

Co-Adaptation in LoRA: Target Placement Effects and Inter-Module Interactions in GPT-2

Stanford CS224N Default Project

Shekhar Sharma

Department of Computer Science
Stanford University
shekhars@stanford.edu

Abstract

Low-Rank Adaptation (LoRA) enables memory-efficient fine-tuning of large language models by injecting trainable low-rank matrices into frozen pretrained weights. While LoRA is widely adopted, the choice of *which* modules to adapt, attention projections, MLP layers, or both, is often made without systematic justification. We conduct a controlled ablation of LoRA target placement on GPT-2 for the paraphrase detection task, varying target modules (attention, MLP, all), rank ($r \in \{4, 8, 16, 32\}$), and model scale (124M, 355M parameters). We find that (1) target placement affects accuracy far more than rank, with MLP-only LoRA outperforming attention-only by 3 points on GPT-2 Small; (2) adapting all modules outperforms either alone, despite attention-only LoRA being substantially weaker than MLP-only adaptation in isolation; and (3) cosine similarity analysis reveals that attention LoRA learns fundamentally different directions when co-adapted with MLP (cosine similarity 0.17 - 0.55) compared to training alone, while MLP directions remain stable (0.59 - 0.83). This *co-adaptation effect* suggests that joint adaptation changes what attention learns, which may help explain why adapting all modules outperforms either target family alone.

1 Key Information

- Mentor: Alisa Levin
- Late Days: 3 applied for the report and code.

2 Introduction

Low-Rank Adaptation (LoRA) [1] is a widely used method for parameter-efficient fine-tuning of large language models. By decomposing weight updates into low-rank matrices $\Delta W = BA$, LoRA enables adaptation with a fraction of the parameters required for full fine-tuning (FFT). The original LoRA paper focused on adapting attention weights rather than MLP weights, and in most experiments applied LoRA to the query and value projections (W_Q and W_V). Many practical LoRA setups have inherited this attention-focused default, though later work has revisited target placement more systematically.

Recent analysis by Schulman [2] found, in their settings, that attention-only LoRA underperforms MLP-only LoRA, and that broader target placement, especially including MLP layers, performs better than attention-only adaptation. However, their discussion was primarily empirical and did not focus on why certain target configurations perform better.

In this work, we conduct a systematic ablation of LoRA target placement on GPT-2 [3] for paraphrase detection task (part of the default project NLP task). We make the following contributions:

1. **Systematic target ablation:** We evaluate three target configurations (attention, MLP, all) across four ranks on GPT-2 Small (124M) and replicate on GPT-2 Medium (355M), establishing that target placement consistently matters more than rank.
2. **Co-adaptation analysis:** Using cosine similarity between learned LoRA weight directions, we find that attention adapters converge to substantially different update directions when co-adapted with MLP adapters compared to training in isolation. This co-adaptation effect provides evidence consistent with an interaction between modules, rather than independent contributions.
3. **Supporting analyses:** We provide layer-selective experiments, Frobenius norm profiling, and FFT weight delta comparisons that contextualize LoRA’s update patterns relative to full fine-tuning.

3 Related Work

Parameter-efficient fine-tuning. Several methods have been proposed to reduce the cost of adapting pretrained language models. Adapter layers [4] insert small bottleneck modules between transformer layers. Prefix-tuning [5] prepends learnable continuous vectors to the input. LoRA [1] decomposes weight updates into low-rank matrices, enabling adaptation without increasing inference latency. QLoRA [6] further reduces memory by quantizing the base model. Our work builds on LoRA and investigates how the *choice of adapted modules* affects both performance and learned representations.

LoRA target selection. The original LoRA paper [1] focused on adapting attention weights, applying LoRA to W_Q and W_V in most experiments. Schulman [2] found that MLP-only LoRA outperforms attention-only LoRA, even when controlling for parameter count. Our work evaluates similar target-placement effects on GPT-2 for paraphrase detection and introduces cosine similarity analysis to investigate possible interactions between adapted modules.

