DIVE into MoE: Diversity-Enhanced Reconstruction of Large Language Models from Dense into Mixture-of-Experts

Yuchen Feng[1,2*], **Bowen Shen**[1,2*], **Naibin Gu**[1,2], **Jiaxuan Zhao**[1,2],
Peng Fu[1,2†], **Zheng Lin**[1,2], **Weiping Wang**[1]

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{fengyuchen,shenbowen,fupeng}@iie.ac.cn

Abstract

Large language models (LLMs) with the Mixture-of-Experts (MoE) architecture achieve high cost-efficiency by selectively activating a subset of the parameters. Despite the inference efficiency of MoE LLMs, the training of extensive experts from scratch incurs substantial overhead, whereas reconstructing a dense LLM into an MoE LLM significantly reduces the training budget. However, existing reconstruction methods often overlook the diversity among experts, leading to potential redundancy. In this paper, we come up with the observation that a specific LLM exhibits notable diversity after being pruned on different calibration datasets, based on which we present a **Div**ersity-**E**nhanced reconstruction method named DIVE. The recipe of DIVE includes domain affinity mining, pruning-based expert reconstruction, and efficient retraining. Specifically, the reconstruction includes pruning and reassembly of the feed-forward network (FFN) module. After reconstruction, we efficiently retrain the model on routers, experts and normalization modules. We implement DIVE on Llama-style LLMs with open-source training corpora. Experiments show that DIVE achieves training efficiency with minimal accuracy trade-offs, outperforming existing pruning and MoE reconstruction methods with the same number of activated parameters. Code is available at: https://github.com/yuchenblah/DIVE.

1 Introduction

Large Language Models (LLMs) based on Transformer (Vaswani et al., 2017) have demonstrated outstanding capabilities across various domains. However, prevailing dense LLMs with all parameters activated during inference involve significicanst computational and memory overheads, hindering the practical deployment of LLM-based services (Zhu et al., 2023; Gu et al., 2024). While
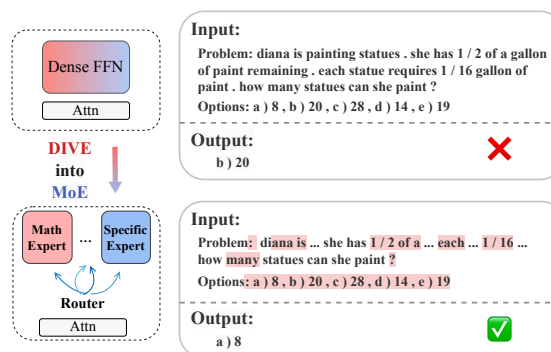

Figure 1: Example inputs and outputs of a dense LLM and DIVE, sampled from MathQA (Amini et al., 2019). Tokens highlighted with the red background are routed to the Math expert. DIVE benefits from the inherent diversity in the dense LLM with specialized experts.

static compression like structured pruning can reduce these overheads, they often incurs downgraded capacity and generalization abilities.

Unlike reducing the size of pre-trained LLMs, Mixture-of-Experts (MoE) LLMs, such as Mixtral 8x7B (Jiang et al., 2024), enable scalable pretraining by activating only a subset of parameters during inference, achieving superior performance with the same or fewer active parameters compared to dense LLMs (Lepikhin et al., 2021). MoE models typically differ from dense LLMs in the feedforward network (FFN), replacing the single FFN with multiple smaller FFN experts. These experts are selectively activated via a parameterized router, while other components like embeddings, language modeling heads, and multi-head attention (MHA) modules remain unchanged.

Although MoE models share similarities with dense LLMs, they are usually pre-trained from scratch, which involves significant costs and potential instability (Wei et al., 2024). To mitigate these issues, there has been increasing interest in converting dense models into MoE architectures. However, existing methods, like duplicating FFN

* Equal Contribution.
† Corresponding Author.

modules (Komatsuzaki et al., 2023) and random splitting (Zhu et al., 2024), often produce homogeneous experts that lack diversity. This necessitates substantial retraining to enhance expert diversity and achieve optimal performance. In this paper, we aim to tackle this challenge from the perspective of expert reconstruction, addressing the following key question:

How can we reconstruct a dense LLM into MoE architecture while enhancing expert diversity and significantly reducing training costs?

In pursuit of enhancing diversity, we investigate structured pruning approaches, which derive smaller but potentially less generalizable models from existing LLMs. Pruning typically relies on a calibration set to assess weight importance (An et al., 2024). Given the aggressive nature of pruning and the limited size of calibration sets compared to pre-training corpora, certain pruning methods exhibit sensitivity to the calibration set. We observe that LLMs pruned with different calibration sets exhibit notable diversity across evaluation domains, and models pruned on specific domain data struggling to generalize. While such sensitivity can be a drawback, it effectively uncovers the inherent diversity in the weights of dense LLMs.

Building on the above insights from pruning, we propose a **Div**ersity-**E**nhancing expert reconstruction method of LLM named DIVE. As illustrated in Figure 1, DIVE leverages the inherent diversity of the original LLMs, and construct experts with varying capabilities. We firstly use domain affinity mining to identify expert domains and deploys pruning-based expert reconstruction. Then, we design an efficient retraining pipeline to recover the model's capabilities, only the reconstructed modules (routers, experts, and normalizations) need to be retrained with parameter-efficient tuning (PEFT). As a summary, our contributions are as follows:

- We introduce DIVE, an effective MoE reconstruction method for dense LLMs that enhances expert diversity, inspired by static pruning techniques.

- We propose an efficient retraining method, designed to minimize training costs. Only less than 1% of the parameters are required to be tuned.

- Experiments show the effectiveness and robustness of DIVE. With a 50% sparsity of the original FFN, DIVE outperforms existing pruning and MoE reconstruction methods, excelling in both language modeling and various downstream tasks.

