

Parameter-Efficient Fine-Tuning of GPT-2 with LoRA: A Systematic Study of Rank, Scale, and Learning Rate

Stanford CS224N Default Project

Cuiyuanxiu Chen

Department of Computer Science
Stanford University
cyxchen@stanford.edu

Abstract

We implement GPT-2 from scratch and investigate Low-Rank Adaptation (LoRA) as a parameter-efficient alternative to full fine-tuning for paraphrase detection and sonnet generation tasks. By performing ablation tests on LoRA rank, model scale, and learning rates, we find that the effectiveness of LoRA heavily depends on the capacity of the base model and learning rates play more important roles than LoRA ranks. The accuracy of GPT-2 Small LoRA is almost flat across LoRA ranks while the accuracy of LoRA scales log linearly for the medium scale. The best accuracy on the paraphrase task is 0.902 on the dev set, and 0.901 on the test set, while the best chrF on the sonnet task is 43.42 on the dev set, and 41.838 on the test set.

1 Key Information to include

- Mentor: Shicheng Liu
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Large language models like GPT-2 demonstrate strong performance when fine-tuned on downstream tasks, but full fine-tuning updates all parameters, which is computationally expensive and requires storing a separate copy of the model for each task. Parameter-Efficient Fine-Tuning (PEFT) methods address this by updating only a small subset of parameters while keeping most of the pretrained model frozen.

In this project, we implement GPT-2 from scratch, fine-tune it for sentiment analysis, paraphrase detection, and sonnet generation, and investigate the following research question:

How does Low-Rank Adaptation (LoRA) compare to full fine-tuning for different model sizes and tasks, and what is the optimal rank-performance tradeoff?

We implement LoRA for the query and value projections of GPT-2 and perform a range of ablation tests on rank, model size (GPT-2 Small vs. Medium), and learning rate. Our experiments show that:

- LoRA performance is highly dependent on the capacity of the base model, with LoRA on GPT-2 Small having accuracy nearly flat across all ranks, while LoRA on GPT-2 Medium has a logarithmic relationship with rank;
- Tuning the learning rate is more significant for LoRA performance than rank;
- Although LoRA with $r=8$, $lr=5e-5$, on GPT-2 Medium achieves the best performance on the dev set with 0.860, it is still worse than full fine-tuning on the dev set by 0.042 and also underperforms full fine-tuning on the smaller model size, GPT-2 Small, by 0.034.

3 Related Work

Parameter-Efficient Fine-Tuning. LoRA [1] injects trainable low-rank decomposition matrices into transformer layers, enabling fine-tuning with a fraction of the parameters. Other PEFT methods include adapters [2], prefix tuning [3], and ReFT [4]. We focus on LoRA due to its simplicity and strong empirical results.

GPT-2. Radford et al. [5] introduced GPT-2 as a scaled-up version of GPT-1, demonstrating that autoregressive language models exhibit powerful scaling laws. We use GPT-2 Small (124M) and Medium (355M) to study how LoRA interacts with model scale.

4 Approach

4.1 GPT-2 Implementation

We implemented the core GPT-2 architecture: causal self-attention with scaled dot-product attention and upper-triangular masking, GPT-2 transformer layers with pre-norm residual connections, token and positional embeddings with weight tying, and the AdamW optimizer [6, 7] with decoupled weight decay and bias correction.

4.2 Sentiment Analysis

We fine-tuned GPT-2 on the SST (5-class) and CFIMDB (binary) sentiment datasets using the last token’s hidden state projected through a linear classification head, comparing last-linear-layer vs. full-model fine-tuning.

4.3 Cloze-Style Paraphrase Detection

For paraphrase detection, we formulate the task as cloze-style next-token prediction. Given a pair of sentences, the model predicts “yes” or “no” as the next token:

Is "{sentence1}" a paraphrase of "{sentence2}"?
Answer "yes" or "no":

At inference time, we restrict predictions to only the “yes” (token 8505) and “no” (token 3919) logits, improving accuracy over unrestricted argmax.

