RETHINKING ADAPTER PLACEMENT: A DOMINANT ADAPTATION MODULE PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-rank adaptation (LoRA) is a widely used parameter-efficient fine-tuning method that places trainable low-rank adapters into frozen pre-trained models. Recent studies show that using fewer LoRA adapters may still maintain or even improve performance, but existing methods still distribute adapters broadly, leaving *where to place a limited number of adapters to maximize performance* largely open. To investigate this, we introduce **PAGE** (**P**rojected **A**dapter **G**radient **E**nergy), a gradient-based sensitivity probe that estimates the initial trainable gradient energy available to each candidate LoRA adapter. Surprisingly, we find that PAGE is highly concentrated on a single shallow FFN down-projection across two model families and four downstream tasks. We term this module the **dominant adaptation module** and show that its layer index is architecture-dependent but task-stable. Motivated by this finding, we propose **DomLoRA**, a placement method that places a single adapter at the dominant adaptation module. With only **0.7%** of vanilla LoRA's trainable parameters, DomLoRA outperforms it on average across downstream tasks, including instruction following, mathematical reasoning, coding, and multi-turn conversation. This method also matches or improves other LoRA variants and reduces training time by up to **2.74**$\times$ compared with broad placement, supporting the dominant adaptation module perspective as a practical placement guideline.

1 INTRODUCTION

Large language models (LLMs) have achieved strong general-purpose performance across many tasks. However, off-the-shelf LLMs are not always optimal for a specific target domain or instruction format, so fine-tuning LLMs on task-specific datasets has become common practice. The most direct strategy is full fine-tuning, which updates all model parameters and often provides strong adaptation performance. However, as model sizes grow, it becomes increasingly expensive in computation, memory, and storage. Parameter-efficient fine-tuning (PEFT) mitigates these costs by updating only a small subset of parameters while approaching full fine-tuning performance (Houlsby et al., 2019).

Among PEFT methods, low-rank adaptation (LoRA) (Hu et al., 2022) is the most widely adopted technique, which places trainable low-rank adapters into frozen pre-trained models. In common practice, these adapters are placed at attention and feed-forward network (FFN) projections in most or all Transformer blocks (Dettmers et al., 2023; He et al., 2022). However, recent studies show that using fewer adapters may still maintain or even improve performance. Some methods reduce the number of adapted *layers* (Yao et al., 2024; Gu et al., 2024; Ogawa et al., 2026) while keeping all module types within the selected layers (Figure 1(a)). Others reduce the adapted *module types* (Xue & Mirzasoleiman, 2026; Kunz, 2025; Hayou et al., 2026), such as selecting only FFN or only attention projections, while applying the selected adapters across all layers (Figure 1(b)). Nevertheless, these methods still keep adapters across multiple modules, raising an open question: **where should a limited number of LoRA adapters be placed to maximize adaptation performance?**

To explore this question, we introduce **PAGE** (**P**rojected **A**dapter **G**radient **E**nergy), a gradient-based sensitivity probe for LoRA placement. PAGE uses the empirical Fisher sensitivity of pretrained projection weights to estimate the initial trainable gradient energy available to each candidate LoRA adapter. Evaluating PAGE for every attention and FFN projection, we find its values markedly concentrated at a single shallow FFN down-projection (Figure 1(c)). We term this module the **dominant adaptation module**, whose layer index varies across model families but remains stable

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

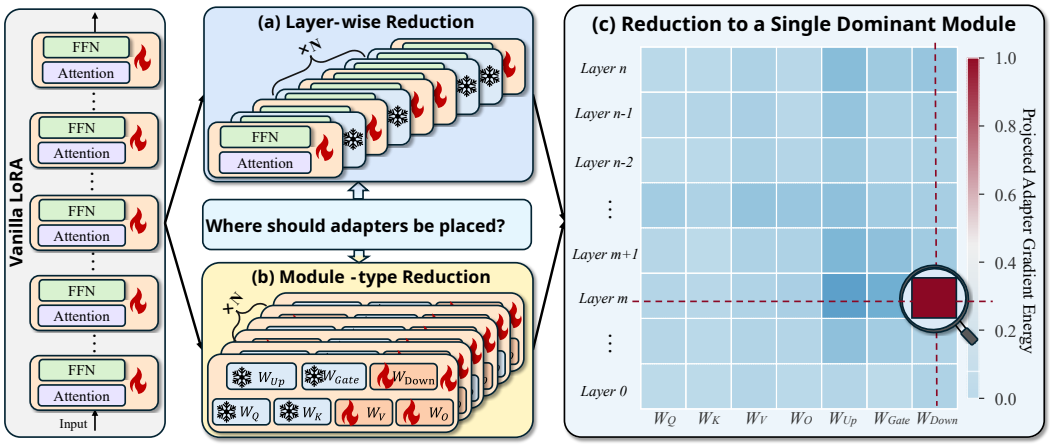


Figure 1: **From broad to dominant placement.** Vanilla LoRA places adapters across many layers and module types. (a) Layer-wise Reduction: fewer layers, all module types retained. (b) Module-type Reduction: fewer module types, all layers retained. (c) Reduction to a Single Dominant Module: PAGE concentrates at one shallow FFN down-projection (magnified), where a single adapter suffices.

across diverse downstream tasks on the same backbone, suggesting that it reflects an intrinsic structural property rather than a task-specific effect.

Guided by this finding, we propose **DomLoRA** (**Dom**inant adaptation Module **LoRA**), a LoRA placement method that applies LoRA only to this module while freezing all other parameters. We evaluate DomLoRA on Qwen3-8B (Yang et al., 2025) and LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024) across various downstream tasks, including instruction following, mathematical reasoning, code generation, and multi-turn conversation. With only **0.7%** of vanilla LoRA's trainable parameters, DomLoRA remains competitive with vanilla LoRA and outperforms it on average. Moreover, DomLoRA can be applied as a plug-and-play method to other LoRA variants, matching or improving their average performance as well (e.g., LoRA+ (Hayou et al., 2024) and DoRA (Liu et al., 2024)), while reducing training time by up to **2.74**$\times$ compared with broad placement. These results show that DomLoRA is effective across different backbones, task domains, and LoRA variants, supporting the dominant module placement as a practical guideline. Key contributions are summarized as follows:

- We introduce **PAGE**, a gradient-based sensitivity probe that estimates each candidate LoRA adapter's initial trainable gradient energy from empirical Fisher sensitivity. Using PAGE, we identify the **dominant adaptation module**, a single shallow FFN down-projection where PAGE is highly concentrated.
- We show that this module is **architecture-dependent but task-stable**: its layer index varies across model families but remains stable across downstream tasks within the same backbone.
- We propose **DomLoRA**, a LoRA placement method that applies one low-rank adapter only to the dominant adaptation module while freezing all other parameters.
- We validate DomLoRA across two model families and four task domains, showing that it outperforms vanilla LoRA on average using only **0.7%** of its trainable parameters. We further show that the same method matches or improves other LoRA variants while reducing their training time by up to **2.74**$\times$.

2 Related Work

Parameter-efficient fine-tuning (PEFT). PEFT adapts pre-trained models by updating a small parameter subset while keeping the backbone frozen. Representative methods include adapter tuning (Houlsby et al., 2019), prefix tuning (Li & Liang, 2021), prompt tuning (Lester et al., 2021), and LoRA (Hu et al., 2022). Later LoRA variants improve adaptive rank allocation (Zhang et al., 2023), weight decomposition (Liu et al., 2024), optimization with asymmetric learning rates (Hayou et al., 2024), and structured low-dimensional parameterization (Kopiczko et al., 2024). These methods

mainly improve how adapters are parameterized or optimized, whereas we focus on where adapters should be placed.

Reducing LoRA adapter placement. Vanilla LoRA commonly places adapters into attention and FFN projections in most or all Transformer blocks, but recent work shows that fewer adapters may still preserve or even improve performance.

One line reduces the number of adapted layers: IST (Yao et al., 2024) scores layer importance and updates only top-ranked layers, while depth-aware and similarity-based methods distinguish shallow from deep layers or identify task-critical layers (Gu et al., 2024; Ogawa et al., 2026).

Another line reduces the adapted module types: MLP-only LoRA can match broader adaptation performance, with the up projection particularly effective (Xue & Mirzasoleiman, 2026); restricting LoRA to feed-forward layers can outperform attention-only placement (Kunz, 2025); and PLoP (Hayou et al., 2026) scores module types by neural-feature-norm alignment and adapts only selected projections.

However, these methods still retain multiple adapters across many or all layers, whereas DomLoRA further reduces LoRA placement to a single adapter at the dominant adaptation module, achieving strong performance with far fewer trainable parameters.

FFN layers and knowledge localization. Prior work suggests that FFN layers play a distinctive role in Transformer representations. Feed-forward layers have been characterized as key-value memories (Geva et al., 2021), and knowledge-neuron (Dai et al., 2022) analyses localize factual associations to specific FFN neurons. GPS (Zhang et al., 2024) further shows that task-relevant parameters are sparse and unevenly distributed across the network. These findings support the view that adaptation demand is non-uniform across modules or depths. However, they do not identify which FFN projection and depth position dominate PEFT adaptation. DomLoRA addresses this gap by showing that empirical Fisher sensitivity concentrates at a single shallow FFN down-projection.

3 Discovering and Exploiting the Dominant Adaptation Module

In this section, we develop DomLoRA by identifying where to place LoRA. We first define module sensitivity as the average squared sample-wise gradient of each pretrained projection weight in Section 3.1. We then use this sensitivity to derive PAGE, which estimates the initial gradient energy received by each candidate LoRA adapter. We next show that PAGE is highly concentrated at one shallow FFN down-projection across models and tasks in Section 3.2. Finally, in Section 3.3, we introduce DomLoRA, which places only one single adapter at the dominant adaptation module.

3.1 From Module Sensitivity to PAGE

We use sample-wise gradients of pretrained projection weights as the starting point for LoRA placement. A large gradient norm indicates that the probe loss is sensitive to perturbations of that projection weight. Since a LoRA adapter modifies the same effective weight through a low-rank update, the gradient of the pretrained weight determines how much gradient can be passed to the LoRA factors at initialization. We therefore formalize the placement signal as the initial gradient energy received by the trainable LoRA parameters.

Measuring Module Sensitivity. Let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$ be a probe set of N samples. For sample i, let $\boldsymbol{s}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$ denote the full prompt-response sequence, indexed as $s_{i,1}, \ldots, s_{i,T_i}$, and let $\mathcal{R}_i \subseteq \{1, \ldots, T_i\}$ denote the supervised target-response positions. We write $p_\theta(s_{i,t} \mid \boldsymbol{s}_{i,<t})$ for the next-token probability assigned by the model with parameters θ. The sample-level probe loss is the mean cross-entropy over these positions:

$$\bar{\ell}_i(\theta) = \frac{1}{|\mathcal{R}_i|} \sum_{t \in \mathcal{R}_i} -\log p_\theta(s_{i,t} \mid \boldsymbol{s}_{i,<t}) \,. \tag{1}$$

For module type k in layer l, let $\boldsymbol{W}_{l,k} \in \mathbb{R}^{d_{\mathrm{out},k} \times d_{\mathrm{in},k}}$ denote the pretrained projection weight. We compute the sample-wise gradient with respect to this weight:

$$\boldsymbol{G}_{i,l,k} = \nabla_{\boldsymbol{W}_{l,k}} \bar{\ell}_i(\theta). \tag{2}$$

This gradient is computed on the original pretrained model before LoRA optimization begins, and is used only to measure the sensitivity of module (l, k). Following the empirical Fisher perspective, we summarize the sample-wise gradients by their average squared norm, which corresponds to the trace of the empirical Fisher block for this projection weight.

Definition 1 (Module Sensitivity)**.** *We use the sample-wise gradients in Eq. equation 2 to define the sensitivity of module (l, k) as*

$$\mathcal{S}_{l,k}^{\mathrm{emp}} = \frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{G}_{i,l,k}\|_{\mathrm{F}}^2 . \tag{3}$$

A larger $\mathcal{S}_{l,k}^{\mathrm{emp}}$ means that small changes to the pretrained projection weight $\boldsymbol{W}_{l,k}$ have a larger first-order effect on the probe loss. Thus, it measures the sensitivity of the module's effective weight. However, LoRA does not train $\boldsymbol{W}_{l,k}$ directly; it trains the low-rank factors $\boldsymbol{A}_{l,k}$ and $\boldsymbol{B}_{l,k}$. We next show how this full-weight sensitivity is transferred to the initial gradients of the LoRA factors.

Bridging Module Sensitivity and Initial LoRA Gradients. A LoRA adapter parameterizes the effective weight of module (l, k) as

$$\boldsymbol{W}_{l,k} = \boldsymbol{W}_{l,k}^{0} + s\, \boldsymbol{B}_{l,k} \boldsymbol{A}_{l,k}, \qquad s = \alpha / r, \tag{4}$$

where $\boldsymbol{B}_{l,k} \in \mathbb{R}^{d_{\mathrm{out},k} \times r}$, $\boldsymbol{A}_{l,k} \in \mathbb{R}^{r \times d_{\mathrm{in},k}}$, and only $\boldsymbol{A}_{l,k}$, $\boldsymbol{B}_{l,k}$ are trainable. Since the effective weight depends on the LoRA factors through the low-rank term $s\boldsymbol{B}_{l,k}\boldsymbol{A}_{l,k}$, the full-weight gradient $\boldsymbol{G}_{i,l,k}$ induces gradients on the LoRA factors through the chain rule.

