STRUCTURED *m*HC: OBJECTIVE-GUIDED RESIDUAL MIXING OVER THE BIRKHOFF POLYTOPE

Anonymous authors
Paper under double-blind review

ABSTRACT

Manifold-Constrained Hyper-Connections (*m*HC) use Sinkhorn–Knopp normalization to approximate doubly stochastic residual mixers; exactly feasible mixers in the Birkhoff polytope compose non-expansively across depth. Doubly stochasticity, however, does not determine how information is exchanged: identity, uniform, and permutation mixers satisfy the same constraint while inducing different cross-stream interactions. We introduce structured *m*HC, which treats the objective used to select a mixer within the Birkhoff polytope as a design variable. We instantiate the framework with a token-dependent linear objective derived from learned mixer scores and compute its mixers with a fixed-depth, differentiable primal–dual method tailored to the row- and column-sum constraints. Under matched protocols and token budgets within each model scale, our method attains lower late-stage training loss than *m*HC-Sinkhorn in 64M dense and 198M-A64M MoE MiniMind-3 models. In a separate 7B comparison after 15B training tokens, it attains mean accuracy gains of 2.47, 2.42, and 0.59 percentage points over *m*HC-Sinkhorn across eight tasks under zero-, five-, and ten-shot evaluation.

1 INTRODUCTION

Residual connections let learned transformations accumulate while preserving a direct path for the hidden state across depth (He et al., 2016). In a standard Transformer, each attention and feed-forward sublayer adds its update to the same residual stream, which therefore serves as the persistent state passed from one sublayer to the next (Vaswani et al., 2017).

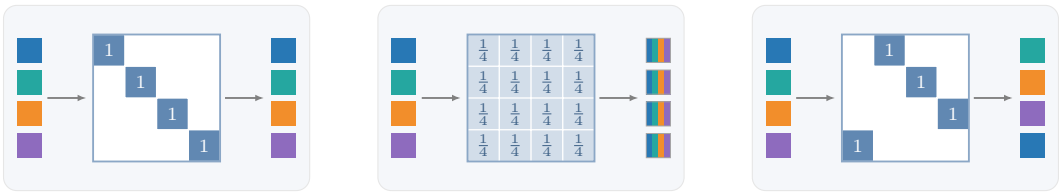
Hyper-Connections (HC) expand the single residual stream into n persistent streams: each layer learns how to read from them, write its output back to them, and mix them along the residual path (Zhu et al., 2025). Because this mixing recurs at every layer, the direct residual contribution passes through a product of mixer matrices that can amplify signals as depth increases if left unconstrained. Manifold-Constrained Hyper-Connections (*m*HC) address this risk by targeting doubly stochastic residual mixers, whose entries are nonnegative and whose rows and columns each sum to one (Xie et al., 2026). Such mixers belong to the Birkhoff polytope $\mathcal{B}_n$, which is closed under multiplication and have spectral norm at most one.

However, the Birkhoff constraint does not determine how residual information is exchanged among streams. The identity $\boldsymbol{I}_n$ passes each stream through unchanged, the uniform matrix $\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top$ replaces every stream with the mean of all streams, and a non-identity permutation matrix $\boldsymbol{P}_n$ reassigns whole streams without averaging them (see Figure 1). All three belong to $\mathcal{B}_n$, yet they induce qualitatively different communication patterns. Mixer selection within $\mathcal{B}_n$ is therefore an architectural choice separate from feasibility.

Standard *m*HC exponentiates an unconstrained residual-mixer matrix and applies Sinkhorn–Knopp normalization, which alternately rescales its rows and columns toward unit sums (Xie et al., 2026; Sinkhorn & Knopp, 1967). Empirical analyses find near-identity mixing beyond early layers in smaller *m*HC models (Alimaskina et al., 2026) and weak late-layer mixing in a large *m*HC model using Sinkhorn–Knopp normalization (Zhao et al., 2026).

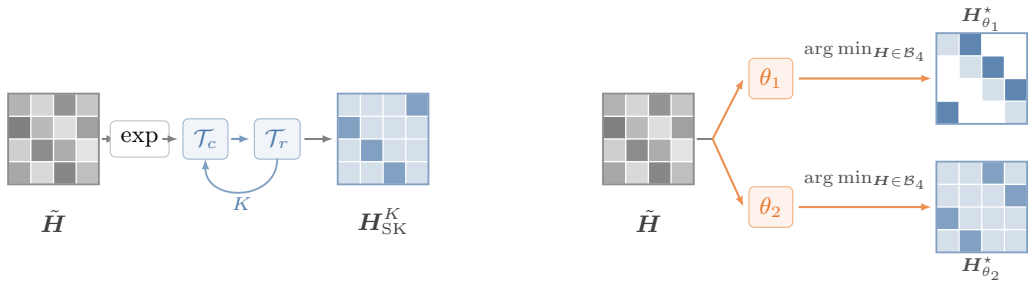
This motivates structured *m*HC, which retains $\mathcal{B}_n$ as the target feasible set while exposing the mixer-selection criterion as a design variable (Figure 2). For each layer l and token t, structured *m*HC defines an ideal target $\boldsymbol{H}^\star_{l,t} \in \arg\min_{\boldsymbol{H}\in\mathcal{B}_n} \theta_{l,t}(\boldsymbol{H})$, where $\theta_{l,t}$ specifies which feasible interactions

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.



(a) Identity (b) Uniform (c) Permutation

Figure 1: **Distinct stream interactions within $\mathcal{B}_4$.** Colored blocks show the input streams (left) and output streams (right): (a) preserves each stream, (b) averages all four inputs into every output, and (c) reassigns whole streams. Blank matrix cells have zero weight; all three mixers are doubly stochastic and have spectral norm one.



(a) Standard *m*HC (b) Structured *m*HC

Figure 2: **The Birkhoff constraint leaves mixer selection open.** (a) Standard *m*HC applies K rounds of alternating row and column normalization to move exponentiated scores toward the doubly stochastic constraints. (b) Structured *m*HC makes the ideal selection criterion explicit: θ_1 and θ_2 illustrate how different objectives can favor different feasible stream couplings for the same unconstrained mixer.

to favor. Whereas Sinkhorn–Knopp normalization corresponds to a fixed entropic projection, this formulation permits different objectives over the same target feasible set and we study a learned linear objective as one instance.

