QUAL-MOE: PRACTICAL DYNAMIC LATENT MIXTURE-OF-EXPERTS WITH QUANTILE BALANCING

Anonymous authors
Paper under double-blind review

ABSTRACT

Fixed Top-k routing assigns every token in a mixture-of-experts (MoE) model the same number of experts, regardless of how much computation it needs. Dynamic routers removes this restriction, but they typically control the average budget only indirectly and handle load balancing separately. We present QUAL-MOE, a dynamic MoE router that uses expert-specific quantile thresholds to determine which tokens each expert accepts. This naturally balances expert loads while keeping the average number of activated experts close to a target, yet allowing the number of experts to vary across tokens. To make this routing practical at scale, we introduce *global normalization*, which uses one normalized score for both expert selection and output weighting, allowing task-loss gradients to directly shape routing decisions. We also develop distributed estimators that aggregate the thresholds across training ranks. A single inference-time coefficient scales the thresholds, letting one trained model trade quality for computation without retraining. At the 1.2B scale, QUAL-MOE outperforms Top-p and DynMoE while keeping the average expert count closer to its target. In 10B-parameter pretraining, it surpasses an auxiliary-loss-balanced Top-k model while activating 11.8% fewer experts at inference. Our analysis shows that allocation varies across task domains, declines with token position during both prefill and decoding, and increases with the number of reasoning steps in synthetic tasks. Finally, combining QUAL-MOE with latent experts makes allocation finer-grained: QUAL-MOE-Latent outperforms an architecture-matched Top-k-Latent model throughout training.

1 INTRODUCTION

Sparse mixture-of-experts (MoE) models scale language-model capacity by activating a subset of experts for each token (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022). Top-k routing fixes the size of this subset: the router chooses which experts process a token, but not how many. This keeps computation predictable, yet spends the same budget on every token, be it a punctuation mark or a key step in a derivation. Dynamic routing resolves this mismatch by letting the number of activated experts vary across tokens (Huang et al., 2024; Guo et al., 2025; Wen et al., 2025).

A dynamic router for large-scale pretraining must satisfy three requirements at once: token-level flexibility in the number of experts, a controlled average budget, and balanced expert load. Existing dynamic routers provide the first, but control the budget only indirectly and rely on separate mechanisms for load balance (Huang et al., 2024; Guo et al., 2025; Zeng et al., 2024; Jin et al., 2026); their realized cost must therefore be tuned in advance, and it drifts as training progresses.

We show that a single rule satisfies all three. Top-k routing solves an assignment problem with two constraints: each token receives exactly k experts, and each expert receives the same number of tokens. Load balance requires only the second. We drop the first and let each expert admit the tokens whose scores exceed its own quantile threshold. Expert loads are then balanced by construction, the average expert count matches the target k, and each token uses as many experts as its own routing scores warrant. We call the resulting dynamic router QUAL-MOE (Figure 1).

QUAL-MOE makes this rule practical for large-scale training. Threshold selection is discrete, so the router scores of unselected experts receive no task-loss signal; *global normalization* normalizes

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

scores across all experts and uses the same score for selection and output weighting, letting task gradients flow through every router score. Three distributed estimators compute the thresholds across training ranks. Because the thresholds are explicit, a single inference-time coefficient λ adjusts the expert budget of a trained model without retraining. Narrower latent experts (Elango et al., 2026) further make each activation cheaper, so that per-token computation can change in finer steps.

Empirically, we first compare QUAL-MOE with dynamic routers at the 1.2B scale. Across varying expert granularities, it outperforms Top-p and DynMoE while holding its target expert count, whereas Top-p drifts above budget and DynMoE collapses to about one expert per token. At the 10B scale, pretrained on 840B tokens, QUAL-MOE surpasses a strong Top-k baseline with slightly fewer experts per token, and rescaling its thresholds at inference cuts expert use by over 11.8% while still outperforming Top-k. Finally, with latent experts, QUAL-MOE-Latent outperforms the same-architecture Top-k model throughout training, gaining more from finer experts than Top-k.

The learned allocation also reveals where the model spends its computation. Allocation differs across domains, with Arabic-MMLU drawing 44% more experts than MMLU. It declines with token position in both prefill and decoding, so the savings grow with sequence length, which suits long-context and multi-turn inference. Within a task, it rises with the number of reasoning steps in synthetic BBH problems, rather than with the annotated difficulty levels of MATH problems.

In summary, we make the following contributions:

- We propose Qual-MoE, which dynamically route experts with quantile balancing in both full-width and fine-grained setting. At 10B scale, it outperforms Top-k while activating 11.8% fewer experts.
- We improve Qual-MoE with global normalization, which closes the gap between expert selection and training object; and threshold estimation approaches, to accurately control target expert k.
- We propose threshold control of QUAL-MOE at inference time, which offer flexible control over quality and computational cost without re-training.
- We comprehensively analyze expert allocation and find that it varies across domains, declines with position, and grows with the number of reasoning steps rather than with annotated difficulty.

2 PRELIMINARY BACKGROUND

Quantile Balancing. Consider an MoE layer with n routed experts and a batch of m tokens. Let $s_{i,j}$ denote the routing affinity between token i and expert j, and $x_{i,j} \in \{0,1\}$ indicate their assignment. For fixed affinities, the ideal balanced Top-k assignment solves the integer program

$$\max_{x_{i,j}\in\{0,1\}} \sum_{i,j} x_{i,j}s_{i,j} \quad \text{s.t.} \quad \sum_{j} x_{i,j} = k,\ \forall i, \qquad \sum_{i} x_{i,j} = q,\ \forall j \qquad q = \frac{mk}{n}. \tag{1}$$

The first constraint fixes the allocation to each token; the second balances the load across experts.

Quantile Balancing (QB) (Kimi Team, 2026) targets the expert-load constraint through expert-specific routing biases. It performs token-wise Top-k selection on biased scores and estimates the biases from quantiles of score margins relative to token-side selection cutoffs. This replaces fixed-step load corrections with a distribution-based update, without introducing an auxiliary balancing loss. However, every token still activates exactly k experts, so per-token computation stays fixed.

LatentMoE. Latent MoE separates the residual width from the width processed by each routed expert (Liu et al., 2025; Elango et al., 2026). For compression factor $\alpha > 1$, shared projections $W_\downarrow$ and $W_\uparrow$ map a token to d/α dimensions and return the mixture to the residual stream:

$$u_t = W_\downarrow h_t, \quad y_t^{\text{lat}} = W_\uparrow \mathcal{R}(\gamma \sum_{e\in S_t} r_{te} E_e^{\text{lat}}(u_t)), \tag{2}$$

where $\mathcal{R}$ is an optional RMS normalization (Zhang & Sennrich, 2019) applied after the expert mixture, as in Kimi K3 (Kimi Team, 2026). Any shared full-width expert is evaluated outside this branch. Narrowing routed experts by α ($\alpha = 2$ in our experiments, doubling 128 experts to 256) permits approximately αn experts at the same leading expert-parameter count. This activates αk experts and preserves the leading routed work while making each routed unit α times narrower.