Lemma 1. *Under the standard LoRA setup (Hu et al., 2022) ($\boldsymbol{B}_{l,k} = \boldsymbol{0}$, with $\boldsymbol{A}_{l,k}$ set by Kaiming-uniform (He et al., 2015)), the trainable gradients at initialization are*

$$\nabla_{\boldsymbol{A}_{l,k}} \bar{\ell}_i = \boldsymbol{0}, \qquad \nabla_{\boldsymbol{B}_{l,k}} \bar{\ell}_i = s\, \boldsymbol{G}_{i,l,k}\, \boldsymbol{A}_{l,k}^{\top}, \tag{5}$$

where $\boldsymbol{G}_{i,l,k}$ is the sample-wise gradient defined in Eq. equation 2.

Proof. See Appendix D.2. □

Lemma 1 shows that the initial LoRA gradient is not the full-weight gradient itself, but its projection through the randomly initialized factor $\boldsymbol{A}_{l,k}$. To compare candidate placements, we therefore measure the energy of this projected gradient, averaged over the initialization of $\boldsymbol{A}_{l,k}$, which motivates the following definition of PAGE.

Definition 2 (Projected Adapter Gradient Energy)**.** *For a candidate LoRA adapter at module (l, k), initialized as in Lemma 1, we define the* Projected Adapter Gradient Energy *(PAGE) as the expected squared norm of the nonzero trainable gradient received at initialization:*

$$\mathrm{PAGE}_{l,k} = \mathbb{E}_{\boldsymbol{A}_{l,k}} \left[\frac{1}{N} \sum_{i=1}^{N} \left\| \nabla_{\boldsymbol{B}_{l,k}} \bar{\ell}_i \right\|_{\mathrm{F}}^2 \right] = \mathbb{E}_{\boldsymbol{A}_{l,k}} \left[\frac{1}{N} \sum_{i=1}^{N} \left\| s\, \boldsymbol{G}_{i,l,k} \boldsymbol{A}_{l,k}^{\top} \right\|_{\mathrm{F}}^2 \right] . \tag{6}$$

The expectation in Eq. equation 6 is taken over the random initialization of $\boldsymbol{A}_{l,k}$. Expanding the squared Frobenius norm via the trace, PAGE can be rewritten as

$$\mathrm{PAGE}_{l,k} = \frac{s^2}{N} \sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{A}_{l,k}} \left[\mathrm{tr}\big(\boldsymbol{G}_{i,l,k}^{\top} \boldsymbol{G}_{i,l,k}\, \boldsymbol{A}_{l,k}^{\top} \boldsymbol{A}_{l,k} \big) \right] . \tag{7}$$

Since $\boldsymbol{G}_{i,l,k}$ is computed on the pretrained model before LoRA optimization, it is fixed with respect to this expectation. Therefore, evaluating PAGE reduces to computing $\mathbb{E}_{\boldsymbol{A}_{l,k}}[\boldsymbol{A}_{l,k}^{\top} \boldsymbol{A}_{l,k}]$ under the LoRA initialization. The following theorem gives this expectation and the expression for PAGE.

Theorem 1. *Let the entries of $\boldsymbol{A}_{l,k} \in \mathbb{R}^{r \times d_{\mathrm{in},k}}$ be drawn i.i.d. from $\mathcal{U}\big(-1/\sqrt{d_{\mathrm{in},k}},\ 1/\sqrt{d_{\mathrm{in},k}}\big)$. Then*

$$\mathbb{E}_{\boldsymbol{A}_{l,k}} \big[\boldsymbol{A}_{l,k}^{\top} \boldsymbol{A}_{l,k} \big] = \frac{r}{3\, d_{\mathrm{in},k}}\, \boldsymbol{I}_{d_{\mathrm{in},k}}, \tag{8}$$

where $\boldsymbol{I}_{d_{\mathrm{in},k}}$ is a $d_{\mathrm{in},k} \times d_{\mathrm{in},k}$ identity matrix. Consequently, PAGE can be written as:

$$\mathrm{PAGE}_{l,k} = \frac{s^2\, r}{3\, d_{\mathrm{in},k}}\, \mathcal{S}_{l,k}^{\mathrm{emp}}. \tag{9}$$

Proof. See Appendix D.3. □

Eq. equation 9 makes PAGE a LoRA placement probe: it estimates how much trainable gradient signal a candidate adapter receives at initialization. With shared r and s, PAGE is determined only by two module-dependent quantities: the empirical sensitivity $\mathcal{S}_{l,k}^{\text{emp}}$ and the input dimension $d_{\text{in},k}$.

3.2 Observation: PAGE Reveals a Dominant Adaptation Module

We evaluate PAGE (Eq. equation 9) for every attention and FFN projection in Qwen3-8B (Yang et al., 2025) and LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024) across four fine-tuning datasets: WizardLM-Evol-Instruct (Xu et al., 2024), Tülu V3 (Lambert et al., 2025), NuminaMath-CoT (Li et al., 2024), and KodCode-V1-SFT-4o (Xu et al., 2025). We show WizardLM-Evol-Instruct in the main text and the full results in Appendix B. For each run, we compute PAGE at seven checkpoints: Steps 0, 1, 1250, 2500, 3750, 5000, and 6250 (6237 for Tülu V3), where Step 6250 is the final checkpoint after one epoch.

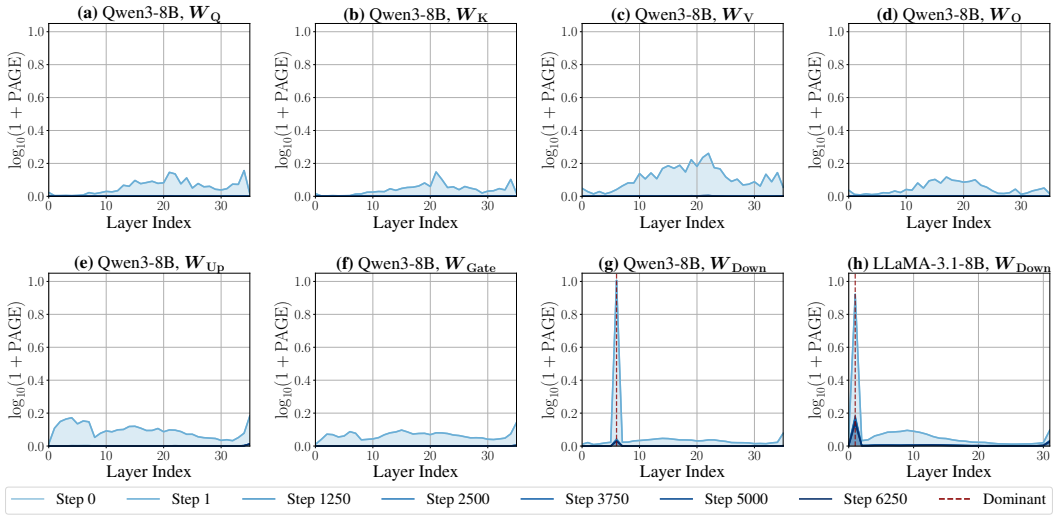


Figure 2: **PAGE across projection modules.** (a)–(g) show all attention and FFN projections of Qwen3-8B, and (h) shows the FFN down-projection of LLaMA-3.1-8B-Instruct. Dashed vertical lines indicate the dominant adaptation module.

PAGE is highly concentrated at one shallow FFN down-projection. As shown in Figure 2, the PAGE values vary substantially across module types and layers. Attention projections and FFN up/gate projections (a)–(f) exhibit relatively small PAGE values that change smoothly across layers.

In contrast, FFN down-projections, (g) for Qwen3-8B and (h) for LLaMA-3.1-8B, exhibit a localized PAGE peak at a shallow layer: Layer 6 in Qwen3-8B and Layer 1 in LLaMA-3.1-8B. We identify this shallow FFN down-projection as the *dominant adaptation module*, and the selection is stable across five data-order seeds under the 32-sample probe (see Appendix I for details). In Appendix C, we show why this module dominates, tracing it to a channel-concentrated down-projection input coinciding with the large error signal at shallow depth.

The identity of the dominant module is a pre-trained structural property. The Layer 6 down-projection of Qwen3-8B is already the top-ranked PAGE candidate at Step 0, before any fine-tuning update, and its ranking does not change at any later checkpoint (Steps 1–6250); the same holds for the Layer 1 down-projection of LLaMA-3.1-8B. The absolute PAGE values decrease as training progresses, since the gradients of all modules shrink as the model fits the data, but this rescaling affects all modules jointly and leaves the ranking intact.

Quantitatively, in Qwen3-8B on WizardLM-Evol-Instruct, the Layer 6 FFN down-projection is only one of 252 candidate projection modules, yet it accounts for 14.7–18.2% of the aggregate PAGE over all projection modules, and for 73.4–79.0% among FFN down-projections alone.

This observation suggests that the dominant adaptation module is already encoded in the pre-trained model and is preserved during fine-tuning rather than created by the fine-tuning process.

3.3 From PAGE to DomLoRA

The PAGE concentration observed in Section 3.2 remains stable across checkpoints and downstream tasks (Appendix B). In both Qwen3-8B and LLaMA-3.1-8B, the dominant PAGE peak consistently appears at a shallow FFN down-projection.

DomLoRA turns this finding into a placement method: it applies a single LoRA adapter at the dominant FFN down-projection selected by Step-0 PAGE, and freezes all remaining parameters. Concretely, we select the layer

$$l^\star = \arg\max_{0 \le l < L} \mathrm{PAGE}_{l,\,\mathrm{down}}, \qquad \boldsymbol{W}_{\mathrm{dom}} = \boldsymbol{W}_{\mathrm{down}}^{(l^\star)}. \tag{10}$$

For the FFN down-projection $\boldsymbol{W}_{\mathrm{down}}^{(l)} \in \mathbb{R}^{d \times d_{\mathrm{ff}}}$, DomLoRA parameterizes the effective weight as

$$\widetilde{\boldsymbol{W}}_{\mathrm{down}}^{(l)} = \boldsymbol{W}_{\mathrm{down}}^{(l)} + \mathbb{I}(l = l^\star)\frac{\alpha}{r}\boldsymbol{B}^\star\boldsymbol{A}^\star, \tag{11}$$

where $\boldsymbol{B}^\star \in \mathbb{R}^{d \times r}$ and $\boldsymbol{A}^\star \in \mathbb{R}^{r \times d_{\mathrm{ff}}}$. Only $\boldsymbol{A}^\star$ and $\boldsymbol{B}^\star$ are trainable; all pretrained weights and all other projection modules remain frozen.

4 Experiments

We evaluate DomLoRA along four main axes: comparison with adapter-placement baselines, generalization to LoRA variants, training-time efficiency, and the necessity of the dominant adaptation module. Specifically, our main experiments are organized around the following research questions:

- **RQ1: How does DomLoRA compare with adapter-placement baselines?** (Section 4.2)
- **RQ2: Does DomLoRA generalize to LoRA variants?** (Section 4.3, Appendix G)
- **RQ3: How much training time does DomLoRA save?** (Section 4.4)
- **RQ4: Is the dominant adaptation module necessary?** (Section 4.5)

Additional analyses in the appendix cover sensitivity to the LoRA rank and learning rate, and PAGE stability across data-order seeds and sample counts:

- **A1: Is DomLoRA sensitive to the LoRA rank and learning rate?** (Appendix H)
- **A2: Is the dominant module identified by PAGE stable across seeds and sample counts?** (Appendix I)

4.1 Experimental Setup

Tasks, datasets, and evaluation. We evaluate DomLoRA on two LLMs: LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024) and Qwen3-8B (Yang et al., 2025). We consider four supervised fine-tuning settings:

- For **general instruction tuning**, we fine-tune each backbone on the 100K subset of Tülu V3 (Lambert et al., 2025) and evaluate them on MMLU (Hendrycks et al., 2021a), TyDiQA (Clark et al., 2020), CommonsenseQA (Talmor et al., 2019), TruthfulQA (Lin et al., 2022), GSM8K (Cobbe et al., 2021), and LogiQA (Liu et al., 2020). We report EM for TyDiQA, and Acc for all other benchmarks.

- For **multi-turn conversational tuning**, we fine-tune LLaMA-3.1-8B-Instruct on 143K examples from WizardLM-Evol-Instruct-V2 (Xu et al., 2024) and evaluate the resulting models on MT-Bench (Zheng et al., 2023). GPT-5.1 serves as the judge under the score-based single-answer grading protocol, and we use the average score over both conversation turns as the evaluation metric.
- For **mathematical reasoning**, we fine-tune LLaMA-3.1-8B-Instruct on the 100K subset of NuminaMath-CoT (Li et al., 2024) and evaluate the resulting models on GSM8K (Cobbe et al., 2021) and MATH-500 (Lightman et al., 2023), using EM as the evaluation metric.
- For **code generation**, we fine-tune LLaMA-3.1-8B-Instruct on the 100K subset of KodCode-V1-SFT-4o (Xu et al., 2025) and evaluate the resulting models on HumanEval (Chen et al., 2021) and HumanEval+ (Liu et al., 2023), using pass@1 as the evaluation metric.

For reproducibility, we evaluate all benchmarks in a zero-shot setting with greedy decoding. Since MT-Bench is scored on a 1–10 scale, we report MT-Bench scores multiplied by 10.

Training details. We use LoRA rank 64 and learning rate 2e-5 with 3% linear warmup followed by cosine decay, and training seed 42 unless otherwise specified. Additional details are in Appendix E.

4.2 Comparing DomLoRA with Adapter-Placement Baselines (RQ1)

We compare DomLoRA with three adapter-placement baselines: vanilla LoRA (Hu et al., 2022) that places adapters broadly across layers and projections, IST (Yao et al., 2024) that reduces adapted layers, and PLoP (Hayou et al., 2026) that reduces adapted module types.

