MoE-I²: Compressing Mixture of Experts Models through Inter-Expert Pruning and Intra-Expert Low-Rank Decomposition

Cheng Yang[*1], Yang Sui[*1,2], Jinqi Xiao[1], Lingyi Huang[1], Yu Gong[1], Yuanlin Duan[1],
Wenqi Jia[3], Miao Yin [3], Yu Cheng[4], Bo Yuan[1]
[1]Rutgers University, [2]Rice University,
[3]The University of Texas at Arlington, [4]The Chinese University of Hong Kong
cheng.yang@rutgers.edu, yang.sui@rice.edu, bo.yuan@soe.rutgers.edu

Abstract

The emergence of Mixture of Experts (MoE) LLMs has significantly advanced the development of language models. Compared to traditional LLMs, MoE LLMs outperform traditional LLMs by achieving higher performance with considerably fewer activated parameters. Despite this efficiency, their enormous parameter size still leads to high deployment costs. In this paper, we introduce a two-stage compression method tailored for MoE to reduce the model size and decrease the computational cost. First, in the inter-expert pruning stage, we analyze the importance of each layer and propose the Layer-wise Genetic Search and Block-wise KT-Reception Field with the non-uniform pruning ratio to prune the individual expert. Second, in the intra-expert decomposition stage, we apply the low-rank decomposition to further compress the parameters within the remaining experts. Extensive experiments on Qwen1.5-MoE-A2.7B, DeepSeek-V2-Lite, and Mixtral-8×7B demonstrate that our proposed methods can both reduce the model size and enhance inference efficiency while maintaining performance in various zero-shot tasks. The code will be available at https://github.com/xiaochengsky/MoEI-2.git

1 Introduction

Large Language Models (LLMs) have recently demonstrated remarkable language understanding and generation proficiency, excelling in complex tasks (Achiam et al., 2023; Touvron et al., 2023a; Wu et al., 2020). However, deploying these models presents substantial challenges due to their significant storage and computational demands. To overcome these issues, the Mixture-of-Experts (MoE) LLM has been proposed (Jiang et al., 2024), which activates only a subset of its parameters during training and inference. For instance, with a smaller model size, the Mixtral-8×7B model with a total of 47B parameters surpasses the performance of dense Transformer models like LLaMA-2-70B (Touvron et al., 2023b). Additionally, Qwen1.5-MoE-A2.7B (Bai et al., 2023) demonstrates highly competitive performance compared to other 7B models, and the recently introduced DeepSeekv2 MoE (DeepSeek-AI, 2024) achieves performance levels comparable to GPT-4, demonstrating the powerful capabilities of MoE models.

MoE models have garnered significant attention recently due to their ability to dynamically select subsets of parameters for each input, enabling efficient handling of diverse tasks. Despite their potential, a notable challenge with MoE models is that they are still burdened by substantial parameter size and computation cost. For example, Mixtral-8×7B (Jiang et al., 2024) not only has 47B parameters but also activates 13B parameters during inference. While this architecture allows for scalability and flexibility, it also introduces complexities and huge memory in deployment and inference, particularly when considering resource constraints and efficiency. Consequently, decreasing and maintaining these large-scale models remains a critical area of research.

Model compression techniques, such as pruning, knowledge distillation, and quantization, have been utilized to slim the model size. (Lu et al., 2024) proposed to reduce the parameter count of MoE models by expert pruning, but it does not reduce the parameters during inference efficiently. (Li et al., 2024) merges several experts into one and applies the low-rank decomposition to further reduce the model size. Although this approach achieves a good compression ratio and performance, it requires calibration and fine-tuning for each downstream task individually, which is not suitable for large-scale LLMs, and time costs are very high.

---

*Equal Contribution.

Several works (Zhou et al., 2021; Sun et al., 2023; Frantar and Alistarh, 2023) focus on unstructured sparsity to decrease the parameters of models while maintaining high performance. However, unstructured pruning struggles to achieve practical acceleration, decrease inference, and save storage without a specific design for hardware and libraries.

To solve these problems, we start by analyzing parameter redundancy in the MoE model from multiple levels. First, since identifying redundant experts using brute-force search (Lu et al., 2024) is infeasible in practice, it is necessary to design efficient methods to reduce the time complexity. Second, we aim to compress as many experts as possible while ensuring that the model maintains its zero-shot performance, rather than being limited to handling a single down-stream task (Li et al., 2024). Finally, our method can adapt to any MoE model, particularly those with a large number of experts and diverse structures, and automatically identifies a suitable compression strategy for each type of MoE model without the need for manual settings.

In this paper, we propose a novel end-to-end framework for MoE models, MoE-I^2, for the task-agnostic compression of the MoE models. To our knowledge, MoE-I^2 is the first end-to-end framework designed task-agnostic for structured compression of MoE LLMs. Our contributions are summarized as follows:

- We introduce a two-stage MoE compression framework for expert slimming that considers both inter-expert and intra-expert relationships.

- In the inter-expert pruning stage, we analyze the importance of each MoE layer and propose a non-uniform pruning ratio for each layer. Then, we find that previous MoE pruning methods lead to high time complexity and local optima. To address these issues, we introduce a layer-wise genetic search to reduce time complexity and a block-wise combination strategy to approximate a global optimum better.

- In the intra-expert decomposition stage, we measure the importance of each expert and assign non-uniform ranks accordingly. Subsequently, we apply a low-rank decomposition to further compress the parameters within each expert in a fine-grained manner.

- We conduct extensive experiments with MoE models, including Qwen1.5-MoE-A2.7B (14.3B), DeepSeek-V2-Lite (16B), and Mixtral-8×7B (47B), across various nine datasets to assess both the generation quality and the zero-shot classification performance, demonstrating the effectiveness of our proposed MoE-I^2 framework.