We solve the resulting optimization problem with a fixed number of unrolled primal–dual updates (Chambolle & Pock, 2011; He & Yuan, 2012; He et al., 2022). Exploiting the row- and column-sum structure reduces each update to elementwise operations, reductions, and broadcasts, so token-specific mixers can be computed in parallel within each residual block.

We evaluate our method against *m*HC-Sinkhorn in matched MiniMind-3 (Gong, 2024) pre-training runs with a 64M dense model and a 198M-total/64M-active MoE model. The proposed method attains lower late-stage training loss at both scales. In a separate 7B dense comparison after 15B training tokens, it also attains higher mean accuracy across eight downstream tasks under zero-, five-, and ten-shot evaluation.

Our contributions are threefold:

- We formulate structured *m*HC as constrained optimization over the Birkhoff polytope, separating the stability constraint from an explicit objective that determines which feasible stream interactions are favored.
- We instantiate this formulation with learned linear costs and derive a fixed-depth, differentiable primal–dual method for the resulting constrained problem. By exploiting the row- and column-sum structure, its updates can be computed in parallel across tokens.
- We evaluate our method in matched 64M dense and 198M-A64M MoE pre-training runs and in a separate 7B dense downstream comparison. It attains lower late-stage training loss in both pre-training settings and higher mean accuracy than *m*HC-Sinkhorn across eight downstream tasks and all three shot settings.

2 Background and Related Work

2.1 Hyper-Connections

We write the update for a single token and omit its token index. Let C denote the hidden width and n the number of residual streams. A standard residual block carries a state $\boldsymbol{x}_l \in \mathbb{R}^{1\times C}$ from block l to block $l+1$ through

$$\boldsymbol{x}_{l+1} = \boldsymbol{x}_l + \mathcal{F}(\boldsymbol{x}_l, \boldsymbol{W}_l), \tag{1}$$

where $\mathcal{F}(\cdot, \boldsymbol{W}_l)$ is the learned block transformation. Hyper-Connections (HC) (Zhu et al., 2025) instead maintain an n-stream state $\boldsymbol{X}_l \in \mathbb{R}^{n\times C}$. A read map $\boldsymbol{H}_l^{\text{pre}} \in \mathbb{R}^{1\times n}$, a write map $\boldsymbol{H}_l^{\text{post}} \in \mathbb{R}^{1\times n}$, and a residual mixer $\boldsymbol{H}_l^{\text{res}} \in \mathbb{R}^{n\times n}$ define its update:

$$\boldsymbol{X}_{l+1} = \boldsymbol{H}_l^{\text{res}} \boldsymbol{X}_l + (\boldsymbol{H}_l^{\text{post}})^\top \mathcal{F}(\boldsymbol{H}_l^{\text{pre}} \boldsymbol{X}_l, \boldsymbol{W}_l). \tag{2}$$

The read map forms the input to $\mathcal{F}$, the write map distributes its output across the streams, and the residual mixer propagates and recombines the incoming state along the direct residual path.

2.2 Manifold-Constrained Hyper-Connections

As blocks are stacked, the residual mixers in the first term of the HC update compose. From block l to block L, their cumulative action is

$$\Pi_{L:l}^{\text{res}} \coloneqq \boldsymbol{H}_{L-1}^{\text{res}} \boldsymbol{H}_{L-2}^{\text{res}} \cdots \boldsymbol{H}_l^{\text{res}}. \tag{3}$$

If the individual mixers are unconstrained, this product can amplify signals along the direct residual path as depth increases. Manifold-Constrained Hyper-Connections (*m*HC) (Xie et al., 2026) control this gain by requiring each residual mixer to lie in the Birkhoff polytope of doubly stochastic matrices,

$$\mathcal{B}_n \coloneqq \left\{ \boldsymbol{H} \in \mathbb{R}^{n\times n} \;\middle|\; \boldsymbol{H}\mathbf{1}_n = \mathbf{1}_n,\ \mathbf{1}_n^\top \boldsymbol{H} = \mathbf{1}_n^\top,\ \boldsymbol{H} \geq 0 \right\}. \tag{4}$$

Here $\mathbf{1}_n$ is the all-ones vector and $\boldsymbol{H} \geq 0$ holds elementwise. Because $\mathcal{B}_n$ is closed under multiplication and $\|\boldsymbol{H}\|_2 \leq \sqrt{\|\boldsymbol{H}\|_1 \|\boldsymbol{H}\|_\infty} = 1$ for every $\boldsymbol{H} \in \mathcal{B}_n$, the direct residual product satisfies $\|\Pi_{L:l}^{\text{res}}\|_2 \leq 1$ when every mixer obeys the constraint exactly.

Standard *m*HC constructs its residual mixer using Sinkhorn–Knopp normalization (Xie et al., 2026; Sinkhorn & Knopp, 1967). Given an unconstrained score matrix $\tilde{\boldsymbol{H}}_l^{\text{res}}$, it exponentiates the entries to obtain $\boldsymbol{M}^{(0)} = \exp(\tilde{\boldsymbol{H}}_l^{\text{res}})$ and alternates column and row normalization:

$$\boldsymbol{M}^{(t)} = \mathcal{T}_r\Big(\mathcal{T}_c\Big(\boldsymbol{M}^{(t-1)}\Big)\Big), \tag{5}$$

where $\mathcal{T}_c$ and $\mathcal{T}_r$ divide each column and row by its current sum, respectively. For the strictly positive $\boldsymbol{M}^{(0)}$, repeated scaling converges to a matrix in $\mathcal{B}_n$. Equivalently, the limit is the generalized-KL projection of $\boldsymbol{M}^{(0)}$ onto $\mathcal{B}_n$, so Sinkhorn–Knopp normalization couples constraint enforcement to a fixed entropic selection criterion (Xie et al., 2026). A fixed iteration budget approximates that limit, and the realized mixer need not satisfy both sets of unit-sum constraints exactly.

