GRAMLORA: CROSS-SCALE OPTIMIZATION TRAJECTORY TRANSFER FOR LORA FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter-efficient fine-tuning (PEFT), particularly low-rank adaptation (LoRA), has become a standard approach for adapting large language models without updating the full parameter space. Existing LoRA variants mainly improve the parameterization, initialization, or optimization of low-rank updates, while adaptation is typically learned independently for each target model. We ask a complementary question: can the optimization trajectory of a smaller model guide LoRA adaptation of a larger one? We introduce **GramLoRA**, a cross-scale framework that transfers the optimization trajectory of a smaller same-family source model to guide LoRA adaptation of a larger target model. We fully fine-tune the source model and convert intermediate parameter residuals into target-compatible rank-r factors. The final source state provides a task-informed, identity-preserving initialization, while intermediate states provide stage-aware directional guidance without changing the downstream objective. Across Qwen2.5-7B, Gemma4-E4B, and Ministral-8B on eight commonsense reasoning tasks, GramLoRA consistently improves adaptation under limited target-model training. With 20% of Commonsense170K for 0.5 epoch, GramLoRA outperforms HiRA by 2.2, 4.6, and 2.9 average points, respectively, and also generalizes to natural language understanding and mathematical reasoning. Its advantage diminishes with extended target-model training, suggesting that transferred source-model optimization information is particularly effective for early target-model adaptation.

1 INTRODUCTION

Parameter-efficient fine-tuning (PEFT) (Houlsby et al., 2019; Ding et al., 2023) has become a standard approach for adapting large language models (LLMs) without the cost of full fine-tuning. Among PEFT methods, low-rank adaptation (LoRA) (Hu et al., 2022) is particularly attractive because it freezes pretrained weights and learns only a low-rank residual BA, substantially reducing the number of trainable parameters while allowing the update to be merged into the base model at inference. Recent LoRA variants improve adaptation from two complementary perspectives. Parameterization-oriented methods, such as DoRA (Liu et al., 2024) and HiRA (Huang et al., 2025), improve the flexibility of low-rank updates. Optimization-oriented methods instead seek to better align LoRA optimization with updates in the full parameter space. For example, LoRA-GA (Wang et al., 2024b) uses the target model's full-parameter gradient to construct a gradient-aligned initialization, while LoRA-Pro (Wang et al., 2025) adjusts LoRA-factor gradients so that their induced updates better approximate the corresponding full-parameter gradients.

A complementary question is whether task-specific adaptation must be learned independently for every model. Prior work suggests that adaptation information can transfer across model scales. Knowledge acquired by smaller models has been reused through behavioral signals or supervision (Mitchell et al., 2024; Wang et al., 2026), while other studies establish correspondence between parameter-efficient adaptations across model sizes and transfer low-rank updates across models (Jin et al., 2023; Zhang et al., 2025; Xia et al., 2025). These findings suggest that task-specific adaptation contains reusable cross-model structure.

Existing approaches, however, primarily transfer the *outcome* of adaptation rather than information from the optimization process itself. Fine-tuning a source model produces not only a final adapted

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.

state, but also a sequence of intermediate states,

$$\theta_0^{\mathrm{src}} \to \theta_1^{\mathrm{src}} \to \cdots \to \theta_T^{\mathrm{src}},$$

which captures how task-specific adaptation evolves during training. We hypothesize that these intermediate states contain useful optimization information beyond the final adapted state alone. This motivates the central question of this work: *can the optimization trajectory of a smaller source model provide useful guidance for LoRA adaptation of a larger target model?*

We introduce **GramLoRA**, a cross-scale framework that transfers optimization information from a smaller source model to guide LoRA adaptation of a larger target model. We fully fine-tune a same-family source model and record intermediate checkpoints during training. The corresponding parameter residuals are converted into target-compatible rank-r factors, forming a cross-scale trajectory prior. GramLoRA uses this prior in two complementary ways: the final source state provides a task-informed, identity-preserving initialization, while intermediate states provide stage-aware directional guidance during target-model training. The downstream objective remains unchanged; the transferred information acts only as an auxiliary optimization prior.

Across three model families, GramLoRA consistently improves adaptation under limited target-model training. With 20% of Commonsense170K (Hu et al., 2023) for 0.5 epoch, GramLoRA outperforms HiRA by 2.2, 4.6, and 2.9 points on Qwen2.5-7B (Yang et al., 2024), Gemma4-E4B (Gemma Team, 2026), and Ministral-8B (Mistral AI, 2026), respectively. The gains also extend to natural language understanding and mathematical reasoning. Ablations show that source-informed initialization provides the largest individual gain, while trajectory guidance yields additional improvements. As the target-model training budget increases, the advantage gradually shrinks and eventually reverses, suggesting that the transferred source prior is particularly effective for early target-model adaptation.

2 Related Work

Parameter-efficient and low-rank adaptation. Parameter-efficient fine-tuning (PEFT) adapts pretrained models while updating only a small fraction of their parameters. Representative approaches include adapters (Houlsby et al., 2019; Pfeiffer et al., 2021), prompt- and prefix-based tuning (Li & Liang, 2021; Lester et al., 2021; Liu et al., 2022b), BitFit (Ben Zaken et al., 2022), and $(IA)^3$ (Liu et al., 2022a). LoRA (Hu et al., 2022) instead parameterizes task-specific updates as low-rank residuals while freezing the pretrained backbone. Subsequent variants improve LoRA through richer parameterizations, such as DoRA (Liu et al., 2024) and HiRA (Huang et al., 2025), dynamic rank allocation (Zhang et al., 2023; Valipour et al., 2023; Lialin et al., 2023), improved scaling and optimization schedules (Hayou et al., 2024; Kalajdzievski, 2023), or quantization-aware adaptation (Dettmers et al., 2023). GramLoRA retains the standard low-rank form and instead transfers optimization information from a smaller source model to guide a larger target model.

Cross-model adaptation transfer. A complementary line of work studies whether adaptation information learned by one model can benefit another. Weak-to-strong generalization uses supervision from weaker models to improve stronger ones (Burns et al., 2023), while other work studies cross-scale representation correspondence (Xue et al., 2025). EFT (Mitchell et al., 2024) transfers small-model adaptation effects at the behavioral level, and LightReasoner (Wang et al., 2026) uses smaller-model signals to construct supervision for larger-model reasoning. In parameter space, Jin et al. (Jin et al., 2023) exploit correspondence between parameter-efficient modules across model scales. LoRAM (Zhang et al., 2025) trains adapters on a smaller surrogate and restores them to the original model, while Cross-LoRA (Xia et al., 2025) aligns low-rank adaptation spaces across heterogeneous LLMs. These methods transfer behaviors, supervision, representations, or adapted parameter states. GramLoRA instead transfers optimization information collected from multiple states along a smaller source model's fine-tuning trajectory.

