COMPEL: Compensated Mixture-of-Experts Pruning with Expert-Layer distribution

Seohee Yoon[1], **Yong Suk Choi**[1]*
[1]Department of Computer Science, Hanyang University, Seoul, Korea
{seohee09, cys}@hanyang.ac.kr

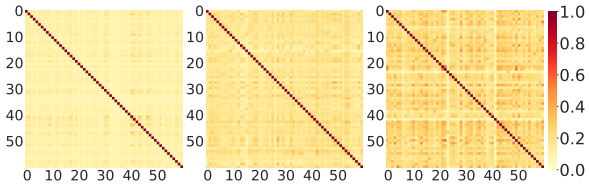
Abstract

Mixture-of-Experts (MoE) architectures have emerged as an effective approach for scaling Large Language Models (LLMs) by activating only a subset of experts during inference. Despite their computational efficiency, MoE models incur a substantial memory bottleneck from maintaining all expert parameters during inference. To address this challenge, numerous MoE pruning methods have been proposed. However, most existing methods adopt uniform pruning across layers, which fails to capture layer-wise variations in expert importance and redundancy. In this paper, we propose **CO**mpensated **M**oE **P**runing with **E**xpert-**L**ayer distribution (COMPEL). COMPEL performs layer-adaptive expert pruning by estimating expert importance using Fisher information and deriving layer importance from layer-wise outlier distributions, enabling pruning decisions that capture layer-wise heterogeneity. Furthermore, to mitigate performance degradation resulting from expert pruning, we propose a Fisher information guided expert weight compensation method. Experimental results on the Qwen1.5-MoE-A2.7B achieve near lossless performance at 25% expert pruning and maintains performance within a 4% margin even at 50% pruning. Moreover, COMPEL consistently outperforms existing pruning methods while substantially reducing inference latency and peak GPU memory usage. [1]

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable performance across a wide range of natural language processing tasks, with recent models such as GPT-4 (Achiam et al., 2023) and Gemini 2.5 (Comanici et al., 2025) exhibiting advanced capabilities in reasoning tasks. This performance improvement is consistent with increasing model size, as evidenced by scaling laws (Bai et al., 2024). However, this trend entails significant computational costs and resource constraints, which limit inference efficiency in resource constrained environments (Wan et al., 2024; Yuan et al., 2024).

To address these challenges, Mixture-of-Experts (MoE) (Fedus et al., 2022) architecture was proposed to achieve computational efficiency by activating only the top-k experts for each token. MoE has enabled the development of efficient models such as Mixtral 8x7B (Jiang et al., 2024). However, MoE models require all parameters to be loaded into memory leading to substantial memory consumption.



(a) Layer 3 (b) Layer 12 (c) Layer 22

Figure 1: Heatmap of Centered Kernel Alignment (CKA) similarities between experts across different MoE layers in Qwen1.5-MoE-A2.7B. Tick labels denote expert indices, and darker colors indicate higher similarity.

Accordingly, existing approaches aim to reduce redundancy in MoE models by applying expert-level pruning with uniform pruning rates across all layers (Lu et al., 2024; Lee et al., 2025). However, these methods often overlook the layer-wise heterogeneity. As illustrated in Figure 1, recent studies on inter expert similarity in MoE models have shown that the similarity between experts varies significantly across layers (Lo et al., 2025). However, applying a uniform pruning rate fails to account for this variability, resulting in the removal of distinct experts even in layers with low redundancy. Such

*Corresponding author.
[1]Our code is available at https://github.com/seohee0925/COMPEL.git

Findings of the Association for Computational Linguistics: ACL 2026, pages 30408–30423
July 2-7, 2026

indiscriminate pruning leads to the loss of unique information and consequently degrades model performance.

Furthermore, most expert pruning methods make pruning decisions based on fixed indicators, without explicitly modeling the relative importance of layers and experts. In contrast, we propose COmpensated MoE Pruning with Expert-Layer distribution (COMPEL), which transforms the pruning metric into a continuous optimization problem. Instead of using raw scores directly, COMPEL learns the importance distribution of experts and layers by aligning it with the structural importance of the network.

Specifically, we estimate expert sensitivity using Fisher information (McGowan et al., 2024; Navarrete et al., 2025), a computationally efficient alternative to the Hessian matrix. To capture layer-wise importance, we extend the Layerwise Outlier Distribution (LOD) (Ling et al., 2024) to MoE architectures. We utilize these estimated values as target distributions and optimize the learnable importance parameters by minimizing the Kullback–Leibler (KL) divergence against them (Luo and Wu, 2020). This ensures that the learned distribution consistently reflects both the expert-level sensitivity and the layer-level outlier ratio.

Within each layer, expert-level importance is estimated using Fisher information to define an expert-level pruning criterion. To quantify layer importance, we measure the KL divergence between the model softmax output distribution and a layer-level importance distribution, where the outlier distribution serves as the target distribution. Our proposed method, COMPEL, not only removes redundant experts but also mitigates pruning induced performance degradation through a compensation mechanism guided by pre computed Fisher information.

2 Related Work

2.1 Mixture of Experts

Mixture-of-Experts (MoE), first introduced by Jacobs et al. (1991), has evolved through the adoption of sparse gating mechanisms (Shazeer et al., 2017) and has become a fundamental building block for scaling Transformer-based architectures (Lepikhin et al., 2021; Li and Zhou, 2024). Contemporary decoder-only LLMs, including Mixtral-8x7B (Jiang et al., 2024), Qwen-MoE (Yang et al., 2024a, 2025a), and DeepSeek-MoE (Dai et al., 2024), exploit this paradigm to substantially increase model capacity while preserving computational efficiency by activating only a sparse subset of experts per token.

Despite these computational advantages, MoE models incur considerable memory overhead, since all experts must reside in memory during inference, which poses challenges for deployment in resource-constrained settings (Huang et al., 2024; Kong et al., 2024). To mitigate this limitation, expert pruning techniques have been proposed.

2.2 Expert Pruning for MoE

Expert pruning in MoE models has been extensively studied to remove redundant experts (Liu et al., 2024c; Yang et al., 2025b). Most existing methods estimate expert importance from local inference statistics to perform structured expert-level pruning (Xie et al., 2024; Liu et al., 2024b; Muzio et al., 2024). Beyond simple expert removal, recent hybrid schemes have also been proposed to combine expert pruning with weight-level and apply either structured expert-level pruning or hybrid schemes that combine expert pruning with weight-level sparsification for further compression (Lee et al., 2025). However, these approaches typically rely on uniform pruning strategies, failing to capture inter-layer heterogeneity. To address this, Bai et al. (2025) and Yang et al. (2025c) proposed layer-adaptive pruning approaches. However, accurately characterizing expert importance remains a significant challenge. Furthermore, existing methods largely lack mechanisms to compensate for the information loss induced by pruning.

