Fine Tune Llama2

by **SUBRATA MONDAL** | **PHONE** | **GMAIL**

Introduction

Large language models (LLMs) are powerful systems that can generate natural language texts for various tasks, such as dialogue generation. LLM has many applications, such as writing, chatting, coding, etc. LLAMA2 is one such advanced and powerful large language model (LLM) for natural language generation (**NLG**), with up to **70 billion parameters**. However, LLAMA2 also has some challenges and limitations, such as high memory and computation requirements, licensing issues, and domain-specific performance.

In this report, I show you how I fine-tuned **LLAMA2 7B Chat**, a **7 billion** parameter model for dialogue generation, on a custom dataset of English quotes from **goodreads.com**. I have used Google Colab and Kaggle Notebook, which are free platforms that provide GPUs for limited usage. I have also used **QLoRA**, a parameter efficient fine-tuning (**PEFT**) technique that combines quantization and **LoRA**, to reduce the size of the model by **4x** or **75%,** from16-bits to 4-bits for efficient loading in low powered systems. The report has three sections: introduction, methodology, deliverables and conclusion.

Methodology

Data

I have used the e English quotes dataset from Hugging Face and Kaggle, which is a collection of **3,910** quotes from **goodreads.com**, a website that allows users to rate and review books. Each quote has the following attributes:

- **quote:** The content of the quote in English
- **author:** The name of the author who said or wrote the quote
- **tags:** A list of keywords that describe the theme or topic of the quote

I chose to use this dataset because it contains diverse and interesting quotes that can be used to generate dialogues with different styles, topics and most importantly lightweight. I also wanted to test the ability of the model to produce relevant and accurate responses based on the quotes and the authors.

Data Preparation: I have mapped the dataset to the tokenizer of LLAMA2 7B Chat, which converts the quotes to numerical tokens that can be processed by the model. The tokenizer is based on the GPT-2 tokenizer, which uses byte-pair encoding (**BPE**) to split the text into subword units.

Model

I have used LLAMA2 7B Chat, a 7 billion parameter LLM for dialogue generation, from Hugging Face. LLAMA2 is a family of LLMs that are trained on a large corpus of text from various sources. The Chat variant of LLAMA2 is optimized for dialogue generation, using techniques such as instruction tuning, QLoRA, and sharding.

The memory requirements of the LLMs are very high and hence, can't be loaded on most computers or platforms. Therefore, I used **quantization** and **PEFT** (Parameter Efficient Fine-Tuning) techniques, such as **QLoRA** using **BitsAndBytes** library, to reduce the model size and memory usage, while preserving the model quality and accuracy.

Quantization is a technique that reduces the size of the model by using fewer bits to represent each weight. A weight is a numerical value that determines how the model processes the input and produces the output. I have used BitsAndBytes, a library that enables quantization during training for LLMs, and configured it to use *4-bit quantization, double quantization, and bfloat16 data type* that reduced the memory usage of the model by **4x** or **75%**.

LoRA is a PEFT technique that improves the performance of the model by adding low-rank adapters to the model. Low-rank adapters are small matrices that are inserted between the layers of the model and can be tuned to adapt the model to a specific task or domain. I have used **peft**, a library that provides PEFT methods, such as LoRA, and configured it to use a **rank reduction factor of 8**, an **alpha value of 32**, a **dropout rate of 0.05**, a **bias type of none**, and a **task type of causal language modeling**.

Training (Fine Tuning)

I have trained the model on the custom dataset of English quotes using the **Trainer** class from transformers's library that provides high-level APIs for NLP Models and Tasks. Enabled **gradient checkpointing** for the model, which reduces the memory usage by trading off some computation time.

