

DoRA: Parameter-Efficient Fine-Tuning for GPT-2 on Cloze Paraphrase Detection and Sonnet Generation

Stanford CS224N {Default}

Aniket Gupta Anjani Pangal Mallika Parulekar

Stanford University

aniketg@stanford.edu apangal@stanford.edu mallika.parulekar@stanford.edu

Abstract

We implemented core components of GPT-2 from scratch and validated the implementation on sentiment classification before adapting it to two downstream tasks: cloze-style paraphrase detection and Shakespearean sonnet generation. On sentiment classification, our GPT-2 implementation achieved dev accuracies of 52.0% on SST and 97.6% on CFIMDB with full fine-tuning. For paraphrase detection, we compared full fine-tuning, MLP-only training, LoRA, and DoRA-based parameter-efficient fine-tuning. Full fine-tuning remained the strongest baseline, while LoRA and DoRA narrowed the gap substantially using less than 3% trainable parameters. We further improved PEFT performance by applying LoRA on *KQV+MLP* on GPT-2, which performed better than restricting updates to *KQV* attention layers alone. Finally, we attempted to reproduce the performance gap between LoRA and DoRA seen in lower ranks in previous work, and were able to see this at a small scale. For sonnet generation, we compared top- p sampling and beam search, added repetition penalty, and tested a two-stage training pipeline that first trains on a broader sonnet corpus and then fine-tunes on Shakespeare. Beam search slightly outperformed top- p on dev chrF in our milestone experiments (41.65 vs. 41.34). However, after incorporating repetition-aware decoding, both methods produced more fluent outputs with less degenerate repetition, and top- p sampling ultimately surpassed beam search (43.6 vs. 43.05). Our results show that parameter-efficient adaptation is effective for paraphrase detection, whereas tasks like sonnet generation benefit from decoding heuristics like repetition penalty.

1 Key Information to include

- Mentor: Simon Kim
- Late Days (if you are using any): 1

Team member	Late days remaining
Aniket Gupta	6
Anjani Pangal	6
Mallika Parulekar	5
Total pooled	17
Team size	3
Late days applied for the report	2

2 Introduction

Full fine-tuning updates all model parameters for each downstream task, which can be costly in memory and compute and is often unnecessary when adapting a pretrained model to a narrow task. This motivates parameter-efficient fine-tuning (PEFT), which preserves the pretrained backbone while

learning only a small number of additional parameters. In this project, we study this tradeoff in the context of the Stanford CS224N final project by adapting GPT-2 to two different tasks: cloze-style paraphrase detection and Shakespearean sonnet generation.

These tasks highlight different aspects of language modeling. Paraphrase detection is a semantic classification problem: given two questions, the model must determine whether they express the same meaning. This is challenging because surface similarity can be misleading. In contrast, sonnet generation is a constrained long-form generation task. Given the first three lines of a Shakespearean sonnet, the model must generate the rest coherently and stylistically consistently despite the small dataset.

For paraphrase detection, we compare full fine-tuning with PEFT methods, focusing on LoRA [1] and DoRA [2]. LoRA applies low-rank updates to frozen pretrained weights, while DoRA further separates weight direction and magnitude to better approximate full fine-tuning. We also study adapter placement and find that applying adapters to both KQV and MLP layers outperforms adapting only KQV. Additionally, we run rank sweeps to compare LoRA and DoRA under controlled settings.

For sonnet generation, we instead rely on full fine-tuning and explore how training and decoding choices affect output quality. We compare beam search and top- p sampling and introduce repetition-aware decoding to reduce degenerate outputs. We also experiment with gradient clipping, weight decay, and a two-stage training setup that pretrains on a broader sonnet corpus before fine-tuning on Shakespeare.

Overall, our results show that the best adaptation strategy depends on the task: PEFT methods are highly competitive for semantic classification, while full fine-tuning combined with careful decoding is more effective for low-resource generation tasks.

3 Related Work

Our project builds on pretrained Transformer language models, parameter-efficient fine-tuning (PEFT), and decoding methods for neural text generation. We use GPT-2 [1], a decoder-only Transformer based on the self-attention architecture of Vaswani et al. [2], as the backbone for both paraphrase detection and sonnet generation.