Optimization-aware LoRA. Another line of work improves LoRA through additional initialization or optimization signals. PiSSA (Meng et al., 2024) initializes low-rank factors from principal singular components of pretrained weights, MiLoRA (Wang et al., 2024a) uses minor singular components, and LoftQ (Li et al., 2024) jointly considers quantization error and low-rank initialization. Gradient-based methods use target-model optimization signals more directly. LoRA-GA (Wang

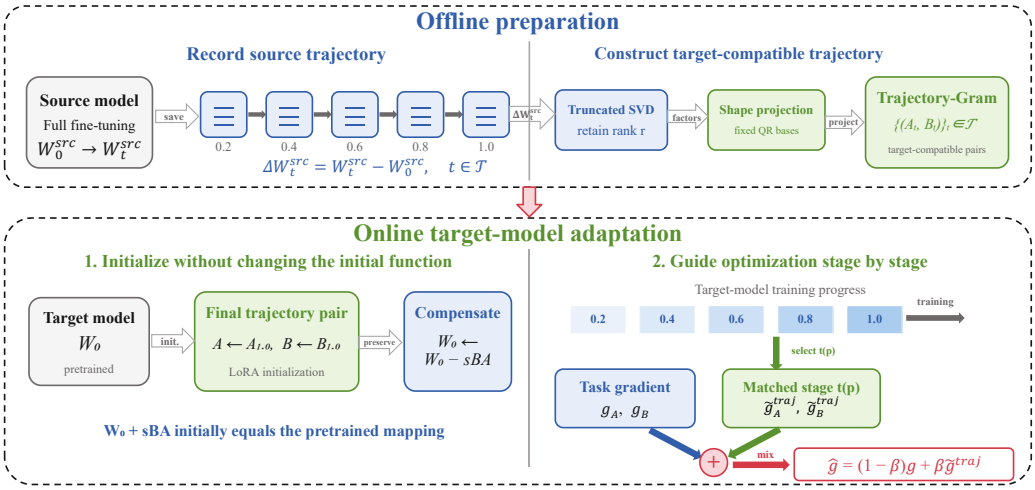


Figure 1: Overview of GramLoRA. A smaller source model provides a target-compatible low-rank trajectory. The final state initializes target LoRA, while intermediate states guide stage-aware optimization.

et al., 2024b) constructs a gradient-aligned initialization from the target model's full-parameter gradient, whereas LoRA-Pro (Wang et al., 2025) adjusts LoRA-factor gradients so that their induced updates better approximate full-parameter gradients. In contrast, GramLoRA derives optimization information from a smaller same-family source model: its final state provides task-informed initialization, while intermediate states provide directional guidance during target-model training. Thus, GramLoRA complements existing optimization-aware methods by transferring cross-scale optimization information rather than relying only on target-model weights or instantaneous gradients.

3 Method

We propose **GramLoRA**, which transfers optimization information from a smaller fully fine-tuned source model to guide LoRA adaptation of a larger target model. As illustrated in Figure 1, GramLoRA first constructs a target-compatible trajectory offline, then uses its final state for initialization and intermediate states for stage-aware optimization.

3.1 Constructing the Target-Compatible Trajectory

Consider a pair of corresponding linear layers in the source and target models. Let $d_{\text{in}}^{\text{src}}$ and $d_{\text{out}}^{\text{src}}$ denote the input and output dimensions of the source layer, and let $d_{\text{in}}^{\text{tgt}}$ and $d_{\text{out}}^{\text{tgt}}$ denote those of the target layer. The frozen target-model weight W_0 and the adapted target weight W have the same shape:

$$W_0, \; W \in \mathbb{R}^{d_{\text{out}}^{\text{tgt}} \times d_{\text{in}}^{\text{tgt}}}.$$

LoRA (Hu et al., 2022) parameterizes the adapted target weight as

$$W = W_0 + sBA, \tag{1}$$

where

$$A \in \mathbb{R}^{r \times d_{\text{in}}^{\text{tgt}}}, \qquad B \in \mathbb{R}^{d_{\text{out}}^{\text{tgt}} \times r},$$

r is the adapter rank, and s is the LoRA scaling factor.

To obtain a transferable optimization prior, we fully fine-tune a smaller same-family source model on the downstream corpus and record checkpoints at normalized training fractions

$$\mathcal{T} = \{0.2, 0.4, 0.6, 0.8, 1.0\}. \tag{2}$$

For each stage $t \in \mathcal{T}$, let

$$W_t^{\mathrm{src}},\ W_0^{\mathrm{src}} \in \mathbb{R}^{d_{\mathrm{out}}^{\mathrm{src}} \times d_{\mathrm{in}}^{\mathrm{src}}}$$

denote the corresponding fine-tuned and pretrained source-model weights, respectively. We define the source-model parameter residual as

$$\Delta W_t^{\mathrm{src}} = W_t^{\mathrm{src}} - W_0^{\mathrm{src}} \in \mathbb{R}^{d_{\mathrm{out}}^{\mathrm{src}} \times d_{\mathrm{in}}^{\mathrm{src}}}. \tag{3}$$

The collection $\{\Delta W_t^{\mathrm{src}}\}_{t \in \mathcal{T}}$ provides a discrete description of the source-model fine-tuning trajectory.

The dense residuals cannot be directly used by the target LoRA adapter: they are generally full-rank and their dimensions may differ from those of the corresponding target layers. We therefore convert each source residual into a target-compatible rank-r factor pair.

For each stage t, we apply truncated SVD to the source residual. Retaining the leading r components gives

$$\Delta W_t^{\mathrm{src}} \approx U_t^{(r)} \Sigma_t^{(r)} (V_t^{(r)})^\top,$$

where

$$U_t^{(r)} \in \mathbb{R}^{d_{\mathrm{out}}^{\mathrm{src}} \times r}, \qquad \Sigma_t^{(r)} \in \mathbb{R}^{r \times r}, \qquad V_t^{(r)} \in \mathbb{R}^{d_{\mathrm{in}}^{\mathrm{src}} \times r}.$$

We split the singular values symmetrically and define

$$B_t^{\mathrm{src}} = U_t^{(r)} (\Sigma_t^{(r)})^{1/2} \in \mathbb{R}^{d_{\mathrm{out}}^{\mathrm{src}} \times r},$$
$$A_t^{\mathrm{src}} = (\Sigma_t^{(r)})^{1/2} (V_t^{(r)})^\top \in \mathbb{R}^{r \times d_{\mathrm{in}}^{\mathrm{src}}}.$$

Consequently, $B_t^{\mathrm{src}} A_t^{\mathrm{src}}$ is a rank-r approximation of $\Delta W_t^{\mathrm{src}}$ with the same $d_{\mathrm{out}}^{\mathrm{src}} \times d_{\mathrm{in}}^{\mathrm{src}}$ shape.

