

Parameter-Efficient Fine-Tuning of GPT-2 using Low-Rank Adaptation (LoRA)

Stanford CS224N Default Project

Christopher Alexander Perez
Department of Computer Science
Stanford University
chrisp26@stanford.edu

Abstract

Large Language Models have shown remarkable capabilities across a variety of NLP tasks, but adapting them to downstream tasks via full fine-tuning is computationally bloated. In this project, I implement a custom GPT-2 model from scratch and evaluate the efficacy of Low-Rank Adaptation (LoRA) as a parameter-efficient alternative. By freezing the pre-trained weights and injecting trainable rank decomposition matrices into the Transformer's attention heads, I aim to show that this efficient adaptation subspace can match the performance of full fine-tuning. Evaluated across Paraphrase Detection and Sonnet Generation, this targeted approach matches full fine-tuning accuracy while providing a vastly superior return on computational investment.

1 Key Information to include

- Mentor: Shicheng Liu
- External Collaborators: None
- Sharing project: No
- Late Days: 0

2 Introduction

The current paradigm of Natural Language Processing relies heavily on pre-training large models on vast corpora and then fine-tuning them on task-oriented datasets. As models like GPT-2 scale up, however, full fine-tuning becomes a huge bottleneck, requiring large amounts of GPU memory to store optimizer states and gradients for every parameter.

To address this, parameter-efficient fine-tuning methods have been developed. This project explores Low-Rank Adaptation (LoRA) [1], which hypothesizes that the changes in weights during task adaptation possess a low "intrinsic rank." Rather than mutating the foundational structure of a model, LoRA isolates the necessary task-specific knowledge into a lower-dimensional space. By implementing a custom GPT-2 architecture and integrating LoRA into the query and value projection layers, this project evaluates whether LoRA can match the performance of traditional full-model fine-tuning.

3 Related Work

The foundational architecture for this project is the Transformer decoder, originally introduced by Vaswani et al. [2] and adapted into the generative pre-trained transformer (GPT) family by Radford et al. [3]. For parameter-efficient fine-tuning, this work is grounded in Hu et al. [1], who demonstrated that injecting trainable low-rank matrices into standard linear layers allows models to match the

performance of full fine-tuning on tasks like RoBERTa and GPT-3, while significantly reducing hardware demands.

4 Approach

Base Architecture: The foundation of the project is a custom implementation of GPT-2, featuring an embedding layer, positional encodings, and stacked causal self-attention layers.

Task 1: Paraphrase Detection. I adapted the standard autoregressive model for classification by passing the input sequence through the GPT-2 backbone and extracting the hidden state of the final valid token in the padded sequence. This hidden state is passed through a linear projection layer, mapping the 2-way classification output back to the GPT-2 vocabulary space using the token IDs for "no" (3919) and "yes" (8505).

Task 2: Sonnet Generation. For autoregressive generation, the hidden states of the transformer are projected back to the vocabulary size. To conserve parameters, I utilized weight-tying by taking the dot product of the hidden states and the transpose of the input word embeddings (`word_embedding.weight.T`).

Low-Rank Adaptation (LoRA): To implement PEFT, I created a custom `LoRALinear` PyTorch module. The base pre-trained weights from HuggingFace were loaded and explicitly frozen (`requires_grad = False`). Two parallel low-rank matrices, A and B , were initialized (with A using Kaiming uniform initialization and B initialized to zero) and scaled by α/r . These modules were dynamically wrapped around the query and value layers within the `CausalSelfAttention` class.

5 Experiments

5.1 Data

The model was evaluated on two primary datasets:

- **Quora Dataset:** Used for paraphrase detection, this dataset contains pairs of questions. The task is to determine if the two questions carry the same semantic meaning.
- **Sonnets Dataset:** A raw text corpus of Shakespearean-style sonnets used to train the autoregressive generation capabilities.

5.2 Evaluation method

Paraphrase detection is evaluated using standard classification **Accuracy** on the dev and test splits. Sonnet generation is evaluated using structural **CHRF** scores to determine how closely the output mimics the linguistic style of the training corpus.

5.3 Experimental details

Training was conducted on Modal cloud infrastructure using NVIDIA A10G GPUs. The model was trained for 10 epochs using the AdamW optimizer with a learning rate of $1e-5$. For the LoRA hyperparameters, I utilized a rank of $r = 8$ and an alpha scaling factor of $\alpha = 32.0$.

Due to the strict memory constraints of the 24GB VRAM on the A10G, batch sizes required careful tuning. Full fine-tuning resulted in Out-Of-Memory (OOM) errors at standard batch sizes, requiring a reduction to a batch size of 16 for Paraphrase Detection and 32 for Sonnet generation.

5.4 Results

The model was evaluated across three distinct downstream tasks to establish baselines and verify the parameter-efficient architecture.

For the baseline sentiment tasks, full-model fine-tuning yielded strong results, achieving **51.1%** Dev and **55.2%** Test accuracy on the SST dataset, alongside **97.1%** Dev accuracy on the CFIMDB dataset.

For the primary Paraphrase Detection task, the baseline (full fine-tuning) model achieved a Dev Accuracy of **89.8%**. When transitioning to the parameter-efficient LoRA implementation, the model achieved a Dev accuracy of **89.5%** and a Test accuracy of **85.7%**. Notably, these LoRA scores successfully outperformed the teaching staff’s full fine-tuning baselines (88.2% Dev / 85.6% Test) despite freezing the vast majority of the 124 million GPT-2 parameters.

For the Sonnet Generation task, the LoRA model achieved a Test CHRF score of **38.212**. While autoregressive generation is highly sensitive to parameter freezing, this score indicates the model successfully learned the structural distribution of the Shakespearean corpus.

6 Analysis

The most significant finding of this project was the difference in hardware economy between full fine-tuning and LoRA. During initial baseline testing, attempting to update all weights within the `CausalSelfAttention` blocks caused immediate CUDA OOM crashes. LoRA completely mitigated this by limiting gradient updates to the extremely small A and B matrices, allowing for faster iteration and more stable training.

Qualitatively, the sonnet generation produced interesting results. While the model struggled with strict rhyming schemes, it was able to somewhat capture the tone of the era, generating lines such as *"Hypnologues of atlas pink with glitter lies"* and *"When animals wonder old maidens became little flashing wild."* This demonstrates that the low-rank subspace was able to effectively capture the stylistic nuances of the dataset.

7 Conclusion

This project successfully implemented a custom GPT-2 architecture and extended it with Low-Rank Adaptation. These results confirm that NLP models can be adapted to highly specific downstream tasks ranging from semantic classification to stylistic text generation, without the immense computational requirements of updating every parameter. Future work could explore tuning the rank (r) and alpha (α) hyperparameters across different attention matrices to optimize even further.

References

- [1] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.
- [2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.