C-GNN-PRUNE: A Unified Graph-Based Framework for Structure-Aware Pruning of Mixture-of-Experts Models

Lin Li, Yan Wang*, Zhuopeng Wang

College of Computer Science, Inner Mongolia University, Hohhot, Inner Mongolia
32409078@mail.imu.edu.cn, cswy@imu.edu.cn, 32409288@mail.imu.edu.cn

Abstract

The Mixture-of-Experts (MoE) architecture has emerged as a promising paradigm for scaling large language models (LLMs) by activating only a sparse subset of experts per input. However, its massive parameter size remains a major obstacle to efficient deployment. Existing pruning methods often ignore two key aspects: the intricate structural dependencies among experts and the heterogeneous importance of different layers. To tackle these issues, we propose C-GNN-PRUNE, a unified and structure-aware compression framework tailored for MoE models. Our method introduces an EntropyGuided Allocation Module that dynamically assigns pruning budgets by leveraging expert activation entropy, enabling adaptive handling of inter-layer heterogeneity. To preserve structural collaboration patterns, we construct an expert interaction graph that fuses functional similarity and routing behavior, and employ a GNN-Based Embedding Module to learn structure-aware expert representations. These embeddings, along with co-activation patterns, are fed into a Community Detection Module to identify expert clusters for structured pruning. Finally, an Activation-Aware Selection Module retains the most critical experts in each community, balancing sparsity and expressiveness. Experiments on multiple open-source MoE models demonstrate that C-GNN-PRUNE consistently outperforms prior methods under various pruning ratios, achieving better trade-offs between compression and accuracy. This framework provides a modular and effective solution for structure-preserving compression of large-scale MoE models.

Introduction

The perfection of model compression lies not in aggressively pruning everything possible, but in identifying the precise point beyond which any further pruning impairs functionality. – **Inspired by Antoine de Saint-Exupéry**

In recent years, in pursuit of superior performance, deep learning models, especially large language models (LLMs), have experienced exponential growth in scale (Devlin et al. 2019; Brown et al. 2020; Dubey et al. 2024). However, this growth has resulted in prohibitively high training and inference costs (Kaddour et al. 2023), prompting the research community to explore more efficient architectures. In this context, the Mixture-of-Experts (MoE) architecture has emerged as a widely adopted paradigm. By sparsely activating only a small subset of parameters for each input token, it effectively scales model capacity while maintaining relatively constant computation costs (Fedus, Zoph, and Shazeer 2022; Du et al. 2022).

Nevertheless, the success of MoE architectures introduces a critical deployment paradox. Although computation is sparse during inference, the full set of model parameters must still reside in memory, leading to substantial memory overhead. As models continue to scale, for example, Mixtral-8x7B (Jiang et al. 2024), DBRX, Snowflake Arctic, DeepSeek-V2 Lite (Liu et al. 2024a), and Qwen1.5-MoE-A2.7B (Team 2024), the number of experts increases dramatically (ranging from 8 to 128), exacerbating this memory bottleneck and making deployment on standard hardware increasingly infeasible. Therefore, it is imperative to prune pre-trained MoE models effectively in order to reduce parameter counts (Guo et al. 2025a).

Existing pruning methods face two fundamental challenges when applied to MoE models. First, many methods treat experts as isolated units, ignoring the complex structural characteristics inherent to MoE architectures. Recent studies have shown that functional homogeneity (i.e., overlapping capabilities) often exists among experts within MoE layers (Lin et al. 2024), and that redundancy increases in deeper layers (Liu et al. 2024d; Xue et al. 2024). Ignoring these structural properties leads to suboptimal pruning and degradation of performance. Second, most methods lack explicit modeling of the mechanisms for collaboration among experts. While some works introduce load-balancing losses (Fedus, Zoph, and Shazeer 2022; Zoph et al. 2022) to reduce underutilization of experts, they do not model inter-expert collaboration directly. The absence of coordination limits the ability of the model to harness the full potential of expert diversity, thereby constraining the effectiveness of pruning.

We argue that the expert set in MoE models is not a flat list but rather a complex and structured system, similar to graph-structured data in domains such as social networks (Perozzi, Al-Rfou, and Skiena 2014) or molecular graphs (Defferrard, Bresson, and Vandergheynst 2016). In such graphs, nodes often form communities, which are clusters with dense internal connections and sparse external links (Red et al. 2011).

*Corresponding author

Exploiting such high-order structure has proven crucial in graph neural networks (GNNs) (Ying et al. 2018), which propagate messages over graph edges (Scarselli et al. 2008) to capture and leverage complex relationships. Motivated by this insight, we propose to model inter-expert relationships as a graph and utilize the expressive power of GNNs to uncover and exploit latent collaboration patterns.

To address the challenges of compressing sparse MoE models while preserving their structural and functional characteristics, we propose C-GNN-PRUNE, a unified and structure-aware pruning framework. Unlike existing methods that rely on heuristic or purely activation-based expert selection, C-GNN-PRUNE explicitly models inter-expert dependencies through graph representation and community detection. It begins with an entropy-guided allocation strategy that adaptively distributes pruning budgets across MoE layers based on the distributional sensitivity of expert activations. Subsequently, a two-stage graph neural network (GNN) module is employed to construct and refine an expert interaction graph by integrating functional output similarity and routing behavior. The learned structure-aware embeddings, combined with activation co-occurrence, enable a principled community partitioning process that reveals functionally redundant expert clusters. Finally, within each cluster, critical experts are retained using activation-aware importance scoring, effectively balancing sparsity and expressiveness.

