It Takes a MAESTRO To Prune Bad Experts

Anonymous ACL submission

Abstract

Sparsely-activated Mixture-of-Experts (MoE) language models achieve remarkable inference efficiency by activating only a small fraction of parameters per token, yet their full expert banks reside in memory at all times, creating a prohibitive deployment bottleneck. Existing structured pruning methods, largely designed for dense transformers, assess expert importance using locally derived heuristics that are blind to the interdependent nature of MoE routing. We introduce MAESTRO (**M**arkov-chain **A**proximated **E**xpert **S**parsification via **T**ransition-based **RO**uting), a structured pruning framework designed for MoE architectures that models autoregressive expert activation trajectories as Ergodic Markov chains whose stationary distributions encode cross-layer dependencies, yielding a globally aware importance heuristic. Evaluated across five diverse domains including Safety, Bias, and Ethics, MAESTRO outperforms state-of-the-art baselines by up to 10.61% in average performance retention under a strict 50% compression regime, while exhibiting substantially lower cross-task variance, indicating that global, routing-congruent pruning produces models that generalize more consistently across heterogeneous tasks.

1 Introduction

Sparsely-activated Mixture-of-Experts (MoE) architectures have emerged as the dominant recipe for scaling large language models without a commensurate increase in inference compute (Jiang et al., 2024; DeepSeek-AI, 2025; Yang et al., 2025; OpenAI et al., 2025; NVIDIA et al., 2025). By replacing the dense feed-forward block of each transformer layer with a bank of E expert sub-networks and a learned router that dispatches each token to only $k \ll E$ of them, MoE models decouple parameter count from per-token FLOPs. Such models routinely instantiate tens to hundreds of billions of parameters while activating only a few billion per token, yielding inference throughput comparable to far smaller dense models.

This same design choice, however, is the source of a sharply contradictory deployment profile. While MoE models are *computationally* efficient at inference time, they remain *spatially* prohibitive: the full bank of experts must reside in memory, regardless of how few are selected for any given token. This results in a deployment bottleneck wherein state-of-the-art MoE models do not fit on commodity accelerators, are expensive to host even on data-centre hardware, and are effectively excluded from on-device, edge, and memory-constrained settings where their per-token compute profile would otherwise be ideal. Closing the gap between activated and total parameters is therefore a central practical problem for the deployment of modern MoE LLMs.

A natural response is structured pruning: permanently removing parameters from the model so that both compute and memory drop simultaneously. The pruning literature on large language models (LLMs), however, has overwhelmingly targeted *dense* transformers (Ma et al., 2023; Ashkboos et al., 2024; Men et al., 2025; Guo et al., 2025; Shopkhoev et al., 2026; Song et al., 2024; Sengupta et al., 2025). Methods such as magnitude- (Han et al., 2016), activation- (Sun et al., 2024), and gradient-based pruning (Ma et al., 2023; Frantar et al., 2022), layer dropping (Shopkhoev et al., 2026; Song et al., 2024; Men et al., 2025), and width reduction (Ma et al., 2023; Ashkboos et al., 2024; Sengupta et al., 2025) are designed around the assumption that every parameter participates in every forward pass and can therefore be scored by global, token-averaged signals. These assumptions are blind to the routing behavior and long-range inter-expert dependencies that exist within MoE models and thus, do not transfer cleanly to them. Additionally, prior MoE-centric efforts tend to rely on local, layer-isolated heuristics such as marginal

performance degradation upon the removal of a subset of experts (Lu et al., 2024a) or router-weighted activations produced at each layer (Xie et al., 2024). Such criteria treat each MoE layer and/or expert independently and ignore the fact that routing decisions are correlated across layers. An expert that is moderately used at layer ℓ may, in practice, only be reached by tokens that subsequently visit a rarely-used expert at layer $\ell+1$; its true contribution to the model's behavior is a property of the joint distribution over routing trajectories, not of any single layer's marginal. A principled importance signal for MoE pruning should therefore capture how routing flows through the entire stack of expert banks, not merely how often each expert is hit in isolation.

In this work, we introduce such a signal in the form of MAESTRO, a framework for **M**arkov-chain **A**pproximated **E**xpert **S**parsification via **T**ransition-based **RO**uting. We model the sequence of $(\texttt{layer}, \texttt{expert})$ pairs visited by a token as a Markov chain over all expert slots of an MoE model, with transitions estimated empirically over a small calibration corpus in an autoregressive manner. This yields a single, time-homogeneous kernel whose stationary distribution reflects global inter-expert dependencies and assigns each expert slot a long-run mass: the fraction of routing time the model spends in that slot under its own generation distribution. Experts with the smallest stationary mass are, by construction, the least frequently visited across all routing trajectories and are the natural candidates for removal.

By leveraging global routing trajectories that arise in MoE models, our proposed framework avoids making locally myopic pruning decisions and instead grounds expert importance in the model's own autoregressive generation behaviour, capturing joint dependencies across the entire architecture that per-layer heuristics structurally cannot.

Contributions. Our contributions are as follows.

- We propose a novel pruning framework, MAESTRO, that formalizes expert-level routing in MoE LLMs as a cyclic Markov chain over states of the form $(\texttt{layer}, \texttt{expert})$.
- We employ the chain's stationary distribution as a theoretically-grounded expert importance heuristic that is congruent with global routing decisions, thus yielding a pruning mechanism that directly incorporates cross-layer dependencies instead of computing pruned configurations from locally derived signals.
- We demonstrate that MAESTRO consistently outperforms current state-of-the-art baselines by up to 10.61% on average performance retention even under a strict 50% compression regime, while demonstrating notably low per-task standard deviation. Our evaluations are conducted on a comprehensive set of 17 benchmarks spanning five diverse domains, including Safety, Bias, and Ethics, across two contemporary MoE families: GPT-OSS and Qwen 3.

2 Related Works

As LLMs continue to scale, their computational demands have grown substantially due to architectural over-parameterization (Frankle and Carbin, 2019; Michel et al., 2019; Ma et al., 2023; K et al., 2025). While techniques such as quantization (Bhandare et al., 2019; Yao et al., 2022; Frantar et al., 2023; Liu et al., 2024) and run-time activation sparsity (Liu et al., 2025; Wang et al., 2025; Luo et al., 2025; Hou et al., 2026) reduce memory and latency costs, they leave the underlying structural redundancy intact. Model pruning addresses this directly. Unstructured methods achieve excellent performance retention at high sparsity ratios (Sun et al., 2024; Frantar and Alistarh, 2023), but sparsity alone does not yield practical speedups without hardware-specific sparse kernels. This has motivated a shift toward structured pruning, which removes contiguous parameter groups such as neurons (Ma et al., 2023; Ashkboos et al., 2024; Sengupta et al., 2025), MLP blocks (Zhong et al., 2025; Zhang et al., 2024), attention heads (Michel et al., 2019; Chen et al., 2025a; Zayed et al., 2024), or entire layers (Men et al., 2025; Song et al., 2024; Shopkhoev et al., 2026), yielding models that are smaller, leaner, and faster without specialized kernel support.

