HERO-MoE: Historical Expert Routing with Scale-Preserving Fusion

Anonymous authors
Paper under double-blind review

Abstract

Mixture-of-Experts (MoE) architectures have become a standard way to scale model capacity while keeping computation sparse, yet routing remains a key determinant of MoE quality and training behavior. Prior empirical studies suggest that MoE routing reflects input semantics and upstream computation across depth, but standard routers do not explicitly use the routing distributions produced by preceding layers. We propose **HERO-MoE**, **H**istorical **E**xpert **RO**uting with Scale-Preserving Fusion, a routing framework that injects historical routing priors into MoE routers by reusing detached, dense routing distributions collected from preceding MoE layers. The key idea is simple: HERO-MoE preserves the original token-conditioned routing branch and adds a residual historical routing contribution before the standard softmax and top-k dispatch. To stabilize this historical signal, HERO-MoE introduces a scale-preserving fusion mechanism that matches the magnitude of historical routing memory to the current hidden representation and accounts for the number of visible historical layers, without introducing an auxiliary routing loss or a fusion-specific tuning parameter. By reusing routing distributions already computed by preceding MoE layers, HERO-MoE improves training-loss reduction with modest end-to-end overhead. The resulting router remains compatible with standard sparse dispatch, including top-k and group-limited routing, and can be inserted into existing MoE backbones with minimal architectural changes. Experiments on an approximately 8B-parameter MoE model with 0.5B active parameters, trained from scratch on 100B tokens, show that HERO-MoE reduces the final loss from 1.6393 to 1.6184, while peak memory and FLOPs increase by only 0.44% and 0.64%, respectively.

1 Introduction

Mixture-of-Experts (MoE) (Liu et al., 2024b; Zhou et al., 2022) has become a widely adopted paradigm for scaling large language models, because it increases model capacity without increasing the computation of every token proportionally. By routing each token to only a small subset of experts, MoE architectures achieve a favorable trade-off between parameter scale and computational efficiency, and have demonstrated strong performance in language modeling, translation, and long-context reasoning. From early sparsely gated models to modern large-scale sparse Transformers, the central mechanism has remained the same: a router selects a small number of experts for each token, thereby determining how expert specialization emerges and how model capacity is utilized (Jacobs et al., 1991; Shazeer et al., 2017; Lepikhin et al., 2020; Liu et al., 2024a; Fedus et al., 2022; Du et al., 2022; Zoph et al., 2022). Consequently, the routing mechanism has become one of the most important components affecting the quality and optimization behavior of sparse MoE models.

Prior empirical studies consistently show that MoE routing reflects input semantics and upstream computation, with structured routing patterns and cross-layer effects extending across depth (Olson et al., 2025; Muennighoff et al., 2025; Li et al., 2026). These findings indicate that preceding routing distributions contain token-specific and computation-aware signals that can complement the current hidden representation. Although a standard router computes logits from the current hidden representation, which carries information from preceding computation, it does not explicitly take the routing distributions produced by preceding layers as additional inputs. Treating routing history as an explicit prior can therefore provide the current router with a direct signal about the same token's previous expert assignments, enabling later expert selection to use both current token features

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.

and the token's preceding routing trajectory. Existing work has begun to exploit related routing structure through indirect regularization or by modifying the routing pathway. Regularization-based methods encourage consistent, non-overlapping, or complementary expert assignments (Omi et al., 2025; Guo et al., 2026; Hu et al., 2026), whereas context-aware routing methods share, propagate, or recurrently update routing-related states across layers (Qiu et al., 2025; Gu et al., 2025; Huang et al., 2026). However, these approaches either require additional regularization terms, loss weights, and method-specific hyperparameter tuning, or modify the routing pathway itself. Directly injecting preceding routing distributions as historical priors while preserving the original current-token routing branch and standard sparse dispatch therefore remains underexplored. This leaves an important design gap in how to exploit routing history without sacrificing the simplicity and compatibility of the standard MoE routing pipeline.

To fill this gap, we propose HERO-MoE, Historical Expert Routing with Scale-Preserving Fusion. The key idea is simple: HERO-MoE reuses the detached, dense routing distributions produced by preceding MoE layers within the same pipeline-parallel stage (Huang et al., 2019) as historical priors for the current router. It preserves the original token-conditioned routing branch and adds a residual logit contribution derived from the routing history before the standard softmax and top-k dispatch. Thus, the router can exploit how the same token was routed in preceding layers without changing the standard sparse dispatch procedure or introducing an additional training objective. The main challenge lies in the scale of this seemingly simple fusion. As the number of visible historical layers grows, directly combining unnormalized routing history can make its contribution either too weak or too dominant, while also causing layer-dependent variations in the router input. HERO-MoE addresses this issue with scale-preserving normalization and depth-aware initialization, which align the historical and current representations and maintain a comparable initial logit scale across layers. Since the historical distributions are detached from the computation graph, the method introduces modest additional computation and does not backpropagate through previous routing decisions.

We evaluate HERO-MoE on an approximately 8B-parameter MoE model with 0.5B active parameters trained from scratch on 100B tokens. HERO-MoE reduces the final training loss from 1.6393 to 1.6184 and improves the average score from 36.57 to 37.42 across 13 general benchmarks, with improvements on 9 tasks. The additional end-to-end cost remains modest: compared with the baseline, HERO-MoE increases total step FLOPs by only 0.64%, peak memory by 0.44%, and total training time by 3.85%. These results show that historical routing can improve both MoE optimization and downstream generalization without requiring an auxiliary routing loss or additional method-specific tuning. Furthermore, the learned historical router provides an interpretable view of layer-wise routing dependencies and expert-level cooperative or competitive relations. In summary, our contributions are threefold:

- We identify and exploit non-trivial cross-layer dependencies in MoE routing by proposing HERO-MoE, a historical expert routing framework that injects stage-local routing priors into the current router.
- We introduce a scale-preserving historical fusion mechanism with depth-aware initialization, which stabilizes the magnitude of historical routing signals without requiring an additional fusion-specific hyperparameter or auxiliary routing loss.
- We demonstrate that HERO-MoE improves training loss and downstream benchmark performance with modest end-to-end overhead, while also providing an interpretable visualization of cross-layer and inter-expert routing relationships.

2 Related Work

In this section, we review studies on MoE routing along two related threads: sparse routing with load balancing and specialization regularization, and context-aware system-level routing designs.

2.1 Sparse MoE Routing and Load Balancing

Mixture-of-Experts research dates back to early conditional computation and hierarchical expert models (Jacobs et al., 1991; Jordan & Jacobs, 1994), and was later scaled to modern neural networks with sparsely gated routing (Shazeer et al., 2017). Subsequent work such as GShard, Switch

Transformers, GLaM, and ST-MoE demonstrated that sparse top-k routing can substantially improve parameter efficiency and training scalability in large language models (Lepikhin et al., 2020; Fedus et al., 2022; Du et al., 2022; Zoph et al., 2022). However, these routers still rely primarily on the current token representation when making dispatch decisions, which makes them vulnerable to expert collapse, uneven utilization, and unstable training dynamics. To mitigate these issues, a number of balancing strategies were proposed, including expert-choice routing and auxiliary-loss-free balancing (Zhou et al., 2022; Wang et al., 2024). These methods improve load balance, but they still treat routing as a mostly local decision problem and do not explicitly model how routing behavior evolves across depth.

Recent work also shows that auxiliary regularization losses can improve specialization and routing consistency without changing the router architecture. Load Balancing Mixture of Experts with Similarity Preserving Routers introduces a similarity-preserving load-balancing loss that encourages consistent expert choices for similar inputs (Omi et al., 2025). Advancing Expert Specialization for Better MoE adds orthogonality and variance losses to reduce expert overlap and sharpen routing decisions (Guo et al., 2026). Synergistic Intra- and Cross-Layer Regularization Losses for MoE Expert Specialization combines intra-layer specialization and cross-layer coupling losses to promote complementary expert usage across depth (Hu et al., 2026). These methods are effective, but they incur additional loss weights and hyperparameter tuning overhead; furthermore, the optimal parameters of each individual method require extra adjustment when different methods operate in conjunction, whereas HERO-MOE improves routing mainly by injecting historical routing priors through a scale-preserving router parameterization.

2.2 Context-Aware and System-Aware Routing

A growing line of work argues that MoE routing should exploit more than the current hidden state alone. RMoE introduces layerwise recurrent routing and shows that cross-layer dependencies can improve expert selection and parameter efficiency (Qiu et al., 2025). Omni-Router shares routing decisions across MoE layers to encourage structured expert usage in speech models (Gu et al., 2025), while MCF-MOE incorporates multi-level context to obtain more consistent expert selection across layers (Huang et al., 2026). Complementary analyses on shared routing geometry and similarity-preserving routers further suggest that routers learn reusable relational structure, and that preserving such structure can reduce redundancy and improve balancing (Labzin et al., 2026; Omi et al., 2025; Ahrac et al., 2026). In parallel, system-oriented MoE frameworks such as DeepSpeed-MoE, FastMoE, Tutel, and MegaBlocks improve training and inference efficiency through better parallelism, dispatch, and kernel design (Rajbhandari et al., 2022; He et al., 2021; Hwang et al., 2023; Gale et al., 2023). Unlike these methods, HERO-MoE integrates historical routing context directly into the router while preserving the standard sparse dispatch procedure, making the routing policy more aware of cross-layer routing structure.