2.3 Related Work

Recent *m*HC variants change how feasible mixers are constructed. *m*HC-lite (Yang & Gao, 2026) uses convex combinations of permutation matrices to obtain exact doubly stochasticity without repeated Sinkhorn–Knopp normalization. KromHC (Zhou et al., 2026) uses Kronecker products of smaller doubly stochastic matrices to retain exact feasibility with fewer parameters as the number of streams grows, while restricting the mixer to a structured subset of $\mathcal{B}_n$. Both methods modify the parameterization and computational path used to produce feasible mixers. Structured *m*HC is complementary: it begins from a specified objective over the target feasible set and derives a differentiable finite-step construction for that objective.

Empirical analyses of smaller *m*HC-lite models find near-identity residual mixers and concentrated read and write activity (Alimaskina et al., 2026). A separate analysis of a larger *m*HC model finds weak late-layer mixing while preserving distinct stream representations and depth-dependent read and write preferences (Zhao et al., 2026). Qwen3.8-Flash-Next (Qwen Team, 2026) and Hy4-preview (Tencent Hy Team, 2026) likewise use identity carryover along the direct residual path while retaining multi-stream read and write operations. Together, these findings reinforce that feasibility alone does not prescribe the amount or pattern of direct-path communication.

3 STRUCTURED mHC

3.1 OBJECTIVE-GUIDED MIXER OPTIMIZATION

Sinkhorn–Knopp normalization fixes an entropic criterion for selecting a mixer while iteratively approaching the doubly stochastic constraints. Structured mHC makes this criterion configurable while retaining $\mathcal{B}_n$ as the target feasible set. For each layer l and token t, an objective $\theta_{l,t} : \mathbb{R}^{n\times n} \to \mathbb{R}$ defines the ideal target mixer by

$$\boldsymbol{H}^{\star}_{l,t} \in \arg\min_{\boldsymbol{H}\in\mathcal{B}_n} \theta_{l,t}(\boldsymbol{H}). \tag{6}$$

An exact solution retains the non-expansive bound of Section 2.2, while $\theta_{l,t}$ determines which feasible stream interactions are preferred. For example, minimizing $-\operatorname{tr}(\boldsymbol{H})$ favors self-coupling, whereas minimizing $\|\boldsymbol{H}\|_F^2$ favors a more even distribution of weights.

Within the framework of Equation 6, we consider a linear objective that provides a direct way to turn learned, token-dependent scores into preferences over feasible mixers. Let $\tilde{\boldsymbol{H}}^{\mathrm{res}}_{l,t} \in \mathbb{R}^{n\times n}$ be the unconstrained residual-mixer matrix produced from learned parameters and the current token state. Its entry $(\tilde{\boldsymbol{H}}^{\mathrm{res}}_{l,t})_{ij}$ scores a coupling from input stream j to output stream i; we convert these scores into costs through

$$\boldsymbol{C}_{l,t} = -\rho\, \tilde{\boldsymbol{H}}^{\mathrm{res}}_{l,t}, \qquad \rho > 0. \tag{7}$$

Since $\rho > 0$, larger scores yield lower costs. For a candidate mixer $\boldsymbol{H}$, the resulting objective is

$$\theta_{l,t}(\boldsymbol{H}) = \langle \boldsymbol{C}_{l,t}, \boldsymbol{H}\rangle = \sum_{i,j=1}^{n} (\boldsymbol{C}_{l,t})_{ij}\boldsymbol{H}_{ij}. \tag{8}$$

Although the objective is a sum of pairwise costs, the Birkhoff constraints couple the corresponding weights: increasing $\boldsymbol{H}_{ij}$ reduces the unit weight available to other pairs sharing output stream i or input stream j. The costs therefore act jointly across the score matrix.

Proposition 1 (Assignment structure of the linear instance)**.** *For each layer l and token t, minimizing $\theta_{l,t}(\boldsymbol{H}) = \langle \boldsymbol{C}_{l,t}, \boldsymbol{H}\rangle$ over $\mathcal{B}_n$ is the linear-programming formulation of a minimum-cost bipartite assignment between input and output streams. Here $\boldsymbol{H}_{ij} \geq 0$ assigns weight to the coupling from input stream j to output stream i at cost $(\boldsymbol{C}_{l,t})_{ij}$, while the unit row and column sums require each output and input stream to carry unit total weight, respectively.*

3.2 PRIMAL–DUAL METHOD

As in Proposition 1, the cost matrix $\boldsymbol{C}_{l,t}$ defines an assignment problem for each layer l and token t. We apply K steps of the primal–dual method proposed in (Chambolle & Pock, 2011), whose convergence was proved in (He & Yuan, 2012) and later refined optimally in (He et al., 2022), and use the resulting $\boldsymbol{H}^{K}_{l,t}$ as the residual mixer. Unrolling these steps within the network allows the training loss to be differentiated through mixer construction to the learned scores that produce $\boldsymbol{C}_{l,t}$. We suppress the layer and token indices in the derivation below.

For a single mixer, let $\boldsymbol{h} = \operatorname{vec}(\boldsymbol{H})$ and $\boldsymbol{c} = \operatorname{vec}(\boldsymbol{C})$, and let $\boldsymbol{A} : \mathbb{R}^{n^2} \to \mathbb{R}^{2n}$ stack the row-sum and column-sum linear maps. The linear instance of Equation 6 then has the compact form

$$\min_{\boldsymbol{h}\geq 0} \boldsymbol{c}^{\top}\boldsymbol{h} \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{h} = \mathbf{1}_{2n}. \tag{9}$$

With column-wise vectorization defined by $\operatorname{vec}(\boldsymbol{H}) = (\boldsymbol{H}_{11}, \ldots, \boldsymbol{H}_{n1}, \boldsymbol{H}_{12}, \ldots, \boldsymbol{H}_{nn})^{\top}$, the row-sum and column-sum maps have the matrix representation

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{1}_n^{\top} \otimes \boldsymbol{I}_n \\ \boldsymbol{I}_n \otimes \mathbf{1}_n^{\top} \end{bmatrix} \in \mathbb{R}^{2n\times n^2}. \tag{10}$$

Here $\otimes$ denotes the Kronecker product. The upper block forms the row sums of $\boldsymbol{H}$, while the lower block forms its column sums.

