# INSIDE THE RESIDUAL STREAM: NEURON-LEVEL MIXING WITH NONLINEAR LOW-RANK BRANCHES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Redesigning residual connections can improve the way information is propagated across deep Transformer layers. Methods such as Hyper-Connection and Attention Residuals enrich connectivity by introducing interactions among different hidden states. However, these approaches primarily operate at the level of whole hidden states, leaving finer-grained interactions within individual states largely unexplored. Frac-Connection explores finer-grained connectivity by partitioning hidden states into multiple streams and applying structured interactions, but its interaction pattern is constrained by a fixed coordinate-wise decomposition. In this work, we propose ResLorB (**Res**idual Connections with Nonlinear **Lo**w-**r**ank **B**ranches), which introduces a lightweight nonlinear low-rank transformation at the block input to enable learnable neuron-level interactions. With only minimal additional parameters and computational overhead, ResLorB consistently improves model performance without requiring specialized optimization or infrastructure support. It is naturally compatible with existing multi-state architectures, including mHC, and the low-rank branch can further serve as an additional interactable state to enable richer interaction patterns. Extensive experiments on DeepSeek-V3-style models scaling up to 3B/9B parameters demonstrate consistent improvements over strong baselines, including a 0.011 reduction in training LM loss and 1.4% improvement in downstream average performance at the 9B scale, while introducing only 1% additional parameters and negligible latency overhead. On 3B-scale models, ResLorB improves downstream performance by approximately 1% on average. Combining ResLorB with existing multi-state architectures further yields non-saturating gains and consistent improvements.

## 1 INTRODUCTION

Large language models (LLMs) have achieved remarkable success through the scaling of Transformer architectures (Vaswani et al., 2017; Brown et al., 2020; Touvron et al., 2023). Despite their simplicity, residual connections (He et al., 2016) play a fundamental role in enabling effective and stable information propagation. As model scaling continues to improve performance (Kaplan et al., 2020), the associated computational and memory costs have grown substantially, motivating the architectural designs that improve model performance with minimal additional overhead.

A line of work explores this trade-off by introducing architectural modifications that diversify connection patterns across layers or representation states. Methods such as Hyper-Connection (HC) (Zhu et al., 2025a) and Attention Residuals (Kimi Team, 2026) introduce multi-state recurrence or cross-layer connections, enriching interactions and improving model performance. While these approaches improve connection diversity across representations, they primarily operate at the level of whole hidden states, i.e., treating each Transformer block input as an atomic unit and focusing on inter-layer or inter-state interactions. As a consequence, the finer-grained intra-state connection diversity remains unexplored. Moreover, since these approaches modify residual pathways at the same state level, multiple modifications may interact with each other and require careful integration, potentially limiting their compatibility (Kimi Team, 2026; Su, 2025).

Inspired by the multi-state interaction paradigm of HC, Frac-Connection (FC) (Zhu et al., 2025b) explores finer-grained connectivity through neuron-level (within-state) feature mixing. Specifically, FC partitions each hidden state into multiple lower-dimensional streams through a fixed reshape



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.

operation and applies structured interactions over them. However, the resulting interaction pattern is constrained by this coordinate-wise decomposition: the set of hidden dimensions that can interact with each other is fixed by the initial *reshape* operation. This rigidity limits the model's ability to adapt neuron-level connections to learned representations, motivating more flexible approaches to intra-state modeling.

In this work, we propose ResLorB (**Res**idual Connections with Nonlinear **Lo**w-**r**ank **B**ranches), which introduces a nonlinear low-rank branch at the input of each Transformer block to enable adaptive neuron-level interactions within the residual stream. Unlike FC, whose interaction pattern is constrained by a predetermined partition of hidden dimensions, ResLorB learns interaction patterns adaptively through a trainable low-rank transformation. This design consistently improves model performance while introducing only minimal additional parameters and computational overhead. Since ResLorB operates at the intra-state level, it is naturally compatible with existing inter-state approaches such as mHC (Xie et al., 2025). Moreover, when integrated with multi-state architectures, the low-rank branch naturally serves as an additional interactable state, leading to richer interaction patterns and additional performance gains. We further provide a theoretical analysis in an in-context learning setting to show the representation benefits of low-rank branches.

We conduct experiments on LLaMA-series (Touvron et al., 2023) and DeepSeek-V3 (Liu et al., 2024a) variants. Specifically, we evaluate ResLorB on models up to 3B/9B parameters trained on up to 200B/450B tokens. Across settings, ResLorB consistently improves training and validation losses over strong baselines. It also outperforms prior intra-state methods such as Frac-Connection on average under the same configuration. These gains are achieved with only about 1% additional parameters and negligible inference overhead. Furthermore, when combined with mHC, ResLorB yields consistent improvements in both training performance and downstream evaluation, demonstrating strong scalability and compatibility with existing multi-state architectures.

Our contributions are summarized as follows:

1. We propose ResLorB, a lightweight nonlinear low-rank branching mechanism that enables adaptive neuron-level interactions in Transformers. We further provide a theoretical analysis to better understand why the proposed design improves model performance.
2. We show that ResLorB is fully compatible with existing inter-state methods such as mHC and that it further improves their performance by acting as an additional interactable state in multi-state frameworks.
3. Extensive experiments on models up to 3B/9B parameters trained on 200B/450B tokens demonstrate consistent improvements in training loss and downstream performance with only around 1% additional parameters and minimal inference overhead.

## 2 Related Work

**Residual and Cross-Layer Connections.** Residual connections enable stable optimization in deep neural networks and form the backbone of modern Transformer architectures. Building on this mechanism, cross-layer connectivity has been explored to enhance information flow between different depths of a network. Representative approaches include DenseNet (Huang et al., 2017), ELMo (Peters et al., 2018), and DenseFormer (Pagliardini et al., 2024), which aggregate or reuse representations across layers via fixed connection patterns. More recent methods such as MUDDFormer (Xiao et al., 2025) and Attention Residuals (Kimi Team, 2026) introduce input-dependent weights to further improve cross-layer interaction. Similar ideas have also been applied to other Transformer components beyond block inputs such as Attention Value (Zhou et al., 2025) and MoE routers (Jin et al., 2025). Despite improving interaction diversity across layers, these methods typically treat each hidden state as an indivisible unit. In contrast, our method enables adaptive neuron-level interactions within each hidden state, providing a finer-grained form of connection diversity.

**Multi-State Methods** Standard Transformers can be viewed as single-state models, where a single residual stream is updated across layers. Multi-state methods such as HC (Zhu et al., 2025a), mHC (Xie et al., 2025), and SiameseNorm (Li et al., 2026) extend this design by introducing multiple parallel streams to enable richer interaction and selection among different hidden-state representations. While these methods diversify interactions across multiple hidden states, recent work has

also explored richer interactions within each hidden state. For example, LAUREL (Menghani et al., 2025) introduces a parallel low-rank branch alongside the residual stream, while Frac-Connection (Zhu et al., 2025b) partitions hidden states into fixed subspaces and applies interaction matrices along the partition dimension, resulting in a structured sparse interaction pattern. Unlike these approaches, ResLorB applies a lightweight low-rank transformation to the Transformer block input to learn adaptive neuron-level interactions.

**Low-Rank Adaptation** Low-rank parameterization is widely used in parameter-efficient fine-tuning of large models (Houlsby et al., 2019; Hu et al., 2022). LoRA introduces low-rank updates to linear layers to reduce the number of trainable parameters while maintaining performance (Hu et al., 2022), with representative variants including ResLoRA (Shi et al., 2024), DoRA (Liu et al., 2024b), AdaLoRA (Zhang et al., 2023), and nonlinear variants (Liu et al., 2025). These methods are primarily developed for fine-tuning rather than improving full-model pretraining. Recently, NOBLE (Smith, 2026) extends nonlinear low-rank adaptation to the pretraining regime by introducing low-rank adapters to all linear transformations in the model. However, it requires careful optimization design such as initialization and learning rate scheduling. Our method instead introduces a lightweight nonlinear low-rank transformation at the block input, providing a simpler design while achieving strong empirical gains.

# 3 PRELIMINARIES

## 3.1 RESIDUAL CONNECTIONS AND TRANSFORMERS

Let $\mathbf{x}_l \in \mathbb{R}^d$ denote the input hidden state entering the $l$-th layer. We denote by $\mathcal{F}_l : \mathbb{R}^d \to \mathbb{R}^d$ the entire Transformer block (e.g., MHA or FFN) of the $l$-th layer, including the normalization layers. The standard Transformer employs residual connections as

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}_l(\mathbf{x}_l).$$

We consider both the Pre-Norm and Sandwich-Norm (Gemma Team et al., 2025) variant, i.e.,

$$\begin{aligned} \text{Pre-Norm:} \quad & \mathcal{F}_l(\mathbf{x}) = \mathrm{Block}(\mathrm{Norm}(\mathbf{x})), \\ \text{Sandwich-Norm:} \quad & \mathcal{F}_l(\mathbf{x}) = \mathrm{Norm}(\mathrm{Block}(\mathrm{Norm}(\mathbf{x}))). \end{aligned}$$

## 3.2 HYPER-CONNECTIONS AND FRAC-CONNECTIONS

Hyper-Connection (HC) extends the standard residual stream into $m$ parallel streams. Specifically, let $\mathbf{S}_l \in \mathbb{R}^{m\times d}$ denote this multi-state tensor, where the first dimension corresponds to the $m$ independent streams. The initial state is formed by repeating the input embedding $\mathbf{x}_0$ across streams:

$$\mathbf{S}_0 = [\mathbf{x}_0; \mathbf{x}_0; \cdots; \mathbf{x}_0] \in \mathbb{R}^{m\times d}. \tag{1}$$

Under the HC paradigm, the multi-state transition is formulated using three connection matrices $\mathbf{H}_l^{pre}, \mathbf{H}_l^{post} \in \mathbb{R}^{1\times m}, \mathbf{H}_l^{res} \in \mathbb{R}^{m\times m}$ that operate along the stream dimension:

$$\mathbf{S}_{l+1} = \mathbf{H}_l^{res}\mathbf{S}_l + \left(\mathbf{H}_l^{post}\right)^\top \mathcal{F}_l(\mathbf{H}_l^{pre}\mathbf{S}_l). \tag{2}$$

These matrices can be either fixed across layers or generated dynamically from the current hidden states. To improve training stability, the Manifold-Constrained Hyper-Connection (mHC) further constrains $\mathbf{H}_l^{res}$ to be doubly stochastic using the Sinkhorn-Knopp algorithm (Xie et al., 2025).

To alleviate the memory overhead of maintaining separate streams in HC, Frac-Connections (FC) mimic this recurrence via a tensor reshaping strategy. Specifically, FC applies a reshape operation to transform the standard hidden state $\mathbf{x}_l \in \mathbb{R}^d$ into a fractionated multi-state tensor $\hat{\mathbf{S}}_l \in \mathbb{R}^{n\times(d/n)}$:

$$\hat{\mathbf{S}}_l = \text{reshape}(\mathbf{x}_l, [n, d/n]), \tag{3}$$

where $n$ is the number of fractions. The same recurrence matrices $\hat{\mathbf{H}}_l^{pre}, \hat{\mathbf{H}}_l^{res} \in \mathbb{R}^{n\times n}, \hat{\mathbf{H}}_l^{post} \in \mathbb{R}^{1\times n}$ are then applied along the fraction dimension, i.e.

$$\hat{\mathbf{S}}_{l+1} = \hat{\mathbf{H}}_l^{res}\hat{\mathbf{S}}_l + \mathrm{Diag}\left(\hat{\mathbf{H}}_l^{post}\right) \text{reshape}\left(\mathcal{F}_l\left(\text{reshape}\left(\hat{\mathbf{H}}_l^{pre}\hat{\mathbf{S}}_l, [1, d]\right)\right), [n, d/n]\right), \tag{4}$$

which preserves the multi-state recurrence structure of HC under a constrained memory budget. An illustration of the architecture of HC and FC is provided in Fig. 1.

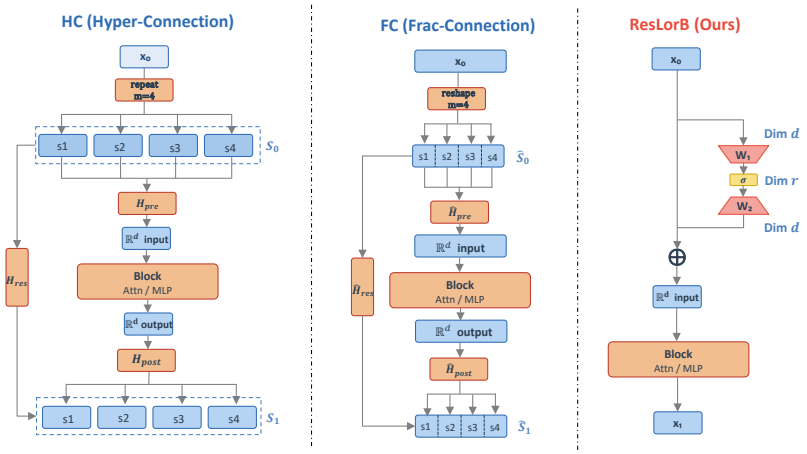



Figure 1: Comparison of three architectures with $m = 4$ streams/fractions. Left: HC splits the hidden state into $m$ parallel streams. Middle: FC splits via dimension fractioning. Right: ResLorB enables neuron-level interactions with low-rank branches.

# 4 Our Method

## 4.1 From Structured Sparse to Low-Rank Interactions

From an equivalent view of Eq. (4), the FC update can be rewritten directly in the original hidden-state space using a Kronecker-structured formulation:

$$\mathbf{x}_{l+1} = (\hat{\mathbf{H}}_l^{res} \otimes \mathbf{I}_{d/n})\mathbf{x}_l + \left(\text{Diag}\left(\hat{\mathbf{H}}_l^{post}\right) \otimes \mathbf{I}_{d/n}\right) \mathcal{F}_l\big((\hat{\mathbf{H}}_l^{pre} \otimes \mathbf{I}_{d/n})\mathbf{x}_l\big), \tag{5}$$

where $\otimes \mathbf{I}_{d/n}$ broadcasts the interaction to the full hidden space. This reveals that FC applies a *structured sparse* transformation: $\hat{\mathbf{H}}_l^{pre} \otimes \mathbf{I}_{d/n}$ is a block-diagonal matrix that fixes which hidden dimensions interact with each other based on the initial partition.

This fixed design is effective for HC/mHC, where the parallel streams are constructed by *repeating the same embedding*, so the stream dimension is naturally aligned across all hidden coordinates. For standard Transformer hidden states, however, no such structural alignment exists, and the fixed interaction pattern cannot adapt to the learned representations.

