RADFREE: GENERAL DEPTH-AWARE ORTHOGONAL RESIDUAL WRITES FOR LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard residual connections in LLMs add entire sublayer outputs to the residual stream. We show that the radial components of these output parallel to the residual state contribute little to language modeling in deep layers, yet drive substantial residual growth. We introduce Radial-Free Residual Write (RadFree), a simple update that selectively removes radial components in late layers using a detached projection reference while preserving perpendicular updates. We evaluate RadFree at frontier scale through modern MoE pre-training up to 125B total parameters. RadFree consistently lowers validation loss and improves performance on nearly all downstream benchmarks, raising average accuracy by approximately 1.5 percentage points across scales. Combined with manifold-constrained hyper-connections (mHC), RadFree improves average accuracy by 2.93 percentage points over the standard residual baseline, more than doubling the 1.29-point gain of mHC alone. RadFree also makes residual growth in deep layers more gradual, reducing final residual RMS by 36–37% across scales. RadFree adds no parameters and achieves zero training and inference overhead through simple kernel fusion, offering a simple and practical way to improve LLM performance.

1 INTRODUCTION

Residual connections (He et al., 2016) are fundamental for representation learning and depth scaling in modern LLMs (Vaswani et al., 2017; Radford et al., 2019). Recent work regulates residual magnitude through normalization (Loshchilov et al., 2025; Kim et al., 2025). Team et al. (2026b) aggregate earlier representations into sublayer inputs, while Zhu et al. (2025); Xie et al. (2025) expand residual capacity through multiple streams. These methods improve how sublayers read residual information and how residual states are maintained, but do not address which components of a sublayer output should be written back. Should every component be added to the residual stream at every depth?

To answer this question, we adopt the geometric perspective of Oh et al. (2026), decomposing each sublayer output into a radial component and a perpendicular component and examining their roles in language modeling. We find that perpendicular components account for nearly all of a residual write's contribution to feature representation and language modeling. In middle and later layers, radial components can be removed with negligible impact on model quality. Yet these radial components drive exponential growth in residual magnitude in deep layers. Such growth can destabilize training and hinder optimization (Kim et al., 2025; Loshchilov et al., 2025; Kosson et al., 2024).

Motivated by these observations, we introduce Radial-Free Residual Write (RadFree), which selectively removes radial components from residual writes in late layers. RadFree projects sublayer outputs onto the orthogonal complement of the current residual state. This constrains residual writes while leaving raw sublayer outputs unconstrained. For multi-stream architectures, RadFree projects outputs onto the orthogonal complement of the streams' linear span, leaving stream mixing to the connection module and naturally complementing Hyper-Connections (Zhu et al., 2025) and mHC (Xie et al., 2025).

We evaluate RadFree at frontier scale by pre-training modern MoE LLMs with up to 125B total parameters, using architectures adopted in recent frontier models. Under matched data and training settings, RadFree consistently lowers validation loss and improves downstream performance across scales, with gains on nearly all 18 benchmarks and an average accuracy improvement of about 1.5 percentage points. It also complements mHC, more than doubling the accuracy gain of mHC alone

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.

over the standard residual baseline. Mechanistic analysis shows that RadFree reduces final residual RMS by approximately 36–37% across all three model scales. The method is simple to implement and adds no parameters. With simple kernel fusion, RadFree delivers these gains with zero training and inference overhead, offering a simple and practical way to improve LLM performance.

To summarize, our contributions are as follows:

- We identify a depth-dependent mismatch between feature representation quality and residual growth, motivating targeted removal of radial writes in late layers.
- We propose RadFree with depth-selective placement and a detached projection reference, and extend it to multi-stream residual connections by projecting writes onto the orthogonal complement of the streams' joint span.
- We demonstrate consistent gains at frontier scale in MoE pre-training up to 125B total parameters. RadFree more than doubles the accuracy gain of mHC alone and achieves zero training and inference overhead through simple kernel fusion.

2 Related Works

Residual stream growth, normalization, and update geometry. Xiong et al. (2020) show that Pre-Norm improves gradient conditioning at initialization. Its unnormalized skip path, however, allows residual magnitudes to grow with depth. Sun et al. (2026) link residual magnitude growth to reduced deep-layer influence and propose LayerNorm Scaling to control this growth and increase the contribution of deeper layers. More broadly, Wang et al. (2024a) stabilize deep Post-Norm Transformers through residual scaling and depth-dependent initialization. To reconcile scale control with gradient flow, Chen & Wei (2026) combine depth-scaled skip connections with normalization at sublayer inputs and after residual addition, while Li et al. (2026) couple normalized and unnormalized streams through shared residual updates. Geometric approaches constrain states or writes: nGPT (Loshchilov et al., 2025) renormalizes learned interpolations of normalized states and sublayer outputs to the unit sphere, whereas Oh et al. (2026) remove write components parallel to the state. RadFree shares this ideal single-stream forward projection but detaches the projection basis, incorporates layer selection, and extends to multi-stream residual connections.

Expanded residual states and multi-stream connections. Expanded-residual designs increase residual capacity without proportionally widening the backbone. Hyper-Connections (Zhu et al., 2025) maintain parallel residual streams with learnable read, write, and mixing operators, improving language modeling performance with little additional computation. mHC (Xie et al., 2025) constrains the residual mixing matrices to be doubly stochastic, improving training stability, model performance, and scalability. Gated Residual (Qwen Team, 2026a) uses elementwise gated reads and removes explicit stream mixing, reducing memory traffic while maintaining performance comparable to mHC. Li et al. (2025) generalize hyper-connections to finer partitions of expanded residual states. Mak & Flanigan (2025) use memory matrices with key-based reads and outer-product writes, while Zhang et al. (2026) use gated delta-rule updates to read and selectively replace stored features.

3 Methodology

3.1 Preliminaries

Standard residual connection. We consider a Pre-Norm Transformer (Xiong et al., 2020) with L layers, each consisting of an attention sublayer and an FFN sublayer. For a single token, let $\boldsymbol{x}_\ell \in \mathbb{R}^d$ denote the residual state entering sublayer ℓ. The residual update is

$$\boldsymbol{x}_{\ell+1} = \boldsymbol{x}_\ell + \boldsymbol{y}_\ell, \qquad \boldsymbol{y}_\ell = F_\ell\big(\mathrm{N}(\boldsymbol{x}_\ell)\big), \qquad \ell = 1, \ldots, 2L, \tag{1}$$

where F_ℓ is the attention or FFN sublayer, N denotes RMSNorm (Zhang & Sennrich, 2019), and $\boldsymbol{y}_\ell$ is the *residual write*. The final residual state is also normalized before the output projection.