To solve Equation 9, the primal–dual method introduces dual vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ for the row- and column-sum constraints, and writes $\boldsymbol{y} = [\boldsymbol{u}; \boldsymbol{v}]$. The equality-constrained problem then has the saddle formulation

$$\min_{\boldsymbol{h}\geq 0} \max_{\boldsymbol{y}\in\mathbb{R}^{2n}} \boldsymbol{c}^{\top}\boldsymbol{h} - \boldsymbol{y}^{\top}(\boldsymbol{A}\boldsymbol{h} - \mathbf{1}_{2n}). \tag{11}$$

Let $\Phi(\boldsymbol{h}, \boldsymbol{y})$ denote the saddle objective in Equation 11. With step sizes $1/r$ and $1/s$, where $r, s > 0$, the primal–dual method takes a proximal step in each variable, using full extrapolation of the new primal iterate (Chambolle & Pock, 2011; He & Yuan, 2012):

$$\begin{cases} \boldsymbol{h}^{k+1} = \arg\min\limits_{\boldsymbol{h} \geq 0} \left\{ \Phi(\boldsymbol{h}, \boldsymbol{y}^k) + \dfrac{r}{2} \|\boldsymbol{h} - \boldsymbol{h}^k\|_2^2 \right\}, \\ \boldsymbol{y}^{k+1} = \arg\max\limits_{\boldsymbol{y} \in \mathbb{R}^{2n}} \left\{ \Phi(2\boldsymbol{h}^{k+1} - \boldsymbol{h}^k, \boldsymbol{y}) - \dfrac{s}{2} \|\boldsymbol{y} - \boldsymbol{y}^k\|_2^2 \right\}. \end{cases} \tag{12}$$

The first minimization retains the nonnegative domain of the mixer, while the unconstrained maximization updates the multipliers for its row and column sums. Since Φ is affine in both variables, completing the square in the primal step and differentiating the dual objective give the explicit updates

$$\boldsymbol{h}^{k+1} = \left[\boldsymbol{h}^k + \frac{1}{r} \left(\boldsymbol{A}^\top \boldsymbol{y}^k - \boldsymbol{c} \right) \right]_+, \tag{13a}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k - \frac{1}{s} \left(\boldsymbol{A}(2\boldsymbol{h}^{k+1} - \boldsymbol{h}^k) - \mathbf{1}_{2n} \right), \tag{13b}$$

where $[\cdot]_+$ projects each entry onto the nonnegative reals. The cost vector enters the primal step through $-\boldsymbol{c}$, while the dual step responds to row- and column-sum errors evaluated at the extrapolated mixer $2\boldsymbol{h}^{k+1} - \boldsymbol{h}^k$.

The usual sufficient condition $rs > \lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A})$ controls the step sizes through the largest eigenvalue of the constraint operator. For the row- and column-sum map in Equation 10, $\boldsymbol{A}\boldsymbol{A}^\top$ has diagonal blocks $n\boldsymbol{I}_n$ and off-diagonal blocks $\mathbf{1}_n \mathbf{1}_n^\top$, while every column of $\boldsymbol{A}$ contains two unit entries. Consequently, its largest and average eigenvalues have different scaling with the number of streams:

$$\lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A}) = 2n, \qquad \bar{\lambda} := \frac{\operatorname{tr}(\boldsymbol{A}^\top \boldsymbol{A})}{n^2} = 2. \tag{14}$$

The largest-eigenvalue condition restricts the product of the step sizes to $(1/r)(1/s) < 1/(2n)$, which becomes smaller as n grows even though $\bar{\lambda}$ remains 2. Motivated by this gap, we follow the average-eigenvalue heuristic used for assignment problems by (He et al., 2022) and choose $rs = 2\bar{\lambda} = 4$. For our $n = 4$ experiments, this lies below the sufficient bound $rs > 8$ and is a heuristic for the fixed-depth computation.

To implement these updates without multiplying by the explicit matrix $\boldsymbol{A}$, we use its action on the primal and dual variables:

$$\boldsymbol{A}\boldsymbol{h} = \begin{bmatrix} \boldsymbol{H}\mathbf{1}_n \\ \boldsymbol{H}^\top \mathbf{1}_n \end{bmatrix}, \tag{15}$$

$$\operatorname{mat}\left(\boldsymbol{A}^\top \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} \right) = \boldsymbol{u}\mathbf{1}_n^\top + \mathbf{1}_n \boldsymbol{v}^\top. \tag{16}$$

Here mat reverses the column-wise vectorization. The first identity computes the row and column sums of $\boldsymbol{H}$; the second forms the adjoint correction at entry (i, j) by adding the row dual u_i and column dual v_j. Thus, applying $\boldsymbol{A}$ and $\boldsymbol{A}^\top$ requires only summation reductions and broadcast additions, without constructing the constraint matrix. These operations also explain the feedback in the updates: when the extrapolated sum of row i exceeds one, Equation 13b decreases u_i, which in turn reduces the next primal update across that row; columns receive the analogous correction. Together with elementwise scaling and projection onto nonnegative entries, these operations give the tensor-form iteration in Algorithm 1.

Each step of Algorithm 1 touches the n^2 mixer entries a constant number of times, giving $O(n^2)$ arithmetic per mixer. Because each token has its own cost matrix and primal–dual state, with no coupling between their updates, the same tensor operations can be batched over all tokens in a residual block. The row- and column-sum operations and dual corrections also carry over to other objectives with tractable primal proximal steps; the objective enters through the primal step. With a fixed K, the network uses the realized mixer $\boldsymbol{H}^K$, which is nonnegative after projection but need not have unit row and column sums exactly. The non-expansive bound in Section 2.2 therefore applies to the ideal constrained target, and is not automatically inherited by the finite-step output. We differentiate through this realized K-step computation for the linear objective in Section 3.3.