Our main contributions are summarized as follows:

- **Structure-aware reformulation of MoE pruning as graph-based community detection.** We propose a novel pruning paradigm by modeling expert interactions through a similarity graph that integrates functional output similarity and routing divergence. This enables the identification of structurally redundant experts via community detection, moving beyond traditional per-expert scoring heuristics.
- **Entropy-guided layer-wise pruning allocation for heterogeneous MoE layers.** To address inter-layer functional disparities, we design an entropy-based budget allocation strategy that quantifies the structural sensitivity of each layer using expert activation entropy, enabling adaptive and globally balanced pruning under a total sparsity constraint.
- **GNN-based expert embedding learning for structure refinement.** We develop a two-stage graph neural network (GNN) module to refine expert relationships by learning structure-aware embeddings. This enhances the modeling of higher-order inter-expert dependencies, providing a robust basis for downstream expert clustering and pruning.

Related Work

MoE models have recently emerged as a crucial method for scaling LLMs. The core idea is to activate only one or a few experts in each MoE layer to improve computational efficiency (Shazeer et al. 2017; Fedus, Zoph, and Shazeer 2022). In sparse MoE architectures, expert selection is controlled by gating functions, with common mechanisms including Top-K activation (Shazeer et al. 2017). Although these mechanisms reduce computational cost, they often suffer from load imbalance. To address this, prior studies have introduced auxiliary losses to balance expert load (Fedus, Zoph, and Shazeer 2022) or employed graph-based methods to model collaboration between experts (Cai et al. 2025; Lin et al. 2024).

As the parameter scale of MoE models grows, expert pruning has become key to reducing model size. Early methods often rely on heuristic pruning based on gating statistics (Zhang et al. 2024). Recent work emphasizes eliminating task-irrelevant experts (Chen et al. 2022; Muzio, Sun, and He 2024) or merging them to preserve knowledge (Liu et al. 2024b). However, these methods often assess experts independently, which limits their ability to exploit redundancy across collaborative structures spanning multiple layers (Lee et al. 2024). Another line of work explores post-training pruning and inference-time optimization to further reduce memory and latency overheads (Lu et al. 2024).

To address this limitation, structure-aware pruning has gained increasing attention. Some work has begun to tackle structural redundancy by clustering functionally similar experts within and across layers before pruning (Guo et al. 2025b). Other studies utilize graph neural networks (GNNs) to model collaboration. For example, GBLM-Pruner (Das, Ma, and Shen 2023) and Pruner-Zero (Dong et al. 2024) integrate gradient information and graph structure to generate pruning masks. Building upon these insights, we propose the C-GNN-PRUNE framework, which constructs expert graphs using expert output similarity and routing divergence, and learns structure-aware embeddings through GNNs to enable more reasonable expert partitioning and pruning.

It is worth noting that the redundancy levels of experts vary across MoE layers. A uniform pruning ratio often results in performance degradation. Wanda (Sun et al. 2024) evaluates neuron importance using activation multiplied by magnitude, applying uniform pruning. In contrast, OWL (Yin et al. 2024) proposes a global constraint optimization strategy to adjust pruning ratios across layers. Inspired by these methods, we design an entropy-based layer sensitivity metric to quantify expert activation entropy, and accordingly allocate the global pruning budget adaptively across MoE layers, thereby improving compression rates while preserving model performance.

Methodology

The proposed C-GNN-PRUNE framework demonstrates a structural pruning method for sparse MoE models, as illustrated in Figure 1. It starts with an entropy-based pruning allocation to adaptively assign pruning ratios per layer. Then, a graph neural network (GNN) learns structure-aware expert embeddings by integrating output similarity and routing behavior. These embeddings, combined with activation co-occurrence, form a new similarity graph used for community detection and clustering. Finally, activation-aware scoring selects critical experts within each cluster for pruning. This pipeline preserves functional diversity and structural importance, achieving efficient compression with minimal performance loss.

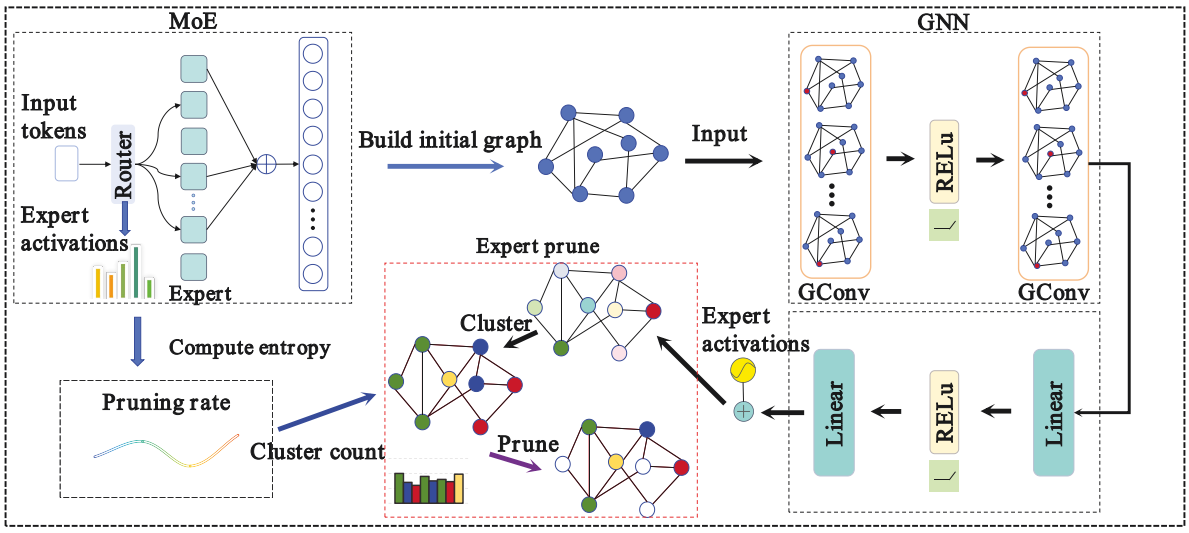

Figure 1: The overall workflow of C-GNN-PRUNE for MoE compression. Input tokens are routed to Top-K experts in the MoE (left), where expert activations are collected to compute entropy-based sensitivities for pruning rate allocation. An expert interaction graph is constructed and refined by a GNN (right), generating embeddings that guide the clustering and pruning of redundant experts (bottom center).