2 Preliminary

2.1 Dense FFN in Transformer

The FFNs in standard Transformer-based dense LLMs (Vaswani et al., 2017) consist of fully connected layers that operate independently for each token. Given an input vector x, the forward pass through the FFN is:

$$FFN(x) = F_{\text{down}}\left(\sigma\left(F_{\text{up}}(x)\right)\right), \quad (1)$$

where F_{up} and F_{down} are linear transformations, and σ stands for activation functions (e.g., ReLU (Agarap, 2018)). For the SwiGLU activation used in Llama-style LLMs,

$$\sigma(F_{\text{up}}(x)) = \varsigma(F_{\text{gate}}) \odot F_{\text{up}}(x), \quad (2)$$

where F_{gate} is an additional linear transformation with the same shape as F_{up}, ς is the Swish function and $\odot$ denotes element-wise multiplication (Shazeer, 2020).

2.2 Sparse FFN in Mixture-of-Experts

Different from dense LLMs, in Transformer-based MoE LLMs, the FFN layers are typically replaced with MoE layers consisting of a router R and n original FFN-type experts $\{FFN_1, FFN_2, \ldots, FFN_n\}$, with only the top-$k$ being activated. The router, often a lightweight feed-forward neural network, learns to optimally allocate inputs to experts based on different data distributions.

Given an input x, the output of an MoE layer is the weighted sum of the outputs from selected experts:

$$o(x) = \sum_{i=1}^{k} w_i(x) \cdot FFN_i(x), \quad (3)$$

where $w_i(x) = \text{Softmax}(\text{TopK}(z(x)))_i$ representing the normalized weight for each expert with the routing logits $z(x)$ produced by the router.

3 Method

Aiming at improving expert diversity in MoE reconstruction, we introduce DIVE. Our method consists of three components: domain affinity mining, pruning-based expert reconstruction, and efficient retraining.
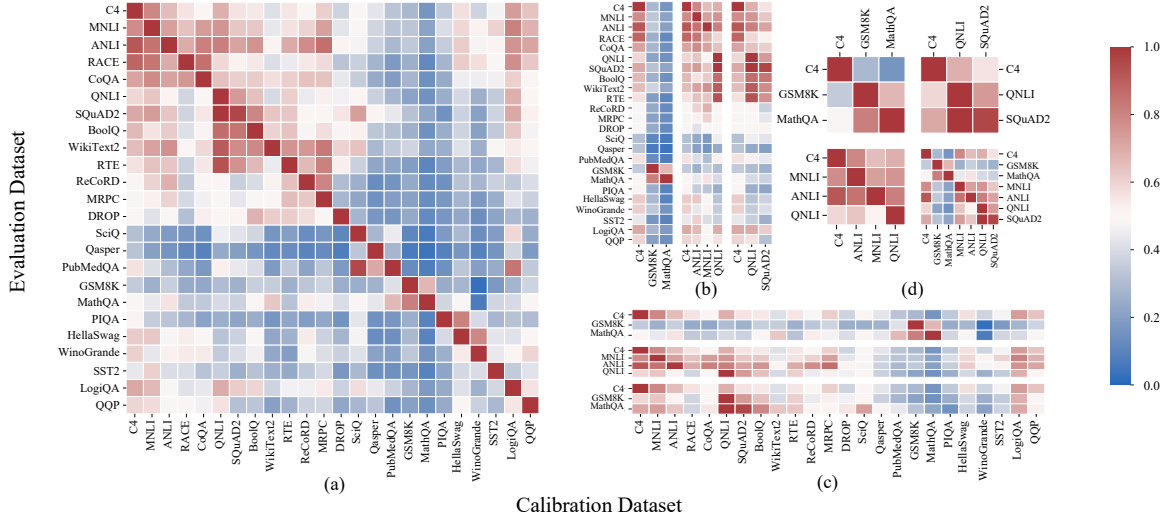
Figure 2: The heatmap shows normalized perplexity (PPL) with a 75% FFN pruning ratio on LLaMA2-7B. According to the definition of normalized PPL in Eq. 5, the redder the color, the better the performance. Subfigure (a) presents the evaluation results across different calibration sets. In (b), we highlight calibration sets from (a) that lead to strong performance on other evaluation tasks, and (c) correspondingly shows that certain evaluation tasks benefit significantly from specific calibration sets. (d) isolates these tasks for a clearer view of the emerging patterns.

3.1 Domain Affinity Mining

We begin with aligning the expert reconstruction with structured pruning for LLMs, applying the same pruning ratio to each FFN. An effective pruning method is FLAP (An et al., 2024), which measures channel importance by assessing the fluctuation across the calibration dataset in the hidden state features. From the observations of performance gaps in FLAP across different calibration datasets (Appendix A.2), we hypothesize that *fluctuation is sensitive to the domain of the calibration sets, resulting in diversity of pruning outcomes*. To verify this, we conduct domain affinity mining on LLaMA2-7B using 24 datasets from various domains (Appendix D.1).

For each calibration task t_i in the set of tasks $\mathcal{T}$, given N samples, we calculate the pruning mask $\mathbf{M}_{t_i}^{\ell}$ base on the fluctuation variance. The importance score for each weight group $\mathbf{W}_{:,j}^{\ell}$ is computed as follows:

$$\mathbf{S}_{:,j,}^{\ell} = \frac{1}{N-1}\sum_{n=1}^{N}(\mathbf{X}_{n,j,:,t_i}^{\ell} - \overline{\mathbf{X}}_{:,j,:,t_i}^{\ell})^2 \cdot ||\mathbf{W}_{:,j}^{\ell}||_2^2,$$

(4)

where $\mathbf{X}_{n,j,:,t_i}^{\ell}$ and $\overline{\mathbf{X}}_{:,j,:,t_i}^{\ell}$ represent the hidden state feature for the n-th sample and the baseline fluctuation in the j-th channel. After pruning, channels with the highest scores are retrained. Importantly, our pruning targets the intermediate dimensions of each FFN, focusing on reconstructing the

MoE layer, as discussed in Section 2.2.