Cross-scale shape projection. The source and target models generally have different layer dimensions, so the source factors cannot be directly reused by the target adapter. For each source–target shape pair, we construct fixed projection matrices from random matrices using QR-derived orthonormal bases:

$$P_{\mathrm{in}} \in \mathbb{R}^{d_{\mathrm{in}}^{\mathrm{src}} \times d_{\mathrm{in}}^{\mathrm{tgt}}}, \qquad P_{\mathrm{out}} \in \mathbb{R}^{d_{\mathrm{out}}^{\mathrm{tgt}} \times d_{\mathrm{out}}^{\mathrm{src}}}.$$

The source factors are mapped to the target-layer shape as

$$A_t = A_t^{\mathrm{src}} P_{\mathrm{in}} \in \mathbb{R}^{r \times d_{\mathrm{in}}^{\mathrm{tgt}}}, \qquad B_t = P_{\mathrm{out}} B_t^{\mathrm{src}} \in \mathbb{R}^{d_{\mathrm{out}}^{\mathrm{tgt}} \times r}. \tag{4}$$

Therefore,

$$B_t A_t \in \mathbb{R}^{d_{\mathrm{out}}^{\mathrm{tgt}} \times d_{\mathrm{in}}^{\mathrm{tgt}}},$$

which matches the shape of the corresponding target-model weight. The same P_{in} and P_{out} are shared across all trajectory stages $t \in \mathcal{T}$ for the same source–target shape pair. They remain fixed after construction and introduce no additional trainable parameters.

Repeating this projection for all trajectory stages yields the target-compatible trajectory

$$\mathcal{G} = \{(A_t, B_t)\}_{t \in \mathcal{T}}. \tag{5}$$

The trajectory is constructed once offline and reused during target-model adaptation.

3.2 Trajectory-Informed Initialization

The final trajectory state contains the source model's accumulated task-specific update after full fine-tuning. We therefore use the corresponding target-compatible factors $(A_{1.0}, B_{1.0})$ to initialize the target LoRA adapter:

$$A \leftarrow A_{1.0}, \qquad B \leftarrow B_{1.0}. \tag{6}$$

Unlike standard LoRA initialization, this provides a non-zero, task-informed starting point derived from the source-model optimization trajectory.

Directly inserting these factors would modify the target model's effective weight at initialization. To preserve the original pretrained forward function, we compensate the frozen base weight as

$$W_0 \leftarrow W_0 - sB_{1.0}A_{1.0}. \tag{7}$$

After this compensation, the effective weight at step 0 remains

$$W_0 + sBA = W_0^{\mathrm{pre}}, \tag{8}$$

where W_0^{pre} denotes the original pretrained weight. Thus, GramLoRA preserves the pretrained model's initial function while starting optimization from source-informed LoRA factors.

3.3 STAGE-AWARE TRAJECTORY GUIDANCE

The final trajectory state provides only the initialization of the target adapter. To further exploit the intermediate source states, we use them as stage-dependent directional priors during target-model optimization.

Let $p \in [0, 1]$ denote the normalized target-model training progress. For the trajectory stages $\mathcal{T} = \{0.2, 0.4, 0.6, 0.8, 1.0\}$, we select the active stage as

$$t(p) = \min \{u \in \mathcal{T} : p \leq u\}. \tag{9}$$

The corresponding target-compatible pair $(A_t, B_t) \in \mathcal{G}$ defines the stage-specific source direction

$$T_t = B_t A_t. \tag{10}$$

The target model is optimized using only the downstream task loss $\mathcal{L}$. Standard back-propagation produces the task gradients

$$g_A = \nabla_A \mathcal{L}, \qquad g_B = \nabla_B \mathcal{L}. \tag{11}$$

We then construct an additional guidance direction that encourages the current LoRA update BA to align with T_t. Using the Frobenius inner product $\langle BA, T_t \rangle_F$, the corresponding descent-compatible directions are

$$g_A^{\text{traj}} = -B^\top T_t, \qquad g_B^{\text{traj}} = -T_t A^\top. \tag{12}$$

Because gradient descent moves in the negative gradient direction, these terms increase the alignment between the current target-model update and the active source-derived direction.

The task and trajectory gradients may have substantially different magnitudes. We therefore normalize the trajectory component to the corresponding task-gradient norm:

$$\widetilde{g}_A^{\text{traj}} = \|g_A\|_F \frac{g_A^{\text{traj}}}{\|g_A^{\text{traj}}\|_F + \varepsilon}, \tag{13}$$

$$\widetilde{g}_B^{\text{traj}} = \|g_B\|_F \frac{g_B^{\text{traj}}}{\|g_B^{\text{traj}}\|_F + \varepsilon}. \tag{14}$$

The gradients passed to the optimizer are then

$$\widehat{g}_A = (1 - \beta) g_A + \beta \widetilde{g}_A^{\text{traj}}, \tag{15}$$

$$\widehat{g}_B = (1 - \beta) g_B + \beta \widetilde{g}_B^{\text{traj}}. \tag{16}$$

This norm matching prevents the source guidance from dominating solely because of its scale, while retaining its directional information.

Stage-aware mixing schedule. Within each trajectory stage, we use stronger guidance immediately after a new source state becomes active and gradually reduce its contribution. Let t_- denote the previous stage boundary, with $t_- = 0$ for the first stage, and define the local progress

$$\lambda = \frac{p - t_-}{t(p) - t_-} \in [0, 1]. \tag{17}$$

Given a base mixing coefficient β_0, we set

$$\beta(\lambda) = \begin{cases} 1.5\beta_0, & \lambda < 0.5, \\ 1.5\beta_0 \dfrac{1 - \lambda}{0.5}, & \lambda \geq 0.5. \end{cases} \tag{18}$$

Thus, each newly activated source state exerts stronger influence in the first half of its stage, after which the guidance gradually decays and the task gradient receives increasing emphasis. No auxiliary loss is introduced; GramLoRA modifies only the LoRA gradients before each optimizer step.

Table 1: Commonsense reasoning under a limited budget (0.5 epoch, 20% training data). The best result in each column within each model block is shown in **bold**.