Algorithm 1 Tensor-form primal–dual updates for one structured mHC mixer

Require: Cost $\boldsymbol{C}$, nonnegative initialization $\boldsymbol{H}^0$, steps K, parameters $r, s > 0$

$\boldsymbol{u}^0 \leftarrow \mathbf{0}$, $\boldsymbol{v}^0 \leftarrow \mathbf{0}$
for $k = 0, \dots, K-1$ **do**
  $\boldsymbol{Q}^k \leftarrow \boldsymbol{H}^k + \frac{1}{r}(\boldsymbol{u}^k \mathbf{1}_n^\top + \mathbf{1}_n (\boldsymbol{v}^k)^\top - \boldsymbol{C})$
  $\boldsymbol{H}^{k+1} \leftarrow [\boldsymbol{Q}^k]_+$; $\boldsymbol{R}^k \leftarrow 2\boldsymbol{H}^{k+1} - \boldsymbol{H}^k$
  $\boldsymbol{u}^{k+1} \leftarrow \boldsymbol{u}^k - \frac{1}{s}(\boldsymbol{R}^k \mathbf{1}_n - \mathbf{1}_n)$
  $\boldsymbol{v}^{k+1} \leftarrow \boldsymbol{v}^k - \frac{1}{s}((\boldsymbol{R}^k)^\top \mathbf{1}_n - \mathbf{1}_n)$
end for
return $\boldsymbol{H}^K$

3.3 Differentiating Through the Updates

The residual block uses the mixer produced after K updates, so the relevant derivative is that of the finite-step map $\mathcal{F}_K : (\boldsymbol{C}, \boldsymbol{H}^0) \mapsto \boldsymbol{H}^K$, with r, s and the zero dual initialization fixed. For the linear objective, a unique exact optimizer remains unchanged under sufficiently small perturbations of $\boldsymbol{C}$ and therefore has zero derivative there. In contrast, $\mathcal{F}_K$ composes affine operations with entrywise projections and is therefore piecewise affine and differentiable almost everywhere. We derive how a scalar training loss $\mathcal{L}(\boldsymbol{H}^K)$ propagates through this map to the costs that encode the learned stream preferences.

The only nonlinear operation in Algorithm 1 is the nonnegative projection. Away from zero, its derivative is the elementwise mask $(\boldsymbol{D}^k)_{ij} = \mathbf{1}[(\boldsymbol{Q}^k)_{ij} > 0]$; at $(\boldsymbol{Q}^k)_{ij} = 0$, where the projection is nondifferentiable, we select the zero subgradient. For perturbations of the cost and the primal–dual state, one iteration then has the linearization

$$\begin{cases} \mathrm{d}\boldsymbol{Q}^k = \mathrm{d}\boldsymbol{H}^k + \dfrac{1}{r}\big(\mathrm{d}\boldsymbol{u}^k \mathbf{1}_n^\top + \mathbf{1}_n (\mathrm{d}\boldsymbol{v}^k)^\top - \mathrm{d}\boldsymbol{C}\big), \\ \mathrm{d}\boldsymbol{H}^{k+1} = \boldsymbol{D}^k \odot \mathrm{d}\boldsymbol{Q}^k, \\ \mathrm{d}\boldsymbol{R}^k = 2\,\mathrm{d}\boldsymbol{H}^{k+1} - \mathrm{d}\boldsymbol{H}^k, \\ \mathrm{d}\boldsymbol{u}^{k+1} = \mathrm{d}\boldsymbol{u}^k - \dfrac{1}{s}\,\mathrm{d}\boldsymbol{R}^k \mathbf{1}_n, \\ \mathrm{d}\boldsymbol{v}^{k+1} = \mathrm{d}\boldsymbol{v}^k - \dfrac{1}{s}\,(\mathrm{d}\boldsymbol{R}^k)^\top \mathbf{1}_n. \end{cases} \tag{17}$$

Taking the adjoint of this linearization gives the vector–Jacobian product needed for backpropagation. In particular, the row and column reductions obey

$$\langle \boldsymbol{a}, \boldsymbol{X}\mathbf{1}_n \rangle + \langle \boldsymbol{b}, \boldsymbol{X}^\top \mathbf{1}_n \rangle = \langle \boldsymbol{a}\mathbf{1}_n^\top + \mathbf{1}_n \boldsymbol{b}^\top, \boldsymbol{X} \rangle_F, \tag{18}$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius inner product. Thus the adjoint of a row- or column-sum reduction is a broadcast addition, as in Equation 16.

The terminal gradient is characterized by $\mathrm{d}\mathcal{L} = \langle \nabla_{\boldsymbol{H}^K} \mathcal{L}, \mathrm{d}\boldsymbol{H}^K \rangle_F$. Write $\bar{X} = \nabla_X \mathcal{L}$ for the adjoint of an intermediate variable accumulated from later iterations. We initialize $\bar{\boldsymbol{H}}^K = \nabla_{\boldsymbol{H}^K} \mathcal{L}$ and $\bar{\boldsymbol{u}}^K = \bar{\boldsymbol{v}}^K = \mathbf{0}$, since the loss depends on the final mixer rather than the final dual variables. At step $k = K-1, \dots, 0$, the two dual updates first contribute the following adjoint to $\boldsymbol{R}^k$; extrapolation and projection then give the adjoint of $\boldsymbol{Q}^k$:

$$\begin{aligned} \boldsymbol{G}^k &= -\frac{1}{s}\big(\bar{\boldsymbol{u}}^{k+1} \mathbf{1}_n^\top + \mathbf{1}_n (\bar{\boldsymbol{v}}^{k+1})^\top\big), \\ \boldsymbol{B}^k &= \boldsymbol{D}^k \odot \big(\bar{\boldsymbol{H}}^{k+1} + 2\boldsymbol{G}^k\big). \end{aligned} \tag{19}$$

Collecting the remaining contributions through the primal update yields

$$\begin{cases} \bar{\boldsymbol{H}}^k = \boldsymbol{B}^k - \boldsymbol{G}^k, & \text{(20a)} \\ \bar{\boldsymbol{u}}^k = \bar{\boldsymbol{u}}^{k+1} + \dfrac{1}{r}\boldsymbol{B}^k \mathbf{1}_n, & \text{(20b)} \\ \bar{\boldsymbol{v}}^k = \bar{\boldsymbol{v}}^{k+1} + \dfrac{1}{r}(\boldsymbol{B}^k)^\top \mathbf{1}_n. & \text{(20c)} \end{cases}$$