Entropy-Based Layer-wise Pruning Allocation

To determine the number of experts to retain in each MoE layer, we introduce an entropy-based layer-wise pruning allocation strategy. This allocation directly defines the number of expert clusters K_l for each layer, which will later guide the structure-driven community partitioning and pruning stages. The motivation is to prune less aggressively in structurally important layers, as indicated by their activation distribution entropy.

Let N_l denote the total number of experts in the l-th MoE layer, and let $p_i^{(l)}$ be the normalized activation frequency of expert i in this layer. We define the structural sensitivity $\mathcal{S}(l)$ as the normalized entropy of the expert activation distribution:

$$\mathcal{S}(l) = \frac{H_l}{\log N_l}, \quad H_l = -\sum_{i=1}^{N_l} p_i^{(l)} \log p_i^{(l)}. \tag{1}$$

A higher entropy indicates more uniform expert usage across samples, implying that the layer plays a more structurally significant role in routing decisions. Such layers should be pruned more conservatively to preserve model expressiveness.

To determine the pruning ratio τ_l for each layer under a total pruning budget τ, where $N_{\text{total}} = \sum_l N_l$ is the total number of experts across all MoE layers, we formulate the following optimization problem:

$$\min_{\{\tau_l\}} \quad \sum_l \left((1-\tau_l) - \mathcal{S}(l)\right)^2 \tag{2}$$

$$\text{subject to} \quad \sum_l \tau_l N_l = \tau N_{\text{total}}. \tag{3}$$

This objective minimizes the discrepancy between the effective retention ratio of each layer $(1-\tau_l)$ and its entropy-based sensitivity score $\mathcal{S}(l)$, so that layers with higher activation entropy retain more experts, while less active layers are pruned more aggressively, under the global pruning constraint.

We solve this constrained problem using projected gradient descent. At each iteration, we compute the partial derivative for each τ_l:

$$\frac{\partial}{\partial \tau_l} = -2\left((1-\tau_l) - \mathcal{S}(l)\right), \tag{4}$$

and update it as:

$$\tau_l \leftarrow \max\left(0, \tau_l - \eta \cdot \frac{\partial}{\partial \tau_l}\right), \tag{5}$$

where η denotes the learning rate.

To ensure the global constraint is satisfied, we normalize all τ_l values after each update step:

$$\tau_l \leftarrow \tau_l \cdot \frac{\tau N_{\text{total}}}{\sum_j \tau_j N_j}. \tag{6}$$

After convergence, we discretize the pruning counts by rounding $\tau_l N_l$ to the nearest integers and correct any residual budget mismatch by adjusting layers with the smallest rounding errors. The resulting number of retained experts in each layer is then used as the target number of expert clusters in the subsequent community partitioning procedure.

This entropy-guided strategy enables globally balanced pruning decisions that account for layer-wise functional diversity, ensuring more informed and adaptive model compression.

GNN-Based Expert Graph Modeling and Refinement

To effectively model the complex inter-expert relationships beyond conventional statistical metrics, we introduce a two-

stage framework that constructs and refines an expert graph, enabling the learning of structure-aware embeddings.

Stage 1: Initial Expert Graph Construction We model the expert set as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $v_i \in \mathcal{V}$ corresponds to an expert f_i, and each edge $(v_i, v_j) \in \mathcal{E}$ encodes the pairwise similarity between experts. The edge weights are represented by a weighted adjacency matrix $\mathbf{A}^{(0)} \in \mathbf{R}^{N \times N}$, which integrates functional similarity and routing divergence:

$$A_{ij}^{(0)} = \omega \cdot \mathrm{sim}_{\mathrm{output}}(f_i, f_j) + (1 - \omega) \cdot \mathrm{sim}_{\mathrm{routing}}(f_i, f_j), \tag{7}$$

where $\mathrm{sim}_{\mathrm{output}}(\cdot, \cdot)$ combines weighted cosine similarity and inverse distance of expert outputs, and $\mathrm{sim}_{\mathrm{routing}}(\cdot, \cdot)$ quantifies routing probability overlap. The hyperparameter $\omega \in [0, 1]$ controls the weight allocation between the two similarity measures and is empirically set through ablation studies, with the final value set to 0.3.

Stage 2: Structure-Aware Expert Embedding Learning We employ a graph autoencoder with a two-layer Graph Convolutional Network (GCN) encoder:

$$\mathbf{H}^{(1)} = \mathrm{ReLU}\big(\mathrm{GCNConv}(\mathbf{X}, \mathbf{A}^{(0)})\big), \tag{8}$$

$$\mathbf{Z} = \mathrm{GCNConv}(\mathbf{H}^{(1)}, \mathbf{A}^{(0)}). \tag{9}$$

where the node feature matrix $\mathbf{X}$ encodes statistical descriptors (mean and variance) of expert outputs. The decoder, composed of a two-layer MLP, reconstructs the adjacency matrix:

$$\hat{\mathbf{A}} = \sigma\left(\mathrm{MLP}(\mathbf{Z}) \cdot \mathrm{MLP}(\mathbf{Z})^{\top}\right), \tag{10}$$

and the learning objective minimizes the reconstruction error:

$$\mathcal{L}_{\mathrm{recon}} = \left\| \hat{\mathbf{A}} - \mathbf{A}^{(0)} \right\|_F^2. \tag{11}$$

This objective enforces the embedding to preserve original expert similarities while capturing higher-order structural dependencies.

