

Evaluating Low-Rank Representation Finetuning for GPT-2 Downstream Tasks

Stanford CS224N {Default}

Alvin Ayuyo

Department of Computer Science
Stanford University
ayuyo17@stanford.edu

1 Key Information

- TA mentor: Sarah Chen
- External collaborators: No
- External mentor: No
- Sharing project: No

Abstract

Large pretrained language models are commonly adapted to downstream tasks through full fine-tuning, which updates millions of parameters and can be computationally expensive. Representation Finetuning (ReFT) has recently been proposed as a parameter-efficient alternative that modifies internal hidden representations rather than directly updating model weights. In this project I evaluate Low-Rank Representation Finetuning (LoReFT) on GPT-2 across multiple downstream tasks including sentiment analysis, paraphrase detection, and sonnet generation.

I show that LoReFT can approach the performance of full fine-tuning on the QQP paraphrase detection benchmark while updating only a small number of parameters. I further evaluate LoReFT on a creative text generation task involving Shakespearean sonnet continuation. While decoding improvements enable full fine-tuning to achieve 26.58 chrF, the best LoReFT configuration reaches 23.89 chrF, suggesting that representation interventions are more effective for classification tasks than for long-form autoregressive generation. These results highlight how intervention placement, capacity, and task structure influence the effectiveness of representation-based adaptation methods.

2 Introduction

Large pretrained language models such as GPT-2 have demonstrated strong performance across a wide range of natural language processing tasks. A common approach to adapting these models to downstream tasks is full fine-tuning, where all model parameters are updated using task-specific data. While effective, full fine-tuning requires updating millions of parameters and can be computationally expensive, particularly when adapting models to multiple tasks.

Parameter-efficient fine-tuning (PEFT) methods aim to address this limitation by adapting pretrained models using a much smaller number of additional parameters. Rather than modifying the entire network, these approaches introduce lightweight modules or low-rank updates that allow the model to specialize to a new task while preserving most of the pretrained parameters. Recent work on Representation Finetuning (ReFT) proposes an alternative approach in which adaptation occurs directly in representation space rather than in weight space.

In this project, I investigate Low-Rank Representation Finetuning (LoReFT), a representation-editing method that applies low-rank transformations to hidden states within a transformer model. Instead of updating the full set of GPT-2 parameters, LoReFT inserts small trainable modules that modify selected hidden representations during the forward pass. This allows the model to adapt to downstream tasks while updating only a small number of parameters.

I evaluate LoReFT on several tasks using GPT-2 as the base model. First, I establish baseline performance on sentiment analysis tasks to verify the implementation of GPT-2 fine-tuning. I then focus on paraphrase detection using the Quora Question Pairs (QQP) dataset, where I compare full fine-tuning against LoReFT and analyze how intervention placement, rank, and optimization hyperparameters affect performance. Finally, I explore a generative task involving Shakespearean sonnet continuation to study how representation interventions behave in long-form autoregressive generation.

My experiments show that LoReFT can approach the performance of full fine-tuning on paraphrase detection while updating far fewer parameters. However, the effectiveness of representation interventions depends strongly on intervention placement and task structure. In generation tasks such as sonnet continuation, LoReFT remains less effective than full fine-tuning, suggesting that representation editing may be more suitable for classification-style tasks than for long-form generative modeling.

3 Related Work

Traditional full fine-tuning modifies all model weights, which can be computationally expensive and memory-intensive for modern large language models. PEFT methods address this issue by freezing the base model and learning small task-specific modifications that enable efficient adaptation. Examples of such approaches include adapter-based methods, low-rank weight updates such as LoRA, and prompt-based tuning methods that optimize additional soft tokens while leaving the model weights unchanged.

Low-rank adaptation methods such as LoRA approximate weight updates using low-rank matrices, allowing the model to learn task-specific behavior while maintaining a small parameter footprint. Prompt-based approaches instead modify the input representation by learning continuous prompt embeddings that guide the model toward desired behaviors without modifying internal weights. While these methods reduce training cost, they still operate primarily by modifying model weights or inputs rather than directly manipulating internal representations.

