COUPLING EXPERTS AND ROUTERS IN MIXTURE-OF-EXPERTS VIA AN AUXILIARY LOSS

Anonymous authors
Paper under double-blind review

ABSTRACT

Traditional Mixture-of-Experts (MoE) models lack explicit constraints to ensure the router's decisions align well with the experts' capabilities, which ultimately limits model performance. To address this, we propose expert-router coupling loss (ERC loss), a lightweight auxiliary loss that couples expert capabilities and the router's decisions. We treat each row of the router matrix as a cluster center for the tokens assigned to a particular expert. From these centers, we create proxy tokens by applying a perturbation with noise. Using these proxy tokens, the ERC loss forces the router and experts to satisfy two constraints: (1) each expert exhibits higher activation for its corresponding proxy token than for any other proxy token, and (2) each proxy token elicits stronger activation in its designated expert than in any other expert. This optimization leads to two key effects: each row of the router matrix is an accurate representation of its expert's capabilities, while each expert develops expertise that closely match the tokens routed to it. Our experiments involve pre-training multiple 3B-parameter MoE-LLMs on trillions of tokens in total, providing detailed evidence of the ERC loss's effectiveness. Our method remains effective and stable as we scale the models up to 15B parameters.[1] Moreover, the ERC loss offers flexible control and quantitative tracking of expert specialization levels during training, providing many valuable insights into MoEs.

1 INTRODUCTION

Mixture-of-Experts (MoE, Shazeer et al., 2017; Fedus et al., 2022; Lepikhin et al., 2021; Zoph et al., 2022) is a core architecture in modern large language models (LLMs). In MoE models, the feed-forward layer is split into multiple small, specialized "experts." A linear classifier, known as the "router," selects which experts process each input token. By activating a few experts per token, MoE balances efficiency with scaled parameter counts, enabling the training of trillion-parameter models.

Ideally, a router should possess an accurate representation of each expert's capabilities to enable effective token routing. However, traditional MoEs offer no explicit constraints to guarantee this. Without direct access to expert parameters (and therefore their true capabilities), routers resort to trial-and-error learning of routing strategies, often resulting in misrouted tokens whose gradients interfere with expert specialization. While some methods (Lv et al., 2025; Pham et al., 2024) incorporated all experts' activations for routing guidance, they incur substantial computational and memory costs due to denser activation. A lightweight and effective solution to better couple routing decisions with true expert capabilities remains an open challenge.

We propose expert-router coupling loss (ERC loss), a novel auxiliary loss for MoE models that tightly couples routers and experts with negligible overhead. The loss is based on interpreting the router parameter matrix $\boldsymbol{R} \in \mathbb{R}^{n \times d}$ as cluster centers, where each row $\boldsymbol{R}[i]$ serves as the center for the token set $\mathcal{X}_i$ routed to expert i. The ERC loss comprises three key steps:

(1) Each $\boldsymbol{R}[i]$ is augmented with bounded random noise $\boldsymbol{\delta}_i$ to obtain $\tilde{\boldsymbol{R}}[i]$, serving as a proxy for tokens in $\mathcal{X}_i$. Here, $\boldsymbol{\delta}_i$ is bounded by half the minimum distance between adjacent cluster centers,

[1] Red text highlights new experiments added during the rebuttal period. Blue text indicates the original content that has been slightly modified.

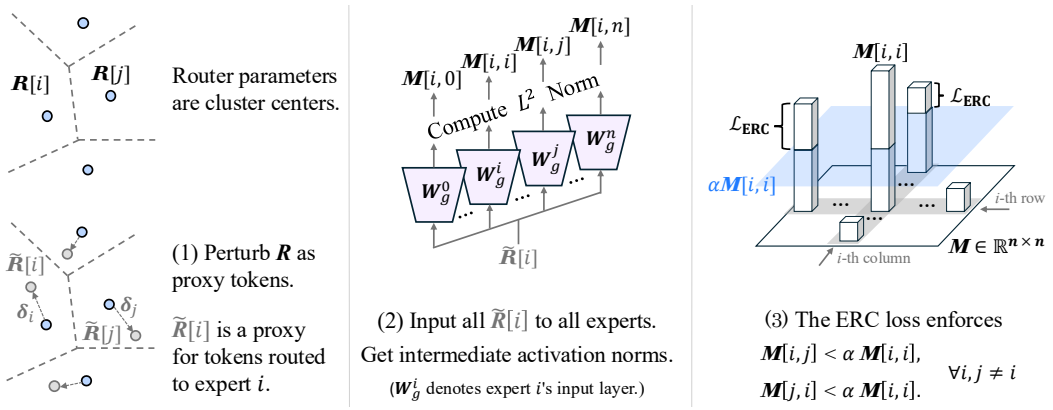


Figure 1: Three steps for computing the expert-router coupling loss.

ensuring that the noise simulates input variations within $\mathcal{X}_i$ while preventing the crossing of cluster boundaries.

(2) Inspired by prior works (Geva et al., 2021; Liu et al., 2023; Lv et al., 2025), the intermediate activation norm serves as an indicator of how well its capabilities align with the token. We measure the intermediate activation norms of all experts that take $\tilde{\boldsymbol{R}}[i]$ as input. This step produces a matrix $\boldsymbol{M} \in \mathbb{R}^{n\times n}$, with $\boldsymbol{M}[i,j]$ being the activation norm from expert j given input $\tilde{\boldsymbol{R}}[i]$.

(3) For all $i \neq j$, the ERC loss imposes a penalty wherever the off-diagonal elements $\boldsymbol{M}[i,j]$ or $\boldsymbol{M}[j,i]$ exceed $\alpha\boldsymbol{M}[i,i]$, where α is a scalar hyperparameter:

$$\mathcal{L}_{\text{ERC}} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j\neq i}^{n}\left(\max\left(\boldsymbol{M}[i,j] - \alpha\boldsymbol{M}[i,i], 0\right) + \max\left(\boldsymbol{M}[j,i] - \alpha\boldsymbol{M}[i,i], 0\right)\right).$$

Minimizing it tightly couples experts and routers through two effects:

- Expert specialization: The proxy token $\tilde{\boldsymbol{R}}[i]$ elicits the strongest activation from expert i versus all other experts. This indicates that expert i is optimized to best match the features of its assigned token cluster $\mathcal{X}_i$.
- Precise token routing: Expert i is most activated by its designated vector $\tilde{\boldsymbol{R}}[i]$ than to any other $\tilde{\boldsymbol{R}}[j]$ for $j \neq i$. This demonstrates that $\boldsymbol{R}[i]$ aligns well with the capabilities of expert i, ensuring that the router assigns to this expert the tokens that need it most.

We conducted large-scale pre-training experiments on models from 3B to 15B parameters, using a total of several trillion tokens. The ERC loss not only significantly enhances model performance and narrows the performance gap with a competitive yet more computationally expensive MoE variant (Lv et al., 2025) but also retains the efficiency of vanilla MoEs.

Furthermore, building on the first effect, we establish that the ERC loss serves as a powerful tool for studying expert specialization. This property arises from two key features of the ERC loss: (1) the specialization level is explicitly controlled by α, and (2) the bound of noise $\boldsymbol{\delta}_i$ provides a quantitative measure for this level. Through this lens, we reveal a trade-off between specialization and model performance. Our findings challenge some beliefs about expert specialization that were derived from small-scale experiments. These novel quantitative and qualitative analysis methods offer new pathways to advance the understanding of MoE models.

In summary, our contributions are twofold:

(1) We propose the ERC loss, a novel auxiliary loss to effectively and efficiently strengthen expert-router coupling in MoE models.

(2) The ERC loss provides an effective lens for studying expert specialization, offering new insights into MoE models.

