TRIBLORA: TRI-FACTOR BLOCKWISE LOW-RANK ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-Rank Adaptation (LoRA) enables parameter-efficient fine-tuning but couples input–output interactions through a common low-dimensional latent structure. Motivated by evidence of functional differentiation and importance heterogeneity within individual layers, we introduce TriBLoRA, a tri-factor block-wise parameterization that partitions the LoRA factors into local input and output blocks and inserts lightweight, dense pair-specific interaction matrices. The shared outer factors define reusable shared local subspaces, while each interaction matrix provides a pair-specific latent transformation. Beyond the core tri-factor parameterization, TriBLoRA also supports non-uniform block-wise rank allocation, enabling adaptation capacity to be redistributed across local regions under a controlled budget. We characterize its feasible update families and approximation properties and analyze when finer block-wise rank reallocation can lower approximation error. Experiments with RoBERTa-base on GLUE and LLaMA-2-7B instruction tuning show improved aggregate performance over rank-8 LoRA with modest parameter overhead, and further gains under appropriately configured block-rank topologies.

1 INTRODUCTION

For a weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, LoRA writes $\Delta W = BA$, where $B \in \mathbb{R}^{d_{\text{out}} \times r}$ and $A \in \mathbb{R}^{r \times d_{\text{in}}}$. This factorization reduces trainable parameters by sharing the same r-dimensional latent factors across all input–output interactions. The common latent structure couples these interactions and globally restricts the update to $\text{rank}(\Delta W) \leq r$. Increasing r relaxes this bottleneck but expands both factors globally, making capacity allocation coarse under tight parameter budgets.

Recent work expands a rank-r LoRA update without proportionally enlarging its shared rank. ReLoRA (Lialin et al., 2024) and COLA (Xia et al., 2024) sequentially accumulate low-rank updates beyond the rank-r ceiling, but require additional multi-stage training. MELoRA (Ren et al., 2025) instead uses n independent block-diagonal rank-r mini-LoRAs to reach rank nr, while BoRA (Li et al., 2025) restores cross-block interactions through pair-specific diagonal modulation under the same ceiling. Thus, capacity can be expanded through sequential accumulation or structural reparameterization.

However, sharing the global rank ceiling nr does not imply the expressive power of rank-nr LoRA: MELoRA removes off-diagonal interactions, whereas BoRA restores them but restricts each pair-specific transformation to diagonal modulation. More fundamentally, LoRA-style sharing efficiently captures reusable structure but may underrepresent interaction-specific variation. Prior analyses have provided evidence of *functional heterogeneity* among components within individual layers (Bau et al., 2017; Fong & Vedaldi, 2018; Voita et al., 2019; Papyan et al., 2020), suggesting that local interactions may share common structure while requiring distinct transformations. This motivates our first question: *can reusable shared structure and pair-specific latent transformations be explicitly separated while retaining LoRA's parameter efficiency?*

A complementary consideration is that adaptation demand may vary within a layer. AdaLoRA (Zhang et al., 2023b) shows that layers can benefit from different rank budgets, while attention-head and fine-grained pruning studies reveal substantial importance differences within individual layers (Michel et al., 2019; Lee et al., 2018; Sanh et al., 2020). We refer to such uneven local adaptation demands as *importance heterogeneity*: different intra-layer regions need not require the same capacity. Yet conventional LoRA has no explicit input- and output-side intra-layer units with

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.

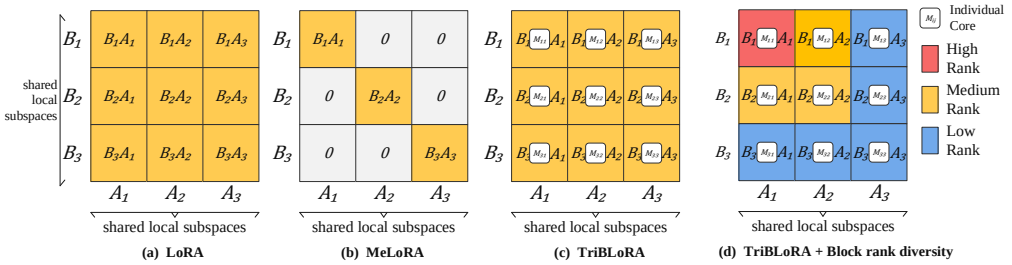

Figure 1: Progression from globally shared low-rank adaptation to pair-specific interactions and block-rank diversity. (a) uses globally shared local factors; (b) retains only block-diagonal interactions; (c) inserts pair-specific interaction matrices M_{ij} between shared outer factors to combine shared local subspaces with pair-specific latent transformations; and (d) redistributes local rank across input- and output-side regions under a controlled budget.

independently adjustable ranks, and existing adaptive-rank methods primarily redistribute capacity across layers or modules. This leads to our second question: *can adaptation capacity be allocated at a finer intra-layer granularity while preserving a controlled parameter budget?*

Motivated by these two forms of intra-layer heterogeneity, we propose TRIBLORA, a tri-factor block-wise low-rank adaptation framework. Specifically, we partition the original LoRA factors into local blocks $\{B_i\}$ and $\{A_j\}$ and parameterize each input–output interaction as

$$\Delta W_{ij} = B_i M_{ij} A_j. \tag{1}$$

The outer factors are reused across interaction pairs, while each lightweight M_{ij} provides a pair-specific transformation. This separates reusable shared structure from individual variation, relaxing cross-interaction coupling while retaining substantial sharing. Building on this core tri-factor structure, we further consider *block-rank diversity*, i.e., non-uniform local ranks across intra-layer input- and output-side regions, redistributing a fixed parameter budget toward more important regions rather than increasing rank uniformly.

Our contributions are threefold. (1) Motivated by *functional heterogeneity* within individual layers, we introduce a tri-factor structure that combines reusable shared local subspaces with lightweight pair-specific latent transformations, increasing interaction specificity while retaining substantial sharing. (2) Building on the core tri-factor structure, we further introduce *block-rank diversity*, which allows input- and output-side local ranks to vary under a controlled budget, redistributing capacity across local regions rather than increasing it uniformly. (3) We characterize the resulting feasible families and approximation properties relative to LoRA and representative block-structured methods, analyze when finer block-wise rank allocation improves approximation, and evaluate both structural effects across multiple tasks and random seeds.

2 RELATED WORK

Expressive LoRA variants and intra-layer structure. A broad range of LoRA variants improve expressiveness by modifying the update structure, including sequential accumulation (Lialin et al., 2024; Xia et al., 2024), high-capacity parameterizations such as MoRA (Jiang et al., 2024) and HiRA (Huang et al., 2025), and alternative factorization or sharing schemes including LoRA-XS (Bałazy et al., 2024), DoRA (Liu et al., 2024), HydraLoRA (Tian et al., 2024), and TopLoRA (Li et al., 2026). More closely related to our formulation, MELoRA (Ren et al., 2025) and BoRA (Li et al., 2025) reorganize intra-layer interactions through spatially separated updates and pair-specific diagonal modulation, respectively. MoSLoRA (Wu et al., 2024) introduces a trainable global dense middle matrix between LoRA factors, whereas TLoRA (Lin et al., 2026) concentrates adaptation in a trainable middle core between fixed outer projections. TriBLoRA differs by combining trainable shared local outer factors with dense pair-specific interaction matrices, thereby retaining cross-block interactions while allowing each pair to learn a general transformation within the shared local subspaces. Meanwhile, prior analyses of neural networks have provided evidence of *functional heterogeneity* among components within individual layers (Bau et al., 2017; Fong & Vedaldi, 2018; Voita

et al., 2019; Papyan et al., 2020), suggesting that finer-grained intra-layer structure may benefit from distinguishing reusable and interaction-specific components. TriBLoRA follows this perspective by combining shared local subspaces with general pair-specific transformations.

Adaptive rank allocation and intra-layer importance heterogeneity. A complementary line of work addresses *importance heterogeneity* by allocating adaptation capacity non-uniformly. AdaLoRA (Zhang et al., 2023b), DyLoRA (Valipour et al., 2023), SoRA (Ding et al., 2023), and IncreLoRA (Zhang et al., 2023a) explore adaptive rank allocation through importance estimation, rank sampling, pruning, or incremental growth, while CSPLoRA (Ding et al., 2026) and GoRA (He et al., 2025) perform one-shot rank planning from pre-training gradient signals before the main adaptation process. At a finer granularity, pruning studies have shown substantial importance differences among components even within the same layer (Michel et al., 2019; Lee et al., 2018; Sanh et al., 2020). Existing adaptive-rank LoRA methods nevertheless allocate capacity primarily across layers or modules. Our formulation extends this granularity to explicit intra-layer input- and output-side units whose capacities can vary across local regions.

Our position. TriBLoRA addresses these two forms of intra-layer heterogeneity within a single block-wise parameterization: pair-specific transformations provide flexibility for functional heterogeneity, while heterogeneous local ranks address importance heterogeneity.

3 METHODOLOGY

Our method is motivated by a simple observation: LoRA gains parameter efficiency through extensive sharing, which captures reusable structure but may underrepresent interaction-specific variation. We therefore introduce a structured decomposition that preserves shared local subspaces while retaining lightweight pair-specific latent transformations. This section introduces the block-structured notation, the TriBLoRA parameterization, and its extension to block-rank diversity.

3.1 PROBLEM SETUP AND UNIFIED NOTATION

Consider a pretrained linear transformation $W^{(0)} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$. Standard LoRA parameterizes its adaptation as $\Delta W = BA$, where $B \in \mathbb{R}^{d_{\text{out}} \times r}$ and $A \in \mathbb{R}^{r \times d_{\text{in}}}$.

To expose the local interaction structure, we partition the output dimension into G blocks with sizes $\{d_i\}_{i=1}^{G}$ and the input dimension into H blocks with sizes $\{c_j\}_{j=1}^{H}$, where $\sum_i d_i = d_{\text{out}}$ and $\sum_j c_j = d_{\text{in}}$. Accordingly, $B = [B_1^\top, \dots, B_G^\top]^\top, A = [A_1, \dots, A_H]$, with $B_i \in \mathbb{R}^{d_i \times r}$ and $A_j \in \mathbb{R}^{r \times c_j}$. The LoRA update can then be written block-wise as

$$\Delta W^{\text{LoRA}} = \begin{bmatrix} B_1A_1 & \cdots & B_1A_H \\ \vdots & \ddots & \vdots \\ B_GA_1 & \cdots & B_GA_H \end{bmatrix} = \begin{bmatrix} \Delta W_{11}^{\text{LoRA}} & \cdots & \Delta W_{1H}^{\text{LoRA}} \\ \vdots & \ddots & \vdots \\ \Delta W_{G1}^{\text{LoRA}} & \cdots & \Delta W_{GH}^{\text{LoRA}} \end{bmatrix}, \Delta W_{ij}^{\text{LoRA}} = B_iA_j. \quad (2)$$

This form makes the parameter sharing explicit: B_i and A_j are reused across all input blocks j and output blocks i, respectively. Such reuse gives LoRA its parameter efficiency, but also couples local interactions because no pair (i, j) has dedicated transformation parameters. Moreover, although each local block may have rank up to r, the complete update still satisfies $\text{rank}(\Delta W^{\text{LoRA}}) \leq r$.

For the theoretical analysis in Section 4, we specialize to the square, equally partitioned setting $d_{\text{out}} = d_{\text{in}} = d, G = H = n, d_i = c_j = m = \frac{d}{n}$. We denote the target adaptation by $\Delta W^\star = [\Delta W_{ij}^\star]_{i,j=1}^n$ and measure approximation quality using the squared Frobenius error

$$E(\Delta W^\star, \Delta W) := \|\Delta W^\star - \Delta W\|_F^2. \quad (3)$$

3.2 SHARED–INDIVIDUAL TRI-FACTOR ADAPTATION

Motivated by prior evidence of intra-layer *functional heterogeneity*, we aim to separate reusable adaptation structure from components that should remain specific to individual interaction pairs. We refer to these two roles as shared local subspaces and pair-specific latent transformations, respectively.

Partitioning the standard LoRA factors as $B = [B_1^\top, \ldots, B_G^\top]^\top$ and $A = [A_1, \ldots, A_H]$, its (i, j)-th interaction is $\Delta W_{ij}^{\mathrm{LoRA}} = B_i A_j = B_i I_r A_j$, so all interaction pairs share the same latent connector. TriBLoRA instead introduces a pair-specific interaction matrix $M_{ij} \in \mathbb{R}^{r \times r}$,

$$\Delta W_{ij} = B_i M_{ij} A_j, \tag{4}$$

The outer factors $\{B_i, A_j\}$ form the *shared* component: B_i is reused across all interactions associated with output region i, while A_j is reused across those associated with input region j. They therefore define reusable local subspaces. In contrast, M_{ij} is specific to pair (i, j) and provides a pair-specific latent transformation within these shared subspaces. Thus, TriBLoRA separates reusable structure from interaction-specific variation while retaining substantial parameter sharing.

Moreover, the full update can be written compactly as $\Delta W = \mathcal{B}\mathcal{M}\mathcal{A}$, where $\mathcal{B} := \mathrm{blkdiag}(B_1, \ldots, B_G)$, $\mathcal{A} := \mathrm{blkdiag}(A_1, \ldots, A_H)$, and $\mathcal{M} := [M_{ij}]$. Accordingly, $\mathcal{B} \in \mathbb{R}^{d_{\mathrm{out}} \times Gr}$, $\mathcal{M} \in \mathbb{R}^{Gr \times Hr}$, and $\mathcal{A} \in \mathbb{R}^{Hr \times d_{\mathrm{in}}}$. Folding the first two factors gives $\Delta W = (\mathcal{B}\mathcal{M})\mathcal{A}$ and hence $\mathrm{rank}(\Delta W) \le \min(Gr, Hr)$. Thus, for $G = H = n$, TriBLoRA has a global rank ceiling of nr and can be viewed as a structured special case of rank-nr LoRA.

The total number of trainable parameters is

$$N_{\mathrm{param}} = r(d_{\mathrm{out}} + d_{\mathrm{in}}) + GHr^2. \tag{5}$$

The first term counts the shared outer factors, whereas GHr^2 is the additional cost of the pair-specific interaction matrices. In the uniform square setting with $G = H = n$ and $d_{\mathrm{out}} = d_{\mathrm{in}} = d$, this becomes $N_{\mathrm{param}} = 2dr + n^2 r^2$, compared with $2dr$ for standard rank-r LoRA. For the small block counts and low local ranks considered in our setting, this additional cost remains modest relative to the shared projection factors. These additional parameters are devoted to pair-specific transformations, while the larger outer projection factors remain shared across multiple interactions.

3.3 Block-Rank Diversity

Section 3.2 assumes a common local rank r, assigning the same nominal adaptation capacity to all intra-layer regions. Under *intra-layer importance heterogeneity*, however, different local regions may have different adaptation demands, making a uniform rank allocation unnecessarily restrictive. TriBLoRA naturally exposes local input- and output-side capacity units, allowing this variation to be modeled at a finer granularity.

Let S_{ij} denote the importance of block interaction (i, j). Following the sharing pattern of the tri-factor structure, the interaction-level importance induces corresponding output- and input-side scores

$$S_B(i) = \sum_j S_{ij}, \qquad S_A(j) = \sum_i S_{ij}. \tag{6}$$

Here, $S_B(i)$ summarizes the adaptation demand associated with output-side region i across its input interactions, while $S_A(j)$ summarizes the corresponding demand of input-side region j across its output interactions.

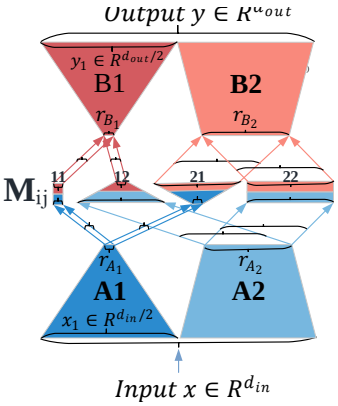

Figure 2: Two-sided block-rank allocation in TriBLoRA.

Motivated by these heterogeneous local demands, we relax the uniform-rank assumption and allow $B_i \in \mathbb{R}^{d_i \times r_{B,i}}$, $A_j \in \mathbb{R}^{r_{A,j} \times c_j}$, and $M_{ij} \in \mathbb{R}^{r_{B,i} \times r_{A,j}}$. Figure 2 illustrates this two-sided generalization: higher-demand regions may receive ranks above r, whereas others release capacity, while the underlying pair-specific tri-factor structure is preserved.

We denote the resulting block-rank topology by

$$\mathcal{T} := (r_{B,1}, \ldots, r_{B,G}; r_{A,1}, \ldots, r_{A,H}). \tag{7}$$

Block-rank diversity therefore redistributes adaptation capacity across intra-layer regions rather than uniformly increasing the rank of the entire layer. The shared-topology and adaptive-degree protocol is provided in Appendix B.4. Two-sided rank coordination is discussed in Appendix A.4.3, and exact budget preservation is derived in Appendix A.4.2.

4 THEORETICAL ANALYSIS

We use the equally partitioned square setting of Section 3.1, with $d_{\text{out}} = d_{\text{in}} = d$, $G = H = n$, $m = d/n$, and local rank $r \leq m$. We analyze representational expressiveness, independently of optimization and generalization, and then consider block-rank diversity as an extension. Throughout this section, BoRA denotes its relaxed family with arbitrary pair-specific diagonal connectors, conservatively enlarging its implemented feasible family.

4.1 SHARED-SUBSPACE AND PAIR-SPECIFIC DECOMPOSITION

For full-rank local factors, write $B_i = Q_i R_i$ and $A_j = S_j P_j^\top$, where $Q_i, P_j \in \mathbb{R}^{m\times r}$ have orthonormal columns and R_i, S_j are invertible. Defining $Z_{ij} := R_i M_{ij} S_j$ gives

$$\widehat{\Delta W}_{ij} = B_i M_{ij} A_j = Q_i Z_{ij} P_j^\top. \tag{8}$$

Here, Q_i is shared across input blocks, P_j across output blocks, and Z_{ij} specifies the transformation for pair (i, j). For fixed bases, let $C_{ij} := Q_i^\top \Delta W_{ij}^\star P_j$. Since Z_{ij} is unrestricted, its optimum is $Z_{ij}^\star = C_{ij}$, yielding

$$E_{\text{TriB}}(Q, P) = \|\Delta W^\star\|_F^2 - \sum_{i,j} \|Q_i^\top \Delta W_{ij}^\star P_j\|_F^2. \tag{9}$$

This provides the structural motivation for TriBLoRA: each outer basis is optimized jointly over the interactions sharing its local region, capturing reusable local subspaces, while each Z_{ij} retains pair-specific freedom within them. The separation holds at the subspace level, since invertible changes of latent coordinates can be absorbed into the cores. Thin QR decomposition and orthogonal projection establish these results; the complete derivation is given in Appendix A.2.

4.2 EXPRESSIVE HIERARCHY AND APPROXIMATION ANALYSIS

Let $\mathcal{F}_{\mathcal{S}}$ denote the feasible update family of structure $\mathcal{S}$, with $\mathcal{F}_{\text{L}}(k) := \{X \in \mathbb{R}^{d\times d} : \text{rank}(X) \leq k\}$ and $E_{\mathcal{F}}^\star := \min_{X\in\mathcal{F}} \|\Delta W^\star - X\|_F^2$.

Proposition 1 (Expressive-family hierarchy)**.** *Under the same partition and local rank,*

$$\mathcal{F}_{\text{L}}(r) \cup \mathcal{F}_{\text{ME}}(n, r) \subseteq \mathcal{F}_{\text{BoRA}}(n, r) \subseteq \mathcal{F}_{\text{TriB}}(n, r) \subseteq \mathcal{F}_{\text{L}}(nr). \tag{10}$$

Consequently, $E_{\text{L},nr}^\star \leq E_{\text{TriB}}^\star \leq E_{\text{BoRA}}^\star \leq \min\{E_{\text{L},r}^\star, E_{\text{ME}}^\star\}$.

LoRA and MELoRA are recovered through restricted diagonal connectors, and relaxed BoRA by restricting TriBLoRA's cores to diagonal matrices; folding $\mathbf{BMA}$ gives the rank-nr upper bound (Appendix A.3.1). The inclusions need not be strict: when $n = 1$, all families reduce to rank-r updates; when $r = 1$, dense and diagonal cores coincide.

Proposition 2 (Approximation effect of interaction constraints)**.** *For fixed local bases, let* $\mathbf{C} = [C_{ij}] \in \mathbb{R}^{nr\times nr}$*. Rank-$r$ LoRA restricts the latent interaction to rank r, whereas TriBLoRA represents the full* $\mathbf{C}$*, giving*

$$E_{\text{L}}(Q, P) - E_{\text{TriB}}(Q, P) = \sum_{k=r+1}^{nr} \sigma_k^2(\mathbf{C}). \tag{11}$$

For each $\mathcal{S} \in \{\text{ME}, \text{BoRA}\}$*, take an optimal baseline solution with full-rank outer factors. Define* $E_{\text{TriB,match}}^{\mathcal{S}}$ *as the minimum error obtained by keeping its local bases* $\{Q_i^{\mathcal{S}}, P_j^{\mathcal{S}}\}$ *fixed and optimizing unrestricted cores, and set* $C_{ij}^{\mathcal{S}} := (Q_i^{\mathcal{S}})^\top \Delta W_{ij}^\star P_j^{\mathcal{S}}$*. Then*

$$E_{\text{ME}}^\star - E_{\text{TriB,match}}^{\text{ME}} = \sum_{i\neq j} \|C_{ij}^{\text{ME}}\|_F^2, \qquad E_{\text{BoRA}}^\star - E_{\text{TriB,match}}^{\text{BoRA}} = \sum_{i,j} \delta_{ij}^{\text{BoRA}}, \tag{12}$$

where $\delta_{ij}^{\text{BoRA}}$ *is the minimum squared residual when approximating* C_{ij}^{BoRA} *by* $R_i^{\text{BoRA}} D_{ij} S_j^{\text{BoRA}}$ *with diagonal* D_{ij}*.*

Interpretation. These gaps isolate the interaction constraints: LoRA imposes a global latent-rank bottleneck, MELoRA removes off-diagonal block interactions, and BoRA retains these interactions but restricts their pair-specific transformations. TriBLoRA removes these restrictions within the shared local subspaces. The matched gains are strictly positive when the respective residual terms are nonzero.

Remaining shared-subspace constraint. Every complete TriBLoRA block row and column has rank at most r. Define the target block-row residual energy as $\epsilon_{\text{out}} := \sum_i \sum_{k>r} \sigma_k^2(\Delta W^\star_{i,:})$ and the block-column residual energy as $\epsilon_{\text{in}} := \sum_j \sum_{k>r} \sigma_k^2(\Delta W^\star_{:,j})$. Then

$$\max\{\epsilon_{\text{out}}, \epsilon_{\text{in}}\} \le E^\star_{\text{TriB}} \le \epsilon_{\text{out}} + \epsilon_{\text{in}}. \tag{13}$$

Unrestricted rank-nr LoRA instead achieves $E^\star_{\text{L},nr} = \sum_{k>nr} \sigma_k^2(\Delta W^\star)$. Thus, a shared global rank ceiling does not imply equivalent expressiveness; TriBLoRA retains local-subspace constraints absent from rank-nr LoRA.

Proof sketch. The Eckart–Young–Mirsky theorem gives Eq. (11). Fixing the baseline factors and relaxing their interaction constraints gives Eq. (12); see Appendix A.3.3. Local rank constraints and joint projection give Eq. (13); the full derivation, including the resulting gap to rank-nr LoRA, is in Appendix A.3.2.

4.3 Block-Rank Topology

Building on heterogeneous-rank analysis (Zeng & Lee, 2024), we consider finer intra-layer allocation using an idealized additive low-rank approximation surrogate with coordinated side ranks and equal rank-unit costs. This isolates allocation granularity from the interaction-core cost and row/column coupling of the actual topology.

