uHC: UNCOUPLED HYPER-CONNECTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Hyper-Connections (HC) extend Transformer residual representations into multiple streams, while Manifold-Constrained Hyper-Connections (mHC) stabilize multi-stream propagation through constrained residual mixing. However, existing HC architectures maintain coupled interfaces between residual streams and the transformation blocks: readout patterns are shared across feature dimensions, and write updates remain aligned across streams, limiting the benefits of residual-stream expansion. We identify this *transformation-interface coupling* as a structural bottleneck. To alleviate this limitation, we propose **Uncoupled Hyper-Connections (uHC)**, which redesigns the transformation interfaces without modifying the underlying Transformer blocks. uHC introduces *Group-Wise Feature Readout* for feature-dependent stream aggregation and *Low-Rank Directional Write-Back* for stream-specific update directions with lightweight overhead. Across three model scales ranging from 46M to 363M parameters, uHC consistently improves validation loss and perplexity over residual connections, mHC, and mHC-lite. At the largest scale, uHC achieves the best performance across all reported metrics, reducing validation loss from 2.9470 for mHC to 2.8188. Further analyses reveal that uHC better utilizes the expanded residual-stream space and achieves more stable optimization, highlighting the importance of decoupling transformation interfaces for scaling multi-stream residual architectures.

1 INTRODUCTION

Residual connections are a fundamental architectural primitive for training deep neural networks (He et al., 2016) and remain integral to modern Transformer architectures (Vaswani et al., 2017). By providing a direct pathway across nonlinear transformations, they facilitate signal and gradient propagation while preserving earlier representations. However, this additive mechanism progressively aggregates layer-wise representations into a single residual state, motivating recent efforts to redesign cross-layer information flow (Xiao et al., 2025; Heddes et al., 2025). Attention Residuals (AttnRes), for instance, replace uniform residual accumulation with adaptive aggregation over earlier representations (Kimi Team et al., 2026). Hyper-Connections (HC), in contrast, expand the residual state into multiple streams and learn how information is routed among them, introducing an additional architectural axis beyond conventional width and depth scaling (Zhu et al., 2025a).

A key stability challenge in HC stems from the multiplicative accumulation of unconstrained residual mixing operators across depth, potentially resulting in substantial signal amplification or attenuation. Manifold-Constrained Hyper-Connections (mHC) mitigate this instability by restricting the residual mixing operator to the Birkhoff polytope through Sinkhorn–Knopp normalization, yielding approximately doubly stochastic mappings (Xie et al., 2026). Subsequent variants further improve the efficiency and flexibility of multi-stream propagation by modifying the parameterization or geometry of residual mixing operators (Yang & Gao, 2026; Zhou et al., 2026; Liu et al., 2026). Collectively, these works improve the stability and flexibility of multi-stream residual propagation.

However, residual mixing governs only how existing stream states propagate across layers, leaving their interaction with the nonlinear block largely unchanged. From this perspective, an HC layer comprises two complementary pathways: a *residual-stream pathway* that propagates and recombines stream states, and a *transformation pathway* consisting of a readout interface, a nonlinear block such as attention or an FFN, and a write-back interface(Zhu et al., 2025a; Xie et al., 2026). Existing mHC variants primarily modify residual mixing within the *residual-stream pathway*, while leaving the readout and write-back interfaces largely unchanged. At readout, all hidden dimensions share

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.

the same stream-routing coefficients, coupling heterogeneous feature subspaces through a common residual-stream mixture; at write-back, each residual stream receives only a scalar-modulated copy of the same output vector, constraining stream-wise updates to a common direction in feature space. We refer to these interface restrictions collectively as *transformation-interface coupling*.

This asymmetry becomes increasingly pronounced as the number of residual streams grows: residual expansion increases the capacity of the residual stream pathway, while the transformation interfaces remain constrained by shared routing patterns and shared update directions. Recent xHC independently addresses the write-back side of this limitation by enriching the write-back basis with additional feature components (Zhang et al., 2026a). We therefore consider a complementary design that jointly relaxes the constraints at both the readout and write-back interfaces.

To this end, we propose **Uncoupled Hyper-Connections (uHC)**, an extension of mHC that relaxes the coupling at both the readout and write-back interfaces. First, **Group-Wise Feature Readout** partitions the hidden dimension into feature groups and assigns each group its own residual-stream routing pattern, allowing different feature subspaces to form different mixtures of the residual streams. Second, **Low-Rank Directional Write-Back** augments the shared block output with stream-specific low-rank corrections, enabling residual streams to receive updates along different feature-space directions rather than only rescaled copies of a common vector. Both modifications are restricted to the transformation interfaces, leaving the attention or FFN block itself unchanged and introducing only minimal additional overhead.

We evaluate uHC across three model scales, from approximately 46M to 363M parameters. Against vanilla residual connections, mHC, and mHC-lite, uHC consistently achieves lower training and validation loss, together with improved perplexity across multiple evaluation corpora. This advantage is most pronounced at the largest scale, where uHC obtains the best result on every reported metric and reduces validation loss from 2.9470 for mHC to **2.8188**. Beyond aggregate performance, our analyses show that uHC induces more distributed readout and non-collinear stream-wise write updates, while exhibiting smoother gradient dynamics than mHC. Its validation-loss advantage over mHC further widens as the residual-stream expansion rate grows from $n = 4$ to $n = 16$ at the medium scale, suggesting that relaxing transformation-interface coupling enables more effective use of additional residual streams. Our contributions are threefold:

- We identify *transformation-interface coupling* as a structural bottleneck in mHC, which restricts how nonlinear transformations interact with the expanded residual-stream space and limits the benefits of residual-stream expansion.
- We propose **Uncoupled Hyper-Connections (uHC)**, which decouple transformation interfaces through Group-Wise Feature Readout and Low-Rank Directional Write-Back, enabling feature-dependent readout and stream-specific updates with lightweight overhead.
- We demonstrate that uHC consistently improves language-modeling performance over existing residual architectures across multiple model scales, while exhibiting more stable optimization and better utilization of additional residual streams.

2 Preliminaries and Motivation

We first revisit Hyper-Connections (HC) (Zhu et al., 2025a) and Manifold-Constrained Hyper-Connections (mHC) (Xie et al., 2026) through a two-pathway decomposition of multi-stream residual computation. This perspective allows us to separate residual-stream propagation from nonlinear feature transformation and reveals a structural asymmetry in how the two pathways utilize the expanded residual-stream space. We characterize this asymmetry as *transformation-interface coupling*, which motivates the design introduced in the following section.