A key challenge in adapting large pretrained models is the cost of full fine-tuning. PEFT methods address this by updating only a small subset of parameters. LoRA [3] models task adaptation through low-rank updates to pretrained weights, while DoRA [4] extends this idea by decomposing weight updates into magnitude and direction. Our paraphrase experiments are motivated by this line of work, and compare full fine-tuning against LoRA and DoRA under different adapter placements.

For paraphrase detection, prior work has shown that pretrained Transformers are highly effective for semantic similarity and sentence-pair tasks. Models such as BERT [5] and Sentence-BERT [6] demonstrate the strength of pretrained representations for capturing paraphrastic meaning. In contrast to these encoder-based approaches, our work studies how a decoder-only GPT-2 model can be adapted to paraphrase classification.

Our sonnet-generation experiments relate to work on neural text decoding. Holtzman et al. [7] show that decoding strategy strongly affects output quality, especially by controlling degeneration and repetition. Motivated by this, we compare beam search and top- p sampling, and study the effect of repetition-aware decoding in a low-resource poetry generation setting.

4 Approach

4.1 Overview and backbone

We first implemented the required GPT-2 components from the default project, including masked multi-head self-attention, transformer layers, token and positional embeddings, and the Adam optimizer update needed for sentiment classification. We used GPT-2 small (124M parameters) as our base architecture throughout the project. Before moving to the extension tasks, we validated the correctness of the implementation by fine-tuning the model on SST and CFIMDB sentiment classification.

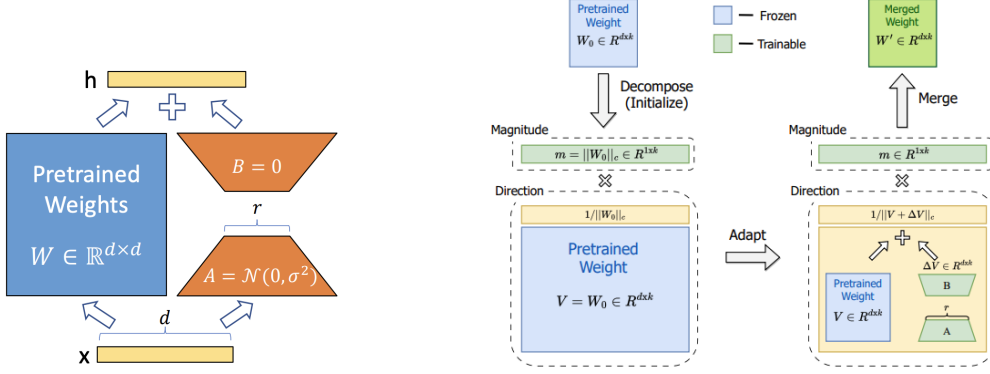


Figure 1: Left: LoRA adds a trainable low-rank update to a frozen pre-trained weight matrix. Right: DoRA decomposes the pre-trained weight into magnitude and direction, learns the magnitude directly, and applies a LoRA-style update to the directional component. Adapted from Hu et al. [3] and Liu et al. [4].

4.2 Paraphrase detection

We formulate paraphrase detection as a cloze-style classification problem using a prompt built from two questions (q_1, q_2) . GPT-2 encodes the prompt autoregressively, and we use the final hidden state h_T to predict the binary label $y \in \{0, 1\}$:

$$\hat{y} = \text{softmax}(W_{\text{cls}}h_T + b_{\text{cls}}). \quad (1)$$

We train using standard cross-entropy loss:

$$\mathcal{L}_{\text{para}} = - \sum_{i=1}^N \log p_{\theta}(y_i | q_{1,i}, q_{2,i}). \quad (2)$$

Our baselines are:

1. **MLP-only**: freeze GPT-2 and train only the classification head.
2. **Full fine-tuning (FFT)**: update all GPT-2 parameters.

We then compare two PEFT methods.

LoRA. For a pre-trained linear layer with weight matrix $W_0 \in \mathbb{R}^{k \times d}$, LoRA freezes W_0 and learns a low-rank update ΔW :

$$W = W_0 + \Delta W, \quad \Delta W = \frac{\alpha}{r} BA, \quad (3)$$

where $A \in \mathbb{R}^{r \times d}$ and $B \in \mathbb{R}^{k \times r}$ are trainable low-rank factors and $r \ll \min(k, d)$ is the LoRA rank. The corresponding linear transformation becomes

$$h = W_0 x + \frac{\alpha}{r} BA x. \quad (4)$$

This lets us adapt the model while training only a small fraction of the parameters [3].