Proposition 3 (Block-rank allocation under a fixed budget)**.** *Let $r_{\ell,b}$ be the effective rank of local unit b in layer ℓ. Under effective-rank budget K, the layer-wise and block-wise topology spaces satisfy $\mathcal{T}_{\text{layer}}(K) \subseteq \mathcal{T}_{\text{block}}(K)$, hence $\mathcal{E}^\star_{\text{block}}(K) \le \mathcal{E}^\star_{\text{layer}}(K)$. For two units a, b in the same layer with common rank r_ℓ, a feasible transfer of one rank from b to a changes the surrogate error by*

$$\Delta\mathcal{E} = -\sigma^2_{r_\ell+1}(\Delta W^\star_a) + \sigma^2_{r_\ell}(\Delta W^\star_b). \tag{14}$$

It is strictly beneficial when $\sigma^2_{r_\ell+1}(\Delta W^\star_a) > \sigma^2_{r_\ell}(\Delta W^\star_b)$.

For the actual equal-block TriBLoRA topology, preserving $\sum_i r_{B,i} = \sum_j r_{A,j} = nr$ keeps the exact parameter count $N_{\text{param}} = 2dr + n^2r^2$ unchanged. However, $\operatorname{rank}(B_i M_{ij} A_j) \le \min(r_{B,i}, r_{A,j})$, and the mean pairwise rank ceiling satisfies $\bar{\rho}_{\text{cap}} := n^{-2} \sum_{i,j} \min(r_{B,i}, r_{A,j}) \le r$.

Interpretation and proof. Finer allocation can reduce the surrogate error under heterogeneous marginal spectral gains. In the actual parameterization, it preserves the budget under the stated constraints but can lower the mean pairwise rank ceiling. Useful topologies therefore balance capacity placement against two-sided rank mismatch. Set inclusion and the Eckart–Young–Mirsky theorem establish the surrogate result; factoring the core cost as $(\sum_i r_{B,i})(\sum_j r_{A,j})$ and using $\min(a, b) \le (a + b)/2$ give the budget and rank-ceiling results. Complete derivations are in Appendices A.4, A.4.2, and A.4.3.

5 Experiments

We organize the experiments around two questions that correspond directly to the two design components of TriBLoRA. First, does the pair-specific dense interaction matrix improve a uniform block-structured adapter? Second, we investigate whether block-rank diversity can improve adaptation by redistributing a fixed adaptation budget across intra-layer regions according to task-dependent demands. Experimental settings are summarized in Appendix B, and additional experiments and analyses are provided in Appendix C.

Table 1: Main comparisons on (a) GLUE and (b) LLaMA-2-7B. * HydraLoRA $r = 3$; $^\diamond$ $r = 4$; * CoLA includes one collapsed seed;

		Method	Params/matrix	MNLI (%)	SST-2 (%)	MRPC (%)	CoLA (%)	QNLI (%)	QQP (%)	RTE (%)	STS-B (%)	Avg. (%)
(a) GLUE	General	LoRA	12,288	87.25±0.06	94.65±0.37	90.28±0.49	61.63±2.25	92.78±0.01	89.28±0.06	75.81±1.25	90.62±0.18	85.29
		AdaLoRA	18,444	87.21±0.09	94.61±0.50	88.53±0.42	58.98±1.37	92.78±0.20	88.78±0.07	60.87±1.89	90.20±0.10	82.75
		HydraLoRA*	11,520	87.05±0.01	94.65±0.43	90.79±0.92	61.89±0.45	92.61±0.12	88.95±0.05	74.84±1.65	90.43±0.17	85.15
		HydraLoRA$^\diamond$	14,592	87.25±0.25	94.69±0.46	91.11±0.29	**62.39**±1.08	92.71±0.07	89.16±0.11	75.09±1.33	90.34±0.10	85.34
	Block	MELoRA	12,288	87.13±0.09	94.53±0.29	89.29±0.85	59.47±0.60	**93.01**±0.07	88.96±0.07	73.18±1.28	90.46±0.21	84.50
		BoRA	12,416	87.24±0.02	94.61±0.20	**91.22**±0.58	49.58±27.75*	92.53±0.30	**89.33**±0.14	72.89±7.29	90.30±0.67	83.46
		TriBLoRA	13,312	87.23±0.22	**94.72**±0.46	90.72±0.45	62.02±0.69	92.64±0.22	89.27±0.08	**76.93**±0.82	**90.73**±0.17	**85.53**
	Tri-F.	TLoRA	12,353	72.26±0.78	90.79±0.18	79.96±0.70	39.28±3.11	84.72±0.11	80.22±0.24	58.77±0.78	83.61±0.22	73.70
		MoSLoRA	12,352	87.13±0.08	94.46±0.40	90.78±0.81	61.03±1.71	92.80±0.18	88.97±0.12	75.34±1.19	90.59±0.17	85.14
		TriBLoRA (K-F)	13,312	**87.29**±0.05	**94.72**±0.41	91.15±0.49	62.19±0.64	92.68±0.05	89.24±0.17	75.85±1.45	90.70±0.24	85.48

		Method	Params/matrix	Best Eval Loss ↓	MMLU (%)	BBH (%)	DROP (%)	WinoGrande (%)	ARC-C (%)	HumanEval (%)	Avg. (6 tasks) (%)
(b) LLaMA-2-7B		Base LLaMA-2-7B	0	–	42.48	31.99	31.52	69.38	41.38	13.17	38.32
	General	LoRA	65,536	0.8311	43.54±0.50	32.94±0.45	32.44±0.17	70.24±0.24	42.53±1.13	15.37±0.38	39.51
		AdaLoRA	98,316	0.8818	42.65±0.17	33.04±0.45	32.01±0.28	69.66±0.62	35.84±1.12	15.10±0.57	38.05
		HydraLoRA*	61,440	0.8321	43.15±0.41	32.83±0.12	31.93±0.69	69.98±0.38	40.12±1.53	14.66±0.68	38.78
		HydraLoRA$^\diamond$	77,824	0.8326	43.57±0.19	33.44±0.17	32.35±0.33	70.23±0.55	41.79±1.64	14.76±0.59	39.36
	Block	MELoRA	65,536	0.8345	43.97±0.20	32.83±0.45	32.46±0.15	70.26±0.12	43.34±0.41	**15.41**±0.39	39.71
		BoRA	65,664	0.8351	43.63±0.29	33.09±0.33	**32.87**±0.59	69.87±0.31	42.24±1.12	14.90±0.22	39.43
		TriBLoRA	66,560	0.8333	**44.18**±0.65	**33.55**±0.34	32.53±0.65	**70.54**±0.58	44.57±1.88	15.00±0.73	**40.06**
	Tri-F.	TLoRA	65,601	1.3083	39.97±0.45	32.71±0.75	30.62±1.25	69.34±0.36	39.47±1.51	12.32±0.73	37.40
		MoSLoRA	65,600	0.8329	43.50±0.32	33.09±0.07	32.35±0.32	70.37±0.23	41.25±1.22	14.56±0.58	39.19
		TriBLoRA (K-F)	66,560	**0.8307**	43.98±0.20	33.17±0.33	32.46±0.23	70.20±0.42	**44.86**±0.77	15.34±0.70	40.00

Benchmarks. We evaluate RoBERTa-base (Liu et al., 2019) on the eight GLUE tasks (Wang et al., 2019): MNLI, SST-2, MRPC, CoLA, QNLI, QQP, RTE, and STS-B, adapting the query and value projections. We additionally instruction-tune LLaMA-2-7B (Touvron et al., 2023) on Alpaca-cleaned (Taori et al., 2023; Yahma, 2023) and evaluate the resulting model on MMLU (Hendrycks et al., 2021), BBH (Suzgun et al., 2022), DROP (Dua et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC-Challenge (Clark et al., 2018), and HumanEval (Chen et al., 2021). Rectangular-identity initialization is the default for TriBLoRA. To provide an additional comparison with tri-factor baselines that use dense random middle-core initialization, we also evaluate a Kaiming–Frobenius (K-F) variant with scale-matched dense interaction cores. Its corresponding optimization setting is described in Appendix B.3. We report mean and sample standard deviation across random seeds, with the best and second-best task means shown in bold and underlined.

5.1 Main Results

Baselines. We compare TriBLoRA with LoRA (Hu et al., 2022), AdaLoRA (Zhang et al., 2023b), HydraLoRA (Tian et al., 2024), MELoRA (Ren et al., 2025), BoRA (Li et al., 2025), MoSLoRA (**?**), and TLoRA (Lin et al., 2026). Unless noted, all methods use rank 8; HydraLoRA uses three experts at $r = 3, 4$, while MELoRA, BoRA, and TriBLoRA use four blocks. BoRA and MoSLoRA are the closest structural references, using pair-specific diagonal and globally shared dense middle transformations, respectively.

Summary of main results. Table 1 reveals three main observations. **(1)** TriBLoRA improves aggregate performance over LoRA and compares favorably with other PEFT baselines. On GLUE, the identity and K-F variants average 85.53 and 85.48, versus 85.29 for LoRA and 85.34 for the strongest HydraLoRA configuration, while attaining top-two results on six of eight tasks. On LLaMA-2-7B, uniform TriBLoRA averages 40.06 over six tasks, versus 39.51 for LoRA and 39.43 for BoRA. It leads on MMLU, BBH, and WinoGrande; BoRA leads on DROP, K-F on ARC-C, and LoRA on HumanEval. **(2)** The two core initializations exhibit complementary behavior. Identity initialization gives the higher overall average on both settings, whereas K-F performs better on several individual GLUE tasks, including MNLI, MRPC, CoLA, and QNLI, and leads on ARC-C in the LLaMA-2-7B evaluation. A more detailed structural comparison with related PEFT methods is provided in Appendix C.2. **(3)** These improvements require only a modest increase in adapter parameters. For fixed n and r, the pair-specific cores contribute n^2r^2 additional parameters per adapted matrix. On GLUE, TriBLoRA uses $13{,}312$ parameters per matrix versus $12{,}288$ for LoRA and remains below HydraLoRA with $r = 4$ ($14{,}592$). On LLaMA-2-7B, the same $1{,}024$ additional

Table 2: Training-time comparison of block number and interaction-matrix structure at $r = 8$.

Method	n	Core	Params/matrix	SST-2 (%)	MRPC (%)	RTE (%)	STS-B (%)	Avg. (%)
LoRA	–	–	12,288	94.65±0.37	90.28±0.49	75.81±1.25	90.62±0.18	87.84
Single-block / global dense interaction								
MoSLoRA	1	Dense	12,352	94.46±0.40	<u>90.78</u>±0.81	75.34±1.19	90.59±0.17	87.79
TriBLoRA	1	Dense	12,352	**94.76**±0.33	90.23±1.01	75.78±1.94	90.62±0.17	87.85
Four-block interaction structure								
TriBLoRA	4	Fixed identity†	12,288	94.30±0.40	89.90±0.29	73.00±1.55	90.31±0.24	86.88
BoRA	4	Diagonal	12,416	94.61±0.20	**91.22**±0.58	72.89±7.29	90.30±0.67	87.26
TriBLoRA	4	Diagonal	12,416	94.69±0.13	90.60±0.56	<u>76.14</u>±1.05	**90.79**±0.23	<u>88.06</u>
TriBLoRA	4	Dense	13,312	<u>94.72</u>±0.46	90.72±0.45	**76.93**±0.82	<u>90.73</u>±0.17	**88.28**

parameters increase the count from 65,536 to 66,560, corresponding to only a 1.56% overhead. Detailed comparisons are provided in Appendix C.1.

5.2 Effect of Interaction-Matrix Structure

At local rank $r = 8$, we evaluate core structure on SST-2, MRPC, RTE, and STS-B. For $n = 4$, TriBLoRA uses fixed-identity, trainable diagonal, or dense cores under the same outer-factor sharing; LoRA, MoSLoRA, BoRA, and $n = 1$ TriBLoRA serve as references. Table 2 shows that four-block dense TriBLoRA reaches 88.28, compared with 87.85 for $n = 1$ and 87.84 for rank-8 LoRA. Within $n = 4$, dense cores improve the average from 88.06 to 88.28 over diagonal cores with 896 additional parameters per matrix, improving SST-2, MRPC, and RTE (largest on RTE: 76.14 → 76.93), while STS-B slightly favors diagonal cores. Thus, dense pair-specific transformations provide a modest aggregate gain over diagonal modulation. Fixed-identity Fixed-identity TriBLoRA is algebraically equivalent to rank-r LoRA but uses a different FP16 arithmetic path; BoRA also differs from diagonal TriBLoRA in parameterization and optimization. Corresponding controls are provided in Appendix C.4.1.

5.3 Effect of Block-Rank Diversity

We next examine intra-layer importance heterogeneity by comparing uniform, layer-adaptive, and block-adaptive allocation under matched rank budgets. Task-specific topologies are constructed using GoRA-style gradient saliency and fixed throughout training; details are provided in Appendix B.4. GLUE uses task-specific layer topologies, whereas LLaMA-2-7B uses the topology estimated from Alpaca training. Given these layer budgets, γ controls within-module rank differentiation without changing the layer-wise total: $\gamma = 0$ gives uniform local ranks, while larger values increase heterogeneity within the range 4–16. We use $\gamma = 1.25$ as the main block-adaptive setting. We first evaluate the resulting block-adaptive configurations with identity initialization and then evaluate K-F initialization on the same topologies. K-F yields a clear gain on GLUE at $\gamma = 0.50$ (85.38 → 85.52) and on LLaMA-2-7B at $\gamma = 1.25$ (40.13 → 40.40), while remaining comparable to identity initialization on GLUE at $\gamma = 1.25$. For each training task, the topology is constructed using seed 42 and then fixed across all reported training seeds. Further construction details and sensitivity to γ are provided in Appendix C.5.1.

Summary of block-rank adaptation. Table 3 reveals three observations across GLUE and LLaMA-2-7B. **(1)** Layer-wise rank adaptation benefits LoRA but provides no consistent improvement for TriBLoRA. On GLUE, Layer-Adaptive LoRA improves several tasks, although the degradation on CoLA largely offsets these gains in the overall average. On LLaMA-2-7B, the improvement is clearer, increasing the LoRA average from 39.51 to 40.10. In contrast, Layer-Adaptive TriBLoRA does not improve over its uniform counterpart in either setting (e.g., 40.06 to 39.85 on LLaMA-2-7B), suggesting that the interaction between layer-level rank allocation and the TriBLoRA parameterization requires further investigation. For GLUE, the particularly strong RTE result of uniform TriBLoRA noticeably affects the eight-task average; we therefore additionally report the seven-task average excluding RTE as a secondary diagnostic. **(2)** Introducing block-wise rank

Table 3: Block-rank diversity under a matched layer budget: (a) GLUE and (b) LLaMA-2-7B.

		Method	MNLI (%)	SST-2 (%)	MRPC (%)	CoLA (%)	QNLI (%)	QQP (%)	RTE (%)	STS-B (%)	Avg. (%)	Avg.$^{\text{w/o RTE}}$ (%)
(a) GLUE	Unif.	LoRA	87.25$_{\pm 0.06}$	94.65$_{\pm 0.37}$	90.28$_{\pm 0.49}$	61.63$_{\pm 2.25}$	92.78$_{\pm 0.01}$	89.28$_{\pm 0.06}$	75.81$_{\pm 1.25}$	90.62$_{\pm 0.18}$	85.29	86.64
		TriBLoRA	87.23$_{\pm 0.22}$	94.72$_{\pm 0.46}$	90.72$_{\pm 0.45}$	62.02$_{\pm 0.69}$	92.64$_{\pm 0.22}$	89.27$_{\pm 0.08}$	**76.93**$_{\pm 0.82}$	90.73$_{\pm 0.17}$	85.53	86.76
		TriBLoRA (K-F)	87.29$_{\pm 0.05}$	94.72$_{\pm 0.41}$	91.15$_{\pm 0.49}$	62.19$_{\pm 0.64}$	92.68$_{\pm 0.05}$	89.24$_{\pm 0.17}$	75.85$_{\pm 1.45}$	90.70$_{\pm 0.24}$	85.48	86.85
	Layer-adapt.	AdaLoRA	87.21$_{\pm 0.09}$	94.61$_{\pm 0.50}$	88.53$_{\pm 0.42}$	58.98$_{\pm 1.37}$	92.78$_{\pm 0.20}$	88.78$_{\pm 0.07}$	60.87$_{\pm 1.89}$	90.20$_{\pm 0.10}$	82.75	85.87
		LoRA	87.26$_{\pm 0.07}$	94.57$_{\pm 0.13}$	90.72$_{\pm 0.77}$	60.41$_{\pm 1.17}$	**92.96**$_{\pm 0.31}$	89.32$_{\pm 0.11}$	76.50$_{\pm 1.30}$	90.68$_{\pm 0.12}$	85.30	86.56
		TriBLoRA	87.25$_{\pm 0.13}$	94.65$_{\pm 0.24}$	90.41$_{\pm 0.12}$	62.02$_{\pm 1.48}$	92.81$_{\pm 0.22}$	**89.34**$_{\pm 0.16}$	75.96$_{\pm 1.13}$	90.87$_{\pm 0.10}$	85.41	86.77
		TriBLoRA (K-F)	87.18$_{\pm 0.06}$	94.88$_{\pm 0.26}$	90.84$_{\pm 0.58}$	61.12$_{\pm 1.39}$	92.71$_{\pm 0.17}$	89.26$_{\pm 0.08}$	76.32$_{\pm 1.56}$	90.78$_{\pm 0.05}$	85.39	86.68
	Block-adapt.	**TriBLoRA**$_{(1.25)}$	87.18$_{\pm 0.05}$	94.88$_{\pm 0.07}$	90.37$_{\pm 0.46}$	**62.78**$_{\pm 1.47}$	92.85$_{\pm 0.05}$	89.30$_{\pm 0.04}$	75.99$_{\pm 1.29}$	90.79$_{\pm 0.11}$	85.52	86.88
		TriBLoRA (K-F)$_{(0.50)}$	**87.38**$_{\pm 0.09}$	94.80$_{\pm 0.13}$	**91.28**$_{\pm 0.85}$	62.11$_{\pm 1.27}$	92.76$_{\pm 0.21}$	89.21$_{\pm 0.16}$	75.78$_{\pm 1.22}$	90.85$_{\pm 0.17}$	85.52	86.91
		TriBLoRA (K-F)$_{(1.25)}$	87.16$_{\pm 0.06}$	94.99$_{\pm 0.33}$	90.93$_{\pm 0.42}$	61.59$_{\pm 0.87}$	92.81$_{\pm 0.18}$	89.30$_{\pm 0.13}$	76.39$_{\pm 1.58}$	90.80$_{\pm 0.16}$	85.50	86.80
		TriBLoRA$_{\text{best-}\gamma}$	87.33$_{\pm 0.13}$$_{(0.25)}$	**95.22**$_{\pm 0.46}$$_{(0.5)}$	91.04$_{\pm 0.38}$$_{(1.75)}$	**62.78**$_{\pm 1.47}$$_{(1.25)}$	92.92$_{\pm 0.26}$$_{(1.0)}$	89.32$_{\pm 0.08}$$_{(1.75)}$	75.99$_{\pm 1.29}$$_{(1.25)}$	**90.94**$_{\pm 0.16}$$_{(1.75)}$	**85.69**	**87.08**

		Method	Best Eval Loss ↓	MMLU (%)	BBH (%)	DROP (%)	WinoGrande (%)	ARC-C (%)	HumanEval (%)	Avg. (6 tasks) (%)
(b) LLaMA-2-7B	Unif.	LoRA	0.8311	43.54$_{\pm 0.50}$	32.94$_{\pm 0.45}$	32.44$_{\pm 0.17}$	70.24$_{\pm 0.24}$	42.53$_{\pm 1.13}$	**15.37**$_{\pm 0.38}$	39.51
		TriBLoRA	0.8333	44.18$_{\pm 0.65}$	33.55$_{\pm 0.34}$	32.53$_{\pm 0.65}$	70.54$_{\pm 0.58}$	44.57$_{\pm 1.88}$	15.00$_{\pm 0.73}$	40.06
		TriBLoRA (K-F)	0.8307	43.98$_{\pm 0.20}$	33.17$_{\pm 0.33}$	32.46$_{\pm 0.23}$	70.20$_{\pm 0.42}$	44.86$_{\pm 0.77}$	15.34$_{\pm 0.70}$	40.00
	Layer-adapt.	AdaLoRA	0.8818	42.65$_{\pm 0.17}$	33.04$_{\pm 0.45}$	32.01$_{\pm 0.28}$	69.66$_{\pm 0.62}$	35.84$_{\pm 1.12}$	15.10$_{\pm 0.57}$	38.05
		LoRA	0.8298	44.22$_{\pm 0.24}$	33.28$_{\pm 0.51}$	32.21$_{\pm 0.44}$	70.83$_{\pm 0.55}$	45.24$_{\pm 0.50}$	14.83$_{\pm 0.50}$	40.10
		TriBLoRA	**0.8297**	44.04$_{\pm 0.39}$	33.11$_{\pm 0.27}$	32.13$_{\pm 0.27}$	70.69$_{\pm 0.59}$	44.06$_{\pm 1.29}$	15.10$_{\pm 0.57}$	39.85
		TriBLoRA (K-F)	0.8301	44.02$_{\pm 0.32}$	**33.65**$_{\pm 0.43}$	32.23$_{\pm 0.49}$	70.37$_{\pm 0.63}$	45.36$_{\pm 1.01}$	14.95$_{\pm 0.39}$	40.10
	Block-adapt.	**TriBLoRA**$_{(1.25)}$	0.8301	44.35$_{\pm 0.28}$	33.41$_{\pm 0.33}$	32.12$_{\pm 0.39}$	70.73$_{\pm 0.52}$	45.12$_{\pm 1.97}$	15.07$_{\pm 0.56}$	40.13
		TriBLoRA (K-F)$_{(1.25)}$	0.8304	44.23$_{\pm 0.27}$	33.41$_{\pm 0.29}$	**32.68**$_{\pm 0.46}$	70.23$_{\pm 0.36}$	**47.05**$_{\pm 0.81}$	14.80$_{\pm 0.35}$	**40.40**
		TriBLoRA$_{\text{best-}\gamma}$	0.8299$_{(1.0)}$	**44.41**$_{\pm 0.23}$$_{(1.5)}$	33.50$_{\pm 0.19}$$_{(1.5)}$	32.26$_{\pm 0.29}$$_{(0.5)}$	**71.02**$_{\pm 0.68}$$_{(1.75)}$	45.12$_{\pm 1.97}$$_{(1.25)}$	15.20$_{\pm 0.55}$$_{(1.0)}$	40.25

differentiation is more effective than layer-wise redistribution alone. With the fixed $\gamma = 1.25$, Block-Adaptive TriBLoRA improves over the layer-adaptive configuration on both benchmark settings, indicating that redistributing a fixed layer budget among local regions can better exploit importance heterogeneity. **(3)** The preferred degree of block-rank differentiation varies across tasks and their underlying topologies. Since each GLUE task has its own layer-level topology, a common γ specifies a shared degree of intra-layer differentiation rather than a task-specific optimum. We therefore use the fixed $\gamma = 1.25$ configuration as the main block-adaptive setting and additionally report the best observed value for each task over the tested γ grid as a sensitivity diagnostic. This per-task upper envelope reaches an overall average of 85.69 and a seven-task average of 87.08. The best observed settings are $\gamma = 0.25$ for MNLI, 0.50 for SST-2, 1.75 for MRPC, QQP, and STS-B, 1.00 for QNLI, and 1.25 for CoLA and RTE. In the LLaMA-2-7B setting, all downstream benchmarks share the topology estimated from Alpaca training; the sensitivity sweep similarly shows that the preferred degree varies across downstream tasks, while $\gamma = 1.25$ remains our fixed main configuration.