2.1 Hyper-Connections

We denote the n residual streams at layer l by $\mathbf{X}_l = [\mathbf{x}_{l,1}; \ldots; \mathbf{x}_{l,n}] \in \mathbb{R}^{n \times C}$, where C is the hidden dimension. Under the two-pathway view, an HC layer can be written as

$$\mathbf{X}_{l+1} = \underbrace{\mathcal{H}_l^{\mathrm{res}} \mathbf{X}_l}_{\mathrm{RSP}_l(\mathbf{X}_l)} + \underbrace{\left(\mathcal{H}_l^{\mathrm{post}}\right)^\top \mathcal{F}_l\left(\mathcal{H}_l^{\mathrm{pre}} \mathbf{X}_l\right)}_{\mathrm{TP}_l(\mathbf{X}_l)}. \tag{1}$$

Here, $\mathcal{F}_l$ denotes a generic transformation sublayer, such as self-attention or a feed-forward network (FFN) in the language models considered in this work. The first term corresponds to the *residual-stream pathway* (RSP), which propagates and mixes representations across residual streams, while the second corresponds to the *transformation pathway* (TP), which reads from the streams, applies $\mathcal{F}_l$, and writes the transformed features back.

The three mappings $\mathcal{H}_l^{\text{pre}}, \mathcal{H}_l^{\text{post}} \in \mathbb{R}^{1\times n}$ and $\mathcal{H}_l^{\text{res}} \in \mathbb{R}^{n\times n}$ govern stream readout, transformed-feature write-back, and residual-stream mixing, respectively.

2.2 Manifold-Constrained Hyper-Connections

mHC (Xie et al., 2026) stabilizes residual-stream propagation by projecting the residual mixing matrix onto the Birkhoff polytope, reducing the feature amplification or attenuation that can accumulate through repeated composition across depth. It constructs a normalized global representation and parameterizes the three mappings as

$$\begin{aligned}
\bar{\mathbf{X}}_l &= \mathrm{RMSNorm}(\mathrm{vec}(\mathbf{X}_l)) \in \mathbb{R}^{1\times nC},\\
\mathcal{H}_l^{\text{pre}} &= \sigma\big(\alpha_l^{\text{pre}} \bar{\mathbf{X}}_l \boldsymbol{\varphi}_l^{\text{pre}} + \mathbf{b}_l^{\text{pre}}\big),\\
\mathcal{H}_l^{\text{post}} &= 2\sigma\big(\alpha_l^{\text{post}} \bar{\mathbf{X}}_l \boldsymbol{\varphi}_l^{\text{post}} + \mathbf{b}_l^{\text{post}}\big),\\
\mathcal{H}_l^{\text{res}} &= \mathrm{Sinkhorn}\big(\alpha_l^{\text{res}}\, \mathrm{mat}\big(\bar{\mathbf{X}}_l \boldsymbol{\varphi}_l^{\text{res}}\big) + \mathbf{b}_l^{\text{res}}\big).
\end{aligned} \tag{2}$$

Here, $\mathrm{mat}(\cdot)$ reshapes an n^2-dimensional vector into an $n \times n$ matrix. The Sinkhorn operator constrains $\mathcal{H}_l^{\text{res}}$ to be approximately doubly stochastic, controlling residual propagation across depth. The sigmoid parameterization constrains the readout and write-back coefficients to positive ranges, while the factor 2 yields unit write-back coefficients at zero-logit initialization.

2.3 Coupling in the Transformation Interfaces

The structural constraint on $\mathcal{H}_l^{\text{res}}$ substantially improves the stability and scalability of multi-stream residual propagation. Nevertheless, the transformation pathway retains the topology inherited from HC:

$$n \text{ streams} \xrightarrow{\mathcal{H}^{\text{pre}}} 1 \text{ feature vector} \xrightarrow{\mathcal{F}} 1 \text{ feature vector} \xrightarrow{\mathcal{H}^{\text{post}}} n \text{ streams}. \tag{3}$$

This creates a structural asymmetry between the two pathways. The RSP maintains multiple residual streams and supports stream-specific routing through an $n \times n$ mixing matrix, whereas the TP interacts with these streams through a single shared C-dimensional representation. We refer to this restriction as *transformation-interface coupling*. It manifests in two forms at the interfaces of the TP: feature-wise routing coupling at readout and directional coupling at write-back.

Feature-wise routing coupling. With $\mathcal{H}_l^{\text{pre}} = [a_{l,1}, \ldots, a_{l,n}]$, the transformation input is

$$\begin{aligned}
\mathbf{u}_l &= \mathcal{H}_l^{\text{pre}} \mathbf{X}_l = \sum_{s=1}^{n} a_{l,s} \mathbf{x}_{l,s},\\
u_{l,c} &= \sum_{s=1}^{n} a_{l,s} x_{l,s,c}, \qquad c = 1, \ldots, C.
\end{aligned} \tag{4}$$

Here, c indexes the hidden dimensions of the C-dimensional representation. The same stream-routing coefficients $\{a_{l,s}\}$ are applied to every hidden dimension, independent of c. Different feature dimensions are therefore coupled through a shared stream-routing pattern, preventing distinct feature subspaces from adopting different residual-stream mixtures. This restricts $\mathcal{F}_l$ to a single stream-aggregated view of the multi-stream representation.

Directional write-back coupling. With $\mathcal{H}_l^{\text{post}} = [b_{l,1}, \ldots, b_{l,n}]$ and $\mathbf{h}_l = \mathcal{F}_l(\mathbf{u}_l)$, the stream updates satisfy

$$\begin{aligned}
\Delta \mathbf{X}_l &= \big(\mathcal{H}_l^{\text{post}}\big)^{\top} \mathbf{h}_l,\\
\Delta \mathbf{x}_{l,s} &= b_{l,s} \mathbf{h}_l, \qquad s = 1, \ldots, n.
\end{aligned} \tag{5}$$

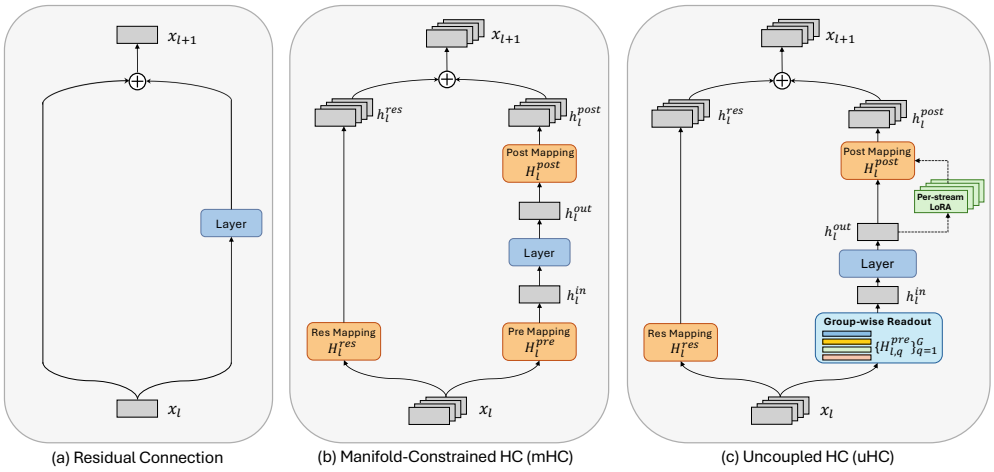

Figure 1: **Architectural comparison of residual connections, mHC, and uHC.** (a) A standard residual connection propagates a single residual stream. (b) Manifold-Constrained Hyper-Connections (mHC) expand the residual pathway into multiple streams with manifold-constrained learned routing. (c) Uncoupled Hyper-Connections (uHC) decouple the transformation interface through group-wise feature readout and stream-specific low-rank write-back.

