Distribution-Consistent Inference for Dynamic Sparse Mixture-of-Experts

Anonymous ACL submission

Abstract

Mixture-of-Experts (MoE) architectures have emerged as a powerful paradigm for scaling model capacity while preserving efficient inference in large foundation models. However, most MoE models use a fixed top-k expert selection policy, assigning the same expert budget to every token even when fewer experts may be sufficient. Inference-time dynamic top-k routing can reduce computation without retraining, but existing methods often overlook the distributional shift caused by deviating from the training-time routing configuration. We show that reducing the number of activated experts consistently increases the RMS scale and variance of SMoE outputs, inducing a representation mismatch that contributes to downstream performance degradation. To address this, we propose *Layer-wise Distribution Alignment* (LDA), a lightweight inference-time correction that uses layer-wise calibration statistics to align reduced-routing representations with the default configuration. Across multiple SMoE LLMs, benchmarks, and routing strategies, LDA recovers much of the lost performance while preserving sparse-inference efficiency with negligible overhead.

1 Introduction

The rapid progress of large language models (LLMs) has been driven in large part by scaling model parameters (Wan et al., 2024), but this trend has also increased computational and memory costs during both training and inference. Sparse Mixture-of-Experts (SMoE) architectures mitigate this bottleneck by decoupling total model capacity from per-token computation (Shazeer et al., 2017; Cai et al., 2025), routing each token to only a small subset of experts to expand capacity while maintaining a manageable inference budget. In practice, most SMoE models employ a fixed top-k routing strategy (Fedus et al., 2022), where a router scores

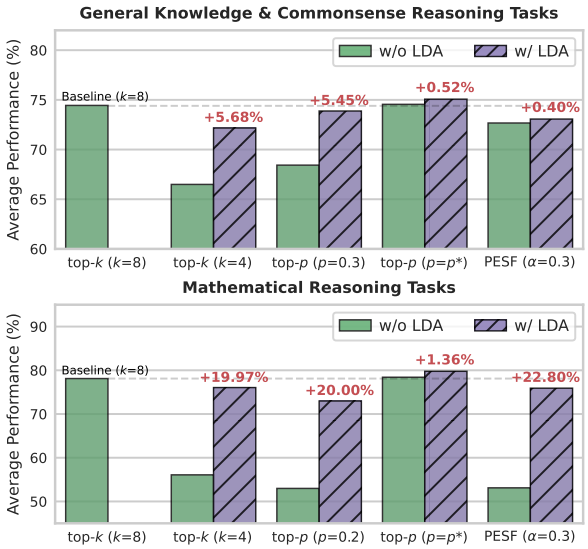


Figure 1: **Comparison of average performance with and without LDA across different routing strategies on Qwen3-30B-A3B.** The dashed line indicates the baseline performance with the top-k routing strategy (k=8). p* denotes the best-performing top-p threshold for each task, reported in Table 9.

all experts and selects the k highest-scoring experts for each token. While simple and predictable, this scheme assigns every token the same expert budget regardless of contextual complexity or information density, resulting in unnecessary computation for tokens that could be processed effectively with fewer experts (Huang et al., 2024).

Recent work on dynamic expert routing addresses this inefficiency by adapting the number of activated experts per token (Li et al., 2023; Zeng et al., 2024), with inference-time variants being particularly attractive because they can be applied directly to pretrained models without retraining (Huang et al., 2025; Chen et al., 2025). However, existing approaches primarily focus on *when* to reduce expert activation, while largely overlooking a critical consequence: the resulting intermediate representations may deviate from the distribution induced by the training-time routing

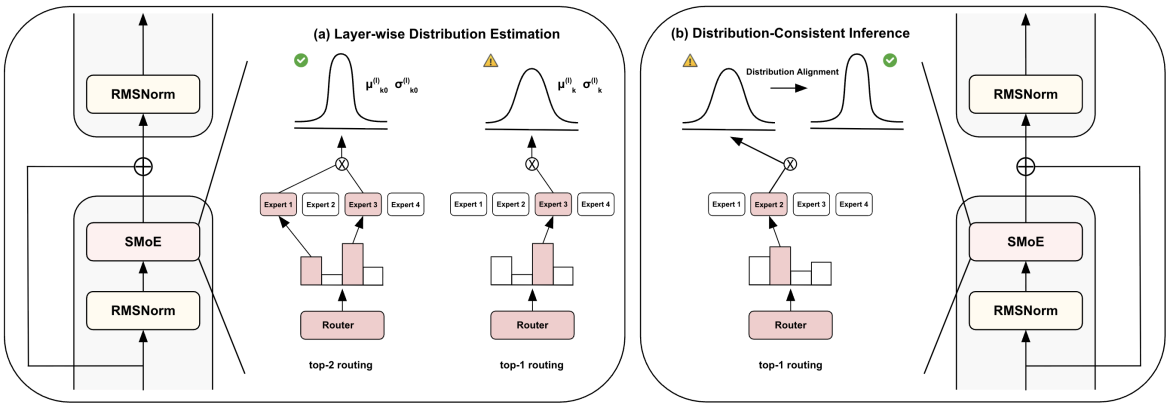


Figure 2: **Overall framework of Layer-wise Distribution Alignment (LDA).** (a): LDA estimates layer-wise reference and target statistics of SMoE outputs from a calibration dataset. (b): At inference time, LDA aligns reduced-routing SMoE outputs to the reference distribution according to k.

configuration.

In this work, we systematically analyze how SMoE representations change as a function of the number of activated experts and find that reducing top-k consistently increases the RMS scale and variance of SMoE outputs. This induces a representation-level mismatch that propagates through layers and contributes to downstream performance degradation. These findings suggest that performance loss under reduced routing is not solely a consequence of diminished expert capacity, but also arises from a correctable distributional shift. Motivated by this observation, we propose *Layer-wise Distribution Alignment* (LDA), a lightweight inference-time correction that aligns representations produced under reduced expert activation with the layer-wise reference distribution of the default top-k_0 routing configuration. LDA applies a per-dimension moment correction after each SMoE layer, requires no retraining or architectural modification, and naturally complements training-free dynamic routing strategies with negligible computational overhead.

We summarize our contributions as follows:

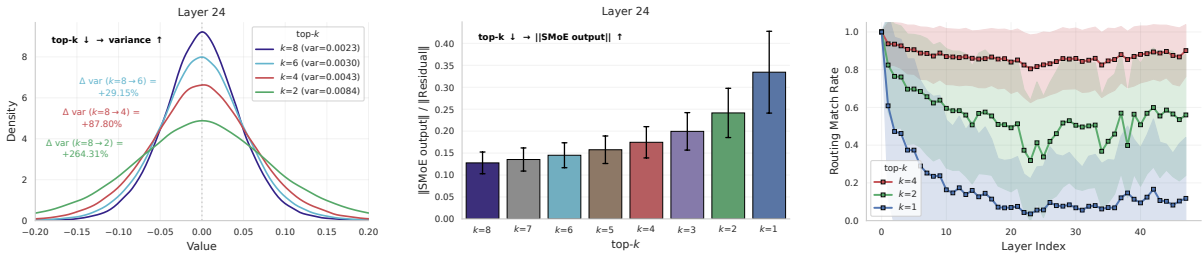
- We show that performance degradation under reduced expert activation is not only due to diminished expert capacity, but is also associated with systematic scale and variance shifts in SMoE representations, a phenomenon not explicitly characterized in prior work.
- We propose *Layer-wise Distribution Alignment* (LDA), a training-free inference-time correction that mitigates this distributional shift using layer-wise calibration statistics.
- We demonstrate that LDA consistently improves the performance–efficiency trade-off across diverse tasks, models, and routing strategies, including top-k routing, adaptive top-p routing, and dynamic expert pruning.

2 Related Works

Sparse Mixture-of-Experts (SMoE) has been proposed to control computational costs by selectively activating only a small set of experts for each token while greatly expanding the total number of parameters. Shazeer et al. (2017) showed that sparse top-k routing is an effective mechanism for scaling model capacity, and this design has since become a standard component of SMoE architectures. Subsequent work extended SMoE to large-scale pretraining (Lepikhin et al., 2021; Fedus et al., 2022), and several recent LLMs have adopted SMoE architectures (Dai et al., 2024; Muennighoff et al., 2025).

Most existing SMoE LLMs are pretrained with a fixed top-k routing configuration (Jiang et al., 2024), which implicitly shapes the representations learned during training. Prior work improves the performance–efficiency trade-off through dynamic expert allocation based on routing confidence (Huang et al., 2024), null experts (Zeng et al., 2024), adaptive top-k selection (Li et al., 2023; Zhong et al., 2025), or training-free expert pruning using token importance (Huang et al., 2025) and expert frequency (Chen et al., 2025).

However, these methods primarily focus on routing strategies, leaving underexplored how reducing the number of activated experts affects the distribution of SMoE outputs. In contrast, we address



(a) SMoE output distribution (b) ||SMoE output|| / ||Residual|| (c) Layer-wise Routing Match Rate

Figure 3: **Observation and analysis of SMoE outputs on Qwen3-30B-A3B.** We use 8×2,048 calibration tokens, and for (a), 100,000 values are randomly sampled. (a): SMoE output distributions become more dispersed as k decreases. (b): The SMoE output scale increases relative to the residual stream as k decreases. (c): The routing match rate (Eq. 6) decreases across layers as k becomes smaller, indicating preservation of the default routing trajectory.

this representation-level mismatch and propose an efficient training-free correction that aligns representations under reduced expert activation. Our method is complementary to existing training-free dynamic routing strategies.

3 Observation and Analysis

3.1 Preliminaries

An SMoE layer consists of a router, a gating mechanism, and a set of N experts $\{E_{\theta_i}\}_{i=1}^{N}$. Given an input $\mathbf{x} \in \mathbb{R}^{D_{\text{in}}}$, the router G_θ produces logits $G_\theta(\mathbf{x}) \in \mathbb{R}^N$, which are converted into routing scores $\mathbf{w} \in \mathbb{R}^N$ through a gating function, such as softmax (Lepikhin et al., 2021) or sigmoid (Lewis et al., 2021):

$$\mathbf{w} = \begin{cases} \text{softmax}(G_\theta(\mathbf{x})), & \text{(softmax gating)}, \\ \sigma(G_\theta(\mathbf{x})), & \text{(sigmoid gating)}. \end{cases}$$

Under top-k routing, only the k experts with the highest routing scores are selected:

$$\mathcal{S} = \text{TopK}(\mathbf{w}, k),$$

where $\mathcal{S}$ denotes the set of selected expert indices. Let $\{w_i\}_{i\in\mathcal{S}}$ denote the routing scores of the selected experts. The selected routing scores can be re-normalized so that they sum to 1:

$$w_i = \begin{cases} \dfrac{w_i}{\sum_{j\in\mathcal{S}} w_j}, & \text{(re-normalization)}, \\ w_i, & \text{(otherwise)}. \end{cases} \quad (1)$$

The output of the SMoE layer is then computed as the weighted sum of the selected expert outputs:

$$\mathbf{y} = \sum_{i\in\mathcal{S}} w_i E_{\theta_i}(\mathbf{x}), \quad (2)$$

where $\mathbf{y} \in \mathbb{R}^{D_{\text{out}}}$ denotes the layer output.