Each iteration contributes $-\boldsymbol{B}^k/r$ to the cost gradient. Summing these contributions and applying $\boldsymbol{C} = -\rho\tilde{\boldsymbol{H}}^{\mathrm{res}}$ gives the gradient along the mixer-construction path:

$$\nabla_{\boldsymbol{C}}\mathcal{L} = -\frac{1}{r}\sum_{k=0}^{K-1}\boldsymbol{B}^k, \qquad \nabla_{\tilde{\boldsymbol{H}}^{\mathrm{res}}}\mathcal{L} = -\rho\,\nabla_{\boldsymbol{C}}\mathcal{L} + \left(\mathrm{D}_{\tilde{\boldsymbol{H}}^{\mathrm{res}}}\boldsymbol{H}^0\right)^*[\bar{\boldsymbol{H}}^0]. \tag{21}$$

The cost gradient in this expression holds $\boldsymbol{H}^0$ fixed; the second term accounts for any score dependence of that initialization and vanishes when $\boldsymbol{H}^0$ is score-independent. Here D denotes the initialization Jacobian, and a superscript $*$ denotes its adjoint. The reverse sweep requires $O(Kn^2)$ arithmetic per mixer using the same reductions, broadcasts, and elementwise operations as the forward pass, without forming a Jacobian or the constraint matrix.

4 Experiments

4.1 Experimental Setup

Models and baselines. We use MiniMind-3 models (Gong, 2024) to study pre-training dynamics and downstream accuracy. The 64M dense and 198M-A64M MoE pre-training runs compare standard Transformer residual connections (Residual) (He et al., 2016; Vaswani et al., 2017), *m*HC with Sinkhorn–Knopp normalization (*m*HC-Sinkhorn) (Xie et al., 2026; Sinkhorn & Knopp, 1967), and two structured *m*HC variants defined by different objectives. Structured–Trace uses the fixed cost $\boldsymbol{C} = -\boldsymbol{I}_n$, which yields the objective $-\operatorname{tr}(\boldsymbol{H})$ and favors same-stream couplings. Note that the exact trace optimum is $\boldsymbol{I}_n$; in our experiments, Structured–Trace uses the mixer returned after K primal–dual updates, whereas *m*HC-Identity uses $\boldsymbol{I}_n$ directly. Structured–Score, our main variant, uses $\boldsymbol{C}_{l,t} = -\rho\,\tilde{\boldsymbol{H}}^{\mathrm{res}}_{l,t}$ from Equation 7, making the cost of each stream pair depend on the learned mixer scores at that layer and token. The 64M dense run also includes fixed identity (*m*HC-Identity) and uniform (*m*HC-Uniform) mixers, representing no cross-stream exchange and uniform averaging along the residual path, respectively. A separate 7B dense comparison evaluates downstream accuracy for *m*HC-Sinkhorn and Structured–Score.

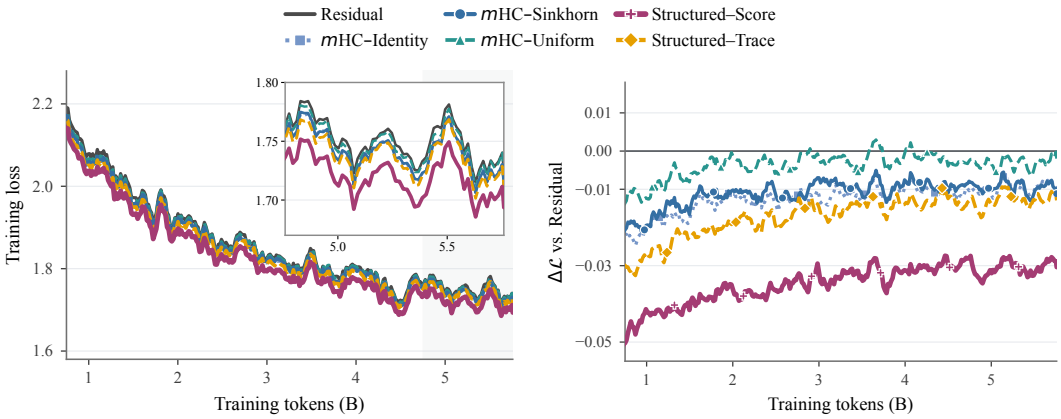
Training data and protocol. Within each model scale, all methods share the tokenizer, preprocessing pipeline, attention and feed-forward architecture, and training budget, so the comparisons isolate the effect of residual-mixer construction. The 64M dense and 198M-A64M MoE models are trained for two epochs on the MiniMind-3 pre-training corpus (Gong, 2024). Both settings use 340-token sequences, a per-device batch size of 32, eight gradient-accumulation steps, and bfloat16 precision; the learning rate warms up to 5×10^{-4} over the first 1% of training and then decays to 5×10^{-5}. For the separate 7B dense comparison, both models are trained on 15B tokens from FineWeb-Edu (Penedo et al., 2024). These runs use 768-token sequences, a per-device batch size of 24, six gradient-accumulation steps, and bfloat16 precision; the learning rate warms up to 2×10^{-4} over the first 1% of training and then decays to 2×10^{-5}.

Implementation. All multi-stream models use $n = 4$ residual streams. For each token, *m*HC-Sinkhorn constructs its mixer with 20 Sinkhorn–Knopp iterations of alternating row and column normalization, whereas Structured–Trace and Structured–Score use $K = 20$ iterations of the primal–dual method in Algorithm 1. For every mixer, we reset the row and column dual variables to $\boldsymbol{u}^0 = \boldsymbol{v}^0 = \boldsymbol{0}$ at the start of each forward pass. We implement all models in PyTorch (Paszke et al., 2019) using Hugging Face Transformers (Wolf et al., 2020); Appendix A provides the architecture and training configurations for all three model scales.