Each non-zero stream update is therefore a scalar multiple of the same feature vector $\mathbf{h}_l$; hence, the write-back matrix $\Delta\mathbf{X}_l \in \mathbb{R}^{n\times C}$ has rank at most one. The write-back coefficients can modulate update magnitudes across streams, but not their feature-space directions. In this sense, the residual streams remain directionally coupled through a shared transformed feature.

Asymmetric scaling with residual expansion. Transformation-interface coupling becomes increasingly restrictive as the number of residual streams grows. The residual mixing matrix $\mathcal{H}_l^{\text{res}} \in \mathbb{R}^{n\times n}$ can have rank up to n, allowing residual expansion to increase the diversity of stream-wise combinations maintained by the RSP. In contrast, the shared transformation readout continues to apply a single routing pattern across all feature dimensions; equivalently, its effective routing matrix $\mathbf{1}_C\mathcal{H}_l^{\text{pre}}$ remains rank one. Likewise, Eq. 5 shows that the write-back matrix remains rank at most one regardless of n.

Thus, residual expansion can increase the routing diversity available to the RSP without a corresponding increase in the number of independent routing patterns or feature-space directions available to the transformation pathway. We provide a complementary null-space interpretation in Appendix C.4. This growing structural asymmetry motivates relaxing both forms of transformation-interface coupling while keeping the core transformation $\mathcal{F}_l$ unchanged.

3 Method

Building on the above analysis, we extend mHC by redesigning both interfaces of the transformation pathway with two orthogonal and complementary mechanisms for relaxing transformation-interface coupling. Specifically, we introduce *Group-Wise Feature Readout* to relax feature-wise routing coupling before the nonlinear transformation (Sec. 3.1), and *Low-Rank Directional Write-Back* to relax directional coupling after the transformation by introducing stream-specific update directions (Sec. 3.2). The parameter efficiency of uHC is further discussed in Sec. 3.3.

3.1 Group-Wise Feature Readout

In Eq. 4, the same stream-routing coefficients are shared across all feature dimensions. We relax this coupling by allowing different feature subspaces to read from the residual streams independently.

Specifically, we partition the hidden dimension into G non-overlapping groups of equal width $d = C/G$, such that $\mathbf{x}_{l,s} = [\mathbf{x}_{l,s}^{(1)}; \ldots; \mathbf{x}_{l,s}^{(G)}]$, where $\mathbf{x}_{l,s}^{(g)} \in \mathbb{R}^d$. We reshape the mHC-normalized state $\bar{\mathbf{X}}_l$ in Eq. 2 as $\tilde{\mathbf{X}}_l = [\tilde{\mathbf{x}}_{l,1}; \ldots; \tilde{\mathbf{x}}_{l,n}] \in \mathbb{R}^{n\times C}$. For each group g, the corresponding normalized feature

slices across streams are concatenated as $\tilde{\mathbf{x}}_l^{(g)} = \mathrm{Concat}(\tilde{\mathbf{x}}_{l,1}^{(g)}, \ldots, \tilde{\mathbf{x}}_{l,n}^{(g)}) \in \mathbb{R}^{nd}$, from which we generate group-specific stream-routing coefficients:

$$\boldsymbol{\alpha}_l^{(g)} = \sigma\left(\gamma_l^{\mathrm{pre}}\, \tilde{\mathbf{x}}_l^{(g)} \boldsymbol{\varphi}_{l,g}^{\mathrm{pre}} + \mathbf{b}_{l,g}^{\mathrm{pre}}\right), \qquad \boldsymbol{\varphi}_{l,g}^{\mathrm{pre}} \in \mathbb{R}^{nd\times n}. \tag{6}$$

Here, γ_l^{pre} plays the same gate-scaling role as α_l^{pre} in the mHC parameterization.

Each group then forms its branch input using its own stream mixture,

$$\mathbf{u}_l^{(g)} = \sum_{s=1}^{n} \alpha_{l,s}^{(g)} \mathbf{x}_{l,s}^{(g)}, \qquad \mathbf{u}_l = \mathrm{Concat}\left(\mathbf{u}_l^{(1)}, \ldots, \mathbf{u}_l^{(G)}\right). \tag{7}$$

This replaces the single feature-shared routing pattern in mHC with G group-specific patterns, allowing different feature subspaces to access different mixtures of the residual streams while preserving the C-dimensional input to the transformation block.

Choice of group number. Unless otherwise stated, we set $G = n$, coupling the granularity of feature-wise routing to the number of residual streams. Each group then contains $d = C/n$ features from every residual stream, so its routing context remains $nd = C$ dimensional. As n increases, the number of independently routed feature groups therefore grows proportionally, while the input dimension of each routing generator remains fixed at C. This choice yields n independently parameterized stream-mixture patterns at the transformation readout, aligning its routing multiplicity with the n output-specific patterns in $\mathcal{H}_l^{\mathrm{res}}$ without introducing an additional architectural hyperparameter. Our ablations further support this choice, with $G = n$ achieving the best performance among the evaluated group configurations.

3.2 Low-Rank Directional Write-Back

At the write-back interface of the transformation pathway, mHC broadcasts a single transformed feature $\mathbf{h}_l$ to all residual streams. Denoting the s-th stream-wise write coefficient by $\beta_{l,s}$, the update is modulated only by this scalar, as formalized in Eq. 5. Consequently, increasing the number of residual streams introduces additional write coefficients without expanding the directional span of the branch updates. Rather than replicating the $O(C^2)$ attention or FFN transformation for every stream, we retain a shared nonlinear branch and augment its output with a lightweight stream-specific low-rank correction.

For each residual stream s at layer l, we introduce $A_{l,s} \in \mathbb{R}^{C\times r}$ and $B_{l,s} \in \mathbb{R}^{r\times C}$, with $r \ll C$, and define

$$\boldsymbol{\delta}_{l,s} = \mathrm{RMSNorm}_r\left(\mathbf{h}_l A_{l,s}\right) B_{l,s}, \tag{8}$$

where the parameter-free RMS normalization is applied over the rank-r bottleneck. The resulting write-back is

$$\boxed{\Delta\mathbf{x}_{l,s} = \beta_{l,s}\left(\mathbf{h}_l + \boldsymbol{\delta}_{l,s}\right)} \tag{9}$$

so that $\beta_{l,s}$ retains its role as the stream-wise write gate, while $\boldsymbol{\delta}_{l,s}$ introduces stream-dependent feature-space directions.