4.2.1 General Instruction Tuning

Table 1 compares DomLoRA with adapter-placement baselines on general instruction tuning, where the LoRA and DomLoRA rows are averaged over three training seeds (per-seed results in Appendix F). Despite using only **0.7%** of vanilla LoRA's trainable parameters (1.2M on LLaMA-3.1-8B and 1.0M on Qwen3-8B), DomLoRA achieves the best average score on both backbones. On LLaMA-3.1-8B, it improves the average from 58.2 to 61.8 over vanilla LoRA. On Qwen3-8B, it also improves the average over vanilla LoRA. Although DomLoRA is not the best on every metric, its strongest average performance suggests that broad adapter placement is not necessary for general instruction tuning.

Table 1: General instruction tuning against adapter-placement baselines. Shaded rows indicate DomLoRA, and **bold** marks the best result within each model group.

Model	Method	Params	MMLU	TyDiQA	CQA	TruthfulQA	GSM8K	LogiQA	Avg.
Qwen	Pretrained	–	67.2	45.9	78.1	**63.8**	**93.8**	**57.3**	67.7
	LoRA	174.6M	71.1	**51.6**	80.3	53.2	92.0	56.3	67.4
	PLoP	68.4M	70.2	48.9	69.2	55.7	91.7	56.1	65.3
	IST	174.6M	71.3	46.0	**81.7**	56.5	91.6	55.8	67.2
	DomLoRA	**1.0M**	**71.9**	50.5	**81.7**	57.4	92.6	57.2	**68.6**
LLaMA	Pretrained	–	**64.2**	48.5	75.8	50.4	**85.3**	**41.9**	61.0
	LoRA	167.8M	63.4	**52.3**	76.5	41.2	77.9	38.2	58.2
	PLoP	65.0M	63.4	49.1	**77.0**	39.2	80.0	38.9	57.9
	IST	167.8M	63.7	48.0	74.4	45.5	77.1	39.3	58.0
	DomLoRA	**1.2M**	63.4	49.7	75.6	**55.9**	84.9	41.2	**61.8**

4.2.2 Mathematical Reasoning, Code Generation, and Multi-Turn Conversation

Table 2 summarizes mathematical reasoning, code generation, and multi-turn conversation results on LLaMA-3.1-8B-Instruct. DomLoRA improves the average over vanilla LoRA from 62.6 to 64.2 while using only 1.2M trainable parameters. The improvement is mainly driven by GSM8K, where it

increases from 81.5 to 86.4, and MT-Bench, where it increases from 57.3 to 61.3. These results show that dominant-module placement remains competitive beyond general instruction tuning.

Table 2: Reasoning, coding, and conversation results on LLaMA-3.1-8B-Instruct. Shaded rows indicate DomLoRA, and **bold** marks the best result.

Method	Params	GSM8K	MATH500	HumanEval	HumanEval+	MT-Bench	Avg.
Pretrained	–	85.3	43.2	56.7	52.4	61.0	59.7
LoRA	167.8M	81.5	48.4	**65.9**	**59.8**	57.3	62.6
PLoP	65.0M	85.0	**48.6**	64.0	59.1	56.4	62.6
IST	167.8M	85.4	47.2	63.4	58.5	57.6	62.4
DomLoRA	**1.2M**	**86.4**	48.2	**65.9**	59.1	**61.3**	**64.2**

4.3 Generalizing Dominant-Module Placement to LoRA Variants (RQ2)

We further evaluate how dominant module placement affects representative LoRA variants. These variants modify different aspects of the LoRA update, including rank allocation (AdaLoRA (Zhang et al., 2023)), magnitude-direction decomposition (DoRA (Liu et al., 2024)), factor-wise learning rates (LoRA+ (Hayou et al., 2024)), low-rank update denoising (NoRM (Jiang et al., 2025)), and block-wise update expressiveness (GraLoRA (Jung et al., 2025)).

Table 3 reports reasoning, coding, and multi-turn conversation for these variants on LLaMA-3.1-8B-Instruct, where dominant-module placement uses two orders of magnitude fewer parameters. The effect is variant-dependent: LoRA+ improves the most, from 62.5 to 65.8, and GraLoRA from 63.5 to 64.4 despite drops on GSM8K and MATH500; NoRM and AdaLoRA decrease by 0.3 and 0.9 points, respectively, while using only 1.1M and 0.1M trainable parameters. General instruction tuning with these variants is reported in Appendix G.

Table 3: LoRA variant results on reasoning, coding, and conversation tasks (LLaMA-3.1-8B-Instruct). Shaded rows indicate dominant-module placement, and **bold** marks the best result.

Method	Dom	Params	GSM8K	MATH500	HumanEval	HumanEval+	MT-Bench	Avg.
NoRM	✗	130.1M	86.4	**50.6**	**70.6**	**60.8**	57.2	**65.1**
	✓	**1.1M**	**86.8**	45.2	**70.6**	**60.8**	**60.4**	64.8
DoRA	✗	169.9M	84.2	**45.0**	64.0	58.5	**56.8**	61.7
	✓	**1.2M**	**84.7**	43.2	**64.6**	**59.8**	56.7	**61.8**
LoRA+	✗	167.8M	85.5	46.4	65.2	57.9	57.3	62.5
	✓	**1.2M**	**87.0**	**48.2**	**70.1**	**64.0**	**59.8**	**65.8**
AdaLoRA	✗	19.4M	86.1	46.0	**63.4**	**58.5**	**61.1**	**63.0**
	✓	**0.1M**	**87.0**	**46.8**	59.1	56.7	60.7	62.1
GraLoRA	✗	167.8M	**86.3**	**49.8**	66.5	57.9	57.2	63.5
	✓	**1.2M**	85.6	45.6	**67.1**	**62.2**	**61.7**	**64.4**

4.4 Training-Time Efficiency (RQ3)

We compare the full training time of dominant-module placement against broad placement for general instruction tuning on Tülu V3 (Lambert et al., 2025). All runs follow the default configuration in Appendix E and train for one epoch on a single H100 GPU. Training time is the wall-clock time from the start of training to the final checkpoint, excluding evaluation.

Table 4 shows that dominant-module placement reduces training time for every LoRA variant, with the largest reduction for DoRA, whose average training time decreases from 7h 49m to 2h 51m ($2.74\times$). The gain is larger for variants with heavier adapter updates, since restricting placement to one module removes most of the adapter-update overhead.

Table 4: Training time (wall-clock, one epoch). Shaded rows indicate dominant-module placement, and **bold** marks the shorter time in each pair.

Method	Dom	LLaMA-3.1-8B	Qwen3-8B	Avg.
LoRA	✗	4h 23m	4h 34m	4h 29m
	✓	**2h 48m**	**2h 49m**	**2h 48m**
DoRA	✗	7h 50m	7h 48m	7h 49m
	✓	**2h 51m**	**2h 52m**	**2h 51m**
LoRA+	✗	4h 14m	4h 39m	4h 27m
	✓	**2h 47m**	**2h 51m**	**2h 49m**
AdaLoRA	✗	6h 16m	6h 03m	6h 10m
	✓	**2h 43m**	**2h 50m**	**2h 46m**
GraLoRA	✗	4h 33m	5h 32m	5h 03m
	✓	**2h 18m**	**2h 52m**	**2h 35m**

4.5 Ablation Study (RQ4)

Since LLaMA-3.1-8B-Instruct has 224 candidate projection modules, fine-tuning each of them is prohibitively expensive. We therefore compare placements spanning high, medium, and low PAGE scores, together with attention-only and FFN-only adaptation. Table 5 shows that the average score decreases monotonically from 63.0 to 55.8 as the PAGE score drops from 1.00 to 0.11, and that the single dominant module outperforms both attention-only (60.2) and FFN-only (60.1) adaptation, indicating that the gain of DomLoRA comes from where the adapter is placed rather than how widely.

Table 5: Ablation study of adaptation target selection on LLaMA-3.1-8B-Instruct. Shaded row indicates DomLoRA, and **bold** marks the best reported score in each column.

Module	Layer	PAGE	MMLU	TyDiQA	CQA	TruthfulQA	GSM8K	MATH500	HumanEval+	Avg.
$\boldsymbol{W}_{\text{down}}$	1	1.00	**64.1**	49.4	76.9	**56.9**	86.4	**48.2**	59.1	**63.0**
$\boldsymbol{W}_{\text{v}}$	0	0.58	63.9	49.6	76.2	49.3	86.4	45.8	56.1	61.0
$\boldsymbol{W}_{\text{v}}$	1	0.52	62.4	50.2	74.8	49.7	**86.5**	45.0	48.8	59.6
$\boldsymbol{W}_{\text{v}}$	9	0.11	59.3	40.6	74.4	43.1	85.6	47.4	51.2	57.4
$\boldsymbol{W}_{\text{o}}$	11	0.11	58.4	35.8	70.1	41.3	83.8	45.4	56.1	55.8
Attention	All	–	63.0	46.0	75.3	44.2	85.5	47.4	**59.8**	60.2
FFN	All	–	63.7	**50.6**	**77.1**	39.7	84.4	47.4	57.9	60.1

5 Conclusion

We introduced PAGE, a gradient-based sensitivity probe that reveals adaptation sensitivity highly concentrating at a single shallow FFN down-projection, which we call the dominant adaptation module. DomLoRA places one LoRA adapter at this module and freezes other parameters. Across two model families and four task domains, it outperforms vanilla LoRA on average with only 0.7% of its trainable parameters, while training up to 2.74× faster than broad placement. It also matches or improves representative LoRA variants, supporting DomLoRA as a plug-and-play placement method.

Limitations. DomLoRA requires an additional PAGE probe before training to identify the dominant adaptation module. Our experiments focus on dense Transformers, extending DomLoRA to MoE models, VLM models, or other architectures is left for future work.

AI USE STATEMENT

We used generative AI tools to polish the writing and improve the readability of this paper, with all AI-assisted text reviewed by the authors. In addition, as described in Section 4.1, we used GPT-5.1 as an automatic judge for MT-Bench evaluation, following the standard LLM-as-a-judge protocol. Apart from these uses, we did not use generative AI tools for research ideation, theoretical derivations, methodology design, or data analysis. We take responsibility for the final content of this work.

REFERENCES

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Systematic outliers in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=rLX7Vyyzus.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020. doi: 10.1162/tacl_a_00317. URL https://aclanthology.org/2020.tacl-1.30/.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.581. URL https://aclanthology.org/2022.acl-long.581/.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1feb87871436031bdc0f2beaa62a049b-Paper-Conference.pdf.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru,

Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik

Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Jihao Gu, Zelin Wang, Yibo Zhang, Ziji Zhang, and Ping Gong. From bottom to top: Extending the potential of parameter efficient fine-tuning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3488–3500, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.204. URL https://aclanthology.org/2024.emnlp-main.204/.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17783–17806. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/hayou24a.html.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. PLoP: Precise LoRA placement for efficient finetuning of large models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3lGkVgNZ5a.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=0RDcd5Axok.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021b. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/be83ab3ecd0db773eb2dc1b0a17836a1-Paper-round2.pdf.

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 Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, June 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

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*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Shuyang Jiang, Yusheng Liao, Ya Zhang, Yanfeng Wang, and Yu Wang. Fine-tuning with reserved majority for noise reduction. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ZV7CLf0RHK.

Yeonjoon Jung, Daehyun Ahn, Hyungjun Kim, Taesu Kim, and Eunhyeok Park. GraLoRA: Granular low-rank adaptation for parameter-efficient fine-tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=8wvOMQ2Olw.

Dawid J. Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. VeRA: Vector-based random matrix adaptation. In *Proceedings of the 2024 International Conference on Learning Representations (ICLR)*, 2024.

Jenny Kunz. Train more parameters but mind their placement: Insights into language adaptation with PEFT. In Richard Johansson and Sara Stymne (eds.), *Proceedings of the Joint 25th Nordic Conference on Computational Linguistics and 11th Baltic Conference on Human Language Technologies (NoDaLiDa/Baltic-HLT 2025)*, pp. 323–330, Tallinn, Estonia, March 2025. University of Tartu Library. ISBN 978-9908-53-109-0. URL https://aclanthology.org/2025.nodalida-1.35/.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243/.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. NuminaMath. https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf, 2024.

Pengxiang Li, Lu Yin, and Shiwei Liu. Mix-LN: Unleashing the power of deeper layers by combining pre-LN and post-LN. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=BChpQU64RG.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *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)*, pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229/.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. LogiQA: A challenge dataset for machine reading comprehension with logical reasoning, 7 2020. URL https://doi.org/10.24963/ijcai.2020/501. Main track.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 21558–21572. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/43e9d647ccd3e4b7b5baab53f0368686-Paper-Conference.pdf.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, July 2024. URL https://proceedings.mlr.press/v235/liu24bn.html.

Keith Ando Ogawa, Bruno Lopes Yamamoto, Lucas Lauton de Alcantara, Lucas Pellicer, Rosimeire Pereira Costa, Edson Bollis, Anna Helena Reali Costa, and Artur Jordao. Layer-wise LoRA fine-tuning: A similarity metric approach, 2026. URL https://arxiv.org/abs/2602.05988.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=F7aAhfitX6.

Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=Orpf8yDjdj.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=CfXh93NDgH.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. KodCode: A diverse, challenging, and verifiable synthetic dataset for coding. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 6980–7008, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.365. URL https://aclanthology.org/2025.findings-acl.365/.

Yihao Xue and Baharan Mirzasoleiman. LoRA is all you need for safety alignment of reasoning LLMs, 2026. URL https://arxiv.org/abs/2507.17075.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Kai Yao, Penglei Gao, Lichun Li, Yuan Zhao, Xiaofeng Wang, Wei Wang, and Jianke Zhu. Layer-wise importance matters: Less memory for better performance in parameter-efficient fine-tuning of large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1977–1992, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.109. URL https://aclanthology.org/2024.findings-emnlp.109/.