By explicitly decoupling graph construction from embedding learning, our method integrates both functional output similarity and routing-based relations, utilizing neighborhood aggregation mechanisms in GNNs to uncover latent and complex dependencies. Compared to traditional statistical descriptors, the learned embeddings exhibit enhanced discriminative capability and serve as a robust foundation for downstream tasks such as expert clustering, redundancy removal, and MoE pruning.

Structure-Driven Expert Community Partitioning

To identify functionally similar experts and enable structured pruning, we partition the expert graph into disjoint clusters based on both structural embeddings and activation co-occurrence. Each resulting cluster represents a group of experts that are likely redundant or interchangeable, and will serve as the basic unit for subsequent pruning decisions.

Let $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N]^{\top} \in \mathbf{R}^{N \times d}$ denote the structure-aware embeddings of N experts, and let $\mathbf{G} \in \{0, 1\}^{|\mathcal{D}| \times N}$ be the binary activation matrix, where $G_{xi} = 1$ if expert i is activated for input $x \in \mathcal{D}$.

We define the similarity between experts i and j as:

$$S_{ij} = \cos(\mathbf{z}_i, \mathbf{z}_j) \cdot \frac{\sum_{x \in \mathcal{D}} \mathbf{I}(g_i(x) = 1 \wedge g_j(x) = 1)}{\sqrt{A_i A_j}}, \tag{12}$$

where $A_i = \sum_{x \in \mathcal{D}} g_i(x)$ is the activation frequency of expert i. The first term captures structural similarity, while the second measures co-activation correlation.

The expert graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is then constructed from the similarity matrix $\mathbf{S}$. We apply the Girvan–Newman algorithm to detect community structures via edge betweenness centrality:

$$\psi(e) = \sum_{s<t} \frac{\sigma_{st}(e)}{\sigma_{st}}, \tag{13}$$

where σ_{st} is the total number of shortest paths between nodes s and t, and $\sigma_{st}(e)$ counts how many of those paths pass through edge e.

Edges with the highest $\psi(e)$ values are iteratively removed:

$$e_k^* = \arg\max_{e \in \mathcal{E}^{(k-1)}} \psi(e), \tag{14}$$

$$\mathcal{G}^{(k)} = \mathcal{G}^{(k-1)} \setminus \{e_k^*\}, \tag{15}$$

$$\mathcal{C}^{(k)} = \mathrm{ConnectedComponents}(\mathcal{G}^{(k)}). \tag{16}$$

This process continues until a predefined number of clusters is reached:

$$|\mathcal{C}^{(k)}| \geq K \quad and \quad \forall C \in \mathcal{C}^{(k)}, |C| > 1, \tag{17}$$

where K is the number of clusters for the current MoE layer.

Importantly, the value of K for each layer is not manually set. Instead, it is derived from the layer-wise pruning allocation strategy described previously, where the number of clusters equals the number of experts to retain in that layer. This ensures that the granularity of community partitioning aligns precisely with the global pruning objective, and that structurally important layers retain more expert clusters.

Activation-Aware Expert Selection

For each community C_k obtained from the partitioning step, we evaluate the importance of an expert $i \in C_k$ based on its cumulative activation frequency during training:

$$\mathrm{imp}(i) = \sum_{x \in \mathcal{D}} g_i(x), \tag{18}$$

where $g_i(x)$ is a binary indicator representing whether expert i was selected for routing on input x.

Experts within each community are ranked by their importance scores, and the most frequently activated ones are retained as the final expert set.

Efficiency Analysis

The pruning process is conducted once offline and completes within several minutes on a single A100 GPU. By reducing the number of experts in each layer, the pruned model requires less memory and achieves more efficient computation during inference, facilitating faster and more resource-friendly deployment in practical settings.

Model	Method	MMLU			GSM8K	HumanEval	Average
		Computer Science	Math	Business			
DeepSeek-V2-Lite	Base	53.00	32.21	49.54	30.94	32.30	31.83
	Random	19.00	12.32	17.53	0.057	0	0.02
	Seer Prune	29.00	26.54	30.09	2.06	0	0.69
	Group&Merge	33.50	24.65	31.64	3.96	1.20	2.58
	C-PRUNE	**51.50**	**33.56**	48.16	26.45	18.90	21.78
	C-GNN-PRUNE (Ours)	50.10	31.35	**49.73**	**28.13**	**23.20**	**24.84**
Qwen1.5-MoE-A2.7B	Base	47.68	34.03	52.45	53.58	49.40	50.99
	Random	14.50	13.81	11.04	10.44	12.90	11.79
	Seer Prune	29.00	25.54	15.10	15.32	26.20	22.24
	Group&Merge	35.50	19.61	40.93	25.38	28.00	31.44
	C-PRUNE	48.00	31.98	40.15	39.40	32.90	37.65
	C-GNN-PRUNE (Ours)	**50.50**	**35.68**	**45.69**	**43.54**	**43.30**	**43.38**

Table 1: Evaluation results of pruning methods on downstream tasks. Results are reported on MMLU (Computer Science, Math, Business), GSM8K, HumanEval, and their averaged performance.