MLP layers as knowledge stores. Geva et al. [7] showed that transformer feed-forward layers function as key-value memories, storing factual associations. This is consistent with our observation that MLP-targeted LoRA outperforms attention-targeted LoRA: if MLP layers play a major role in storing task-relevant information, they may also be a natural site for task-specific adaptation.

Intrinsic dimensionality. Aghajanyan et al. [8] demonstrated that fine-tuning operates in a low-dimensional subspace, providing theoretical grounding for why low-rank methods like LoRA are effective. Our observation that rank has diminishing returns beyond $r = 16$ is consistent with this perspective.

4 Approach

4.1 LoRA Formulation

For a pretrained weight matrix $W_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, LoRA parameterizes the update as:

$$W = W_0 + \frac{\alpha}{r}BA \quad (1)$$

where $B \in \mathbb{R}^{d_{\text{out}} \times r}$, $A \in \mathbb{R}^{r \times d_{\text{in}}}$, $r \ll \min(d_{\text{in}}, d_{\text{out}})$ is the rank, and α is a scaling hyperparameter (set to 32 in all experiments, following [2]). A is initialized with Kaiming uniform and B with zeros, so the initial forward pass is identical to the base model. During fine-tuning, W_0 is frozen and only A , B are updated.

4.2 Target Configurations

Conceptually, each GPT-2 transformer block includes four attention weight matrices (W_Q , W_K , W_V , W_O) and two MLP matrices. We group these into three target configurations:

- **Attention (attn):** query (W_Q), key (W_K), value (W_V), and output projection (W_O)
- **MLP (mlp):** intermediate up-projection ($W_{\text{up}}: d \rightarrow 4d$) and down-projection ($W_{\text{down}}: 4d \rightarrow d$)

- **All:** all six projections

In all configurations, we add an untied linear output projection from hidden dimension d to vocabulary size V , which is always trainable. For paraphrase detection, the model uses this projection to score the verbalizer tokens “yes” and “no” at the final token position.

4.3 Analysis Methods

Frobenius norm profiling. For each LoRA-adapted layer, we compute $\|BA\|_F$ to measure the magnitude of the learned update. This was done in order to find out which layers receive the largest adaptations under each target configuration.

Cosine similarity. To measure co-adaptation, we compare the learned LoRA directions between configurations. For a given layer l and projection p , we compute the flattened product $\text{vec}(B_l^{(c)} A_l^{(c)})$ for configuration c and measure:

$$\text{sim}(l, p) = \frac{\text{vec}(B_l^{\text{all}} A_l^{\text{all}}) \cdot \text{vec}(B_l^{\text{standalone}} A_l^{\text{standalone}})}{\|\text{vec}(B_l^{\text{all}} A_l^{\text{all}})\| \cdot \|\text{vec}(B_l^{\text{standalone}} A_l^{\text{standalone}})\|} \quad (2)$$

where “standalone” is either attn-only or mlp-only. A cosine similarity near 1 indicates the configuration learns similar directions regardless of what other modules are adapted and near 0 indicates the directions are orthogonal.

FFT weight delta comparison. We compare the layer-wise distribution of LoRA updates to the weight changes from full fine-tuning by computing $\|W_l^{\text{FFT}} - W_l^{\text{base}}\|_F$ per layer and overlaying with LoRA norms (both normalized to $[0, 1]$).

5 Experiments

5.1 Data

We evaluate on the Quora Question Pairs paraphrase detection task, using the provided split for the project: 283K training pairs and 40K development pairs. The input is formatted as: Is “{s1}” a paraphrase of “{s2}”? Answer “yes” or “no”: and the model predicts the next token.

5.2 Evaluation Method

We report development set accuracy as our primary metric. All comparisons use the same evaluation protocol: the checkpoint with the highest development accuracy over 5 training epochs.

5.3 Experimental Details

Models. GPT-2 Small (124M parameters, 12 layers, $d = 768$) and GPT-2 Medium (355M parameters, 24 layers, $d = 1024$), both initialized from HuggingFace pretrained weights. The untied output projection adds 38.6M parameters for Small (768×50257) and 51.5M for Medium (1024×50257).

Training. AdamW optimizer [9], learning rate 1×10^{-5} , batch size 32 (effective), 5 epochs. LoRA scaling factor $\alpha = 32$ for all configurations. All experiments use a single NVIDIA A100 80GB GPU except where noted.

