Introduction

Large language models like GPT-2, RoBERTa, and DeBERTa excel at natural language understanding (NLU) tasks, but adapting them to different downstream tasks via full fine-tuning is costly both in time and memory, often making it infeasible at scale. Our project reproduces and evaluates the method introduced in "*LoRA: Low-Rank Adaptation of Large Language Models*" (Hu et al., ICLR 2022), which proposes freezing pre-trained weights and injecting low-rank matrices which are trained instead of updating all pre-trained weights. This technique drastically reduces trainable parameters, adds no inference latency, and remains compatible with standard Transformer architectures without sacrificing performance. We test LoRA's practicality and effectiveness by replicating key experiments on a subset of GLUE tasks and comparing LoRA's results with those of traditional fine-tuning.

Chosen Result

The paper applies LoRA to the self-attention modules of RoBERTa, DeBERTa, GPT-2, and GPT-3. We choose to restrict our implementation of LoRA to RoBERTa-base and DeBERTa-v2-xlarge. DeBERTa-v2-xlarge is larger than RoBERTa-base, so our experiments allow us to assess the effectiveness of LoRA across different sized models as well as across different downstream tasks.

Since the experiments in the paper were all conducted for 30 to 80 epochs, we were unable to directly replicate all of them. We replicate the exact experiment from the paper on one task with RoBERTa-base, and also run some of our own experiments to compare LoRA with varying rank size and full fine-tuning.

These results highlight LoRA's ability to match or exceed full fine-tuning performance while using orders of magnitude fewer trainable parameters. Demonstrating this is essential to validate the core claim of the paper—that low-rank updates are a parameter-efficient yet high-performance alternative to conventional fine-tuning.

Methodology

We fine-tune RoBERTa-base on the sentence pair classification tasks MRPC (paraphrase identification) and RTE (textual entailment identification), and DeBERTa-v2-xlarge on SST-2 (binary sentiment classification) with and without LoRA.

As aforementioned, we were unable to directly replicate the full experimental setup of the original paper due to compute limitations. For our custom experiments, we made the following modifications:

- Limited training to 10 epochs for most runs
- Used heuristically selected hyperparameters (for training efficiency)
- Trained LoRA with ranks 4, 8, and 16 to assess the significance of rank on LoRA's performance
- Fine-tuned full model with the same hyperparameters to provide baseline comparison with LoRA experimental runs

Model performance on all tasks was measured by accuracy, consistent with the paper's reported metrics. All models are modified and trained using PyTorch and Hugging Face Transformers. To apply LoRA, we first freeze all parameters (set requires_grad = False) in a given model. Then, we create a LoRA layer that initializes two trainable, low-rank matrices A and B, and stores the pre-trained weight matrix W_0. The LoRA layer is injected into the model, replacing the key, query projection layers in the self-attention modules. The forward pass for the LoRA model is defined below (Equation 3 in the LoRA paper).

$$h = W_0 x + \Delta W x = W_0 x + BAx$$

Results & Analysis

We now present the results of our experiments of LoRA with the RoBERTa and DeBERTa models.

Figures 1 and 2 show the accuracy on the MRPC and RTE datasets for different LoRA implementations and full finetuning. For the MRPC dataset (Figure 1), the performance of the r16 model approaches that of the full finetuning after approximately 5 epochs. However, lower ranks (r4, r8) consistently underperform. Similar patterns emerge for the RTE dataset (Figure 2), though with a more pronounced performance gap between all LoRA variants and full fine-tuning.

