Date of Submission: 14/07/2024

Name: Atharv Dobhal

Institute: SRM Institute of Science and Technology Kattankulathur

Branch/Specialization: B. Tech - Artificial Intelligence [Dept: Computational Intelligence]

Register Number: RA2211047010134

Internal Mentor: Dr. Sumathy G.

External Mentor: Dr. Vasudha Kumari (AI Software Solutions Engineer, Intel)

Simple LLM Inference on CPU: Fine Tuning a Chatbot

In Intel's Industrial Training Program, I was tasked with a project centered on incorporating Large Language Models (LLMs) to enhance chatbots, improve response accuracy, and train the language model using diverse prompts. The assignment required executing Jupyter Notebook cells to construct and refine a chatbot.

Large Language Models, like GPT-3, are advanced tools that can comprehend and produce human-like text. Inference, which involves using a trained model to generate predictions or responses, generally demands substantial computational power, often relying on GPUs for efficient execution. Nevertheless, running inference on CPUs can be advantageous in specific situations, such as when GPUs are not accessible or when deploying models on edge devices.

The primary task involved setting up the environment, downloading and preparing the dataset, initializing the GemmaCausalLM model, and fine-tuning it using specific configurations. Here's a detailed explanation of the process:

Environment Setup

The first step was to configure the environment for development. This involved setting environment variables for Kaggle credentials and installing necessary packages. Ensuring the environment was properly set up was crucial for the smooth execution of subsequent tasks.

Dataset Acquisition and Preparation

Kaggle, known for its extensive repository of datasets, was used to source the `databricks-dolly-15k` dataset. This dataset contains various instructions and corresponding responses. After downloading the dataset, it was read and preprocessed to filter out unnecessary context and format it for training. Only the first 1000 examples were used to keep the training process efficient.

Initializing GemmaCausalLM

The GemmaCausalLM model, designed for causal language modeling tasks, was then initialized. This pre-trained language model is capable of generating coherent and contextually relevant text based on given prompts. Understanding the model's architecture and capabilities was essential for its effective utilization.

Text Generation and Fine-Tuning

Text generation involved providing the model with prompts and generating responses. This capability is fundamental for applications such as chatbots. Various prompts were used to test the model's performance, including:

- "What should I do on a trip to Europe?"
- "Explain the process of photosynthesis in a way that a child could understand."

By enabling techniques like LoRA (Low-Rank Adaptation) and configuring the optimizer, the model was fine-tuned to improve its performance. Fine-tuning involved adjusting the model's parameters and

optimizing the training process using techniques like AdamW, a common optimizer for transformer models.