This augmentation relaxes the rank-one constraint of the original write-back. Unlike xHC (Zhang et al., 2026a), which applies Gram–Schmidt orthogonalization to its temporally augmented write-back components, we impose no explicit orthogonality constraint on the stream-specific corrections. This allows $\boldsymbol{\delta}_{l,s}$ to introduce additional feature-space directions while adaptively modulating the original feature contribution.

Mid-RMSNorm Parameterization. Normalizing within the rank-r bottleneck makes the directional path approximately invariant to positive rescaling of the down-projection. For any $c > 0$, up to the stabilizing ϵ term,

$$\mathrm{RMSNorm}_r\left(\mathbf{h}_l(cA_{l,s})\right) \approx \mathrm{RMSNorm}_r\left(\mathbf{h}_l A_{l,s}\right). \tag{10}$$

With the intermediate normalization, $A_{l,s}$ determines the low-rank representation independently of its overall scale, while $B_{l,s}$ governs how this normalized representation is mapped back to the hidden space. This separation decouples the roles of the down- and up-projections, removing an otherwise unconstrained scaling degree of freedom without introducing additional affine parameters.

3.3 Parameter Efficiency

Despite introducing additional flexibility at both transformation interfaces, uHC remains lightweight by only adding small routing and low-rank correction modules without replicating the nonlinear transformation blocks.

Group-Wise Feature Readout. A direct implementation that predicts routing coefficients for all feature groups from the full residual state requires $O(Gn^2C)$ parameters. Instead, uHC restricts each routing generator to its corresponding feature group. Since each group contains $d = C/G$ features from all residual streams, the total parameter count is

$$N_{\text{group}} = G(nd)n = n^2C, \tag{11}$$

which matches the projection parameter budget of the original mHC readout and remains independent of the number of feature groups.

Low-Rank Directional Write-Back. Instead of introducing stream-specific dense transformations with $O(nC^2)$ parameters, uHC adopts lightweight low-rank corrections for each residual stream. Each stream requires two projections of size $C \times r$ and $r \times C$, resulting in

$$N_{\text{dir}} = 2nCr \tag{12}$$

additional parameters per layer. With $r \ll C$, uHC introduces stream-specific update directions while maintaining the efficiency and scalability of HC.

4 Experiments

4.1 Experimental Setup

Training configuration. Our experiments are implemented on top of the mHC-Lite codebase (Yang & Gao, 2026), which is built upon nanoGPT (Karpathy, 2022). Following mHC-Lite, we use the same model sizes and network depths while adopting longer training schedules to ensure sufficient convergence. We consider three model scales: S (6 layers, $\sim$45M parameters, 2.62B training tokens), M (12 layers, $\sim$0.12B parameters, 3.54B training tokens), and L (24 layers, $\sim$0.36B parameters, 7.86B training tokens). All models are trained on OpenWebText with a context length of 1024 using two NVIDIA A800 GPUs. Unless otherwise specified, we set the expansion rate to $n = 4$ and the write-back rank to $r = 8$. For each layer l, we initialize the group-wise readout by replicating the mHC one-hot readout across feature groups, $\mathbf{H}_l^{\text{pre}} = \mathbf{1}_G(\mathbf{h}_l^{\text{pre}})^\top$, so that the initial routing is functionally equivalent to mHC. Complete training configurations and initialization details are provided in Appendix A.

Evaluation protocol. For each run, we evaluate the checkpoint with the lowest OpenWebText validation loss. We report perplexity on the OpenWebText validation split (Gokaslan et al., 2019) and the WikiText-103 test split (Merity et al., 2017), together with zero-shot perplexity on LAMBADA (Paperno et al., 2016). We further evaluate cross-domain generalization on five corpora from Paloma (Magnusson et al., 2024): C4, Dolma v1.5, Falcon RefinedWeb, RedPajama, and WikiText-103. Unless otherwise stated, mechanistic analyses are conducted on the L-scale checkpoints using 64 fixed 1024-token OpenWebText sequences. Detailed evaluation settings are provided in Appendix C.1.

4.2 Main Results

Pretraining performance. Table 1 reports validation loss and perplexity across the three model scales, with uHC achieving the lowest validation loss among the evaluated HC variants at every scale. It also improves most perplexity metrics at the S and M scales. At the L scale, the advantage becomes most pronounced: uHC maintains a lower late-stage training loss, reduces validation loss from 2.9470 for mHC to 2.8188, and achieves the best result on every reported metric. These results show that the pretraining gains persist across scales and become increasingly consistent as model size grows.

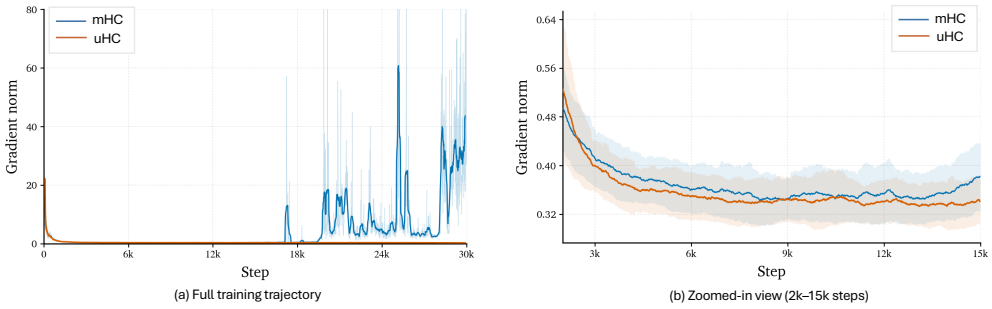

Figure 2: **Gradient-norm dynamics during training.** (a) Full training trajectory with 200-step moving averages shown as solid curves and raw gradient norms shown in lighter colors. (b) Zoomed-in view over steps 2k–15k using 1000-step moving averages; shaded regions indicate the within-window standard deviation.

Table 1: Validation loss and perplexity comparison across model scales under scale-specific training budgets. Lower is better for all metrics.

Scale	Method	Val Loss ↓	Evaluation PPL ↓			Paloma Word PPL ↓				
			LAMBADA	WT103	OWT-val	C4	Dolma	Falcon	RedPajama	WikiText
S	Residual	3.4505	248.92	63.41	31.43	235.19	**562.14**	362.89	**2420.79**	128.29
	mHC	3.3805	160.11	**54.68**	29.57	229.50	587.02	377.77	2794.91	124.18
	mHC-lite	3.3766	160.77	56.36	29.47	224.47	590.88	362.67	2908.43	137.77
	uHC (Ours)	**3.3747**	**148.14**	54.98	**29.37**	**220.43**	594.47	**340.39**	2811.69	**121.27**
M	Residual	3.1681	94.18	44.04	23.80	175.47	429.60	261.71	1999.86	88.69
	mHC	3.0892	**58.18**	39.32	**21.86**	158.01	427.65	240.34	2075.42	83.17
	mHC-lite	3.0950	60.25	38.23	22.03	159.32	383.25	250.80	1667.72	80.91
	uHC (Ours)	**3.0814**	64.19	**34.74**	**21.86**	**147.90**	**360.35**	**214.75**	**1464.05**	**77.73**
L	Residual	2.8536	30.20	29.53	17.17	112.67	269.99	165.83	1163.53	60.72
	mHC	2.9470	42.75	32.37	18.81	114.40	251.20	149.84	913.80	71.96
	mHC-lite	2.9029	34.51	30.82	18.02	112.13	266.25	156.28	1120.53	69.61
	uHC (Ours)	**2.8188**	**26.12**	**23.05**	**16.47**	**104.68**	**213.19**	**145.67**	**724.62**	**57.85**