3.2 Observation: Reduced top-k Amplifies Representation Variance and Scale

We focus on SMoE architectures with routing score re-normalization (Eq. 1), which is commonly used in recent SMoE LLMs.

Empirical Observation We empirically observe that reducing top-k monotonically increases the variance of SMoE outputs. As shown in Fig. 3a, the output distribution becomes more dispersed as fewer experts are activated. Alongside this variance increase, the RMS magnitude of the SMoE output also grows, making it larger relative to the residual stream, as shown in Fig. 3b. These observations suggest that reduced expert activation changes not only the routing sparsity, but also the scale of the resulting representation.

Theoretical Insight We provide a simple analysis explaining why reducing top-k amplifies the RMS scale of the SMoE output (Eq. 2) under routing score re-normalization, where $\sum_{i\in\mathcal{S}} w_i = 1$. The RMS is defined as

$$\text{RMS}(\mathbf{y}) = \sqrt{\frac{1}{D}\sum_{i=1}^{D} y_i^2} = \frac{\|\mathbf{y}\|_2}{\sqrt{D}}, \quad (3)$$

which is determined by the squared norm of $\mathbf{y}$. Expanding this term gives,

$$\mathbb{E}\left[\|\mathbf{y}\|_2^2\right] = \sum_{i\in\mathcal{S}} w_i^2 \mathbb{E}\left[\|E_{\theta_i}(\mathbf{x})\|_2^2\right] + \sum_{\substack{i,j\in\mathcal{S} \\ i\neq j}} w_i w_j \mathbb{E}\left[\langle E_{\theta_i}(\mathbf{x}), E_{\theta_j}(\mathbf{x})\rangle\right].$$

Under the mild assumption that the average scale of activated expert outputs does not vary substantially

across top-k settings and that pairwise correlations between expert outputs are weak,

$$\mathbb{E}\left[\|\mathbf{y}\|_2^2\right] \approx \sigma^2 \sum_{i\in\mathcal{S}} w_i^2 \propto \sum_{i\in\mathcal{S}} w_i^2,$$

where σ^2 denotes the average squared norm of the expert outputs and is treated as a constant. Thus, the RMS scale of the SMoE output is mainly governed by the concentration of the normalized routing scores.

When reducing top-k, the remaining routing scores are re-normalized after removing the lower-ranked scores:

$$\tilde{w}_i = \frac{w_i}{1 - w_k}, \quad i = 1, \ldots, k-1.$$

Let $A = \sum_{i=1}^{k-1} w_i^2$ denote the sum of squared routing scores over the remaining $k-1$ scores, then

$$\sum_{i=1}^{k-1} \tilde{w}_i^2 = \frac{A}{(1-w_k)^2} > A + w_k^2 = \sum_{i=1}^{k} w_i^2, \quad (4)$$

where the inequality follows from $w_i \geq w_k$ for $i < k$. Therefore, reducing top-k increases the sum of squared normalized scores, which leads to a larger expected squared norm of the SMoE output and explains the RMS amplification. A detailed derivation is provided in Appendix A.

3.3 Structural Analysis

Modern SMoE LLM architecture Most modern SMoE LLMs adopt residual connections (He et al., 2016) within Pre-LN Transformer architectures (Xiong et al., 2020). In this structure, the SMoE output is added to the residual stream, and the combined representation is normalized in the subsequent block. A commonly used normalization layer is RMSNorm (Zhang and Sennrich, 2019), which rescales a representation by its RMS (Eq. 3):

$$\mathrm{RMSNorm}(\mathbf{x}) = \boldsymbol{\alpha} \odot \frac{\mathbf{x}}{\mathrm{RMS}(\mathbf{x})}, \quad (5)$$

where $\mathbf{x} \in \mathbb{R}^D$ denotes an input representation and $\boldsymbol{\alpha} \in \mathbb{R}^D$ denotes a learnable scaling vector. Thus, changes in the scale of the SMoE output can directly affect the scale balance between the existing residual stream and the newly added weighted expert output.

Scale Imbalance between SMoE Output and Residual In a residual block, the combined representation can be written as $\mathbf{z} = \mathbf{r} + \mathbf{y}$, where $\mathbf{r} \in \mathbb{R}^D$ denotes the residual stream and $\mathbf{y} \in \mathbb{R}^D$ denotes the SMoE output.

The observation above suggests that reducing top-k increases the scale of the SMoE output relative to the residual stream. When the RMS scale of the SMoE output $\mathbf{y}$ increases, the combined representation $\mathbf{z}$ becomes more strongly influenced by the weighted expert output.

Since RMSNorm rescales the entire representation by a single RMS value, an enlarged SMoE output increases the normalization denominator $\mathrm{RMS}(\mathbf{z})$. As it grows, components carried by the residual stream $\mathbf{r}$ can be relatively suppressed after the normalization layer, even if they contain useful information.

Routing Trajectory Mismatch In SMoE models, experts are often specialized to process tokens from different domains or tasks (Dong et al., 2025; Herbst et al., 2026), making it important for each token to be routed to appropriate experts. However, the scale imbalance discussed above can distort the representation passed to subsequent blocks, which may in turn affect the router's expert selection. To quantify this effect, we define *Routing Match Rate* as

$$\text{Routing Match Rate} = \frac{\left|\mathcal{E}_k \cap \mathcal{E}_{k_0}^{(k)}\right|}{\left|\mathcal{E}_{k_0}^{(k)}\right|}, \quad (6)$$

where $\mathcal{E}_k$ denotes the expert set selected under reduced top-k routing, and $\mathcal{E}_{k_0}^{(k)}$ denotes the top-k subset of experts selected under the default top-k_0 routing. A higher routing match rate indicates that reduced top-k routing follows a routing path more similar to the default training configuration.

As shown in Fig. 3c, the routing match rate decreases as the number of activated experts is reduced. This degradation is especially pronounced for smaller k, where the routing path rapidly diverges from the default top-k_0 routing across layers. Consequently, these results show that reduced top-k routing also alters the subsequent routing trajectory across layers.

4 Layer-wise Distribution Alignment (LDA)

Motivated by the observation and analysis in Section 3, we use a lightweight inference-time align-

top-k	Mean	Variance	Skewness	Kurtosis (Fisher)
	Layer 12			
k=8	$0.0000_{\pm 0.0051}$	$\mathbf{0.0012_{\pm 0.0003}}$	$-0.0040_{\pm 0.1165}$	$1.0793_{\pm 0.4076}$
k=4	$0.0000_{\pm 0.0063}$	$\mathbf{0.0022_{\pm 0.0005}}$	$-0.0043_{\pm 0.1140}$	$1.1111_{\pm 0.4467}$
k=2	$0.0000_{\pm 0.0073}$	$\mathbf{0.0044_{\pm 0.0009}}$	$-0.0060_{\pm 0.1229}$	$1.3246_{\pm 0.5706}$
	Layer 24			
k=8	$0.0002_{\pm 0.0060}$	$\mathbf{0.0022_{\pm 0.0013}}$	$0.0010_{\pm 0.1110}$	$0.7520_{\pm 0.6319}$
k=4	$0.0002_{\pm 0.0069}$	$\mathbf{0.0042_{\pm 0.0020}}$	$0.0004_{\pm 0.1055}$	$0.8080_{\pm 0.6136}$
k=2	$0.0002_{\pm 0.0078}$	$\mathbf{0.0083_{\pm 0.0031}}$	$0.0007_{\pm 0.1193}$	$1.0866_{\pm 0.9998}$
	Layer 36			
k=8	$0.0002_{\pm 0.0125}$	$\mathbf{0.0063_{\pm 0.0056}}$	$0.0012_{\pm 0.1850}$	$1.5756_{\pm 1.3413}$
k=4	$0.0003_{\pm 0.0154}$	$\mathbf{0.0121_{\pm 0.0096}}$	$0.0001_{\pm 0.1803}$	$1.3168_{\pm 1.0057}$
k=2	$0.0005_{\pm 0.0184}$	$\mathbf{0.0243_{\pm 0.0156}}$	$0.0005_{\pm 0.2134}$	$1.5302_{\pm 1.2054}$

Table 1: **Per-dimension moment statistics of SMoE outputs across different top-k.** We report mean$_{\pm \text{std}}$ for each moment. Only the variance consistently shows a substantial increase as k decreases.

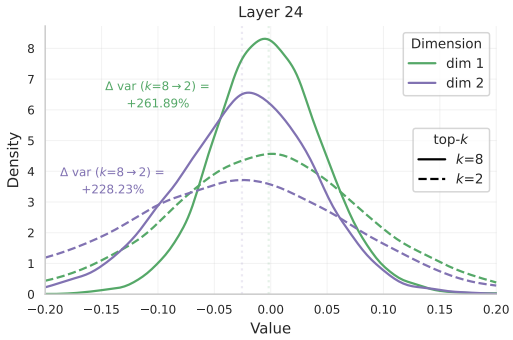


Figure 4: **Per-dimension distribution of SMoE outputs on Qwen3-30B-A3B.** Each dimension shows different degrees of variance amplification under reduced top-k routing.

ment to mitigate the representation scale and variance mismatch induced by reduced top-k routing. The key idea is to use the layer-wise representation statistics under the default top-k_0 configuration as a reference, and align the representations produced under reduced top-k routing to the reference distribution. A detailed pseudo-code description is provided in Appendix G.

4.1 Per-dimension Moment Alignment

Original Representation Space Since SMoE outputs computed under top-k_0 or different top-k settings lie in the same representation space, we align their statistics in the original representation space instead of applying an additional projection or non-linear transformation.

Per-dimension Alignment Theoretical insight as discussed in Section 3.2 doesn't imply that the amplification is not uniform across dimensions. As shown in Fig. 4, the variance shift differs substantially across dimensions, suggesting that a single global scaling factor is insufficient to correct the mismatch. We therefore apply a per-dimension correction.

First- and Diagonal Second-order Moment Alignment As shown in Table 1, the significant and consistent shift appears in the variance, while higher-order moments such as skewness and kurtosis exhibit weaker and less consistent changes. Thus, aligning the per-dimension mean and standard deviation provides a simple affine correction that targets the dominant observed shift, while avoiding the computational cost of full-covariance alignment or higher-order transformations.

4.2 Layer-wise Distribution Estimation

We estimate the required statistics using a calibration set. For each SMoE layer l, we compute the per-dimension mean and standard deviation of the SMoE output under the default top-k_0 configuration and under reduced top-k routing. We denote the reference statistics from the default configuration as $\boldsymbol{\mu}_{k_0}^{(l)}, \boldsymbol{\sigma}_{k_0}^{(l)} \in \mathbb{R}^D$, and the target statistics from reduced routing as $\boldsymbol{\mu}_{k}^{(l)}, \boldsymbol{\sigma}_{k}^{(l)} \in \mathbb{R}^D$.

During calibration, hidden states are propagated through preceding layers using the default top-k_0 routing. This prevents distribution shifts from accumulating across layers during statistics estimation and provides a stable reference for measuring how each layer's SMoE output changes under reduced top-k routing.

