**Batch: C - 3 Roll No.: 16010122096**

**Experiment / assignment / tutorial No: 09**

|  |
| --- |
| **Title:** Text Generation using GPT-2 |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Objectives:**

- To introduce students to the concept of Generative AI.

- To understand how GPT-2 generates text using prompt-based inputs.

- To explore how different decoding strategies and hyper-parameters affect generated output.

**Expected Outcome of Experiment:**

|  |  |
| --- | --- |
| **Course Outcome** | **After successful completion of the course students should be able to** |
| **CO 4** | Analyze applications of AI and understand planning & learning processes in advanced AI applications |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# Resources & References

1. HuggingFace GPT-2: <https://huggingface.co/gpt2> , last retrieved on April 02,2025  
2. Illustrated Transformer: <https://jalammar.github.io/illustrated-transformer/> , last retrieved on April 02,2025  
3. GPT-2 Paper: <https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf> , last retrieved on April 02,2025  
4. HuggingFace Blog on Generation: <https://huggingface.co/blog/how-to-generate> , last retrieved on April 02,2025

5. <https://cloud.google.com/ai/generative-ai?hl=en> , last retrieved on April 02,2025  
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Text generation is a natural language processing (NLP) task where a model creates meaningful and coherent text based on a given input or prompt. It is a core capability of Generative AI and is commonly achieved using deep learning models, especially transformer-based architectures like GPT (Generative Pre-trained Transformer). These models are trained on massive corpora of text data and learn language patterns, grammar, and context. In text generation, the model predicts the next word (or token) one step at a time, using the context of the previous words, and continues until it reaches a desired length or stopping condition. Applications include story writing, code generation, chatbots, summarization, and creative content creation.

**Generative AI** refers to a branch of artificial intelligence that is capable of creating new content — such as text, images, music, or even code — based on patterns learned from large datasets. Unlike traditional AI that focuses on classification or prediction tasks, generative AI learns to produce original data that resembles its training input. In natural language processing (NLP), generative AI models like GPT can generate human-like text when given a prompt, making them useful for applications like chatbots, story writing, summarization, and more.

**GPT-2 (Generative Pre-trained Transformer 2)** is a powerful language model developed by OpenAI that uses deep learning to generate human-like text. It is based on the transformer architecture and was trained on a large corpus of internet text. Given a starting prompt, GPT-2 can continue writing coherent and contextually relevant sentences. It works by predicting the next word in a sequence using the context of previous words, making it capable of generating creative content, answering questions, translating text, and more.

In text generation, decoding strategies are methods used to determine which words the model should generate next. Common strategies include **Greedy Search** (picks the most probable word), **Beam Search** (keeps multiple best options), and Sampling methods like Top-k and Top-p that add randomness.

In the context of **text generation** using large language models like GPT-2, the process of "**hyper-parameter tuning**" refers to adjusting the generation parameters to control how the text is generated. These hyper-parameters influence the model’s **creativity**, **coherence, repetition, and overall quality of the output**.

## Common Hyper-parameters for Text Generation

|  |  |  |
| --- | --- | --- |
| **Hyperparameter** | **Description** | **Typical Range / Values** |
| max\_length | Total number of tokens (words + punctuation) to generate. | 20–100 |
| temperature | Controls randomness. Lower = more conservative, Higher = more creative. | 0.5 – 1.5 (default = 1.0) |
| top\_k | Randomly samples from the top K most probable next words. | 30–100 |
| top\_p (nucleus) | Chooses from the smallest set of tokens whose cumulative probability ≥ p. | 0.8 – 0.95 |
| do\_sample | Enables sampling instead of greedy or beam decoding. | True / False |
| repetition\_penalty | Penalizes repeated phrases. Higher = less repetition. | 1.0 (no penalty), 1.2+ |
| num\_beams | Number of beams used in **Beam Search** to explore multiple paths. | 1 (greedy), 3–10 |
| early\_stopping | Stops generation when all beams are finished. Used with beam search. | True / False |
| num\_return\_sequences | Number of output sequences to return for each input prompt. | 1 – 5 |

## How The Hyper-parameters Affect the Output:

|  |  |
| --- | --- |
| **Parameter** | **Effect on Output** |
| temperature=1.5 | More diverse and creative output |
| temperature=0.5 | Safer and more repetitive output |
| top\_k=50 | Random, but constrained to top 50 choices |
| top\_p=0.9 | More dynamic; considers word distribution |
| repetition\_penalty=1.2 | Avoids looping or repetitive phrases |
| num\_beams=5 | Improves fluency and quality at the cost of speed |

## Examples:

## Default (greedy):

*model.generate(inputs, max\_length=50)*

### High temperature (more creative):

*model.generate(inputs, max\_length=50, temperature=1.5)*

### Top-k sampling:

*model.generate(inputs, max\_length=50, do\_sample=True, top\_k=50)*

### Top-p (nucleus sampling):

*model.generate(inputs, max\_length=50, do\_sample=True, top\_p=0.9)*

### Beam search:

*model.generate(inputs, max\_length=50, num\_beams=5, early\_stopping=True)*

## Instructions for Students:

1. Open Google Colaband create a new notebook**.**
2. Copy and run the following base codeto generate text using the GPT-2 model:

------------------------------------------------------------------------------------------------

*!pip install transformers torch*

*from transformers import GPT2LMHeadModel, GPT2Tokenizer*

*# Load model and tokenizer*

*model = GPT2LMHeadModel.from\_pretrained("gpt2")*

*tokenizer = GPT2Tokenizer.from\_pretrained("gpt2")*

*# Write your prompt*

*prompt = “Artificial Intelligence will revolutionize education because”*

*inputs = tokenizer.encode(prompt, return\_tensors="pt")*

*# Generate text*

*outputs = model.generate(inputs, max\_length=50, temperature=0.7, num\_return\_sequences=1)*

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

*-------------------------------------------------------------------------------------------------------*

1. Now**,** experiment by adding different hyper-parameters to the generate() function one by one and in combinations. Some hyper-parameters you must try:
   * temperature
   * top\_k
   * top\_p
   * num\_beams
   * repetition\_penalty
   * do\_sample
   * max\_length
   * num\_return\_sequences
2. Run at least 5 different configurations, each with a new combination of hyper-parameters.
3. For each configuration, record:
   * The generated output
   * A short observation (Was it more creative? repetitive? random? logical?)

**Record your observations: (Add maximum outputs as you can add)**

**CODE:**

!pip install transformers torch

from transformers import GPT2LMHeadModel, GPT2Tokenizer

# load the pretrained GPT-2 model and its tokenizer

model = GPT2LMHeadModel.from\_pretrained("gpt2")

tokenizer = GPT2Tokenizer.from\_pretrained("gpt2")

# prepare input prompt and corresponding attention mask

encoded = tokenizer(

    "Artificial Intelligence will revolutionize education because",

    return\_tensors="pt",

    return\_attention\_mask=True

)

inputs = encoded["input\_ids"]

attention\_mask = encoded["attention\_mask"]

# generate text with your chosen hyper‑parameters

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=60,

    temperature=0.7,

    top\_k=50,

    do\_sample=True

)

# display the output clearly

print("\n=== Generation Output ===\n" +

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

**1]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=60,

    temperature=0.7,

    top\_k=50,

    do\_sample=True

)

Output received: Artificial Intelligence will revolutionize education because it will offer a new way to learn and to create new skills. We already have Google Glass, Apple's Siri, and Facebook's Face ID, which are all using artificial intelligence in their everyday lives, but AI will also allow us to share our experiences online

Explanation/Learning:

 **Coherence & focus:** The moderate temperature (0.7) plus top‑k sampling keeps the model on‐topic—it references familiar AI products (Google Glass, Siri) and links them sensibly to education.

 **Creativity vs. safety:** You get reasonably novel phrasing (“offer a new way to learn”) without risking complete off‐the‐wall text.

 **Brand/name drops:** The model draws on well‐known examples, demonstrating how top‑k encourages higher‐probability tokens (common brand names).

**2]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=60,

    temperature=1.5,

    do\_sample=True

)

Output received: Artificial Intelligence will revolutionize education because its core value is freedom in the knowledge of a self informed community, even if only with greater security.

The problem is that AI won't replace education, but that education will simply ensure that any potential teacher, and the community we have to educate you,

Explanation/Learning:

 **High creativity:** A temperature of 1.5 yields more abstract, philosophical language (“freedom in the knowledge of a self informed community”).

 **Risk of vagueness:** Phrases can become lofty and semantically loose.

 **Potential incoherence:** The second sentence trails off—longer sampling with high temperature can lead to meandering or incomplete thoughts.

**3]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=30,

    temperature=1,

    top\_p=0.9,

    do\_sample=True

)

Output received: Artificial Intelligence will revolutionize education because we will learn what is required in order to become smart. It is about empowering students, not just those who

Explanation/Learning:

 **Conciseness & abrupt cutoff:** The strict max\_length=30 forces a short, punchy statement, but it ends mid‑sentence.

 **Balanced randomness:** Temperature 1.0 with nucleus sampling (top‑p 0.9) still produces coherent language, but you sacrifice completion when you severely limit length.

**4]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=40,

    num\_beams=5,

    early\_stopping=True

)

Output received: Artificial Intelligence will revolutionize education because it will make it possible for students to learn about the world around them. It will also make it possible for students to learn about the world around them.

Explanation/Learning:

 **Deterministic & repetitive:** Beam search finds the highest‐probability sequences, often repeating phrases (“learn about the world around them”) to maximize likelihood.

 **Low creativity:** No sampling means minimal variation and a tendency to loop on the same idea.

 **Use case:** Good for safe, conservative text but poor if you want diversity.

**5]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=50,

    temperature=1.2,

    top\_p=0.8,

    repetition\_penalty=2.0,

    do\_sample=True

)

Output received: Artificial Intelligence will revolutionize education because of its ability to deliver a rich learning experience in students from all socioeconomic backgrounds."

- The New York Times.