Effect of the K-F initialization–optimization setting. The initialization results provide two further observations. **(1)** In both the uniform and layer-adaptive settings, the local ranks are kept equal within each module, resulting in square M_{ij} matrices. Under these settings, K-F initialization does not consistently outperform identity initialization. **(2)** Under block adaptation, heterogeneous side ranks make many M_{ij} rectangular, making the initialization increasingly dependent on the resulting topology. On GLUE, we further evaluate K-F at two selected adaptive degrees, $\gamma = 0.50$ and $\gamma = 1.25$. At $\gamma = 0.50$, K-F improves the overall average from 85.38 to 85.52, whereas at $\gamma = 1.25$ it remains comparable to identity initialization (85.50 vs. 85.52). On LLaMA-2-7B, K-F at $\gamma = 1.25$ improves the six-task average from 40.13 to 40.40. These results suggest that the benefit of dense initialization depends on the heterogeneous block topology rather than being uniform across adaptive degrees. Detailed analyses are provided in Appendices B.3 and C.5.1.

5.4 Block Granularity and Learned Topology

Appendices C.3 and C.5 provide the complete block-number and adaptive-topology analyses. **(1)** Increasing n from 2 to 4 consistently improves TriBLoRA, while further subdivision yields smaller and less consistent gains. This suggests that finer partitions improve pair-specific interaction modeling up to a moderate granularity; beyond this point, more cores and fragmented optimization make the additional nominal capacity harder to exploit. **(2)** The resulting parameter–performance trade-off remains favorable. On GLUE, TriBLoRA with $n = 4$ achieves 85.53 using $13{,}312$ parameters per matrix, exceeding LoRA with $r = 10$ (85.19, $15{,}360$ parameters) and $r = 12$ (85.40, $18{,}432$ parameters). On LLaMA-2-7B, it reaches 40.06 with $66{,}560$ parameters, compared with 39.94 for rank-16 LoRA using $131{,}072$ parameters. **(3)** The adaptive-topology analysis reveals a complementary trade-off: stronger block-rank differentiation enables more importance-aware capacity placement but can reduce the pairwise rank ceiling under side-rank mismatch. Consequently, both the preferred block granularity and adaptive degree remain task and topology dependent.

AI USE STATEMENT

In this work, we used generative AI tools to help develop theoretical models and conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose and refine hypotheses, design and provide feedback on research methodology and experiments, and assist with method implementation.

We additionally used generative AI tools to create and edit software code, draft and revise parts of the research paper, summarize and analyze existing literature, identify relevant literature, and format bibliographic references.

All AI-assisted outputs were reviewed by the authors before being incorporated into the work. Mathematical claims, derivations, and proofs were checked by the authors for consistency with the stated assumptions and definitions. AI-assisted code and method implementations were inspected, executed, and validated through experiments and numerical checks. Experimental results reported in the paper were obtained from the authors' actual implementations and experimental runs. Literature suggestions and bibliographic information were checked against the corresponding original sources before inclusion.

The authors made the final decisions regarding the theoretical formulation, research hypotheses, methodology, experimental design, interpretation, and manuscript content. We take responsibility for the final content of this work, including text, mathematical claims, code, experimental conclusions, and other artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproducibility throughout the paper and appendix. Complete derivations and proofs for the theoretical results are given in Appendix A. Appendix B specifies the experimental protocol, including hardware and benchmark settings, random-seed protocols, training hyperparameters, interaction-matrix initialization and optimization settings, and the gradient-based procedure used to construct block-rank topologies before fine-tuning. GLUE experiments use task-dependent sets of 3, 5, or 10 random seeds, while the LLaMA-2-7B experiments target five training seeds; reported statistics are computed over completed runs. The block-rank topology is constructed once from the prescribed probing data and then fixed across the corresponding training runs. Anonymous source code and experiment configurations will be available at `https://anonymous.4open.science/r/TriBLoRA-FF88/`.

REFERENCES

Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. Lora-xs: Low-rank adaptation with extremely small number of parameters. In *ECAI 2025: 28th European Conference on Artificial Intelligence 25-30 October 2025, Bologna, Italy-Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025) Proceedings*, pp. 3194–3201. SAGE Publications 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, 2024.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 3319–3327. IEEE, 2017.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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.

Huiming Ding, Xiaochen Li, Jianhui Ma, Xu An, Yihui Yang, and Zhenyu Tan. CSPLoRA: Confidence-guided structure planning for low-rank adaptation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. To appear.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. *arXiv preprint arXiv:2311.11696*, 2023.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2368–2378, 2019. doi: 10.18653/v1/N19-1246. URL https://aclanthology.org/N19-1246/.

Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8730–8738. IEEE, 2018.

Haonan He, Peng Ye, Yuchen Ren, Yuan Yuan, Luyang Zhou, Shucun Ju, and Lei Chen. GoRA: Gradient-driven adaptive low rank adaptation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 126541–126569, 2025.

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*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

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.

Qiushi Huang, Tom Ko, Zhan Zhuang, Lilian Tang, and Yu Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=TwJrTz9cRS.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, et al. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*, 2024.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

Shiwei Li, Xiandi Luo, Haozhao Wang, Xing Tang, Ziqiang Cui, Dugang Liu, Yuhua Li, Xiuqiang He, and Ruixuan Li. BoRA: Towards more expressive low-rank adaptation with block diversity. *arXiv preprint arXiv:2508.06953*, 2025.

Shiwei Li, Xiandi Luo, Haozhao Wang, Xing Tang, Ziqiang Cui, Dugang Liu, Yuhua Li, Xiuqiang He, and Ruixuan Li. Beyond higher rank: Token-wise input-output projections for efficient low-rank adaptation. *Advances in Neural Information Processing Systems*, 38:136414–136435, 2026.

Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. ReLoRA: High-rank training through low-rank updates. In *International Conference on Learning Representations*, 2024.

Weicheng Lin, Yi Zhang, Jiawei Dang, and Liang-Jie Zhang. Tlora: Task-aware low rank adaptation of large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 29252–29269, 2026.

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 *Proceedings of the 41st International Conference on Machine Learning*, pp. 32100–32121, 2024.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Vardan Papyan, Xiao Y Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40): 24652–24663, 2020.

Pengjie Ren, Chengshun Shi, Shiguang Wu, Mengqi Zhang, Zhaochun Ren, Maarten de Rijke, Zhumin Chen, and Jiahuan Pei. MELoRA: Mini-ensemble low-rank adapters for parameter-efficient fine-tuning. *arXiv preprint arXiv:2402.17263*, 2025.

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.

Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in neural information processing systems*, 33:20378–20389, 2020.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following LLaMA model. GitHub repository, 2023. URL `https://github.com/tatsu-lab/stanford_alpaca`.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *Advances in Neural Information Processing Systems*, 37: 9565–9584, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. Dylora: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 3274–3287, 2023.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 5797–5808, 2019.

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. URL `https://openreview.net/forum?id=rJ4km2R5t7`.

Taiqiang Wu, Jiahao Wang, Zhe Zhao, and Ngai Wong. Mixture-of-subspaces in low-rank adaptation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7880–7899, 2024.

Wenhan Xia, Chengwei Qin, and Elad Hazan. Chain of lora: Efficient fine-tuning of language models via residual learning. *arXiv preprint arXiv:2401.04151*, 2024.

Yahma. Alpaca-cleaned. Hugging Face dataset, 2023. URL `https://huggingface.co/datasets/yahma/alpaca-cleaned`.

Yuchen Zeng and Kangwook Lee. The expressive power of low-rank adaptation. In *International Conference on Learning Representations*, volume 2024, pp. 5078–5123, 2024.

Feiyu Zhang, Liangzhi Li, Junhao Chen, Zhouqiang Jiang, Bowen Wang, and Yiming Qian. IncreLoRA: Incremental parameter allocation method for parameter-efficient fine-tuning. *arXiv preprint*, 2023a.

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

APPENDIX OVERVIEW

The appendix is organized into five parts that complement the main paper from theoretical, experimental, diagnostic, implementation, and future-work perspectives.

Appendix A provides the complete theoretical development underlying Sections 3–4. It establishes the common notation and orthogonal shared-subspace representation, derives the expressive-family and approximation relations of TriBLoRA with LoRA, MELoRA, and BoRA, and analyzes block-rank topology, exact parameter-budget preservation, and the practical consequences of two-sided rank diversity.

Appendix B specifies the experimental protocol used throughout the paper. It describes the benchmark environments, random seeds, and training hyperparameters, followed by a detailed analysis of interaction-matrix initialization, early optimization dynamics, and the relative learning rate of the pair-specific cores.

Appendix C provides additional empirical and structural analyses. It quantifies adapter cost, positions TriBLoRA relative to related PEFT structures, studies block granularity and parameter efficiency, examines the functional role of the learned interaction matrices, and analyzes adaptive block-rank topology. The latter includes adaptive-degree sensitivity, the interaction between block topology and core initialization, structural rank loss, topology visualization, and within-module rank-placement sensitivity.

Appendix D discusses implementation considerations for blockwise interaction computation. It presents algebraically equivalent computational organizations of TriBLoRA, analyzes their computation and memory trade-offs, and discusses the numerical sensitivity of equivalent FP16/BF16 computation paths. These engineering considerations are separated from the structural contribution of the TriBLoRA parameterization.

Finally, Appendix E summarizes the current limitations and directions for future work. It discusses the incomplete characterization of Kaiming–Frobenius initialization, task-dependent variability on GLUE, the unresolved interaction between block granularity and interaction-matrix optimization, and the need for adaptive-topology criteria that explicitly balance importance-aware specialization against structural rank loss.

A ADDITIONAL THEORETICAL ANALYSIS

This appendix provides the derivations supporting Section 4. Appendix A.1 summarizes notation and scope. Appendices A.2 and A.3 establish the shared-subspace representation, expressive-family hierarchy, and approximation comparisons. Appendix A.4 collects the block-rank analysis, covering idealized allocation benefit, exact parameter-budget preservation, and the pairwise rank-ceiling trade-off.

A.1 NOTATION AND SCOPE

We follow the block-structured notation and equally partitioned square setting of Sections 3.1 and 4, with $m = d/n$ and uniform local rank $r \leq m$ unless otherwise stated. The analysis concerns the minimum squared Frobenius approximation error of a target update, independently of optimization dynamics and generalization. Table 4 summarizes the notation used in the proofs.

Where invertible local coordinate transforms are used, the outer factors are assumed to have rank r, as specified in Appendix A.2. BoRA denotes its relaxed structural family with arbitrary pair-specific

diagonal connectors. The idealized effective-rank allocation in Appendix A.4.1 is distinguished from the exact parameter count and two-sided rank constraints in Appendices A.4.2 and A.4.3.

Table 4: Notation used in the theoretical analysis.

Symbol	Meaning
d, n, m, r	Matrix dimension, block count, block width, and uniform local rank
$\Delta W^\star, \widehat{\Delta W}$	Target update and its structured approximation
$\Delta W_{ij}^\star$	Target interaction block (i, j)
B_i, A_j, M_{ij}	Shared output/input factors and pair-specific interaction core
$\mathcal{B}, \mathcal{M}, \mathcal{A}$	Global block matrices in the TriBLoRA factorization
$Q_i, P_j;\ R_i, S_j$	Orthonormal local bases and their coordinate transforms
$Z_{ij} = R_i M_{ij} S_j$	Interaction core in orthogonal local coordinates
$C_{ij} = Q_i^\top \Delta W_{ij}^\star P_j$	Projected target interaction
$\mathcal{F}_\mathcal{S}, \mathcal{F}_\mathrm{L}(k)$	Feasible family of structure $\mathcal{S}$; updates of rank at most k
$E_\mathcal{F}^\star$	Minimum squared Frobenius error over $\mathcal{F}$
$\sigma_k(X)$	k-th largest singular value of X
$\Delta W_{i,:}^\star, \Delta W_{:,j}^\star$	Target block row and block column
$\epsilon_\mathrm{out}, \epsilon_\mathrm{in}$	Residual spectral energies beyond rank r in block rows and columns
$r_{\ell,b}, K$	Local effective rank and total effective-rank budget
$\mathcal{T}_\mathrm{layer}(K), \mathcal{T}_\mathrm{block}(K)$	Feasible layer-wise and block-wise topology spaces
$r_{B,i}, r_{A,j}$	Output- and input-side ranks of the actual topology
$N_\mathrm{param}(\mathcal{T})$	Parameter count of topology $\mathcal{T}$
$\rho_{ij}^\mathrm{cap}, \bar{\rho}_\mathrm{cap}$	Individual and mean pairwise rank ceilings

A.2 Proof of Shared-Subspace and Pair-Specific Decomposition

Motivated by prior evidence of intra-layer *functional heterogeneity*, we provide the complete derivation of the results in Section 4.1. Throughout this subsection, we consider the non-degenerate case in which the local outer factors have rank r. Rank-deficient factors can be treated analogously by replacing r with their effective rank.

Equivalent orthogonal representation. Recall that a TriBLoRA interaction block is parameterized as

$$\widehat{\Delta W}_{ij} = B_i M_{ij} A_j, \tag{15}$$

where

$$B_i \in \mathbb{R}^{m\times r}, \qquad A_j \in \mathbb{R}^{r\times m}, \qquad M_{ij} \in \mathbb{R}^{r\times r}. \tag{16}$$

Applying thin QR decompositions to B_i and $A_j^\top$ gives

$$B_i = Q_i R_i, \qquad A_j = S_j P_j^\top, \tag{17}$$

where

$$Q_i, P_j \in \mathbb{R}^{m\times r}, \qquad Q_i^\top Q_i = P_j^\top P_j = I_r, \tag{18}$$

and $R_i, S_j \in \mathbb{R}^{r\times r}$ are invertible.

Substituting Eq. (17) into Eq. (15) yields

$$\begin{aligned} \widehat{\Delta W}_{ij} &= Q_i R_i M_{ij} S_j P_j^\top \\ &= Q_i Z_{ij} P_j^\top, \end{aligned} \tag{19}$$

where

$$Z_{ij} := R_i M_{ij} S_j. \tag{20}$$

Because R_i and S_j are invertible, the mapping between M_{ij} and Z_{ij} is bijective:

$$M_{ij} = R_i^{-1} Z_{ij} S_j^{-1}. \tag{21}$$

Hence, allowing an arbitrary $M_{ij} \in \mathbb{R}^{r\times r}$ is equivalent to allowing an arbitrary $Z_{ij} \in \mathbb{R}^{r\times r}$. The orthogonal representation therefore does not change the feasible TriBLoRA update family; it only changes the latent coordinate system.

Optimal pair-specific interaction. For fixed shared local bases Q_i and P_j, consider the approximation problem for block (i, j):

$$\min_{Z_{ij}} \left\| \Delta W_{ij}^{\star} - Q_i Z_{ij} P_j^{\top} \right\|_F^2 . \tag{22}$$

Define the projected target interaction

$$C_{ij} := Q_i^{\top} \Delta W_{ij}^{\star} P_j. \tag{23}$$

The approximation error admits the following orthogonal decomposition:

$$\begin{aligned} &\left\| \Delta W_{ij}^{\star} - Q_i Z_{ij} P_j^{\top} \right\|_F^2 \\ &= \left\| \Delta W_{ij}^{\star} - Q_i Q_i^{\top} \Delta W_{ij}^{\star} P_j P_j^{\top} \right\|_F^2 + \left\| Q_i (C_{ij} - Z_{ij}) P_j^{\top} \right\|_F^2 \\ &= \left\| \Delta W_{ij}^{\star} \right\|_F^2 - \| C_{ij} \|_F^2 + \| C_{ij} - Z_{ij} \|_F^2 . \end{aligned} \tag{24}$$

The second equality follows from the orthonormality of Q_i and P_j.

The first two terms in Eq. (24) are independent of Z_{ij}. Therefore,

$$Z_{ij}^{\star} = C_{ij} = Q_i^{\top} \Delta W_{ij}^{\star} P_j. \tag{25}$$

The corresponding minimum block-wise error is

$$E_{ij}^{\mathrm{TriB}}(Q_i, P_j) = \left\| \Delta W_{ij}^{\star} \right\|_F^2 - \left\| Q_i^{\top} \Delta W_{ij}^{\star} P_j \right\|_F^2 . \tag{26}$$

Global reduced objective. Because the interaction blocks occupy disjoint entries of the update matrix, the global Frobenius error is the sum of the block-wise errors:

$$\begin{aligned} E_{\mathrm{TriB}}(Q, P) &= \sum_{i,j} E_{ij}^{\mathrm{TriB}}(Q_i, P_j) \\ &= \| \Delta W^{\star} \|_F^2 - \sum_{i,j} \left\| Q_i^{\top} \Delta W_{ij}^{\star} P_j \right\|_F^2 . \end{aligned} \tag{27}$$

Equivalently, optimizing the shared local subspaces amounts to

$$\max_{\substack{Q_i^{\top} Q_i = I_r \\ P_j^{\top} P_j = I_r}} \sum_{i,j} \left\| Q_i^{\top} \Delta W_{ij}^{\star} P_j \right\|_F^2 . \tag{28}$$

Why the outer subspaces represent shared structure. Equation (28) also makes the sharing property explicit. Fix all input-side bases $\{P_j\}$ and consider the optimization of a particular output-side basis Q_i. Its objective is

$$\sum_j \left\| Q_i^{\top} \Delta W_{ij}^{\star} P_j \right\|_F^2 = \mathrm{tr}\left(Q_i^{\top} H_i^{\mathrm{out}} Q_i \right), \tag{29}$$

where

$$H_i^{\mathrm{out}} := \sum_j \Delta W_{ij}^{\star} P_j P_j^{\top} (\Delta W_{ij}^{\star})^{\top}. \tag{30}$$

Thus, at a coordinate-wise optimum, the columns of Q_i span the leading r-dimensional eigenspace of H_i^{out}.

Similarly, fixing $\{Q_i\}$ gives

$$\sum_i \left\| Q_i^{\top} \Delta W_{ij}^{\star} P_j \right\|_F^2 = \mathrm{tr}\left(P_j^{\top} H_j^{\mathrm{in}} P_j \right), \tag{31}$$

where

$$H_j^{\mathrm{in}} := \sum_i (\Delta W_{ij}^{\star})^{\top} Q_i Q_i^{\top} \Delta W_{ij}^{\star}. \tag{32}$$

Hence, the columns of P_j span the leading r-dimensional eigenspace of H_j^{in} at a coordinate-wise optimum.

These expressions show that Q_i is determined by aggregate information from all interaction pairs sharing output block i, while P_j is determined by all pairs sharing input block j. In contrast, Eq. (25) shows that $Z_{ij}^\star$ depends on the individual target interaction $\Delta W_{ij}^\star$ once the shared subspaces are fixed.

Therefore, TriBLoRA admits the decomposition

$$\underbrace{\{Q_i, P_j\}}_{\text{shared local subspaces}} \quad + \quad \underbrace{\{Z_{ij}\}}_{\text{pair-specific transformations}} . \tag{33}$$

This separation holds at the subspace level; individual coordinates are not unique because invertible changes of latent basis can be absorbed into the interaction matrices.

A.3 Expressive Hierarchy and Approximation Analysis

We now establish the expressive and approximation results summarized in Section 4.2. The analysis proceeds in three steps. Appendix A.3.1 first derives the global-rank properties and feasible-family inclusions among LoRA, MELoRA, BoRA, and TriBLoRA. Appendix A.3.2 then positions TriBLoRA between rank-r and unrestricted rank-nr LoRA, separating the gain from relaxing global interaction coupling from the remaining shared-local-subspace constraint. Finally, Appendix A.3.3 compares TriBLoRA with MELoRA and BoRA under matched local subspaces, isolating the approximation penalties introduced by removing cross-block interactions or restricting pair-specific transformations to diagonal families.

A.3.1 Proof of Expressive Families and Rank Structure

We provide the complete derivation of the rank and feasible-family results in Section 4.2. Throughout this subsection, we use the uniform n-block partition with block width $m = d/n$ and local rank $r \le m$.

Let

$$\mathcal{D}_r := \left\{ D \in \mathbb{R}^{r\times r} : D \text{ is diagonal} \right\}. \tag{34}$$

Global rank capacity. A standard rank-r LoRA update satisfies

$$\Delta W_{\text{L}} = BA, \qquad B \in \mathbb{R}^{d\times r}, \quad A \in \mathbb{R}^{r\times d}, \tag{35}$$

and hence

$$\operatorname{rank}(\Delta W_{\text{L}}) \le r. \tag{36}$$

For TriBLoRA, recall the global factorization

$$\Delta W_{\text{TriB}} = \mathcal{B}\mathcal{M}\mathcal{A}, \tag{37}$$

where

$$\mathcal{B} \in \mathbb{R}^{d\times nr}, \qquad \mathcal{M} \in \mathbb{R}^{nr\times nr}, \qquad \mathcal{A} \in \mathbb{R}^{nr\times d}. \tag{38}$$

Therefore,

$$\operatorname{rank}(\Delta W_{\text{TriB}}) \le \min\left\{\operatorname{rank}(\mathcal{B}), \operatorname{rank}(\mathcal{M}), \operatorname{rank}(\mathcal{A})\right\} \le nr. \tag{39}$$

MELoRA consists of n spatially disjoint rank-r blocks:

$$\Delta W_{\text{ME}} = \operatorname{blkdiag}\left(B_1A_1, \ldots, B_nA_n\right). \tag{40}$$

Hence,

$$\operatorname{rank}(\Delta W_{\text{ME}}) = \sum_{i=1}^{n} \operatorname{rank}(B_iA_i) \le nr. \tag{41}$$

The BoRA family contains MELoRA as a special case, as shown below, and therefore also admits updates of rank nr. Consequently, when all local blocks have rank r, the upper bound nr is attainable by MELoRA, BoRA, and TriBLoRA.

Local rank constraints of BoRA and TriBLoRA. Consider the i-th complete block row of a TriBLoRA update:

$$\begin{aligned}\Delta W_{i,:}^{\mathrm{TriB}} &= [B_i M_{i1} A_1 \quad \cdots \quad B_i M_{in} A_n] \\ &= B_i \left[M_{i1} A_1 \quad \cdots \quad M_{in} A_n\right].\end{aligned} \tag{42}$$

Therefore,

$$\operatorname{rank}\left(\Delta W_{i,:}^{\mathrm{TriB}}\right) \le \operatorname{rank}(B_i) \le r. \tag{43}$$

Similarly, the j-th block column can be written as

$$\Delta W_{:,j}^{\mathrm{TriB}} = \begin{bmatrix} B_1 M_{1j} \\ \vdots \\ B_n M_{nj} \end{bmatrix} A_j, \tag{44}$$

which gives

$$\operatorname{rank}\left(\Delta W_{:,j}^{\mathrm{TriB}}\right) \le \operatorname{rank}(A_j) \le r. \tag{45}$$

BoRA has exactly the same outer-factor sharing structure,

$$\Delta W_{ij}^{\mathrm{BoRA}} = B_i \Sigma_{ij} A_j, \qquad \Sigma_{ij} \in \mathcal{D}_r, \tag{46}$$

so replacing M_{ij} by Σ_{ij} in the preceding arguments yields

$$\operatorname{rank}\left(\Delta W_{i,:}^{\mathrm{BoRA}}\right) \le r, \qquad \operatorname{rank}\left(\Delta W_{:,j}^{\mathrm{BoRA}}\right) \le r. \tag{47}$$

Thus, BoRA and TriBLoRA may achieve global rank nr, while every complete local block row and block column remains confined to an r-dimensional shared subspace.

