LightMoE: Task-Aware Expert Availability Management for Memory-Efficient MoE-LLM Inference

Puhan Luo, Yunhao Yao, Junyang Wang, Junyang Zhang*, Xiang-Yang Li
University of Science and Technology of China, Hefei, China
luopuhan@mail.ustc.edu.cn, yaoyunhao@mail.ustc.edu.cn, iswangjy@mail.ustc.edu.cn,
zhangjunyang@mail.ustc.edu.cn, xiangyangli@ustc.edu.cn

Abstract

Mixture-of-Experts (MoE) models offer a promising path for scaling model capacity, yet their massive memory footprint poses significant challenges for deployment on resource-constrained edge devices. Existing solutions, such as static pruning or dynamic offloading, often struggle to balance model accuracy with inference latency due to irreversible information loss or prohibitive I/O overhead. In this paper, we propose LightMoE, a novel framework for memory-efficient MoE inference that exploits the inherent functional redundancy and temporal locality of expert activation. LightMoE employs a frequency-aware expert initialization strategy to retain a compact core of resident experts and introduces a similarity-based redirection mechanism to compensate for missing experts without incurring I/O costs. Furthermore, it incorporates a lightweight runtime manager that performs coarse-grained, task-level expert replacement to adapt to shifting data distributions. Empirical evaluations on representative edge platforms demonstrate that LightMoE achieves a superior accuracy-efficiency trade-off, improving average accuracy by 4.3% over static pruning and 2.4% over dynamic swapping methods, while maintaining inference latency comparable to strictly pruned models.

1 Introduction

Large language models (LLMs) (OpenAI et al., 2024; Yang et al., 2025) have rapidly scaled in recent years, and the Mixture-of-Experts (MoE) (Lepikhin et al., 2020; Fedus et al., 2022; Liu et al., 2024) architecture has emerged as a key paradigm for increasing model capacity without proportionally increasing inference cost. By activating only a small subset of experts per token, MoE models significantly reduce active parameters during inference while achieving strong performance. Meanwhile, the growing demand for on-device and edge deployment, such as embodied intelligence (Liu et al., 2025) and autonomous agents (Tran et al., 2025), makes efficient MoE inference increasingly important under strict memory constraints (Yi et al., 2025; Xue et al., 2025).

However, a fundamental tension exists: while MoE reduces computational cost via sparse activation, the requirement to keep the massive expert pool memory-resident creates a prohibitive footprint for edge devices. Prior work has explored expert pruning (Lu et al., 2024; Zhou et al., 2025) and quantization (Li et al., 2024; Fu et al., 2025), which reduces memory at the cost of accuracy, as well as expert prefetching or swapping (Huang et al., 2024a; Yi et al., 2025), which reduces peak memory usage but introduces I/O overhead and runtime complexity. Critically, these approaches overlook a key characteristic of edge inference: strong spatial and temporal locality. Unlike general cloud workloads, edge requests typically arrive in semantically correlated streams. This observation raises a pivotal question: Can we leverage this locality to maintain only a compact, task-relevant expert subset in memory, performing expert replacement only when task semantics shift, thereby preserving model capability with minimal overhead?

To investigate this question, we conduct an empirical analysis of expert usage in MoE-LLM. Our analysis reveals that expert usage is simultaneously sparse, highly skewed, and partially redundant. (1) Expert activation is highly imbalanced despite explicit load balancing during training: the top 50% of experts account for over 90% of activations. (2) Removing low-frequency experts leads to significant performance degradation, indicating their non-negligible importance. (3) Experts exhibit substantial functional overlap, with some experts being highly similar and thus substitutable. (4) While different tasks favor different high-frequency experts, their expert subsets partially overlap, suggesting the existence of a shared expert core.

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 21124–21134
July 2-7, 2026 ©2026 Association for Computational Linguistics

These observations reveal both opportunities and challenges. On one hand, experts can be categorized by activation frequency to reduce the number of resident experts and lower the memory footprint. On the other hand, missing low-frequency experts can still cause severe accuracy loss, creating a fundamental tension between minimizing the resident expert set and preserving model performance. Moreover, task distributions may evolve over time, making static expert configurations suboptimal. This raises two challenges: how to mitigate expert misses without expensive I/O operations, and how to adapt to changing task distributions efficiently.

To address these challenges, we propose **LightMoE**, a task-aware expert availability management framework for memory-efficient MoE inference. LightMoE explicitly models expert availability as a primary concern during inference and comprises three components. First, a frequency-based initialization strategy categorizes experts into resident, replaceable, and removable groups, substantially reducing memory usage. Second, a similarity-aware expert redirection mechanism preserves accuracy when selected experts are unavailable. Third, a lightweight runtime expert manager performs task-level expert replacement based on recent activation patterns, enabling adaptation to task shifts with minimal overhead. Experimental results demonstrate that LightMoE achieves a better accuracy-efficiency trade-off, improving average accuracy by 4.3% over static pruning and 2.4% over dynamic swapping, while maintaining low latency comparable to parameter-reduced models.