2 BACKGROUND

Mixture-of-Experts Our description follows the prevailing SwiGLU structure used by advanced LLMs (Qwen, 2024; DeepSeek-AI, 2025; OpenAI, 2025). An MoE layer consists of n experts, where each expert i is parameterized by three matrices: $\boldsymbol{W}_g^i \in \mathbb{R}^{d\times D}$, $\boldsymbol{W}_p \in \mathbb{R}^{d\times D}$, and $\boldsymbol{W}_o \in \mathbb{R}^{D\times d}$. The layer also includes a router with the weight matrix $\boldsymbol{R} \in \mathbb{R}^{n\times d}$, which takes a token $\boldsymbol{x} \in \mathbb{R}^d$ as input and outputs an expert weight[2] vector:

$$\boldsymbol{w} = \text{softmax}(\boldsymbol{x}\boldsymbol{R}^\top) \in \mathbb{R}^n.$$

Typically, the top-K experts with the highest expert weights are selected to process the token. The processing of $\boldsymbol{x}$ by expert i is given by:

$$E_i(\boldsymbol{x}) = \left(\text{SiLU}(\boldsymbol{x}\boldsymbol{W}_g^i) \odot (\boldsymbol{x}\boldsymbol{W}_p^i)\right) \boldsymbol{W}_o^i,$$

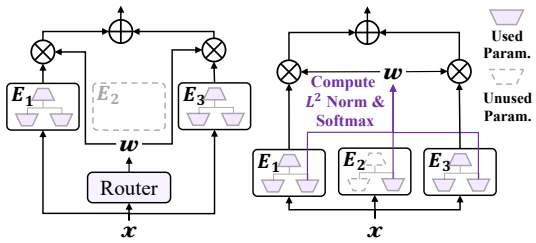
where $\odot$ denotes element-wise multiplication. The final output of the entire MoE layer is the weighted sum of the outputs of the selected experts:

$$\sum_k^K \boldsymbol{w}[k] E_k(\boldsymbol{x}), \text{ where } k \in \text{Top-K}(\boldsymbol{w}).$$

Expert-router coupling via denser activation Autonomy-of-Experts (AoE; Lv et al., 2025) encodes the routing function into expert parameters. AoE factorizes $\boldsymbol{W}_g$ into two r-rank matrices $\boldsymbol{W}_{down}^i \in \mathbb{R}^{d\times r}$ and $\boldsymbol{W}_{up}^i \in \mathbb{R}^{r\times D}$. Each expert processes a token up to the point after the $\boldsymbol{W}_{down}^i$ projection. The expert weight vector is computed using the activation norm at this stage:

$$\boldsymbol{w} = \text{softmax}\left(\{\|\boldsymbol{x}\boldsymbol{W}_{down}^i\| \text{ for } i = 1, \ldots, n\}\right).$$

The top-K experts exhibiting the highest activation norms are selected to continue their forward computation, and the others are terminated early. This norm-based selection is justified by the fact that the activation norm of MLPs represents how well their capabilities match their inputs (Geva et al., 2021; Liu et al., 2023). The computational overhead of AoE scales with the number of tokens during both training and inference. Moreover, this inefficiency worsens as the number of experts n increases or the selection count K decreases. These limitations hinder the scalability and practical deployment of AoE in LLMs.



(a) Mixture-of-Experts (b) Autonomy-of-Experts

Figure 2: The overview of MoE and AoE models.

Pham et al. (2024) use experts' *final* output norms to supervise router logits. There is no inference overhead but the model is fully dense-activated during training, contradicting the core sparsity principle of MoE. Therefore, we include it only for background discussion, not as a baseline.

3 METHOD

After analyzing the strengths and limitations of prior work, we distill three design principles to ensure a lightweight, effective, and practically applicable enhancement for expert-router coupling in MoE-LLMs:

(1) Routers must be retained in MoE architectures to preserve routing efficiency.

(2) An auxiliary loss that enables interaction between experts and routers can strengthen their coupling.

(3) The loss must have complexity independent of the number of input tokens and must not introduce activation density beyond that of a vanilla MoE.

Below, we introduce expert-router coupling loss, which fulfills all these principles.

[2]In this paper, "weight" refers to the relative contribution of each expert's output or the strength of the loss function. Please carefully distinguish between "weight" and "parameter."

3.1 EXPERT-ROUTER COUPLING LOSS

The expert-router coupling (ERC) loss is motivated by a clustering-based interpretation of MoE routing: The routing mechanism in traditional MoE models can be interpreted as a clustering process, where router parameters $\boldsymbol{R} \in \mathbb{R}^{n \times d}$ are viewed as n cluster centers. For any input token $\boldsymbol{x} \in \mathbb{R}^d$, the router computes an n-dimensional logit vector representing the weight assigned to each expert. Specifically, the weight for expert i is derived from the inner product between $\boldsymbol{x}$ and the cluster center $\boldsymbol{R}[i]$. When $\boldsymbol{x}$ belongs to the cluster centered at $\boldsymbol{R}[i]$, this inner product is maximized (under the premise that the rows of $\boldsymbol{R}$ have comparable magnitude, which is generally the case), making expert i the top choice.

A key advantage of this clustering view is that it enables probing an expert's responsiveness to a set of tokens without feeding every token to all experts, unlike prior methods (See §2). Instead, we leverage each cluster center $\boldsymbol{R}[i]$ as a proxy for tokens routed to expert i (denoted as $\mathcal{X}_i$), enabling us to derive intermediate activations and evaluate how well the expert aligns with a proxy token.

Our ERC loss is computed in three key steps:

(1) For each cluster center $\boldsymbol{R}[i]$, we create a perturbed proxy token $\tilde{\boldsymbol{R}}[i] = \boldsymbol{R}[i] \odot \boldsymbol{\delta}_i$. $\boldsymbol{\delta}_i \in \mathbb{R}^d$ is bounded multiplicative random noise, which we elaborate in §3.2. This noise ensures the proxy generalizes to tokens in $\mathcal{X}_i$. **Notably, the corrupted $\tilde{\boldsymbol{R}}$ is used only for loss computation**; routing still uses the clean $\boldsymbol{R}$ to compute router logits, as in standard MoEs.

(2) Each proxy token is processed by the $\boldsymbol{W}_g$ parameter of all n experts, yielding a total of n^2 intermediate activations. The L^2 norm of each activation is computed to form a matrix $\boldsymbol{M} \in \mathbb{R}^{n \times n}$, where $\boldsymbol{M}[i, j]$ corresponds to the norm from expert j given input $\tilde{\boldsymbol{R}}[i]$:

$$\boldsymbol{M}[i, j] = \left\| \tilde{\boldsymbol{R}}[i] \cdot \boldsymbol{W}_g^j \right\|$$

(3) To enforce expert-router coupling, for all i and $j \neq i$, the ERC loss imposes two constraints, where a scalar $\alpha \in [0, 1]$ determines their strength:

$$\boldsymbol{M}[i, j] < \alpha \boldsymbol{M}[i, i] \tag{1}$$

$$\boldsymbol{M}[j, i] < \alpha \boldsymbol{M}[i, i] \tag{2}$$

Constraint 1 ensures the proxy token $\tilde{\boldsymbol{R}}[i]$ activates its corresponding expert i more than any other expert j. Since tokens similar to $\boldsymbol{R}[i]$ are routed to expert i, and given their similarity to $\tilde{\boldsymbol{R}}[i]$, they also elicit a stronger activation in expert i than in other experts. This strongest activation indicates that expert i is optimized to develop capabilities best suited to $\mathcal{X}_i$ (Lv et al., 2025).

Constraint 2 requires that expert i responds more strongly to its own proxy token $\tilde{\boldsymbol{R}}[i]$ than by any other $\tilde{\boldsymbol{R}}[j]$. This ensures each $\boldsymbol{R}[i]$ accurately represents expert i, guaranteeing that tokens most needing expert expert i are correctly routed to it.

As α decreases, the two constraints become stricter, thereby enforcing stronger expert-router coupling. Additionally, α enables flexible regulation of specialization: a smaller α increases the gap between $\boldsymbol{M}[i, i]$ and $\boldsymbol{M}[i, j]$, reflecting greater expert specialization as experts exhibit more differentiated responses to the same inputs. This feature makes the ERC loss a useful tool for investigating expert specialization and provides deeper insight into MoE behavior, as demonstrated in §4.2.

We translate these two constraints into expert-router coupling loss, formally defined as:

$$\mathcal{L}_{\text{ERC}} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j \neq i}^{n} \left(\max\left(\boldsymbol{M}[i, j] - \alpha \boldsymbol{M}[i, i], 0 \right) + \max\left(\boldsymbol{M}[j, i] - \alpha \boldsymbol{M}[i, i], 0 \right) \right) \tag{3}$$

The three steps for computing expert-router coupling loss are illustrated in Figure 1. For implementation details, we provide PyTorch-style pseudocode in Figure 9.

3.2 BOUNDED RANDOM NOISE FOR GENERATING PROXY TOKENS

The perturbed proxy token $\tilde{\boldsymbol{R}}[i] = \boldsymbol{R}[i] \odot \boldsymbol{\delta}_i$ makes expert i's coupling generalizes effectively from $\boldsymbol{R}[i]$ alone to $\mathcal{X}_i$. To ensure the perturbed point $\tilde{\boldsymbol{R}}_i$ remains within its original cluster, we require

a bounded perturbation. We therefore model the noise $\boldsymbol{\delta}_i$ as a multivariate uniform distribution, $\boldsymbol{\delta}_i \sim \mathcal{U}(1-\epsilon_i, 1+\epsilon_i)^d$. Let $j = \arg\min_{j^* \neq i} \|\boldsymbol{R}[i] - \boldsymbol{R}[j^*]\|$ be the nearest cluster center. For the noise level ϵ to be sufficient to avoid perturbing the cluster, it must satisfy:

$$\epsilon_i \leq \frac{\|\boldsymbol{R}[i] - \boldsymbol{R}[j]\|}{2\|\boldsymbol{R}[i]\|}. \tag{4}$$

The derivation of this bound is provided in Appendix A. We set ϵ_i to its maximum value, i.e., the right-hand side of this inequality. Notably, the value of ϵ_i is dynamically computed at each layer and every training step.

