

Differentiable Routing

Probabilistic Inference

Sparse MoE

tl;dr

“We cast MoE routing as probabilistic inference over cardinality-constrained expert subsets, yielding exact marginal-based router gradients that improve expert utilization and diversity, with a natural extension to adaptive per-token expert budgets.”

Why

- MoE scales by activating only a small subset of experts per token.
- Training still depends on hard **top- k routing**, a discrete operator with no useful gradient through the selection event.
- Standard training updates selected experts but gives the router little signal about **alternative expert subsets**.

$$\frac{\partial y(x; r)}{\partial r_i} = \underbrace{\sum_{\substack{S \subseteq [M] \\ |S|=k}} \mathbb{I}[S = S_{\text{top-}k}(r)] \frac{\partial y_S(x; r)}{\partial r_i}}_{\text{softmax-weight path}} + \underbrace{\sum_{\substack{S \subseteq [M] \\ |S|=k}} y_S(x; r) \frac{\partial \mathbb{I}[S = S_{\text{top-}k}(r)]}{\partial r_i}}_{\text{discrete-selection path (ignored!)}}$$

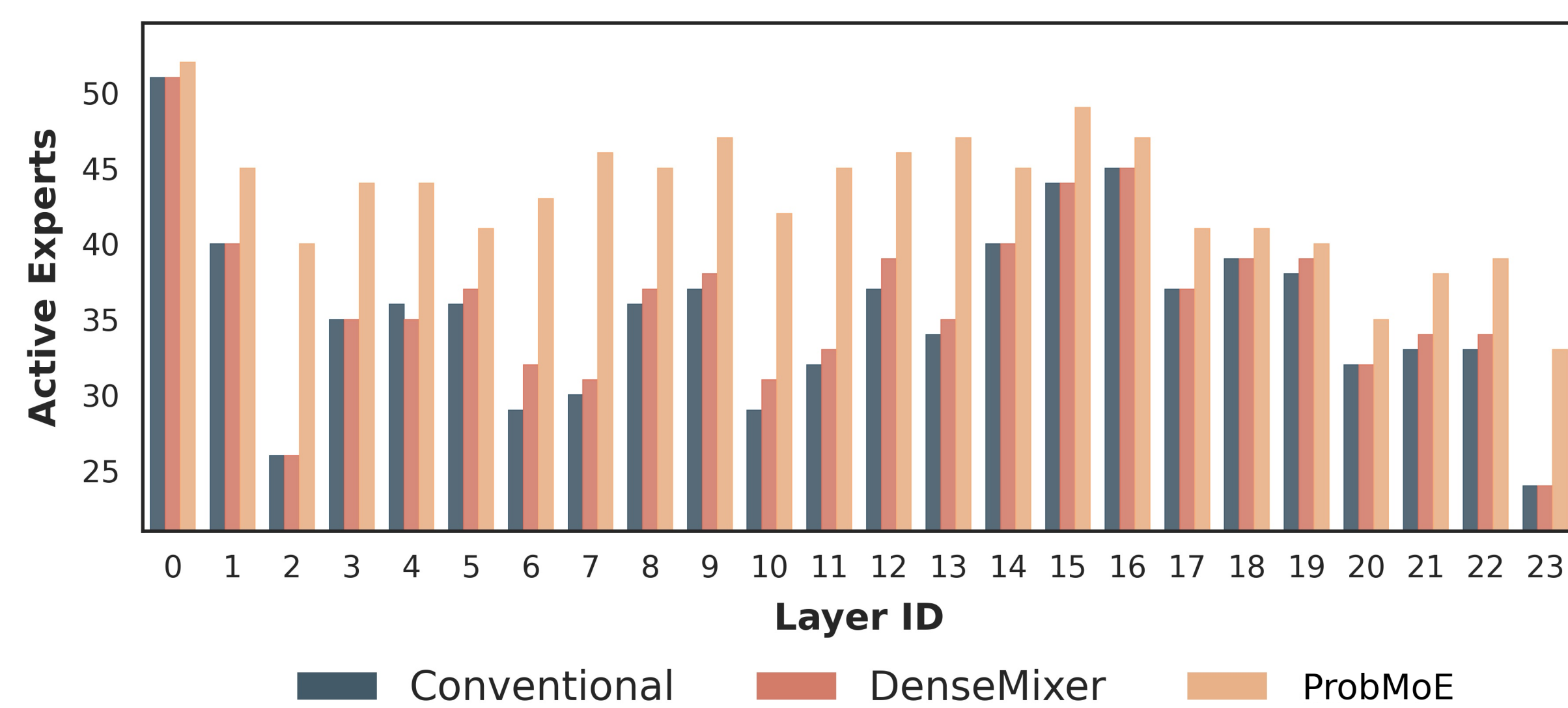
Existing Methods

Conventional MoE treats $S_{\text{top-}k}(r)$ as fixed in the backward pass. Router gradients therefore flow only through the softmax weights of the selected experts:

$$\frac{\partial \mathcal{L}}{\partial r_i} = \sum_{j \in S_{\text{top-}k}} \left\langle \frac{\partial \mathcal{L}}{\partial y_j}, f_j(x) \right\rangle \frac{\partial \pi_j}{\partial r_i}$$

This gives no direct gradient for the discrete selection mechanism. Dense straight-through estimators add proxy gradients for expert scores, but the learning signal remains tied to deterministic top- k routing instead of a distribution over expert subsets.

- Hard top- k encourages concentrated routing patterns.