To obtain more flexible interactions, we replace this fixed structured matrix with a learnable low-rank transformation. The input to each Transformer block is modified as:

$$\tilde{\mathbf{x}}_l = \mathbf{x}_l + \mathbf{W}_l^2\, \sigma(\mathbf{W}_l^1 \mathbf{x}_l), \tag{6}$$

where $\mathbf{W}_l^1 \in \mathbb{R}^{r\times d}$, $\mathbf{W}_l^2 \in \mathbb{R}^{d\times r}$, $r < d$, and $\sigma(\cdot) = \text{ReLU}(\cdot)$ by default. We also consider the commonly used SwiGLU as an alternative nonlinearity in Section 5. The nonlinear activation between the two low-rank projections increases the flexibility of the learned interaction pattern with negligible computational cost. The transformed state $\tilde{\mathbf{x}}_l$ then serves as input to the standard Transformer block

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}_l(\tilde{\mathbf{x}}_l).$$

In total, ResLorB introduces additional $4drL$ parameters, where $L$ is the number of layers.

We place the low-rank branch only at the block input rather than at the block output or at both locations for two reasons. First, the transformed representation participates throughout the entire block computation while immediately benefiting from the existing pre-normalization. Second, a single pre-block branch introduces fewer additional computation pathways, which is also more favorable for training stability. The pseudo-code of ResLorB is provided in Appendix A.

## 4.2 Theoretical Analysis

Since the low-rank branch can represent the zero transformation, ResLorB is at least as expressive as the corresponding baseline. To better understand how the low-rank branches improve the repre-

sentation capability, we provide a theoretical analysis in an in-context learning setting for nonlinear regression, similar to (Sun et al., 2025). Specifically, let $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n+1)} \sim \mathrm{Unif}(\mathbb{S}^{d-1})$ be i.i.d. samples, and define the nonlinear feature mapping $g(\mathbf{x}) = \mathbf{x} + (\mathbf{1}^\top \mathbf{x})^2 \mathbf{1}$. The labels are generated according to $y^{(i)} = \mathbf{w}^\top g(\mathbf{x}^{(i)})$, where $\mathbf{w} \sim \mathcal{N}(0, I_d)$ is the task vector sampled independently for each task. We consider the standard in-context learning input format

$$\mathbf{Z} = \begin{pmatrix} \mathbf{x}^{(1)} & \cdots & \mathbf{x}^{(n)} & \mathbf{x}^{(n+1)} \\ y^{(1)} & \cdots & y^{(n)} & 0 \end{pmatrix},$$

where the last token corresponds to the query point. Following prior analyses (Von Oswald et al., 2023; Mahankali et al., 2024), we consider a single linear attention layer with residual connections as the baseline model. The baseline model $f_0$ and ResLorB $f_1$ can be expressed as

$$f_0(\mathbf{Z}) = \mathbf{Z} + \frac{1}{n}\,\mathrm{Attn}(\mathbf{Z}), \qquad f_1(\mathbf{Z}) = \mathbf{Z} + \frac{1}{n}\mathrm{Attn}(\mathbf{Z} + \mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{Z})).$$

The in-context risk is defined as

$$R_{\mathbf{Z}}(f_i) = \mathbb{E}_{\mathbf{Z},\mathbf{w}}\left[\left(f_i(\mathbf{Z})_{[d+1,n+1]} + y^{(n+1)}\right)^2\right], \quad i = 0, 1.$$

The following theorem shows that ResLorB achieves a strictly smaller in-context risk than the baseline model.

**Theorem 1.** *Consider the above in-context learning setting. Let $n \geq d$ and $r \geq d^{\frac{3}{4}}$. Then it holds that*

$$\inf R_{\mathbf{Z}}(f_0) - \inf R_{\mathbf{Z}}(f_1) = \Omega(d) > 0.$$

The proof is provided in Appendix B. Intuitively, the baseline model can only operate on the original hidden states, whereas ResLorB introduces neuron-level interactions that enable the hidden states to be lifted and refined before attention, leading to improved in-context learning performance.

### 4.3 Extensions

ResLorB is naturally compatible with both HC/mHC and FC due to its lightweight pre-block design. Since the HC/mHC projection already produces the block input, the ResLorB branch can be directly inserted after $\mathbf{H}_l^{pre}\mathbf{S}_l$, yielding a straightforward combined formulation as illustrated in Fig. 2(a):

$$\tilde{\mathbf{x}}_l = \mathbf{H}_l^{pre}\mathbf{S}_l + \mathbf{W}_l^2\sigma(\mathbf{W}_l^1\mathbf{H}_l^{pre}\mathbf{S}_l). \tag{7}$$

We refer to this direct combination as *mHC-ResLorB*. The resulting $\tilde{\mathbf{x}}_l$ is then processed by the standard Transformer block. ResLorB can be combined with FC in the same manner.

Beyond direct compatibility, the low-rank branch can serve as an additional interactable state within multi-state frameworks. Taking mHC as an example, instead of reusing the same HC projection between the residual path and the low-rank branch, we allow the branch to maintain an independent interaction projection, i.e.,

$$\tilde{\mathbf{x}}_l = \mathbf{H}_l^{pre,1}\mathbf{S}_l + \mathbf{W}_l^2\sigma(\mathbf{W}_l^1\mathbf{H}_l^{pre,2}\mathbf{S}_l), \tag{8}$$

where $\mathbf{H}_l^{pre,2}$ follows the same parameterization scheme as $\mathbf{H}_l^{pre,1}$ but is instantiated independently. As illustrated in Fig. 2(b), we refer to this finer-grained variant as *mHC-ResLorB+*. This design allows the low-rank branch to interact with the multi-state representation through an independent projection, introducing richer interaction patterns with only negligible additional overhead.

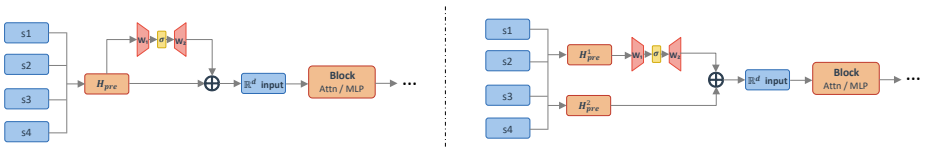



Figure 2: Illustration of ResLorB combined with mHC. (a) Left: *mHC-ResLorB*: the low-rank branch directly reuses the mHC pre-projection. (b) Right: *mHC-ResLorB+*: the low-rank branch maintains an independent pre-projection and acts as an additional interactable state.

Table 1: Validation loss of different intra-state methods on LLaMA models trained on C4. The numbers in parentheses denote additional parameter overhead.

| Model | Baseline | LAUREL | FC | NOBLE | ResLorB |
|---|---|---|---|---|---|
| 60M | 3.359 | 3.361 (+1.35%) | 3.338 (+0.30%) | 3.339 (+6.70%) | 3.344 (+1.80%) |
| 130M | 3.125 | 3.132 (+1.32%) | 3.118 (+0.59%) | 3.116 (+6.42%) | 3.116 (+1.32%) |
| 350M | 2.868 | 2.904 (+1.71%) | 2.863 (+0.28%) | 2.850 (+8.33%) | 2.859 (+1.71%) |

# 5 EXPERIMENTS

**Overview.** We evaluate ResLorB across a range of architectures and training scales. We first conduct experiments on LLaMA-series models to compare ResLorB with existing intra-state methods in Section 5.1. We then scale up to a DeepSeek-V3-style 3B model trained with 200B tokens (Xie et al., 2025), where we evaluate ResLorB under both Pre-Norm and Sandwich-Norm settings and investigate its compatibility with mHC and FC in Section 5.2. We further validate ResLorB+mHC at a larger scale on a 9B model trained with 450B tokens in Appendix D.2. We also provide analyses on training overhead and low-rank branch contributions in Section 5.3, and ablate key design choices including nonlinear activation function and rank selection in Section 5.4. Additional analyses, ablations, and training statistics are provided in Appendix D.

**Setup.** For LLaMA experiments, we consider model scales ranging from 60M to 350M parameters. All models are trained on the C4 dataset (Raffel et al., 2020) with the AdamW optimizer (Loshchilov & Hutter, 2019) for approximately one epoch following the Chinchilla scaling law (Hoffmann et al., 2022) on 8 NVIDIA RTX 4090 GPUs with bfloat16 mixed precision. For scaling experiments, we train a DeepSeek-V3-style 3B model on 200B tokens using 64 NVIDIA A800 GPUs, and further train a 9B model with 450B tokens using 32 NVIDIA H200 GPUs to evaluate ResLorB+mHC. Pretrained checkpoints are evaluated on downstream benchmarks including BBH (Suzgun et al., 2023), DROP (Dua et al., 2019), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), PIQA (Bisk et al., 2020), TriviaQA (Joshi et al., 2017), ARC-C (Clark et al., 2018). More details are provided in Appendix C.

## 5.1 RESULTS ON LLAMA MODELS

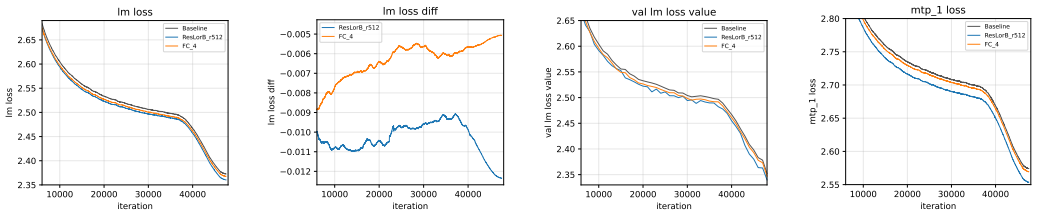
We first evaluate ResLorB on LLaMA-style Transformer models with Pre-Norm, ranging from 60M to 350M parameters. We compare against the standard baseline and representative intra-state methods: LAUREL (Menghani et al., 2025), Frac-Connection (FC) (Zhu et al., 2025b), and NOBLE (Smith, 2026). We carefully tune FC and implement it following an mHC-style formulation (see Appendix C.4). For ResLorB, we use bottleneck ranks of 64 or 48 depending on the model size, resulting in only approximately 1–2% additional parameters.

Table 1 summarizes the validation loss results. Compared with both the standard Transformer and LAUREL, ResLorB consistently improves validation loss across all model scales under similar parameter overhead, demonstrating the effectiveness of the proposed pre-block low-rank branch. Moreover, ResLorB achieves performance competitive with NOBLE while requiring far fewer additional parameters (1–2% vs. 6–8%), highlighting its superior parameter efficiency. Compared with FC, ResLorB also achieves competitive or better performance despite a modest increase in parameter count. The relative advantage over FC generally increases with model scale, a trend that becomes more evident on the DeepSeek experiments in the following subsection.

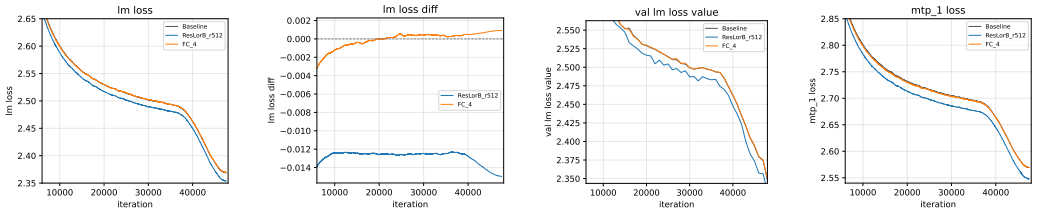
## 5.2 SCALING TO LARGER SCALE

**Scaling and Downstream Performance.** To evaluate scalability, we conduct experiments on a DeepSeek-V3-style 3B model (Xie et al., 2025) trained on 200B tokens under both Pre-Norm and Sandwich-Norm settings. We additionally incorporate the low-rank branch into the MTP (Multi-Token Prediction) module.

Fig. 3 presents the dynamics of ResLorB throughout pretraining. Under both normalization schemes, ResLorB consistently achieves lower training and validation loss than both the standard baseline and




(a) LM Loss (b) LM Loss Diff. (c) Val. LM Loss (d) MTP-1 Loss




(e) LM Loss (f) LM Loss Diff. (g) Val. LM Loss (h) MTP-1 Loss

Figure 3: Training dynamics of ResLorB on the DeepSeek-V3-style 3B model under Pre-Norm (top) and Sandwich-Norm (bottom). From left to right: training LM loss, LM loss difference relative to the baseline, validation LM loss, and MTP-1 loss.

Table 2: Downstream performance (%) of pretrained 3B models. Higher is better.

| Norm | Method | BBH | DROP | HellaSwag | MMLU | PIQA | TriviaQA | ARC-C | AVG. |
|---|---|---|---|---|---|---|---|---|---|
| Pre-Norm | Baseline | **26.91** | 15.16 | 63.13 | 25.00 | **75.41** | 35.05 | 38.14 | 39.83 |
| | ResLorB | 26.02 | **17.81** | **63.60** | **26.04** | 74.65 | **37.10** | **40.78** | **40.86** |
| Sandwich-Norm | Baseline | 26.08 | 13.33 | 62.75 | **27.98** | **74.76** | 35.43 | 37.88 | 39.74 |
| | ResLorB | **27.18** | **15.22** | **64.04** | 27.38 | 73.99 | **36.71** | **40.27** | **40.68** |

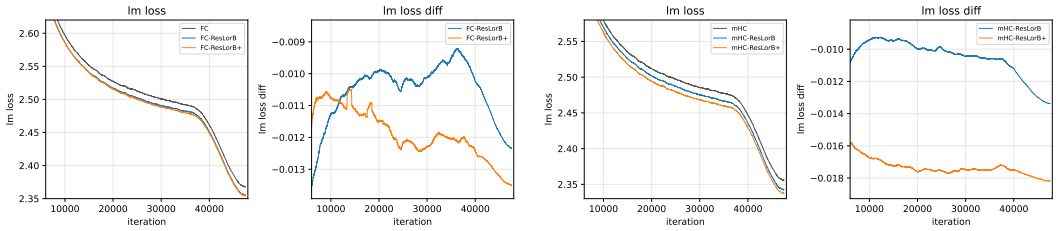
FC. More importantly, the performance gap continues to increase during the learning-rate annealing stage, indicating that the gains remain non-saturating as optimization proceeds. A similar trend is observed for the auxiliary MTP objective, suggesting that the benefit of ResLorB is not limited to the primary language modeling loss. Furthermore, under the Sandwich-Norm setting, ResLorB yields larger improvements, which we hypothesize to be due to the improved magnitude regulation introduced by the additional output normalization.

We further evaluate the pretrained checkpoints on a suite of downstream benchmarks in Table 2. The improvements in pretraining consistently translate to stronger downstream performance under both normalization schemes, demonstrating that the gains achieved by ResLorB generalize beyond training loss.

**Compatibility.** We further evaluate the compatibility of ResLorB with existing multi-state architectures. Since FC exhibits larger gains under the Pre-Norm setting (Fig. 3), we evaluate their combination in this regime. In contrast, mHC is evaluated under the Sandwich-Norm setting, where the stronger baseline provides a more challenging setting for assessing complementary improvements.

Fig. 4 summarizes the training dynamics of combined models. Under Pre-Norm (left), combining ResLorB with FC consistently improves over both the standard baseline and FC alone. Similarly, under Sandwich-Norm (right), mHC-ResLorB further improves upon mHC, despite the strong baseline performance already achieved by mHC. These results suggest that the benefit of ResLorB remains effective on top of existing multi-state architectures. We further validate the effectiveness of mHC-ResLorB at a larger scale on a 9B model trained with 450B tokens in Appendix D.2.