3.3 Efficiency analysis

Theoretical training efficiency In a standard MoE layer, T tokens are processed by K experts, resulting in a total computational cost of $6TKDd$ FLOPs. expert-router coupling loss introduces only $2n^2Dd$ additional FLOPs, a cost that is negligible in practical pre-training setups where K is often in the millions. In contrast, AoE introduces an additional overhead of $2T(n-K)dr$ FLOPs (recall that r is AoE's factorization rank; see §2). Given that typical MoE-LLMs operate at sparsity levels far below 25% (i.e., $n > 4K$), this overhead ratio exceeds r/D, making it prohibitive. A detailed breakdown of the FLOP calculations supporting the above theoretical analysis is provided in Appendix B.1.

Empirical training overhead The efficiency of our method is confirmed in practice. The ERC loss maintains low overhead during LLM pre-training with multiple parallelism strategies, adding only 0.2–0.8% overhead in our experiments. We provide a complete analysis of these real-world distributed conditions and measured throughputs in Appendix B.2.

Overhead-free inference Our method incurs no additional inference overhead as the auxiliary loss is not applied. However, AoE retains the same forward computation, carrying over the associated overhead.

4 Experiments

4.1 Experimental settings

We compare the ERC-loss-augmented MoE against both the vanilla MoE and AoE baselines. All models are trained from scratch with 3B parameters. This parameter size is chosen because it represents the largest scale at which we could successfully train the AoE model under our available resources. Our implementation is based on OLMoE (Muennighoff et al., 2025). The models comprise 12 layers with $d = 1536$ and $D = 768$. Each Transformer (Vaswani et al., 2017) layer has 16 attention heads and $n = 64$ experts, where $K = 8$ experts are selected per token. For the AoE model, we set $r = 512$ to ensure consistent total parameter count. The number of activated parameters is 500M. Each model is trained on 500B tokens from the open-source dataset `dolmap-v1.5-sample` (Soldaini et al., 2024), using a batch size of 3 million tokens. We use the AdamW optimizer (Loshchilov & Hutter, 2019) with $(\beta_1, \beta_2) = (0.9, 0.95)$, a weight decay of 0.1, and a learning rate of 4e-4 with a cosine schedule decaying to 4e-5. A load balancing loss (Fedus et al., 2022) with a weight of 0.01 is applied consistently in all experiments.

For simplicity, the loss weight of the ERC loss is fixed at 1, and we use $\alpha = 1$ by default if not specified.

We evaluate LLMs on following tasks: ARC-Challenge (Clark et al., 2018), CommonsenseQA (Talmor et al., 2019), COPA (Roemmele et al., 2011), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), OpenbookQA (Mihaylov et al., 2018), SciQ (Welbl et al., 2017), Social IQa (Sap et al., 2019), WinoGrande (Sakaguchi et al., 2021), and MMLU (Hendrycks et al., 2021a).

4.2 Performance, efficiency, and load balancing

Figure 3(a) reports the average accuracy across all tasks and task-specific results are presented in Figure 10. It shows that the ERC-loss-augmented MoE achieves stable performance gains, which significantly outperforms the vanilla MoE and narrows the gap between AoE and MoE.

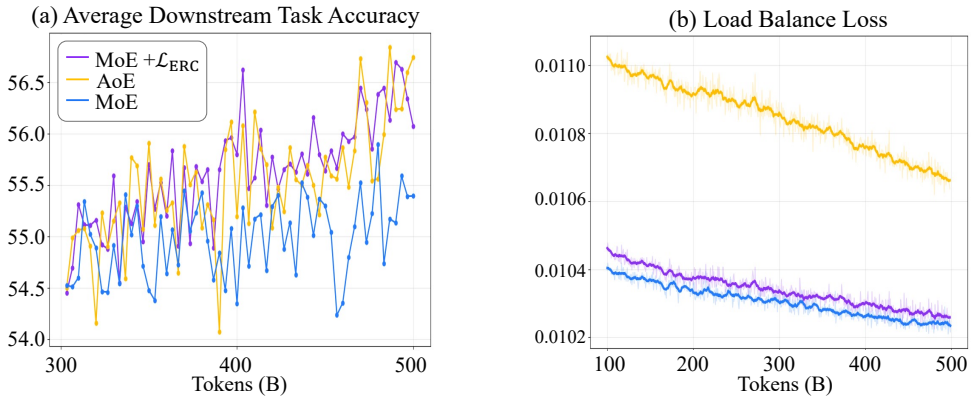


Figure 3: The 3B-parameter MoE model augmented with ERC loss achieves substantial and stable performance gains, while maintaining comparable load balancing to the baseline. For detailed task-specific results, please refer to Figure 10.

Table 1: Scaling to 15B parameters: ERC loss improves performance on more challenging benchmarks.

	MMLU	C-Eval	MMLU-Pro	AGI-Eval	BBH	MATH	GSM8K	TriviaQA
MoE	63.2	67.5	31.0	42.0	44.3	25.7	45.2	47.2
MoE + $\mathcal{L}_{\text{ERC}}$	64.6	69.0	31.9	44.2	45.6	26.1	45.8	49.1

In terms of efficiency, MoE models with and without ERC loss have nearly identical throughput and memory costs. By contrast, AoE requires **1.6**× more training hours and **1.3**× higher memory usage, limiting further scaling due to impractical training times and out-of-memory issues.

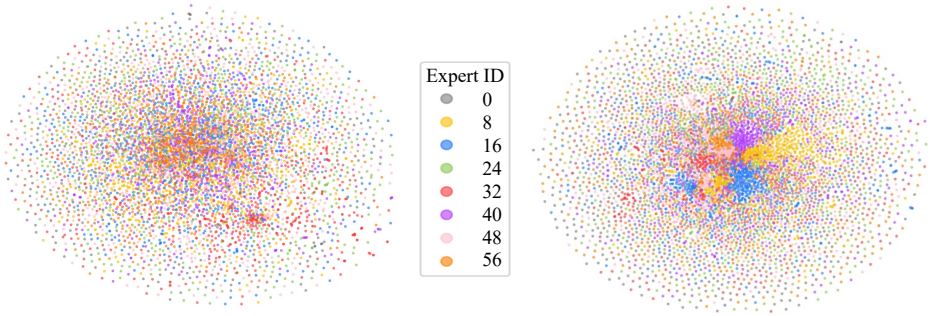
Expert-router coupling loss is compatible with the load balancing loss. As shown in Figure 3(b), the difference in load balancing loss between MoE combined with $\mathcal{L}_{\text{ERC}}$ and the vanilla MoE is on the order of 10^{-5}. This difference is negligible given that the overall load balancing loss magnitude remains around 10^{-2}. By comparison, the loss difference between AoE and vanilla MoE is approximately 4×10^{-4}. Although this difference is still small relative to the overall loss magnitude, it is notably larger than the difference exhibited by ours.

4.3 Validating ERC Loss in 15B-Parameter MoEs

We scale models to 15 billion parameters by increasing n to 256 (keeping K=8) and doubling the model depth. This configuration results in a total of 15B parameters with approximately 700M activated. Other training hyper-parameters largely follow the setup in Section 4.1. As a large-scale, high-sparsity model, the AoE method failed to train due to overly costly and is thus omitted from comparison. Table 1 shows that the benefits of the ERC loss persist across various public benchmarks more challenging than those used for 3B models, including MMLU (Hendrycks et al., 2021a), C-Eval (Huang et al., 2023), MMLU-Pro (Wang et al., 2024b), AGI-Eval (Zhong et al., 2024), BBH (Suzgun et al., 2023), MATH (Hendrycks et al., 2021b), GSM8K (Cobbe et al., 2021), and TriviaQA (Joshi et al., 2017). The consistent performance improvements demonstrate that our method effectively addresses the expert-router decoupling problem even at scale. Throughout this large-scale training, we observed no loss spikes or abnormal gradients.

4.4 The ERC loss is an effective tool for exploring expert specialization

With the ERC loss, experts are more specialized, as they exhibit greater discrimination between tokens they process and those they do not, compared to the ERC loss is not used. An intuitive demonstration of this specialization comes from visualizing expert parameters. Following (Yang et al., 2025), we use t-SNE (van der Maaten & Hinton, 2008) to project each row of $\boldsymbol{W}_g^i$ (where $i \bmod 8 = 0$) from layer 6 (the middle depth) onto a 2D point. As shown in Figure 4, experts in vanilla



(a) MoE (b) MoE + $\mathcal{L}_{\text{ERC}}$

Figure 4: t-SNE projections of $\boldsymbol{W}_g$ in MoE experts trained without and with the ERC loss. Our ERC loss provides greater expert specialization.

