Masks Can be Learned as an Alternative to Experts

Peiyu Liu[1], **Tianwen Wei**[3] , **Bo Zhu**[3], **Wayne Xin Zhao**[2*] , **Shuicheng Yan**[3,4]
[1] University of International Business and Economics
[2] Renmin University of China, [3] Skywork AI, [4] National University of Singapore
`liupeiyustu@163.com,tianwen.wei@kunlun-inc.com,zhubo@singularity-ai.com,`
`batmanfly@gmail.com,shuicheng.yan@gmail.com`

Abstract

In this work, we investigate how to sparsify a pre-trained dense large language model into a mixture-of-experts (MoE) architecture for faster inference. Our approach applies mask matrix to the activations for each expert, constrained by L_0 regularization to minimize the number of activated parameters. To ensure minimal performance loss under this constraint, we initialize the model with all parameters active and progressively sparsify it during training. This approach proves more efficient than one-shot sparsification techniques, which typically require significant resources for performance recovery. Moreover, our approach automatically identifies shared, token-specific, and inactive experts, allowing for more efficient allocation of computational resources. Through extensive experiments, we achieve up to 97% performance retention on downstream tasks with only 50% of the feed-forward parameters activated in dense models. Beyond improving inference efficiency, this strategy of sharing computational units among experts provides a principled foundation for building more scalable and generalizable MoE architectures, paving the way for future expert-based model designs. Our code is available at `https://github.com/lpyhdzx/Mixture-of-Masks`.

1 Introduction

Under the guidance of scaling laws, the parameter count in large language models (LLMs) has continued to rise, with models ranging from LLaMA 7B to 70B parameters. To alleviate the substantial computational burden associated with model inference and deployment, various model compression techniques have been proposed. However, their application to LLMs often results in unacceptable degradation of performance. Thus, a critical challenge remains: *how to effectively reduce inference computation without sacrificing model performance?*

* Corresponding author.

Table 1: Comparison of MoM and MoE. “Flexibility” refers to the adaptability in expert structure design, “Mem” indicates memory usage, and “Training Cost” reflects the computational budget required for training.

Methods	Flexibility	Mem	Training Cost
MoE	Regid	High	High
MoEfication	Regid	Low	Low
MoM	Flexible	Low	Minimal

One promising direction is sparse activation. A notable example is the Mixture-of-experts (MoE) approach, which designs multiple expert structures with extensive parameters but activates only a subset during computation This limits the number of active parameters and effectively mitigates the computational load. Despite the effectiveness of current sparse activation methods, they typically require training from scratch, which incurs prohibitive computational costs. An alternative research direction explores converting existing dense models into sparsely activated ones. Techniques such as MoEfication (Zhang et al., 2022), LLaMA-MoE (Zhu et al., 2024), and Turbo Sparse (Song et al., 2024) exemplify this approach by treating specific dimensions [1] of the weights in the feed-forward network (FFN) as expert structures, selectively activating these dimensions during forward computation. Although these methods avoid the need to retrain from scratch, they rely on heuristic-based expert construction (*e.g.,* equally partitioning the dimensions of weight matrices across experts), which neglects the varying significance of different dimensions within large language models. This can lead to suboptimal performance, as it overlooks the fact that some dimensions can be pruned while others can be shared across experts.

[1]In this paper, we use the term *dimension* to refer to each feature channel in the hidden state, along with the corresponding rows or columns in the projection weight matrices.

To address these challenges, our approach is guided by the principle of *maximizing efficiency while maintaining model performance and structure integrity*. Building on this principle, and drawing inspiration from MoEfication (Zhang et al., 2022), we focus on a key component of Transformer-based models, *i.e.,* FFN. Specifically, we propose transforming the dense FFN structure into a sparse MoE module, leveraging a routing mechanism to enable the selective activation of parameters. However, achieving activated sparsity with MoEfication style is non-trivial due to the following practical challenges: (1) identifying the varying importance of different weight dimensions. (2) minimizing performance degradation.

To this end, we propose a novel sparsification framework for large language models, named Mixture-of-Masks (MoM) . MoM introduces a learning-based mechanism that dynamically selects and activates a subset of parameters through binary mask matrices. These masks are integrated into the FFN structure, serving as substitutes for traditional expert modules. By training the masks with L_0 norm constraints, MoM adaptively learns which dimensions to share, specialize, or prune, enabling token-specific expertization without relying on heuristic-based construction methods. To ensure a smooth sparsification process, masks are initialized with all ones, preserving the dense model's original structure at the beginning of training. This design minimizes performance degradation while allowing the model to gradually prune parameters and integrate multiple compression techniques during continued pre-training.

We conducted comprehensive experiments to evaluate the performance of MoM, focusing on model accuracy restoration, data efficiency, and inference costs. In publicly available evaluation benchmarks, MoM outperformed existing methods with fixed expert allocation, restoring 97% of the dense model's accuracy compared to 90% achieved by MoEfication (Zhang et al., 2022). MoM effectively maintains model performance while exhibiting superior data efficiency during training. In addition, starting from the original dense model, MoM gradually prunes parameters with minimal accuracy loss, achieving the compression target after processing just 10B tokens. In contrast, methods with static expert partitioning introduce significant structural changes, resulting in prolonged training times to restore model accuracy.