Despite these advances, progress has been largely confined to dense Transformer architectures. Many existing methods either lack support for MoE models entirely or naively transpose dense pruning heuristics onto MoE architectures. For instance, MoE-Pruner (Xie et al., 2024) extends Wanda's weight-magnitude-times-activation heuristic (Sun et al., 2024) to MoE models by simply multiplying it with router weights. While this incorporates a MoE-specific signal, it overlooks the deeper structural properties and inter-component dependencies that arise from an expert-centric design. This has motivated the emergence of MoE-centric pruning methods (Lu et al., 2024b; Lee et al., 2025; Hu

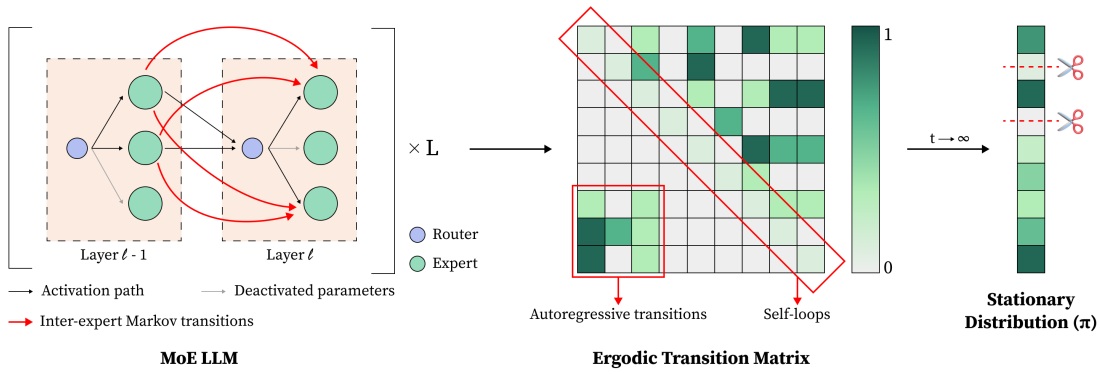

Figure 1: Schematic of MAESTRO's methodology. MAESTRO visualizes expert routing in MoE LLMs in the form of an Ergodic Markov chain and uses its stationary distribution as a global heuristic to remove redundant experts from each layer.

et al., 2026; Lasby et al., 2026) that propose heuristics such as activation variability scores (Hu et al., 2026) and router-weighted expert norms (Lasby et al., 2026) to better capture architectural nuances. While effective, these methods suffer from a critical drawback: they use locally derived heuristics to drive global pruning decisions. We hypothesize that such myopic heuristics are bound to overlook important global dependencies, leading to concessions in post-compression performance.

Beyond pruning, expert merging has emerged as a complementary strategy for compressing MoE models. Rather than discarding experts outright, merging methods (He et al., 2023; Chen et al., 2025b; Li et al., 2024; Muqeeth et al., 2024) combine redundant experts into a smaller set, preserving the collective knowledge encoded across them. For instance, MEO (He et al., 2023) employs router-weighted averaging of expert weights to merge experts based on routing similarity. In contrast, HC-SMoE (Chen et al., 2025b) leverages hierarchical clustering to merge functionally similar experts. While effective, Lasby et al. (2026) have recently shown that expert merging methods introduce an *irreducible error* due to loss of fine-grained router control, serving as a strong motivator for exploring increasingly advanced pruning strategies.

3 Methodology

Figure 1 displays our proposed calibration-driven procedure for structurally pruning MoE language models. The central idea is to formulate top-k expert routing within the model as a Markov chain over all states of the form $(\texttt{layer}, \texttt{expert})$ and to identify and remove redundant experts as those whose steady-state probabilities under this chain are smallest. Since expert tensors are physically sliced rather than masked at inference-time, the resulting model is strictly smaller in overall parameter count and memory footprint.

3.1 Preliminaries and Problem Setup

Let $\mathcal{M}$ denote a MoE model with L layers, each containing a set of E experts $\{f_{\ell,1}, \ldots, f_{\ell,E}\}$ and a router $g_\ell : \mathbb{R}^d \to \mathbb{R}^E$. For an input token representation $h \in \mathbb{R}^d$ at layer ℓ, the router produces routing weights r_ℓ where the routing weight for expert e is computed as follows:

$$r_\ell^{(e)} = \frac{\exp\left[g_\ell(h)_e\right]}{\sum_{e'} \exp\left[g_\ell(h)_{e'}\right]}, \quad 1 \le e' \le E \tag{1}$$

The router's top-k operator selects an index set $\mathcal{E}_\ell(h) \subset \{1, \ldots, E\}$ with $|\mathcal{E}_\ell(h)| = k$. The layer output, A_ℓ, is then the weighted sum over the selected experts, using the routing weights from Equation 1 re-normalized over $\mathcal{E}_\ell(h)$. Formally,

$$A_\ell = \sum_{e \in \mathcal{E}_\ell(h)} \hat{r}_\ell^{(e)}(h)\, f_{\ell,e}(h) \tag{2}$$

$$\text{where} \quad \hat{r}_\ell^{(e)}(h) = \frac{r_\ell^{(e)}}{\sum_{e' \in \mathcal{E}_\ell(h)} r_\ell^{(e')}} \tag{3}$$

Given a target compression ratio $\rho \in (0, 1)$, our objective is to select a set $\mathcal{R} \subset \{(\ell, e) : 1 \le \ell \le L,\ 1 \le e \le E\}$ of $K = \lfloor \rho \cdot LE \rfloor$ expert slots to delete from $\mathcal{M}$, yielding a pruned model $\tilde{\mathcal{M}}$.

3.2 Routing as a Markov Chain over Experts

Let $\mathcal{S} = \{(\ell, e) : 1 \le \ell \le L,\ 1 \le e \le E\}$ index the LE expert slots in the model. As a token traverses the network, it activates a sequence of states in $\mathcal{S}$. Inspired by Markovian principles, we propose

that, conditioned on the experts selected at layer ℓ, the experts selected at layer $\ell{+}1$ are governed by a transition kernel that depends only on the layer index and the current selection. This memoryless assumption, motivated by the fact that the router at layer $\ell{+}1$ consumes only the residual stream produced by layer ℓ, induces a Markov chain over $\mathcal{S}$. In order to derive a single steady-state (stationary) probability distribution of this chain, we induce Ergodicity in it, *i.e.*, ensure irreducibility (all states are reachable) and aperiodicity (not all paths between any two states have the same period). While the former is achieved by connecting the final layer back to the first layer, the latter is the resultant of a smoothing technique that applies a small self-loop to each state. The resulting chain admits a unique stationary distribution π, which can be interpreted as the long-run probability with which routing visits each expert slot. Experts whose stationary mass is smallest contribute least to overall computation and are therefore the natural targets for removal. Since π is derived from long token trajectories across the model's layers, it encodes long range inter-expert dependencies and structural routing patterns which are often invisible to local heuristics.

3.3 Calibration and Transition-Count Estimation

We estimate the per-layer transition kernels empirically from a small calibration corpus. For each calibration batch, the model is run *autoregressively* for T_{seq} generation steps, with each predicted token appended to the rolling context. Notably, in contrast to teacher-forced passes, autoregressive rollouts ensure that transition statistics reflect the distribution the model itself induces over expert activations during inference.

Let $\mathbf{e}_\ell \in \mathbb{Z}^{B\times T_{\text{seq}}\times k}$ represent the top-k index tensor produced by the router at layer ℓ. We accumulate pairwise transition counts from experts in layers $\ell-1$ to those in layer ℓ into a per-layer matrix $C^{(\ell)} \in \mathbb{R}^{E\times E}$. Specifically, $C^{(\ell)}_{ij}$ counts the number of token-steps for which expert i was selected at layer $\ell{-}1$ and expert j was selected at layer ℓ. Formally,

$$C^{(\ell)}_{ij} = \sum_{b=1}^{B}\sum_{t=1}^{T_{\text{seq}}} \mathbf{1}\Big[i \in \mathbf{e}^{(b,t)}_{\ell-1}\Big] \cdot \mathbf{1}\Big[j \in \mathbf{e}^{(b,t)}_{\ell}\Big], \quad (4)$$

where the sum ranges over all *non-padding* tokens (b,t) across the batch. These per-layer matrices are accumulated across all batches and autoregressive steps and cached to disk so subsequent runs can re-use the calibration statistics.