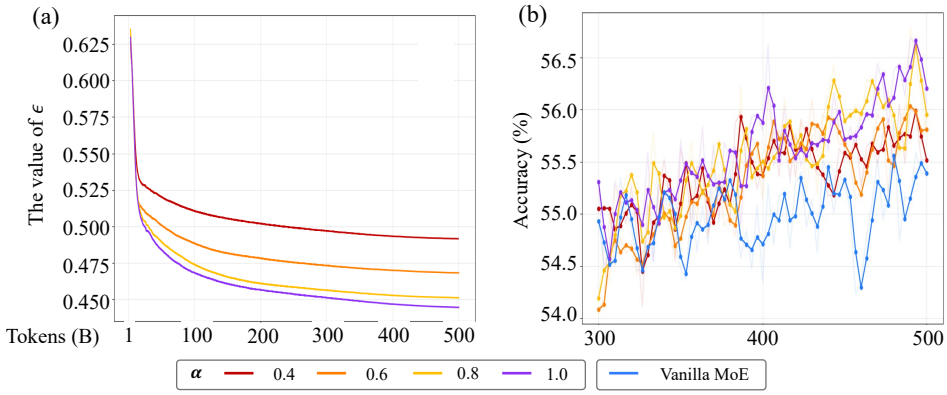


Figure 5: (a) Since routers are deeply coupled with experts, the distance between neighboring cluster centers (i.e., the maximum noise level ϵ) quantitatively reflects changes in expert specialization during training, which is controlled by α. (b) Downstream performance across different values of α.

MoE lack specialization, as their parameter features do not form meaningful clusters. By contrast, MoE enhanced with the ERC loss exhibits distinct clusters, indicating specialized capabilities.

Beyond merely promoting specialization, the ERC loss can also serve as a powerful tool for exploring it. We show this capability through two features below and an example use case in Section 4.5.

Feature 1: α enables a controllable investigation into optimal specialization. In the ERC loss, α governs the coupling strength between experts and the router. When $\alpha = 0$, the ERC loss encourages $\boldsymbol{R}[i]$ to be orthogonal to the parameters of other experts, thereby maximizing specialization. Conversely, when $\alpha \to 1$, the loss permits smaller differences in how all experts' responsiveness to $\boldsymbol{R}[i]$, thus reducing specialization. Notably, $\alpha = 1$ only weakens the ERC loss's constraints to their maximum extent; it still retains a degree of specialization stronger than the spontaneously emerged specialization in a vanilla MoE model.

Feature 2: ϵ provides a quantitative measure for specialization. The noise level ϵ exhibits a strong correlation with α, and it can reflect changes in expert specialization throughout the training process. This correlation exists because as α increases, experts are allowed to be more homogeneous. This growing homogeneity among experts, in turn, reduces the separation between the cluster centers in the router as they are tightly coupled. A smaller separation between cluster centers ultimately derives a smaller ϵ. Thus, ϵ is a quantitative metric tracking expert specialization.

Experiments and discussion. The following experiments support these two features. In Figure 5(a), we plot ϵ at each training step across a parameter search over $\alpha \in \{0.4, 0.6, 0.8, 1.0\}$. Consistent with our analysis, increasing α which reduces expert specialization indeed leads to a corresponding decrease in ϵ. Note that measuring router cluster distance is uninformative in vanilla

MoE training without the ERC loss, as the router and experts are uncoupled and cluster distances do not reflect expert capability dynamics. We further compared downstream task performance across different values of α. Figure 5(b) shows that all tested α values outperform the vanilla MoE model. This not only confirms the robust effectiveness of the ERC loss but also demonstrates that the specialization spontaneously formed by vanilla MoE models is inadequate.

Several previous studies (Guo et al., 2025; Liu et al., 2024; Hendawy et al., 2024) have suggested that enforcing orthogonality among experts can enhance MoE performance. However, these claims are primarily based on small-scale fine-tuning experiments conducted on well-pretrained models. As shown in Figure 5(b), pursuing extreme orthogonality is not advisable, as model performance degrades with stricter ERC loss constraints. This highlights a trade-off between promoting expert specialization and maintaining effective collaboration, a balance that is underdiscussed in previous works. More intuitively, while our ERC loss can achieve expert orthogonality by setting $\alpha = 0$, we observe that this strict constraint can even impair convergence during large-scale pre-training (Figure 6). These findings challenge the applicability and effectiveness of strict expert orthogonality in large-scale pre-training settings, suggesting that the orthogonality obtained during fine-tuning may merely make experts specialized for a specific domain more distinct. In Appendix C.3, we further show that the lack of "perfect" orthogonality among router embeddings is also not a critical weakness for pre-training MoE models.

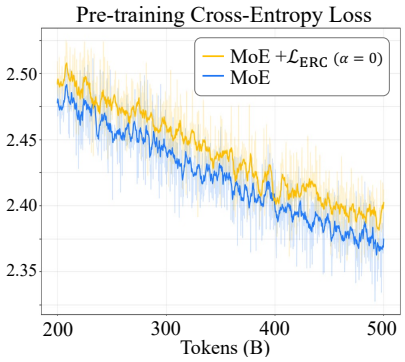


Figure 6: Enforcing expert orthogonality ($\alpha = 0$) impairs convergence.

4.5 How specialized should experts be? An early exploration across sparsity

While $\alpha = 1$ was optimal for the MoE sparsity settings and architectural hyperparameters discussed above, the peak performance at $\alpha = 0.8$ in Figure 5 suggests that other values may yield better results under different model configurations. This raises a question: How specialized should experts be? More concretely, how should α be tuned for different model architectures to achieve better performance?

An intuition is that when the MoE is very sparse (with a small K/n), the selected combination of experts must be generalist enough to cover the diverse requirements of processing any given token. Over-specialization (an α that is too small) risks that this small set of experts cannot adequately handle the input, thereby hurting performance. Conversely, when K/n is large, the system can afford to include more specialized experts, as their collective capacity is more likely to cover the input's needs.

To validate this, we pre-trained models with $n = 64$ experts, varying $K \in \{4, 8, 16\}$ and $\alpha \in \{0.4, 0.6, 0.8, 1.0\}$. For each (K, α) pair, we trained on 100B tokens. All other hyper-parameters followed Section 4.1, and we report the average downstream score across in Figure 7. The results confirm our intuition: for $K = 4$ and 8, $\alpha = 1.0$ performs best; while $\alpha = 0.6$ is acceptable for K=16.

Based on these findings, we provide a practical guideline for tuning α when applying the ERC loss to custom models. Given that industrial MoEs operate with high sparsity (e.g., $K/n \ll 8/64$), we recommend using $\alpha = 1$ as a robust default, requiring no further tuning. For research on smaller models or denser activations, $\alpha = 1.0$ remains a safe and convenient choice, while $\alpha < 1$ may yield more benefits but requires case-specific tuning. Furthermore, this experiment confirms that the ERC loss serves as a tool for studying specialization, thus supporting the claims in Section 4.4.

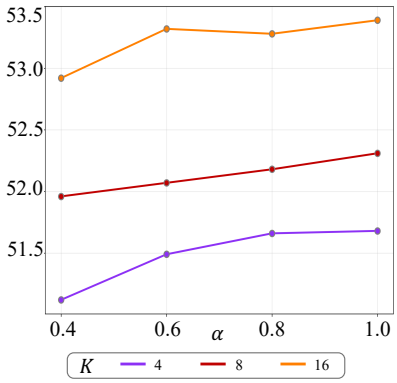


Figure 7: Downstream performance versus MoE sparsity (K/n) and α.

4.6 Ablation studies

Our ablation studies: (1) explain our rationale for selecting $\|\tilde{\boldsymbol{R}}\boldsymbol{W}_g\|$ to compute $\boldsymbol{M}$ as the default choice; (2) demonstrate that the random noise δ enables the generalization of coupling; (3) show that the ERC loss cannot be reduced to geometric constraints applied to experts or routers separately (e.g., router embedding orthogonality); (4) discuss the impact of $\alpha > 1$; and (5) verify that the model decreases the ERC loss by learning meaningful coupling rather than by manipulating parameter norms. Due to page limits, we include these experiments in Appendix C.

5 Related works

Auxiliary loss for MoEs Auxiliary losses are crucial for training large-scale MoE models. Most existing work in this area focuses primarily on enhancing training stability. For instance, many studies have proposed auxiliary losses to address load balancing challenges (Fedus et al., 2022; Qiu et al., 2025; Wang et al., 2024a); Zoph et al. (2022) introduced the z-loss, which penalizes excessively large logits in the gating network to enable stable training. Our ERC loss is the first tailored to strengthen the expert-router coupling. Other related auxiliary losses enhancing expert specialization or orthogonality are discussed below.

Expert specialization Dai et al. (2024) introduced a shared expert to handle general capabilities, encouraging the others to be more specialized. Guo et al. (2025) proposes an auxiliary loss to minimize the pairwise projections of the selected top-K experts' outputs for each token, reducing expert overlap but incurring high cost due to K^2 cosine similarity calculations per token. Other methods scale the number of tiny experts to millions, making each expert more atomic and thus more specialized (Yang et al., 2025; Park et al., 2025; He, 2024), but are memory-bounded. Beyond efficiency, these methods face three major limitations: (1) no quantitative control over specialization degree; (2) no exploration of the specialized-generalized ability trade-off; and (3) failure to strengthen expert-router coupling. Our method addresses all three, both efficiently and effectively.

