Fine-tuning Language Models for a Q&A Bot

This report fine-tunes LLaMA-3B, Bloom-3B, and Falcon-1B LLMs on the Alpaca dataset for question-answering. Quantization and QLorA reduced computational demands. Graphic visualizing learning rates and losses. LLaMA-3B performed best on a diverse question set, but all models struggled with complex math/reasoning. GPU utilization, applications, and future research directions are discussed.

Leandro Bello

Introduction

This technical report explores the training process of large language models (LLMs), where demonstrates the fine-tuning of LLaMA-3B, Bloom-3B, and Falcon-1B – for a question-answering (Q&A) bot using the Alpaca dataset.

Background

Language Models and Fine-tuning

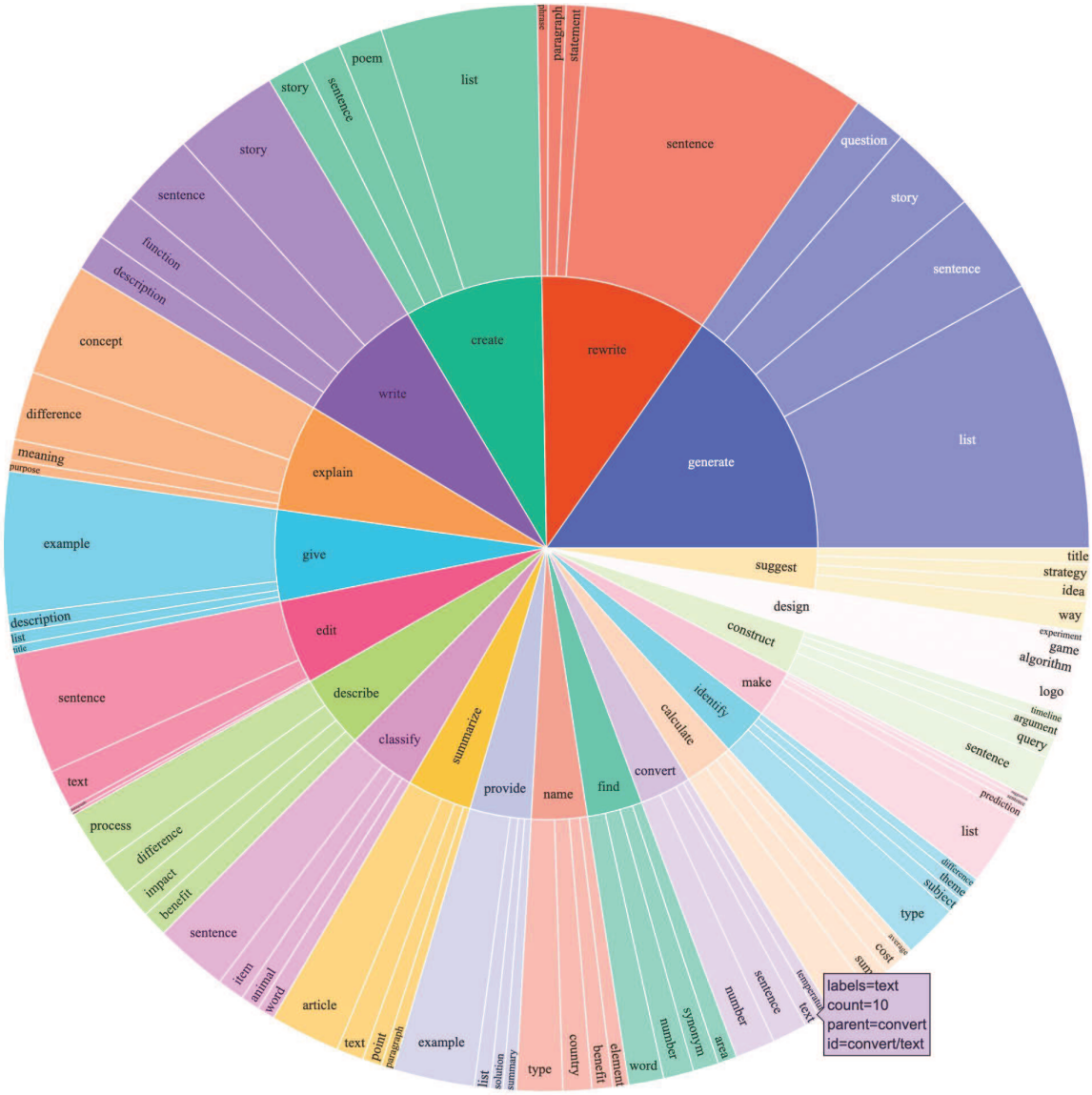
Large language models (LLMs) are transformer-based neural networks trained on massive amounts of text data, allowing them to understand and generate human-like text. These models have revolutionized the field of NLP, demonstrating remarkable capabilities in tasks such as text generation, summarization, question-answering, and more.