Configurations. We evaluate $3 \text{ targets} \times 4 \text{ ranks} = 12$ LoRA configurations on GPT-2 Small, plus full fine-tuning (FFT, all 124M backbone parameters trainable) and head-only (all GPT-2 backbone parameters frozen, only the output projection trains) as baselines. We replicate the $r = 16$ ablation on GPT-2 Medium.

5.4 Results

Table 1 presents our main results. There are three key observations:

Target placement > rank. Moving from attention to MLP at any fixed rank improves accuracy by ~ 3 points. Moving from MLP to all adds ~ 1 point. In contrast, increasing rank from 4 to 32 (an $8 \times$

Table 1: Paraphrase detection dev accuracy on GPT-2 Small (124M). Target placement consistently affects accuracy more than rank. LoRA Params column shows additional backbone parameters from LoRA adapters; the shared output projection (38.6M params for Small) is trainable in all configurations including head-only.

Target	LoRA Params	r=4	r=8	r=16	r=32
Attention	295K - 2.4M	0.807	0.806	0.806	0.807
MLP	369K - 2.9M	0.833	0.835	0.836	0.837
All	664K - 5.3M	0.840	0.839	0.846	0.846
FFT	all 124M backbone			0.888	
Head-only	0 (backbone frozen)			0.644	

increase in LoRA parameters) yields less than 1 point improvement within any target configuration. Attention-only LoRA is essentially flat across all ranks (0.806 - 0.807), indicating rank saturation even at $r = 4$.

Head-only establishes a floor. With only the output projection trainable and all GPT-2 backbone parameters frozen, the model achieves 0.644, substantially below any LoRA configuration. This shows that adapting internal model weights matters for this task, and that even the weakest LoRA configuration (attention-only, 0.807) provides meaningful adaptation of the base model’s representations.

All > MLP despite weak attention. Attention-only LoRA contributes just 0.807 in isolation, yet adding attention adapters on top of MLP (moving from mlp to all) consistently improves accuracy by ~ 1 point. This suggests the improvement from adapting all modules is not a simple sum of independent contributions.

Leaderboard results. Separately from the LoRA ablation, we submitted our best full fine-tuning configuration (GPT-2 Large) to the course test leaderboards. For paraphrase detection: dev accuracy 0.906, test accuracy 0.869. For sonnet generation: dev chrF 42.34, test chrF 41.62. The LoRA target ablation study presented in this paper was conducted on GPT-2 Small and Medium; the leaderboard submissions use a larger model with full fine-tuning.

6 Analysis

6.1 Co-Adaptation Between Target Modules

The observation that all > MLP, despite attention-only being weak, suggests an interaction between the adapted modules. To investigate, we compute the cosine similarity between the learned LoRA weight directions (BA) from the “all” configuration and the corresponding standalone (attn-only or mlp-only) configuration, per layer and per projection.

Figure 1 reveals an interesting asymmetry. When MLP adapters are also present (the “all” configuration), attention LoRA learns update directions that are nearly orthogonal to what it learns in isolation. The key projection is most affected, with cosine similarities as low as 0.04 at $r = 4$. In contrast, MLP projections maintain moderate-to-high alignment (0.40 - 0.69) regardless of whether attention is co-adapted.

This pattern is consistent across ranks: higher rank increases similarity somewhat (at $r = 32$, attention similarity rises to 0.31 - 0.55), but the differential between attention divergence and MLP stability persists at every rank we tested.

We refer to this as a *co-adaptation effect*. The cosine similarity data establishes that attention LoRA converges to substantially different weight directions when MLP adapters are jointly trained, while MLP directions remain relatively stable. This indicates that the benefit of adapting all modules is not a simple sum of independent contributions, the presence of MLP adaptation changes what attention learns. One possible explanation is that attention-only LoRA must simultaneously handle both representational and task-specific adaptation when MLP layers are frozen, while the presence of MLP adapters allows attention to specialize in complementary functions. However, confirming this hypothesis would require causal interventions beyond the scope of this work.