3.4 Global Transition Matrix and Stationary Distribution

We combine the per-layer count matrices into a single global matrix $\mathbf{T}$, which we partition into an $L\times L$ grid of $E\times E$ blocks. Mathematically,

$$\mathbf{T}[\ell,\,\ell'] = \begin{cases} C^{(\ell')} & \text{if } \ell' = \ell+1 \bmod L \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where $\mathbf{T}[\ell,\ell']$ represents the block at block-index (ℓ,ℓ'). More explicitly,

$$\mathbf{T} = \begin{pmatrix} 0 & C^{(1)} & & \\ & 0 & \ddots & \\ & & \ddots & C^{(L-1)} \\ C^{(0)} & & & 0 \end{pmatrix} \quad (6)$$

The final block $\mathbf{T}[L{-}1,\,0] = C^{(0)}$ closes the cycle. The autoregressive nature of the model, wherein the token generated at the last layer is fed to the first layer at the next timestep, makes this cyclic closure a natural choice, allowing us to capture inter-expert transitions based on the model's inherent distribution and behavior.

We then row-normalize $\mathbf{T}$ to obtain a stochastic matrix $\mathbf{P}$:

$$\mathbf{P}_{ij} \;=\; \frac{\mathbf{T}_{ij}}{\max\!\big(1,\,\sum_{j'}\mathbf{T}_{ij'}\big)}, \quad (7)$$

where the denominator is clipped to one to handle expert slots that are never activated on the calibration set. The empirical chain, while irreducible due to its autoregressive modeling, is periodic. To tackle this, we apply an ε-smoothing operator that adds a small self-loop to every state,

$$\mathbf{P}_\varepsilon \;=\; \frac{(1-\varepsilon)\,\mathbf{P} + \varepsilon\,\mathbf{I}}{Z}, \quad (8)$$

where ε is the hyperparameter that controls this smoothing and Z is the row-sum re-normalizer. The resulting kernel is irreducible and aperiodic on the support of the calibration trajectories, so the Perron–Frobenius theorem guarantees a unique stationary distribution π satisfying $\pi^\top\mathbf{P}_\varepsilon = \pi^\top$. We compute π by power iteration as follows:

$$\pi^\top_{t+1} \;=\; \pi^\top_t\,\mathbf{P}_\varepsilon, \quad (9)$$

where $\pi_0 = \frac{1}{LE}\mathbf{1}$ (uniform initial probabilities) and the iteration terminates when $\|\pi_{t+1} - \pi_t\|_1 < \tau$ or after N_τ iterations, whichever condition is hit first.

3.5 Expert Selection

Let σ_ℓ be the permutation that sorts all expert slots of layer ℓ in ascending order of π-mass. We mark the bottom $K = \lfloor \rho \cdot E \rfloor$ experts as redundant:

$$\mathcal{R}_\ell = \{\sigma_\ell(1), \sigma_\ell(2), \ldots, \sigma_\ell(K)\}. \tag{10}$$

The selected set of redundant experts, $\mathcal{R}_\ell$, is chosen as a consequence of the model's global routing patterns under a uniformity constraint that ensures no layer is pruned overly aggressively.

3.6 Structural Removal of Redundant Experts

Let $\bar{\mathcal{E}}_\ell = \{1, \ldots, E\} \setminus \mathcal{R}_\ell$ denote the retained experts in layer ℓ. We slice the expert-bank tensors along the expert dimension as follows:

$$\mathbf{W}_\ell^{(i)} \leftarrow \mathbf{W}_\ell^{(i)}[\bar{\mathcal{E}}_\ell, :, :] \tag{11}$$

$$\mathbf{b}_\ell^{(i)} \leftarrow \mathbf{b}_\ell^{(i)}[\bar{\mathcal{E}}_\ell, :] \tag{12}$$

where $\mathbf{W}_\ell^{(i)}$ is the ith expert-bank tensor in layer ℓ and $\mathbf{b}_\ell^{(i)}$ is its corresponding bias. The router weight $\mathbf{W}_\ell^{\text{router}} \in \mathbb{R}^{E \times d}$ has its rows restricted to $\bar{\mathcal{E}}_\ell$, eliminating the logits associated with deleted experts. The router's top-k parameter is clamped to $\min(k, |\bar{\mathcal{E}}_\ell|)$ so that selection remains well-defined in layers where the survivor count has fallen below the original k. Owing to the structural removal of experts, the pruned model contains strictly fewer parameters and consumes proportionally less memory without requiring additional masking or runtime bookkeeping to realize these gains.

3.7 Recovery Fine-Tuning

Aggressive pruning unavoidably perturbs the model's output distribution. We recover task performance with a brief LoRA-based recovery fine-tuning stage (Hu et al., 2021). Crucially, LoRA adapters are inserted on each layer's attention projections ($\{\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V, \mathbf{W}^O\}$) while the expert weights and router parameters are kept frozen. This ensures that the structural decisions made by the Markov-chain analysis are preserved; the adapter learns to compensate by reshaping attention behavior around the surviving experts rather than by re-mixing the routing distribution.

4 Experimental Setup

Model Families. We benchmark MAESTRO on two different MoE-based models, namely, GPT-OSS-20B[1] (OpenAI et al., 2025) and Qwen-3-30B[2] (Yang et al., 2025). Specifically, GPT-OSS-20B model contains 32 experts per layer in contrast to 128 experts in Qwen-3-30B, allowing us to evaluate MAESTRO under both, expert-sparse and expert-dense routing paradigms.

Baselines. We compare MAESTRO against a naive, random pruning baseline and three contemporary baselines: an expert merging framework, HC-SMoE (Chen et al., 2025b), and two MoE-pruning methods, Mosaic Pruning (MoP) (Hu et al., 2026) and REAP (Lasby et al., 2026). All baselines were manually adapted to support the evaluated model families and our evaluation framework. To ensure a fair comparison, all configurations are held constant across baselines, with only the core pruning methodology varying. We provide all implementation details in Appendix A. Note that MOP holds each expert's output for every calibration token in memory. While this is manageable for GPT-OSS-20B (32 experts per layer), it requires a prohibitively high amount of memory for Qwen-3-30B (128 experts per layer). As a result, we are unable to run MOP on Qwen and report its results only for GPT OSS.

Evaluation Tasks. We evaluate all methods on a comprehensive testbed of 17 tasks across five domains. We measure generative aptitude by computing log-perplexity on the WikiText-2 (Merity et al., 2016) and LAMBADA (Paperno et al., 2016) datasets. PIQA (Bisk et al., 2020), PROST (Aroca-Ouellette et al., 2021), and CommonsenseQA (Talmor et al., 2019) are employed to gauge world understanding and common sense reasoning. We assess domain-specific capabilities via science-based reasoning tasks such as ARC-Easy and ARC-Challenge (Clark et al., 2018), and a medical question-answering dataset called MedQA (Jin et al., 2021). Additionally, we employ OpenbookQA (Mihaylov et al., 2018) for multi-domain advanced question-answering and reasoning. We use the BLiMP (Warstadt et al., 2020), BoolQ (Clark et al., 2019), LAMBADA (Paperno et al., 2016), Winogrande (Sakaguchi et al., 2021), and COQA (Reddy et al., 2019) benchmarks to quantify natural language understanding and inference

[1] https://huggingface.co/openai/gpt-oss-20b
[2] https://huggingface.co/Qwen/Qwen3-30B-A3B