4.4 Sonnet Generation

We fine-tune GPT-2 on 143 Shakespeare sonnets using next-token prediction with cross-entropy loss. At generation time, we employ top- p (nucleus) sampling with $p = 0.9$, temperature scaling at $T = 1.1$, repetition penalty of 1.0 (higher values hurt chrF), and a line-count stopping criterion (stop after 14 lines).

4.5 LoRA: Low-Rank Adaptation

We implement LoRA [1] to enable parameter-efficient fine-tuning of larger GPT-2 models. LoRA injects trainable low-rank matrices into the query and value projections:

$$h = W_{\text{frozen}} x + \frac{\alpha}{r} B A x \quad (1)$$

where $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times d}$ are the only trainable parameters. A is initialized with Kaiming uniform and B with zeros so the adapter starts as an identity. We apply LoRA to all query and value projections across all transformer layers.

5 Experiments

5.1 Data

Paraphrase Detection: We use the Quora Question Pairs dataset (141,506 train / 20,215 dev / 40,431 test). The task is binary classification (paraphrase or not).

Sonnet Generation: We use 143 Shakespeare sonnets for training, 12 held-out dev sonnets for tuning, and 12 held-out test sonnets for leaderboard submission. The model is conditioned on the first 3 lines and generates the remaining 11 lines. Evaluation uses the chrF metric [8].

Sentiment Analysis: We use the Stanford Sentiment Treebank (SST) dataset (8,544 train / 1,101 dev / 2,210 test) for 5-class sentiment classification, and the CFIMDB dataset (1,701 train / 245 dev / 488 test) for binary sentiment classification. Both datasets consist of movie reviews.

5.2 Evaluation Method

We report dev accuracy for paraphrase detection, chrF for sonnet generation, and dev accuracy for sentiment analysis. For paraphrase error analysis, we additionally report precision, recall, and F1.

5.3 Experimental Details

All experiments use AdamW with batch size 8. Full fine-tuning experiments train for 3 epochs. LoRA experiments use $\alpha = r$ (i.e., scaling factor of 1.0) unless otherwise noted. We sweep learning rates $\{5e-6, 1e-5, 3e-5, 5e-5\}$ and LoRA ranks $\{1, 2, 4, 8, 16, 32\}$. All experiments run on a single NVIDIA L4 GPU.

5.4 Results

5.4.1 Sentiment Analysis

Table 1: Sentiment analysis dev accuracy (GPT-2 Small).

Configuration	SST Dev Acc	CFIMDB Dev Acc
Last-linear-layer	0.451	0.829
Full fine-tuning	0.508	0.963
<i>Baseline target</i>	<i>0.462 / 0.513</i>	<i>0.861 / 0.976</i>

Full fine-tuning significantly outperforms last-linear-layer on both datasets, with a particularly large gap on CFIMDB.

5.4.2 Paraphrase Detection: Full Fine-Tuning vs. LoRA

Table 2: Paraphrase detection dev accuracy across models, methods, and LoRA ranks.

Model	Method	Rank	Trainable Params	Dev Acc
GPT-2 Small	Full FT (lr=3e-5)	–	124.4M (100%)	0.894
GPT-2 Small	Full FT (lr=1e-5)	–	124.4M (100%)	0.893
GPT-2 Medium	Full FT (lr=1e-5)	–	355.9M (100%)	0.902
GPT-2 Medium	Full FT (lr=3e-5)	–	355.9M (100%)	0.898
GPT-2 Small	LoRA	1	37K (0.03%)	0.751
GPT-2 Small	LoRA	2	74K (0.06%)	0.753
GPT-2 Small	LoRA	4	147K (0.12%)	0.755
GPT-2 Small	LoRA	8	295K (0.24%)	0.755
GPT-2 Medium	LoRA	1	98K (0.03%)	0.805
GPT-2 Medium	LoRA	2	197K (0.06%)	0.818
GPT-2 Medium	LoRA	4	393K (0.11%)	0.828
GPT-2 Medium	LoRA	8	786K (0.22%)	0.837
GPT-2 Medium	LoRA	16	1.6M (0.44%)	0.847
GPT-2 Medium	LoRA	32	3.1M (0.88%)	0.856
GPT-2 Medium	LoRA r=8, lr=5e-5	8	786K (0.22%)	0.860