3 Method

In this section, we formalize the complete HERO-MoE framework. We first provide an overview of the routing pipeline, and then describe the router parameterization, scale-preserving initialization strategy, and stage-local historical routing mechanism in detail.

Figure 1 illustrates the overall routing flow of HERO-MoE. For each MoE layer, the current token representation is processed by the original router branch, while the historical routing memory collected from preceding layers is fed into the residual router branch. The two branches produce original router logits and residual historical logits, which are added to obtain the final router logits. The resulting logits are converted into routing probabilities for top-k expert selection and weighted expert aggregation. Meanwhile, the dense routing output of the current layer is concatenated with the previous history to form the updated historical memory for subsequent MoE layers.

3.1 Parameterization

Consider a Transformer with L MoE layers. Let $\boldsymbol{X}^{(l)} \in \mathbb{R}^{N\times d}$ denote the token hidden representations entering the l-th MoE layer, where N is the sequence length or batch token count, d is the hidden dimension, $l \in \{1, \ldots, L\}$ is the layer index, L is the total number of MoE layers, and E is

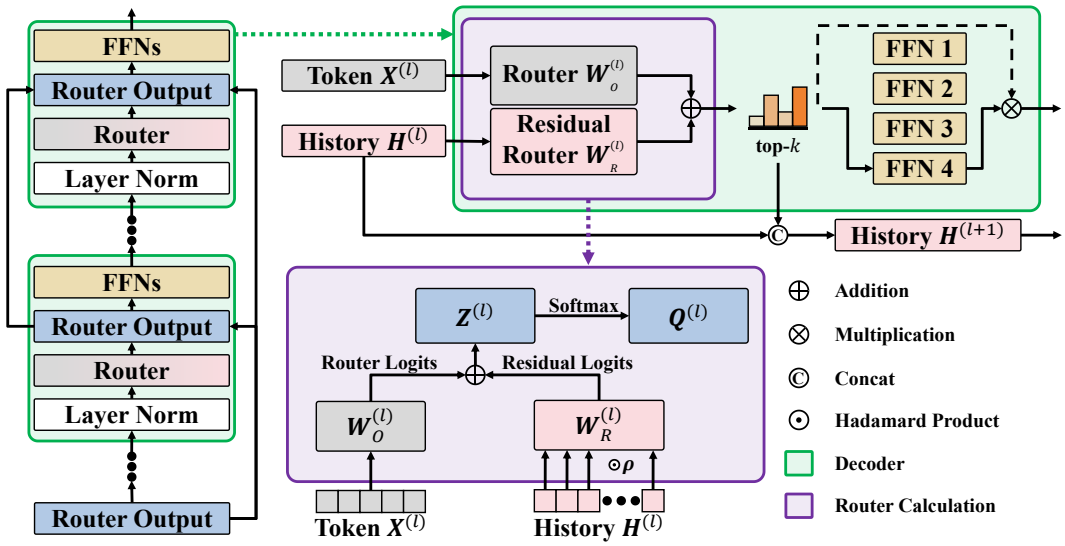

Figure 1: Pipeline of HERO-MoE, including the original router branch, historical residual branch, sparse dispatch, and history update.

the total number of experts. We use $t \in \{1, \ldots, N\}$ for the token index and $e \in \{1, \ldots, E\}$ for the expert index. The t-th row of $\boldsymbol{X}^{(l)}$ is denoted by $\boldsymbol{x}_t^{(l)}$.

In a conventional sparse MoE layer, the router computes expert logits solely from the current hidden representations. We denote this original routing pathway by the subscript O and compute its logits as

$$\boldsymbol{Z}_O^{(l)} = \boldsymbol{X}^{(l)} \boldsymbol{W}_O^{(l)\top} \in \mathbb{R}^{N \times E}, \qquad \boldsymbol{W}_O^{(l)} \in \mathbb{R}^{E \times d}, \tag{1}$$

where $\boldsymbol{W}_O^{(l)}$ is the trainable weight block corresponding to the original router.

Given a router logit matrix $\boldsymbol{Z}^{(l)}$, where $\boldsymbol{Z}^{(l)} = \boldsymbol{Z}_O^{(l)}$ for a conventional MoE layer, the standard softmax-based sparse MoE dispatch procedure first converts the logit vector of token t, denoted by $\boldsymbol{Z}_t^{(l)}$, into a probability distribution over the E experts:

$$\boldsymbol{Q}_t^{(l)} = \mathrm{softmax}\left(\boldsymbol{Z}_t^{(l)}\right) \in \mathbb{R}^E. \tag{2}$$

For each token, let k be the number of selected experts. The operator $\mathrm{TopK}(\cdot, k)$ returns the indices of the k largest entries, yielding the selected expert set

$$\mathcal{A}_t = \mathrm{TopK}\left(\boldsymbol{Q}_t^{(l)}, k\right), \qquad |\mathcal{A}_t| = k. \tag{3}$$

For a selected expert $e \in \mathcal{A}_t$, using e' as a dummy expert index, the selected probabilities are converted into normalized routing weights as

$$w_{t,e} = \frac{Q_{t,e}^{(l)}}{\sum_{e' \in \mathcal{A}_t} Q_{t,e'}^{(l)} + \epsilon}, \qquad e \in \mathcal{A}_t, \tag{4}$$

where $\epsilon > 0$ is a small constant for numerical stability. The outputs of the selected expert networks are then aggregated according to these normalized weights:

$$\boldsymbol{y}_t^{(l)} = \sum_{e \in \mathcal{A}_t} w_{t,e} f_e\left(\boldsymbol{x}_t^{(l)}\right), \tag{5}$$

where $f_e(\cdot)$ denotes the network of expert e and $\boldsymbol{y}_t^{(l)}$ is the output for token t.

To incorporate routing history while preserving this standard MoE pathway, we propose the history-aware residual router, referred to as HERO-MoE. Let M_l denote the number of preceding MoE

layers whose routing distributions are visible to layer l, let $\boldsymbol{H}^{(l)} \in \mathbb{R}^{N\times M_l E}$ denote the corresponding concatenated routing memory, and let $\boldsymbol{\rho}^{(l)} \in \mathbb{R}^{N\times 1}$ denote a token-wise scale-preserving coefficient vector broadcast along the feature dimension. We use $\odot$ for element-wise multiplication with row-wise broadcasting. The construction of $\boldsymbol{H}^{(l)}$ and the computation of $\boldsymbol{\rho}^{(l)}$ are detailed in Sections 3.3 and 3.2, respectively. The final router logits are defined as

$$\boldsymbol{Z}^{(l)} = \boldsymbol{Z}_O^{(l)} + \boldsymbol{Z}_R^{(l)}, \qquad \boldsymbol{Z}_R^{(l)} = \left(\boldsymbol{\rho}^{(l)} \odot \boldsymbol{H}^{(l)}\right) \boldsymbol{W}_R^{(l)\top}, \qquad \boldsymbol{W}_R^{(l)} \in \mathbb{R}^{E\times M_l E}, \tag{6}$$

where the subscript R denotes the residual routing branch and $\boldsymbol{W}_R^{(l)}$ is its trainable weight block. In the implementation, $\boldsymbol{X}^{(l)}$ and $\boldsymbol{\rho}^{(l)} \odot \boldsymbol{H}^{(l)}$ are concatenated along the feature dimension and passed through a single linear projection whose weight is conceptually partitioned as $[\boldsymbol{W}_O^{(l)} \,\|\boldsymbol{W}_R^{(l)}]$; the additive form above is algebraically equivalent to this implementation. When no historical layer is available, $M_l = 0$ and $\boldsymbol{Z}_R^{(l)} = \boldsymbol{0}$, so $\boldsymbol{Z}^{(l)} = \boldsymbol{Z}_O^{(l)}$. The resulting logits $\boldsymbol{Z}^{(l)}$ are then passed to the configured sparse-MoE routing policy; the standard softmax, top-k selection, and weighted aggregation above describe the default path.

3.2 Initialization and Scaling Strategy

The original and residual routing branches receive inputs with different effective dimensions. The original branch takes the current hidden representation of dimension d, while the residual branch takes a concatenated routing history whose dimension can grow with the depth of the network. Without an appropriate initialization scale, the residual logits could become increasingly large in deeper layers as more historical routing features are incorporated, leading to a depth-dependent routing scale and overly confident initial assignments.

To keep the initial magnitude of the combined router logits comparable across layers, we use a depth-aware initialization bound

$$b_l = \frac{1}{\sqrt{d + (l-1)E}}. \tag{7}$$

The term $d+(l-1)E$ represents the current hidden dimension together with the maximum historical feature dimension available before layer l in a single-stage setting. In the implementation, the router weight is a single matrix

$$\boldsymbol{W}^{(l)} = \left[\boldsymbol{W}_O^{(l)} \| \boldsymbol{W}_R^{(l)}\right] \in \mathbb{R}^{E\times(d+M_l E)}, \tag{8}$$

where the residual block is omitted when $M_l = 0$. Its entries are initialized from the same uniform distribution:

$$\boldsymbol{W}^{(l)} \sim \mathcal{U}(-b_l, b_l), \tag{9}$$

where $\mathcal{U}(-b_l, b_l)$ denotes the uniform distribution on $[-b_l, b_l]$. The bound depends on the global layer index l rather than the number of locally visible layers, so that the initialization of a given layer is not changed by the choice of pipeline partition. The first MoE layer has no preceding history and therefore uses only the original router block.