RMSNorm computes $\mathrm{N}(\boldsymbol{x}) = \boldsymbol{\gamma} \odot \boldsymbol{x} / \mathrm{RMS}(\boldsymbol{x})$ where $\mathrm{RMS}(\boldsymbol{x}) = \|\boldsymbol{x}\|/\sqrt{d}$ and $\boldsymbol{\gamma}$ is a learned gate. For $\boldsymbol{x} \neq \mathbf{0}$ and $c > 0$, $\mathrm{N}(c\boldsymbol{x}) = \mathrm{N}(\boldsymbol{x})$. Thus, each sublayer after RMSNorm is sensitive only to the direction of the residual stream, but not its magnitude.

Hyper-connections. Hyper-connections(HC) (Zhu et al., 2025) extend standard residual connections by maintaining n parallel residual streams. Stacking the streams as the rows of $\boldsymbol{X}_\ell \in \mathbb{R}^{n \times d}$, the update is

$$\boldsymbol{X}_{\ell+1} = \mathcal{H}_{\mathrm{res}}\, \boldsymbol{X}_\ell + \mathcal{H}_{\mathrm{post}}\, \boldsymbol{y}_\ell^\top, \qquad \boldsymbol{y}_\ell = F_\ell\big(\mathrm{N}((\mathcal{H}_{\mathrm{pre}}\, \boldsymbol{X}_\ell)^\top)\big), \tag{2}$$

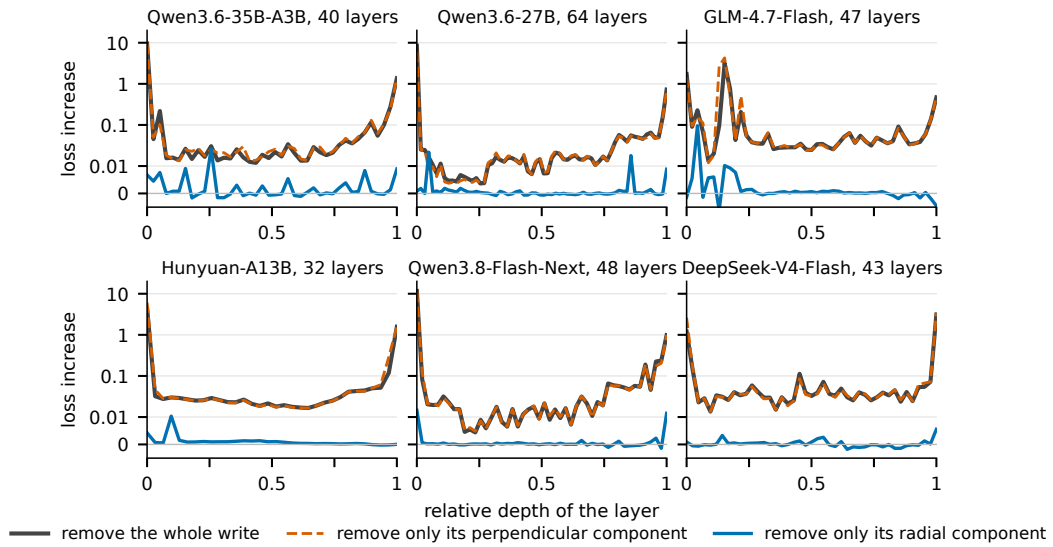


Figure 1: Loss increases from removing the full residual write, its perpendicular component, or its radial component in six open models. Each sublayer is intervened on separately. The horizontal axis is the depth of the layer divided by the depth of the model, and the vertical axis is the loss increase.

where $\mathcal{H}_{\text{pre}} \in \mathbb{R}^{1\times n}$ reads a mix of the streams into the sublayer, $\mathcal{H}_{\text{post}} \in \mathbb{R}^{n\times 1}$ writes the output back to every stream, and $\mathcal{H}_{\text{res}} \in \mathbb{R}^{n\times n}$ mixes the streams among themselves. The three matrices are computed from $\boldsymbol{X}_\ell$ by a learned gate, and variants of HC differ in how they constrain $\mathcal{H}_{\text{res}}$.

3.2 The Two Components of a Residual Write

For clarity, we drop the sublayer index ℓ from equation 1 and write the residual update as $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{y}$. The sublayer output $\boldsymbol{y}$ splits into a **radial component** parallel to $\boldsymbol{x}$ and a **perpendicular component**,

$$\boldsymbol{y} = \alpha\,\boldsymbol{x} + \boldsymbol{p}, \qquad \alpha = \frac{\boldsymbol{x}^\top \boldsymbol{y}}{\|\boldsymbol{x}\|^2}, \qquad \boldsymbol{p} = \boldsymbol{P}_{\boldsymbol{x}}\,\boldsymbol{y}, \qquad \boldsymbol{P}_{\boldsymbol{x}} = \boldsymbol{I} - \frac{\boldsymbol{x}\boldsymbol{x}^\top}{\|\boldsymbol{x}\|^2}, \tag{3}$$

The updated state is $\boldsymbol{z} = (1+\alpha)\boldsymbol{x} + \boldsymbol{p}$. Since $\boldsymbol{p} \perp \boldsymbol{x}$, the rotation angle φ between $\boldsymbol{x}$ and $\boldsymbol{z}$ satisfies

$$\tan\varphi = \frac{1}{1+\alpha}\frac{\|\boldsymbol{p}\|}{\|\boldsymbol{x}\|}. \tag{4}$$

3.3 Effects of Residual Write Components Across Depth

To evaluate the contribution of each component to language modeling, we perform component ablations on six open models: Qwen3.6-35B-A3B (Qwen Team, 2026c), Qwen3.6-27B (Qwen Team, 2026b), GLM-4.7-Flash (Team et al.), Hunyuan-A13B, Qwen3.8-Flash-Next (Qwen Team, 2026a), and DeepSeek-V4-Flash (Xu et al., 2026). The last two use HC with four residual streams; for these models, the decomposition in equation 3 is taken relative to the mixed input to each sublayer. We compare three interventions: removing the full write $\boldsymbol{y}$, only the perpendicular component $\boldsymbol{p}$, or only the radial component $\alpha\boldsymbol{x}$. Each intervention is applied to one sublayer at a time. We evaluate on 500 public documents covering web text, encyclopedias, code, mathematics, and reasoning, each truncated to 2048 tokens. For each layer and intervention, we report the median per-document loss increase relative to the unmodified model.