While pre-trained LLMs possess broad knowledge and language understanding, fine-tuning these models on task-specific datasets can further enhance their performance and tailor them to specialized applications. Fine-tuning involves updating the model's parameters using a smaller, domain-specific dataset, enabling the model to adapt to the nuances and requirements of the target task.

The Alpaca Dataset

The Alpaca collection consists of 52,000 directives generated by OpenAI's text-davinci-003 model. These instructions cover a range of topics including text summarization, fashion, mathematics, food, and more. They are commonly utilized for refining LLM models.

The creation of the Alpaca dataset prioritized inclusivity, ensuring it encompasses various activities applicable to LLMs. The diagram below illustrates the breadth of topics within the Alpaca instructions. However, relying solely on such a varied dataset for refining LLMs tailored to specific purposes may not yield optimal results.



https://github.com/gururise/AlpacaDataCleaned/blob/main/assets/parse_analysis.png

By fine-tuning LLMs on the Alpaca dataset, researchers and developers can create AI assistants that adhere to ethical principles and prioritize the dissemination of truthful and beneficial information.

Methodology

Model Selection and Preprocessing

For this project, three state-of-the-art LLMs were selected: LLaMA-3B, Bloom-3B, and Falcon-1B. These models were chosen for their good performance and not-so-big size.

- **LLaMA-3B**: the LLaMA-3B model, developed by Meta AI, is a large language model with approximately 3 billion parameters. Known for its strong performance in various NLP tasks, this model serves as a powerful foundation for fine-tuning on the Alpaca dataset.
- **Bloom-3B**: the Bloom-3B model, created by BigScience, is another large language model with approximately 3 billion parameters. This model has demonstrated remarkable capabilities in text generation and understanding, making it a promising candidate for fine-tuning on the Alpaca dataset.
- **Falcon-1B**: the Falcon-1B model, developed by Technology Innovation Institute, is a smaller language model with approximately 1 billion parameters. Despite its more compact size, this model has shown promising results in various NLP tasks, offering an interesting comparison to the larger models in the evaluation.

- **Quantization**: to address the computational demands of fine-tuning these massive LLMs, quantization techniques were employed. Low-Rank Adaptation (LoRA) is an advanced technique for fine-tuning models. Instead of adjusting all the parameters in the extensive weight matrix of the large pre-trained language model, LoRA concentrates on refining two smaller matrices designed to approximate the larger one. These compact matrices form the LoRA adapter. After fine-tuning, this adapter is incorporated into the pre-trained model and utilized during the inference stage. On the other hand, exist another method called quantized LoRA (QLoRA), which is a highly memory-optimized variation of the LoRA method that enhances efficiency by storing the pre-trained model's weights in GPU memory with 4-bit quantization, a reduction from LoRA's 8-bit weight representation.

By combining these techniques, the memory footprint and computational requirements of the fine-tuning process were significantly reduced, allowing for the training of these models on consumer-grade hardware, specifically on a RTX 2070 Max-Q card, with only 8 GB of RAM.