DoRA. DoRA starts by decomposing the pre-trained weight into a magnitude term and a directional term [4]. Let $\|\cdot\|_c$ denote the column-wise ℓ_2 norm of a matrix. We write

$$m = \|W_0\|_c, \quad V = \frac{W_0}{\|W_0\|_c}, \quad (5)$$

where m stores the column magnitudes and V stores the normalized direction. DoRA keeps the pre-trained directional matrix V as the base and applies a LoRA-style update to the direction:

$$\Delta V = \frac{\alpha}{r} BA. \quad (6)$$

For each column j , the adapted weight is

$$W_{:,j} = m'_j \cdot \frac{V_{:,j} + \Delta V_{:,j}}{\|V_{:,j} + \Delta V_{:,j}\|_2}, \quad (7)$$

where m'_j is a learnable magnitude parameter initialized from the pretrained weight. Intuitively, LoRA asks the low-rank update to capture both scale and direction, whereas DoRA lets the model learn magnitude separately and uses the low-rank branch only for directional adaptation.

Adapter placement. We study not only *which* PEFT method to use, but also *where* to apply it. Specifically, we compare adaptation on:

1. attention projections only (KQV), and
2. attention plus feed-forward layers ($KQV+MLP$).

This is one of our key project ablations. In our final experiments, applying LoRA/DoRA to $KQV+MLP$ performed better than restricting adaptation to KQV alone. The layers are labeled in 2.

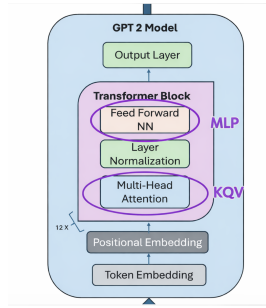


Figure 2: GPT Architecture with DoRA/LoRA targets labeled

4.3 Sonnet generation

For sonnet generation, we fine-tune GPT-2 as a conditional language model. Each training example is a sonnet, and at inference time the model is conditioned on the first three lines and generates the remainder autoregressively.

Unlike paraphrase detection, our main focus here is not PEFT but the interaction between full fine-tuning and decoding. We compare:

1. **Beam search**, which searches for high-likelihood continuations,
2. **Top- p sampling**, which samples from the smallest set of tokens whose cumulative probability exceeds p [7].

We additionally apply repetition-aware decoding to reduce degenerate loops.

To improve performance in the low-resource Shakespeare setting, we also use a two-stage training pipeline. We first fine-tune GPT-2 on the **Deep-speare SONNET corpus** [8], which provides a broader collection of English sonnets and helps the model learn general poetic structure, sonnet-style language, and longer-range verse patterns. We then further fine-tune this intermediate model on the Shakespeare sonnet dataset. This intermediate training step is intended to give the model a better stylistic prior before specializing to Shakespearean diction and continuation.

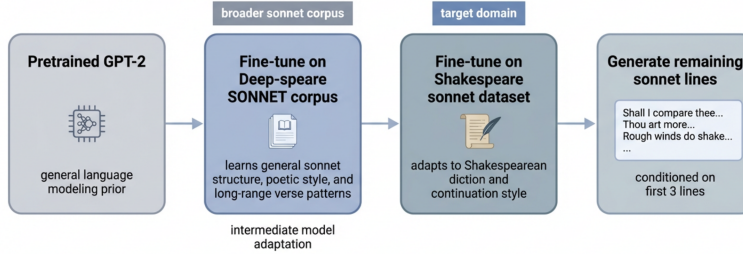


Figure 3: Two-stage fine-tuning pipeline for sonnet generation.

In addition to decoding, we experiment with gradient clipping and weight decay during training. We also experiment with masked loss, which ignores the padded tokens. Together, these experiments let us study how much generation quality can be improved through a combination of full fine-tuning, better initialization via intermediate sonnet-domain training, and decoding controls.