Figure 1 plots the loss increases from the three interventions against relative depth for all six models. We make two observations:

- A sublayer contributes to language modeling primarily through the perpendicular component of its write. Removing this component produces loss increases comparable to removing the full write at most depths.
- The radial component is largely dispensable in the middle and later layers, where its removal causes little change in loss. Its contribution is concentrated in the shallowest layers of most models and the final layer of four of the six models.

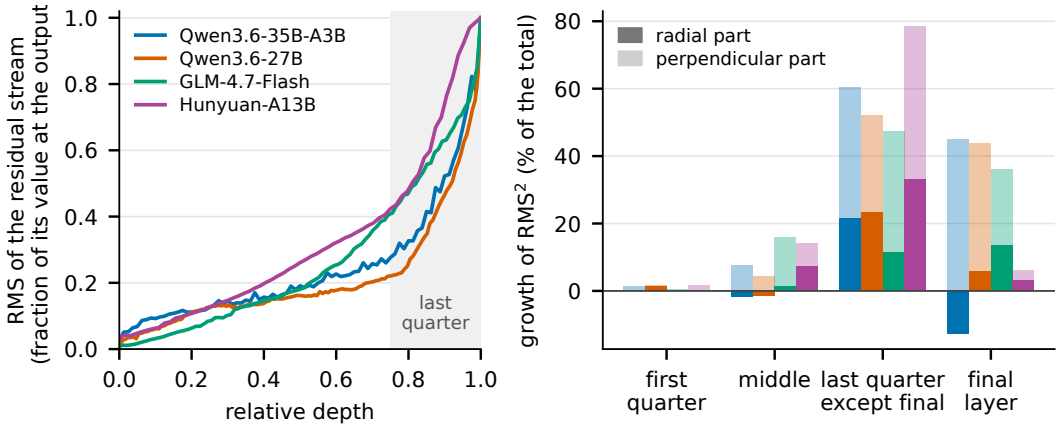


Figure 2: **Left:** Residual RMS at each sublayer input in four single-stream models. Each curve is normalized by the model's final residual RMS. Shading marks the last quarter of layers. **Right:** Radial and perpendicular contributions to the growth in squared residual RMS. Contributions are summed within each depth segment and reported as percentages of the total growth over all layers.

3.4 Residual Growth and Its Consequences

We examine residual growth in the four models with standard residual connections introduced in Section 3.3. Figure 2 (left) plots the residual RMS across depth, normalized by its final value. Growth is concentrated in the last quarter of the layers: the RMS increase over this segment equals 57%–78% of the final RMS.

To identify the source of this growth, we decompose the increase in squared RMS into radial and perpendicular contributions. Figure 2 (right) shows these contributions summed over four depth segments. Radial components in the last quarter, excluding the final layer, account for 11%–33% of the total increase in squared RMS over the network. In these layers, radial components therefore cause substantial residual growth despite their limited contribution to language modeling performance. As the residual norm grows, larger perpendicular updates are required to produce the same directional change. In the four models, the mean sublayer output RMS in the last quarter, excluding the final layer, is 5–11 times that in the first quarter.

This unnecessary growth has practical costs. Since RMSNorm fixes the scale of the sublayer input, deeper layers must compensate with larger weights to produce the larger outputs that the growing stream demands, which contributes to training instability (Kim et al., 2025; OLMo et al., 2024), complicates low-precision training and quantization (Liu et al., 2025; Dettmers et al., 2022; Xiao et al., 2023), and makes the effective learning rate uneven across depth (Loshchilov et al., 2025; Kosson et al., 2024). These observations motivate removing radial components in layers where they drive residual growth while offering limited benefits for language modeling.

3.5 Radial-Free Residual Write

We propose *Radial-Free Residual Write* (RadFree), which removes the radial component from the outputs of selected sublayers. The residual update becomes

$$\boldsymbol{x}_{\ell+1} = \boldsymbol{x}_\ell + \boldsymbol{P}_{\mathrm{sg}(\boldsymbol{x}_\ell)}\, \boldsymbol{y}_\ell, \tag{5}$$

where sg denotes the stop-gradient operator and $\boldsymbol{P}_{\boldsymbol{x}}$ is the projection defined in equation 3. This removes the radial contribution to residual growth while preserving the perpendicular update. The projection is applied per token and adds no parameters. We compute it in FP32 to reduce rounding errors that can leave a radial component in the projected write.

RadFree makes the residual write orthogonal to $\boldsymbol{x}_\ell$ while allowing the sublayer output to contain a radial component. An alternative is to encourage orthogonality by adding an auxiliary loss that penalizes the radial component of the raw sublayer output. We find that this approach degrades model performance (Section 4.3).

Placement. We apply RadFree to both attention and FFN sublayers in the last quarter of layers, excluding the final layer. In this region, radial components contribute substantially to residual growth but little to language modeling performance (Figures 1 and 2). We retain the final layer because

radial removal there still increases loss in several models. This placement achieves the highest average benchmark accuracy among the placements evaluated in Section 4.2.

Gradient design. We use stop-gradient to block gradients through the projection basis. Let $\boldsymbol{g} = \partial\mathcal{L}/\partial\boldsymbol{x}_{\ell+1}$ and $\boldsymbol{J} = \partial\boldsymbol{y}_\ell/\partial\boldsymbol{x}_\ell$. The sublayer output receives the projected gradient, while gradients propagate to the residual stream through the identity path and the sublayer input:

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{y}_\ell} = \boldsymbol{P}_{\boldsymbol{x}_\ell}\,\boldsymbol{g}, \qquad \frac{\partial\mathcal{L}}{\partial\boldsymbol{x}_\ell} = \boldsymbol{g} + \boldsymbol{J}^\top \boldsymbol{P}_{\boldsymbol{x}_\ell}\,\boldsymbol{g}. \tag{6}$$

Without stop-gradient, differentiating the projection matrix introduces two additional terms in the residual gradient (see Appendix A for the derivation):

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{x}_\ell} = \boldsymbol{g} + \boldsymbol{J}^\top \boldsymbol{P}_{\boldsymbol{x}_\ell}\,\boldsymbol{g} - \alpha_\ell\,\boldsymbol{P}_{\boldsymbol{x}_\ell}\,\boldsymbol{g} - \frac{\boldsymbol{x}_\ell^\top \boldsymbol{g}}{\|\boldsymbol{x}_\ell\|^2}\,\boldsymbol{p}_\ell. \tag{7}$$

To motivate the use of stop-gradient theoretically, we examine how these additional terms couple the discarded radial component to gradient propagation. The radial component is removed from the forward write, and the projected gradient at the sublayer output gives it no direct learning signal. Yet, through the first additional term, its coefficient α_ℓ scales the perpendicular contribution of the identity path by $1 - \alpha_\ell$. This weakens that contribution when $0 < \alpha_\ell < 1$ and reverses it when $\alpha_\ell > 1$. Thus, a coefficient with no direct learning signal can affect part of the gradient passed to earlier sublayers. Stop-gradient removes these additional terms while retaining the gradients through the identity path and the sublayer input.