Gradient stability. We next examine whether uncoupling the transformation pathway also affects gradient dynamics. Figure 2(a) shows the gradient-norm trajectories over the full training process. While mHC develops pronounced fluctuations and sharp spikes in the later stages of training, uHC maintains a substantially smoother trajectory throughout optimization. The zoomed-in view in Figure 2(b) further highlights the stable gradient dynamics of uHC, with the gradient norm remaining within a consistently narrower range. The result suggests that relaxing the coupling in the transformation pathway leads to more stable optimization at larger model scale.

4.3 Analysis of the Transformation Pathway

Our central hypothesis is that additional residual streams are most useful when the transformation pathway can both *read* from multiple streams and *write* stream-specific information back. We therefore analyze the two interfaces separately.

4.3.1 Stream Utilization in Readout

Layer-wise readout patterns. We first examine how the transformation pathway aggregates information from residual streams through the learned H_{pre} readout. Effective multi-stream utilization should avoid persistent concentration on a small subset of streams while preserving layer-dependent readout preferences. Different layers are therefore expected to develop distinct readout patterns rather than uniform aggregation.

For transformation type $b \in \{\text{Attn}, \text{MLP}\}$, we define the normalized readout weight of stream s at layer l as

$$p_{l,s}^{(b)} = \frac{\left|h_{l,s}^{\text{pre},(b)}\right|}{\sum_{j=1}^{n}\left|h_{l,j}^{\text{pre},(b)}\right|}. \tag{13}$$

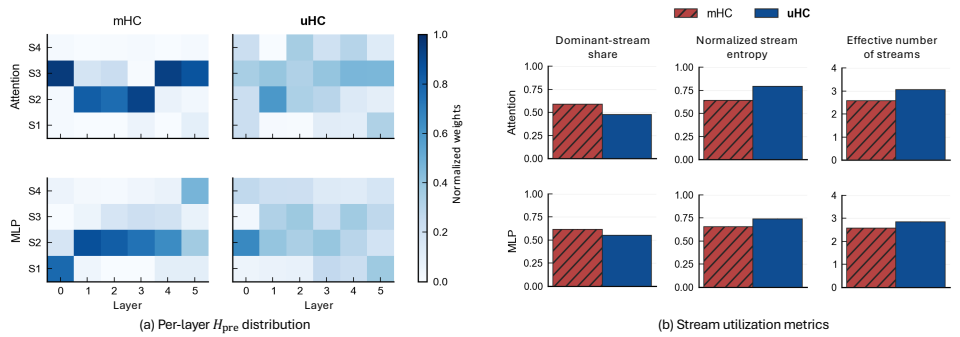

Figure 3: **Readout utilization in the transformation pathway.** (a) Layer-wise normalized H_{pre} weights over residual streams for mHC and uHC; darker colors indicate larger weights. (b) Token-level stream utilization statistics, including dominant-stream share, normalized stream entropy, and effective number of streams. uHC exhibits more distributed readout patterns across residual streams.

Figure 3(a) visualizes the normalized H_{pre} weights in the first six layers, where the single-stream dominance is most evident. Compared with mHC, uHC exhibits more distributed readout patterns, particularly in the attention branch. In mHC, shallow-layer H_{pre} weights often follow a nearly one-hot pattern, concentrating the readout contribution on a single residual stream while leaving others weakly utilized. Such sparse and imbalanced readout can weaken the effective gradient signal received by shallow-layer readout parameters, leading to an under-utilized transformation pathway. In contrast, uHC enables more balanced H_{pre} distributions, allowing multiple residual streams to participate consistently.

Stream utilization statistics. To further quantify this behavior beyond the layer-wise visualization, we measure the token-level distribution of readout contributions. Let $c_{q,s}$ denote the contribution magnitude of stream s in feature group q. We normalize the contributions within each group and average across groups:

$$p_s = \frac{1}{Q}\sum_{q=1}^{Q}\frac{c_{q,s}}{\sum_{j=1}^{n} c_{q,j}}, \qquad \sum_{s=1}^{n} p_s = 1, \tag{14}$$

where $Q = 1$ for mHC and $Q = 4$ for uHC. We summarize the resulting stream distribution using three statistics: the dominant-stream share, normalized entropy, and effective number of streams:

$$\begin{aligned} S_{\text{dom}}(\mathbf{p}) &= \max_s p_s, \\ H_{\text{norm}}(\mathbf{p}) &= -\frac{1}{\log n}\sum_s p_s \log p_s, \qquad N_{\text{eff}}(\mathbf{p}) = \exp\left(-\sum_s p_s \log p_s\right). \end{aligned} \tag{15}$$

Lower S_{dom} and higher H_{norm} and N_{eff} indicate more balanced stream utilization. As shown in Figure 3(b), uHC achieves more balanced readout contributions for both attention and MLP branches, suggesting that the transformation pathway can exploit a larger fraction of the available residual streams. This utilization pattern is related to the expert balancing problem in Mixture-of-Experts(MOE) models, where avoiding routing collapse and maintaining balanced expert utilization are important for preventing under-utilized experts (Lepikhin et al., 2021; Dai et al., 2024).

4.3.2 Stream-Specific Write-Back

Directional diversity of write updates. We examine whether uHC relaxes the directional constraint of the write interface. In mHC, all streams receive scalar-rescaled versions of the same transformed feature, resulting in collinear updates. uHC introduces stream-specific low-rank write directions, reducing the mean pairwise cosine similarity of write updates from 1.000 to 0.926 and increasing the effective rank from 1.000 to 1.293. This demonstrates that uHC breaks the strict collinearity of the shared write interface while preserving alignment among streams.

Layer-wise routing diversity. We further analyze whether uHC learns more differentiated write routing patterns across layers. We compute the mean pairwise Jensen–Shannon divergence between normalized $\mathcal{H}^{\text{post}}$ routing distributions across layers, normalized by $\log 2$. For the attention branch, uHC increases the diversity from 0.278 to 0.329, while the MLP branch remains similar (0.269 vs.

0.255). Together with the stream-utilization analysis, these results indicate that uHC enables more flexible layer-dependent write routing without requiring uniformly increased routing diversity.