Model	CR	Method	Generative (Log-PPL ↓)	World (Acc ↑)	Domain (Acc ↑)	NLU & NLI (Acc ↑)	Safety (Score ↑)	Avg RP (%) (↑)	Std RP (%) (↓)
GPT-OSS-20B	-	Base	2.17	65.62	56.82	74.16	58.17	-	-
	25%	Random	2.39	63.60	52.53	72.40	58.66	95.38	6.00
		HC-SMoE	2.54	57.86	47.07	72.55	57.26	90.46	8.60
		MOP	2.31	63.99	53.04	72.85	56.59	95.63	5.10
		REAP	2.21	64.19	55.46	73.57	57.52	97.67	2.70
		MAESTRO	2.18	64.68	57.30	73.92	57.97	**98.90**	**1.90**
	50%	Random	2.95	52.35	44.78	65.21	57.82	83.92	11.50
		HC-SMoE	3.00	56.39	39.26	69.17	56.24	83.71	12.50
		MOP	2.56	60.06	43.06	71.90	54.85	88.41	10.80
		REAP	2.47	58.49	44.70	71.17	56.62	89.37	10.40
		MAESTRO	2.40	60.64	50.61	71.84	56.14	**92.59**	**6.30**
Qwen-3-30B	-	Base	1.97	69.43	63.29	76.65	56.60	-	-
	25%	Random	2.19	62.67	53.57	74.49	54.00	91.79	7.10
		HC-SMoE	2.35	59.62	46.65	72.06	56.20	87.47	11.00
		REAP	1.93	68.86	59.77	76.58	54.26	98.13	4.30
		MAESTRO	1.86	70.00	61.56	76.47	53.95	**99.40**	**3.50**
	50%	Random	2.86	54.65	44.97	61.06	54.18	78.77	13.50
		HC-SMoE	3.23	47.37	36.61	65.54	53.70	74.58	16.60
		REAP	2.45	63.56	53.54	72.48	54.45	90.25	**7.10**
		MAESTRO	2.04	65.82	52.07	75.71	53.81	**93.40**	10.50

Table 1: Post-compression performance of GPT-OSS-20B and Qwen3-30B at compression ratios of 25% and 50% after undergoing RFT. MAESTRO demonstrates excellent performance consistently across diverse MoE-based architectures. Best results are highlighted in boldface.

capabilities. Notably, unlike prior MoE baselines, we assess models from a lens of Safety, Bias, and Ethics via popular benchmarks such as Winogender (Rudinger et al., 2018), TruthfulQA (Lin et al., 2022), and Moral Stories (Emelin et al., 2021).

5 Results

Table 1 provides quantitative performance metrics of various pruning methods on two architecturally distinct MoE models: GPT-OSS-20B and Qwen3-30B. We find MAESTRO to be consistently superior across both architectures, demonstrating that it is not a method tuned to a single model family but rather a general-purpose pruner whose Markov-chain formulation transfers across diverse MoE topologies. On GPT-OSS-20B, MAESTRO outperforms the best baseline (REAP) by relatively 1.26% at 25% compression and by 3.60% at 50% compression. On Qwen3-30B, the relative margins are equally strong: 1.29% over REAP at 25% and 3.49% at 50% compression. Even under a strict 50% compression regime, MAESTRO surpasses HC-SMoE by up to 10.61% on GPT-OSS-20B and by 25.23% on Qwen3-30B, underscoring its robustness under aggressive compression.

Task-level analysis. A closer inspection of individual tasks reveals that the gains of MAESTRO are not uniform but are most pronounced in the Generative and Domain dimensions, which are arguably the most sensitive indicators of expert specialization. On GPT-OSS-20B at 25% compression, MAESTRO achieves a Domain accuracy of 57.30, which not only surpasses every baseline but also *exceeds the unpruned base model* (56.82), suggesting that removing low-salience experts can act as a mild regularizer by eliminating redundant or noisy routing paths. Its Log-PPL of 2.18 is within 0.01 of the base (2.17), effectively recovering near-lossless generative quality that no baseline achieves. A strikingly similar pattern emerges on Qwen3-30B: at 25% compression, MAESTRO achieves a Log-PPL of 1.86, which *beats* the unpruned base model (1.97), and a World Knowledge accuracy of 70.00 that again *surpasses* the base (69.43). These results, unlikely to be coincidental, corroborate our central claim that modeling expert routing as an Ergodic Markov chain and retaining experts with the highest stationary probabilities per layer preserves the experts that collectively carry the most global information flow, rather than those that merely appear frequently in local or layer-wise snapshots while ensuring no layer is over-pruned.

Graceful degradation under aggressive compression. A notable property of MAESTRO is its graceful degradation as the compression ratio increases. On GPT-OSS-20B, moving from 25% to 50% compression, MAESTRO incurs an average RP drop of only 6.31 percentage points, compared to drops of 7.22 (MOP), 6.75 (HC-SMoE), 11.46 (Random),

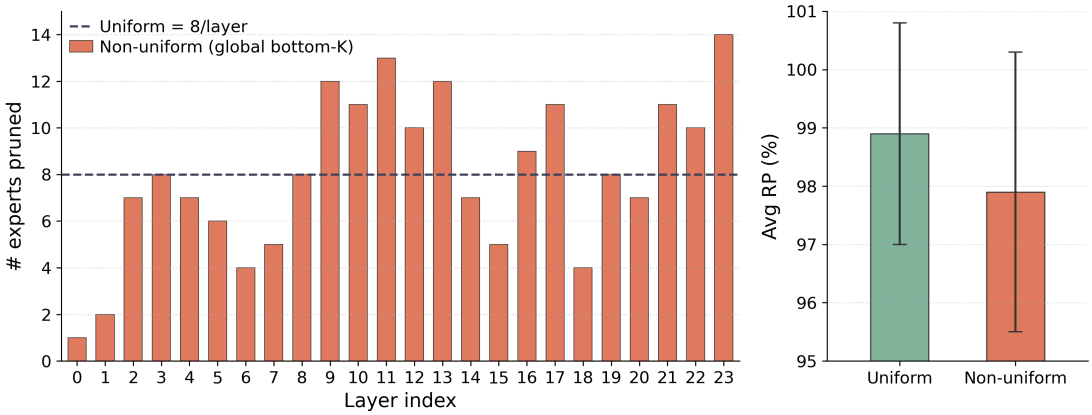

Figure 2: Comparison of uniform vs. non-uniform expert pruning strategies applied to GPT-OSS-20B at 25% compression. Uniform pruning (8 experts/layer) achieves a higher average retained performance (RP) compared to non-uniform pruning, indicating that allowing unconstrained global pruning can over-prune sensitive layers and hurt overall model quality, despite the flexibility it affords.

and 8.30 (REAP). This trend can be observed on Qwen3-30B as well: MAESTRO drops by just 6.00 percentage points, versus 13.02 (Random), 12.89 (HC-SMoE), and 7.88 (REAP). This suggests that the stationary distribution provides a well-calibrated, architecture-agnostic global ranking of expert importance: even when a large fraction of experts is removed, the retained set still faithfully represents the model's aggregate routing behavior regardless of the underlying MoE topology. Notably, on GPT-OSS-20B, MAESTRO at 50% compression (92.59% RP) still outperforms HC-SMoE at the much lighter 25% setting (90.46% RP); the analogous gap is even wider on Qwen3-30B, where MAESTRO at 50% (93.40%) eclipses HC-SMoE at 25% (87.47%) by 6.78%.

Consistency as a proxy for routing coherence. Beyond raw performance, the standard deviation of RP serves as a proxy for how well a pruned model generalizes across heterogeneous tasks. On GPT-OSS-20B, MAESTRO achieves the lowest Std RP at both compression ratios (1.90% at 25% and 6.30% at 50%), compared to the next best competitor at each level (REAP: 2.70%; REAP: 10.40%, respectively). On Qwen3-30B at 25% compression, MAESTRO again records the lowest standard deviation (3.50%) versus REAP (4.30%), Random (7.10%), and HC-SMoE (11.00%). At 50% compression on Qwen3-30B, REAP achieves a marginally lower Std RP (7.10%) compared to MAESTRO (10.50%), though MAESTRO retains a substantially higher average RP (93.40% vs. 90.25%), indicating that the small consistency advantage of REAP comes at the cost of significantly greater performance loss. The improvement in consistency is particularly striking at 50% compression on GPT-OSS-20B, where competing methods exhibit standard deviations of 10–12%, roughly twice that of MAESTRO. Local heuristics such as the ones used by baseline methods tend to over-rely on task-dominant experts, often turning a blind eye to experts that are moderately active across various tasks and anchor cross-task consistency. This structural blind spot manifests in the form of high variance. By leveraging a globally relevant importance score, MAESTRO bypasses this issue, yielding models that are much more consistent across diverse benchmarks.