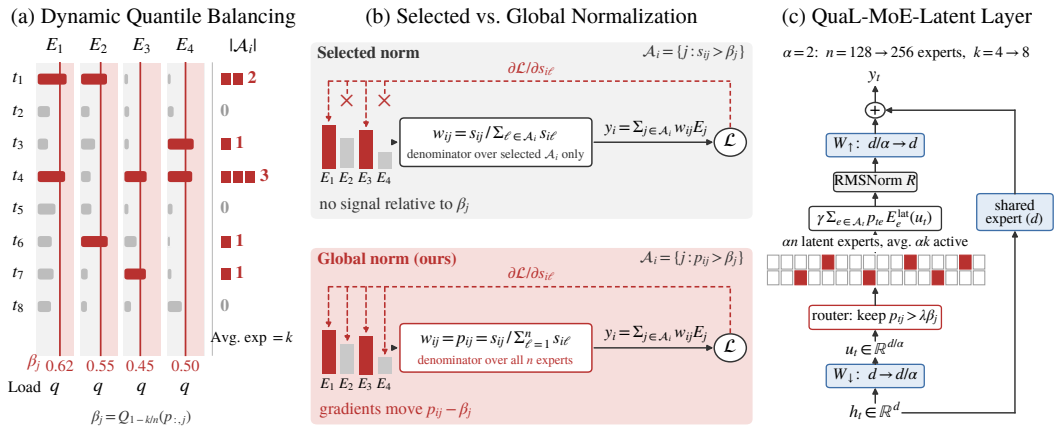


Figure 1: Overview of QuaL-MoE. **(a)** Dynamic routing with quantile balancing: each expert admits the tokens above its threshold. **(b)** Global normalization lets every router score receive task-loss gradients. **(c)** Latent experts make each activation cheaper and allocation finer-grained.

3 QuaL-MoE

QuaL-MoE develops a practical dynamic **L**atent **MoE** with **Qua**ntile Balancing, built from four components: a loss-free per-expert quantile rule that dynamically allocates experts (§3.1); a global normalization that connects the discrete selection to the task loss (§3.2); distributed estimators that realize the thresholds across training ranks (§3.3); and a single inference-time knob that trades quality against compute without retraining (§3.4). We apply this routing to both full-width experts (QuaL-MoE) and latent experts (QuaL-MoE-Latent) for finer-grained allocation.

3.1 Dynamic quantile thresholding

The expert-load constraints in Eq. 1 already imply q assignments per expert. Maintaining this budget therefore does not require a fixed Top-k allocation to every token. QuaL-MoE removes the token-side constraint and retains the target load of each expert:

$$\max_{x_{i,j}\in\{0,1\}} \sum_{i,j} x_{i,j} s_{i,j} \quad \text{s.t.} \quad \sum_{i} x_{i,j} = q,\ \forall j \qquad q = \frac{mk}{n}. \tag{3}$$

The resulting problem becomes expert-choice routing (Zhou et al., 2022): each expert selects its q highest-scoring tokens. For expert j, the threshold $\beta_j^\star$ is the $(1 - k/n)$ quantile of its scores:

$$\beta_j^\star = Q_{1-k/n}(s_{:,j}), \qquad \mathcal{A}_i^\star = \left\{ j : s_{i,j} > \beta_j^\star \right\}, \tag{4}$$

where $\mathcal{A}_i^\star$ is the resulting set of experts selected for token i. Each expert receives q tokens, so $\frac{1}{m}\sum_i |\mathcal{A}_i^\star| = k$, although the number of experts assigned to an individual token can vary.

To avoid making routing decisions depend on other tokens in the current batch, we maintain thresholds from previous training steps. At step t, token i selects $\mathcal{A}_i^{(t)} = \{j : s_{i,j}^{(t)} > \beta_j^{(t)}\}$. The current scores update the thresholds for the next step:

$$\beta_j^{(t+1)} = \rho\beta_j^{(t)} + (1-\rho)\widehat{Q}_{1-k/n}\left(s_{:,j}^{(t)}\right) \tag{5}$$

where $\widehat{Q}$ estimates the quantile over the distributed batch and $\rho \in [0, 1)$ controls temporal smoothing.

Eq. 4 describes the exact current-batch assignment, which requires the full batch before routing; Eq. 5 provides a causal online approximation. Running thresholds therefore target balanced expert loads and an average allocation of k, rather than enforcing either exactly on every batch.

3.2 ALIGNING SELECTION WITH THE TASK LOSS VIA GLOBAL NORMALIZATION

The discrete QUAL-MOE expert selection is non-differentiable. During back-propagation, the selected set $\mathcal{A}_i$ is effectively treated as fixed. Therefore, the router can only learn from the differentiable gate weights multiplying the expert outputs (Panda et al., 2024). To align expert selection with loss optimization, we propose global normalization: given router logits $z_{i,j}$ and sigmoid routing score $s_{i,j} = \sigma(z_{i,j})$ (DeepSeek-AI, 2024), each score is normalized by the sum over all experts:

$$p_{i,j} = \frac{s_{i,j}}{\sum_{\ell=1}^{n} s_{i,\ell}}.$$

With global normalization, $p_{i,j}$ replaces $s_{i,j}$ in both expert selection and threshold estimation. The shared denominator couples the router scores: increasing one expert's share reduces the shares of all other experts. We compare this design against two alternative normalizations in §5.4.

3.3 DISTRIBUTED THRESHOLD ESTIMATORS

To approximate the global per-expert threshold $\beta_j^{(t)}$ in a distributed system, we design three estimators:

- QUANTILE-AVG. Each worker computes a local order-statistic threshold, with the quota determined by its local token count. An all-reduce then averages the resulting n thresholds across workers.
- QUANTILE-HIST. Workers first obtain a common minimum and maximum score for each expert. They bin local scores into 256 equal-width intervals and sum the histograms across workers. The estimator locates the bin containing descending rank $q + 1$ and interpolates within that bin.
- QUANTILE-BS. Workers bracket each threshold by the global minimum and maximum scores. At each iteration, they count local scores strictly above the midpoint and sum those counts. If the global count exceeds q, the midpoint becomes the lower bound; otherwise, it becomes the upper bound. After 16 iterations, the estimator returns the upper bound as the threshold.[1]

3.4 INFERENCE-TIME BUDGET CONTROL

At inference, we introduce a single coefficient $\lambda > 0$ that scales all trained thresholds, letting a deployed model flexibly trade compute for quality without any parameter update or threshold re-estimation. Token i then selects expert j when its score exceeds the scaled threshold:

$$\mathcal{A}_i = \{j : p_{i,j} > \lambda \beta_j\}$$

Lowering λ admits more experts per token and raises the budget; conversely, $\lambda > 1$ lowers it. Setting $\lambda = 1$ recovers the trained operating point. We investigate its performance in §5.2.

4 EXPERIMENTS

We first compare QUAL-MOE with other dynamic routers at the 1.2B scale (§5.1) and then scale it to 10B parameters (§5.2). We further analyze how it allocates experts (§5.3), ablate normalization and threshold estimation (§5.4), and finally extend it to finer-grained latent experts (§5.5).

Models and data. We train decoder-only Transformer MoE models on TxT360v2 (Institute of Foundation Models, 2026b), a mixture of web, code, mathematical, and multilingual corpora. Experiments are conducted at two model scales: 1) the *1.2B experiments* compare QUAL-MOE with Top-p, DynMoE, and fixed Top-k baselines at different target counts $\bar{k}$; 2) the *10B experiments* scale the comparison with Top-k routing to both standard and latent MoE architectures. The 1.2B experiments (MoE-1.2B-0.8B-8E4A) train for 30B tokens with eight routed experts and target budgets of four and two. The standard 10B model (MoE-10B-1B-128E4A) trains for 840B tokens with 128 routed experts per MoE layer, activating four per token. Its latent variant has 256 experts and a target of eight activated experts. Appendix A details the architectures (Table 3) and hyperparameters. Both scales use the same source sampling weights and a sequence length of 8,192 tokens.