4.4 Implementation details and project-specific contributions

We use the provided GPT-2 baseline from the CS224N default project as the starting point, and implement the task-specific adaptation ourselves. In particular, our project contributions include:

1. Implementing and evaluating LoRA and DoRA for paraphrase detection,
2. Performing an adapter-placement ablation (KQV vs. $KQV+MLP$),
3. Studying effect of pre-training on general sonnet corpus on sonnet generation task,
4. Adding repetition-aware decoding and additional optimization experiments for sonnet generation.

5 Experiments

5.1 Data

We use four datasets across the project. For sentiment validation, we use SST [9] and CFIMDB[10], following the default project pipeline. For paraphrase detection, we use the Quora subset provided in the assignment, with 141,506 training examples, 20,215 dev examples, and 40,431 test examples.

For sonnet generation, we use the Shakespeare sonnet dataset provided in the CS224N project as our target dataset. For intermediate domain training, we use the **Deep-speare SONNET corpus** [8]. To avoid leakage, we exclude Shakespearean sonnets from the general corpus before training, leaving **2664 training** examples and **332 validation** examples. We first fine-tune GPT-2 on this filtered general sonnet corpus and then fine-tune further on the Shakespeare dataset.

5.2 Evaluation method

For sentiment classification, we report accuracy on the dev set. For paraphrase detection, we report dev accuracy and macro-F1 during model selection, and we use the test leaderboard accuracy for the required final project submission. For sonnet generation, we evaluate with chrF [11], which compares generated text and reference text using character-level n -grams. We also report qualitative findings related to repetition.

5.3 Experimental details

All experiments use GPT-2 small as the backbone. In our milestone experiments, paraphrase models were trained on a single NVIDIA L4 GPU (24GB) with batch size 64 and AdamW [12]. Full fine-tuning used learning rates $\{10^{-5}, 3 \times 10^{-5}\}$, while LoRA swept ranks $r \in \{4, 8, 16, 32, 64\}$ and learning rates in $\{10^{-5}, 10^{-4}, 10^{-3}\}$. We initially adapted attention projections only, and each epoch took roughly 45 minutes in the LoRA setting.

For the final paraphrase experiments, we mirrored this setup for DoRA and added an adapter-placement ablation comparing KQV versus $KQV+MLP$. Unless otherwise noted, we kept the same batch size and training budget for fair comparison.

For sonnet generation, we fine-tune GPT-2 as a conditional language model and run a focused ablation over decoding, regularization, and intermediate domain adaptation. We vary the learning rate (10^{-3} , 10^{-4} , 10^{-5}), repetition penalty (1.0–1.5), weight decay (0, 0.01, 0.1), and gradient clipping (max norm 1.0), and compare top- p sampling against beam search. We additionally explore a masked loss function that ignores the padded tokens. We also evaluate a two-stage pipeline in which GPT-2 is first fine-tuned on the filtered Deep-speare SONNET corpus and then adapted to Shakespeare. Most experiments use GPT-2 small, with limited exploratory runs on GPT-2 medium and GPT-2 large. Exact details of the experimented configurations are present in B.1

5.4 Results

Sentiment validation Table 1 shows the sentiment results we use as an implementation sanity check. As expected, full fine-tuning improves over training only the last layer, and performance is much stronger on the binary CFIMDB dataset than on the 5-way SST dataset.

Dataset	Finetuned Layer	Dev Accuracy	Staff Baseline
SST	Last Linear Layer	0.471	0.462
SST	Full Model	0.520	0.513
CFIMDB	Last Linear Layer	0.869	0.861
CFIMDB	Full Model	0.976	0.976

Table 1: Sentiment classification results used to validate our GPT-2 implementation.

Paraphrase detection Our results are shown in Table 2. Full fine-tuning achieved the strongest dev accuracy, but PEFT methods came close while training only a small fraction of the parameters. Both LoRA and DoRA benefited from broader adapter placement, and updating both $KQV+MLP$ outperformed updating only KQV . This supports our hypothesis that PEFT should not be restricted to attention alone. For more detailed results refer to Table 5 where you can see that adding PEFT to the MLP layers to 1-2% improvement, with the most improvement at lower ranks.