2.3 Structured Compression

Recent LLM compression methods have explored structured compression and decomposition to reduce the computational and memory costs of dense models, including dimensionality reduction-based structured compression (Ashkboos et al., 2024) and joint module-level decomposition of consecutive Transformer subcomponents (Lin et al., 2025). More recently, this line of research has been extended to MoE models through inter-expert pruning with intra-expert decomposition (Yang et al., 2024b) and decomposition-based compression tailored to MoE (Li et al., 2025b). However, these approaches primarily focus on structural reduction or expert redundancy removal, and therefore do not address layer-wise heterogeneity under sparse routing. In particular, although MoDeGPT (Lin et al., 2025) uses layer-wise signals to guide glob-

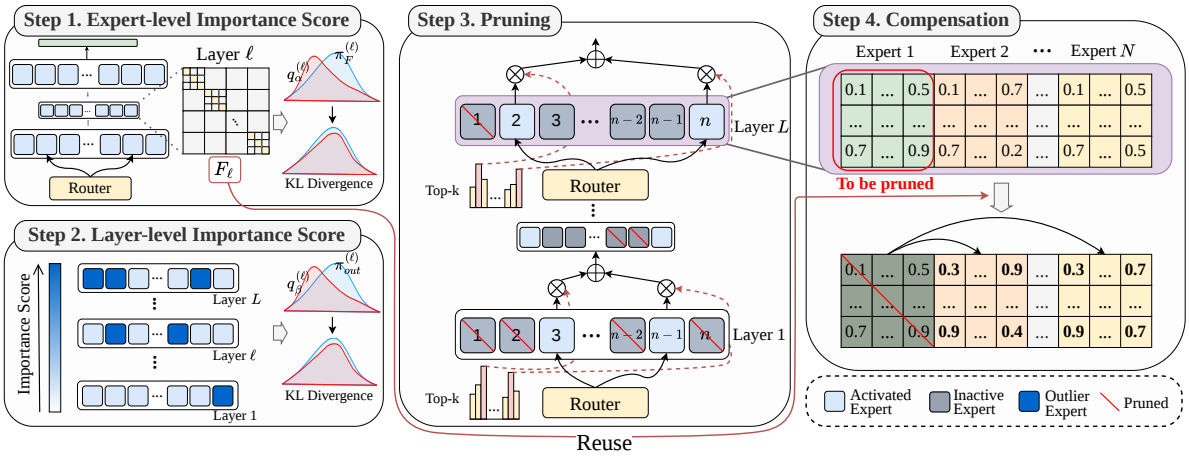


Figure 2: Overview of our proposed COMPEL. Expert importance at each layer is quantified using Fisher-based sensitivity, while layer importance is defined by outlier-aware expert activation statistics. The blue curves indicate the target importance distributions, and the red curves correspond to the learned distributions parameterized by α (expert-level) and β (layer-level), both optimized by minimizing KL divergence. Experts are pruned according to the resulting joint importance score, and Fisher-based weight compensation is subsequently applied to prevent performance degradation caused by pruning.

ally informed compression, it remains a module-level compression framework and does not explicitly model expert-level selection or information loss induced by pruning in sparse MoE layers.

2.4 Second-order Information for Compensation

Second-order information has been widely used to compensate for performance degradation caused by structural pruning, most notably in the Optimal Brain Surgeon (OBS) framework proposed by Hassibi and Stork (1992). Recent studies have extended OBS-style compensation to LLMs, demonstrating that curvature-aware updates can effectively mitigate pruning-induced errors (Ling et al., 2024). However, exact computation of the Hessian matrix is infeasible for large-scale models due to its quadratic complexity with respect to the number of parameters (Kang et al., 2025). Consequently, the Fisher information matrix (FIM) is commonly adopted as a tractable surrogate for second-order curvature information (Navarrete et al., 2025).

The FIM is defined as the expected outer product of gradients of the negative log-likelihood,

$$F = \mathbb{E}_{p(D|\theta)} \left[\nabla_\theta \ell(\theta; D) \nabla_\theta \ell(\theta; D)^\top \right] \quad (1)$$

and coincides with the expected Hessian at a local optimum under maximum likelihood estimation. This relationship enables Fisher-based criteria to provide reliable estimates of parameter importance and supports pruning and compensation beyond first-order approximations (Navarrete et al., 2025; Garcia et al., 2023).

Despite these advantages, applying second-order compensation to Mixture-of-Experts (MoE) models remains challenging. Token-dependent routing leads to sparse and asynchronous expert activation, which conflicts with the parameter co-activation assumptions of conventional OBS-based formulations. Accordingly, Li et al. (2025a) limit the use of second-order information to expert importance estimation, relying on diagonal FIM approximations and OBS-style loss increase estimates. In this setting, our goal is not to derive a full OBS-style compensation for MoE, but to use Fisher as a tractable curvature proxy to guide a practical compensation mechanism compatible with sparse expert activation.

3 Method

3.1 Preliminary: Mixture-of-Experts

In an MoE layer, the dense Feed-Forward Network (FFN) in a Transformer block is replaced with a collection of N independent expert networks, denoted as $\{FFN_i\}_{i=1}^{N}$, together with a routing function that determines expert assignment on a per token basis (Shazeer et al., 2017; Fedus et al., 2022).

Given an input token representation x, the router computes a routing logit l_i for each expert i. Instead of activating all experts, only the top-k experts with the highest routing logits are selected. The routing weights of the selected experts are then

normalized using a softmax operation restricted to the top-k set (Shazeer et al., 2017):

$$w_i = \frac{\exp(l_i)}{\sum_{j \in \text{Top-}k} \exp(l_j)}, \quad i \in \text{top-}k \tag{2}$$

The output of the MoE layer is computed as a weighted sum of the outputs produced by the selected experts:

$$o(x) = \sum_{i \in \text{Top-k}} w_i \cdot FFN_i(x) \tag{3}$$

Each expert FFN_i follows the standard Transformer feed-forward architecture (Vaswani et al., 2017) and is parameterized independently:

$$FFN_i(x) = W_{\text{down}}^{(i)}\left(\sigma\left(W_{\text{up}}^{(i)}x\right)\right) \tag{4}$$

where $W_{\text{up}}^{(i)}$ and $W_{\text{down}}^{(i)}$ denote expert-specific projection matrices, and $\sigma(\cdot)$ is a non-linear activation function.

As a result, only a subset of experts participates in the computation for each token, while the parameters of all experts are maintained across layers.

3.2 Expert Importance Estimation

Expert-level Importance Score. Ashkboos et al. (2024); Men et al. (2025); Xia et al. (2023) have demonstrated the effectiveness of gradient-based sensitivity measures for LLM compression and pruning. Motivated by these findings, we employ a Fisher-based criterion to estimate expert-level importance. Extending the curvature-based sensitivity framework (LeCun et al., 1989) to the expert-level, we define a score $F_{l,i}$ that characterizes the contribution of expert i in layer l by capturing the local curvature of the loss function $\mathcal{L}$ with respect to the expert-specific parameters $\theta_{l,i}$. Consequently, $F_{l,i}$ serves as a proxy for expert importance by reflecting the sensitivity of model performance to parameter perturbations. The score $F_{l,i}$ is defined as follows:

$$F_{l,i} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{cal}}}\left[\left\|\nabla_{\theta_{l,i}} \log p(y \mid x;\theta)\right\|_2^2\right] \tag{5}$$

where this formulation approximates the diagonal of the FIM (Kirkpatrick et al., 2017).