- Alternative expert subsets receive little exploration signal.
- Expert under-utilization can persist even with dense STE proxies.



ProbMoE activates more experts across Qwen layers.

ProbMoE Solution

Rather than treating expert selection as a fixed discrete decision in the backward pass, ProbMoE considers the distribution over feasible expert subsets and trains the router by optimizing the expected loss under this distribution, defined as below:

$$\mathcal{J}(\theta) = \mathbb{E}_{S \sim \mathbb{P}_r(\cdot | |S|=k)} [\mathcal{L}(y_S(x; r))],$$

This objective optimizes the router over a distribution of feasible expert subsets, allowing alternative expert combinations to influence training.

Exact- k Subset Distribution

Let $p_i = \sigma(r_i)$ denote the Bernoulli selection probability for expert i . These independent Bernoulli variables induce an unconstrained product distribution over expert subsets. ProbMoE conditions this distribution on selecting exactly k experts:

$$\mathbb{P}_r(S \mid |S| = k) = \frac{1}{Z_k} \prod_{i \in S} p_i \prod_{j \notin S} (1 - p_j), \quad |S| = k.$$

Forward and Backward

Forward. ProbMoE samples an expert subset from the exact- k distribution:

$$S \sim \mathbb{P}_r(S \mid |S| = k).$$

The MoE layer computes the subset-specific output using the sampled mask.

Backward. Let $z_i \in \{0, 1\}$ indicate whether expert i is selected, and let $m_i = \mathbb{P}_r(z_i = 1 \mid |S| = k)$ be its exact marginal. ProbMoE combines the sampled forward mask, exact marginals, and soft routing weights with a straight-through estimator:

$$w = (\text{stopgrad}(z - m) + m) \odot \pi.$$

Thus the forward pass stays sparse, while router gradients are driven by marginal probabilities over feasible expert subsets.