Method	Pruning Rate	ARC-e	BoolQ	Hella	MMLU	OBQA	RTE	Wino	Average
None	0	84.01	85.35	64.88	67.88	35.00	70.40	75.93	69.06
Random	0.25	78.49	81.99	59.02	60.77	33.40	66.79	**75.85**	65.19
Frequency		78.83	77.03	59.38	55.18	33.60	57.40	75.69	62.44
EEMoE		**81.94**	83.64	**61.60**	58.72	33.00	67.87	75.37	66.02
STUN		81.82	83.09	60.84	**63.34**	31.60	68.59	72.69	66.00
C-GNN-PRUNE (Ours)		80.47	**84.49**	61.19	61.40	**32.60**	**69.31**	74.74	**66.31**
Random	0.5	68.35	78.59	53.32	49.23	**29.20**	62.82	69.93	58.78
Frequency		73.61	76.97	54.01	41.48	26.20	57.04	**73.48**	57.54
Wanda		74.16	76.64	53.16	52.21	27.00	63.90	70.96	59.72
EEMoE		78.16	81.35	**57.66**	47.30	29.00	61.37	72.85	61.10
C-GNN-PRUNE (Ours)		**80.03**	**83.73**	56.16	**49.76**	28.60	**66.79**	72.37	**62.49**

Table 2: Evaluation results of different pruning methods on Mixtral 8x7B across seven benchmarks: ARC-e, BoolQ, HellaSwag, MMLU, OBQA, RTE, and WinoGrande. None denotes the unpruned model. Due to availability constraints in prior work, some baselines only report results for one pruning ratio; thus, we present 25% and 50% settings in separate blocks for clarity.

Experiments

Experiment Setting

Model and Computational Setup: Our experiments are based on three state-of-the-art MoE models: DeepSeek-V2-Lite (Liu et al. 2024a), Qwen1.5-MoE-A2.7B (Team 2024), and Mixtral 8x7B (Jiang et al. 2024). These models represent diverse architectural designs and parameter scales, allowing us to thoroughly validate the generality and robustness of our method. The original unpruned models serve as the baseline reference for all comparative experiments.

Datasets and Evaluation Protocols: We evaluate the proposed method on a variety of benchmarks.

For DeepSeek-V2-Lite and Qwen1.5-MoE-A2.7B, we use selected subsets of MMLU (Wang et al. 2024), GSM8K (Cobbe et al. 2021), and HumanEval (Liu et al. 2023, 2024c).

For Mixtral 8×7B, we adopt the standardized EleutherAI LM Harness framework (Gao et al. 2024) for evaluation. The included benchmarks are ARC-e (Clark et al. 2018), BoolQ (Mihaylov et al. 2018), HellaSwag (Zellers et al. 2019), MMLU (Hendrycks et al. 2020), OBQA (Mihaylov et al. 2018), RTE (Wang et al. 2018), and WinoGrande (Sakaguchi et al. 2021).

Evaluation on Lightweight MoE Models

To assess the effectiveness of our proposed C-GNN-PRUNE method in practical pruning scenarios, we first conduct evaluations on two representative lightweight MoE models: DeepSeek-V2-Lite (15.7B) and Qwen1.5-MoE-A2.7B (14.3B). As summarized in Table 1, we compare our method against several established baselines under a uniform pruning rate of 20%, including Random, Group&Merge (Guo et al. 2025a), Seer Prune (Muzio, Sun, and He 2024), and C-PRUNE (Guo et al. 2025a).Bold numbers indicate the best performance in each group (this applies throughout).

For DeepSeek-V2-Lite, C-GNN-PRUNE achieves the highest scores on three key tasks: Business (49.73), GSM8K (28.13), and HumanEval (23.20). Although its performance on the Computer Science (50.10) and Math (31.35) subsets of MMLU is slightly lower than that of C-PRUNE, the differences remain small. Overall, C-GNN-PRUNE attains an average score of 24.84, outperforming all other methods under the same pruning ratio.

On Qwen1.5-MoE-A2.7B, C-GNN-PRUNE consistently outperforms all baselines across most benchmarks. It achieves the best scores on Mathematics (35.68), Business (45.69), GSM8K (43.54), and HumanEval (43.30), leading to the highest overall average score of 43.38. Compared to C-PRUNE, the improvements are substantial, particularly in

reasoning-heavy tasks such as GSM8K and HumanEval.

Taken together, these results demonstrate that C-GNN-PRUNE delivers competitive or superior performance across a range of tasks. Even in cases where its performance on a specific benchmark is not the best, the method compensates with stronger results on other tasks, leading to the highest overall scores. This indicates that C-GNN-PRUNE is a robust and generalizable pruning strategy capable of maintaining model utility across diverse evaluation metrics.

Evaluation on Large-Scale MoE Model

We further evaluate C-GNN-PRUNE on the large-scale Mixtral 8×7B model. Results under 25% and 50% pruning rates are reported in Table 2, alongside several baselines including Random, Frequency, Wanda (Sun et al. 2024), STUN (Lee et al. 2024), and EEMoE(Lu et al. 2024).

Under the 25% pruning rate, C-GNN-PRUNE achieves the best results on multiple tasks such as BoolQ, RTE, and OBQA. It performs slightly worse than EEMoE on ARC-e and HellaSwag, with marginal differences (within 1.5 points). On MMLU and WinoGrande, all methods yield comparable results. These findings demonstrate that while C-GNN-PRUNE may not dominate every individual task, it consistently maintains competitive performance without significant degradation.

At the 50% pruning rate, the superiority of C-GNN-PRUNE becomes more apparent. It achieves the highest overall average score (62.49), outperforming all baselines on most tasks. Notably, it exhibits clear gains over both EEMoE and Wanda on BoolQ, RTE, and ARC-e. Although its score on HellaSwag is slightly lower than EEMoE, the overall performance remains robust and stable.