Our ablations support this design: removing stop-gradient largely eliminates the improvements in both validation loss and downstream task performance (Section 4.2).

Extension to hyper-connections. Hyper-connections mix the existing residual streams through $\mathcal{H}_{\text{res}}$. The sublayer output can introduce an additional mixing term if it contains a component in the linear span of these streams. To see this, write $\boldsymbol{y}_\ell^\top = \boldsymbol{c}\,\boldsymbol{X}_\ell + \boldsymbol{p}_\ell^\top$, where $\boldsymbol{c} \in \mathbb{R}^{1\times n}$ and $\boldsymbol{p}_\ell$ is orthogonal to every residual stream, i.e., $\boldsymbol{X}_\ell\boldsymbol{p}_\ell = \boldsymbol{0}$. Substituting this decomposition into equation 2 gives

$$\boldsymbol{X}_{\ell+1} = \big(\mathcal{H}_{\text{res}} + \mathcal{H}_{\text{post}}\,\boldsymbol{c}\big)\boldsymbol{X}_\ell + \mathcal{H}_{\text{post}}\,\boldsymbol{p}_\ell^\top. \tag{8}$$

The component in the span adds the rank-one term $\mathcal{H}_{\text{post}}\,\boldsymbol{c}$ to the residual mixing matrix. RadFree removes this additional term, leaving the mixing of existing streams to $\mathcal{H}_{\text{res}}$ and retaining only the output component orthogonal to all streams.

To implement this projection, we compute an orthonormal basis $\boldsymbol{Q} \in \mathbb{R}^{d\times k}$ for the row space of $\text{sg}(\boldsymbol{X}_\ell)$, where $k = \text{rank}(\boldsymbol{X}_\ell)$. The basis is computed per token using Gram–Schmidt Orthogonalization (Golub & Van Loan, 2013). The residual update becomes

$$\boldsymbol{X}_{\ell+1} = \mathcal{H}_{\text{res}}\,\boldsymbol{X}_\ell + \mathcal{H}_{\text{post}}\,\big((\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\,\boldsymbol{y}_\ell\big)^\top. \tag{9}$$

Since the projected output is orthogonal to every input stream, it is also orthogonal to every mixed stream in $\mathcal{H}_{\text{res}}\boldsymbol{X}_\ell$. With a single residual stream, this projection reduces to $\boldsymbol{P}_{\boldsymbol{x}_\ell}$ in equation 5.

4 Experiments

4.1 Main Results

We evaluate RadFree through frontier-scale pre-training of modern MoE LLMs, using architectures and training methods adopted by recent frontier models. Our experiments cover standard Pre-Norm residual connections at three model scales with up to 125B total parameters, and further assess RadFree in multi-stream architectures based on mHC.

Setup. Our baseline models follow the architecture of recent open LLMs (Xiao et al., 2026; Zeng et al., 2026; Xu et al., 2026; Team et al., 2026a; DeepSeek-AI, 2026). They are MoE with multi-head latent attention (Liu et al., 2024a) with head-wise output gating (Qiu et al., 2026), sliding-window attention (Beltagy et al., 2020) in three of every four layers, and one multi-token prediction module (Gloeckle et al., 2024; Liu et al., 2024b). Every MoE sublayer has 144 fine-grained routed experts with top-8 routing and one shared expert (Dai et al., 2024). We evaluate three model sizes: a 17B-total, 1.5B-activated model (17B-A1.5B), a 67B-total, 5B-activated model (67B-A5B), and a 125B-total, 10B-activated model (125B-A10B). All models use a sequence length of 8192, the

Table 1: Validation loss and benchmark accuracy (%) of RadFree on the standard residual connection at three scales and on manifold-constrained hyper-connections (mHC) with four streams. Base is the baseline, and +RadFree adds RadFree to mHC as described in Section 3.5. The Average row of each domain reports the average over its benchmarks, and the overall average is taken over all 18 benchmarks. The best result within each group of columns is in bold.

	17B		**67B**		**125B**		**17B multi-stream**	
	Base	RadFree	Base	RadFree	Base	RadFree	mHC	+RadFree
Validation loss ↓	1.9842	**1.9790**	1.7855	**1.7754**	1.6903	**1.6835**	1.9548	**1.9518**
Knowledge								
MMLU	53.30	**54.29**	64.98	**67.10**	73.40	**74.45**	54.58	**55.39**
MMLU-Pro	22.93	**23.38**	39.71	**40.40**	49.60	**50.79**	24.31	**24.70**
ARC-C	**65.29**	**65.29**	82.47	**88.32**	**91.07**	**91.07**	66.32	**68.04**
AGIEval	40.99	**41.98**	54.50	**56.10**	62.44	**64.58**	42.76	**44.75**
CMMLU	55.17	**57.58**	70.49	**73.46**	79.47	**80.69**	57.69	**58.44**
C-Eval	54.06	**54.46**	69.72	**71.50**	77.75	**79.45**	55.55	**56.35**
Xiezhi	61.17	**62.24**	71.96	**73.29**	76.81	**77.24**	62.46	**63.40**
Average	50.42	**51.32**	64.83	**67.17**	72.93	**74.04**	51.95	**53.01**
Reasoning								
BBH	38.59	**40.59**	56.64	**60.77**	69.11	**72.06**	41.23	**41.88**
PIQA	60.55	**67.08**	80.52	**80.74**	82.37	**84.93**	62.68	**63.77**
SIQA	59.11	**61.90**	67.98	**70.70**	72.55	**74.81**	59.93	**61.11**
C3	**77.64**	77.23	89.10	**90.14**	93.26	**94.03**	76.44	**77.70**
Average	58.97	**61.70**	73.56	**75.59**	79.32	**81.46**	60.07	**61.12**
Mathematics								
GSM8K	38.82	**40.11**	64.97	**68.76**	74.83	**77.94**	42.38	**44.39**
MATH	26.96	**27.92**	**49.84**	48.64	61.04	**63.04**	28.64	**30.32**
DROP	41.02	**43.24**	63.02	**63.26**	75.97	**76.61**	39.89	**46.92**
APE210K	23.52	**23.69**	44.17	**44.80**	57.34	**58.54**	27.49	**27.52**
Average	32.58	**33.74**	55.50	**56.37**	67.30	**69.03**	34.60	**37.29**
Code								
HumanEval+	23.17	**24.39**	**35.98**	34.15	43.90	**45.12**	21.95	**25.61**
MBPP+	35.34	**37.34**	48.87	**49.62**	52.13	**55.39**	35.84	**38.60**
BigCodeBench	19.30	**19.65**	**37.89**	36.14	44.21	**44.91**	20.00	**20.71**
Average	25.94	**27.13**	**40.91**	39.97	46.75	**48.47**	25.93	**28.31**
Overall average	44.27	**45.69**	60.71	**62.10**	68.74	**70.31**	45.56	**47.20**