We further align the magnitude of the historical routing memory with that of the current hidden representation. When $M_l > 0$, let $\mathrm{RMS}(\cdot)$ denote the root-mean-square magnitude computed independently for each row. We define the token-wise scale-preserving coefficient as

$$\boldsymbol{\rho}^{(l)} = \frac{\mathrm{RMS}(\boldsymbol{X}^{(l)})}{\mathrm{RMS}(\boldsymbol{H}^{(l)}) + \epsilon} \cdot \sqrt{\frac{l-1}{M_l}}, \tag{10}$$

where $\epsilon > 0$ is a small constant for numerical stability. The first factor performs dynamic magnitude alignment, while the second factor compensates for the reduction in visible historical layers introduced by pipeline parallelism. For any representation vector $\boldsymbol{u} \in \mathbb{R}^D$ with i-th component u_i, the RMS magnitude is

$$\mathrm{RMS}(\boldsymbol{u}) = \sqrt{\frac{1}{D}\sum_{i=1}^{D} u_i^2}. \tag{11}$$

In the implementation, the RMS statistics are detached from the gradient graph, so $\boldsymbol{\rho}^{(l)}$ acts as a stop-gradient scaling factor. Thus, $\rho^{(l)}$ calibrates the effective energy of the historical input across depths and pipeline partitions, while the initialization strategy keeps the initial router-logit scale comparable.

3.3 STAGE-LOCAL HISTORICAL ROUTING

In large-scale distributed training, deep models are typically partitioned across multiple pipeline (Huang et al., 2019) stages. Transmitting historical routing features across pipeline stage boundaries would introduce additional communication overhead and may disrupt the execution schedule. Therefore, HERO-MoE restricts historical routing retrieval to the current pipeline stage. For layer l, we define the set of visible predecessor layers as

$$\mathcal{S}(l) = \{\, j \mid j < l,\ \text{layer } j \text{ resides on the same pipeline stage as layer } l \,\}. \tag{12}$$

The number of visible historical layers is

$$M_l = |\mathcal{S}(l)|. \tag{13}$$

Thus, only preceding layers on the same pipeline stage contribute to the historical routing memory, avoiding cross-stage communication while preserving the execution schedule.

For each historical layer $j \in \mathcal{S}(l)$, we retrieve its dense, pre-Top-k routing probability distribution:

$$\boldsymbol{Q}^{(j)} = \mathrm{StopGrad}\left(\mathrm{softmax}\left(\boldsymbol{Z}^{(j)}\right)\right) \in \mathbb{R}^{N\times E}. \tag{14}$$

Here, $\boldsymbol{Z}^{(j)}$ denotes the router logits computed from the current representation at layer j and its available historical routing input. The distribution $\boldsymbol{Q}^{(j)}$ is stored immediately after these logits are computed and before the subsequent configured top-k routing policy is applied. The operator $\mathrm{StopGrad}(\cdot)$ detaches its input from the gradient graph, preventing gradients from the current router from propagating through previous routing decisions.

Enumerate the visible predecessor indices as $j_1, \ldots, j_{M_l}$, where j_m is the m-th visible predecessor for $m \in \{1, \ldots, M_l\}$. The historical routing memory is constructed by concatenating their routing distributions along the feature dimension:

$$\boldsymbol{H}^{(l)} = \left[\boldsymbol{Q}^{(j_1)}, \boldsymbol{Q}^{(j_2)}, \ldots, \boldsymbol{Q}^{(j_{M_l})}\right] \in \mathbb{R}^{N\times M_l E}, \tag{15}$$

where $\{j_1, \ldots, j_{M_l}\} = \mathcal{S}(l)$. The t-th row of $\boldsymbol{H}^{(l)}$ corresponds to the routing history of the same token represented by the t-th row of $\boldsymbol{X}^{(l)}$. At layer l, this history is scaled and mapped to $\boldsymbol{Z}_R^{(l)}$, added to $\boldsymbol{Z}_O^{(l)}$, and then passed through the standard sparse MoE routing procedure described in Section 3.

4 EXPERIMENTS

In this section, we conduct empirical evaluation of HERO-MoE. We first detail the experimental setup including datasets, model configurations and evaluation benchmarks, then present the effectiveness results and further analyze the learned inter-layer routing patterns. All training runs were performed on NPUs, each with 64 GB of NPU memory.

4.1 EXPERIMENT SETTINGS

Datasets. The training dataset includes open-source datasets: dclm (Li et al., 2024b), map-cc (Du et al., 2024), ultrachat (Ding et al., 2023), tuluv3 (Lambert et al., 2024), finemath (Liu et al., 2024c), megamath (Zhou et al., 2025), as well as high-quality proprietary self-collected datasets related to education that are not publicly available.

Model Configuration. The experimental evaluation defaults to a MoE model built with our custom architecture (close to Deepseek-MoE (Liu et al., 2024b)), which has approximately 8B total parameters and 0.5B activated parameters. Pre-training with random initialization adopts sequences of 4K tokens, with a total training corpus of roughly 100B tokens. Specifically, the model comprises 30 Transformer layers with a hidden dimension of 1024. The attention layer adopts Multi-Head Latent Attention (Liu et al., 2024a) (MLA, 16 attention heads, Query LoRA rank of 512). The feed-forward layer employs an MoE architecture with 128 experts (intermediate size of 640), dynamically routing to Top-4 experts per token. The learning rate follows a cosine decay schedule from 2×10^{-3} to

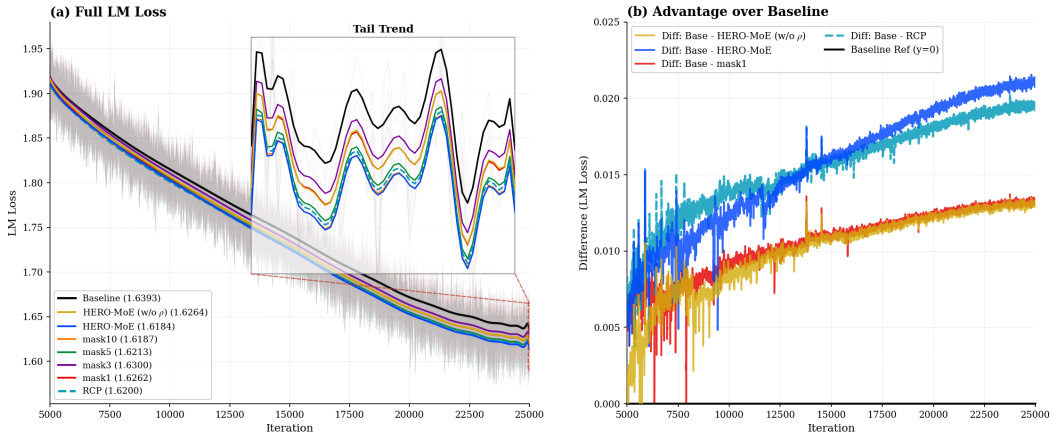

Figure 2: Training loss of the Baseline, RCP, HERO-MoE, historical-window ablations, and the no-scaling ablation.

5×10^{-5} with a global batch size of 1024 across 25,000 training steps. The training results before introducing HERO-MoE are denoted as Baseline.

Evaluation Metrics. The general benchmarks cover 13 typical task categories, including mathematical reasoning (GSM8K (Cobbe et al., 2021), MathQA (Amini et al., 2019), MATH (Hendrycks et al., 2021)), commonsense question answering (ARC-C, ARC-C-ZH, ARC-E (Clark et al., 2018)), comprehensive cognitive reasoning (MMLU (Hendrycks et al., 2020), CMMLU (Li et al., 2024a), BBH (Suzgun et al., 2023)), text coherence assessment (LAMBADA (Paperno et al., 2016)), code generation (HumanEval (Chen, 2021)), and domain-specific tasks (HellaSwag (Zellers et al., 2019), PiPA (Kim et al., 2025)), which span both English and Chinese bilingual scenarios.

4.2 Effectiveness of HERO-MoE

Training Loss. Figure 2(a) presents the full training-loss trajectories of different routing-enhancement methods, with the inset providing a zoomed-in view of the final training stage. Here, the baseline denotes the raw MoE model without historical routing, RCP (Hu et al., 2026) denotes the adjacent-layer route consolidation method, and mask1, mask3, mask5, and mask10 denote ablation variants that use only recent historical layers. The no-scaling ablation removes the scale-preserving factor in Eq. (10) while retaining the historical routing branch. The historical-routing variants generally achieve lower final losses than the baseline. Full HERO-MoE achieves the lowest final loss, around 1.6184 compared with 1.6393 for the baseline, while mask10 performs very closely, indicating that a medium-range history captures most of the benefit. The no-scaling variant also benefits from historical routing, but converges to a higher final loss than the full method, showing that the scale-preserving operation is important for effectively incorporating routing history. RCP also improves over the baseline, but remains weaker than HERO-MoE and mask10, showing that, in our setting, HERO-MoE achieves lower training loss without introducing an additional loss objective or requiring hyperparameter tuning.

Figure 2(b) reports the loss advantage over the baseline throughout training. The full HERO-MoE method maintains the largest advantage among the compared variants, whereas the no-scaling ablation exhibits a smaller improvement, further demonstrating the value of the scale-preserving operation. The ablation results also show that an overly short history limits the performance gain: mask1 provides a noticeably smaller improvement, suggesting that routing decisions benefit from richer historical trajectories beyond the immediately preceding layer. Moreover, the advantage of HERO-MoE over the baseline continues to increase during training, whereas the improvement of RCP saturates earlier. These results indicate that HERO-MoE provides sustained benefits for long-term MoE optimization.