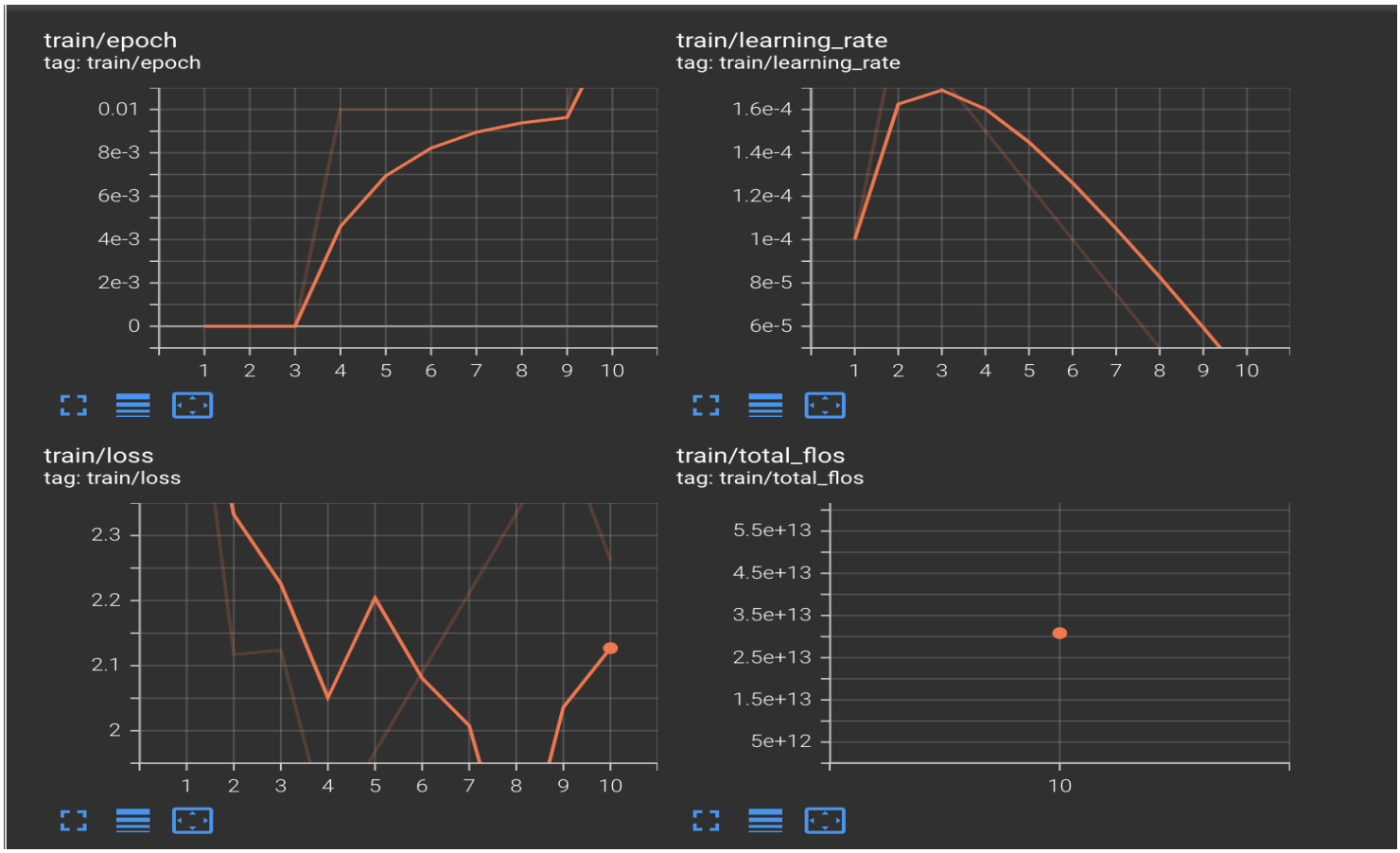
Also used **DataCollatorForLanguageModeling** object from transformers, which prepares the data for **causal language modeling**, which is the task of generating text from given some context.

- **batch_size: 1** - model processes one quote at a time
- **gradient_accumulation_steps: 4** - number of iterations that are performed before updating the model weights
- **warmup_steps: 2** - number of iterations performed before reaching the maximum learning rate
- **max_steps: 10** - total number of iterations performed during the training process
- **learning_rate: 2e-4** - amount of change applied to the model weights in each iteration

- **fp16: True** - indicates to use 16-bit floating point precision for the model weights
- **logging_steps: 1** - number of iterations performed before saving the logs to the output directory
- **output_dir: “outputs”** - path where the model checkpoints and the logs are saved during the training process
- **optim: “paged_adamw_8bit”** - updates the model weights based on the gradient and the learning rate

Evaluation

Captured the performance of the model by launching a TensorBoard instance, which displays the logs in a graphical interface. The logs contain the metrics and the losses of the training process, such as the perplexity, the learning rate, the train loss, and the eval loss. Used the interface to visualize and analyze the performance of the model during the training process.



Deliverables

Stored all the deliverables in Gdrive - two notebooks, one demo video, and a report.

- **Source Code:** Fine tuned the LLM on Colab and Kaggle Notebook, contains two notebook **finetune** and **inference**. You might face a problem with the dataset, in kaggle load_dataset was throwing an error so manually add the data in notebook and then converted it to a DatasetDict.
- Streamlit App Links: Due to high size, not able to host it properly, instead look at my other apps:
    - BERT FINETUNED
    - FASTAI

Conclusion

In this report, I have presented my work on fine-tuning LLAMA2 7B Chat, a 7 billion parameter LLM for dialogue generation, on a custom dataset of English quotes from **goodreads.com**. Used Google Colab and Kaggle Notebook for fine tuning using QLoRA PEFT with 4-bits quantization.

Findings:

- Successfully fine-tuned the LLM on a custom dataset using QLoRA, a Parameter Efficient Fine Tuning technique that reduced the model size by 4x or 75% using 4-bits weight representation.
- Improved the model's performance by adding low-rank adapters to the model using LoRA, which increased the model complexity and expressiveness without increasing the number of parameters, and allowed the model to adapt to the specific domain of dialogue generation based on quotes and authors.

Limitations:

- Faced multiple out-of-memory errors, crashes during the training and loading the full pretrained model to merge with the train adapters and unlink
- Used a small and simple dataset for fine-tuning, which may not capture the diversity and complexity of real-world dialogues.
- Used a fixed system prompt for inference, which may not suit the different styles and preferences of the users.
- Did not perform a rigorous evaluation of the model output, such as using metrics, human ratings, etc.

Suggestions:

- Can use a larger and more diverse dataset for fine-tuning, such as the Persona-Chat dataset, which contains conversations between two speakers with different personas.
- Can use a dynamic system prompt for inference, which can adapt to the user input and the context of the conversation or techniques such as prompt masking or end of sequence token to create dynamic prompts that can be fine-tuned with the model.
- Can perform a more comprehensive evaluation of the model output, such as using metrics, such as BLEU, ROUGE, or METEOR, human ratings, or ablation studies, to measure the quality, accuracy, and relevance of the model output.
- Can explore other techniques and methods for fine-tuning LLAMA2, such as using **mixed-precision quantization**, **quantization-aware training**, or **quantization noise injection**, to further reduce the model size and improve the model performance. We can also use **ONNX**, a framework that enables interoperability and optimization of models across different platforms and devices, to **reduce the latency** and **inference time** of the model.