Mengxia Yu, De Wang, Qi Shan, Colorado J Reed, and Alvin Wan. The super weight in large language models, 2025. URL https://arxiv.org/abs/2411.07191.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=lq62uWRJjiY.

Zhi Zhang, Qizhe Zhang, Zijun Gao, Renrui Zhang, Ekaterina Shutova, Shiji Zhou, and Shanghang Zhang. Gradient-based parameter selection for efficient fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28566–28577, June 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot Arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91f18a1287b398d378ef22505bf41832-Paper-Datasets_and_Benchmarks.pdf.

A Appendix

The appendix provides additional theoretical details, empirical evidence, and experimental results.

- Appendix B reports additional PAGE maps across datasets and backbones.
- Appendix C explains why PAGE peaks at a shallow FFN down-projection.
- Appendix D provides detailed derivations for PAGE and its connection to full-weight empirical Fisher.
- Appendix E lists training hyperparameters, variant-specific settings, and evaluation prompts.

- Appendix F reports multi-seed results of LoRA and DomLoRA on general instruction tuning.
- Appendix G reports general instruction tuning results for LoRA variants.
- Appendix H reports the sensitivity analysis of DomLoRA to the LoRA rank, scaling factor, and learning rate.
- Appendix I examines the stability of the dominant module identified by PAGE across data-order seeds and sample counts.

B Additional PAGE Evidence

This section provides the complete PAGE maps across datasets and backbones. Step 0 is computed on the pretrained backbone, while the later checkpoint curves are diagnostic re-probes on frozen checkpoints under the same PAGE protocol rather than gradients of the trained adapter factors.

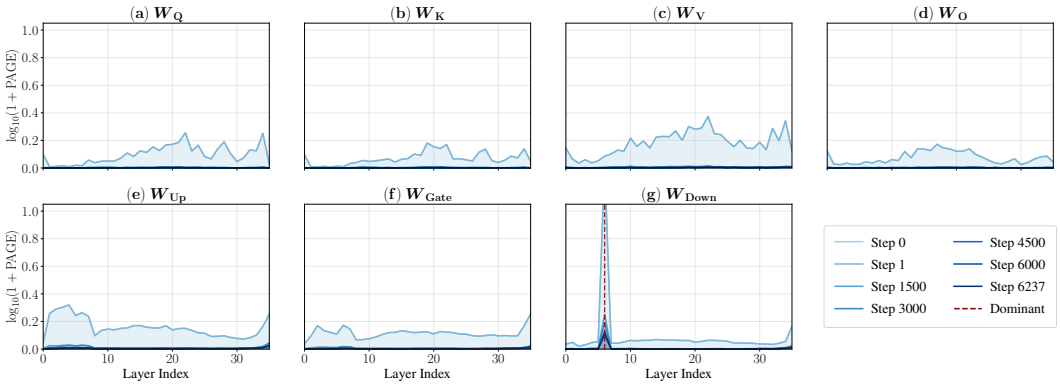


Figure 3: PAGE of all projection modules in Qwen3-8B on Tülu V3.

Figure 3 shows that, for Qwen3-8B on Tülu V3, PAGE is sharply concentrated at the Layer 6 FFN down-projection. In particular, the W_{down} panel in Figure 3(g) exhibits a localized peak aligned with the dashed vertical line, while the attention projections and FFN up/gate projections in Figure 3(a)–(f) remain substantially lower or more diffuse. The peak is already visible at Step 0 and remains aligned across later checkpoints, suggesting that the ranking is already visible at initialization and remains stable under the reported checkpoint re-probes.

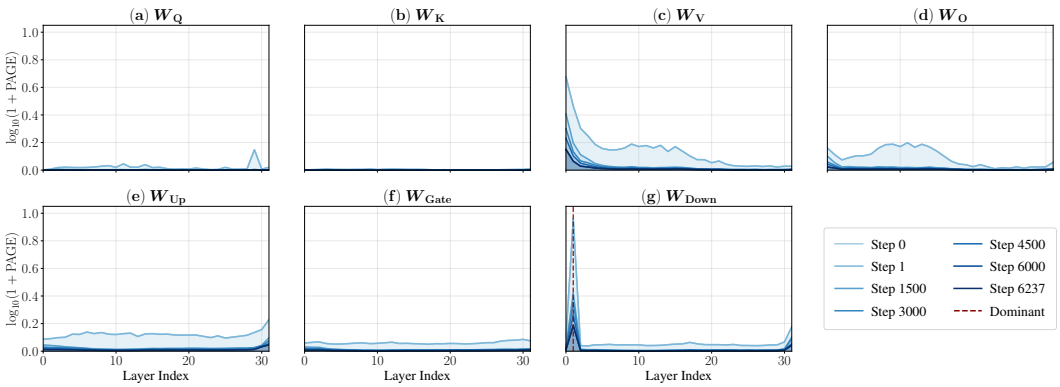


Figure 4: PAGE of all projection modules in LLaMA-3.1-8B-Instruct on Tülu V3.

Figure 4 reveals a different dominant layer for LLaMA-3.1-8B-Instruct: the strongest PAGE peak appears at the Layer 1 FFN down-projection. As shown in Figure 4(g), this peak is highly localized, whereas several attention projections in Figure 4(c)–(d) show broader elevated values over shallow layers. These attention elevations are less localized and less stable than the down-projection peak, supporting the use of the FFN down-projection as the dominant placement target.

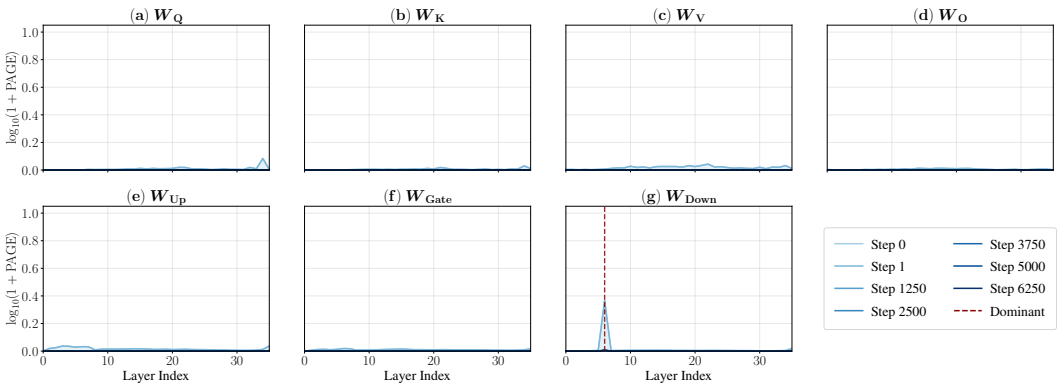


Figure 5: PAGE of all projection modules in Qwen3-8B on NuminaMath-CoT.

Figure 5 demonstrates that the dominant PAGE peak for Qwen3-8B on NuminaMath-CoT again occurs at the Layer 6 FFN down-projection. The W_{down} panel in Figure 5(g) matches the dominant layer observed on Tülu V3, while the other projections in Figure 5(a)–(f) have much smaller PAGE values. This indicates that the same Qwen3-8B module is selected even when the probe data shifts from instruction-following to mathematical reasoning.

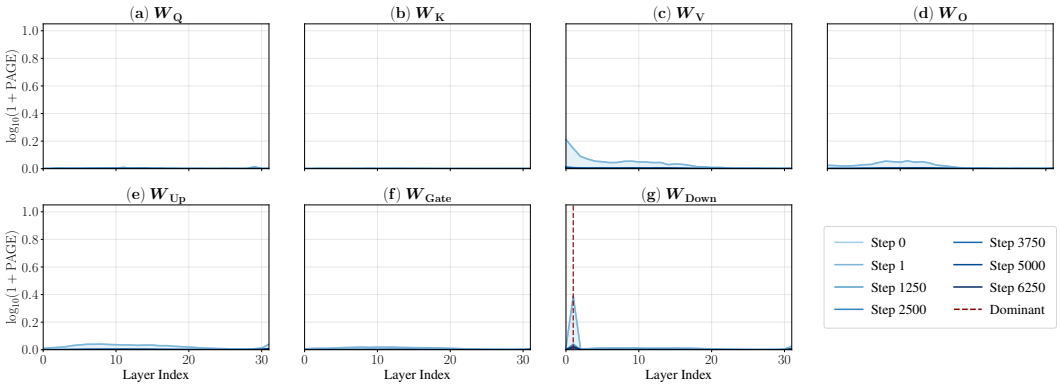


Figure 6: PAGE of all projection modules in LLaMA-3.1-8B-Instruct on NuminaMath-CoT.

Figure 6 reports that LLaMA-3.1-8B-Instruct again selects the Layer 1 FFN down-projection on NuminaMath-CoT. Compared with the Tülu V3 setting, the non-FFN projections in Figure 6(a)–(f) are weaker and smoother, making the down-projection peak in Figure 6(g) more isolated. The repeated selection of Layer 1 suggests that the dominant module is stable for this backbone rather than tied to a particular data distribution.

Figure 7 shows that Qwen3-8B on KodCode still has its largest PAGE peak at the Layer 6 FFN down-projection. In Figure 7(g), the peak is narrow and clearly separated from the rest of the layer-wise curve. By contrast, the attention projections and FFN up/gate projections in Figure 7(a)–(f) do not form a comparable localized maximum. Thus, the same dominant module is recovered on code-generation data.

Figure 8 reveals that LLaMA-3.1-8B-Instruct on KodCode again concentrates PAGE at the Layer 1 FFN down-projection. Although shallow attention projections in Figure 8(c)–(d) exhibit elevated values at early checkpoints, these signals are broader and decay over training. In contrast, the W_{down} peak in Figure 8(g) remains the most localized and stable signal, making it the consistent placement target for this backbone.

Figure 9 demonstrates that the strongest PAGE peak for Qwen3-8B on WizardLM again appears at the Layer 6 FFN down-projection. The W_{down} panel in Figure 9(g) contains a clear localized peak, while the remaining projections in Figure 9(a)–(f) show only moderate and smooth layer-wise

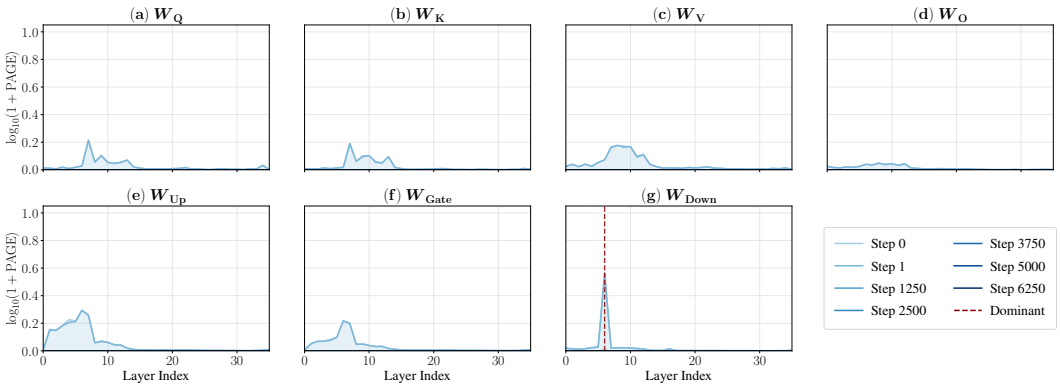


Figure 7: PAGE of all projection modules in Qwen3-8B on KodCode.

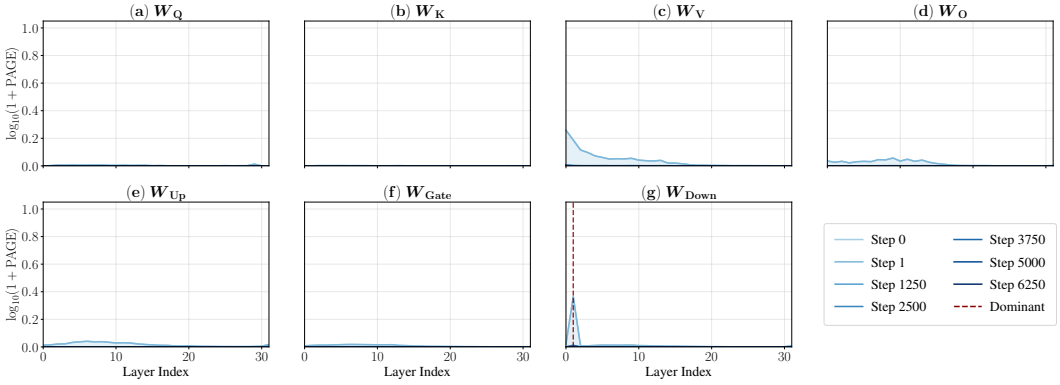


Figure 8: PAGE of all projection modules in LLaMA-3.1-8B-Instruct on KodCode.

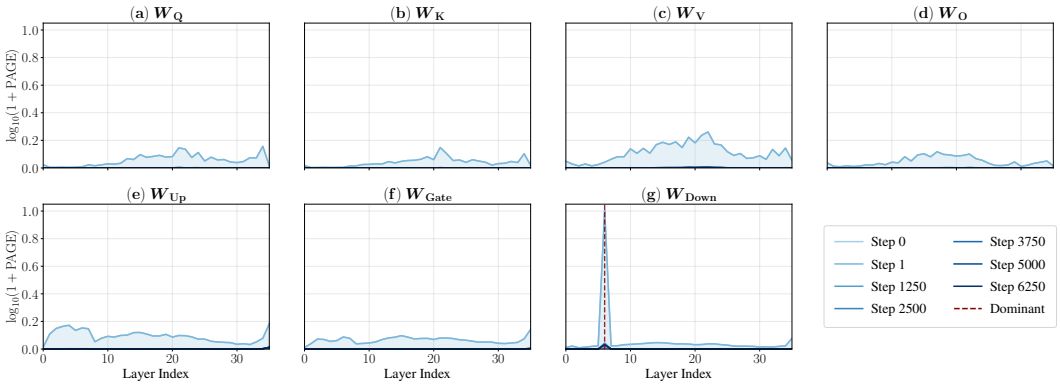


Figure 9: PAGE of all projection modules in Qwen3-8B on WizardLM.

variations. This confirms that Qwen3-8B consistently selects the same dominant module across the evaluated datasets.

