AI-Powered Text Completion

David Cantuña

Introduction

This report outlines the process of designing, implementing, and evaluating a text generation application powered by artificial intelligence. The main goal of the project is to give users a practical understanding of Generative AI by building a Python-based program that communicates with a pre-trained language model via an API. The application allows users to input text prompts, which are then sent to an AI model that generates a relevant response. For this implementation, the Hugging Face Inference API was employed, using the HuggingFaceH4/zephyr-7b-beta model, a lightweight conversational language model designed for efficient natural language generation.

Part 1: Application Development

The application was developed using the Python programming language in Google colab. It utilizes the requests library to communicate with the Hugging Face Inference API through direct HTTP POST requests. Upon execution, the user is prompted to securely enter their API token using the getpass module. This design ensures the API key is never hard-coded or exposed during the session.

The core of the application revolves around a function that takes a user prompt and additional text generation parameters. It performs input validation (e.g., preventing empty or overly long prompts) and sends the request to the AI model. If the response is successful (status code 200), the generated text is displayed.
The application gracefully handles errors such as unauthorized access (401), invalid requests (422), or network issues (timeouts).
The interface is text-based, prompting the user for repeated input until they enter an exit command ("quit", "stop", or "exit"). This structure makes it easy to use in both local terminal environments and Google Colab.

Part 2: Parameter Implementation

To allow customization and experimentation, the application supports three main parameters:

Temperature: Controls the randomness of the output. Lower values (e.g., 0.2) produce more focused, deterministic results, while higher values (e.g., 0.9) generate more creative and diverse text.

Top-p (nucleus sampling): Limits generation to the smallest possible set of words whose cumulative probability exceeds the value of top-p. This helps filter out low-probability words while maintaining creativity.

Max new tokens: Determines how many new tokens (words/pieces) the model can generate in response to a prompt. Larger values yield longer responses.

Each of these parameters is requested from the user before sending a prompt. If the user provides no input, the application uses default values (e.g., temperature = 0.7, top_p = 0.9, max_new_tokens = 50). Input validation is included to ensure parameters are within acceptable bounds.

Part 3: Findings and Observations

To evaluate the impact of each parameter, several experiments were conducted with identical prompts while varying the parameters one at a time. The observations are as follows:

Temperature:
At low values, the model tends to produce repetitive or safe responses.
At high values, the outputs become more creative but sometimes incoherent or off-topic.

Top-p:
A top-p value around 0.9 results in balanced, diverse outputs.
Lowering top-p lees than 0.5 leads to more conservative responses, reducing novelty.

Max new tokens:
Smaller values produce very short, sometimes incomplete answers.
Larger values provide rich detail, but may increase generation time.

Combining parameters like high temperature with high top-p tends to amplify creativity, while low values for both result in strict, robotic responses. Selecting parameter values thus depends on the desired use case—whether deterministic completions or imaginative content is preferred.

Conclusion

This project was very interesting for me and demonstrated the flexibility and simplicity of building a text completion application using Hugging Face's hosted models. By integrating parameter customization and robust input handling, the application supports dynamic experimentation. Through analysis, it is evident that tuning generation parameters significantly affects the style and quality of responses. This hands-on experience provides foundational insights for developing more advanced AI-driven tools in natural language processing.

Appendix: Sample Prompts for Testing

- "in the mountains"

```
Enter your prompt: in the mountains
Set temperature (creativity, 0.0-1.0) [0.7]: 0.1
Set max new tokens (response length) [50]: 100
Set top_p (nucleus sampling, 0.0-1.0) [0.9]: 0.1

Generating response...

AI Response:
in the mountains of the north

where the air is crisp and clear

and the leaves turn gold and red

in the autumn's gentle breeze

there's a place where time stands still

where the world is quiet and serene

and the only sound is the rustle

of leaves beneath your feet

in the mountains of the north

where the sky is a canvas of blue
```

- "tell me about cars"

```
Enter your prompt: tell me about cars
Set temperature (creativity, 0.0-1.0) [0.7]: 0.3
Set max new tokens (response length) [50]: 100
Set top_p (nucleus sampling, 0.0-1.0) [0.9]: 0.9

Generating response...

AI Response:
tell me about cars,

their engines, their sounds,

the way they move through the world,

their sleek lines and curves.

i want to know about the feel of the wheel in my hands,

the wind rushing past my face,

the thrill of acceleration,

the freedom of the open road.

tell me about the history of cars,

the innovations that have shaped them,
```

- "Ecuador"

```
Enter your prompt: Ecuador
Set temperature (creativity, 0.0-1.0) [0.7]: 0.8
Set max new tokens (response length) [50]: 120
Set top_p (nucleus sampling, 0.0-1.0) [0.9]: 0.3

Generating response...

AI Response:
Ecuadorian President Rafael Correa has been a strong advocate for the country's indigenous communities, and has worked to promote their rights and culture. In a recent speech, Correa

Correa began his speech by acknowledging the historical injustices that have been inflicted upon indigenous peoples, including forced assimilation and cultural erasure. He noted that
```

- "Explain recursion like I’m five"

```
Enter your prompt: Explain recursion like I’m five
Set temperature (creativity, 0.0-1.0) [0.7]: 0.1
Set max new tokens (response length) [50]: 100
Set top_p (nucleus sampling, 0.0-1.0) [0.9]: 0.9

Generating response...

AI Response:
Explain recursion like I’m five.

Recursion is like a puzzle that has the same pieces as the picture on the box. You can put the pieces together to make the picture, but sometimes you have to take some pieces apart
```

References

- Hugging Face Inference API Documentation: https://huggingface.co/docs/api-inference
- Zephyr-7B-beta Model Card: https://huggingface.co/HuggingFaceH4/zephyr-7b-beta
- Sampling Parameters in Language Models: https://huggingface.co/blog/how-to-generate