To evaluate the pruning results, we measure the normalized perplexity (PPL) on evaluation tasks for models pruned using various calibration task sets. Given the variability of PPL values across different evaluation tasks, we introduce the definition of normalized PPL as follows:

$$\text{norm}(p)_{i,j} = \frac{\min(p)_{:,j}}{p_{i,j}},$$

(5)

where $p_{i,j}$ denotes the original PPL of the i-th calibration datasets on the j-th evaluation datasets, and $\min(p)_{:,j}$ denotes the minimum PPL on the j-th evaluation dataset.

Figure 2 shows evaluation results after pruning across all calibration sets with the FFN pruning ratio of 75%. We identify key phenomena that shed light on the effects of pruning across different calibration sets:

(1) Calibration sets from similar domains exhibit strong correlations. For mathematic QA tasks (MathQA and GSM8K), the model pruned with one calibration set demonstrates strong generalization on the other task, indicating that pruning can effectively capture domain-specific expertise.

(2) Affinity is more sensitive to data than task types. The affinity between QNLI and other NLI tasks (e.g., ANLI and MNLI) is not as strong as that with SQuAD2, which stems from both QNLI
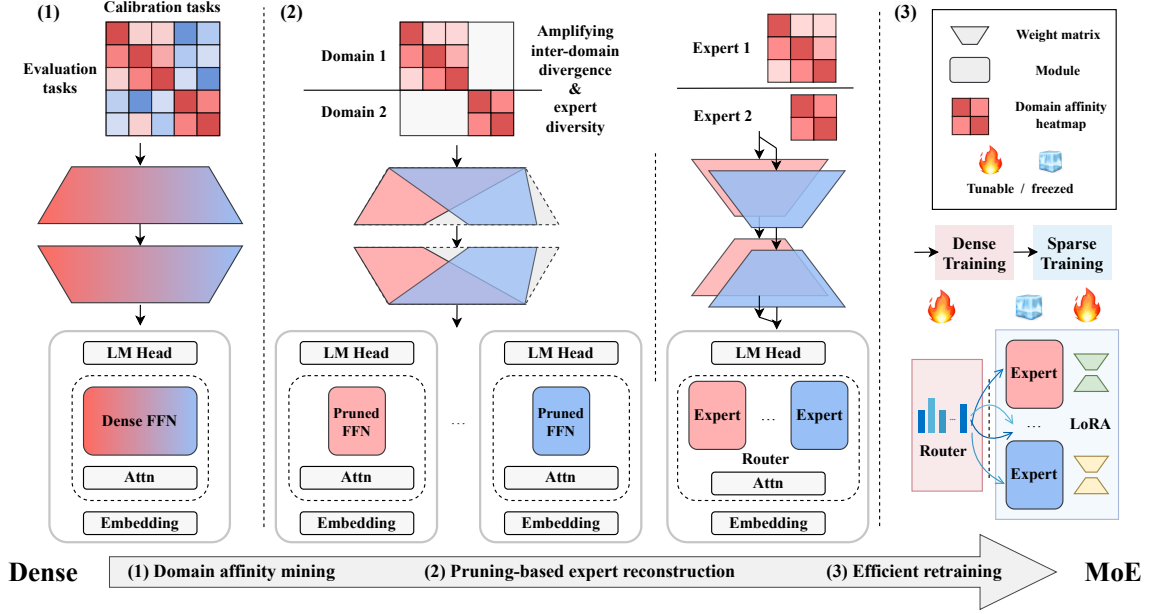
19377

Figure 3: Demonstration of diversity-enhanced reconstruction of LLMs from dense to MoE. In each lane, the upper part exhibits the expert division with the domain affinity heatmap. The reconstruction of FFNs is concretely illustrated in the middle part. And, the modifications to the overall LLM architecture are depicted in the lower part.

and SQuAD2 being derived from SQuAD1.1 data (Rajpurkar et al., 2018).

(3) C4 exhibits generalization as a calibration set. C4, a cleaned web crawl corpus (Raffel et al., 2020), shows broad generalization when used as a calibration set, reflecting its diverse linguistic features and robustness.

We further perform domain affinity mining on various LLM backbones (Appendix D.2). These findings back up our hypothesis and motivate pruning with affinity domains to achieve diversity-enhanced MoE reconstruction.

3.2 Pruning-Based Expert Reconstruction

In this section, we describe how to conduct pruning-based reconstruction of domain-specific experts upon the above domain affinity mining. The reconstruction process is defined as follows: Given a dense LLM $\mathcal{M}$, obtain a converted sparse MoE LLM $\tilde{\mathcal{M}}$, with FFN modules replaced by MoE layers. Each MoE layer is composed of a router and multiple experts, which are pruned from the original dense FFNs.

We begin by calculating the domain correlations. The correlation between each pair of calibration sets is computed with the Pearson correlation coefficient:

$$\text{corr}(d^1, d^2) = \frac{\text{cov}(d^1, d^2)}{\sigma(d^1)\sigma(d^2)}, \qquad (6)$$

where $\text{cov}(d^1, d^2)$ is the covariance between d^1 and d^2, and $\sigma(d^1)$ and $\sigma(d^2)$ are the standard deviations of d^1 and d^2, respectively. The correlation value measures the linear correlation between two datasets, with values ranging from -1 to 1.

We use hierarchical clustering to group the calibration datasets into 8 clusters based on Eq. 6, and construct final calibration datasets for pruning domain-specific experts by uniformly mixing data within each cluster. For each cluster, we prune an LLM on intermediate dimensions of the FFNs on it. The pruned FFNs are then restructured as experts, with each layer guided by a randomly initialized noisy router to form the MoE layers, replacing the original FFNs.