Method	Trainable %	Dev Accuracy	Dev F1
MLP-only	–	0.639	–
Full fine-tuning (leaderboard)	100%	0.898	–
LoRA (best milestone config)	1.93%	0.866	0.858
LoRA on KQV only	2.83%	0.875	0.867
LoRA on KQV+MLP	7.09%	0.882	0.875
DoRA on KQV only	2.85%	0.871	0.863
DoRA on KQV+MLP	7.14%	0.882	0.875

Table 2: Paraphrase detection results

We were also interested in attempting to reproduce the results from [4] that showed that DoRA provided particularly high gains over LoRA for smaller ranks, due to its ability to decouple the magnitude and direction of weight updates, which improves parameter efficiency under tight rank constraints. We did see DoRA outperform LoRA for smaller ranks, but by a very narrow margin, as seen in Figure 4 .

Ensemble exploration. We also investigated whether combining LoRA and DoRA predictions could improve over either model alone. At rank 64, the two models agreed on 95.9% of dev examples (with 96.1% accuracy when agreeing), but disagreed on the remaining 4.1%. To understand these disagreements, we computed the Jaccard similarity, the ratio of shared words to total unique words, between each Q1 + Q2 question pair. Pairs with very high overlap are often paraphrases, while pairs with low overlap rarely are; the ambiguous middle range (0.4–0.6) is where models are most likely to disagree. We found that DoRA outperformed LoRA in this moderate-overlap region, correctly

classifying 12 more examples than LoRA in the 0.4–0.6 Jaccard range. Based on this, we tested an overlap-based routing strategy that defers to DoRA for moderate-overlap pairs and LoRA otherwise, but the best ensemble gained only 8 additional correct predictions out of 40,429 dev examples, a negligible improvement. The high agreement rate between the two models suggests that LoRA and DoRA learn similar decision boundaries on this task, limiting the potential for complementary ensembling.

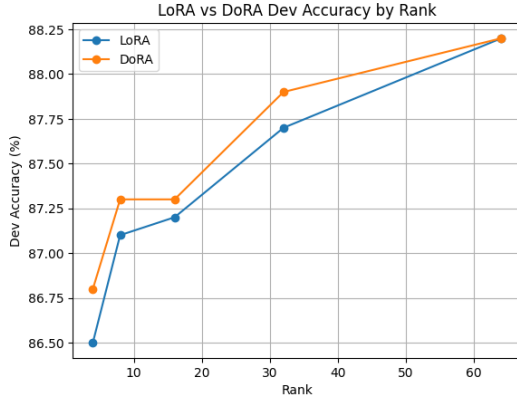


Figure 4: Dev accuracy across adaptation ranks for LoRA and DoRA. DoRA slightly outperforms LoRA at lower ranks, though the margin is smaller than reported in [4].

Sonnet generation The full sonnet-generation sweep is summarized in Table 3. In the milestone setting, beam search only slightly outperformed top- p sampling (41.65 vs. 41.34 chrF), suggesting that decoding alone could not fully solve the task. In the final sweep, however, repetition-aware decoding produced the largest improvement: top- p sampling with repetition penalty reached 43.67 chrF, while the best beam-search configuration reached 43.05. Two-stage training on the filtered Deep-speare sonnet corpus before Shakespeare fine-tuning was also competitive at 43.58 chrF, but its gains were less consistent across downstream learning rates and decoding choices. Exploratory GPT-2 medium and large runs did not outperform the best GPT-2 small configuration, indicating that under our compute budget, decoding and training stability mattered more than simply scaling model size.

Method	Dev chrF
Top- p sampling (milestone baseline)	41.343
Beam search (milestone baseline)	41.648
Beam + repetition penalty + regularization	43.050
Top- p + repetition penalty	43.667
General sonnet pretrain → Shakespeare fine-tune	43.582
GPT-2 large exploratory run	43.010

Table 3: Representative sonnet-generation results from the configuration sweep. Scores are dev chrF from the final evaluation of each saved model.

Overall, the results suggest two different lessons across tasks. For paraphrase detection, PEFT can recover most of the benefit of full fine-tuning, especially when adapters are inserted more broadly. For sonnet generation, gains are harder to obtain because the dataset is tiny and the quality bottleneck is split across model adaptation, optimization, and decoding.

6 Analysis

PEFT works better when adaptation is not too narrow One key from the final paraphrase experiments is that adapting $KQV+MLP$ performed better than adapting only KQV . This is intuitively reasonable: semantic equivalence requires both contextual interaction (attention) and nonlinear

feature transformation (feed-forward layers), and is in line with what we heard from John Schulman’s talk in class.