In summary, this paper makes the following contributions:

- We propose LightMoE, a holistic framework that decouples logical expert selection from physical memory residency, enabling efficient MoE inference under strict edge constraints.
- We conduct an empirical analysis revealing substantial expert functional redundancy and task-specific locality, and introduce similarity-based expert redirection and hysteresis-aware task-level expert replacement mechanisms to ensure robust performance with a minimal resident set.
- We implement LightMoE on representative edge platforms and popular MoE-LLM, demonstrating the effectiveness of our method across diverse tasks.

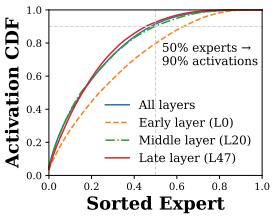

Figure 1: Layer-aware cumulative distribution frequencies of expert activation during MoE inference.

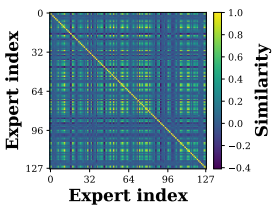

Figure 2: Heatmap of expert activation similarity across 128 experts in the MoE model.

2 Motivation and Data-Driven Analysis

To motivate the design of LightMoE, we first conduct a data-driven analysis of expert usage using Qwen3-30B-A3B (Yang et al., 2025) and different tasks as an example. Our goal is to understand whether the activation patterns expose opportunities for reducing memory footprint while maintaining model performance. Through a series of empirical observations, we identify three key insights that collectively suggest expert availability can be managed more efficiently at inference time.

2.1 Skewed and Layer-Dependent Expert Activation

We begin by analyzing expert activation frequencies during inference. Although MoE models are typically trained with explicit load-balancing objectives, we find that expert usage at inference time is highly skewed. Fig. 1 shows the cumulative distribution frequency of expert activations, where experts are sorted by their activation frequency. Surprisingly, the top 50% of experts account for over 90% of total activations, while the remaining experts are only rarely selected.

Meanwhile, this imbalance is not uniform across the network. As illustrated in Fig. 1, different layers exhibit varying degrees of skewness in expert activation. In some layers, activations are concentrated on an even smaller subset of experts, whereas some layers display a relatively flatter distribution. This layer-dependent behavior indicates that expert redundancy and importance are not consistent throughout the model, suggesting that a one-size-fits-all expert retention strategy may be suboptimal.

Insight 1. *For a given task distribution, only a compact and layer-dependent subset of experts is*

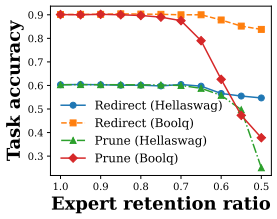

Figure 3: Impact of expert retention ratio and methods on different task accuracy.

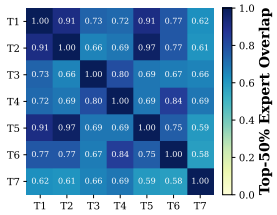

Figure 4: Task-to-task overlap of the top 50% most frequently selected experts.

frequently used and needs to remain resident in memory.

2.2 Low-Frequency Experts: Important yet Substitutable

The skewed activation distribution (Fig. 1) might suggest that rare experts can be removed to reduce memory, but our analysis refutes this intuition. Fig. 3 reveals that accuracy degradation is non-linear: performance is preserved at 30% pruning but collapses beyond 50%. This suggests that low frequency implies low importance only up to a point; beyond this redundancy threshold, experts often contribute critical capabilities that cannot be discarded. Inspired by existing work, researchers have found significant functional redundancy within the expert space (Lo et al., 2025; Dai et al., 2024). By analyzing the cosine similarity of the activation vector of different experts (Fig. 2), we find that most experts can find other experts with high similarity to them. This widespread functional overlap suggests that while low-frequency experts are important, their functionality might be approximated by other experts. As shown in Fig. 3, a simple similarity-based redirect strategy achieves higher accuracy than direct pruning.

Insight 2. *Low-frequency experts cannot be simply ignored due to their importance. However, leveraging functional redundancy allows us to substitute them by redirecting inputs to similar experts.*

2.3 Task-Specific Shared Core vs. Variability

While the previous analyses suggest that a compact expert subset suffices for a single task, a critical question remains: can a single fixed subset support the diverse tasks encountered in real-world deployments? We analyze expert activation patterns across multiple downstream tasks. Fig. 4 illustrates the overlap between task-specific high-frequency experts. We observe that while different tasks activate distinct expert subsets, they also exhibit non-trivial overlap. This indicates the existence of a *shared core* of experts that are universally useful across tasks. The coexistence of task-specific variability and a shared core implies that maintaining a static resident set is suboptimal, as it incurs frequent misses when the task shifts. In contrast, this structure suggests that expert sets can be adapted at a coarser granularity than tokens: by keeping the shared core resident and dynamically switching only a small number of task-relevant experts, the model can efficiently adapt to changing contexts.