Table 1: General benchmark comparison between the baseline MoE and HERO-MoE.

Model	GSM8K	MathQA	MATH	ARC-C	ARC-C-ZH	ARC-E	HellaSwag	BBH	MMLU	CMMLU	HumanEval	LAMBADA	PIPA	Avg.
Baseline	10.99	**30.99**	8.14	35.32	30.97	**70.96**	40.62	**29.3**	42.1	**43.2**	15.24	45.55	72.03	36.57
HERO-MoE	**12.21**	30.95	**9.6**	**37.63**	**31.83**	70.92	**42.01**	29.09	**42.22**	42.38	**18.9**	**46.52**	**72.2**	**37.42**
Δ	1.22	−0.04	1.46	2.31	0.86	−0.04	1.39	−0.21	0.12	−0.82	3.66	0.97	0.17	0.85

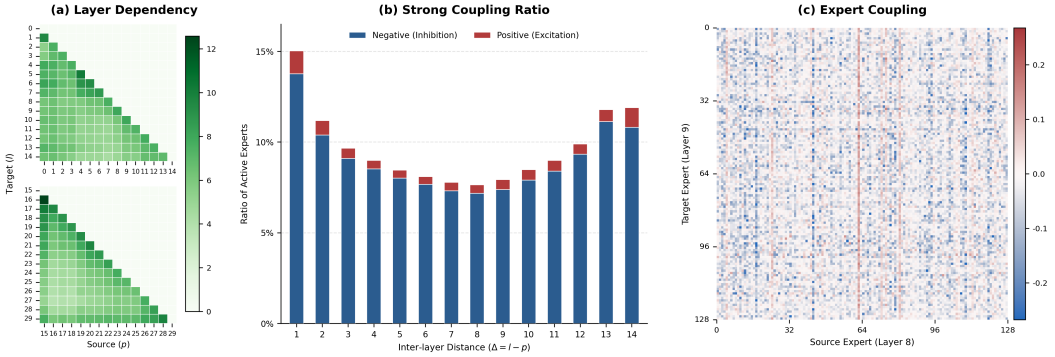

Figure 3: Layer-wise historical dependencies, distance-wise strong coupling ratios, and a representative expert-level coupling matrix learned by HERO-MoE.

General Benchmarks. Table 1 compares the baseline MoE and HERO-MoE on multiple general benchmarks. Overall, our method improves the average score from 36.57 to 37.42, bringing an average gain of +0.85. Among the 13 tasks, HERO-MoE achieves improvements on 9 tasks.

The most notable gains appear on HumanEval (+3.66), ARC-C (+2.31), MATH (+1.46), HellaSwag (+1.39), and GSM8K (+1.22). These tasks usually require stronger reasoning, code generation, or complex semantic modeling abilities, indicating that HERO-MoE improves not only the training loss but also downstream task performance. Meanwhile, HERO-MoE shows slight drops on MathQA, ARC-E, BBH, and CMMLU, suggesting that historical routing enhancement is not monotonically beneficial for all tasks. One possible reason is that historical routing changes the expert specialization pattern, making its benefit less obvious on some knowledge-oriented or multiple-choice tasks.

Overall, by introducing historical routing information, HERO-MoE improves both the training behavior and generalization performance of MoE models. The average performance gain is stable, with especially clear benefits on mathematics, coding, and complex reasoning tasks.

4.3 Findings of HERO-MoE

Figure 3(a) visualizes the layer-wise dependency strength learned by the historical routing branch in HERO-MoE. Each lower-triangular entry represents the routing dependency from a predecessor layer p to a target layer l, where darker green indicates a stronger influence of the historical layer on the current router. We observe that the learned dependencies are not limited to adjacent layers, but extend across multiple preceding layers. Although entries near the diagonal are generally stronger, visible non-zero dependencies also appear between more distant layers. This result suggests that routing history forms a cross-depth trajectory rather than a purely adjacent-layer effect, providing useful routing priors for subsequent MoE layers.

To quantify how expert-level interactions vary with layer distance, Figure 3(b) reports the ratio of experts involved in strong negative and positive couplings at different inter-layer distances $\Delta = l - p$. Strong coupling is most prevalent between adjacent layers and decreases over intermediate distances. However, the ratio rises again for larger distances, especially at $\Delta = 13$ and $\Delta = 14$, indicating that while short-range interactions dominate overall, a non-negligible subset of distant layer pairs still exhibits strong expert-level dependencies. Across distances, negative couplings account for the larger proportion, suggesting that cross-layer routing history often encodes expert competition or suppression in addition to cooperative relationships.

Table 2: Time, memory and FLOPs comparison between Baseline and HERO-MoE.

Method	Peak Memory	FLOPs (Router only)	FLOPs (Total)	Forward Time	Router Step Time	Total Train Time
Baseline	5440.8 MiB	1.55 T	211.83 T	32.579 ms	0.14 s / iter	71.88 h
HERO-MoE	5464.9 MiB	2.90 T	213.18 T	37.984 ms	0.54 s / iter	74.65 h
Δ	+0.44%	+87.50%	+0.64%	+16.59%	+281.14%	+3.85%

Figure 3(c) provides a representative expert-level coupling matrix from Layer 8 to Layer 9. The horizontal axis denotes source-layer experts, and the vertical axis denotes target-layer experts. Each entry measures the influence of a source expert on a target expert: red indicates positive cooperation, blue indicates negative competition, and white indicates a near-neutral relation. Vertical red and blue patterns indicate source experts that broadly promote or suppress multiple target experts, whereas vertical white patterns indicate source experts with weak influence on most subsequent experts. Horizontal white patterns indicate target experts that are relatively insensitive to historical expert choices and may be determined primarily by the current token representation. Scattered dark red and dark blue entries further reveal strong pairwise cooperative and competitive relations between specific expert pairs. Additional expert-level coupling matrices for other layer pairs are provided in Appendix Figure 4.

Overall, Figure 3 provides a hierarchical view of the routing structures learned by HERO-MoE and offers a direct perspective for studying cross-layer routing dynamics, expert specialization, and expert composition mechanisms in sparse MoE models.

4.4 Cost of HERO-MoE

As shown in Table 2, we evaluate the computational cost of HERO-MoE against the Baseline. HERO-MoE introduces only a small memory overhead, with peak training memory increasing by 0.44% (+24.2 MiB). The expanded gating dimension increases the standalone router FLOPs from 1.55 T to 2.90 T (+87.50%) and the router step time from 0.14 s/iter to 0.54 s/iter. However, because routing accounts for only a small fraction of the full model computation, the total model step FLOPs increase from 211.83 T to 213.18 T (+0.64%). The end-to-end forward latency increases by 5.405 ms ($\approx$ 0.18 ms per layer), and the overall training wall-clock time increases from 71.88 h to 74.65 h over 25,000 steps, corresponding to a 3.85% overhead. These measurements show that HERO-MoE introduces noticeable router-side overhead but maintains modest end-to-end training overhead in our evaluated setting.

5 Conclusion

In this paper, we introduced HERO-MoE, a historical expert routing framework for sparse Mixture-of-Experts models. Instead of making routing decisions solely from the current token representation, HERO-MoE reuses detached routing distributions from preceding MoE layers within the same pipeline stage and injects them into the current router through a scale-preserving historical residual branch. This design keeps the standard sparse dispatch procedure unchanged, avoids cross-stage communication, and introduces historical routing priors without adding an auxiliary routing loss.

Experiments show that HERO-MoE improves both training dynamics and downstream performance. Compared with the baseline MoE, HERO-MoE reduces the final training loss from 1.6393 to 1.6184 and improves the average benchmark score from 36.57 to 37.42 across 13 evaluation tasks. Ablation studies further show that using richer historical routing trajectories is more effective than relying only on very short history windows. In addition, the trained historical router provides a direct way to visualize layer-wise routing dependencies and expert-level cooperative or competitive relations, offering new insights into how sparse expert models organize routing behavior across depth.

Overall, our results suggest that routing history is a useful and underexplored signal for improving MoE training. By incorporating historical routing information in a lightweight and stage-local manner, HERO-MoE provides a simple, practical, and interpretable direction for building more effective sparse expert models. A current limitation is that routing information is not shared across pipeline-parallel stages, which may limit HERO-MoE's effectiveness at high pipeline-parallel degrees. Future work will extend HERO-MoE to support cross-stage information transfer.

AI ASSISTANCE

The AI support mainly covers text fluency optimization, logical structure sorting and grammar correction, and all related outputs have been manually verified and revised by the authors.

REFERENCES

Sagi Ahrac, Noya Hochwald, and Mor Geva. Routers learn the geometry of their experts: Geometric coupling in sparse mixture-of-experts. *arXiv preprint arXiv:2605.12476*, 2026.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pp. 2357–2367, 2019.

Mark Chen. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

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

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3029–3051, 2023.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, pp. 5547–5569. PMLR, 2022.

Xinrun Du, Zhouliang Yu, Songyang Gao, Ding Pan, Yuyang Cheng, Ziyang Ma, Ruibin Yuan, Xingwei Qu, Jiaheng Liu, Tianyu Zheng, et al. Chinese tiny llm: Pretraining a chinese-centric large language model. *arXiv preprint arXiv:2404.04167*, 2024.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Trevor Gale, Deepak Narayanan, Cliff Young, and Matei Zaharia. Megablocks: Efficient sparse training with mixture-of-experts. *Proceedings of Machine Learning and Systems*, 5:288–304, 2023.