Muon optimizer (Jordan et al., 2024; Liu et al., 2025), and a warmup-stable-decay learning rate schedule (Wen et al., 2024). All models are trained on a data mixture covering English, Chinese, code, mathematics, and reasoning corpora.

In all models, RadFree is applied to about the last quarter of the layers and leaves the final layer unchanged, following Section 3.5. Specifically, it is applied to both attention and the MoE sublayers of layers 21 to 27 in the 17B-A1.5B and 67B-A5B models and of layers 38 to 49 in the 125B-A10B model. Each RadFree model and its baseline use the same training data and hyperparameters, differing only in the residual update. Full hyperparameters and training details are provided in Appendix B.

We report the loss on a held-out validation set, and we evaluate the downstream performance of all models using OpenCompass (Contributors, 2023) on 18 benchmarks across four domains: Knowledge (MMLU (Hendrycks et al., 2020), MMLU-Pro (Wang et al., 2024b), ARC-C (Clark et al., 2018), AGIEval (Zhong et al., 2024), CMMLU (Li et al., 2024), C-Eval (Huang et al., 2023), Xiezhi (Gu et al., 2024)), Reasoning (BBH (Suzgun et al., 2022), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), C3 (Sun et al., 2020)), Mathematics (GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), DROP (Dua et al., 2019), APE210K (Zhao et al., 2020)), and Code (HumanEval+ and MBPP+ (Liu et al., 2023), BigCodeBench (Zhuo et al., 2025)).

Standard Residual Connection. Table 1 compares RadFree with the standard residual baseline across three model scales. Averaged over the 18 benchmarks, RadFree improves accuracy by 1.42,

Table 2: Ablations on the 17B-A1.5B model. Layers are numbered from 1 to 28. Forward and Backward indicate whether the radial component is removed from the write and from the gradient at the sublayer output, respectively. Stop-grad. indicates whether the projection basis is detached. Δ Loss is the validation loss minus that of the baseline. Avg. accuracy is the mean accuracy over the 18 benchmarks in Table 1.

	Layers	**Forward**	**Backward**	**Stop-grad.**	**Δ Loss $\downarrow$**	**Avg. acc (%) $\uparrow$**
(1) Baseline	none				0	44.27
(2) RadFree (default)	21–27	✓	✓	✓	$\mathbf{-0.0052}$	**45.69**
(3) All layers	1–28	✓	✓	✓	$+0.0040$	44.04
(4) Middle layers	11–18	✓	✓	✓	$+0.0051$	42.91
(5) Every third layer	3, 6, …, 27	✓	✓	✓	-0.0003	44.12
(6) Tail and final layer	21–28	✓	✓	✓	-0.0031	45.50
(7) No stop-gradient	21–27	✓	✓	×	$+0.0002$	44.32
(8) Forward only	21–27	✓	×	✓	diverges	
(9) Backward only	21–27	×	✓	✓	diverges	

1.39, and 1.57 percentage points at 17B-A1.5B, 67B-A5B, and 125B-A10B, respectively. The gains are comparable in magnitude across scales, indicating that the benefits of RadFree persist as model size increases from 17B to 125B total parameters.

These improvements extend across domains and individual tasks. RadFree increases the knowledge, reasoning, and mathematics averages at every scale and outperforms the baseline in 48 of the 54 benchmark comparisons. The 125B-A10B model achieves the largest average gain and matches or exceeds the baseline on all 18 benchmarks. At this scale, MBPP+, GSM8K, and BBH each improve by approximately 3 percentage points, demonstrating gains across code generation, mathematical problem solving, and multi-step reasoning.

Multi-Stream Residual Connection. We evaluate RadFree at the 17B-A1.5B scale using manifold-constrained hyper-connections (mHC). We set the number of residual streams to 4 and obtain a doubly stochastic $\mathcal{H}_{\text{res}}$ through Sinkhorn iterations. mHC alone raises the average accuracy of the single-stream baseline from 44.27 to 45.56. Adding RadFree as described in Section 3.5 raises it further to 47.20, and all 18 benchmarks improve. These results show that RadFree complements mHC, improving average accuracy by a further 1.64 points. RadFree also reduces validation loss at all three model scales and with the multi-stream residual connection.

4.2 Ablations

We ablate two design choices of RadFree: the layers where it is applied and the gradient design of the projection. All ablations use the 17B-A1.5B model. All variants use the same training data and hyperparameters as the baseline in Section 4.1. Each ablation changes one design choice relative to the default RadFree configuration. Table 2 reports the change in validation loss and downstream benchmark performance relative to the baseline.

Placement. The default configuration applies RadFree to layers 21–27, leaving the final layer unchanged. We compare it with four alternatives: all layers (3), eight middle layers (4), every third layer (5), and layers 21–28, including the final layer (6). Applying RadFree to all layers (3), as in Oh et al. (2026), increases validation loss by 0.0040 and reduces average accuracy by 0.23 points relative to the baseline. The middle placement (4) also worsens both metrics: validation loss increases by 0.0051, and average accuracy decreases by 1.36 points. Applying RadFree to every third layer (5) reduces validation loss by only 0.0003, compared with 0.0052 for the default placement, and does not improve average accuracy. Only the default placement (2) and its variant including the final layer (6) improve both validation loss and average accuracy, with accuracy gains of 1.42 and 1.23 points, respectively. These results agree with the probes in Section 3.3. Including the final layer increases validation loss by 0.0021 and reduces average accuracy by 0.19 points relative to the default placement, supporting our choice to leave the final layer unchanged.