Insight 3. *Task-level switching can minimize loading overhead by updating only a subset of experts when task semantics change, rather than at the token level.*

3 Method

Inspired by the insights mentioned above, we design the LightMoE, a task-aware expert availability management framework for the memory-efficient MoE inference. As shown in the Fig. 5, LightMoE consists of three modules. First, **Frequency-Aware Expert Selection** (Sec. 3.1) initializes the memory with a permanent core ($\mathcal{E}_s$) and a high-utility subset of replaceable experts ($\mathcal{E}_d$) based on offline profiling. Second, **Similarity-Based Expert Redirection** (Sec. 3.2) analyzes the expert similarity in the offline phase and redirects missing experts to their most similar and available experts in the online phase. Finally, **Runtime Expert Management** (Sec. 3.3) handles temporal distribution shifts by monitoring demand and asynchronously updating the replaceable $\mathcal{E}_d$ subset to match changing task contexts.

3.1 Frequency-Aware Expert Selection

To maximize model accuracy under a strict memory budget, we identify the optimal subset of experts via offline profiling. Using a calibration dataset with N tokens sampled from the training dataset, we calculate the activation frequency $f_{l,e}$ for each expert e in layer l:

$$f_{l,e} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left(e \in \mathcal{E}_l^{(i)}\right), \tag{1}$$

where $\mathcal{E}_l^{(i)}$ denotes the activated experts for token x_i. Experts are then ranked by frequency within each layer.

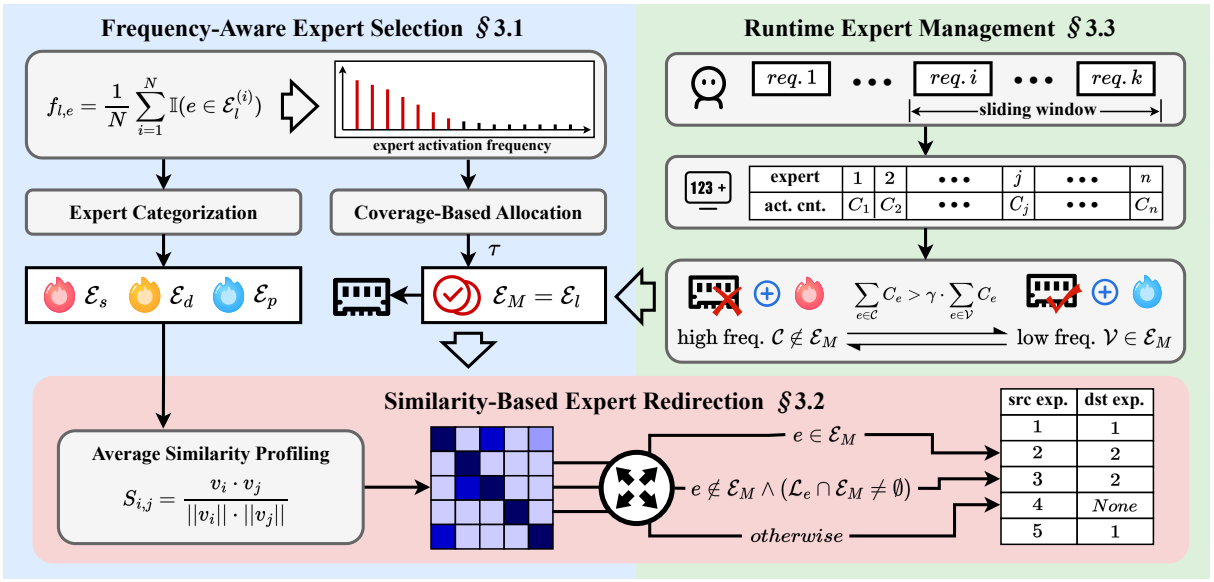

Figure 5: Framework Overview.

Coverage-Based Allocation. After obtaining the expert activation frequencies, we calculate a differentiated number of resident experts for each layer based on the global retention ratio. A fixed top-k policy causes accuracy degradation in layers with dispersed activation distributions shown in Fig. 1. Instead, we adopt a *cumulative coverage strategy*: for each layer, we retain the minimal set of top-ranked experts $\mathcal{E}_l$ whose summed frequency exceeds a global threshold δ (e.g., 90% of total activations). We tune δ so that the total count $\sum |\mathcal{E}_l|$ satisfies the global expert retention ratio. This dynamically allocates more experts to layers with a smoother activation distribution, balancing information preservation across the model.

Expert Categorization. Based on this allocation, experts are classified into three groups: (1) *Resident Experts* $\mathcal{E}_s$, pinned in memory to handle the majority of activation. (2) *Replaceable Experts* $\mathcal{E}_d$, non-resident experts with moderate frequency. They can be replaced between different tasks. (3) *Removable Experts* $\mathcal{E}_p$, which have negligible frequency in most tasks and are safely pruned. During inference, the experts in the memory $\mathcal{E}_M$ consist of the permanent resident experts set $\mathcal{E}_s$ and a part of the replaceable experts set $\mathcal{E}_d$ selected for the current task and the expert retention ratio.

3.2 Similarity-Based Expert Redirection

While frequency-aware categorization defines which experts are resident versus replaceable, the absence of specific $\mathcal{E}_d$ or $\mathcal{E}_p$ experts during inference may still degrade accuracy. To mitigate this, we propose *Similarity-Based Expert Redirection*, which dynamically redirects inputs from missing experts to their most functionally similar experts currently available in the memory.