2 Related Works

2.1 Mixture-of-Experts LLMs

MoE-LLMs have gained significant attention in recent years due to their ability to scale efficiently while maintaining high performance. MoE models divide the network into several experts and dynamically select a subset of these experts for each input, which reduces computational overhead and enhances scalability. (Shazeer et al., 2017) introduced the MoE model in their work on the Sparsely-Gated Mixture-of-Experts Layer, and (Lepikhin et al., 2020) further advanced the MoE architecture by demonstrating its scalability to trillions of parameters while retaining manageable computation costs by distributing the experts across multiple devices. With the recent advancements in decoder-only architecture(Touvron et al., 2023a), MoE models built on this structure have become increasingly popular (Jiang et al., 2024). In this paper, we focus on how to build an end-to-end framework to solve post-training expert pruning and decomposition for MoE LLMs to decrease computation and storage.

2.2 Compression on MoE LLMs

Recent advancements in large language models have underscored the need to reduce parameter sizes and latency (Ma et al., 2023). Compression techniques for language models include network pruning (Xu et al., 2021), knowledge distillation (Sun et al., 2019, 2020), quantization (Yao et al., 2022), decomposition (Hsu et al., 2022; Yuan et al., 2023; Wang et al., 2024), and methods like early exit (Xin et al., 2020). Building on these techniques, pruning, and sparsity is crucial for MoE models, which often have up to 95% of parameters dedicated to experts. Pruning MoE models involves removing less important experts or neurons to reduce the number of active parameters during inference. For example, (Kim et al., 2021) retains the most activated experts to enhance machine translation MoE models, while (Koishekenov et al., 2022) introduces gate statistics-

based pruning during decoding. Although effective, these methods are mostly confined to linguistic models in machine translation. (Chen et al., 2022) dropping-while-training approach progressively removes non-essential experts for specific tasks, tested on Switch Transformers (Fedus et al., 2022). The merge-compression (Li et al., 2024) method and EPP (Lu et al., 2024) approach, which is similar to ours, consider pruning and skipping in MoE models but face challenges in reducing computational costs. Given a pruned or sparse model, finetuning aims to restore performance on original tasks. Recent studies on LLMs (Sun et al., 2023; Ma et al., 2023) focus on pruning linear layers, but these methods often fail to reduce computing costs without specialized hardware or libraries. Efficient post-finetuning expert pruning and sparsity methods for task-agnostic MoE LLMs remain underexplored. This gap highlights the need for advanced techniques to effectively balance pruning and sparsity while maintaining or enhancing performance across various tasks.

3 Method

In this section, we introduce the details of our proposed framework, MoE-I^2, which consists of three stages: Inter-Expert Pruning stage (Sec. 3.1), Intra-Expert Decomposition stage (Sec. 3.2), and finetuning stage (Sec. 3.3). The overall pipeline is shown in Figure 1.

3.1 Inter-Expert Pruning

In this stage, our goal is to prune individual unimportant experts to reduce the parameter size and computational cost. It raises two crucial questions: (1) *Given an overall pruning ratio, how many experts should be pruned in each layer?* (2) *How to determine which experts to prune?*

3.1.1 Layer Importance Analysis

To answer the first question, we start by analyzing the importance of each layer. The layer importance of i-th layer, denoted by I_i, is defined as the average loss degradation by removing individual experts within this layer. Specifically, to calculate I_i in the i-th layer, we first calculate the expert importance. We consecutively pruning j-th expert in the i-th layer, denoted by $e_{i,j}$, where $j = 1, 2, \cdots, M_i$. The M_i represents the total number of experts in the i-th layer. Next, each pruned model predicts the next token with the calibration samples. The expert importance of $e_{i,j}$ is calculated as:

$$I_{i,j} = \sum_B \mathcal{L}(\mathcal{X}, \{\mathbf{E}_i\} \setminus \{e_{i,j}\}) \qquad (1)$$

where $\{\mathbf{E}_i\} = \{e_{i,1}, e_{i,2}, \cdots, e_{i,M_i}\}$ denotes the set of all experts in i-th layer. $\mathcal{X}$ represents the calibration dataset, and B denotes the batch size. $\mathcal{L}$ denotes the output of the MoE model under the condition that the j-th expert in the i-th layer is removed. Once we have determined the importance score of the j-th expert in the i-th layer, the overall importance score of i-th layer is defined as $I_i = \sum_{j=1}^{E_i} I_{i,j}$. Given the overall pruning rate, we normalize the layer importance to obtain the pruning rate for each layer.

Following this paradigm, we demonstrate the layer importance for Mixtral-8×7B (Jiang et al., 2024), Qwen1.5-MoE-A2.7B (Bai et al., 2023), and DeepSeek-V2-Lite (DeepSeek-AI, 2024) as shown in Figure 2. Note that the previous work (Lu et al., 2024) overlooks the varying importance of layers and simply applies a uniform pruning ratio to each layer, leading to a suboptimal solution. In contrast, our analysis shows that some models perform in ways that largely diverge from this strategy. For example, the analysis of DeepSeek-V2-Lite (Figure 2) reveals that layer importance rapidly increases with depth, indicating that deeper layers are more sensitive than shallower ones.

3.1.2 Inter-Expert Pruning Strategy

To answer the second question, it is required to identify a combination of N experts that have the least impact on prediction loss. Previous work (Lu et al., 2024) utilizes brute-force search to find the least impactful combination of N experts within each layer. However, this method presents two significant drawbacks. First, the brute-force search has high time complexity, making it *extremely time-consuming*, especially when pruning the MoE with a large number of experts. For example, Qwen1.5-MoE-A2.7B and DeepSeek-V2-Lite have 60 and 64 experts per layer, respectively. If 25% of experts need to be pruned, (Lu et al., 2024) needs to traverse C_{15}^{60} and C_{16}^{64} times for each layer respectively, which is unacceptable in terms of time consumption. Second, it restricts the search space within the current layer, only achieving a *local optimum* and potentially missing a more globally optimal solution.