We further compare the FC-ResLorB+ and mHC-ResLorB+ variants described in Eq. (8). As shown in Fig. 4, FC-ResLorB+ and mHC-ResLorB+ achieve comparable or better performance than the direct-combined variants, indicating that allowing the low-rank branch to interact with multi-state representations independently can provide additional flexibility. Additional downstream comparisons are provided in Appendix D.1.




(a) Pre-Norm: FC (b) Sandwich-Norm: mHC

Figure 4: Training dynamics of ResLorB combined with existing multi-state architectures. (a) Pre-Norm with FC and (b) Sandwich-Norm with mHC. Each panel reports the training LM loss and its difference from the corresponding baseline, comparing the baseline, the shared-projection formulation (FC/mHC-ResLorB), and the independent-projection formulation (FC/mHC-ResLorB+).

Table 3: Training overhead of ResLorB on the DeepSeek-V3-style 3B model under the Sandwich-Norm setting. Total and activated parameters include the parameters of MTP layers.

| Method | Total Params | Activated Params | TFLOP/s/GPU | Max Allocated Memory |
|---|---|---|---|---|
| Baseline | 3.18B | 0.79B | 57.7 | 44.54 GB |
| ResLorB | 3.21B | 0.82B | 56.0 | 46.09 GB |

### 5.3 Overhead and Analysis

**Training Overhead.** Table 3 summarizes the training overhead introduced by ResLorB on the DeepSeek-V3-style 3B model under the Sandwich-Norm setting. All measurements are obtained under the same training configuration (EP=8, TP=PP=CP=1, global batch size of 1024, and sequence length of 4096). ResLorB introduces only approximately 1% and 4% additional total and activated parameters, respectively. Meanwhile, the additional low-rank computation results in only a minor reduction in training throughput and a small increase in maximum allocated GPU memory. Furthermore, since the low-rank branch only modifies the block input and does not introduce additional KV states, it does not incur extra KV-cache overhead during autoregressive decoding. These results demonstrate that ResLorB provides an effective capacity improvement with limited computational and memory overhead.

**Branch Contribution.** To verify that the low-rank branch does not collapse to a negligible perturbation, we analyze its magnitude relative to the original residual representation at the final checkpoint. Fig. 5 reports the norm ratio of the low-rank branch for both the attention and MLP branches under Pre-Norm and Sandwich-Norm settings. The ratio remains stable throughout training and stays at roughly 25–30% across both normalization schemes, indicating that the low-rank branch remains active while providing a moderate contribution to the residual stream.

### 5.4 Ablation Studies

**Low-Rank Branch Design.** We first investigate the design choices of the low-rank branch, including the activation function and branch placement. Specifically, we compare identity (linear), ReLU, and SwiGLU activations, as well as different low-rank branch locations, including pre-block, post-block, and pre- and post-block placements under the Sandwich-Norm setting.

Table 4 shows that both nonlinearity and branch placement are important for ResLorB. The linear branch performs worse than nonlinear alternatives, suggesting that the nonlinear feature transformation provides additional modeling capacity. Meanwhile, ReLU and SwiGLU achieve comparable performance, and we adopt ReLU as the default activation due to its simpler and more lightweight implementation.

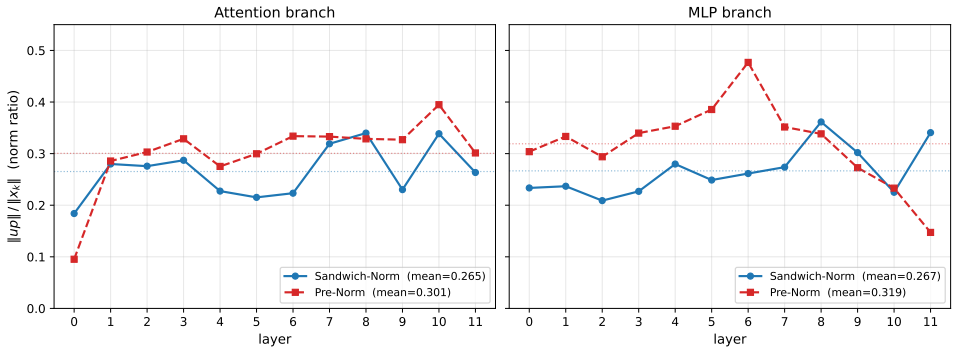



Figure 5: Norm ratio of the low-rank branch relative to the original residual representation at the final checkpoint. The left and right panels correspond to the attention and MLP branches, respectively, with separate curves for Pre-Norm and Sandwich-Norm.

Table 4: Ablation of low-rank branch design choices.

| **Configuration** | **Val. Loss** |
|---|---|
| Identity (Linear) | 2.347 |
| SwiGLU | 2.333 |
| Post-Block (ReLU) | 2.339 |
| Pre- & Post-Block (ReLU, $r/2$) | 2.334 |
| ResLorB (ReLU, Pre-Block) | 2.333 |

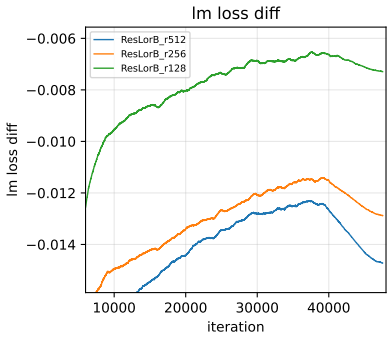



Figure 6: Rank scaling of ResLorB.

For branch placement, the proposed pre-block design consistently achieves the best performance compared with post-block and pre- and post-block alternatives, indicating that applying the low-rank transformation before Transformer computation is more effective.

**Rank Scaling.** We further study the effect of the bottleneck rank $r$ that controls the capacity of the low-rank branch on the DeepSeek-V3-style 3B model. Starting from the default setting of $r = 512$, we compare smaller ranks of $r = 256$ and $r = 128$. As shown in Fig. 6, reducing the rank to 256 results in only marginal performance degradation, indicating that ResLorB can achieve a favorable performance-efficiency trade-off with a smaller bottleneck dimension. However, further reducing the rank to 128 leads to a larger performance gap, although it still consistently outperforms the baseline. These results suggest that although ResLorB remains effective with compact low-rank branches, sufficient rank capacity is still beneficial for capturing richer feature interactions.

## 6 Conclusions

In this paper, we proposed ResLorB, a lightweight architectural modification that introduces a nonlinear low-rank branch at the input of each Transformer block to enable adaptive neuron-level interactions. Extensive experiments on models up to 3B/9B parameters trained on 200B/450B tokens show that ResLorB consistently improves validation loss and downstream performance over strong baselines, with only about 1–2% additional parameters and negligible inference overhead. Moreover, it is fully compatible with existing multi-state architectures such as mHC, yielding further complementary gains. A theoretical analysis in an in-context learning setting further provides a preliminary understanding of the effectiveness of the proposed design. In future work, we plan to validate ResLorB on larger and more extreme sparse MoE models and to gain a deeper understanding of the underlying mechanisms.

## AI USE STATEMENT

AI tools were used solely for language editing and proofreading to improve the clarity and readability of this manuscript. All technical content, experiments, analyses, and conclusions were developed and verified by the authors.

## REPRODUCIBILITY STATEMENT

Appendix A provides the pseudo-code of the ResLorB branch, covering both the ReLU and SwiGLU variants. Appendix C documents the model configurations, training hyperparameters, and evaluation protocol for all experiments, together with the implementation details of the ResLorB branch and of the Frac-Connection and mHC baselines, including the rank and initialization settings. Appendix D reports the additional downstream evaluations, the 9B-scale results, the training dynamics, and the ablation studies on the design choices of ResLorB.

## REFERENCES

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *AAAI Conference on Artificial Intelligence*, 2020.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 2020.

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.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 2023.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

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.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, DDL Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *ArXiv preprint arXiv:2203.15556*, 2022.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *International Conference on Machine Learning*, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *conference on Computer Vision and Pattern Recognition*, 2017.

Peng Jin, Bo Zhu, Yuan Li, and Shuicheng Yan. Moe++: Accelerating mixture-of-experts methods with zero-computation experts. In *International Conference on Learning Representations*, 2025.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Annual Meeting of the Association for Computational Linguistics*, 2017.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *ArXiv preprint arXiv:2001.08361*, 2020.

Kimi Team. Attention residuals. *ArXiv preprint arXiv:2603.15031*, 2026.

Tianyu Li, Dongchen Han, Zixuan Cao, Haofeng Huang, Mengyu Zhou, Ming Chen, Erchao Zhao, Xiaoxi Jiang, Guanjun Jiang, and Gao Huang. Siamesenorm: Breaking the barrier to reconciling pre/post-norm. *ArXiv preprint arXiv:2602.08064*, 2026.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *ArXiv preprint arXiv:2412.19437*, 2024a.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *International Conference on Machine Learning*, 2024b.

Ziyue Liu, Ruijie Zhang, Zhengyang Wang, Mingsong Yan, Zi Yang, Paul D Hovland, Bogdan Nicolae, Franck Cappello, Sui Tang, and Zheng Zhang. Cola: Compute-efficient pre-training of llms via low-rank activation. In *Conference on Empirical Methods in Natural Language Processing*, 2025.

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

Arvind Mahankali, Tatsunori Hashimoto, and Tengyu Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. In *International Conference on Learning Representations*, 2024.

Gaurav Menghani, Ravi Kumar, and Sanjiv Kumar. Laurel: Learned augmented residual layer. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *International Conference on Machine Learning*, 2025.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *ArXiv preprint arXiv:2512.13961*, 2025.

Matteo Pagliardini, Amirkeivan Mohtashami, Francois Fleuret, and Martin Jaggi. Denseformer: Enhancing information flow in transformers via depth weighted averaging. *Advances in Neural Information Processing Systems*, 2024.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn A. Walker, Heng Ji, and Amanda Stent (eds.), *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

Zihan Qiu, Zeyu Huang, Kaiyue Wen, Peng Jin, Bo Zheng, Yuxin Zhou, Haofeng Huang, Zekun Wang, Xiao Li, Huaqing Zhang, et al. A unified view of attention and residual sinks: Outlier-driven rescaling is essential for transformer training. *ArXiv preprint arXiv:2601.22966*, 2026a.

Zihan Qiu, Zekun Wang, Xiao Li, Yanpeng Li, Yang Xu, Yixuan Wang, Huaqing Zhang, Rui Men, Bochao Mao, Chengruidong Zhang, et al. On the design of qwen3. 8-next architecture: Evaluation, efficiency, and training stability. *ArXiv preprint arXiv:2608.30320*, 2026b.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, et al. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *Advances in Neural Information Processing Systems*, 2026c.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

Shuhua Shi, Shaohan Huang, Minghui Song, Zhoujun Li, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. Reslora: Identity residual mapping in low-rank adaption. In *Findings of the Association for Computational Linguistics*, 2024.

Ethan Smith. Noble: Accelerating transformers with nonlinear low-rank branches. *ArXiv preprint arXiv:2603.06492*, 2026.

Jianlin Su. ¡english translated title¿. Scientific Spaces Blog, 2025. URL `https://spaces.ac.cn/archives/11664`.

Haoyuan Sun, Ali Jadbabaie, and Navid Azizan. On the role of transformer feed-forward layers in nonlinear in-context learning. *ArXiv preprint arXiv:2501.18187*, 2025.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics*, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *ArXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, 2023.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *ArXiv preprint arXiv:2408.15664*, 2024.

Zhoutong Wu, Yimu Zhang, Cong Fang, and Zhouchen Lin. Separation and bias of deep equilibrium models on expressivity and learning dynamics. *Advances in Neural Information Processing Systems*, 2024.

Da Xiao, Qingye Meng, Shengping Li, and Xingyuan Yuan. Muddformer: Breaking residual bottlenecks in transformers via multiway dynamic dense connections. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *International Conference on Machine Learning*, 2025.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Kuai Yu, et al. mhc: Manifold-constrained hyper-connections. *ArXiv preprint arXiv:2512.24880*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In Anna Korhonen, David R. Traum, and Lluís Màrquez (eds.), *Conference of the Association for Computational Linguistics*, 2019.

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. *ArXiv preprint arXiv:2303.10512*, 2023.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient LLM training by gradient low-rank projection. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *International Conference on Machine Learning*, 2024.

Zhanchao Zhou, Tianyi Wu, Zhiyun Jiang, Fares Obeid, and Zhenzhong Lan. Value residual learning. In *Annual Meeting of the Association for Computational Linguistics*, 2025.

Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. Hyper-connections. In *International Conference on Learning Representations*, 2025a.

Defa Zhu, Hongzhi Huang, Jundong Zhou, Zihao Huang, Yutao Zeng, Banggu Wu, Qiyang Min, and Xun Zhou. Frac-connections: Fractional extension of hyper-connections. *ArXiv preprint arXiv:2503.14125*, 2025b.

# APPENDIX

## A PSEUDO-CODE OF RESLORB

Algorithm 1 summarizes the forward computation of ResLorB. The low-rank branch is applied to the input of every Transformer block, and its nonlinearity $\sigma(\cdot)$ can be either ReLU (the default) or SwiGLU.

**Algorithm 1** ResLorB: nonlinear low-rank branch at the Transformer block input

**Require:** block input $\mathbf{x}$; rank $r$; nonlinearity type $\tau \in \{\mathrm{ReLU}, \mathrm{SwiGLU}\}$
**Ensure:** transformed block input $\tilde{\mathbf{x}}$
**if** $\tau = \mathrm{ReLU}$ **then**
  $\mathbf{h} \leftarrow \mathrm{ReLU}(\mathbf{W}^1\mathbf{x})$ ▷ $\mathbf{W}^1 \in \mathbb{R}^{r\times d}$
**else**
  $\mathbf{g} \leftarrow \mathbf{W}^{\mathrm{gate}}\mathbf{x}, \quad \mathbf{u} \leftarrow \mathbf{W}^{\mathrm{up}}\mathbf{x}$ ▷ $\mathbf{W}^{\mathrm{gate}}, \mathbf{W}^{\mathrm{up}} \in \mathbb{R}^{r'\times d}, \ \ r' = \lfloor 2r/3 \rfloor$
  $\mathbf{h} \leftarrow \mathrm{SiLU}(\mathbf{g}) \odot \mathbf{u}$
**end if**
$\tilde{\mathbf{x}} \leftarrow \mathbf{x} + \mathbf{W}^2\mathbf{h}$ ▷ $\mathbf{W}^2 \in \mathbb{R}^{d\times r}$ ($\mathbb{R}^{d\times r'}$ for SwiGLU)
**return** $\tilde{\mathbf{x}}$

The transformed input $\tilde{\mathbf{x}}$ is then processed by the standard Transformer block, i.e., $\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}_l(\tilde{\mathbf{x}}_l)$. Both variants introduce approximately $4drL$ additional parameters, where $L$ is the number of layers.

## B PROOF OF THEOREM 1

We first recall the nonlinear in-context learning setting considered in the theorem. Then, we introduce several auxiliary results that will be used in the proof.