More recent research has explored representation-level interventions as an alternative mechanism for controlling model behavior. Prior interpretability work has demonstrated that neural representations in language models encode structured semantic information, suggesting that manipulating hidden activations could steer model outputs. Several studies have proposed techniques for modifying internal representations using steering vectors or concept-editing methods that intervene in the model’s residual stream to influence generated outputs. These methods demonstrate that model behavior can be altered without traditional fine-tuning, highlighting the importance of internal representations in controlling model outputs.

Building on this work, the Representation Finetuning (ReFT) framework proposes learning interventions that operate directly on hidden representations rather than modifying model weights. Instead of updating parameters of the underlying language model, ReFT trains lightweight transformation functions that adjust internal activations during the forward pass. A particularly efficient instantiation of this framework, Low-rank Linear Subspace ReFT (LoReFT), performs edits within a low-dimensional subspace of the hidden representation, allowing the model to adapt to downstream tasks while training significantly fewer parameters than conventional PEFT methods [1]. Empirical evaluations show that this approach can achieve competitive or superior performance across several benchmarks while maintaining substantially higher parameter efficiency.

The ReFT framework is also closely connected to work in mechanistic interpretability. Prior research has shown that many linguistic and semantic properties are encoded linearly within neural representations, motivating the idea that targeted interventions in representation subspaces can influence model behavior. Representation-level fine-tuning therefore bridges interpretability and model adaptation.

This project builds on these developments by applying representation-level interventions within a specific downstream task setting. While prior work demonstrates the effectiveness of ReFT-style interventions across several NLP benchmarks, my experiments investigate how these techniques behave under different training configurations and evaluation conditions. By exploring these adaptations, this project contributes to understanding the practical behavior of representation-based fine-tuning methods.

4 Approach

My goal is to improve classification performance while maintaining parameter efficiency during fine-tuning. I begin by adapting a pretrained language model to the task and then explore representation-level fine-tuning methods that modify internal activations rather than updating the full model parameters.

4.1 Representation Fine-Tuning

Beyond conventional fine-tuning, I explore a representation-level intervention approach inspired by the Representation Fine-Tuning (ReFT) framework [1]. The key idea is to modify internal hidden representations during the forward pass while keeping the pretrained model weights frozen.

Let h denote a hidden representation produced by the base model. ReFT introduces a learned intervention function

$$\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^d$$

that modifies the representation before it is passed to subsequent layers. The intervened representation is

$$h' = \Phi(h).$$

Rather than updating the full model parameters, the intervention learns a small set of parameters that adjust hidden activations.

4.2 Low-Rank Representation Intervention

Following the LoReFT formulation [1], the intervention operates within a low-dimensional subspace of the hidden representation. Let $R \in \mathbb{R}^{r \times d}$ be a projection matrix defining a subspace of rank r , where $r \ll d$.

The intervention function is defined as

$$\Phi(h) = h + R^T(Wh + b - Rh),$$

where

$$W \in \mathbb{R}^{r \times d}, \quad b \in \mathbb{R}^r$$

are learned parameters.

This formulation modifies the hidden representation only within the subspace spanned by R . Because the rank r is small, the number of trainable parameters is significantly smaller than full model fine-tuning.

Intuitively, the intervention learns how to adjust internal representations to make them more predictive for the downstream task.

4.3 Training Objective

The model is trained using the same cross-entropy objective as the baseline models. However, the parameters optimized during training differ depending on the configuration.

- **Linear head baseline:** optimize only classification head parameters.
- **Full fine-tuning:** optimize all model parameters.
- **Representation fine-tuning:** optimize only the intervention parameters (R, W, b) and the classification head.

This design allows me to directly compare parameter efficiency and performance across different adaptation strategies.

4.4 Implementation Details

All experiments were implemented in PyTorch using the HuggingFace Transformers library. The GPT-2 architecture and pretrained weights were obtained from the public HuggingFace repository. The sentiment classification head, training pipeline, and experimental framework were implemented by myself.