In addition, we also conducted an in-depth analysis to shed light on why MoM works well. Upon analyzing the experts obtained through MoM training, we observed that the experts were automatically divided into shared experts, independent experts, and ineffective experts. Both shared and ineffective experts can be excluded from routing, thereby reducing the model's inference costs and further improving efficiency. This observation is consistent with conclusions from some of the most advanced model structures (Dai et al., 2024; Yang et al., 2024), opening new directions for us to explore the characteristics of MoE architectures.

2 Methods

We now detail our proposed method, Mixture-of-Masks (MoM), which is designed to produce compact models by selectively activating a subset of parameters. This approach achieves sparsity and computational efficiency while maintaining strong performance within a modest resource budget.

2.1 Preliminary

We first present the background for our approach to mixture-of-experts architecture and the pruning methods.

Mixture-of-Experts. MoE enhances the Transformer architecture by introducing multiple Feed-Forward Networks (FFNs), known as "experts", within each Transformer block. During computation, only a subset of these experts is activated based on the input, significantly increasing model capacity while keeping computational costs low. Formally, the output of MoE architecture y can be computed as:

$$h = \Sigma_{i=1}^{n} p_i(x) E_i(x), \tag{1}$$

where $p_i(x)$ and $E_i(x)$ are the gate value and the output vector of the i-th expert for a given input x, respectively. Inspired by this, recent works (Zhang et al., 2022; Zhu et al., 2024) have shown that transforming a dense model into an MoE structure effectively achieves activation sparsity. However, current methods randomly assign hidden dimensions to experts, disregarding the fact that different dimensions contribute unequally to model performance. Such suboptimal assignments can lead to notable performance degradation, highlighting the need for approaches that construct expert modules that adapt to both model structure and the underlying pretraining data.

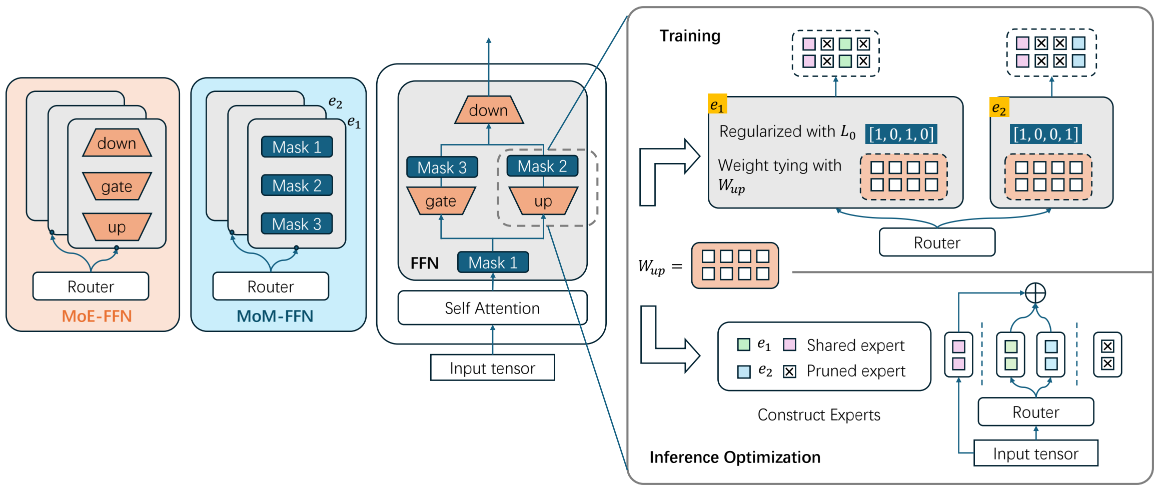


Figure 1: Overview of MoM architecture. MoM-FFN trains multiple masks as experts instead of multiple copies. For training, the masks are regularized by L_0 normalization. For inference, we construct experts based on the learned expert allocation pattern. Specifically, columns that are never routed to any expert are pruned (crossed box), while columns that are consistently routed across all cases are explicitly shared among all experts (pink box).

Model Pruning. Pruning aims to achieve sparsity in large models by removing less important weights or components. Common approaches include structured pruning (removing specific structures) and unstructured pruning (removing individual weights). However, for large language models, scaling laws indicate that a large number of parameters is crucial for optimal performance. Directly reducing the total number of parameters can harm the model's capacity. Therefore, we propose the concept of "activation pruning", which maintains the total number of parameters while pruning only the activated ones. This approach aims to preserve the model's advanced capabilities while reducing computational costs. In this context, we follow the study (Louizos et al., 2017) of L_0 regularization to constrain the sparsity of large language models.

2.2 Constructing Experts by Masks

Following the work (Zhang et al., 2022), we treat the dimensions of weights in FFN as the minimal unit, and *experts* are constructed by grouping multiple dimensions together. Instead of manually assigning dimensions to experts, our objective is to dynamically group related dimensions into experts based on their interrelationships. In this section, we introduce **M**ixture-**o**f-**M**asks (MoM),a mask-based expert construction approach that enables dynamic selection of dimensions.