Some works (Guo et al., 2025; Liu et al., 2024; Hendawy et al., 2024) maximize specialization by training orthogonal experts, but their evaluations are limited to small-scale fine-tuning (or reinforcement learning). In contrast, our ERC loss allows for orthogonality when $\alpha = 0$, yet we find this value hinders convergence during pre-training, with optimal performance achieved at $\alpha \gg 0$. These results challenge the practicality of expert orthogonality in large-scale pre-training.

Contrastive learning Constraints 1 and 2 bear similarity to contrastive learning (Chen et al., 2020; van den Oord et al., 2019; Khosla et al., 2020). In MoE research, Luo et al. (2024) applied contrastive learning to expert outputs, encouraging specialization. Baidu-ERNIE-Team (2025) enforces router embedding orthogonality. However, naively applying contrastive learning to either routers or experts leaves the weak expert-router coupling unaddressed.

6 Conclusions

The weak coupling between router decisions and expert capabilities limits MoE models in multiple important aspects. We propose expert-router coupling loss that tightly couples router parameters with their corresponding experts. The proposed ERC loss improves MoE-based LLMs on downstream tasks while incurring negligible training overhead. In addition, it exhibits several desirable properties that not only provide deeper insight into the behavior of MoE models but also offer a promising tool for future research on expert specialization.

Statements on ethics, reproducibility, and LLM usage

Our research does not raise ethical issues. For reproducibility, we used public data and code, and provide algorithm code in Figure 9. We used LLMs solely for typo checking.

REFERENCES

Baidu-ERNIE-Team. Ernie 4.5 technical report, 2025.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020. URL https://arxiv.org/abs/2002.05709.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *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. 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300/.

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, 2018. URL https://arxiv.org/abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models, 2024. URL https://arxiv.org/abs/2401.06066.

DeepSeek-AI. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

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. URL http://jmlr.org/papers/v23/21-0998.html.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446.

Hongcan Guo, Haolang Lu, Guoshun Nan, Bolun Chu, Jialin Zhuang, Yuan Yang, Wenhao Che, Sicong Leng, Qimei Cui, and Xudong Jiang. Advancing expert specialization for better moe, 2025. URL https://arxiv.org/abs/2505.22323.

Xu Owen He. Mixture of a million experts, 2024. URL https://arxiv.org/abs/2407.04153.

Ahmed Hendawy, Jan Peters, and Carlo D'Eramo. Multi-task reinforcement learning with mixture of orthogonal experts. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=aZH1dM3GOX.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021a.

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. *NeurIPS*, 2021b.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, 2023.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 18661–18673. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qrwe7XHTmYb.

Boan Liu, Liang Ding, Li Shen, Keqin Peng, Yu Cao, Dazhao Cheng, and Dacheng Tao. Diversifying the mixture-of-experts representation for language models with orthogonal optimizer, 2024. URL https://arxiv.org/abs/2310.09762.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Ré, and Beidi Chen. Deja vu: contextual sparsity for efficient llms at inference time. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Tongxu Luo, Jiahe Lei, Fangyu Lei, Weihao Liu, Shizhu He, Jun Zhao, and Kang Liu. Moelora: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models, 2024. URL https://arxiv.org/abs/2402.12851.

Ang Lv, Ruobing Xie, Yining Qian, Songhao Wu, Xingwu Sun, Zhanhui Kang, Di Wang, and Rui Yan. Autonomy-of-experts models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=8BIDrYWCeg.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering, 2018. URL https://arxiv.org/abs/1809.02789.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. Olmoe: Open mixture-of-experts language models, 2025. URL https://arxiv.org/abs/2409.02060.

OpenAI. Gpt-oss series, 8 2025. URL https://openai.com/index/introducing-gpt-oss/.

Jungwoo Park, Ahn Young Jin, Kee-Eung Kim, and Jaewoo Kang. Monet: Mixture of monosemantic experts for transformers. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=1Ogw1SHY3p.

Quang Pham, Giang Do, Huy Nguyen, TrungTin Nguyen, Chenghao Liu, Mina Sartipi, Binh T. Nguyen, Savitha Ramasamy, Xiaoli Li, Steven Hoi, and Nhat Ho. Competesmoe – effective training of sparse mixture of experts via competition, 2024.

Zihan Qiu, Zeyu Huang, Bo Zheng, Kaiyue Wen, Zekun Wang, Rui Men, Ivan Titov, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Demons in the detail: On implementing load balancing loss for training specialized mixture-of-expert models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5005–5018, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.249. URL https://aclanthology.org/2025.acl-long.249/.

Qwen. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*, 2011.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454.

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. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. *arXiv preprint*, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.824. URL https://aclanthology.org/2023.findings-acl.824/.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *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. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019. URL https://arxiv.org/abs/1807.03748.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts, 2024a. URL https://arxiv.org/abs/2408.15664.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024b.

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

Xingyi Yang, Constantin Venhoff, Ashkan Khakzar, Christian Schroeder de Witt, Puneet K. Dokania, Adel Bibi, and Philip Torr. Mixture of experts made intrinsically interpretable. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=6QERrXMLP2.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. AGIEval: A human-centric benchmark for evaluating foundation models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 2299–2314, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.149. URL https://aclanthology.org/2024.findings-naacl.149/.

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, 2022. URL https://arxiv.org/abs/2202.08906.

A Determining the maximum multiplicative noise level

$\boldsymbol{\delta}_i$ is a random vector where each component $\boldsymbol{\delta}_{i,k}$ follows a uniform distribution $\mathcal{U}(1-\epsilon, 1+\epsilon)$, and all components are mutually independent. The perturbed point is given by:

$$\tilde{\boldsymbol{R}}_i = (\boldsymbol{\delta}_{i,1}(\boldsymbol{R}_{i,1}), \boldsymbol{\delta}_{i,2}(\boldsymbol{R}_{i,2}), \ldots, \boldsymbol{\delta}_{i,d}(\boldsymbol{R}_{i,d}))$$

To ensure that $\tilde{\boldsymbol{R}}_i$ remains in the same cluster as $\boldsymbol{R}_i$, it must satisfy:

$$\|\tilde{\boldsymbol{R}}_i - \boldsymbol{R}_i\|^2 < \|\tilde{\boldsymbol{R}}_i - \boldsymbol{R}_j\|^2,$$

where $j = \arg\min_{j^* \neq i} \|\boldsymbol{R}[i] - \boldsymbol{R}[j]\|$.

Expanding the squared norms on both sides of the inequality yields:

$$\|\tilde{\boldsymbol{R}}_i - \boldsymbol{R}_i\|^2 = \sum_{k=1}^{d} (\boldsymbol{\delta}_{i,k} - 1)^2 (\boldsymbol{R}_{i,k})^2$$

$$\|\tilde{\boldsymbol{R}}_i - \boldsymbol{R}_j\|^2 = \sum_{k=1}^{d} (\boldsymbol{\delta}_{i,k}\boldsymbol{R}_{i,k} - \boldsymbol{R}_{j,k})^2$$

Substituting into the inequality and simplifying gives:

$$\sum_{k=1}^{d} [2\boldsymbol{\delta}_{i,k}(\boldsymbol{R}_{i,k}(\boldsymbol{R}_{j,k} - \boldsymbol{R}_{i,k}) + (\boldsymbol{R}_{i,k}^2 - \boldsymbol{R}_{j,k}^2)] < 0$$

To ensure this inequality holds for all realizations of $\boldsymbol{\delta}_i$, we consider the worst-case scenario that maximizes the left-hand side. Define:

$$A_k = 2\boldsymbol{R}_{i,k}(\boldsymbol{R}_{j,k} - \boldsymbol{R}_{i,k}), \quad B = \sum_{k=1}^{d} (\boldsymbol{R}_{i,k}^2 - \boldsymbol{R}_{j,k}^2),$$

so the inequality becomes:

$$\sum_{k=1}^{d} A_k \boldsymbol{\delta}_{i,k} + B < 0. \tag{5}$$

The worst-case $\boldsymbol{\delta}_{i,k}$ is chosen to maximize $\sum A_k \boldsymbol{\delta}_{i,k}$:

$$\boldsymbol{\delta}_{i,k} = \begin{cases} 1+\epsilon & \text{if } A_k > 0, \\ 1-\epsilon & \text{if } A_k < 0. \end{cases}$$