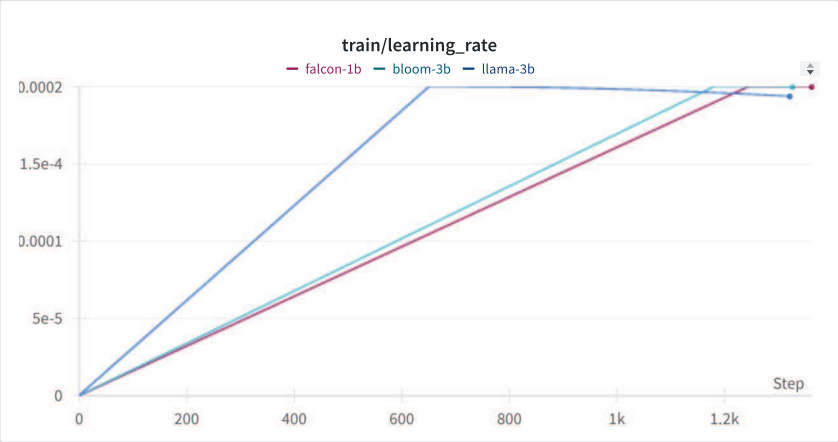
Training setup

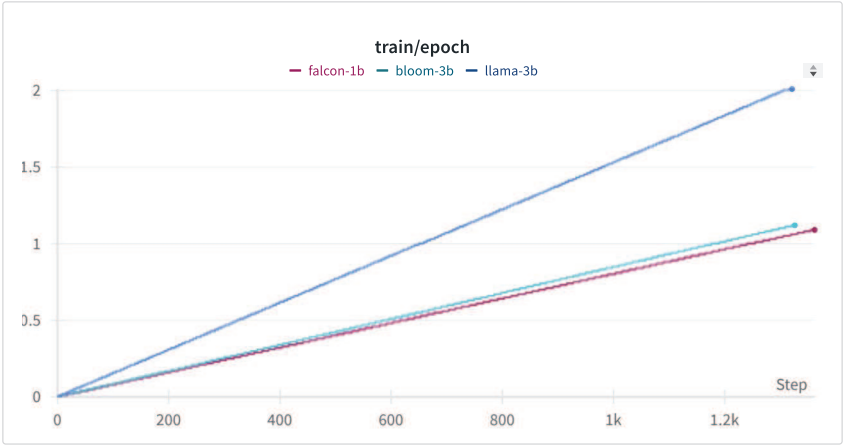
- **Learning Rate and Warmup**: for the fine-tuning process, a warmup and decrease learning rate strategy was employed. This approach involves gradually increasing the learning rate during the initial warmup phase, followed by a gradual decrease in the learning rate as training progresses. This technique has been shown to improve model convergence and performance in many machine learning tasks.
- **Epochs and Batch Sizes**: due to the computational constraints of fine-tuning large language models, the number of epochs and batch sizes varied for each model. The llama-3b model was trained for 2 epochs with a batch size of 4, while the bloom-3b and falcon-1b models were trained for 1 epoch with batch sizes of 2 and 1, respectively. The difference in epochs and batch sizes was primarily influenced by the model sizes and the available computational resources.
- **Hardware Utilization**: given the computational demands of fine-tuning large language models, the GPU utilization was closely monitored throughout the training process. Despite the quantization techniques employed, the fine-tuning process still required significant GPU resources, with all models utilizing nearly 100% of the available GPU memory.

Training

Learning Rate Visualization

In this section, the learning rate visualization for each of the three fine-tuned LLMs is showcased.