4.3 Distribution-Consistent Inference

At inference time, when an SMoE layer uses reduced top-k routing, we align its output to the reference statistics of the corresponding default top-k_0 configuration. Given the SMoE output $\mathbf{y}^{(l)} \in \mathbb{R}^D$ at layer l, LDA applies the following per-dimension affine transformation:

$$\hat{\mathbf{y}}^{(l)} = \begin{cases} \mathbf{y}^{(l)}, & (k = k_0), \\ \boldsymbol{\sigma}_{k_0}^{(l)} \odot \dfrac{\mathbf{y}^{(l)} - \boldsymbol{\mu}_{k}^{(l)}}{\boldsymbol{\sigma}_{k}^{(l)} + \epsilon} + \boldsymbol{\mu}_{k_0}^{(l)}, & (k < k_0). \end{cases}$$

This operation restores the first- and diagonal second-order statistics of reduced top-k representations toward those of the default training configuration. Since it only requires an element-wise affine transformation, LDA introduces negligible inference overhead and requires no additional training or parameter updates. It can also be combined with existing training-free dynamic routing strategies,

Method		w/ LDA	Avg. k	*General Knowledge & Commonsense Reasoning Tasks*								
				MMLU	**Hella.**	**Wino.**	**ARC-E**	**ARC-C**	**CQA**	**SciQ**	**PIQA**	**Average**
Baseline (k=8)		✗	8.00	**77.87%**	59.52%	70.24%	79.80%	53.16%	79.12%	96.50%	79.33%	74.44%
top-k	k=4	✗	4.00	70.72%	55.73%	60.06%	69.57%	40.78%	65.52%	92.80%	76.77%	66.49%
		✓	4.00	75.56%	57.12%	67.72%	77.44%	47.44%	77.31%	95.90%	78.89%	72.17%
	k=2	✗	2.00	27.97%	36.02%	50.99%	41.29%	24.83%	22.36%	70.90%	61.48%	41.98%
		✓	2.00	67.40%	49.43%	58.41%	71.42%	40.27%	66.17%	93.70%	74.97%	65.22%
top-p	p=0.3	✗	5.62	75.35%	57.85%	66.22%	70.33%	43.60%	60.61%	95.00%	78.51%	68.43%
		✓	5.43	76.46%	58.23%	70.96%	79.38%	51.62%	79.28%	96.00%	79.11%	73.88%
	p=0.2	✗	3.64	58.25%	51.68%	58.25%	56.61%	34.39%	50.45%	88.90%	72.85%	58.92%
		✓	3.38	73.05%	54.65%	64.88%	74.28%	44.20%	76.74%	95.30%	77.26%	70.04%
	p=p*	✗	7.80	77.80%	59.57%	70.72%	79.71%	53.33%	78.95%	96.40%	79.38%	74.55%
		✓	7.52	77.81%	**59.75%**	**71.98%**	**80.26%**	**53.92%**	**80.51%**	**96.70%**	**79.71%**	75.08%
PESF (α=0.3)		✗	7.87	76.20%	56.81%	68.90%	78.11%	50.34%	77.95%	94.80%	78.29%	72.67%
		✓	7.87	76.30%	57.49%	69.53%	79.08%	50.51%	78.29%	94.80%	78.46%	73.07%

Method		w/ LDA	Avg. k	*Mathematical Reasoning Tasks*			*Code Generation Tasks*			*Instruction-Following Tasks*
				GSM8K	**MATH**	**Average**	**MBPP**	**HumanEval**	**Average**	**IFEval**
Baseline (k=8)		✗	8.00	88.02%	68.20%	78.11%	**72.60%**	84.76%	78.68%	26.25%
top-k	k=4	✗	4.00	74.15%	38.00%	56.08%	34.80%	66.46%	50.63%	24.95%
		✓	4.00	87.49%	64.60%	76.05%	68.00%	72.56%	70.28%	24.77%
	k=2	✗	2.00	00.99%	00.00%	00.50%	00.00%	00.61%	00.31%	09.43%
		✓	2.00	68.54%	42.60%	55.57%	44.40%	36.59%	40.50%	17.74%
top-p	p=0.3	✗	7.21	86.35%	67.40%	76.88%	67.20%	77.44%	72.32%	28.10%
		✓	7.17	**88.93%**	68.00%	78.47%	71.00%	79.88%	75.44%	**30.13%**
	p=0.2	✗	5.46	73.16%	32.80%	52.98%	36.00%	46.95%	41.48%	24.03%
		✓	5.40	85.75%	60.20%	72.98%	61.60%	73.17%	67.39%	24.40%
	p=p*	✗	7.87	88.02%	68.80%	78.41%	72.40%	84.15%	78.28%	28.84%
		✓	7.66	**88.93%**	**70.60%**	79.77%	**72.60%**	**85.37%**	78.99%	**30.13%**
PESF (α=0.3)		✗	7.78	87.62%	18.60%	53.11%	0.00%	76.21%	38.11%	25.13%
		✓	7.78	87.41%	64.40%	75.91%	39.40%	79.26%	59.33%	27.54%

Table 2: **Performance across a wide range of downstream tasks under different routing strategies on Qwen3-30B-A3B.** We use deterministic algorithms for the evaluations. Avg. k denotes the average number of activated experts, and p* denotes the best-performing top-p threshold for each task, reported in Table 9. Results with the highest performance are highlighted in **red**.

as it operates independently of how the reduced number of activated experts is selected.

5 Experiments

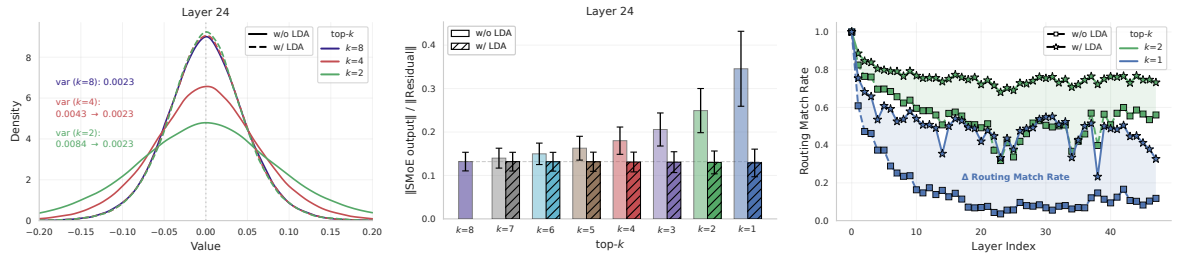
5.1 Experimental Setup

We evaluate LDA on recent SMoE LLMs, including Qwen3-30B-A3B (Team, 2025), kanana-2-30b-a3b-instruct (LLM, 2025) based on DeepSeek-V3 architecture (DeepSeek-AI et al., 2025), and GLM-4.7-Flash (Team et al., 2025). We consider top-k routing, dynamic top-p routing (Huang et al., 2024), and PESF (Chen et al., 2025) as inference-time routing baselines, and compare each reduced-routing with and without LDA across 13 benchmarks: 8 general knowledge & commonsense reasoning tasks, 2 mathematical reasoning tasks, 2 code generation tasks, and 1 instruction-following task. Layer-wise statistics are estimated using 4×2048 tokens sampled from C4 (Dodge et al., 2021) calibration subset. We report accuracy-based metrics for all benchmark tasks. Additional details on baselines, models, benchmarks, evaluation settings, and implementation are provided in Appendix B.

5.2 Performance across Routing Strategies

Table 2 reports results under different inference-time routing strategies, including fixed top-k routing, dynamic top-p routing, and PESF. Across these strategies, performance degradation becomes more pronounced as the average number of activated experts decreases.

Under fixed top-k routing, LDA substantially improves over reduced top-k baselines, particularly in low-k regimes. When k is reduced to 4 or 2, the vanilla baseline suffers large drops across tasks, whereas LDA preserves considerably higher performance under the same expert budget. This suggests that the degradation caused by reducing expert activation is not solely attributable to lower



(a) SMoE output distribution (b) ||SMoE output|| / ||Residual|| (c) Layer-wise Routing Match Rate

Figure 5: **Effect of LDA on SMoE outputs in Qwen3-30B-A3B.** We use 2,048 held-out calibration tokens, and for (a), 100,000 values are randomly sampled. (a): LDA aligns reduced top-k SMoE output distributions toward the default $k = 8$ distribution. (b): LDA suppresses the increased SMoE output scale relative to the residual stream. (c) LDA increases the routing match rate (Eq. 6) across layers, indicating reduced routing trajectory mismatch.

Task	top-k	w/o LDA	w/ LDA	p-value
MMLU	k=6	$76.92\%_{\pm 0.00\%}$	$77.34\%_{\pm 0.00\%}$	**0.0341**
	k=4	$70.72\%_{\pm 0.00\%}$	$75.56\%_{\pm 0.00\%}$	**1.9e-57**
MATH	k=6	$58.00\%_{\pm 0.89\%}$	$65.40\%_{\pm 0.49\%}$	**1.3e-04**
	k=4	$35.72\%_{\pm 1.61\%}$	$61.24\%_{\pm 1.65\%}$	**8.4e-05**
MBPP	k=6	$68.56\%_{\pm 0.64\%}$	$70.80\%_{\pm 1.29\%}$	**0.0308**
	k=4	$43.56\%_{\pm 3.81\%}$	$68.00\%_{\pm 0.38\%}$	**3.7e-04**

Table 3: **Statistical significance of LDA improvements.** For MMLU, we apply McNemar's test. For MATH and MBPP, we apply paired t-test and report $\text{mean}_{\pm \text{CI95}}$ over 5 random seeds. LDA yields statistically significant improvements ($p < 0.05$) across all settings.

expert capacity, but is also closely tied to the representation variance and scale mismatch analyzed in Section 3.

We observe a similar tendency under dynamic top-p routing and PESF. For top-p routing, LDA consistently improves performance at the same threshold, and with an appropriate p value, it can match or exceed the default top-k_0 performance while using fewer experts on average. For PESF, LDA also improves performance over the corresponding dynamic expert-pruning baseline, indicating that LDA is effective beyond fixed reduced top-k routing. Overall, these results show that LDA can be combined with diverse training-free dynamic routing strategies by correcting the representation mismatch induced by reduced expert activation.

We provide results on additional models in Appendix D, where LDA yields consistent improvements under reduced expert activation budgets.

5.3 Statistical Significance of Improvements

To assess whether the observed improvements are attributable to evaluation noise, we conduct statistical significance tests under representative reduced top-k settings. For multiple-choice tasks such as MMLU (Hendrycks et al., 2021a), where predictions are deterministic, we apply McNemar's test (McNemar, 1947) to paired per-example correctness, comparing reduced top-k routing with and without LDA. For generation tasks such as MATH (Hendrycks et al., 2021b) and MBPP (Austin et al., 2021), we evaluate each method across five random seeds and use a paired t-test (Student, 1908) to compare task-level performance. As shown in Table 3, LDA yields statistically significant improvements ($p < 0.05$) across all evaluated settings. We provide the specific null hypotheses and additional testing details in Appendix C.

5.4 Effect of LDA

We analyze how LDA affects representations under reduced top-k routing. As shown in Fig. 5a, LDA aligns the per-dimension distribution of SMoE outputs toward the default top-k_0 configuration. After correction, outputs from different reduced top-k settings exhibit comparable means and variances, indicating that LDA mitigates the distributional mismatch identified in Section 3.2. This correction also stabilizes the relative scale of the SMoE output. Fig. 5b shows that the ratio between the SMoE output scale and the residual stream remains close to that of the default setting after applying LDA, suggesting that LDA reduces the scale imbalance amplified under reduced routing.

