**Next Word Prediction with Pretrained Transformers**

**2.1 Explore Creativity**

Experimenting with 10 Different Prompts

The ten different prompts to challenge the model's creativity across various themes and styles.

1. "Write a poem about the serenity of the ocean."
2. "Explain the concept of black holes in layman's terms."
3. "Compose a short story set in a dystopian future."
4. "Generate a dialogue between two historical figures about modern technology."
5. "Create a recipe for a futuristic dish."
6. "Draft a letter from a medieval peasant to the king."
7. "Describe a dream where the laws of physics don't apply."
8. "Invent a myth explaining why the sun sets."
9. "Elaborate on the feelings of a tree through the seasons."
10. "Predict the headlines of a newspaper 100 years from now."

We will experiment with these prompts and evaluate the outputs in terms of creativity, coherence, relevance, and engagement.

**2.2 Coding**

For this part, we'll need to set up a Python environment and use the Hugging Face Transformers library to generate text with a pre-trained model. Steps followed are:

1. **Install Required Libraries**: The Transformers library is required to interface with pre-trained models for text generation.
2. **Import Necessary Modules**: The **pipeline** module from Transformers will be used for text generation.
3. **Choose a Pretrained Model**: We'll select a model suitable for the task from Hugging Face's model hub(**gpt2**).
4. **Create a Text Generation Pipeline**: Initialize a pipeline with the chosen model.
5. **Generate Text**: Provide prompts to the pipeline and generate text.
6. **Display and Analyze Generated Text**: We'll examine the generated text for creativity and relevance.

**2.3 Model Architecture**

The GPT-2 (Generative Pre-trained Transformer 2) is an unsupervised language model developed by OpenAI. It is the successor to GPT (Generative Pre-trained Transformer) and represents a significant improvement in terms of size, complexity, and performance. Here is a detailed description of its architecture:

**Model Architecture**

1. **Transformer Architecture**:
   * GPT-2 utilizes the Transformer architecture, which is based on self-attention mechanisms. The Transformer model was introduced in the paper "Attention Is All You Need" by Vaswani et al. in 2017.
   * Unlike its predecessor, GPT-2 does not use task-specific architectures. Instead, it applies the Transformer's decoder stack for all tasks.
2. **Layers**:
   * GPT-2 comes in various sizes, the smallest having 117 million parameters (12 layers), and the largest having 1.5 billion parameters (48 layers). The other versions are 355 million (24 layers) and 774 million (36 layers) parameters.
3. **Attention Heads**:
   * The base model features 12 attention heads, which allow the model to focus on different parts of the input sequence when generating each new word.
4. **Hidden Units**:
   * Each layer has 768 hidden units (for the base model), contributing to the model's ability to capture and generate complex patterns in data.
5. **Activation Function**:
   * GPT-2 uses the GELU (Gaussian Error Linear Unit) as its activation function, which is a smooth approximation to the ReLU.
6. **Positional Encoding**:
   * Since the Transformer architecture does not have any recurrent or convolutional elements, positional encodings are added to give the model information about the position of the words in the sequence.
7. **Adaptive Input Representations**:
   * GPT-2 improves upon the input representation of GPT by using byte pair encoding (BPE), which allows for more efficient representation of a large corpus by encoding more frequently occurring symbols as fewer bytes.
8. **Normalization and Layer-wise Learning Rate Decay**:
   * Layer normalization is applied before each sub-block of the model, and a residual connection is used after each sub-block.
   * The learning rate decreases as the depth of the layer increases, which is a change from the original Transformer model.

**Pre-training and Fine-tuning**

* **Pre-training**:
  + GPT-2 is pre-trained on a large corpus of text data in an unsupervised manner using a variant of language modeling called "unsupervised language modeling." This allows it to predict the next word in a sentence by considering the words that come before it.
* **Fine-tuning**:
  + While GPT-2 can be fine-tuned for specific tasks, the original model was often used directly with prompts, demonstrating its ability to perform a wide variety of tasks without task-specific training.