Substituting these values gives:

$$\sum_{k=1}^{d} A_k + \epsilon \sum_{k=1}^{d} |A_k| + B < 0. \tag{6}$$

Now simplify $\sum A_k + B$:

$$\begin{aligned} \sum A_k + B &= 2\sum \boldsymbol{R}_{i,k}\boldsymbol{R}_{j,k} - 2\sum \boldsymbol{R}_{i,k}^2 + \sum \boldsymbol{R}_{i,k}^2 - \sum \boldsymbol{R}_{j,k}^2 \\ &= 2\sum \boldsymbol{R}_{i,k} R_{j,k} - \sum \boldsymbol{R}_{i,k}^2 - \sum \boldsymbol{R}_{j,k}^2 \\ &= -\left(\sum \boldsymbol{R}_{i,k}^2 - 2\sum \boldsymbol{R}_{i,k}\boldsymbol{R}_{j,k} + \sum \boldsymbol{R}_{j,k}^2\right) \\ &= -\|\boldsymbol{R}_i - \boldsymbol{R}_j\|^2 \end{aligned} \tag{7}$$

Substituting equation 7 into equation 6 yields:

$$-\|\boldsymbol{R}_i - \boldsymbol{R}_j\|^2 + 2\epsilon \sum_{k=1}^{d} |\boldsymbol{R}_{i,k}(\boldsymbol{R}_{j,k} - \boldsymbol{R}_{i,k})| < 0$$

Solving for ϵ gives:

$$\epsilon_{\max} < \frac{\|\boldsymbol{R}_j - \boldsymbol{R}_i\|^2}{2\sum_{k=1}^{d} \|\boldsymbol{R}_{i,k}(\boldsymbol{R}_{j,k} - \boldsymbol{R}_{i,k})\|}$$

However, computing the denominator of this expression is relatively complex. To balance the efficiency of loss calculation, we instead adopt a tighter upper bound for ϵ.

By the Cauchy-Schwarz Inequality, the following relationship holds:

$$\sum_{k=1}^{d} \|\boldsymbol{R}_{i,k}(\boldsymbol{R}_{j,k} - \boldsymbol{R}_{i,k})\| \leq \|\boldsymbol{R}_i\| \cdot \|\boldsymbol{R}_j - \boldsymbol{R}_i\|$$

Thus, we have:

$$\epsilon_{\max} = \frac{\|\boldsymbol{R}_j - \boldsymbol{R}_i\|^2}{2\sum_{k=1}^{d} |(\boldsymbol{R}_{i,k}(\boldsymbol{R}_{j,k} - \boldsymbol{R}_{i,k})\|} \geq \frac{\|\boldsymbol{R}_j - \boldsymbol{R}_i\|^2}{2\|\boldsymbol{R}_i\| \cdot \|\boldsymbol{R}_j - \boldsymbol{R}_i\|} = \frac{\|\boldsymbol{R}_j - \boldsymbol{R}_i\|}{2\|\boldsymbol{R}_i\|}$$

The term on the right-hand side of the final inequality is the value of ϵ we used in the main text. This choice ensures that the perturbed $\tilde{\boldsymbol{R}}[i]$ remains closer to $\boldsymbol{R}[i]$ than to any other $\boldsymbol{R}[j \neq i]$ at all times.

B Efficiency Analysis

Appendix B.1 analyzes the ideal FLOPs cost breakdown of the vanilla MoE, as well as the overhead introduced by AoE and ERC loss. Appendix B.2 discusses efficiency with consideration of the multiple parallelism strategies used in real-world MoE pre-training. Both analyses demonstrate the practicality of our method.

B.1 FLOPs cost breakdown of three methods

MoE forward Each expert in a MoE layer involves the following operations, with their respective FLOP counts:

- Two matrix multiplications of dimension $T \times d$ with $d \times D$, accounting for $4TdD$ FLOPs. These correspond to the linear transformations parameterized by $\boldsymbol{W}_g$ and $\boldsymbol{W}_p$.
- One element-wise multiplication of $T \times D$ tensors and one SiLU activation applied to a $T \times D$ tensor. The computational cost of these operations is negligible compared to the matrix multiplications.
- One matrix multiplication of dimension $T \times D$ with $D \times d$, contributing $2TDd$ FLOPs. This corresponds to the linear transformation parameterized by $\boldsymbol{W}_o$.

Summing these components gives a total of $6TdD$ FLOPs per expert. For K experts processing T tokens, the total computational cost is therefore $6KTdD$ FLOPs.

Computational overhead of AoE AoE factorizes the expert matrix $\boldsymbol{W}_g \in \mathbb{R}^{D\times d}$ into two low-rank matrices of rank r. To maintain the same number of parameters as the original matrix, we require $dr + Dr = Dd$, which gives $r = \frac{Dd}{d+D}$.

The change in FLOPs compared to an MoE is:

$$T\left(\underbrace{2ndr}_{\text{All experts use } \boldsymbol{W}_{\text{down}}} + \underbrace{2KDr}_{\text{Top-}K\text{ experts use } \boldsymbol{W}_{\text{up}}} - \underbrace{2KDd}_{\text{Top-}K\text{ experts use original } \boldsymbol{W}_g}\right),$$

where T is the number of tokens. Substituting the value of r and simplifying leads to an extra computational cost of:

$$2T(n-K)dr.$$

Computational overhead of expert-router coupling loss It introduces n^2 matrix multiplications, each operating on tensors of shape $1 \times d$ and $d \times D$. In total, this results in $2n^2Dd$ extra FLOPs.

B.2 THROUGHPUTS UNDER MULTIPLE PARALLELISM STRATEGIES

We now assess the overhead of the ERC loss in a realistic large-scale pre-training setup that employs both data parallelism (DP) and expert parallelism (EP). As derived in our previous analysis, the computational cost of the ERC loss is equivalent to a forward pass on $n^2/3$ tokens. When distributed across devices, the costs are:

- Base MoE forward: $K \cdot T$ / `dp_size`
- ERC overhead: $n \cdot (n$ / `ep_size`$) / 3$

Consider training our 15B-parameter model with the configuration: $K = 8$, $T = 3 \times 10^6$, $n = 256$, `dp_size` = 64, and `ep_size` = 8. In this scenario, the ERC overhead constitutes a mere 0.72% of the base model's forward pass cost. This theoretical estimate is consistent with our empirical measurements: we observe a throughput of 62.03B tokens/day for the baseline versus 61.52B tokens/day for our model, representing only a 0.82% reduction. With a smaller $n = 64$, as in our 3B models trained with `dp_size`=32 and `ep_size`=1 (i.e., EP disabled), the overhead ratio drops further to 0.18%. This analysis confirms the practical efficiency of our method.

C ABLATION STUDIES

C.1 COMPUTING $\boldsymbol{M}$ WITH DIFFERENT ACTIVATIONS

We considered five candidates for calculating $\boldsymbol{M}$: using the norms of (a) $\tilde{\boldsymbol{R}}\boldsymbol{W}_g$, (b) $\tilde{\boldsymbol{R}}\boldsymbol{W}_p$, (c) SiLU$(\tilde{\boldsymbol{R}}\boldsymbol{W}_g)$, (d) the post-SwiGLU activations (i.e., SiLU$(\tilde{\boldsymbol{R}}\boldsymbol{W}_g) \odot \tilde{\boldsymbol{R}}\boldsymbol{W}_p$), and (e) experts' final outputs (i.e., $(\text{SiLU}(\tilde{\boldsymbol{R}}\boldsymbol{W}_g) \odot \tilde{\boldsymbol{R}}\boldsymbol{W}_p)\boldsymbol{W}_o$). As shown in Figure 8 C.1, $\tilde{\boldsymbol{R}}\boldsymbol{W}_g$ is the most effective among all alternatives. While using the final output achieves comparable performance, it incurs a higher cost. We therefore adopt $\tilde{\boldsymbol{R}}\boldsymbol{W}_g$ as our default choice.

C.2 RANDOM NOISE $\boldsymbol{\delta}$ ENABLES THE GENERALIZATION OF COUPLING

The random noise $\boldsymbol{\delta}$ allows $\tilde{\boldsymbol{R}}[i]$ to better capture the samples within $\mathcal{X}_i$. To validate its importance, we conducted an ablation study where we trained an MoE with the ERC loss but removed $\boldsymbol{\delta}$. Specifically, we computed $\boldsymbol{M}$ directly using the original $\boldsymbol{R}$ instead of the noise-augmented $\tilde{\boldsymbol{R}}$. As shown in Figure 8 C.2, removing $\boldsymbol{\delta}$ greatly degrades performance. This is because the coupling between routers and experts becomes overfitted to $\boldsymbol{R}$, failing to generalize to the real inputs that $\boldsymbol{R}[i]$s represent.

C.3 COMPARISON WITH CONTRASTIVE REGULARIZATION SOLELY ON ROUTERS

In Section 4.4, we showed that overly strict contrastive regularization on experts can be detrimental during pre-training. Here, we extend this analysis to contrastive regularization applied solely to routers. We compare our ERC loss with the router orthogonalization loss (Baidu-ERNIE-Team, 2025), which requires $\hat{\boldsymbol{R}}$ (the row-wise normalization of $\boldsymbol{R}$) to satisfy:

$$\hat{\boldsymbol{R}}\hat{\boldsymbol{R}}^\top = \boldsymbol{I}.$$

As shown in Figure 8 C.3, the orthogonalization loss yields only limited gains. We attribute this to our finding that the router embeddings in our baseline MoE model are already nearly orthogonal, with an average absolute cosine similarity of 0.15. This value corresponds to angles between router embeddings mostly ranging from arccos(0.15) = 81° to arccos(-0.15) = 99°. Notably, we do not imply that all MoEs always have nearly orthogonal router embeddings, as this may depend on the data or specific architecture; we report this only as a characteristic of our models, which explains the limited gains from the orthogonalization loss.