We further examine whether this correction affects subsequent routing behavior. Fig. 5c compares the routing match rate in Eq. 6 before and after LDA. Across layers, LDA consistently increases the match rate relative to the reduced top-k baseline, especially for smaller k. This indicates that

LDA helps preserve routing trajectories closer to those induced by the default routing configuration.

top-k	w/ LDA	ShareGPT		NuminaMath-1.5		TFLOPs↓
		TPOT(ms)↓	Throughput(tok/s)↑	TPOT(ms)↓	Throughput(tok/s)↑	
k=8	✗	33.22	910.71	29.97	1035.71	≈ 14.48
k=6	✗	30.14	999.78	27.01	1147.42	≈ 12.63
	✓	30.26	995.09	26.30	1177.90	≈ 12.63
k=4	✗	26.00	1156.26	22.57	1368.72	≈ 10.77
	✓	25.97	1153.90	22.02	1402.42	≈ 10.77
k=2	✗	20.11	1477.55	17.13	1780.97	≈ 8.92
	✓	20.29	1461.31	16.74	1819.19	≈ 8.92

Table 4: **Inference efficiency comparison with and without LDA.** We report TPOT, output token throughput, and TFLOPs across different top-k settings on ShareGPT and NuminaMath-1.5 in Qwen3-30B-A3B. TFLOPs are computed with a sequence length of 2,048. The results show that applying LDA introduces negligible additional inference overhead.

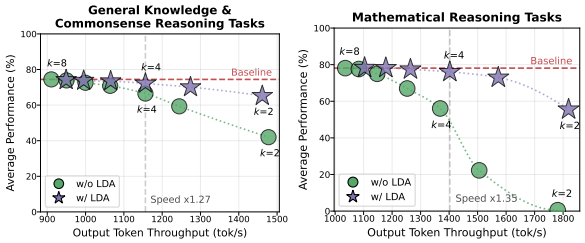


Figure 6: **Performance–throughput trade-off on Qwen3-30B-A3B.** We compare the average performance and output token throughput under different top-k settings. LDA consistently shifts the trade-off curve upward. Throughput is measured using ShareGPT and NuminaMath-1.5 for general reasoning and mathematical reasoning, respectively.

5.5 Trade-off between Inference Efficiency and Performance

We evaluate inference efficiency in a realistic serving setting using `vLLM` framework (Kwon et al., 2023), measuring *Time Per Output Token* (TPOT) and *Output Token Throughput*. All experiments are conducted on a single NVIDIA RTX PRO 6000 Blackwell Max-Q GPU (96GB). Our analysis focuses on the performance–efficiency trade-off induced by varying the number of activated experts. To capture domain-dependent inference behavior, we use the ShareGPT dataset (Chiang et al., 2023) for general knowledge & commonsense reasoning, and NuminaMath-1.5 (LI et al., 2024) for mathematical reasoning.

Table 4 reports the inference metrics, and Fig. 6 visualizes the trade-off between average task performance and output token throughput. As the number of activated experts decreases, inference becomes more efficient, leading to lower FLOPs and higher throughput. However, under vanilla top-k routing, this efficiency gain comes with a substantial performance drop, especially at smaller k values.

In contrast, LDA improves this trade-off by preserving much of the task performance while retaining the efficiency benefit of reduced expert activation. At the same reduced top-k setting, LDA consistently achieves higher average performance than vanilla top-k routing with comparable inference cost. This is because LDA only applies a per-dimension affine transformation to the SMoE output at each layer, whose additional computational cost is $O(LD)$ across the entire model, where L is the number of SMoE layers and D is the hidden dimension. This overhead is negligible compared to the entire computation cost, and it does not require additional training or parameter updates.

This improved trade-off is consistent across both evaluated domains but is especially pronounced in mathematical reasoning, where vanilla reduced top-k routing otherwise suffers catastrophic degradation. Ultimately, these results demonstrate that LDA effectively decouples throughput gains from severe performance penalties, offering a highly practical solution for efficient SMoE serving.

6 Conclusion

In this work, we studied the performance degradation of Sparse Mixture-of-Experts (SMoE) models under reduced expert activation from the perspective of representation scale and distribution mismatch. We observed that reducing top-k amplifies the scale and variance of SMoE outputs, disturbing the balance between the weighted expert output and the residual stream. To address this issue, we proposed *Layer-wise Distribution Alignment* (LDA), a training-free inference-time correction that aligns reduced top-k representations with the layer-wise statistics of the default training configuration. LDA is lightweight, requiring only an element-wise affine transformation with $O(LD)$ additional cost across the model, and is compatible with training-free dynamic routing strategies. Experiments across multiple SMoE LLMs and diverse downstream tasks show that LDA consistently mitigates performance degradation under reduced expert activation and improves the performance–efficiency trade-off across routing strategies. These results highlight the importance of maintaining representation scale and distribution consistency for dynamic routing in SMoE models.

Limitations

LDA provides a training-free inference-time correction for the representation scale and distribution mismatch induced by reduced expert activation, but it also has several limitations.

First, LDA summarizes each layer-wise distribution using per-dimension mean and standard deviation. This design makes the method lightweight and stable, but it does not capture full covariance structure, distributional asymmetry, or higher-order statistics. Although our empirical analysis shows that variance is the most consistent and substantial shift under reduced top-k routing, more expressive alignment methods may further improve performance in certain layers or tasks.

Second, LDA requires a calibration stage to estimate layer-wise reference and target statistics. Our ablation studies show that LDA is relatively robust to the choice and size of the calibration dataset, but it is not completely calibration-free. Reducing this calibration cost further or developing online statistic estimation methods could improve the practicality of LDA in deployment settings.

Finally, our dynamic top-p routing experiments use manually selected thresholds. With an appropriate threshold, LDA can achieve performance higher than the default routing baseline while using fewer activated experts. However, this threshold must be selected for each task, which introduces an additional task-specific design choice. So, developing training-free dynamic routing strategies that automatically determine the number of activated experts without task-specific tuning, and combining them with LDA, can be an important direction for future work.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Yonatan Bisk, Rowan Zellers, Ronan Le bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7432–7439.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2025. A Survey on Mixture of Experts in Large Language Models . *IEEE Transactions on Knowledge & Data Engineering*, 37(07):3896–3915.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Yuanteng Chen, Yuantian Shao, Peisong Wang, and Jian Cheng. 2025. EAC-MoE: Expert-selection aware compressor for mixture-of-experts large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12942–12963, Vienna, Austria. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv: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. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv: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. 2024. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1280–1297, Bangkok, Thailand. Association for Computational Linguistics.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zican Dong, Han Peng, Peiyu Liu, Xin Zhao, Dong Wu, Feng Xiao, and Zhifeng Wang. 2025. Domain-specific pruning of large mixture-of-experts models with few-shot demonstrations. In *Advances in Neural Information Processing Systems*, volume 38, pages 103552–103577. Curran Associates, Inc.

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

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *Advances in neural information processing systems*.

Jeremy Herbst, Jae Hee Lee, and Stefan Wermter. 2026. The expert strikes back: Interpreting mixture-of-experts language models at expert level. *Preprint*, arXiv:2604.02178.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024. Harder task needs more experts: Dynamic routing in MoE models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12883–12895.

Wei Huang, Yue Liao, Jianhui Liu, Ruifei He, Haoru Tan, Shiming Zhang, Hongsheng Li, Si Liu, and XIAOJUAN QI. 2025. Mixture compressor for mixture-of-experts LLMs gains more. In *The Thirteenth International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

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

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. Base layers: Simplifying training of large, sparse models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6265–6274. PMLR.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath.

Jiamin Li, Qiang Su, Yitao Yang, Yimin Jiang, Cong Wang, and Hong Xu. 2023. Adaptive gating in mixture-of-experts based language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3577–3587, Singapore.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. In *Proceedings of Machine Learning and Systems*, volume 6, pages 87–100.

Kanana LLM. 2025. Kanana-2 llm.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Evan Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, and 5 others. 2025. OLMoe: Open mixture-of-experts language models. In *The Thirteenth International Conference on Learning Representations*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *Preprint*, arXiv:1907.10641.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Student. 1908. The probable error of a mean. *Biometrika*, 6(1):1–25.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

GLM Team, Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, Kedong Wang, Lucen Zhong, Mingdao Liu, Rui Lu, Shulin Cao, Xiaohan Zhang, Xuancheng Huang, Yao Wei, and 152 others. 2025. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *Preprint*, arXiv:2508.06471.

Qwen 3 Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, and Mi Zhang. 2024. Efficient large language models: A survey. *Transactions on Machine Learning Research*. Survey Certification.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10524–10533. PMLR.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Zihao Zeng, Yibo Miao, Hongcheng Gao, Hao Zhang, and Zhijie Deng. 2024. AdaMoE: Token-adaptive routing with null experts for mixture-of-experts language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6223–6235, Miami, Florida, USA. Association for Computational Linguistics.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Shuzhang Zhong, Ling Liang, Yuan Wang, Runsheng Wang, Ru Huang, and Meng Li. 2025. Adapmoe: Adaptive sensitivity-based expert gating and management for efficient moe inference. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '24. Association for Computing Machinery.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Layer Index	$\mathbb{E}(\lvert\cos(E_{\theta_{i\in S_{k_0}}}(\mathbf{x}), E_{\theta_{j\in S_{k_0}}}(\mathbf{x}))\rvert)$	$\mathbb{E}(\lVert E_{\theta_{i\in S_k}}(\mathbf{x})\rVert_2)$			$\sum_{i\in S_k} w_i^2$		
		k=8	k=4	k=2	k=8	k=4	k=2
12	$0.0321_{\pm 0.0080}$	$3.2769_{\pm 0.6915}$	$3.5664_{\pm 0.8044}$	$3.7829_{\pm 0.9823}$	$0.1455_{\pm 0.0208}$	$0.2675_{\pm 0.0228}$	$0.5125_{\pm 0.0217}$
24	$0.0333_{\pm 0.0089}$	$4.8823_{\pm 0.8127}$	$5.2075_{\pm 0.9377}$	$5.4284_{\pm 1.2085}$	$0.1401_{\pm 0.0211}$	$0.2634_{\pm 0.0226}$	$0.5102_{\pm 0.0211}$
36	$0.0319_{\pm 0.0093}$	$7.4705_{\pm 1.2000}$	$8.3009_{\pm 1.5161}$	$8.9204_{\pm 2.0796}$	$0.1440_{\pm 0.0221}$	$0.2664_{\pm 0.0239}$	$0.5122_{\pm 0.0229}$

Table 5: **Empirical statistics related to the mild assumptions in Qwen3-30B-A3B.** We use 8×2,048 calibration tokens and report mean$_{\pm\text{std}}$ for each measurement. The results show near-zero off-diagonal cosine similarity, average activated expert output norms that remain within a comparable range and tend to increase as k decreases, and increased routing-score concentration as k decreases.

A Theoretical Insight

We provide additional details for the theoretical insight in Section 3.2. We first derive the inequality 4. We then provide empirical measurements related to the mild assumptions used in the approximation.

A.1 Derivation of Eq. 4

We derive Eq. 4, which shows that removing the smallest selected routing score and re-normalizing the remaining scores increases the sum of squared routing scores.