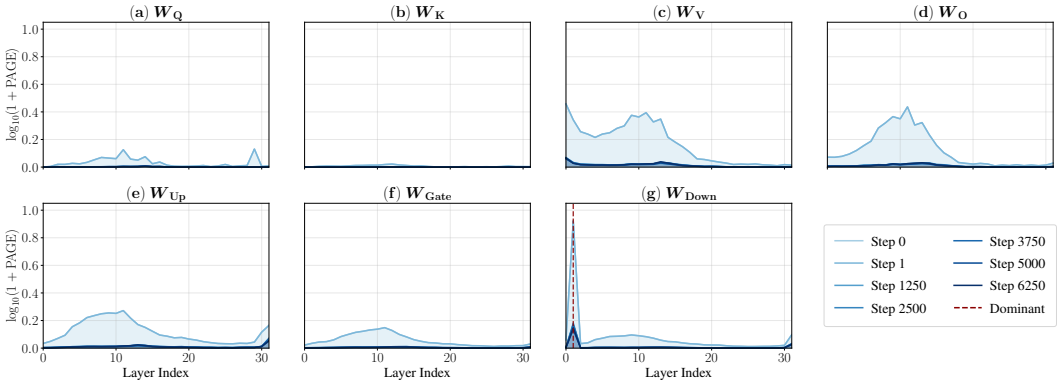


Figure 10: PAGE of all projection modules in LLaMA-3.1-8B-Instruct on WizardLM.

Figure 10 reports that PAGE for LLaMA-3.1-8B-Instruct on WizardLM is again concentrated at the Layer 1 FFN down-projection. The attention and FFN up/gate projections in Figure 10(a)–(f) show only small secondary variations and do not approach the peak in Figure 10(g). Together with the other LLaMA figures, this indicates that the dominant module depends on the backbone but remains stable across downstream datasets.

C Why PAGE Peaks at a Shallow FFN Down-Projection

Section 3.2 identified the dominant module as a shallow FFN down-projection, and this section examines two measured factors associated with the observed peak at Step 0. First, the input to the down-projection concentrates an exceptional amount of energy in a single channel. Second, the backward error signal decays sharply with depth, so it remains large only in shallow layers. We then verify that the ranking reflects which channels the sensitivity flows through, not the raw magnitude of these channels.

The dominant module's input is concentrated in a single channel. A projection's per-sample weight gradient is an outer product summed over positions, $\boldsymbol{G} = \boldsymbol{\Delta}^\top \boldsymbol{X}$, so its energy decomposes exactly across input channels, $\|\boldsymbol{G}\|_{\mathrm{F}}^2 = \sum_j \|[\boldsymbol{G}]_{:,j}\|_2^2$. By Cauchy–Schwarz, each channel satisfies $\|[\boldsymbol{G}]_{:,j}\|_2^2 \le D\,\|[\boldsymbol{X}]_{:,j}\|_2^2$, where $D = \|\boldsymbol{\Delta}\|_{\mathrm{F}}^2$ is the output-gradient energy. Thus, for a fixed output-gradient energy, low input energy limits a channel's possible contribution to the weight gradient. We compare input-energy and gradient-energy concentration directly in the measured modules, without using PAGE.

The dominant module is extreme on both measures. A single input channel holds 92% (Qwen3-8B, $\boldsymbol{W}_{\mathrm{down}}$@6) and 93% (LLaMA-3.1-8B-Instruct, $\boldsymbol{W}_{\mathrm{down}}$@1) of the input energy, and its top 32 channels hold 99.9% and 97.3%, respectively (Figure 11b). Across layers, the down-projection type reaches the largest single-channel input energy of any projection type—up to $250\times$ (Qwen3-8B) and $27\times$ (LLaMA-3.1-8B-Instruct) the strongest competitor (Figure 11a). At the dominant module, the gradient concentrates on the very channel that carries this forward outlier: its single largest input channel also carries 91% (Qwen3-8B) and 97% (LLaMA-3.1-8B-Instruct) of the module's gradient energy, far above typical modules (for example, 37% for $\boldsymbol{W}_{\mathrm{v}}$@22 in Qwen3-8B and 30% for $\boldsymbol{W}_{\mathrm{v}}$@0 in LLaMA-3.1-8B-Instruct). This structure likely follows from the down-projection's role: it is the projection that reads the gated SwiGLU product and writes directly to the residual stream. Prior work has documented large-magnitude activation and weight outliers at early down-projection inputs (Sun et al., 2024; An et al., 2025; Yu et al., 2025), but only as a forward-pass or inference phenomenon. We connect this location to fine-tuning: adaptation sensitivity concentrates on the very channel that carries the forward outlier, which makes it a single dominant module for LoRA placement.

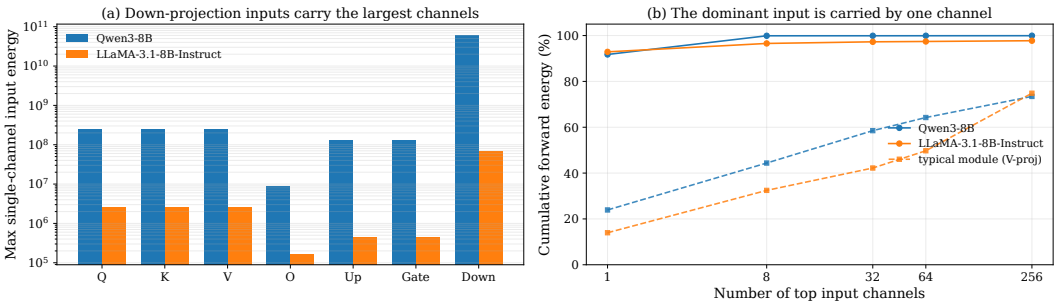


Figure 11: **Down-projection inputs carry exceptional single-channel energy.** (a) Largest single-channel input energy across layers, per module type (log scale); the down-projection type exceeds the strongest competing type by 250× (Qwen3-8B) and 27× (LLaMA-3.1-8B-Instruct). (b) Cumulative forward energy versus number of top input channels, for the dominant down-projection (solid) and a typical V-projection (dashed).

The backward error signal is large only at shallow depth. We measure the residual-stream error signal $E_l = \mathbb{E}_i \|\partial \bar{\ell}_i / \partial \boldsymbol{r}_l\|_2^2$ at each layer, again at Step 0. For the down-projection this is exactly the output-gradient energy D from above, because the down-projection writes straight into the residual stream (other projections differ by intervening Jacobians). From the shallowest to the deepest layer, the signal drops by 3,735× in Qwen3-8B and 940× in LLaMA-3.1-8B-Instruct (Figure 12). A gradient energy such as PAGE necessarily contains this factor; what is empirical here is that the factor stays large only at shallow depth, matching reports of vanishing gradients in deep Pre-LN Transformers (Sun et al., 2025; Li et al., 2025).

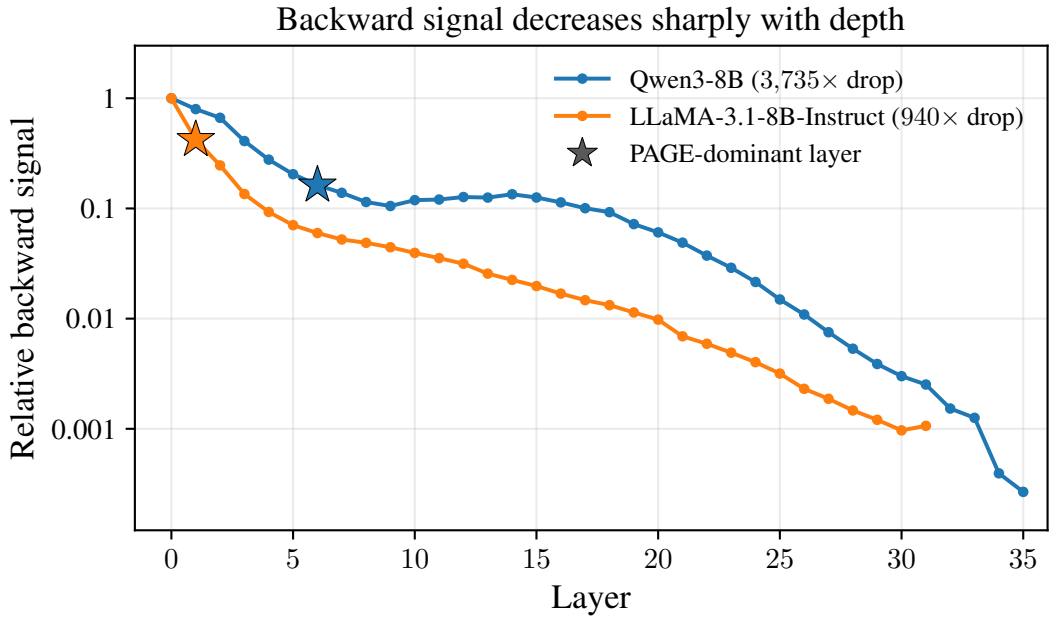


Figure 12: **The backward error signal decays with depth.** Relative residual-stream backward signal E_l/E_0 per layer (pretrained backbones, Step 0); the total drop from the shallowest to the deepest layer is given in the legend. Stars mark the PAGE-dominant layers—layer 6 (Qwen3-8B) and layer 1 (LLaMA-3.1-8B-Instruct)—each shallow enough to still receive a large backward signal.

The two properties select the dominant module jointly, and the near-misses show that neither suffices alone. The deep $\boldsymbol{W}_{\text{down}}$@35 in Qwen3-8B has a comparably concentrated input (top-32 forward energy 0.97, against 1.00 at layer 6) and its largest channel carries about 17× more input energy, but its backward signal is about 600× weaker and its gradient no longer follows the dominant input

channel, leaving its PAGE roughly $38\times$ below layer 6's—a small bump, not the peak. At the other end, $\boldsymbol{W}_{\text{down}}@0$ sits at the largest backward signal with a comparably concentrated input (top-32 energy 0.96), yet its dominant channel carries $274\times$ less energy than layer 6's and its PAGE is $335\times$ lower. The up/gate projections at layer 6 fail on the input side instead: they share the depth, but their largest channel carries $88\times$ less energy. The same joint requirement holds in LLaMA-3.1-8B-Instruct.

PAGE ranking under channel attenuation. The dominance does not come from a few large-magnitude channels. For the mild attenuation settings tested, scaling the 32 dominant input channels by α preserves the top rank with small changes in probe loss. Stronger attenuation increases the loss and can change the ranking, especially for LLaMA-3.1-8B-Instruct. The rank moves only when the loss does ($\alpha \leq 0.1$ for LLaMA-3.1-8B-Instruct, $\alpha = 0$ for Qwen3-8B), that is, only once attenuation is strong enough to damage the model—at which point the ranking change conflates attenuation with damage, since removal also reroutes gradient to the block's up and gate projections. Scaling 32 random channels, or another module's top channels, leaves the ranking unchanged (Table 6).

Table 6: Channel-scaling test: PAGE rank of the dominant module and the probe-set loss (mean cross-entropy on the 32 probe examples) as its 32 largest input channels are scaled by α. The rank stays at 1 for the mild settings tested; stronger attenuation increases the loss and eventually changes the ranking, with full removal ($\alpha = 0$) producing the strongest change.

Backbone	α	**PAGE rank**	**Probe loss**
Qwen3-8B	1.0	1	1.549
	0.5	1	1.544
	0.25	1	1.541
	0.1	1	1.531
	0.05	1	1.536
	0 (removed)	7	1.981
LLaMA-3.1-8B-Instruct	1.0	1	0.984
	0.5	1	0.981
	0.25	1	0.965
	0.1	4	1.150
	0.05	4	3.137
	0 (removed)	37	3.534

In short, the observed PAGE peak coincides with two measured properties: highly concentrated input-channel sensitivity at the down-projection and a strong residual-stream error signal at shallow depth. We do not explain why the dominant adaptation module sits at Layer 6 in Qwen3-8B and Layer 1 in LLaMA-3.1-8B-Instruct; this is a property of the pretrained model that we measure but do not derive.

D Detailed Derivations for PAGE

This appendix provides the supporting derivations for Section 3.1. We first give an empirical Fisher interpretation of the module sensitivity. We then prove Lemma 1. Finally, we derive the PAGE expression in Theorem 1, including the moment identity in Eq. equation 8 and the relationship between PAGE and module sensitivity.

Throughout this appendix, for matrices $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{m\times n}$, we use the Frobenius inner product

$$\langle \boldsymbol{X}, \boldsymbol{Y} \rangle_{\text{F}} = \operatorname{tr}\big(\boldsymbol{X}^\top \boldsymbol{Y}\big)\,,$$

and for a matrix $\boldsymbol{M} \in \mathbb{R}^{m\times n}$, we use the Frobenius norm

$$\|\boldsymbol{M}\|_{\text{F}}^2 = \operatorname{tr}\big(\boldsymbol{M}^\top \boldsymbol{M}\big) = \sum_{a=1}^{m}\sum_{b=1}^{n} M_{ab}^2.$$

All gradients are evaluated at the pretrained model before LoRA optimization begins. Unless otherwise specified, expectations over $\boldsymbol{A}_{l,k}$ are taken with respect to its random initialization.