LLMs	Methods	BoolQ	PIQA	SIQA	ARC-c	ARC-e	OBQA	HellaS.	WinoG.	Avg.
Qwen2.5-7B	LoRA	63.4	82.3	72.4	82.1	88.4	82.0	85.8	70.8	78.4
	DoRA	68.9	86.1	76.5	87.2	93.5	86.6	88.4	74.7	82.7
	HiRA	67.4	87.2	77.2	89.7	**96.5**	88.0	89.5	82.1	84.7
	PiSSA	70.2	88.4	**81.0**	88.0	95.3	88.8	**92.8**	83.3	85.9
	LoRA-GA	68.2	88.0	79.1	**89.8**	96.1	87.2	92.2	81.8	85.3
	LoRA-Pro	64.5	**89.1**	78.2	83.6	96.0	84.0	83.3	70.3	81.1
	GramLoRA	**71.8**	88.7	80.7	89.0	96.3	**91.4**	92.6	**84.9**	**86.9**
Gemma4-E4B	LoRA	62.5	80.5	71.8	77.1	86.5	77.8	81.1	72.6	76.2
	DoRA	66.9	84.3	76.2	80.9	90.7	82.1	85.4	77.0	80.7
	HiRA	**67.8**	85.5	74.5	**83.6**	**93.1**	81.6	74.9	74.2	79.4
	GramLoRA	67.2	**86.4**	**80.9**	80.7	91.2	**84.8**	**94.3**	**86.7**	**84.0**
Ministral-8B	LoRA	64.2	83.5	73.9	84.4	90.9	85.4	87.0	76.9	80.8
	DoRA	70.2	84.9	73.7	71.9	84.3	80.6	91.8	80.6	79.7
	HiRA	66.4	86.2	76.2	88.7	95.5	87.0	88.5	81.1	83.7
	GramLoRA	**71.2**	**88.3**	**79.3**	**89.7**	**95.7**	**90.6**	**93.7**	**84.1**	**86.6**

4 EXPERIMENTS

We evaluate GramLoRA across commonsense reasoning, mathematical reasoning, and natural language understanding. Our experiments address six questions: (i) whether source-model trajectory priors improve target-model LoRA adaptation; (ii) whether the gains generalize across model families and task domains; (iii) how much of the improvement comes from source-informed initialization versus trajectory-guided optimization; (iv) how the benefit changes as the target-model training budget increases; (v) how source–target compatibility affects trajectory transfer; and (vi) how source–target compatibility affects trajectory transfer.

4.1 EXPERIMENTAL SETUP

Models. We evaluate Qwen2.5-7B (Yang et al., 2024), Gemma4-E4B (Gemma Team, 2026), and Ministral-8B (Mistral AI, 2026). For each target model, we fully fine-tune a source model from the same family and record checkpoints at normalized training fractions $\mathcal{T} = \{0.2, 0.4, 0.6, 0.8, 1.0\}$. The source models are Qwen2.5-0.5B, Gemma4-E2B, and Ministral-3B, respectively. Source-model fine-tuning and trajectory construction are performed offline; the source model is not used during target-model optimization. Unless otherwise specified, all LoRA-based methods use rank $r = 32$ on the Q, K, V, U, and D projections.

Tasks. For commonsense reasoning, we follow the Commonsense170K protocol of LLM-Adapters (Hu et al., 2023) and evaluate BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), ARC-Challenge (ARC-c) and ARC-Easy (ARC-e) (Clark et al., 2018), OpenBookQA (OBQA) (Mihaylov et al., 2018), HellaSwag (HellaS) (Zellers et al., 2019), and WinoGrande (WinoG) (Sakaguchi et al., 2021). We report individual accuracies and their unweighted eight-task average. We additionally evaluate mathematical reasoning on GSM8K (Cobbe et al., 2021) and natural language understanding on six GLUE tasks (Wang et al., 2019).

Baselines. We compare GramLoRA with LoRA (Hu et al., 2022), DoRA (Liu et al., 2024), and HiRA (Huang et al., 2025) under matched target-model data and training schedules. On Qwen2.5-7B, we additionally compare with PiSSA (Meng et al., 2024), LoRA-GA (Wang et al., 2024b), and LoRA-Pro (Wang et al., 2025), covering both initialization-aware and optimization-aware LoRA variants.

Optimization. For Commonsense170K, the primary short-training setting uses 20% of the training mixture for 0.5 epoch. All methods use the same fixed subset. Unless otherwise noted, adapters

Table 2: GLUE results on Qwen2.5-7B with 20% training data. CoLA uses Matthews correlation, STS-B uses Pearson correlation, and the remaining tasks use accuracy. The best result in each column is shown in **bold**.

Methods	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	Avg.
LoRA	93.6	79.4	58.2	71.7	51.5	86.1	73.4
DoRA	93.8	81.4	58.6	86.5	51.8	86.9	76.5
HiRA	93.6	81.6	58.2	88.5	52.7	86.1	76.8
PiSSA	94.7	82.2	58.2	92.2	56.4	86.2	78.3
LoRA-GA	**95.8**	78.4	**59.3**	91.3	57.7	86.8	78.2
LoRA-Pro	93.4	81.6	58.2	90.4	57.6	87.8	78.2
GramLoRA	93.8	**84.3**	58.8	**93.2**	**87.1**	**89.5**	**84.4**

are trained with AdamW (Loshchilov & Hutter, 2019), learning rate 3×10^{-4}, per-device batch size 8, gradient accumulation 2, and 100 warmup steps. We use bfloat16 and gradient checkpointing. The default GramLoRA coefficient is $\beta_0 = 0.05$. Full implementation details are provided in Appendix A.

4.2 Main Results on Commonsense Reasoning

Table 1 reports the main results using 20% of Commonsense170K for 0.5 epoch. GramLoRA achieves the highest eight-task average across all three target-model families. Compared with HiRA, it improves the average by $+2.2$ points on Qwen2.5-7B (86.9 vs. 84.7), $+4.6$ on Gemma4-E4B (84.0 vs. 79.4), and $+2.9$ on Ministral-8B (86.6 vs. 83.7). Relative to vanilla LoRA, the gains are $+8.5$, $+7.8$, and $+5.8$ points, respectively. On Qwen2.5-7B, GramLoRA also outperforms PiSSA (85.9), LoRA-GA (85.3), and LoRA-Pro (81.1). These results show that GramLoRA remains competitive against both standard LoRA variants and methods that explicitly modify initialization or target-model optimization.

4.3 Generalization across Task Domains

We evaluate six GLUE (Wang et al., 2019) tasks on Qwen2.5-7B using 20% of each task's labeled training set. Adapters are applied to the Q and V projections with rank $r = 16$. As shown in Table 2, GramLoRA achieves an average score of 84.4, compared with 76.8 for HiRA and 73.4 for LoRA. It also outperforms PiSSA, LoRA-GA, and LoRA-Pro on the overall average. The largest gains are observed on RTE, QNLI, and MRPC, while performance on SST-2 and CoLA remains closer across methods. These results indicate that the transferred optimization prior is not specific to commonsense reasoning and remains effective on standard natural language understanding tasks. We additionally evaluate mathematical reasoning on GSM8K, where GramLoRA also outperforms HiRA and LoRA under the limited-data setting; full results are provided in Appendix B.1.

4.4 Where Does the Gain Come From?

The main results show that GramLoRA substantially improves LoRA, but do not reveal whether the gain mainly comes from source-informed initialization or from using the source trajectory during target-model optimization. We therefore construct several controlled variants on Qwen2.5-7B under the main Commonsense170K setting (20% data, 0.5 epoch).