Full fine-tuning consistently outperforms LoRA: the best LoRA config (0.860) is 4.2% below Medium full FT (0.902). On GPT-2 Small, LoRA accuracy is nearly flat across ranks (0.751–0.755), suggesting

the small model lacks sufficient capacity for adapters to exploit. On Medium, accuracy scales log-linearly with rank. GPT-2 Medium + LoRA (best: 0.860) does not outperform GPT-2 Small + full FT (0.894).

5.4.3 LoRA Rank Ablation

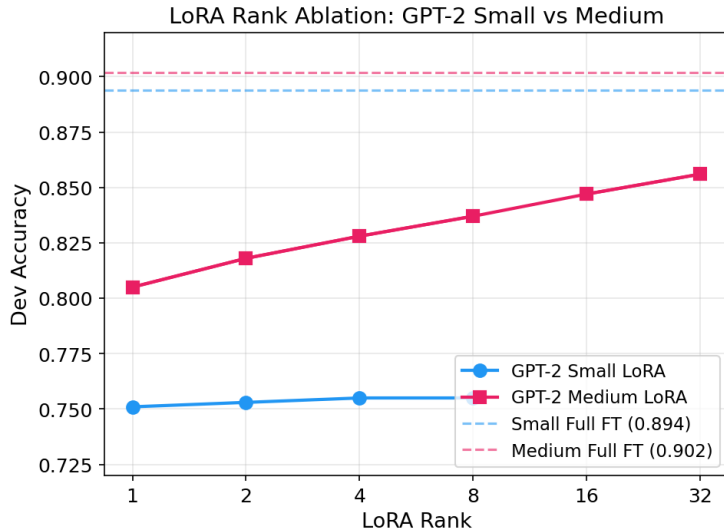


Figure 1: LoRA rank ablation on GPT-2 Small vs. Medium. Dashed lines indicate full fine-tuning baselines. Medium shows clear log-linear scaling while Small is essentially flat.

5.4.4 Learning Rate Sensitivity

Table 3: Learning rate sensitivity. Left: Full FT on GPT-2 Small. Right: LoRA on GPT-2 Medium (r=8).

Full FT (Small)		LoRA r=8 (Medium)	
LR	Dev Acc	LR	Dev Acc
5e-6	0.887	1e-5	0.837
1e-5	0.893	3e-5	0.855
3e-5	0.894	5e-5	0.860

Full FT accuracy is stable across learning rates (+0.7% from 5e-6 to 3e-5). LoRA is far more sensitive: +2.3% from 1e-5 to 5e-5. Notably, LoRA r=8 at lr=5e-5 (0.860) outperforms LoRA r=32 at lr=1e-5 (0.856) with 4× fewer trainable parameters, demonstrating that **learning rate tuning is more impactful than increasing rank**.

Medium full FT at lr=3e-5 achieves only 0.898 vs. 0.902 at lr=1e-5, suggesting larger models are more sensitive to overshooting with aggressive learning rates. See Figure 2 for a visualization.