To implement this, we adopt a LLaMA (Touvron et al., 2023) style decoder-only model with N Transformer layers. Let $\mathbf{W}_{gate}, \mathbf{W}_{up} \in \mathbb{R}^{e \times d}$ denote the weight matrices for the gate and up projections, respectively, and let $\mathrm{Swish}(\cdot)$ be the activation function. The output $y \in \mathbb{R}^d$ of FFN can be described as follows:

$$\begin{aligned} h &= x\mathbf{W}_{up} \odot \mathrm{Swish}(x\mathbf{W}_{gate}), \\ y &= h\mathbf{W}_{down}. \end{aligned} \tag{2}$$

Our goal is to insert mask variables (denoted as $\mathbf{v} \in \mathbb{R}^d$) at various positions in this formulation to achieve sparse activation of different components. Depending on where the masks are inserted, we then introduce our method within two steps: (1) *basic masking method* that selects and masks intermediate hidden dimensions shared by both projections in the FFN, and (2) *fine-grained strategies* that apply independent masking to the two projection weights to further increase sparsity.

Basic Masking Method. The basic characteristic of the FFN structure is that expanding through the gate and up components can increase model capacity, but it also introduces significant redundancy. Our approach involves adding a mask module with values {0,1} after the gate and up outputs. Then, the output of the FFN becomes:

$$h = [x\mathbf{W}_{up} \odot \mathrm{Swish}(x\mathbf{W}_{gate})] \odot \mathbf{v}. \tag{3}$$

We frame mask determination as a constrained optimization problem, allowing the masks to be dynamically learned during training rather than statically assigned, as was done in previous work (Zhang et al., 2022; Zhu et al., 2024). This dynamic approach allows dimensions corresponding to similar tokens to be grouped together after training, aligning with the core idea of the MoE structure, *i.e.,* similar tokens activate similar sets of parameters, improving both efficiency and specialization. Finally, the sparsity is computed as:

$$R(\mathbf{v}) = \frac{\sum_{i=1}^{d} \mathbb{I}(v_i = 0)}{d}, \quad (4)$$

where $\mathbb{I}$ is a indicator function and d is the dimensionality of the vector $\mathbf{v}$.

Fine-grained Masks Strategies. While the basic masking method provides an initial reduction in redundancy, further improvements can be achieved by targeting specific components of the FFN with more fine-grained masking strategies. This is because different components, such as the gate and up projections, contribute unequally to overall model performance and may benefit from different sparsity levels (Song et al., 2024), thereby enabling more targeted and effective pruning.

Then, we extend the masking approach to fine-grained modules (*i.e.,* gate, up, and hidden states separately). For gate and up projections, the final sparsity is calculated as $R_{FFN} = (R_{gate} \odot R_{up})$. To further improve sparsity, we also apply masking to the FFN inputs. This is based on the observation that only a small subset of input dimensions typically needs to be projected into higher-dimensional space. The final sparsity is thus calculated as $R_h \odot R_{FFN}$.

2.3 Training with L_0 regularization

Building on the mask construction strategy described earlier, the final set of experts is determined by the parts of the model that are retained by the learned masks. To increase sparsity and reduce the number of active parameters, we frame this as a constrained optimization problem. Specifically, the objective is to learn mask matrices that dynamically select sub-dimensions conditioned on the input tokens, mirroring the expert selection behavior in MoE architectures.

Inspired by the L_0 regularization method (Louizos et al., 2017), we parameterize the masks to model hard concrete distributions. These distributions are defined on the interval $[0, 1]$ but concentrate their probability mass at 0 or 1, enabling discrete decisions to either prune or retain specific dimensions. In addition, by starting with all parameters active, the model is progressively sparsified during training, ensuring minimal performance loss.

To formalize this process, let l, and E represent the number of layers and the number of experts per layer, respectively. Given a target sparsity ratio R_t, the optimization objective for each layer is defined as:

$$L_{mask} = \sum^{l} \sum^{E} (R_e - R_t) + (R_e - R_t)^2, \quad (5)$$

where R_e denotes the actual sparsity ratio of a layer after applying the learned mask. At the beginning of training, we initialize $R_e = 1.0$, indicating no sparsity is applied. As training progresses, the mask is learned dynamically to gradually reduce the sparsity ratio toward the target R_t , typically set to values such as 0.25 or 0.5. This objective encourages the model to meet the desired sparsity level while mitigating potential performance degradation.

Since each expert learns independently, the model naturally categorizes dimensions into three types: *shared dimensions* (across all experts), *independent dimensions* (specific to individual experts), and *unused dimensions* (not allocated to any expert). By automating this process, we reduce the risk of introducing prior biases and improve the efficiency of the model's sparse activation mechanism. Then we will introduce inference optimization based on identified expert types.

2.4 Inference Optimization via Expert Pattern Identification

In this section, we leverage the expert patterns derived from L_0 regularization to optimize inference, using tailored strategies for shared, independent, and redundant experts:

• **Shared experts.** Shared experts are dimensions that remain active across all experts. These are processed only once, as their outputs can be reused across different inputs, thereby reducing memory usage and computational load.

• **Independent experts.** For independent experts, we introduce a routing mechanism that selectively activates experts, following the standard MoE routing strategy. This approach helps to significantly decrease computational costs by activating only the necessary experts.