We consider i.i.d. inputs $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n+1)} \sim \mathrm{Unif}\left(\mathbb{S}^{d-1}\right)$, and define $g(\mathbf{x}) = \mathbf{x} + \left(\mathbf{1}^\top\mathbf{x}\right)^2 \mathbf{1}$. The task vector is sampled from $\mathbf{w} \sim \mathcal{N}(0, I_d)$, and the labels are generated by $y^{(i)} = \mathbf{w}^\top g(\mathbf{x}^{(i)})$. The input sequence is

$$\mathbf{Z} = \begin{pmatrix} \mathbf{x}^{(1)} & \cdots & \mathbf{x}^{(n)} & \mathbf{x}^{(n+1)} \\ y^{(1)} & \cdots & y^{(n)} & 0 \end{pmatrix}.$$

For any model $f$, define the in-context risk as

$$R_{\mathbf{Z}}(f) = \mathbb{E}_{\mathbf{Z},\mathbf{w}}\left[\left(f(\mathbf{Z})_{[d+1,n+1]} + \mathbf{w}^\top g\left(\mathbf{x}^{(n+1)}\right)\right)^2\right].$$

We consider a linear attention block with residual connection as the baseline model $f_0$, i.e.,

$$f_0(\mathbf{Z}) = \mathbf{Z} + \frac{1}{n}\,\mathrm{Attn}_{\mathbf{P},\mathbf{Q}}(\mathbf{Z}), \qquad \mathrm{Attn}_{\mathbf{P},\mathbf{Q}}(\mathbf{Z}) = \mathbf{P}\,\mathbf{Z}\mathbf{M}\,(\mathbf{Z}^\top\mathbf{Q}\mathbf{Z}), \tag{9}$$

where $\mathbf{P}$ is the value weight matrix, $\mathbf{Q}$ is the reparameterization of $\mathbf{W}_k^\top\mathbf{W}_Q$, and $\mathbf{M} = \mathrm{diag}(I_n, 0)$ is the causal mask. Our ResLorB model $f_1$ can be expressed as

$$f_1(\mathbf{Z}) = \mathbf{Z} + \frac{1}{n}\mathrm{Attn}_{\mathbf{P},\mathbf{Q}}(\mathbf{Z} + \mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{Z})), \tag{10}$$

where $\sigma(\cdot)$ is the ReLU activation function by default.

In the proof, we also consider an 'ideal' ResLorB model $f_2$ defined as follows:

$$f_2(\mathbf{Z}) = \mathbf{Z} + \frac{1}{n}\mathrm{Attn}_{\mathbf{P},\mathbf{Q}}(\mathbf{Z}'), \quad \mathbf{Z}' = \begin{pmatrix} g\left(\mathbf{x}^{(1)}\right) & \cdots & g\left(\mathbf{x}^{(n)}\right) & g\left(\mathbf{x}^{(n+1)}\right) \\ y^{(1)} & \cdots & y^{(n)} & 0 \end{pmatrix}. \tag{11}$$

The ideal model $f_2$ assumes that the low-rank branch can exactly implement the nonlinear feature lifting $g(\mathbf{x}) = \mathbf{x} + \left(\mathbf{1}^\top\mathbf{x}\right)^2 \mathbf{1}$ before the attention operation.

We next introduce the theoretical results from Sun et al. (2025) that will be used in our proof.

**Lemma 2** (Proposition 11, Sun et al. (2025)). *Let* $\mathbf{X} = \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n+1)}\right)$ *and* $f_0$ *be defined as above. For any fixed task vector* $\mathbf{w}$*, it holds that*

$$\mathbb{E}_{\mathbf{X}}\left[\left(f_0(\mathbf{Z})_{[d+1,n+1]} + \mathbf{w}^\top g\left(\mathbf{x}^{(n+1)}\right)\right)^2\right] \geq \min_{\boldsymbol{\beta}} \mathbb{E}_{\mathbf{X}}\left[\left(\boldsymbol{\beta}^\top \mathbf{x}^{(n+1)} + \mathbf{w}^\top g\left(\mathbf{x}^{(n+1)}\right)\right)^2\right].$$

**Lemma 3** (Corollary of Lemma 12, Sun et al. (2025)). *Let* $\mathbf{Z}'$ *be defined as in Eq. (11). For any fixed task vector* $\mathbf{w}$ *and* $\mathbf{x} \sim$ *Unif* $\left(\mathbb{S}^{d-1}\right)$*, denote* $\mathbf{u} = g(\mathbf{x})$ *and* $v = \mathbf{w}^\top \mathbf{u}$*. Further define*

$$\Lambda = \mathbb{E}_{\mathbf{x}}\left[\mathbf{u}\mathbf{u}^\top\right], \quad \Phi = \mathrm{Cov}_{\mathbf{x}}(\mathbf{u}v), \quad \xi = \mathbb{E}_{\mathbf{x}}[\mathbf{u}v].$$

*Then for a baseline model with* $\boldsymbol{\Gamma} = -\Lambda^{-1}$ *and*

$$\mathbf{P} = \begin{pmatrix} 0_{d\times d} & 0 \\ 0 & 1 \end{pmatrix}, \qquad \mathbf{Q} = \begin{pmatrix} \boldsymbol{\Gamma} & 0 \\ 0 & 0 \end{pmatrix},$$

*the prediction error satisfies*

$$\mathbb{E}_{\mathbf{X}}\left[\left(f_0(\mathbf{Z}')_{[d+1,n+1]} + \mathbf{w}^\top g\left(\mathbf{x}^{(n+1)}\right)\right)^2\right] = \frac{1}{n}\,\mathrm{tr}\left(\Lambda^{-1}\Phi\right).$$

The following lemma gives an upper bound for a two-layer ReLU NN to approximate quadratic function.

**Lemma 4** (Lemma 7 in Wu et al. (2024)). *For any* $r \in \mathbb{N}^+$*, there exists a two-layer ReLU FNN* $\varphi : \mathbb{R} \to \mathbb{R}$ *with width* $2r$ *such that*

$$\left|\varphi(x) - x^2\right| \leq 4r^{-2},$$

*for all* $x \in [-1, 1]$.

We now begin our proof.

*Proof.* We will first use Lemma 2 for the baseline model and Lemma 3 for the 'ideal' ResLorB model to give an approximation bound. Then we will use Lemma 4 to analyze the error between the ideal ResLorB and true ResLorB model.

**Baseline model.** Denote

$$L_i = \mathbb{E}_{\mathbf{X}}\left[\left(f_i(\mathbf{Z})_{[d+1,n+1]} + \mathbf{w}^\top g\left(\mathbf{x}^{(n+1)}\right)\right)^2\right], \quad i = 0, 1, 2.$$

By Lemma 2, for any fixed task vector $\mathbf{w}$,

$$L_0 \geq \min_{\boldsymbol{\beta}} \mathbb{E}_{\mathbf{x}}\left[\left(\boldsymbol{\beta}^\top \mathbf{x} + \mathbf{w}^\top g(\mathbf{x})\right)^2\right].$$

Substituting $g(\mathbf{x}) = \mathbf{x} + (\mathbf{1}^\top \mathbf{x})^2 \mathbf{1}$, we have

$$\boldsymbol{\beta}^\top \mathbf{x} + \mathbf{w}^\top g(\mathbf{x}) = (\boldsymbol{\beta} + \mathbf{w})^\top \mathbf{x} + (\mathbf{w}^\top \mathbf{1})(\mathbf{1}^\top \mathbf{x})^2.$$

Since $\mathbf{x}$ is sampled from the uniform distribution over $\mathbb{S}^{d-1}$, the linear term is orthogonal to the quadratic term. Therefore, the optimal choice is $\boldsymbol{\beta}^* = -\mathbf{w}$, and

$$L_0 \geq (\mathbf{w}^\top \mathbf{1})^2 \mathbb{E}_{\mathbf{x}}\left[(\mathbf{1}^\top \mathbf{x})^4\right].$$

Let $z = \mathbf{1}^\top \mathbf{x}$. Using the fourth-order moment of the uniform distribution on $\mathbb{S}^{d-1}$,

$$\mathbb{E}[z^4] = \frac{3\|\mathbf{1}\|_2^4}{d(d+2)} = \frac{3d^2}{d(d+2)} = \frac{3d}{d+2}.$$

Thus,

$$L_0 \geq \frac{3d}{d+2}(\mathbf{w}^\top \mathbf{1})^2.$$

Taking expectation over $\mathbf{w} \sim \mathcal{N}(0, I_d)$ yields

$$\inf R_{\mathbf{Z}}(f_0) = \mathbb{E}_{\mathbf{w}}[L_0] \geq \frac{3d}{d+2}\mathbb{E}_{\mathbf{w}}[(\mathbf{w}^\top \mathbf{1})^2] = \frac{3d^2}{d+2}.$$

**Ideal ResLorB.** Next we consider the 'ideal' ResLorB model $f_2$. With the notations in Lemma 3, we have

$$L_2(\mathbf{w}) = \frac{1}{n} \operatorname{tr}\left(\Lambda^{-1}\Phi\right),$$

where

$$\Lambda = \mathbb{E}_{\mathbf{x}}\left[\mathbf{u}\mathbf{u}^\top\right], \quad \Phi = \operatorname{Cov}_{\mathbf{x}}(\mathbf{u}v), \quad \mathbf{u} = g(\mathbf{x}), \quad v = \mathbf{w}^\top \mathbf{u}.$$

We first characterize the covariance matrix $\Phi$ and the second moment matrix $\Lambda$. Let

$$z = \mathbf{1}^\top \mathbf{x}.$$

Since $\mathbf{x} \sim \operatorname{Unif}(\mathbb{S}^{d-1})$, the random variable $z$ satisfies

$$\mathbb{E}[z^2] = 1, \qquad \mathbb{E}[z^4] = \frac{3d}{d+2}.$$

For the second moment matrix, we have

$$\Lambda = \mathbb{E}\left[g(\mathbf{x})g(\mathbf{x})^\top\right].$$

Expanding $g(\mathbf{x})$, we obtain

$$\Lambda = \mathbb{E}[\mathbf{x}\mathbf{x}^\top] + \mathbb{E}[z^2\mathbf{x}]\mathbf{1}^\top + \mathbf{1}\mathbb{E}[z^2\mathbf{x}]^\top + \mathbb{E}[z^4]\mathbf{1}\mathbf{1}^\top.$$

By the symmetry of the spherical distribution,

$$\mathbb{E}[z^2\mathbf{x}] = 0,$$

and therefore

$$\Lambda = \frac{1}{d}I_d + \frac{3d}{d+2}\mathbf{1}\mathbf{1}^\top.$$

Define

$$P_\parallel = \frac{1}{d}\mathbf{1}\mathbf{1}^\top, \qquad P_\perp = I - P_\parallel.$$

Then

$$\Lambda = \frac{1}{d}P_\perp + \left(\frac{1}{d} + \frac{3d^2}{d+2}\right)P_\parallel.$$

Hence,

$$\Lambda^{-1} = dP_\perp + \frac{d(d+2)}{3d^3+d+2}P_\parallel = dP_\perp + O(d^{-1})P_\parallel.$$

Next we bound $\operatorname{tr}(\Lambda^{-1}\Phi)$. By direct calculation,

$$\mathbf{u}v = \left(\mathbf{x} + z^2\mathbf{1}\right)\left(\mathbf{w}^\top\mathbf{x} + (\mathbf{1}^\top\mathbf{w})z^2\right),$$

which can be decomposed as

$$\mathbf{u}v = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4,$$

where

$$\mathbf{s}_1 = \mathbf{x}(\mathbf{w}^\top\mathbf{x}), \qquad \mathbf{s}_2 = (\mathbf{1}^\top\mathbf{w})z^2\mathbf{x},$$
$$\mathbf{s}_3 = z^2\mathbf{1}(\mathbf{w}^\top\mathbf{x}), \qquad \mathbf{s}_4 = (\mathbf{1}^\top\mathbf{w})z^4\mathbf{1}.$$

Recall that

$$\Lambda^{-1} = dP_\perp + \lambda_\parallel^{-1}P_\parallel,$$

where

$$\lambda_\parallel^{-1} = \frac{d(d+2)}{3d^3+d+2} = O(d^{-1}).$$

Hence,

$$\operatorname{tr}(\Lambda^{-1}\Phi) = d\operatorname{tr}(P_\perp\Phi) + \lambda_\parallel^{-1}\operatorname{tr}(P_\parallel\Phi).$$

Noting that $P_\perp^\top P_\perp = P_\perp$ and $P_\parallel^\top P_\parallel = P_\parallel$, we then have

$$\operatorname{tr}(A\Phi) = \mathbb{E}_{\mathbf{x}}\left[\|A(\mathbf{u}v - \mathbb{E}_{\mathbf{x}}[\mathbf{u}v])\|^2\right], \quad A = P_\perp, P_\parallel.$$

We first consider the orthogonal component. Since $\mathbf{s}_3$, $\mathbb{E}[\mathbf{s}_3]$ and $\mathbf{s}_4$, $\mathbb{E}[\mathbf{s}_4]$ are parallel to $\mathbf{1}$, they vanish after projection by $P_\perp$. Therefore,

$$P_\perp(\mathbf{u}v - \mathbb{E}[\mathbf{u}v]) = P_\perp\left(\mathbf{s}_1 + \mathbf{s}_2 - \mathbb{E}[\mathbf{s}_1 + \mathbf{s}_2]\right).$$

By the inequality $\|a+b\|^2 \le 2\|a\|^2 + 2\|b\|^2$, we have

$$\operatorname{tr}(P_\perp\Phi) \le 2\mathbb{E}\left[\|P_\perp(\mathbf{s}_1 - \mathbb{E}[\mathbf{s}_1])\|^2\right] + 2\mathbb{E}\left[\|P_\perp\mathbf{s}_2\|^2\right],$$

where we used $\mathbb{E}[\mathbf{s}_2] = 0$ due to spherical symmetry.