Dynamic- k Routing

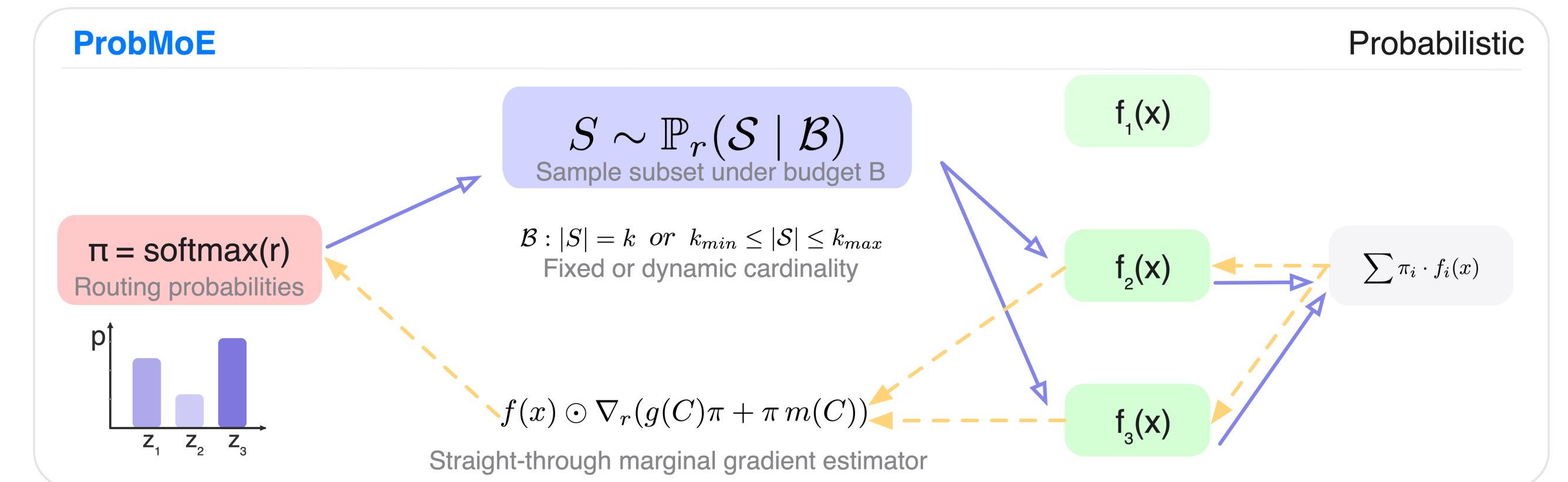
Because ProbMoE is defined as a probabilistic distribution over expert subsets, it naturally extends from fixed-cardinality routing to adaptive compute. Dynamic- k relaxes the exact- k constraint and instead conditions on a range of feasible subset sizes:

$$k_{\min} \leq |S| \leq k_{\max}.$$

This yields a range-constrained expert subset distribution:

$$\mathbb{P}_r(S \mid k_{\min} \leq |S| \leq k_{\max}) = \frac{1}{Z^*} \prod_{i \in S} p_i \prod_{j \notin S} (1 - p_j).$$

The router can therefore choose fewer or more experts per token while preserving the same marginal-based training principle.



Results

Table 1. Performance comparison across benchmarks. Top-3 are highlighted; darker colors indicate better performance.

Model	GSM	Law	Translation	MBPP	Summary	MMLU	Overall MMLU	Stem
OLMoE Backbone (Top-$k = 8$)								
Base Model	15.85	5.70	11.09	19.80	7.40	53.40		44.91
Frozen Router	44.88	22.50	28.29	17.80	36.05	53.97		46.18
Conventional	45.94	25.00	27.56	23.20	33.70	54.04		46.46
DenseMixer	47.00	27.90	30.32	24.40	37.50	53.95		46.18
ProbMoE	50.19	29.00	31.63	22.80	39.29	53.69		45.54
Qwen backbone (Top-$k = 4$)								
Base Model	38.69	18.20	16.53	38.84	28.29	60.85		52.27
Frozen Router	53.37	33.01	32.75	35.20	38.29	61.08		53.19
Conventional	53.30	29.50	30.00	32.80	39.00	61.03		53.16
DefaultMoE	51.00	31.20	24.00	33.20	38.40	—		—
SparseMixer	1.30	3.40	3.50	0.00	2.10	—		—
ReMoE	46.30	25.50	16.99	33.00	25.80	—		—
DenseMixer	54.97	30.75	33.75	34.00	41.00	61.03		52.87
ProbMoE	53.29	34.40	39.23	35.00	44.40	61.05		53.82