• **Redundant experts.** Redundant experts are dimensions that are never routed across any of the experts. These dimensions are pruned, as their contribution to model performance is negligible, further reducing the total number of parameters.

Interestingly, several advanced studies (Dai et al., 2024) have manually divided experts into shared and independent groups, arguing that shared experts capture common knowledge while independent experts focus on domain-specific tasks. Our findings after applying MoM are consistent with this, but in our case, the model automatically learns this division. To further examine this automatically learned expert allocation pattern, we analyze the learned expert distributions in Section 3.4, which reveal architectural principles for effective expert assignment.

3 Experiments

In this section, we first set up the experiments and then report the results and analysis. Then we conduct a detailed analysis for different MoE settings.

3.1 Experimental Setup

Datasets. By continuing pre-training process, we aim to restore the performance when selectively activating a subset of the parameters. So we use a mixture of several data sources to cover several domains, including: (1) RedPajama (Computer, 2023), a mixture of CommonCrawl, C4, Github, Wikipedia, Books, arXiv, and StackExchange. We try to cover a diverse set of domains for a better performance restoration. (2) Dolma (Soldaini et al., 2024), built from a diverse mixture of web content, scientific papers, code, public-domain books, social media, and encyclopedic materials. (3) SkyPile (Wei et al., 2023), a large-scale Chinese dataset containing approximately 150B tokens. For evaluation, we follow the study (Wei et al., 2023; Zhu et al., 2024) and utilize HellaSwag to evaluate the model ability since the performance on HellaSwag is reported to grow smoothly during pre-training.

For a comprehensive assessment of downstream tasks, we follow Sheared LLaMA (Xia et al., 2024) and use lm-evaluation-harness package (Gao et al., 2024) to evaluate the following tasks: BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SiQA (Welbl et al., 2017), HellaSwag (Zellers et al., 2019) and ARC easy (Clark et al., 2018).

Implementation. For the implementation of continued pre-training setting, we utilize the open-source SkyWork model (Wei et al., 2023) with 300M parameters for our experiments. SkyWork provides a general LLaMA-style model framework, ensuring that our method can be easily transferred to other similar frameworks. Additionally, since all data associated with this model is accessible, it provides a fair platform for comparing the effectiveness of different methods. Based on this model, we start from a checkpoint trained with 200 billion tokens. According to the Section 2.2, we provide four variants of different masking strategies: MoM, MoM$_{FH}$, MoM$_{FW}$ and MoM$_{FWH}$:

- **MoM** serves as the base variant, applying coarse-grained masks to the intermediate activations in the FFN module.
- **MoM$_{FH}$** extends MoM by introducing fine-grained masks over the input hidden state dimensions.
- **MoM$_{FW}$** applies fine-grained masking to the FFN weights, specifically separating the *gate* and *up* projections.
- **MoM$_{FWH}$** combines both MoM$_{FH}$ and MoM$_{FW}$, applying masks to both hidden states and weights to achieve higher sparsity.

Subsequently, we assess how well each method recovers model performance under limited training resources. To further demonstrate the scalability of our approach, we also conduct experiments on a larger LLaMA-3-8B model (AI@Meta, 2024). In the next section, we will present the detailed experimental results.

Baseline Models. Here we introduce relevant methods as our baselines.

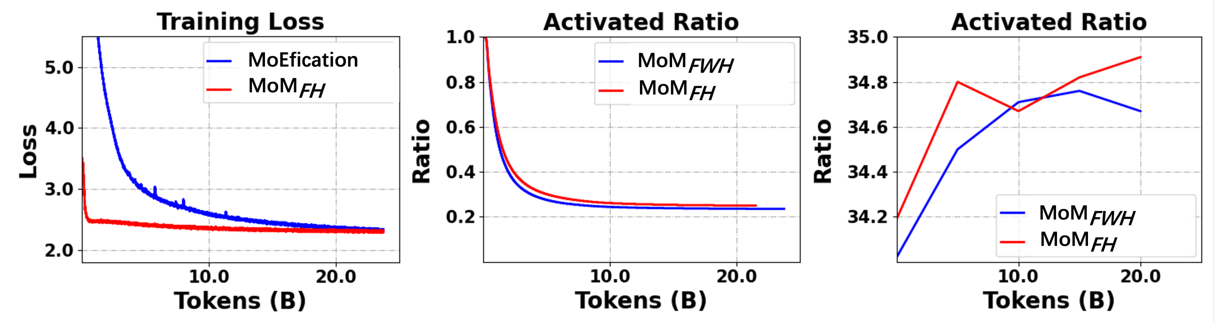
- **MoEfication** (Zhang et al., 2022) for sparse activation. MoEfication converts dense models into a MoE version by splitting the FFN weights into multiple partitions as experts, with dimensions evenly distributed across experts.
- **Pruning.** We additionally employ model pruning as a baseline to validate the effectiveness of activation-based compression in comparison to full parameter pruning. Specifically, when the total number of experts is set to 1, our method degenerates into conventional pruning approaches, reducing the total number of parameters. We use this configuration as a variant of pruning to provide a comparative baseline.