3.3 Efficient Retraining

Since the LLM is partially reorganized, it is essential to retrain the model on a large-scale dataset to recover its general performance. Considering the sparse gradients and non-differentiability raised by the top-k operation in routers, we perform a two-stage retraining: **(1) dense training** for routers using a tiny amount of data, with all experts being activated, and **(2) sparse training** for experts

and normalization modules on a large dataset with PEFT, only activating the correlated experts.

During the dense training stage, we introduce a temperature coefficient t to routers as in Eq.7. This operation aims to align the routing behavior better with sparse inference. Further clarifications of the two-stage retraining design are in Appendix B.2 and B.3.

$$w_i(x) = \text{Softmax}\left(\frac{z(x)}{t}\right)_i. \quad (7)$$

We use Low-Rank Adaptation (LoRA) (Hu et al., 2022) to retrain the experts, with fully fine-tuning the routers and the normalization modules. There is no need to update the parameters of MHA and other modules, as verified by experiments in Section 4.4.2.

3.4 Overview of DIVE

Generally, DIVE is designed with the workflow described in Figure 3 and Algorithm 1. With the original dense LLM $\mathcal{M}$ and target expert number N, $\mathcal{M}$ is first pruned and evaluated on different pairs of tasks. Then, the clustering-based data selection is performed with the distance metric of task correlation. With the grouped datasets $\mathcal{G}$, $\mathcal{M}$ is pruned with calibration data sampled from $\mathcal{G}$ into N dense LLM $\{\mathcal{M}'_1, \cdots, \mathcal{M}'_N\}$. As $\mathcal{M}'_N$ are with the same parameters with the only exception of FFN modules, they can be merged into an MoE architecture $\tilde{\mathcal{M}}$. Finally, a two-stage retraining on $\tilde{\mathcal{M}}$ FFN modules with PEFT recovers the performance and yields the final MoE LLM $\tilde{\mathcal{M}}^*$.

Algorithm 1 Workflow of DIVE

Input: dense LLM $\mathcal{M}$, expert number N, tasks $\mathcal{T}$
Output: MoE LLM $\tilde{\mathcal{M}}^*$
1: **for** $(t_i, t_j) \in \mathcal{T} \times \mathcal{T}$ **do**
2:     $P_{i,j} \leftarrow \text{PPL}(\text{PRUNE}(\mathcal{M}, t_i), t_j)$
3: **end for**
4: $\mathcal{G} \leftarrow \text{CLUSTER}(\text{NORM}(P), N, \text{dist} = corr)$
5: **for** $i \leftarrow 1$ to N **do**
6:     $\mathcal{C}_i \leftarrow \text{SAMPLE}(\mathcal{G}_i)$
7:     $\mathcal{M}'_i \leftarrow \text{PRUNE}(\mathcal{M}, \mathcal{C}_i)$
8: **end for**
9: $\mathcal{M}' \leftarrow \text{RECONSTRUCTION}(\{\mathcal{M}'_1, \cdots, \mathcal{M}'_N\})$
10: $\text{FREEZE}(\mathcal{M}')$ and $\text{UNFREEZE}(\mathcal{M}'.Router)$
11: $\tilde{\mathcal{M}} \leftarrow \text{TRAIN}(\mathcal{M}'.Router)$
12: $\text{FREEZE}(\tilde{\mathcal{M}})$ and $\text{UNFREEZE}(\tilde{\mathcal{M}}.LN)$
13: $\tilde{\mathcal{M}}^* \leftarrow \text{TRAIN}(\tilde{\mathcal{M}}.LN \cup \tilde{\mathcal{M}}.FFN.LoRA)$
14: **return** $\tilde{\mathcal{M}}^*$

Method	FFN Size	WikiText2 1024	WikiText2 2048	LAMBADA
LLM-Pruner	50%	17.59	15.62	56.66
FLAP	50%	14.51	12.84	33.22
LLaMA-MoE 1/8	50% × 1	19.57	17.26	87.27
DIVE 1/8 (Ours)	50% × 1	**13.52**	**11.95**	**24.84**
LLaMA-MoE 2/8	25% × 2	20.78	18.46	121.00
DIVE 2/8 (Ours)	25% × 2	**16.58**	**14.67**	**51.95**

Table 1: Perplexity on WikiText2 and LAMBADA language modeling. WikiText2 commonly exceeds the maximum length of the model, and the length of LAMBADA is below 1024. Best results are in **bold**.

4 Experiments

4.1 Experimental Settings

Baselines. We implement experiments on TinyLlama-1.1B (Zhang et al., 2024), a Llama-style LLM with dense SwiGLU FFNs (Shazeer, 2020). We compare DIVE with existing structured pruning and MoE reconstruction methods for LLMs, including LLM-Pruner (Ma et al., 2023), FLAP (An et al., 2024), and LLaMA-MoE (Zhu et al., 2024). We set the same size of 50% of a single original FFN for all methods and apply identical retraining procedures. FLAP is implemented on a mixed calibration set of all 24 datasets. Additionally, to make LLaMA-MoE adaptable to reconstruct MoE models with various sizes, we make adjustments to its expert construction method, as detailed in Appendix A.

Retraining Setup. We use the SlimPajama dataset for retraining, which is a cleaned and deduplicated version of the RedPajama dataset, containing 627B tokens of training data (Soboleva et al., 2023). Experiments of MoE reconstruction methods are conducted on randomly sampled 0.5B tokens for dense training of routers and 5B tokens for sparse training, with a sequence length of 1024, if not specified. Experiments of pruning methods are aligned with the sparse training stage. We set the temperature coefficient to 0.05 for DIVE 1/8 and 0.5 for DIVE 2/8. Detailed hyper-parameters are provided in Appendix B.1.