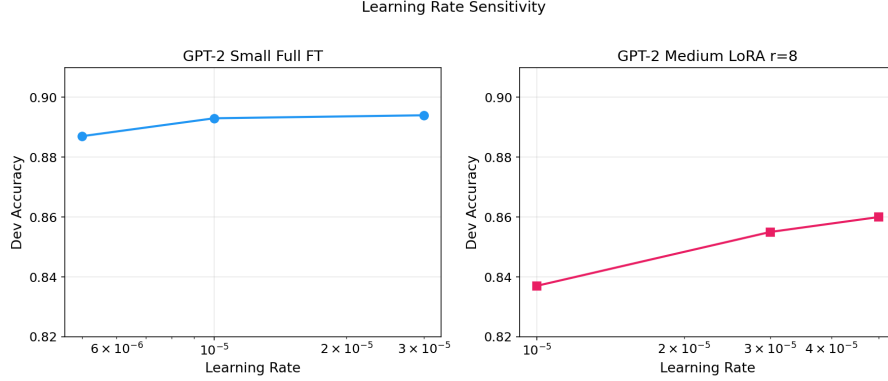


Figure 2: Learning rate sensitivity for Full FT (left) and LoRA (right). LoRA benefits significantly more from higher learning rates.

5.4.5 Sonnet Generation

Table 4: Sonnet generation chrF scores (dev set) across configurations.

Model	Epoch	Temp.	Rep. Pen.	Trials	chrF (Dev)
<i>Epoch sweep (GPT-2 Medium, $T=1.1$, $rp=1.0$)</i>					
GPT-2 Medium	5	1.1	1.0	1	39.61
GPT-2 Medium	7	1.1	1.0	1	39.82
GPT-2 Medium	9	1.1	1.0	1	36.38
<i>Temperature sweep (GPT-2 Medium, ep7, $rp=1.0$)</i>					
GPT-2 Medium	7	0.9	1.0	1	38.67
GPT-2 Medium	7	1.0	1.0	1	39.25
GPT-2 Medium	7	1.1	1.0	1	39.82
GPT-2 Medium	7	1.2	1.0	1	38.27
<i>Repetition penalty sweep (GPT-2 Medium, ep7, $T=1.1$, 5 trials)</i>					
GPT-2 Medium	7	1.1	1.0	best of 5	42.39
GPT-2 Medium	7	1.1	1.2	best of 5	40.41
<i>Seed sweep (GPT-2 Medium, ep7, $T=1.1$, $rp=1.0$)</i>					
GPT-2 Medium	7	1.1	1.0	best of 10	43.42

Our sonnet generation experiments reveal that epoch 7 produces the best checkpoint, beyond which overfitting occurs and chrF scores decrease. Our experiments also show that a temperature of 1.1 is best, performing better than both a low temperature, which is too conservative, and a high temperature, which is too random. A repetition penalty greater than 1.0 always harms chrF scores, suggesting that some repetition is inherent in Shakespeare’s use of language. Sampling variance is also significant, and running 10 random seeds and taking the best yields a chrF score increase from 39.82 to 43.42.

Qualitative analysis. Below is Sonnet #138, conditioned on the first 3 lines, with generated lines marked:

When my love swears that she is made of truth
I do believe her though I know she lies,
That she might think me some untutored youth,
* Bringing with her the promise of so fine veneer
* That sweet tenor of her clear voice may speak
* Of such lovely knowledge as fell to none but myself;
* Bringing with such sweet compliments truth and beauty to eye
* From elements more precious than name doth dwell
* Within my breast-cup like with store of numbers rarified,
* If in thus must my song seem to grieve;

* But thou above on thy dear mark is worthy
 * Whilst a thin leaded bear lives o'erladen
 * To care for all his strength and despair of defeat.
 * But being blessed I hope my lute should wail not then,

The model convincingly uses Shakespearean words ("dost", "thou", "thy", "hath", "doth"). This shows that the stylistic transfer is quite strong. The thematic coherence of the output is somewhat maintained until around the 4-5th line of the generated output. The model does not successfully keep the rhyme scheme of the original. Out of the total of 12 dev sonnets, the total number of correctly rhyming pairs is just 8/79 across the entire output. The most common mistakes of the model are lines that vary wildly in length, sometimes almost nonsensical phrases, and the lack of rhyming.