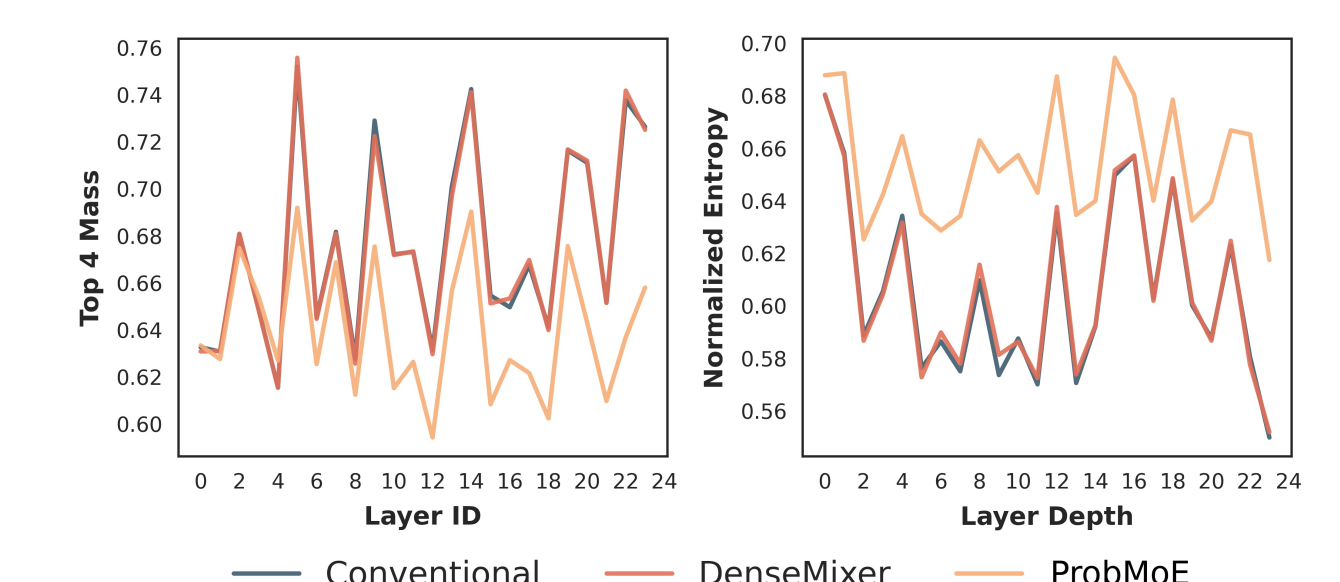
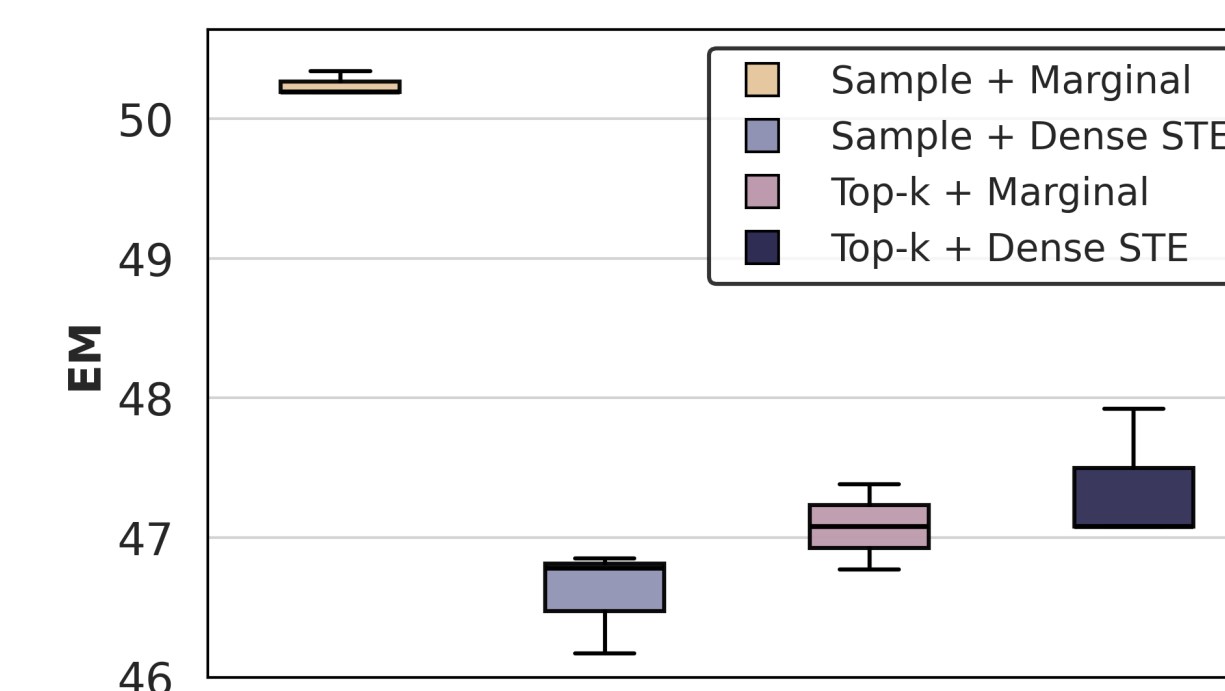
GSM: exact match; Law, Translation, Summary: LLM-as-judge; MBPP and MMLU: LM Evaluation Harness.