To assess the relative importance of experts within each layer, we normalize the expert-level Fisher sensitivity scores to obtain a target distribution,

$$\pi_F^{(l,i)} = \frac{F_{l,i}}{\sum_j F_{l,j}}, \tag{6}$$

where $\pi_F^{(l,i)}$ denotes the normalized importance of expert i in layer l.

We parameterize a Softmax distribution $q_\alpha^{(l)}$ with learnable expert-level importance parameters $\alpha_i^{(l)}$ and train it to match the target distribution $\pi_F^{(l)}$. The parameters α are optimized by minimizing the KL divergence,

$$\mathcal{L}_{\text{Fisher-}\alpha} = \mu_\alpha \sum_l \text{KL}\left(\pi_F^{(l)} \,\|\, q_\alpha^{(l)}\right). \tag{7}$$

Through this optimization, $\alpha_{l,i}$ serves as a learnable expert-level importance score that reflects the relative contribution of each expert.

Layer-level Importance Score. In addition to expert-level importance, we consider the relative importance of layers, grounded in observations that layers contribute unevenly to model inference and exhibit differing redundancy patterns (Men et al., 2025).

Following prior work on activation outliers (Yin et al., 2023; Sun et al., 2023), we quantify layer-level importance based on the distribution of expert activations within each layer. This distribution effectively characterizes what is referred to as the LOD in structured pruning (Ling et al., 2024), where higher outlier concentrations indicate greater importance to pruning.

Let $A_{l,i}$ denote the activation magnitude of expert i in layer l, and let $\bar{A}_l$ be the average activation of layer l. We define a layer-wise outlier ratio D_l as

$$D_l = \frac{1}{E_l}\sum_{i=1}^{E_l} \mathbb{I}\left(A_{l,i} > M \cdot \bar{A}_l\right) \tag{8}$$

where M serves as a threshold multiplier defining the outlier boundary. Any expert with an activation magnitude exceeding M times the layer-wise average $\bar{A}_l$ is classified as an outlier. This criterion aligns with Yin et al. (2023), who identify that activations surpassing this relative magnitude are critical for maintaining model performance. Thus, D_l strictly quantifies the proportion of these high-magnitude experts in each layer.

To compare importance across layers, we normalize D_l to obtain a layer-level target distribution,

$$\pi_{\text{out}}^{(l)} = \frac{D_l}{\sum_k D_l} \tag{9}$$

We optimize the layer-level importance parameters β_l by minimizing the KL divergence between

$\pi_{\text{out}}^{(l)}$ and a parameterized distribution q_β,

$$\mathcal{L}_{\text{out-}\beta} = \mu_\beta \cdot \mathrm{KL}\left(\pi_{\text{out}}^{(l)} \,\|\, q_\beta\right) \tag{10}$$

The learned parameter β_l serves as a layer-level importance score that modulates the pruning intensity applied to each MoE layer.

3.3 Pruning and Compensation

Expert Pruning Criterion. We define an adaptive pruning criterion that jointly captures expert-level and layer-level importance, leveraging the methodology proposed in Bai et al. (2025). For each expert i in layer l, the pruning score is computed as the product of the expert-level importance term $\alpha_i^{(l)}$ and the corresponding layer-level importance factor $\beta^{(l)}$. This multiplicative form follows a hierarchical factorization in which the expert-level term determines the within-layer ranking of experts, while the layer-level factor calibrates the pruning intensity across layers. Because the layer-level factor is shared by all experts in layer l, it preserves the within-layer ordering induced by the expert-level term while enabling globally comparable ranking under a fixed pruning budget.

The resulting score quantifies the relative impact of removing an expert on overall model performance, comparable to recent scoring-based pruning metrics (Sun et al., 2023).

Fisher-based Expert Compensation. After pruning, a compensation step is performed to mitigate the performance degradation caused by the removal of experts. While second-order compensation methods based on the Optimal Brain Surgeon (OBS) framework have proven effective for dense models (Frantar and Alistarh, 2023), directly applying them to the sparse and dynamic structure of MoE remains computationally prohibitive (Li et al., 2025a). Under a local second-order approximation of the loss, OBS-style compensation admits a closed-form update in terms of the inverse Hessian H^{-1}. However, such a derivation is difficult to obtain for MoE, where token-dependent sparse routing leads to asynchronous expert activation and violates the parameter co-activation assumptions underlying conventional OBS formulations.

To address this, we propose a computationally efficient compensation mechanism that redistributes the importance of pruned experts to the surviving experts. We assume that experts with higher Fisher information values possess greater capability to absorb the functionality of removed components. Accordingly, rather than uniformly distributing the pruned weights, we redistribute the parameters of the pruned set $R^{(l)}$ to the surviving set $S^{(l)}$ proportional to the receiver's importance

$$\theta_{l,j} \leftarrow \theta_{l,j} + \gamma \sum_{i \in R^{(l)}} \left(\frac{F_{l,j}}{\sum_{k \in S^{(l)}} F_{l,k} + \epsilon} \cdot \theta_{l,i} \right), \quad \forall j \in S^{(l)} \tag{11}$$

where $F_{l,j}$ is the Fisher importance score of the survived expert j as defined in Sec. 3.2, and the denominator acts as a normalization term across the surviving set $S^{(l)}$. The hyperparameter γ modulates the overall magnitude of compensation, while ϵ ensures numerical stability.

By prioritizing experts with higher sensitivity, this heuristic effectively concentrates the pruned information into the most critical surviving experts, approximating the recovery of representational power without the computational overhead of second-order optimization.

4 Experiment

4.1 Model Settings

To evaluate the effectiveness of expert-level pruning, we conduct experiments using MoE models composed of multiple experts. The detailed configurations of the models used in our experiments are summarized below.

- **Qwen1.5-MoE-A2.7B** (Team, 2024): A 24-layer model with 60 routed experts and 4 shared experts per layer. It contains 14.3B total parameters, of which 2.7B are activated during inference.
- **DeepSeek-V2-Lite** (Liu et al., 2024a): A 27-layer model with 64 routed experts and 2 shared experts per layer. The model has 16B total parameters, with 2.4B parameters activated at inference time.
- **OLMoE-1B-7B-0924** (Muennighoff et al., 2024): A 16-layer model with 64 routed experts per layer. It comprises 7B total parameters, with 1B parameters activated during inference.

