Body, Request
2  from pydantic import BaseModel, Field
3  from app.generate import generate_text
4  import torch
5  from fastapi.staticfiles import StaticFiles
6  from fastapi.middleware.cors import CORSMiddleware
7  import logging
8
9  # Setup logging
10 logging.basicConfig(level=logging.INFO)
11 logger = logging.getLogger(__name__)
12
13 app = FastAPI(
14     title="Text Generation System",
15     version="1.0.0",
16     description="API for generating text using a fine-tuned GPT-2 model",
17     openapi_tags=[{
18         "name": "Generation",
19         "description": "Text generation endpoints"
20     }]
21 )
22
```

# DEPLOYMENT AND RESULTS

Request body required

```
{  "prompt": "If quantum computers become mainstream, cybersecurity will need to",  "max_new_tokens": 100,  "temperature": 0.3,  "top_p": 0.95,  "repetition_penalty": 1.5}
```

Execute

Request URL

http://127.0.0.1:8000/generate

Server response

Code	Details						
200	<p>Response body</p> <pre>{  "status": "success",  "generated_text": "If quantum computers become mainstream, cybersecurity will need to be a top priority for the government.\n\nThe government is facing a major challenge from the cybersecurity industry as it tries to find ways to protect its data. The government has been trying to get the technology to the public and to provide some kind of protection to its computers. But the security industry is not ready to give up on the idea of protecting its systems. So the Government is looking at ways that it can help the industry.\n\nIn the meantime, the Department of Homeland Security is working",  "parameters": {    "prompt": "If quantum computers become mainstream, cybersecurity will need to",    "max_new_tokens": 100,    "temperature": 0.3,    "top_p": 0.95,    "repetition_penalty": 1.5  }}</pre> <p>Response headers</p> <pre>access-control-allow-origin: *content-length: 788content-type: application/jsondate: Thu, 24 Apr 2025 13:03:50 GMTserver: uvicorn</pre> <p>Responses</p> <table><thead><tr><th>Code</th><th>Description</th><th>Links</th></tr></thead><tbody><tr><td>200</td><td>Successful text generation</td><td></td></tr></tbody></table> <p>Activate Windows No links</p>	Code	Description	Links	200	Successful text generation	
Code	Description	Links					
200	Successful text generation						





THANK YOU