Table 2. Dynamic- k vs. Exact- k .

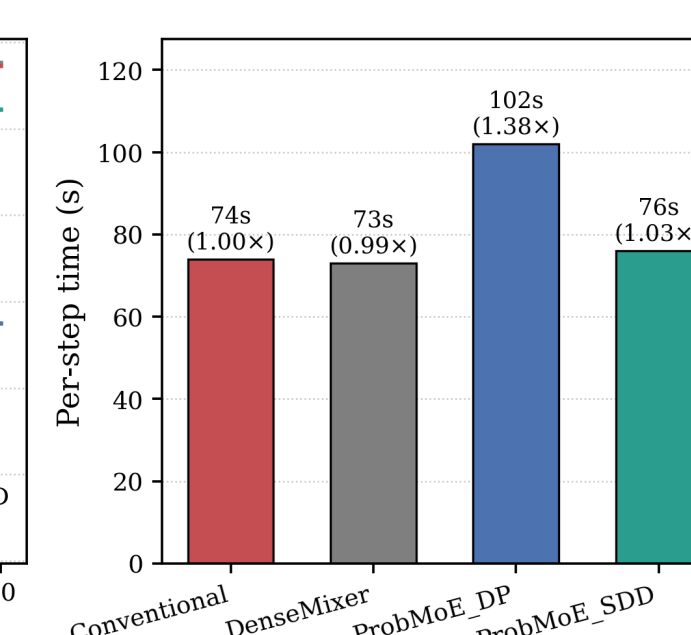
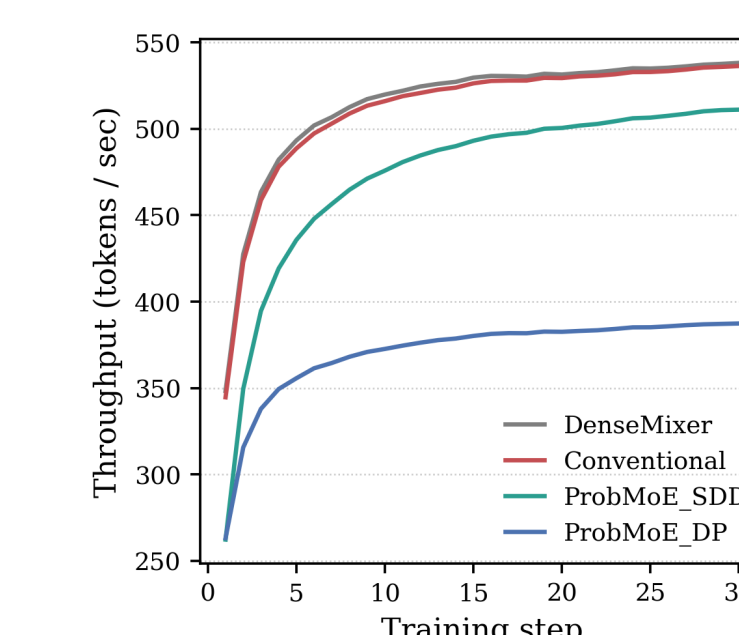
Dataset	OLMoE	Qwen1.5
GSM	-1.82 (80.00%)	-4.29 (75.00%)
Law	-0.04 (84.50%)	+2.70 (75.00%)
Translation	+0.36 (82.00%)	+3.22 (75.00%)

Table 3. Bias, variance, and gradient error.