Table 3: Ablation of source-informed initialization and trajectory guidance on Qwen2.5-7B.

Variant	Training-time guidance	Avg.
LoRA	None	78.4
Init only	None	84.6
Final-state guidance	$T_{1.0}$ throughout	86.3
Shuffled trajectory	Shuffled T_t	86.4
GramLoRA	Stage-matched T_t	**86.9**

Initialization provides the largest individual gain. The *Init only* variant uses the final projected source state to initialize the target LoRA factors but disables trajectory guidance during training. This alone improves the eight-task average from 78.4 to 84.6, showing that the final source state provides a strong task-informed initialization.

Table 4: Commonsense reasoning on the full training set using the same 0.5-epoch schedule. The best result in each column within each model block is shown in **bold**.

LLMs	Methods	BoolQ	PIQA	SIQA	ARC-c	ARC-e	OBQA	HellaS.	WinoG.	Avg.
Qwen2.5-7B	LoRA	65.9	83.3	74.0	83.8	90.3	85.2	88.3	79.2	81.3
	DoRA	70.5	87.7	77.7	87.5	94.0	89.4	92.4	84.0	85.4
	HiRA	71.2	**89.7**	77.5	**89.1**	**96.3**	**90.4**	94.3	87.1	87.0
	GramLoRA	**72.8**	88.2	**81.2**	86.9	95.8	90.2	**94.6**	**88.6**	**87.3**
Gemma4-E4B	LoRA	64.9	82.5	73.4	77.2	86.8	82.2	87.8	78.4	79.2
	DoRA	69.1	86.4	77.7	81.0	91.0	86.6	91.9	82.9	83.5
	HiRA	68.5	**88.0**	79.6	**85.5**	**93.4**	**87.6**	93.6	84.8	85.1
	GramLoRA	**72.5**	87.2	**82.2**	83.2	92.8	87.2	**95.1**	**88.6**	**86.1**
Ministral-8B	LoRA	65.4	84.4	75.8	84.4	90.9	84.6	89.7	81.5	82.1
	DoRA	71.4	89.4	78.9	88.5	95.0	90.6	93.6	87.2	86.8
	HiRA	68.9	88.4	**82.3**	**90.6**	**96.3**	**91.4**	93.5	86.8	87.3
	GramLoRA	**73.0**	**90.0**	81.5	87.8	95.8	89.8	**94.5**	**88.1**	**87.6**

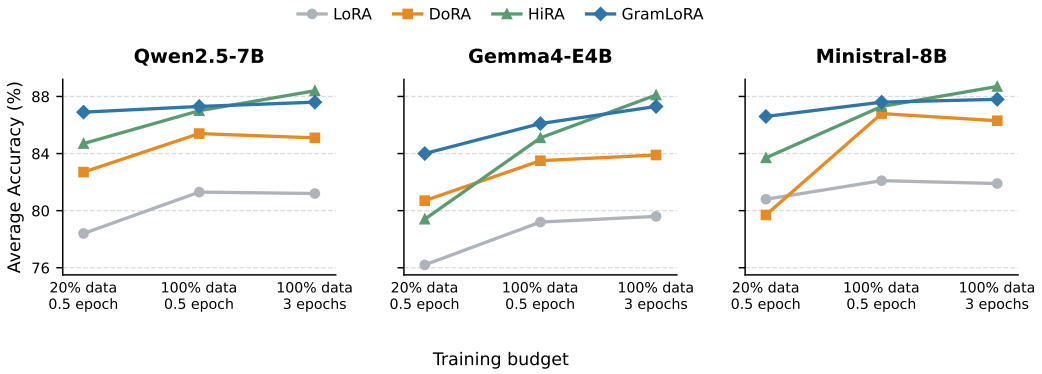


Figure 2: Performance of LoRA, DoRA, HiRA, and GramLoRA under increasing target-model training budgets.

Training-time guidance provides additional improvement. Keeping the same initialization while using the final source state throughout training increases the average to 86.3. Thus, the transferred source information remains useful beyond initialization alone.

Intermediate trajectory states provide a smaller additional gain. When the same intermediate states are used in a shuffled order, the average reaches 86.4, whereas the default stage-matched trajectory achieves the best result of 86.9. The relatively small difference indicates that much of the training-time benefit comes from the transferred source directions themselves, while preserving their stage correspondence provides an additional improvement.

Overall, the results suggest that GramLoRA benefits from two complementary sources of transfer: a strong source-informed starting point and continued trajectory guidance during target-model optimization. Detailed per-task ablations, including the effect of identity-preserving compensation, are provided in Appendix E.

4.5 Effect of the Target-Model Training Budget

We next examine how the benefit of GramLoRA changes as the target model receives more optimization. We consider three budgets: 0.5 epoch on 20% of the training data, 0.5 epoch on the full training set, and 3 epochs on the full training set.

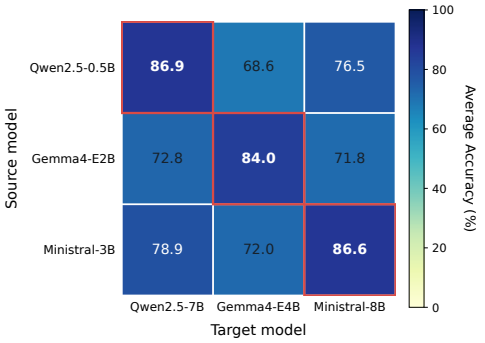


Figure 3: Source-model compatibility.

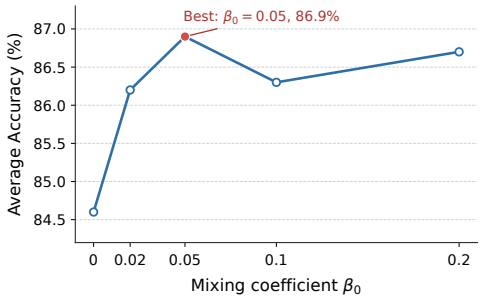


Figure 4: Sensitivity to β_0.

As shown in Figure 2, GramLoRA achieves its largest advantage under the limited-data setting. When the same short schedule is applied to the full training set, the gap over HiRA becomes much smaller. After three full epochs, HiRA slightly surpasses GramLoRA on all three model families.

This consistent crossover suggests that the transferred source prior primarily improves early target-model adaptation, while its relative benefit diminishes as target-specific optimization accumulates. Complete task-level results and exact GramLoRA–HiRA differences are reported in Appendix C.

4.6 Source-Model Compatibility

We finally examine whether the transferred trajectory acts as a generic low-rank prior or depends on compatibility between the source and target models. We swap the three source-model trajectories across the three target-model families while keeping the target-model training configuration fixed.