Evaluation. For the 64M dense and 198M-A64M MoE comparisons, we align checkpoints by the number of processed tokens and measure each method's training-loss difference from Residual. For method m at token count τ, we define

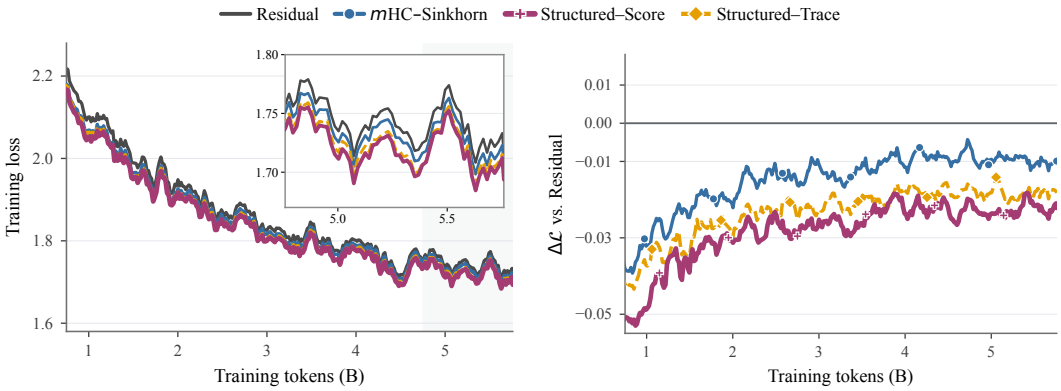
$$\Delta\mathcal{L}_m(\tau) = \mathcal{L}_m(\tau) - \mathcal{L}_{\mathrm{Residual}}(\tau). \tag{22}$$

A negative value therefore means that method m attains lower loss after processing the same number of tokens. We summarize late-stage behavior by averaging the unsmoothed $\Delta\mathcal{L}_m$ values over the



(a) Training loss (b) Loss gap to Residual

Figure 3: **Pre-training dynamics of the 64M dense model.** (a) Training loss and (b) its matched-token difference from Residual after the shared initial 0.75B-token transient. The shaded interval marks the final 1B tokens enlarged in the inset.



(a) Training loss (b) Loss gap to Residual

Figure 4: **Pre-training dynamics of the 198M-A64M MoE model.** (a) Training loss and (b) its matched-token difference from Residual after the shared initial 0.75B-token transient. The shaded interval marks the final 1B tokens enlarged in the inset.

final 20 matched checkpoints; the curves in Figures 3 and 4 use centered seven-checkpoint moving averages for visualization.

For the separate 7B comparison, we use the LM Evaluation Harness (Gao et al., 2024) to measure zero-, five-, and ten-shot accuracy on ARC-Easy (Clark et al., 2018), PIQA (Bisk et al., 2020), WinoGrande (Sakaguchi et al., 2020), RTE (Dagan et al., 2006), COPA (Roemmele et al., 2011), HellaSwag (Zellers et al., 2019), SciQ (Welbl et al., 2017), and CommitmentBank (CB) (de Marneffe et al., 2019). For each shot setting, we report per-task accuracy and its unweighted mean across the eight tasks.

4.2 Main Results

Figures 3 and 4 show a similar ordering among mixer constructions under matched token budgets. *m*HC-Sinkhorn attains a modest reduction relative to Residual, while the structured variants generally produce larger reductions as training progresses. Structured–Score maintains the lowest late-stage loss in both the dense and MoE runs, indicating that training dynamics are sensitive to the rule used to construct mixers toward the shared Birkhoff-constrained target.

Table 1: **Structured–Score attains higher mean downstream accuracy in the 7B dense comparison.** Both checkpoints are trained for 15B tokens on FineWeb-Edu. Values are percentages; **bold** marks the better method within each shot setting. Avg. is the unweighted task mean; Δ Avg. denotes the gain over *m*HC-Sinkhorn in percentage points.

Shots	Method	ARC-E	PIQA	WinoG.	RTE	COPA	HellaS.	SciQ	CB	Avg.	Δ Avg.
Zero-shot	*m*HC-Sinkhorn	**31.19**	53.97	**51.78**	**52.35**	59.00	26.97	45.40	26.79	43.43	—
	Structured–Score	31.14	**55.39**	50.36	50.90	**63.00**	**28.05**	**49.10**	**39.29**	**45.90**	**+2.47**
Five-shot	*m*HC-Sinkhorn	**30.89**	54.24	49.96	51.26	57.00	26.45	44.80	41.07	44.46	—
	Structured–Score	30.13	**54.84**	**50.36**	**54.51**	**58.00**	**27.62**	**47.80**	**51.79**	**46.88**	**+2.42**
Ten-shot	*m*HC-Sinkhorn	29.88	54.03	**51.54**	**50.90**	54.00	26.04	**40.30**	41.07	43.47	—
	Structured–Score	**30.26**	**54.84**	49.88	50.54	**57.00**	**26.91**	40.20	**42.86**	**44.06**	**+0.59**

We quantify this late-stage separation by averaging $\Delta\mathcal{L}_m$ over the final 20 matched checkpoints. Structured–Score reduces loss relative to Residual by 0.030 for the 64M dense model and by 0.022 for the 198M-A64M MoE model, compared with reductions of 0.010 and 0.011, respectively, for *m*HC-Sinkhorn. Thus, the score-based construction yields a larger late-stage reduction in both runs, with the same ordering in the dense and sparsely activated MoE architectures.

Table 1 evaluates downstream performance in a separate 7B comparison. Relative to *m*HC-Sinkhorn, Structured–Score raises mean accuracy from 43.43 to 45.90 in the zero-shot setting and from 44.46 to 46.88 in the five-shot setting, corresponding to gains of 2.47 and 2.42 percentage points. At ten shots, mean accuracy increases from 43.47 to 44.06, a smaller but still positive gain of 0.59 points. Across the 24 task–shot combinations, Structured–Score achieves higher accuracy on 17, including improvements on PIQA, COPA, and HellaSwag under all three shot settings; the two largest task-level gains occur on CB at zero and five shots. The 7B evaluation therefore complements the smaller pre-training comparisons: Structured–Score attains higher mean accuracy under all three prompting regimes.