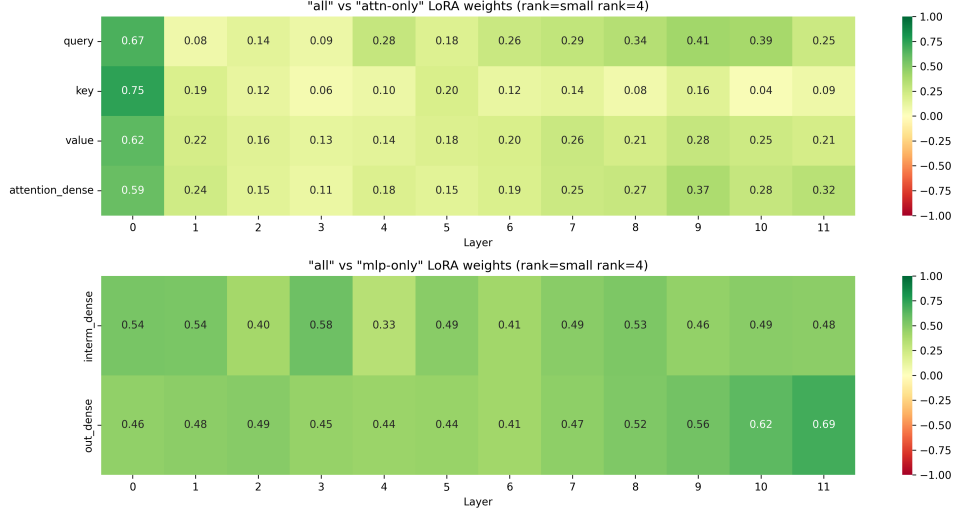


Figure 1: Cosine similarity between LoRA weights learned in the “all” configuration vs. standalone configurations, GPT-2 Small at $r = 4$. **Top:** attention projections (all vs. attn-only). **Bottom:** MLP projections (all vs. mlp-only). Attention directions diverge substantially (sim ≈ 0.04 - 0.28 in middle layers), while MLP directions remain relatively aligned (sim ≈ 0.40 - 0.69).

We note that different random seeds could produce some directional variance between independent runs. However, the systematic contrast, attention directions diverge while MLP directions converge, within the same checkpoint pair, argues against a purely stochastic explanation.

6.2 Update Magnitude by Layer

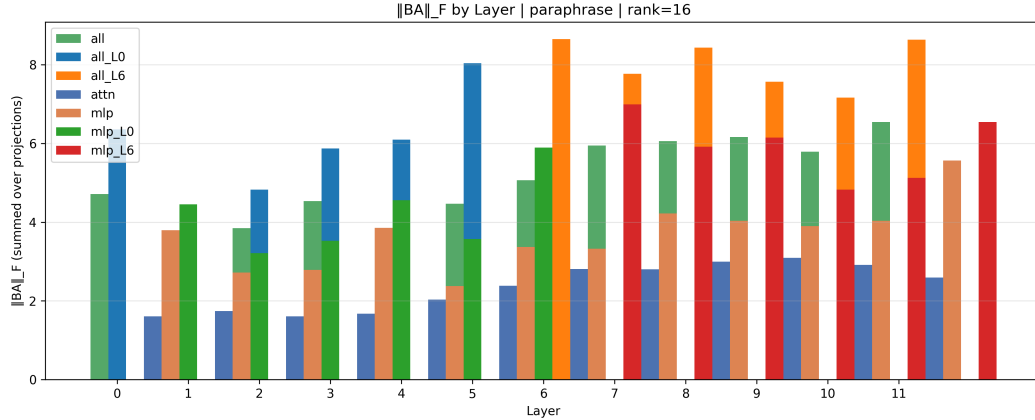


Figure 2: Frobenius norm of learned LoRA updates ($\|BA\|_F$) by layer for GPT-2 Small at $r = 16$. Green (all) dominates at every layer. Updates concentrate in later layers across all target configurations.

Figure 2 shows the per-layer Frobenius norm of the learned LoRA updates. The “all” configuration produces the largest updates at every layer, consistent with its superior accuracy. All configurations show increasing norms toward later layers, suggesting that task-specific adaptation concentrates near the output. We follow it up with in our next analysis.