Gradient design. RadFree projects the forward write and the gradient at the sublayer output, and applies stop-gradient to the projection basis. We separately remove stop-gradient (7), the backward projection (8), and the forward projection (9). Without stop-gradient (7), the projection follows the gradient design of Oh et al. (2026), and RadFree's gains largely disappear. Validation loss is 0.0002

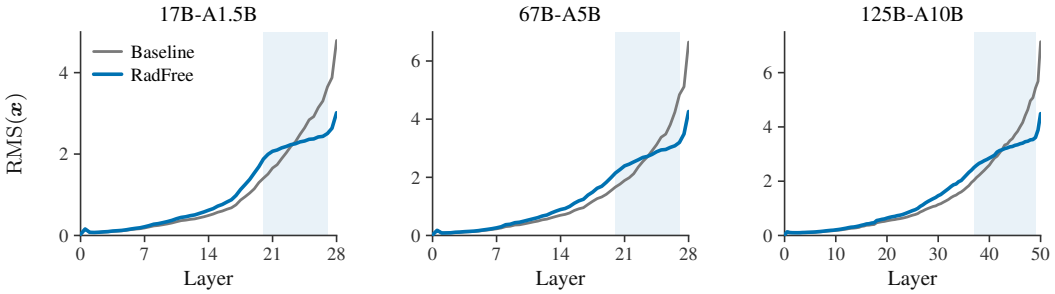


Figure 3: Residual RMS near the end of training, computed over tokens and hidden dimensions. Layer 0 denotes the embedding output; half-integer and integer positions show the states after attention and feed-forward residual updates, respectively. The final point precedes the final normalization. Shaded regions mark the RadFree projection windows.

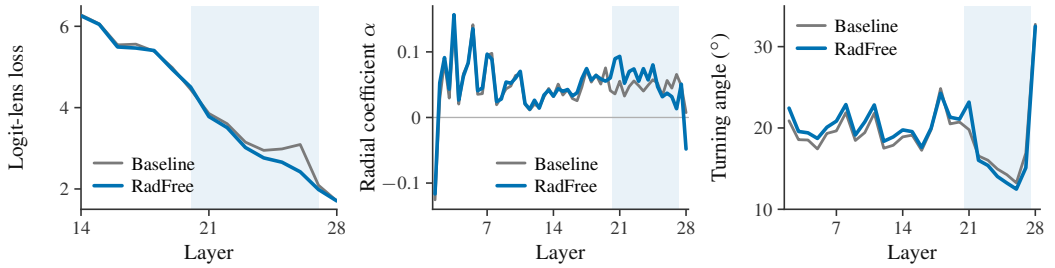


Figure 4: Mechanistic profiles of the 17B-A1.5B models at the end of training. **Left:** Logit-lens cross-entropy. **Center:** Mean radial coefficient α of the raw sublayer output before projection. **Right:** Layerwise turning angle. Shading marks the RadFree window.

higher than the baseline, and average accuracy improves by only 0.05 points. The variants with only the forward projection (8) or only the backward projection (9) both diverge, while the default configuration trains stably. These ablations show that stop-gradient is necessary for performance gains, and projection in both the forward and backward passes is fundamental for stable training.

4.3 Mechanistic Analysis

We examine RadFree's effects on residual growth and intermediate representations, and the distinction between constraining residual writes and raw sublayer outputs.

Reduced residual norm growth. Figure 3 compares residual RMS profiles near the end of training across three model scales. RadFree enters the RadFree window with a slightly higher residual RMS than the baseline. Within the window, its RMS grows much more slowly. The final residual-stream RMS is approximately 36–37% lower than the baseline across all three scales. These profiles show that RadFree curbs rapid residual growth in deep layers, thus reducing the need for progressively larger perpendicular updates to produce comparable directional changes. Such control of residual growth may also facilitate scaling Transformers to greater depths.

Improved intermediate representation. We assess the quality of intermediate representations in the final 17B-A1.5B checkpoints by applying each model's final normalization and output head to the residual state after each layer. Figure 4 (left) shows lower logit-lens (nostalgebraist, 2020) cross-entropy for RadFree throughout layers 21–27. Its loss decreases at every layer in this window, whereas the baseline's increases at layers 25 and 26; at layer 26, the losses are 2.42 and 3.09, respectively. These results indicate that RadFree improves intermediate representation quality. Figure 4 (right) further shows that RadFree maintains layerwise rotations comparable to those of the baseline.

Orthogonality at the residual write. Figure 4 (center) shows the radial coefficient α of each sublayer's raw output before projection. In RadFree, the α coefficient still remains positive in every projected sublayer, even though the branch receives a gradient orthogonal to the residual state. This suggests that an orthogonal output gradient does not ensure independent learning of the radial and perpendicular components, since updates to shared sublayer parameters can change both. We therefore hypothesize that suppressing the discarded radial component may also hinder learning of the perpendicular component.

Table 4: Forward-pass memory accesses per token of the residual write (in elements, hidden size d) and E2E training throughput (tokens/s/GPU). Percentages are relative to the baseline

	Memory accesses		Throughput		
Residual write	**Reads**	**Writes**	**17B-A1.5B**	**67B-A5B**	**125B-A10B**
Standard	$\mathbf{2}\boldsymbol{d}$	$\boldsymbol{d}$	7966	5155	2236
RadFree, unfused	$\mathbf{13}\boldsymbol{d}$	$\mathbf{8}\boldsymbol{d}$	7813 (-1.9%)	4908 (-4.8%)	2162 (-3.3%)
FP32 precision conversions	$3d$	$3d$			
Normalization and projection	$8d$	$4d$			
Residual merge	$2d$	d			
RadFree, fused	$\mathbf{2}\boldsymbol{d}$	$\boldsymbol{d}$	7952	5153	2237

To test this hypothesis, we add an auxiliary loss $\lambda\,\mathbb{E}[\alpha^2]$ to penalize the radial coefficients of the attention and FFN outputs in layers 21–27 of the 17B-A1.5B model, using $\lambda = 50$. The penalty variants are trained for 10,000 steps (about 31.5B tokens). We find that the penalty reduces the magnitude of α to about 0.002, but increases validation loss by 0.0052 with standard writes and by 0.0117 relative to RadFree without the penalty (Table 3). Both RadFree variants use the same projected write rule, so the latter comparison measures the effect of additionally constraining the raw output during training.

Table 3: Validation loss relative to the standard baseline. The reported loss excludes the auxiliary penalty.

Write	**No penalty**	**Penalty**
Standard	0	$+0.0052$
RadFree	-0.0061	$+0.0056$

The loss increase suggests that penalizing the raw radial component can also hinder learning of the perpendicular update. RadFree avoids this additional training constraint by enforcing orthogonality through projection at the residual write, while leaving the raw sublayer output unconstrained.