4.2 Implementation Details

We empirically determined the outlier threshold as $M = 5.0$ for the LOD Formulation (8). To facilitate the transfer of knowledge from pruned to

Sparsity	Method	PPL ↓	ARC-c	ARC-e	BoolQ	HellaSwag	PIQA	OBQA	RTE	WinoGrande	Avg.
Qwen1.5-MoE-A2.7B											
0%	None	7.06	56.57	82.79	83.12	58.38	80.03	37.00	72.92	76.16	68.37
25%	NAEE	10.11	47.56	77.65	75.72	52.69	75.92	34.52	67.80	75.85	63.46
	STUN	9.46	51.84	79.93	80.18	57.96	78.07	33.14	69.76	73.48	65.55
	MoE-I2	9.10	50.96	78.42	76.94	56.21	77.88	33.29	70.84	71.02	64.44
	DiEP	8.97	52.61	79.48	81.02	57.42	78.64	35.27	70.68	73.91	66.13
	COMPEL	**7.94**	**53.67**	**80.62**	**83.62**	**60.52**	**81.33**	**37.34**	**72.40**	**77.22**	**68.34**
50%	NAEE	13.84	43.78	72.54	72.52	49.95	72.96	30.80	65.82	67.86	59.53
	STUN	10.23	45.27	75.86	74.18	52.06	74.03	31.26	66.37	70.94	61.25
	MoE-I2	9.14	45.92	70.38	71.06	53.41	73.62	31.58	67.02	66.11	59.89
	DiEP	10.51	44.82	72.36	71.24	52.91	72.18	31.94	66.02	64.27	59.47
	COMPEL	**8.03**	**49.57**	**78.83**	**77.43**	**56.52**	**76.59**	**34.81**	**69.84**	**72.66**	**64.53**
DeepSeek-V2-Lite											
0%	None	10.22	43.52	75.67	74.62	54.99	79.22	31.00	62.09	67.01	61.02
25%	NAEE	11.15	38.27	70.41	69.18	50.37	74.56	27.68	58.14	62.73	56.42
	STUN	10.89	41.20	72.96	71.02	54.02	77.05	28.92	58.77	64.72	58.58
	MoE-I2	11.49	39.94	73.92	72.85	52.64	76.48	29.35	60.41	62.91	58.56
	DiEP	11.34	41.05	73.60	73.02	53.60	76.62	29.07	59.96	64.22	58.89
	COMPEL	**10.65**	**42.66**	**74.85**	**74.03**	**55.27**	**78.31**	**30.64**	**61.88**	**66.05**	**60.46**
50%	NAEE	11.51	33.62	63.41	63.88	46.27	68.92	24.38	54.44	58.06	51.62
	STUN	11.03	36.18	67.42	66.05	49.11	72.45	25.71	57.40	60.34	54.33
	MoE-I2	12.62	35.44	65.98	67.21	49.65	70.84	26.46	57.19	58.47	53.91
	DiEP	11.46	36.05	68.15	65.42	50.12	72.12	26.05	56.02	60.88	54.35
	COMPEL	**10.87**	**39.18**	**71.34**	**69.74**	**52.22**	**74.56**	**28.74**	**58.81**	**63.49**	**57.26**
OLMoE-1B-7B-0924											
0%	None	13.52	47.10	78.28	74.71	57.93	80.09	33.20	52.35	68.19	61.98
25%	NAEE	13.96	42.30	73.65	69.88	53.10	75.42	30.84	48.10	65.20	57.81
	STUN	14.64	45.86	76.92	72.84	56.08	78.01	32.76	50.62	66.94	59.92
	MoE-I2	13.76	44.98	75.80	71.36	55.12	77.10	31.40	50.10	66.02	59.23
	DiEP	13.84	46.38	76.44	73.10	56.43	76.42	32.28	51.08	67.14	60.28
	COMPEL	**13.66**	**48.62**	**78.95**	**75.96**	**58.74**	**81.20**	**33.96**	**53.18**	**69.84**	**62.56**
50%	NAEE	14.09	38.92	69.84	65.12	49.08	71.36	28.94	46.20	61.30	53.85
	STUN	14.84	41.86	70.48	68.04	51.72	70.98	29.88	44.62	60.88	56.68
	MoE-I2	13.97	41.10	71.36	66.92	52.14	73.22	29.66	45.20	61.74	55.17
	DiEP	14.01	42.34	72.06	67.38	51.98	73.10	29.92	45.48	60.02	55.29
	COMPEL	**13.76**	**45.82**	**76.12**	**71.28**	**55.34**	**75.04**	**30.10**	**48.86**	**62.88**	**58.18**

Table 1: Zero-shot performance of expert pruning methods on Qwen1.5-MoE-A2.7B, DeepSeek-V2-Lite, and OLMoE-1B-7B-0924. Expert sparsity denotes the proportion of pruned experts (25% and 50%). PPL is perplexity on WikiText2, and Avg. denotes the average accuracy across tasks in the LM Evaluation Harness. Bold and underlined values indicate the best and second-best results, respectively.

retained experts during the weight compensation phase, the scaling factor was fixed at $\gamma = 0.1$ to modulate the magnitude of the weight updates. We optimized the learnable importance parameters, α and β, using the Adam optimizer. This optimization process was executed over 3 epochs with a batch size of 4 and a learning rate of 5e-3. All experimental evaluations were conducted in a computational environment equipped with four NVIDIA RTX 3080 GPUs.

4.3 Evaluation Setup

Calibration and Pruning efficiency. The C4 (Colossal Clean Crawled Corpus) dataset (Raffel et al., 2020) was utilized for calibration to ensure broad generalization without domain-specific bias. Specifically, calibration was performed using 128 sequences with a length of 1024 tokens. This setup serves as the basis for the pruning-stage efficiency reported in Table 4. Additional results regarding different sequences are provided in Appendix A.4.

Inference Efficiency. We evaluated the practical efficiency by measuring token generation throughput and peak GPU memory usage under a consistent evaluation setup.

Task-level Performance. We report results on eight zero-shot benchmarks: ARC-easy and ARC-

Model	Method	Mem (GB) ↓	Mem. Ratio ↓	TTFT (s) ↓	ITL (ms) ↓	Throughput (tok/s) ↑	Speedup ↑
Qwen1.5-MoE-A2.7B	None	9.48	1.00×	0.35	22.12	45.21	1.00×
	NAEE	8.82	0.93×	0.33	19.15	52.23	1.16×
	STUN	8.54	0.90×	0.31	18.28	54.71	1.21×
	MoE-I2	8.35	0.88×	0.30	17.15	58.32	1.29×
	DiEP	8.16	0.86×	0.28	15.97	62.64	1.39×
	COMPEL	**7.78**	**0.82×**	**0.26**	**14.41**	**69.42**	**1.54×**
DeepSeek-V2-Lite	None	9.95	1.00×	0.41	24.63	40.60	1.00×
	COMPEL	**8.45**	**0.85×**	**0.31**	**16.34**	**61.18**	**1.51×**
OLMoE-1B-7B-0924	None	6.12	1.00×	0.24	17.95	55.72	1.00×
	COMPEL	**4.83**	**0.79×**	**0.18**	**12.48**	**80.15**	**1.44×**

Table 2: Inference-time latency, throughput, and peak GPU memory. All values are measured under the same inference setup, and Speedup is normalized to the unpruned model.

challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), RTE (Wang et al., 2018), and WinoGrande (Sakaguchi et al., 2021), evaluated using the EleutherAI LM Evaluation Harness (Gao et al., 2024). To provide a more fine-grained analysis of pruning effects on language modeling capability, we report perplexity (PPL) on WikiText-2 (Merity et al., 2016). Additional perplexity results on the Penn Treebank (PTB) (Marcus et al., 1993) and LAMBADA (Paperno et al., 2016) are provided in Appendix A.1.