D.1 Empirical Fisher Interpretation of Module Sensitivity

In Definition 1, the module sensitivity $\mathcal{S}_{l,k}^{\mathrm{emp}}$ is defined in Eq. equation 3 as the average squared Frobenius norm of the sample-wise gradients in Eq. equation 2. We now show that this quantity is exactly the trace of the empirical Fisher block for the pretrained projection weight $\boldsymbol{W}_{l,k}$.

Recall from Eq. equation 2 that, for sample i and module (l, k), the sample-wise gradient with respect to the pretrained projection weight is

$$\boldsymbol{G}_{i,l,k} = \nabla_{\boldsymbol{W}_{l,k}} \bar{\ell}_i(\theta), \qquad \boldsymbol{W}_{l,k} \in \mathbb{R}^{d_{\mathrm{out},k} \times d_{\mathrm{in},k}}. \tag{12}$$

Here, l indexes the Transformer layer, k indexes the projection type, and $\boldsymbol{G}_{i,l,k}$ has the same shape as $\boldsymbol{W}_{l,k}$. Eq. equation 12 gives the first-order response of the sample loss $\bar{\ell}_i(\theta)$ to perturbations of this projection weight.

To write the empirical Fisher block, we need a parameter vector rather than a matrix-shaped gradient. We therefore vectorize the matrix gradient in Eq. equation 12. Let $\mathrm{vec}(\cdot)$ denote column-wise vectorization. We define

$$\boldsymbol{g}_{i,l,k} = \mathrm{vec}(\boldsymbol{G}_{i,l,k}) \in \mathbb{R}^{d_{\mathrm{out},k} d_{\mathrm{in},k}}. \tag{13}$$

This vector contains all entries of $\boldsymbol{G}_{i,l,k}$ and represents the sample-wise gradient in the parameter space of $\boldsymbol{W}_{l,k}$.

Using the vectorized gradients in Eq. equation 13, we define the empirical Fisher block for module (l, k) as

$$\boldsymbol{F}_{l,k}^{\mathrm{emp}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{g}_{i,l,k} \boldsymbol{g}_{i,l,k}^{\top}. \tag{14}$$

Eq. equation 14 is the empirical Fisher block associated with the pretrained projection weight $\boldsymbol{W}_{l,k}$. It is the average outer product of sample-wise gradients and records the gradient energy along directions in the parameter space of this projection weight.

To compare different modules with a scalar quantity, we take the trace of Eq. equation 14. This gives

$$\begin{aligned} \mathrm{tr}\Big(\boldsymbol{F}_{l,k}^{\mathrm{emp}}\Big) &= \mathrm{tr}\left(\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{g}_{i,l,k} \boldsymbol{g}_{i,l,k}^{\top}\right) \\ &= \frac{1}{N} \sum_{i=1}^{N} \mathrm{tr}\big(\boldsymbol{g}_{i,l,k} \boldsymbol{g}_{i,l,k}^{\top}\big)\,. \end{aligned} \tag{15}$$

The second equality in Eq. equation 15 uses the linearity of the trace.

For each sample-wise gradient vector, we simplify the remaining trace term. Using $\mathrm{tr}(\boldsymbol{a}\boldsymbol{a}^{\top}) = \boldsymbol{a}^{\top}\boldsymbol{a}$, we obtain

$$\mathrm{tr}\big(\boldsymbol{g}_{i,l,k} \boldsymbol{g}_{i,l,k}^{\top}\big) = \boldsymbol{g}_{i,l,k}^{\top} \boldsymbol{g}_{i,l,k} = \|\boldsymbol{g}_{i,l,k}\|_2^2\,. \tag{16}$$

Eq. equation 16 turns the trace of each outer product into the squared Euclidean norm of the corresponding sample-wise gradient vector.

Substituting Eq. equation 16 into Eq. equation 15, we get

$$\mathrm{tr}\Big(\boldsymbol{F}_{l,k}^{\mathrm{emp}}\Big) = \frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{g}_{i,l,k}\|_2^2\,. \tag{17}$$

Eq. equation 17 shows that the Fisher trace is the average squared Euclidean norm of the vectorized sample-wise gradients.

Finally, we substitute $\boldsymbol{g}_{i,l,k} = \mathrm{vec}(\boldsymbol{G}_{i,l,k})$ from Eq. equation 13. Since vectorization preserves the squared norm of a matrix, we obtain

$$\|\boldsymbol{g}_{i,l,k}\|_2^2 = \|\mathrm{vec}(\boldsymbol{G}_{i,l,k})\|_2^2 = \|\boldsymbol{G}_{i,l,k}\|_{\mathrm{F}}^2\,. \tag{18}$$

Applying Eq. equation 18 to every term in Eq. equation 17, we obtain

$$\begin{aligned}\operatorname{tr}\Big(\boldsymbol{F}_{l,k}^{\mathrm{emp}}\Big) &= \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{G}_{i,l,k}\|_{\mathrm{F}}^{2} \\ &= \mathcal{S}_{l,k}^{\mathrm{emp}}.\end{aligned} \tag{19}$$

The last equality follows from Eq. equation 3 in Definition 1. Thus, the module sensitivity $\mathcal{S}_{l,k}^{\mathrm{emp}}$ is exactly the trace of the empirical Fisher block for $\boldsymbol{W}_{l,k}$. This explains why the average squared sample-wise gradient norm in Definition 1 can be interpreted as empirical Fisher sensitivity.

D.2 Proof of Lemma 1 (Initial LoRA Gradients)

Lemma. *Under the standard LoRA initialization $\boldsymbol{B}_{l,k} = \mathbf{0}$, with $\boldsymbol{A}_{l,k}$ initialized by the Kaiming-uniform rule, the initial trainable gradients of the LoRA factors are*

$$\nabla_{\boldsymbol{A}_{l,k}}\bar{\ell}_i = \mathbf{0}, \qquad \nabla_{\boldsymbol{B}_{l,k}}\bar{\ell}_i = s\,\boldsymbol{G}_{i,l,k}\,\boldsymbol{A}_{l,k}^{\top}, \tag{20}$$

where $\boldsymbol{G}_{i,l,k}$ is the sample-wise full-weight gradient defined in Eq. equation 2.

Proof. We prove the lemma in two steps. First, we derive the gradients with respect to the LoRA factors for arbitrary $\boldsymbol{A}_{l,k}$ and $\boldsymbol{B}_{l,k}$. Second, we substitute the standard LoRA initialization $\boldsymbol{B}_{l,k} = \mathbf{0}$.

Step 1: Derive the general LoRA-factor gradients. Recall from Eq. equation 4 that, for module (l, k), the LoRA-augmented effective weight is

$$\boldsymbol{W}_{l,k} = \boldsymbol{W}_{l,k}^{0} + s\,\boldsymbol{B}_{l,k}\boldsymbol{A}_{l,k}, \qquad s = \alpha/r. \tag{21}$$

Here, $\boldsymbol{W}_{l,k}^{0} \in \mathbb{R}^{d_{\mathrm{out},k}\times d_{\mathrm{in},k}}$ is the frozen pretrained projection weight, $\boldsymbol{B}_{l,k} \in \mathbb{R}^{d_{\mathrm{out},k}\times r}$ and $\boldsymbol{A}_{l,k} \in \mathbb{R}^{r\times d_{\mathrm{in},k}}$ are trainable LoRA factors, r is the LoRA rank, and $s = \alpha/r$ is the LoRA scaling coefficient. This equation shows that the loss depends on $\boldsymbol{A}_{l,k}$ and $\boldsymbol{B}_{l,k}$ only through the low-rank update $s\boldsymbol{B}_{l,k}\boldsymbol{A}_{l,k}$.

To compute the gradients of the LoRA factors, we track how a small change in $\boldsymbol{A}_{l,k}$ or $\boldsymbol{B}_{l,k}$ changes the effective weight $\boldsymbol{W}_{l,k}$, and then how this change affects the loss. We first take the differential of Eq. equation 21. Since $\boldsymbol{W}_{l,k}^{0}$ is frozen, its differential is zero. Therefore, we obtain

$$\mathrm{d}\boldsymbol{W}_{l,k} = s\,\mathrm{d}\boldsymbol{B}_{l,k}\,\boldsymbol{A}_{l,k} + s\,\boldsymbol{B}_{l,k}\,\mathrm{d}\boldsymbol{A}_{l,k}. \tag{22}$$

Eq. equation 22 separates the change in effective weight into the part caused by $\mathrm{d}\boldsymbol{B}_{l,k}$ and the part caused by $\mathrm{d}\boldsymbol{A}_{l,k}$.

We next relate this change to the sample loss. By the definition of $\boldsymbol{G}_{i,l,k}$ in Eq. equation 2, the first-order loss differential for sample i is

$$\mathrm{d}\bar{\ell}_i = \langle \boldsymbol{G}_{i,l,k},\ \mathrm{d}\boldsymbol{W}_{l,k}\rangle_{\mathrm{F}}\,. \tag{23}$$

Eq. equation 23 states that, to first order, the loss change is the Frobenius inner product between the full-weight gradient and the change in effective weight.

Substituting Eq. equation 22 into Eq. equation 23, we get

$$\mathrm{d}\bar{\ell}_i = s\,\langle \boldsymbol{G}_{i,l,k},\ \mathrm{d}\boldsymbol{B}_{l,k}\,\boldsymbol{A}_{l,k}\rangle_{\mathrm{F}} + s\,\langle \boldsymbol{G}_{i,l,k},\ \boldsymbol{B}_{l,k}\,\mathrm{d}\boldsymbol{A}_{l,k}\rangle_{\mathrm{F}}\,. \tag{24}$$

Eq. equation 24 expresses the same loss differential as the sum of a $\mathrm{d}\boldsymbol{B}_{l,k}$ term and a $\mathrm{d}\boldsymbol{A}_{l,k}$ term. We next rewrite these two terms so that $\mathrm{d}\boldsymbol{B}_{l,k}$ and $\mathrm{d}\boldsymbol{A}_{l,k}$ appear as the second arguments of Frobenius inner products.

For the $\mathrm{d}\boldsymbol{B}_{l,k}$ term, we expand the Frobenius inner product using its trace form and move factors by cyclic invariance of the trace:

$$\begin{aligned}\langle \boldsymbol{G}_{i,l,k},\ \mathrm{d}\boldsymbol{B}_{l,k}\boldsymbol{A}_{l,k}\rangle_{\mathrm{F}} &= \operatorname{tr}\big(\boldsymbol{G}_{i,l,k}^{\top}\mathrm{d}\boldsymbol{B}_{l,k}\boldsymbol{A}_{l,k}\big) \\ &= \operatorname{tr}\big(\boldsymbol{A}_{l,k}\boldsymbol{G}_{i,l,k}^{\top}\mathrm{d}\boldsymbol{B}_{l,k}\big) \\ &= \operatorname{tr}\Big(\big(\boldsymbol{G}_{i,l,k}\boldsymbol{A}_{l,k}^{\top}\big)^{\top}\,\mathrm{d}\boldsymbol{B}_{l,k}\Big) \\ &= \big\langle \boldsymbol{G}_{i,l,k}\boldsymbol{A}_{l,k}^{\top},\ \mathrm{d}\boldsymbol{B}_{l,k}\big\rangle_{\mathrm{F}}\,.\end{aligned} \tag{25}$$

Eq. equation 25 identifies the coefficient of $\mathrm{d}\boldsymbol{B}_{l,k}$ as $\boldsymbol{G}_{i,l,k}\boldsymbol{A}_{l,k}^{\top}$.

Similarly, for the $\mathrm{d}\boldsymbol{A}_{l,k}$ term, we again use the trace form of the Frobenius inner product:

$$\begin{aligned}\langle \boldsymbol{G}_{i,l,k},\ \boldsymbol{B}_{l,k}\mathrm{d}\boldsymbol{A}_{l,k}\rangle_{\mathrm{F}} &= \mathrm{tr}\big(\boldsymbol{G}_{i,l,k}^{\top}\boldsymbol{B}_{l,k}\mathrm{d}\boldsymbol{A}_{l,k}\big)\\ &= \mathrm{tr}\Big(\big(\boldsymbol{B}_{l,k}^{\top}\boldsymbol{G}_{i,l,k}\big)^{\top}\,\mathrm{d}\boldsymbol{A}_{l,k}\Big)\\ &= \big\langle \boldsymbol{B}_{l,k}^{\top}\boldsymbol{G}_{i,l,k},\ \mathrm{d}\boldsymbol{A}_{l,k}\big\rangle_{\mathrm{F}}.\end{aligned} \tag{26}$$

Eq. equation 26 identifies the coefficient of $\mathrm{d}\boldsymbol{A}_{l,k}$ as $\boldsymbol{B}_{l,k}^{\top}\boldsymbol{G}_{i,l,k}$.

Substituting Eq. equation 25 and Eq. equation 26 into Eq. equation 24, we obtain

$$\mathrm{d}\bar{\ell}_i = \big\langle s\,\boldsymbol{G}_{i,l,k}\boldsymbol{A}_{l,k}^{\top},\ \mathrm{d}\boldsymbol{B}_{l,k}\big\rangle_{\mathrm{F}} + \big\langle s\,\boldsymbol{B}_{l,k}^{\top}\boldsymbol{G}_{i,l,k},\ \mathrm{d}\boldsymbol{A}_{l,k}\big\rangle_{\mathrm{F}}. \tag{27}$$

This equation expresses the loss differential directly in terms of the perturbations of the two LoRA factors.