To mitigate these challenges, we leverage *Genetic Search* (Grefenstette, 1993; Alam et al., 2020) with *KT-Receptive Field* methods to enhance search efficiency and concurrently identify the least im-
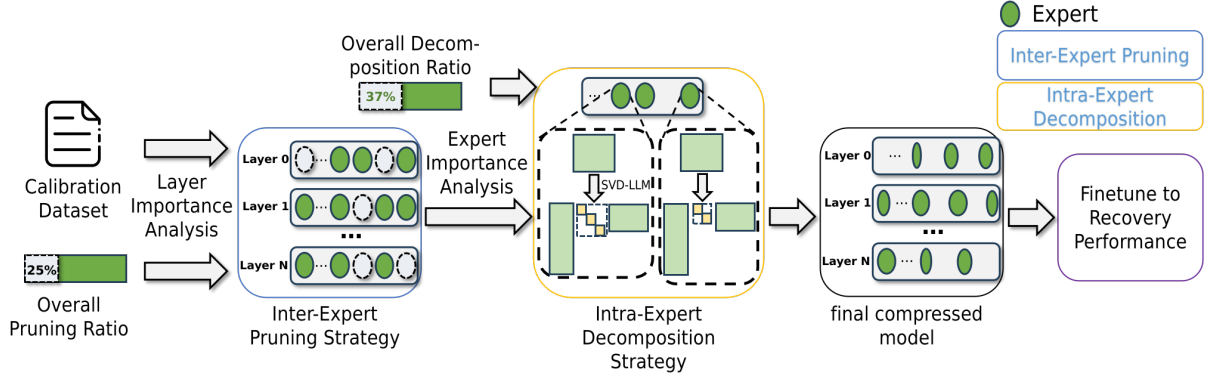
Figure 1: The three-stage pipeline of MoE-I². The first stage (left) represents *Inter-Expert Pruning*, where MoE-I² conducts the *Layer Importance Analysis* on the target MoE model. By using a predefined overall pruning rate, it determines varying pruning ratios of different layers. Subsequently, the unimportant experts in MoE are determined by Layer-wise Genetic Search and Block-wise KT-Reception Field. The MoE is pruned accordingly. The second stage (middle) represents *Intra-Expert Analysis*. Similarly, MoE-I² automatically performs *Expert Importance Analysis* on the pruned model and using a predefined overall decomposition rate, applies varying ranks and low-rank decomposition to different experts, resulting in a final compressed model. The third stage (right) shows that we fine-tuned the compressed MoE model to recover performance.

pactful combinations of experts on a more global scale.

Layer-wise Genetic Search. To avoid extreme time consumption caused by brute-force search (Lu et al., 2024), we leverage the genetic search to select the M candidate combinations in each layer.

For the i-th layer, we define all possible pruning combinations as C_{P_i}. Here, P_i represents the number of experts to be be pruned in the i-th layer. Given that there are M_i experts in the i-th layer, C_{P_i} denotes the number of combinations for selecting P_i experts to prune from the total of M_i experts.

In the initial stage of Genetic Search, we first initialize a population $\{C_{P_i,1}, C_{P_i,2}, \ldots, C_{P_i,N}\}$, where the population size $N = 100$. We then calculate the loss for each combination in the population:

$$\mathcal{L}_i^n = \sum_B \|\mathcal{F}_i(\mathcal{X}) - \mathcal{F}_i(\mathcal{X}, \{\mathbf{E}_i\} \setminus C_{P_i,n}))\|_F \quad (2)$$

where $\mathcal{F}_i$ represents the output of layer i of the MoE model, and $\|\cdot\|_F$ donates Frobenius norm.

We select the combinations with the smallest loss from $C_{P_i,n}$ as parents. Using union and random sampling, we generate offspring combinations. Each individual in the offspring population undergoes some mutations, where a few experts to be pruned are randomly replaced. This process is repeated iteratively in 50 steps and we can obtain the optimal a few combinations of expert pruning

as candidate combinations in the i-th layer.

Block-wise KT-Reception Field. After obtaining the n candidate combinations, we only keep K best combinations with the smallest loss in each layer as the candidate combinations to be used for the block-level optimization. We aim to select one of the K combinations from each layer such that they minimize the output loss. During this selection process, instead of only considering the importance of experts in just the current layer (Lu et al., 2024), we extend the scope of candidate selection from one layer to T layers, achieving a block-wise combination. Specifically, we partition all layers into $\lceil \frac{L}{T} \rceil$ blocks. Within each block, we select the combination in a brute-force scheme. Given K candidates in each layer, and considering there are T layers in one block, we traverse all possible combinations by selecting one combination from each of T layers, yielding a total of K^T options. Subsequently, we calculate the output loss and select the optimal combinations for pruning. The pipeline is shown in Figure 3.

Expert Pruning. Given the to-be-pruned experts, we conduct the expert pruning operation by removing the entire expert in a structured manner.