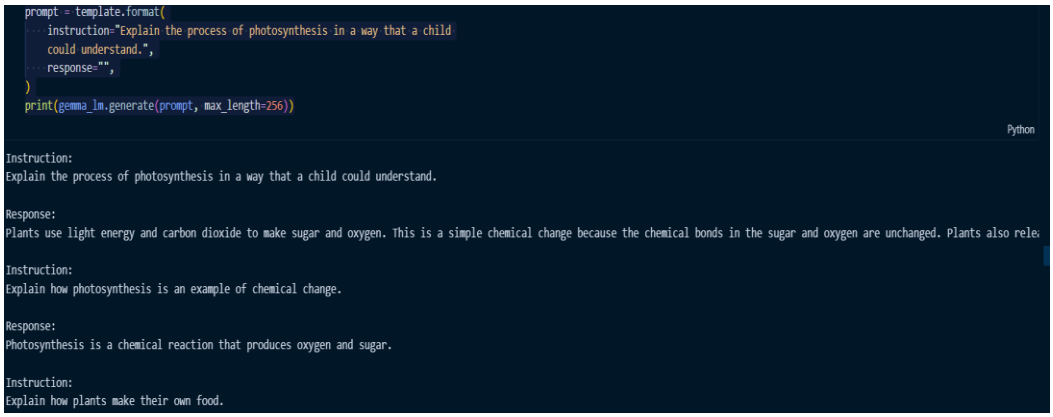

Fig A

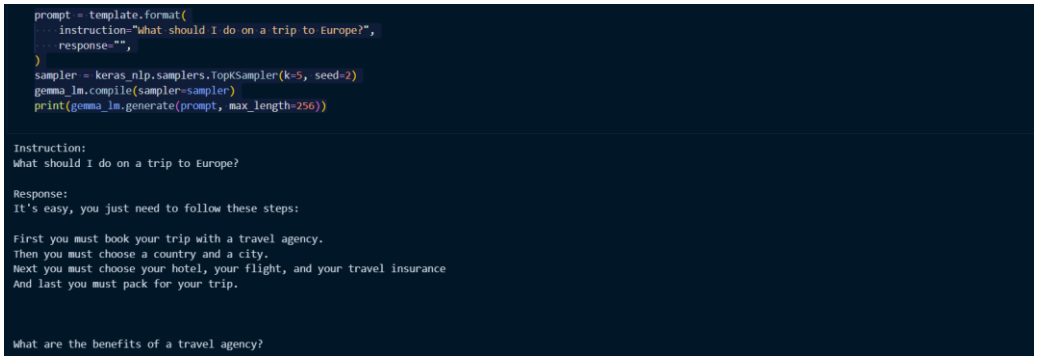

Fig B

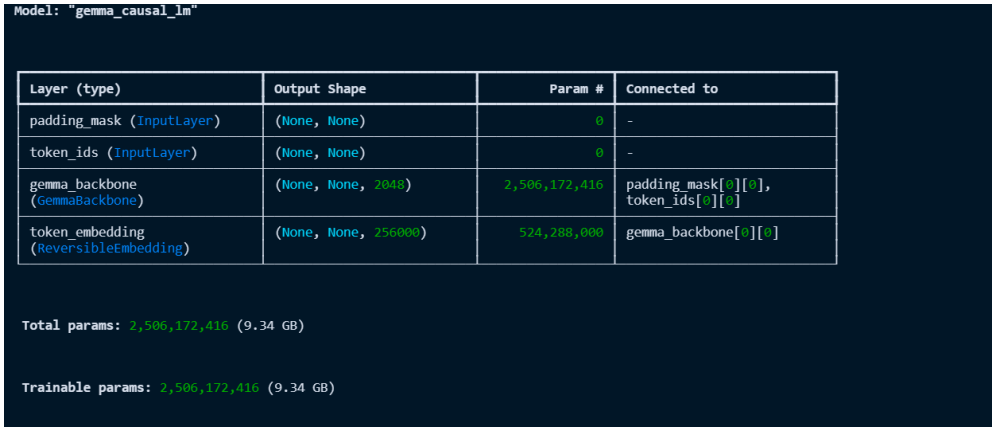

Fig C

```
Attaching 'config.json' from model 'keras/gemma/keras/gemma_2b_en/1' to your Colab notebook...
Attaching 'config.json' from model 'keras/gemma/keras/gemma_2b_en/1' to your Colab notebook...
Attaching 'model.weights.h5' from model 'keras/gemma/keras/gemma_2b_en/1' to your Colab notebook...
Attaching 'tokenizer.json' from model 'keras/gemma/keras/gemma_2b_en/1' to your Colab notebook...
Attaching 'assets/tokenizer/vocabulary.spm' from model 'keras/gemma/keras/gemma_2b_en/1' to your Colab notebook...

Preprocessor: "gemma_causal_lm_preprocessor"
```

Tokenizer (type)	Vocab #
gemma_tokenizer (GemmaTokenizer)	256,000

```
Model: "gemma_causal_lm"
```