We report additional results and analyze them in Appendix C.

6 Discussion

Uniform v/s Non-Uniform Pruning. Figure 2 demonstrates that under a non-uniform regime, MAESTRO tends to over-prune certain layers, even compressing them by almost twice as much as their uniformly pruned counterparts. For instance, layers 9, 11, 13, and 23 are pruned by 12 to 14 experts, forcing these layers to rely on a disproportionately small set of experts, leading to a performance bottleneck that outweighs the benefits of a dynamically allocated compression budget, causing the non-uniform variant to perform worse by a full percentage point on average and also exhibit higher per-task standard deviation.

Influence of Autoregressive Expert Transition Counting on Pruning Decisions. Figure 3 demonstrates clear differences in information encoded when expert transitions are accumulated over

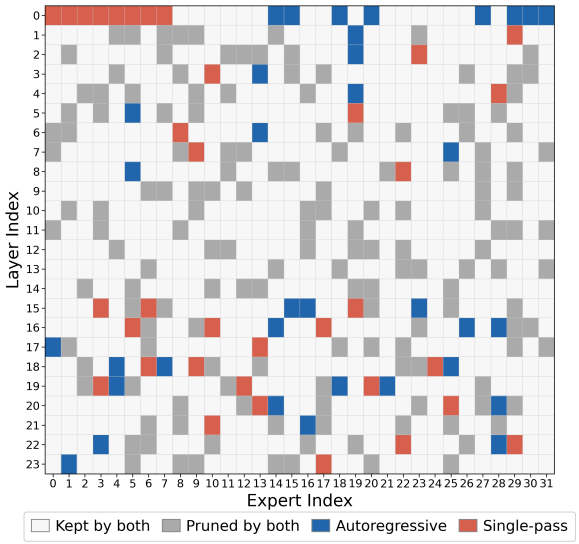

Figure 3: Pruning decisions made by MAESTRO versus those obtained by ablating its autoregressive transition counting to naive expert activation frequency, across 24 layers of GPT-OSS-20B at 25% compression. As evident, loss of autoregressive context leads to qualitatively different and inferior pruning decisions.

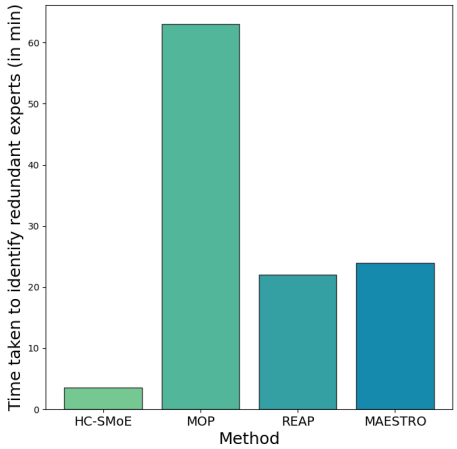

Figure 4: Comparison of the run-times of different pruning methods on GPT OSS at a compression ratio of 25%. Since the RFT process is method-agnostic, we omit it for this analysis.

$T_{seq} = 100$ autoregressive steps versus when they are computed over a single pass. The divergence is most pronounced in Layer 0, where single-pass counting prunes a near-contiguous block of low-index experts (0–7), reflecting a positional activation bias rather than any meaningful structural property of the routing graph. MAESTRO, by contrast, selects an entirely disjoint set in this layer, identifying experts whose low stationary probability under the autoregressive Markov chain marks them as genuinely peripheral to the model's learned routing topology. This distinction matters: early layers process token-level representations that heavily influence downstream routing decisions, so errors in pruning at this depth are likely to compound across layers. Conversely, Layers 9–14 exhibit near-total agreement between the two heuristics (grey), suggesting that a subset of experts are unambiguously redundant and recoverable by any reasonable importance metric. We provide corresponding task-specific performance results in Appendix C.

Method Run-time Comparison. Figure 4 compares the time it takes for each method to compress an instance of GPT-OSS-20B by 25%. We observe that HC-SMoE, an expert merging baseline, operates the fastest due to its rapid agglomerative clustering approach, taking roughly 3.5 minutes for the same. This excellent efficiency, however, comes at a notable cost in retained performance, as shown in Table 1. In contrast, MOP, a stronger baseline, takes around 63 minutes for compression, owing to its expensive expert similarity process, followed by iterative clustering. REAP completes the task in approximately 22 minutes, benefiting from its routing-based expert selection which avoids exhaustive pairwise comparisons. MAESTRO, our proposed method, achieves compression in around 24 minutes, a time that is comparable to REAP and nearly 3× faster than MOP, while delivering superior retained performance (Table 1). MAESTRO's modest overhead over REAP is attributable to its autoregressive counting of inter-expert transitions as opposed to the single-pass importance scores that REAP operates on. Overall, MAESTRO strikes a strong balance between computational efficiency and compression quality, making it a practical choice for large-scale MoE compression.

7 Conclusion

In this work, we present, MAESTRO, a novel structured pruning framework designed specifically for MoE LLMs. By leveraging Markov chains to model autoregressive routing decisions, MAESTRO grounds its decisions in globally relevant heuristics rather than locally-derived, single-pass activation statistics. MAESTRO outperforms state-of-the-art baselines by up to 10% even under an extreme compression ratio of 50%. Additionally, MAESTRO exhibits superior cross-task consistency and robustness across multiple MoE architectures.

8 Limitations

We identify two main limitations with our work. First, MAESTRO's routing model rests on a first-order Markov assumption: the experts selected at layer $\ell + 1$ depend only on those selected at layer ℓ, not on earlier layers. While this yields a tractable and theoretically grounded stationary distribution, it may underestimate the importance of experts whose relevance is conditioned on longer routing histories; exploring higher-order chain approximations is a natural avenue for future work. Furthermore, MAESTRO currently operates at the granularity of whole experts: it either retains or discards an expert in its entirety. This binary decision may be suboptimal in settings where a moderate compression budget would be better served by a combination of expert pruning and intra-expert weight reduction, and integrating MAESTRO's global routing signal with width- or depth-level pruning criteria represents another promising direction.

9 Ethical Considerations

This work presents MAESTRO, a structured pruning framework for compressing MoE LLMs. We reflect on the ethical dimensions of this research below.

Usage of Existing Artifacts. This work builds upon publicly available model checkpoints and datasets. Specifically, we use GPT-OSS-20B (OpenAI et al., 2025) and Qwen3-30B (Yang et al., 2025), accessed via their respective HuggingFace repositories in accordance with their published model cards and terms of use. For calibration and recovery fine-tuning, we use the SlimOrca dataset (Lian et al., 2023), which is released under the Apache 2.0 license. Evaluation is conducted using the Language Model Evaluation Harness (Gao et al., 2024), and LoRA-based fine-tuning is performed via the PEFT library (Mangrulkar et al., 2022), both of which are open-source. We strictly adhere to the terms of use of each of the artifacts used by us and will release our own under the MIT License upon publication.

Intended Use. MAESTRO is designed to reduce the memory footprint and deployment cost of MoE models, with the goal of making them more accessible in resource-constrained settings. While we believe democratizing access to capable models is broadly beneficial, we acknowledge that lowering the barrier to deployment also lowers the barrier to misuse. Compressed models that fit on commodity hardware may be more easily deployed in adversarial contexts without institutional oversight. We encourage users of this framework to adhere to the intended use policies of the underlying model families.

Safety and Bias Under Compression. A central concern with any model compression technique is whether it disproportionately degrades safety-relevant capabilities. To this end, we explicitly evaluate all methods on Safety, Bias, and Ethics benchmarks (Winogender, TruthfulQA, and Moral Stories), which is notably absent from most prior pruning work. Our results indicate that MAESTRO's safety-relevant scores remain broadly stable across compression ratios, and are competitive with or superior to baselines. Nevertheless, we caution that benchmark performance on these tasks is an imperfect proxy for real-world safety behavior, and we strongly recommend that practitioners conduct targeted safety evaluations before deploying any compressed model in a production setting.