4.4 System Efficiency

In a naive implementation of RadFree, the additional kernel launches and memory traffic for precision conversion and projection lie on the critical path, reducing throughput. We fuse these operations with residual addition. The fused forward kernel reads $\boldsymbol{x}$ and $\boldsymbol{y}$ once and writes only the updated state, matching the standard residual addition's $2d$ reads and d writes per token (Table 4). The backward kernel reuses $\boldsymbol{x}$ retained for normalization backward to compute the gradient at the sublayer output $\boldsymbol{P_x g}$, without saving projection intermediates. For multi-stream residuals, basis orthonormalization is also performed on chip within the fused residual update.

We evaluate end-to-end training throughput using Megatron (Shoeybi et al., 2019) on NVIDIA H800 GPUs, comparing standard residual updates with unfused and fused RadFree under matched training configurations at each of three model scales. As shown in Table 4, unfused RadFree reduces throughput by 1.9–4.8%, while fused RadFree incurs zero overhead across all three scales. For inference, we applied the same kernel fusion in SGLang (Zheng et al., 2024) and also observed zero overhead relative to the baseline in both prefill and decode.

5 Conclusion and Outlook

We introduced RadFree, which controls residual growth through selective radial removal and a detached projection reference. Across MoE models up to 125B total parameters, RadFree consistently improves language modeling performance and complements mHC, with negligible overhead using fused kernels.

Limitations and Future Work. Our current design uses a fixed projection window, and its benefits for depth scaling remain to be established. Future work will explore adaptive placement across models and training stages, and evaluate whether controlling residual growth improves optimization in deeper networks.

AI USE STATEMENT

We used generative AI tools to assist with drafting and polishing the manuscript, including improving clarity, organization, and the presentation of experimental analyses. We developed the core research ideas and take responsibility for the final content and conclusions.

ETHICS STATEMENT

This work studies residual architectures and training methods for language models. Although RadFree does not target specific downstream applications, models trained with it remain subject to concerns about bias, privacy, and misuse. Our evaluation focuses on language modeling performance and training behavior; responsible deployment requires additional assessment of safety and fairness in the intended use setting.

REPRODUCIBILITY STATEMENT

We describe the RadFree formulation, layer placement, gradient design, and extension to hyper-connections in Section 3.5. Appendix A derives the gradients through the projection basis, and Appendix B provides model configurations and training hyperparameters. The component ablations, design ablations, and throughput evaluation are described in Sections 3.3, 4.2, and 4.4, respectively. We will release the code upon acceptance.

REFERENCES

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Chen Chen and Lai Wei. Post-layernorm is back: Stable, expressive, and deep. *arXiv preprint arXiv:2601.19895*, 2026.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

OpenCompass Contributors. Opencompass: A universal evaluation platform for foundation models. `https://github.com/open-compass/opencompass`, 2023.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 1280–1297, 2024.

DeepSeek-AI. Deepseek-v4.1-flash: Pushing the limits of kv cache compression, 2026.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35: 30318–30332, 2022.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*, 2024.

Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU press, 2013.

Zhouhong Gu, Xiaoxuan Zhu, Haoning Ye, Lin Zhang, Jianchen Wang, Yixin Zhu, Sihang Jiang, Zhuozhi Xiong, Zihan Li, Weijie Wu, et al. Xiezhi: An ever-updating benchmark for holistic domain knowledge evaluation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 18099–18107, 2024.

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*, pp. 770–778, 2016.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Yao Fu, et al. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *Advances in neural information processing systems*, 36:62991–63010, 2023.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL `https://kellerjordan.github.io/posts/muon/`.

Jeonghoon Kim, Byeongchan Lee, Cheonbok Park, Yeontaek Oh, Beomjun Kim, Taehwan Yoo, Seongjin Shin, Dongyoon Han, Jinwoo Shin, and Kang Min Yoo. Peri-ln: Revisiting normalization layer in the transformer architecture. *arXiv preprint arXiv:2502.02732*, 2025.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Atli Kosson, Bettina Messmer, and Martin Jaggi. Analyzing & reducing the need for learning rate warmup in gpt training. *Advances in Neural Information Processing Systems*, 37:2914–2942, 2024.

Baisheng Li, Banggu Wu, Bole Ma, Bowen Xiao, Chaoyi Zhang, Cheng Li, Chengyi Wang, Chengyin Xu, Chi Zhang, Chong Hu, et al. Virtual width networks. *arXiv preprint arXiv:2511.11238*, 2025.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. Cmmlu: Measuring massive multitask language understanding in chinese. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11260–11285, 2024.

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, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024a.

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*, 2024b.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Ilya Loshchilov, Cheng-Ping Hsieh, Simeng Sun, and Boris Ginsburg. ngpt: Normalized transformer with representation learning on the hypersphere. In *International Conference on Learning Representations*, volume 2025, pp. 74014–74038, 2025.

Brian Mak and Jeffrey Flanigan. Residual matrix transformers: Scaling the size of the residual stream. *arXiv preprint arXiv:2506.22696*, 2025.

Rob nostalgebraist. Interpreting gpt: The logit lens. *Blog Post*, 2020.

Giyeong Oh, Woohyun Cho, Siyeol Kim, Suhwan Choi, and Youngjae Yu. Revisiting residual connections: Orthogonal updates for stable and efficient deep networks. *Advances in Neural Information Processing Systems*, 38:71725–71759, 2026.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

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*, 38:100092–100118, 2026.

Qwen Team. On the design of Qwen3.8-Next architecture: Evaluation, efficiency, and training stability. Technical report, Alibaba Group, August 2026a.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model, April 2026b. URL `https://qwen.ai/blog?id=qwen3.6-27b`.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026c. URL `https://qwen.ai/blog?id=qwen3.6-35b-a3b`.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 4463–4473, 2019.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. Investigating prior knowledge for challenging chinese machine reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:141–155, 2020.

Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. *Advances in Neural Information Processing Systems*, 38: 163104–163136, 2026.

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

G Team, A Zeng, X Lv, Q Zheng, Z Hou, B Chen, C Xie, C Wang, D Yin, H Zeng, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models, 2025a. *URL https://arxiv.org/abs/2508.06471*.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026a.

Kimi Team, Guangyu Chen, Yu Zhang, Jianlin Su, Weixin Xu, Siyuan Pan, Yaoyu Wang, Yucheng Wang, Guanduo Chen, Bohong Yin, et al. Attention residuals. *arXiv preprint arXiv:2603.15031*, 2026b.