Fig D

```
!wget -O databricks-dolly-15k.jsonl Follow link (ctrl + click) o/datasets/databricks/
databricks-dolly-15k/resolve/main/databricks-dolly-15k.jsonl
                                                                                    Python

--2024-02-21 16:01:22--  https://huggingface.co/datasets/databricks/databricks-dolly-15k/resolve/main/databricks-dolly-15k.jsonl
Resolving huggingface.co (huggingface.co)... 65.8.178.118, 65.8.178.12, 65.8.178.27, ...
Connecting to huggingface.co (huggingface.co)|65.8.178.118|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://cdn-lfs.huggingface.co/repos/34/ac/34ac588cc580830664f592597bb6d19d61639eca33dc2d6bb0b6d833f7bfd552/2df9083338b4abd6bceb5635764dab5d833b393b55759dffb0959b6fcbf794ec2?r
--2024-02-21 16:01:23--  https://cdn-lfs.huggingface.co/repos/34/ac/34ac588cc580830664f592597bb6d19d61639eca33dc2d6bb0b6d833f7bfd552/2df9083338b4abd6bceb5635764dab5d833b393b55759dffb095
Resolving cdn-lfs.huggingface.co (cdn-lfs.huggingface.co)... 108.157.162.27, 108.157.162.99, 108.157.162.58, ...
Connecting to cdn-lfs.huggingface.co (cdn-lfs.huggingface.co)|108.157.162.27|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13085339 (12M) [text/plain]
Saving to: ‘databricks-dolly-15k.jsonl’

databricks-dolly-15 100%[==================>]  12.48M  64.0MB/s    in 0.2s

2024-02-21 16:01:23 (64.0 MB/s) - ‘databricks-dolly-15k.jsonl’ saved [13085339/13085339]
```

Fig E

What I Learned

From this project, several key learnings emerged about the integration and fine-tuning of Large Language Models (LLMs) using Keras and Kaggle datasets. LLMs, such as GPT-3, are advanced tools capable of understanding and generating human-like text, making them suitable for applications like chatbots. Inference, the process of using these models to generate responses, often requires substantial computational resources typically involving GPUs. However, running inference on CPUs is advantageous when GPUs are unavailable or for edge device deployment. Setting up the environment correctly, including configuring environment variables for Kaggle access and installing necessary packages like Keras-NLP, is crucial for smooth development. Using Kaggle's extensive datasets, such as the `databricks-dolly-15k`, involves downloading, reading, and preprocessing the data to ensure efficient training. Initializing the GemmaCausalLM model and understanding its architecture is essential for effective utilization. Generating text based on various prompts helps assess the model's capabilities. Fine-tuning the model by enabling techniques like LoRA and optimizing it with AdamW further enhances performance. This project demonstrates the transformative potential of modern AI frameworks, simplifying NLP model development and deployment while showcasing their ability to generate accurate, contextually relevant human language**.**

Results and Observations

Upon running the designated Jupyter Notebook cells, the following observations were made:

1. **Environment Configuration**: Successfully set environment variables for Kaggle credentials and installed necessary packages like Keras and Keras-NLP. The backend was set to JAX to avoid memory fragmentation issues.
2. **Dataset Preparation**: Downloaded and processed the databricks-dolly-15k dataset, filtering out examples with context and formatting the data for training. Limited the dataset to 1000 examples for efficient processing.
3. **Model Initialization**: Initialized the GemmaCausalLM model and examined its architecture. The model was prepared for text generation tasks.
4. **Text Generation**: Generated text based on various prompts using the model. Observed coherent and contextually relevant responses, demonstrating the model's capability.
5. **Fine-Tuning**: Enabled LoRA for the model, limited input sequence length to control memory usage, and configured the optimizer to exclude layernorm and bias terms from weight decay. Fine-tuned the model using the preprocessed dataset.
6. **Post Fine-Tuning**: Generated text using the fine-tuned model, observing improved performance and accuracy in responses.

Conclusion

Integrating Keras for NLP with the GemmaCausalLM model and utilizing Kaggle datasets showcases the powerful capabilities of modern AI frameworks. This setup simplifies the development and deployment of NLP models, ensuring flexibility and performance optimization. By leveraging these tools, developers can create sophisticated NLP applications that understand and generate human language with remarkable accuracy. This integration highlights the transformative potential of NLP in various fields, including automated customer service, content creation, and more, contributing to advancements in today's technology landscape.