Figure 3 shows a clear diagonal pattern. For each target model, the strongest result is obtained when the source comes from the same model family: Qwen2.5-0.5B for Qwen2.5-7B, Gemma4-E2B for Gemma4-E4B, and Ministral-3B for Ministral-8B. Cross-family transfer consistently performs worse than same-family transfer, although the magnitude of degradation varies across source–target pairs. This result suggests that the usefulness of the transferred optimization trajectory depends strongly on source–target compatibility, rather than acting as an arbitrary low-rank prior. The exact numerical results are reported in Appendix F.

4.7 Sensitivity to the Mixing Coefficient

We further examine whether GramLoRA is sensitive to the mixing coefficient β_0. We sweep $\beta_0 \in \{0, 0.02, 0.05, 0.1, 0.2\}$ on Qwen2.5-7B under the main Commonsense170K setting.

As shown in Figure 4, setting $\beta_0 = 0$, which disables training-time trajectory guidance, yields an eight-task average of 84.6. All positive values improve over this initialization-only baseline, with averages ranging from 86.2 to 86.9. The default $\beta_0 = 0.05$ achieves the best result of 86.9, while the remaining positive settings stay within 0.7 points.

These results indicate that the benefit of trajectory guidance does not depend on a narrowly tuned mixing coefficient. We therefore use $\beta_0 = 0.05$ throughout the main experiments.

5 Conclusion

We introduced **GramLoRA**, which transfers optimization information from a smaller same-family model to guide large-model LoRA adaptation. The final source state provides task-informed initialization, while intermediate states provide stage-dependent guidance without changing the downstream objective. Across three model families, GramLoRA improves short-budget adaptation and generalizes to GLUE and GSM8K. Ablations show that initialization contributes most of the gain, with trajectory guidance providing additional improvements. The advantage decreases with more target-model training, suggesting that GramLoRA mainly accelerates early adaptation rather than improving the final attainable solution.

AI USE STATEMENT

Generative AI tools were used to assist with language polishing, manuscript organization, and presentation refinement. All methodological decisions, implementations, experiments, numerical results, and scientific conclusions were produced, reviewed, and verified by the authors. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

This work studies parameter-efficient adaptation of LLMs and does not involve human subjects, personally identifiable information, or the collection of private data. Our experiments use publicly available benchmarks and LLMs. GramLoRA is a general model-adaptation technique and may inherit the limitations, biases, and potentially harmful behaviors of the underlying pretrained models and training data. Improving adaptation efficiency may also lower the cost of modifying or specializing language models, which could enable both beneficial and undesirable downstream applications. We do not introduce mechanisms specifically intended to generate harmful content, bypass safety safeguards, or target particular individuals or groups. All experimental results are reported in accordance with the described settings, and the authors are responsible for the integrity and reproducibility of the reported findings.

REPRODUCIBILITY STATEMENT

We provide the information necessary to reproduce our experiments, including the datasets, source and target models, adapted modules, training budgets, optimization hyperparameters, trajectory checkpoints, and evaluation protocols. For the main Commonsense170K experiments, all methods use the same fixed training subset and matched target-model training schedules. We report the random seed, learning rate, batch size, gradient accumulation, LoRA rank and scaling factor, trajectory stages, mixing coefficient, and evaluation interval in Appendix A. Task-specific settings for GLUE and GSM8K are also documented in the appendix.

The construction of the source-model trajectory and its mapping to target-compatible low-rank factors are described in Section 3, including the initialization, compensation, and training-time guidance mechanisms. We additionally provide task-level results for extended training budgets, component ablations, module choices, hyperparameter sensitivity, cross-family transfer, and optimization dynamics in the appendix. Code and experiment configurations will be made available to facilitate reproduction of the reported results.

REFERENCES

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of ACL*, 2022.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7432–7439, 2020.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeff Wu. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, pp. 2924–2936, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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. *arXiv preprint arXiv:2110.14168*, 2021.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, 2023.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5:220–235, 2023.

Gemma Team. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. In *International Conference on Machine Learning*, 2024.

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 *International Conference on Machine Learning*, 2019.

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.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5254–5276, 2023.

Qiushi Huang, Tom Ko, Zhan Zhuang, Lilian Tang, and Yu Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *International Conference on Learning Representations*, 2025.

Feihu Jin, Jiajun Zhang, and Chengqing Zong. Parameter-efficient tuning for large language model without calculating its gradients. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 321–330, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.22. URL https://aclanthology.org/2023.emnlp-main.22/.

Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with LoRA. *arXiv preprint arXiv:2312.03732*, 2023.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of EMNLP*, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of ACL*, 2021.

Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. LoftQ: LoRA-fine-tuning-aware quantization for large language models. In *International Conference on Learning Representations*, 2024.

Vladislav Lialin, Namrata Shivagunde, Sherin Muckatira, and Anna Rumshisky. ReLoRA: High-rank training through low-rank updates. *arXiv preprint arXiv:2307.05695*, 2023.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*, 2022a.

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 *International Conference on Machine Learning*, 2024.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 61–68, 2022b. doi: 10.18653/v1/2022.acl-short.8. URL https://aclanthology.org/2022.acl-short.8/.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *Advances in Neural Information Processing Systems*, 2024.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018.

Mistral AI. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

Eric Mitchell, Rafael Rafailov, Archit Sharma, Chelsea Finn, and Christopher D Manning. An emulator for fine-tuning large language models using small language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Eo7kv0sllr.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of EACL*, 2021.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of EMNLP-IJCNLP*, pp. 4463–4473, 2019.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. DyLoRA: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*, 2023.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.

Hanqing Wang, Yixia Li, Shuo Wang, Guanhua Chen, and Yun Chen. MiLoRA: Harnessing minor singular components for parameter-efficient LLM finetuning. *arXiv preprint arXiv:2406.09044*, 2024a.

Jingyuan Wang, Yankai Chen, Zhonghang Li, and Chao Huang. LightReasoner: Can small language models teach large language models reasoning? In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2635–2663, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.122. URL https://aclanthology.org/2026.acl-long.122/.

Shaowen Wang, Linxi Yu, and Jian Li. LoRA-GA: Low-rank adaptation with gradient approximation. In *Advances in Neural Information Processing Systems*, 2024b.

Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. LoRA-pro: Are low-rank adapters properly optimized? In *International Conference on Learning Representations*, 2025.

Fei Xia, Mingyang Liao, Yuyang Fang, Defang Li, Yantong Xie, Weikang Li, Yang Li, Deguo Xia, and Jizhou Huang. Cross-LoRA: A data-free LoRA transfer framework across heterogeneous LLMs. *arXiv preprint arXiv:2508.05232*, 2025.