6.3 Layer-Selective Experiments

To test whether layers with larger norm updates are also the most important, we restrict LoRA ($r = 16$, target=all) to either layers 0 - 5 or layers 6 - 11.

Table 2: Layer-selective LoRA results on GPT-2 Small ($r = 16$). Results are from single runs; margins may not exceed run-to-run variance.

Target	Layers 0 - 5	Layers 6 - 11	All 12
All	0.830	0.823	0.846
MLP	0.819	0.809	0.836

We observe that early layers (0 - 5) yield slightly higher accuracy than late layers (6 - 11) for both targets (Table 2), despite late layers having larger update norms (Figure 2). However, these results are from single training runs and the margins (0.7 - 1.0 points) may not exceed run-to-run variance. We therefore note this as an observation rather than a definitive finding. Neither half matches the performance of adapting all layers, confirming that full-layer adaptation remains beneficial.

6.4 Scale Replication on GPT-2 Medium

To assess whether our findings generalize across model scale, we replicate the $r = 16$ target ablation on GPT-2 Medium (355M parameters, 24 layers).

Table 3: Dev accuracy at $r = 16$: GPT-2 Small (124M) vs. Medium (355M).

Target	Small	Medium
Attention	0.806	0.853
MLP	0.836	0.855
All	0.846	0.864

The ordering $\text{all} > \text{MLP} \geq \text{attention}$ holds at both scales (Table 3), and adapting all modules provides a clear advantage over either target alone on Medium (+1 point). The MLP - attention gap narrows substantially on Medium (0.2 points vs. 3.0 points on Small), though we note this is from single runs and the margin may not be significant.

Cosine similarity analysis on GPT-Medium confirms the same co-adaptation pattern: attention projections in the “all” configuration diverge from standalone attention (similarity 0.10 - 0.75), while MLP projections remain more aligned (0.47 - 0.86). The co-adaptation effect is robust across scales.

7 Conclusion

We conducted a systematic ablation of LoRA target placement on GPT-2 for paraphrase detection and identified a *co-adaptation effect* between attention and MLP adapters. Our main findings are:

1. Target placement affects accuracy far more than rank: moving from attention to all-module LoRA yields 4 points, while quadrupling rank yields <1 point.
2. When co-adapted with MLP, attention LoRA learns directions nearly orthogonal to its standalone solution (cosine similarity as low as 0.04), while MLP directions remain stable. This co-adaptation, not simple additive contribution, could explain why adapting all modules outperforms MLP alone.
3. The co-adaptation effect replicates on GPT-2 Medium (355M), suggesting it is not specific to small models.

Limitations. Our evaluation is limited to a single task (paraphrase detection) and model family (GPT-2). The layer-selective and scale comparisons are based on single runs without seed variance analysis. We did not investigate whether the co-adaptation effect holds for other PEFT methods or at scales beyond 355M parameters.

Future work. Extending the cosine similarity analysis to other tasks and model families would test the generality of the co-adaptation effect. Causal interventions (e.g., patching activations from one configuration into another) could provide stronger mechanistic evidence. Investigating whether LoRA rank can be allocated non-uniformly across layers, informed by the update patterns we observe, is another promising direction.

References

- [1] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.
- [2] John Schulman. LoRA without regret. Thinking Machines blog post, <https://thinkingmachines.ai/blog/lora/>, 2025. In collaboration with Thinking Machines.
- [3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.
- [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*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 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 and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597. Association for Computational Linguistics, 2021.
- [6] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, 2023.
- [7] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495. Association for Computational Linguistics, 2021.
- [8] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328. Association for Computational Linguistics, 2021.
- [9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

A Appendix

A.1 Training Dynamics

Figure 3 shows development accuracy across epochs for all configurations. FFT separates from LoRA configurations immediately and maintains its lead throughout training. The head-only baseline remains flat near 0.64, confirming that the pretrained representations alone are insufficient for the task. Among LoRA configurations, clear tiers emerge: all > mlp > attn. All configurations converge by epoch 3 - 4, indicating that 5 epochs is sufficient for this task.