Expert Similarity Profiling. We quantify expert functional similarity using the calibration dataset, the same as we mentioned in Sec. 3.1. For every pair of experts (i, j) in the same layer, we compute the average cosine similarity of their output activation vectors across the calibration samples. Let $\mathbf{v}_e \in \mathbb{R}^d$ denote the aggregated output activation vector of expert e. The pairwise similarity matrix $\mathbf{S}$ is computed as:

$$\mathbf{S}_{i,j} = \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\|\|\mathbf{v}_j\|}. \tag{2}$$

Based on $\mathbf{S}$, we construct a substitution list $\mathcal{L}_e$ for each expert e, containing other experts sorted by descending similarity scores, filtered by a minimum similarity threshold τ to ensure functional relevance.

Dynamic Redirection Logic. When the gate router $\mathcal{G}$ selects a target expert e, our redirection function $\Phi(e)$ maps it to an optimal destination within $\mathcal{E}_M$:

$$\Phi(e) = \begin{cases} e, & \text{if } e \in \mathcal{E}_M; \\ \arg\max_{k \in \mathcal{L}_e \cap \mathcal{E}_M} S_{e,k}, & \text{if } e \notin \mathcal{E}_M \\ & \quad \wedge (\mathcal{L}_e \cap \mathcal{E}_M \neq \emptyset); \\ \text{Next}(\mathcal{G}), & \text{otherwise.} \end{cases} \tag{3}$$

The first case handles *hit*, i.e., it is used directly if e is physically loaded. The second case performs *redirection*: if the target expert is unavailable (missing), we select the highest-ranked substitute that is currently in the memory $\mathcal{E}_M$. The third case is a *fallback*: if no sufficiently similar resident expert exists (i.e., $\mathcal{L}_e \cap \mathcal{E}_M$ is empty), we mask expert e and allow the gate $\mathcal{G}$ to select the next-best expert based on the original routing logits.

Since $\mathcal{E}_M$ typically changes only at task boundaries, the mapping $\Phi(\cdot)$ is updated infrequently. Thus, at inference time, redirection is reduced to a simple $O(1)$ table lookup, incurring negligible overhead while effectively mitigating the accuracy loss from missing experts.

3.3 Runtime Expert Management

While offline profiling establishes a solid baseline, real-world inference often encounters distribution shifts that static initialization cannot handle. To bridge this gap, we introduce the *Runtime Expert Manager* to dynamically adapt the available expert set $\mathcal{E}_M$ to online variations.

Online Utility Monitoring. We specifically target the replaceable subset $\mathcal{E}_d$, leaving the static core $\mathcal{E}_s$ untouched to ensure stability. It maintains a lightweight activation counter C_e for all experts (including those currently not in the memory) within a sliding window (e.g., 10 requests). Crucially, C_e records the original routing decisions rather than the redirected experts, ensuring that the counter tracks the true demand regardless of whether an expert is currently resident.

Hysteresis-Based Replacement. To prevent high I/O overhead caused by transient noise, we employ a robust batch replacement strategy. At regular intervals, we identify two expert groups per layer: the *Eviction Set* $\mathcal{V}$ (the k lowest-frequency experts in $\mathcal{E}_d$) and the *Candidate Set* $\mathcal{C}$ (the k highest-frequency experts currently not in memory). A replacement is triggered only if the candidates' collective utility significantly exceeds the eviction by a stability margin γ:

$$\sum_{e \in \mathcal{C}} C_e > \gamma \cdot \sum_{e \in \mathcal{V}} C_e, \tag{4}$$

where $\gamma > 1.0$ acts as a hysteresis factor. This ensures that updates occur only in response to sustained semantic shifts rather than sporadic outliers.

Asynchronous Update Loop. Once triggered, we can execute an asynchronous update pipeline to close the adaptation loop. It pre-fetches the parameters of $\mathcal{C}$ to overwrite $\mathcal{V}$ and simultaneously updates the redirection map $\Phi(\cdot)$ (defined in Eq. 3). This ensures that subsequent requests immediately utilize the newly loaded experts or find updated redirection experts for the displaced ones, maintaining high accuracy with low runtime overhead.

4 Evaluation

4.1 Experimental Setup

Platforms. We conduct experiments on two distinct hardware platforms to represent different deployment scenarios: (1) NVIDIA A100 (80GB), representing a high-performance edge server environment; and (2) NVIDIA Jetson AGX Orin (64GB), representing a resource-constrained embedded edge device. To simulate real-time streaming inference common in edge applications, the batch size is set to 1 for all experiments.

Models and Benchmarks. We evaluate our framework on two representative MoE models with varying scales and the number of experts: Qwen3-30B-A3B (Yang et al., 2025), a large-scale model with 128 experts per layer, and OLMoE-1B-7B (Muennighoff et al., 2024), a compact model with 64 experts per layer. These models cover a wide range of parameter sizes and sparsity levels. We employ the EleutherAI LM Harness (Gao et al., 2024) library to conduct comprehensive testing across eight diverse datasets covering common sense reasoning, reading comprehension, and mathematical solving: OpenBookQA (Mihaylov et al., 2018), ARC-Easy, ARC-Challenge (Clark et al., 2018), Winogrande (ai2, 2019), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), BoolQ (Christopher et al., 2019), and GSM8K (Cobbe et al., 2021). We report the zero-shot accuracy for most tasks (5-shot for GSM8K) to measure the model's generalization capability.