In summary, C-GNN-PRUNE consistently demonstrates competitive or superior task-level performance across pruning settings. Despite minor performance drops on specific benchmarks, it achieves the highest average scores under both 25% and 50% pruning rates. These results confirm the robustness and generalizability of C-GNN-PRUNE on large-scale MoE architectures.

Ablation Studies

We perform a series of ablation studies to assess the contribution of key components in the C-GNN-PRUNE framework. All experiments are conducted under a fixed pruning rate of 20%, with consistent training configurations. Two representative MoE models are used: Qwen1.5-MoE-A2.7B and Mixtral 8×7B. This setup ensures broad validation across different MoE architectures.

Impact of Graph Construction Weight ω We first examine the effect of the graph construction parameter ω, which balances expert output similarity and routing divergence in edge computation. Table 3 shows that $\omega = 0.3$ achieves the highest performance on most benchmarks, indicating that combining both signals yields more informative expert graphs. In contrast, using either signal alone results in noticeable performance drops. This validates the design of a dual-factor graph that captures both functional and routing-level behaviors.

ω	ARC-e	BoolQ	Hella	OBQA	RTE	Wino
0.0	78.87	83.36	58.93	**34.40**	66.79	73.64
0.1	76.35	84.46	58.93	32.40	68.59	73.09
0.2	77.44	84.19	59.02	31.80	64.26	72.38
0.3	**80.47**	84.49	**61.19**	32.60	**69.31**	**74.74**
0.4	77.69	**84.68**	59.26	32.60	63.90	73.32
0.5	77.86	82.75	59.06	33.00	64.62	72.85
0.6	80.35	84.28	61.14	31.80	68.23	74.35
0.7	80.13	83.09	61.18	32.60	66.79	73.72
0.8	79.97	83.27	61.10	32.20	69.31	72.06
0.9	78.07	83.52	59.39	32.80	62.45	72.85
1.0	77.95	76.26	59.29	31.80	65.34	73.64

Table 3: Ablation study on graph construction weight ω, conducted on the Mixtral 8×7B model across six benchmarks. $\omega = 0.0$ and $\omega = 1.0$ use only output similarity or routing divergence, respectively.

Effectiveness of GNN-Based Embeddings and Entropy-Guided Allocation To evaluate the contribution of key design components in our pruning framework, we perform ablation experiments on Qwen1.5-MoE-A2.7B using the MMLU dataset. Specifically, we assess (1) the GNN-based structure-aware embedding module and (2) the entropy-guided layer-wise pruning allocation. Each ablation removes one component while keeping the rest of the framework intact. The domain-wise results are summarized in Table 4.

Method	Computer Science	Math	Business
w/o GNN	44.99	34.59	42.18
w/o Allocation	45.49	34.68	43.29
Ours	**50.50**	**35.68**	**45.69**

Table 4: Ablation study on Qwen1.5-MoE-A2.7B for MMLU. Results compare the full model with variants excluding the GNN encoder or entropy-guided expert allocation.

Impact of Structure-Aware Embeddings. In the GNN-free variant, we remove the graph autoencoder and directly perform community detection on the initial expert similarity graph. As shown in Table 4, this leads to consistent accuracy degradation across all MMLU domains. The average score drops from 43.96 to 40.59, with Computer Science and Business experiencing the largest declines. This highlights the importance of capturing higher-order structural dependencies among experts, especially for tasks with complex activation patterns.

Qualitative evidence further supports this finding. Figure 2 compares raw expert activations with the GNN-learned embeddings. The latter form clearer, more discriminative clusters.

Impact of Entropy-Guided Allocation. We further investigate the effect of adaptive layer-wise pruning allocation. In the ablated variant, we apply a fixed pruning ratio uniformly across all MoE layers, without considering sensitivity. As shown in Table 4, this results in a noticeable drop in performance, especially in the CS and Business domains. The

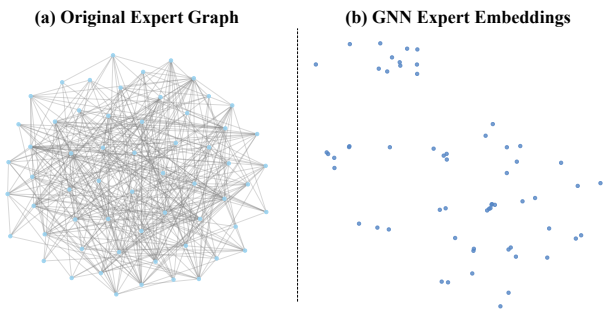

Figure 2: Expert embeddings before and after GNN modeling. (a) Raw activation. (b) GNN-learned embeddings.

comparison confirms that uniform allocation fails to preserve critical capacity in sensitive layers.

To interpret this behavior, we analyze layer-wise entropy scores $S(l)$ and the corresponding assigned pruning rates τ_l in the full model. As shown in Figure 3, layers with higher entropy, which reflects greater activation diversity and task importance, are pruned more conservatively. This allocation ensures structural preservation where needed, while still reducing redundancy in less critical layers.

Effect of Clustering Strategy in Expert Partitioning To assess the impact of the clustering strategy used in expert partitioning, we conduct ablation experiments on the Mixtral 8×7B model using six representative evaluation datasets: ARC-e, BoolQ, HellaSwag, OBQA, RTE, and WinoGrande. The goal is to evaluate how different clustering algorithms affect pruning performance under a unified expert similarity graph.

Specifically, we replace the Girvan-Newman algorithm in our method with standard unsupervised clustering techniques, including KMeans, Spectral Clustering, and Agglomerative Clustering. All methods operate on the same similarity graph, constructed using structure-aware embeddings and co-activation statistics.