Zijin Gu, Tatiana Likhomanenko, and Navdeep Jaitly. Omni-router: Sharing routing decisions in sparse mixture-of-experts for speech recognition. In *2025 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 1–8. IEEE, 2025.

Hongcan Guo, Haolang Lu, Guoshun Nan, Bolun Chu, Jialin Zhuang, Yuan Yang, Wenhao Che, Xinye Cao, Sicong Leng, Qimei Cui, et al. Advancing expert specialization for better moe. *Advances in Neural Information Processing Systems*, 38:48767–48809, 2026.

Jiaao He, Jiezhong Qiu, Aohan Zeng, Zhilin Yang, Jidong Zhai, and Jie Tang. Fastmoe: A fast mixture-of-expert training system. *arXiv preprint arXiv:2103.13262*, 2021.

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

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

Rizhen Hu, Yuan Cao, Boao Kong, Mou Sun, and Kun Yuan. Synergistic intra-and cross-layer regularization losses for moe expert specialization. *arXiv preprint arXiv:2602.14159*, 2026.

Shuhan Huang, Naifan Zhang, Yuanbo Tang, Yang Li, and Wai Kin Victor Chan. Multi-level context modeling for consistent expert selection in mixture-of-experts. *arXiv preprint arXiv:2607.16427*, 2026.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Advances in neural information processing systems*, 32, 2019.

Changho Hwang, Wei Cui, Yifan Xiong, Ziyue Yang, Ze Liu, Han Hu, Zilong Wang, Rafael Salas, Jithin Jose, Prabhat Ram, et al. Tutel: Adaptive mixture-of-experts at scale. *Proceedings of Machine Learning and Systems*, 5:269–287, 2023.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Takyoung Kim, Janvijay Singh, Shuhaib Mehri, Emre Can Acikgoz, Sagnik Mukherjee, Nimet Beyza Bozdag, Sumuk Shashidhar, Gokhan Tur, and Dilek Hakkani-Tür. Pipa: A unified evaluation protocol for diagnosing interactive planning agents. *arXiv preprint arXiv:2505.01592*, 2025.

Kirill Labzin, Stepan Kulibaba, Artem Dzhalilov, and Artem Gorokhov. Evidence for shared routing geometry and dynamics in sparse mixture-of-experts. *arXiv preprint arXiv:2609.02404*, 2026.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

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. *arXiv preprint arXiv:2006.16668*, 2020.

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

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Scott Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282, 2024b.

Wengang Li, Lingqi Zhang, Toshio Endo, and Mohamed Wahib. Understanding cross-layer contributions to mixture-of-experts routing in llms. In *The Fourteenth International Conference on Learning Representations*, 2026.

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

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

Yan Liu, Renren Jin, Ling Shi, Zheng Yao, and Deyi Xiong. Finemath: A fine-grained mathematical evaluation benchmark for chinese large language models. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 2024c.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, et al. Olmoe: Open mixture-of-experts language models. In *International Conference on Learning Representations*, volume 2025, pp. 62061–62121, 2025.

Matthew Lyle Olson, Neale Ratzlaff, Musashi Hinck, Man Luo, Sungduk Yu, Chendi Xue, and Vasudev Lal. Probing semantic routing in large mixture-of-expert models. In *EMNLP (Findings)*, pp. 18263–18278, 2025.

Nabil Omi, Siddhartha Sen, and Ali Farhadi. Load balancing mixture of experts with similarity preserving routers. *arXiv preprint arXiv:2506.14038*, 2025.

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 (volume 1: Long papers)*, pp. 1525–1534, 2016.

Zihan Qiu, Zeyu Huang, Shuang Cheng, Yizhi Zhou, Ivan Titov, Jie Fu, et al. Layerwise recurrent router for mixture-of-experts. In *International Conference on Learning Representations*, volume 2025, pp. 14179–14205, 2025.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. In *International conference on machine learning*, pp. 18332–18346. PMLR, 2022.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Fan Zhou, Zengzhi Wang, Nikhil Ranjan, Zhoujun Cheng, Liping Tang, Guowei He, Zhengzhong Liu, and Eric P Xing. Megamath: Pushing the limits of open math corpora. *arXiv preprint arXiv:2504.02807*, 2025.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.

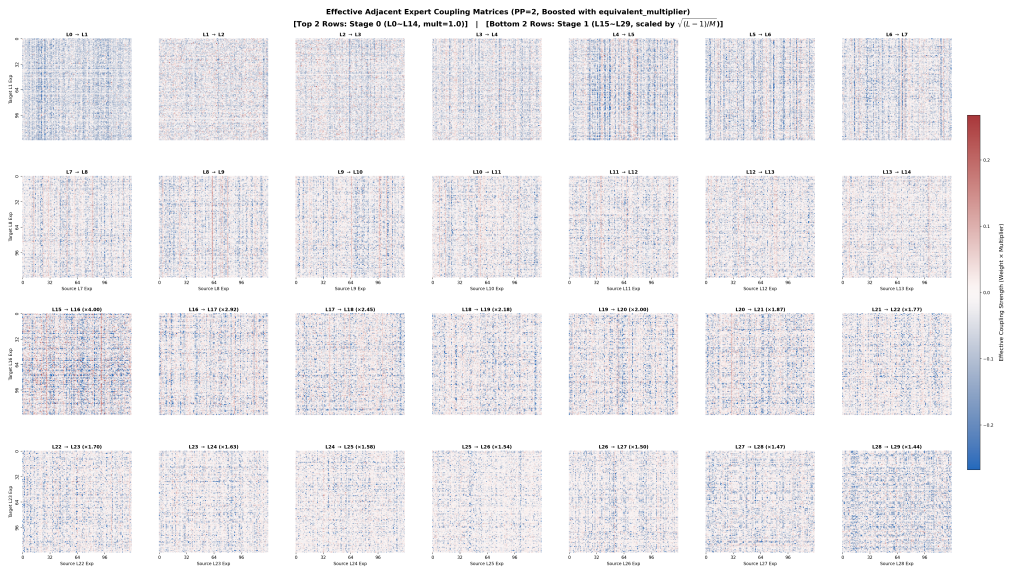

Figure 4: Expert-level coupling matrices for all adjacent MoE-layer pairs learned by HERO-MoE.

APPENDIX

A MORE EXPERT-LEVEL SAMPLES

To complement the representative coupling matrix shown in Figure 3(c), Figure 4 presents the expert-level coupling matrices for all adjacent MoE-layer pairs, and the later fourteen plots are multiplied by the second factor in Eq. (10). Across different layer pairs, the matrices exhibit structured positive and negative regions rather than unstructured noise, indicating that the historical routing branch captures non-trivial cooperation and competition between experts. These patterns are not identical across depth: some source experts exert broad promoting or suppressing effects on multiple target experts, whereas others have only localized or weak influence. Likewise, several target experts remain relatively insensitive to the preceding routing state, suggesting that their assignments are determined primarily by the current token representation. Overall, Figure 4 shows that the coupling observed in Figure 3(c) is not an isolated case, but reflects a broader, layer-dependent organization of cross-layer expert interactions learned by HERO-MoE.

B EXPERT ROUTING DYNAMICS

Figure 5 illustrates the layer-wise expert activation distribution across all 30 layers and 128 experts, evaluated on multiple benchmarks. In the baseline model, Expert 94 in Layer 18 monopolizes 13.74% of all tokens, creating a severe bottleneck and exacerbating the straggler problem in distributed All-to-All communication. In contrast, HERO-MoE reduces the peak expert load by 31.2%, down to 9.45%, thereby effectively alleviating cluster communication congestion. More importantly, HERO-MoE accelerates model training convergence while maintaining balanced expert utilization.

C ADDITIONAL RESULTS ON QWEN3

Figure 6(a) compares the training loss of Qwen3 (Yang et al., 2025) with and without HERO-MoE under the Qwen3 experimental configuration. HERO-MoE generally achieves a lower loss, especially during the later training stage, reducing the final loss from 1.9056 to 1.8985. Figure 6(b) reports the loss advantage over the original Qwen3 model, computed as the Qwen3 loss minus the loss of Qwen3 with HERO-MoE. Without model-specific tuning, although the advantage fluctuates

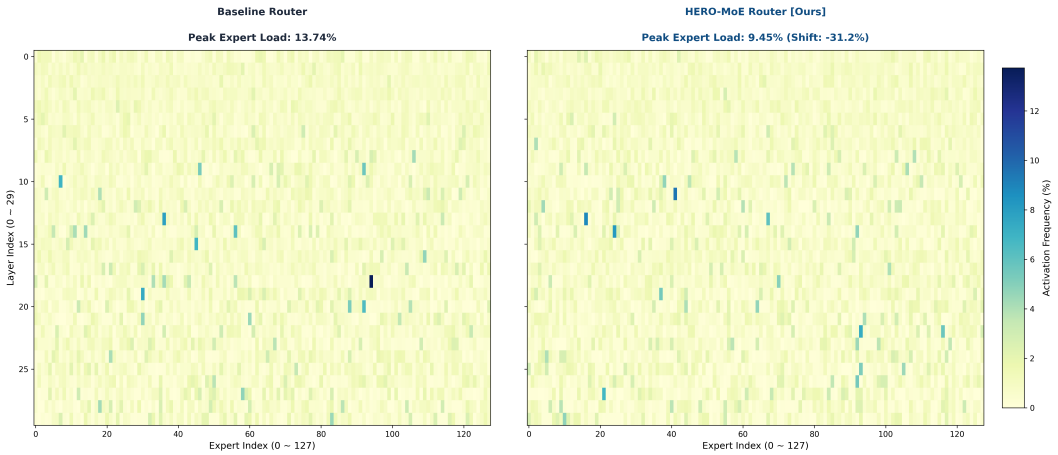

Figure 5: Expert utilization frequency across 30 layers and 128 experts.