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*, 30, 2017.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10):6761–6774, 2024a.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024b.

Kaiyue Wen, Zhiyuan Li, Jason Wang, David Hall, Percy Liang, and Tengyu Ma. Understanding warmup-stable-decay learning rates: A river valley loss landscape perspective. *arXiv preprint arXiv:2410.05192*, 2024.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, et al. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*, 2026.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099. PMLR, 2023.

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.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International conference on machine learning*, pp. 10524–10533. PMLR, 2020.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

Yifan Zhang, Yifeng Liu, Mengdi Wang, and Quanquan Gu. Deep delta learning. *arXiv preprint arXiv:2601.00417*, 2026.

Wei Zhao, Mingyue Shang, Yang Liu, Liang Wang, and Jingming Liu. Ape210k: A large-scale and template-rich dataset of math word problems. *arXiv preprint arXiv:2009.11506*, 2020.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Livia Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37: 62557–62583, 2024.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. In *Findings of the association for computational linguistics: NAACL 2024*, pp. 2299–2314, 2024.

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*, volume 2025, pp. 97183–97219, 2025.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, et al. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. In *International Conference on Learning Representations*, volume 2025, pp. 66602–66656, 2025.

A Gradient through the Projection Basis

This appendix derives the two additional terms of equation 7 and the direction in which they move $\boldsymbol{x}$. We hold the branch output $\boldsymbol{y}$ and the incoming gradient $\boldsymbol{g}$ fixed and write $\hat{\boldsymbol{x}} = \boldsymbol{x}/\|\boldsymbol{x}\|$. The RadFree write is $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{y} - \boldsymbol{d}(\boldsymbol{x})$, where $\boldsymbol{d}(\boldsymbol{x}) = (\hat{\boldsymbol{x}}^\top \boldsymbol{y})\,\hat{\boldsymbol{x}}$ is the removed radial component. The vector $\boldsymbol{x}$ enters this expression twice. The first occurrence is the identity path, and the second is the projection basis, which decides what part of $\boldsymbol{y}$ is removed.

The additional terms. For any fixed vector $\boldsymbol{a}$ we have $\nabla_{\boldsymbol{x}}(\hat{\boldsymbol{x}}^\top \boldsymbol{a}) = \boldsymbol{P}_{\boldsymbol{x}}\,\boldsymbol{a}/\|\boldsymbol{x}\|$. With $\hat{\boldsymbol{x}}^\top \boldsymbol{y}/\|\boldsymbol{x}\| = \alpha$ and $\boldsymbol{P}_{\boldsymbol{x}}\,\boldsymbol{y} = \boldsymbol{p}$, the Jacobian of the removed component is

$$\frac{\partial \boldsymbol{d}}{\partial \boldsymbol{x}} = \frac{\hat{\boldsymbol{x}}\,\boldsymbol{p}^\top}{\|\boldsymbol{x}\|} + \alpha\,\boldsymbol{P}_{\boldsymbol{x}}. \tag{10}$$

The gradient that reaches $\boldsymbol{x}$ through the basis is therefore

$$-\Big(\frac{\partial \boldsymbol{d}}{\partial \boldsymbol{x}}\Big)^\top \boldsymbol{g} = -\alpha\,\boldsymbol{P}_{\boldsymbol{x}}\,\boldsymbol{g} - \frac{\boldsymbol{x}^\top \boldsymbol{g}}{\|\boldsymbol{x}\|^2}\,\boldsymbol{p}, \tag{11}$$

which gives the two terms of equation 7. Both are orthogonal to $\boldsymbol{x}$. Adding the first to the identity path gives $\boldsymbol{g} - \alpha\,\boldsymbol{P}_{\boldsymbol{x}}\,\boldsymbol{g} = (\hat{\boldsymbol{x}}^\top \boldsymbol{g})\,\hat{\boldsymbol{x}} + (1-\alpha)\,\boldsymbol{P}_{\boldsymbol{x}}\,\boldsymbol{g}$, so the part of $\boldsymbol{g}$ that is orthogonal to $\boldsymbol{x}$ is multiplied by $1 - \alpha$ and the radial part is unchanged.

B Experimental Details

Table 5 lists the model configurations and training hyperparameters of the three model scales and of the four-stream mHC model. Each RadFree model shares all entries with its baseline and differs only in the residual write of the RadFree layers.

Table 5: Model configurations and training hyperparameters. The mHC column is the four-stream hyper-connection model of Table 1. In the attention window pattern, G denotes a global-attention layer and 512 a sliding-window layer with window size 512.

	Parameters	17B-A1.5B	67B-A5B	125B-A10B	17B-A1.5B mHC
Optimizer	optimizer	Muon (Jordan et al., 2024), AdamW (Kingma & Ba, 2014)			
	Muon momentum	0.95			
	AdamW (β_1, β_2)	(0.9, 0.95)			
	lr-schedule	warmup-stable-decay, exponential decay			
	max, min lr	(3.97e-4, 3.97e-5)	(2.8e-4, 2.8e-5)	(2.35e-4, 2.35e-5)	(3.97e-4, 3.97e-5)
	warmup steps, stable-ratio	(500, 0.80)			
	weight-decay, grad_clip	(0.1, 1.0)			
Batch	global batch size	384	768	1024	384
	context-length	8192			
	#training tokens	92B	241B	397B	92B
Backbone	#params (total, activated)	(17B, 1.5B)	(67B, 5B)	(125B, 10B)	(17B, 1.5B)
	hidden dim.	2112	4096	4096	2112
	#layers (dense, MoE)	(1, 27)	(1, 27)	(1, 49)	(1, 27)
	#attention heads	16	32	32	16
	MLA (q, kv) lora rank	(1024, 512)	(1536, 512)	(1536, 512)	(1024, 512)
	(qk, rope, v) head dim.	(128, 64, 128)			
	attention window pattern	(G, 512, 512, 512)×7; 125B-A10B: G, (G, 512, 512, 512)×12, G			
	rotary base	50,000			
	#experts (routed, shared), topK	(144, 1), 8			
	FFN size (dense, MoE)	(6144, 672)	(12288, 1376)	(12288, 1408)	(6144, 672)
	MTP layers	1			
Routing	bias update rate, balance loss coef.	(0.001, 1e-4)			
Data	vocab size	152,064			
mHC	#streams, Sinkhorn iterations	–	–	–	(4, 20)
RadFree	layers	21–27	21–27	38–49	21–27