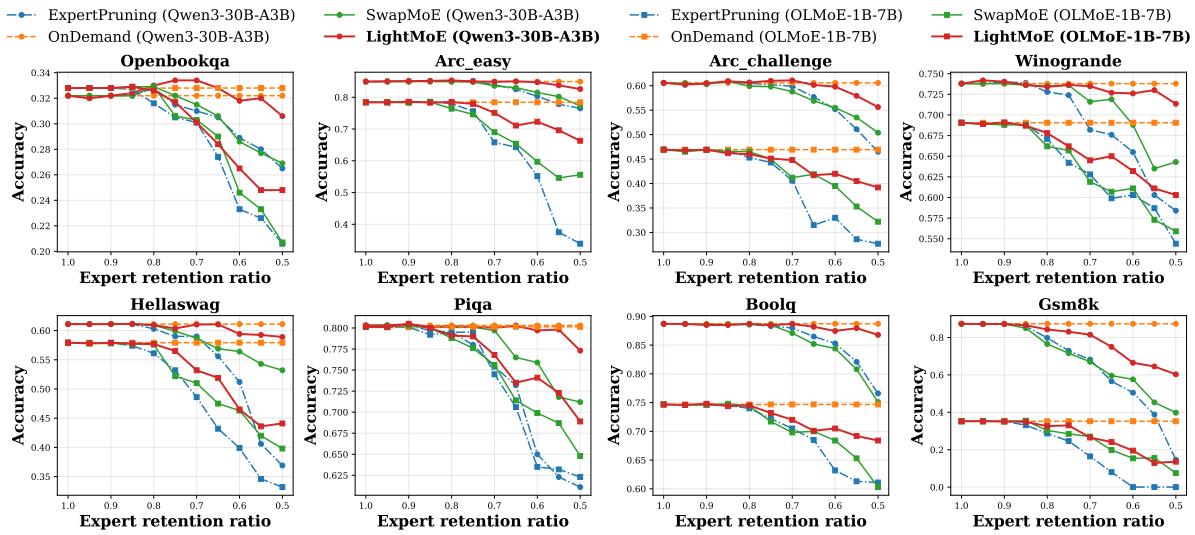

Figure 6: Performance degradation analysis under memory constraints. LightMoE effectively preserves accuracy compared to ExpertPruning and SwapMoE as the expert retention ratio decreases from 1.0 to 0.5.

Baselines. We compare our proposed method against the original MoE model and three competitive baselines: (1) Expert Pruning: A static compression method adapted from (Lu et al., 2024), which retains a fixed budget of experts by employing a layer-wise reconstruction loss minimization strategy to identify the most critical experts, permanently removing the rest. (2) On-Demand Loading: A naive offloading strategy where experts are stored in host memory and fetched to the GPU synchronously upon activation, representing the lower bound of dynamic systems. (3) SwapMoE (Kong et al., 2024): A dynamic inference method with an asynchronous expert swapping mechanism that overlaps expert transfer with computation, enabling token-level expert replacement with minimal latency overhead. (4) MoE-Infinity (Xue et al., 2024): A lossless dynamic inference approach that preserves the original MoE computation by loading missing experts on demand. It leverages expert prediction and token-level caching to reduce I/O frequency, making it conceptually similar to an optimized on-demand loading strategy. Comparison with these baselines demonstrates the efficacy of our method in balancing memory footprint, latency, and accuracy.

4.2 Main Results

Accuracy in different expert retention ratios. We evaluate different expert management strategies under memory constraints by simulating an edge inference scenario with a dynamic request stream, where only a subset of experts can be retained in memory. We gradually reduce the expert retention ratio from 1.0 to 0.5 and measure the resulting accuracy degradation across eight benchmarks.

As shown in Fig. 6, On-Demand expert loading preserves the original computation by fetching missing experts on demand, and thus achieves the same accuracy as the original model across all retention ratios. However, it incurs substantial latency overhead due to frequent expert loading and therefore mainly serves as an accuracy upper bound. Meanwhile, expert pruning methods maintain reasonable accuracy at high retention ratios by removing redundant experts, but suffer from increasing accuracy degradation as more informative experts are pruned under higher compression. SwapMoE improves over static pruning by dynamically updating active experts and achieves better accuracy under moderate compression. When the retention ratio becomes low, however, its performance degrades rapidly because many requested experts remain unavailable within each update interval and are mapped to virtual experts, introducing approximation errors.