Evaluation Metrics. The language modeling ability is evaluated with perplexity on Wiki-Text2 (Merity et al., 2017) and LAMBADA (Paperno et al., 2016). We use lm-evaluation-harness (Gao et al., 2023) to evaluate the performance on downstream tasks. Following LLaMA-MoE (Zhu et al., 2024) and FLAP (An et al., 2024),

Method	FFN Size	SciQ	PIQA	WinoGrande	ARC-e	ARC-c (25)	MathQA
TinyLlama-1.1B	100%	89.30	72.60	59.43	61.66	35.67	24.32
LLM-Pruner	50%	79.20	63.17	51.46	42.85	22.87	21.68
FLAP	50%	80.50	66.10	**55.72**	47.69	24.49	**22.61**
LLaMA-MoE 1/8	50% × 1	76.30	61.10	52.17	43.18	22.70	21.84
DIVE 1/8 (Ours)	50% × 1	**83.00**	**68.12**	52.72	**51.77**	**24.91**	22.45
LLaMA-MoE 2/8	25% × 2	77.10	60.66	**54.93**	40.70	21.84	22.18
DIVE 2/8 (Ours)	25% × 2	**81.10**	**64.53**	52.49	**46.68**	**22.70**	**23.02**

Method	FFN Size	HellaSwag (10)	LogiQA	BoolQ	OBQA	MMLU (5)	Average
TinyLlama-1.1B	100%	46.45	21.51	55.99	25.20	26.79	47.18
LLM-Pruner	50%	33.17	**21.04**	59.02	16.20	24.61	39.57
FLAP	50%	35.37	20.28	**61.86**	17.60	23.39	41.42
LLaMA-MoE 1/8	50% × 1	33.24	18.74	61.71	16.60	25.16	39.34
DIVE 1/8 (Ours)	50% × 1	**37.09**	19.35	58.81	**20.20**	**25.40**	**42.17**
LLaMA-MoE 2/8	25% × 2	31.93	19.20	**61.93**	16.20	25.00	39.24
DIVE 2/8 (Ours)	25% × 2	**32.86**	**22.12**	59.88	**18.00**	**26.78**	**40.92**

Table 2: Few-shot performance of pruning and MoE reconstruction methods on TinyLlama-1.1B, measured by accuracy on downstream tasks. The FFN size with a multiplier of ×1 or ×2 indicates the number of experts, while dense models do not use the multiplier. The best results are highlighted in **bold**.

our evaluation covers 11 commonly used benchmarks for LLMs: 0-shot of SciQ (Welbl et al., 2017), PIQA (Bisk et al., 2020), WinoGrande (Sakaguchi et al., 2020), ARC-e (Clark et al., 2018), MathQA (Amini et al., 2019), LogiQA (Liu et al., 2020), BoolQ (Clark et al., 2019) and OBQA (Mihaylov et al., 2018), 25-shot ARC-c (Clark et al., 2018), 10-shot HellaSwag (Zellers et al., 2019) and 5-shot MMLU (Hendrycks et al., 2021).

4.2 Main Results

4.2.1 Language Modeling

Table 1 presents the language modeling results of DIVE and other methods after retraining. Models are grouped by single FFN size: 50% × 1 and 25% × 2. The perplexity scores demonstrate that DIVE 1/8 significantly outperforms other methods with the same number of activated parameters. In particular, compared to the second-best results, DIVE 1/8 reduces the perplexity by 8.38 on LAMBADA, and by 0.99 and 0.89 on WikiText2 with sequence lengths of 1024 and 2048, respectively.

We further analyze the performance of MoE reconstruction methods during retraining, focusing on the perplexity for WikiText2 with the sequence length of 1024 and the training loss. As shown in Figure 4, our models consistently outperform LLaMA-MoE counterparts, during retraining with 1B to 5B tokens. Notably, DIVE 1/8 trained on merely 1B tokens surpasses FLAP trained on 5B tokens, highlighting the efficiency of our method.
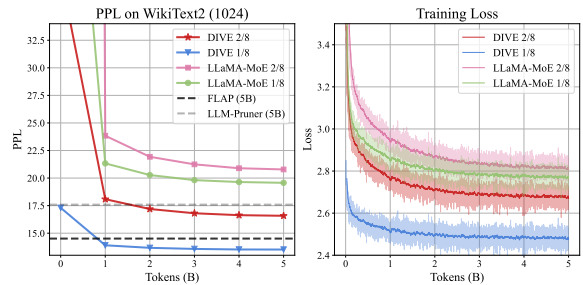
Figure 4: Perplexity curves of DIVE and the baselines, consisting of perplexity on WikiText2 (left) and training loss for DIVE and LLaMA-MoE (right). Loss curves are smoothed with a 0.05B token window.

4.2.2 Downstream Tasks

Table 2 presents the few-shot performance of accuracy across 11 downstream tasks. DIVE 1/8 achieves the best average performance among 50% × 1 models, excelling in SciQ, PIQA, ARC-e, ARC-c, HellaSwag, OBQA, and MMLU. It improves accuracy on BoolQ by 3.55% compared to the original model. And while DIVE 2/8 slightly lags behind DIVE 1/8 and FLAP on average, it outperforms LLM-Pruner and LLaMA-MoE 2/8 by 1.35% and 1.68%, despite LLM-Pruner benefiting from a more refined pruning ratio.