4.4 Scaling with the Number of Residual Streams

We further examine whether additional residual streams translate into usable model capacity. At the M scale, we train mHC and uHC with $n \in 4, 8, 16$ under otherwise identical configurations. As shown in Table 2, uHC exhibits stronger scaling with the number of streams. From $n = 4$ to $n = 16$, uHC reduces validation loss by 0.0067, LAMBADA perplexity by 3.63, and OWT-val perplexity by 0.44, compared with 0.0030, 0.36, and 0.12 for mHC, respectively. The improvement also remains pronounced from $n = 8$ to $n = 16$, suggesting that uncoupled transformation interfaces make more effective use of additional residual streams.

Table 2: Scaling with the number of residual streams at the M scale. Parenthesized values denote changes from $n = 4$.

Method	n	Val Loss ↓	LAMBADA ↓	WT103 ↓	OWT-val ↓
mHC	4	3.0892	58.18	39.32	21.86
	8	$3.0869_{(-0.0023)}$	$58.02_{(-0.16)}$	$38.34_{(-0.98)}$	$21.86_{(-0.00)}$
	16	$3.0862_{(-0.0030)}$	$57.82_{(-0.36)}$	$37.85_{(-1.47)}$	$21.74_{(-0.12)}$
uHC	4	3.0814	64.19	34.74	21.86
	8	$3.0785_{(-0.0029)}$	$62.52_{(-1.67)}$	$34.54_{(-0.20)}$	$21.69_{(-0.17)}$
	16	$3.0747_{(-0.0067)}$	$60.56_{(-3.63)}$	$34.17_{(-0.57)}$	$21.42_{(-0.44)}$

4.5 Ablation Studies

We ablate the two components of uHC: group-wise readout and low-rank write-back. As shown in Table 3, each component consistently improves over mHC, while combining them achieves the best validation loss and perplexity performance. The larger gain from low-rank write-back highlights the importance of relaxing the shared write direction, while the improvement from group-wise readout demonstrates the benefit of more flexible stream aggregation before transformation.

Table 3: Component ablation of group-wise readout and low-rank write-back. Lower is better for all metrics.

Method	**Val Loss ↓**	**Evaluation PPL ↓**			**Paloma Word PPL ↓**				
		LAMBADA	**WT103**	**OWT-val**	**C4**	**Dolma**	**Falcon**	**RedPajama**	**WikiText**
mHC	2.9470	42.75	32.37	18.81	114.40	251.20	149.84	913.80	71.96
+ Group Readout	2.8843	<u>33.68</u>	30.12	<u>17.29</u>	110.04	262.91	151.82	<u>820.47</u>	69.24
+ Low-Rank Write-Back	<u>2.8629</u>	34.52	<u>27.58</u>	18.05	**103.26**	<u>238.18</u>	<u>149.71</u>	865.09	<u>62.14</u>
uHC (Both)	**2.8188**	**26.12**	**23.05**	**16.47**	<u>104.68</u>	**213.19**	**145.67**	**724.62**	**57.85**

5 Conclusion

In this work, we revisit multi-stream residual architectures from the transformation pathway and identify coupled readout and write-back as a structural limitation of existing Hyper-Connections. We introduce uHC, which enables feature groups to access different residual-stream mixtures and each stream to receive a distinct write-back direction. Across multiple model scales, uHC consistently improves language-modeling performance and optimization stability, with greater improvements as the number of residual streams increases. Our analysis further reveals more diverse readout and write-back patterns, suggesting that additional residual streams are better utilized when their interaction with the transformation branch is made more flexible. Due to resource constraints, our experiments have not yet been extended to larger-scale models. Future work can further explore the joint design of residual-stream propagation and transformation-pathway interaction to better support architectural scaling.

REFERENCES

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 1352–1361. PMLR, 2021.

Damai Dai et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *ICLR*, 2024.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Haoqiang Guo, Xuyi Chen, Bo Ke, Yishu Lei, Ziyang Xu, Shikun Feng, Ximen, and Wenhan Luo. oHC: Orthogonal hyper-connections on SO(4) via quaternions, 2026. URL `https://arxiv.org/abs/2609.02672`.

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. doi: 10.1109/CVPR.2016.90.

Mike Heddes, Adel Javanmard, Kyriakos Axiotis, Gang Fu, Mohammadhossein Bateni, and Vahab Mirrokni. DeepCrossAttention: Supercharging transformer residual connections. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 22881–22903. PMLR, 2025.

Andrej Karpathy. nanogpt. `https://github.com/karpathy/nanoGPT`, 2022. GitHub repository.

Kimi Team, Guangyu Chen, Yu Zhang, Jianlin Su, Weixin Xu, Siyuan Pan, Yaoyu Wang, Yucheng Wang, Guanduo Chen, Bohong Yin, Yutian Chen, Junjie Yan, Ming Wei, Y. Zhang, Fanqing Meng, Chao Hong, Xiaotong Xie, Shaowei Liu, Enzhe Lu, Yunpeng Tai, Yanru Chen, Xin Men, Haiqing Guo, Y. Charles, Haoyu Lu, Lin Sui, Jinguo Zhu, Zaida Zhou, Weiran He, Weixiao Huang, Xinran Xu, Yuzhi Wang, Guokun Lai, Yulun Du, Yuxin Wu, Zhilin Yang, and Xinyu Zhou. Attention residuals, 2026. URL `https://arxiv.org/abs/2603.15031`.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *ICLR*, 2021.

Zhaoyi Liu, Haichuan Zhang, and Ang Li. Beyond the birkhoff polytope: Spectral-sphere-constrained hyper-connections, 2026. URL `https://arxiv.org/abs/2603.20896`.

Cheng Luo, Zefan Cai, and Junjie Hu. Delta attention residuals, 2026. URL `https://arxiv.org/abs/2605.18855`.

Ian Magnusson, Akshita Bhagia, Valentin Hofmann, Luca Soldaini, Ananya Harsh Jha, Oyvind Tafjord, Dustin Schwenk, Evan Pete Walsh, Yanai Elazar, Kyle Lo, Dirk Groeneveld, Iz Beltagy, Hannaneh Hajishirzi, Noah A. Smith, Kyle Richardson, and Jesse Dodge. Paloma: A benchmark for evaluating language model fit. In *Advances in Neural Information Processing Systems*, 2024.

Brian Mak and Jeffrey Flanigan. Residual matrix transformers: Scaling the size of the residual stream. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 42712–42729. PMLR, 2025.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=Byj72udxe`.

Giyeong Oh, Woohyun Cho, Siyeol Kim, Suhwan Choi, and Youngjae Yu. Revisiting residual connections: Orthogonal updates for stable and efficient deep networks. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2409.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1525–1534, 2016. doi: 10.18653/v1/P16-1144.

Biswa Sengupta, Jinhua Wang, and Leo Brunswic. JPmHC dynamical isometry via orthogonal hyper-connections, 2026. URL https://arxiv.org/abs/2602.18308.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 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, 2024. doi: 10.1109/TPAMI.2024.3386927.