A.2 Cosine Similarity Across Ranks

Figures 4 and 5 show cosine similarity between “all” and standalone configurations at $r = 16$ and $r = 32$ on GPT-2 Small. Higher rank moderately increases similarity (compare to $r = 4$ in Figure 1), but the core pattern persists: attention directions diverge while MLP directions remain stable.

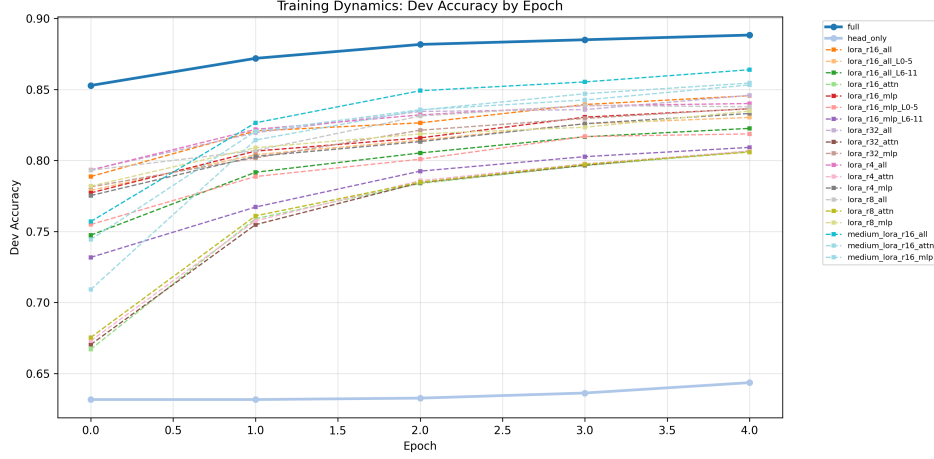


Figure 3: Development accuracy by epoch for all configurations.

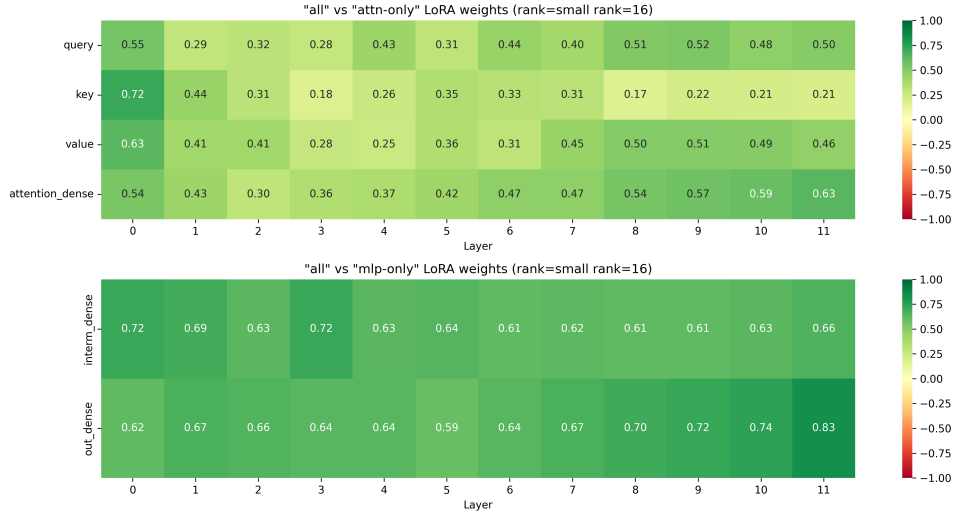


Figure 4: Cosine similarity at $r = 16$, GPT-2 Small.

A.3 GPT-2 Medium Cosine Similarity

Figure 6 shows the co-adaptation pattern on GPT-2 Medium (24 layers) at $r = 16$. The same asymmetry holds: attention projections diverge between “all” and “attn-only” (similarity 0.10 - 0.75), while MLP projections remain more aligned (0.47 - 0.86), confirming scale robustness.