LoRA is a special case of the structural BoRA family. Consider an arbitrary rank-r LoRA update

$$X_{\mathrm{L}} = BA. \tag{48}$$

Partition B along its rows and A along its columns:

$$B = \begin{bmatrix} B_1 \\ \vdots \\ B_n \end{bmatrix}, \qquad A = [A_1 \quad \cdots \quad A_n]. \tag{49}$$

Its (i, j)-th block is

$$(X_{\mathrm{L}})_{ij} = B_i A_j. \tag{50}$$

Choosing

$$\Sigma_{ij} = I_r, \qquad \forall i, j, \tag{51}$$

in the structural BoRA parameterization gives

$$B_i \Sigma_{ij} A_j = B_i A_j. \tag{52}$$

Therefore,

$$\mathcal{F}_{\mathrm{L}}(r) \subseteq \mathcal{F}_{\mathrm{BoRA}}(n, r). \tag{53}$$

MELoRA is a special case of the structural BoRA family. A MELoRA update has

$$\Delta W_{ij}^{\mathrm{ME}} = \begin{cases} B_i A_i, & i = j, \\ 0, & i \ne j. \end{cases} \tag{54}$$

Choose

$$\Sigma_{ij} = \begin{cases} I_r, & i = j, \\ 0, & i \ne j. \end{cases} \tag{55}$$

Since both I_r and 0 belong to $\mathcal{D}_r$, this is feasible in the BoRA family and reproduces MELoRA exactly. Hence,

$$\mathcal{F}_{\mathrm{ME}}(n, r) \subseteq \mathcal{F}_{\mathrm{BoRA}}(n, r). \tag{56}$$

Notice that $\mathcal{F}_{\mathrm{L}}(r)$ and $\mathcal{F}_{\mathrm{ME}}(n, r)$ are not required to contain one another. The two inclusions Eqs. (53) and (56) instead show that their union is contained in the BoRA family:

$$\mathcal{F}_{\mathrm{L}}(r) \cup \mathcal{F}_{\mathrm{ME}}(n, r) \subseteq \mathcal{F}_{\mathrm{BoRA}}(n, r). \tag{57}$$

BoRA is a special case of TriBLoRA. For every BoRA update, choose

$$M_{ij} = \Sigma_{ij}, \qquad \forall i, j. \tag{58}$$

Since TriBLoRA permits arbitrary $M_{ij} \in \mathbb{R}^{r\times r}$ and every diagonal matrix is a valid $r \times r$ matrix,

$$\mathcal{F}_{\mathrm{BoRA}}(n, r) \subseteq \mathcal{F}_{\mathrm{TriB}}(n, r). \tag{59}$$

TriBLoRA is contained in rank-nr LoRA. Using the global representation,

$$\Delta W_{\mathrm{TriB}} = \mathcal{B}\mathcal{M}\mathcal{A}, \tag{60}$$

define

$$\widetilde{B} := \mathcal{B}\mathcal{M} \in \mathbb{R}^{d\times nr}, \qquad \widetilde{A} := \mathcal{A} \in \mathbb{R}^{nr\times d}. \tag{61}$$

Then

$$\Delta W_{\mathrm{TriB}} = \widetilde{B}\widetilde{A}, \tag{62}$$

which is a valid rank-at-most-nr LoRA factorization. Therefore,

$$\mathcal{F}_{\mathrm{TriB}}(n, r) \subseteq \mathcal{F}_{\mathrm{L}}(nr). \tag{63}$$

Combining Eqs. (57), (59), and (63) gives the family relation stated in the main text:

$$\mathcal{F}_{\mathrm{L}}(r) \cup \mathcal{F}_{\mathrm{ME}}(n, r) \subseteq \mathcal{F}_{\mathrm{BoRA}}(n, r) \subseteq \mathcal{F}_{\mathrm{TriB}}(n, r) \subseteq \mathcal{F}_{\mathrm{L}}(nr). \tag{64}$$

Ordering of optimal approximation errors. Recall that for a feasible family $\mathcal{F}$,

$$E^{\star}_{\mathcal{F}} = \min_{X\in\mathcal{F}} \|\Delta W^{\star} - X\|_F^2 . \tag{65}$$

For any two feasible families $\mathcal{F}_1 \subseteq \mathcal{F}_2$, minimization over the larger family cannot increase the optimal error:

$$E^{\star}_{\mathcal{F}_2} \le E^{\star}_{\mathcal{F}_1}. \tag{66}$$

Applying Eq. (66) to Eq. (64) yields

$$E^{\star}_{\mathrm{L},nr} \le E^{\star}_{\mathrm{TriB}} \le E^{\star}_{\mathrm{BoRA}}. \tag{67}$$

Moreover, because the BoRA family contains both $\mathcal{F}_{\mathrm{L}}(r)$ and $\mathcal{F}_{\mathrm{ME}}(n, r)$,

$$E^{\star}_{\mathrm{BoRA}} \le E^{\star}_{\mathrm{L},r}, \qquad E^{\star}_{\mathrm{BoRA}} \le E^{\star}_{\mathrm{ME}}. \tag{68}$$

Therefore,

$$E^{\star}_{\mathrm{L},nr} \le E^{\star}_{\mathrm{TriB}} \le E^{\star}_{\mathrm{BoRA}} \le \min\left\{E^{\star}_{\mathrm{L},r}, E^{\star}_{\mathrm{ME}}\right\}. \tag{69}$$

This completes the proof of the global-rank bounds and the feasible-family error ordering.

A.3.2 Proof of TriBLoRA Between Rank-r and Rank-nr LoRA

We provide the complete derivation of the results in Section 4.2. The comparison with rank-r LoRA is performed under matched local subspaces, whereas the comparison with rank-nr LoRA characterizes the structural approximation loss induced by TriBLoRA's shared rank-r local subspaces.

Matched-subspace comparison with rank-r LoRA. Let

$$Q := \mathrm{blkdiag}(Q_1, \ldots, Q_n) \in \mathbb{R}^{d\times nr}, \qquad P := \mathrm{blkdiag}(P_1, \ldots, P_n) \in \mathbb{R}^{d\times nr}, \tag{70}$$

where $Q_i^{\top}Q_i = P_j^{\top}P_j = I_r$. For these fixed shared local subspaces, define the projected target interaction

$$\mathbf{C} := Q^{\top}\Delta W^{\star}P = [C_{ij}]_{i,j=1}^{n} \in \mathbb{R}^{nr\times nr}. \tag{71}$$

For any latent interaction matrix $Z \in \mathbb{R}^{nr\times nr}$, orthonormality gives

$$\left\|\Delta W^{\star} - QZP^{\top}\right\|_F^2 = \|\Delta W^{\star}\|_F^2 - \|\mathbf{C}\|_F^2 + \|\mathbf{C} - Z\|_F^2. \tag{72}$$

Thus, the first term is determined entirely by the chosen local subspaces Q and P, while the second term measures the additional restriction imposed on interactions within those subspaces. TriBLoRA places no restriction on the individual blocks of Z. Therefore,

$$Z^{\star}_{\mathrm{TriB}} = \mathbf{C}, \tag{73}$$

and its optimal error under the fixed local subspaces is

$$E_{\mathrm{TriB}}(Q,P) = \left\|\Delta W^{\star} - Q\mathbf{C}P^{\top}\right\|_F^2. \tag{74}$$

Hence, TriBLoRA removes the latent-interaction error, but it does not remove the approximation loss caused by the shared local subspaces themselves.

Under the same fixed local subspaces, a standard rank-r LoRA update can be written as

$$\widehat{\Delta W}_{\mathrm{L}} = QRSP^{\top}, \tag{75}$$

where

$$R \in \mathbb{R}^{nr\times r}, \qquad S \in \mathbb{R}^{r\times nr}. \tag{76}$$

Its latent interaction matrix is therefore

$$Z_{\mathrm{L}} = RS, \qquad \mathrm{rank}(Z_{\mathrm{L}}) \le r. \tag{77}$$

Conversely, every $Z \in \mathbb{R}^{nr\times nr}$ with $\mathrm{rank}(Z) \le r$ admits such a factorization. Thus, with the local subspaces held fixed, optimizing rank-r LoRA reduces to

$$\min_{\mathrm{rank}(Z)\le r} \|\mathbf{C} - Z\|_F^2. \tag{78}$$

By the Eckart–Young–Mirsky theorem, the optimum is the best rank-r approximation of $\mathbf{C}$, denoted by $\mathbf{C}_r$, and

$$\|\mathbf{C} - \mathbf{C}_r\|_F^2 = \sum_{k=r+1}^{nr} \sigma_k^2(\mathbf{C}). \tag{79}$$

Substituting this result into Eq. (72) gives

$$E_{\mathrm{L}}(Q,P) - E_{\mathrm{TriB}}(Q,P) = \sum_{k=r+1}^{nr} \sigma_k^2(\mathbf{C}). \tag{80}$$

Therefore, this matched-subspace comparison isolates only the effect of cross-pair latent interaction coupling. Both methods incur the same shared-subspace approximation error, while rank-r LoRA further restricts the projected interaction matrix to rank r. TriBLoRA removes this additional interaction restriction, but remains constrained by the choice of the shared local subspaces Q and P.

Lower bound induced by shared output subspaces. Consider an arbitrary feasible TriBLoRA update $\widehat{\Delta W}$. Its i-th complete block row is

$$\widehat{\Delta W}_{i,:} = Q_i \begin{bmatrix} Z_{i1}P_1^{\top} & \cdots & Z_{in}P_n^{\top} \end{bmatrix}. \tag{81}$$

Therefore,

$$\mathrm{rank}\left(\widehat{\Delta W}_{i,:}\right) \le r. \tag{82}$$

By the Eckart–Young–Mirsky theorem,

$$\left\|\Delta W^{\star}_{i,:} - \widehat{\Delta W}_{i,:}\right\|_F^2 \ge \sum_{k>r} \sigma_k^2\left(\Delta W^{\star}_{i,:}\right). \tag{83}$$

Since different block rows occupy disjoint entries,

$$\begin{aligned} \left\|\Delta W^{\star} - \widehat{\Delta W}\right\|_F^2 &= \sum_{i=1}^{n} \left\|\Delta W^{\star}_{i,:} - \widehat{\Delta W}_{i,:}\right\|_F^2 \\ &\ge \sum_{i=1}^{n}\sum_{k>r} \sigma_k^2\left(\Delta W^{\star}_{i,:}\right) = \epsilon_{\mathrm{out}}. \end{aligned} \tag{84}$$

Since this holds for every feasible TriBLoRA approximation,

$$E^{\star}_{\mathrm{TriB}} \ge \epsilon_{\mathrm{out}}. \tag{85}$$

Lower bound induced by shared input subspaces. The argument is symmetric for the j-th block column:

$$\widehat{\Delta W}_{:,j} = \begin{bmatrix} Q_1 Z_{1j} \\ \vdots \\ Q_n Z_{nj} \end{bmatrix} P_j^\top, \tag{86}$$

and hence

$$\operatorname{rank}\left(\widehat{\Delta W}_{:,j}\right) \le r. \tag{87}$$

Applying the best rank-r approximation bound to every block column and summing over j gives

$$E^\star_{\mathrm{TriB}} \ge \epsilon_{\mathrm{in}}. \tag{88}$$

Combining Eqs. (85) and (88) yields

$$E^\star_{\mathrm{TriB}} \ge \max\left\{\epsilon_{\mathrm{out}}, \epsilon_{\mathrm{in}}\right\}. \tag{89}$$

Constructive upper bound. We next construct a feasible TriBLoRA approximation whose error is at most $\epsilon_{\mathrm{out}} + \epsilon_{\mathrm{in}}$.

For every block row i, let $\bar{Q}_i \in \mathbb{R}^{m\times r}$ contain the leading r left singular vectors of $\Delta W^\star_{i,:}$. Define

$$\bar{Q} := \operatorname{blkdiag}(\bar{Q}_1, \dots, \bar{Q}_n), \qquad \Pi_Q := \bar{Q}\bar{Q}^\top. \tag{90}$$

By construction,

$$\|(I - \Pi_Q)\Delta W^\star\|_F^2 = \epsilon_{\mathrm{out}}. \tag{91}$$

Similarly, for each block column j, let $\bar{P}_j \in \mathbb{R}^{m\times r}$ contain the leading r right singular vectors of $\Delta W^\star_{:,j}$, and define

$$\bar{P} := \operatorname{blkdiag}(\bar{P}_1, \dots, \bar{P}_n), \qquad \Pi_P := \bar{P}\bar{P}^\top. \tag{92}$$

Then

$$\|\Delta W^\star(I - \Pi_P)\|_F^2 = \epsilon_{\mathrm{in}}. \tag{93}$$

Consider the doubly projected update

$$\widehat{\Delta W}_{\mathrm{cand}} := \Pi_Q \Delta W^\star \Pi_P. \tag{94}$$

Its (i,j)-th block is

$$\begin{aligned}(\widehat{\Delta W}_{\mathrm{cand}})_{ij} &= \bar{Q}_i \bar{Q}_i^\top \Delta W^\star_{ij} \bar{P}_j \bar{P}_j^\top \\ &= \bar{Q}_i \underbrace{\left(\bar{Q}_i^\top \Delta W^\star_{ij} \bar{P}_j\right)}_{Z^{\mathrm{cand}}_{ij}} \bar{P}_j^\top.\end{aligned} \tag{95}$$

Hence, $\widehat{\Delta W}_{\mathrm{cand}}$ is a feasible TriBLoRA update.

Its residual can be decomposed as

$$\begin{aligned}\Delta W^\star - \Pi_Q \Delta W^\star \Pi_P = (I - \Pi_Q)\Delta W^\star \\ + \Pi_Q \Delta W^\star (I - \Pi_P).\end{aligned} \tag{96}$$

The two terms in Eq. (96) are orthogonal in the Frobenius inner product because

$$(I - \Pi_Q)\Pi_Q = 0. \tag{97}$$

Therefore,

$$\begin{aligned}&\left\|\Delta W^\star - \widehat{\Delta W}_{\mathrm{cand}}\right\|_F^2 \\ &= \|(I - \Pi_Q)\Delta W^\star\|_F^2 + \|\Pi_Q \Delta W^\star (I - \Pi_P)\|_F^2 \\ &\le \epsilon_{\mathrm{out}} + \epsilon_{\mathrm{in}},\end{aligned} \tag{98}$$

where the last inequality follows because an orthogonal projection cannot increase the Frobenius norm:

$$\|\Pi_Q \Delta W^\star (I - \Pi_P)\|_F \le \|\Delta W^\star (I - \Pi_P)\|_F. \tag{99}$$

Since the globally optimal TriBLoRA approximation cannot be worse than this feasible construction,

$$E^{\star}_{\mathrm{TriB}} \le \epsilon_{\mathrm{out}} + \epsilon_{\mathrm{in}}. \tag{100}$$

Combining Eqs. (89) and (100) proves

$$\max\{\epsilon_{\mathrm{out}}, \epsilon_{\mathrm{in}}\} \le E^{\star}_{\mathrm{TriB}} \le \epsilon_{\mathrm{out}} + \epsilon_{\mathrm{in}}. \tag{101}$$

Structural gap to unrestricted rank-nr LoRA. By the Eckart–Young–Mirsky theorem, unrestricted rank-nr LoRA achieves

$$E^{\star}_{\mathrm{L},nr} = \sum_{k>nr} \sigma_k^2\left(\Delta W^{\star}\right). \tag{102}$$

Moreover, from Appendix A.3.1,

$$\mathcal{F}_{\mathrm{TriB}}(n,r) \subseteq \mathcal{F}_{\mathrm{L}}(nr), \tag{103}$$

and therefore

$$E^{\star}_{\mathrm{TriB}} - E^{\star}_{\mathrm{L},nr} \ge 0. \tag{104}$$

Subtracting $E^{\star}_{\mathrm{L},nr}$ from the two sides of Eq. (101) and combining the lower bound with nonnegativity gives

$$\left[\max\{\epsilon_{\mathrm{out}}, \epsilon_{\mathrm{in}}\} - E^{\star}_{\mathrm{L},nr}\right]_+ \le E^{\star}_{\mathrm{TriB}} - E^{\star}_{\mathrm{L},nr} \le \epsilon_{\mathrm{out}} + \epsilon_{\mathrm{in}} - E^{\star}_{\mathrm{L},nr}. \tag{105}$$

Thus, the difference between TriBLoRA and unrestricted rank-nr LoRA is not determined solely by their common global rank ceiling. Instead, it is controlled by the residual spectral energy that cannot be represented within the shared rank-r local input and output subspaces.

Special case: globally rank-nr target. If

$$\mathrm{rank}(\Delta W^{\star}) \le nr, \tag{106}$$

then $E^{\star}_{\mathrm{L},nr} = 0$ and Eq. (105) reduces to

$$\max\{\epsilon_{\mathrm{out}}, \epsilon_{\mathrm{in}}\} \le E^{\star}_{\mathrm{TriB}} \le \epsilon_{\mathrm{out}} + \epsilon_{\mathrm{in}}. \tag{107}$$

Hence, even when unrestricted rank-nr LoRA can represent the target exactly, TriBLoRA may retain a nonzero approximation floor if the target requires more than r independent directions within some local block row or block column.

A.3.3 Proof of Approximation Gains over MELoRA and BoRA

We provide the complete derivation of the matched approximation results in Section 4.2. The purpose of the matched comparison is to isolate interaction-level expressiveness: for each reference method, we first take its own optimal local subspaces and then keep these subspaces fixed while relaxing only its interaction constraint to the dense TriBLoRA interaction matrix.

Throughout this subsection, we use the non-degenerate local-rank setting in which the outer factors have rank r. Rank-deficient cases can be handled analogously by replacing r with the corresponding effective rank.

Common fixed-subspace decomposition. For any orthonormal local bases $Q_i, P_j \in \mathbb{R}^{m\times r}$, define

$$C_{ij} := Q_i^{\top} \Delta W^{\star}_{ij} P_j. \tag{108}$$

As established in Eq. (24), for any latent interaction $Z_{ij} \in \mathbb{R}^{r\times r}$,

$$\begin{aligned} &\left\|\Delta W^{\star}_{ij} - Q_i Z_{ij} P_j^{\top}\right\|_F^2 \\ &= \underbrace{\left\|\Delta W^{\star}_{ij}\right\|_F^2 - \|C_{ij}\|_F^2}_{E^{\mathrm{sub}}_{ij}(Q_i,P_j)} + \|C_{ij} - Z_{ij}\|_F^2 . \end{aligned} \tag{109}$$

Here, $E^{\mathrm{sub}}_{ij}(Q_i, P_j)$ is the irreducible approximation error induced by the selected local subspaces, while the second term measures any additional error caused by restricting the interaction matrix.

Since TriBLoRA allows arbitrary Z_{ij}, it can choose

$$Z^{\star}_{ij} = C_{ij}, \tag{110}$$

and therefore incurs no additional interaction-matrix error once the local subspaces are fixed.

Matched comparison with MELoRA. Let $\{Q_i^{\mathrm{ME}}, P_i^{\mathrm{ME}}\}_{i=1}^n$ denote local subspaces associated with an optimal MELoRA approximation, and define

$$C_{ij}^{\mathrm{ME}} := (Q_i^{\mathrm{ME}})^\top \Delta W_{ij}^\star P_j^{\mathrm{ME}}. \tag{111}$$

MELoRA permits only block-diagonal interactions. Hence, for $i = j$, the latent interaction can be optimized within the corresponding local subspaces, whereas for $i \neq j$ the reconstructed block is fixed to zero. Its optimal error can therefore be written as

$$\begin{aligned} E_{\mathrm{ME}}^\star = &\sum_i \left[\|\Delta W_{ii}^\star\|_F^2 - \left\|C_{ii}^{\mathrm{ME}}\right\|_F^2 \right] \\ &+ \sum_{i \neq j} \left\|\Delta W_{ij}^\star\right\|_F^2 . \end{aligned} \tag{112}$$

Now construct a matched TriBLoRA approximation using exactly the same subspaces $\{Q_i^{\mathrm{ME}}, P_j^{\mathrm{ME}}\}$, but allowing every interaction matrix to be optimized independently. Using Eq. (109),

$$E_{\mathrm{TriB,match}}^{\mathrm{ME}} = \sum_{i,j} \left[\left\|\Delta W_{ij}^\star\right\|_F^2 - \left\|C_{ij}^{\mathrm{ME}}\right\|_F^2 \right]. \tag{113}$$

Subtracting Eq. (113) from Eq. (112) gives

$$E_{\mathrm{ME}}^\star - E_{\mathrm{TriB,match}}^{\mathrm{ME}} = \sum_{i \neq j} \left\|C_{ij}^{\mathrm{ME}}\right\|_F^2 \geq 0. \tag{114}$$

Therefore, the matched gain is exactly the projected target energy contained in the spatial interactions that MELoRA removes. The gain is strictly positive if and only if

$$C_{ij}^{\mathrm{ME}} \neq 0 \tag{115}$$

for at least one $i \neq j$.

Finally, since the globally optimized TriBLoRA solution is no worse than the matched construction,

$$E_{\mathrm{TriB}}^\star \leq E_{\mathrm{TriB,match}}^{\mathrm{ME}}, \tag{116}$$

and thus

$$E_{\mathrm{ME}}^\star - E_{\mathrm{TriB}}^\star \geq \sum_{i \neq j} \left\|C_{ij}^{\mathrm{ME}}\right\|_F^2 . \tag{117}$$

Matched comparison with BoRA. We next consider the BoRA family defined in Section 4.2. Let an optimal BoRA solution have outer factors

$$B_i^{\mathrm{BoRA}} = Q_i^{\mathrm{BoRA}} R_i^{\mathrm{BoRA}}, \qquad A_j^{\mathrm{BoRA}} = S_j^{\mathrm{BoRA}} (P_j^{\mathrm{BoRA}})^\top, \tag{118}$$

where Q_i^{BoRA} and P_j^{BoRA} are orthonormal local bases and $R_i^{\mathrm{BoRA}}, S_j^{\mathrm{BoRA}} \in \mathbb{R}^{r \times r}$ are invertible.

Define the projected target interaction

$$C_{ij}^{\mathrm{BoRA}} := (Q_i^{\mathrm{BoRA}})^\top \Delta W_{ij}^\star P_j^{\mathrm{BoRA}}. \tag{119}$$

Let

$$\mathcal{D}_r := \left\{ D \in \mathbb{R}^{r \times r} : D \text{ is diagonal} \right\}. \tag{120}$$

Under the orthogonal coordinates in Eq. (118), a BoRA interaction matrix must belong to

$$\mathcal{Z}_{ij}^{\mathrm{BoRA}} := \left\{ R_i^{\mathrm{BoRA}} D S_j^{\mathrm{BoRA}} : D \in \mathcal{D}_r \right\}. \tag{121}$$

Hence, for the fixed BoRA-optimal factors, the pair-specific interaction residual is

$$\delta_{ij}^{\mathrm{BoRA}} := \min_{D \in \mathcal{D}_r} \left\| C_{ij}^{\mathrm{BoRA}} - R_i^{\mathrm{BoRA}} D S_j^{\mathrm{BoRA}} \right\|_F^2 . \tag{122}$$

Applying Eq. (109), the optimal BoRA error can be decomposed as

$$E^{\star}_{\mathrm{BoRA}} = \sum_{i,j} \Big[\big\|\Delta W^{\star}_{ij}\big\|_F^2 - \big\|C^{\mathrm{BoRA}}_{ij}\big\|_F^2 + \delta^{\mathrm{BoRA}}_{ij} \Big]. \tag{123}$$

Now keep the BoRA-optimal geometric subspaces $\{Q^{\mathrm{BoRA}}_i, P^{\mathrm{BoRA}}_j\}$ unchanged, but replace the restricted interaction family in Eq. (121) with unrestricted TriBLoRA interaction matrices. The optimal choice is

$$Z_{ij} = C^{\mathrm{BoRA}}_{ij}, \tag{124}$$

and therefore

$$E^{\mathrm{BoRA}}_{\mathrm{TriB,match}} = \sum_{i,j} \Big[\big\|\Delta W^{\star}_{ij}\big\|_F^2 - \big\|C^{\mathrm{BoRA}}_{ij}\big\|_F^2 \Big]. \tag{125}$$

Subtracting Eq. (125) from Eq. (123) gives

$$E^{\star}_{\mathrm{BoRA}} - E^{\mathrm{BoRA}}_{\mathrm{TriB,match}} = \sum_{i,j} \delta^{\mathrm{BoRA}}_{ij} \geq 0. \tag{126}$$

The matched gain is strictly positive if and only if at least one projected target interaction does not belong to its admissible BoRA interaction matrix family:

$$C^{\mathrm{BoRA}}_{ij} \notin \mathcal{Z}^{\mathrm{BoRA}}_{ij} \tag{127}$$

for some pair (i, j).