3.2 Intra-Expert Decomposition

In this stage, we propose to further compress the remaining experts in a fine-grained way by perform-
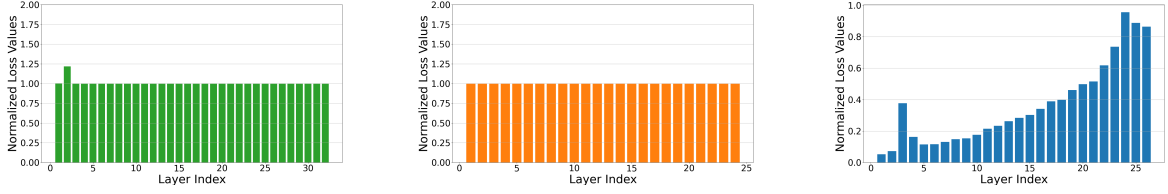
Figure 2: Importance analysis of Mixtral-8×7B (left), Qwen1.5-MoE-A2.7B (middle), and DeepSeek-V2-Lite (right) models. A larger loss indicates greater importance. For Mixtral-8×7B and Qwen1.5-MoE-A2.7B, the importance of the different layers is relatively consistent, but for DeepSeek-V2-Lite, the importance increases as one approaches the output layer.
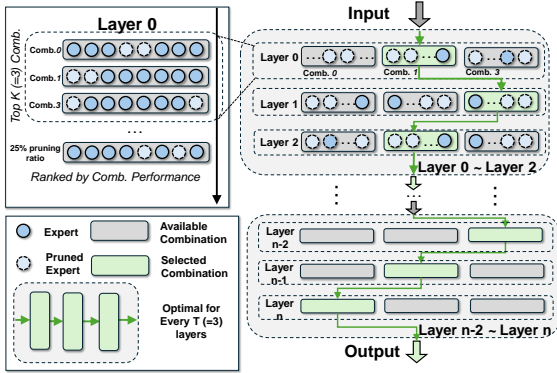

Figure 3: The process of *KT-Receptive Filed* in Mixtral-8×7B for satisfied 25% pruning ratio. In this case, the number of candidate combinations per layer is $K = 3$, and the number of layers per block is $T = 3$. For each layer, we select K optimal candidates using the Layer-wise Genetic Search (top-left). Within a consecutive sequence of T layers, we employ the Block-wise KT-Reception Field to identify the best-performing combination within that block (T layers).

ing the low-rank decomposition on the parameters within each intra-expert.

3.2.1 Expert Importance Analysis

As mentioned in (Chi et al., 2022), each expert has varying levels of importance. To achieve better compression performance, instead of applying a uniform compression ratio, we aim to retain more parameters in the important experts and fewer in the less important ones. That leads us to assign higher ranks to the more important experts and lower ranks to the less important ones. Therefore, to calculate the varying ranks, we analyze the relative importance of each expert. Based upon the previous analysis in Sec. 3.1.1, we adopt the same importance metric, $I_{i,j}$ in Eq. 1, as the expert importance.

To determine the varying ranks of each expert,

we begin by calculating basic uniform rank values. Given the overall compression ratio in the second stage, and considering that the structure of all experts is entirely consistent, we directly calculate the target average rank for each expert after decomposition, which is denoted as $\mathcal{R}_a$. By considering the important score of each expert, we calculate the rank values for experts $e_{i,j}$ as:

$$\mathcal{R}_{ij} = \left\lfloor \frac{(I_{ij} + \epsilon)^\alpha}{\sum_{j=1}^{M'_i}(I_{ij} + \epsilon)^\alpha} \cdot \mathcal{R}_a \cdot M'_i \right\rfloor \quad (3)$$

Here, M'_i represents the number of experts remaining in layer i of the model obtained after *Intra-Expert Pruning*, and α denotes the smooth factor used to avoid overly linearizing the distribution of rank values, set as 0.15. ϵ is set to 1×10^{-6} to avoid the numerical issue.

3.2.2 Intra-Expert Decomposition Strategy

Singular Value Decomposition (SVD) is a general technique to reduce parameter size by decomposing a large dense matrix into two smaller low-rank matrices. Compared to the Vanilla SVD, which only focuses on the initial weight matrix, (Wang et al., 2024) generates activation by truncation-aware data whitening and provides hierarchical closed-form updates for model compression. Inspired by SVD-LLM (Wang et al., 2024) working on dense models, we extend SVD-LLM to MoE models by integrating the non-uniform ranks $R_{i,j}$ in Sec. 3.2.1.

3.3 Efficient Fine-tuning

To mitigate performance degradation caused by the two-stage compression, we fine-tune the MoE by updating the weights. Instead of adjusting all weights, we integrate LoRA (Hu et al., 2021), a low-rank approximation technique, into the post-training of the pruned model. The overall algorithm is illustrated in Alg. 1.

Algorithm 1 The Algorithm of MoE-I^2

Inputs: Initial Model $\mathcal{M}$, Target Pruning Ratio $\mathcal{P}_S$, Expert Decomposition Rate $\mathcal{D}$, Calibration Sample $\mathcal{S}_c$, Finetune Sample $\mathcal{S}_f$.
Outputs: Compressed MoE-I^2, $\mathcal{M}_f$
1: **for** each layer l_i in $\mathcal{M}$ **do**
2: $\mathcal{I}_i \leftarrow$ Layer Importance Analysis with $\mathcal{S}_c$ via Sec. 3.1.1;
3: **end for**
4: $\mathcal{M}_p \leftarrow$ Inter-Expert Pruning($\mathcal{M}$, $\mathcal{S}_c$, $\mathcal{P}_S$, $\mathcal{I}$) via Sec. 3.1.2;
5: **for** each layer l_i in $\mathcal{M}_p$ **do**
6: $\mathcal{R}_{i,j} \leftarrow$ Expert Importance Analysis via Sec. 3.2.1;
7: **end for**
8: $\mathcal{M}_c \leftarrow$ Intra-Expert Decomposition($\mathcal{M}_p$, $\mathcal{S}_c$, $\mathcal{D}$, $\mathcal{R}$) via Sec. 3.2.2;
9: $\mathcal{M}_f \leftarrow$ Low-Rank Finetune($\mathcal{M}_c$, $\mathcal{S}_f$) via Sec. 3.3;

---

4 Experiments

4.1 Experimental Settings

Model Settings. To demonstrate the effectiveness of our method, we conducted experiments on three MoE models: Qwen1.5-MoE-A2.7B (14.3B), DeepSeek-V2-Lite (16B), and Mixtral-8×7B (47B). Mixtral-8×7B has a larger number of parameters and relatively fewer experts (8 experts per layer in total 32 layers). On the other hand, Qwen1.5-MoE-A2.7B and DeepSeek-V2-Lite have fewer parameters but a greater number of experts (60 and 64 experts per layer in a total of 24 and 26 layers, respectively).