This result further demonstrates that weak coupling between routers and experts is a more critical issue than imperfect orthogonality in router embeddings. The significant gains from ERC, even when applied to a baseline with already near-orthogonal routers, provide clear evidence.

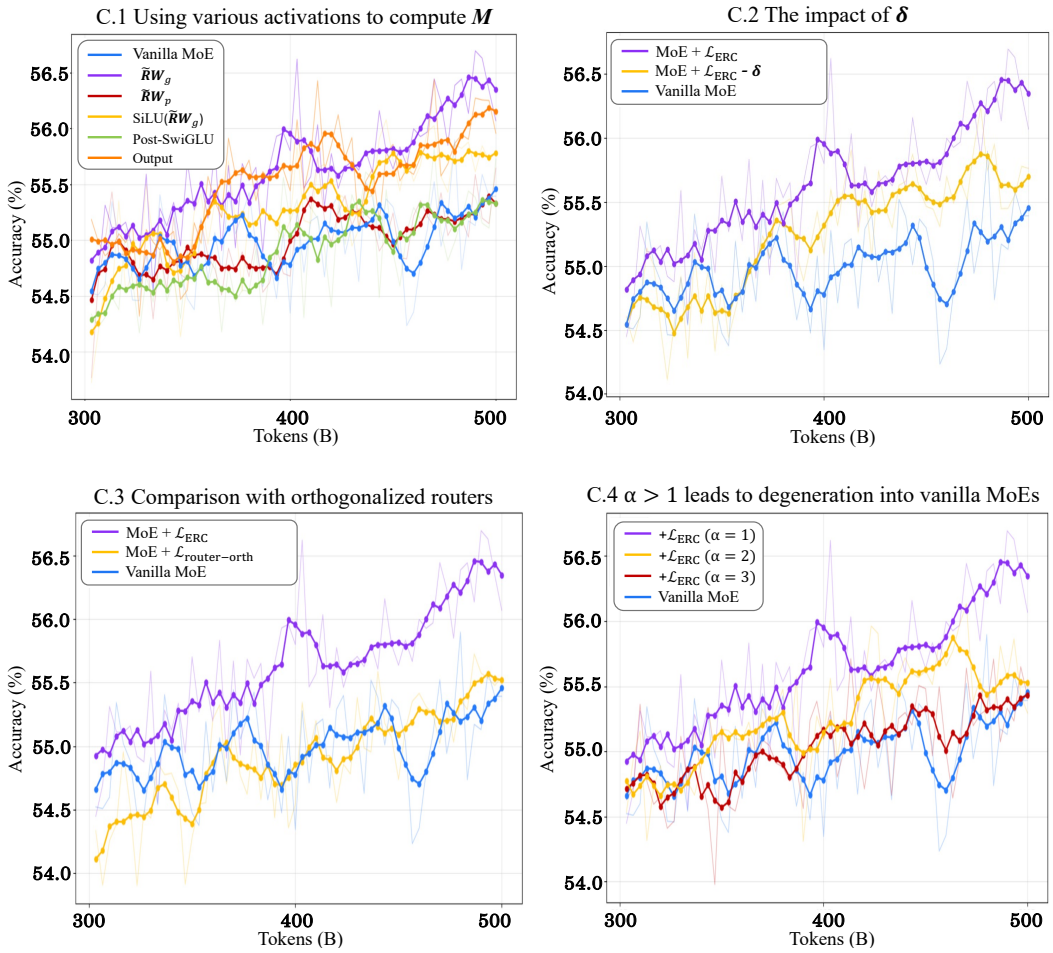


Figure 8: Results of ablation studies C.1, C.2, C.3 and C.4. For detailed task-specific results, please refer to Figure 10.

Furthermore, it is important to note that even if both routers and experts are orthogonalized, there is no guarantee that each $\boldsymbol{R}[i]$ will be aligned with $\boldsymbol{W}_g^i$. Therefore, the ERC loss cannot be reduced to contrastive techniques applied individually to routers or experts, such as orthogonalization loss.

C.4 What happens if $\alpha > 1$?

Some readers might be interested in the value of α at which the ERC loss degenerates to no effective constraints, and the trained model consequently degenerates to a vanilla MoE. For our baseline MoE, we seek the minimum α that zeros the ERC loss computed from the $\mathbf{M}$ matrices of the last checkpoint. Table 2 shows that achieving zero ERC loss across all layers requires $\alpha = 5$ in our pre-trained vanilla MoE baseline. This provides direct evidence that the router-expert coupling in the vanilla MoE is very weak.

We further pre-trained 3B MoE models with the ERC loss at α values of 2 and 3. It is important to note that this experiment is to only demonstrate the effects of loosening the ERC constraints. We do not recommend using $\alpha > 1$ in practice, as it contradicts our core motivation: the router and experts will shift from a state of no mismatch toward looser coupling constraints, ultimately causing the model to degenerate into a vanilla MoE. As shown in Figure 8 C.4, the model with $\alpha = 2$ yields only limited improvement, while the model with $\alpha = 3$ shows almost no improvement over the vanilla MoE.

Table 2: Post-hoc ERC loss evaluation of the vanilla MoE across α values. For a clear and concise demonstration, the loss values in this table are computed using the original $\boldsymbol{R}$ rather than $\tilde{\boldsymbol{R}}$, making the results deterministic.

Layer	Value of α				
	1	2	3	4	5
0	0.87	0.69	0.26	0.00	0.00
1	0.42	0.28	0.10	0.00	0.00
2	0.45	0.19	0.00	0.00	0.00
3	0.25	0.15	0.00	0.00	0.00
4	0.28	0.08	0.00	0.00	0.00
5	0.24	0.22	0.00	0.00	0.00
6	0.22	0.15	0.00	0.00	0.00
7	0.21	0.13	0.00	0.00	0.00
8	0.15	0.05	0.00	0.00	0.00
9	0.16	0.00	0.00	0.00	0.00
10	0.21	0.09	0.00	0.00	00.00
11	0.50	0.44	0.20	0.20	0.00

C.5 Do models reduce ERC loss through manipulating parameter norms?

This is a frequent question, as some readers assume that simply increasing or decreasing the norms of certain router embeddings or experts will increase the diagonal entries of $\boldsymbol{M}$, thereby reducing the ERC loss. Below, we (1) explain that any attempt to reduce one term of the ERC loss by manipulating norms will simultaneously increase other terms, and (2) present detailed parameter norms as evidence.

The term $\boldsymbol{M}[i,j]$ can be expressed as $\|\boldsymbol{R}[i]\|\|\boldsymbol{W}_g^j\|\cos\theta_{i,j}$, where $\theta_{i,j}$ denotes the angle between $\boldsymbol{R}[i]$ and $\boldsymbol{W}_g^j$. Scaling up $\|\boldsymbol{W}_g^i\|$ decreases the loss from i-th row in $\boldsymbol{M}$ (as the second term below increases):

$$\|\boldsymbol{R}[i]\|\|\boldsymbol{W}_g^j\|\cos\theta_{i,j} - \|\boldsymbol{R}[i]\|\|\boldsymbol{W}_g^i\|\cos\theta_{i,i}$$

However, simultaneously, it increases the loss term from every $j \neq i$ rows (as the first term below increases):

$$\|\boldsymbol{R}[j]\|\|\boldsymbol{W}_g^i\|\cos\theta_{j,i} - \|\boldsymbol{R}[j]\|\|\boldsymbol{W}_g^j\|\cos\theta_{j,j}.$$

This logic is symmetric: any attempt to manipulate the norms of $\boldsymbol{R}$ or $\boldsymbol{W}_g$ (whether increasing or decreasing them) to reduce one part of the loss will increase another. This property ensures that the overall ERC loss is minimized only when the router embedding norms are kept comparable and a meaningful coupling is established between routers and their experts.

As shown in the first four columns of Table 3, the average parameter norms for models trained with and without the ERC loss are comparable. Meanwhile, the lower standard deviation under the ERC loss reflects more consistent norms across both router embeddings and experts. In the last two columns of the table, we present the ERC loss for each model. The ERC loss is significantly higher in the baseline model despite its similar average parameter norms.

Table 3: The first four columns show parameter norms for models trained with and without ERC loss, while the last two show the corresponding layer-wise ERC loss. These results show that MoE + $\mathcal{L}_{\text{ERC}}$ learns a meaningful coupling, rather than trivially minimizing the loss through norm manipulation. All values are evaluated on the last checkpoint.

