

Fine-Tuning GPT-2 for Sentiment Analysis, Paraphrase Detection, and Sonnet Generation with Parameter-Efficient Adaptation

Stanford CS224N Default Project

Carl Liu
Stanford University
cslu@stanford.edu

Zikun Zhu
Stanford University
zikun@stanford.edu

Abstract

Pre-trained language models such as GPT-2 [1] achieve strong performance on diverse NLP tasks through fine-tuning, but updating all parameters is computationally expensive and risks overfitting on small datasets. We investigate parameter-efficient fine-tuning via LoRA [2] as an alternative to full fine-tuning for sentiment analysis, paraphrase detection, and sonnet generation. We implement GPT-2 (124M parameters) from scratch, load official pre-trained weights, and compare three adaptation strategies—last-linear-layer, full fine-tuning, and LoRA ($r=4$). On sentiment classification, LoRA achieves the best dev accuracy on both SST (0.516) and CFIMDB (0.980) while training only $\sim 149\text{K}$ parameters (0.12% of the model). For paraphrase detection, full fine-tuning reaches 0.892 dev accuracy on Quora, outperforming LoRA at 0.833. For sonnet generation, full fine-tuning achieves a test chrF score of 41.935. Our results suggest that LoRA’s low-rank constraint provides an effective regularizer for single-sentence classification on small datasets, while more complex cross-sentence tasks may require higher-rank adaptations or full parameter updates.

1 Introduction

Large pre-trained language models have transformed NLP by learning general linguistic representations from massive text corpora that transfer effectively to downstream tasks [3, 1]. The standard approach—fine-tuning all model parameters on task-specific data—achieves strong results but is computationally expensive and risks overfitting, particularly on small datasets. As models scale to hundreds of millions or billions of parameters, the cost of storing and deploying separate fine-tuned copies per task becomes prohibitive.

Parameter-efficient fine-tuning (PEFT) methods address this challenge by updating only a small fraction of model parameters while keeping the rest frozen. Approaches include inserting bottleneck adapter modules [4], prepending learnable continuous prompts [5], and injecting low-rank decompositions into weight matrices [2]. Among these, LoRA (Low-Rank Adaptation) is particularly attractive because it introduces no additional inference latency—the low-rank updates can be merged back into the original weights after training—and adds negligible parameters.

In this work, we implement GPT-2 Small (124M parameters) [1] entirely from scratch, load official pre-trained weights, and systematically compare three adaptation strategies across three diverse NLP tasks: sentiment classification on SST [6] and CFIMDB, paraphrase detection on Quora [7], and sonnet generation on Shakespeare. Our extension is LoRA applied to the query and value projections of all attention layers with rank $r=4$. We find that LoRA achieves the best performance on sentiment classification while training only 0.12% of the model’s parameters, but full fine-tuning remains

superior for the more complex paraphrase detection task. We provide qualitative error analysis across all three tasks, examining how the adaptation strategy affects model behavior on specific examples.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes our model architecture and LoRA implementation. Section 4 presents experimental results, and Section 5 provides qualitative analysis. We conclude in Section 6.

2 Related Work

Pre-trained language models. The Transformer architecture [8] introduced self-attention as the core mechanism for sequence modeling. GPT [3] demonstrated that pre-training a decoder-only Transformer on large text corpora with a language modeling objective produces representations that transfer well to downstream tasks. GPT-2 [1] scaled this approach, showing that larger models trained on more diverse data exhibit emergent zero-shot capabilities. In contrast to bidirectional models like BERT [9], GPT-2’s autoregressive design naturally supports both classification (via the final hidden state) and generation tasks, making it well-suited for our multi-task evaluation.

Parameter-efficient fine-tuning. Full fine-tuning of pre-trained models updates all parameters, which is costly and can overfit on small datasets. Houlsby et al. [4] proposed adapter modules—small bottleneck layers inserted between Transformer blocks—that achieve within 0.4% of full fine-tuning performance while adding only 3–4% parameters. Prefix tuning [5] optimizes continuous task-specific vectors prepended to attention keys and values, learning only 0.1% of parameters. LoRA [2] takes a different approach: for a pre-trained weight matrix W_0 , it parameterizes the update as a low-rank decomposition $\Delta W = BA$ where $B \in \mathbb{R}^{d \times r}$ and $A \in \mathbb{R}^{r \times d}$ with $r \ll d$. Crucially, LoRA introduces no inference overhead because ΔW can be absorbed into W_0 after training, unlike adapters which add sequential computation. This property motivates our choice of LoRA as the extension for this project.

