1. Objective:

Create a generative text model using GPT or LSTM to generate coherent paragraphs based on specific topics.

1. Implementation Steps:
   1. Using GPT (Preferred for Advanced Results)
2. Set Up the Environment:

Install Python libraries:

Pip install transformers torch

1. Load a Pre-trained GPT Model: Use Hugging Face’s transformers library.

From transformers import GPT2LMHeadModel, GPT2Tokenizer

# Load pre-trained GPT-2 model and tokenizer

Model\_name = “gpt2”

Model = GPT2LMHeadModel.from\_pretrained(model\_name)

Tokenizer = GPT2Tokenizer.from\_pretrained(model\_name)

1. Generate Text Based on a Prompt:

Def generate\_text(prompt, max\_length=100):

Inputs = tokenizer.encode(prompt, return\_tensors=”pt”)

Outputs = model.generate(inputs, max\_length=max\_length, num\_return\_sequences=1, temperature=0.7)

Return tokenizer.decode(outputs[0], skip\_special\_tokens=True)

# Example usage

Prompt = “Explain the importance of renewable energy”

Print(generate\_text(prompt))

1. Test with User Prompts:

Input prompts like:

“Write a paragraph about artificial intelligence.”

“Explain the benefits of healthy eating habits.”

* 1. Using LSTM (For Learning Purpose)

1. Prepare Data:

Use a dataset (e.g., text corpus like Wikipedia or news articles).

Preprocess text (tokenization, padding).

1. Build an LSTM Model:

From keras.models import Sequential

From keras.layers import LSTM, Dense, Embedding

Model = Sequential([

Embedding(input\_dim=5000, output\_dim=128),

LSTM(128, return\_sequences=True),

LSTM(128),

Dense(128, activation=’relu’),

Dense(5000, activation=’softmax’)

])

Model.compile(optimizer=’adam’, loss=’categorical\_crossentropy’)

1. Train the Model:

Use a large text corpus for training.

Implement sequence prediction (next word/character).

1. Generate Text:

Feed a starting sequence and predict the next words iteratively.

1. Deliverable: Notebook

Prepare a Jupyter Notebook containing:

1. Introduction:

Brief explanation of generative text models (GPT vs. LSTM).

1. Code Implementation:

GPT model setup and usage.

(Optional) LSTM model training and text generation.

1. Examples:

User prompts and generated text outputs.

1. Conclusion:

Insights on the model’s performance.

Sample Notebook Structure:

Section 1: Introduction

Overview of text generation.

Explanation of GPT/LSTM.

Section 2: Setting up GPT

Install libraries, load model, and generate text.

Section 3: Results

Display generated paragraphs for different prompts.

Section 4: Observations

Comment on quality and coherence of text.