For the first term,

$$\mathbb{E}\left[\|\mathbf{s}_1\|^2\right] = \mathbb{E}\left[(\mathbf{w}^\top\mathbf{x})^2\|\mathbf{x}\|^2\right] = \mathbb{E}\left[(\mathbf{w}^\top\mathbf{x})^2\right] = \frac{\|\mathbf{w}\|^2}{d}.$$

For the second term,

$$\mathbb{E}\left[\|\mathbf{s}_2\|^2\right] = (\mathbf{1}^\top\mathbf{w})^2\mathbb{E}[z^4]\|\mathbf{x}\|^2 = (\mathbf{1}^\top\mathbf{w})^2\frac{3d}{d+2}.$$

Therefore,

$$\operatorname{tr}(P_\perp\Phi) = O\left(\frac{\|\mathbf{w}\|^2}{d} + (\mathbf{1}^\top\mathbf{w})^2\right).$$

For the parallel component, it suffices to bound

$$\mathbb{E}\left\|P_\parallel\mathbf{s}_1\right\|^2 + \mathbb{E}\left\|P_\parallel\mathbf{s}_2\right\|^2 + \mathbb{E}\left\|P_\parallel\mathbf{s}_3\right\|^2 + \mathbb{E}\left\|P_\parallel\mathbf{s}_4\right\|^2.$$

For $\mathbf{s}_1$ and $\mathbf{s}_2$, since $P_\parallel$ is an orthogonal projection, we have

$$\mathbb{E}\|P_\parallel\mathbf{s}_1\|^2 \le \mathbb{E}\|\mathbf{s}_1\|^2 = \frac{\|\mathbf{w}\|^2}{d},$$

and

$$\mathbb{E}\|P_\parallel\mathbf{s}_2\|^2 \le \mathbb{E}\|\mathbf{s}_2\|^2 = (\mathbf{1}^\top\mathbf{w})^2\mathbb{E}[z^4].$$

Therefore, the contribution from the first two terms is

$$O\left(\frac{\|\mathbf{w}\|^2}{d} + (\mathbf{1}^\top\mathbf{w})^2\right).$$

For $\mathbf{s}_3$ and $\mathbf{s}_4$, both vectors lie in $\operatorname{span}(\mathbf{1})$, and hence $P_\parallel\mathbf{s}_i = \mathbf{s}_i$ for $i = 3, 4$. For $\mathbf{s}_3$, we have

$$\|\mathbf{s}_3\|^2 = \|z^2\mathbf{1}(\mathbf{w}^\top\mathbf{x})\|^2 = dz^4(\mathbf{w}^\top\mathbf{x})^2.$$

Using the bounded fourth moment of spherical distributions,

$$\mathbb{E}\left[z^4(\mathbf{w}^\top\mathbf{x})^2\right] = O\left(\frac{\|\mathbf{w}\|^2}{d}\right),$$

we obtain

$$\mathbb{E}\|\mathbf{s}_3\|^2 = O(\|\mathbf{w}\|^2).$$

Similarly,

$$\|\mathbf{s}_4\|^2 = d(\mathbf{1}^\top\mathbf{w})^2z^8,$$

and therefore

$$\mathbb{E}\|\mathbf{s}_4\|^2 = d(\mathbf{1}^\top\mathbf{w})^2\mathbb{E}[z^8].$$

Since

$$\mathbb{E}[z^8] = O(1)$$

for $\mathbf{x} \sim \mathrm{Unif}(\mathbb{S}^{d-1})$, we have

$$\mathbb{E}\|\mathbf{s}_4\|^2 = O\left(d(\mathbf{1}^\top \mathbf{w})^2\right).$$

Combining the four terms gives

$$\mathrm{tr}(P_\parallel \Phi) = O\left(\|\mathbf{w}\|^2 + d(\mathbf{1}^\top \mathbf{w})^2\right).$$

Therefore, we obtain

$$\begin{aligned}\mathrm{tr}(\Lambda^{-1}\Phi) &= d\,\mathrm{tr}(P_\perp \Phi) + \lambda_\parallel^{-1}\,\mathrm{tr}(P_\parallel \Phi) \\ &= O\left(\|\mathbf{w}\|^2 + d(\mathbf{1}^\top \mathbf{w})^2\right) + O(d^{-1})O\left(\|\mathbf{w}\|^2 + d(\mathbf{1}^\top \mathbf{w})^2\right).\end{aligned}$$

Taking expectation over $\mathbf{w} \sim \mathcal{N}(0, I_d)$, using

$$\mathbb{E}\|\mathbf{w}\|^2 = d, \qquad \mathbb{E}(\mathbf{1}^\top \mathbf{w})^2 = d,$$

gives

$$\mathbb{E}_{\mathbf{w}}\,\mathrm{tr}(\Lambda^{-1}\Phi) = O(d^2).$$

Therefore,

$$R_{\mathbf{Z}}(f_2) = \mathbb{E}_{\mathbf{w}}[L_2(\mathbf{w})] = O\left(\frac{d^2}{n}\right).$$

**ResLorB.** Finally we consider the approximation error between the ideal ResLorB model $f_2$ and the true ResLorB model $f_1$.

Since $\mathbf{x} \sim \mathrm{Unif}(\mathbb{S}^{d-1})$, we have

$$|\mathbf{1}^\top \mathbf{x}| \le \|\mathbf{1}\|_2 \|\mathbf{x}\|_2 = \sqrt{d}.$$

Therefore,

$$t(\mathbf{x}) = \frac{\mathbf{1}^\top \mathbf{x}}{\sqrt{d}} \in [-1, 1].$$

By Lemma 4, there exists a two-layer ReLU network $\varphi$ satisfying

$$|\varphi(t) - t^2| \le 4r^{-2},$$

for all $t \in [-1, 1]$. We construct the approximate nonlinear lifting as

$$\hat{g}(\mathbf{x}) = \mathbf{x} + d\varphi\left(\frac{\mathbf{1}^\top \mathbf{x}}{\sqrt{d}}\right)\mathbf{1}.$$

Then the approximation error satisfies

$$\Delta(\mathbf{x}) = \hat{g}(\mathbf{x}) - g(\mathbf{x}) = d\left(\varphi(t(\mathbf{x})) - t(\mathbf{x})^2\right)\mathbf{1}.$$

Hence,

$$\|\Delta(\mathbf{x})\| \le d \cdot 4r^{-2} \cdot \|\mathbf{1}\| = 4d^{3/2} r^{-2}.$$

Denote

$$\delta_f = f_1(\mathbf{Z})_{[d+1,n+1]} - f_2(\mathbf{Z})_{[d+1,n+1]}.$$

Then,

$$\begin{aligned}L_1 - L_2 &= \mathbb{E}\left[\left(f_1(\mathbf{Z})_{[d+1,n+1]} + \mathbf{w}^\top g(\mathbf{x}^{(n+1)})\right)^2\right] - \mathbb{E}\left[\left(f_2(\mathbf{Z})_{[d+1,n+1]} + \mathbf{w}^\top g(\mathbf{x}^{(n+1)})\right)^2\right] \\ &= \mathbb{E}[\delta_f^2] + 2\mathbb{E}\left[\delta_f\left(f_2(\mathbf{Z})_{[d+1,n+1]} + \mathbf{w}^\top g(\mathbf{x}^{(n+1)})\right)\right].\end{aligned} \tag{12}$$

Using the explicit form of the linear attention operator,

$$f_2(\mathbf{Z})_{[d+1,n+1]} = \frac{1}{n}\sum_{i=1}^{n} y^{(i)} (u^{(i)})^\top \Gamma u^{(n+1)},$$

and

$$f_1(\mathbf{Z})_{[d+1,n+1]} = \frac{1}{n}\sum_{i=1}^{n} y^{(i)}(\hat{u}^{(i)})^\top \Gamma \hat{u}^{(n+1)},$$

where

$$u^{(i)} = g(\mathbf{x}^{(i)}), \qquad \hat{u}^{(i)} = \hat{g}(\mathbf{x}^{(i)}).$$

Let

$$\Delta^{(i)} = \hat{u}^{(i)} - u^{(i)}.$$

Then,

$$\delta_f = \frac{1}{n}\sum_{i=1}^{n} y^{(i)}\Big((\Delta^{(i)})^\top \Gamma u^{(n+1)} + (u^{(i)})^\top \Gamma \Delta^{(n+1)} + (\Delta^{(i)})^\top \Gamma \Delta^{(n+1)}\Big).$$

We first bound $\mathbb{E}[\delta_f^2]$. Using $(a+b+c)^2 \le 3(a^2+b^2+c^2)$, it suffices to bound the three terms in the above expansion. For the first term, since $\Delta^{(i)}$ is parallel to $\mathbf{1}$, we have

$$\Gamma \Delta^{(i)} = -\lambda_\parallel^{-1}\Delta^{(i)},$$

where $\lambda_\parallel = \Theta(d)$ is the eigenvalue of $\Lambda$ along the $\mathbf{1}$ direction. Therefore,

$$\begin{aligned} a := \left|\frac{1}{n}\sum_{i=1}^{n} y^{(i)}(\Delta^{(i)})^\top \Gamma u^{(n+1)}\right| &\le O(d^{-1})\frac{1}{n}\sum_{i=1}^{n}|y^{(i)}|\|\Delta^{(i)}\|\|u^{(n+1)}\| \\ &\le O(d^{1/2}r^{-2})\|u^{(n+1)}\|\frac{1}{n}\sum_{i=1}^{n}|y^{(i)}|, \end{aligned}$$

where we used $\left\|\Delta^{(i)}\right\| \le 4d^{3/2}r^{-2}$. Consequently,

$$\mathbb{E}[a^2] \le O(dr^{-4})\mathbb{E}[\|u\|^2]\mathbb{E}[y^2] = O(d^3 r^{-4}\|\mathbf{w}\|^2),$$

where $\mathbb{E}[\|u\|^2] = O(d)$ and

$$\mathbb{E}[y^2] = \mathbf{w}^\top \Lambda \mathbf{w} = O(d\|\mathbf{w}\|^2).$$

For the second term, similarly,

$$b := \left|\frac{1}{n}\sum_{i=1}^{n} y^{(i)}(u^{(i)})^\top \Gamma \Delta^{(n+1)}\right| \le O(d^{1/2}r^{-2})\frac{1}{n}\sum_{i=1}^{n}|y^{(i)}|\|u^{(i)}\|.$$

Thus,

$$\mathbb{E}[b^2] \le O(dr^{-4})\mathbb{E}[y^2\|u\|^2] \le O(dr^{-4})\sqrt{\mathbb{E}[y^4]\mathbb{E}[\|u\|^4]} = O(d^3 r^{-4}\|\mathbf{w}\|^2).$$

For the third term,

$$\begin{aligned} c := \left|\frac{1}{n}\sum_{i=1}^{n} y^{(i)}(\Delta^{(i)})^\top \Gamma \Delta^{(n+1)}\right| &\le O(d^{-1})\frac{1}{n}\sum_{i=1}^{n}|y^{(i)}|\|\Delta^{(i)}\|\|\Delta^{(n+1)}\| \\ &\le O(d^2 r^{-4})\frac{1}{n}\sum_{i=1}^{n}|y^{(i)}|. \end{aligned}$$

Therefore,

$$\mathbb{E}[c^2] \le O(d^4 r^{-8})\mathbb{E}[y^2] = O(d^5 r^{-8}\|\mathbf{w}\|^2) \le O(d^2 r^{-4}\|\mathbf{w}\|^2),$$

where we use $r > d^{\frac{3}{4}}$ in the last inequality. Combining the above bounds, we obtain

$$\mathbb{E}[\delta_f^2] = O(d^3 r^{-4}\|\mathbf{w}\|^2).$$

By the Cauchy–Schwarz inequality, Eq. equation 12 gives

$$\begin{aligned} |L_1 - L_2| &\le \mathbb{E}[\delta_f^2] + 2\sqrt{\mathbb{E}[\delta_f^2]}\sqrt{\mathbb{E}\left[\left(f_2(\mathbf{Z})_{[d+1,n+1]} + \mathbf{w}^\top g(\mathbf{x}^{(n+1)})\right)^2\right]} \\ &= \mathbb{E}[\delta_f^2] + 2\sqrt{\mathbb{E}[\delta_f^2]L_2}. \end{aligned}$$

Using

$$\mathbb{E}[\delta_f^2] = O(d^3 r^{-4} \|\mathbf{w}\|^2)$$

and

$$L_2 = O\left(\frac{d^2}{n}\right),$$

we obtain

$$|L_1 - L_2| \le O(d^3 r^{-4} \|\mathbf{w}\|^2) + O\left(d^{5/2} r^{-2} \frac{\|\mathbf{w}\|}{\sqrt{n}}\right).$$

Taking expectation over $\mathbf{w} \sim \mathcal{N}(0, I_d)$, where

$$\mathbb{E}[\|\mathbf{w}\|^2] = d, \qquad \mathbb{E}[\|\mathbf{w}\|] = O(\sqrt{d}),$$

yields

$$\mathbb{E}_{\mathbf{w}}[|L_1 - L_2|] = O(d^4 r^{-4}) + O\left(\frac{d^3 r^{-2}}{\sqrt{n}}\right).$$

Combining all above, we have

$$R_{\mathbf{Z}}(f_0) - R_{\mathbf{Z}}(f_1) \ge \frac{3d^2}{d+2} - O\left(\frac{d^2}{n}\right) - O(d^4 r^{-4}) - O\left(\frac{d^3 r^{-2}}{\sqrt{n}}\right).$$

When $n > d$ and $r \ge d^{3/4}$, the first term is $\Theta(d)$, while all remaining terms are at most $O(d)$. Therefore,

$$R_{\mathbf{Z}}(f_0) - R_{\mathbf{Z}}(f_1) = \Omega(d),$$

which proves the theorem.

□

**Remark 5.** *In Sun et al. (2025), the authors show that, when approximating certain quadratic functions, a deep network consisting only of linear attention layers cannot achieve better accuracy than a linear predictor, whereas a deep network consisting of a bilinear layer composed with attention can substantially improve the approximation capability. Compared with their setting, our analysis omits the post-attention MLP block while quantifying the error induced by the ReLU low-rank branch. This design choice potentially leaves the MLP block in ResLorB available for more complex functionalities, such as key-value memorization or knowledge storage (Geva et al., 2021; Dai et al., 2022).*

# C Experimental Details

This appendix provides detailed model configurations, training hyperparameters, and implementation settings for all experiments.

## C.1 LLaMA-Series Experiments

**Model Configurations.** Table 5 summarizes the model configurations used throughout the LLaMA experiments. Following the Chinchilla scaling law, all models are trained for approximately one data epoch.

Table 5: Model configurations for LLaMA experiments.

| Model | Layers | Hidden | FFN | Heads | Vocab | Tokens |
|---|---|---|---|---|---|---|
| 60M | 8 | 512 | 1376 | 8 | 32K | 1.4B |
| 130M | 12 | 768 | 2048 | 12 | 32K | 2.6B |
| 350M | 24 | 1024 | 2736 | 16 | 32K | 7.8B |

For the baseline configurations, we follow the implementation and hyperparameter settings of LAUREL and NOBLE described in their original papers whenever applicable. Frac-Connection (FC) is

implemented following the mHC-style parameterization, where the interaction matrices are parameterized by sigmoid gates with the scaling parameter $\alpha$ initialized to 0.01. The number of fractions is fixed to 4 throughout all experiments.

For ResLorB, the bottleneck ranks are set to 64, 48, and 64 for the 60M, 130M, and 350M models, respectively, corresponding to approximately 1–2% additional parameters. The low-rank matrices are initialized using the same Gaussian initialization as the other trainable parameters in the model.

**Training Hyperparameters** Unless otherwise specified, all LLaMA models are trained using AdamW with the default $\beta_1$ and $\beta_2$ values in PyTorch, a weight decay of 0.1, and bfloat16 mixed precision. Weight decay is not applied to normalization layers or bias parameters. The global batch size is set to 512 and the sequence length is 256.