Da Xiao, Qingye Meng, Shengping Li, and Xingyuan Yuan. MUDDFormer: Breaking residual bottlenecks in transformers via multiway dynamic dense connections. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 68440–68458. PMLR, 2025.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Kuai Yu, Liang Zhao, Shangyan Zhou, Zhean Xu, Zhengyan Zhang, Wangding Zeng, Shengding Hu, Yuqing Wang, Jingyang Yuan, Lean Wang, and Wenfeng Liang. mHC: Manifold-constrained hyper-connections, 2026. URL https://arxiv.org/abs/2512.24880.

Yongyi Yang and Jianyang Gao. mHC-lite: You don't need 20 sinkhorn–knopp iterations, 2026. URL https://arxiv.org/abs/2601.05732.

Xiangdong Zhang, Xiaohan Qin, Sunan Zou, Tuo Dai, Xiaoming Shi, Huaijin Wu, Yebin Yang, Zhuo Xia, Shaofeng Zhang, Lin Yao, Yuliang Liu, Yu Cheng, and Junchi Yan. xHC: Expanded hyper-connections, 2026a. URL https://arxiv.org/abs/2607.14530.

Yifan Zhang, Yifeng Liu, Mengdi Wang, and Quanquan Gu. Deep delta learning, 2026b. URL https://arxiv.org/abs/2601.00417.

Wuyang Zhou, Yuxuan Gu, Giorgos Iacovides, and Danilo Mandic. KromHC: Manifold-constrained hyper-connections with kronecker-product residual matrices. In *Forty-third International Conference on Machine Learning*, 2026.

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, 2025b. URL https://arxiv.org/abs/2503.14125.

A Model and Training Configurations

The main experiments are conducted at three model scales, with the corresponding architecture and training configurations summarized in Table 4.

Table 4: Model architecture and training configurations across different scales. The learning rate is linearly warmed up to the peak value, followed by a constant phase when applicable and cosine decay to the minimum value.

Configuration	**S**	**M**	**L**
Layers	6	12	24
Heads	8	12	16
Hidden dimension	512	768	1024
Parameters	46.26 M	127.69 M	363.44 M
Batch size	32	24	8
Gradient accumulation steps	8	8	16
Training steps	10,000	18,000	30,000
Warmup steps	200	200	500
Cosine decay start step	200	200	25,500
Peak learning rate	1×10^{-3}	6×10^{-4}	3×10^{-4}
Minimum learning rate	1×10^{-4}	6×10^{-5}	3×10^{-5}
Total training tokens	2.62B	3.54B	7.86B

B Related Work

Residual Connections. Residual connections have become a fundamental component of deep neural networks and Transformers (He et al., 2016; Vaswani et al., 2017). Early work mainly improved optimization by controlling residual-branch magnitude and gradient propagation (Bachlechner et al., 2021; Wang et al., 2024), while recent studies increasingly treat the residual state and cross-layer information flow as architectural design choices. Representative examples include learned aggregation across depths, matrix-valued residual states, and constrained residual updates (Xiao et al., 2025; Heddes et al., 2025; Mak & Flanigan, 2025; Oh et al., 2025; Kimi Team et al., 2026; Luo et al., 2026; Zhang et al., 2026b). Complementary to these single-stream or depth-wise redesigns, Hyper-Connections (HC) expand the residual pathway into multiple persistent streams with learned routing among them (Zhu et al., 2025a).

Hyper-Connections. Hyper-Connections maintain multiple residual streams and use learned read, residual-mixing, and write-back maps to communicate with each transformation branch (Zhu et al., 2025a). Subsequent work has primarily improved the stability, efficiency, and expressivity of this multi-stream routing. Frac-Connections reduces the memory cost of stream expansion (Zhu et al., 2025b), while mHC and its variants impose structured constraints on the residual mixer to improve conditioning and propagation stability (Xie et al., 2026; Yang & Gao, 2026; Zhou et al., 2026; Sengupta et al., 2026; Liu et al., 2026; Guo et al., 2026). xHC instead enriches the write-back interface to promote more diverse residual updates (Zhang et al., 2026a). However, the transformation pathway remains comparatively constrained: readout typically uses a shared stream aggregation across feature channels, while write-back remains strongly coupled across streams. Jointly improving these two interfaces with lightweight routing therefore remains underexplored.

C Appendix

C.1 Evaluation Details

We use the same evaluation protocol for all compared methods. Models are evaluated in evaluation mode and in FP32, using the GPT-2 BPE tokenizer implemented with `tiktoken`.

Checkpoint selection. During training, validation loss is evaluated every 500 optimization steps. At each evaluation, cross-entropy is averaged over 200 randomly sampled mini-batches of 1024-token sequences from the OpenWebText validation split. We retain the checkpoint with the lowest estimated validation loss for downstream evaluation.

OpenWebText. For final evaluation, we recompute perplexity deterministically over all complete non-overlapping 1024-token windows of the OpenWebText validation split. Cross-entropy is accumulated over all evaluated tokens, and perplexity is computed as the exponential of the resulting average cross-entropy.

WikiText-103. We evaluate on the `wikitext-103-raw-v1` test split. The corpus is tokenized as a continuous sequence and evaluated using rolling windows of length 1024 with a stride of 512. Only newly introduced tokens in each overlapping window contribute to the reported perplexity.

LAMBADA. We evaluate LAMBADA zero-shot using the `lambada_openai` task from the EleutherAI LM Evaluation Harness, with `num_fewshot=0`. The harness is interfaced with our models through a custom adapter using the same GPT-2 tokenizer as during training.

Paloma. We evaluate C4, Dolma v1.5, Falcon RefinedWeb, RedPajama, and WikiText-103 using the corresponding official Paloma tasks in the EleutherAI LM Evaluation Harness. We use the harness's `loglikelihood_rolling` protocol and report its word-level perplexity for each corpus.

Mechanistic analyses. Unless otherwise specified, mechanistic analyses are performed on the L-scale models using the same 64 fixed, non-overlapping OpenWebText sequences of length 1024 for all compared methods. Routing, stream-state, write-back, and gradient statistics are collected at both attention and MLP hyper-connection sites and aggregated over the evaluated tokens.

C.2 Ablation

Normalization placement. We further compare NoNorm, PreNorm, MidNorm, and PostNorm in the low-rank write branch. MidNorm achieves the lowest validation loss (**2.8188**, compared with **2.9483**, **2.8871**, and **2.9147** for NoNorm, PreNorm, and PostNorm, respectively). This result is consistent with the role of intermediate normalization discussed in Sec. 3.2, which removes the global scale degree of freedom of the down-projection while leaving the up-projection free to control the returned correction.

C.3 Efficient Implementation of uHC

uHC introduces additional routing operations for group-wise read and stream-specific write. We therefore apply two execution-level optimizations that preserve the mathematical definition of the model.