By the definition of gradients with respect to $\boldsymbol{B}_{l,k}$ and $\boldsymbol{A}_{l,k}$, the same differential can also be written as

$$\mathrm{d}\bar{\ell}_i = \big\langle \nabla_{\boldsymbol{B}_{l,k}}\bar{\ell}_i,\ \mathrm{d}\boldsymbol{B}_{l,k}\big\rangle_{\mathrm{F}} + \big\langle \nabla_{\boldsymbol{A}_{l,k}}\bar{\ell}_i,\ \mathrm{d}\boldsymbol{A}_{l,k}\big\rangle_{\mathrm{F}}. \tag{28}$$

Comparing Eq. equation 27 with Eq. equation 28, we obtain the general LoRA factor gradients:

$$\nabla_{\boldsymbol{B}_{l,k}}\bar{\ell}_i = s\,\boldsymbol{G}_{i,l,k}\boldsymbol{A}_{l,k}^{\top}, \qquad \nabla_{\boldsymbol{A}_{l,k}}\bar{\ell}_i = s\,\boldsymbol{B}_{l,k}^{\top}\boldsymbol{G}_{i,l,k}. \tag{29}$$

Eq. equation 29 gives the gradients for arbitrary values of the LoRA factors.

Step 2: Apply the standard LoRA initialization. We now specialize Eq. equation 29 to the standard LoRA initialization. At initialization, $\boldsymbol{B}_{l,k} = \boldsymbol{0}$, while $\boldsymbol{A}_{l,k}$ is a Kaiming-uniform initialized matrix. Substituting $\boldsymbol{B}_{l,k} = \boldsymbol{0}$ into Eq. equation 29 gives

$$\nabla_{\boldsymbol{A}_{l,k}}\bar{\ell}_i = s\,\boldsymbol{0}^{\top}\boldsymbol{G}_{i,l,k} = \boldsymbol{0}, \qquad \nabla_{\boldsymbol{B}_{l,k}}\bar{\ell}_i = s\,\boldsymbol{G}_{i,l,k}\boldsymbol{A}_{l,k}^{\top}. \tag{30}$$

Thus, only $\boldsymbol{B}_{l,k}$ receives a nonzero gradient at initialization, and this gradient is the full-weight gradient $\boldsymbol{G}_{i,l,k}$ projected through the initialized factor $\boldsymbol{A}_{l,k}$. This recovers Eq. equation 5 and proves Lemma 1. □

D.3 Derivation of the PAGE Expression

Theorem. *Let the entries of $\boldsymbol{A}_{l,k} \in \mathbb{R}^{r\times d_{\mathrm{in},k}}$ be drawn i.i.d. from $\mathcal{U}\big(-1/\sqrt{d_{\mathrm{in},k}},\ 1/\sqrt{d_{\mathrm{in},k}}\big)$. Then*

$$\mathbb{E}_{\boldsymbol{A}_{l,k}}\big[\boldsymbol{A}_{l,k}^{\top}\boldsymbol{A}_{l,k}\big] = \frac{r}{3\,d_{\mathrm{in},k}}\,\boldsymbol{I}_{d_{\mathrm{in},k}}, \tag{31}$$

where $\boldsymbol{I}_{d_{\mathrm{in},k}}$ is a $d_{\mathrm{in},k}\times d_{\mathrm{in},k}$ identity matrix. Consequently, PAGE can be written as:

$$\mathrm{PAGE}_{l,k} = \frac{s^2\,r}{3\,d_{\mathrm{in},k}}\,\mathcal{S}_{l,k}^{\mathrm{emp}}. \tag{32}$$

Proof. We prove Theorem 1 in two steps. First, we compute the initialization moment $\mathbb{E}_{\boldsymbol{A}_{l,k}}[\boldsymbol{A}_{l,k}^{\top}\boldsymbol{A}_{l,k}]$, which proves Eq. equation 31. Second, we substitute this moment into the trace form of PAGE in Eq. equation 7 to derive Eq. equation 32.

Step 1: Compute the initialization moment. We first prove Eq. equation 31. Let $\boldsymbol{A}_{l,k} \in \mathbb{R}^{r\times d_{\mathrm{in},k}}$ be the LoRA factor in Theorem 1. For clarity, let

$$A_{qj} = (\boldsymbol{A}_{l,k})_{qj}, \qquad 1 \le q \le r, \quad 1 \le j \le d_{\mathrm{in},k}.$$

Here, q indexes the rank dimension of the LoRA factor, and j indexes the input dimension of the projection weight. Under the initialization assumption in Theorem 1, the entries A_{qj} are independent and identically distributed as

$$A_{qj} \sim \mathcal{U}\left(-\frac{1}{\sqrt{d_{\mathrm{in},k}}}, \frac{1}{\sqrt{d_{\mathrm{in},k}}}\right). \tag{33}$$

From the uniform distribution in Eq. equation 33, we compute the mean and second moment of each entry as

$$\mathbb{E}_{\boldsymbol{A}_{l,k}}[A_{qj}] = 0, \qquad \mathbb{E}_{\boldsymbol{A}_{l,k}}[A_{qj}^2] = \frac{1}{3\, d_{\mathrm{in},k}}. \tag{34}$$

Eq. equation 34 provides the entry-wise moments needed for the matrix expectation. We compute this expectation by evaluating the (j, j')-th entry of $\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}$:

$$\left(\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right)_{jj'} = \sum_{q=1}^{r} A_{qj} A_{qj'}, \qquad 1 \le j, j' \le d_{\mathrm{in},k}. \tag{35}$$

The expectation in Eq. equation 35 depends on whether the two column indices j and j' are equal. When $j \neq j'$, the product $A_{qj}A_{qj'}$ contains two independent zero-mean entries. When $j = j'$, it becomes the squared entry A_{qj}^2. We therefore treat the off-diagonal and diagonal cases separately.

For the off-diagonal case $j \neq j'$, taking expectation on both sides of Eq. equation 35 gives

$$\begin{aligned} \mathbb{E}_{\boldsymbol{A}_{l,k}}\left[\left(\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right)_{jj'}\right] &= \sum_{q=1}^{r} \mathbb{E}_{\boldsymbol{A}_{l,k}}[A_{qj}A_{qj'}] \\ &= \sum_{q=1}^{r} \mathbb{E}_{\boldsymbol{A}_{l,k}}[A_{qj}]\mathbb{E}_{\boldsymbol{A}_{l,k}}[A_{qj'}] \\ &= 0. \end{aligned} \tag{36}$$

The second equality uses the independence of A_{qj} and $A_{qj'}$, and the last equality uses the zero mean in Eq. equation 34. Thus, every off-diagonal entry of the expected matrix is zero.

For the diagonal case $j = j'$, Eq. equation 35 becomes

$$\left(\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right)_{jj} = \sum_{q=1}^{r} A_{qj}^2. \tag{37}$$

Taking expectation on both sides of Eq. equation 37 and using the second moment in Eq. equation 34, we obtain

$$\begin{aligned} \mathbb{E}_{\boldsymbol{A}_{l,k}}\left[\left(\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right)_{jj}\right] &= \sum_{q=1}^{r} \mathbb{E}_{\boldsymbol{A}_{l,k}}[A_{qj}^2] \\ &= \sum_{q=1}^{r} \frac{1}{3\, d_{\mathrm{in},k}} \\ &= \frac{r}{3\, d_{\mathrm{in},k}}. \end{aligned} \tag{38}$$

Thus, every diagonal entry of the expected matrix equals $r/(3d_{\mathrm{in},k})$.

The off-diagonal result in Eq. equation 36 shows that all off-diagonal entries are zero, while the diagonal entries result in Eq. equation 38 shows that every diagonal entry equals $r/(3d_{\mathrm{in},k})$. Therefore, the expected matrix is a scalar multiple of the identity matrix:

$$\mathbb{E}_{\boldsymbol{A}_{l,k}}\left[\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right] = \frac{r}{3\, d_{\mathrm{in},k}}\, \boldsymbol{I}_{d_{\mathrm{in},k}}. \tag{39}$$

Eq. equation 39 proves Eq. equation 8 in Theorem 1.

Step 2: Substitute the initialization moment into PAGE. We now derive Eq. equation 9. Recall from Eq. equation 7 that PAGE can be written as

$$\mathrm{PAGE}_{l,k} = \frac{s^2}{N} \sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{A}_{l,k}}\left[\mathrm{tr}\left(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k}\, \boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right)\right]. \tag{40}$$

Here, $\boldsymbol{G}_{i,l,k}$ is the sample-wise full-weight gradient from Eq. equation 2, and the expectation is taken over the random initialization of $\boldsymbol{A}_{l,k}$.

Since $\boldsymbol{G}_{i,l,k}$ is computed on the pretrained model before LoRA optimization begins, it is fixed with respect to the random initialization of $\boldsymbol{A}_{l,k}$. Therefore, the expectation in Eq. equation 40 acts only on $\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}$. Moving this expectation inside the trace gives

$$\mathbb{E}_{\boldsymbol{A}_{l,k}}\left[\operatorname{tr}\left(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k}\, \boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right)\right] = \operatorname{tr}\left(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k}\, \mathbb{E}_{\boldsymbol{A}_{l,k}}\left[\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right]\right). \tag{41}$$

Eq. equation 41 isolates the only random term $\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}$

Substituting Eq. equation 39 into Eq. equation 41, we get

$$\operatorname{tr}\left(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k}\, \mathbb{E}_{\boldsymbol{A}_{l,k}}\left[\boldsymbol{A}_{l,k}^\top \boldsymbol{A}_{l,k}\right]\right) = \operatorname{tr}\left(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k}\, \frac{r}{3\, d_{\mathrm{in},k}}\, \boldsymbol{I}_{d_{\mathrm{in},k}}\right). \tag{42}$$

Eq. equation 42 replaces the random projection term with a deterministic scalar multiple of the identity matrix.

Because $r/(3d_{\mathrm{in},k})$ is a scalar and $\boldsymbol{I}_{d_{\mathrm{in},k}}$ is the identity matrix, the right-hand side of Eq. equation 42 becomes

$$\operatorname{tr}\left(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k}\, \frac{r}{3\, d_{\mathrm{in},k}}\, \boldsymbol{I}_{d_{\mathrm{in},k}}\right) = \frac{r}{3\, d_{\mathrm{in},k}} \operatorname{tr}\left(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k}\right) = \frac{r}{3\, d_{\mathrm{in},k}} \left\|\boldsymbol{G}_{i,l,k}\right\|_{\mathrm{F}}^2. \tag{43}$$

The last equality in Eq. equation 43 uses the Frobenius norm identity $\|\boldsymbol{G}_{i,l,k}\|_{\mathrm{F}}^2 = \operatorname{tr}(\boldsymbol{G}_{i,l,k}^\top \boldsymbol{G}_{i,l,k})$.

Substituting Eq. equation 43 into Eq. equation 40, we obtain

$$\mathrm{PAGE}_{l,k} = \frac{s^2}{N} \sum_{i=1}^{N} \frac{r}{3\, d_{\mathrm{in},k}} \left\|\boldsymbol{G}_{i,l,k}\right\|_{\mathrm{F}}^2. \tag{44}$$

Eq. equation 44 shows that PAGE is proportional to the average squared Frobenius norm of the sample-wise full-weight gradients.

Factoring out the constants s^2, r, and $3d_{\mathrm{in},k}$ from Eq. equation 44, we obtain

$$\mathrm{PAGE}_{l,k} = \frac{s^2 r}{3\, d_{\mathrm{in},k}} \left(\frac{1}{N} \sum_{i=1}^{N} \left\|\boldsymbol{G}_{i,l,k}\right\|_{\mathrm{F}}^2\right). \tag{45}$$

The term in parentheses is exactly the module sensitivity $\mathcal{S}_{l,k}^{\mathrm{emp}}$ defined in Eq. equation 3. Substituting Eq. equation 3 into Eq. equation 45, we get

$$\mathrm{PAGE}_{l,k} = \frac{s^2 r}{3\, d_{\mathrm{in},k}}\, \mathcal{S}_{l,k}^{\mathrm{emp}}. \tag{46}$$

This completes the proof. □

E Experimental Details

Hardware and default setup. We run training on a single NVIDIA H100 GPU using PyTorch. Unless otherwise specified, we use the default training configuration in Table 7.

Variant-specific settings. Table 8 lists the additional hyperparameters and placement settings used by the LoRA variants and placement baselines. All other settings follow Table 7. For AdaLoRA, r_{target} denotes the target average rank, and t_{final} denotes the number of final fixed-rank steps. Because AdaLoRA uses a dynamic rank schedule, its realized rank and parameter count may vary across runs.

Evaluation prompts. We use the following task-specific prompts:

- **MATH** (Hendrycks et al., 2021b) and **GSM8K** (Cobbe et al., 2021): "Please solve this problem step by step. Put your final answer in \boxed{}."
- **CommonsenseQA** (Talmor et al., 2019): "Let's think step by step. Please format the final answer at the end of the response as: The answer is {answer}."

Table 7: Default training configuration.

Setting	Value
Precision	`bfloat16`, TF32 enabled
DeepSpeed stage	ZeRO-2
Epochs	1
Per-device batch size	4
Gradient accumulation steps	4
Effective batch size	16
Maximum sequence length	3,072
Learning-rate schedule	Linear warmup, then cosine decay
Peak learning rate	2×10^{-5}
Warmup ratio	0.03 of total training steps
Warmup initial learning rate	0
Weight decay	0
LoRA rank r	64
LoRA α	128
LoRA dropout	0.05
Trainable bias	None
Gradient checkpointing	Enabled

Table 8: Variant-specific hyperparameters and placement settings.