We sweep the peak learning rate over the following search grid:

$$\left\{10^{-3}, 1.5 \times 10^{-3}, 2 \times 10^{-3}, 2.5 \times 10^{-3}, 3 \times 10^{-3}\right\}.$$

The peak learning rates are set to $3 \times 10^{-3}$, $1.5 \times 10^{-3}$, and $1 \times 10^{-3}$ for the 60M, 130M, and 350M models, respectively. We adopt a learning-rate schedule consisting of 10% linear warmup followed by cosine decay to $10^{-4}$ of the peak learning rate. No gradient clipping or dropout is utilized throughout training.

Our implementation is based on the official GaLore (Zhao et al., 2024) codebase with minor fixes to the validation pipeline and total batch-size computation.

### C.2 DeepSeek-V3-Series Experiments

**Model Configurations** We adopt the 3B DeepSeek-V3-style architecture following the configuration described in Table 5 in Xie et al. (2025). The model consists of a Mixture-of-Experts architecture with the same hidden dimension, attention configuration, MLA (Multi-head Latent Attention) components, and expert configurations as the original setting. Unless otherwise specified, all architectural components follow the original implementation.

For completeness, the main configurations relevant to our experiments are summarized as follows. The model uses a hidden dimension of 1280, 16 attention heads, 64 experts with top-6 routing, and one MTP layer. The MLA module uses a Q-LoRA rank of 384, a KV-LoRA rank of 512, and QK/V head dimensions of 128. For Sandwich-Norm experiments, the output normalization scale is initialized to 0.03.

For experiments with mHC, we follow the official implementation with four interaction streams and initialize the interaction scaling parameter $\alpha$ to 0.01. For ResLorB, the bottleneck rank is set to 512 throughout all DeepSeek-3B experiments. The low-rank matrices are initialized using the same Gaussian initialization as the Transformer parameters.

**Training Hyperparameters** The 3B model is trained on a mixed dataset based on Dolma 3 (Olmo et al., 2025) using AdamW with $\beta_1 = 0.9$ and $\beta_2 = 0.95$, a weight decay of 0.1, and gradient clipping with a maximum norm of 1.0. Weight decay is disabled for normalization layers and bias parameters. Training is performed with a mixed-precision strategy using bfloat16 for the majority of compute-intensive operations, while keeping numerically sensitive operations in float32. The global batch size for pretraining is 1024 and the sequence length is 4096.

We adopt the WSD learning-rate schedule with a peak learning rate of $8.6 \times 10^{-4}$ and a minimum learning rate of $7 \times 10^{-6}$. The learning rate is linearly warmed up over the first 8.3B tokens and decays during the final 50B tokens.

### C.3 Evaluation Protocol

For downstream evaluation, we evaluate pretrained checkpoints on seven representative language understanding benchmarks, including ARC-Challenge, BBH, DROP, HellaSwag, MMLU, PIQA, and TriviaQA. All evaluations are conducted using the `lm-evaluation-harness` (Gao et al., 2023) framework with the same evaluation protocol across all models.

Table 6: Ablation study of different Frac-Connection implementations and initialization strategies on LLaMA models.

| Method | 60M Loss | 130M Loss |
|---|---|---|
| Original FC | 3.392 | 3.161 |
| mHC-style FC (No Tune) | 3.437 | 3.207 |
| mHC-style FC, $\mathbf{H}^{res} = \mathbf{I}$ (No Tune) | 3.368 | 3.137 |
| mHC-style FC (Tuned Init) | 3.343 | **3.116** |
| mHC-style FC, $\mathbf{H}^{res} = \mathbf{I}$ (Tuned Init) | **3.338** | 3.118 |

Following standard practice, we use zero-shot evaluation for ARC-Challenge and PIQA, while applying few-shot prompting for the remaining tasks. Specifically, we use 3-shot evaluation for BBH and DROP, 10-shot evaluation for HellaSwag, and 5-shot evaluation for MMLU and TriviaQA. We report normalized accuracy for ARC-Challenge, HellaSwag, and PIQA, exact match for BBH, F1 score for DROP, accuracy for MMLU, and exact match after whitespace normalization for TriviaQA.

### C.4 Implementation Details

**Frac-Connection** For all experiments involving Frac-Connection (FC), we adopt an implementation based on the mHC formulation (Xie et al., 2025), which provides a more general parameterization of the interaction matrices. Specifically, following Eq. (7) of mHC, given the reshaped input $\hat{\mathbf{S}}_l \in \mathbb{R}^{n \times d/n}$, we first compute $\hat{\mathbf{s}}'_l = \mathrm{RMSNorm}(\mathrm{flatten}(\hat{\mathbf{S}}_l)) \in \mathbb{R}^{1 \times d}$. The interaction matrices are then parameterized as

$$\begin{aligned}
\hat{\mathbf{H}}_l^{pre} &= \sigma_s(\hat{\alpha}_l^{pre}(\hat{\mathbf{s}}'_l \varphi_l^{pre}) + \mathbf{b}_l^{pre}), \\
\hat{\mathbf{H}}_l^{post} &= 2\sigma_s(\hat{\alpha}_l^{post}(\hat{\mathbf{s}}'_l \varphi_l^{post}) + \mathbf{b}_l^{post}), \\
\hat{\mathbf{H}}_l^{res} &= \mathrm{SK}(\hat{\alpha}_l^{res}\mathrm{mat}(\hat{\mathbf{s}}'_l \varphi_l^{res}) + \mathbf{b}_l^{res}),
\end{aligned}$$

where $\varphi_l^{post} \in \mathbb{R}^{d \times n}, \varphi_l^{pre}, \varphi_l^{res} \in \mathbb{R}^{d \times n^2}$, $\sigma_s(\cdot)$ denotes the sigmoid function and $\mathrm{SK}(\cdot)$ denotes the Sinkhorn-Knopp normalization.

Recent observations suggest that directly using an identity matrix for the residual interaction component can be more effective than enforcing a doubly stochastic constraint (Su, 2025). Therefore, we evaluate several FC variants, including the original FC formulation, the full mHC-style FC, and the mHC-style FC with $\hat{\mathbf{H}}_l^{res} = \mathbf{I}$. We further investigate different initialization strategies for the interaction matrices. The corresponding results on LLaMA-60M and LLaMA-130M are summarized in Table 6. Specifically, $\varphi^{pre}$, $\varphi^{post}$, and $\varphi^{res}$ are initialized to 0. In the "no tuning" setting, all learnable parameters $\mathbf{b}^{pre}, \mathbf{b}^{post}$ and $\mathbf{b}^{res}$ are initialized to 0, while in the "init tuning" setting, we initialize $\hat{\mathbf{H}}^{pre}$ close to identity matrices by setting the diagonal elements of $\mathbf{b}^{pre}$ to 4 and the off-diagonal elements to $-4$, resulting in $\sigma(4) \approx 0.98$.

As shown in Table 6, the mHC-style FC with identity residual interaction and tuned initialization achieves the near-best performance among all variants. Therefore, we use this configuration for all FC-based experiments reported in the main paper.

Following the default configuration in the original FC paper (Zhu et al., 2025b), we use 4 streams as the default setting for all FC experiments. We additionally conduct a preliminary comparison between 4-stream and 8-stream FC on a DeepSeek-V3-style 3B model under the Pre-Norm setting. As shown in Fig. 7, increasing the number of streams from 4 to 8 yields nearly identical LM loss, suggesting limited benefits from further increasing the number of streams. This observation is also consistent with the trend reported in Fig.4 of Zhu et al. (2025b), where increasing the number of streams from 2 to 4 provides only marginal additional gains. Therefore, we use 4 streams as the default configuration for FC throughout the paper. These results suggest that the performance advantage of ResLorB over FC is unlikely to arise simply from introducing more parameters, and is more plausibly related to the difference in their interaction mechanisms: FC relies on a fixed coordinate-wise decomposition that restricts which dimensions can interact, whereas ResLorB enables learnable neuron-level mixing through a trainable low-rank transformation.

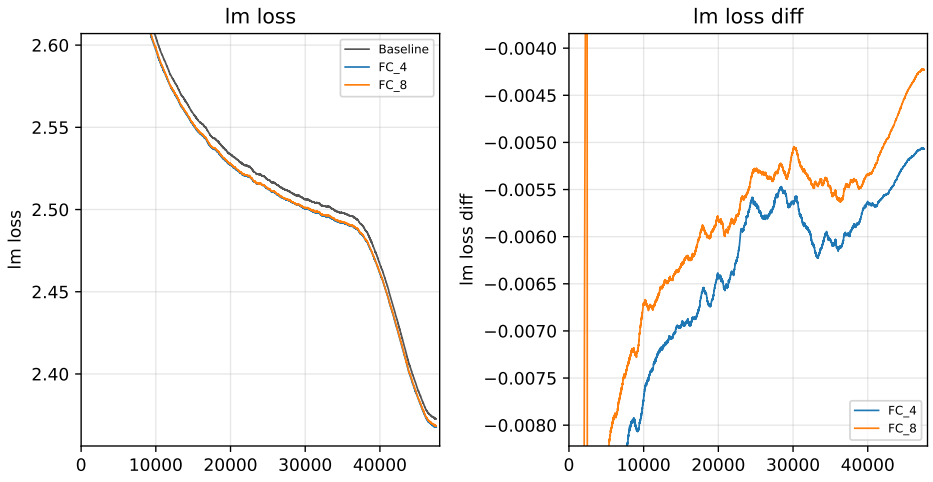



Figure 7: Comparison of 4-stream and 8-stream Frac-Connection on the DeepSeek-V3-style 3B model under the Pre-Norm setting. Increasing the number of streams from 4 to 8 provides no additional improvement in LM loss.

**Low-Rank Branch Variants.** In addition to the default ReLU-based pre-block ResLorB, we investigate several alternative designs of the low-rank branch, including different nonlinear transformations and branch placements in Section 5.4.

Recall that the default ResLorB used throughout the paper applies a nonlinear low-rank transformation before each Transformer block:

$$\tilde{\mathbf{x}}_l = \mathbf{x}_l + \mathbf{W}_l^2 \sigma(\mathbf{W}_l^1 \mathbf{x}_l),$$
$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}_l(\tilde{\mathbf{x}}_l),$$

where $\mathbf{W}_l^1 \in \mathbb{R}^{r\times d}$, $\mathbf{W}_l^2 \in \mathbb{R}^{d\times r}$, and $\sigma(\cdot) = \mathrm{ReLU}(\cdot)$. We also consider a SwiGLU-based low-rank branch with a matched parameter budget:

$$\tilde{\mathbf{x}}_l = \mathbf{x}_l + \mathbf{W}_l^{down}\left((\mathbf{W}_l^{gate}\mathbf{x}_l) \odot (\mathbf{W}_l^{up}\mathbf{x}_l)\right),$$

where $\mathbf{W}_l^{gate}, \mathbf{W}_l^{up} \in \mathbb{R}^{r'\times d}$, $\mathbf{W}_l^{down} \in \mathbb{R}^{d\times r'}$, and $r' = \lfloor 2r/3 \rfloor$ is chosen to approximately match the parameter count of the ReLU branch.

Beyond the default pre-block placement, we further evaluate two alternative branch locations. For the post-block variant, the low-rank transformation is applied after the Transformer block output:

$$\mathbf{y}_l = \mathcal{F}_l(\mathbf{x}_l),$$
$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathbf{y}_l + \mathbf{W}_l^2 \sigma(\mathbf{W}_l^1 \mathbf{y}_l).$$

For the pre- and post-block variant, both branches are applied simultaneously:

$$\tilde{\mathbf{x}}_l = \mathbf{x}_l + \mathbf{W}_l^{pre,2} \sigma(\mathbf{W}_l^{pre,1} \mathbf{x}_l),$$
$$\mathbf{y}_l = \mathcal{F}_l(\tilde{\mathbf{x}}_l),$$
$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathbf{y}_l + \mathbf{W}_l^{post,2} \sigma(\mathbf{W}_l^{post,1} \mathbf{y}_l).$$

For fair comparison, the total parameter budget of different placement variants is matched by adjusting the branch rank.

# D Additional Experiments

## D.1 Downstream Results

To further evaluate the compatibility of ResLorB with existing interaction mechanisms, we report downstream performance of the combined architectures on DeepSeek-V3-style 3B model, including

Table 7: Downstream performance (%) of combined architectures on the DeepSeek-V3-style 3B model. Higher is better.

| Norm | Method | BBH | DROP | HellaSwag | MMLU | PIQA | TriviaQA | ARC-C | AVG. |
|---|---|---|---|---|---|---|---|---|---|
| Pre-Norm | FC | 25.73 | 17.25 | 63.05 | 25.20 | 74.81 | 35.36 | 39.16 | 40.08 |
| | FC-ResLorB | 25.36 | **17.36** | 64.02 | **27.31** | **75.68** | **37.02** | 39.51 | 40.89 |
| | FC-ResLorB+ | **27.29** | 17.09 | **64.22** | 26.48 | 74.54 | 35.80 | **41.13** | **40.94** |
| Sandwich-Norm | mHC | 27.02 | 15.74 | 63.59 | 25.99 | 74.48 | 36.74 | 36.35 | 39.99 |
| | mHC-ResLorB | **28.94** | 14.33 | 64.56 | **26.57** | **75.24** | **38.34** | 37.88 | 40.84 |
| | mHC-ResLorB+ | 27.37 | **15.79** | **64.76** | 25.94 | 74.48 | 38.06 | **41.81** | **41.17** |

FC-ResLorB, FC-ResLorB+, mHC-ResLorB, mHC-ResLorB+ variants under their corresponding normalization settings. As shown in Table 7, the improvements observed during pretraining consistently translate to downstream tasks, demonstrating that ResLorB provides complementary benefits when combined with both intra-state and inter-state architectures.

### D.2 Further Scaling

To further evaluate the scalability of ResLorB, we conduct experiments on a larger DeepSeek-V3-style 9B model trained with 450B tokens. Due to computational constraints, we focus on the combination of ResLorB with mHC and compare it against the mHC baseline. The model configuration follows Table 5 in Xie et al. (2025). Based on this configuration, we fix the residual interaction matrix $\mathbf{H}^{res}$ in mHC as the identity matrix and adopt the Sandwich-Norm setting. The post-norm scale parameter is initialized with 0.003. For ResLorB, we employ a SwiGLU activation function in the low-rank branch and set the rank to 768, resulting in only approximately 1% additional parameters.

For optimization, we adopt the Kimi-style Muon optimizer. Following common practice, embedding parameters and parameters with dimension one are optimized using AdamW. To improve the optimization stability of the newly introduced low-rank parameters, the weights of the ResLorB branch are also optimized using AdamW instead of Muon. Other optimizer configurations, including $\beta$ coefficients and weight decay, as well as the mixed-precision strategy, follow the settings described in Appendix C.2. We train the models with a global batch size of 2048 and a sequence length of 4096 tokens. The learning rate follows a WSD schedule with a peak learning rate of $5.9 \times 10^{-4}$ and a minimum learning rate of $7 \times 10^{-6}$. Specifically, the learning rate is linearly warmed up over the first 16.7B tokens, followed by a stable phase and a cosine decay phase over the final 50B tokens.