Yihao Xue, Jiping Li, and Baharan Mirzasoleiman. Representations shape weak-to-strong generalization: Theoretical insights and empirical predictions. In *International Conference on Machine Learning*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Jun Zhang, Jue WANG, Huan Li, Lidan Shou, Ke Chen, Yang You, Guiming Xie, Xuejian Gong, and Kunlong Zhou. Train small, infer large: Memory-efficient loRA training for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=s7DkcgpRxL.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023.

A Experimental Details

A.1 Datasets

Commonsense reasoning. We follow the Commonsense170K mixture of LLM-Adapters (Hu et al., 2023), containing 170,420 training pairs. Evaluation covers BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), ARC-Challenge (ARC-c) and ARC-Easy (ARC-e) (Clark et al., 2018), OpenBookQA (OBQA) (Mihaylov et al., 2018), HellaSwag (HellaS) (Zellers et al., 2019), and WinoGrande (WinoG) (Sakaguchi et al., 2021). We report accuracy on each task and their unweighted average. The 20% setting uses a fixed random subset shared across all methods.

Mathematical reasoning. For GSM8K (Cobbe et al., 2021), we use 20% of the training data for 0.5 epoch with rank $r = 32$, matching the main limited-budget commonsense setting.

Natural language understanding. We evaluate six GLUE (Wang et al., 2019) tasks on Qwen2.5-7B: SST-2, MRPC, CoLA, QNLI, RTE, and STS-B. Each training set is reduced to 20%. The public validation set is partitioned into validation and held-out test splits and is not subsampled. We report accuracy on SST-2, MRPC, QNLI, and RTE, Matthews correlation on CoLA, and Pearson correlation on STS-B.

A.2 Hyperparameters

Table 5 summarizes the default configuration for Commonsense170K and GSM8K. Table 6 reports the GLUE-specific settings. Unless otherwise specified, methods within the same comparison use the same data split, rank, target modules, optimizer, and training schedule.

Table 5: Default hyperparameters for Commonsense170K and GSM8K.

Hyperparameters	**Value**
Rank r	32
LoRA α	64
Target modules	Q, K, V, U, D
Optimizer	AdamW
Learning rate	3×10^{-4}
Learning-rate schedule	Linear decay, 100 warmup steps
Per-device batch size	8
Gradient accumulation	2 (effective batch size 16)
Precision	bfloat16
Seed	36
Evaluation interval	80 steps
Mixing coefficient β_0	0.05
Trajectory stages $\mathcal{T}$	$\{0.2, 0.4, 0.6, 0.8, 1.0\}$
Main budget	0.5 epoch, 20% data
Full-data short budget	0.5 epoch, 100% data

Table 6: GLUE hyperparameters on Qwen2.5-7B. All tasks use 20% of the training data and adapters on Q, V.

Hyperparameters	**SST-2**	**MRPC**	**CoLA**	**QNLI**	**RTE**	**STS-B**
Rank r	16					
Maximum sequence length	512					
Optimizer	AdamW					
Mixing coefficient β_0	0.05					
Epochs	8	10	10	8	12	16

Table 7: GSM8K accuracy on Qwen2.5-7B under the main limited-data setting (20% training data, 0.5 epoch, $r = 32$, $\alpha = 64$).

Methods	**GSM8K (%)**
LoRA	76.2
DoRA	77.4
HiRA	80.5
GramLoRA	**82.1**

The source models are Qwen2.5-0.5B for Qwen2.5-7B, Gemma4-E2B for Gemma4-E4B, and Ministral-3B for Ministral-8B. Source-model fine-tuning and target-compatible trajectory construction are performed offline; no source-model forward pass is required during target-model adaptation.

B Additional Task Results

B.1 Mathematical Reasoning on GSM8K

We evaluate GSM8K (Cobbe et al., 2021) on Qwen2.5-7B under the same limited target-model budget as the main commonsense experiment. As shown in Table 7, GramLoRA reaches 82.1% accuracy, outperforming HiRA by 1.6 points and LoRA by 5.9 points. These exact values are those reported in the current appendix.

B.2 GLUE Training Details

For GLUE, adapters are applied to the Q and V projections with rank $r = 16$ and maximum sequence length 512. We train for 8 epochs on SST-2, 10 on MRPC, 10 on CoLA, 8 on QNLI, 12 on RTE, and 16 on STS-B. The source model is Qwen2.5-0.5B and the default mixing coefficient is $\beta_0 = 0.05$. All methods use the same task-specific data splits and training schedules.

C Results under Larger Training Budgets

The main paper analyzes how the relative advantage of GramLoRA changes as target-model optimization increases. Here we report the complete 3-epoch results and the exact GramLoRA–HiRA differences across the three target-model training budgets.

C.1 Three-Epoch Full-Data Results

Table 8 reports Commonsense170K results after three full epochs. HiRA obtains the highest eight-task average on all three target models, with 88.4, 88.1, and 88.7 on Qwen2.5-7B, Gemma4-E4B, and Ministral-8B, respectively, while GramLoRA obtains 87.6, 87.3, and 87.8.

C.2 Relative Performance across Training Budgets

Table 9 provides the exact GramLoRA–HiRA differences corresponding to the budget analysis in Section 4.5. The differences change from positive under short training to negative after three full epochs.

Table 8: Commonsense reasoning after three full epochs on the complete training set. The best result in each column within each model block is shown in **bold**.

LLMs	Methods	BoolQ	PIQA	SIQA	ARC-c	ARC-e	OBQA	HellaS.	WinoG.	Avg.
Qwen2.5-7B	LoRA	63.4	84.1	74.1	82.7	89.8	85.4	89.5	81.1	81.2
	DoRA	69.1	87.9	77.7	86.8	93.8	88.4	93.2	84.2	85.1
	HiRA	72.8	**91.0**	**81.8**	**89.5**	**96.0**	**92.4**	**95.3**	88.3	**88.4**
	GramLoRA	**72.9**	88.3	81.2	87.9	95.8	90.6	95.1	**88.8**	87.6
Gemma4-E4B	LoRA	64.7	83.1	73.7	76.6	86.6	82.4	88.1	81.3	79.6
	DoRA	68.6	87.4	77.5	81.1	90.8	86.3	92.2	85.1	83.9
	HiRA	**73.7**	**91.0**	**81.7**	86.3	94.5	**93.0**	**95.6**	**89.0**	**88.1**
	GramLoRA	72.7	89.0	81.6	**87.5**	**95.7**	90.6	94.8	86.4	87.3
Ministral-8B	LoRA	65.4	84.5	75.0	83.9	90.6	86.0	89.5	80.7	81.9
	DoRA	72.9	87.9	79.9	87.8	94.0	89.0	93.5	85.8	86.3
	HiRA	**75.4**	**90.3**	**81.9**	**91.2**	**96.9**	92.2	94.4	87.5	**88.7**
	GramLoRA	69.9	90.1	**81.9**	89.0	96.2	**92.8**	**94.6**	**88.0**	87.8

Table 9: Signed difference in eight-task average accuracy between GramLoRA and HiRA. Positive values favor GramLoRA.