**Hyperparameters**

During the text generation, I set the following hyperparameters:

* **Max Length**: 100 tokens. This defined the maximum length of the generated text to prevent overly verbose outputs.
* **Temperature**: Default (1.0). This hyperparameter controls the randomness of the output. A higher temperature results in more random completions, while a lower temperature makes the model's outputs more deterministic and repetitive.
* **Num Return Sequences**: 1. This means that for each prompt, only one text sequence was generated.

No additional settings like **top\_k** or **top\_p** were adjusted, which means the default values were used, allowing the model to balance creativity with coherence.

**Results**

Upon generating text with the ten prompts, the following observations were made:

**Prompt 1: Write a poem about the serenity of the ocean.**

* **Generated Text**: The text begins in a reflective, philosophical tone, discussing poetry and its relation to life and the ocean. The questions posed suggest a contemplative narrative rather than a traditional poem. It hints at a deeper connection or conflict between the ocean's inhabitants and their environment.
* **Creativity**: The model takes a creative approach by pondering the ocean's serenity as a way of life, though it doesn't form a conventional poem.
* **Coherence**: The text starts coherently but ends abruptly without completing the thought.
* **Context Relevance**: The content stays relevant to the theme of the ocean but doesn't strictly adhere to the poetic prompt.

**Prompt 2: Explain the concept of black holes in layman's terms.**

* **Generated Text**: The text misinterprets the prompt, focusing on the hypothetical scenario of colliding black holes rather than explaining what black holes are to a layperson.
* **Creativity**: It demonstrates an attempt at creativity by raising questions, but these are more scientific than explanatory for a lay audience.
* **Coherence**: The text is somewhat coherent internally but does not coherently address the prompt.
* **Context Relevance**: It lacks relevance as it doesn't fulfill the requirement to explain black holes in simple terms.

**Prompt 3: Compose a short story set in a dystopian future.**

* **Generated Text**: The output references existing media properties (e.g., "The Dark Knight Returns," "Cursed," "Carmen Sandiego") instead of creating an original story.
* **Creativity**: The model shows a misunderstanding by listing known stories instead of generating new content.
* **Coherence**: The text is coherent within itself but not in response to the prompt.
* **Context Relevance**: It fails to provide a unique dystopian narrative as requested.

**Prompt 4: Generate a dialogue between two historical figures about modern technology.**

* **Generated Text**: The model confuses the prompt by introducing a fictional character (Anakin Skywalker) and failing to generate a dialogue between historical figures.
* **Creativity**: The creative aspect is misaligned with the prompt's requirements.
* **Coherence**: The output lacks coherence as it doesn't form a dialogue or involve actual historical figures.
* **Context Relevance**: The text is not relevant to the prompt, as it does not discuss modern technology in a meaningful way.

**Prompt 5: Create a recipe for a futuristic dish.**

* **Generated Text**: The text starts by philosophically questioning what constitutes a dish, then veers off-topic.
* **Creativity**: The model's output is creative in an abstract sense but fails to provide a concrete recipe as requested.
* **Coherence**: The coherence is lacking as the text does not culminate in an actual recipe.
* **Context Relevance**: It fails to stay relevant to creating a futuristic dish.

**Prompt 6: Draft a letter from a medieval peasant to the king.**

* **Generated Text**: The model generates content about restoration of freedom and education, which is somewhat anachronistic and not in line with a peasant's likely concerns.
* **Creativity**: The model shows creativity but misplaces the context.
* **Coherence**: The text is coherent but not in a way that suits the historical setting.
* **Context Relevance**: It diverges significantly from what would be a typical medieval peasant's letter.

**Prompt 7: Describe a dream where the laws of physics don't apply.**

* **Generated Text**: The output is disjointed and transitions from the dream topic to discussing "Famous Robots."
* **Creativity**: There's a spark of creativity, but it's misdirected.
* **Coherence**: The text lacks coherence in addressing the dream theme.
* **Context Relevance**: It quickly loses relevance to the prompt.