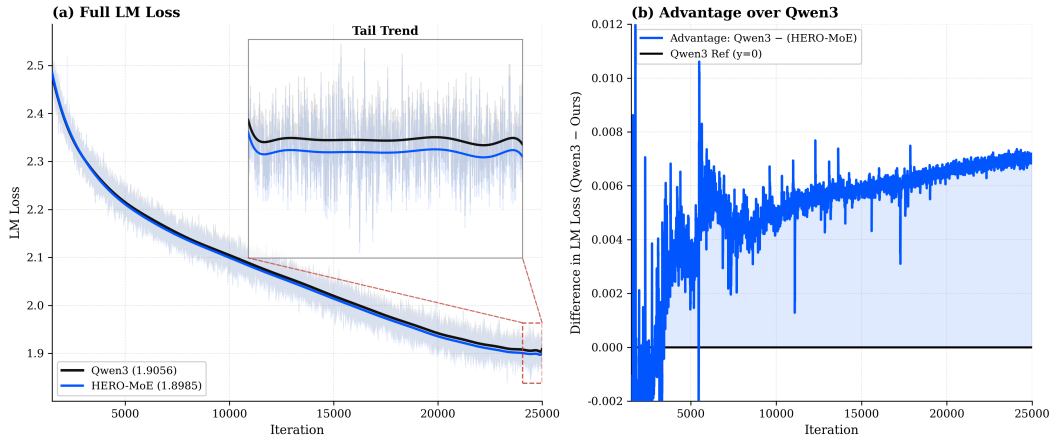

Figure 6: Training loss and loss advantage of Qwen3 with HERO-MoE.

considerably during early training, it remains predominantly positive and continues to increase during the later stage. These results further demonstrate the generality of HERO-MoE across MoE models with different architectures.

D Mathematical Properties of HERO-MoE

This appendix formalizes the structural properties of HERO-MoE. We use the same notation as in the method section. In particular, let $\mathcal{S}(l)$ be the set of preceding MoE layers on the same pipeline stage as layer l, and let $M_l = |\mathcal{S}(l)|$. The results below concern the explicit historical routing path. They establish structural and optimization properties of the proposed parameterization; improvements in training loss, expert utilization, and downstream accuracy remain empirical claims.

D.1 Notation and Assumptions

For $M_l > 0$, write the detached historical distributions as

$$\boldsymbol{Q}^{(j)} = \mathrm{StopGrad}\Big(\mathrm{softmax}\Big(\boldsymbol{Z}^{(j)}\Big)\Big) \in \mathbb{R}^{N\times E}, \qquad j \in \mathcal{S}(l), \tag{16}$$

and define

$$\boldsymbol{H}^{(l)} = \Big[\boldsymbol{Q}^{(j_1)}, \boldsymbol{Q}^{(j_2)}, \ldots, \boldsymbol{Q}^{(j_{M_l})}\Big] \in \mathbb{R}^{N\times M_l E}. \tag{17}$$

For a token row, let

$$r_x = \mathrm{RMS}\Big(\boldsymbol{x}_t^{(l)}\Big), \qquad r_h = \mathrm{RMS}\Big(\boldsymbol{h}_t^{(l)}\Big). \tag{18}$$

Define the scale-preserving coefficient as

$$\rho_t^{(l)} = \frac{r_x}{r_h + \epsilon}\sqrt{\frac{l-1}{M_l}}, \qquad \epsilon > 0. \tag{19}$$

When an exact identity is stated, we first consider the idealized limit $\epsilon = 0$ and then give the corresponding expression for the finite-ϵ case.

D.2 History-aware router decomposition

Proposition 1 (Baseline recovery). If $M_l = 0$, HERO-MoE exactly reduces to the conventional linear MoE router. If $M_l > 0$, the history-aware router is algebraically equivalent to an original-state branch plus a historical residual branch.

Proof. Partition the router weight as

$$\boldsymbol{W}^{(l)} = \Big[\boldsymbol{W}_O^{(l)} \Big\| \boldsymbol{W}_R^{(l)}\Big], \qquad \boldsymbol{W}_O^{(l)} \in \mathbb{R}^{E\times d}, \quad \boldsymbol{W}_R^{(l)} \in \mathbb{R}^{E\times M_l E}. \tag{20}$$

The input to the single linear projection is

$$\boldsymbol{U}^{(l)} = \Big[\boldsymbol{X}^{(l)} \Big\| \boldsymbol{\rho}^{(l)} \odot \boldsymbol{H}^{(l)}\Big]. \tag{21}$$

Block matrix multiplication gives

$$\begin{aligned} \boldsymbol{Z}^{(l)} &= \boldsymbol{U}^{(l)}\boldsymbol{W}^{(l)\top} \\ &= \boldsymbol{X}^{(l)}\boldsymbol{W}_O^{(l)\top} + \Big(\boldsymbol{\rho}^{(l)} \odot \boldsymbol{H}^{(l)}\Big)\boldsymbol{W}_R^{(l)\top}. \end{aligned} \tag{22}$$

When $M_l = 0$, the second block is absent, so $\boldsymbol{Z}^{(l)} = \boldsymbol{X}^{(l)}\boldsymbol{W}_O^{(l)\top}$, which is the baseline router. □

Corollary 1 (Unchanged sparse dispatch). HERO-MoE changes only the logits supplied to the configured routing operator. Consequently, the subsequent softmax, top-k selection, group-limited selection, capacity handling, and token dispatch operations are unchanged. In particular, for any routing operator $\mathcal{R}$ acting on router logits,

$$\text{HERO-MoE routing} = \mathcal{R}\Big(\boldsymbol{Z}_O^{(l)} + \boldsymbol{Z}_R^{(l)}\Big), \tag{23}$$

with the same operator $\mathcal{R}$ as in the baseline model.

D.3 Gradient isolation of historical routing

Proposition 2 (Stop-gradient isolation). Let θ_j denote parameters that produce the router logits $\boldsymbol{Z}^{(j)}$ of a preceding layer. Along the explicit historical path, the current layer satisfies

$$\frac{\partial \boldsymbol{H}^{(l)}}{\partial \theta_j} = \boldsymbol{0}, \qquad \frac{\partial \boldsymbol{Z}_R^{(l)}}{\partial \theta_j} = \boldsymbol{0}. \tag{24}$$

Proof. For every visible predecessor j, the historical routing distribution is

$$\boldsymbol{Q}^{(j)} = \mathrm{StopGrad}\Big(\mathrm{softmax}\Big(\boldsymbol{Z}^{(j)}\Big)\Big). \tag{25}$$

By definition of the stop-gradient operator, its value is preserved in the forward pass while its derivative with respect to its input is zero. Therefore,

$$\frac{\partial \boldsymbol{Q}^{(j)}}{\partial \theta_j} = \boldsymbol{0}. \tag{26}$$

Concatenation is linear with respect to its inputs, so the same property holds for $\boldsymbol{H}^{(l)}$. Since the RMS statistics used in $\boldsymbol{\rho}^{(l)}$ are also treated as stop-gradient quantities, the product $\boldsymbol{\rho}^{(l)} \odot \boldsymbol{H}^{(l)}$ has no gradient through the explicit historical path. Multiplication by the trainable matrix $\boldsymbol{W}_R^{(l)\top}$ therefore yields Eq. (24). This statement does not remove gradients that may reach earlier layers through the ordinary hidden-state path $\boldsymbol{X}^{(l)}$. □

D.4 SCALE-PRESERVING HISTORICAL FUSION

Proposition 3 (RMS calibration identity). For a token with $r_h > 0$, the scaled historical vector

$$\widetilde{\boldsymbol{h}}_t^{(l)} = \rho_t^{(l)} \boldsymbol{h}_t^{(l)} \tag{27}$$

satisfies

$$\mathrm{RMS}\Big(\widetilde{\boldsymbol{h}}_t^{(l)}\Big) = r_x \sqrt{\frac{l-1}{M_l}} \frac{r_h}{r_h + \epsilon}. \tag{28}$$

In the idealized case $\epsilon = 0$,

$$\mathrm{RMS}\Big(\widetilde{\boldsymbol{h}}_t^{(l)}\Big) = r_x \sqrt{\frac{l-1}{M_l}}. \tag{29}$$

Proof. The coefficient $\rho_t^{(l)}$ is a nonnegative scalar shared by all components of the token row. Hence

$$\mathrm{RMS}\Big(\rho_t^{(l)} \boldsymbol{h}_t^{(l)}\Big) = \rho_t^{(l)}\, \mathrm{RMS}\Big(\boldsymbol{h}_t^{(l)}\Big) = \rho_t^{(l)} r_h. \tag{30}$$

Substituting Eq. (19) proves Eq. (28), and setting $\epsilon = 0$ proves Eq. (29). □

Corollary 2 (Equivalent historical energy). Let the historical vector have dimension $M_l E$. For $\epsilon = 0$,

$$\Big\|\widetilde{\boldsymbol{h}}_t^{(l)}\Big\|_2^2 = (l-1) E r_x^2. \tag{31}$$

Thus, after RMS alignment, the visible history has the same squared energy as $l-1$ virtual historical blocks of width E, regardless of the number M_l of blocks that are locally visible. For $\epsilon > 0$, the finite-ϵ expression is

$$\Big\|\widetilde{\boldsymbol{h}}_t^{(l)}\Big\|_2^2 = (l-1) E r_x^2 \left(\frac{r_h}{r_h + \epsilon}\right)^2, \tag{32}$$

which differs from the ideal identity only by the numerical-stability factor in parentheses. If $r_h = 0$, the historical vector is already zero; the positive ϵ prevents an undefined division but cannot recover information that is absent from the history tensor.