Tasks and evaluation. Sentiment analysis on the Stanford Sentiment Treebank (SST) [6] is a standard benchmark with 5-class fine-grained labels. Paraphrase detection on the Quora Question Pairs dataset [7] tests whether two questions express equivalent meaning. For text generation evaluation, we use chrF [10], a character-level n-gram metric that is more robust to morphological variation than word-level metrics. For generation, we employ nucleus (top- p) sampling [11], which dynamically adjusts the candidate token set to avoid the degeneration observed with pure top- k or greedy decoding.

3 Approach

Base model. We implement GPT-2 (124M parameters) [1] from scratch following the default project specification, including masked multi-head self-attention with 12 heads and 768-dimensional hidden states, pre-layer-normalized Transformer decoder blocks [8], byte-pair-encoding tokenization, and learned positional embeddings supporting up to 1,024 tokens. We load the official pre-trained weights from HuggingFace, converting from their fused Conv1D format (a single matrix for Q, K, V) into separate linear projections with transposed weights. The output logits share weights with the input token embeddings (weight tying). For optimization, we implement AdamW with decoupled weight decay [12, 13].

Sentiment classification. For sentiment analysis, we extract the hidden state at the last non-padding token position and project it through a dropout layer ($p=0.3$) and a linear classifier to produce logits over the label space. SST uses 5 classes (very negative to very positive) and CFIMDB uses 2 classes (negative/positive). We compare three baselines: the *last-linear-layer* approach freezes all GPT-2 parameters and trains only the classifier head (3,845 trainable parameters for SST), *full fine-tuning* updates all parameters, and *LoRA* freezes the base model while training low-rank adaptations plus the classifier head.

Paraphrase detection. For paraphrase detection [7], we formulate the task as a cloze-style prompt: given two questions, the input is formatted as Question 1: "{s1}" Question 2: "{s2}" Are these questions asking the same thing? and the model generates “yes” or “no.” We

extract the last token’s hidden state and project it to 2-class logits. We fine-tune with both full parameter updates and LoRA on the Quora training set.

Sonnet generation. For sonnet generation, we fine-tune GPT-2 as a causal language model on 131 Shakespeare sonnets. The model learns next-token prediction with teacher forcing and cross-entropy loss. At inference, we condition on the first three lines of a held-out sonnet and generate the remaining lines using top- p nucleus sampling [11] with temperature $T=1.2$ and $p=0.9$, producing up to 128 tokens.

LoRA (our extension). We implement LoRA [2] by injecting trainable low-rank matrices into the query and value projections of each attention layer. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times d}$, we add a low-rank update $\Delta W = BA$ where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times d}$, and $r \ll d$. The modified forward pass computes:

$$h = W_0 x + \frac{\alpha}{r} \cdot BAx \quad (1)$$

where α is a scaling hyperparameter. During fine-tuning, only A and B are trained while W_0 remains frozen. We initialize A with Kaiming uniform and B with zeros so that $\Delta W = 0$ at the start of training, preserving the pre-trained model’s behavior before any gradient updates.

For GPT-2 with $d=768$ and $r=4$, each LoRA pair adds $2 \times 768 \times 4 = 6,144$ parameters. Applied to Q and V projections across all 12 layers, LoRA adds $12 \times 2 \times 6,144 = 147,456$ trainable parameters, or 0.12% of the full model. We set $\alpha=1$ throughout. We implement LoRA ourselves as a `LoRALinear` module that wraps the original `nn.Linear` layer.

4 Experiments

4.1 Data

We use four datasets from the default project: SST [6] (8,544 train / 1,101 dev, 5-class sentiment), CFIMDB (1,707 / 245, binary sentiment), Quora (283,003 / 40,429, binary paraphrase detection), and Shakespeare Sonnets (131 train / 12 dev / 12 test, generation). No additional data was used.