Since

$$E^{\star}_{\mathrm{TriB}} \leq E^{\mathrm{BoRA}}_{\mathrm{TriB,match}}, \tag{128}$$

we further obtain

$$E^{\star}_{\mathrm{BoRA}} - E^{\star}_{\mathrm{TriB}} \geq \sum_{i,j} \delta^{\mathrm{BoRA}}_{ij}. \tag{129}$$

Interpretation of the two structural gaps. The two matched comparisons isolate distinct limitations:

$$\begin{aligned} \text{MELoRA:} \quad & \sum_{i \neq j} \|C^{\mathrm{ME}}_{ij}\|_F^2, \\ \text{BoRA:} \quad & \sum_{i,j} \delta^{\mathrm{BoRA}}_{ij}. \end{aligned} \tag{130}$$

The MELoRA term measures energy in interaction channels that are completely removed by its spatially block-diagonal structure. The BoRA term instead measures the residual within existing interaction channels caused by restricting each pair-specific transformation to the family $R_i D S_j$ with diagonal D.

TriBLoRA removes both restrictions while preserving shared local subspaces: it retains all spatial block pairs and permits a dense $r \times r$ interaction matrix within each pair.

A.4 Block-Rank Allocation: Benefit, Budget, and Rank Ceilings

This subsection provides the derivations supporting Section 4.3. We first establish the allocation benefit in an idealized surrogate, then derive exact parameter-budget preservation and the pairwise rank-ceiling constraint for the actual two-sided TriBLoRA parameterization.

A.4.1 IDEALIZED ALLOCATION BENEFIT

Surrogate and feasible topologies. Let $\mathcal{I}_\ell$ be the local adaptation units in layer ℓ, with effective ranks $r_{\ell,b}$ for $b \in \mathcal{I}_\ell$. We use three simplifying assumptions: input- and output-side ranks are coordinated into a single effective rank, each rank unit has equal cost, and local approximation errors are evaluated independently. For a target local update $\Delta W^\star_{\ell,b}$, the Eckart–Young–Mirsky theorem gives

$$e_{\ell,b}(r_{\ell,b}) := \min_{\mathrm{rank}(X)\le r_{\ell,b}} \|\Delta W^\star_{\ell,b} - X\|_F^2 = \sum_{k>r_{\ell,b}} \sigma_k^2(\Delta W^\star_{\ell,b}). \tag{131}$$

For $\tau := \{r_{\ell,b}\}_{\ell,b}$, define the additive surrogate $\mathcal{E}(\tau) := \sum_\ell \sum_{b\in\mathcal{I}_\ell} e_{\ell,b}(r_{\ell,b})$. This effective-rank formulation isolates allocation granularity from the core cost and row/column coupling of the full parameterization.

Under budget K, the feasible topology spaces are

$$\mathcal{T}_{\mathrm{block}}(K) := \left\{ \tau : r_{\ell,b} \in \mathbb{Z}_{\ge 0},\ \sum_\ell \sum_{b\in\mathcal{I}_\ell} r_{\ell,b} \le K \right\}, \tag{132}$$

$$\mathcal{T}_{\mathrm{layer}}(K) := \{\tau \in \mathcal{T}_{\mathrm{block}}(K) : r_{\ell,b} = r_\ell,\ \forall b \in \mathcal{I}_\ell,\ \forall \ell\}. \tag{133}$$

Writing $\mathcal{E}^\star_s(K) := \min_{\tau\in\mathcal{T}_s(K)} \mathcal{E}(\tau)$ for $s \in \{\mathrm{layer}, \mathrm{block}\}$, set inclusion gives

$$\mathcal{T}_{\mathrm{layer}}(K) \subseteq \mathcal{T}_{\mathrm{block}}(K) \quad\Longrightarrow\quad \mathcal{E}^\star_{\mathrm{block}}(K) \le \mathcal{E}^\star_{\mathrm{layer}}(K). \tag{134}$$

Budget-preserving rank transfer. Equation (131) implies that adding one rank to a unit of rank r reduces its error by $\sigma^2_{r+1}(\Delta W^\star_{\ell,b})$, whereas removing one rank increases its error by $\sigma^2_r(\Delta W^\star_{\ell,b})$. Consider two units a, b in the same layer and suppress their common layer index. If $r_b \ge 1$ and unit a has remaining rank capacity, the transfer $r'_a = r_a + 1$, $r'_b = r_b - 1$ preserves the budget and gives

$$\Delta\mathcal{E} := \mathcal{E}(\tau') - \mathcal{E}(\tau) = -\sigma^2_{r_a+1}(\Delta W^\star_a) + \sigma^2_{r_b}(\Delta W^\star_b). \tag{135}$$

The transfer strictly improves approximation when $\sigma^2_{r_a+1}(\Delta W^\star_a) > \sigma^2_{r_b}(\Delta W^\star_b)$: the recovered target energy exceeds the energy lost at the donor unit.

For a layer-wise topology, $r_a = r_b = r_\ell$, recovering the exchange condition in Section 4.3. In particular, if an optimal layer-wise topology $\tau^\star_{\mathrm{layer}}$ admits a feasible transfer satisfying $\sigma^2_{r_\ell+1}(\Delta W^\star_a) > \sigma^2_{r_\ell}(\Delta W^\star_b)$, then

$$\mathcal{E}^\star_{\mathrm{block}}(K) \le \mathcal{E}(\tau') < \mathcal{E}(\tau^\star_{\mathrm{layer}}) = \mathcal{E}^\star_{\mathrm{layer}}(K). \tag{136}$$

A.4.2 EXACT PARAMETER-BUDGET PRESERVATION

We now return to the actual two-sided parameterization. Consider G equally sized output blocks and H equally sized input blocks, with $d_i = d_{\mathrm{out}}/G$ and $c_j = d_{\mathrm{in}}/H$, and retain the complete interaction grid. Relative to a uniform rank-r configuration, let a heterogeneous topology $\mathcal{T}$ preserve the two side-rank sums:

$$\sum_{i=1}^{G} r_{B,i} = Gr, \qquad \sum_{j=1}^{H} r_{A,j} = Hr. \tag{137}$$

Counting the entries of the outer factors and pair-specific cores yields

$$\begin{aligned} N_{\mathrm{param}}(\mathcal{T}) &= \sum_i d_i r_{B,i} + \sum_j c_j r_{A,j} + \sum_{i,j} r_{B,i} r_{A,j} \\ &= \frac{d_{\mathrm{out}}}{G} \sum_i r_{B,i} + \frac{d_{\mathrm{in}}}{H} \sum_j r_{A,j} + \left(\sum_i r_{B,i}\right)\left(\sum_j r_{A,j}\right) \\ &= r(d_{\mathrm{out}} + d_{\mathrm{in}}) + GHr^2. \end{aligned} \tag{138}$$

Thus, redistribution preserves both the total outer-factor cost and the total core cost, although individual factor dimensions change. For $d_{\mathrm{out}} = d_{\mathrm{in}} = d$ and $G = H = n$, this becomes $N_{\mathrm{param}}(\mathcal{T}) = 2dr + n^2r^2$. The same argument applies module by module with r replaced by the prescribed module rank r_ℓ, establishing equal parameter counts between layer-adaptive and block-adaptive configurations with the same module-level budgets.

Scope. The equality requires equal block sizes on each side, separate preservation of the two side-rank sums, and a complete interaction grid. With unequal block sizes, the outer-factor costs are weighted rank sums; with sparse interactions, the core cost no longer factors into the product of side-rank sums. Preserving only a combined rank sum also need not preserve the parameter count.

A.4.3 Pairwise Rank-Ceiling Trade-off

For the heterogeneous topology, $B_i \in \mathbb{R}^{d_i \times r_{B,i}}$, $A_j \in \mathbb{R}^{r_{A,j} \times c_j}$, and $M_{ij} \in \mathbb{R}^{r_{B,i} \times r_{A,j}}$. Each interaction therefore satisfies

$$\operatorname{rank}(B_i M_{ij} A_j) \le \rho_{ij}^{\mathrm{cap}} := \min(r_{B,i}, r_{A,j}). \tag{139}$$

Increasing only one side cannot raise this ceiling while the other side remains the bottleneck. Across the complete grid, define the mean pairwise rank ceiling as $\bar{\rho}_{\mathrm{cap}} := (GH)^{-1} \sum_{i,j} \min(r_{B,i}, r_{A,j})$. Using $\min(a, b) \le (a + b)/2$ and Eq. (137),

$$\begin{aligned} \bar{\rho}_{\mathrm{cap}} &\le \frac{1}{2GH} \sum_{i,j} (r_{B,i} + r_{A,j}) \\ &= \frac{1}{2} \left(\frac{1}{G} \sum_{i} r_{B,i} + \frac{1}{H} \sum_{j} r_{A,j} \right) = r. \end{aligned} \tag{140}$$

Equality requires $r_{B,i} = r_{A,j}$ for every interacting pair. On a complete grid, this reduces to the uniform rank-r topology. We refer to the reduction in the mean pairwise ceiling under non-uniform allocation as *structural rank loss*. This quantity is a structural upper bound, not the numerical rank of a trained update or a task-weighted measure of approximation quality.

A one-sided alternative, $\Delta W_i = B_i M_i A_{\mathrm{share}}$, has rank at most $\min(r_{B,i}, r_A)$, where r_A is the globally shared input rank. TriBLoRA instead permits both local input and output capacities to vary while retaining cross-block interactions.

Joint interpretation. The idealized analysis identifies the value of allocating rank where marginal spectral gains are larger, while the actual parameterization preserves the budget under the stated constraints and introduces pairwise rank bottlenecks. Useful topologies balance importance-aware capacity placement against these two-sided constraints. Appendix C.5 examines this balance through adaptive-degree, rank-ceiling, and rank-placement analyses.

B Experimental Setup

This appendix specifies the experimental protocol used for the results in the main paper and Appendix C. Appendix B.1 describes the hardware environments, benchmarks, evaluation settings, and random-seed protocol. Appendix B.2 reports the training hyperparameters for RoBERTa-base on GLUE and LLaMA-2-7B on Alpaca-cleaned. Appendix B.3 then describes the TriBLoRA-specific initialization and optimization settings, including interaction-matrix initialization, early optimization dynamics, and learning-rate scaling of the pair-specific cores. Finally, Appendix B.4 describes the one-shot gradient-based probing procedure used to estimate local importance and construct the block-adaptive rank topology before fine-tuning.

Additional empirical comparisons and diagnostic analyses are reported in Appendix C.

B.1 Experimental Environment, Benchmarks, and Random Seeds

GLUE experiments are conducted on NVIDIA GeForce RTX 3090 GPUs, while LLaMA-2-7B experiments use NVIDIA GeForce RTX 4090 and RTX A6000 GPUs.

GLUE. We fine-tune RoBERTa-base (Liu et al., 2019) on eight tasks from GLUE (Wang et al., 2019), adapting the query and value projections. MNLI evaluates natural language inference across multiple genres; SST-2 evaluates sentiment classification; MRPC and QQP evaluate paraphrase

identification; CoLA evaluates linguistic acceptability; QNLI evaluates question–sentence entailment; RTE evaluates textual entailment; and STS-B measures graded semantic similarity. Results are reported using the corresponding task metrics.

Among the four smaller tasks, MRPC, CoLA, and STS-B use five seeds $\{41, 42, 43, 44, 45\}$, whereas RTE uses ten seeds $\{41, \ldots, 50\}$ because of its higher variability. The four larger tasks, MNLI, SST-2, QNLI, and QQP, use three seeds $\{41, 42, 43\}$, reflecting their more stable results and the higher cost of additional runs.

LLaMA-2-7B. We fine-tune LLaMA-2-7B (Touvron et al., 2023) on Alpaca-cleaned (Taori et al., 2023; Yahma, 2023) and evaluate the resulting models on downstream benchmarks. MMLU (Hendrycks et al., 2021) measures broad subject knowledge through multiple-choice questions; BBH (Suzgun et al., 2022) contains challenging language and reasoning tasks; DROP (Dua et al., 2019) evaluates reading comprehension requiring discrete reasoning over passages; WinoGrande (Sakaguchi et al., 2021) evaluates commonsense reasoning through pronoun resolution; and ARC-Challenge (ARC-C) (Clark et al., 2018) evaluates grade-school science reasoning using challenging multiple-choice questions. HumanEval (Chen et al., 2021) additionally evaluates code generation through functional correctness under unit tests.

LLaMA-2-7B experiments target five training seeds. All methods use the same Alpaca training examples, prompt template, sequence length, optimizer schedule, and target modules. Within each downstream benchmark, decoding and evaluation settings are held fixed across methods.

Unless otherwise stated, we report the mean and sample standard deviation over completed runs. Cross-task averages weight all included tasks equally.

B.2 Training Hyperparameters

GLUE. The GLUE training protocol uses the task-specific settings shown below. For TriBLoRA, the listed learning rate corresponds to the outer-factor learning rate.

Task	Learning rate	Batch size	Epochs	Max. length
MNLI	5×10^{-4}	64	10	256
SST-2	5×10^{-4}	64	30	256
MRPC	4×10^{-4}	64	30	256
CoLA	4×10^{-4}	32	20	256
QNLI	4×10^{-4}	64	20	256
QQP	4×10^{-4}	128	25	256
RTE	4×10^{-4}	64	20	512
STS-B	4×10^{-4}	64	25	256

The GLUE protocol uses weight decay 0.1, warm-up ratio 0.1, and adapter dropout 0.05. The default TriBLoRA configuration uses $n = 4$ and $r = 8$, with $\alpha = 2r = 16$, corresponding to an adapter scale of 2.

LLaMA-2-7B. For LLaMA experiments, we fine-tune LLaMA-2-7B on Alpaca-cleaned while adapting the query and value projections. We use a learning rate of 3×10^{-4}, an effective batch size of 128 with micro-batch size 4 and gradient accumulation, four training epochs, and a maximum sequence length of 256. AdamW uses linear learning-rate decay, 100 warm-up steps, weight decay 0, and adapter dropout 0.05. We hold out $2{,}000$ examples for validation and evaluate and save checkpoints every 100 optimizer steps. All downstream evaluations use the checkpoint selected by validation loss. The default TriBLoRA configuration uses $n = 4$ and $r = 8$. We set $\alpha = 2r$, giving an adapter scale of 2.

B.3 Interaction-Matrix Initialization and Optimization Dynamics

TriBLoRA inserts a pair-specific dense interaction matrix between the shared outer factors:

$$\Delta W_{ij} = B_i M_{ij} A_j, \tag{141}$$

where $B_i \in \mathbb{R}^{d_i \times r_{B,i}}$, $A_j \in \mathbb{R}^{r_{A,j} \times c_j}$, and $M_{ij} \in \mathbb{R}^{r_{B,i} \times r_{A,j}}$. For both initialization schemes, we set $B_i^{(0)} = 0$ and initialize A_j using the standard Kaiming initialization adopted in LoRA.

Unless explicitly stated otherwise, TriBLoRA uses rectangular-identity initialization,

$$M_{ij}^{(0)} = I_{r_{B,i} \times r_{A,j}}, \tag{142}$$

together with an interaction-matrix learning rate $\eta_M = 0.3\,\eta_{\text{outer}}$ ($\lambda_M = 0.3$). When $r_{B,i} = r_{A,j} = r$, this reduces to the square identity I_r. Experiments using the alternative Kaiming–Frobenius initialization are explicitly denoted as *K-F*; in these experiments, M_{ij} is initialized as a dense random matrix whose Frobenius norm is matched to that of the corresponding rectangular identity, and we use $\lambda_M = 1.0$ unless otherwise specified.

The initialization of M_{ij} determines the initial transformation between the two local subspaces and also affects the early optimization dynamics. We therefore analyze the interaction-matrix optimization in four stages. We first introduce the default rectangular-identity initialization and then the scale-matched dense Kaiming–Frobenius (K-F) alternative. We next derive how these choices enter the early gradients when $B_i^{(0)} = 0$, before empirically studying the interaction-matrix learning-rate ratio relative to the shared outer factors.

B.3.1 Identity-based initialization

Our default initialization follows direct compatibility with standard LoRA. In the equal-rank case, $r_{B,i} = r_{A,j} = r$, setting

$$M_{ij}^{(0)} = I_r \tag{143}$$

gives $B_i M_{ij}^{(0)} A_j = B_i A_j$. Thus, before the pair-specific interaction matrices deviate from the identity, each local factorization follows the interaction rule induced by the corresponding LoRA block decomposition. This provides a natural starting point from which pair-specific transformations can be learned during training.

When the two side ranks differ, $r_{B,i} \neq r_{A,j}$, we instead use the rectangular identity $I_{r_{B,i} \times r_{A,j}}$, whose first $k_{ij} := \min(r_{B,i}, r_{A,j})$ diagonal entries are one and whose remaining entries are zero. This preserves the identity mapping over the shared latent coordinates, while introducing a deterministic coordinate alignment at initialization.

B.3.2 Kaiming–Frobenius initialization

As an alternative to the default identity initialization, we additionally consider Kaiming–Frobenius (K-F) initialization. Inspired by initialization strategies used in related tri-factor and structured LoRA methods, we first sample

$$\widetilde{M}_{ij} \sim \text{Kaiming}(r_{B,i} \times r_{A,j})\,, \tag{144}$$

and then normalize its Frobenius norm to that of the corresponding rectangular identity:

$$M_{ij}^{(0)} = \sqrt{k_{ij}}\,\frac{\widetilde{M}_{ij}}{\|\widetilde{M}_{ij}\|_F}, \qquad k_{ij} = \min(r_{B,i}, r_{A,j}). \tag{145}$$

Because $\|I_{r_{B,i} \times r_{A,j}}\|_F = \sqrt{k_{ij}}$, the two schemes have the same overall interaction-matrix magnitude, whereas K-F distributes this magnitude across a dense transformation. In the equal-rank case, $\|M_{ij}^{(0)}\|_F = \sqrt{r}$.

Unlike diagonal modulation schemes, we impose no positivity or diagonal constraint on M_{ij}. Both positive and negative entries and off-diagonal interactions are retained, allowing the core to model general pair-specific transformations between the two local subspaces.

B.3.3 Early optimization dynamics

Since $B_i^{(0)} = 0$, the initial adapter update satisfies

$$\Delta W_{ij}^{(0)} = B_i^{(0)} M_{ij}^{(0)} A_j^{(0)} = 0, \tag{146}$$

regardless of the interaction-matrix initialization, so the adapted model initially matches the pre-trained model.

Let $\Gamma_{ij} := \partial \mathcal{L}/\partial \Delta W_{ij}$ denote the upstream gradient for pair (i, j). Omitting the common adapter scaling factor, the three gradients are

$$\frac{\partial \mathcal{L}}{\partial B_i} = \sum_{j=1}^{H} \Gamma_{ij} A_j^\top M_{ij}^\top, \tag{147}$$

$$\frac{\partial \mathcal{L}}{\partial A_j} = \sum_{i=1}^{G} M_{ij}^\top B_i^\top \Gamma_{ij}, \tag{148}$$

$$\frac{\partial \mathcal{L}}{\partial M_{ij}} = B_i^\top \Gamma_{ij} A_j^\top. \tag{149}$$

At initialization, $B_i^{(0)} = 0$ implies

$$\left.\frac{\partial \mathcal{L}}{\partial A_j}\right|_{t=0} = 0, \qquad \left.\frac{\partial \mathcal{L}}{\partial M_{ij}}\right|_{t=0} = 0, \tag{150}$$

whereas

$$\left.\frac{\partial \mathcal{L}}{\partial B_i}\right|_{t=0} = \sum_{j=1}^{H} \Gamma_{ij} (A_j^{(0)})^\top (M_{ij}^{(0)})^\top \tag{151}$$

is generally nonzero. Thus, B_i begins to adapt immediately; A_j and M_{ij} receive nonzero gradients only after B_i has moved away from zero. Although both schemes yield the same zero initial update, the initial interaction matrix therefore directly modulates the first nonzero gradient of B_i. Identity initialization preserves the LoRA latent mapping, whereas Kaiming–Frobenius initialization introduces scale-controlled dense mixing from the first update.

After B_i becomes nonzero, the shared outer factors aggregate gradients over multiple interaction paths: $\nabla_{B_i} \mathcal{L}$ sums over all H input-side blocks and $\nabla_{A_j} \mathcal{L}$ sums over all G output-side blocks. In contrast, $\nabla_{M_{ij}} \mathcal{L}$ contains only the gradient of pair (i, j). This is an aggregation difference, not a difference in the number of backward passes: all parameter groups are updated in the same pass, but their gradient magnitudes and statistics need not be comparable.

B.3.4 Interaction-Matrix Learning Rate

We parameterize the interaction-matrix learning rate as

$$\eta_M = \lambda_M \eta_{\text{outer}}, \qquad \eta_A = \eta_B = \eta_{\text{outer}}, \tag{152}$$

where λ_M controls the optimization rate of pair-specific interaction matrices relative to shared outer factors. For a parameter group X, its relative update magnitude under gradient descent is approximately

$$\rho_X := \frac{\|\Delta X\|_F}{\|X\|_F + \epsilon} \approx \eta_X \frac{\|\nabla_X \mathcal{L}\|_F}{\|X\|_F + \epsilon}. \tag{153}$$

Consequently, equal nominal learning rates do not generally imply equal relative update magnitudes. With AdamW, first- and second-moment estimates further weaken a direct correspondence between nominal learning rate and optimization speed.

We vary λ_M while holding the outer-factor learning rate, adapter rank, block number, and remaining hyperparameters fixed. The ablations use $n = 4$ and $r = 8$, with ten random seeds for RTE and five each for MRPC, CoLA, and STS-B. Under identity-based initialization, the best four-task average occurs at $\lambda_M = 0.3$; under Kaiming–Frobenius initialization, it occurs at $\lambda_M = 1.0$. Raising the interaction-matrix learning rate further does not produce a consistent improvement. We therefore use identity-based initialization with its corresponding interaction-matrix learning-rate setting unless otherwise specified. The Kaiming–Frobenius and identity-based results are reported in Table 5.

Table 5: Effect of the interaction-matrix learning-rate ratio under identity-based and Kaiming–Frobenius initialization.

Method	LR ratio λ_M	Params/matrix	RTE (%)	MRPC (%)	CoLA (%)	STS-B (%)	Avg. (%)
LoRA	–	12,288	75.81±1.25	90.28±0.49	61.63±2.25	90.62±0.18	79.59
TriBLoRA	0.1	13,312	76.10±1.37	90.29±0.61	62.27±2.02	90.72±0.26	79.85
TriBLoRA	0.3	13,312	76.93±0.82	90.72±0.45	62.02±0.69	90.73±0.17	**80.10**
TriBLoRA	0.6	13,312	74.91±2.06	90.30±0.57	62.36±2.01	90.68±0.08	79.56
TriBLoRA	0.8	13,312	75.88±1.20	90.10±0.45	62.92±1.75	90.62±0.17	79.88
TriBLoRA	1.0	13,312	75.78±1.30	90.52±0.53	62.17±1.94	90.72±0.29	79.80
TriBLoRA	2.0	13,312	75.85±1.27	90.58±0.62	62.12±1.29	90.72±0.20	79.82
TriBLoRA	4.0	13,312	75.34±4.57	90.47±0.37	62.00±1.30	90.78±0.17	79.65
TriBLoRA (K-F)	0.1	13,312	74.80±1.17	90.61±0.60	60.41±0.89	90.87±0.17	79.17
TriBLoRA (K-F)	0.3	13,312	75.05±2.48	90.31±0.29	60.94±1.35	90.79±0.16	79.27
TriBLoRA (K-F)	0.6	13,312	73.54±7.50	90.59±0.23	60.92±1.81	90.77±0.13	78.95
TriBLoRA (K-F)	0.8	13,312	74.95±1.07	91.07±0.70	61.08±1.06	90.74±0.23	79.46
TriBLoRA (K-F)	1.0	13,312	75.85±1.45	91.15±0.49	62.19±0.64	90.70±0.24	**79.97**
TriBLoRA (K-F)	2.0	13,312	76.06±1.49	90.67±0.56	60.53±1.47	90.72±0.24	79.49
TriBLoRA (K-F)	4.0	13,312	75.63±0.92	90.72±0.49	62.02±1.57	90.84±0.18	79.80

B.4 Gradient-Based Block-Rank Topology Construction

The block-adaptive topology is determined *before* the main fine-tuning process and does not use the performance of a trained adapter. Inspired by one-shot gradient-based rank planning such as GoRA (He et al., 2025), we first probe the pretrained model to estimate task-dependent local importance, then deterministically map these statistics to integer input- and output-side ranks under a fixed module-level budget. The mapping takes the prescribed module-level base rank r_ℓ from the corresponding layer-level allocation as input.