A.4 Parameter Counts

Table 4 lists the LoRA parameter counts for each configuration on GPT-2 Small. The shared output projection (38.6M parameters for Small, 51.5M for Medium) is trainable across all configurations. FFT additionally trains all backbone parameters.

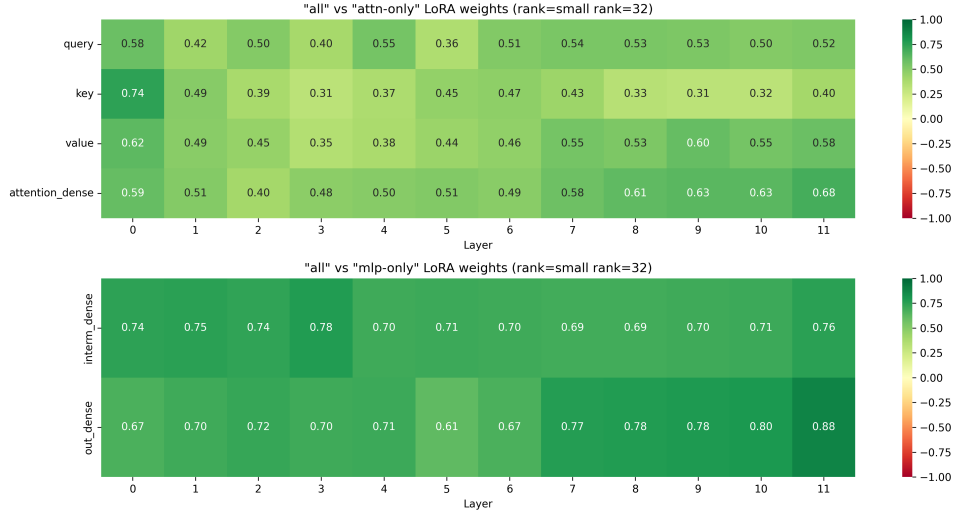


Figure 5: Cosine similarity at $r = 32$, GPT-2 Small.

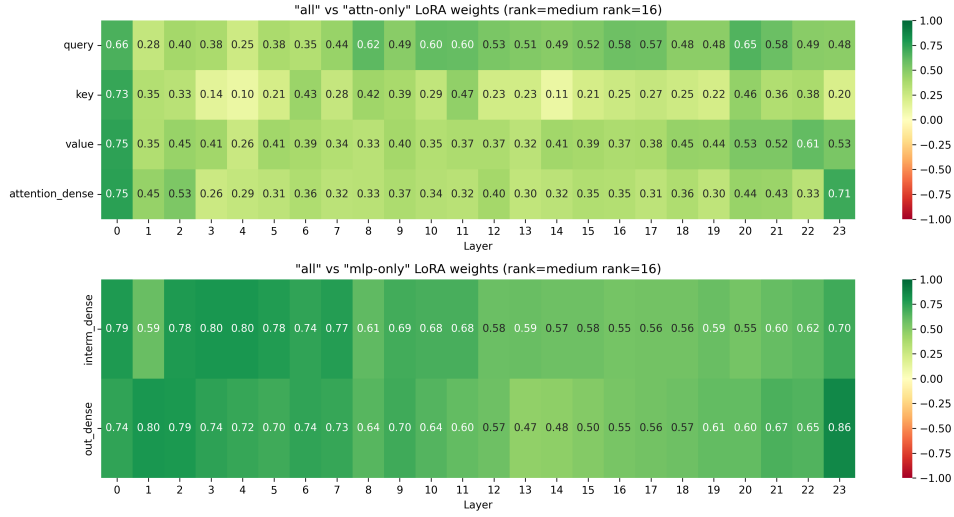


Figure 6: Cosine similarity at $r = 16$, GPT-2 Medium (24 layers).

Table 4: LoRA parameter counts by configuration (GPT-2 Small).

Configuration	LoRA Params
Attention, $r = 4$	294,912
MLP, $r = 4$	368,640
All, $r = 4$	663,552
Attention, $r = 16$	1,179,648
MLP, $r = 16$	1,474,560
All, $r = 16$	2,654,208
Attention, $r = 32$	2,359,296
MLP, $r = 32$	2,949,120
All, $r = 32$	5,308,416