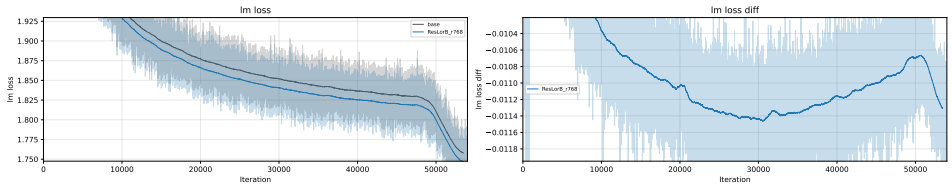
The training curves are shown in Figure 8. Under this larger-scale training setting, mHC-ResLorB consistently outperforms the mHC baseline, achieving a reduction of 0.011 in training LM loss and approximately 0.02 in MTP-1 loss. These results demonstrate that the benefits of ResLorB are preserved beyond the 3B-scale regime, validating its scalability under realistic large-scale pretraining settings.

We further evaluate the pretrained models on downstream benchmarks at the end of training. As shown in Table 8, mHC-ResLorB improves the average performance from 44.1 to 45.5 across the ten recommended benchmarks, corresponding to a 1.4 percentage-point gain. ResLorB improves performance on 8 out of the 10 benchmarks, with particularly clear gains on MBPP, GSM8K, and MATH. These downstream results further show that the improvements observed in training loss translate into consistent gains across a diverse set of language understanding, reasoning, and code-generation tasks.

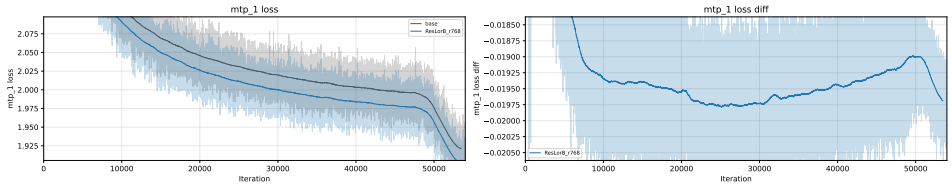
### D.3 Training Dynamics

**Training Stability.** We further analyze the optimization dynamics of ResLorB by tracking the gradient norm throughout pretraining. Fig. 9 compares ResLorB with the baseline model under both Pre-Norm and Sandwich-Norm settings.

Under the Sandwich-Norm configuration, ResLorB consistently exhibits smaller gradient norms than the baseline, suggesting a more stable optimization trajectory. Under the Pre-Norm configuration, the gradient norms also remain smaller compared with the baseline, although occasional moderate spikes appear during training. These observations indicate that ResLorB does not intro-




(a) Training LM loss and loss difference




(b) MTP-1 loss and loss difference

Figure 8: Training dynamics of mHC-ResLorB and mHC on the DeepSeek-V3-style 9B model trained with 450B tokens. Each subfigure contains the original loss curves (left) and the corresponding loss difference relative to the mHC baseline (right). mHC-ResLorB consistently achieves lower LM loss and MTP-1 loss throughout training.

Table 8: Downstream evaluation of the 9B models at different training stages. Results are reported on the ten recommended benchmarks.

| **Benchmark** | **mHC** | **mHC-ResLorB** |
|---|---|---|
| TriviaQA | **51.71** | 51.41 |
| MBPP | 42.00 | **48.20** |
| HellaSwag | **72.42** | 72.05 |
| LAMBADA | 72.39 | **72.52** |
| DROP | 23.43 | **23.98** |
| ARC-Challenge | 48.98 | **50.34** |
| GSM8K | 26.76 | **28.96** |
| MMLU | 49.34 | **50.27** |
| MATH | 13.78 | **15.96** |
| BBH | 39.99 | **41.50** |
| **Average** | 44.1 | **45.5** |

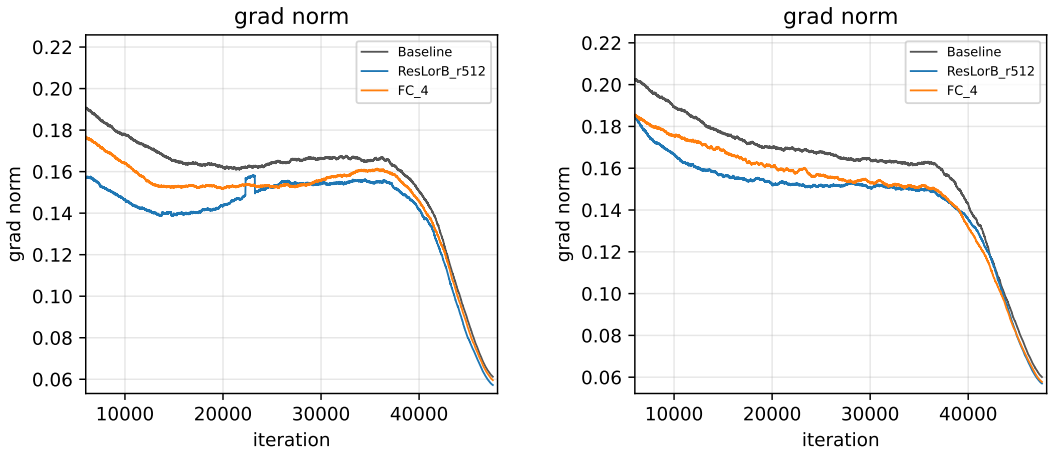
duce significant optimization instability, while Sandwich-Norm provides a particularly favorable regime for training ResLorB.

**Load Balancing.** Since ResLorB modifies the block input representation, we further investigate whether the additional low-rank transformation affects the routing behavior of MoE layers. We analyze two commonly used load balancing metrics on the DeepSeek-V3-style 3B model: sequence-level load balancing loss and maximum routing violation.

Following DeepSeek-V3 (Liu et al., 2024a), the sequence-level load balancing loss is defined as

$$L_{\text{seq}} = \alpha N_r \sum_{i=1}^{N_r} f_i P_i, \tag{13}$$

where $N_r$ denotes the total number of routed experts, $f_i$ denotes the normalized routing frequency of expert $i$ within a sequence and $P_i$ represents the average normalized routing score. We further measure the maximum routing violation (MaxVio) (Wang et al., 2024), which quantifies the largest




(a) Pre-Norm (b) Sandwich-Norm

Figure 9: Gradient norm during training under different normalization schemes. ResLorB maintains comparable or smaller gradient norms compared with the baseline, indicating stable optimization behavior under both Pre-Norm and Sandwich-Norm settings.

deviation of expert load from the ideal balanced allocation:

$$\text{MaxVio} = \max_i \left| \frac{N_r c_i}{K_r T} - 1 \right|, \tag{14}$$

where $K_r$ is the number of experts each token is routed to, $T$ is the total number of tokens, where $c_i$ denotes the number of tokens routed to expert $i$.

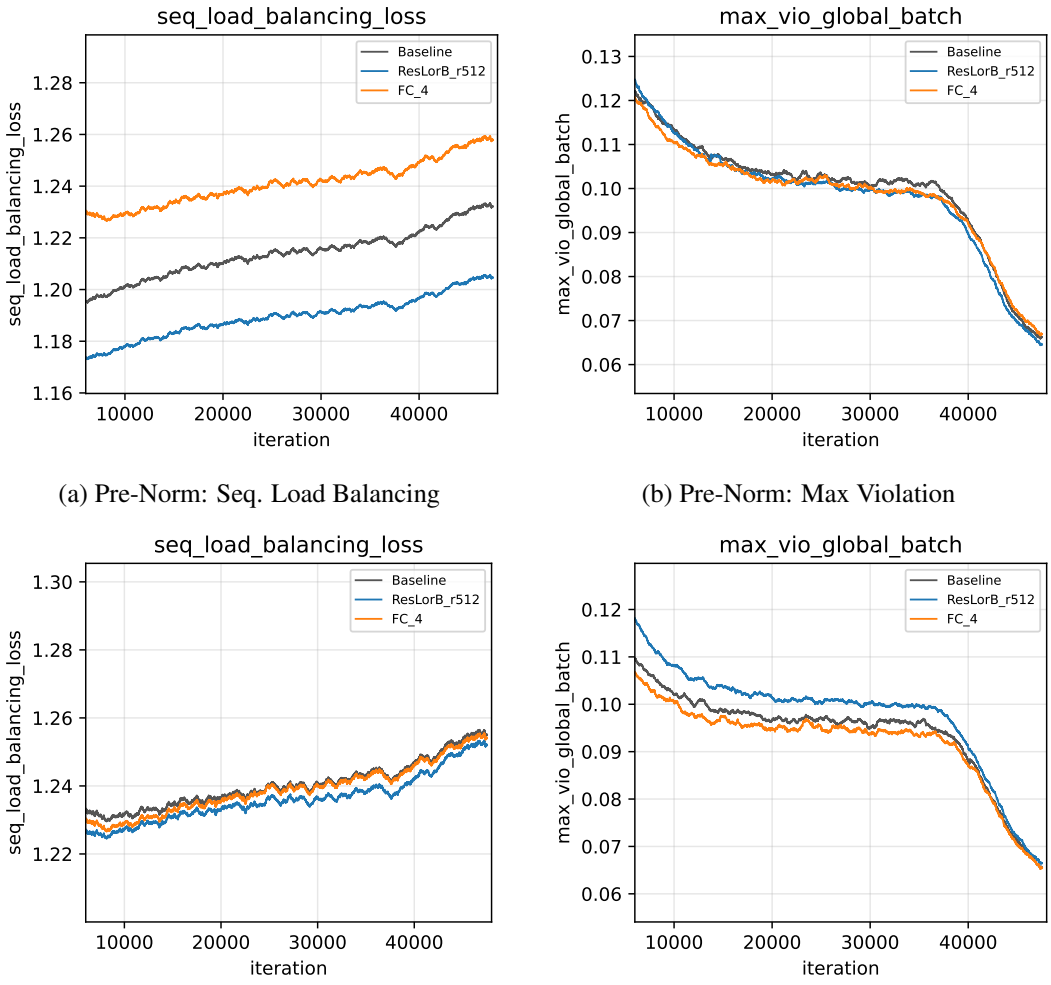
Fig. 10 presents the training dynamics of these metrics under different normalization schemes and architectural settings. Across both Pre-Norm and Sandwich-Norm configurations, ResLorB exhibits nearly identical load balancing behavior compared with the baseline model, with negligible differences in both metrics throughout training. These results indicate that the low-rank branch does not introduce additional routing imbalance and preserves stable MoE routing behavior.

### D.4 More Analysis

**Adjacent-Layer Cosine Similarity.** We further examine the median cosine similarity between the representations at consecutive Transformer blocks. Fig. 11 compares ResLorB with the baseline under Pre-Norm and Sandwich-Norm settings. Under Pre-Norm, ResLorB generally exhibits lower cosine similarity across layers, indicating less redundant transformations between consecutive blocks. Under Sandwich-Norm, the effect is less uniform, with the cosine similarity varying across layers. This suggests that the normalization scheme can have a more complex influence on inter-layer representation similarity, while ResLorB generally maintains a comparable level of representation similarity to the baseline.

**Residual Sink.** The low-rank branch in ResLorB structurally resembles a two-layer additive gate (Qiu et al., 2026c), which motivates us to further examine its effect on residual sinks, following the analysis of Qiu et al. (2026a). We consider two complementary statistics: (1) the mean absolute activation of each hidden dimension across all tokens and layers, and (2) the maximum absolute activation within each layer.

Fig. 12 compares the top-20 dimensions ranked by their mean absolute activation between ResLorB and the baseline. The results are shown under both Sandwich-Norm and Pre-Norm. ResLorB consistently reduces the magnitude of the most dominant dimension, while dimensions in the middle ranks tend to exhibit slightly larger magnitudes and those in the lower ranks tend to exhibit smaller magnitudes than the baseline. This suggests a more balanced distribution of activation magnitude




(a) Pre-Norm: Seq. Load Balancing

(b) Pre-Norm: Max Violation

(c) Sandwich-Norm: Seq. Load Balancing

(d) Sandwich-Norm: Max Violation

Figure 10: MoE load balancing behavior of ResLorB and the baseline model under different normalization schemes. ResLorB closely follows the baseline in both sequence-level load balancing loss and maximum routing violation, indicating that the low-rank branch does not interfere with expert routing behavior.

across dimensions, which may be related to the finer-grained neuron-level interactions introduced by the low-rank branch.

We further examine the maximum absolute activation at each layer in Fig. 13. For the 12-layer DeepSeek-V3-style 3B model, ResLorB generally exhibits smaller maximum activations than the baseline across most layers, with the main exceptions occurring at the deepest layers. In particular, the largest activation is observed at the final layer, where ResLorB produces a larger maximum than the baseline. Nevertheless, the mean maximum activation across layers is lower with ResLorB, suggesting an overall tendency to alleviate residual sinks, while potentially introducing outlier activations in specific deep layers.

### D.5 More Ablations

**Architecture and Parameter Allocation Alternatives.** We further investigate alternative architectural designs and parameter allocation strategies to better understand the source of ResLorB's gains. Specifically, we compare ResLorB with four variants in the DeepSeek-V3-style 3B setting. The first two are alternative formulations of the low-rank branch: (1) a direct residual injection

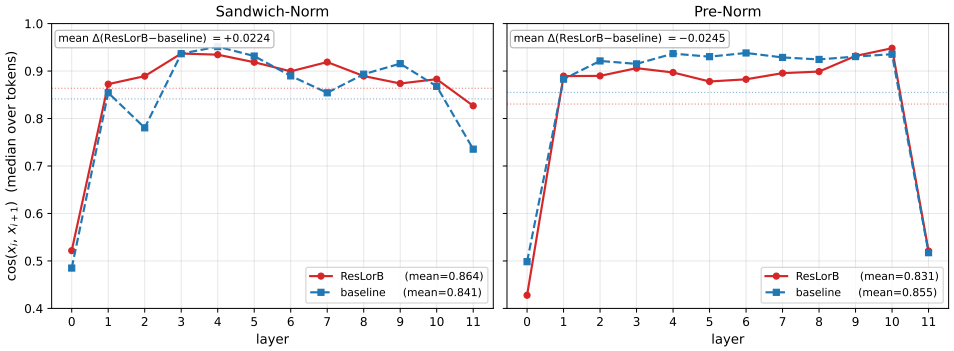



Figure 11: Median cosine similarity between the input and output representations of consecutive Transformer blocks. The left and right panels correspond to Sandwich-Norm (left) and Pre-Norm (right), respectively. Each panel compares ResLorB with the baseline. Under Pre-Norm, ResLorB generally exhibits lower cosine similarity across layers, while the effect is less uniform under Sandwich-Norm.