For representation interventions, I implemented the low-rank transformation modules and integrated them into the forward pass of the transformer network. This allowed me to modify hidden representations during training without altering the pretrained model parameters.

By comparing these approaches, I aim to understand the trade-off between parameter efficiency and downstream task performance when adapting large language models.

5 Experiments

This section summarizes the datasets, evaluation metrics, experimental setup, and results for paraphrase detection and sonnet generation.

5.1 Datasets

Dataset	Description
Quora Question Pairs (QQP)	Sentence-pair classification task where the model predicts whether two questions express the same semantic meaning. Used to evaluate semantic similarity and paraphrase detection.
Shakespeare Sonnets	Corpus of Shakespearean sonnets used for conditional text generation. During evaluation, the model receives the first three lines of a held-out sonnet and generates the continuation.

Table 1: Datasets used in the experiments.

5.2 Evaluation Metrics

Task	Metric
Paraphrase Detection	Classification accuracy on the QQP development set.
Sonnet Generation	chrF score, which measures character n -gram overlap between generated text and reference text and captures lexical and morphological similarity.

Table 2: Evaluation metrics used for each task.

5.3 Experimental Setup

All experiments use GPT-2 as the base architecture. I compare full fine-tuning with the parameter-efficient LoReFT method.

Method	Description
Full Fine-Tuning	All GPT-2 parameters are updated during training.
LoReFT	GPT-2 backbone parameters are frozen while low-rank representation interventions are learned. Only intervention parameters and the task-specific output head are trained.

Table 3: Model configurations compared in the experiments.

For GPT-2 with hidden dimension $d = 768$, LoReFT introduces a small number of trainable parameters depending on the intervention rank r .

Rank r	Trainable Parameters
8	$\sim 12.4\text{K}$
16	$\sim 24.6\text{K}$
32	$\sim 49.4\text{K}$

Table 4: Approximate number of parameters introduced by LoReFT interventions.

5.4 Results: Paraphrase Detection

The initial LoReFT configuration intervenes on the final non-padding token representation in the last two transformer layers, achieving a QQP development accuracy of 0.725.

Configuration	QQP Accuracy
Initial LoReFT (last 2 layers)	0.725
All-layer intervention ($r = 8$)	0.760
$r = 16$ (all layers)	0.770
$r = 32$ (all layers)	0.776
Prefix-suffix intervention ($p = 4, s = 6$)	0.792

Table 5: Ablation results for LoReFT on paraphrase detection.

Optimization strength also significantly affects performance.

The final LoReFT model achieves 0.847 development accuracy, narrowing the performance gap with full fine-tuning to only 0.033 while updating far fewer parameters.

5.5 Results: Sonnet Generation

The tuned full fine-tuning model achieves a development chrF score of 25.37.

The final decoding configuration uses temperature 0.8, top_p 0.9, top_k 40, and no_repeat_ngram_size = 3.

5.6 LoReFT for Sonnet Generation

Suffix interventions perform poorly because the suffix position shifts during autoregressive generation, causing the model to repeatedly modify its own outputs and accumulate errors. Scaling the base model to GPT-2 medium also slightly reduces performance, suggesting that larger models require retuned intervention capacity and optimization settings.

5.7 Summary of Results

Overall, these experiments show that LoReFT can achieve competitive performance with significantly fewer trainable parameters, but its effectiveness depends strongly on intervention design. For paraphrase detection, increasing intervention depth and rank substantially improves performance. For generation tasks, careful control of prefix interventions is crucial, while suffix interventions can destabilize autoregressive decoding.

Training Configuration	QQP Accuracy
Initial configuration	0.792
3 epochs, LR 5×10^{-5}	0.835
5 epochs, batch size 32, LR 10^{-4}	0.847
Full fine-tuning baseline	0.880

Table 6: Effect of optimization hyperparameters on LoReFT performance.