4.4 Baselines

We compare our proposed method against four representative pruning techniques to evaluate COMPEL. NAEE (Lu et al., 2024) enumerates possible expert combinations and searches for the optimal subset that minimizes the reconstruction loss. STUN (Lee et al., 2025) proposes a multi-stage strategy that simultaneously achieves structured and unstructured sparsity by sequentially applying expert pruning followed by unstructured pruning. MoE-I2 (Yang et al., 2024b) performs structural compression by combining inter-expert pruning and intra-expert low-rank decomposition. Finally, DiEP (Bai et al., 2025) performs adaptive MoE compression through differentiable expert pruning, enabling pruning decisions to be optimized in a continuous manner.

5 Results

5.1 Main Results

Downstream Tasks. Table 1 presents a comparative evaluation of COMPEL and existing pruning methods across three MoE models under various sparsity levels. The results demonstrate that COMPEL consistently achieves the best performance across all models and sparsity regimes. Notably, COMPEL exhibits strong robustness under the 50% sparsity setting, where the risk of information loss is most pronounced, substantially outperforming baseline approaches. For instance, on Qwen1.5-MoE at 50% sparsity, COMPEL improves the average score by approximately 5 percentage points over STUN, and shows clear advantages on reasoning-intensive benchmarks such as ARC-c and BoolQ. Moreover, under the 25% sparsity setting, COMPEL attains lossless compression across all models. In the case of OLMoE, COMPEL even surpasses the original dense model in terms of average accuracy, indicating that removing redundant experts can contribute to improved generalization.

Perplexity. We measure perplexity on WikiText-2 with a sequence length of 1024 to evaluate the impact of pruning on language modeling quality. As shown in Table 3, pruning without compensation leads to increased perplexity as sparsity increases, whereas Fisher-based compensation consistently alleviates this degradation. At 25% sparsity, compensated models achieve perplexity close to the unpruned model, and even at 50% sparsity, they outperform variants without compensation.

5.2 Efficiency Analysis

Inference Cost. To evaluate the impact of pruning on inference efficiency, we report peak GPU memory usage, latency, and throughput, as shown in Table 2. Inference latency is measured using Time to First Token (TTFT) and Inter-Token Latency (ITL), which respectively quantify the prefill latency and the per-token decoding cost, following

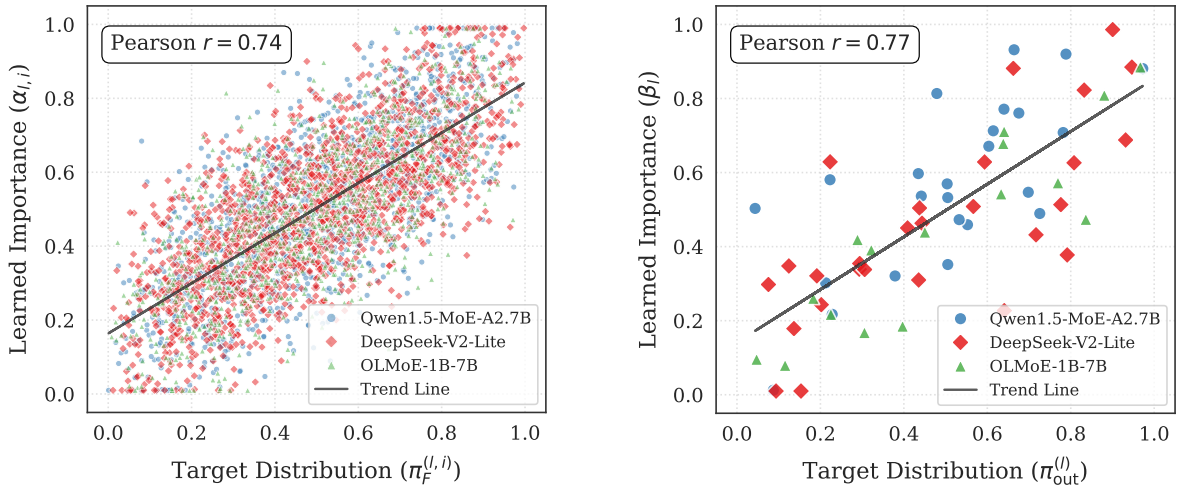


Figure 3: Correlation between the learned importance parameters and their corresponding target distributions. The relationship between expert-level importance α and the Fisher-based target distribution π_F is shown on the left, while the correlation between layer-level importance β and the outlier-based target distribution π_{out} is shown on the right.

Model	Sparsity	Prun.	Comp.	PPL ↓
Qwen1.5-MoE-A2.7B	0%			7.06
	25%	✓		8.06
	25%	✓	✓	**7.94**
	50%	✓		8.97
	50%	✓	✓	**8.03**
DeepSeek-V2-Lite	0%			10.22
	25%	✓		10.78
	25%	✓	✓	**10.65**
	50%	✓		11.06
	50%	✓	✓	**10.87**
OLMoE-1B-7B-0924	0%			13.52
	25%	✓		14.90
	25%	✓	✓	**13.66**
	50%	✓		14.06
	50%	✓	✓	**13.76**

Table 3: WikiText2 perplexity (PPL) under different sparsity settings. Prun. corresponds to pruning, and Comp. corresponds to compensation.

Method	Pruning Time (h)	Mem (GB)	GPU ↓
None	–	9.48	1.00×
DiEP	2.21	7.78	1.26×
COMPEL	**0.66**	**6.20**	**0.99×**

Table 4: Comparison of computational efficiency during the pruning stage on Qwen1.5-MoE-A2.7B. Pruning Time denotes the total optimization time, including importance estimation and compensation.

Configuration	Prun.	Comp.	Avg. ↑	PPL ↓
None			68.37	7.06
Expert-level (α)	✓		59.10	12.89
Layer-level (β)	✓		58.85	13.05
Joint (α, β)	✓		61.40	11.54
COMPEL	✓	✓	**64.53**	**8.03**

Table 5: Ablation study of COMPEL. We report average accuracy (Avg.) over LM Evaluation Harness and perplexity (PPL) on WikiText-2.

the definition in Chitty-Venkata et al. (2025). Detailed metric definitions and experimental settings are provided in Appendix B. COMPEL consistently improves inference efficiency. In particular, COMPEL reduces peak GPU memory usage to approximately 0.8× of the unpruned model while achieving around 1.5× throughput speedup, together with lower TTFT and ITL.

Optimization Cost. We evaluate the optimization cost of COMPEL during the pruning stage and compare it with DiEP. As summarized in Table 4, COMPEL incurs substantially lower optimization overhead than DiEP while maintaining a comparable memory footprint. This improvement arises from avoiding iterative pruning optimization and instead leveraging calibration-based importance estimation and layer-adaptive pruning.

