Instruction Tuning of Lightweight Language Models with QLoRA

Student Jiangtao Xu
Promotor Dr. Qingbiao Li
Co-promotors Chunlin Tian

1 Abstract

Lightweight pretrained models like DistilGPT-2 offer efficient inference and are well-suited for edge deployment. However, task adaptation still requires memory-intensive fine-tuning. In this work, we apply QLoRA—combining 4-bit quantization with Low-Rank Adaptation—to efficiently fine-tune DistilGPT-2 on a subset of the Alpaca dataset. We develop a complete instruction-tuning pipeline, including data preprocessing, QLoRA implementation, inference generation, and model evaluation. Our results show that QLoRA achieves comparable instruction-following performance to full fine-tuning, while updating only 0.48% of model parameters and reducing GPU memory usage by over 80%. On the Alpaca test set and the Self-Instruct dataset, QLoRA improves ROUGE-L by nearly 48% over the base model. The QLoRA-tuned model demonstrates effective instruction-following in scenario-based evaluations—including simple math, daily help, and emotional support—typical of mobile assistants and offline chatbots. These findings demonstrate that QLoRA is a practical solution for instruction tuning in low-resource environments.

2 Methodology

2.1 Model Choice

We adopt **DistilGPT-2**, an 82M-parameter distilled version of GPT-2 (124M), designed as a faster and lighter alternative through knowledge distillation. Despite its significantly smaller size, DistilGPT-2 maintains performance comparable to GPT-2 on both understanding and generation tasks: it achieves an average GLUE score of 77.1 (vs 78.6) and a perplexity of 21.13 on Wikitext-103 (vs 21.30) [1]. These results make it a strong candidate for instruction tuning in low-resource settings. Moreover, it is fully supported by the HuggingFace Transformers library, allowing seamless integration with LoRA and quantization techniques.

2.2 Data Preprocessing

We use a subset of the **Alpaca instruction dataset**, which contains instruction-response pairs generated by GPT-4 across diverse tasks such as question answering, summarization, and reasoning. Its high quality, task diversity, and manageable size make it well-suited for fine-tuning lightweight models on instruction-following tasks without requiring large-scale computational resources.

The original Alpaca prompt format adopts a verbose, chat-style template designed to align with conversational models like ChatGPT. While helpful for large-scale instruction tuning, this format introduces substantial prompt overhead—typically adding over 40 extra tokens per sample. For lightweight models such as DistilGPT-2, this added verbosity can be counterproductive: it consumes valuable context window space and may distract the model from the core instruction. To address this, we adopt a more concise template tailored for small models:

- If `input` is non-empty: `Instruction:\n<instruction>\n\Input:\n<input>\n\n Response:`
- If `input` is empty: `Instruction:\n<instruction>\n\n Response:`

This minimal format preserves the essential instruction-following structure while reducing token overhead, making it more efficient and better suited for training resource-constrained models.

2.3 QLoRA Implementation

To efficiently fine-tune the model, we implement **QLoRA** following Dettmers et al. [2]. The method combines low-bit quantization and low-rank adaptation efficiently and modularly. Its core components include:

- **4-bit quantization**: We use the `bitsandbytes` library to load the base model in 4-bit NF4 quantized format. NF4 combines 4-bit weight storage with double quantization and Gaussian group-wise scaling, yielding minimal memory footprint and high task accuracy.
- **LoRA adapters**: Using the HuggingFace PEFT library, we inject LoRA adapters into the attention and MLP layers of the transformer, with a configuration tailored for small models: rank $r = 8$, scaling factor $\alpha = 16$, and dropout rate `lora_dropout` $= 0.05$.

QLoRA integration: QLoRA first loads the base model in 4-bit quantized format and inserts LoRA adapters into selected linear layers. The output of a LoRA-injected layer is computed as:

$$\boldsymbol{Y}^{\text{BF16}} = \boldsymbol{X}^{\text{BF16}} \cdot \text{DoubleDequant}(\boldsymbol{W}^{\text{NF4}}, \boldsymbol{c}^{\text{FP32}}, \boldsymbol{c}^{\text{k-bit}}) + \boldsymbol{X}^{\text{BF16}} \cdot \boldsymbol{L}_1^{BF16} \cdot \boldsymbol{L}_2^{BF16} \tag{1}$$

Here, $\boldsymbol{W}^{\text{NF4}}$ is the 4-bit quantized weight matrix, and $\boldsymbol{c}^{\text{FP32}}$, $\boldsymbol{c}^{k\text{-bit}}$ are the associated dequantization scaling factors. $\boldsymbol{L}_1^{BF16}$ and $\boldsymbol{L}_2^{BF16}$ are the trainable LoRA matrices, stored in BF16 precision. $\boldsymbol{X}_{\text{BF16}}$ represents the input activation to the linear layer, and $\boldsymbol{Y}_{\text{BF16}}$ is the corresponding output activation.

In this design, the vast majority of model weights remain frozen and are stored in 4-bit quantized format, while only the LoRA adapters—typically accounting for just 1–3% of the total parameters—are updated during training and stored in BF16 precision. This combination of low-bit quantization and low-rank adaptation significantly reduces memory usage and training cost, making QLoRA highly efficient for fine-tuning under resource constraints.