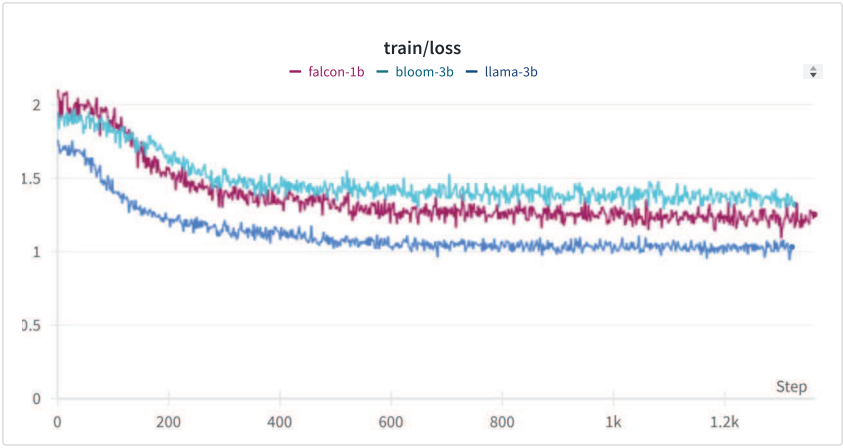


Import panel Add panel

For the learning rate schedule, the LLaMA-3B model employed a linear decrease after the warmup phase. In contrast, the Falcon and Bloom models only exhibited the warmup behavior, as the warmup was set to 10% of the total epochs. Since LLaMA-3B had a larger batch size, it allowed for an additional epoch of training beyond the warmup period, during which the linear decrease in learning rate was applied.

Loss Visualization

In addition training losses were logged, providing a comprehensive view of the models' performance during the fine-tuning process.



Import panel Add panel

- **LLaMA-3B:** the loss visualization for the lLLaMA-3B model reveals a steady decrease in the training loss over the course of the two epochs, indicating that the model is effectively learning from the Alpaca dataset. Notably, the technical report highlights that the LLaMA-3B model achieved the lowest loss among the three evaluated models.

- **Bloom-3B**: the Bloom-3B model's loss visualization shows a similar decreasing trend, albeit over a single epoch. While the final loss value is higher compared to the LLaMA-3B model, the visualization provides valuable insights into the model's learning dynamics and potential for further fine-tuning.
- **Falcon-1B**: the Falcon-1B model's loss visualization illustrates its training progression on the complex Q&A task. Although the model started with a higher initial loss compared to the others, by the end of the single training epoch, its final loss value was in the middle range, lower than the LLaMA-3B model but higher than the Bloom model's loss. This suggests that while the smaller Falcon-1B model faced challenges, it managed to adapt to the task reasonably well, outperforming the larger Bloom-3B model in terms of the final loss achieved.

Evaluation and Inference

Question Set

To comprehensively evaluate the fine-tuned LLMs, a diverse set of 14 questions was curated, covering various domains, including general knowledge, coding problems, reasoning, and mathematical problems. This diverse question set aimed to thoroughly test the models' capabilities and provide insights into their strengths and weaknesses.

- What is the capital city of Australia?
- Can you explain the theory of relativity in simple terms?
- How would you code a function in Python to sum two vectors?
- Who wrote the novel "To Kill a Mockingbird"?
- What is the chemical formula for table salt?
- How would you calculate the area of a circle given its radius?
- Who was the first person to walk on the moon?
- Can you name three common cybersecurity threats?
- What is the meaning of the phrase "a bird in the hand is worth two in the bush"?
- How would you explain the concept of sustainable development?
- Who painted the famous artwork "The Starry Night"?
- Can you describe the process of photosynthesis in plants?
- How would you solve the following equation: 3x + 5 = 17?
- How would you explain the concept of machine learning to someone with no technical background?
- What do you think about the future of the artificial intelligence?