3.2 Main Results

Comparing with MoEfication. First, we show dense downstream task evaluation results on both dense models and activation pruning methods. As shown in Table 2, MoM uses limited training tokens and outperforms MoEfication in all tasks. Specifically, MoM preserves 97% of original dense

Table 2: Models with MoM outperforms publicly available methods of sparsification. Models with "†" are our reproduced result.

Model (#tokens for training)	**#Activated**	**BoolQ**	**PIQA**	**SiQA**	**HellaSwag (10)**	**ARC-E**	**Average**
		Commonsense & Reading Comprehension					
Dense (200B)	100%	58.4	67.8	39.1	36.9	49.5	50.3
MoEfication (20B)†	50%	59.4	58.5	36.5	29.3	42.0	45.1
MoM (20B)	75%	**60.0**	**66.9**	36.3	**35.3**	46.6	49.0
MoM$_{FH}$ (20B)	50%	59.5	65.6	**37.2**	34.9	**48.2**	**49.1**



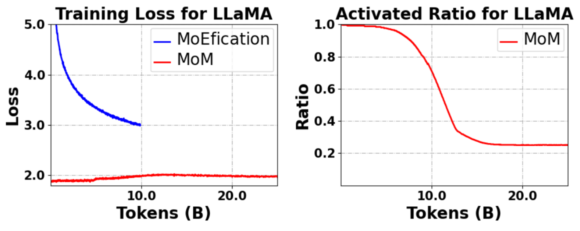
(a) Training Loss. (b) Activated Sparsity. (c) Accuracy of HellaSwag.

Figure 2: Model loss and activated sparsity. (a) shows the comparison between MoM and MoEfication. (b) and (c) illustrate the compression rate and downstream task performance of our method under the fine-grained masking strategy.

model's performance (49.1 *vs.* 50.3), while MoEficaiton only preserves around 90% (45.1 *vs.* 50.3).

As for the data efficiency, we observe obviously from Figure 2 (a), that our method (red curve) quickly converges to the same loss as the MoEfication (blue curve), whereas MoEfication requires nearly 20B tokens to achieve a similar level. This demonstrates that MoM achieves better data efficiency through its near-lossless compression strategy, where learned masks minimize performance degradation more effectively than one-shot sparsification methods like MoEfication.

As for the effect of our method during the compression process, Figure 2 (b,c) shows that the recovery of model performance remains stable across various compression rates. Specifically, performance recovery stays within 92% of the dense model (*i.e.,* 34.2 *vs.* 36.9), indicating minimal degradation even with significant compression. In the early stages of training, there is a slight drop in performance, despite a low loss value, but this is quickly corrected as training continues. The overall trend suggests that our method ensures performance stabilizes and recovers effectively. These results confirm the robustness of our approach, demonstrating that it achieves substantial compression without severely affecting model accuracy.

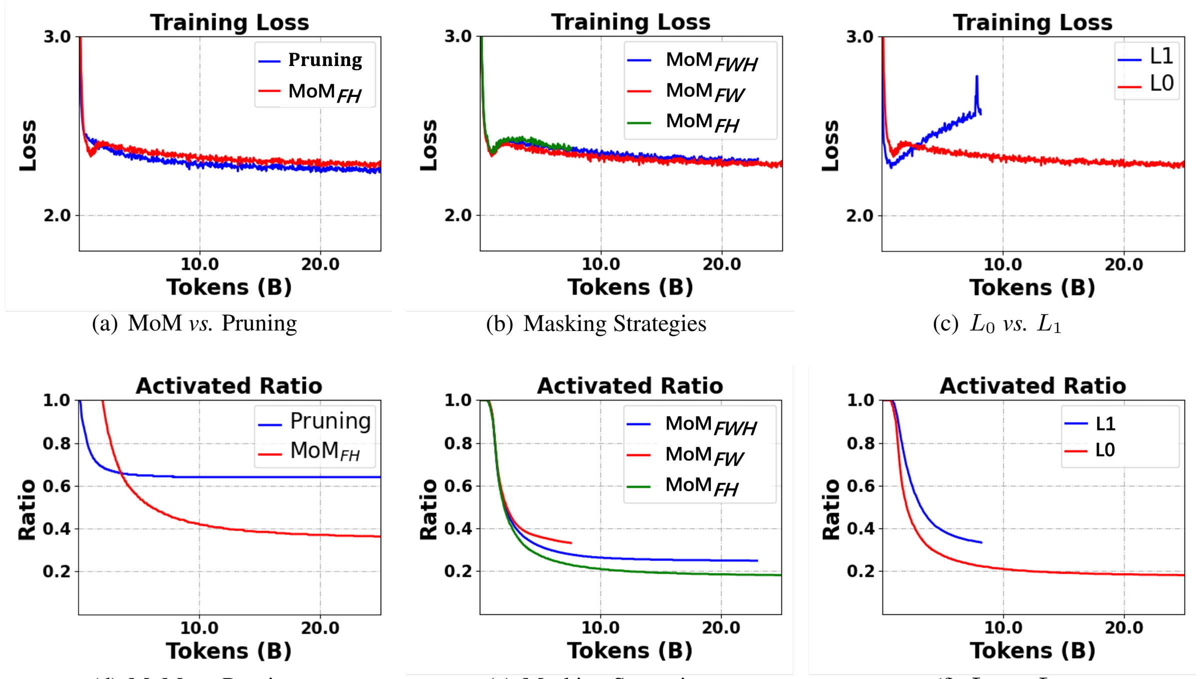


(a) Loss (b) Sparsity

Figure 3: Extending experiments on LLaMA-3-8B.