Variant	Additional settings
LoRA+	lr_ratio $= 8.0$ (the learning rate for matrix B is $8\times$ that for A)
AdaLoRA	$r_{\text{init}} = 64$, $r_{\text{target}} = 8$, $T_{\text{sched}} = 6{,}237$, $t_{\text{init}} = 623$, $t_{\text{final}} = 1{,}871$, $\Delta T = 10$, $\beta_1 = \beta_2 = 0.85$, $\lambda_{\text{orth}} = 0.1$
GraLoRA	$k = 4$
PLoP	Selects W_V, W_O, and W_{up} via its neural-feature-norm (NFN) criterion
IST	Dynamically selects 8 of the 32 layers to update at each step, using its default configuration

- **MMLU** (Hendrycks et al., 2021a) and **LogiQA** (Liu et al., 2020): output only the option letter.
- **TruthfulQA** (Lin et al., 2022): use the question-and-options prompt in the evaluation data, ending with `Answer:`, without an additional task-specific instruction.
- **TyDiQA** (Clark et al., 2020): extract the exact shortest answer span from the context without translation, formatted as `Answer: {answer}`.
- **HumanEval+** (Liu et al., 2023): output only the Python function definition, without explanations, examples, markdown code blocks, or comments.

F Per-Seed Results on General Instruction Tuning

To make the averages reported in Table 1 traceable, we list the three per-seed results and their mean $\pm$ sample standard deviation below, using the same training settings as Appendix E.

As shown in Table 9, DomLoRA obtains a higher average than vanilla LoRA at all three seeds on both backbones, with mean averages of 68.6 versus 67.4 on Qwen3-8B and 61.8 versus 58.2 on LLaMA-3.1-8B-Instruct. The per-benchmark comparison is not uniform, however, because TyDiQA and CQA favor LoRA for some seeds, whereas TruthfulQA and GSM8K favor DomLoRA in every run. Thus, the three-seed results support a consistent advantage in the reported average under the shared training configuration rather than uniform improvements on every benchmark.

Table 9: Per-seed general instruction tuning results. ✓ and ✗ denote DomLoRA and vanilla LoRA, respectively; shaded italic rows report the mean ± sample standard deviation over the three seeds.

Model	Dom	Seed	MMLU	TyDiQA	CQA	TruthfulQA	GSM8K	LogiQA	Avg.
Qwen	✓	42	71.7	49.7	81.7	57.5	92.4	57.5	68.4
		777	72.1	51.1	82.4	57.0	92.6	57.1	68.7
		979	71.9	50.8	81.0	57.6	92.9	57.1	68.6
		Mean	*71.9±0.2*	*50.5±0.7*	*81.7±0.7*	*57.4±0.3*	*92.6±0.3*	*57.2±0.2*	*68.6±0.2*
	✗	42	71.7	47.1	80.8	48.8	92.0	55.9	66.1
		777	70.4	53.5	80.4	55.1	92.4	55.8	67.9
		979	71.2	54.1	79.7	55.8	91.7	57.1	68.3
		Mean	*71.1±0.7*	*51.6±3.9*	*80.3±0.6*	*53.2±3.9*	*92.0±0.4*	*56.3±0.7*	*67.4±1.2*
LLaMA	✓	42	64.1	49.4	76.9	56.9	86.7	41.3	62.6
		777	63.3	50.8	74.9	56.1	83.5	41.9	61.8
		979	62.9	48.8	74.9	54.7	84.5	40.3	61.0
		Mean	*63.4±0.6*	*49.7±1.0*	*75.6±1.2*	*55.9±1.1*	*84.9±1.6*	*41.2±0.8*	*61.8±0.8*
	✗	42	63.8	49.8	76.4	40.0	78.3	38.1	57.7
		777	63.6	55.3	76.7	42.0	77.6	38.4	58.9
		979	62.9	51.9	76.3	41.6	77.7	38.1	58.1
		Mean	*63.4±0.5*	*52.3±2.8*	*76.5±0.2*	*41.2±1.1*	*77.9±0.4*	*38.2±0.2*	*58.2±0.6*

G General Instruction Tuning with LoRA Variants (RQ2)

Table 10 reports results for LoRA variants on general instruction tuning across Qwen3-8B and LLaMA-3.1-8B-Instruct. Replacing broad placement with dominant-module placement reduces trainable parameters by two orders of magnitude and still improves the average score for most variants. The largest gain is obtained by DoRA on LLaMA-3.1-8B-Instruct, whose average score increases from 57.6 to 61.6; GraLoRA also improves by 3.9 points. On Qwen3-8B, DoRA and AdaLoRA improve from 66.4 to 68.5 and from 67.1 to 69.3, respectively, while NoRM is the only variant that decreases, by 0.2 points.

H Sensitivity to the LoRA Rank and Learning Rate (A1)

We examine the sensitivity of DomLoRA to the learning rate and the LoRA rank–scaling factor pair on LLaMA-3.1-8B. Table 11 sweeps the learning rate with (r, α) fixed to $(64, 128)$. The average score peaks at $2e-5$; a tenfold larger rate drops the average to 56.4, with the largest drop on CommonsenseQA. Table 12 sweeps (r, α) with $\alpha = 2r$ at learning rate $2e-5$. The average score varies from 61.7 to 62.8 across the three settings. These results indicate that the performance gain of dominant-module placement does not rely on a specific LoRA rank, while the learning rate should be kept at the standard value.

I PAGE Stability across Seeds and Sample Counts (A2)

Section 3.2 shows that the dominant adaptation module already appears at Step 0 and persists throughout fine-tuning. This section examines whether the same module is identified when the data subset used for PAGE computation varies.

Data-order seeds. We repeat the Step 0 PAGE evaluation on both backbones with five data-order seeds (42, 777, 979, 1234, and 31337). Each seed shuffles the Tülu V3 training set and draws a fresh subset of 32 samples; all other settings follow Section 3.2. Since the PAGE score scale varies with the sample subset, we compare module rankings rather than raw scores. Table 13 reports the top-two

Table 10: LoRA variant results on general instruction tuning across Qwen3-8B and LLaMA-3.1-8B-Instruct. Shaded rows indicate dominant-module placement, and **bold** marks the best result within each method pair.

Model	Method	Dom	Params	MMLU	TyDiQA	CQA	TruthfulQA	GSM8K	LogiQA	Avg.
Qwen3	NoRM	✗	156.7M	71.3	**51.7**	**81.4**	**62.7**	93.3	56.8	**69.5**
		✓	**0.9M**	**71.5**	48.0	81.2	62.4	**94.0**	**58.4**	69.3
	DoRA	✗	176.0M	71.4	46.3	78.8	53.9	91.8	56.4	66.4
		✓	**1.1M**	**71.6**	**50.0**	**82.0**	**57.0**	**92.6**	**57.5**	**68.5**
	LoRA+	✗	174.6M	70.6	48.7	**80.3**	55.7	**92.9**	**57.8**	67.7
		✓	**1.0M**	**70.8**	**48.9**	80.2	**60.8**	**92.9**	55.9	**68.3**
	AdaLoRA	✗	22.3M	**71.1**	**50.5**	80.2	52.3	91.9	56.5	67.1
		✓	**0.1M**	70.4	48.6	**81.6**	**63.2**	**93.4**	**58.4**	**69.3**
	GraLoRA	✗	174.6M	71.1	50.0	79.9	**54.8**	**92.0**	55.3	67.2
		✓	**1.0M**	**71.8**	**50.3**	**81.6**	53.0	91.9	**57.8**	**67.7**
LLaMA	NoRM	✗	141.9M	**65.7**	**50.7**	**78.5**	52.0	86.4	41.0	**62.4**
		✓	**1.1M**	64.9	49.1	75.9	**54.7**	**86.8**	**43.2**	**62.4**
	DoRA	✗	169.9M	**63.6**	**49.7**	**75.8**	40.6	78.2	37.8	57.6
		✓	**1.2M**	63.4	49.2	74.3	**55.8**	**86.1**	**40.6**	**61.6**
	LoRA+	✗	167.8M	63.0	48.5	73.5	**48.3**	81.7	38.4	58.9
		✓	**1.2M**	**65.0**	**49.7**	**76.0**	44.8	**86.7**	**41.8**	**60.7**
	AdaLoRA	✗	21.1M	64.1	47.8	**76.1**	**51.3**	84.1	39.2	60.4
		✓	**0.1M**	**65.1**	**49.4**	75.2	50.7	**84.4**	**42.2**	**61.2**
	GraLoRA	✗	167.8M	63.7	48.5	74.1	45.2	80.5	37.9	58.3
		✓	**1.2M**	**64.3**	**49.2**	**75.8**	**55.4**	**85.1**	**43.6**	**62.2**

Table 11: Sensitivity of DomLoRA to the learning rate on LLaMA-3.1-8B, with (r, α) fixed to $(64, 128)$.

LR	MMLU	TyDiQA	CQA	TruthfulQA	GSM8K	MATH500	HumanEval+	MT-Bench	Avg.
$2e-4$	63.1	46.9	43.7	57.6	82.9	44.6	54.3	58.0	56.4
$2e-5$	64.1	49.4	76.9	56.9	**86.4**	**48.2**	**59.1**	61.3	**62.8**
$5e-6$	**64.6**	**50.2**	76.4	48.8	86.9	48.0	54.9	**62.6**	61.6

modules for each seed. The Layer 6 down-projection of Qwen3-8B ranks first in all five seeds, with a margin of roughly $10\times$ over the runner-up, and the Layer 1 down-projection of LLaMA-3.1-8B ranks first in all five seeds.

Number of samples. Figure 13 varies the number of PAGE samples from 1 to 32 across the four fine-tuning datasets. The Layer 6 down-projection of Qwen3-8B remains the top-1 candidate in all 128 dataset–sample combinations, and the Layer 1 down-projection of LLaMA-3.1-8B-Instruct in 127 of 128; the single exception is KodCode-V1-SFT-4o with one sample, where $\boldsymbol{W}_{\mathrm{v}}$ at Layer 0 is marginally higher. PAGE-based selection is therefore robust to the number of samples, and the probe itself is cheap: 32 samples take 80–120 seconds on a single H100 GPU, a negligible overhead relative to fine-tuning.

Table 12: Sensitivity of DomLoRA to the LoRA rank and scaling factor on LLaMA-3.1-8B, with $\alpha = 2r$ and learning rate $2e-5$.

(r, α)	MMLU	TyDiQA	CQA	TruthfulQA	GSM8K	MATH500	HumanEval+	MT-Bench	Avg.
(16, 32)	64.8	49.8	**79.2**	55.2	85.9	47.2	54.9	**61.6**	62.3
(32, 64)	**64.8**	**51.2**	77.1	53.9	85.1	45.4	54.9	60.9	61.7
(64, 128)	64.1	49.4	76.9	**56.9**	**86.4**	**48.2**	**59.1**	61.3	**62.8**

Table 13: Top-ranked modules under five data-order seeds at Step 0.

Backbone	Seed	Top-1	Top-2
Qwen3-8B	42	$\boldsymbol{W}_{\text{down}}@6$	$\boldsymbol{W}_{\text{v}}@22$
	777	$\boldsymbol{W}_{\text{down}}@6$	$\boldsymbol{W}_{\text{v}}@22$
	979	$\boldsymbol{W}_{\text{down}}@6$	$\boldsymbol{W}_{\text{v}}@22$
	1234	$\boldsymbol{W}_{\text{down}}@6$	$\boldsymbol{W}_{\text{v}}@22$
	31337	$\boldsymbol{W}_{\text{down}}@6$	$\boldsymbol{W}_{\text{v}}@22$
LLaMA-3.1-8B-Instruct	42	$\boldsymbol{W}_{\text{down}}@1$	$\boldsymbol{W}_{\text{v}}@0$
	777	$\boldsymbol{W}_{\text{down}}@1$	$\boldsymbol{W}_{\text{v}}@1$
	979	$\boldsymbol{W}_{\text{down}}@1$	$\boldsymbol{W}_{\text{v}}@0$
	1234	$\boldsymbol{W}_{\text{down}}@1$	$\boldsymbol{W}_{\text{v}}@0$
	31337	$\boldsymbol{W}_{\text{down}}@1$	$\boldsymbol{W}_{\text{v}}@0$

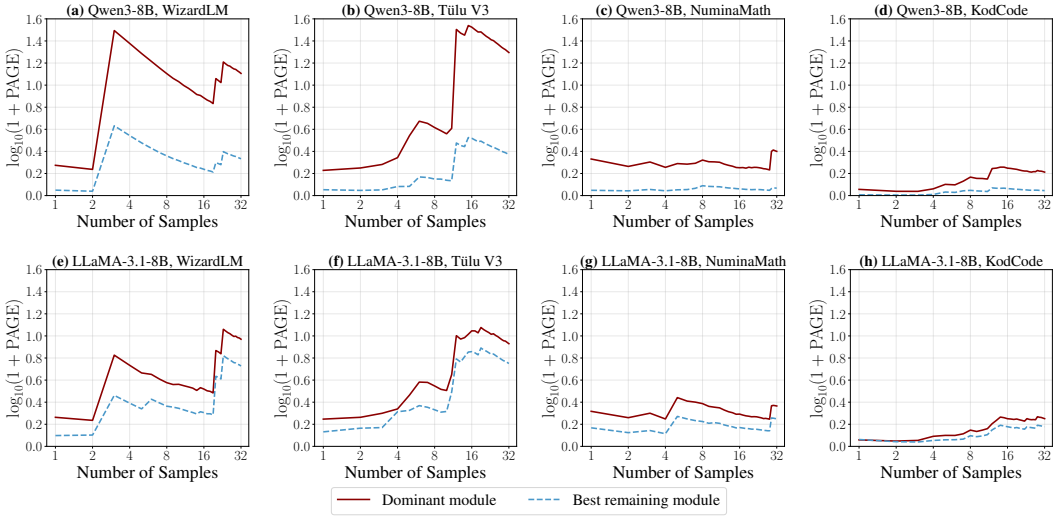


Figure 13: **PAGE under varying sample sizes.** Each panel compares the dominant module with the best remaining module as the number of samples varies from 1 to 32. The top row shows Qwen3-8B, and the bottom row shows LLaMA-3.1-8B-Instruct.