Assume that the selected routing scores are ordered as

$$w_1 \geq w_2 \geq \cdots \geq w_k > 0,$$

and normalized such that

$$\sum_{i=1}^{k} w_i = 1.$$

After removing the smallest selected score w_k, the remaining $k-1$ scores are re-normalized as

$$\tilde{w}_i = \frac{w_i}{1-w_k}, \quad i = 1, \ldots, k-1.$$

Let A denote the sum of squared routing scores over the remaining $k-1$ scores before re-normalization,

$$A = \sum_{i=1}^{k-1} w_i^2.$$

Then,

$$\sum_{i=1}^{k-1} \tilde{w}_i^2 = \sum_{i=1}^{k-1} \left(\frac{w_i}{1-w_k}\right)^2 = \frac{A}{(1-w_k)^2}.$$

We now show that this quantity is larger than the original squared sum over all k selected scores,

$$\frac{A}{(1-w_k)^2} > A + w_k^2.$$

Multiplying both sides by $(1-w_k)^2 > 0$,

$$A > (A + w_k^2)(1-w_k)^2.$$

Rearranging the inequality gives

$$A\, w_k(2-w_k) > \left(w_k(1-w_k)\right)^2.$$

Since $w_i \geq w_k$ for all $i < k$,

$$A = \sum_{i=1}^{k-1} w_i^2 \geq w_k \sum_{i=1}^{k-1} w_i = w_k(1-w_k).$$

Using $A \geq w_k(1-w_k)$ and $0 < w_k < 1$,

$$A\, w_k(2-w_k) \geq w_k(1-w_k)\, w_k(2-w_k).$$

Moreover, since $2 - w_k > 1 - w_k$,

$$w_k(1-w_k)\, w_k(2-w_k) > \left(w_k(1-w_k)\right)^2.$$

Thus,

$$A\, w_k(2-w_k) > \left(w_k(1-w_k)\right)^2,$$

which proves

$$\frac{A}{(1-w_k)^2} > A + w_k^2.$$

This shows that removing the smallest routing score and re-normalizing the remaining scores increases the sum of squared routing scores.

A.2 Empirical Support for Mild Assumptions

The theoretical insight in Section 3.2 relies on mild assumptions that the average scale of activated expert outputs does not vary substantially across different top-k settings and that pairwise correlations between selected expert outputs are weak. To empirically examine these assumptions, we define the corresponding measurements and report statistics based on activated expert outputs and normalized routing scores in Table 5.

We first measure the average absolute off-diagonal cosine similarity between activated expert outputs:

$$\mathbb{E}\left[\left|\cos\left(E_{\theta_{i\in S_{k_0}}}(\mathbf{x}), E_{\theta_{j\in S_{k_0}}}(\mathbf{x})\right)\right|\right], \quad i \neq j,$$

which measures the magnitude of directional correlation between different activated expert outputs. As shown in Table 5, the value remains around 0.03 across layers, indicating that activated expert outputs have weak pairwise correlations.

We then measure the average norm of activated expert outputs and the sum of normalized routing scores under different top-k settings:

$$\mathbb{E}\left[\left\|E_{\theta_{i\in S_k}}(\mathbf{x})\right\|_2\right], \quad \sum_{i\in S_k} w_i^2.$$

Table 5 compares these two quantities across different top-k settings. The average norm of activated expert outputs remains within a comparable range and slightly increases as k decreases, indicating that reduced top-k routing does not reduce the average scale of activated expert outputs. In contrast, the sum of squared normalized routing scores, $\sum_{i\in S_k} w_i^2$, increases much more substantially as k decreases. This suggests that the increase in SMoE output scale is mainly associated with the increased concentration of normalized routing weights. Together with the near-zero off-diagonal cosine similarity, these results provide empirical support for the approximation used in Section 3.2 and additional evidence for the RMS amplification mechanism under reduced top-k routing.

B Experimental Setup

We provide further details on our experimental setup, including baselines, models, benchmarks, and evaluation protocols.

B.1 Baselines

We consider three routing strategies as baselines: fixed top-k routing, dynamic top-p routing (Huang et al., 2024) and PESF (Chen et al., 2025). For each routing strategy, we compare reduced expert activation with and without LDA to validate the effect of distribution alignment.

top-k Routing top-k routing is the standard routing strategy used in many SMoE LLMs, where each token activates a fixed number of experts. This setting provides a controlled comparison of LDA by keeping the expert budget identical between the vanilla reduced top-k baseline and its LDA-applied counterpart.

top-p Routing top-p routing is a dynamic expert routing strategy that selects experts until the cumulative routing score exceeds a threshold p. Unlike fixed top-k routing, the number of activated experts can vary across tokens depending on the routing distribution. To preserve the inference–efficiency motivation, we restrict the maximum number of activated experts so that it does not exceed the default top-k_0 configuration. This setting allows us to evaluate whether LDA remains effective when the number of activated experts is determined dynamically at inference time. The evaluated threshold values are described in Appendix B.4.

PESF Pruning based on Expert-Selection Frequency (PESF) is a dynamic expert pruning method applied at inference time. For each input sequence, PESF counts how many times each expert is selected by the router. Then, An expert is pruned if its selection count is smaller than a predefined threshold:

$$c_i < \frac{l \times K}{N} \times \alpha,$$

where c_i denotes the number of times expert E_{θ_i} is selected, l is the input sequence length, K is the number of selected experts per token before pruning, N is the total number of experts, and α is the pruning threshold. Since the set of retained experts can vary across each token, PESF can be viewed as an inference-time dynamic routing strategy that changes the number of activated experts. We set $\alpha \in \{0.3, 0.7\}$ following prior work.

B.2 Models

We evaluate LDA on recent SMoE LLMs with diverse architectures to verify that it is not limited to a specific model structure and works consistently across different architectural designs.

Qwen3-30B-A3B (Team, 2025) This model has 30.5B total parameters, with 3.3B activated parameters per token. Each MoE layer consists of 128 experts, among which 8 routed experts are activated per token. The routing scores are computed using softmax gating.

kanana-2-30b-a3b-instruct (LLM, 2025) This model is based on the DeepSeek-V3 architecture (DeepSeek-AI et al., 2025) and has 30B total parameters with approximately 3B activated parameters per token. Each MoE layer consists of 128

Task	Metric	Few-shot	Max Seq. Length	Max Gen. Tokens
General Knowledge & Commonsense Reasoning Tasks				
MMLU (Hendrycks et al., 2021a)	Accuracy	0-shot	4096	-
Hellaswag (Zellers et al., 2019)	Accuracy	0-shot	4096	-
Winogrande (Sakaguchi et al., 2019)	Accuracy	0-shot	4096	-
ARC-Easy (Clark et al., 2018)	Accuracy	0-shot	4096	-
ARC-Challenge (Clark et al., 2018)	Accuracy	0-shot	4096	-
CommonsenseQA (Talmor et al., 2019)	Accuracy	0-shot	4096	-
ScienceQA (Welbl et al., 2017)	Accuracy	0-shot	4096	-
PIQA (Bisk et al., 2020)	Accuracy	0-shot	4096	-
Mathematical Reasoning Tasks				
GSM8K (Cobbe et al., 2021)	Exact Match	8-shot (single-turn)	4096	1024
MATH (Hendrycks et al., 2021b)	Exact Match	4-shot (single-turn)	4096	1024
Code Generation Tasks				
MBPP (Austin et al., 2021)	Pass@1	3-shot (single-turn)	4096	1024
HumanEval (Chen et al., 2021)	Pass@1	0-shot	4096	1024
Instruction-Following Tasks				
IFEval (Zhou et al., 2023)	Accuracy[prompt-level]	0-shot	4096	1024

Table 6: **Benchmark-specific evaluation settings.** We report the evaluation metric, few-shot, max sequence length, and max generation tokens for each benchmark.

experts, among which 8 experts (2 shared experts and 6 routed experts) are activated per token. The routing scores are computed using sigmoid gating.

GLM-4.7-Flash (Team et al., 2025) This model has 30B total parameters with approximately 3B activated parameters per token. Each MoE layer consists of 64 experts, among which 5 experts (1 shared expert + 4 routed experts) are activated per token. The routing scores are calculated with sigmoid gating.

B.3 Benchmarks

We evaluate LDA across four domains of benchmarks: general knowledge & commonsense reasoning, mathematical reasoning, code generation, and instruction-following.

General Knowledge & Commonsense Reasoning Tasks: MMLU (Hendrycks et al., 2021a), Hellaswag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2019), ARC (ARC-Easy, ARC-Challenge) (Clark et al., 2018), CommonsenseQA (CQA) (Talmor et al., 2019), ScienceQA (SciQ) (Welbl et al., 2017), and PIQA (Bisk et al., 2020)

Mathematical Reasoning Tasks: GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021b)

Code Generation Tasks: MBPP (Austin et al., 2021) and HumanEval (Chen et al., 2021)

Instruction-Following Tasks: IFEval (Zhou et al., 2023)

B.4 Evaluation Protocols

We use `lm_eval` framework (Gao et al., 2024) for benchmark evaluations. For reproducibility, we use the default random seed configuration provided by the evaluation framework. We categorize the evaluated benchmarks into option-based tasks and generation-based tasks, and detailed benchmark-specific settings are provided in Table 6.

GPU Resources All experiments were conducted on a single NVIDIA RTX PRO 6000 Blackwell Max-Q GPU (96GB), using bfloat16 precision for inference without quantization.

Option-based Tasks These tasks require the model to select an answer from a given set of candidates. This category includes the general knowledge and commonsense reasoning tasks described in Appendix B.3 Given an input prompt, we compute the probability of each candidate answer from the final logits and select the candidate with the highest probability as the model prediction. This allows deterministic evaluation.

Generation-based Tasks These tasks require the model to generate an output sequence, which is then filtered and compared against the reference answer. This category includes the mathematical reasoning, code generation, and instruction-following tasks described in Appendix B.3. We use greedy decoding for deterministic evaluation and follow the task-specific filtering and answer extraction procedures provided by `lm_eval` framework.