5 Conclusion

We introduced structured *m*HC, which formulates residual-mixer construction as objective-guided optimization over the Birkhoff polytope. The target feasible set encodes the non-expansive composition property of *m*HC, while an explicit objective specifies which admissible stream interactions the construction favors. We instantiated the framework with Structured–Score, a token-dependent linear objective derived from learned residual-mixer scores, and developed a fixed-depth, differentiable primal–dual method whose tensor-form updates exploit the row- and column-sum structure without materializing the constraint matrix. Under matched token budgets, Structured–Score attained the lowest late-stage loss in both the 64M dense and 198M-A64M MoE runs. Averaged over the final 20 checkpoints, its reductions relative to Residual were 0.030 and 0.022, compared with 0.010 and 0.011 for *m*HC-Sinkhorn. In a separate 7B dense comparison after 15B training tokens, Structured–Score also attained higher mean accuracy across eight downstream tasks under zero-, five-, and ten-shot evaluation. These results support mixer construction as a consequential design choice under a shared Birkhoff-constrained target. Our main evaluation focuses on a learned linear objective with four residual streams in MiniMind-3; studying other objective classes, stream counts, model families, and training scales will clarify the broader scope of objective-guided mixer construction.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing the proposed method. Additionally, we used generative AI tools to improve the readability, organization, and formatting of the manuscript. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Ekaterina Alimaskina, Gleb Molodtsov, and Aleksandr Beznosikov. Analyzing stream collapse in Hyper-Connections: From diagnosis to mitigation, 2026. URL https://arxiv.org/abs/2606.03483.

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

Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011. doi: 10.1007/s10851-010-0251-1.

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.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges: Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment*, volume 3944 of *Lecture Notes in Computer Science*, pp. 177–190. Springer, 2006. doi: 10.1007/11736790_9.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. The CommitmentBank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23 (2):107–124, 2019. doi: 10.18148/sub/2019.v23i2.601.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The Language Model Evaluation Harness, 2024. URL https://zenodo.org/records/12608602.

Jingyao Gong. Minimind: Train a tiny llm from scratch, 2024. URL https://github.com/jingyaogong/minimind. GitHub repository, accessed 2026.

Bingsheng He and Xiaoming Yuan. Convergence analysis of primal-dual algorithms for a saddle-point problem: from contraction perspective. *SIAM Journal on Imaging Sciences*, 5(1):119–149, 2012.

Bingsheng He, Feng Ma, Shengjie Xu, and Xiaoming Yuan. A generalized primal-dual algorithm with improved convergence condition for saddle point problems. *SIAM Journal on Imaging Sciences*, 15(3):1157–1183, 2022. doi: 10.1137/21M1453463.

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.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale, 2024. URL https://arxiv.org/abs/2406.17557.

Qwen Team. On the design of Qwen3.8-Next architecture: Evaluation, efficiency, and training stability. Technical report, Alibaba Group, August 2026. URL https://arxiv.org/abs/2608.30320.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford University, March 2011.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

Tencent Hy Team. Hy4-preview model card. https://huggingface.co/tencent/Hy4-preview, 2026. Accessed September 2026.

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*, 2017.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, Copenhagen, Denmark, 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413/.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Clara von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

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. *m*HC: 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.

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

Pengxiang Zhao, Xing Li, Xianzhi Yu, Wei Guo, and Zhenhua Dong. How does *m*hc use its residual streams? selective routing and near-identity mixing. Manuscript, 2026.

Wuyang Zhou, Yuxuan Gu, Giorgos Iacovides, and Danilo Mandic. Kromhc: Manifold-constrained hyper-connections with kronecker-product residual matrices, 2026. URL https://arxiv.org/abs/2601.21579.

Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. Hyper-connections, 2025. URL https://arxiv.org/abs/2409.19606.

A Detailed Experimental Configurations

Model scales. Table 2 summarizes the configurations used in our pre-training and downstream comparisons. The 64M dense and 198M-A64M MoE models share the same eight-layer backbone, hidden size, and attention configuration. The MoE model replaces each dense feed-forward block with four routed experts, of which one is active for each token; this increases total capacity to 198M parameters while keeping the active parameter count near 64M. The 7B dense model increases both depth and width, using 34 layers with a hidden size of 4096, and serves as the larger-scale setting for downstream evaluation.

Controlled comparisons. Within each model scale, all methods use the same backbone, data, optimizer, and training budget, so the residual-path construction is the primary experimental variable. All multi-stream variants use $n = 4$ residual streams. mHC-Sinkhorn applies 20 Sinkhorn–Knopp normalization iterations, while Structured–Trace and Structured–Score apply 20 primal–dual iterations. The 64M dense and 198M-A64M MoE experiments run for two epochs on MiniMind-3 data, whereas the 7B models are trained on 15B tokens from FineWeb-Edu. Because the model scales use different data and sequence lengths, all quantitative comparisons are made between methods within the same scale.

Table 2: **Model and training configurations.** Architecture, mixer, and optimization settings for the three model scales used in our experiments.

Attribute	**Dense 64M**	**MoE 198M-A64M**	**Dense 7B**
Backbone	MiniMind-3 decoder-only Transformer		
Model type	Dense	MoE	Dense
Total parameters	64M	198M	7B
Active parameters	64M	64M	7B
Layers	8	8	34
Hidden size	768	768	4096
FFN dimension	2432	2432	12864
Attention heads	8	8	32
KV heads	4	4	8
Head dimension	96	96	128
Routed experts	–	4	–
Active experts	–	1	–
Vocabulary size	6400		
Position embedding	RoPE		
RoPE base θ	10^6		
Max position embeddings	2048	2048	4096
Layer norm	RMSNorm, $\epsilon = 10^{-6}$		
Streams n (multi-stream methods)	4		
Sinkhorn–Knopp iterations	20		
Primal–dual iterations K	20		
Training data	MiniMind-3	MiniMind-3	FineWeb-Edu
Training budget	2 epochs	2 epochs	15B tokens
Sequence length	340	340	768
Per-device batch size	32	32	24
Gradient accumulation	8	8	6
Optimizer	Fused AdamW		
AdamW betas	$(0.9, 0.95)$		
Weight decay	0.1		
Peak learning rate	5×10^{-4}	5×10^{-4}	2×10^{-4}
Minimum learning rate	5×10^{-5}	5×10^{-5}	2×10^{-5}
LR scheduler	Cosine decay		
Warmup ratio	1%		
Gradient clipping	1.0		
Precision	bfloat16		