

# GEN AI

## Lab-2

**NAME-** Dhrushaj Achar

**SRN-** PES2UG23CS171

**CLASS-** CSE 6C

Category 6: Creative Writing & Content (Expansion on Generation)

Project Title- Joke Punchline Generator

Goal: Create a system that takes a joke setup as input (e.g., "Why did the chicken cross the road?") and uses AI to generate a funny punchline to complete the joke.

Technology: Text generation using Hugging Face's `pipeline('text-generation')`

Abstract:

This project implements an AI-powered joke punchline generator that uses natural language processing to complete joke setups. The system leverages the GPT-2 language model through Hugging Face's Transformers library to generate creative and contextually relevant punchlines. Users can input any joke setup, and the AI will automatically generate a completion, making it useful for entertainment, creative writing, or understanding how language models work with humour.

What I understood from this project:

- Text Generation Pipeline: Hugging Face provides a simple pipeline('text-generation') interface that abstracts away the complexity of loading and using pre-trained language models.
- GPT-2 Model: GPT-2 is a transformer-based language model trained on a large corpus of text. It can predict what comes next in a sequence, making it suitable for completing joke setups.
- Parameters Matter: The quality of generated text depends on parameters like:
  - max\_new\_tokens: Controls how long the generated punchline will be
  - temperature: Controls randomness (higher = more creative, lower = more predictable)

- `do_sample`: Enables sampling for more diverse outputs

### Core Functionality:

- Loads the GPT-2 model using Hugging Face's Transformers library
- Takes joke setups as input (either from predefined examples or user input)
- Generates AI-powered punchlines using text generation
- Provides an interactive mode for users to test their own joke setups

### OUTPUT:

The screenshot shows a Jupyter Notebook interface with two main panels. The left panel displays the Python code for `joke_generator.py`, and the right panel shows the terminal output.

**Code in `joke_generator.py`:**

```
1  """
2  SRN- PES2UG23CS171
3  NAME- Dhrushaj Achar
4  CLASS- 6C CSE
5
6  """
7
8  import warnings
9  from transformers import pipeline
10
11 # Suppress warnings
12 warnings.filterwarnings('ignore')
13
14 # Initialize text generation pipeline
15 print("Loading AI model...")
16 generator = pipeline('text-generation', model='gpt2')
17
18 def generate_punchline(setup):
19     """Generate a punchline using AI text generation."""
20
21     prompt = f"{setup} "
22
23     # Generate text
24     result = generator(
25         prompt,
26         max_new_tokens=30,
27         num_return_sequences=1,
28         temperature=0.8,
29         do_sample=True,
30         pad_token_id=generator.tokenizer.eos_token_id
31     )
32
33     # Extract the generated text and remove the setup
34     generated_text = result[0]['generated_text']
35     punchline = generated_text[len(setup):].strip()
36
37     return punchline
38
```

**Terminal Output:**

```
Punchline: "
The witness answered: "Yeah, we could hear it. It was in the distance.
"I took pictures. It was
Enter joke setup:
Thanks for the laughs!
PS C:\Users\dhrus\Downloads> python joke_generator.py
Loading AI model...
Device set to use cpu
=== Joke Punchline Generator ===
Examples:
Setup: Why did the chicken cross the road?
Punchline: "A cow would have been a good target for a hit, but you knew right away what was on the way. "The cattle were still
Setup: Why do programmers prefer dark mode?
Punchline: The most obvious answer is that it has nothing to do with the code, it's just that when you run it, it gets messed up.
Setup: How many developers does it take to change a light bulb?
Punchline: Does it take half a person to change a light bulb? Does it take five to change a light bulb? How long do
Setup: Knock knock!
Punchline: ____ | | "I'm sure you're not exactly the sort of person to expect things from. I don't just mean an insult to the
--- Try your own! ---
Enter joke setup: Why is Dhrushaj the smartest engineer?
Punchline: It seems that the answer to Dhrushaj's question is that he is actually the smartest of the lot. He knows the fundamentals
Enter joke setup: Why are women bad drivers?
Punchline: Let's look at why they are. Let's say that she gets caught in a car and doesn't go home. The
Enter joke setup: PES is the best university
Punchline: of the best in the world.
My own personal favorite area is my school, of course, and while it's not as big as I
Enter joke setup: How to make an engineer laugh
Punchline: _____
I think it's ok to laugh if you like. Just be honest.
In my experience it's easy enough to do if
Enter joke setup: []
```

### Technical Implementation:

- Used `pipeline('text-generation', model='gpt2')` as the core AI engine
- Configured generation parameters for optimal joke completion
- Implemented warning suppression for cleaner output
- Added error handling for graceful exit (Ctrl+C)

### User Experience:

- Displays example jokes with AI-generated punchlines on startup
- Provides an interactive loop where users can input custom joke setups
- Clean, formatted output showing both setup and generated punchline

### How It Works

- The program initializes by loading the GPT-2 model (downloads on first run)
- When given a joke setup, it treats it as a text prompt
- The AI model analyses the context and generates a continuation
- The generated text is extracted and presented as the punchline
- Users can try multiple jokes in the interactive mode

### Challenges Faced

- Warning Messages: Initially, the model generated warnings about conflicting parameters. I resolved this by using `max_new_tokens` instead of `max_length` and suppressing unnecessary warnings.
- Model Size: The GPT-2 model is about 500MB, so the first run requires downloading it.
- Output Quality: Text generation can sometimes produce unexpected results, so parameter tuning was necessary.

### Future Improvements

- Add support for larger models (GPT-2 Medium/Large) for better joke quality
- Implement fine-tuning on a joke dataset for more humorous outputs
- Add a rating system to evaluate punchline quality
- Create a web interface for easier access

### Conclusion

This project successfully demonstrates the application of AI text generation for creative purposes. By using the Hugging Face Transformers library and the GPT-2 model, I was able to create a functional joke punchline generator that showcases how modern NLP models can understand context and generate human-like text. The project helped me understand the practical implementation of text generation pipelines and the importance of parameter tuning in AI applications.