Training setting	Qwen2.5-7B	Gemma4-E4B	Ministral-8B
0.5 epoch, 20% data	$+2.2$	$+4.6$	$+2.9$
0.5 epoch, 100% data	$+0.3$	$+1.0$	$+0.3$
3 epochs, 100% data	-0.8	-0.8	-0.9

D Optimization Dynamics under a Limited Budget

Figure 5 compares the evaluation-loss trajectories of HiRA and GramLoRA on Qwen2.5-7B under the main limited-budget setting. Both methods are evaluated every 80 optimizer steps over 0.5 epoch.

GramLoRA obtains a lower evaluation loss at 10 of the 13 evaluation checkpoints and reduces the area under the evaluation-loss curve by 22.3%. This provides complementary evidence that the transferred source prior improves optimization efficiency under a limited target-model budget. We treat evaluation loss as an optimization diagnostic rather than as a direct substitute for downstream task accuracy.

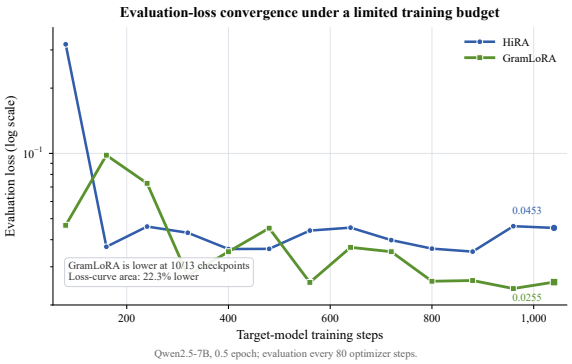


Figure 5: Evaluation-loss trajectories of HiRA and GramLoRA on Qwen2.5-7B under the main limited-budget setting. GramLoRA exhibits lower evaluation loss over most of the training trajectory.

E Additional Ablations

E.1 Detailed Component Ablation

Section 4.4 reports the average performance of the main trajectory ablations. Table 10 provides the complete task-level results.

Source-informed initialization improves the eight-task average from 78.4 to 84.6. Final-state guidance and shuffled trajectory guidance further increase the average to 86.3 and 86.4, respectively, while stage-matched GramLoRA reaches 86.9.

Table 10: Detailed component ablation on Qwen2.5-7B under the main limited-data setting.

Methods	**BoolQ**	**PIQA**	**SIQA**	**ARC-c**	**ARC-e**	**OBQA**	**HellaS.**	**WinoG.**	**Avg.**
LoRA	63.4	82.3	72.4	82.1	88.4	82.0	85.8	70.8	78.4
w/o mixing (init only)	68.0	86.3	78.8	88.1	95.8	88.6	91.9	79.4	84.6
w/o compensation	69.6	86.6	79.7	89.2	95.9	89.4	92.1	82.5	85.6
Final-state guidance	69.7	87.7	80.3	89.3	95.7	90.5	91.7	**85.3**	86.3
Shuffled trajectory	70.1	**88.7**	79.9	**89.8**	**96.3**	88.2	**93.3**	84.9	86.4
GramLoRA	**71.8**	**88.7**	**80.7**	89.0	**96.3**	**91.4**	92.6	84.9	**86.9**

E.2 Effect of Identity-Preserving Compensation

GramLoRA initializes the LoRA factors from the final projected source state and compensates the frozen base weight as

$$W_0 \leftarrow W_0 - sB_{1.0}A_{1.0},$$

so that introducing non-zero LoRA factors does not change the pretrained model's effective weight at step zero.

Removing this compensation reduces the eight-task average from 86.9 to 85.6. Without compensation, the projected source update is directly injected into the initial forward map. Thus, this ablation evaluates the role of preserving the pretrained function rather than removing trajectory guidance itself.

E.3 Choice of Adapted Modules

We compare attention-only (Q, K, V), MLP-only (U, D), and combined adaptation on Qwen2.5-7B using the full Commonsense170K training set for 0.5 epoch. GramLoRA remains strong under all three configurations, with eight-task averages of 87.3, 88.0, and 87.3, respectively.

Table 11: Module ablation on Qwen2.5-7B using the full training set for 0.5 epoch.

Methods	Modules	BoolQ	PIQA	SIQA	ARC-c	ARC-e	OBQA	HellaS.	WinoG.	Avg.
LoRA	Q, K, V	65.4	82.2	73.0	82.4	89.3	82.4	86.9	75.0	79.6
	U, D	59.2	83.2	74.4	84.6	89.8	85.4	87.2	80.0	80.5
	All	65.9	83.3	74.0	83.8	90.3	85.2	88.3	79.2	81.3
GramLoRA	Q, K, V	70.5	89.5	80.6	88.7	**96.0**	90.8	**94.8**	87.4	87.3
	U, D	**73.4**	**89.7**	**81.7**	87.9	**96.0**	**92.0**	94.7	**88.8**	**88.0**
	All	72.8	88.2	81.2	86.9	95.8	90.2	94.6	88.6	87.3

E.4 Detailed Mixing-Coefficient Results

Table 12 reports the task-level results corresponding to the sensitivity analysis in Section 4.7. The average ranges from 86.2 to 86.9 for all positive β_0 values tested, while $\beta_0 = 0$ recovers the initialization-only result of 84.6.

Table 12: Sensitivity to the mixing coefficient β_0 on Qwen2.5-7B under the main limited-data setting.

β_0	BoolQ	PIQA	SIQA	ARC-c	ARC-e	OBQA	HellaS.	WinoG.	Avg.
0	68.0	86.3	78.8	88.1	95.8	88.6	91.9	79.4	84.6
0.02	70.6	88.0	80.1	89.2	95.9	89.6	92.5	84.1	86.2
0.05	71.8	88.7	80.7	89.0	96.3	91.4	92.6	84.9	**86.9**
0.1	70.7	87.3	79.6	89.3	96.1	89.8	92.9	84.3	86.3
0.2	70.4	88.6	79.9	88.7	96.4	90.0	93.7	85.6	86.7

F Cross-Family Transfer Details

Figure 3 in the main paper visualizes trajectory transfer across source–target model families. Table 13 reports the corresponding exact values.

For all three target models, the strongest performance is obtained with the same-family source model. Cross-family transfer remains possible but degrades substantially for several source–target pairs, indicating that the transferred prior depends strongly on source–target compatibility.

Table 13: Cross-family trajectory transfer under the main limited-data setting. Rows denote source models and columns denote target models. Same-family results are shown in bold.

Source \ Target	Qwen2.5-7B	Gemma4-E4B	Ministral-8B
Qwen2.5-0.5B	**86.9**	68.6	76.5
Gemma4-E2B	72.8	**84.0**	71.8
Ministral-3B	78.9	72.0	**86.6**