2.4 Training Setup

We fine-tune the model using HuggingFace's `Trainer` API. To accommodate limited resources, we use a small batch size of 1 and apply gradient accumulation over 16 steps. The learning rate is fixed at 1e-4, which was reported to be stable for both full and parameter-efficient fine-tuning in the original QLoRA implementation [2].

For comparison, we also perform full-parameter fine-tuning on the same model and dataset. The results show a clear contrast in resource demands:

- **Full fine-tuning**: all 82M parameters are updated; peak GPU memory usage: 7849 MiB.
- **QLoRA**: only 0.48% of parameters (0.295M) are trainable; peak GPU memory usage: 1394 MiB.

2.5 Inference and Evaluation

After fine-tuning, we generate responses using HuggingFace's `pipeline` with sampling-based decoding. We apply nucleus sampling ($p = 0.95$), top-$k = 50$, and temperature $= 0.7$ to balance diversity and fluency.

To comprehensively evaluate model performance, we adopt three evaluation types using **ROUGE-L** as the metric:

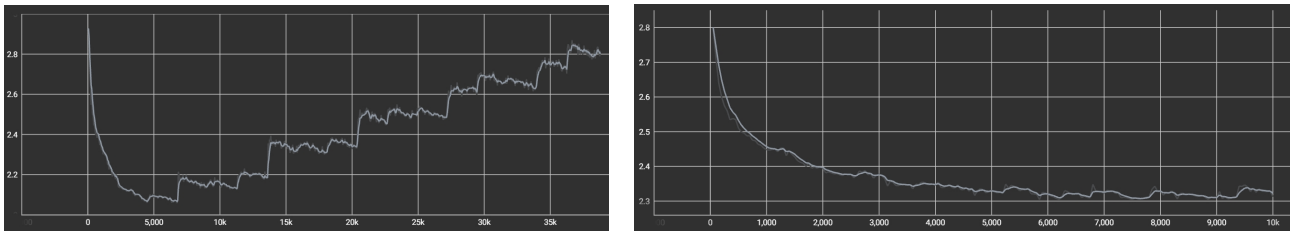
- **Alpaca Test Set**: A reserved portion of the Alpaca dataset, unseen during training, is used to evaluate in-domain instruction-following ability.
- **Self-Instruct Benchmark**: Although structurally similar to Alpaca, this dataset includes a broader set of task types and serves as an out-of-domain generalization benchmark.
- **Scenario-Based Evaluation**: We manually design practical test cases that reflect common deployment needs of lightweight models, such as simple math questions, story generation, basic knowledge reasoning, and emotional support. These cases help assess the model's real-world utility beyond benchmark-style prompts.

3 Results and Discussion

3.1 Model Convergence

To assess the convergence behavior of different fine-tuning strategies, we record the evaluation loss every 50 steps during training. Figure 1a and Figure 1b show the validation loss curves for full-parameter fine-tuning and QLoRA, respectively.

In full-parameter fine-tuning, the evaluation loss drops from 2.85 to 2.07 within approximately 5,000 steps, indicating rapid convergence. For QLoRA, the loss decreases from 2.80 to 2.32 over about 6,000 steps, demonstrating a slightly slower but stable convergence pattern. Overall, both approaches show effective convergence on the Alpaca subset. While full fine-tuning converges faster, QLoRA achieves comparable generalization with significantly fewer trainable parameters and lower memory usage.



(a) Full-parameter fine-tuning (b) QLoRA fine-tuning

Figure 1: Validation loss curves for full fine-tuning (left) and QLoRA (right).

3.2 Quantitative Evaluation

We evaluate three models—no fine-tuning, full fine-tuning, and QLoRA fine-tuning—on two test sets: alpaca test set and Self-Instruct benchmark. ROUGE-L is used to measure similarity between generated and reference responses. Results are shown in Table 1.

Model	ROUGE-L (Alpaca)	ROUGE-L (Self-Instruct)
No Fine-Tuning	10.32	5.85
Full Fine-Tuning	14.58	7.38
QLoRA Fine-Tuning	15.25	8.71

Table 1: ROUGE-L scores of different models on Alpaca and Self-Instruct test sets.

The QLoRA model significantly improves generation quality over the base model. On the Alpaca test set, ROUGE-L increases from 10.32 to 15.25—a 47.78% improvement—matching the performance of full fine-tuning. On Self-Instruct dataset, which presents out-of-distribution instructions, QLoRA also generalizes well, raising ROUGE-L from 5.85 to 8.71—a 48.89% gain—and slightly outperforming full fine-tuning. These results demonstrate that QLoRA substantially enhances instruction-following ability while requiring significantly less GPU memory.

3.3 Scenario-Based Evaluation

Lightweight models are often deployed on edge devices such as mobile assistants and offline chatbots, where tasks like factual Q&A, daily help, and emotional support are common. To simulate such use cases, we evaluate five representative scenarios: simple math, general knowledge, daily assistance, emotional support, and story generation. In each case, we compare responses from the QLoRA-tuned model and the base DistilGPT-2.