Proof. By definition of RMS for a vector in $\mathbb{R}^{M_l E}$,

$$\Big\|\widetilde{\boldsymbol{h}}_t^{(l)}\Big\|_2^2 = M_l E\ \mathrm{RMS}\Big(\widetilde{\boldsymbol{h}}_t^{(l)}\Big)^2. \tag{33}$$

Substitution of Eq. (28) gives

$$\Big\|\widetilde{\boldsymbol{h}}_t^{(l)}\Big\|_2^2 = M_l E \left[r_x \sqrt{\frac{l-1}{M_l}} \frac{r_h}{r_h + \epsilon}\right]^2, \tag{34}$$

which simplifies to Eq. (32). Setting $\epsilon = 0$ gives Eq. (31). □

Interpretation. The factor $\sqrt{(l-1)/M_l}$ is therefore an energy compensation term. When pipeline parallelism exposes fewer predecessor layers ($M_l < l-1$), the visible history is amplified so that its total effective energy remains comparable to a single-stage router with $l-1$ historical blocks. This does not claim that the individual expert probabilities become uniform or that routing load is automatically balanced; those are separate learned properties.

D.5 INITIALIZATION AND LOGIT-SCALE STABILITY

Proposition 4 (Depth-independent initial logit variance). For a layer with $M_l > 0$, assume that the entries of the router weight are independent and sampled from

$$w_{e,i}^{(l)} \sim \mathcal{U}(-b_l, b_l), \qquad b_l = \frac{1}{\sqrt{d + (l-1)E}}, \tag{35}$$

and that the weights are independent of the router input. Suppose further that, for some $\sigma_x^2 > 0$,

$$\mathbb{E}\left[\Big\|\boldsymbol{x}_t^{(l)}\Big\|_2^2\right] = d\sigma_x^2, \qquad \mathbb{E}\left[\Big\|\widetilde{\boldsymbol{h}}_t^{(l)}\Big\|_2^2\right] = (l-1)E\sigma_x^2. \tag{36}$$

Then, under the idealized $\epsilon = 0$ calibration, every initial router logit has variance

$$\mathrm{Var}\Big(Z_{t,e}^{(l)}\Big) = \frac{\sigma_x^2}{3}, \tag{37}$$

independently of the layer depth l and the number of locally visible history layers M_l.

When $M_l = 0$, the historical branch is absent. Under the same initialization bound, the router receives only the current hidden representation, and the logit variance becomes

$$\mathrm{Var}\Big(Z_{t,e}^{(l)}\Big) = \frac{d}{d + (l-1)E}\,\frac{\sigma_x^2}{3}. \tag{38}$$

For the first layer, where $l = 1$, this reduces to $\sigma_x^2/3$.

Proof. Let

$$\boldsymbol{u}_t^{(l)} = \Big[\boldsymbol{x}_t^{(l)} \Big\| \widetilde{\boldsymbol{h}}_t^{(l)}\Big] \tag{39}$$

and let $D_l = d + (l-1)E$. For a uniform random variable on $[-b_l, b_l]$, the variance is $b_l^2/3$. Since the weights have zero mean and are independent of the input,

$$\begin{aligned}
\mathrm{Var}\Big(Z_{t,e}^{(l)}\Big) &= \frac{b_l^2}{3}\mathbb{E}\left[\Big\|\boldsymbol{u}_t^{(l)}\Big\|_2^2\right] \\
&= \frac{1}{3D_l}\left(d\sigma_x^2 + (l-1)E\sigma_x^2\right) \\
&= \frac{\sigma_x^2}{3}.
\end{aligned} \tag{40}$$

For layers with $M_l > 0$, the number of locally visible history layers does not appear in the final expression because Eq. (31) calibrates the effective historical energy to $(l-1)E\sigma_x^2$. □

Remark. Proposition 4 is a variance calculation under explicit independence and second-moment assumptions; it is not a guarantee about the trained network. When $\epsilon > 0$, the historical energy is attenuated by the factor in Eq. (32). Under the same assumptions, the historical contribution to the initial router-logit variance is therefore reduced by the corresponding factor.

D.6 Branch-wise Initial Logit Variance Decomposition

The preceding analysis characterizes the variance of the combined router logits. We further decompose this variance into contributions from the original current-token branch and the historical residual branch. Consider a layer with $M_l > 0$ and the idealized calibration $\epsilon = 0$. For token t and expert e, write

$$Z_{O,t,e}^{(l)} = \boldsymbol{x}_t^{(l)}\left(\boldsymbol{w}_{O,e}^{(l)}\right)^\top, \qquad Z_{R,t,e}^{(l)} = \widetilde{\boldsymbol{h}}_t^{(l)}\left(\boldsymbol{w}_{R,e}^{(l)}\right)^\top, \tag{41}$$

so that

$$Z_{t,e}^{(l)} = Z_{O,t,e}^{(l)} + Z_{R,t,e}^{(l)}. \tag{42}$$

Proposition 5 (Branch-wise variance decomposition). Under the assumptions of Proposition 4, let

$$D_l = d + (l-1)E.$$

Assume that the entries of the original and residual router weight blocks are independently sampled from $\mathcal{U}(-1/\sqrt{D_l}, 1/\sqrt{D_l})$. Then the initial logit variances of the two branches are

$$\mathrm{Var}\Big(Z_{O,t,e}^{(l)}\Big) = \frac{d}{d + (l-1)E}\,\frac{\sigma_x^2}{3}, \tag{43}$$

and

$$\mathrm{Var}\Big(Z_{R,t,e}^{(l)}\Big) = \frac{(l-1)E}{d + (l-1)E}\,\frac{\sigma_x^2}{3}. \tag{44}$$

Moreover, the two branches are uncorrelated at initialization:

$$\mathrm{Cov}\Big(Z^{(l)}_{O,t,e}, Z^{(l)}_{R,t,e}\Big) = 0. \tag{45}$$

Consequently,

$$\mathrm{Var}\Big(Z^{(l)}_{t,e}\Big) = \mathrm{Var}\Big(Z^{(l)}_{O,t,e}\Big) + \mathrm{Var}\Big(Z^{(l)}_{R,t,e}\Big) = \frac{\sigma_x^2}{3}. \tag{46}$$

Proof. Because the entries of each router weight block have variance $1/(3D_l)$, the original branch satisfies

$$\begin{aligned}\mathrm{Var}\Big(Z^{(l)}_{O,t,e}\Big) &= \frac{1}{3D_l}\mathbb{E}\left[\left\|\boldsymbol{x}_t^{(l)}\right\|_2^2\right] \\ &= \frac{d\sigma_x^2}{3D_l} = \frac{d}{d+(l-1)E}\frac{\sigma_x^2}{3}.\end{aligned} \tag{47}$$

Similarly, using the idealized energy identity in Eq. (31),

$$\begin{aligned}\mathrm{Var}\Big(Z^{(l)}_{R,t,e}\Big) &= \frac{1}{3D_l}\mathbb{E}\left[\left\|\widetilde{\boldsymbol{h}}_t^{(l)}\right\|_2^2\right] \\ &= \frac{(l-1)E\sigma_x^2}{3D_l} = \frac{(l-1)E}{d+(l-1)E}\frac{\sigma_x^2}{3}.\end{aligned} \tag{48}$$

The covariance is zero because the two weight blocks are independently sampled and have zero mean. Adding the two branch variances gives Eq. (46). □

Interpretation. The scale-preserving initialization keeps the total initial router-logit variance independent of layer depth, but it does not force the original and historical branches to have equal contributions. Their idealized variance fractions are

$$\eta_O^{(l)} = \frac{d}{d+(l-1)E}, \qquad \eta_R^{(l)} = \frac{(l-1)E}{d+(l-1)E}, \qquad \eta_O^{(l)} + \eta_R^{(l)} = 1. \tag{49}$$

Thus, under the global fan-in used by HERO-MoE, the relative contribution of the historical branch increases with depth while the total logit scale remains controlled. The number of locally visible history layers M_l does not appear in these idealized variance fractions because the scale-preserving factor compensates for the corresponding historical energy. With finite $\epsilon > 0$, the historical contribution is attenuated in a token-dependent manner, and the exact residual-branch variance depends on the distribution of r_h.

D.7 Theoretical implications

The preceding propositions establish the following properties of HERO-MoE:

1. The proposed router is a strict extension of the baseline router: removing historical information recovers the original routing function exactly.
2. Historical routing priors provide an additional decision signal without creating an explicit gradient path through earlier routing distributions.
3. For layers with $M_l > 0$, the factor $\sqrt{(l-1)/M_l}$ compensates for the number of visible historical layers, making the calibrated effective historical energy independent of M_l in the idealized $\epsilon = 0$ setting.
4. The historical branch modifies only the router logits. Therefore, any standard sparse routing operator can be applied after the proposed router without changing its definition.