Joint Gate Projection. Several routing variables are predicted from the same normalized residual representation. A direct implementation evaluates these projections independently:

$$P_\alpha = ZW_\alpha, \qquad P_{\mathrm{grp}} = ZD, \qquad P_\beta = ZW_\beta, \tag{16}$$

where $Z \in \mathbb{R}^{M\times nC}$ denotes the normalized input and D is the dense representation of the compact group-read projection. Since these projections share the same input, we concatenate their output dimensions and evaluate them using a single matrix multiplication:

$$W_J = [W_\alpha \mid D \mid W_\beta], \qquad P = ZW_J, \tag{17}$$

followed by splitting P into the corresponding routing logits. This preserves the model parameters and routing equations while reducing the execution overhead of multiple small projections. Under the matched L-scale benchmark, joint projection reduces the uHC training-microstep latency from 313.30 ms to 293.65 ms, corresponding to a 6.27% reduction.

Selective Width Recomputation. The hyper-connection width path computes the branch input, residual mixing, and write gates:

$$(u, R, \beta) = \mathcal{W}(X; \theta). \tag{18}$$

To reduce activation memory, we recompute $\mathcal{W}$ during the backward pass instead of retaining its intermediate activations. The recomputation is restricted to the hyper-connection width path and does not include the attention or FFN transformations.

This provides a direct memory–compute trade-off. Under the same L-scale setting, selective width recomputation reduces peak allocated memory from 26.24 GiB to 16.16 GiB, corresponding to a

38.4% reduction. The training-microstep latency increases from 272.47 ms to 310.21 ms, an overhead of 13.85%.

Table 5: Execution-level optimizations for uHC on the L-scale model. Each row is evaluated against its corresponding matched baseline.

Optimization	Baseline	Optimized	Change
Joint projection	313.30 ms	293.65 ms	-6.27% latency
Width recomputation	26.24 GiB	16.16 GiB	-38.4% memory

Thus, joint projection improves execution efficiency, whereas selective width recomputation provides a memory-efficient configuration at the cost of additional computation. Both optimizations preserve the mathematical definition of uHC.

C.4 Common Null Directions under Group-Wise Readout

We provide a complementary null-space view of the group-wise readout defined in Sec. 3.1. Consider a fixed token and layer, and stack the routing vectors of the G feature groups as

$$\mathbf{A} = \begin{bmatrix} \mathbf{h}_1^{\mathrm{pre}\top} \\ \vdots \\ \mathbf{h}_G^{\mathrm{pre}\top} \end{bmatrix} \in \mathbb{R}^{G\times n}. \tag{19}$$

A stream-space direction $\mathbf{v} \in \mathbb{R}^n$ is invisible to every feature group if and only if

$$\mathbf{A}\mathbf{v} = \mathbf{0}. \tag{20}$$

Hence, the dimension of the stream subspace simultaneously discarded by all groups is

$$\dim \ker(\mathbf{A}) = n - \mathrm{rank}(\mathbf{A}). \tag{21}$$

For the shared readout of mHC, every feature group uses the same routing vector, so $\mathrm{rank}(\mathbf{A}) = 1$ and there are necessarily $n-1$ common null directions. Group-wise readout allows the rows of $\mathbf{A}$ to differ, reducing this common null subspace as their rank increases. In particular, if $\mathrm{rank}(\mathbf{A}) = n$, no non-zero stream direction is simultaneously ignored by all feature groups.

This does not increase the total rank of the readout. Assuming non-zero routing vectors, both shared and group-wise readouts map $\mathbb{R}^{nC}$ to $\mathbb{R}^C$ with rank C and nullity $(n-1)C$. Group-wise routing therefore changes the distribution of the discarded stream directions across feature subspaces, rather than reducing the overall dimensional compression.

C.5 Probing Gradient Instability

Our training diagnostics reveal pronounced gradient-norm instability in mHC, whereas uHC exhibits substantially more stable gradients. We further probe the transformation pathway to identify structural differences associated with this behavior.

In mHC, the transformed feature written to residual stream s is

$$\mathbf{w}_s = \beta_s \mathbf{h}, \tag{22}$$

where the transformed feature $\mathbf{h} \in \mathbb{R}^C$ is shared across all streams and β_s is a stream-specific scalar gate. Therefore,

$$\mathbf{W} = [\mathbf{w}_1; \ldots; \mathbf{w}_n] = \boldsymbol{\beta}\mathbf{h}^\top, \qquad \mathrm{rank}(\mathbf{W}) \le 1, \tag{23}$$

such that all non-zero stream-wise writes are constrained to the same feature direction. The corresponding gradient with respect to the shared transformation feature is

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}} = \sum_{s=1}^{n} \beta_s \frac{\partial \mathcal{L}}{\partial \mathbf{w}_s}, \tag{24}$$

so gradients from different streams are aggregated through the same transformation feature without a stream-specific transformation before aggregation.

Table 6: Probing the transformation write interface. Write geometry is measured from forward activations, while update statistics are inferred from existing AdamW optimizer states.

Metric	mHC	uHC
Write pairwise cosine ↓	1.000	0.926
Write directional PR ↑	1.000	1.123
Dynamic-β maximum update share ↓	44.85%	31.19%
Dynamic-β effective streams ↑	3.175	3.909

uHC relaxes this constraint through stream-specific low-rank corrections,

$$\mathbf{w}_s = \beta_s \left(\mathbf{h} + \boldsymbol{\delta}_s \right), \tag{25}$$

which introduce non-identical write directions and stream-specific Jacobians in the backward pass.

Our activation probing confirms that these additional degrees of freedom are effectively used by the trained model. At the transformation write interface, the mean pairwise cosine similarity across streams decreases from exactly 1.000 in mHC to 0.926 in uHC, while the directional participation ratio increases from 1.000 to 1.123. Thus, uHC breaks the strict rank-one collinearity of the mHC write interface, although the resulting stream-wise writes remain substantially correlated.

A consistent difference is also observed in the optimizer-side update allocation. For the dynamic write-gate parameters, the largest-stream share of the inferred AdamW update decreases from 44.85% in mHC to 31.19% in uHC, while the effective number of updated streams increases from 3.175 to 3.909 out of four streams. In contrast, the corresponding concentration of the static write gates changes only marginally (60.60% versus 59.39%). The redistribution is therefore localized to the dynamic transformation interface rather than reflecting a uniform change across all parameters. Since these quantities are reconstructed AdamW updates rather than raw gradients, we treat them as supporting evidence of less concentrated optimization rather than a direct measurement of gradient stability.

Importantly, this behavior should not be interpreted as global decorrelation of the residual streams. The complete residual states do not exhibit consistently lower cross-stream similarity under uHC. The observed difference is therefore better localized to the transformation-path interface targeted by our design.

Taken together with the directly observed gradient-norm instability, these results suggest that the unstable optimization behavior of mHC is associated with strong cross-stream coupling at the transformation write interface. uHC removes the exact rank-one write constraint, realizes non-identical stream-specific write directions in trained models, and exhibits a markedly more balanced update allocation at the corresponding dynamic write gates. While this analysis does not establish the write constraint as the unique cause of gradient instability, it identifies a concrete structural difference that is consistent with the improved gradient stability observed during training.