Decoding Configuration	chrF
Baseline tuned model	25.37
+ top_k = 40	26.35
+ no-repeat 3-gram blocking	26.58

Table 7: Effect of decoding strategies on sonnet generation quality.

6 Analysis

Beyond quantitative evaluation, I analyze the behavior of these models in order to better understand when LoReFT succeeds and when it fails. I focus on three aspects: (1) representation intervention design for paraphrase detection, (2) decoding behavior in sonnet generation, and (3) task-dependent differences between parameter-efficient fine-tuning and full fine-tuning.

6.1 Effect of Intervention Placement

One of the most important design decisions in representation fine-tuning is *where* in the model to apply the intervention. The initial LoReFT implementation intervened only on the final non-padding token representation in the last two transformer layers. This configuration achieved a QQP development accuracy of 0.725.

Expanding the intervention depth proved significantly more effective than expanding the number of edited token positions. When LoReFT was applied across all transformer layers while editing the final four non-padding tokens, development accuracy increased from 0.725 to 0.760. This suggests that intervention depth plays a larger role than suffix width, because deeper interventions allow edited representations to propagate through more of the network’s computation.

Further improvements were obtained by increasing the rank of the LoReFT transformation. Increasing rank from $r = 8$ to $r = 32$ increased development accuracy from 0.760 to 0.776, indicating that higher-rank low-dimensional subspaces allow the intervention to express more flexible representation edits.

These results suggest that LoReFT performance depends strongly on both intervention depth and subspace capacity. In contrast to conventional fine-tuning, where improvements typically come from larger models or longer training, LoReFT improvements arise from better placement of representation edits within the network.

6.2 Parameter Efficiency vs. Performance

Despite using far fewer trainable parameters, LoReFT achieved competitive performance relative to full fine-tuning. My best LoReFT configuration achieved 0.847 development accuracy on QQP, compared to 0.880 for full fine-tuning. This result is notable because LoReFT trains only a small number of parameters in the intervention module while keeping the GPT-2 backbone frozen.

This result highlights a key advantage of representation fine-tuning: rather than modifying all weights in the network, the model can adapt to a task by selectively modifying hidden representations in a low-dimensional subspace. These findings support the hypothesis that many task-relevant features are already present in pretrained representations and only require targeted adjustments.

Configuration	chrF
Prefix-only LoReFT ($p = 4, r = 16$)	23.48
Prefix-only LoReFT ($p = 8, r = 16$)	19.66
Rank $r = 32$ (all layers)	23.89
Last four layers only	20.22
Suffix intervention ($s = 2$)	8.39
GPT-2 medium (same config)	22.23

Table 8: LoReFT ablation results for sonnet generation.

6.3 Qualitative Analysis of Sonnet Generation

I also inspected generated sonnet continuations to better understand the effects of decoding strategies and representation interventions. The baseline tuned model often produced fluent language but frequently repeated phrases or short patterns, which degraded long-range coherence.

Introducing decoding constraints significantly improved generation quality. In particular, applying top- k filtering with $k = 40$ increased the chrF score from 25.37 to 26.35, and adding no-repeat 3-gram blocking further increased the score to 26.58. Inspection of generated examples confirmed that repetition control played a critical role in maintaining coherent poetic continuations. Without repetition constraints, the model frequently repeated entire phrases or partial lines.

Lowering the sampling temperature also improved fluency by reducing overly diverse or incoherent outputs. The final decoding configuration produced noticeably more Shakespeare-like continuations while still occasionally failing to maintain rhyme structure or long-range thematic consistency.

6.4 Task Sensitivity of Representation Interventions

The experiments revealed that LoReFT behaves differently depending on the task. For paraphrase detection, both prefix and suffix interventions were beneficial, and expanding intervention depth consistently improved performance.

In contrast, sonnet generation was much more sensitive to intervention placement. Prefix-only interventions worked reasonably well, achieving a chrF score of 23.89, but suffix interventions dramatically reduced performance to 8.39 chrF. This failure mode occurs because suffix tokens change during autoregressive generation, causing the model to repeatedly modify representations of tokens it has just generated. This creates a feedback loop where small errors accumulate over time.