In contrast, LightMoE consistently achieves the best accuracy under constrained memory. By redirecting the missing experts to the most semantically similar available ones, LightMoE effectively alleviates expert unavailability while preserving semantic relevance. Consequently, LightMoE consistently outperforms the baselines across most settings, achieving an average accuracy improvement of 4.3% over ExpertPruning and 2.4% over SwapMoE across all models, datasets, and expert reten-

Method	Memory ↓	Latency ↓	Accuracy ↑
50% Expert Retention-A100-Qwen3			
On-demand		3.57×	0.711
MoE-Infinity		2.86×	0.711
ExpertPruning	0.53×	0.65×	0.496
SwapMoE		0.71×	0.573
LightMoE		**0.73**×	**0.654**
70% Expert Retention-AGX Orin-OLMoE			
On-demand		2.32×	0.594
MoE-Infinity		1.96×	0.594
ExpertPruning	0.74×	0.82×	0.512
SwapMoE		0.89×	0.532
LightMoE		**0.91**×	**0.554**

Table 1: Memory, latency, and accuracy comparison. LightMoE achieves the better accuracy-latency trade-off under the different devices, models and memory constraints.

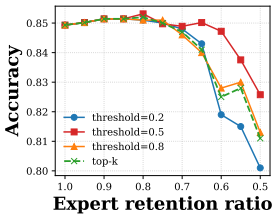

Figure 7: Impact of expert retention ratio and similarity threshold on task accuracy.

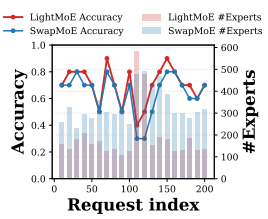

Figure 8: The accuracy and stability of runtime expert management under task shifts.

tion ratios. When averaging accuracy across all retention ratios for each model and dataset pair, LightMoE improves up to a 9.0% absolute accuracy improvement on GSM8K.

We further observe that baseline methods suffer more severe degradation on reasoning-intensive benchmarks such as GSM8K, whereas LightMoE remains robust even at low retention ratios. Moreover, accuracy drops more rapidly on the smaller OLMoE than on Qwen3 as retention decreases. We attribute this to lower over-parameterization and more uniform expert utilization in smaller models, which makes them inherently more sensitive to expert compression.

Accuracy and latency trade-off. We further evaluate the overhead of different methods. Table 1 presents the system performance comparison across different devices, models, and compression ratios. Memory and latency are reported as ratios relative to the original model, while accuracy is averaged across all evaluated datasets.

As shown in the Table 1, although the on-demand method loading serves as an accuracy upper bound, its reliance on frequent I/O operations for fetching missing experts results in prohibitive latency overhead (3.57× on A100), making it impractical for real-time edge deployment. Conversely, the ExpertPruning method achieves the lowest latency by permanently discarding parameters, but this comes at the cost of severe accuracy degradation. SwapMoE offers a middle ground but still suffers from accuracy loss due to the unavailability of experts between update intervals. Notably, LightMoE achieves the most favorable trade-off. By leveraging *similarity-based redirection* to compensate for missing experts rather than incurring expensive I/O, LightMoE significantly outperforms SwapMoE in accuracy (e.g., +8.1% absolute gain on Qwen3) while maintaining a low latency overhead. This confirms that our approach effectively bridges the gap between memory efficiency and model performance without introducing the latency bottlenecks typical of dynamic offloading systems.

4.3 Analysis of LightMoE

Impact of similarity threshold. In Sec. 3.2, we used similarity-based redirection to select experts, and we further explored the impact of the setting of the similarity threshold on the results. As shown in Fig. 7, LightMoE consistently outperforms the Top-k baseline because the latter simply selects the next-ranked expert based on routing logits without considering functional similarity, sometimes leading to semantic mismatch. We further observe that the threshold τ controls a critical trade-off: setting τ too high (e.g., 0.8) prevents finding the valid substitute experts, causing the strategy to degenerate into Top-k, while setting it too low (e.g., 0.2) introduces noise from dissimilar experts. Empirically, $\tau = 0.5$ achieves the optimal accuracy by effectively balancing substitute quality and coverage.

Impact of runtime expert management. We evaluate the stability of runtime expert management under a task-shifted inference workload. We construct a streaming sequence of 200 requests, with a task change occurring at the 100th request, and report rolling accuracy and expert replacement counts over a sliding window of 10 requests. As shown in the Fig. 8, LightMoE responds more promptly to task changes and recovers accuracy faster after the shift. Within a stable task segment, expert redirection effectively compensates for miss-

Variant	M1	M2	M3	Avg. Acc. ↑
LightMoE	✓	✓	✓	0.654
w/o M1	×	✓	✓	0.632
w/o M2	✓	×	✓	0.621
w/o M3	✓	✓	×	0.627

Table 2: Ablation study of LightMoE under a fixed expert retention ratio of 50%. M1: frequency-aware expert selection; M2: similarity-based expert redirection; M3: runtime expert management.

ing experts, resulting in both higher accuracy and stable expert usage with infrequent replacements. In contrast, SwapMoE performs layer-wise expert swapping, incurring approximately 70% more expert replacements on average and exhibiting slower adaptation to task changes. These results demonstrate the advantage of task-level expert availability management in achieving efficient and stable adaptation.

4.4 Ablation Study

We demonstrated the effectiveness of the proposed module through an ablation study. Under a fixed expert retention ratio of 50%. To isolate the contribution of each component, we replace one module at a time with a reasonable baseline design. Specifically, for M1, we replace frequency-aware expert selection with a norm-based initialization strategy that selects experts according to the ℓ_2 norm of their weights. For M2, we disable similarity-based redirection and revert to the original top-k routing behavior, where missing experts are not redirected. For M3, we remove runtime expert management and adopt a static expert set determined offline without online adaptation.