Gradient-based importance probing. For each task, we use the first $T = 100$ mini-batches of the training data. The pretrained parameters remain unchanged and dropout is disabled, while gradients of the task loss are computed with respect to the adapted query and value projection weights. For an adapted matrix W_ℓ, we use $n = 4$ blocks per side and estimate the (i, j)-th path saliency as

$$S_{\ell,ij} = \frac{1}{T} \sum_{t=1}^{T} \sum_{p \in \mathcal{B}_{\ell,ij}} \left| W_{\ell,p} \frac{\partial \mathcal{L}_t}{\partial W_{\ell,p}} \right|, \qquad T = 100, \tag{154}$$

where $\mathcal{B}_{\ell,ij}$ contains the entries of block (i, j). The output- and input-side scores are $S_{B,\ell}(i) = \sum_j S_{\ell,ij}$ and $S_{A,\ell}(j) = \sum_i S_{\ell,ij}$.

Module-wise heterogeneity calibration. For each side, we measure relative dispersion using the population coefficient of variation,

$$\mathrm{CV}_{B,\ell} = \frac{\mathrm{std}_0(S_{B,\ell})}{\max(\mathrm{mean}(S_{B,\ell}), 10^{-12})}, \qquad \mathrm{CV}_{A,\ell} = \frac{\mathrm{std}_0(S_{A,\ell})}{\max(\mathrm{mean}(S_{A,\ell}), 10^{-12})},$$

and combine the two sides as $h_\ell = \sqrt{(\mathrm{CV}_{A,\ell}^2 + \mathrm{CV}_{B,\ell}^2)/2}$. For each task, let q_{50} and q_{90} denote the median and 0.9 quantile of $\{h_\ell\}$ over all adapted modules. To avoid a degenerate division when the median heterogeneity is numerically zero, define $\widetilde{q}_{50} := \max(q_{50}, 10^{-12})$. The calibrated expansion factor is

$$\rho_{\mathrm{base},\ell} = \begin{cases} 1 + 0.25\, h_\ell / \widetilde{q}_{50}, & h_\ell \le q_{50}, \\ 1.25 + 0.25\, (h_\ell - q_{50})/(q_{90} - q_{50}), & q_{50} < h_\ell < q_{90}, \\ 1.50, & h_\ell \ge q_{90}, \end{cases} \tag{155}$$

followed by clipping to $[1, 1.50]$. When $q_{90} - q_{50} \le 10^{-12}$ and $h_\ell > q_{50}$, we directly set $\rho_{\mathrm{base},\ell} = 1.50$.

Adaptive degree and target maximum rank. For a module with prescribed base rank r_ℓ, both sides preserve the same rank budget,

$$\sum_{i=1}^{4} r_{B,\ell,i} = \sum_{j=1}^{4} r_{A,\ell,j} = 4r_\ell, \qquad 4 \le r_{B,\ell,i}, r_{A,\ell,j} \le 16. \tag{156}$$

The largest feasible local rank under this budget is $m_{\mathrm{feas},\ell} = \min(16, 4r_\ell - 12)$. Given adaptive degree γ, we set

$$\rho_{\gamma,\ell} = \min\left\{1 + \gamma(\rho_{\mathrm{base},\ell} - 1),\ 1.50,\ \frac{m_{\mathrm{feas},\ell}}{r_\ell}\right\}, \qquad m_{\mathrm{cont},\ell} = r_\ell \rho_{\gamma,\ell}.$$

The target maximum local rank is the nearest feasible integer,

$$m_\ell = \arg\min_{m \in \{r_\ell, \ldots, m_{\mathrm{feas},\ell}\}} |m - m_{\mathrm{cont},\ell}|, \tag{157}$$

with half-integer ties resolved toward the smaller integer. For example, $r_\ell = 8$, $\rho_{\mathrm{base},\ell} = 1.25$, and $\gamma = 1.25$ give $m_{\mathrm{cont},\ell} = 10.5$ and hence $m_\ell = 10$. When $\gamma = 0$, the implementation directly returns the uniform topology $[r_\ell, r_\ell, r_\ell, r_\ell]$ on both sides.

Continuous rank targets. For either side, let $S = (S_1, \ldots, S_4)$ denote its importance vector and $\bar{S}$ its mean. We first form $z_k = S_k / \max(\bar{S}, 10^{-12}) - 1$. If $\max_k z_k > 10^{-12}$, we normalize $u_k = z_k / \max_j z_j$. Since $\sum_k z_k = 0$, this also gives $\sum_k u_k = 0$. If all four scores are equal, we use a deterministic fallback: the smallest-index maximum receives $u_k = 1$, and the other three entries receive $-1/3$. The continuous targets are then

$$t_{B,\ell,i} = r_\ell + (m_\ell - r_\ell) u_{B,\ell,i}, \qquad t_{A,\ell,j} = r_\ell + (m_\ell - r_\ell) u_{A,\ell,j}, \tag{158}$$

whose sums remain $4r_\ell$ on each side.

Joint constrained integer projection. We do not independently clip or round the continuous targets. Instead, define the feasible set

$$\mathcal{C}_\ell(m_\ell) = \left\{ x \in \{4, \ldots, 16\}^4 : \sum_k x_k = 4r_\ell,\ \max_k x_k = m_\ell \right\}. \tag{159}$$

The output- and input-side ranks are jointly selected from $\mathcal{C}_\ell(m_\ell) \times \mathcal{C}_\ell(m_\ell)$ by minimizing

$$J = \sum_j (r_{A,\ell,j} - t_{A,\ell,j})^2 + \sum_i (r_{B,\ell,i} - t_{B,\ell,i})^2. \tag{160}$$

Thus, integerization, box constraints, the exact side-rank budgets, and the target maximum rank are enforced simultaneously rather than through independent rounding followed by budget correction.

If multiple candidates attain the same primary objective, we resolve ties deterministically. First, we maximize residual path alignment

$$\sum_{i,j} R_{\ell,ij} \min(r_{B,\ell,i}, r_{A,\ell,j}), \qquad R_{\ell,ij} = S_{\ell,ij} - \bar{S}_{\ell,i\cdot} - \bar{S}_{\ell,\cdot j} + \bar{S}_{\ell,\cdot\cdot}. \tag{161}$$

Remaining ties are resolved by maximizing $\sum_{i,j} \min(r_{B,\ell,i}, r_{A,\ell,j})$, corresponding to the total pairwise rank ceiling. If a tie still remains, we choose the lexicographically smallest vector $(r_{A,\ell,1}, \ldots, r_{A,\ell,4}, r_{B,\ell,1}, \ldots, r_{B,\ell,4})$. Floating-point ties use a relative tolerance of 10^{-12}.

Topology freezing and reuse. For a specified γ, the resulting topology is fully determined by the prescribed module-level ranks, probing saliency, and task-level calibration statistics, and is frozen before fine-tuning. For GLUE, each task has its own topology estimated from that task's probing data, and all training seeds for the same task reuse the topology constructed with probing seed 42. For LLaMA-2-7B, the topology is estimated once from the Alpaca training task and then shared across all downstream evaluation benchmarks.

C Additional Experiments and Analyses

This appendix complements the main experiments by examining TriBLoRA from five progressively more detailed perspectives.

Appendix C.1 quantifies the adapter parameter budget and shows how the pair-specific interaction matrices affect parameter cost under both uniform and block-adaptive ranks. Appendix C.2 positions TriBLoRA relative to closely related PEFT designs by separating interaction coverage, global versus pair-specific transformations, and diagonal versus dense middle factors. Appendix C.3 studies block granularity and compares the resulting parameter–performance trade-off with higher-rank LoRA controls. Appendix C.4 then moves from structural availability to learned functional utilization by combining training-time interaction-matrix restrictions with post-hoc interventions on trained cores. Finally, Appendix C.5 analyzes adaptive block-rank topology, including adaptive-degree sensitivity, interaction-matrix initialization under heterogeneous ranks, structural rank loss, topology visualization, and within-module rank-placement sensitivity.

C.1 Parameter Budgets and Adapter Cost

We first quantify the parameter cost of TriBLoRA and its relation to the block-rank allocation used in our experiments. For an adapted square projection of dimension d, rank-r LoRA contains

$$N_{\mathrm{LoRA}} = 2dr \tag{162}$$

trainable parameters. With an $n \times n$ block partition, uniform TriBLoRA retains the same outer-factor cost and introduces n^2 dense $r \times r$ interaction matrices:

$$N_{\mathrm{TriB}} = 2dr + n^2 r^2. \tag{163}$$

For comparison, replacing the dense interaction matrices with pair-specific diagonal connectors requires

$$N_{\mathrm{diag}} = 2dr + n^2 r. \tag{164}$$

The same total parameter budget is preserved under block-rank diversity. For a module with base rank r_ℓ, let $r_{B,\ell,i}$ and $r_{A,\ell,j}$ denote the output- and input-side local ranks, respectively. Our allocation preserves the two side-rank sums:

$$\sum_{i=1}^{n} r_{B,\ell,i} = \sum_{j=1}^{n} r_{A,\ell,j} = n r_\ell. \tag{165}$$

With equally sized blocks, the outer-factor cost therefore remains $2dr_\ell$, while the total interaction-matrix cost is

$$\begin{aligned} \sum_{i=1}^{n}\sum_{j=1}^{n} r_{B,\ell,i} r_{A,\ell,j} &= \left(\sum_{i} r_{B,\ell,i}\right)\left(\sum_{j} r_{A,\ell,j}\right) \\ &= n^2 r_\ell^2. \end{aligned} \tag{166}$$

Hence, Layer-Adaptive and Block-Adaptive TriBLoRA have identical parameter budgets for a fixed layer topology; block adaptation changes only how the available rank is distributed within each module.

For the uniform setting, the relative overhead over standard LoRA is

$$\frac{N_{\mathrm{TriB}} - N_{\mathrm{LoRA}}}{N_{\mathrm{LoRA}}} = \frac{n^2 r}{2d}. \tag{167}$$

Table 6: Per-matrix adapter cost for the RoBERTa-base GLUE setting ($d_{\text{in}} = d_{\text{out}} = 768$). Adapter footprint includes fixed adapter factors when applicable, whereas trainable parameters count only parameters updated during optimization. Relative overhead is measured against rank-8 LoRA. Counts exclude the classification head.

Method	Adaptation structure	Setting	Additional structure	Footprint	Trainable	Overhead
LoRA	BA	$r = 8$	None	12,288	12,288	–
LoRA (budget control)	BA	$r = 9$	None	13,824	13,824	$+12.50\%$
AdaLoRA	Adaptive low-rank factors	$r_{\text{init}} = 12$	Adaptive rank allocation	18,444	18,444	$+50.10\%$
HydraLoRA	Shared A, three gated B experts	$r = 3$	Three B experts	11,520	11,520	-6.25%
HydraLoRA	Shared A, three gated B experts	$r = 4$	Three B experts	14,592	14,592	$+18.75\%$
MELoRA	$\Delta W_{ii} = B_i A_i$	$n = 4,\ r = 8$	Diagonal block pairs only	12,288	12,288	0%
BoRA	$B_i D_{ij} A_j$	$n = 4,\ r = 8$	n^2 diagonal connectors	12,416	12,416	$+1.04\%$
MoSLoRA	BMA	$r = 8$	One global dense $r \times r$ core	12,352	12,352	$+0.52\%$
TLoRA	$C_{\mathfrak{f}} T A_{\mathfrak{f}}$	$r = 8$	Fixed outer factors, trainable core	12,353	65	$+0.53\%$
TriBLoRA	$B_i M_{ij} A_j$	$n = 4,\ r = 8$	n^2 dense pair-specific cores	13,312	13,312	$+8.33\%$

Thus, for fixed n and r, the absolute interaction-matrix cost $n^2 r^2$ is independent of the projection dimension, while its relative contribution decreases as d increases. In addition to same-rank comparisons, we use the nearest integer-rank LoRA configuration as a parameter-budget control when applicable.

For RoBERTa-base, TriBLoRA adds only $1{,}024$ parameters per adapted matrix over rank-8 LoRA, corresponding to an 8.33% overhead. Its $13{,}312$ parameters lie between rank-8 and rank-9 LoRA and remain below HydraLoRA with $r = 4$ ($14{,}592$ parameters). The additional cost has a direct structural interpretation: MoSLoRA adds one global dense $r \times r$ core, BoRA adds n^2 pair-specific diagonal connectors, and TriBLoRA adds n^2 pair-specific dense interaction matrices.

For LLaMA-2-7B, where $d = 4096$, rank-8 LoRA contains $65{,}536$ parameters per adapted matrix. The same additional $1{,}024$ TriBLoRA parameters increase this count to $66{,}560$, corresponding to only 1.56% overhead. Thus, the relative cost of pair-specific interaction modeling becomes smaller for wider projections.

C.2 Structural Comparison with Related PEFT Designs

To complement the benchmark comparison in the main experiments, we analyze the PEFT baselines most closely related to TriBLoRA from two complementary structural perspectives. First, we compare how block-based methods retain and parameterize local input–output interactions. Second, we compare how tri-factor methods organize their outer and middle factors. The GLUE and LLaMA-2-7B averages are reproduced from the main experiments to connect these structural distinctions with empirical performance. Parameter costs are reported separately in Appendix C.1.

Block interaction structure. Table 7 compares how block-structured methods organize local input–output interactions. MELoRA increases aggregate rank through isolated local updates but removes all off-diagonal block pairs. It scores below LoRA on GLUE (84.50 vs. 85.29), but above LoRA on LLaMA-2-7B (39.71 vs. 39.51). BoRA restores all interaction pairs through pair-specific diagonal transformations, but remains restricted to coordinate-wise modulation. TriBLoRA retains all interaction pairs while allowing each pair to learn a dense transformation within shared local subspaces, achieving averages of 85.53 and 40.06. These results suggest that both interaction coverage and the flexibility of the pair-specific transformation are important.

Tri-factor parameterization. Table 7 shows that introducing a dense middle factor alone is not sufficient. TLoRA concentrates adaptation in a global trainable core between fixed outer projections, while MoSLoRA jointly trains the outer factors and a global dense middle matrix. However,

$$\Delta W = BMA = (BM)A, \tag{168}$$

so MoSLoRA remains a global reparameterization of the rank-r LoRA function class rather than expanding its representational family. BoRA moves from a globally shared transformation to pair-specific ones, but restricts each D_{ij} to be diagonal. TriBLoRA instead combines trainable shared

Table 7: Structural comparison of related parameter-efficient adaptation methods: (a) block interaction structures and (b) tri-factor parameterizations. Performance columns reproduce averages from the main experiments. †The GLUE BoRA aggregate includes one collapsed CoLA seed.

(a) Block interaction structures

Method	Motivation	Formulation	Interaction structure	GLUE	LLaMA
LoRA	Global parameter-efficient low-rank adaptation	$\Delta W_{ij} = B_i A_j$	All block pairs are retained but share the same rank-r interaction factorization	85.29	39.51
MELoRA	Increase aggregate rank through isolated local updates	$\Delta W_{ii} = B_i A_i$, $\Delta W_{ij} = 0$ $(i \neq j)$	Only diagonal block pairs are retained; cross-block interactions are removed	84.50	39.71
BoRA	Restore cross-block interactions with pair-specific modulation	$\Delta W_{ij} = B_i D_{ij} A_j$	All pairs are retained, with a pair-specific diagonal transformation	83.46†	39.43
TriBLoRA	Separate reusable local subspaces from pair-specific interaction modeling	$\Delta W_{ij} = B_i M_{ij} A_j$	All pairs are retained; dense M_{ij} enables pair-specific mixing within shared local subspaces	**85.53**	**40.06**

(b) Tri-factor parameterizations

Method	Motivation	Formulation	Factor structure and functional role	GLUE	LLaMA
TLoRA	Concentrate adaptation into a compact trainable core	$\Delta W = C_{\text{fix}} M A_{\text{fix}}$	Fixed global outer factors with one trainable global dense core	73.70	37.40
MoSLoRA	Improve optimization through global latent-space mixing	$\Delta W = BMA$	Trainable global factors with one dense global core; $BMA = (BM)A$, so the function class remains rank-r LoRA	85.14	39.19
BoRA	Introduce low-cost pair-specific transformations	$\Delta W_{ij} = B_i D_{ij} A_j$	Trainable local outer factors with pair-specific diagonal cores	83.46†	39.43
TriBLoRA	Combine reusable local subspaces with flexible pair-specific transformations	$\Delta W_{ij} = B_i M_{ij} A_j$	Trainable local outer factors with pair-specific dense cores	**85.53**	**40.06**

local outer factors with pair-specific dense interaction matrices, allowing each local input–output pair to learn a general transformation within the shared subspaces.

Summary. Taken together, the two comparisons isolate three structural choices: whether cross-block interactions are retained, whether the middle transformation is global or pair-specific, and whether that transformation is diagonal or dense. TriBLoRA combines complete interaction coverage, shared trainable local subspaces, and dense pair-specific transformations. The empirical comparison therefore suggests that its advantage is associated not merely with introducing an additional middle factor or partitioning the weight matrix, but with the combination of shared local structure and flexible pair-specific interaction modeling.

C.3 Effect of Block Number and Parameter Efficiency

We study block granularity from two complementary perspectives. First, we vary the block number n at a fixed local rank $r = 8$ to determine how finer input–output partitioning affects TriBLoRA. Increasing n simultaneously introduces more localized input/output subspaces, more pair-specific interactions, and a larger global rank ceiling nr.

Table 8: Effect of the block number n at local rank $r = 8$, together with higher-rank LoRA controls: **(a)** GLUE and **(b)** LLaMA-2-7B.

	Method	n	r	Params/matrix	MNLI (%)	SST-2 (%)	MRPC (%)	CoLA (%)	QNLI (%)	QQP (%)	RTE (%)	STS-B (%)	Avg. (%)
(a) GLUE	LoRA	1	8	12,288	87.25±0.06	94.65±0.37	90.28±0.49	61.63±2.25	92.78±0.01	89.28±0.06	75.81±1.25	90.62±0.18	85.29
	LoRA	1	10	15,360	87.37±0.16	95.15±0.33	90.20±0.55	62.18±0.86	92.82±0.07	89.49±0.15	73.72±7.50	90.61±0.28	85.19
	LoRA	1	12	18,432	87.39±0.20	94.88±0.29	90.46±0.44	61.72±0.92	92.76±0.18	89.60±0.05	75.74±1.19	90.65±0.20	85.40
	LoRA	1	16	24,576	87.37±0.13	94.88±0.18	90.81±0.61	62.16±1.08	92.86±0.02	89.66±0.07	77.33±1.73	90.83±0.13	85.74
	TriBLoRA	2	8	12,544	87.21±0.19	94.50±0.40	90.21±0.76	61.78±1.31	92.89±0.08	89.31±0.05	75.96±1.74	90.71±0.19	85.32
	TriBLoRA	4	8	13,312	87.23±0.22	94.72±0.46	90.72±0.45	62.02±0.69	92.64±0.22	89.27±0.08	76.93±0.82	90.73±0.17	85.53
	TriBLoRA	6	8	14,592	87.39±0.30	94.95±0.30	90.78±0.88	61.87±2.29	92.81±0.08	89.29±0.08	75.38±1.30	90.73±0.26	85.40
	TriBLoRA	8	8	16,384	87.28±0.28	94.76±0.54	90.89±0.75	62.83±2.77	92.82±0.15	89.33±0.06	75.99±1.84	90.76±0.25	85.58

	Method	n	r	Params/matrix	Best Eval Loss ↓	MMLU (%)	BBH (%)	DROP (%)	WinoGrande (%)	ARC-C (%)	HumanEval (%)	Avg. (6 tasks) (%)
(b) LLaMA-2-7B	LoRA	1	8	65,536	0.8311	43.54±0.50	32.94±0.45	32.44±0.17	70.24±0.24	42.53±1.13	15.37±0.38	39.51
	LoRA	1	10	81,920	0.8299	43.37±0.26	32.85±0.29	32.20±0.34	70.69±0.44	42.58±1.55	15.32±0.53	39.50
	LoRA	1	12	98,304	0.8292	43.55±0.30	33.04±0.28	32.02±0.32	70.28±0.50	43.86±1.76	15.12±0.59	39.64
	LoRA	1	16	131,072	0.8278	43.74±0.30	33.41±0.13	32.35±0.50	70.36±0.31	44.57±0.87	15.24±0.81	39.94
	TriBLoRA	2	8	65,792	0.8332	43.78±0.25	33.01±0.40	32.39±0.43	70.56±0.61	42.87±0.95	15.29±0.69	39.65
	TriBLoRA	4	8	66,560	0.8333	44.18±0.65	33.55±0.34	32.53±0.65	70.54±0.58	44.57±1.88	15.00±0.73	40.06
	TriBLoRA	6	8	67,840	0.8304	44.18±0.50	33.50±0.21	32.10±0.37	70.32±0.20	45.68±1.40	15.05±0.44	40.14
	TriBLoRA	8	8	69,632	0.8311	44.29±0.59	33.41±0.43	32.37±1.03	70.21±0.65	42.78±1.39	15.05±0.72	39.69

Second, because increasing n also increases the number of interaction-matrix parameters, we compare against higher-rank LoRA controls to determine whether the observed gains can be explained simply by a larger adapter budget. Table 8 reports both comparisons on GLUE and LLaMA-2-7B. We first examine the effect of block granularity and its optimization trade-off, and then analyze parameter efficiency relative to increasing the global LoRA rank.

Effect of block granularity. Increasing the block number from $n = 2$ to $n = 4$ produces a clear improvement in both settings. On GLUE, the average increases from 85.32 to 85.53, while on LLaMA-2-7B the six-task average increases from 39.65 to 40.06. This trend is consistent with the structural analysis in Section 3: a finer partition provides more localized input–output subspaces, enables more fine-grained pair-specific transformations, and increases the global rank ceiling to nr.

The benefit does not increase monotonically with n. On GLUE, $n = 8$ reaches 85.58, only slightly above the $n = 4$ result, while $n = 6$ gives 85.40. On LLaMA-2-7B, $n = 6$ remains comparable to $n = 4$ (40.14 vs. 40.06), whereas $n = 8$ decreases to 39.69. Thus, most of the gain from increasing block granularity is already realized around $n = 4$, while further subdivision provides only limited or inconsistent improvements.

One possible explanation is the increasing optimization difficulty of the tri-factor parameterization. As discussed in Appendix B.3, the shared outer factors and pair-specific cores receive gradients through different aggregation structures and may therefore benefit from different optimization scales. In these experiments, we keep the optimization configuration fixed rather than retuning it for each n. Moreover, increasing n introduces more pair-specific cores and a more fragmented factorization. Although this raises the nominal interaction capacity and rank ceiling, the additional capacity may become harder to optimize effectively.

Parameter efficiency. The higher-rank LoRA controls show that the improvement of TriBLoRA cannot be attributed simply to a larger adapter budget. On GLUE, TriBLoRA with $n = 4$ uses $13{,}312$ parameters per adapted matrix and achieves 85.53, exceeding LoRA with $r = 10$ ($15{,}360$ parameters, 85.19) and $r = 12$ ($18{,}432$ parameters, 85.40). Increasing LoRA to $r = 16$ eventually reaches 85.74, but requires $24{,}576$ parameters per matrix.