Evaluation and Datasets. To evaluate the performance in a task-agnostic setting, we mainly adopt LLama-Pruner (Ma et al., 2023) evaluation methodology, conducting zero-shot task classification across common sense reasoning datasets such as BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC-easy (Clark et al., 2018), ARC-challenge (Clark et al., 2018), and OpenbookQA (Mihaylov et al., 2018). Meanwhile, our model evaluates results in multiple-choice tasks or generates answers in open-ended generation tasks (Gao et al., 2021). Furthermore, we supplement our evaluation with a zero-shot perplexity (PPL) analysis on WikiText2 (Merity et al., 2016) and PTB (Marcus et al., 1993).

Implementation Details. During the expert pruning phase, we use the same data as the (Lu et al., 2024), which is 2048 randomly sampled data from the C4 (Raffel et al., 2020) dataset as calibration data. In the expert decomposition phase, we also use 2048 randomly sampled data from Alpaca (Taori et al., 2023) as calibration data to conduct the importance analysis. For the finetuning phase, similar to LLM-Pruner (Ma et al., 2023), we use Alpaca as the finetuning training set, totaling approximately 50k samples. The batch size is set as 64 and learning rates are from 3e-4 to 5e-4. The experiments are conducted on 4 A100-80G GPUs.

4.2 Main Results

MoE-I^2 Results. Table 1 presents the zero-shot performance of the models after applying the MoE-I^2 framework. It is evident that pruning 25% of the expert parameters results in only a slight performance loss. However, after finetuning the compressed mode with only 2 epochs, the performance can even surpass that of the original model, especially with an improvement of over 2% on the DeepSeek-V2-Lite model. This observation suggests that pruning 25% of the experts in the first step is lossless. In the second step, we choose to further compress the pruned model with an approximate 40% compression ratio via low-rank decomposition. Finally, we perform the finetuning stage. As a result, we can see that while ensuring a reduction of more than 50% in expert parameters, the model's performance is largely preserved.

Zero-shot Performance Comparisons with Existing Methods. Table 2 shows the zero-shot performance of the pruned model by comparing Wanda (Sun et al., 2023), EEP (Lu et al., 2024), and our *Inter-Expert Pruning* method under the same sparsity rate. Our method demonstrates significant advantages over Wanda and EEP.

PPL Comparisons with Existing Methods. Table 3 shows the zero-shot perplexity(PPL) of the pruned model by comparing EEP, and our *Inter-Expert Pruning* method under the same sparsity rate. Our method demonstrates significant advantages over EEP.

Inference Speedup with Existing Methods. Table 4 shows the speedup of three models by comparing Wanda (Sun et al., 2023), EEP (Lu et al., 2024), and MoE-I^2 method.

4.3 Ablation Studies

Comparison of MoE-I^2 and its Components. Table 5 demonstrates the necessity of the components within the MoE-I^2 framework. It shows that MoE-I^2 has a significant advantage when compared to applying only *Inter-Expert Pruning* or *Intra-Expert Decomposition* individually.

Table 1: Zero-shot performance of three models under our MoE-I² Framework. The average is calculated among seven classification datasets. "P" denotes the Inter-Expert Pruning operation, "D" represents the Intra-Expert Decomposition operation, and "F" indicates the "Fine-tuning" operation based on LoRA. "Params" represents the percentage reduction in the number of expert parameters. In the Inter-Expert Pruning stage, we prune 25% of the experts. During the Intra-Expert Decomposition stage, for the Mixtral-8×7B model, we decompose the remaining experts with an average rank of 2048, further reducing the parameters by approximately 37.5%. For the Qwen1.5-MoE-A2.7B and DeepSeek-V2-Lite models, we perform decomposition with an average rank of 512, further reducing the parameters by approximately 38.6%.

Model	Method	Params↓	ARC-c	ARC-e	BoolQ	HellaSwag	OBQA	RTE	WinoGrande	Average
8×7B	baseline	0	57.17	84.01	85.35	64.88	35.00	70.40	75.93	67.53
8×7B	P	25%	51.79	81.36	84.07	61.99	32.80	71.12	75.85	65.57
8×7B	P+F	25%	56.23	82.49	86.42	64.48	36.00	72.92	74.98	**67.65**
8×7B	P+D	51.79%	40.70	71.51	67.83	45.34	26.00	61.37	67.56	54.33
8×7B	MoE-I²	51.79%	52.20	78.22	82.62	61.07	34.00	72.20	71.50	**64.55**
Qwen	baseline	0	41.89	73.11	79.76	57.90	30.40	70.04	68.67	60.25
Qwen	P	25%	38.57	70.37	73.30	55.84	29.80	64.98	67.25	57.16
Qwen	P+F	25%	45.14	75.93	78.01	57.83	32.80	71.12	68.51	**61.33**
Qwen	P+D	53.98%	37.71	65.91	71.41	49.34	29.40	64.26	67.88	55.13
Qwen	MoE-I²	53.98%	41.13	71.68	75.08	53.08	30.80	66.43	66.54	**57.82**
DeepSeek	baseline	0	46.93	78.37	79.82	58.70	34.60	60.65	71.35	61.49
DeepSeek	P	25%	45.31	74.62	67.95	57.38	33.20	59.93	70.01	58.34
DeepSeek	P+F	25%	47.44	78.16	79.79	60.32	35.40	74.56	71.35	**63.86**
DeepSeek	P+D	53.98%	38.48	71.42	70.09	48.15	27.80	60.65	65.98	54.65
DeepSeek	MoE-I²	53.98%	42.58	71.80	76.79	55.16	32.60	70.76	67.64	**59.62**