5.3 Ablation Studies

Sparsity	Var($\log \alpha$)	Var($\log \beta$)	Corr(joint, $\log \alpha$)	Corr(joint, $\log \beta$)
25%	0.41	0.18	0.44	0.72
50%	0.37	0.24	0.65	0.68

Table 6: Log-space dominance analysis of the joint importance score.

Effectiveness of Pruning Criterion. Table 5 shows that pruning based on a single criterion, either expert-level (α) or layer-level (β), leads to substantial performance degradation. Using only one criterion reduces accuracy by 9 and increases perplexity by 6 relative to the unpruned model.

Jointly modeling expert- and layer-level importance mitigates this degradation. With Fisher-based compensation, accuracy is recovered by 3 and perplexity is reduced by 3 compared to pruning without compensation.

Robustness of the Joint Pruning Score. Table 6 reports log-space variance and correlation statistics of the joint score. Neither component exhibits overwhelming variance dominance, and the joint score remains substantially correlated with both terms. These results support the multiplicative combination of expert-level and layer-level importance.

Alignment of Learned Importance Parameters. Figure 3 evaluates the alignment between the learned importance parameters and their target distributions using the Pearson correlation coefficient (Pearson, 1896), a standard measure of linear association between continuous variables (Schober et al., 2018). We observe positive correlations, with coefficients of 0.74 at the expert level and 0.77 at the layer level, indicating that the learned importance parameters closely follow the intended pruning criteria. In the expert-level analysis, each point represents an expert across all MoE layers, while in the layer-level analysis, each point corresponds to an MoE layer.

6 Conclusion

In this paper, we propose COMPEL, an optimization-based pruning framework for MoE models that integrates adaptive expert pruning with compensation. COMPEL leverages Fisher-based expert importance and outlier-based layer sensitivity to guide layer-adaptive pruning, and applies Fisher-based compensation to mitigate performance degradation caused by expert removal. As a result, COMPEL preserves model performance while reducing memory usage and accelerating inference.

Limitations

COMPEL has two limitations due to computational constraints. First, experiments were restricted to models up to 16B, leaving validation on larger architectures like Mixtral-8×7B for future work. Second, we could not perform full fine-tuning to jointly optimize the learnable parameters (α, β) with the model weights, which could potentially yield further improvements.

Acknowledgments

This work was supported by the Institute of Information and communications Technology Planning and evaluation (IITP) grant (No.RS-2025-25422680, No. RS-2020-II201373), and the National Research Foundation of Korea (NRF) grant (No. RS-2025-00520618) funded by the Korean Government (MSIT).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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. In *The Twelfth International Conference on Learning Representations*.

Guangji Bai, Zheng Chai, Chen Ling, Shiyu Wang, Jiaying Lu, Nan Zhang, Tingwei Shi, Ziyang Yu, Mengdan Zhu, Yifei Zhang, and 1 others. 2024. Beyond efficiency: A systematic survey of resource-efficient large language models. *arXiv preprint arXiv:2401.00625*.

Sikai Bai, Haoxi Li, Jie Zhang, Zicong Hong, and Song Guo. 2025. Diep: Adaptive mixture-of-experts compression through differentiable expert pruning. *arXiv preprint arXiv:2509.16105*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Krishna Teja Chitty-Venkata, Sylvia Howland, Golara Azar, Daria Soboleva, Natalia Vassilieva, Siddhisanket Raskar, Murali Emani, and Venkatram Vishwanath. 2025. Moe-inference-bench: Performance

evaluation of mixture of expert large language and vision models. In *Proceedings of the SC'25 Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1502–1511.

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 North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2924–2936. 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. *arXiv preprint arXiv:1803.05457*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

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

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

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International conference on machine learning*, pages 10323–10337. PMLR.

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.

Jezabel R Garcia, Federica Freddi, Stathi Fotiadis, Maolin Li, Sattar Vakili, Alberto Bernacchia, and Guillaume Hennequin. 2023. Fisher-legendre (fishleg) optimization of deep neural networks. In *The Eleventh International Conference on Learning Representations*.

Babak Hassibi and David Stork. 1992. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5.

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

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

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, and 1 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Yuhan Kang, Zhongdi Luo, Mei Wen, Yang Shi, Jun He, Jianchao Yang, Zeyu Xue, Jing Feng, and Xinwang Liu. 2025. Hwpq: Hessian-free weight pruning-quantization for llm compression and acceleration. *arXiv e-prints*, pages arXiv–2501.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

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

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

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*, pages 13660–13676.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*.

Ke Li, Zheng Yang, Zhongbin Zhou, Feng Xue, Zhonglin Jiang, and Wenxiao Wang. 2025a. Heapr: Hessian-based efficient atomic expert pruning in output space. *arXiv preprint arXiv:2509.22299*.

Wei Li, Lujun Li, Hao Gu, You-Liang Huang, Mark G. Lee, Shengjie Sun, Wei Xue, and Yike Guo. 2025b. Moe-SVD: Structured mixture-of-experts LLMs

compression via singular value decomposition. In *Forty-second International Conference on Machine Learning*.

Ziyue Li and Tianyi Zhou. 2024. Your mixture-of-experts llm is secretly an embedding model for free. *arXiv preprint arXiv:2410.10814*.

Chi-Heng Lin, Shangqian Gao, James Seale Smith, Abhishek Patel, Shikhar Tuli, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. 2025. ModeGPT: Modular decomposition for large language model compression. In *The Thirteenth International Conference on Learning Representations*.

Gui Ling, Ziyang Wang, and Qingwen Liu. 2024. Slimgpt: Layer-wise structured pruning for large language models. *Advances in Neural Information Processing Systems*, 37:107112–107137.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, and 1 others. 2024a. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*.

Enshu Liu, Junyi Zhu, Zinan Lin, Xuefei Ning, Matthew B Blaschko, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. 2024b. Efficient expert pruning for sparse mixture-of-experts language models: Enhancing performance and reducing inference costs. *arXiv preprint arXiv:2407.00945*.

Jiacheng Liu, Peng Tang, Wenfeng Wang, Yuhang Ren, Xiaofeng Hou, Pheng-Ann Heng, Minyi Guo, and Chao Li. 2024c. A survey on inference optimization techniques for mixture of experts models. *arXiv preprint arXiv:2412.14219*.

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

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

Jian-Hao Luo and Jianxin Wu. 2020. Neural network pruning with residual-connections and limited-data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1458–1467.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Jamie McGowan, Wei Sheng Lai, Weibin Chen, Henry Aldridge, Jools Clarke, Jezabel Garcia, Rui Xia, Yilei Liang, Guillaume Hennequin, and Alberto Bernacchia. 2024. Efficient model compression techniques with fishleg. *arXiv preprint arXiv:2412.02328*.

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.

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

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. Association for Computational Linguistics.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, and 5 others. 2024. Olmoe: Open mixture-of-experts language models. *Preprint*, arXiv:2409.02060.