As shown in the Table 2, removing any individual module leads to a consistent accuracy degradation, indicating that each component contributes positively to overall performance. In particular, disabling similarity-based redirection (w/o M2) results in the largest drop, highlighting the importance of handling low-frequency expert misses during inference. These results demonstrate that the three components are complementary and jointly enable memory-efficient and accurate MoE inference.

5 Related Work

MoE Compression and Sparsity. Several studies aim to reduce the memory footprint of MoE models by pruning or restructuring experts (Zhou et al., 2025). Lu et al. (Lu et al., 2024) propose post-training expert pruning and skipping based on routing statistics, removing or skipping low-usage experts to reduce model size with limited accuracy degradation. Lee et al. (Lee et al., 2025) propose STUN, a one-shot structured pruning method that removes experts and weights simultaneously at inference time. He et al. (He et al., 2023) merge multiple experts into a single network to eliminate redundancy, but require architectural modification and retraining. These methods permanently alter the expert set and may degrade performance when rare but important experts are removed, whereas our approach preserves the original model and avoids irreversible pruning.

MoE Experts Offloading. Another line of work improves MoE inference efficiency through expert loading and swapping across devices. Huang et al. (Huang et al., 2024a) propose expert buffering and dynamic gating to cache frequently used experts on GPU and offload others to CPU. Kamahori et al. (Kamahori et al., 2024) present FIDDLER, a system that dynamically schedules experts across CPU and GPU to hide data transfer latency. SwapMoE (Kong et al., 2024) proposes an asynchronous expert swapping mechanism that overlaps expert transfer with computation, enabling token-level expert replacement with minimal latency overhead. These approaches rely on accurate prediction of future expert usage and often operate at token-level granularity, which can incur nontrivial I/O overhead. In contrast, our method reduces expert transfers by redirecting missing experts to similar resident ones and performs expert switching at the task or request level.

Adaptive and Efficient MoE Routing. Several studies improve MoE inference efficiency by optimizing expert routing or modifying MoE architectures (Huang et al., 2024b). Yue et al. (Yue et al., 2024) propose Ada-K routing, which uses a RL–based allocator to determine the number of experts per token dynamically. Longkai et al. (Longkai et al., 2025) introduce HookMoE, inserting lightweight compensation modules before MoE layers to alleviate accuracy degradation when fewer experts are activated. Zhong et al. (Zhong et al., 2024) present AdapMoE, a framework that adaptively adjusts expert activation via sensitivity analysis and reduces inference overhead through prefetching and cache management. While these methods enhance efficiency by dynamically con-

trolling expert activation, the overall memory footprint remains unchanged. Our work is complementary to this line of research, as it focuses on expert availability and memory footprint rather than activation control.

6 Conclusion

In this paper, we presented LightMoE, a framework enabling memory-efficient MoE inference on edge devices. By combining frequency-aware expert selection, similarity-based redirection for missing experts, and a runtime expert replacement mechanism, LightMoE effectively decouples model capacity from strict memory constraints. Experimental results demonstrate that LightMoE achieves a better accuracy-efficiency trade-off, improving average accuracy by 4.3% over static pruning and 2.4% over dynamic swapping. This work provides a practical solution for deploying large-scale MoE models in resource-constrained environments.

7 Limitations

First, LightMoE assumes the existence of functional overlap among experts to enable effective similarity-based redirection when selected experts are unavailable. While our empirical analysis confirms substantial redundancy in the evaluated MoE models, this assumption may be weaker for architectures with highly specialized experts or alternative training objectives in the future. Second, LightMoE is designed for edge inference scenarios with correlated and slowly evolving request streams. In settings with highly dynamic or adversarial task distributions, task-level expert replacement may need to occur more frequently, which could diminish the benefits of maintaining a small resident expert set.

Ethical Considerations

Potential Risks. This work focuses on improving the deployment efficiency of Mixture-of-Experts (MoE) large language models via expert-level sparsification, routing optimization, and runtime expert management. The methods involve modifications to the model and its experts for efficiency, but do not change the underlying training data or the semantic behavior of the model, and thus do not introduce new ethical risks beyond those already associated with large language models. Nonetheless, real-world deployment, especially on edge or on-device, requires careful consideration of privacy, responsible use, and mitigation of pre-existing model biases. Edge deployment can reduce cloud data transmission, but improper design may still expose sensitive information.

Information About Use of AI Assistants. This work utilized AI assistants (e.g., ChatGPT) solely for the purposes of language polishing, grammar correction, and refining the clarity of the text. All scientific ideas, experimental designs, data analyses, and the final manuscript content are the original work of the authors and have been manually verified.

Data Privacy and Content Safety. The datasets employed in this study (including OpenBookQA, ARC, Winogrande, HellaSwag, PIQA, BoolQ, and GSM8K) are all standard, publicly available benchmarks widely established in the NLP community. We did not curate or release any new datasets. Consequently, issues regarding personally identifying information (PII) or offensive content are consistent with those of the original open-source releases, and no additional anonymization was performed.