Method		w/ LDA	Avg. k	*General Knowledge & Commonsense Reasoning Tasks*								
				MMLU	**Hella.**	**Wino.**	**ARC-E**	**ARC-C**	**CQA**	**SciQ**	**PIQA**	**Average**
Baseline (k=2+6)		✗	2+6.00	73.63%	**62.16%**	72.14%	85.69%	58.79%	65.68%	97.00%	80.69%	74.47%
top-k	k=2+4	✗	2+4.00	71.81%	61.13%	71.27%	84.43%	55.20%	59.95%	96.50%	79.49%	72.47%
		✓	2+4.00	73.33%	61.31%	72.69%	86.03%	56.57%	65.85%	97.20%	80.52%	74.19%
	k=2+2	✗	2+2.00	60.87%	53.76%	58.96%	74.28%	45.56%	46.36%	90.20%	75.08%	63.13%
		✓	2+2.00	68.65%	55.09%	65.82%	83.92%	53.50%	63.31%	96.20%	78.18%	70.58%
	k=2+1	✗	2+1.00	23.89%	32.28%	51.30%	35.27%	20.56%	19.41%	54.40%	56.09%	36.65%
		✓	2+1.00	49.42%	44.04%	56.12%	70.62%	37.80%	43.41%	92.90%	70.46%	58.10%
top-p	p=0.2	✗	2+3.45	69.21%	59.50%	66.61%	81.19%	50.60%	57.08%	95.00%	77.86%	69.63%
		✓	2+3.52	70.43%	59.50%	69.38%	84.72%	55.89%	67.24%	96.70%	77.97%	72.73%
	p=0.1	✗	2+2.50	23.66%	31.72%	50.12%	35.27%	20.22%	19.82%	42.00%	59.09%	35.24%
		✓	2+1.93	58.92%	49.05%	60.77%	78.49%	45.99%	58.48%	94.60%	73.50%	64.98%
	p=p*	✗	2+5.93	73.62%	62.01%	72.38%	85.69%	59.13%	65.60%	97.00%	80.63%	74.50%
		✓	2+5.75	**73.91%**	62.06%	**73.80%**	**85.88%**	**59.48%**	**66.18%**	**97.30%**	**80.90%**	74.94%
PESF (α=0.3)		✗	2+5.94	73.03%	61.09%	72.16%	84.93%	57.32%	64.78%	97.00%	79.70%	73.76%
		✓	2+5.94	73.23%	61.09%	72.69%	85.05%	57.93%	65.11%	97.00%	79.81%	73.99%

Method		w/ LDA	Avg. k	*Mathematical Reasoning Tasks*			*Code Generation Tasks*			*Instruction-Following Tasks*
				GSM8K	**MATH**	**Average**	**MBPP**	**HumanEval**	**Average**	**IFEval**
Baseline (k=2+6)		✗	2+6.00	87.95%	64.60%	76.28%	69.80%	82.93%	76.37%	42.33%
top-k	k=2+4	✗	2+4.00	83.78%	55.80%	69.79%	65.40%	81.32%	73.36%	41.04%
		✓	2+4.00	86.81%	66.40%	76.61%	68.20%	81.71%	74.96%	40.30%
	k=2+2	✗	2+2.00	49.66%	10.80%	30.23%	17.40%	37.80%	27.60%	25.14%
		✓	2+2.00	79.53%	55.60%	67.57%	62.40%	82.93%	72.67%	38.82%
	k=2+1	✗	2+1.00	00.23%	00.00%	00.12%	00.00%	00.00%	00.00%	07.95%
		✓	2+1.00	33.97%	13.00%	23.49%	29.40%	32.32%	30.86%	23.66%
top-p	p=0.2	✗	2+4.41	79.30%	46.60%	62.95%	54.60%	73.78%	64.19%	38.82%
		✓	2+4.78	81.73%	59.40%	70.57%	63.20%	84.15%	73.68%	41.77%
	p=0.1	✗	2+3.72	00.30%	00.00%	00.15%	00.00%	00.00%	00.00%	14.60%
		✓	2+3.74	59.82%	38.20%	49.01%	46.60%	61.59%	54.10%	31.79%
	p=p*	✗	2+5.94	87.79%	64.00%	75.90%	69.20%	82.93%	76.07%	44.73%
		✓	2+5.80	**88.48%**	**66.20%**	77.34%	**70.20%**	**84.15%**	77.18%	**46.58%**
PESF (α=0.3)		✗	2+5.79	86.80%	61.20%	74.00%	66.20%	84.14%	75.17%	43.99%
		✓	2+5.79	87.03%	63.40%	75.30%	67.00%	85.36%	76.18%	44.91%

Table 7: **Performance across a wide range of downstream tasks under different routing strategies on kanana-2-30b-a3b-instruct.** We use deterministic algorithms for the evaluations. Avg. k denotes the average number of activated experts, and p* denotes the best-performing top-p threshold for each task, reported in Table 9. Results with the highest performance are highlighted in **red**.

top-p Evaluation For top-p routing, we sweep the threshold over $p \in \{0.1, 0.2, \ldots, 0.9\}$. In the main results, p^* denotes the threshold that achieves the best average performance for each task among the evaluated values.

C Statistical Significance Test

We use McNemar's test (McNemar, 1947) for option-based tasks and paired t-test (Student, 1908) for generation-based tasks.

McNemar's Test For option-based tasks, predictions are obtained deterministically from the same set of examples. Thus, we use McNemar's test to compare paired correctness between reduced routing with and without LDA. Let A denote the event that an example is correct only with LDA, and B denote the event that an example is correct only without LDA. The null hypothesis is defined as

$$H_0 : P(A) = P(B).$$

This test examines whether the number of examples corrected only by LDA is statistically different from the number of examples correctly predicted only by the baseline.

Paired t-test For generation-based tasks, we evaluate 5 random seeds under the default generation configuration and compare task-level performance using a paired t-test. Let $X_{\text{w/ LDA}}$ and $X_{\text{w/o LDA}}$ denote the performance under the same evaluation setting, The null hypothesis is defined as

$$H_0 : \mathbb{E}[X_{\text{w/ LDA}} - X_{\text{w/o LDA}}] = 0.$$

This test examines whether the mean performance difference induced by LDA is statistically different from zero.

Method		w/ LDA	Avg. k	*General Knowledge & Commonsense Reasoning Tasks*								
				MMLU	**Hella.**	**Wino.**	**ARC-E**	**ARC-C**	**CQA**	**SciQ**	**PIQA**	**Average**
Baseline (k=1+4)		✗	1+4.00	70.67%	61.06%	73.71%	82.41%	55.46%	72.32%	96.60%	80.09%	74.04%
top-k	k=1+1	✗	1+1.00	36.95%	40.99%	52.80%	51.09%	27.65%	24.41%	76.90%	65.72%	47.06%
		✓	1+1.00	58.08%	48.93%	60.77%	71.00%	40.53%	55.12%	91.30%	73.88%	62.45%
top-p	p=0.1	✗	1+1.42	48.43%	47.76%	57.45%	62.45%	36.51%	35.87%	87.50%	69.96%	55.74%
		✓	1+1.41	64.33%	52.23%	66.38%	77.57%	47.69%	67.56%	95.60%	76.80%	68.52%
PESF (α=0.7)		✗	1+3.56	63.85%	57.60%	70.00%	79.12%	50.08%	64.29%	90.60%	77.09%	69.07%
		✓	1+3.57	64.70%	57.87%	70.71%	80.59%	50.68%	67.89%	93.70%	78.12%	70.53%

Method		w/ LDA	Avg. k	*Mathematical Reasoning Tasks*			*Code Generation Tasks*			*Instruction-Following Tasks*
				GSM8K	**MATH**	**Average**	**MBPP**	**HumanEval**	**Average**	**IFEval**
Baseline (k=1+4)		✗	1+4.00	84.15%	19.40%	51.77%	40.80%	75.61%	58.21%	48.43%
top-k	k=1+1	✗	1+1.00	01.97%	00.00%	0.99%	00.80%	01.22%	02.01%	09.06%
		✓	1+1.00	63.76%	00.00%	31.88%	35.60%	32.93%	34.27%	23.29%
top-p	p=0.1	✗	1+2.09	21.53%	15.20%	18.37%	19.60%	32.92%	26.26%	20.51%
		✓	1+2.07	70.73%	40.80%	55.77%	47.80%	57.32%	52.56%	33.09%
PESF (α=0.7)		✗	1+3.41	83.85%	1.00%	42.43%	46.00%	54.87%	50.44%	35.48%
		✓	1+3.41	83.39%	7.80%	45.60%	52.60%	56.34%	54.57%	37.70%

Table 8: **Performance across a wide range of downstream tasks under different routing strategies on GLM-4.7-Flash.** We use deterministic algorithms for the evaluations. Avg. k denotes the average number of activated experts.

Model	w/ LDA	MMLU	Hella.	Wino.	ARC-E	ARC-C	CQA	SciQ	PIQA	GSM8K	MATH	MBPP	HumanEval	IFEval
Qwen3-30B-A3B	✗	0.8	0.8	0.7	0.6	0.7	0.5	0.7	0.8	0.7	0.8	0.7	0.8	0.4
	✓	0.8	0.8	0.6	0.5	0.5	0.4	0.7	0.7	0.3	0.7	0.7	0.7	0.3
kanana-2-30b-a3b-instruct	✗	0.7	0.7	0.7	0.8	0.7	0.7	0.5	0.5	0.6	0.8	0.5	0.6	0.6
	✓	0.6	0.8	0.3	0.6	0.8	0.8	0.4	0.6	0.4	0.4	0.5	0.6	0.5

Table 9: **Selected p* values across models and downstream tasks.** p* denotes the best-performing threshold among $p \in \{0.1, 0.2, \ldots, 0.9\}$ under top-p routing.

D Performance on Other Models

We provide additional results on kanana-2-30b-a3b-instruct and GLM-4.7-Flash to examine whether the effectiveness of LDA is limited to Qwen3-30B-A3B. As described in Appendix B.2, these models differ from Qwen3-30B-A3B in their expert configurations, use of shared experts, and gating functions. Overall, the results show a similar tendency to the main results in Section 5.2: reduced expert activation degrades vanilla routing performance, while LDA mitigates this degradation under the different routing strategies.

D.1 kanana-2-30b-a3b-instruct

Table 7 reports the results on kanana-2-30b-a3b-instruct. Under fixed top-k routing, performance decreases as the number of activated experts is reduced, and the degradation becomes more pronounced in smaller k settings. However, applying LDA consistently improves performance over the corresponding reduced top-k baseline across all task categories.

Under the top-p routing, lowering the threshold p reduces the average number of activated experts and degrades vanilla performance, whereas LDA improves performance under the same top-p setting. In the top-p* setting, LDA achieves higher performance than the default top-k_0 routing baseline across diverse tasks, indicating that LDA also remains effective under dynamic routing.

We also observe consistent improvements when LDA is combined with PESF. This suggests that LDA is not limited to fixed reduced top-k or dynamic top-p routing, but can also mitigate performance degradation under dynamic expert pruning.

D.2 GLM-4.7-Flash

Table 8 reports the results on GLM-4.7-Flash. We observe a similar pattern to the other models. Under aggressive reduced top-k routing, vanilla performance drops severely, whereas LDA substantially mitigates this degradation under the same expert budget. A similar trend is observed under dynamic top-p routing and PESF, where LDA improves performance over the corresponding routing baseline

Method	MMLU			GSM8K			MBPP		
	k=6	k=4	k=2	k=6	k=4	k=2	k=6	k=4	k=2
top-k	76.92%	70.72%	27.97%	86.81%	74.15%	00.99%	68.20%	34.80%	00.00%
top-k w/ ECS	72.72%	24.69%	24.56%	77.71%	01.06%	00.00%	54.60%	00.00%	00.00%
top-k w/ RMSS	77.30%	75.33%	65.17%	87.94%	86.65%	66.11%	**70.60%**	67.60%	**44.00%**
top-k w/ LDA	**77.34%**	**75.56%**	**67.40%**	**88.63%**	**87.49%**	**68.54%**	**70.60%**	**68.00%**	**44.00%**

Table 10: **Comparison with simple scaling baselines on Qwen3-30B-A3B.** ECS, RMSS, and LDA denote Expert-Count Scaling, RMS Scaling, and Layer-wise Distribution Alignment, respectively. We use deterministic algorithms for the evaluations. LDA generally outperforms simple scaling baselines under reduced top-k routing.

in most task categories.