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

Ivo Gollini Navarrete, Nicolas Mauricio Cuadrado, Jose Renato Restom, Martin Takáč, and Samuel Horváth. 2025. Fishing for cheap and efficient pruners at initialization. *arXiv preprint arXiv:2502.11450*.

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. Association for Computational Linguistics.

Karl Pearson. 1896. Vii. mathematical contributions to the theory of evolution.—iii. regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, (187):253–318.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

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.

Patrick Schober, Christa Boer, and Lothar A Schwarte. 2018. Correlation coefficients: appropriate use and interpretation. *Anesthesia & analgesia*, 126(5):1763–1768.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, and Mi Zhang. 2024. Efficient large language models: A survey. *Transactions on Machine Learning Research*. Survey Certification.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pages 353–355.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2023. Sheared llama: Accelerating language model pre-training via structured pruning. *CoRR*, abs/2310.06694.

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. *arXiv preprint arXiv:2410.12013*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Yuanlin Duan, Wenqi Jia, Miao Yin, Yu Cheng, and Bo Yuan. 2024b. MoE-i^2: Compressing mixture of experts models through inter-expert pruning and intra-expert low-rank decomposition. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10456–10466. Association for Computational Linguistics.

Haoqi Yang, Luohe Shi, Qiwei Li, Zuchao Li, Ping Wang, Bo Du, Mengjia Shen, and Hai Zhao. 2025b. Faster moe llm inference for extremely large models. *arXiv preprint arXiv:2505.03531*.

Xican Yang, Yuanhe Tian, and Yan Song. 2025c. Moe pathfinder: Trajectory-driven expert pruning. *arXiv preprint arXiv:2512.18425*.

Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Mykola Pechenizkiy, Yi Liang, Zhangyang Wang, and Shiwei Liu. 2023. Outlier weighed layerwise sparsity (owl): A missing secret sauce for pruning llms to high sparsity. *arXiv preprint arXiv:2310.05175*.

Zhihang Yuan, Yuzhang Shang, Yang Zhou, Zhen Dong, Zhe Zhou, Chenhao Xue, Bingzhe Wu, Zhikai Li, Qingyi Gu, Yong Jae Lee, and 1 others. 2024. Llm inference unveiled: Survey and roofline model insights. *arXiv preprint arXiv:2402.16363*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

A Additional Experimental Results

A.1 Perplexity Evaluation

We evaluate zero-shot language modeling performance on WikiText2, PTB, and LAMBADA using the standard sliding-window evaluation protocol. Table 9 reports the perplexity of pruned models in comparison with the baselines. As sparsity increases, perplexity generally degrades across all methods. Nevertheless, COMPEL consistently exhibits greater robustness to pruning than competing approaches, achieving the lowest perplexity among the baselines and incurring only a marginal degradation relative to the unpruned model.

A.2 Hardware Efficiency

We evaluate hardware efficiency by measuring peak GPU memory consumption during inference. Table 7 reports the memory usage in gigabytes, together with the relative reduction compared to the unpruned model. Notably, COMPEL consistently reduces GPU memory usage to around 80% of the unpruned baseline across all backbones, matching or slightly improving upon competing methods such as NAEE and DiEP.

Model	Method	Mem(GB)	GPU ↓
Qwen	None	9.48	1.00×
	NAEE	8.82	0.93×
	STUN	8.54	0.90×
	MoE-I2	8.35	0.88×
	DiEP	8.16	0.86×
	COMPEL	7.78	0.82×
DeepSeek	None	9.95	1.00×
	NAEE	9.25	0.93×
	STUN	9.05	0.91×
	MoE-I2	8.86	0.89×
	DiEP	8.66	0.87×
	COMPEL	8.45	0.85×
OLMoE	None	6.12	1.00×
	NAEE	5.63	0.92×
	STUN	5.39	0.88×
	MoE-I2	5.26	0.86×
	DiEP	5.14	0.84×
	COMPEL	4.83	0.79×

Table 7: Inference-time GPU memory consumption for different pruning methods. Reported values correspond to peak GPU memory usage (GB), with GPU memory normalized by the unpruned model.

A.3 Runtime Analysis

Table 8 presents the detailed time breakdown of the proposed COMPEL framework across three different MoE models. All experiments were conducted on a server equipped with four NVIDIA RTX 3080 GPUs. For the Qwen1.5-MoE-A2.7B model, the entire pipeline is completed in approximately 0.66 hours, demonstrating the practical efficiency of our method even on consumer-grade hardware. As shown in the table, the Optimization step accounts for the majority of the time cost. This is primarily due to the computation of importance scores and gradient-based adjustments required to identify redundant experts. The Pruning step involves simple weight masking operations with negligible latency, while the Compensation step efficiently recovers model performance, occupying approximately 20% of the total budget. The results indicate that the time cost of COMPEL scales reasonably with the model size, verifying its applicability to larger MoE architectures.

Model	Step	Time (h)	Proportion (%)
Qwen	Optimization	0.52	78.8
	Pruning	0.01	1.5
	Compensation	0.13	19.7
DeepSeek	Optimization	1.92	78.4
	Pruning	0.04	1.6
	Compensation	0.49	20.0
OLMoE	Optimization	0.98	78.4
	Pruning	0.02	1.6
	Compensation	0.25	20.0

Table 8: Time breakdown of COMPEL across different MoE backbones. The proportion indicates the relative time cost of each step within the full COMPEL pipeline.

A.4 Experiments on Calibration Dataset Sizes

We analyze the impact of calibration set size by varying the number of calibration sequences sampled from the C4 dataset. Table 10 reports average performance on standard benchmarks as the number of sequences ranges from 1 to 256. Performance improves with increasing calibration size and reaches its maximum at 128 sequences, whereas a further increase to 256 sequences results in a slight degradation. Based on this observation, we fix the calibration set size to 128 sequences for all experiments.

Model	Sparsity	Method	WikiText2 ↓	PTB ↓	LAMBADA ↓
Qwen1.5-MoE-A2.7B	0%	None	7.06	13.45	31.56
	25%	NAEE	10.11	16.81	44.83
		STUN	9.46	16.14	41.62
		MoE-I2	9.10	15.23	38.90
		DiEP	<u>8.97</u>	<u>14.39</u>	<u>35.11</u>
		COMPEL	**7.94**	**14.38**	**34.91**
	50%	NAEE	13.84	20.85	52.37
		STUN	10.23	17.21	47.92
		MoE-I2	<u>9.14</u>	15.87	41.26
		DiEP	10.51	<u>14.66</u>	36.83
		COMPEL	**8.03**	**14.65**	**36.65**
DeepSeek-V2-Lite	0%	None	10.22	48.61	79.20
	25%	NAEE	11.15	60.76	96.53
		STUN	<u>10.89</u>	57.36	<u>88.72</u>
		MoE-I2	11.49	65.81	104.30
		DiEP	11.34	<u>53.47</u>	92.11
		COMPEL	**10.65**	**53.45**	**81.83**
	50%	NAEE	11.51	63.19	112.41
		STUN	<u>11.03</u>	58.33	101.27
		MoE-I2	12.62	66.50	110.84
		DiEP	11.46	<u>55.90</u>	<u>87.93</u>
		COMPEL	**10.87**	**55.88**	**87.61**
OLMoE-1B-7B-0924	0%	None	13.52	26.80	27.90
	25%	NAEE	13.96	27.87	30.92
		STUN	14.64	27.60	30.53
		MoE-I2	<u>13.76</u>	27.47	<u>29.82</u>
		DiEP	13.84	<u>27.20</u>	31.68
		COMPEL	**13.66**	**27.18**	**28.45**
	50%	NAEE	14.09	28.41	33.81
		STUN	14.84	30.55	35.93
		MoE-I2	<u>13.97</u>	28.14	31.42
		DiEP	14.01	<u>27.87</u>	<u>29.72</u>
		COMPEL	**13.76**	**27.45**	**29.20**