The contrast is more pronounced on LLaMA-2-7B. TriBLoRA with $n = 4$ uses $66{,}560$ parameters per matrix and reaches a six-task average of 40.06, whereas LoRA with $r = 16$ reaches 39.94 using $131{,}072$ parameters. The $n = 6$ configuration similarly reaches 40.14 with only $67{,}840$ parameters. These comparisons indicate that allocating additional capacity to pair-specific block interactions can provide a more favorable parameter–performance trade-off than increasing the global LoRA rank alone.

Overall, the block-number study suggests that finer partitioning is beneficial up to a moderate granularity, but increasing nominal capacity beyond this point does not guarantee proportional perfor-

mance gains. We therefore use $n = 4$ as the default configuration, which provides a favorable balance between interaction granularity, representational capacity, parameter cost, and optimization complexity.

C.4 Interaction-Matrix Structure and Post-hoc Sensitivity

The structural comparison in Appendix C.2 establishes how TriBLoRA differs from related PEFT parameterizations. Section 5.2 further evaluates the effect of block granularity and interaction-matrix structure during training. This appendix provides the corresponding controls and diagnostic analyses.

Appendix C.4.1 discusses the single-block and diagonal-core references and examines the fixed-identity control in detail. Appendix C.4.2 then studies the sensitivity of trained models to post-hoc interventions on the learned cores, while Appendix C.4.3 summarizes their implications together with the initialization and optimization analysis in Appendix B.3.

C.4.1 Training-Time Structural Ablation

The complete training-time results are reported in Table 2 in the main text. We use four GLUE tasks, SST-2, MRPC, RTE, and STS-B, and local rank $r = 8$. Under the uniform-rank setting $r_{B,i} = r_{A,j} = r$, the three TriBLoRA core structures are fixed identity ($M_{ij} = I_r$, frozen), trainable diagonal ($M_{ij} = D_{ij}$, with diagonal D_{ij}), and trainable dense ($M_{ij} \in \mathbb{R}^{r\times r}$). These restrictions are imposed throughout optimization and therefore control which pair-specific transformations are available during training.

Single-block reference. With $n = 1$, TriBLoRA reduces to $\Delta W = B_1 M_{11} A_1$, which belongs to the same feasible rank-r update family as a global dense tri-factor parameterization such as MoSLoRA. On the four tasks, single-block TriBLoRA obtains an average of 87.85%, compared with 87.79% for MoSLoRA. Their optimization settings are not fully matched: single-block TriBLoRA inherits the default interaction-matrix learning-rate ratio $\lambda_M = 0.3$, whereas the MoSLoRA middle factor uses the same nominal learning rate as its outer factors. At $n = 1$, the cross-pair gradient aggregation present in the multi-block setting also disappears. We therefore treat this comparison as an optimization-sensitive reference rather than evidence of a difference in representational power.

Increasing to $n = 4$ introduces multiple shared local subspaces and pair-specific cores. The dense four-block configuration reaches 88.28%, compared with 87.85% for the single-block configuration and 87.84% for rank-8 LoRA.

Diagonal-core comparison and relation to BoRA. Within four-block TriBLoRA, the diagonal and dense variants retain the same outer-factor sharing structure and train pair-specific middle factors, differing primarily in whether the interaction matrices are constrained to diagonal modulation or allowed to be dense. Their four-task averages are 88.06% and 88.28%, respectively. The dense variant improves SST-2, MRPC, and RTE, while the diagonal variant is slightly higher on STS-B. The largest task-level difference is on RTE ($76.14\% \rightarrow 76.93\%$).

BoRA provides a complementary pair-specific diagonal reference but is not identical to the controlled diagonal TriBLoRA variant. Its connector parameterization, initialization, and optimization protocol differ, and its RTE result exhibits substantially larger variance in our experiments. We therefore use the within-TriBLoRA diagonal-to-dense comparison as the cleaner control for interaction-matrix flexibility, while BoRA remains an external structural reference.

Fixed-identity equivalence to LoRA. The fixed-identity configuration requires separate consideration because it is algebraically equivalent to a block decomposition of standard rank-r LoRA. With $M_{ij} = I_r$, each interaction satisfies $\Delta W_{ij} = B_i A_j$. Defining $B = [B_1^\top, \ldots, B_n^\top]^\top$ and $A = [A_1, \ldots, A_n]$ therefore gives $\Delta W = BA$. Hence, for matched outer factors, fixed-identity TriBLoRA and rank-r LoRA represent exactly the same update family.

For the controlled comparison, we directly partition and copy the LoRA A and B tensors together with the classifier head into the corresponding fixed-identity TriBLoRA model. The production implementations nevertheless evaluate the algebraically identical update through different floating-

Table 9: Post-hoc interventions on learned TriBLoRA interaction matrices under identity and Kaiming–Frobenius (K-F) initialization. No model is retrained after intervention.

Initialization	Core treatment	RTE	MRPC	CoLA	STS-B	MNLI	SST-2	QNLI	QQP	Avg.
Identity	Original dense M_{ij}	76.93±0.82	90.72±0.45	62.02±0.69	90.83±0.17	87.24±0.22	94.53±0.29	92.62±0.20	89.27±0.08	85.52
	Diagonalized M_{ij}	76.57±0.84	90.60±0.44	62.05±1.08	90.86±0.17	87.26±0.09	94.38±0.11	92.68±0.24	89.19±0.10	85.45
	Identity replacement	76.46±0.88	90.64±0.41	61.80±1.06	90.86±0.17	87.18±0.05	94.30±0.24	92.70±0.18	89.18±0.13	85.39
	Pair permutation	76.46±0.70	90.62±0.60	62.32±0.83	90.86±0.16	87.19±0.17	94.36±0.38	92.65±0.16	89.18±0.10	85.45
K-F	Original dense M_{ij}	75.81±1.42	91.15±0.49	62.19±0.64	90.87±0.22	87.29±0.07	94.72±0.41	92.68±0.06	89.24±0.17	85.49
	Diagonalized M_{ij}	51.16±3.68	38.22±24.58	2.69±2.46	21.69±3.88	34.49±2.47	78.17±9.90	51.21±0.80	57.68±8.32	41.91
	Identity replacement	49.17±3.13	53.28±27.64	6.47±8.43	3.12±12.50	34.56±2.57	51.95±3.50	50.71±0.28	44.24±12.15	36.69
	Pair permutation	50.79±1.77	59.37±16.45	5.29±2.53	9.95±13.14	34.15±1.67	60.78±2.25	51.44±1.11	43.40±1.53	39.40

point computation paths. Standard LoRA uses a single concatenated A projection followed by B, whereas the blockwise TriBLoRA path computes local projections separately and accumulates them before applying the corresponding output-side factors.

C.4.2 POST-HOC INTERVENTIONS ON LEARNED CORES

The training-time experiments above examine how restricting the available interaction structure affects the solution learned during optimization. However, allowing dense pair-specific interaction matrices does not by itself establish that a trained model ultimately relies on their additional degrees of freedom. We therefore perform complementary post-hoc interventions on completed checkpoints without retraining.

All experiments use the uniform $n = 4$, $r = 8$ configuration. We separately analyze checkpoints trained with the default identity initialization and with Kaiming–Frobenius (K-F) initialization. The K-F checkpoints use an interaction-matrix learning-rate ratio of $\lambda_M = 1.0$. The two initialization schemes correspond to separate checkpoint cohorts and are not pooled.

We consider three interventions. **(i)** *Diagonalization* removes all off-diagonal entries of each learned M_{ij}. **(ii)** *Identity replacement* replaces each learned core with I_r. **(iii)** *Pair permutation* preserves the learned core matrices but reassigns them across block pairs. These interventions probe how sensitive the trained predictor is to removing off-diagonal entries, replacing the learned cores by identities, or reassigning cores across block pairs in the learned latent coordinate system. Accordingly, they characterize how the learned parameterization depends on the realized interaction-core structure. For pair permutation, three random permutations are evaluated and averaged within each training seed.

RTE is evaluated over ten training seeds, MRPC, CoLA, and STS-B over five, and MNLI, SST-2, QNLI, and QQP over three. STS-B uses Pearson correlation in this post-hoc evaluation, whereas the main trained-model table reports the checkpoint-selection score used during training.

Identity-initialized checkpoints. The identity-initialized models are largely insensitive to all three post-hoc interventions. The average score changes only from 85.52 to 85.45 after diagonalization, 85.39 after identity replacement, and 85.45 after pair permutation. In particular, removing the off-diagonal entries of M_{ij} produces almost no aggregate degradation, and replacing the learned interaction matrices with identities preserves most of the original performance.

These results indicate that the final identity-initialized predictor depends only weakly on the fully dense and pair-specific structure of the learned interaction matrices. The optimization therefore appears to retain a strong identity-like pathway, with much of the learned adaptation remaining functional even when dense latent mixing or pair-specific core assignment is removed.

K-F-initialized checkpoints. K-F initialization produces a qualitatively different behavior. The original dense-core checkpoints obtain an average score of 85.49, whereas diagonalization reduces it to 41.91. Identity replacement further reduces the average to 36.69, and pair permutation gives 39.40.

The three interventions provide complementary evidence about learned core utilization. K-F-initialized checkpoints are substantially more sensitive to all three interventions. The large degrada-

tion after diagonalization indicates that the trained predictor depends strongly on the learned dense core realization in its current latent coordinates. Likewise, the degradation after identity replacement and pair permutation shows strong dependence on the learned core values and their association with particular block pairs. Together, these interventions show that K-F checkpoints depend more strongly on the realized core structure, indicating stronger co-adaptation between the shared outer factors and interaction cores.

Paired loss evidence. Paired evaluation losses show the same qualitative distinction. Under identity initialization, intervention-induced loss changes are small and mixed in sign: the loss increases on only $4/8$, $4/8$, and $3/8$ tasks after diagonalization, identity replacement, and pair permutation, respectively. Under K-F initialization, the corresponding losses increase on $7/8$, $8/8$, and $8/8$ tasks. RTE is the only exception under diagonalization and also exhibits substantially higher variance.

Because raw loss scales differ across tasks, we use these results only as within-task paired evidence rather than averaging loss changes across benchmarks. Their consistency with Table 9 nevertheless supports the same functional distinction between the two initialization schemes.

C.4.3 Implications for Interaction-Matrix Utilization

The training-time and post-hoc analyses address distinct but complementary questions. Training-time restrictions determine which pair-specific transformations are available during optimization, whereas post-hoc interventions reveal which properties of those transformations are actually used by the final predictor.

The training-time results show that, within the controlled $n = 4$ TriBLoRA setting, relaxing the interaction constraint from diagonal to dense yields a further improvement in average performance. The fixed-identity endpoint provides a useful structural reference, but its direct comparison with LoRA is affected by numerical differences between algebraically equivalent FP16 computation paths and should therefore not be interpreted as a difference in representational capacity.

The post-hoc results additionally show that structural availability does not necessarily imply functional utilization. With identity initialization, the trained models remain largely robust to perturbations of the learned cores in the current latent coordinate system. The K-F setting, in contrast, produces models that are substantially more sensitive to diagonalization, identity replacement, and core reassignment. Together with the analysis in Appendix B.3, these results indicate stronger co-adaptation between the shared outer factors and interaction cores under the K-F setting.

Taken together, these observations distinguish the *availability* of interaction-specific degrees of freedom from their *utilization*. The dense TriBLoRA parameterization makes general pair-specific transformations available during training, while initialization and optimization influence how strongly the resulting predictor depends on the learned core realization. This distinction complements the initialization and interaction-matrix learning-rate analysis in Appendix B.3.

C.5 Adaptive Block-Rank Topology Analysis

Block-adaptive allocation operates on top of a prescribed layer-level rank topology. For each adapted module ℓ, the layer topology first determines a module-level rank r_ℓ. Block adaptation then redistributes this budget across the local output- and input-side blocks while preserving

$$\sum_{i=1}^{n} r_{B,\ell,i} = \sum_{j=1}^{n} r_{A,\ell,j} = n r_\ell. \tag{169}$$

Accordingly, $\gamma = 0$ recovers the prescribed layer topology, where all local blocks within the same module have rank r_ℓ, whereas increasing γ controls only the degree of within-module rank differentiation.

The analysis proceeds in four stages, followed by a joint interpretation. Appendix C.5.1 first studies performance sensitivity to the adaptive degree γ and examines how interaction-matrix initialization interacts with increasingly heterogeneous and rectangular local cores. Appendix C.5.2 then quantifies the corresponding structural rank-loss effect through the average pairwise rank ceiling. Appendix C.5.3 visualizes how a fixed layer-level budget is redistributed across local input- and

Table 10: Complete adaptive-degree results on (a) GLUE and (b) LLaMA-2-7B.

		Method	MNLI (%)	SST-2 (%)	MRPC (%)	CoLA (%)	QNLI (%)	QQP (%)	RTE (%)	STS-B (%)	Avg. (%)	Avg. w/o RTE (%)
(a) GLUE	Unif.	LoRA	87.25±0.06	94.65±0.37	90.28±0.49	61.63±2.25	92.78±0.01	89.28±0.06	75.81±1.25	90.62±0.18	85.29	86.64
		TriBLoRA	87.23±0.22	94.72±0.46	90.72±0.45	62.02±0.69	92.64±0.22	89.27±0.08	**76.93**±0.82	90.73±0.17	**85.53**	86.76
	Layer-adapt.	LoRA	87.26±0.07	94.57±0.13	90.72±0.77	60.41±1.17	**92.96**±0.31	89.32±0.11	76.50±1.30	90.68±0.12	85.30	86.56
		TriBLoRA	87.25±0.13	94.65±0.24	90.41±0.12	62.02±1.48	92.81±0.22	**89.34**±0.16	75.96±1.13	90.87±0.10	85.41	86.77
		TriBLoRA (K-F)	87.18±0.06	94.88±0.26	90.84±0.58	61.12±1.39	92.71±0.17	89.26±0.08	76.32±1.56	90.78±0.05	85.39	86.68
	Block-adapt.	**TriBLoRA**$_{(0.25)}$	87.33±0.13	94.76±0.33	90.88±0.46	61.73±0.95	92.90±0.12	89.27±0.05	74.69±2.57	90.83±0.21	85.30	86.82
		TriBLoRA$_{(0.50)}$	87.07±0.34	**95.22**±0.46	90.76±0.28	61.82±1.49	92.89±0.21	89.30±0.16	75.05±2.69	**90.93**±0.12	85.38	86.86
		TriBLoRA$_{(0.75)}$	87.13±0.05	94.95±0.41	90.26±0.25	61.40±1.16	92.68±0.18	89.31±0.07	75.56±1.06	90.87±0.19	85.27	86.66
		TriBLoRA$_{(1.00)}$	87.23±0.07	95.03±0.29	90.65±0.37	61.64±0.75	92.92±0.26	89.29±0.13	73.32±7.33	90.86±0.21	85.12	86.80
		TriBLoRA$_{(1.25)}$	87.18±0.05	94.88±0.07	90.37±0.46	**62.78**±1.47	92.85±0.05	89.30±0.04	75.99±1.29	90.79±0.11	85.52	86.88
		TriBLoRA$_{(1.50)}$	87.31±0.13	95.11±0.46	90.30±0.46	62.05±1.49	92.76±0.13	89.31±0.13	75.42±1.28	90.84±0.19	85.39	86.81
		TriBLoRA$_{(1.75)}$	87.20±0.23	95.07±0.20	91.04±0.38	62.15±0.92	92.82±0.09	89.32±0.08	75.63±1.12	**90.94**±0.16	85.52	**86.93**
		TriBLoRA$_{(2.00)}$	87.19±0.09	94.92±0.37	90.78±0.52	61.43±1.81	92.74±0.30	89.28±0.16	75.45±1.18	90.80±0.13	85.32	86.73
		TriBLoRA (K-F)$_{(0.50)}$	**87.38**±0.09	94.80±0.13	**91.28**±0.85	62.11±1.27	92.76±0.21	89.21±0.16	75.78±1.22	90.85±0.17	85.52	**86.91**
		TriBLoRA (K-F)$_{(1.25)}$	87.16±0.06	94.99±0.33	90.93±0.42	61.59±0.87	92.81±0.18	89.30±0.13	76.39±1.58	90.80±0.16	85.50	86.80

		Method	Best Eval Loss ↓	MMLU (%)	BBH (%)	DROP (%)	WinoGrande (%)	ARC-C (%)	HumanEval (%)	Avg. (6 tasks) (%)
(b) LLaMA-2-7B	Unif.	LoRA	0.8311	43.54±0.50	32.94±0.45	32.44±0.17	70.24±0.24	42.53±1.13	**15.37**±0.38	39.51
		TriBLoRA	0.8333	44.18±0.65	33.55±0.34	32.53±0.65	70.54±0.58	44.57±1.88	15.00±0.73	40.06
	Layer-adapt.	LoRA	0.8298	44.22±0.24	33.28±0.51	32.21±0.44	70.83±0.55	45.24±0.50	14.83±0.50	40.10
		TriBLoRA	**0.8297**	44.04±0.39	33.11±0.27	32.13±0.27	70.69±0.59	44.06±1.29	15.10±0.57	39.85
		TriBLoRA (K-F)	0.8301	44.02±0.32	**33.65**±0.43	32.23±0.49	70.37±0.63	45.36±1.01	14.95±0.39	40.10
	Block-adapt.	**TriBLoRA**$_{(0.50)}$	0.8299	44.19±0.43	33.13±0.25	32.26±0.29	70.64±0.69	44.16±1.73	14.85±0.46	39.87
		TriBLoRA$_{(0.75)}$	0.8300	44.31±0.28	33.29±0.50	32.08±0.29	70.77±0.91	44.86±1.06	15.10±0.72	40.07
		TriBLoRA$_{(1.00)}$	0.8299	44.27±0.31	33.30±0.31	32.12±0.23	70.61±0.50	44.69±1.56	15.20±0.55	40.03
		TriBLoRA$_{(1.25)}$	0.8301	44.35±0.28	33.41±0.33	32.12±0.39	70.73±0.52	45.12±1.97	15.07±0.56	40.13
		TriBLoRA$_{(1.50)}$	0.8302	**44.41**±0.23	33.50±0.19	32.10±0.44	70.89±0.63	44.98±1.94	14.95±0.64	40.14
		TriBLoRA$_{(1.75)}$	0.8302	44.38±0.36	33.42±0.29	32.13±0.30	**71.02**±0.68	44.62±1.78	15.15±0.64	40.12
		TriBLoRA$_{(2.00)}$	0.8303	44.26±0.21	33.42±0.29	32.08±0.45	70.89±0.69	44.56±2.17	15.17±0.71	40.06
		TriBLoRA (K-F)$_{(1.25)}$	0.8304	44.23±0.27	33.41±0.29	**32.68**±0.46	70.23±0.36	**47.05**±0.81	14.80±0.35	**40.40**

output-side blocks. Appendix C.5.4 subsequently controls for the rank multiset and tests whether the particular within-module placement of those ranks matters. Finally, Appendix C.5.5 combines these observations into a trade-off among importance-aware specialization, structural rank utilization, and task-dependent rank placement.

C.5.1 Adaptive-Degree Sensitivity

We evaluate the adaptive-degree rule described in Section 5.3 over a fixed grid of γ values while preserving the same module-wise side-rank sums. Layer-Adaptive LoRA, Layer-Adaptive TriBLoRA, and Block-Adaptive TriBLoRA use the same importance information and total budget wherever applicable. This comparison separates the effect of the importance estimator from the effect of allocation granularity.

Table 10 reports the complete available adaptive-degree sweeps on GLUE and LLaMA-2-7B. The uniform and layer-adaptive rows are included as references; a single value of γ is applied across all tasks within each fixed-degree row. The K-F rows change the interaction-matrix initialization and its learning-rate ratio together, so they should be compared with the corresponding non-K-F rows at the same γ.

Effect of adaptive degree. The results show that stronger within-module rank differentiation is not monotonically beneficial. On GLUE, different tasks favor different adaptive degrees, and the aggregate performance fluctuates rather than increasing consistently with γ. Under the default identity initialization, $\gamma = 1.25$ and $\gamma = 1.75$ both reach 85.52% overall. Excluding RTE, $\gamma = 1.75$ reaches 86.93%, compared with 86.88% at $\gamma = 1.25$; the average then decreases to 85.32% (86.73% without RTE) at $\gamma = 2.00$. The preferred degree is task dependent. For example, SST-2 obtains its highest reported block-adaptive score at $\gamma = 0.50$, CoLA at $\gamma = 1.25$, and MRPC, QQP, and STS-B at $\gamma = 1.75$.

A similar pattern appears on LLaMA-2-7B. Under identity initialization, the six-task average increases from 39.87% at $\gamma = 0.50$ to 40.13–40.14% around $\gamma = 1.25$–1.50, and then decreases to 40.06% at $\gamma = 2.00$. Thus, increasing γ does not simply increase useful capacity. Instead, it changes how a fixed layer-level budget is distributed among local blocks. As shown in the structural analysis below, stronger differentiation simultaneously increases importance-aware specialization and reduces the average pairwise rank ceiling.

Core initialization under block adaptation. Block adaptation also changes the geometry of the interaction matrices and therefore makes their initialization particularly relevant. As described in Appendix B.3, the default TriBLoRA configuration uses rectangular-identity initialization,

$$M_{ij}^{(0)} = I_{r_{B,i} \times r_{A,j}}, \tag{170}$$

together with an interaction-matrix learning-rate ratio $\lambda_M = 0.3$. When the local ranks are equal, this reduces to the standard square identity and provides a natural coordinate-aligned mapping between the two local latent spaces.

This property becomes less straightforward under block adaptation. Increasing γ deliberately creates heterogeneous local ranks, and many interaction matrices therefore satisfy

$$r_{B,i} \neq r_{A,j}. \tag{171}$$

For such a rectangular core, the identity initialization only establishes one-to-one correspondence over the shared dimensionality

$$k_{ij} = \min(r_{B,i}, r_{A,j}), \tag{172}$$

while the remaining possible cross-coordinate interactions are initially absent. Therefore, although identity initialization is well aligned with the uniform square-core setting, it may be less naturally matched to the heterogeneous latent geometries deliberately introduced by block-adaptive rank allocation.

Kaiming–Frobenius (K-F) initialization provides a complementary alternative. As defined in Appendix B.3, we initialize

$$M_{ij}^{(0)} = \sqrt{k_{ij}}\, \frac{\widetilde{M}_{ij}}{\|\widetilde{M}_{ij}\|_F}, \qquad k_{ij} = \min(r_{B,i}, r_{A,j}), \tag{173}$$

where $\widetilde{M}_{ij}$ is Kaiming initialized. This normalization matches the Frobenius-norm scale of the corresponding rectangular identity while initializing a dense transformation between the two local latent spaces. The K-F setting additionally uses $\lambda_M = 1.0$ rather than the default $\lambda_M = 0.3$.

The distinction is illustrated schematically by

$$\underbrace{I_{r_{B,i} \times r_{A,j}}}_{\text{coordinate-aligned}} \qquad \text{vs.} \qquad \underbrace{M_{ij}^{(0)}\ \text{dense}}_{\text{cross-coordinate coupling}} \quad . \tag{174}$$

Under increasingly heterogeneous block ranks, the former preserves a specific aligned pathway, whereas the latter couples all available local coordinates from the beginning of optimization. This suggests that core initialization and block topology should not be viewed independently: the topology determines the dimensions of the local latent spaces, while the initialization determines how these spaces are initially connected.

Empirical interaction between topology and initialization. The K-F results provide partial support for this interpretation, while also showing that the benefit is not universal. On GLUE at $\gamma = 0.50$, the K-F configuration reaches an overall average of 85.52%, compared with 85.38% under identity initialization. Excluding RTE, the corresponding averages are 86.91% and 86.86%. The task-level effect can be more pronounced; for example, MRPC improves from 90.76% to 91.28%.

At $\gamma = 1.25$, however, K-F and identity initialization are essentially comparable on GLUE: the overall averages are 85.50% and 85.52%, respectively. This indicates that dense initialization is not uniformly better for every block topology. Rather, its usefulness appears to depend on the particular degree and pattern of local rank heterogeneity.