Baselines. We compare QUAL-MOE against three established routing baselines: (i) *Top-k* (Shazeer et al., 2017): selects a fixed number of experts for every token; (ii) *Top-p* (Huang et al., 2024): uses

[1] After 16 iterations the threshold is localized to 2^{-16} of the initial score range.

Table 1: Routing methods in the 1.2B experiments at nominal budgets $k = 4$ and $k = 2$.

Metric	1.2B–8E4A			1.2B–8E2A			Uncontrolled
	Top-k $k = 4$	QuaL-MoE $\bar{k}^\star = 4$	Top-p $p = 0.85$	Top-k $k = 2$	QuaL-MoE $\bar{k}^\star = 2$	Top-p $p = 0.7$	DynMoE
Mean routed experts	4.000	3.989	4.328	2.000	1.983	2.040	1.048
Held-out PPL ↓	4.991	**4.990**	5.013	5.102	**5.082**	5.151	5.824
Avg. downstream tasks ↑	32.780	**32.884**	32.744	32.269	**32.456**	32.139	30.092

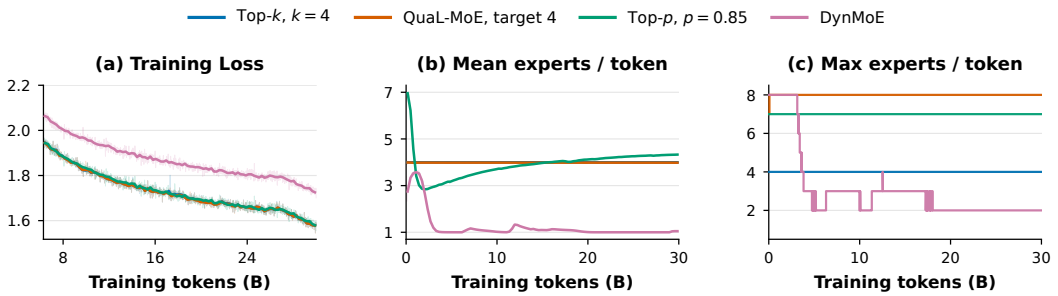


Figure 2: Training dynamics of the 1.2B experiments. Top-k fixes $k = 4$, QuaL-MoE targets four experts on average, Top-p uses $p = 0.85$, and DynMoE adaptively selects experts without a target $\bar{k}$. a fixed cumulative-probability cutoff p to select experts; and (iii) *DynMoE* (Guo et al., 2025): uses per-expert sigmoid gates to adaptively determine the activated experts, without a target $\bar{k}$.

Unless otherwise stated, QuaL-MoE uses globally normalized sigmoid scores, no selected-set renormalization, Quantile-Avg, and threshold decay $\rho = 0.3$. Top-k uses loss-free balancing biases (Wang et al., 2024a) and an entropy-balancing auxiliary loss with coefficient 10^{-4}. In the 1.2B experiments, we use Top-p cutoffs $p = 0.85$ and $p = 0.7$ for comparisons at nominal budgets $k = 4$ and $k = 2$, respectively. For DynMoE, expert pruning and growth are performed every 100 updates.

Evaluation. Validation perplexity is evaluated on held-out web, multilingual, code, academic, and other text slices, together with performance on 21 downstream benchmarks listed in Appendix A. To present routing behavior, we report the mean activated expert count $\bar{k}$, the maximum activated expert count $k_{\max}$ in dynamic approaches, and expert-load imbalance measured by MaxVio (Wang et al., 2024a). To investigate inference-time budget control, we freeze the 10B QuaL-MoE checkpoint and multiply its thresholds by $\lambda \in \{0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.4\}$ without retraining.

5 Results

5.1 Comparison with other dynamic routers

We first compare QuaL-MoE with Top-k, Top-p, and DynMoE in the 1.2B experiments (Table 1). At nominal $k = 4$, QuaL-MoE attains the best average downstream score (32.884) and held-out perplexity (4.990) of all methods. At nominal $k = 2$, it achieves a mean score of 32.456, compared with 32.269 for Top-k and 32.139 for Top-p. DynMoE adaptively activates experts but achieves a clearly higher training loss and lower downstream scores than all other methods. Beyond predictive quality, QuaL-MoE provides more direct control over average expert use. As shown in Figure 2, Top-p's mean expert count drifts from approximately 3.4 to 4.3 in the nominal four-expert setting, whereas QuaL-MoE stays close to its target. Moreover, selecting p in Top-p requires tuning to approximate a desired expert budget, and a fixed p does not maintain that budget as training progresses. In contrast, QuaL-MoE specifies the target count directly and adapts its thresholds to track it.

Takeaway 1. QuaL-MoE *combines stronger predictive quality with more stable control of* $\bar{k}$.

5.2 Scaling to 10B parameters

We then scale QuaL-MoE to 10B parameters and test whether it improves predictive quality over fixed Top-k, and whether its thresholds let one trained model support different inference budgets.

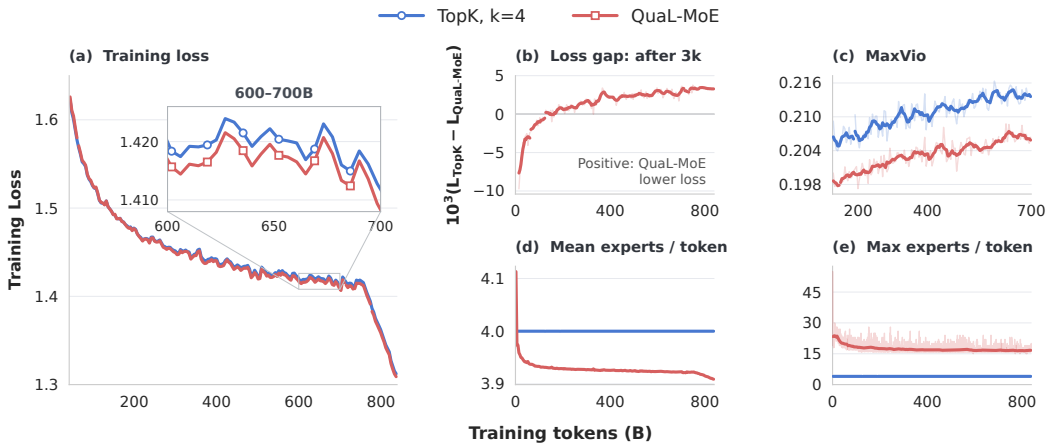


Figure 3: Training dynamics of the MoE-10B-1B-128E4A experiment. (a) Training loss after 10k steps. (b) Loss gap $\Delta L = L_{\text{Top-}k} - L_{\text{QuaL-MoE}}$; positive values favor QuaL-MoE. (c) Load imbalance (MaxVio; lower is better). (d,e) Mean and max routed experts per token.

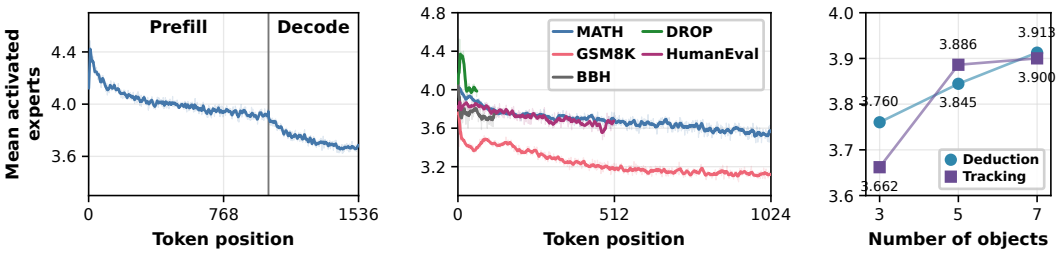
Table 2: Inference-time quality–budget trade-off. The trained threshold is $\lambda = 1.0^*$. Avg. $\bar{k}$ measures the average activated experts per token during inference by QuaL-MoE; reducing λ increases the expert budget. Mean_{21} averages 21 downstream task scores (all tasks in Table 4). AMMLU: Arabic-MMLU; DROP and TriviaQA report F1, HumanEval and MBPP pass@1.