Table 2: Zero-shot performance comparison with EEP (Lu et al., 2024) and Wanda (Sun et al., 2023)

Model	Method	Params↓	ARC-c	ARC-e	BoolQ	HellaSwag	OBQA	RTE	WinoGrande	Average
8×7B	EEP	25%	51.62	81.94	83.64	61.60	33.00	67.87	75.37	65.01
8×7B	P	25%	51.79	81.36	84.07	61.99	32.80	71.12	75.85	**65.57**
8×7B	Wanda	50%	42.06	74.16	76.64	53.16	27.00	63.90	70.96	58.27
8×7B	EEP	50%	48.89	78.16	81.35	57.66	29.00	61.37	72.85	61.33
8×7B	P	50%	48.38	78.66	81.41	58.35	27.00	64.62	74.19	**61.80**

Table 3: Zero-shot performance of experiment results of comparison with EEP and Wanda. "↓" indicates that lower values are better.

Model	Method	Params↓	WikiText2↓	PTB↓
8×7B	baseline	0%	6.24	107.24
8×7B	EEP	25%	8.16	141.1
8×7B	P	25%	**8.01**	**133.38**
8×7B	EEP	50%	11.02	207.4
8×7B	P	50%	**10.1**	**185.2**

Table 4: Inference speedup performance comparison with EEP (Lu et al., 2024) and Wanda (Sun et al., 2023) at a compression rate of 50% . "↓" indicates that lower values are better.

Model	Method	Mem (GB)↓	Speedup	Average
8×7B	baseline	87.7	1.0×	67.53
8×7B	EEP	45.78	1.20×	61.33
8×7B	Wanda	50.01	0.91×	58.27
8×7B	MoE-I²	43.49	**1.28×**	64.55
Qwen	baseline	26.67	1.0×	60.25
Qwen	MoE-I²	14.14	**1.12×**	57.82
DeepSeek	baseline	29.26	1.0×	61.49
DeepSeek	MoE-I²	15.03	**1.13×**	59.62

Impact of Genetic Search. For Qwen1.5-MoE-A2.7B and DeepSeek-V2-Lite models, which have 60 and 64 experts per layer respectively, we only iterate 50 times for Genetic Search. As shown in Figure 4, the loss has converged in the majority of layers. Using EEP (Lu et al., 2024) for combinatorial search would result in unimaginable time complexity. For instance, if pruning 25% of the experts, EEP would require searching C_{15}^{60} and C_{16}^{64} times for each layer respectively. Table 6 presents the performance of the pruned models obtained through our *Inter-Expert Pruning* compared to *Random* and *TopLoss* methods in terms of zero-shot performance of average(among seven classification datasets) and perplexity tasks. The *TopLoss* denotes that we individually select the P_i least important experts in the

current layer to prune instead of considering the expert combination used in Genetic Search. As observed, Genetic Search has a significant advantage over other methods with similar low time costs on seven classification tasks and PPL.

Impact of KT-Receptive Field. As shown in Figure 5, we also observe that a large KT-Receptive Field is not always the best during calibration. This is partially because we only use a small amount of data for calibration (2048 samples selected from the C4 dataset). Additionally, there is a significant difference between the C4 dataset and the seven datasets used for zero-shot validation. Simply in-

Table 5: Comparison of zero-shot performance of the MoE-I^2 framework and its components. To ensure the same compression ratio and ease of computation as much as possible, when performing the "D+F", we set the average rank value of the experts as $\frac{1}{4}$ of expert dimension, which is 352.

| Model	Method	Params ↓	ARC-c	ARC-e	BoolQ	HellaSwag	OBQA	RTE	WinoGrande	Average
8x7B	P+F	50%	50.43	78.79	82.42	59.12	32.00	70.40	74.03	63.88
8x7B	D+F	51.35%	46.08	75.34	81.41	54.02	27.80	72.20	68.27	60.73
8×7B	MoE-I^2	51.79%	52.20	78.22	82.62	61.07	34.00	72.20	71.50	**64.55**
Qwen	P+F	50%	41.89	69.15	75.20	53.97	30.20	64.98	62.43	56.83
Qwen	D+F	57.81%	36.69	69.01	74.56	47.29	29.40	72.92	68.27	56.88
Qwen	MoE-I^2	53.98%	41.13	71.68	75.08	53.08	30.80	66.43	66.54	**57.82**
DeepSeek	P+F	50%	39.51	70.16	68.17	53.37	26.40	64.98	63.14	55.11
DeepSeek	D+F	57.81%	69.68	70.33	74.19	51.98	29.20	71.12	67.01	57.64
DeepSeek	MoE-I^2	53.98%	42.58	71.80	76.79	55.16	32.60	70.76	67.64	**59.62**
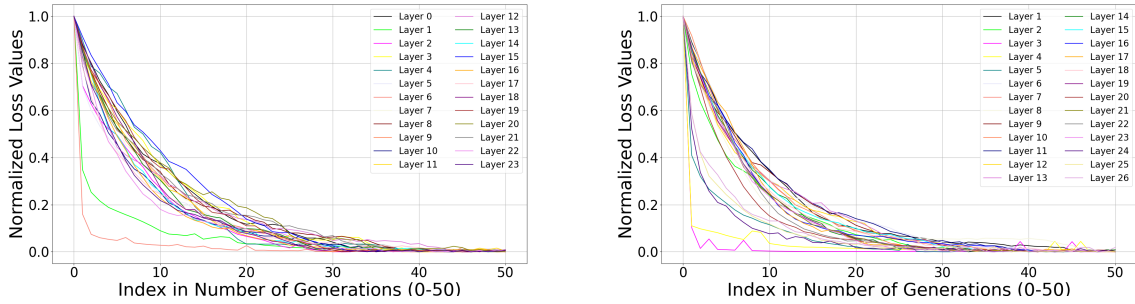
Figure 4: The left and right figures represent the loss convergence for each layer of Qwen1.5-MoE-A2.7B and DeepSeek-V2-Lite during the Genetic Search process, respectively. As shown in the figures, after 50 iterations, nearly all layers have converged.