DoRA provides gains over LoRA for paraphrase detection at smaller ranks, although not as much as in the original paper [4] We hypothesize that this smaller margin may be due to the fact that the datasets in the original paper were common-sense reasoning, which is a much more difficult task than paraphrase detection. Thus it is possible, that for this task, DoRA and LoRA performed more similarly for smaller ranks since the capacity of the model wasn’t fully exhausted.

Decoding improvements are the main source of sonnet gains. Repetition-aware decoding was the most effective change for sonnet generation, reducing the short loops and repeated phrases that often dominated low-resource outputs. Once repetition was controlled, the gap between top- p sampling and beam search became small, suggesting that limited Shakespeare training data is a more important bottleneck than the decoding method itself. Representative examples are shown in Appendix B.2.

Intermediate sonnet-domain pretraining is helpful but fragile. Pretraining on the filtered Deep-sea corpus before Shakespeare fine-tuning provided a useful prior for sonnet structure and style. However, its benefits were not consistently robust across downstream settings, and performance remained sensitive to the fine-tuning and decoding configuration. We therefore view intermediate pretraining as promising, but fragile in this low-resource setting.

7 Conclusion

In this project, we implemented GPT-2 and studied its adaptation to two different downstream tasks: cloze-style paraphrase detection and Shakespearean sonnet generation. On paraphrase detection, full fine-tuning remained the strongest baseline, but parameter-efficient methods closed much of the gap while using far fewer trainable parameters. Our final experiments further showed that adapter placement matters: updating $KQV+MLP$ was better than updating only KQV , indicating that feed-forward layers are an important part of the adaptation bottleneck. On sonnet generation, the strongest result came from top- p sampling with repetition-aware decoding, which reached 43.67 dev chrF and outperformed the best beam-search run at 43.05. Two-stage sonnet-domain pretraining was also competitive at 43.58, but its gains were sensitive to the downstream fine-tuning setup, and larger GPT-2 variants did not improve over the best small-model run under the same budget.

Taken together, our results show that small, task-specific interventions can substantially improve GPT-2 adaptation under limited compute. For paraphrase detection, the most effective interventions were parameter-efficient updates placed in the right parts of the network. For sonnet generation, the most effective interventions were decoding controls and a stable fine-tuning recipe rather than additional model scale. Future work could study structure-aware objectives for poetry, richer qualitative evaluation beyond chrF, and whether transfer from broader poetic corpora can be made more robust through better curriculum design or decoding tuned for verse.

Team contributions (Required for multi-person team)

Aniket Gupta: Implemented the base gpt2 architecture. Implemented LoRA for paraphrase detection. Implemented the base sonnet generation, along with beam search, additionally other optimizations such as repetition penalty, general corpus training, etc. Contributed to milestone and final report paper writing.

Anjani Pangal: Implemented the AdamW optimizer for sentiment fine-tuning. Improved DoRA with gradient detachment for stable magnitude–direction decoupling, and extended LoRA/DoRA adapter placement to MLP layers of the GPT-2 backbone. Conducted hyperparameter sweeps across ranks, explored LoRA–DoRA ensemble methods, and contributed to milestone and final report writing.

Mallika Parulekar: Implemented classifier.py from the starter code for the milestone and implemented DoRA from scratch. Also implemented sweep across ranks for DoRA / LoRA, as well as the ability to enter debug mode with fewer examples in order to ensure data is saved correctly. Ran multiple experiments for paraphrase detection and contributed to milestone and final report paper writing.

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] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.
- [3] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [4] Shih-Yang Liu, Zheng-Xin Wang, Yung-Sung Chen, Yung-Yu Lin, Yu-Chieh Cheng, Ching-Yun Yeh, Wei-Chen Chen, and Yi-Hsuan Yang. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.
- [6] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of EMNLP-IJCNLP*, pages 3982–3992, 2019.
- [7] 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.
- [8] Jey Han Lau, Trevor Cohn, Timothy Baldwin, Julian Brooke, and Adam Hammond. Deep-speare: A joint neural model of poetic language, meter and rhyme. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1948–1958, 2018.
- [9] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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 (EMNLP)*, pages 1631–1642, 2013.
- [10] Divyansh Kaushik, Eduard Hovy, and Zachary C. Lipton. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations (ICLR)*, 2020.
- [11] 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, Lisbon, Portugal, 2015. Association for Computational Linguistics.
- [12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