These additional results suggest that LDA is not restricted to a single SMoE architecture. Across different model architectures and routing strategies, LDA consistently improves reduced-routing performance, supporting the importance of maintaining representation scale and distribution consistency under reduced expert activation.

E Comparison with Simple Scaling Baselines

To examine whether the effect of LDA can be explained solely by scalar rescaling of SMoE outputs, we compare LDA with two simple scaling baselines: Expert-Count Scaling and RMS Scaling. These baselines are training-free and applied at inference time, similar to LDA. However, unlike LDA, they apply a single scalar correction to the entire SMoE output and do not align per-dimension distribution statistics. Thus, this comparison allows us to examine whether global scale correction is sufficient, or whether per-dimension distribution alignment is necessary for preserving downstream performance under reduced expert activation.

Expert-Count Scaling Expert-Count Scaling is a naive scaling baseline that rescales the SMoE output according to the ratio between the default and reduced numbers of activated experts. Given the SMoE output $\mathbf{y}^{(l)}$ at layer l, it applies

$$\hat{\mathbf{y}}^{(l)} = \mathbf{y}^{(l)} \cdot \frac{k_0}{k},$$

where k_0 is the default number of activated experts and k is the reduced number of activated experts. This baseline tests whether performance degradation under reduced expert activation can be mitigated by a simple multiplier based only on the expert count.

RMS Scaling RMS Scaling rescales the SMoE output using the average RMS scale estimated for each layer and top-k setting. Given the SMoE output $\mathbf{y}^{(l)}$ at layer l, it applies

$$\hat{\mathbf{y}}^{(l)} = \mathbf{y}^{(l)} \cdot \frac{\rho_{k_0}^{(l)}}{\rho_k^{(l)}},$$

where $\rho_k^{(l)} = \mathbb{E}[\mathrm{RMS}(\mathbf{y}_k^{(l)})]$ denotes the average RMS of SMoE outputs at layer l under top-k routing. Unlike Expert-Count Scaling, RMS Scaling uses calibration statistics to directly correct the average output scale. However, it still applies a single scalar correction and does not account for dimension-wise distributional mismatch.

Table 10 compares LDA with simple scaling baselines. Expert-Count Scaling leads to substantial performance degradation compared to the vanilla reduced top-k baseline. This indicates that the performance drop under reduced expert activation cannot be addressed by a naive expert-count-based multiplier. Since routing scores are re-normalized, the effect of reducing k is not simply proportional to the number of activated experts.

RMS Scaling provides a stronger baseline. Compared to the vanilla reduced top-k baseline, RMS Scaling substantially mitigates performance degradation across tasks, especially in low-k settings. This result supports our analysis that the scale of SMoE outputs plays an important role in maintaining downstream performance under reduced expert activation.

However, LDA generally achieves higher performance than the simple scaling baselines. Unlike Expert-Count Scaling and RMS Scaling which apply a single scalar correction to the entire SMoE output, LDA aligns the per-dimension mean and standard deviation of reduced top-k representations to those of the default top-k_0 configuration. The consistent improvement of LDA over the simple

scaling baselines suggests that the mismatch induced by reduced expert activation is not only a global scale shift, but also involves dimension-wise distributional changes. These results empirically provide additional evidence that per-dimension distribution alignment is more effective than global scaling for mitigating routing-induced representation distribution mismatch.

F Ablation Study

LDA estimates layer-wise distribution statistics from a calibration dataset and uses them to align SMoE outputs under reduced expert activation. Since the correction is determined by calibration statistics, its effectiveness may depend on the choice of calibration dataset and the number of calibration samples. We therefore conduct ablation studies to examine the robustness of LDA with respect to these factors.

However, our calibration procedure relies on activation-level statistics of SMoE outputs, similar to post-training compression methods. Prior work on activation-aware quantization and activation-based pruning has shown that activation-based statistics can be estimated effectively from relatively small calibration sets (Sun et al., 2024) and are often robust across calibration data choices (Lin et al., 2024). This suggests that the role of calibration data is primarily to provide stable estimates of the general representation distribution, rather than to encode task-specific knowledge.

Nevertheless, because LDA explicitly uses calibration statistics for inference-time correction, we empirically examine whether calibration dataset choice and calibration sample size affect downstream performance. We first compare calibration datasets from different domains and then vary the number of calibration samples used to estimate the layer-wise statistics. These ablations provide additional evidence that LDA remains stable across calibration settings.

F.1 Effect of Calibration Dataset Choice

We first examine the effect of calibration dataset choice. Table 11 compares downstream performance when the layer-wise distribution statistics are estimated from different calibration datasets, including C4, GSM8K, MATH, and MBPP. All settings use the same number of calibration tokens, and apply LDA under reduced top-k routing.

Across different calibration datasets, the performance remains stable. For MMLU, GSM8K, and MBPP, calibration with C4 achieves performance comparable to calibration with task-specific or domain-specific datasets. Although small variations appear across tasks and top-k settings, no calibration dataset consistently outperforms the others. These results indicate that using a dataset from the same downstream domain does not necessarily yield better performance.

This observation supports the role of calibration data in LDA. The calibration dataset is not used to inject task-specific knowledge or adapt the model to a particular domain. Instead, it is used to estimate layer-wise representation statistics under different routing configurations. Therefore, even when calibration datasets differ in domain, they can provide sufficiently stable estimates of the general representation distribution of SMoE outputs.

Calibration Dataset	MMLU		GSM8K		MBPP	
	k=6	k=4	k=6	k=4	k=6	k=4
C4	77.34%	75.56%	88.63%	87.49%	70.60%	68.00%
GSM8K	77.28%	75.52%	88.10%	87.41%	70.60%	68.40%
MATH	77.32%	75.59%	88.09%	87.26%	70.00%	67.80%
MBPP	77.35%	75.57%	88.09%	86.65%	71.00%	67.60%

Table 11: **Effect of calibration dataset choice on Qwen3-30B-A3B.** All settings use top-k routing with LDA and 4×2,048 calibration tokens for layer-wise distribution estimation. We use deterministic algorithms for the evaluations.

# of tokens	MMLU		GSM8K		MBPP	
	k=6	k=4	k=6	k=4	k=6	k=4
1×2048	77.39%	75.67%	88.32%	87.02%	70.00%	67.60%
2×2048	77.29%	75.69%	88.55%	87.26%	70.40%	67.00%
4×2048	77.34%	75.56%	88.63%	87.49%	70.60%	68.00%
8×2048	77.35%	75.52%	88.63%	87.71%	70.60%	67.00%

Table 12: **Effect of calibration sample size on Qwen3-30B-A3B.** All settings use top-k routing with LDA and C4 as the calibration dataset. We use deterministic algorithms for the evaluations.

F.2 Effect of Calibration Sample Size

We next examine the effect of the number of calibration samples. Table 12 compares downstream performance when varying the number of calibration tokens used to estimate the layer-wise distribution statistics. We use C4 as the calibration dataset and apply LDA under reduced top-k routing.

Across different calibration sample sizes, the performance remains relatively stable. Increasing the number of calibration tokens does not lead to consistent performance improvements, and even a small number of calibration samples provides

competitive results. This indicates that the layer-wise mean and standard deviation of SMoE outputs can be estimated sufficiently well from a small calibration set. In addition, the lower performance variation across calibration sample sizes further suggests that LDA is not overly sensitive to a particular calibration subset and can be applied with a small calibration cost.

G Algorithms

We provide pseudo-code descriptions of our method. Algorithm 1 describes the calibration procedure, where layer-wise distribution statistics are estimated for each top-k setting using an MoE model and a calibration dataset. Algorithm 2 describes the inference procedure, where the estimated layer-wise statistics are applied to align SMoE outputs under reduced expert activation.

H Additional Plots

We provide additional analysis figures for Qwen3-30B-A3B. Figure 7 shows the layer-wise distributions of SMoE outputs under different top-k settings. Figure 8 shows the corresponding distributions after applying LDA. These figures provide additional evidence that reduced top-k changes the SMoE output distribution across all layers, while LDA aligns it toward the default top-k_0 configuration.

AI Assistant Usage Statement

AI assistants were used solely for language polishing, grammar correction, and improving clarity of presentation. All technical content, experimental design, and analysis are entirely our own.

Algorithm 1 Layer-wise Distribution Estimation

Input: calibration dataset $\mathcal{D}$, SMoE model $\mathcal{M}$
Output: layer-wise distribution statistics $\mathcal{S}$

$L \leftarrow$ number of layers in $\mathcal{M}$
$k_0 \leftarrow$ default top-k routing value of $\mathcal{M}$
$\mathcal{K} \leftarrow \{1, 2, \ldots, k_0\}$

▷ **Step 1: Sample calibration data**
Sample a calibration mini-batch $X \sim \mathcal{D}$

▷ **Step 2: Estimate layer-wise distributions**
$H^{(0)} \leftarrow \mathrm{Embed}_{\mathcal{M}}(X)$
for $l = 0, 1, \ldots, L-1$ **do**
  if $\mathrm{SMoE} \in \mathrm{Layer}_{\mathcal{M}}^{(l)}$ **then**
    ▷ **Estimate the statistics for each top-k setting**
    for $k \in \mathcal{K}$ **do**
      $Y_k^{(l)} \leftarrow \mathrm{SMoEOutput}_{\mathcal{M}}^{(l)}(H^{(l)}; k)$
      $\mu_k^{(l)} \leftarrow \mathrm{Mean}_{\text{token-wise}}\left(Y_k^{(l)}\right)$ ▷ Per-dimension mean
      $\sigma_k^{(l)} \leftarrow \mathrm{Std}_{\text{token-wise}}\left(Y_k^{(l)}\right)$ ▷ Per-dimension standard deviation
      Store $\{\mu_k^{(l)}, \sigma_k^{(l)}\}$ in $\mathcal{S}$
    end for
  end if
  ▷ **Propagate with default top-k_0 routing**
  $H^{(l+1)} \leftarrow \mathrm{Layer}_{\mathcal{M}}^{(l)}(H^{(l)}; k_0)$
end for

Return $\mathcal{S}$

Algorithm 2 Distribution-Consistent Inference

Input: input data X, distribution statistics $\mathcal{S}$, SMoE model $\mathcal{M}$, routing value k
Output: model output $\hat{X}$

$L \leftarrow$ number of layers in $\mathcal{M}$
$k_0 \leftarrow$ default top-k routing value of $\mathcal{M}$

$H^{(0)} \leftarrow \mathrm{Embed}_{\mathcal{M}}(X)$
for $l = 0, 1, \ldots, L-1$ **do**
  ▷ **Propagate to the next layer**
  if $\mathrm{SMoE} \in \mathrm{Layer}_{\mathcal{M}}^{(l)}$ **then**
    ▷ **Forward with top-k routing**
    $Y_k^{(l)} \leftarrow \mathrm{SMoEOutput}_{\mathcal{M}}^{(l)}(H^{(l)}; k)$
    if $k < k_0$ **then**
      ▷ **Apply layer-wise distribution alignment**
      $\{\mu_{k_0}^{(l)}, \sigma_{k_0}^{(l)}\} \leftarrow \mathcal{S}[l][k_0]$ ▷ Reference statistics
      $\{\mu_k^{(l)}, \sigma_k^{(l)}\} \leftarrow \mathcal{S}[l][k]$ ▷ Target statistics
      $\hat{Y}^{(l)} \leftarrow \sigma_{k_0}^{(l)} \odot \dfrac{Y_k^{(l)} - \mu_k^{(l)}}{\sigma_k^{(l)} + \epsilon} + \mu_{k_0}^{(l)}$ ▷ Per-dimension correction
    else
      $\hat{Y}^{(l)} \leftarrow Y_k^{(l)}$
    end if
    $H^{(l+1)} \leftarrow \mathrm{Layer}_{\mathcal{M}}^{(l)}(H^{(l)}; k, \hat{Y}^{(l)})$
  else
    $H^{(l+1)} \leftarrow \mathrm{Layer}_{\mathcal{M}}^{(l)}(H^{(l)})$
  end if
end for