To demonstrate the scaling effect, we extend to the LLaMA3-8B model (see Figure 3). As for data preparation, existing work has shown that more complex datasets are often required to recover the model after compression, including data ratios (Xia et al., 2024) and larger data sizes (Zhu et al., 2024).

Therefore, we adopt a classic dataset preparation pipeline to ensure a fair comparison (Wei et al., 2023). The results show that our method can still achieve faster model compression on the 8B model. It is worth noting that in LLaMA-8B, the compression process can be completed more quickly, requir-



(a) MoM *vs.* Pruning (b) Masking Strategies (c) L_0 *vs.* L_1

(d) MoM *vs.* Pruning (e) Masking Strategies (f) L_0 *vs.* L_1

Figure 4: Influence of Masking Strategies for different metrics. Figures (a,d) denote the comparison with pruning. Figures (b,e) denote the ablation of different masking strategies. Figures (c,f) denote the ablation study of different learning strategies.

ing only a budget of 15B tokens. However, model recovery is a more prolonged process. Overall, the model performance gradually improves, while the recovery process for MoEfication might be a more long-term task. This result demonstrates that our method offers greater data efficiency compared to MoEfication.

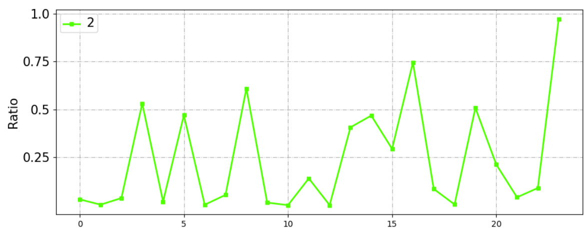
Comparing with Pruning. In order to demonstrate the strength of reducing activated parameters over pruning total parameters, we design two comparison ablation experiments: (1) continue pre-train with static sparsity. We set the sparsity to {25%, 50%} and do continue pre-train to see whether the performance can restore to the original level. This study is to test the ability to restore performance after pruning. (2) continue pre-train with dynamic sparsity. We learn the masks using L_0 normalization similar to MoM. While, we constrain the number of experts to only 1. Specifically, as shown in Figure 4 (a,d), direct pruning methods fail to match the compression rate achieved by MoM when maintaining comparable performance. This highlights the advantage of our progressive, mask-based approach, which enables more effective sparsification without sacrificing accuracy.

3.3 Detail Analysis

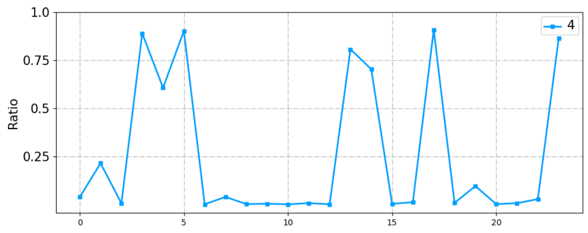
Here we provide detailed studies of two important aspects of learning masks: masking strategies and learning strategies.

Masking Strategies. As introduced in Section 3.1, we categorize the masking strategies based on their target modules. Then, to further compare the impact of these strategies on model performance, we continue pretraining the 300M models on 20B tokens and report the evaluations on the HellaSwag dataset in Figures 4 (b) and (e). From the sparsity ratio, we find that MoM$_{FH}$ achieves a lower sparsity ratio than the others. Meanwhile, these compression gains sacrifice the performance as we can see from the evaluation in Hellaswag. Therefore, we recommend using MoM$_{FH}$ when performance is the priority. If a larger training budget is available, MoM$_{FWH}$ can be advantageous as it results in more sparsified models, which may be better suited for low-resource machines.

Learning strategies. In practice, optimizing binary masks can be challenging due to their discrete nature. Therefore, it is crucial to design an appropriate technique for learning effective masks. Popular approaches include normalization methods



(a) Expert-2.



(b) Expert-4.

Figure 5: Visulizaton of experts selection.

such as L_1 and L_0 regularization. To evaluate the effectiveness of these techniques, we performed an ablation study and present the results in Figure 4 (c, f). As shown in the figure, applying L_1 regularization results in a significant degradation in model performance at the early stages of training, with the loss rapidly increasing. This indicates that L_1 is not well-suited for sparsification tasks. Consequently, we halted the L_1 experiment after training with less than 10B tokens, as the sparsity achieved was considerably lower compared to L_0. In contrast, the L_0 regularization technique proved to be much more effective in achieving sparsity, validating its suitability for tasks involving sparse activation.

3.4 Analysis for the Experts

Experts Selection Across Layers. As shown in Section 2.4, our method automatically learns to assign dimensions into *shared*, *independent*, and *pruned* experts. To further understand this result, we visualize the experts at different layers in the Figure 5. We observe varying levels of preference for the experts across layers. For example, Expert 2 shows a relatively even level of participation, with activation remaining below 50% and spread across all layers. In contrast, Expert 4 exhibits activation in some layers that reaches approximately 80%, but the number of activated layers remains relatively low, around 30%, which maintains higher efficiency.