4.2 Evaluation method

SST and CFIMDB are evaluated with classification accuracy on the dev set. Quora paraphrase detection uses binary accuracy, with test predictions submitted to the Gradescope leaderboard. Sonnet generation is evaluated with the chrF metric [10], computed via the `sacrebleu` library. All evaluations are automatic.

4.3 Experimental details

All experiments use GPT-2 Small (124M) with pre-trained weights, trained on a single NVIDIA L4 GPU. Table 1 summarizes the hyperparameters for each experiment.

Experiment	LR	Batch	Epochs	Trainable Params
Sentiment: last-linear-layer	10^{-3}	64	10	3,845
Sentiment: full fine-tuning	10^{-5}	64	10	124.4M
Sentiment: LoRA ($r=4$)	10^{-3}	64	10	$\sim 149K$
Paraphrase: full fine-tuning	10^{-5}	8	10	124.4M
Paraphrase: LoRA ($r=4$)	10^{-4}	8	10	$\sim 149K$
Sonnet generation	10^{-5}	8	10	124.4M

Table 1: Hyperparameters for all six experiments. LoRA uses rank $r=4$ and scaling $\alpha=1$. All models use dropout $p=0.3$ and seed 11711. Sonnet generation uses temperature $T=1.2$ and top- $p=0.9$ for inference.

For LoRA experiments, we use a higher learning rate (10^{-3} for sentiment, 10^{-4} for paraphrase) than full fine-tuning (10^{-5}), consistent with the recommendation in Hu et al. [2] that LoRA modules benefit from larger learning rates since fewer parameters receive gradient updates.

4.4 Results

Table 2 reports our main results across all tasks.

Method	Params	SST Dev	CFIMDB Dev	Quora Dev	Quora Test	Sonnet chrF
Last-linear-layer	3,845	0.451	0.829	—	—	—
Full fine-tuning	124.4M	0.508	0.963	0.892	0.861 [†]	41.935 [‡]
LoRA ($r=4$)	~149K	0.516	0.980	0.833	—	—
Expected baseline	—	0.462 / 0.513	0.861 / 0.976	—	—	—

Table 2: Results across all tasks. Bold indicates best performance per column. Expected baselines (last-linear / full) are from the project handout. [†]Paraphrase test accuracy from Gradescope leaderboard. [‡]Sonnet chrF is the test leaderboard score (Rank 22). All other scores are dev set accuracy.

Sentiment classification. LoRA ($r=4$) achieves the **highest accuracy on both sentiment tasks**—0.516 on SST and 0.980 on CFIMDB—while training only 0.12% of the model parameters. Full fine-tuning reaches 0.508 and 0.963 respectively, and the last-linear-layer baseline obtains 0.451 and 0.829. Our baseline numbers fall slightly below the expected values from the handout; the largest gap is 3.2 points on CFIMDB last-linear-layer, which we attribute to random seed and hyperparameter differences. We believe the low-rank constraint acts as an implicit regularizer: with only 149K trainable parameters, LoRA is less prone to overfitting on these relatively small sentiment datasets than full fine-tuning with 124M parameters.

Paraphrase detection. Full fine-tuning achieves 0.892 dev accuracy, well above LoRA at 0.833—a 5.9 percentage point gap. On the test leaderboard, full fine-tuning obtains 0.861 accuracy. Unlike sentiment classification, paraphrase reasoning requires comparing two full sentences presented as a single concatenated input. A rank-4 decomposition appears insufficient to reshape the cross-sentence attention patterns needed for this task. We note that LoRA uses a $10\times$ higher learning rate (10^{-4} vs. 10^{-5}) to compensate for the smaller parameter count, but the gap persists.

Sonnet generation. Our model achieves a test chrF score of **41.935**, ranking 22nd on the class leaderboard. We use full fine-tuning for generation since the entire model must adapt to Shakespearean English. The dev chrF (27.7) is notably lower than the test score, likely due to the very small evaluation set (12 sonnets each) and the stochastic nature of nucleus sampling.

5 Analysis

5.1 Sentiment classification error analysis

To understand how adaptation strategies differ, we examine specific SST dev examples where methods disagree. Table 3 shows representative cases.