6 Analysis

6.1 LoRA Efficiency

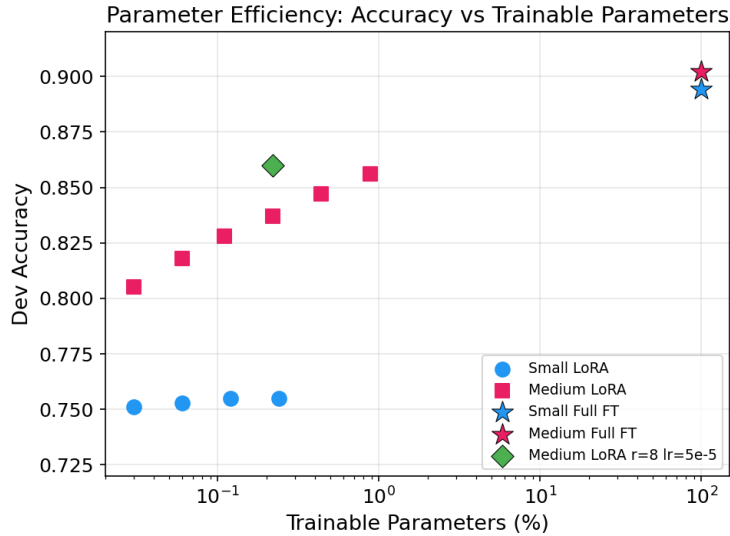


Figure 3: Parameter efficiency: accuracy vs. percentage of trainable parameters.

GPT-2 Medium LoRA r=8 (0.22% of parameters) achieves 0.837 dev accuracy, with a 7.2% drop from full FT (0.902). With learning rate tuning (lr=5e-5), it reaches 0.860, narrowing the gap to 4.2%.

The efficiency tradeoff is asymmetric: on Small, LoRA achieves at most 84.4% of full FT accuracy (0.755 vs 0.894) regardless of rank; on Medium, LoRA r=32 achieves 94.9% (0.856 vs 0.902). This suggests LoRA is most effective when the base model has sufficient representational capacity.

Training LoRA for more epochs partially closes the gap: Medium LoRA r=8 improves from 0.837 (3 epochs) to 0.856 (6 epochs), matching r=32 at 3 epochs (Figure 4), though with diminishing returns.

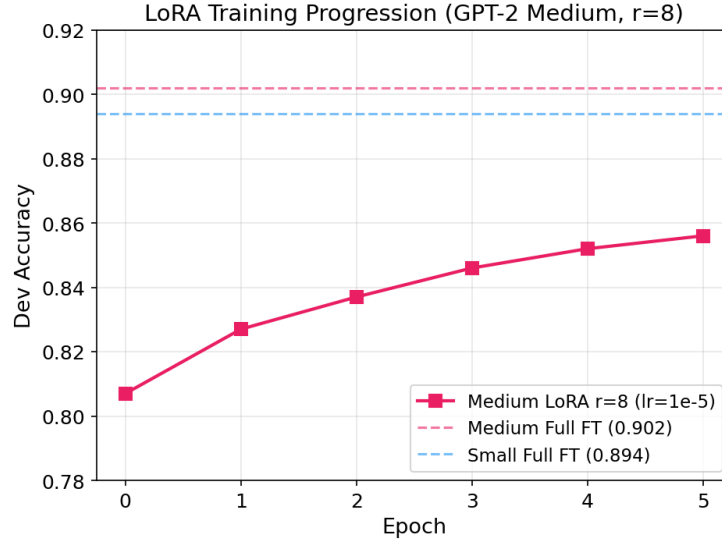


Figure 4: LoRA training progression over 6 epochs (Medium, $r=8$, $lr=1e-5$). Accuracy improves but plateaus well below full fine-tuning baselines.