$\hat{X} \leftarrow \mathrm{Output}_{\mathcal{M}}(H^{(L)})$
Return $\hat{X}$

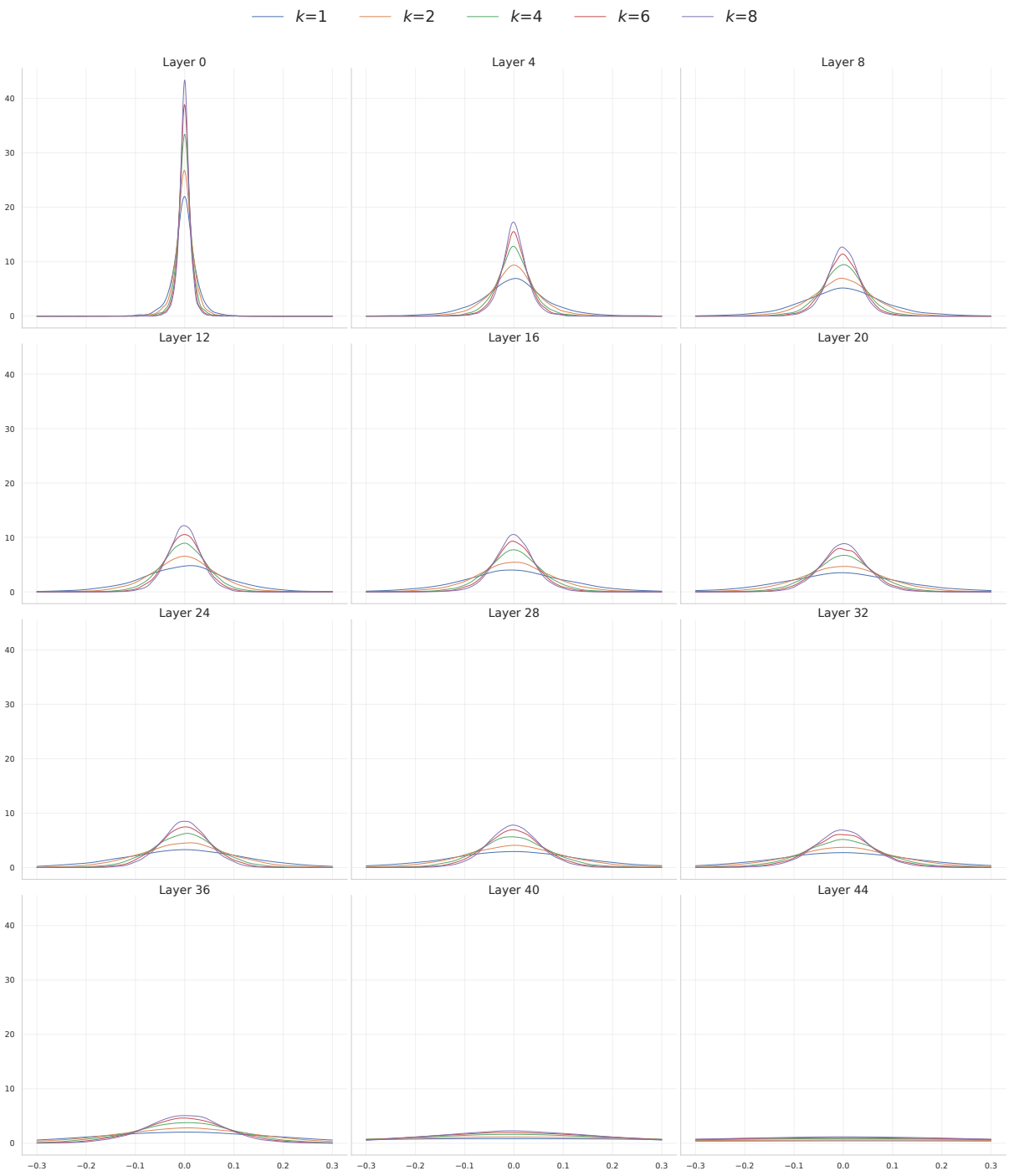


Figure 7: **Layer-wise distributions of SMoE outputs under different top-k settings before applying LDA on Qwen3-30B-A3B.** We use 2,048 held-out calibration tokens and sample 100,000 values. The distributions become more dispersed under smaller k across layers, consistent with Fig. 3a.

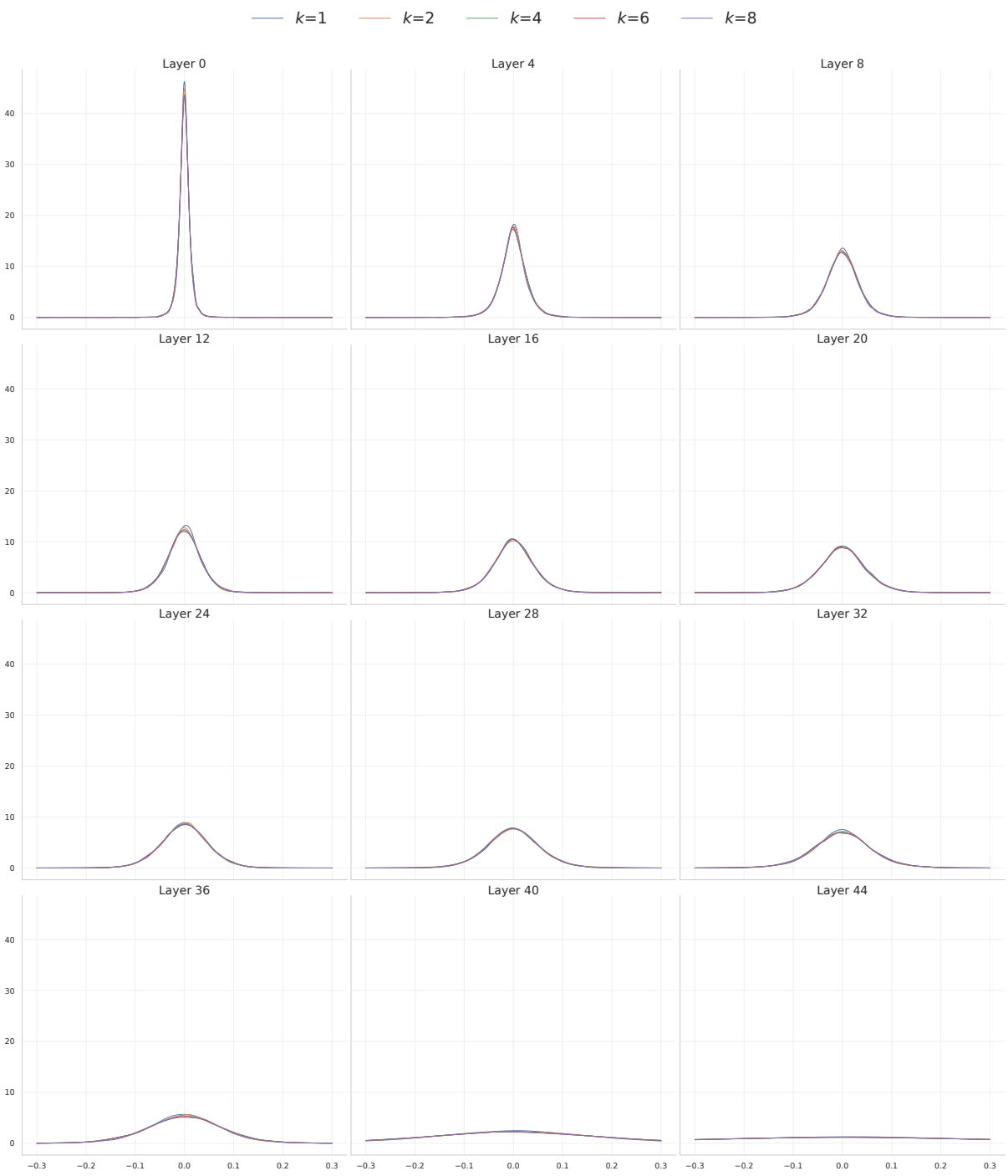


Figure 8: **Layer-wise distributions of SMoE outputs under different top-k settings after applying LDA on Qwen3-30B-A3B.** We use 2,048 held-out calibration tokens and sample 100,000 values. Compared with Fig. 7, the layer-wise distributions become more closely aligned with the default top-k_0 setting.

Artifact	Description	License
MMLU	General knowledge and reasoning benchmark; English; auxiliary_train/dev/val/test = 99,842/285/1,531/14,042 samples.	MIT Lincense
HellaSwag	Commonsense reasoning benchmark; English; train/val/test = 39,905/10,042/10,003 samples.	MIT Lincense
WinoGrande	Commonsense coreference reasoning benchmark; English; train/val/test = 40,398/1,267/1,767 samples.	CC-BY Lincense
ARC-Easy	Science question answering benchmark; English; train/val/test = 2,251/570/2,376 samples.	CC BY-SA 4.0 Lincense
ARC-Challenge	Challenging science question answering benchmark; English; train/val/test = 1,119/299/1,172 samples.	CC BY-SA 4.0 Lincense
CommonsenseQA	Commonsense question answering benchmark; English; train/val/test = 9,741/1,221/1,140 samples.	MIT Lincense
SciQ	Science question answering benchmark; English; train/val/test = 11,679/1,000/1,000 samples.	CC BY-NC 3.0 Lincense
PIQA	Physical commonsense reasoning benchmark; English; train/val/test = 16,113/1,838/3,084 samples.	Academic Free License v3.0
GSM8K	Grade-school mathematical reasoning benchmark; English; train/test = 7,473/1,319 samples.	MIT Lincense
MATH500	Competition-level mathematical reasoning benchmark; English; 500 samples.	MIT Lincense
MBPP	Python code generation benchmark from natural language descriptions; English/Python; train/val/test = 374/90/500 samples.	CC BY 4.0 Lincense
HumanEval	Python code generation benchmark; English/Python; 164 samples.	MIT Lincense
IFEval	Instruction-following benchmark with verifiable constraints; English; 541 samples.	Apache 2.0 Lincense
Qwen3-30B-A3B	Sparse MoE LLM used for evaluation.	Apache 2.0 Lincense
kanana-2-30b-a3b-instruct	Sparse MoE LLM used for evaluation.	Kanana License Agreement
GLM-4.7-Flash	Sparse MoE LLM used for evaluation.	MIT License
lm_eval	Evaluation framework for language model benchmarks.	MIT Lincense
vLLM	Inference framework for efficient LLM serving and evaluation.	Apache 2.0 Lincense

Table 13: **Documentation of experimental artifacts.** We report summary of the datasets, models, and frameworks used in our experiments, including descriptions, split sizes, and licenses.