Table 6: Zero-shot performance of average and perplexity of comparison with "Inter-Expert Pruning", *Random*, and *TopLoss*.

Model	Method	Params↓	Average	WikiText2↓	PTB↓
Qwen	baseline	0%	60.25	7.06	13.51
Qwen	Random	25%	55.34	9.38	16.73
Qwen	TopLoss	25%	56.51	8.06	15.39
Qwen	P	25%	57.16	**8.01**	15.17
DeepSeek	baseline	0%	61.49	10.22	46.43
DeepSeek	Random	25%	43.93	48.05	628.97
DeepSeek	TopLoss	25%	57.00	**11.34**	67.67
DeepSeek	P	25%	**58.34**	11.49	**65.80**

creasing the values of K and T can lead to overfitting on the calibration dataset. Empirically, $K = 3$ and $T = 3$ can achieve the best performance.

Impact of Non-uniform Pruning Ratio. We can observe in Figure 2 that the importance of different layers in the DeepSeek-V2-Lite model varies significantly. Table 7 demonstrates that this distinction in layer importance is effective. Compared to the balanced pruning ratio used by Mixtral-8×7B and Qwen1.5-MoE-A2.7B, the imbalance pruning ratio applied to DeepSeek-V2-Lite results in better model performance.

Impact of Different Ranks. Table 8 shows that selecting an imbalanced rank approach yields better performance for all experts within the same layer. This phenomenon highlights the differences among experts and indicates that different ranks should be
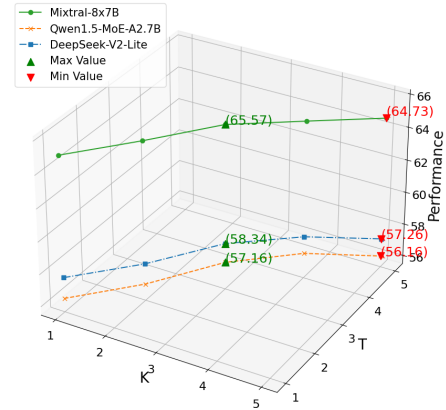

Figure 5: The impact of K and T on the performance of models Mixtral-8×7B, Qwen1.5-MoE-A2.7B, DeepSeek-V2-Lite.

assigned to different experts.

Impact of Experts Pruning, Layers, and Blocks Pruning. Table 9 shows our expert pruning method (Genetic Search) demonstrates significant advantages over concurrent approaches, such as Layer Pruning and Block Pruning (He et al., 2024). Our Genetic Search can retain more performance (1.32% vs. 3.19% performance drop) while maintaining a higher pruning rate (23.95% vs. 15.51% pruning ratio). Note that since (He et al., 2024) presents normalized zero-shot accuracy results, we have also normalized our results for fairness.

Table 7: Zero-shot performance of experiment results produced by Inter-Expert Pruning of comparison with Imbalance (Imba.) and Balance (Ba.) pruning ratio in DeepSeek-V2-Lite.

Model	Method	Params↓	ARC-c	ARC-e	BoolQ	HellaSwag	OBQA	RTE	WinoGrande	Average
DeepSeek	Ba.	25%	44.20	73.91	68.26	57.07	32.00	57.76	69.93	57.59
DeepSeek	Imba.	25%	45.31	74.62	67.95	57.38	33.20	59.93	70.01	**58.34**
DeepSeek	Ba.	50%	31.74	60.19	61.28	45.34	22.40	50.90	60.62	47.50
DeepSeek	Imba.	50%	31.74	61.87	61.74	44.79	23.60	54.87	56.67	**47.90**

Table 8: Zero-shot performance of experiment results produced by Intra-Expert Decomposition of comparison with Imbalance (Imba.) and Balance (Ba.) rank in same layer in three models.

Model	Rank(avg)	Type	ARC-c	ARC-e	BoolQ	HellaSwag	OBQA	RTE	WinoGrande	Average
8x7B	2048	Ba.	43.66	73.45	74.03	54.31	27.40	67.92	69.55	58.62
8x7B	2048	Imba.	43.94	73.95	74.56	55.91	27.80	68.23	69.85	**59.18**
8x7B	1550	Ba.	33.70	63.43	62.57	47.29	22.00	62.45	62.98	50.63
8x7B	1550	Imba.	34.59	63.67	62.59	47.68	22.00	63.05	63.15	**50.96**
Qwen	704	Ba.	40.19	72.94	77.95	54.50	30.40	68.95	69.06	59.14
Qwen	704	Imba.	40.44	73.40	77.74	54.54	31.60	68.95	69.30	**59.43**
Qwen	352	Ba.	35.92	67.55	73.64	44.09	26.40	70.04	67.17	54.97
Qwen	352	Imba.	36.26	67.89	73.15	44.34	27.20	72.20	66.69	**55.39**
DeepSeek	704	Ba.	43.60	76.94	77.77	53.98	30.40	62.82	69.22	59.25
DeepSeek	704	Imba.	44.11	77.19	78.50	54.20	30.40	63.54	69.30	**59.61**
DeepSeek	352	Ba.	33.45	65.11	63.05	39.07	25.20	61.75	64.88	50.35
DeepSeek	352	Imba.	34.04	65.95	63.76	39.53	25.80	60.29	65.19	**50.65**

Table 9: Performance of Pruning on Mixtral-8×7B between our Genetic Search and C-MoE (He et al., 2024). "P" denotes ours Inter-Expert Pruning operation (Genetic Search). "E[n/m]" denotes dropping n out of m of experts per MoE layer on average. "L[n/m]", "B[n/m]" represents dropping n out of m corresponding modules with Layer Drop and Block Drop respectively. These three methods are described in (He et al., 2024).