However, in this static report, only 3 of the question-answer pairs are visibly presented. If the reader is interested in examining the complete table consisting of all 14 questions and their corresponding answers from the various models, please refer to the link provided at the beginning of the report. This link will grant access to the comprehensive information, allowing to thoroughly analyze the full set of inferences made across the different models and questions.

```
runs.summary["questions"]
```

	id	question	llama-3b	bloom-3b	falcon-1b
1	1	What is the capital city of Australia?	The capital city of Australia is Canberra, located in the Australian Capital Territory.	The capital city of Australia is Canberra, a federal capital city located in the state of Victoria.	Australia's capital city is Canberra, located on the eastern side of the country along the River Burar. Canberra is the seat of government of the Australian Capital Territory, the country's administrative and legislative capital, and the capital of the Australian state of Canberra. It is also the capital of the Australian territory of New South Wales.
2	2	Can you explain the theory of relativity in simple terms?	Theory of relativity is a scientific theory that describes the behavior of objects and forces in the Universe, based on the principles of Euclidean geometry, special relativity and general relativity. One of the fundamental	... Relativity theory is a set of mathematical principles describing the relationship between two or more inertial frames of reference that are moving relative to one another. The first fundamental principle of relativity is the law of inverse	Theory of relativity is the scientific theory that describes the nature of time and space in terms of moving bodies and the laws that govern those bodies' relationships with each other. It is one of the most influential and well-known theories in physics, and its fundamental

			concepts of relativity is the equivalence principle, which states that the speed of an object's motion (relative to an inertial frame) is not affected by the motion of the object	squares, which states that the distance travelled by objects moving with a constant speed relative to one another is inversely proportional to the square of the speed. This	principle is that matter and energy are related by what is known as the force of gravity. In a nutshell, the theory of relativity states that the laws of
3	3	How would you code a function in Python to sum two vectors?	Here's an example of a function in Python that adds two vectors: ```python def vector_sum(v1, v2): # Calculate the inner product of the two vectors inner_product = np.dot(v1, v2) # Find the sum of the inner product return inner_product ``` In this example, `np.dot` is a function from the `numpy`	# Define a function with 2 arguments: `x` and `y`, representing vectors of length `n`. def sum(x, y): # Sum the two vectors together. return x + y # Call the function with two vectors: `x` and `y`. sum(x, y) # Check the result of the sum: `sum(x, y)`. print(sum(x, y))	import pandas as pddef sum(a, b): return a + bdef sum_of_2(a, b): return a * 2 and b * 2import math%= sum_of_2(a, b)print("sum_of_2()", sum_of_2(a, b))print("sum(2, 2) = ", sum(2, 2))print("sum(2, 2) = ", sum(2, 2))print("sum(2, 2) = ", sum(2, 2))print("sum(2, 2) = ", sum(2, 2))

1 - 3 of 15 Export as CSV Columns... Reset table

Import panel Add panel

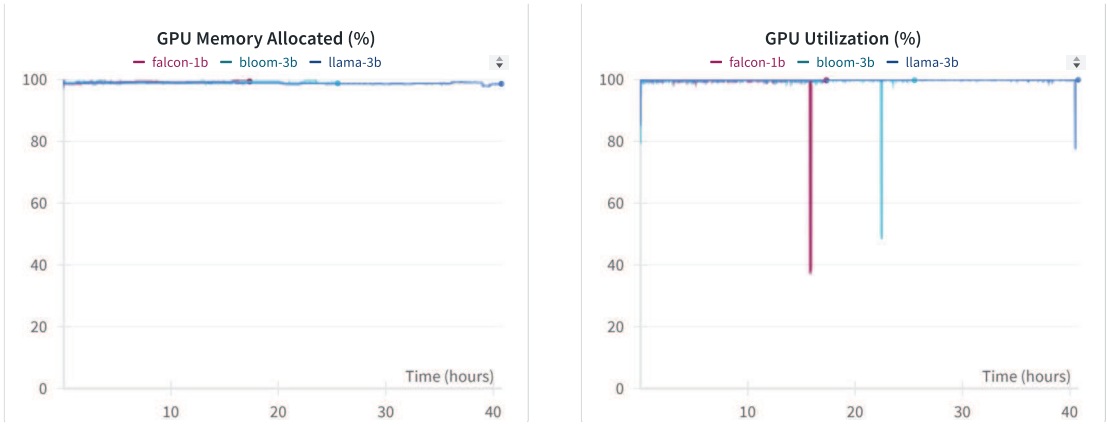
- **LLaMA-3B**: the LLaMA-3B model demonstrated impressive performance across the question set, providing accurate and informative responses to both general knowledge and coding-related queries. However, the technical report notes that the model exhibited some limitations in handling complex mathematical and reasoning problems.
- **Bloom-3B**: the Bloom-3B model's performance was comparable to the LLaMA-3B model, excelling in general knowledge questions. Nonetheless, it encountered similar challenges when faced with intricate mathematical and reasoning problems, and it couldn't answer some of them.
- **Falcon-1B**: despite the smaller size in parameters, the Falcon model did very well on the mathematical problem, being the only model to answer this question correctly. It also offeres great details in most of its answers.