Router / λ	Avg. $\bar{k}$	PPL ↓	Mean_{21} ↑	MMLU	AMMLU	MATH	DROP	BBH	TriviaQA	HumanEval	MBPP
Top-k	4.000	3.789	52.240	55.88	52.26	44.90	44.47	32.88	46.55	44.51	46.40
QuaL-MoE, 0.6	7.299	3.765	52.773	**56.60**	53.50	44.74	**46.65**	35.20	**47.50**	45.73	**50.00**
QuaL-MoE, 0.7	5.976	**3.762**	**52.886**	56.33	53.39	45.58	46.40	35.08	47.37	47.56	48.60
QuaL-MoE, 0.8	5.065	3.763	52.768	56.23	**53.69**	46.04	46.25	**35.63**	47.30	46.95	49.20
QuaL-MoE, 0.9	4.410	3.768	52.671	56.28	53.17	45.92	45.83	34.91	47.23	46.95	48.80
QuaL-MoE, 1.0^*	3.916	3.776	52.697	56.05	52.84	**46.10**	45.71	35.09	47.23	**52.44**	48.40
QuaL-MoE, 1.1	3.529	3.789	52.311	56.08	52.58	45.70	45.79	34.62	47.08	47.56	47.40
QuaL-MoE, 1.2	3.212	3.805	52.122	55.54	52.38	45.88	45.44	34.40	47.22	50.00	48.40
QuaL-MoE, 1.4	2.713	3.850	51.539	54.90	52.31	45.32	44.64	34.27	46.96	49.39	45.60

Training dynamics. At the trained threshold ($\lambda = 1$), QuaL-MoE achieves higher aggregate downstream performance and lower held-out perplexity while using slightly fewer experts than architecture-matched Top-k (Table 2). These results show that the observed quality improvement can be achieved with lower routing cost. To examine how the comparison develops during training, Figure 3 tracks loss and routing behavior in the 128-expert models. QuaL-MoE develops a consistent loss advantage throughout training. Over the final 10k updates, the mean activated expert count $\bar{k}$ of QuaL-MoE is 3.912. Their mean loss gap is around 0.006, corresponding to approximately 2.4B additional training tokens. Individual tokens can nevertheless receive substantially more computation: the maximum activated experts $k_{\max}$ ranges from 15 to 20. Thus, allocation remains variable even when the average stays close to four. QuaL-MoE also maintains slightly more balanced expert load, with mean MaxVio of 0.203 versus 0.211 for Top-k despite using no auxiliary loss.

Inference control. We then explore whether scaling the routing thresholds during inference allows a trained QuaL-MoE model to reduce inference-time expert use while preserving predictive quality (Table 2). At $\lambda = 1.1$, QuaL-MoE uses 3.529 experts per token on the held-out evaluation, compared with 4.0 for Top-k, a reduction of 11.8%. Its downstream task mean remains slightly better, at 52.311 versus 52.240, with similar validation perplexity. Threshold adjustment thus reduces routed-expert activations while retaining the quality advantage over the fixed-count baseline.

Increasing the inference budget can improve quality, but the gains saturate at higher budgets. Lowering λ from 1.0 to 0.7 increases mean expert use from 3.916 to 5.976 and improves the task mean from 52.697 to 52.886. Compared with $\lambda = 0.6$, $\lambda = 0.7$ uses 18.1% fewer experts while achieving a higher aggregate task mean and lower perplexity. This suggests that threshold calibration can avoid unnecessary expert computation. Moreover, the influence of inference budget varies across tasks. Although $\lambda = 0.7$ achieves the highest aggregate task mean, HumanEval and MATH perform best at



(a) Held-out prefill/decode **(b)** Downstream decode **(c)** BBH object count

Figure 4: Expert Allocation. Allocation declines with position in held-out continuations (a) and downstream decoding (b), while expert allocation increases with object count in BBH tasks (c).

the trained threshold $\lambda = 1.0$; MMLU and DROP F1 perform best at $\lambda = 0.6$. λ thus allows users to flexibly adjust the inference budget of QUAL-MOE at deployment without re-training.

Takeaway 2. QUAL-MOE *outperforms Top-k with lower routing cost, and it offers flexible inference-time control over expert use, letting one trained model serve different compute budgets.*

5.3 EXPERT ALLOCATION ACROSS INPUTS

A natural intuition behind dynamic routing is that expert allocation varies with task domain, language, or complexity. We investigate whether QUAL-MOE exhibits these patterns under fixed routing thresholds, and further examine how allocation changes across token positions.

Task domain. We first examine whether the same routing thresholds produce different expert budgets across downstream tasks (Table 5 in Appendix B). We observe that allocation varies across workloads: Arabic-MMLU receives approximately 44% more experts than MMLU (5.679 vs. 3.938), whereas GSM8K activates only 3.335 experts per token on average, about 17% below the target of four. QUAL-MOE-Latent shows the same extremes, with Arabic-MMLU highest and GSM8K lowest.

Token position. We next examine whether expert allocation changes as a sequence develops. As shown in Figure 4, expert use generally declines with position within both prefill and decoding (panels a and b). Downstream generation also exhibits distinct allocation profiles across tasks. The decline in expert use at later positions suggests increasing savings in expert computation over fixed Top-k as sequences grow. This makes QUAL-MOE well suited to long-context and multi-turn generation.

Within-task difficulty. Beyond differences between tasks, greater structural complexity leads to more expert use (Figure 4(c)). Increasing object count from three to seven in BBH (Suzgun et al., 2023) examples raises expert allocation by 4.1% for logical deduction and 6.5% for object tracking. Each additional object adds steps to the reasoning chain, so allocation follows the structure of the reasoning. MATH (Hendrycks et al., 2021b) difficulty levels, by contrast, measure mathematical depth, which does not translate into harder token-level generation in a natural-language CoT; accordingly, mean allocation stays similar from level 1 to level 5 in both generated and reference solutions.

We also test whether chain-of-thought (CoT) prompting (Wei et al., 2022) increases expert use. On the BBH questions, mean allocation over the question tokens is 3.894 with direct prompting and 3.855 with CoT prompting. Under a longer generation allowance, CoT uses fewer experts per output token. Longer generations thus draw fewer experts per token, matching the positional trend.

Takeaway 3. QUAL-MOE *spends fewer experts as sequences grow longer and more on problems that require more reasoning steps, rather than following annotated difficulty labels. The decline with position makes* QUAL-MOE *well suited to long-context and multi-turn inference workloads.*

5.4 NORMALIZATION AND THRESHOLD ESTIMATION

Even when average expert use stays near its target, individual tokens may activate many experts. We examine how score normalization affects these extremes and overall model quality. Additionally, we explore how threshold estimation affects control of the average expert budget.