A Paraphrase Detection

A.1 Paraphrase detection hyperparameters: milestone

In our milestone experiments, all paraphrase models used GPT-2 small on a single NVIDIA L4 GPU with batch size 64 and AdamW. Full fine-tuning used learning rate 1×10^{-5} for 10 epochs. LoRA swept ranks $r \in \{8, 16, 64\}$ and learning rates in $\{10^{-5}, 10^{-4}, 10^{-3}\}$. In the final stage, we reused the same backbone and training protocol while varying the adapter method (LoRA vs. DoRA) and placement (*KQV* vs. *KQV+MLP*). Details of this are in Table 4

Method	Rank	LR	Epochs	Trainable %	Dev Acc
Full fine-tune	–	$1e^{-5}$	10	100%	0.896
Projection head only	–	$1e^{-5}$	10	–	0.639
LoRA	64	$1e^{-3}$	2	1.93%	0.866
LoRA	16	$1e^{-4}$	5	0.55%	0.863
LoRA	8	$1e^{-4}$	5	0.32%	0.861
LoRA	16	$1e^{-3}$	2	0.55%	0.856
LoRA	8	$1e^{-5}$	15	0.32%	0.843

Table 4: Milestone paraphrase hyperparameter sweep (subset of runs shown).

A.2 Paraphrase detection trainable parameters

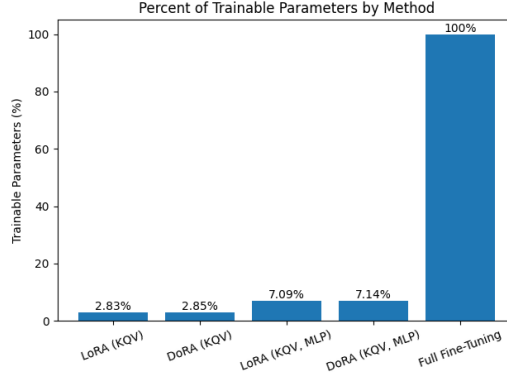


Figure 5: Percent of parameters that are trainable for each Method Type (assumed Rank 64 for PEFT)

A.3 Paraphrase detection improvement after adding PEFT in MLP layers

Rank	LoRA		DoRA	
	KQV	Δ w/ MLP	KQV	Δ w/ MLP
4	84.8	+1.7	84.9	+1.9
8	85.1	+2.0	85.2	+2.1
16	86.2	+1.0	86.1	+1.2
32	86.7	+1.0	86.8	+1.1
64	87.5	+0.7	87.1	+1.1

Table 5: Improvement in dev accuracy from adding MLP adaptation to Q,K,V adapters. Δ denotes the gain relative to adapting Q,K,V alone.

Idx	gpt2	Masked loss	LR	Epochs	Rep. Pen.	WD	Grad Clip	General	Decode	Best dev chrF
0	small	yes	10^{-3}	20	1.0	0.0	no	no	top- <i>p</i>	37.57
1	small	yes	10^{-4}	20	1.0	0.0	no	no	top- <i>p</i>	35.58
2	small	no	10^{-4}	20	1.0	0.0	no	no	top- <i>p</i>	36.16
3	small	no	10^{-3}	20	1.0	0.0	no	no	top- <i>p</i>	38.30
4	small	no	10^{-3}	20	1.2	0.0	no	no	top- <i>p</i>	41.34
5	small	yes	10^{-3}	20	1.2	0.0	no	no	top- <i>p</i>	43.67
6	small	yes	10^{-3}	20	1.2	0.01	no	no	top- <i>p</i>	42.68
7	small	yes	10^{-3}	20	1.2	0.1	no	no	top- <i>p</i>	43.06
8	small	yes	10^{-3}	20	1.5	0.1	no	no	top- <i>p</i>	43.08
9	small	yes	10^{-3}	20	1.5	0.1	yes	no	top- <i>p</i>	43.33
10	small	yes	10^{-3}	20	1.1	0.1	yes	no	top- <i>p</i>	42.75
11	small	yes	10^{-3}	10	1.2	0.01	yes	yes	top- <i>p</i>	43.58
12	small	yes	10^{-4}	10	1.2	0.01	–	yes	top- <i>p</i>	42.89
13	small	yes	10^{-5}	10	1.2	0.01	–	yes	top- <i>p</i>	41.93
14	small	yes	10^{-4}	20	1.2	0.01	–	yes	top- <i>p</i>	41.67
15	small	yes	10^{-3}	10	1.2	0.01	no	no	beam ($w = 5$)	43.35
16	small	yes	10^{-3}	10	1.2	0.01	yes	no	beam ($w = 5$)	43.05
17	small	yes	10^{-4}	10	1.2	0.01	yes	yes	beam ($w = 5$)	36.46
18	small	yes	10^{-4}	10	1.2	0.01	yes	no	beam ($w = 5$)	34.92
19	small	yes	10^{-4}	25	1.2	0.01	yes	no	beam ($w = 5$)	42.22
20	medium	yes	10^{-3}	20	1.2	0.0	no	no	top- <i>p</i>	42.76
21	large	yes	10^{-3}	20	1.2	0.0	yes	no	top- <i>p</i>	43.01