Environmental Impact. Model compression is, at its core, a technique for reducing the computational resources required for inference, and we therefore expect the net effect of this work on energy consumption to be positive. That said, the calibration and recovery fine-tuning stages of MAESTRO do incur a one-time computational cost.

Broader Societal Impact. By enabling high-quality compression of state-of-the-art MoE models, this work contributes to a trend of increasing model accessibility. We believe this is net positive for researchers and practitioners in low-resource settings, but note that it does not address the upstream costs of training the base models, which remain substantial and are concentrated among a small number of well-resourced organizations.

References

Stéphane Aroca-Ouellette, Cory Paik, Alessandro Roncone, and Katharina Kann. 2021. PROST: Physical reasoning about objects through space and time. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4597–4608, Online. Association for Computational Linguistics.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. Slicegpt: Compress large language models by deleting rows and columns. *Preprint*, arXiv:2401.15024.

Aishwarya Bhandare, Vamsi Sripathi, Deepthi Karkada, Vivek Menon, Sun Choi, Kushal Datta, and Vikram

Saletore. 2019. Efficient 8-bit quantization of transformer neural machine language translation model. *Preprint*, arXiv:1906.00532.

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*.

Hao Chen, Haoze Li, Zhiqing Xiao, Lirong Gao, Qi Zhang, Xiaomeng Hu, Ningtao Wang, Xing Fu, and Junbo Zhao. 2025a. ALPS: Attention localization and pruning strategy for efficient adaptation of large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11764–11780, Vienna, Austria. Association for Computational Linguistics.

I-Chun Chen, Hsu-Shen Liu, Wei-Fang Sun, Chen-Hao Chao, Yen-Chang Hsu, and Chun-Yi Lee. 2025b. Retraining-free merging of sparse moe via hierarchical clustering.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the No rth American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Denis Emelin, Ronan Le Bras, Jena D. Hwang, Maxwell Forbes, and Yejin Choi. 2021. Moral stories: Situated reasoning about norms, intents, actions, and their consequences. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 698–718, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *Preprint*, arXiv:1803.03635.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. Gptq: Accurate post-training quantization for generative pre-trained transformers. *Preprint*, arXiv:2210.17323.

Elias Frantar, Sidak Pal Singh, and Dan Alistarh. 2022. Optimal brain compression: a framework for accurate post-training quantization and pruning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

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.

Jialong Guo, Xinghao Chen, Yehui Tang, and Yunhe Wang. 2025. Slimllm: Accurate structured pruning for large language models. *Preprint*, arXiv:2505.22689.

Song Han, Huizi Mao, and William J. Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *Preprint*, arXiv:1510.00149.

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.

Wenwen Hou, Xinyuan Song, and Shiwei Liu. 2026. Acttail: Global activation sparsity in large language models. *Preprint*, arXiv:2603.12272.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Wentao Hu, Mingkuan Zhao, Shuangyong Song, Xiaoyan Zhu, Xin Lai, and Jiayin Wang. 2026. Mosaic pruning: A hierarchical framework for generalizable pruning of mixture-of-experts models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(26):21885–21893.

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

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14).

Thennal D K, Tim Fischer, and Chris Biemann. 2025. Large language models are overparameterized text encoders. In *Proceedings of the 10th Workshop on Representation Learning for NLP (RepL4NLP-2025)*, pages 170–184, Albuquerque, NM. Association for Computational Linguistics.

Mike Lasby, Ivan Lazarevich, Nish Sinnadurai, Sean Lie, Yani Ioannou, and Vithursan Thangarasa. 2026. REAP the experts: Why pruning prevails for one-shot moe compression. In *The Fourteenth International Conference on Learning Representations*.

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, Vienna, Austria. Association for Computational Linguistics.

Pingzhi Li, Zhenyu Zhang, Prateek Yadav, Yi-Lin Sung, Yu Cheng, Mohit Bansal, and Tianlong Chen. 2024. Merge, then compress: Demystify efficient smoe with hints from its routing policy. *Preprint*, arXiv:2310.01334.

Wing Lian, Guan Wang, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. Slimorca: An open dataset of gpt-4 augmented flan reasoning traces, with verification.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

James Liu, Pragaash Ponnusamy, Tianle Cai, Han Guo, Yoon Kim, and Ben Athiwaratkun. 2025. Training-free activation sparsity in large language models. *Preprint*, arXiv:2408.14690.

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. 2024. QLLM: Accurate and efficient low-bitwidth quantization for large language models. In *International Conference on Learning Representations (ICLR)*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. 2024a. 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.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. 2024b. 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.

Yuqi Luo, Chenyang Song, Xu Han, Yingfa Chen, Chaojun Xiao, Xiaojun Meng, Liqun Deng, Jiansheng Wei, Zhiyuan Liu, and Maosong Sun. 2025. Sparsing law: towards large language models with greater activation sparsity. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems*.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, Benjamin Bossan, and Marian Tietz. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2025. ShortGPT: Layers in large language models are more redundant than you expect. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20192–20204, Vienna, Austria. Association for Computational Linguistics.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Preprint*, arXiv:1905.10650.

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 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Mohammed Muqeeth, Haokun Liu, and Colin Raffel. 2024. Soft merging of experts with adaptive routing. *Preprint*, arXiv:2306.03745.

NVIDIA, :, Aaron Blakeman, Aaron Grattafiori, Aarti Basant, Abhibha Gupta, Abhinav Khattar, Adi Renduchintala, Aditya Vavre, Akanksha Shukla, Akhiad Bercovich, Aleksander Ficek, Aleksandr Shaposhnikov, Alex Kondratenko, Alexander Bukharin, Alexandre Milesi, Ali Taghibakhshi, Alisa Liu, Amelia Barton, and 340 others. 2025. Nvidia nemotron 3: Efficient and open intelligence. *Preprint*, arXiv:2512.20856.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, and 108 others. 2025. gpt-oss-120b gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: An imperative style, high-performance deep learning library. *Preprint*, arXiv:1912.01703.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

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

Ayan Sengupta, Siddhant Chaudhary, and Tanmoy Chakraborty. 2025. You only prune once: Designing calibration-free model compression with policy learning. In *The Thirteenth International Conference on Learning Representations*.

Dmitriy Shopkhoev, Ammar Ali, Magauiya Zhussip, Valentin Malykh, Stamatios Lefkimmiatis, Nikos Komodakis, and Sergey Zagoruyko. 2026. Replaceme: Network simplification via depth pruning and transformer block linearization. *Preprint*, arXiv:2505.02819.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. 2024. Sleb: Streamlining llms through redundancy verification and elimination of transformer blocks. In *Proceedings of the 41st International Conference on Machine Learning*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2024. A simple and effective pruning approach for large language models. *Preprint*, arXiv:2306.11695.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Ming Wang, Miao Zhang, Xuebo Liu, and Liqiang Nie. 2025. Weight-aware activation sparsity with constrained Bayesian optimization scheduling for large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1086–1098, Suzhou, China. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yanyue Xie, Zhi Zhang, Ding Zhou, Cong Xie, Ziang Song, Xin Liu, Yanzhi Wang, Xue Lin, and An Xu. 2024. Moe-pruner: Pruning mixture-of-experts large language model using the hints from its router. *Preprint*, arXiv:2410.12013.

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.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: efficient and affordable post-training quantization for large-scale transformers. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Abdelrahman Zayed, Gonçalo Mordido, Samira Shabanian, Ioana Baldini, and Sarath Chandar. 2024. Fairness-aware structured pruning in transformers. In *Proceedings of the Thirty-Eighth AAAI*

Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

Yang Zhang, Yawei Li, Xinpeng Wang, Qianli Shen, Barbara Plank, Bernd Bischl, Mina Rezaei, and Kenji Kawaguchi. 2024. Finercut: Finer-grained interpretable layer pruning for large language models. *Preprint*, arXiv:2405.18218.