Sentence	Gold	Full	LoRA	Last
“The Queen of the Damned can not be said to suck.”	3	4	3	3
“It’s a remarkably solid . . . tour de force.”	4	4	3	3
“No one goes unindicted here, which is probably for the best.”	2	4	3	1

Table 3: SST dev examples where methods disagree. Labels: 0=very negative, 1=negative, 2=neutral, 3=positive, 4=very positive. Bold indicates correct prediction. “Full” = full fine-tuning, “Last” = last-linear-layer.

Full fine-tuning tends to over-predict sentiment intensity: it assigns “very positive” (4) to the hedged double-negation “can not be said to suck” and to the ironic “no one goes unindicted.” LoRA, constrained to a low-rank subspace, produces more conservative predictions that are closer to the neutral center of the label space. This conservatism paradoxically helps overall accuracy by avoiding

confident errors on ambiguous sentences. Conversely, LoRA sometimes under-predicts strong superlatives like “tour de force,” where full fine-tuning’s greater expressiveness correctly captures the intensity.

On binary CFIMDB (0.980 vs. 0.963), the two methods largely agree, suggesting that binary sentiment—where the neutral class does not exist—is simple enough that LoRA’s constrained updates suffice. The regularization benefit is most pronounced on the small CFIMDB dataset (1,707 training examples), where overfitting is a greater risk.

5.2 Paraphrase detection error analysis

We analyze cases where full fine-tuning correctly classifies but LoRA fails, which accounts for the 5.9-point accuracy gap. Table 4 shows representative examples.

Question 1	Question 2	Gold	Full	LoRA
“Why is Lake Tahoe so low?”	“Why is Lake Tahoe named Lake Tahoe?”	No	No	Yes
“What is vaccination?”	“What is in a vaccine?”	No	No	Yes
“Who are the people with the most followers on Quora?”	“Who is the most popular writer on Quora?”	Yes	Yes	No

Table 4: Quora dev examples where full fine-tuning is correct but LoRA errs. Bold indicates correct prediction.

Two dominant error patterns emerge for LoRA. First, **false positives on high-overlap non-paraphrases**: questions sharing entities and syntactic structure but asking fundamentally different things (“Lake Tahoe so low” vs. “named Lake Tahoe”) fool LoRA into predicting paraphrase. Second, **false negatives on low-overlap true paraphrases**: semantically equivalent questions with different phrasing (“most followers” vs. “most popular writer”) are missed by LoRA because the surface forms differ substantially.

These patterns suggest that rank-4 LoRA produces attention updates that rely more heavily on lexical overlap and less on deeper semantic comparison. Full fine-tuning, with access to all 124M parameters, develops richer cross-sentence representations that distinguish genuine semantic equivalence from superficial word sharing. A higher LoRA rank (e.g., 8 or 16) applied to Q, K, and V projections might close this gap by allowing more expressive attention modifications.

5.3 Sonnet generation analysis

We qualitatively evaluate the generated sonnets against the true Shakespeare sonnets. The model is conditioned on the first three lines (provided as input) and must generate the remaining lines.

Strengths. The model successfully captures surface-level Shakespearean style, producing archaic constructions (“thee,” “thou,” “dost,” “thy”) and poetic vocabulary. The first 1–2 generated lines after the prompt often maintain reasonable coherence with the prompt’s theme, as in Sonnet 7:

*In loving thee thou know’st I am forsworn,
But thou art twice forsworn, to me love swearing;
In act thy bed-vow broke, and new faith torn;
Within my love for thee commitst perpetual lyre;
To snatch thine heart and sleep away, however,
The steadfast behestings showst’st three moods entirely[...]*

Failure modes. Several systematic failures emerge. (1) **Meter degradation**: iambic pentameter is maintained for at most 1–2 generated lines before line lengths become irregular. (2) **Rhyme scheme collapse**: the ABAB CDCD EFEF GG scheme is not reliably maintained; the model does not learn to plan ahead for end-of-line rhymes. (3) **Semantic incoherence**: generated lines often become word salad by the second quatrain (e.g., “grey brag went on lose”). (4) **Tokenization artifacts**: Sonnet 8 contains a Chinese character and French word mid-line, likely BPE tokens from GPT-2’s pre-training data surfacing during high-temperature sampling. (5) **Incomplete generation**: several sonnets end with truncated phrases (e.g., “Give the” in Sonnet 9).