Table 9: Zero-shot evaluation results for structured expert pruning across MoE models at different sparsity levels (0%, 25%, and 50%). Performance is measured on WikiText2, PTB, and LAMBADA. Bold and underlined values indicate the best and second-best results within each sparsity setting, respectively.

B Performance Metrics

We provide detailed definitions of the performance metrics used to evaluate inference efficiency, following the methodology described in Chitty-Venkata et al. (2025). We focus on three key metrics: Throughput, Time to First Token (TTFT), and Inter-Token Latency (ITL).

Throughput. Throughput measures the overall processing efficiency of the hardware by calculating the total number of tokens processed per second. It accounts for both the input prompt processing (prefill) and the output token generation (decode). It is defined as:

$$\text{Throughput} = \frac{\text{Batch Size} \times (\text{Input Tokens} + \text{Output Tokens})}{\text{End-to-End Inference Latency}} \tag{12}$$

where End-to-End Inference Latency is the total time from prompt submission to the generation of the final output token.

Time to First Token (TTFT). TTFT measures the latency from receiving an input prompt to the generation of the first output token, and is commonly used as an indicator of system responsiveness. In our experiments, TTFT is measured by constraining the maximum output length to one token and recording the corresponding generation time.

Inter-Token Latency (ITL). ITL characterizes the average latency between consecutive output tokens and reflects the per-token decoding speed of the model. It is computed by subtracting the initial

Number of Sequence	Avg.
1	62.86
2	63.61
4	63.64
16	63.93
32	63.95
64	64.01
128	64.53
256	63.28

Table 10: Effect of the number of evaluation sequences. Avg. denotes the average accuracy across tasks in the LM Evaluation Harness

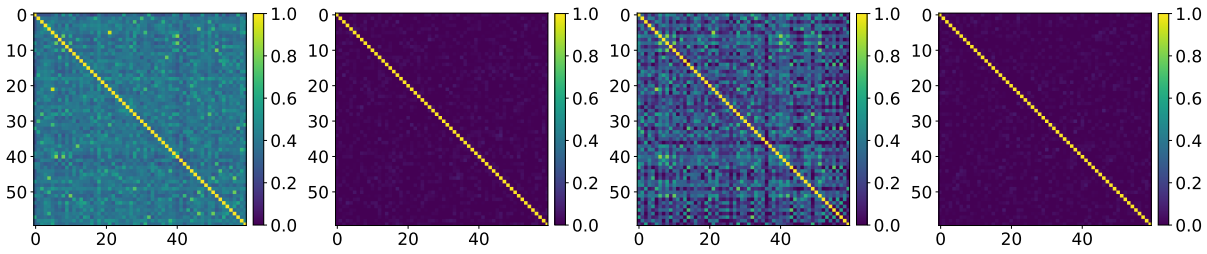
prefill latency (TTFT) from the end-to-end latency:

$$\text{ITL} = \frac{\text{End-to-End Latency} - \text{TTFT}}{\text{Batch Size} \times \text{Output Tokens} - 1} \quad (13)$$

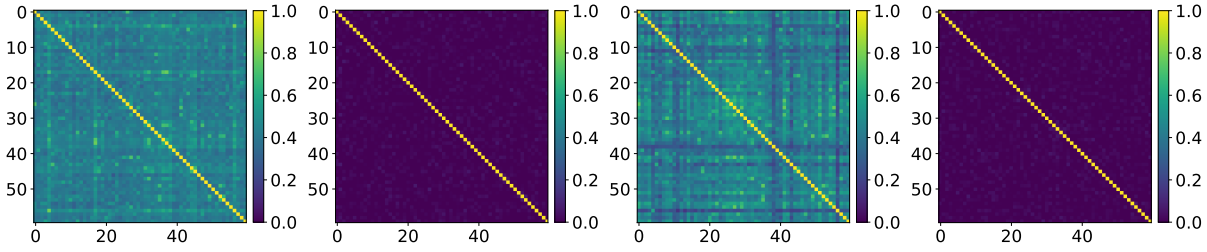
This formulation isolates the decoding phase and enables a focused evaluation of the per-token generation cost.

C Analysis of Expert Redundancy

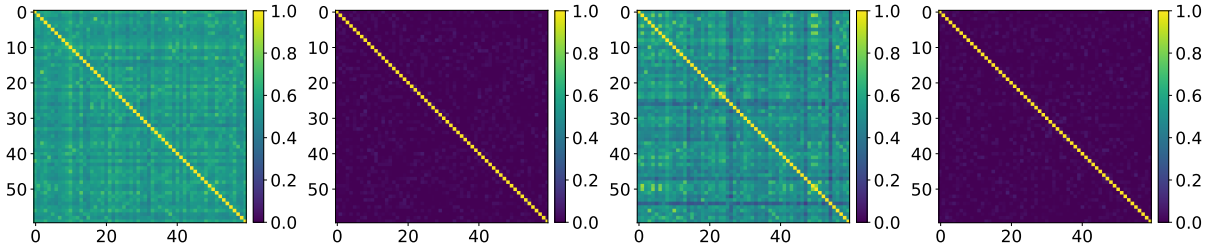
To investigate the representational diversity and redundancy among experts in Qwen1.5-MoE-A2.7B, we analyze the pairwise correlations of expert parameters by computing the cosine similarity between the weight matrices of experts within the same layer, following the neuron-level averaging methodology described in Chitty-Venkata et al. (2025). We represent the projection matrices (W_{up}, W_{gate}, W_{down}) and the router's gate weights of each expert as low-dimensional centroid vectors by averaging along the feature dimension, and subsequently calculate their pairwise similarity to generate the heatmaps shown in Figure 4. Our visualization reveals that while experts in shallower layers exhibit lower similarity, indicating diverse feature extraction capabilities, those in deeper layers demonstrate increasing redundancy with distinct clusters, suggesting a convergence towards similar representations. This trend is consistent across the router, gate, and projection weights, implying that the gating mechanism tends to group experts with similar transformation logic, thereby directing tokens to functionally redundant expert clusters in the deeper stages of the network.



(a) Layer 3



(b) Layer 12



(c) Layer 22

Figure 4: Pairwise cosine similarity heatmaps of expert parameters in Qwen1.5-MoE-A2.7B.