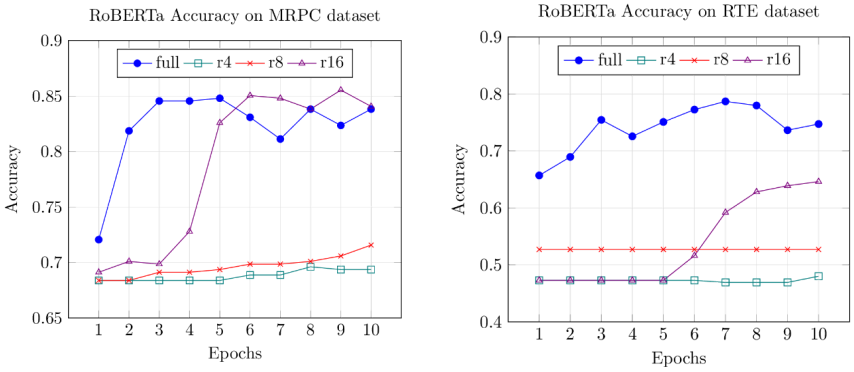

Figure 1 (left) and Figure 2 (right)

Figure 3 shows the result of our direct reproduction experiment with RoBERTa-base. In Table 2, the paper cites a peak accuracy of 95.1% with hyperparameters given in Table 10 of Appendix D. In our reproduction, we reached a best SST-2 accuracy of 91.8% with rank 8—falling slightly short of the original but still supporting the core insight that LoRA performs competitively with full fine-tuning while using far fewer parameters.

The DeBERTa-v2-xlarge model (Figure 4) demonstrated higher absolute performance but maintained the pattern of full fine-tuning outperforming LoRA variants.

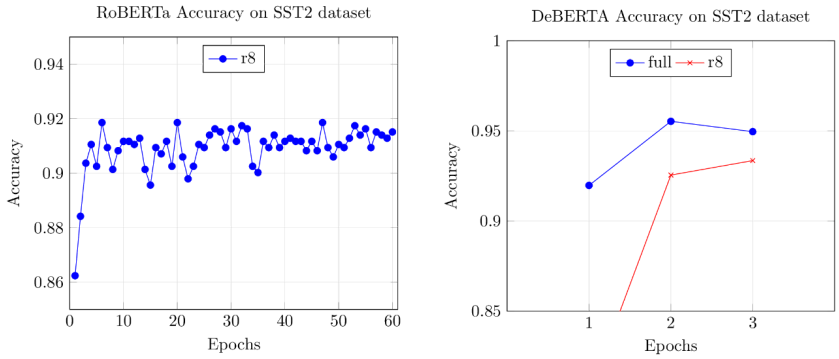

Figure 3 (left) and Figure 4 (right)

Overall, our reimplementation yielded results that partially align with the original paper's findings but we noticed a larger difference between LoRA and full finetuning. This could be because we used different initializations for hyperparameters or learning rates compared to the original authors. Additionally, due to compute and time constraints, we ran smaller-scale experiments than those in the paper. These limitations may have led to the differences in our results. From our results, we can conclude that certain LoRA configuration parameters may match the performance of full fine tuning at the lower cost of memory and training time.

Reflections

Reimplementing LoRA revealed interesting insights about the tradeoff between parameter efficiency and model performance. We learned that LoRA rank makes a huge impact on this tradeoff and that different ranks can cause performance to vary. We also observed varying levels of success across different NLP datasets and tasks, suggesting that parameter-efficient techniques may need to be adjusted between models and tasks. We also ran into some problems with reproducibility potentially due to missing details on the original implementation.

In the future, we would be interested in testing more hyperparameters for these datasets to improve model performance, as well as try LoRA on larger models such as GPT-2. In addition, we would be intrigued to explore what tasks are most suitable for LoRA. Despite not fully matching the original results, we learned a lot about the practical considerations for parameter-efficient finetuning.

References

He, Pengcheng, et al. "Deberta: Decoding-enhanced bert with disentangled attention." *arXiv preprint arXiv:2006.03654* (2020).

Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." *ICLR* 1.2 (2022): 3.

Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." *arXiv preprint arXiv:1907.11692* (2019).

Wang, Alex, et al. "GLUE: A multi-task benchmark and analysis platform for natural language understanding." *arXiv preprint arXiv:1804.07461* (2018).