All outputs are manually tagged (e.g., *Correct*, *Off-topic*) to assess instruction understanding and response quality. Table 2 summarizes the results. The base model frequently fails to follow instructions, producing irrelevant content. In contrast, the QLoRA model generates more contextually appropriate and relevant responses. However, it still struggles with tasks requiring precise reasoning—such as math questions.

4 Conclusion and Future Work

In this report, we implemented and evaluated QLoRA—a parameter-efficient fine-tuning method that combines 4-bit quantization with low-rank adaptation—on the lightweight DistilGPT-2 model using a small subset of the Alpaca instruction dataset. Compared to full-parameter fine-tuning, our method offers the following advantages and limitations under this task.

Advantages of QLoRA:

- **Parameter efficiency**: Full fine-tuning updates all 82M parameters of DistilGPT-2, while QLoRA updates only 0.48% (0.295M), greatly reducing the optimization footprint.
- **Memory usage**: In our setup, QLoRA requires only 1394 MiB of GPU memory—an 82% reduction compared to 7849 MiB for full fine-tuning—making it feasible even on entry-level GPUs like the GTX 1650 (4GB).
- **Instruction-following performance**: Despite updating far fewer parameters, QLoRA achieves comparable results. On the Alpaca test set, it reaches a ROUGE-L of 15.25 vs. 14.58 for full fine-tuning; on Self-Instruct, 8.71 vs. 7.38.

Scenario	Query	QLoRA-tuned Output	Tags	Base DistilGPT-2 Output	Tags
Simple math	If an apple costs 2 yuan and a banana costs 3 yuan, how much is 2 apples and 1 banana ?	In the case of an apple costing 2 yuan and a banana costing 5 yuan, it would be 6.	Partially correct	We can assume that the price of two bananas would be twice as high. Therefore we are.....	Too long Incorrect
Basic knowledge	Briefly explain what high blood pressure is?	Low blood pressure can cause low blood pressure and potentially fatal heart attacks.	Helpful	If you're reading this, I recommend that the following links go to your.....	Off-topic
Daily help	Should you wash your hands before eating?	You could rinse yourself thoroughly after consuming a meal and eat as much food, but it is important for the person's health and well-being.	Too long	Yes, this is the same as what weve seen so far in our review of "The Man with a Mouth-Inch Wrenching Clothe. So please make sure.....	Off-topic
Emotional support	I just feel really down today and I don't know why.	Because when you're feeling like the world is not getting better, it's time to take a closer look at your own.	relevant	It's been a few years since the first article of my "The Game" blog, but now it has become clear that this is not.....	Off-topic
Story generation	Tell a short story about a rabbit who learns to be brave.	Once upon a time, the Rabbit had become an unstoppable force in her quest for survival. However, she was suddenly overcome by fierce competition from vicious enemies and began fighting back against them all at once.....	Correct	No, no! It is not easy to read and understand the way you interact with animals (i.e., in my experience this process might require some explaining). This technique works by telling people what they know but also letting them get.....	Off-topic

Table 2: Qualitative comparison between QLoRA and base DistilGPT-2 outputs across practical scenarios.

- **Modularity and reusability**: LoRA adapters can be stored and swapped independently of the base model, enabling fast iteration and deployment across multiple tasks or domains without retraining the full model.

Potential Limitations:

- **Layer Support Constraints**: Some layers (e.g., `Conv1D`) are not fully supported by current quantization libraries, requiring manual implementation and limiting compatibility with older architectures.
- **Uncertain Capacity Impact**: While 4-bit quantization yields effective results in our evaluation, it remains unclear whether it compromises the model's representational capacity—especially in small models—calling for broader validation across tasks.
- **Generalization Boundaries**: While QLoRA performs well on Alpaca and Self-Instruct, its effectiveness on more diverse and complex tasks—such as long-context generation, structured outputs, and multi-turn dialogues—remains underexplored.

Future Work:

- **Robustness Across Tasks**: While QLoRA achieves strong results on in-domain tasks like Alpaca, performance tends to decline on out-of-domain datasets such as Self-Instruct. Future work could explore multi-task fine-tuning on diverse instruction datasets to enhance generalization and task robustness.
- **Lower-bit Quantization**: QLoRA currently uses 4-bit quantization. Investigating ultra-low-bit regimes may further reduce memory usage. Clarifying the trade-off between compression and performance is essential for reliable deployment in constrained environments.
- **Scalability to Larger Models**: Applying QLoRA to mid-sized models such as LLaMA-7B or Mistral could help assess whether its memory and performance benefits scale with model size.

References

[1] Tianda Li, et al. *A Short Study on Compressing Decoder-Based Language Models*. arXiv preprint arXiv:2110.08460, 2021.

[2] Tim Dettmers, Artidoro Pagnoni, Lukas Zettlemoyer, and Mike Lewis. *QLoRA: Efficient Finetuning of Quantized LLMs*. arXiv preprint arXiv:2305.14314, 2023.