Longguang Zhong, Fanqi Wan, Ruijun Chen, Xiaojun Quan, and Liangzhi Li. 2025. BlockPruner: Fine-grained pruning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5065–5080, Vienna, Austria. Association for Computational Linguistics.

A Implementation Details

MAESTRO has been implemented in PyTorch (Paszke et al., 2019) using Huggingface's Transformers library (Wolf et al., 2020). We conduct all experiments on a cloud-based GPU server consisting of 8 NVIDIA A100 GPUs, each compirising of 80GB of VRAM. For post-compression recovery fine-tuning (RFT), we employ the PEFT library (Mangrulkar et al., 2022) and conduct evaluation using the Language Model Evaluation Harness (Gao et al., 2024).

We estimate expert transition counts (Section 3.3) by calibrating the model on 250 samples of the SlimOrca dataset (Lian et al., 2023), each with a context length of 512 tokens, achieved via padding or truncation. All transitions are counted over $T_{seq} = 100$ autoregressive generation steps. Notably, we utilize masking to ensure that padding tokens are not considered when counting transitions. In order to induce aperiodicity in the resulting stochastic transition matrix, we employ ε-smoothing, where $\varepsilon = 10^{-5}$ (Equation 8). In order to compute the stationary distribution of our final transition matrix, we employ power iteration with a tolerance of $\tau = 10^{-6}$ under a maximum of $N_\tau = 1000$ (Section 3.4) iterations to balance accuracy with computational efficiency.

After compression, we conduct RFT on 2,000 SlimOrca conversations with 1,024 tokens each, using LoRA adapters of rank r=16 and scaling α=16 (no dropout) that are inserted on the attention projections of each layer. Training is done over a single epoch using the AdamW optimizer (Loshchilov and Hutter, 2017) with a peak learning rate of 2×10^{-4}, a linear warm-up over the first 10% of steps followed by linear decay, and a batch size of 2.

B Detailed Per-Task Performance

Tables 2 and 3 report per-task performance for all methods on GPT-OSS-20B and Qwen3-30B respectively, across both compression ratios.

GPT-OSS-20B. At 25% compression, MAESTRO recovers near-lossless generative quality, with Lambada Log-PPL (1.86) marginally surpassing the unpruned base (1.87) and PIQA accuracy (78.35) also exceeding the base (78.13), suggesting that removing low-stationary-probability experts can have a mild regularising effect on specific tasks. The most pronounced gains over baselines appear in domain-specific tasks: MAESTRO leads on MedQA by a substantial margin at both compression levels (60.49 vs. 58.13 for REAP at 25%; 51.93 vs. 37.63 for REAP at 50%), indicating that salient domain-specialized experts are well-preserved by the stationary distribution heuristic. At 50% compression, performance on NLU tasks degrades gracefully and remains competitive with or above baselines, while Safety and Ethics scores are broadly stable across all methods, suggesting these capabilities are robust to expert pruning at the ratios studied.

Qwen3-30B. The results on Qwen3-30B reinforce and strengthen the trends observed on GPT-OSS-20B. At 25% compression, MAESTRO exceeds the base model on four tasks, Wikitext Log-PPL (2.35 vs. 2.52), Lambada Log-PPL (1.38 vs. 1.42), PIQA (81.18 vs. 80.69), and CommonsenseQA (80.43 vs. 79.12), pointing to a consistent regularization effect that is more pronounced in the larger model. Even at 50% compression, MAESTRO retains a Log-PPL of 1.46 on Lambada, within 0.04 of the uncompressed base, a margin that no other method approaches. HC-SMoE degrades most severely at this compression level, dropping to 74.58% average RP with a standard deviation of 16.60%, highlighting the instability of clustering-based approaches under aggressive pruning. Overall, the per-task breakdown confirms that MAESTRO's aggregate gains reported in the main results are not driven by outlier tasks but reflect broad, consistent performance retention across generative, knowledge, domain, and language understanding benchmarks.

CR	Method	Generative		World Understanding			Domain-Specific			
		Wikitext (Log-PPL ↓)	**Lambada** (Log-PPL ↓)	**PIQA** (Acc ↑)	**PROST** (Acc ↑)	**CommonsenseQA** (Acc ↑)	**OpenbookQA** (Acc ↑)	**ARC-Easy** (Acc ↑)	**ARC-Challenge** (Acc ↑)	**MedQA** (Acc ↑)
-	Base	2.46	1.87	78.13	45.93	72.81	39.80	79.08	52.39	64.81
25%	Random	2.68	2.09	77.80	46.32	66.67	38.80	74.58	47.18	49.57
	HC-SMoE	2.85	2.24	70.78	35.80	66.99	36.40	62.79	43.52	45.56
	MOP	2.64	1.99	77.04	43.69	71.25	40.20	73.23	48.21	50.51
	REAP	2.53	1.89	77.37	44.44	70.76	39.60	74.54	49.57	58.13
	MAESTRO	2.50	1.86	78.35	45.03	70.68	40.00	78.11	50.60	60.49
50%	Random	3.15	2.74	71.76	35.32	49.96	34.60	65.40	42.92	36.21
	HC-SMoE	3.27	2.74	68.50	36.21	64.46	32.40	54.97	34.64	35.04
	MOP	2.87	2.25	72.52	36.97	70.68	35.60	60.77	37.54	38.39
	REAP	2.77	2.16	73.56	35.40	66.50	37.40	62.29	41.47	37.63
	MAESTRO	2.74	2.06	77.20	37.90	66.83	39.00	69.02	42.49	51.93

(continued)

CR	Method	NLU & NLI					Safety, Bias & Ethics				
		BLIMP (Acc ↑)	**BoolQ** (Acc ↑)	**Lambada** (Acc ↑)	**Winogrande** (Acc ↑)	**CoQA** (Acc ↑)	**Winogender** (Acc ↑)	**TruthfulQA** (Acc ↑)	**Moral Stories** (Acc ↑)	**Avg RP (%)** (↑)	**Std RP (%)** (↓)
-	Base	81.04	86.70	60.61	67.64	74.80	63.06	58.57	52.88	-	-
25%	Random	80.67	85.47	57.42	65.11	73.30	62.92	57.52	55.53	95.38	6.00
	HC-SMoE	79.84	85.54	55.68	67.17	74.50	63.61	54.90	53.27	90.46	8.60
	MOP	79.66	84.56	58.55	67.48	74.00	61.11	55.86	52.81	95.63	5.10
	REAP	78.36	86.48	60.66	68.51	73.84	62.92	56.66	52.98	97.67	2.70
	MAESTRO	79.11	86.85	60.72	68.43	74.50	63.47	57.22	53.22	**98.90**	**1.90**
50%	Random	78.35	78.53	46.98	58.17	64.00	60.56	56.13	56.78	83.92	11.50
	HC-SMoE	79.21	82.72	47.41	64.09	72.40	60.42	54.94	53.37	83.71	12.50
	MOP	79.90	86.06	54.73	65.90	72.90	57.92	53.49	53.15	88.41	10.80
	REAP	77.10	85.02	56.67	64.48	72.60	61.11	55.24	53.52	89.37	10.40
	MAESTRO	78.15	84.92	57.37	65.75	73.00	61.25	53.52	53.66	**92.59**	**6.30**

Table 2: Detailed task-wise performance results for various pruners on GPT-OSS-20B across compression ratios of 25% and 50% with RFT. Best results are highlighted in boldface.