Together, these results provide a theoretical explanation for the three design goals of HERO-MoE: compatibility with standard sparse MoE routing, decoupling of historical priors from earlier router gradients, and calibrated historical signal magnitude across depth and different numbers of visible historical layers.

Algorithm 1: Depth-aware Initialization of the HERO-MoE Router

Input : Total number of MoE layers L, current layer index l, hidden dimension d, number of experts E, pipeline-stage assignment $\pi(\cdot)$

Output: Router weight $\boldsymbol{W}^{(l)}$

// 1. Determine the visible historical layers

$\mathcal{S}(l) \leftarrow \{j \mid j < l,\ \pi(j) = \pi(l)\}$;

$M_l \leftarrow |\mathcal{S}(l)|$;

// 2. Determine the physical router dimension

$D_l \leftarrow d + M_l E$;

// $D_l = d$ when no preceding layer is visible

// 3. Use the global depth to determine the initialization scale

$b_l \leftarrow \dfrac{1}{\sqrt{d + (l-1)E}}$;

// 4. Initialize the single concatenated router matrix

Initialize each entry $W_{e,i}^{(l)} \sim \mathcal{U}(-b_l, b_l)$ for $e \in \{1, \ldots, E\}$ and $i \in \{1, \ldots, D_l\}$;

if $M_l = 0$ **then**

  $\boldsymbol{W}_O^{(l)} \leftarrow \boldsymbol{W}^{(l)}$;

  $\boldsymbol{W}_R^{(l)} \leftarrow \emptyset$;

else

  Partition $\boldsymbol{W}^{(l)}$ as $\boldsymbol{W}^{(l)} = \left[\boldsymbol{W}_O^{(l)} \middle\| \boldsymbol{W}_R^{(l)}\right]$;

  $\boldsymbol{W}_O^{(l)} \in \mathbb{R}^{E \times d}$, $\boldsymbol{W}_R^{(l)} \in \mathbb{R}^{E \times M_l E}$;

return $\boldsymbol{W}^{(l)}$

E ALGORITHM OF HERO-MOE

The pseudocode of HERO-MoE is divided into two procedures. Algorithm 1 initializes the history-aware router with a depth-aware global fan-in. Algorithm 2 performs stage-local history retrieval, scale-preserving fusion, router-logit computation, and standard sparse expert selection.

Algorithm 2: Stage-local Historical Routing in HERO-MoE

Input : Current hidden states $\boldsymbol{X}^{(l)} \in \mathbb{R}^{N \times d}$, current layer index l, pipeline-stage assignment $\pi(\cdot)$, preceding router logits $\{\boldsymbol{Z}^{(j)}\}_{j<l}$, router weight $\boldsymbol{W}^{(l)}$, numerical constant $\epsilon > 0$, sparse routing configuration $\mathcal{C}$

Output: Router logits $\boldsymbol{Z}^{(l)}$, routing weights $\boldsymbol{G}^{(l)}$, selected expert indices $\boldsymbol{I}^{(l)}$, dense routing distribution $\boldsymbol{Q}^{(l)}$

// 1. Identify stage-local routing history

$\mathcal{S}(l) \leftarrow \{j \mid j < l,\ \pi(j) = \pi(l)\}$;

$M_l \leftarrow |\mathcal{S}(l)|$;

if $M_l = 0$ **then**

  $\boldsymbol{U}^{(l)} \leftarrow \boldsymbol{X}^{(l)}$;

else

  // 2. Collect detached dense routing distributions

  Enumerate $\mathcal{S}(l) = \{j_1, j_2, \ldots, j_{M_l}\}$;

  for $m \leftarrow 1$ **to** M_l **do**

    $\boldsymbol{Q}^{(j_m)} \leftarrow \texttt{StopGrad}\Big(\texttt{softmax}\Big(\boldsymbol{Z}^{(j_m)}\Big)\Big)$;

  $\boldsymbol{H}^{(l)} \leftarrow \left[\boldsymbol{Q}^{(j_1)}, \boldsymbol{Q}^{(j_2)}, \ldots, \boldsymbol{Q}^{(j_{M_l})}\right]$;

  // 3. Apply token-wise scale-preserving fusion

  $\boldsymbol{r}_x \leftarrow \texttt{RMS}\Big(\boldsymbol{X}^{(l)}\Big)$;

  $\boldsymbol{r}_h \leftarrow \texttt{RMS}\Big(\boldsymbol{H}^{(l)}\Big)$;

  $\boldsymbol{\rho}^{(l)} \leftarrow \dfrac{\boldsymbol{r}_x}{\boldsymbol{r}_h + \epsilon}\sqrt{\dfrac{l-1}{M_l}}$;

  $\widetilde{\boldsymbol{H}}^{(l)} \leftarrow \boldsymbol{\rho}^{(l)} \odot \boldsymbol{H}^{(l)}$;

  // 4. Concatenate current features and historical features

  $\boldsymbol{U}^{(l)} \leftarrow \left[\boldsymbol{X}^{(l)} \middle\| \widetilde{\boldsymbol{H}}^{(l)}\right]$;

// 5. Compute history-aware router logits

$\boldsymbol{Z}^{(l)} \leftarrow \boldsymbol{U}^{(l)} \boldsymbol{W}^{(l)\top}$;

// 6. Form dense routing probabilities for the current layer

$\boldsymbol{Q}^{(l)} \leftarrow \texttt{softmax}\Big(\boldsymbol{Z}^{(l)}\Big)$;

// 7. Apply the configured sparse routing policy

$(\boldsymbol{G}^{(l)}, \boldsymbol{I}^{(l)}) \leftarrow \texttt{SparseRoute}\Big(\boldsymbol{Z}^{(l)}, \boldsymbol{Q}^{(l)}, \mathcal{C}\Big)$;

// $\mathcal{C}$ may specify top-k, group-limited routing, capacity, and top-k weight normalization

return $\boldsymbol{Z}^{(l)},\ \boldsymbol{G}^{(l)},\ \boldsymbol{I}^{(l)},\ \boldsymbol{Q}^{(l)}$

F VISUALIZATION AND STATISTICAL DETAILS

This section describes the data processing and statistics used to generate Figures 3.

Layer-wise dependency and expert-level coupling. The routing analysis uses the trained router weights of a 30-layer MoE model with $E = 128$ experts and hidden dimension $d = 1024$. The model is partitioned into two pipeline-parallel stages, with 15 layers per stage. The implementation uses zero-based layer indices $0, \ldots, 29$, whereas the equations in the main text use one-based layer indices. Thus, an implementation layer index l_0 corresponds to the mathematical layer index $l = l_0 + 1$.

For a target layer l and a preceding layer p located in the same pipeline-parallel stage, let

$$\boldsymbol{W}_{\mathrm{R}}^{(l,p)} \in \mathbb{R}^{E \times E}$$

denote the residual-router weight block associated with the routing history from layer p to layer l. For each visible predecessor, the block is extracted from the columns following the first d current-token features. The layer-wise dependency strength shown in Figure 3(a) is computed as

$$D_{l,p} = \left\| \boldsymbol{W}_{\mathrm{R}}^{(l,p)} \right\|_{\mathrm{F}} \sqrt{\frac{l-1}{M_l}}, \qquad p < l, \tag{50}$$

where M_l is the number of preceding layers visible on the same pipeline-parallel stage. The multiplier is the implementation equivalent of the depth-compensation factor in Eq. (**??**); it is computed as $\sqrt{l_0/M_l}$ using the zero-based layer index l_0. Entries corresponding to cross-stage layer pairs are set to zero because HERO-MoE uses stage-local routing history. Therefore, Figure 3(a) contains two independent 15×15 diagonal blocks.

To quantify how strong expert-level interactions vary with layer distance, we define

$$\mathcal{W}_\Delta = \left\{ \left(\boldsymbol{W}_{\mathrm{R}}^{(l,p)} \right)_{a,b} : l - p = \Delta,\ p \text{ and } l \text{ are in the same stage} \right\}, \tag{51}$$

where $\Delta \in \{1, \ldots, 14\}$ and $a, b \in \{1, \ldots, E\}$. A single global threshold is computed from all historical-router entries:

$$\tau = \mathrm{Percentile}_{90} \left(\left\{ |w| : w \in \bigcup_{\Delta=1}^{14} \mathcal{W}_\Delta \right\} \right). \tag{52}$$

For each inter-layer distance Δ, the negative and positive strong-coupling ratios are then calculated as

$$r_\Delta^- = \frac{100}{|\mathcal{W}_\Delta|} \sum_{w \in \mathcal{W}_\Delta} \mathbb{1}[w < -\tau], \qquad r_\Delta^+ = \frac{100}{|\mathcal{W}_\Delta|} \sum_{w \in \mathcal{W}_\Delta} \mathbb{1}[w > \tau]. \tag{53}$$

Figure 3(b) visualizes r_Δ^- and r_Δ^+ as stacked bars. Thus, this panel reports the fraction of strongly negative or positive coupling entries among all expert-pair weights at a given layer distance, rather than the fraction of activated experts.

Finally, Figure 3(c) visualizes the raw residual weight block for the implementation pair Layer 8 $\rightarrow$ Layer 9. The horizontal axis corresponds to source experts, and the vertical axis corresponds to target experts. The color range is symmetric around zero and is clipped at the 99.8th percentile of the absolute matrix values to reduce the influence of extreme entries. Positive and negative colors therefore indicate the signs of learned residual-router coefficients. These signs serve as a weight-based structural proxy for positive and negative routing influence, rather than a direct causal measurement of expert cooperation or competition.