Results are presented in Table 5. Our method achieves the highest accuracy on four of the six datasets: ARC-e (80.47), HellaSwag (61.19), OBQA (32.60), and RTE (69.31). Compared to KMeans and Spectral Clustering, it shows consistent improvements with margins up to +4.33 points. While Agglomerative Clustering slightly outperforms our method on BoolQ and WinoGrande, the gains are limited and not consistent across benchmarks. These results highlight the advantage of using community detection based on edge betweenness in expert partitioning.

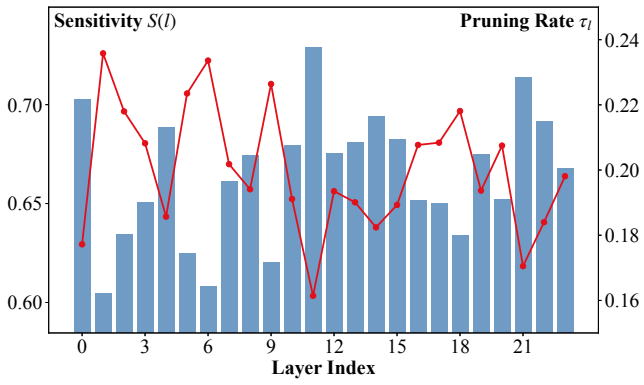

Figure 3: Layer sensitivity scores $S(l)$ and allocated pruning rates τ_l in Qwen1.5-MoE-A2.7B.

Dataset	Ours	KMeans	Spectral	Agglomerative
ARC-e	**80.47**	76.94	79.50	78.87
BoolQ	84.49	80.52	84.62	**85.50**
Hella	**61.19**	59.01	59.03	59.08
OBQA	**32.60**	31.40	31.80	30.80
RTE	**69.31**	64.98	64.26	67.15
Wino	74.74	**75.06**	73.88	74.51

Table 5: Performance comparison of different clustering methods under a unified expert partitioning framework on Mixtral 8×7B. All methods operate on the same structure-aware similarity graph.

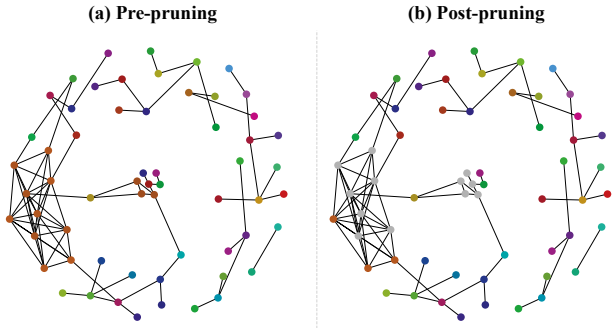

Figure 4: Expert graph before (a) and after (b) merging. Pruned experts are shown in gray.

To qualitatively illustrate the effect of our clustering strategy, Figure 4 presents the expert similarity graph before and after applying our community-based merging method. The visualization highlights the formation of modular expert groups, with pruned experts marked in gray. This structure reflects the ability of our method to expose redundant experts while preserving diversity across clusters, thereby supporting more effective and structured pruning.

Conclusion

In this paper, we propose C-GNN-PRUNE, a unified graph-based framework for structure-aware pruning of Mixture-of-Experts models. By constructing an expert interaction graph and leveraging a Graph Neural Network (GNN), our method learns embeddings that capture both functional similarity and routing behavior. We further introduce an entropy-guided strategy for adaptive pruning budget allocation and apply community detection to identify redundant expert clusters for structured pruning. Experiments on multiple open-source MoE models demonstrate the robustness and effectiveness of our approach across various pruning ratios.

Acknowledgments

This work was supported in part by Natural Science Foundation of China under Grants 62562048 and 62162047, Key Research and Development and Technology Transfer Program of Inner Mongolia Autonomous Region under Grants 2025YFHH0110 and 2025KJHZ0030, Natural Science Foundation of Inner Mongolia under Grants 2023MS06017 and 2023ZD18, Inner Mongolia Youth Science and Technology Talents support project under Grant NJYT24034, Inner Mongolia Autonomous Region higher Education Carbon Peak and Carbon Neutral Research Project under Grant STZX202322, Ministry of Education, and the Fund of Supporting the Reform and Development of Local Universities (Disciplinary Construction).

References

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Cai, W.; Jiang, J.; Wang, F.; Tang, J.; Kim, S.; and Huang, J. 2025. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*.

Chen, T.; Huang, S.; Xie, Y.; Jiao, B.; Jiang, D.; Zhou, H.; Li, J.; and Wei, F. 2022. Task-specific expert pruning for sparse mixture-of-experts. *arXiv preprint arXiv:2206.00277*.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv e-prints*, arXiv–1803.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*.

Das, R. J.; Ma, L.; and Shen, Z. 2023. Beyond Size: How Gradients Shape Pruning Decisions in Large Language Models. *CoRR*.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J.; Doran, C.; and Solorio, T., eds., *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)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

Dong, P.; Li, L.; Tang, Z.; Liu, X.; Pan, X.; Wang, Q.; and Chu, X. 2024. Pruner-zero: evolving symbolic pruning metric from scratch for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, 11346–11374.

Du, N.; Huang, Y.; Dai, A. M.; Tong, S.; Lepikhin, D.; Xu, Y.; Krikun, M.; Zhou, Y.; Yu, A. W.; Firat, O.; et al. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, 5547–5569. PMLR.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv e-prints*, arXiv–2407.