Model	Method	Mem(GB)	ARC-c	BoolQ	HellaSwag	OBQA	RTE	WinoGrande	Average	Δ↓
8×7B	baseline(Ours/EEP)	87.7	59.81	84.92	83.97	47.00	71.12	76.32	70.52	-
8×7B	P	66.7	56.66	83.46	81.72	46.40	71.12	75.85	69.02	↓ **1.32**
8×7B	baseline (He et al., 2024)	87.7	59.4	84.2	84.00	46.80	70.40	75.60	70.07	-
8×7B	E2/8	66.7	53.20	77.70	80.50	46.20	55.60	76.80	65.00	↓ 5.07
8×7B	L8/32	66.6	47.70	85.30	75.20	40.40	69.70	74.60	65.42	↓ 4.65
8×7B	B5/32	74.1	51.30	85.30	78.70	42.00	69.70	74.30	66.88	↓ 3.19

5 Conclusion

In this paper, we explore the efficiency of current large-scale MoE models and propose a general end-to-end compression framework, MoE-I^2, that addresses the issue of parameter redundancy in MoE models. In our approach, we first conduct the layer importance analysis and *Inter-Expert Pruning* for different MoE models. Subsequently, we perform the expert important analysis based on the pruned model, ensuring appropriate target ranks of each expert when performing the *Intra-Expert Decomposition*. Our MoE-I^2 framework significantly reduces the parameters of MoE models maintaining high performance. In the future, we aim to support a wider variety of MoE models with larger parameters, enhancing their deployability.

Limitations

Our proposed framework, MoE-I^2, can perform end-to-end compression on any MoE model and adaptively find suitable pruning and decomposition strategies for the target MoE model. By compressing the model at multiple fine-grants, we ensure optimal compression while maintaining model performance, making it more suitable for deployment. Despite these advantages, due to computational limitations, we have not yet tested our framework on larger MoE models such as Mixtral-8×22B (141B), and DeepSeek-V2 (236B). We aim to gradually test these larger MoE models in future work.

Ethics Statement

Our research focuses on developing an end-to-end framework for the compression of Mixture-of-Experts (MoE) large language models (LLMs). By enhancing model compression techniques, we aim to significantly reduce the model size and improve inference efficiency, ensuring these improvements do not come at the cost of performance. While our work contributes to the advancement of deploying sophisticated LLMs more effectively, we recognize the ethical considerations inherent in this field. These include the need to address potential biases in the models, ensure the responsible and

fair use of LLMs, and safeguard privacy. We are committed to transparency by making our compression framework publicly available. We urge the community to apply our work ethically, with careful attention to the broader societal impacts of deploying compressed LLMs.

References

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

Tanweer Alam, Shamimul Qamar, Amit Dixit, and Mohamed Benaida. 2020. Genetic algorithm: Reviews, implementations, and applications. *arXiv preprint arXiv:2007.12673*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tianyu Chen, Shaohan Huang, Yuan Xie, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. 2022. Task-specific expert pruning for sparse mixture-of-experts. *arXiv preprint arXiv:2206.00277*.

Zewen Chi, Li Dong, Shaohan Huang, Damai Dai, Shuming Ma, Barun Patra, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, et al. 2022. On the representation collapse of sparse mixture of experts. *Advances in Neural Information Processing Systems*, 35:34600–34613.

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. *arXiv preprint arXiv:1905.10044*.

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*.

DeepSeek-AI. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*.

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, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, et al. 2021. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, page 8.

John J Grefenstette. 1993. Genetic algorithms and machine learning. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 3–4.

Shwai He, Daize Dong, Liang Ding, and Ang Li. 2024. Demystifying the compression of mixture-of-experts through a unified framework. *arXiv preprint arXiv:2406.02500*.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Language model compression with weighted low-rank factorization. *arXiv preprint arXiv:2207.00112*.

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

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, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andres Felipe Cruz Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021. Scalable and efficient moe training for multitask multilingual models. *arXiv preprint arXiv:2109.10465*.

Yeskendir Koishekenov, Alexandre Berard, and Vassilina Nikoulina. 2022. Memory-efficient nllb-200: Language-specific expert pruning of a massively multilingual machine translation model. *arXiv preprint arXiv:2212.09811*.

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

Pingzhi Li, Zhenyu Zhang, Prateek Yadav, Yi-Lin Sung, Yu Cheng, Mohit Bansal, and Tianlong Chen. 2024. Merge, then compress: Demystify efficient SMoe with hints from its routing policy. In *The Twelfth International Conference on Learning Representations*.

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. *arXiv preprint arXiv:2402.14800*.

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

Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

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. *arXiv preprint arXiv:1809.02789*.

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.

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. *arXiv preprint arXiv:1701.06538*.

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*.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Siqi Sun, Zhe Gan, Yu Cheng, Yuwei Fang, Shuohang Wang, and Jingjing Liu. 2020. Contrastive distillation on intermediate representations for language model compression. *arXiv preprint arXiv:2009.14167*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2024. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Yiquan Wu, Kun Kuang, Yating Zhang, Xiaozhong Liu, Changlong Sun, Jun Xiao, Yueting Zhuang, Luo Si, and Fei Wu. 2020. De-biased court's view generation with causality. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 763–780, Online. Association for Computational Linguistics.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating bert inference. *arXiv preprint arXiv:2004.12993*.

Dongkuan Xu, Ian EH Yen, Jinxi Zhao, and Zhibin Xiao. 2021. Rethinking network pruning–under the pre-train and fine-tune paradigm. *arXiv preprint arXiv:2104.08682*.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. 2023. Asvd: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Aojun Zhou, Yukun Ma, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, and Hongsheng Li. 2021. Learning n: m fine-grained structured sparse neural networks from scratch. *arXiv preprint arXiv:2102.04010*.