CR	Method	Generative		World Understanding			Domain-Specific			
		Wikitext (Log-PPL ↓)	**Lambada** (Log-PPL ↓)	**PIQA** (Acc ↑)	**PROST** (Acc ↑)	**CommonsenseQA** (Acc ↑)	**OpenbookQA** (Acc ↑)	**ARC-Easy** (Acc ↑)	**ARC-Challenge** (Acc ↑)	**MedQA** (Acc ↑)
-	Base	2.52	1.42	80.69	48.47	79.12	44.80	78.91	56.23	73.21
25%	Random	2.69	1.68	76.33	36.73	74.94	42.00	64.94	47.70	59.62
	HC-SMoE	2.89	1.81	70.89	36.71	71.25	36.00	64.02	43.77	42.81
	REAP	2.42	1.44	79.92	45.50	81.16	44.40	72.31	52.13	70.23
	MAESTRO	2.35	1.38	81.18	48.39	80.43	44.80	76.05	52.65	72.74
50%	Random	3.22	2.50	71.44	27.73	64.78	33.00	62.67	42.49	41.71
	HC-SMoE	3.58	2.89	61.26	26.88	53.97	32.40	51.35	34.73	27.97
	REAP	2.83	2.07	74.21	41.52	74.94	39.80	70.33	50.00	54.05
	MAESTRO	2.61	1.46	79.38	36.85	81.25	42.60	72.18	48.55	44.93

(continued)

CR	Method	NLU & NLI					Safety, Bias & Ethics				
		BLIMP (Acc ↑)	**BoolQ** (Acc ↑)	**Lambada** (Acc ↑)	**Winogrande** (Acc ↑)	**CoQA** (Acc ↑)	**Winogender** (Acc ↑)	**TruthfulQA** (Acc ↑)	**Moral Stories** (Acc ↑)	**Avg RP (%)** (↑)	**Std RP (%)** (↓)
-	Base	82.95	88.72	64.62	70.88	76.10	66.53	52.56	50.71	-	-
25%	Random	81.97	81.50	63.21	70.80	75.00	59.31	51.88	50.80	91.79	7.10
	HC-SMoE	84.07	73.49	60.43	69.61	72.70	64.17	53.32	51.12	87.47	11.00
	REAP	82.36	85.51	67.26	71.98	75.80	57.92	54.25	50.61	98.13	4.30
	MAESTRO	82.18	84.65	67.88	72.06	75.60	59.44	52.71	49.69	**99.40**	**3.50**
50%	Random	76.50	49.82	50.17	61.80	67.00	58.47	49.55	54.51	78.77	13.50
	HC-SMoE	80.95	73.46	42.07	62.12	69.10	57.08	49.71	54.30	74.58	16.60
	REAP	81.26	81.84	56.59	69.22	73.50	56.67	54.95	51.73	90.25	**7.10**
	MAESTRO	81.41	82.78	66.18	73.09	75.10	57.92	53.66	49.84	**93.40**	10.50

Table 3: Detailed task-wise performance results for various pruners on Qwen3-30B across compression ratios of 25% and 50% with RFT. Best results are highlighted in boldface.

C Additional Results

Table 4 provides additional results of various ablative variants of MAESTRO, allowing for a deeper analysis of the necessity and effectiveness of its components.

– **UNI.** This variant removes the uniform pruning constraint from MAESTRO and prunes experts with the globally minimum stationary probabilities. As analyzed in Section 6, while removing the uniformity constraint leads to a variant that is more flexible and can dynamically allocate the pruning budget across layers, it can also lead to configurations where certain layers are over-pruned, a bottleneck that reflects clearly in the model's post-compression performance metrics (97.90% v/s 98.90% avg RP) and task-wise consistency

Variant	Generative (Log-PPL ↓)	World (Acc ↑)	Domain (Acc ↑)	NLU & NLI (Acc ↑)	Safety (Score ↑)	Avg RP (%) (↑)	Std RP (%) (↓)
MAESTRO	2.18	64.68	57.30	73.92	57.97	**98.90**	**1.90**
– UNI	2.19	63.85	55.78	74.03	57.37	97.90	2.40
– AUTO	2.21	64.35	56.05	73.71	57.18	97.76	2.10
– RFT	5.12	59.42	55.56	47.85	57.28	78.54	26.40
REAP – RFT	5.02	59.60	52.75	48.67	57.06	77.91	25.60

Table 4: Additional results for MAESTRO obtained after compressing GPT-OSS-20B by 25%.

# Calibration Samples	Generative (Log-PPL ↓)	World (Acc ↑)	Domain (Acc ↑)	NLU & NLI (Acc ↑)	Safety (Score ↑)	Avg RP (%) (↑)	Std RP (%) (↓)
250	2.18	64.68	57.30	73.92	57.97	98.90	**1.90**
50	2.20	63.95	56.43	74.00	57.75	98.22	2.30
100	2.19	64.87	56.58	74.26	57.76	98.77	2.20
500	2.19	65.28	56.98	74.03	57.83	**98.93**	2.20

Table 5: Effect of modifying the number of calibration samples used by MAESTRO to estimate inter-expert transitions. While performance seems to be near constant across all regimes, cross-task variance fluctuates by a much larger degree. All results have been obtained after RFT. Best results are in boldface.

(26.32% higher standard deviation than base variant). Empirically, ensuring the stability of the pruning process leads to better overall performance even if it comes at the cost of decisional flexibility, motivating us to the adopt the uniform regime.

– AUTO. This variant ablates the autoregressive expert transition counting component from MAESTRO. Consequently, MAESTRO's Markov modeling is performed using single-pass activation heuristics which fail to capture the model's long range routing behavior (Section 6). As a result, this variant incurs a relative drop of 1.15% in average performance, along with a 10.53% increase in standard deviation across tasks, demonstrating the effectiveness of autoregressive routing trajectories at aiding cross-task generalization and consistency.

– RFT. We also explore the effectiveness of performing post-compression RFT via the – RFT variant which ablates this exact process. We observe that this variant performs considerably worse than MAESTRO's base variant on average performance retention (20.59% relative drop) as well as cross-task consistency (1,289.47% higher variance). We also compare this variant with a non-RFT variant of our best baseline, REAP, denoted as REAP – RFT to determine whether RFT is required only by our method or if it is an unavoidable by-product of compression. To this end, we observe that MAESTRO's – RFT variant outperforms REAP – RFT by almost 1%, albeit with a marginally worse cross-task standard deviation. More importantly, these results clearly demonstrate that compression induces a significant perturbation in the model's output distribution. RFT allows the fine-grained re-calibration of this output distribution so that model performance is recovered to an extent. Therefore, RFT is a vital part of the pruning pipeline and plays an essential role in maximizing post-compression model alignment and performance.

Impact of Size of Calibration Dataset. We also analyze the effect of the number of calibration samples used to accumulate routing trajectories on post-compression performance and provide these results in Table 5. We calibrate MAESTRO on 50, 100, and 500 samples and observe that in all cases, its average performance retention remains almost the same, differing by less than a percent across all variants. However, the cross-task standard deviation for each variant exhibits a much larger shift from the baseline. When calibrated on 50 samples, MAESTRO suffers from a relatively 21.05% higher cross-task standard deviation while under the 100 and 500 sample regimes, its standard deviation increases by 15.79%. We hypothesize that smaller calibration sets (50 and 100 samples) do not provide MAESTRO with enough context tokens to model expert transitions holistically, forcing it to rely on the limited number of domains spanned by the tokens and optimizing for performance primarily on these domains, leading to higher cross-task variance when evaluated on a diverse set of benchmarks. On the other hand, larger calibration sets (500 samples) tend to induce a frequency-based distributional bias towards domains that are most commonly seen in

the data, leading to better performance on related benchmarks, but a disproportionate loss of performance in other domains. Recognizing this behavior, we opted to utilize 250 calibration samples as this is a fair middle-ground that provides MAESTRO with enough cross-domain context to model routing behavior in a generalizable manner without biasing it towards any one particular domain, as reflected by its excellent performance and consistency.