To illustrate the changes of model capabilities during retraining process, we present performance curves on ARC-e and PIQA in Figure 6. As the results on these benchmarks gradually grow, DIVE 1/8 consistently outperforms all other methods re-

Figure 5: Routing distribution of DIVE 1/8 (left) and DIVE 2/8 (right), shown as activation ratios of the corresponding experts across all layers. The indexes of experts correspond to the domain clustering results in Appendix D.2. Note that, since DIVE achieves expert initialization through pruning, the IDs of domain-specific experts remain consistent across layers.
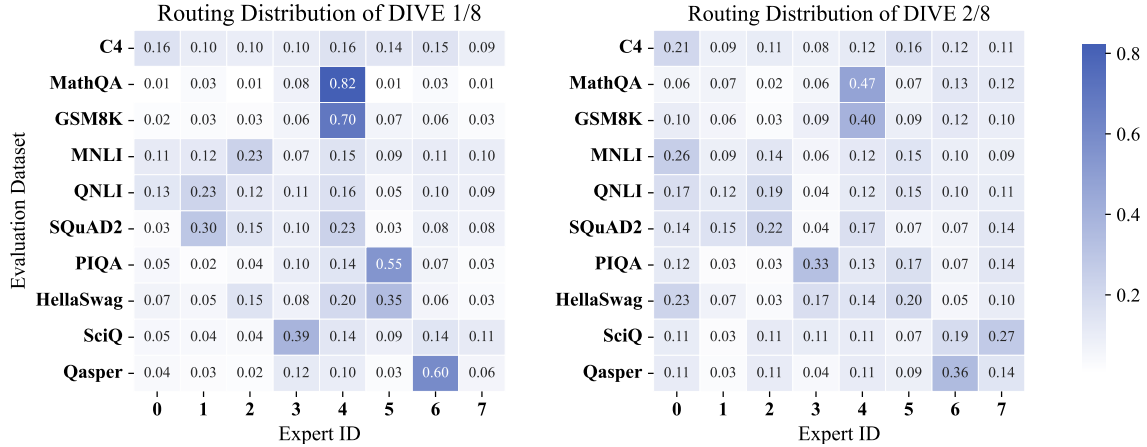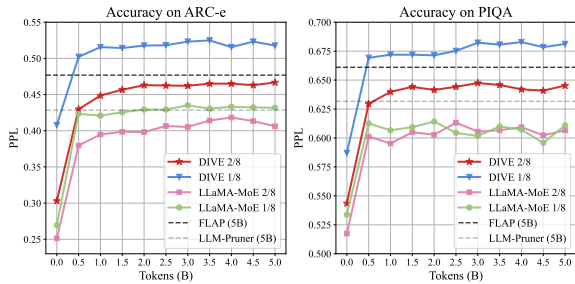

Figure 6: Performance curves of DIVE and baselines on 0-shot ARC-e (left) and PIQA (right). Solid lines indicate the performance changes of models during retraining to 5B tokens, and dashed lines represent the performance of models retrained on 5B tokens.

trained on 5B tokens. Meanwhile, compared with LLaMA-MoE models, both DIVE 1/8 and DIVE 2/8 demonstrate superior performance. We further retrain the MoE models on 15B tokens to validate the effectiveness of our approach (Appendix C).

4.3 Diversity Analysis

To assess the diversity of DIVE models, we analyze routing distributions across 24 pruning datasets, with representative results shown in Figure 5. The values in heatmaps represent activation ratios, calculated as the frequency of tokens routed to a specific expert index across all layers, normalized by the total token activations in each evaluation set. Notably, since our MoE layers are derived through pruning, the expert indices in each layer are fully aligned with the domain clusters.

Based on the analysis, we find several obser-

vations aligned with the phenomena discussed in Section 3.1:

1. After dense training of routers, most task tokens (such as MathQA, GSM8K, PIQA, and HellaSwag) are correctly routed to their corresponding experts.

2. Certain tasks (QNLI and SQuAD2) exhibit sensitivity to data structure, activating Expert 1 due to shared features, while Expert 2 is mainly triggered by other NLI tasks.

3. Tokens in C4 demonstrate strong generality, activating experts in a balanced manner with nearly uniform distribution.

4. Compared to DIVE 1/8, DIVE 2/8 shows smoother activation ratios across experts due to the increased number of activated experts, aligning with expectations.

These findings confirm the success of MoE diversity and routing allocation. Full heatmaps along with a case study are provided in Appendix E.

4.4 Ablation Studies

4.4.1 Domain Affinity Mining

To validate the effectiveness of our domain affinity mining (D.A.M) approach, which enhances the diversity of MoE models, we compare DIVE reconstructed with D.A.M against models using randomly selected calibration data from all 24 datasets. We categorize the downstream tasks into in-distribution (ID) tasks and out-of-distribution

	DIVE 2/8	- D.A.M.
WikiText2 ↓	18.09	20.02 (+1.93)
LAMBADA ↓	63.45	76.03 (+12.58)
ID tasks ↑	47.88	47.08 (-0.80)
OOD tasks ↑	27.67	26.15 (-1.52)

Table 3: Performance of DIVE 2/8 with and without D.A.M, retrained on 1B tokens. Tasks marked with downward arrows (WikiText2 and LAMBADA) are measured by perplexity and inversely correlated with performance, others (ID and OOD tasks) are downstream tasks measured by accuracy.
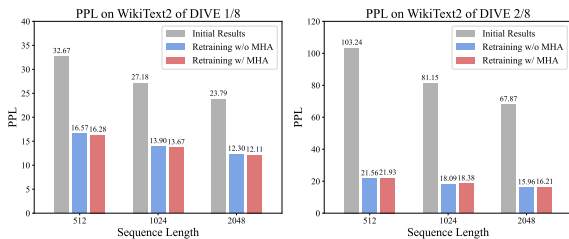

Figure 7: Comparison of DIVE 1/8 (left) and DIVE 2/8 (right) perplexity results on WikiText2, retrained on 1B tokens with and without MHA modules.

(OOD) tasks, based on whether they belong to the 24 datasets used for pruning.