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

Gao, L.; Tow, J.; Abbasi, B.; Biderman, S.; Black, S.; DiPofi, A.; Foster, C.; Golding, L.; Hsu, J.; Le Noac'h, A.; Li, H.; McDonell, K.; Muennighoff, N.; Ociepa, C.; Phang, J.; Reynolds, L.; Schoelkopf, H.; Skowron, A.; Sutawika, L.; Tang, E.; Thite, A.; Wang, B.; Wang, K.; and Zou, A. 2024. The Language Model Evaluation Harness.

Guo, H.; Yao, J.; Wang, B.; Du, J.; Cao, S.; Di, D.; Zhang, S.; and Li, Z. 2025a. Cluster-Driven Expert Pruning for Mixture-of-Experts Large Language Models. *arXiv preprint arXiv:2504.07807*.

Guo, H.; Yao, J.; Wang, B.; Du, J.; Cao, S.; Di, D.; Zhang, S.; and Li, Z. 2025b. Cluster-Driven Expert Pruning for Mixture-of-Experts Large Language Models. *arXiv preprint arXiv:2504.07807*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Jiang, A. Q.; Sablayrolles, A.; Roux, A.; Mensch, A.; Savary, B.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Hanna, E. B.; Bressand, F.; et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Kaddour, J.; Harris, J.; Mozes, M.; Bradley, H.; Raileanu, R.; and McHardy, R. 2023. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169*.

Lee, J.; Hwang, S.-w.; Qiao, A.; Campos, D. F.; Yao, Z.; and He, Y. 2024. STUN: Structured-Then-Unstructured Pruning for Scalable MoE Pruning. *CoRR*.

Lin, B.; Tang, Z.; Ye, Y.; Cui, J.; Zhu, B.; Jin, P.; Zhang, J.; Ning, M.; and Yuan, L. 2024. MoE-LLaVA: Mixture of Experts for Large Vision-Language Models. *CoRR*.

Liu, A.; Feng, B.; Wang, B.; Wang, B.; Liu, B.; Zhao, C.; Dengr, C.; Ruan, C.; Dai, D.; Guo, D.; et al. 2024a. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*.

Liu, E.; Zhu, J.; Lin, Z.; Ning, X.; Blaschko, M. B.; Yan, S.; Dai, G.; Yang, H.; and Wang, Y. 2024b. Efficient Expert Pruning for Sparse Mixture-of-Experts Language Models: Enhancing Performance and Reducing Inference Costs. *CoRR*.

Liu, J.; Xia, C. S.; Wang, Y.; and Zhang, L. 2023. Is Your Code Generated by ChatGPT Really Correct? Rigorous Evaluation of Large Language Models for Code Generation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Liu, J.; Xie, S.; Wang, J.; Wei, Y.; Ding, Y.; and Zhang, L. 2024c. Evaluating Language Models for Efficient Code Generation. In *First Conference on Language Modeling*.

Liu, Q.; Wu, X.; Zhao, X.; Zhu, Y.; Xu, D.; Tian, F.; and Zheng, Y. 2024d. When moe meets llms: Parameter efficient fine-tuning for multi-task medical applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1104–1114.

Lu, X.; Liu, Q.; Xu, Y.; Zhou, A.; Huang, S.; Zhang, B.; Yan, J.; and Li, H. 2024. Not All Experts are Equal: Efficient Expert Pruning and Skipping for Mixture-of-Experts Large Language Models. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 6159–6172. Bangkok, Thailand: Association for Computational Linguistics.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2381–2391. Brussels, Belgium: Association for Computational Linguistics.

Muzio, A.; Sun, A.; and He, C. 2024. Seer-moe: Sparse expert efficiency through regularization for mixture-of-experts. *arXiv preprint arXiv:2404.05089*.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710.

Red, V.; Kelsic, E. D.; Mucha, P. J.; and Porter, M. A. 2011. Comparing community structure to characteristics in online collegiate social networks. *SIAM review*, 53(3): 526–543.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9): 99–106.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1): 61–80.

Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; and Dean, J. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations*.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2024. A Simple and Effective Pruning Approach for Large Language Models. In *The Twelfth International Conference on Learning Representations*.

Team, Q. 2024. Qwen1. 5-moe: Matching 7b model performance with 1/3 activated parameters.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In Linzen, T.; Chrupała, G.; and Alishahi, A., eds., *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 353–355. Brussels, Belgium: Association for Computational Linguistics.

Wang, Y.; Ma, X.; Zhang, G.; Ni, Y.; Chandra, A.; Guo, S.; Ren, W.; Arulraj, A.; He, X.; Jiang, Z.; et al. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

Xue, F.; Zheng, Z.; Fu, Y.; Ni, J.; Zheng, Z.; Zhou, W.; and You, Y. 2024. OpenMoE: An Early Effort on Open Mixture-of-Experts Language Models. In *Forty-first International Conference on Machine Learning*.

Yin, L.; Wu, Y.; Zhang, Z.; Hsieh, C.-Y.; Wang, Y.; Jia, Y.; Li, G.; Jaiswal, A.; Pechenizkiy, M.; Liang, Y.; et al. 2024. Outlier weighed layerwise sparsity (OWL) a missing secret sauce for pruning LLMs to high sparsity. In *Proceedings of the 41st International Conference on Machine Learning*, 57101–57115.

Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; and Leskovec, J. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 974–983.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Zhang, Z.; Liu, X.; Cheng, H.; Xu, C.; and Gao, J. 2024. Diversifying the Expert Knowledge for Task-Agnostic Pruning in Sparse Mixture-of-Experts. *CoRR*.

Zoph, B.; Bello, I.; Kumar, S.; Du, N.; Huang, Y.; Dean, J.; Shazeer, N.; and Fedus, W. 2022. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*.