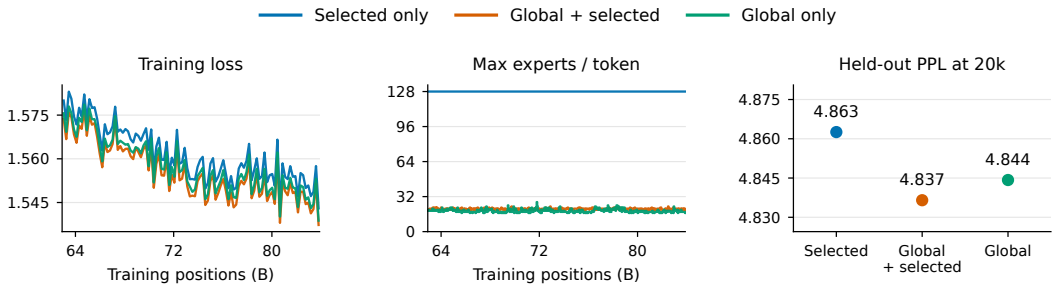


Figure 5: **Routing Score Normalization Variants**. Selected-only routing at times activates all 128 experts; both global variants have smaller maxima and lower held-out PPL and training loss

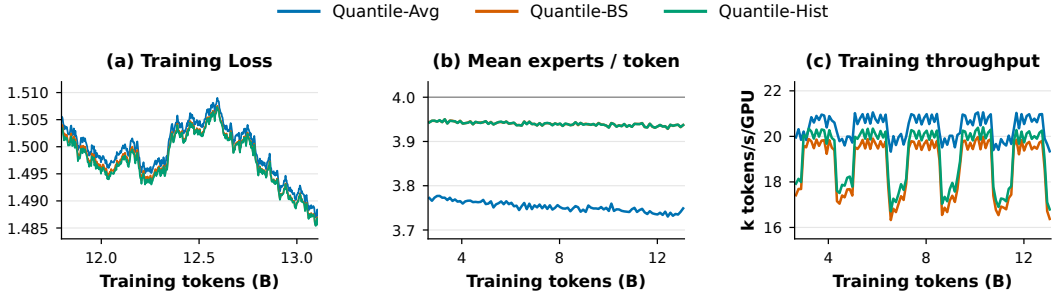


Figure 6: **Threshold Estimation Comparison**. QUANTILE-BS and QUANTILE-HIST keep the average expert count within 1.6% of the target of four, compared with 6.4% for QUANTILE-AVG, at similar training losses. QUANTILE-AVG attains the highest throughput of the three estimators.

Routing score normalization. We compare global normalization with two variants (Figure 5): *selected-only*, which selects experts by the raw score $s_{i,j}$ and renormalizes the output weights over the selected set, and *global + selected*, which selects by $p_{i,j}$ but also renormalizes over the selected set. Selected-only normalization repeatedly activates all 128 experts for some tokens, whereas the maxima stay at 27 or fewer with either globally normalized variant. This reduction occurs both with and without selected-set renormalization, suggesting that normalizing scores before expert selection helps suppress excessively dense activation. Both globally normalized variants also achieve lower mean training loss and lower validation perplexity. These results support global normalization as a simple way to improve both routing sparsity and training quality in threshold-based dynamic routing.

Threshold estimation. Figure 6(b) compares how distributed quantile estimators realize the target average expert count. QUANTILE-BS and QUANTILE-HIST undershoot the target $k = 4$ by 1.6%, compared with 6.4% for QUANTILE-AVG, despite similar training losses. However, QUANTILE-AVG requires only one all-reduce of n thresholds and is thus more cost-efficient than the other two.

5.5 LATENT EXPERT GRANULARITY

The benefits of flexible expert allocation may depend on the expert architecture. We investigate whether they extend to latent MoE by comparing QUAL-MOE with architecture-matched Top-k.

Latent vs. full-width experts. Figure 7 compares both routers with full-width and latent experts. Latent experts consistently benefit QUAL-MOE. QUAL-MOE-Latent attains lower validation loss than full-width QUAL-MOE at every evaluated checkpoint, and lower training loss. This gain is larger than the corresponding gain for Top-k, whose latent variant improves validation loss by only 0.008–0.010. Finer-grained experts therefore help QUAL-MOE more than fixed-count routing.

Comparison with architecture-matched Top-k. With the same latent architecture and target budget, QUAL-MOE-Latent outperforms Top-k-Latent throughout training. Its validation loss is lower at every checkpoint, and its training loss is lower. QUAL-MOE-Latent achieves the lowest training and validation losses among all four configurations. As in the full-width setting, the gain comes from redistributing experts across tokens rather than from a larger fixed budget: while Top-k-Latent always routes to eight latent experts, individual QUAL-MOE-Latent tokens activate up to 26–76 of them (mean maximum 36.3), about twice the full-width QUAL-MOE range when measured in latent units.

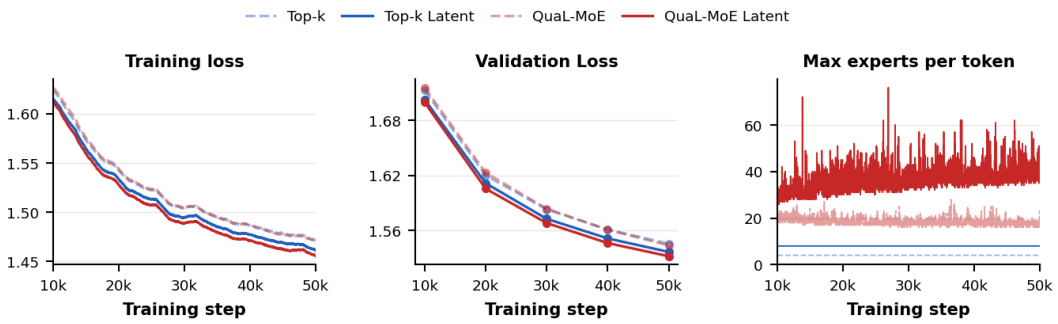


Figure 7: Latent and full-width experts under QuaL-MoE and Top-k routing. QuaL-MoE-Latent improves on full-width QuaL-MoE and outperforms architecture-matched Top-k-Latent.

Takeaway 4. *Latent experts strengthen* QuaL-MoE *more than Top-k:* QuaL-MoE-*Latent improves on full-width* QuaL-MoE *and consistently outperforms architecture-matched Top-k-Latent.*

6 Related Work

Adaptive expert budgets. Dynamic routing methods differ in how they distribute computation across tokens. MoE-Dynamic varies the expert count using routing confidence (Huang et al., 2024), while DynMoE learns token-wise activation counts and adapts the expert population during training (Guo et al., 2025). AdaMoE introduces null experts so fixed-cardinality selection can activate different numbers of real experts (Zeng et al., 2024). SeqTopK pools the budget across a sequence (Wen et al., 2025), and DTop-p combines cumulative-probability selection with a proportional–integral controller for average sparsity (Jin et al., 2026). We study expert-wise quantile targets, which connect the aggregate activation budget to the traffic received by each individual expert.

Load balancing and quantile routing. Balancing expert traffic is a central requirement of sparse MoE training (Shazeer et al., 2017; Fedus et al., 2022). Loss-Free Balancing updates expert selection biases from recent loads (Wang et al., 2024a), and fixed-Top-k QB derives its bias updates from score quantiles (Kimi Team, 2026). Expert Choice gives each expert a fixed token bucket, allowing token-wise computation to vary (Zhou et al., 2022). Expert Threshold routing, considered as our concurrent work, compares scores with per-expert quantile thresholds to enable variable token-wise expert counts (Sun et al., 2026). Both are concurrent to our work. QuaL-MoE develops a practical dynamic routing framework by introducing a globally normalized routing score shared by expert selection and output weighting, enabling task-loss gradients to shape routing decisions; designing distributed estimators for scalable threshold computation; and integrating the resulting router with LatentMoE to support finer-grained dynamic computation.