Table 6: Full sonnet-generation configuration sweep. “General” indicates intermediate training on the filtered Deep-speare sonnet corpus before Shakespeare fine-tuning.

B Sonnet generation

B.1 Additional Experiment Details

We explored a broad configuration sweep for sonnet generation covering training objective, decoding, regularization, intermediate sonnet-domain pretraining, and model scale. The full list of runs is provided in Table 6 for completeness and reproducibility.

Table 6 lists the full sonnet-generation configuration sweep explored in our experiments.

B.2 Qualitative Sonnet Examples

The examples below illustrate a common failure mode of the baseline sonnet model: repeated phrases, repeated line openings, and local looping. After adding repetition-aware decoding, the generations become substantially less repetitive and more diverse, although they may still drift semantically or lose global coherence. We refer to these examples in the main analysis when discussing the importance of decoding controls.

Example 1

Prompt

*Whoever hath her wish, thou hast thy will,
And will to boot, and will in overplus.
More than enough am I that vex thee still,*

Before repetition penalty

*Than my poor self for evermore.
For all thy glory, for all thy love’s gain,
And all thy fading beauty’s forlorn
For all thy fading sweets and all thy fading sweets,
And all thy fading beauty’s forlorn
For all thy fading sweets and all thy fading sweets
And all thy lovely allurement,
And all thy fading sweets and all thy fading sweets*

*And all thy fading sweets and all thy fading sweets
And all thy lovely self for evermore.
And all thy fading love's forlorn
And all thy fading sweets and all thy*

After repetition penalty

*even till the dull wood with base infection meet.
Thou art as fair and dearest of all living things; yet for thyself so poor is best
acquainted
That every word shall be mine own sweet self-loving spite, being made by others
but those whose voices are themselves alone.
O, let them be so tickled
They wouldst know it was not their sport when they first took this delight:
"Tis better to die young if life's composition tell."
Thus do they live remembered well! Save them, then, for they live a torment where
love lives least ill,"
Since '*

Example 2

Prompt

*O, call not me to justify the wrong
That thy unkindness lays upon my heart;
Wound me not with thine eye but with thy tongue;*

Before repetition penalty

*Lend me thy hand, that with my broken heart,
Injustice doth break, and is it right to say
That thou hast no right to forgive?
If thou livest, thou hast no honor to give
If thou livest, thou hast no honor to show
If thou livest, thou hast no worth to show
If thou livest, thou hast no honor to show
If thou livest, thou hast no wealth to show
If thou livest, thou hast no wealth to give,
If thou livest, thou hast no heir to sell,
If thou livest, thou hast no title to be.*

After repetition penalty

*Thy outward appearance is fairer than mine own?
Mine eyes are raven black, and their faces bright red beauteous prime
Of these blazon tell, why thou dost breathe in these cheeks:
My lips turn pale when I see thee grow'st. Sweetest things turn sourest by their
deeds—such lips as mine do grow!
Mine eyelids' crests even so, yet they themselves seem more deep-sinks now, making
them look fresh indeed.
Thy brow complexion dimmed after a thousand winters shall besiege thy face."
Tired with this all would make my love eng*