Then we analyze the roles of shared, independent, and pruned experts across layers and their relationship to activation sparsity. Specifically, "8-hit" dimensions are those routed to all eight experts, indicating shared usage (blue bars), while "0-hit" dimensions are never selected and are considered pruned (red bars) (see Figure 6 in Appendix A.2). Our analysis reveals two notable patterns that shed light on how expert roles evolve across model layers: **(1) in the earlier layers, a larger number of experts are pruned, indicating that the model primarily focuses on general, token-agnostic representations.** This results in higher activation sparsity, as many parameters contribute little to early-stage processing. As the model progresses to deeper layers, sparsity decreases, suggesting that more experts are needed to capture increasingly complex semantic features. **(2) In the final layers (21-23), we observe a rise in shared experts, even though these layers may handle more complex and nuanced semantic tasks.** This implies that, despite the increased task complexity, there are underlying patterns or features that remain consistent across tasks, captured effectively by shared experts. This discovery reflects the emergence of shared representations and may underpin the model's generalization ability. Together, our findings offer valuable insights into the interpretability and efficiency of deep MoE models, showing how expert roles evolve across layers. Understanding these dynamics could lead to more efficient model architectures that balance the trade-off between task-specific adaptations and shared knowledge extraction.

Experts Selection Across Tasks. Then we empirically investigate whether different experts contain domain-specific information. For the dataset, we use the benchmark of MMLU where the tasks are categories into four groups (Hendrycks et al., 2021). First, we collect the output of the gate projections across all the layers and form a gate distribution vector of the dimension of 8 (experts per layer) $\times$ 24 (layers). Then we calculate the cosine similarity of the vectors and report the results in the Figure 6 (b) in Appendix A.2. Higher similarity indicates that tasks activate experts in a similar pattern, suggesting shared domain characteristics; lower similarity implies domain divergence. We observe a clear distinction between the `STEM` and `humanities` subjects, as shown by the clustering patterns in the heatmap. Additionally, three history tasks—`high school european history`, `high school US history`, and `high school world history`—exhibit strong correlations with each other, more so than with other tasks. This is likely due to the significant overlap in the subject matter across these history topics, which makes them more similar compared to other tasks.

Notably, even though our experts are constructed using masks rather than the traditional MoE approach, they still successfully learn to capture domain-specific information and categorize tokens based on their content. This demonstrates that our approach retains the essential characteristics of MoE models while offering greater flexibility.

4 Related Work

Pruning. Existing models are often impractical to deploy due to their large parameter count. A direct solution to this issue is pruning (Xia et al., 2024), which involves the removal of model weights. Pruning generally follows two primary approaches. The first approach is structured pruning (Xia et al., 2024), which typically achieves higher compression rates and enhances inference efficiency. However, this method often results in significant performance degradation due to the coarse granularity of pruning, which inadequately preserves essential weights. Consequently, extensive retraining is often necessary to recover model performance. The second approach is unstructured pruning (Song et al., 2024; Wang et al., 2024), which eliminates non-essential weight values. This finer-grained method effectively retains important weights, resulting in minimal performance loss. However, it does not substantially improve inference speed. The traditional work focus on reducing the total parameters which may not against the spirit of scaling laws (Kaplan et al., 2020): the large language models where the superior ability comes from a large number of parameters.

Sparse Methods. In contrast to pruning, activating fewer parameters during computation maintains model capabilities without increasing computational load. A common approach is the Mixture of Experts structure (Fedus et al., 2022; Lepikhin et al., 2020), where multiple FFN structures serve as experts, with only a subset activated during computation, reducing parameter usage. Numerous studies have validated the efficiency of this method in large-scale models, such as Mixtral (Jiang et al., 2024), which implements a standard MoE at a 7B scale, and DeepSeek (Dai et al., 2024), which combines shared and unique experts for different tasks. Existing pre-trained models can also be converted into MoE structures through "MoEfication" (Zhang et al., 2022), successfully applied to models from BERT to Llama-MoE (Zhu et al., 2024). However, these structural changes often cause performance degradation, which this paper seeks to address.

5 Conclusion

We propose Mixture-of-Masks (MoM), a novel approach to transforming dense models into sparsely activated architectures that achieve high efficiency while maintaining strong performance. With MoM, we achieved 97% of the performance of the dense counterpart, activating only 50% of the FFN parameters, significantly reducing computational costs within a 10B parameter training budget. Compared to traditional MoE, MoM demonstrates superior efficiency in both parameter utilization and computational overhead. Beyond its performance advantages, MoM provides valuable insights into the distribution of experts, uncovering key design principles that can guide the development of more interpretable and efficient MoE architectures. These findings deepen our understanding of optimizing sparse models and open new avenues for balancing performance and efficiency in large-scale language models. Next, we will extend MoM to attention and others for further parameter efficiency.

Acknowledgments

This work was partially supported by National Natural Science Foundation of China under Grant No. 92470205. Peiyu Liu and Xin Zhao are the corresponding authors.