As the loss and performance remain relatively stable during training from 1B to 5B tokens in Figure 4 and 6, we conduct our ablation studies with 1B tokens to balance the computational efficiency and the validity of results. As shown in Table 3, DIVE 2/8 with D.A.M reduces the perplexity on WikiText2 with the sequence length of 1024 and LAMBADA by 1.93 and 12.58, and achieves accuracy improvements of 0.80% and 1.52% over the model without D.A.M. This underscores the critical role of domain affinity mining in the reconstruction process and highlights the importance of diversity in improving model performance. Detailed information of task categories and full results are provided in Appendix F.

4.4.2 Retraining Targets

In this section, we explain the rationale behind our efficient retraining targets. We focus on perplexity results on WikiText2 across various sequence lengths to evaluate general performance.

Figure 7 shows the performance of the initial reconstructed MoE models and models retrained on 1B tokens with and without MHA modules. For DIVE 1/8, cutting the MHAs out of the tunable parameters only leads to less than a 0.29 degra-

dation in performance. For DIVE 2/8, the model retrained without MHA surprisingly outperforms the one with MHA by up to 0.37. These results support the retraining targets of our method, indicating the MHA modules are unnecessary to be retrained.

5 Related Work

5.1 Dense Compression

Statically pruning LLMs to an optimal size enables efficient deployment (Xia et al., 2022; Shen et al., 2022), while posing challenges due to performance impacts and the need for retraining adjustments. Gradient-based methods have been extensively studied, such as LLM-Pruner (Ma et al., 2023) using Taylor expansion derived from optimal brain damage (OBD) (LeCun et al., 1989), and Sheared-LLaMA (Xia et al., 2024), employing parameterized masks. In contrast, gradient-free methods like TransAct (Shen et al., 2024), typically use activation magnitude for co-designing pruning metrics and architectures, enhancing general capabilities but reducing internal diversity. FLAP (An et al., 2024) emphasizes domain-specific diversity, as discussed in Section 3.1.

5.2 Sparse Acceleration

Beyond static compression, dynamic computation methods like early exiting (Del Corro et al., 2023) and layer skipping (Raposo et al., 2024) accelerate LLM inference by reducing model depth, though they may increase memory demands. To address this, sparsity-aware offloading techniques (e.g., LLM-in-a-flash (Alizadeh et al., 2024) and PowerInfer (Song et al., 2023)) exploit sparse activation patterns for efficient computation. Given the successful deployment and acceleration of the Mixtral MoE LLM (Xue et al., 2024), DIVE appears highly compatible with these offloading strategies.

5.3 From Dense to Sparse

While dense LLMs dominate the landscape, agilely derive an MoE LLM from existing dense LLMs minimize the sunk cost. MoEfication (Zhang et al., 2022) introduces expert construction strategies for T5, including random splitting, parameter clustering, and co-activation graph methods. On decoder-only architectures, sparse up-cycling (Komatsuzaki et al., 2023) initializes experts using FFN modules, as demonstrated by Skywork-MoE (Wei et al., 2024), which expands a 13B dense LLM into a 146B MoE model with 16 diverse experts. Expert

diversity proves crucial, as diverse experts outperform duplicated ones despite higher early-stage training loss. LLaMA-MoE (Zhu et al., 2024) advocates for a random split approach which divide the dense FFNs into non-overlapping experts.

6 Conclusion

In this work, we propose DIVE, a diversity-enhanced MoE reconstruction method which significantly reduces the training cost. The three-phase framework of DIVE consists of domain affinity mining, pruning-based expert reconstruction, and efficient retraining, enabling a seamless and effective reconstruction from dense to MoE LLMs. Experiments on language modeling and downstream tasks validate the efficacy of our approach with perplexity reduction and improvements on task accuracy. The analysis of diversity validates the correlation between pruning and MoE expert diversity in enhancing overall performance, while ablation studies highlight the robustness of our method.

Limitations

Although DIVE demonstrates generalization across different models and sizes, the models used in experiments are no larger than 7B due to resource constraints. Future efforts could focus on scaling up model sizes and exploring broader domains to further validate the effectiveness of DIVE, as splitting-based methods and up-cycling have proven effective in larger models.

Ethics Statement

Our work focuses on efficiently reconstructing dense LLMs into MoE architectures while enhancing the diversity of the reconstructed models. The datasets used in our study are publicly available and widely recognized in the research community, sourced from open-source repositories. No proprietary or sensitive data were used during either training or evaluation. Moreover, our methods are purely algorithmic and do not promote or amplify harmful biases by design.

Acknowledgement

References

AF Agarap. 2018. Deep learning using rectified linear units (relu). *ArXiv preprint*, abs/1803.08375.

Keivan Alizadeh, Seyed Iman Mirzadeh, Dmitry Belenko, S. Khatamifard, Minsik Cho, Carlo C Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. 2024. LLM in a flash: Efficient large language model inference with limited memory. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12562–12584, Bangkok, Thailand. Association for Computational Linguistics.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. 2024. Fluctuation-based adaptive structured pruning for large language models. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 10865–10873. AAAI Press.

Yonatan Bisk, Rowan Zellers, Ronan LeBras, 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 *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro

Nakano, et al. 2021. Training verifiers to solve math word problems. *ArXiv preprint*, abs/2110.14168.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online. Association for Computational Linguistics.

Luciano Del Corro, Allie Del Giorno, Sahaj Agarwal, Bin Yu, Ahmed Awadallah, and Subhabrata Mukherjee. 2023. Skipdecode: Autoregressive skip decoding with batching and caching for efficient llm inference. *ArXiv preprint*, abs/2307.02628.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

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. 2023. A framework for few-shot language model evaluation.