Explanation/Learning:

 **Moderate creativity:** Temperature 1.2 + top‑p 0.8 yields varied phrasing without going completely off‑rails.

 **Repetition penalty effect:** Discourages verbatim reuse, so the text stays fresh.

 **Hallucination risk:** The model invented a “– The New York Times” citation, showing how sampling can introduce plausible but false attributions.

**6]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=70,

    temperature=1.0,

    top\_p=0.9,

    repetition\_penalty=1.5,

    do\_sample=True,

    num\_return\_sequences=3

)

Output received: Artificial Intelligence will revolutionize education because it is fundamentally transforming the way people work and what they see, not only from an algorithmic perspective but also to a practical use-case. We believe that AI may be applied beyond teaching jobs or simply by using them for good purposes like helping students learn different skills at school (like math), which isn't

Explanation/Learning:

* **Balanced creativity:** Temperature 1.0 + top\_p 0.9 yields coherent yet flexible responses.
* **Repetition penalty effect:** repetition\_penalty 1.5 minimizes redundancy across sequences.
* **Diversity boost:** num\_return\_sequences 3 lets you explore multiple valid continuations in one go.

**7]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=60,

    num\_beams=4,

    early\_stopping=True,

    repetition\_penalty=1.2,

    num\_return\_sequences=2

)

Output received: Artificial Intelligence will revolutionize education because it will make it possible for students to learn about the world around them.

"We're going to be able to teach people about the world around them, and we're going to be able to teach them how to think about the world around them,"

Explanation/Learning:

* **Deterministic quality:** num\_beams 4 produces high‑probability, well‑structured text.
* **Slight repetition control:** repetition\_penalty 1.2 reduces verbatim reuse without stifling fluency.
* **Safe variation:** early\_stopping + 2 beams balances variation with reliable stopping criteria.

**8]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=50,

    temperature=0.5,

    top\_k=30,

    do\_sample=True,

    repetition\_penalty=1.3

)

Output received: Artificial Intelligence will revolutionize education because it is the only way to develop new skills and knowledge in a rapidly changing world.

Explanation/Learning:

* **Conservative creativity:** Low temperature 0.5 + top\_k 30 focuses on top tokens for precise output.
* **Repetition control:** repetition\_penalty 1.3 prevents looping on the same phrases.
* **Precision over flair:** This setup yields clear, factual statements but limits imaginative phrasing.

**9]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=80,

    temperature=1.4,

    top\_k=90,

    top\_p=0.95,

    do\_sample=True,

    num\_return\_sequences=2

)

Output received: Artificial Intelligence will revolutionize education because they simply are not suitable for everyday work.

They're also capable of stealing jobs, and have caused massive health impacts (possibly more with these hacks than our society already knows when looking to avoid losing jobs due to malware-linked activity) due to the simple fact that machines lack the processing power left over before they should be able to use human creativity and

Explanation/Learning:

* **High creativity:** Temperature 1.4 + top\_p 0.95 fosters bold, varied language.
* **Broad sampling:** top\_k 90 opens the token pool wide, boosting unpredictability.
* **Coherence risk:** Such liberal settings can produce tangential or fragmented continuations.

**10]**

Parameters and their values:

outputs = model.generate(

    inputs,

    attention\_mask=attention\_mask,

    pad\_token\_id=tokenizer.eos\_token\_id,

    max\_length=65,

    num\_beams=6,

    repetition\_penalty=1.6,

    early\_stopping=True

)

Output received: Artificial Intelligence will revolutionize education because it will make it easier for students to learn about the world around them.

"It's going to change the way we think about the world," he said. "It's going to change the way we think about the future."

Explanation/Learning:

* **Polished structure:** num\_beams 6 with early\_stopping delivers concise, well‑formed outputs.
* **Strong repetition penalty:** repetition\_penalty 1.6 ensures minimal redundancy for a clean style.
* **Conservative tone:** Beam search–driven generation yields safe, deterministic text ideal for formal contexts.

**Post Lab Questions:**

1. What is a pre-trained language model? Give two examples.

A pre‑trained language model is a neural network trained on a large corpus of text to learn statistical patterns of language (grammar, semantics, context). After this general training, it can be fine‑tuned or used directly for downstream tasks.

* Examples: BERT, GPT‑3.

1. How does GPT-2 generate text?

GPT‑2 generates text autoregressively: given an initial prompt, it predicts the next token by sampling from the model’s probability distribution over its vocabulary. It then feeds that token back in, repeats the prediction step, and continues until reaching a stopping criterion (e.g., end‑of‑text token or max length).

1. What is the role of prompt engineering in Generative AI?

Prompt engineering involves crafting or formatting the input text (the “prompt”) to steer the model toward desired outputs. Good prompts can improve relevance, coherence, style, and factual accuracy without changing the underlying model.

1. List three applications of Generative AI in NLP.

 **Text summarization:** Condensing long documents into concise summaries.

 **Machine translation:** Converting text from one language to another.

 **Conversational agents:** Powering chatbots and virtual assistants for interactive dialogue.

**Conclusion:**

GPT-2 generates diverse text outputs, and tuning decoding parameters significantly impacts creativity, coherence, and repetition in generated responses.