This observation highlights an important difference between classification and generation tasks. In classification tasks such as paraphrase detection, the entire input sequence is static and representation edits remain stable. In autoregressive generation tasks, however, the input sequence evolves during decoding, which makes suffix interventions unstable.

6.5 Impact of Model Size

Finally, I evaluated whether larger base models improve LoReFT performance by replacing GPT-2 small with GPT-2 medium. Surprisingly, performance decreased from 23.89 to 22.23 chrF under the same intervention configuration. This suggests that larger models do not automatically benefit from the same intervention design and may require different ranks, learning rates, or intervention placements.

This result further emphasizes that LoReFT performance depends heavily on intervention design rather than simply model scale.

6.6 Summary of Insights

Overall, my analysis reveals several important properties of representation fine-tuning:

- Intervention depth is more important than intervention width.
- Increasing low-rank capacity improves performance but shows diminishing returns.
- Decoding strategies play a major role in generation quality.

- Representation interventions behave differently across classification and generation tasks.

These findings suggest that representation fine-tuning is a promising parameter-efficient alternative to full fine-tuning, but its success depends strongly on careful design of intervention placement and optimization settings.

7 Conclusion

In this project, I investigated representation-level fine-tuning using the LoReFT framework as a parameter-efficient alternative to conventional full fine-tuning. My experiments explored how low-rank representation interventions can adapt pretrained language models to both classification and generation tasks while updating only a small number of parameters.

For paraphrase detection on the QQP dataset, I found that LoReFT can achieve performance close to full fine-tuning while training dramatically fewer parameters. Starting from an initial development accuracy of 0.725, I improved LoReFT performance through several design choices, including increasing intervention depth, expanding the low-rank subspace, and adjusting optimization hyperparameters. Our final configuration achieved 0.847 development accuracy, narrowing the gap to full fine-tuning (0.880) to only 0.033 while maintaining strong parameter efficiency.

For sonnet generation, I observed that decoding strategies played a major role in generation quality. Adjusting temperature, applying top- k sampling, and introducing no-repeat n -gram blocking improved chrF scores from 25.37 to 26.58. When applied to this task, LoReFT was more sensitive to intervention design. A prefix-only intervention with $p = 4$ and rank $r = 32$ achieved a chrF score of 23.89, demonstrating that representation-level edits can partially adapt the model while still falling short of the full fine-tuning baseline.

My experiments also revealed several broader insights about representation fine-tuning. First, intervention depth was often more important than the number of edited token positions. Second, increasing the rank of the intervention improved performance but exhibited diminishing returns. Third, intervention placement must match the structure of the task: suffix interventions were useful in paraphrase detection but destabilized autoregressive generation. These observations suggest that representation-level fine-tuning works best when the intervention location aligns with how the model uses contextual information.

Despite these promising results, my work has several limitations. LoReFT performance proved sensitive to intervention placement, rank, and optimization hyperparameters, requiring extensive experimentation to achieve strong results. In addition, while LoReFT substantially reduced the number of trainable parameters, applying interventions across many transformer layers increased computational cost during training.

Future work could explore several directions. First, automated methods for selecting optimal intervention layers and positions could make representation fine-tuning easier to apply in practice. Second, more expressive intervention functions or adaptive rank selection may further improve performance. Finally, applying representation-level interventions to larger language models or more diverse tasks could help clarify when parameter-efficient representation editing is most effective.

Overall, my results demonstrate that representation-level fine-tuning provides a compelling middle ground between frozen models and full fine-tuning. By editing internal representations rather than updating all model weights, LoReFT can achieve competitive performance while dramatically reducing the number of trainable parameters.

References

- [1] Zheng Wang Atticus Geiger Dan Jurafsky Christopher D. Manning Christopher Potts Zhengxuan Wu, Aryaman Arora. Reft: Representation finetuning for language models. In *arXiv*, 2024.