Expert granularity. Smaller experts allow a routing decision to allocate computation in finer increments. DeepSeekMoE combines smaller routed experts with shared experts (Dai et al., 2024), and XMoE couples small experts with threshold-based selection (Yang et al., 2024). MoLAE introduces expert computation in a shared lower-dimensional space (Liu et al., 2025), and LatentMoE projects tokens into a narrower latent space so that more, smaller experts can be activated at similar cost (Elango et al., 2026); Kimi K3 adopts a normalized variant at scale (Kimi Team, 2026). We adopt this latent factorization as a compatible branch for quantile routing: the compression factor sets the size of one activation, while the target mean expert count sets how many such units are used.

7 Conclusion

We presented QuaL-MoE, a dynamic MoE router in which each expert admits tokens above a quantile threshold, trained with global normalization and distributed threshold estimators. At 1.2B, it outperforms Top-p and DynMoE while holding its target budget; at 10B, it outperforms Top-k with fewer experts, and a single coefficient λ adjusts its budget at inference without retraining. Its allocation declines with token position and grows with reasoning steps, and latent experts strengthen it further, making per-token computation controllable in training and adjustable at inference.

AI USE STATEMENT

We used large language models as writing assistants to polish the language and improve the clarity of the manuscript, and to help write a small number of plotting scripts for figures. All research ideas, method design, experiments, and analyses were carried out by the authors. The authors reviewed and verified all AI-assisted text and code, checked every reported number against the underlying experimental records, and take full responsibility for the content of this paper.

REPRODUCIBILITY STATEMENT

Section 3 specifies the routing scores, quantile-based threshold updates, global normalization, distributed threshold estimators, and the inference-time scaling factor λ. Section 4 and Appendix A describe the model configurations, training budgets, data, baselines and their hyperparameters, evaluation benchmarks, decoding settings, and how activated-expert counts are measured. All compared methods share the same pretraining corpus, sampling weights, sequence length, and evaluation protocol. We will publicly release the training and evaluation code, including the QUAL-MOE router implementation, all model and training configurations, the data processing and sampling configurations, and the evaluation scripts, together with the data needed to reproduce our results.

REFERENCES

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

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL https://arxiv.org/abs/2107.03374.

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 *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2924–2936, 2019. doi: 10.18653/v1/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. *arXiv preprint arXiv:1803.05457*, 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. *arXiv preprint arXiv:2110.14168*, 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. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 1280–1297, 2024. doi: 10.18653/v1/2024.acl-long.70. URL https://aclanthology.org/2024.acl-long.70/.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024. URL https://arxiv.org/abs/2412.19437.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2368–2378, 2019. doi: 10.18653/v1/N19-1246.

Venmugil Elango, Nidhi Bhatia, Roger Waleffe, Rasoul Shafipour, Tomer Asida, Abhinav Khattar, Nave Assaf, Maximilian Golub, Joey Guman, Tiyasa Mitra, Ritchie Zhao, Ritika Borkar, Ran Zilberstein, Mostofa Patwary, Mohammad Shoeybi, and Bita Rouhani. LatentMoE: Toward optimal accuracy per FLOP and parameter in mixture of experts. *arXiv preprint arXiv:2601.18089*, 2026. URL https://arxiv.org/abs/2601.18089.

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 https://www.jmlr.org/papers/v23/21-0998.html.

Yongxin Guo, Zhenglin Cheng, Xiaoying Tang, Zhaopeng Tu, and Tao Lin. Dynamic mixture of experts: An auto-tuning approach for efficient transformer models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2405.14297.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a. URL https://arxiv.org/abs/2009.03300.

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. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2021b. URL https://arxiv.org/abs/2103.03874.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. Harder task needs more experts: Dynamic routing in MoE models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 12883–12895, 2024. doi: 10.18653/v1/2024.acl-long.696. URL https://aclanthology.org/2024.acl-long.696/.

Institute of Foundation Models. K2-Horizon. Hugging Face, 2026a. URL https://huggingface.co/collections/IFM/k2-horizon-6a73e8c7519e17b64f550a79.

Institute of Foundation Models. TxT360-v2. Hugging Face, 2026b. URL https://huggingface.co/datasets/IFM/TxT360-v2.

Can Jin, Hongwu Peng, Mingcan Xiang, Qixin Zhang, Xiangchi Yuan, Amit Hasan, Ohi Dibua, Yifan Gong, Yan Kang, and Dimitris N. Metaxas. DTop-p MoE: Sparsity-controlled dynamic Top-p MoE for foundation model pre-training. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2512.13996.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1601–1611, 2017. doi: 10.18653/v1/P17-1147.

Kimi Team. Kimi K3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026. URL https://arxiv.org/abs/2607.24653.

Fajri Koto, Haonan Li, Sara Shatnawi, Jad Doughman, Abdelrahman Sadallah, Aisha Alraeesi, Khalid Almubarak, Zaid Alyafeai, Neha Sengupta, Shady Shehata, Nizar Habash, Preslav Nakov, and Timothy Baldwin. ArabicMMLU: Assessing massive multitask language understanding in Arabic. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 5622–5640, 2024. doi: 10.18653/v1/2024.findings-acl.334.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural Questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019. doi: 10.1162/tacl_a_00276.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 785–794, 2017. doi: 10.18653/v1/D17-1082.

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

Zehua Liu, Han Wu, Ruifeng She, Xiaojin Fu, Xiongwei Han, Tao Zhong, and Mingxuan Yuan. MoLAE: Mixture of latent experts for parameter-efficient language models. *arXiv preprint arXiv:2503.23100*, 2025. URL https://arxiv.org/abs/2503.23100.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018. doi: 10.18653/v1/D18-1260.

Ashwinee Panda, Vatsal Baherwani, Zain Sarwar, Benjamin Therien, Sambit Sahu, Stephen Rawls, Supriyo Chakraborty, and Tom Goldstein. Dense backpropagation improves routing for sparsely-gated mixture-of-experts. In *Proceedings of the 4th NeurIPS Efficient Natural Language and Speech Processing Workshop*, volume 262 of *Proceedings of Machine Learning Research*, pp. 81–101. PMLR, 2024. URL https://proceedings.mlr.press/v262/panda24a.html.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 4462–4472, 2019. doi: 10.18653/v1/D19-1454.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. 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.

Hanchi Sun, Yixin Liu, Yonghui Wu, and Lichao Sun. Expert threshold routing for autoregressive language modeling with dynamic computation allocation and load balancing. *arXiv preprint arXiv:2603.11535*, 2026. URL https://arxiv.org/abs/2603.11535.

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 *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023. doi: 10.18653/v1/2023.findings-acl.824.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*, 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, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b. URL https://arxiv.org/abs/2406.01574.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2201.11903.

Tiansheng Wen, Yifei Wang, Aosong Feng, Long Ma, Xinyang Liu, Yifan Wang, Lixuan Guo, Bo Chen, Stefanie Jegelka, and Chenyu You. Route experts by sequence, not by token. *arXiv preprint arXiv:2511.06494*, 2025. URL https://arxiv.org/abs/2511.06494.

Yuanhang Yang, Shiyi Qi, Wenchao Gu, Chaozheng Wang, Cuiyun Gao, and Zenglin Xu. XMoE: Sparse models with fine-grained and adaptive expert selection. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11664–11674, 2024. doi: 10.18653/v1/2024.findings-acl.694. URL https://aclanthology.org/2024.findings-acl.694/.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019. doi: 10.18653/v1/P19-1472.