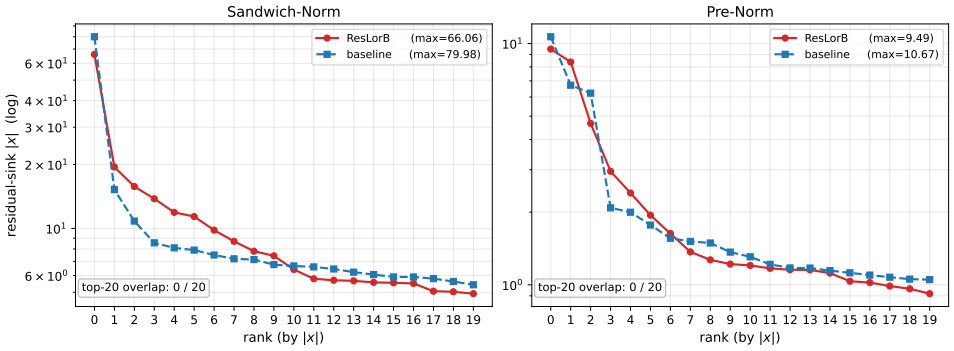



Figure 12: Mean absolute activation of the top-20 hidden dimensions ranked by their average magnitude across tokens and layers under Sandwich-Norm (left) and Pre-Norm (right). ResLorB reduces the magnitude of the most dominant dimensions while redistributing activation magnitude toward middle-ranked dimensions.

variant that explicitly adds the low-rank transformation to the residual stream,

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathbf{W}_l^2\sigma(\mathbf{W}_l^1\mathbf{x}_l) + \mathcal{F}_l\left(\mathbf{x}_l + \mathbf{W}_l^2\sigma(\mathbf{W}_l^1\mathbf{x}_l)\right), \tag{15}$$

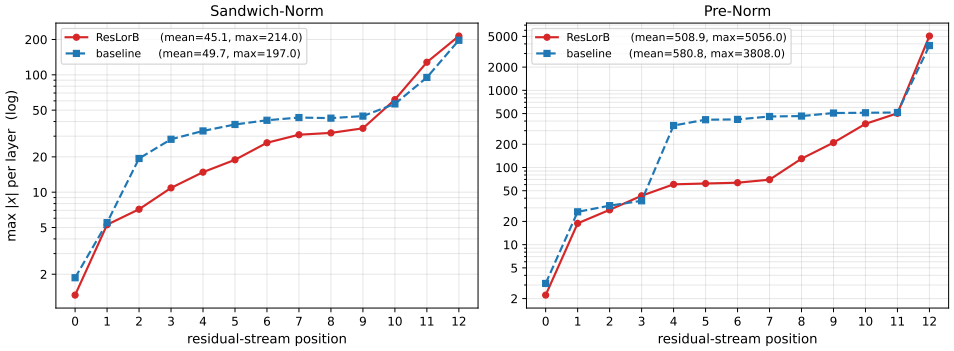



Figure 13: Maximum absolute activation at each Transformer layer under Sandwich-Norm (left) and Pre-Norm (right). ResLorB generally exhibits smaller maximum activations across layers, while larger outlier activations may still occur at the deepest layers.

and (2) a gated input modulation variant:

$$\tilde{\mathbf{x}}_l = \sigma_s(\mathbf{x}_l \mathbf{W}_g) \odot \mathbf{x}_l. \tag{16}$$

The remaining two variants serve as parameter allocation baselines: (3) activating eight experts instead of six in each MoE layer, and (4) adding one additional Transformer layer including an attention and an MoE block, increasing the depth from 12 to 13 layers.

Fig. 14 compares these variants under both Pre-Norm and Sandwich-Norm settings. Across both normalization schemes, the proposed ResLorB consistently achieves the best performance.

For the architectural alternatives, directly injecting the low-rank branch into the residual stream performs similarly to ResLorB under Pre-Norm, but exhibits a much larger gap under Sandwich-Norm. One possible explanation is that, under Sandwich-Norm, explicitly adding the low-rank branch to the residual stream effectively changes the depth and scaling behavior of each layer, and may therefore require retuning of the corresponding training configuration, such as the learning rate. In contrast, ResLorB preserves the original residual stream and introduces the low-rank transformation only to the block input, allowing the original model configuration to be retained. The gated input modulation variant performs substantially worse than ResLorB, despite introducing a comparable or larger number of parameters. This suggests that the neuron-level interactions enabled by the nonlinear low-rank transformation are more important than feature-wise scaling alone.

We next compare ResLorB with alternative ways of allocating additional model capacity. ResLorB slightly outperforms adding one additional Transformer layer, while consistently outperforming the eight-expert variant by a larger margin. Importantly, ResLorB introduces only approximately $1\%$ additional total parameters and $4\%$ additional activated parameters (see Table 3), compared with approximately $7\%$ additional total parameters (3.41B) and $5\%$ additional activated parameters (0.783B) for adding one Transformer layer. Moreover, increasing the model depth also introduces additional KV-cache overhead during autoregressive inference, whereas ResLorB does not increase KV-cache. Activating eight experts does not increase the total parameter count, but increases the number of activated parameters by approximately $10\%$ (0.823B). These results suggest that ResLorB provides a more parameter-efficient way of allocating additional model capacity than simply increasing model depth or the number of activated experts.

**Comparisons with Gated Residual.** We also compare ResLorB with the contemporary Gated Residual (GR) introduced in Qwen3.8-Flash-Next (Qiu et al., 2026b), which similarly introduces a low-rank gating mechanism at the read and write operations of mHC. Following the notation of Eq. (2), GR first normalizes each stream independently with its own gain $\boldsymbol{\gamma}_i \in \mathbb{R}^d$, and predicts an *elementwise* gate from all streams through a low-rank bottleneck:

$$\hat{\mathbf{S}}_{l,i} = \mathrm{RMSNorm}(\mathbf{S}_{l,i}; \boldsymbol{\gamma}_i), \quad i = 1, \dots, m, \tag{17}$$

$$\mathbf{G}_l = \mathrm{unvec}\Big(\sigma_s\big(\mathbf{W}_l^u \,\mathrm{SiLU}\big(\tfrac{1}{m}\mathbf{W}_l^d \,\mathrm{vec}(\hat{\mathbf{S}}_l)\big)\big)\Big) \in \mathbb{R}^{m \times d}, \tag{18}$$

where $\mathbf{W}_l^d \in \mathbb{R}^{r_g \times md}$ and $\mathbf{W}_l^u \in \mathbb{R}^{md \times r_g}$ form the bottleneck of rank $r_g = d/8$, and $\mathrm{vec}(\cdot)$ stacks the $m$ streams into a vector of length $md$. The block input is the average of the gated streams,

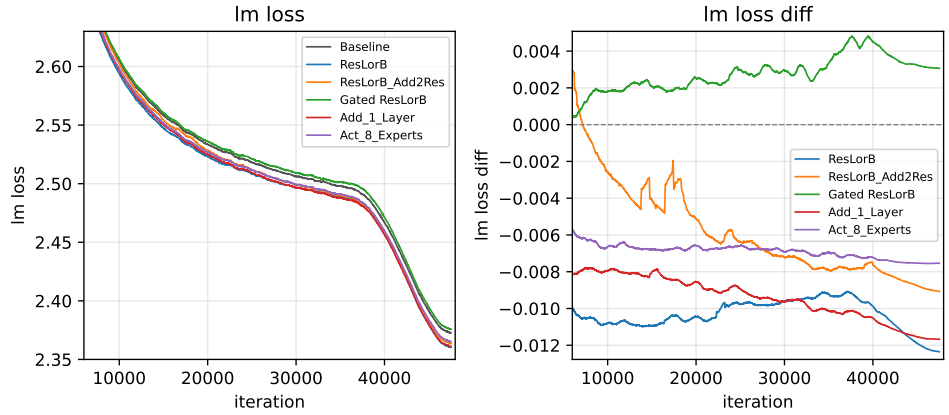
$$\mathbf{x}_l = \frac{1}{m} \sum_{i=1}^{m} \mathbf{G}_{l,i} \odot \hat{\mathbf{S}}_{l,i}, \tag{19}$$

which generalizes the read operator $\mathbf{H}_l^{pre}$ from a per-stream scalar to an elementwise gate. The write operator is also data-dependent, but remains a per-stream scalar,

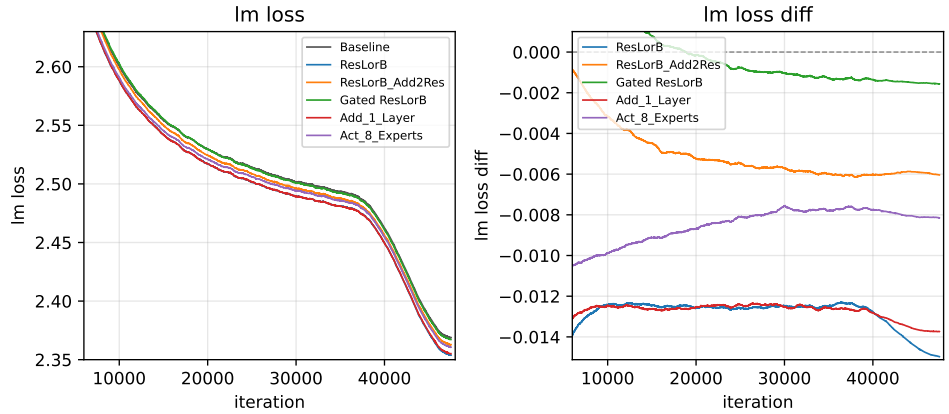
$$\mathbf{s}_l = 2\sigma_s\Big(\frac{1}{m}\mathbf{W}_l^w \,\mathrm{vec}(\hat{\mathbf{S}}_l)\Big) \in \mathbb{R}^{1 \times m}, \qquad \mathbf{S}_{l+1} = \mathbf{S}_l + \mathbf{s}_l^\top \mathcal{F}_l(\mathbf{x}_l), \tag{20}$$

with $\mathbf{W}_l^w \in \mathbb{R}^{m \times md}$. Finally, the mixing matrix is removed altogether by setting $\mathbf{H}_l^{res} = \mathbf{I}$. Since the read in Eq. (19) already performs normalization and gating, the standalone pre-normalization of the Transformer block is removed as well. GR also discards the static terms used in HC/mHC, requires no special initialization, and is instantiated separately for the attention and MLP sub-blocks of each layer.

We conduct the comparison under the DeepSeek-V3-style 3B setting. Following the configuration in (Qiu et al., 2026b), we set the rank of GR to $d/8$, while using rank $512$ for ResLorB. Therefore, GR




(a) Pre-Norm



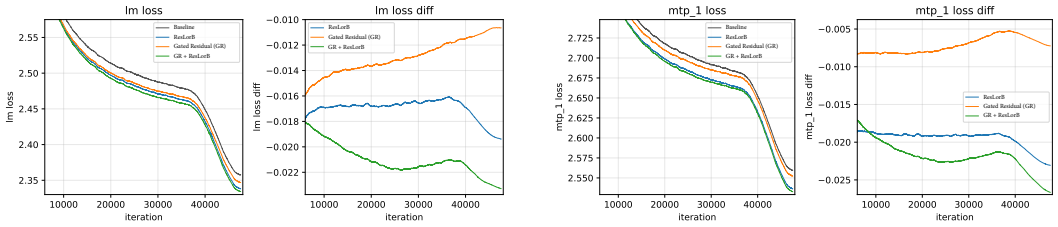

(b) Sandwich-Norm

Figure 14: Comparison of alternative architectural designs and parameter allocation strategies under different normalization schemes. ResLorB consistently achieves the best performance across all comparisons.

introduces slightly more additional parameters than ResLorB in this setting. Following the settings of GR, we set $\mathbf{H}^{res}$ to be identity matrix for the baseline and also GR/ResLorB. We additionally evaluate a combined variant by incorporating both GR and ResLorB.

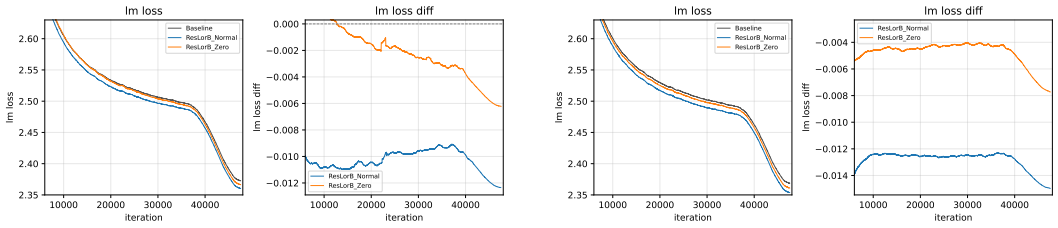
As shown in Figure 15, ResLorB consistently achieves lower training LM loss and MTP-1 loss compared with GR. Combining GR with ResLorB further improves performance, although the additional gain over ResLorB alone is relatively limited. This may be partially attributed to the normalization placement in GR: by folding the pre-block normalization into the read operation, GR removes the standalone Pre-Norm of the Transformer block. This change may affect the stability of the block input representation when combining with ResLorB. Overall, our experiments suggest that an additive low-rank branch at the block input is more effective than the multiplicative gate adopted by GR, and the two remain complementary.

**Initialization.** We further study the effect of initialization for the low-rank branch. Following the common practice in parameter-efficient fine-tuning methods such as LoRA, one natural choice is to initialize the branch output to zero, which allows the model to start from the original architecture. We compare this strategy with the default initialization, where the low-rank matrices are initialized in the same way as other trainable weights.




(a) Training LM loss (b) Training MTP-1 loss

Figure 15: Comparison between ResLorB and Gated Residual (GR) under the DeepSeek-V3-style 3B setting. ResLorB consistently achieves lower training LM loss and MTP-1 loss, while combining GR with ResLorB provides additional improvements.




(a) Pre-Norm (b) Sandwich-Norm

Figure 16: Effect of initialization strategies for the low-rank branch. Default weight initialization consistently outperforms zero initialization under both normalization schemes.

Fig. 16 shows the training curves under both normalization schemes. The default initialization consistently achieves better performance than zero initialization. This suggests that, unlike parameter-efficient adaptation settings where preserving the pretrained model behavior is desirable, allowing the low-rank branch to participate from the beginning is more beneficial during large-scale pretraining.

## E Limitations and Future Directions

First, although ResLorB introduces only a small number of additional total and activated parameters in our experimental settings, we have not yet extensively evaluated its behavior on extremely large-scale MoE models with more aggressive sparsity. In such settings, the balance between total parameters and activated parameters may become a more important design consideration, and the optimal rank configuration of the low-rank branch may require further investigation. One potential direction is to introduce sparse activation selection strategies for the low-rank branch, similar to expert routing in MoE models, to further reduce the activated parameter overhead. Moreover, since the low-rank branch contains substantially fewer parameters than typical MoE experts, it may not require full expert parallelism. These directions may further improve the scalability of ResLorB for large-scale sparse architectures. Moreover, the additional activated parameters introduced by ResLorB remain comparable to or smaller than commonly used gating mechanisms, suggesting that its overhead can be effectively controlled.

Second, this paper provides a preliminary investigation of how the neuron-level interaction introduced by ResLorB improves representation capacity and model performance. Our empirical analyses, including adjacent-layer cosine similarity (Fig. 11) and maximum absolute activation (Fig. 13), reveal several interesting behaviors but are also affected by factors such as normalization schemes and layer depth. A more comprehensive mechanistic understanding of how nonlinear low-rank branches reshape residual representations remains an interesting direction for future work.