The difference is more visible on LLaMA-2-7B. At $\gamma = 1.25$, K-F increases the six-task average from 40.13% to 40.40%. The largest individual gain occurs on ARC-C, which increases from 45.12% to 47.05%, although the effects across the remaining tasks are mixed. Consequently, the results are consistent with the hypothesis that dense core initialization can be better matched to some heterogeneous block-adaptive topologies, but they do not support a universal advantage over identity initialization.

An additional caveat is that the K-F configuration changes both the initialization of M_{ij} and its learning-rate ratio. The comparison should therefore be interpreted as an initialization–optimization setting rather than a strictly isolated initialization ablation. Nevertheless, its behavior is consistent with the post-hoc analysis in Appendix C.4.2: identity-initialized checkpoints remain largely insensitive to removal or permutation of the learned dense cores, whereas K-F checkpoints exhibit substantially stronger dependence on off-diagonal mixing and pair-specific core assignment. Together, these results suggest that K-F not only changes the initial coupling between heterogeneous local latent spaces, but can encourage stronger dependence on the dense interaction-core parameterization during training.

Connection to structural rank loss. The initialization effect does not remove the structural consequence of increasing γ. Regardless of how M_{ij} is initialized, the pairwise update satisfies

$$\operatorname{rank}(B_i M_{ij} A_j) \le \min(r_{B,i}, r_{A,j}). \tag{175}$$

Thus, stronger rank differentiation simultaneously changes two properties: the geometry that the interaction matrix must connect and the structural rank ceiling of each pair. The former makes the choice of core initialization increasingly relevant, while the latter introduces the structural rank-loss trade-off analyzed in Appendix C.5.2.

Overall, the adaptive-degree results suggest that block adaptation should be understood jointly through topology and core optimization. Increasing γ can provide more importance-aware specialization, but it also creates more heterogeneous local latent spaces and greater pairwise rank loss. Dense K-F initialization can better exploit some of these heterogeneous configurations, whereas the rectangular identity retains a more restricted coordinate-aligned initial pathway. The preferred balance between these effects is consequently task- and topology-dependent.

C.5.2 Structural Rank Loss under Increasing Heterogeneity

To understand the non-monotonic behavior above, we examine the structural effect of increasing rank heterogeneity. For interaction pair (i, j) in module ℓ,

$$\operatorname{rank}(B_{\ell,i} M_{\ell,ij} A_{\ell,j}) \le \min(r_{B,\ell,i}, r_{A,\ell,j}). \tag{176}$$

We therefore define the *mean pairwise rank ceiling* of a topology as

$$r_{\text{eff}} = \frac{1}{Ln^2} \sum_{\ell=1}^{L} \sum_{i=1}^{n} \sum_{j=1}^{n} \min(r_{B,\ell,i}, r_{A,\ell,j}), \tag{177}$$

where L denotes the number of adapted projection modules.

This quantity measures nominal structural capacity rather than the numerical rank of a trained update. The learned outer factors and interaction matrices may realize a smaller numerical rank.

Block adaptation preserves the total core parameter budget. From Eq. (169),

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{n} r_{B,\ell,i} r_{A,\ell,j} &= \left(\sum_{i} r_{B,\ell,i} \right) \left(\sum_{j} r_{A,\ell,j} \right) \\ &= n^2 r_\ell^2. \end{aligned} \tag{178}$$

However, preservation of parameter count does not imply preservation of pairwise rank capacity. Since

$$\min(a, b) \le \frac{a+b}{2}, \tag{179}$$

we obtain

$$\begin{aligned} \frac{1}{n^2} \sum_{i,j} \min(r_{B,\ell,i}, r_{A,\ell,j}) &\le \frac{1}{2n^2} \sum_{i,j} (r_{B,\ell,i} + r_{A,\ell,j}) \\ &= r_\ell. \end{aligned} \tag{180}$$

Thus, increasingly heterogeneous input- and output-side ranks can reduce the average pairwise rank ceiling despite an unchanged module-level budget.

Table 11: Structural effect of adaptive degree relative to the prescribed layer topology. At $\gamma = 0$, block ranks recover the corresponding layer-level allocation. Increasing γ redistributes the same side-rank budget within each module. The mean pairwise rank ceiling is defined in Eq. (177). Changed modules are measured relative to the corresponding $\gamma = 0$ topology. Both panels also report the local rank range, pairwise capacity range, mean side ranks, and the numbers of changed rank cells and modules.

(a) GLUE

γ	r_{eff}	Δ vs. layer	Local min	Local max	Pair min	Pair max	Mean A	Mean B	Changed cells	Changed modules
0.00	8.000	+0.000	5	16	5	16	8.00	8.00	0/1536	0/192
0.25	7.771	−0.229	4	16	4	16	8.00	8.00	464/1536	90/192
0.50	7.509	−0.491	4	16	4	16	8.00	8.00	909/1536	161/192
0.75	7.313	−0.688	4	16	4	16	8.00	8.00	1084/1536	178/192
1.00	7.156	−0.844	4	16	4	16	8.00	8.00	1171/1536	183/192
1.25	7.024	−0.976	4	16	4	16	8.00	8.00	1216/1536	183/192
1.50	6.948	−1.052	4	16	4	16	8.00	8.00	1248/1536	183/192
1.75	6.860	−1.140	4	16	4	16	8.00	8.00	1271/1536	183/192
2.00	6.814	−1.186	4	16	4	16	8.00	8.00	1287/1536	183/192

(b) LLaMA-2-7B

γ	r_{eff}	Δ vs. layer	Local min	Local max	Pair min	Pair max	Mean A	Mean B	Changed cells	Changed modules
0.00	8.000	+0.000	4	11	4	11	8.00	8.00	0/512	0/64
0.25	7.801	−0.199	4	12	4	12	8.00	8.00	140/512	31/64
0.50	7.567	−0.433	4	13	4	13	8.00	8.00	277/512	58/64
0.75	7.368	−0.632	4	14	4	14	8.00	8.00	332/512	63/64
1.00	7.169	−0.831	4	15	4	15	8.00	8.00	368/512	63/64
1.25	7.010	−0.990	4	16	4	16	8.00	8.00	396/512	63/64
1.50	6.905	−1.095	4	16	4	16	8.00	8.00	408/512	63/64
1.75	6.801	−1.199	4	16	4	16	8.00	8.00	412/512	63/64
2.00	6.703	−1.297	4	16	4	16	8.00	8.00	425/512	63/64

Table 11 reveals a monotonic structural trend despite the non-monotonic performance curves. On GLUE, the mean mean pairwise rank ceiling decreases from 8.00 at $\gamma = 0$ to 7.02 at $\gamma = 1.25$ and 6.81 at $\gamma = 2.0$. The number of modules whose within-module topology differs from the layer allocation increases rapidly and saturates at 183 out of 192 modules from $\gamma = 1.0$ onward.

LLaMA-2-7B exhibits the same structural behavior. Its mean pairwise rank ceiling decreases from 8.00 to 7.01 at $\gamma = 1.25$ and to 6.70 at $\gamma = 2.0$. Meanwhile, the mean A- and B-side ranks remain exactly 8.00, confirming that this reduction does not arise from a smaller rank budget. Instead, the local rank range expands from 4–11 under the layer topology to 4–16 once stronger block differentiation is applied, and the number of changed local rank cells increases to $425/512$ at $\gamma = 2.0$.

These results expose two competing effects of increasing γ:

$$\underbrace{\text{importance-aware specialization}}_{\gamma\uparrow} \quad \text{vs.} \quad \underbrace{\text{pairwise structural rank loss}}_{\gamma\uparrow}. \tag{181}$$

At moderate adaptive degrees, directing more rank toward important local blocks can improve the use of the available budget. At larger degrees, however, the increasingly unequal side ranks continue to reduce the pairwise rank ceilings through $\min(r_{B,\ell,i}, r_{A,\ell,j})$. The performance trends in Table 10 are consistent with this trade-off: moderate differentiation can be beneficial, whereas stronger differentiation eventually saturates or declines despite using the same number of parameters.

We emphasize that this correspondence does not establish structural rank loss as the sole cause of the performance variation. Rather, it identifies a concrete structural cost that grows monotonically with γ and therefore provides a plausible explanation for why increasingly aggressive rank differentiation need not yield monotonic gains.

C.5.3 Visualization of Within-Module Rank Redistribution

The aggregate statistics in Table 11 do not show where the redistributed ranks are placed. We therefore visualize the prescribed layer topology together with its block-adaptive counterpart.

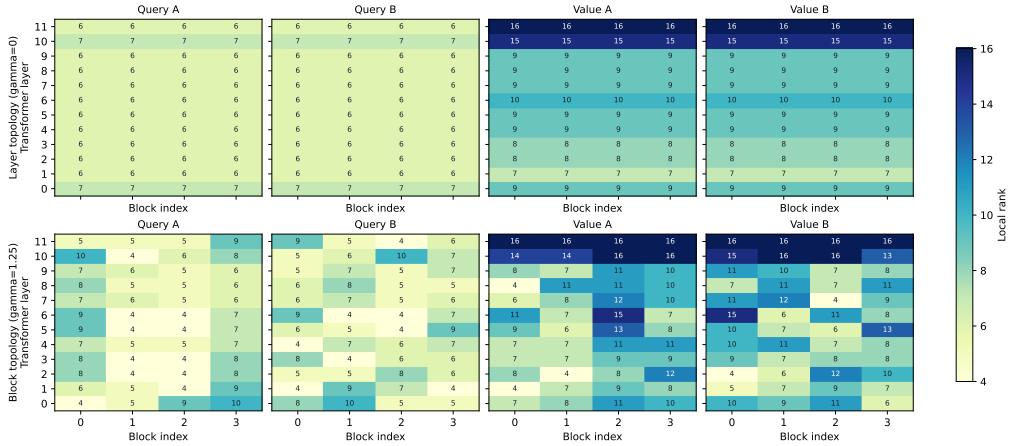

Figure 3: Representative layer-level and block-adaptive rank topologies for CoLA. The top row shows the prescribed layer topology ($\gamma = 0$), where all local blocks within the same module share the module-level rank. The bottom row shows the corresponding block-adaptive topology at $\gamma = 1.25$. Query and value projections are shown separately for the input-side (A) and output-side (B) ranks. Block adaptation changes the within-module distribution of rank while preserving the corresponding side-rank sums.

For each query or value projection, the layer topology assigns the same module-level rank r_ℓ to all four local blocks. Block adaptation retains the same side-rank sums but redistributes them according to the estimated local importance.

The visualization highlights that block adaptation is not equivalent to simply changing a scalar layer rank. After redistribution, different blocks within the same query or value projection can receive substantially different ranks while preserving the same total budget. Some modules remain close to their original allocation, whereas others exhibit strong intra-module differentiation. The resulting mismatch between input- and output-side local ranks gives rise to the pairwise capacity reduction quantified in Eq. (177).

C.5.4 Sensitivity to Within-Module Rank Placement

The adaptive-degree analysis determines how strongly rank is redistributed. We next ask a separate question: once the local rank multiset is fixed, does the particular placement of those ranks matter?

For each selected task, we fix the underlying layer topology, the selected adaptive degree, the A- and B-side rank multisets within every module, the total parameter budget, the interaction-matrix initialization, and the learning-rate protocol. We then independently permute the local A and B ranks within every query and value module. Three random topology permutations are generated using topology seeds 101, 102, and 103. Consequently, the guided and permuted configurations use exactly the same rank values and parameter budgets; only the association between rank values and block positions changes.

CoLA and MRPC use training seeds 41–45. SST-2 and QNLI use seeds 41–43, following the large-task protocol. Any SST-2 or QNLI runs using seeds 44–45 are excluded from this comparison.

Table 12 shows that sensitivity to rank placement is task dependent. CoLA exhibits a consistent advantage for the guided topology over all three random permutations, with differences of approximately 0.7–1.1 points. SST-2 shows the same directional pattern in the currently available runs. These cases indicate that, for some tasks, assigning larger ranks to the importance-selected local blocks contributes beyond simply selecting the rank multiset.

MRPC, in contrast, is almost invariant to the tested permutations: the guided and randomized topologies all obtain approximately 91.0%. QNLI also shows no consistent advantage for the guided placement, with two random permutations slightly exceeding the guided result and one falling below it.

Table 12: Sensitivity to within-module rank placement at the selected adaptive degree. Guided denotes the importance-based block topology. Perm-101/102/103 independently permute the local A- and B-side ranks within every query/value module while preserving the same rank multisets, parameter budgets, initialization, and optimization protocol. Values are task means $\pm$ sample standard deviations.

Task	γ	Guided	Perm-101	Perm-102	Perm-103
CoLA	1.25	$62.78_{\pm 1.47}$	$61.66_{\pm 0.97}$	$61.99_{\pm 1.56}$	$62.09_{\pm 1.47}$
MRPC	1.75	$91.04_{\pm 0.38}$	$91.06_{\pm 0.56}$	$91.02_{\pm 0.80}$	$90.98_{\pm 0.73}$
SST-2	0.50	$95.22_{\pm 0.46}$	$94.95_{\pm 0.60}$	$94.69_{\pm 0.40}$	$94.80_{\pm 0.46}$
QNLI	1.00	$92.92_{\pm 0.26}$	$92.99_{\pm 0.16}$	$93.01_{\pm 0.09}$	$92.82_{\pm 0.09}$

For these tasks, the degree of within-module rank differentiation appears to be more consequential than the precise locations receiving the individual ranks.

The permutation experiment therefore separates two aspects of block-adaptive allocation:

$$\underbrace{\text{rank differentiation}}_{\text{which ranks are allocated}} \quad \text{and} \quad \underbrace{\text{rank placement}}_{\text{where they are assigned}}. \tag{182}$$

The former determines the extent of local specialization and the associated structural rank loss, whereas the latter determines whether the importance-guided spatial assignment itself provides additional benefit.

C.5.5 Overall Interpretation

Taken together, the adaptive-topology analyses reveal a trade-off between importance-aware specialization and structural rank utilization. Starting from a fixed layer topology, increasing γ redistributes the same module-level rank budget more aggressively across local blocks. This allows the allocation to emphasize locally important regions, but simultaneously increases mismatches between output- and input-side local ranks and therefore reduces the average pairwise rank ceiling.

This structural trade-off is consistent with the non-monotonic adaptive-degree results. Moderate differentiation can improve the allocation of capacity, whereas larger γ continues to reduce structural effective rank without producing corresponding monotonic performance gains. The preferred degree is therefore task dependent rather than universal.

The rank-placement experiment exposes a second source of task dependence. For some tasks, the particular blocks receiving larger ranks contribute to performance, whereas other tasks are comparatively insensitive to permutations that preserve exactly the same rank multiset and parameter budget. Block-adaptive allocation should therefore be understood as controlling two distinct properties: *how strongly* the layer-level budget is differentiated within a module and *where* the resulting local ranks are placed.

D Implementation and Numerical Diagnostics

This appendix separates the TriBLoRA parameterization from its computational realization. Appendix D.1 describes algebraically equivalent computation paths and their computation–memory trade-offs. Appendix D.2 then examines finite-precision differences using the fixed-identity control.

D.1 Computational Organization

The TriBLoRA parameterization specifies the interaction structure of the adapter but does not prescribe a unique computational realization. In particular, for an output-side block i, the adapter contribution can be written as

$$Y_i = \sum_{j=1}^{n} B_i M_{ij} A_j X_j, \tag{183}$$

where X_j denotes the input associated with input-side block j. A direct pair-wise implementation may evaluate the operations associated with each (i, j) independently. Although this follows the parameterization literally, it can lead to many small matrix multiplications and kernel launches.

This effect becomes more pronounced as the block number n increases. Individual local dimensions become smaller while the number of interaction pairs grows as n^2. Consequently, a straightforward implementation can fragment otherwise large matrix multiplications into many smaller operations, which may reduce GPU utilization and increase launch and scheduling overhead. Backward propagation must additionally retain and process the corresponding block-wise intermediate activations. These effects concern the computational realization of TriBLoRA rather than a restriction of its parameterization.

Reusing input-side projections. A first optimization follows directly from the sharing structure of TriBLoRA. Since A_j is shared by all interaction pairs associated with input block j, its projection

$$Z_j = A_j X_j \tag{184}$$

needs to be evaluated only once within a forward pass. Equation (183) then becomes

$$Y_i = \sum_{j=1}^{n} B_i M_{ij} Z_j. \tag{185}$$

Thus, an implementation need not independently recompute $A_j X_j$ for every pair (i, j). The projected activations $\{Z_j\}$ can instead be shared across all output-side interactions connected to the same input block.

Associating the outer factor after interaction aggregation. Using distributivity, the same computation can be reorganized as

$$U_i = \sum_{j=1}^{n} M_{ij} Z_j, \qquad Y_i = B_i U_i. \tag{186}$$

This form applies the shared output-side factor B_i only once for output block i, rather than once for every interaction pair. The remaining pair-specific computation is concentrated in the small latent-space multiplications $M_{ij} Z_j$.

This organization is particularly natural for TriBLoRA because B_i is shared over all input blocks connected to output region i, while M_{ij} is the only pair-specific transformation. It reduces redundant applications of the large outer factors without changing the underlying adapter parameterization.

Precomposing the output factor and interaction matrix. An alternative association is to define

$$C_{ij} = B_i M_{ij}, \tag{187}$$

which gives

$$Y_i = \sum_{j=1}^{n} C_{ij} Z_j. \tag{188}$$

Within a forward pass, each C_{ij} can be reused over all examples and sequence positions in the corresponding activation tensor. Operations with the same matrix shapes may also be organized using batched or grouped matrix multiplication.

During training, however, C_{ij} remains part of the differentiable computation graph because it depends on both B_i and M_{ij}. It therefore cannot be treated as a persistent precomputed weight across optimizer steps: after either factor is updated, the previous C_{ij} becomes stale. This form trades latent-space aggregation for a different set of matrix shapes, and whether it is faster than Eq. (186) depends on the batch size, sequence length, block number, local rank, and hardware.

Global block-matrix realization. TriBLoRA can also be expressed in the compact global form

$$\Delta W = \mathcal{B}\mathcal{M}\mathcal{A}, \tag{189}$$

where

$$\mathcal{B} = \mathrm{blkdiag}(B_1, \ldots, B_n), \qquad \mathcal{A} = \mathrm{blkdiag}(A_1, \ldots, A_n), \tag{190}$$

and

$$\mathcal{M} = \begin{bmatrix} M_{11} & \cdots & M_{1n} \\ \vdots & \ddots & \vdots \\ M_{n1} & \cdots & M_{nn} \end{bmatrix}. \tag{191}$$

For uniform local rank r, $\mathcal{M} \in \mathbb{R}^{nr \times nr}$, and the global update has

$$\mathrm{rank}(\Delta W) \le nr. \tag{192}$$

For a block-partitioned input $X = [X_1^\top, \ldots, X_n^\top]^\top$, the forward computation can therefore be organized conceptually as

$$Z = \mathcal{A}X, \qquad U = \mathcal{M}Z, \qquad Y = \mathcal{B}U. \tag{193}$$

This representation exposes larger matrix operations and removes the need for Python-level iteration over all interaction pairs. Importantly, it does not require explicitly materializing the full $d_{\mathrm{out}} \times d_{\mathrm{in}}$ update matrix.

A naive dense realization of $\mathcal{A}$ and $\mathcal{B}$ would, however, store and process the zero entries introduced by their block-diagonal structure. A practical implementation should therefore preserve the block structure using grouped or batched GEMM, fused kernels, or equivalent structured operators rather than explicitly constructing dense block-diagonal matrices.

Computation and memory trade-offs. The alternative forms above are algebraically equivalent but expose different computational trade-offs. Equation (186) minimizes repeated applications of the large output-side factors by performing interaction aggregation in the latent space. Equation (188) instead precomposes B_i and M_{ij} within each forward pass, which can be advantageous when the resulting matrices are reused over many tokens. The global form in Eq. (193) exposes the entire nr-dimensional latent interaction and may allow more efficient batched or fused execution.

These reorganizations do not remove the intrinsic n^2 pair-specific core structure. Under uniform rank r, the interaction matrices still contain $n^2 r^2$ trainable parameters, and applying the dense latent interaction still requires processing all corresponding pairwise couplings. Moreover, larger fused computations may increase intermediate activation memory. Consequently, reducing kernel count alone does not determine the preferred implementation. Runtime, GPU utilization, activation memory, and backward cost should be evaluated jointly for the target workload.

D.2 Numerical Diagnostics of the Fixed-Identity Control

Algebraic equivalence. With uniform local rank r and all cores fixed to $M_{ij} = I_r$, define $B = [B_1^\top, \ldots, B_n^\top]^\top$ and $A = [A_1, \ldots, A_n]$. Then $\Delta W_{ij} = B_i A_j$ and assembling all block pairs gives $\Delta W = BA$. Thus, fixed-identity TriBLoRA and rank-r LoRA represent exactly the same update for matched outer factors.

One-step FP16 audit. We copy the LoRA outer factors into the corresponding blocks and use identical classifier parameters, input examples, optimizer settings, and CPU/CUDA random-number-generator states. The production block-wise path computes local input projections and accumulates them before applying the output factors. We compare this with a diagnostic concatenated path using the same arithmetic organization as LoRA. Table 13 reports the initial forward differences and the A/B gradient differences associated with the first nonzero update.

Conclusion. The concatenated path also reproduces LoRA's one-step parameter updates exactly. These checks identify arithmetic-path sensitivity in the block-wise FP16 computation: small gradient perturbations near zero can be amplified by AdamW despite algebraic equivalence. The contribution of this effect to the full benchmark-level gap remains unquantified. Accordingly, the fixed-identity result serves as an implementation control within the same rank-r update family.

Table 13: One-step GPU FP16 audit of fixed-identity TriBLoRA against standard LoRA. Entries are maximum absolute differences; all cores remain fixed to I_r.

Quantity	Block-wise path	Concatenated path
Initial logits	0	0
Initial loss	0	0
First-update A/B gradients	3.81×10^{-6}	0

E Limitations and Future Work

Several limitations remain and motivate further investigation.

First, our study of interaction-matrix initialization is not yet exhaustive. The current experiments compare the default rectangular-identity initialization with Kaiming–Frobenius (K-F) initialization only for a subset of adaptive degrees and settings. The results suggest that K-F can be more effective for some heterogeneous block topologies, particularly when the local interaction matrices become rectangular, but the benefit is not uniform across tasks or adaptive degrees. A more systematic study that separates initialization from the interaction-matrix learning-rate ratio and covers a broader range of topologies would be needed to characterize this effect more conclusively.

Second, performance on GLUE remains sensitive to task-specific optimization and random variation. Although TriBLoRA improves the aggregate results, the gain is not uniform across all tasks, and the current results on MNLI and QQP show only limited improvement over standard LoRA. This variability makes joint hyperparameter selection across heterogeneous GLUE tasks challenging. A broader evaluation on larger and more stable benchmarks would help determine more precisely when the additional block-interaction structure is most beneficial.

Third, the interaction between block granularity and core optimization remains underexplored. Increasing the block number n changes not only the number and size of local interactions, but also the number of pair-specific cores and the gradient statistics seen by the shared outer factors and interaction matrices. In the current experiments, we use the same interaction-matrix learning-rate rule across different values of n rather than retuning it jointly with block granularity. The weaker gains observed for larger partitions, such as $n = 8$, may therefore reflect both structural saturation and increased optimization difficulty. Future work could study the joint scaling of n, λ_M, and other optimization parameters, as well as more efficient implementations for fine-grained block structures.

Finally, block-adaptive rank allocation is not universally beneficial for every topology. Because the pairwise interaction rank is bounded by $\min(r_{B,i}, r_{A,j})$, stronger within-module rank differentiation can introduce structural rank loss even when the total parameter budget is preserved. Our results show that appropriately chosen adaptive topologies can improve fine-tuning performance at the same parameter budget, but the preferred adaptive degree and rank placement are task dependent. The current topologies are selected using empirical calibration rather than a theoretically optimal allocation rule. Developing an allocation criterion that explicitly balances importance-aware specialization against structural rank loss is therefore an important direction for future work.