References

AI@Meta. 2024. Llama 3 model card.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

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 *NAACL*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind

Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Together Computer. 2023. Redpajama: an open dataset for training large language models.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 1280–1297. Association for Computational Linguistics.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23:120:1–120:39.

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, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *CoRR*, abs/2401.04088.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

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*.

Christos Louizos, Max Welling, and Diederik P. Kingma. 2017. Learning sparse neural networks through l_0 regularization. *CoRR*, abs/1712.01312.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Raghavi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. 2024. Dolma: an open corpus of three trillion tokens for language model pretraining research. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 15725–15788. Association for Computational Linguistics.

Yixin Song, Haotong Xie, Zhengyan Zhang, Bo Wen, Li Ma, Zeyu Mi, and Haibo Chen. 2024. Turbo sparse: Achieving LLM SOTA performance with minimal activated parameters. *CoRR*, abs/2406.05955.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hongyu Wang, Shuming Ma, Ruiping Wang, and Furu Wei. 2024. Q-sparse: All large language models can be fully sparsely-activated. *CoRR*, abs/2407.10969.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. 2023. Skywork: A more open bilingual foundation model. *Preprint*, arXiv:2310.19341.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text, NUT@EMNLP 2017, Copenhagen, Denmark, September 7, 2017*, pages 94–106. Association for Computational Linguistics.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared llama: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 technical report. *CoRR*, abs/2407.10671.

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*.

Zhengyan Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022. Moefication: Transformer feed-forward layers are mixtures of experts. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 877–890. Association for Computational Linguistics.

Tong Zhu, Xiaoye Qu, Daize Dong, Jiacheng Ruan, Jingqi Tong, Conghui He, and Yu Cheng. 2024. Llama-moe: Building mixture-of-experts from llama with continual pre-training. *arXiv preprint arXiv:2406.16554*.

A Appendix

A.1 Analysis of the Hyperparameters

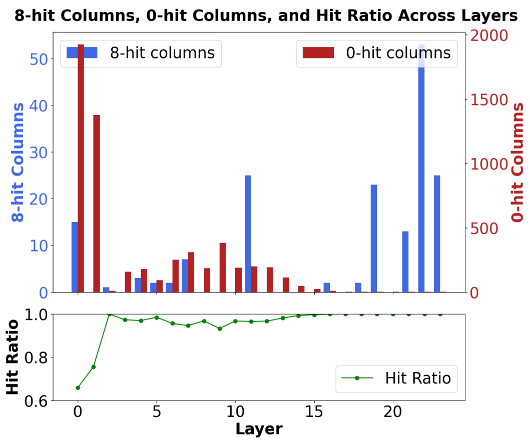
Table 3 summarizes the training hyperparameters used in our experiments, including the learning rate, mask learning rate, batch size, weight decay, and Adam optimizer parameters. These configurations are kept consistent across all training setups unless otherwise specified.

Initial Learning Rate: lr	2e-5
Mask Learning Rate: mlr	$lr * 1e4$
Global Batch Size: bs	2048
Weight Decay: wd	0.1
Adam Parameters: $beta1$,$beta2$	0.9,0.95

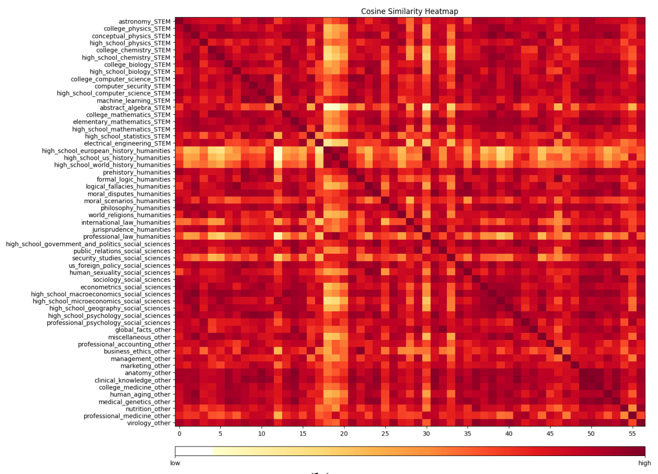
Table 3: Training hyperparameters.

A.2 Visualization of the Gating.

First, we analyze the roles of shared, independent, and pruned experts across layers and their relationship to activation sparsity. Specifically, we use 8-hit dimensions to represent shared experts (blue bars) and 0-hit dimensions to represent pruned experts (red bars) in Figure 6 (a). Then we empirically investigate whether different experts contain domain-specific information. Specifically, we calculate the cosine similarity of the vectors and report the results in the Figure 6 (b). We observe a clear distinction between the `STEM` and `humanities` subjects, as shown by the clustering patterns in the heatmap. Additionally, three history tasks—`high school european history`, `high school US history`, and `high school world history`—exhibit strong correlations with each other, more so than with other tasks. This is likely due to the significant overlap in the subject matter across these history topics, which makes them more similar compared to other tasks.



(a)



(b)

Figure 6: Analysis of the experts. (a) denotes the visualization of experts' selection and (b) denotes the routing distribution similarity across MMLU 57 tasks.