Overview

The Sustainable Development Goals (SDG) Text Generation Application was developed to generate coherent and creative text based on user input prompts related to the SDGs. By leveraging state-of-the-art generative AI models, the application enables users to explore various SDG topics through AI-generated narratives. It also compares the performance of different models to determine which is the most effective for this task.

Key Features

1. **Interactive Text Generation**: Users can input SDG-related prompts, and the application generates text that aligns with the provided topic.
2. **Model Comparison**: The application evaluates different models, such as GPT-2, BLOOM, and GPT-Neo, comparing outputs based on fluency, relevance, creativity, and grammatical correctness.
3. **User Interface**: Built using Streamlit, the interface is designed to be user-friendly, allowing easy prompt entry and seamless interaction with AI models.
4. **Performance Metrics**: The application evaluates generated text on the following criteria:
   - **Fluency**: The natural flow and coherence of the text.
   - **Relevance**: The alignment between the generated text and the user’s input.
   - **Creativity**: Originality and diversity of ideas.
   - **Grammatical Correctness**: The adherence to grammar rules.

Tools and Libraries

- **Python**: The primary programming language used for this project.
- **Transformers Library**: Provided by Hugging Face, it offers easy access to pre-trained models, such as GPT-2, BLOOM, and GPT-Neo.
- **Torch Library**: Used to handle deep learning computations.
- **Streamlit and Gradio**: Libraries used to create an interactive user interface.

Model Selection

Two models were primarily evaluated:

1. **GPT-2**: Known for producing coherent and high-quality text, it was used due to its ease of integration and versatile performance across prompts.
2. **BLOOM (560m)**: Chosen for evaluation due to its balance between computational efficiency and performance. The model demonstrated strong capabilities in generating relevant and contextually accurate text.

Implementation Process

1. **Setting up the Environment**: The development environment was set up using Google Colab for its accessible resources and efficient Python integration. Libraries such as transformers, torch, and Streamlit were installed to handle AI model operations and build the user interface.

2. **Model and Tokenizer Initialization**: The model and tokenizer were initialized using Hugging Face's transformers library. This step ensured that the application could handle text generation efficiently. For instance, BLOOM-560m was loaded using the following code:

```
model_name = "bigscience/bloom-560m"

tokenizer = AutoTokenizer.from_pretrained(model_name)

model = AutoModelForCausalLM.from_pretrained(model_name)
```

3. **Text Generation Function:** A simple text generation function was implemented to generate output based on user prompts. Here's the code:

```
def generate_text (prompt, max_length=100):

inputs = tokenizer (prompt, return_tensors="pt")

outputs = model. generate(inputs["input_ids"], max_length=max_length,
num_return_sequences=1)

return tokenizer. decode (outputs [0], skip_special_tokens=True)
```

4. **User Interface:** Streamlit and Gradio were used to create an interactive and accessible interface. Users can enter SDG-related prompts and instantly receive AI-generated text. The Streamlit framework allows for a clean and responsive UI:

```
def main ():

st. title ("SDG Text Generation App")

prompt = st.text_input ("Enter your prompt related to SDGs")

if prompt:

generated_text = generate_text(prompt)

st. write ("Generated Text:", generated_text)
```

Evaluation

To evaluate the performance of the models, several SDG-related prompts were used (e.g., “How can we achieve affordable and clean energy?”). The following criteria were used for assessment:

1. Coherence and Creativity: Models were tested for their ability to produce logically consistent and creative outputs. BLOOM performed well, generating diverse ideas while maintaining coherence.
2. Relevance: The models were evaluated on how closely their outputs aligned with the prompt topics. GPT-2 generally delivered more relevant text compared to BLOOM, which occasionally produced less focused responses.
3. Grammatical Correctness: Both models performed similarly in terms of grammar, but GPT-2 had a slight advantage in generating more readable and grammatically sound text.

Results and Conclusion

GPT-2 was selected for the final implementation due to its superior performance in fluency, relevance, and overall readability. Although BLOOM-560m demonstrated strengths in creativity and computational efficiency, it was occasionally less consistent in aligning with the given prompts. Future improvements could involve incorporating larger models or fine-tuning existing ones for SDG-specific content.

By comparing multiple models, the SDG Text Generation Application offers valuable insights into how generative AI can be applied to promote and explore global development goals. Future iterations could integrate more sophisticated user interfaces, enhance performance, and fine-tune models to focus on specific SDGs.