Model Comparison and Discussion

The evaluation of the three models, LLaMA-3B, Falcon-1B, and Bloom, on the Q&A task reveals notable distinctions in their performance. Overall, the LLaMA-3B model exhibited superior results, potentially attributable to the extended training duration spanning two epochs. However, it is essential to acknowledge that the additional epoch facilitated by the larger batch size may have contributed to this advantage. Notably, the Falcon-1B model, despite being the smallest among the three, demonstrated a commendable ability to generate high-quality responses, suggesting its potential for further improvement with increased training epochs. The Bloom model's performance fell between the other two models, indicating a balanced trade-off between model size and training duration. Collectively, these findings underscore the intricate interplay between model architecture, capacity, and the allocated computational resources in achieving optimal performance on natural language processing tasks.

GPU Utilization Analysis

A critical aspect of fine-tuning large language models is the computational resources required, particularly the GPU utilization. The technical report presents an analysis of the GPU utilization during the fine-tuning process, demonstrating that all three models utilized nearly 100% of the available GPU memory, despite the quantization and QLorA techniques employed.



This analysis underscores the significant computational demands of fine-tuning LLMs and highlights the need for efficient techniques and hardware acceleration to enable wider adoption and deployment of these models.

The plot further illustrates the training time invested in each model, highlighting the computational efficiency of the Falcon-1B model. Remarkably, the Falcon-1B model required only approximately one-third of the training time compared to the larger LLaMA-3B model.

Conclusion

Summary of Findings

The technical report summarizes the key findings from the fine-tuning and evaluation of the LLaMA-3B, Bloom-3B, and Falcon-1B models on the Alpaca dataset for a Q&A bot application. It highlights the effectiveness of wandb in logging and visualizing various aspects of the training process, enabling deeper insights into model performance and learning dynamics.

The report emphasizes the superior performance of the LLaMA-3B model, which achieved the lowest loss and demonstrated impressive capabilities across the diverse question set. However, it also acknowledges the limitations of all three models in handling complex mathematical and reasoning problems, presenting opportunities for future research and model improvement.

Future Work

To further enhance the performance of the models on the Q&A task, several avenues for future work can be explored. Firstly, extending the training duration by increasing the number of epochs could potentially yield improved results, as evidenced by the LLaMA 3B model's performance after two epochs. Additionally, experimenting with different batch sizes may uncover optimal configurations that strike a balance between computational efficiency and model convergence. Evaluating alternative optimizers and learning rate schedules could also contribute to more efficient training and better generalization capabilities. Moreover, incorporating perplexity as an evaluation metric would provide a valuable and widely adopted measure of model quality, enabling comprehensive comparisons with other language models and benchmarking efforts.

Sources

- https://medium.com/@metechsolutions/llm-by-examples-using-lora-and-qlora-0284318b7b3d
- https://www.comet.com/site/blog/how-i-leveraged-the-alpaca-dataset-to-fine-tune-the-llama2-model-based-on-contrastive-few-shot-learning/