Layer	$\|\boldsymbol{R}[i]\|$		$\|\boldsymbol{W}_g^i\|$		$\mathcal{L}_{\text{ERC}}$ Values	
	Baseline	+$\mathcal{L}_{\text{ERC}}$	Baseline	+$\mathcal{L}_{\text{ERC}}$	Baseline	+$\mathcal{L}_{\text{ERC}}$
0	1.85±0.39	1.67±0.31	25.46±3.93	24.14±3.02	0.87	0.00
1	1.25±0.13	1.13±0.12	30.14±0.68	29.42±0.69	0.42	0.00
2	1.17±0.12	1.07±0.09	30.63±0.77	29.88±0.76	0.45	0.00
3	1.10±0.08	1.01±0.07	30.18±0.77	29.42±0.78	0.25	0.00
4	1.03±0.08	0.89±0.05	30.59±1.21	29.88±1.09	0.28	0.00
5	0.93±0.08	0.87±0.06	30.33±1.13	29.86±1.06	0.24	0.00
6	0.86±0.08	0.83±0.07	30.65±1.15	29.82±1.11	0.22	0.00
7	0.82±0.07	0.75±0.06	30.56±1.20	29.96±1.16	0.21	0.00
8	0.77±0.06	0.76±0.06	30.46±1.02	29.82±0.88	0.15	0.00
9	0.80±0.07	0.74±0.06	30.58±0.88	29.86±0.79	0.16	0.00
10	0.74±0.08	0.69±0.06	30.80±1.03	30.16±0.89	0.21	0.00
11	0.80±0.14	0.73±0.10	32.03±1.46	31.50±1.26	0.50	0.00

```python
import torch
import torch.nn as nn
import PseudoExpertClass

class MoE(nn.Module):

    def __init__(self, args):
        super().__init__()

        self.experts = PseudoExpertClass(args)
        self.R = torch.nn.Parameter(torch.empty(
            args.n, args.d))

        self.alpha = args.alpha

    def erc_loss(self, M):
        row_diff = (M - self.alpha * torch.diag(M).unsqueeze(1))
        row_diff_clamped = torch.clamp(row_diff, min=0.0)

        col_diff = (M - self.alpha * torch.diag(M).unsqueeze(0))
        col_diff_clamped = torch.clamp(col_diff, min=0.0)

        mask = torch.ones_like(A) - torch.eye(A.size(0), device=A.device)
        total_diff = (row_diff_clamped + col_diff_clamped) * mask

        return total_diff.mean()

    def get_noisy_router(self, R):
        with torch.no_grad():
            norm_R = torch.norm(R, dim=1)
            distances = torch.cdist(R, R, p=2)
            distances.fill_diagonal_(float('inf'))
            min_dist, _ = torch.min(distances, dim=1)
            eps = min_dist / 2 / norm_R

            low = (1 - eps).unsqueeze(1)
            high = (1 + eps).unsqueeze(1)
            noise = torch.rand_like(R)
        return (low + noise * (high - low)) * R

    def forward(self, x):

        erc_loss = 0.0
        if self.training:
            R = self.get_noisy_router(self.R)
            M = torch.norm(torch.einsum('jDd,id->ijD', self.experts.Wg,
                R), dim=-1)
            erc_loss = self.erc_loss(M)

        logits = x.view(-1, x.shape[-1]) @ self.R.T
        scores = logits.softmax(dim=-1)
        expert_weights, expert_indices = torch.topk(scores, dim=-1)

        return self.experts(x, expert_weights, expert_indices), erc_loss
```

Figure 9: Pseudo code for expert-router coupling loss in PyTorch.

Figure 3

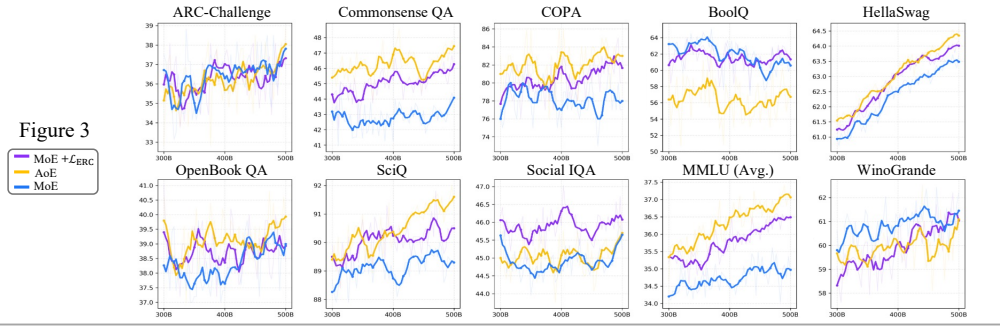


Figure 8 C.1

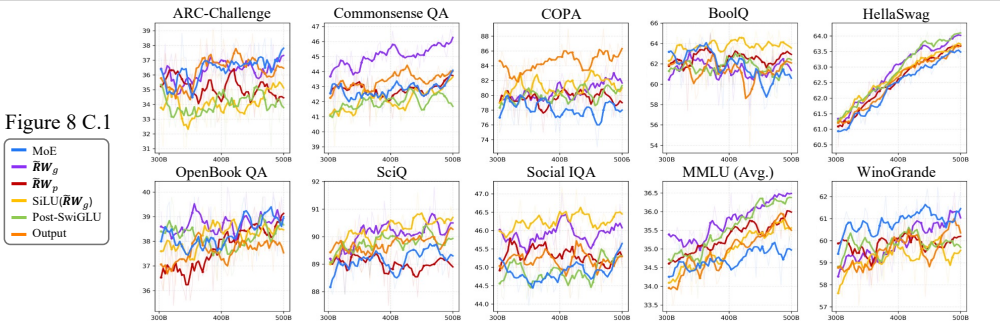


Figure 8 C.2

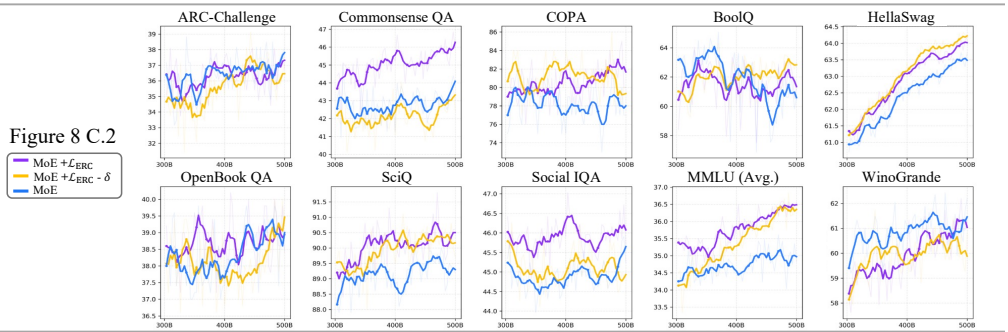


Figure 8 C.3

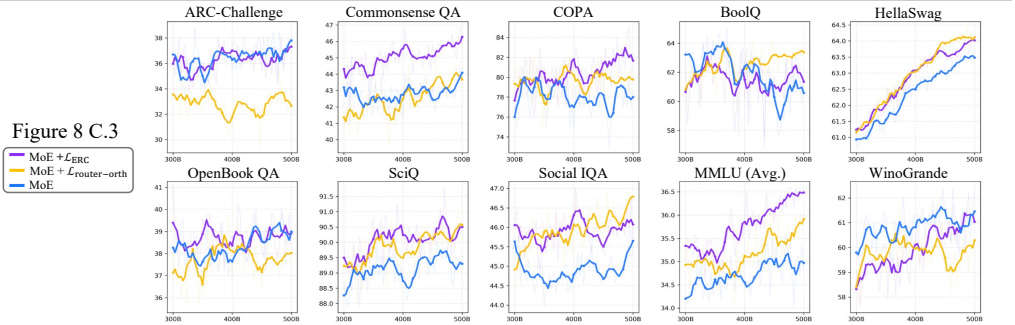


Figure 8 C.4

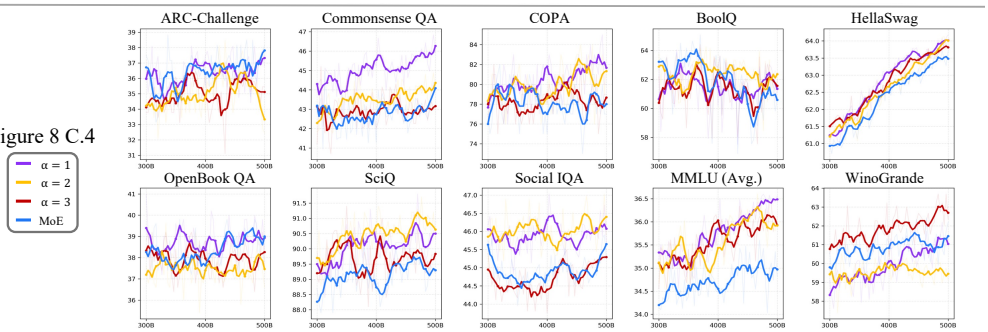


Figure 10: Task-specific downstream results for previous experiments.