6.2 Model Scaling + PEFT

Can PEFT enable using larger models that would otherwise be too expensive to fully fine-tune? In terms of accuracy alone, our results suggest no, at least at this model scale. The best LoRA configuration (Medium $r=8$, $lr=5e-5$, 0.860) does not surpass GPT-2 Small full fine-tuning (0.894)—a 3.4% gap.

However, LoRA provides significant practical advantages beyond accuracy. By freezing the base model and only training 0.22% of parameters, LoRA drastically reduces GPU memory usage. Gradients and optimizer states are only maintained for the adapter weights. In our setup (single NVIDIA L4 GPU), this meant we could run many more LoRA experiments (rank sweeps, LR sweeps, epoch sweeps) in the time it took to run a single Medium full FT experiment, which required approximately 7 hours per epoch. Full fine-tuning of GPT-2 Medium was the most expensive experiment in our study, and our limited compute budget prevented us from exploring larger models (e.g., GPT-2 Large) with full FT. LoRA would make such exploration feasible.

Furthermore, LoRA benefits disproportionately from model scaling: Small→Medium improves full FT by only +0.8% (0.894→0.902) but improves LoRA $r=8$ by +8.2% (0.755→0.837). Extrapolating this trend, LoRA on a larger base model (e.g., GPT-2 Large, 774M) could potentially close the gap to full fine-tuning while remaining tractable on limited hardware.

6.3 Error Analysis (Paraphrase Detection)

We analyzed errors from the best model (Medium full FT, 0.902 acc). The confusion matrix: TP=13,721, FP=2,785, TN=22,752, FN=1,171, yielding precision=0.831 and recall=0.921.

False positives (2,785) are the dominant error type. These tend to be question pairs with high lexical overlap (avg word overlap = 0.388) but different intent, e.g. “What is an incognito window?” vs. “What does an Incognito Window do?”

False negatives (1,171) are true paraphrases using substantially different wording—e.g., “Which is the best private bank to open a saving account in India?” vs. “What bank is good for a savings account in India?”

Sentence length analysis reveals that true negatives have significantly larger length differences between sentence pairs (avg 4.6 words) compared to false positives (2.8 words) and true positives (2.4 words).

6.4 Prediction Restriction

By restricting paraphrase predictions to only the “yes” (token 8505) and “no” (token 3919) logits rather than taking argmax over the full 50,257-token vocabulary, we avoid interference from semantically equivalent but lexically different tokens (e.g., “Yes”, “sure”, “yeah”). Without restriction, the model may assign highest probability to such variants, leading to incorrect predictions even when it has correctly learned the paraphrase relationship.

7 Leaderboard Results

Table 5: Leaderboard scores.

Task	Dev Score	Test Score
Paraphrase Detection (Accuracy)	0.902	0.901
Sonnet Generation (chrF)	43.42	41.838

8 Conclusion

We implemented LoRA adapters for GPT-2 and conducted a systematic study of parameter-efficient fine-tuning. Our key findings are: (1) LoRA effectiveness scales with base model capacity—Small is flat across ranks while Medium scales log-linearly; (2) learning rate matters more than rank— $r=8$ at $lr=5e-5$ outperforms $r=32$ at $lr=1e-5$ with $4\times$ fewer parameters; (3) LoRA does not close the gap to full fine-tuning at this scale (best 0.860 vs. 0.902), though it enables faster experimentation under limited compute; and (4) sonnet generation captures Shakespearean style but not structure (10.1% rhyme adherence).

References

- [1] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.
- [2] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019.
- [3] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation, 2021.
- [4] Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. Reft: Representation finetuning for language models, 2024.
- [5] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI*, 2019. Accessed: 2024-11-15.
- [6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
- [7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.
- [8] Maja Popović. chrF: character n-gram F-score for automatic MT evaluation. In Ondřej Bojar, Rajan Chatterjee, Christian Federmann, Barry Haddow, Chris Hokamp, Matthias Huck, Varvara Logacheva, and Pavel Pecina, editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, September 2015. Association for Computational Linguistics.