Zihao Zeng, Yibo Miao, Hongcheng Gao, Hao Zhang, and Zhijie Deng. AdaMoE: Token-adaptive routing with null experts for mixture-of-experts language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 6223–6235, 2024. doi: 10.18653/v1/2024.findings-emnlp.361. URL https://aclanthology.org/2024.findings-emnlp.361/.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1910.07467.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M. Dai, Zhifeng Chen, Quoc V. Le, and James Laudon. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/2f00ecd787b432c1d36f3de9800728eb-Abstract-Conference.html.

A EXPERIMENTAL DETAILS

Table 3 lists the architectures of the three model families we train. Comparisons use the same architecture within each suite; token counts denote nominal training positions.

Table 3: Model families and evaluation endpoints. The latent suite doubles the expert pool while halving routed input width. Counts exclude the shared expert in each MoE layer.

	1.2B Experiments	10B QUAL-MOE	10B QUAL-MOE-Latent
Layers / MoE layers	24 / 21	26 / 23	26 / 23
Residual width	1,536	1,024	1,024
Routed experts	8	128	256
Routed input width	1,536	1,024	512
Nominal active experts	4	4	8
Shared expert	1	1	1
Vocabulary	64,256	250,624	250,624

Training configurations. The small-model suite has residual width 1,536, 21 MoE layers, and eight routed experts. At nominal $k = 4$, it has 1B non-embedding parameters and 600M active non-embedding parameters (1.2B and 0.8B with embeddings, hence MoE-1.2B-0.8B-8E4A). Training uses context length 8,192, global batch 256, seed 42, and 14,306 updates (30B training tokens, including padding and repeated data). Its 42-source mixture uses the 64,256-token K2-Horizon-0.9B tokenizer (Institute of Foundation Models, 2026a). The Top-k balancing-bias update rate is 0.001.

The full-width suite has residual width 1,024, 23 MoE layers, and 128 routed experts. It uses the 250,624-token K2-Horizon-375B-A23B tokenizer (Institute of Foundation Models, 2026a), context length 8,192, and global batch 512. The latent variant retains the residual architecture and narrows the routed input width to 512 ($\alpha = 2$), and doubles the routed experts to 256 in total.

Downstream tasks. The 21 primary metrics are BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), Natural Questions F1 (Kwiatkowski et al., 2019), PIQA (Bisk et al., 2020), Social IQA (Sap et al., 2019), WinoGrande (Sakaguchi et al., 2020), OpenBookQA (Mihaylov et al., 2018), ARC-Easy and ARC-Challenge (Clark et al., 2018), RACE-Middle and RACE-High (Lai et al., 2017), COPA (Roemmele et al., 2011), MMLU macro accuracy (Hendrycks et al., 2021a), Arabic-MMLU macro accuracy (Koto et al., 2024), TriviaQA F1 (Joshi et al., 2017), GSM8K (Cobbe et al., 2021), BBH exact match (Suzgun et al., 2023), DROP F1 (Dua et al., 2019), HumanEval pass@1 (Chen et al., 2021), MBPP pass@1 (Austin et al., 2021), and MATH exact match (Hendrycks et al., 2021b). Scores use their native metrics on a 0–100 scale. Mean_{21} is their unweighted arithmetic mean.

A.1 ALLOCATION-PROFILE MEASUREMENT

Figure 4 uses the full-width QUAL-MOE model at 200k updates and $\lambda = 1$. Counts average 23 routed layers, excluding the shared expert. Each panel uses its own input cohort and token span.

BBH object counts. Panel (c) uses 250 examples in each three-, five-, and seven-object variant of logical deduction and tracking shuffled objects (1,500 examples, weighted equally). Allocation is measured at the two positions predicting the reference answer, starting at the final prompt token.

Difficulty and CoT comparisons. For all 5,000 MATH problems, we compare levels using generated-answer prediction and teacher-forced reference-solution prediction regions. Generated-answer prediction includes positions through the first EOS or the 512-token cap; 2,295 responses reach the cap. Reference-solution prediction uses the dataset solution, including the final prompt position that predicts its first token. Question-input measurements exclude BOS and few-shot examples.

The BBH prompting comparison uses the same 2,500 questions from ten structural groups, with 250 examples per group. Direct and CoT question-input measurements cover identical current-question token sequences, totaling 300,718 positions, under their respective prompt contexts. Generated-output prediction measurements include the final prompt position and continue through the first EOS or the output cap: 32 tokens for direct prompting and 256 for CoT. Total routed expert assignments in this region are 7,015,237 and 52,217,758 for direct and CoT prompting, respectively.

Table 4: Complete 21-task threshold sweep at 200k updates. Scores use native percentage metrics (higher is better). Increasing the expert budget produces different responses across tasks.

Task	Top-k	QuaL-MoE threshold multiplier λ							
	$k = 4$	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.4
MMLU	55.88	56.60	56.33	56.23	56.28	56.05	56.08	55.54	54.90
Arabic-MMLU	52.26	53.50	53.39	53.69	53.17	52.84	52.58	52.38	52.31
ARC-Easy	77.00	77.04	76.58	77.04	76.32	76.62	76.19	75.52	75.52
ARC-Challenge	44.29	44.64	44.38	44.89	43.86	43.18	43.95	44.64	44.03
COPA	82.00	85.00	85.00	84.00	85.00	85.00	84.00	82.00	84.00
PIQA	76.61	77.31	77.37	77.31	77.20	77.04	77.48	77.31	76.66
HellaSwag	51.67	51.87	51.91	51.88	51.59	51.52	51.48	51.24	50.74
WinoGrande	64.09	61.96	63.06	62.04	62.98	61.56	62.75	61.80	59.83
BoolQ	72.87	73.43	73.64	73.58	73.24	72.97	72.51	71.56	70.61
MATH	44.90	44.74	45.58	46.04	45.92	46.10	45.70	45.88	45.32
DROP F1	44.47	46.65	46.40	46.25	45.83	45.71	45.79	45.44	44.64
GSM8K	66.49	62.32	62.40	61.56	61.03	59.67	59.89	58.83	56.71
BBH	32.88	35.20	35.08	35.63	34.91	35.09	34.62	34.40	34.27
TriviaQA F1	46.55	47.50	47.37	47.30	47.23	47.23	47.08	47.22	46.96
Natural Questions F1	25.59	25.00	25.30	25.03	25.10	24.97	24.36	24.07	23.70
Social IQA	41.91	42.22	42.12	42.02	42.12	42.02	42.07	41.97	41.91
OpenBookQA	27.60	27.40	28.40	27.40	28.20	28.40	27.40	27.60	27.60
RACE-Middle	57.66	58.08	58.08	58.36	58.36	58.29	58.15	57.45	56.62
RACE-High	41.40	42.05	42.08	41.74	42.00	41.54	41.51	41.31	40.99
HumanEval	44.51	45.73	47.56	46.95	46.95	52.44	47.56	50.00	49.39
MBPP	46.40	50.00	48.60	49.20	48.80	48.40	47.40	48.40	45.60
21-task mean	52.240	52.773	52.886	52.768	52.671	52.697	52.311	52.122	51.539

Table 5: Mean routed-expert counts by task at the 200k QuaL-MoE checkpoint, using $\lambda = 1$ and excluding the shared expert. The same router allocates different average budgets across tasks. Abbreviations: AMMLU: Arabic-MMLU; TQ: TriviaQA; H-Eval: HumanEval.