These failures reflect the fundamental difficulty of generating structured poetry with a small fine-tuning corpus (131 sonnets). The model learns local stylistic patterns but fails to capture the global structural constraints of the sonnet form.

6 Conclusion

We implemented GPT-2 (124M parameters) from scratch and compared three adaptation strategies—last-linear-layer, full fine-tuning, and LoRA ($r=4$)—across sentiment classification, paraphrase detection, and sonnet generation. Our key finding is that LoRA’s effectiveness is **task-dependent**: its implicit regularization through low-rank constraints achieves the best accuracy on sentiment classification (SST 0.516, CFIMDB 0.980) with only 0.12% trainable parameters, but full fine-tuning remains superior for the more complex paraphrase detection task (0.892 vs. 0.833). For sonnet generation, full fine-tuning achieves a test chrF of 41.935. Our error analysis reveals that LoRA produces more conservative predictions that avoid confident errors on ambiguous sentiment examples, while its limited parameter budget hinders the cross-sentence reasoning needed for paraphrase detection.

Future work. Several directions could strengthen these findings. A **LoRA rank sweep** across $r \in \{2, 4, 8, 16\}$ would characterize the parameter-performance trade-off and determine whether higher ranks close the 5.9-point gap on paraphrase detection. Applying LoRA to **key projections and MLP layers**, in addition to query and value, could provide more expressive adaptations. For sonnet generation, **constrained decoding** methods that enforce rhyme and meter constraints could address the structural failures we observed. Finally, evaluating LoRA on larger GPT-2 variants (Medium, Large) would test whether the regularization benefit scales with model size.

Limitations. Training loss curves were not saved before the cloud VM was deleted. We submitted full fine-tuning paraphrase predictions to the test leaderboard but did not submit LoRA paraphrase predictions, preventing a direct test-set comparison. LoRA was not applied to sonnet generation.

7 Team Contributions

Carl Liu: Carl led the parameter-efficient adaptation portion of the project by implementing the LoRA modules and integrating them into our GPT-2 fine-tuning pipeline. He also supported the experimental setup needed to evaluate how lightweight adaptation compared with standard fine-tuning across our tasks.

Zikun Zhu: Zikun implemented the GPT-2 model and core training pipeline, ran the leaderboard submissions, and synthesized the project findings in the final report. He also helped connect the experimental results to the broader narrative of the paper.

References

- [1] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.
- [2] 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. *arXiv preprint arXiv:2106.09685*, 2021.
- [3] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.
- [4] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799, 2019.
- [5] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 4582–4597, 2021.

- [6] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.
- [7] Samuel Fernando and Mark Stevenson. A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics*, pages 45–52, 2008.
- [8] 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.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186, 2019.
- [10] Maja Popović. chrF: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, 2015.
- [11] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.
- [12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

A Generated Sonnets

Below are two representative generated test sonnets (out of 12 total). The first three lines of each sonnet (in normal font) are the prompt; the remaining lines (in italics) are model-generated via top- p sampling ($T=1.2$, $p=0.9$).

Sonnet 0. Those lips that Love’s own hand did make
 Breathed forth the sound that said “I hate”
 To me that languished for her sake;
*Of hair beauty records e’en house youth like plenitude;
 Then the slow striving spent leisure wasted
 Between his lips was unsaid boast, and that grey brag went on lose.
 Or though folly no doubt linger o’er
 Your cruel well-deserved turning, I’m afraid
 All knew that your fond torment loves you so.
 There aught I only wished, lovest thou least I,
 To hear love to copy what spake you;
 So they thought they shared such original sorrows,
 For it disturbed my heart so to see thee that its grief is*

Sonnet 7. In loving thee thou know’st I am forsworn,
 But thou art twice forsworn, to me love swearing;
 In act thy bed-vow broke, and new faith torn;
*Within my love for thee commitst perpetual lyre;
 To snatch thine heart and sleep away, however,
 The steadfast behestings showst’st three moods entirely;
 Thou that dwelleth fever delights therein,
 Then bestowing upon me sorrow whereof thou takest no rest
 Since I this day plead thine enemies for thine disgrace.
 That suffers what weary age may from thine bound,
 To arms thou dost prefer the days to my eternal age.*