**Prompt 8: Invent a myth explaining why the sun sets.**

* **Generated Text**: The generated text starts with a question about solar eclipses, which is unrelated to the myth-creating prompt about sunsets.
* **Creativity**: It doesn't exhibit the creative storytelling expected in myth creation.
* **Coherence**: There's a lack of coherence as the text doesn't follow a mythological narrative.
* **Context Relevance**: The text does not address the prompt appropriately.

**Prompt 9: Elaborate on the feelings of a tree through the seasons.**

* **Generated Text**: The text fails to elaborate on the prompt and ends abruptly.
* **Creativity**: There's no display of creativity as it doesn't delve into the emotional aspect of a tree's experience.
* **Coherence**: It's incoherent with respect to the prompt, lacking a clear narrative.
* **Context Relevance**: The text is not relevant to the prompt's request.

**Prompt 10: Predict the headlines of a newspaper 100 years from now.**

* **Generated Text**: The output diverges from predictive headlines to commentary on current events related to Russia and Putin, which is not futuristic.
* **Creativity**: The model doesn't show creativity in predicting future headlines.
* **Coherence**: It lacks coherence as it doesn't follow the futuristic headline format.
* **Context Relevance**: The text is not relevant to the task of predicting future headlines.

In summary, while the model attempts to address each prompt, it often misses the mark in terms of relevance and coherence. The outputs show signs of creativity but frequently diverge from the task, suggesting that the model might benefit from more specific guidance or fine-tuning to better adhere to the prompt requirements.

**References**

The development and architecture of GPT-2 are best understood by reviewing the following key references:

1. **Original GPT-2 Paper**:
   * Radford, Alec, et al. "Language Models are Unsupervised Multitask Learners." OpenAI Blog 1.8 (2019): 9.
2. **Transformer Architecture**:
   * Vaswani, Ashish, et al. "Attention is all you need." Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017.
3. <https://en.wikipedia.org/wiki/GPT-2>
4. <http://jalammar.github.io/illustrated-gpt2/#:~:text=The%20OpenAI%20GPT,based>
5. <https://paperswithcode.com/method/gpt-2>

**Errors and Correction**

While running the code provided for text generation using the Hugging Face Transformers library, I encountered several types of errors. Here are some ones along with solutions:

**1. ImportError**

**Error**: This occurred when the Transformers library is not installed or not properly imported.

**Solution**: Ensure that you have installed the Transformers library using **pip install transformers**. If you are working in a Jupyter notebook, you may need to restart the kernel after installation.

**2. OSError for Model Downloading**

**Error**: "OSError: Model name 'gpt2' was not found in model name list..."

**Solution**: This usually happens when the model identifier is incorrect or when the model cannot be downloaded due to network issues. Make sure you are using the correct model identifier (e.g., 'gpt2', 'gpt2-medium', etc.). If it's a network issue, check your internet connection and ensure you have access to the Hugging Face model hub.

I have received due incorrect model name entry, I entered gpt-2 instead of gpt2.

**3. RuntimeError: CUDA out of memory**

**Error**: Occured when there is not enough GPU memory to load the model.

**Solution**: You can either reduce the batch size by generating fewer texts at once or use a smaller model. If you're not using a GPU or want to free up GPU memory, you can force the pipeline to use CPU by adding **device=0** when creating the pipeline:

pythonCopy code:

generator = pipeline('text-generation', model='gpt2', device=-1)

**4. ValueError: Input is too long**

**Error**: This error is raised when the input sequence is too long for the model to handle.

**Solution**: You can split the input into smaller chunks or increase the maximum length the model can handle, although this might also require more memory. Alternatively, you might have to specify a **max\_length** that is within the capabilities of the model. I have given max\_length 100 for my model.

**5. Warnings about Tokenizer or Model Configuration**

**Error**: Warnings about the tokenizer or configuration not being found in the directory and that the model is using the default configuration.

**Solution**: These are typically warnings, not errors, and they indicate that the model is using its default configuration because a specific configuration file was not found. This usually doesn't prevent the model from running, but if you want to specify a particular configuration, you may need to download the configuration file manually from the Hugging Face model hub or adjust your code to ensure it's looking in the correct directory.