Model	MMLU	AMMLU	ARC-E	ARC-C	COPA	PIQA	HellaSwag	WinoGrande	BoolQ	MATH	DROP	GSM8K	BBH	TQ	NQ	H-Eval	MBPP
QuaL-MoE	3.938	5.679	4.010	4.011	3.876	3.934	3.940	3.921	4.081	3.765	4.092	3.335	3.822	4.068	4.320	3.726	3.803
QuaL-MoE-Latent	4.041	5.017	4.198	4.184	4.088	4.242	4.154	4.100	4.244	3.729	4.058	3.364	3.795	4.239	4.330	3.593	3.619

B Complete Task Results

B.1 10B threshold sweep

Table 4 reports all 21 tasks at every evaluated threshold multiplier of the 10B QuaL-MoE model, and Table 5 reports its mean routed-expert count on each of the 17 downstream tasks.

B.2 1.2B experiments

The Top-p evaluation uses the same final 14,306-update checkpoint horizon, K2-Horizon-0.9B tokenizer, benchmark populations, prompts, few-shot examples and generation limits as the other eight-expert evaluations. Each condition covers 116 leaf tasks and 93,884 examples. Mean_{21} gives the 21 headline percentages equal weight, as in the main-text comparison of Table 1.

Table 6 reports the primary routing region separately for each task. Counts are total routed assignments divided by the number of included token–layer positions, with 21 MoE layers and 1 shared expert. For generative tasks, the answer-prediction region includes the final prompt position that predicts the first output and ends with the position predicting the first EOS. For choice-scoring tasks, the conditioned-input region includes the candidate branches, which can repeat the shared question prefix.

Table 6: Per-task mean routed experts at the 30B-position endpoints. Each value pools expert assignments over token–layer positions across examples and 21 routed MoE layers.

Benchmark	1.2B–8E4A			1.2B–8E2A			Uncontrolled
	Top-k $k = 4$	QuaL-MoE $\bar{k}^\star = 4$	Top-p $p = 0.85$	Top-k $k = 2$	QuaL-MoE $\bar{k}^\star = 2$	Top-p $p = 0.7$	DynMoE
MMLU	4.000	4.069	4.420	2.000	1.963	2.038	1.048
Arabic-MMLU	4.000	4.062	4.422	2.000	2.137	2.144	1.047
ARC-Easy	4.000	3.835	4.206	2.000	1.838	1.842	1.046
ARC-Challenge	4.000	3.856	4.238	2.000	1.850	1.852	1.046
COPA	4.000	3.775	4.415	2.000	1.748	1.896	1.045
PIQA	4.000	3.829	4.238	2.000	1.834	1.869	1.046
HellaSwag	4.000	3.790	4.432	2.000	1.831	1.932	1.047
WinoGrande	4.000	3.762	4.459	2.000	1.786	1.927	1.046
BoolQ	4.000	3.830	4.293	2.000	1.897	1.916	1.047
MATH	4.000	4.088	4.332	2.000	2.025	2.184	1.048
DROP F1	4.000	4.280	4.513	2.000	2.061	2.118	1.048
GSM8K	4.000	4.888	4.686	2.000	2.256	2.430	1.048
BBH	4.000	4.956	4.857	2.000	2.320	2.559	1.048
TriviaQA F1	4.000	3.952	4.245	2.000	2.003	1.880	1.048
NQ F1	4.000	4.027	4.470	2.000	2.095	1.969	1.048
Social IQA	4.000	3.796	4.441	2.000	1.853	1.879	1.046
OpenBookQA	4.000	3.842	4.391	2.000	1.786	1.891	1.045
RACE-Middle	4.000	3.947	4.582	2.000	1.900	1.992	1.047
RACE-High	4.000	3.863	4.531	2.000	1.880	1.973	1.048
HumanEval	4.000	4.451	4.522	2.000	2.224	2.352	1.048
MBPP	4.000	4.328	4.595	2.000	2.221	2.381	1.048

Table 7: MMLU-Pro-CoT question accuracy at final checkpoint, with $\lambda = 1$. Generation uses a 512-token output budget; QuaL-MoE scores higher while activating fewer routed experts.

Model	Scored N	Accuracy (%)	Cap hits / generated questions
Top-k	12,032	27.144	12,020 / 12,032
QuaL-MoE	12,032	28.657	11,984 / 12,032

Table 8: MMLU-Pro-CoT expert allocation in Table 7. Means pool routed assignments over 23 MoE layers and the included token positions.

Model	Prefill $\bar{k}$	Decode $\bar{k}$	Combined $\bar{k}$
Top-k	4.0000	4.0000	4.0000
QuaL-MoE	3.9583	3.6732	3.8777

B.3 MMLU-Pro with CoT

The full-width Top-k and QuaL-MoE checkpoints at 200k updates are evaluated on all 12,032 MMLU-Pro (Wang et al., 2024b) questions in 14 subjects, using their trained K2-Horizon-375B-A23B tokenizer, no chat template, a 4,096-token context, and five subject-matched demonstrations. Generation is greedy with a 512-token cap. A local adapter selects subject-matched demonstrations and extracts the final answer against the gold choice. Accuracy is per-question micro accuracy.

Table 7 reports accuracy: QuaL-MoE reaches 28.66%, compared with 27.14% for Top-k, while activating fewer experts. Unparseable answers are all counted as incorrect.

Prefill and decode routing are disjoint. Prefill includes BOS; decode counts consumed generated inputs before EOS. The combined count pools assignments over both regions. QuaL-MoE's prefill, decode and combined counts are 3.9583, 3.6732 and 3.8777 (Table 8); Top-k is 4 throughout.

B.4 Downstream performance with latent experts.

Table 9 compares QuaL-MoE-Latent with an architecture-matched Top-k-Latent baseline at 100k updates. QuaL-MoE-Latent achieves a higher overall Mean21 score, 47.713 versus 47.267. The

Table 9: Per-task downstream scores of the latent models at 100k updates, the last checkpoint evaluated for both. DROP, TriviaQA and NQ report F1.

Model	MMLU	Ar.-MMLU	ARC-E	ARC-C	COPA	PIQA	HellaSw.	WinoG.	BoolQ	MATH	DROP
Top-k-Latent ($k = 8$)	**49.69**	48.62	**72.98**	37.34	76.00	75.41	48.15	60.93	68.23	**35.08**	**38.86**
QUAL-MOE-Latent ($\bar{k}^\star = 8$)	49.47	**49.42**	72.56	**38.28**	**80.00**	**75.95**	**48.48**	**61.17**	**68.96**	34.78	38.67
Model	GSM8K	BBH	TriviaQA	NQ	SIQA	OBQA	RACE-M	RACE-H	HumanEval	MBPP	Mean_{21}
Top-k-Latent ($k = 8$)	**51.48**	30.47	**38.25**	19.42	**42.22**	26.80	**56.27**	39.99	37.80	38.60	47.267
QUAL-MOE-Latent ($\bar{k}^\star = 8$)	50.72	**31.01**	37.60	**20.29**	41.97	**28.40**	55.36	**40.02**	**39.02**	**39.84**	**47.713**

gains are spread across several benchmarks, including ARC-Challenge, COPA, PIQA, HellaSwag, WinoGrande, BoolQ, BBH, NQ, OpenBookQA, HumanEval, and MBPP. Although Top-k remains slightly better on some tasks such as MMLU, MATH, GSM8K, and TriviaQA, the aggregate result shows that dynamic allocation continues to provide an advantage when combined with finer-grained latent experts.