Acknowledgments

The research is partially supported by Quantum Science and Technology-National Science and Technology Major Project (QNMP) 2021ZD0302900 and China National Natural Science Foundation with No.92567301, 62132018, 62231015, "Pioneer" and "Leading Goose" R&D Program of Zhejiang", 2023C01029, and 2023C01143 and USTC Kunpeng-Ascend Scientific and Education Innovation Excellence Center.

References

2019. Winogrande: An adversarial winograd schema challenge at scale.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Clark Christopher, Lee Kenton, Chang Ming-Wei, Kwiatkowski Tom, Collins Michael, and Toutanova Kristina. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*.

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. *arXiv:1803.05457v1*.

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. *arXiv preprint arXiv:2110.14168*.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, and 1 others. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*.

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.

Zhongqian Fu, Ning Ding, Kai Han, Xianzhi Yu, Xiaosong Li, Xinghao Chen, Yehui Tang, and Yunhe Wang. 2025. Eaquant: Enhancing post-training quantization for moe models via expert-aware optimization. *arXiv preprint arXiv:2506.13329*.

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.

Shwai He, Run-Ze Fan, Liang Ding, Li Shen, Tianyi Zhou, and Dacheng Tao. 2023. Merging experts into one: Improving computational efficiency of mixture of experts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14685–14691, Singapore. Association for Computational Linguistics.

Haiyang Huang, Newsha Ardalani, Anna Sun, Liu Ke, Hsien-Hsin S Lee, Shruti Bhosale, Carole-Jean Wu, and Benjamin Lee. 2024a. Toward efficient inference for mixture of experts. *Advances in Neural Information Processing Systems*, 37:84033–84059.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024b. 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, Bangkok, Thailand. Association for Computational Linguistics.

Keisuke Kamahori, Tian Tang, Yile Gu, Kan Zhu, and Baris Kasikci. 2024. Fiddler: Cpu-gpu orchestration for fast inference of mixture-of-experts models. *arXiv preprint arXiv:2402.07033*.

Rui Kong, Yuanchun Li, Qingtian Feng, Weijun Wang, Xiaozhou Ye, Ye Ouyang, Linghe Kong, and Yunxin Liu. 2024. Swapmoe: Serving off-the-shelf moe-based large language models with tunable memory budget. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6710–6720.

Jaeseong Lee, Seung-won Hwang, Aurick Qiao, Daniel F Campos, Zhewei Yao, and Yuxiong He. 2025. Stun: Structured-then-unstructured pruning for scalable moe pruning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13660–13676.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

Pingzhi Li, Xiaolong Jin, Yu Cheng, and Tianlong Chen. 2024. Examining post-training quantization for mixture-of-experts: A benchmark.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Huaping Liu, Di Guo, and Angelo Cangelosi. 2025. Embodied intelligence: A synergy of morphology, action, perception and learning. *ACM Computing Surveys*, 57(7):1–36.

Ka Man Lo, Zeyu Huang, Zihan Qiu, Zili Wang, and Jie Fu. 2025. A closer look into mixture-of-experts in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4427–4447.

Cheng Longkai, Along He, Mulin Li, Xie Xueshuo, and Tao Li. 2025. HookMoE: A learnable performance compensation strategy of mixture-of-experts for LLM inference acceleration. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 31582–31594, Suzhou, China. Association for Computational Linguistics.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. 2024. Not all experts are equal: Efficient expert pruning and skipping for mixture-of-experts large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6159–6172, Bangkok, Thailand. Association for Computational Linguistics.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, and 5 others. 2024. Olmoe:

Open mixture-of-experts language models. *Preprint*, arXiv:2409.02060.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*.

Leyang Xue, Yao Fu, Zhan Lu, Luo Mai, and Mahesh Marina. 2024. Moe-infinity: Efficient moe inference on personal machines with sparsity-aware expert cache. *arXiv preprint arXiv:2401.14361*.

Leyang Xue, Yao Fu, Zhan Lu, Luo Mai, and Mahesh Marina. 2025. Moe-infinity: Efficient moe inference on personal machines with sparsity-aware expert cache. *Preprint*, arXiv:2401.14361.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Rongjie Yi, Liwei Guo, Shiyun Wei, Ao Zhou, Shangguang Wang, and Mengwei Xu. 2025. Edgemoe: Empowering sparse large language models on mobile devices. *IEEE Transactions on Mobile Computing*.

Tongtian Yue, Longteng Guo, Jie Cheng, Xuange Gao, Hua Huang, and Jing Liu. 2024. Ada-k routing: Boosting the efficiency of moe-based llms. In *The Thirteenth International Conference on Learning Representations*.

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*.

Shuzhang Zhong, Ling Liang, Yuan Wang, Runsheng Wang, Ru Huang, and Meng Li. 2024. 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*, pages 1–9.

Yixiao Zhou, Ziyu Zhao, Dongzhou Cheng, Zhiliang Wu, Jie Gui, Yi Yang, Fei Wu, Yu Cheng, and Hehe Fan. 2025. Dropping experts, recombining neurons: Retraining-free pruning for sparse mixture-of-experts LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 15169–15186, Suzhou, China. Association for Computational Linguistics.