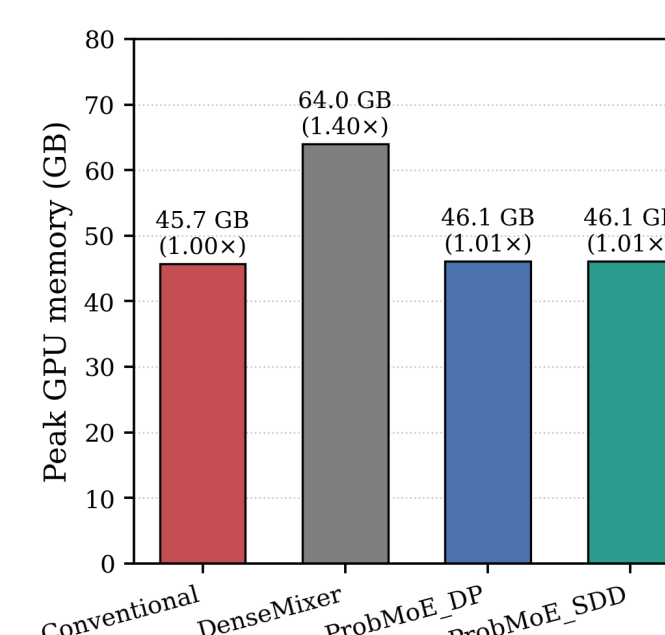
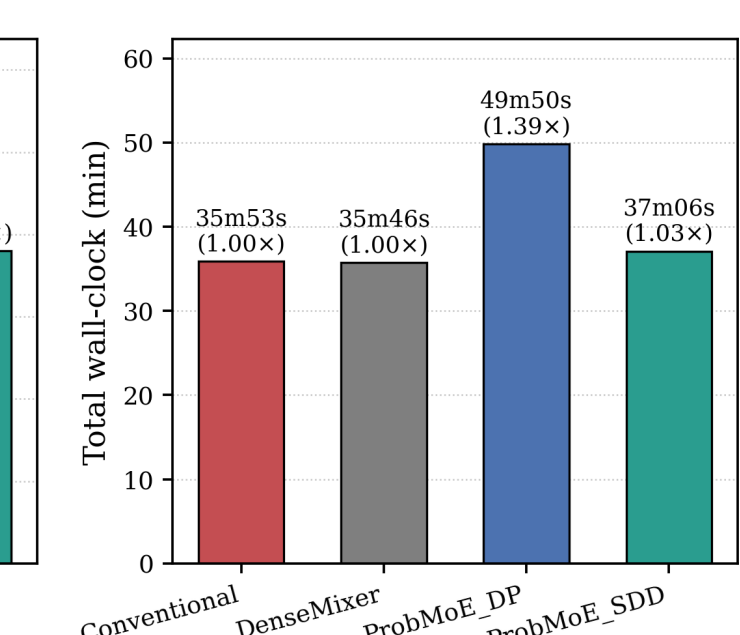
Model	Bias	Variance	Error
ProbMoE	0.0702 ± 0.0426	0.0076 ± 0.0110	0.0967 ± 0.0659
DenseMixer	0.1052 ± 0.0839	0.0201 ± 0.0427	0.1349 ± 0.0995



Ablation over routing distribution shape.



Routing concentration and entropy.



Wall-clock comparison on GSM.

Key takeaway

ProbMoE reframes MoE routing as probabilistic inference over expert subsets, replacing hard Top- K selection with a principled routing distribution that gives more informative router gradients, improves expert utilization, and naturally extends to Dynamic- K routing for adaptive sparse computation.