Naibin Gu, Peng Fu, Xiyu Liu, Bowen Shen, Zheng Lin, and Weiping Wang. 2024. Light-PEFT: Lightening parameter-efficient fine-tuning via early pruning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 7528–7541, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

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*, abs/2401.04088.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. 2023. Sparse upcycling: Training mixture-of-experts from dense checkpoints. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

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

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. Gshard: Scaling giant models with conditional computation and automatic sharding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3622–3628. ijcai.org.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

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

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. 2024. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *ArXiv preprint*, abs/2404.02258.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 3505–3506. ACM.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. 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 8732–8740. AAAI Press.

Noam Shazeer. 2020. Glu variants improve transformer. *ArXiv preprint*, abs/2002.05202.

Bowen Shen, Zheng Lin, Yuanxin Liu, Zhengxiao Liu, Lei Wang, and Weiping Wang. 2022. COST-EFF: Collaborative optimization of spatial and temporal efficiency with slenderized multi-exit language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1719–1730, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Bowen Shen, Zheng Lin, Daren Zha, Wei Liu, Jian Luan, Bin Wang, and Weiping Wang. 2024. Pruning large language models to intra-module low-rank architecture with transitional activations. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9781–9793, Bangkok, Thailand. Association for Computational Linguistics.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. 2023. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yixin Song, Zeyu Mi, Haotong Xie, and Haibo Chen. 2023. Powerinfer: Fast large language model serving with a consumer-grade gpu. *ArXiv preprint*, abs/2312.12456.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059, Dublin, Ireland. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Tianwen Wei, Bo Zhu, Liang Zhao, Cheng Cheng, Biye Li, Weiwei Lü, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Liang Zeng, et al. 2024. Skywork-moe: A deep dive into training techniques for mixture-of-experts language models. *ArXiv preprint*, abs/2406.06563.

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*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

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

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.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1513–1528, Dublin, Ireland. Association for Computational Linguistics.

Zhenliang Xue, Yixin Song, Zeyu Mi, Le Chen, Yubin Xia, and Haibo Chen. 2024. Powerinfer-2: Fast large language model inference on a smartphone. *ArXiv preprint*, abs/2406.06282.

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*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *ArXiv preprint*, abs/2401.02385.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *ArXiv preprint*, abs/1810.12885.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

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*, pages 877–890, Dublin, Ireland. 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. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15913–15923, Miami, Florida, USA. Association for Computational Linguistics.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2023. A survey on model compression for large language models. *ArXiv preprint*, abs/2308.07633.

A Baseline Details

A.1 Expert Reconstruction of LLaMA-MoE

LLaMA-MoE (Zhu et al., 2024) presents two groups of expert reconstruction methods: Neuron-Independent and Neuron-Sharing, with each category containing two distinct methods. We initially consider selecting the best method from each group, specifically Independent$_{\text{Random}}$ and Sharing$_{\text{Inter}}$, based on their reported performance. However, due to the restriction of Sharing$_{\text{Inter}}$, which requires setting aside neurons shared by most experts as independent residual blocks, it is not suitable for reconstructing 1/8 MoE models. So we adapt Independent$_{\text{Random}}$, which is originally supports reconstructing models with a fixed total number of parameters. We make revision to allow it to be adaptive to varying FFN sizes and to enable a fair comparison.

We redefine expert reconstruction as the task of partitioning intermediate neurons into equal-sized subsets. Given a universal set U that contains the indices of all intermediate neurons $\{1, 2, \ldots, d_h\}$, we randomly divide U into equal sized n subsets $S_1, S_2, \ldots, S_n$, potentially constructing experts of varying sizes as:

$$\bigcup_{i=1}^{n} S_i = U. \qquad (8)$$

In this context, we omit the original condition $\bigcap_{i=1}^{n} S_i = \varnothing$ from LLaMA-MoE.

A.2 Discussion on FLAP

For the structured pruning method in FLAP (An et al., 2024), given a calibration set C and N samples, we first compute the baseline fluctuation of the j-th channel in the input feature of layer ℓ with the average sample fluctuation variance:

$$\overline{\mathbf{X}}_{:,j,:,C}^{\ell} = \frac{1}{NL} \sum_{n=1}^{N} \sum_{k=1}^{L} \mathbf{X}_{n,j,k,C}^{\ell}. \qquad (9)$$

Then, we calculate the fluctuation variance for each input feature in layer ℓ, which gives the importance score $\mathbf{S}_{:,j,C}^{\ell}$ for the weight group $\mathbf{W}_{:,j}^{\ell}$:

$$\mathbf{S}_{:,j,C}^{\ell} = \frac{1}{N-1} \sum_{n=1}^{N} (\mathbf{X}_{n,j,:,C}^{\ell} - \overline{\mathbf{X}}_{:,j,:,C}^{\ell})^2 \cdot \|\mathbf{W}_{:,j}^{\ell}\|_2^2. \qquad (10)$$

The pruning mask $\mathbf{M}_C^{\ell}$ is determined using these importance scores, by retaining the channels with the highest scores. The output of each pruned layer can be formulated as:

$$\mathbf{W}_C^{\ell} \mathbf{X}_C^{\ell} \approx (\mathbf{M}_C^{\ell} \odot \mathbf{W}^{\ell}) \mathbf{X}_C^{\ell} + \mathbf{B}_{0,C}^{\ell}, \qquad (11)$$

with $\mathbf{B}_{0,C}^{\ell} = \mathbf{W}^{\ell}((1 - \mathbf{M}_C^{\ell}) \odot \overline{\mathbf{X}}_C^{\ell})$ representing the bias of the relevant linear layer.

For all pruning processes, we set the calibration size to 1024 and the sample length to 256. To assess the impact of the calibration set, we conduct an experiment comparing language modeling and downstream task performance using FLAP pruning on C4 and a mixed set of 24 datasets, alongside our DIVE 1/8 with only routers trained.

As shown in Table 4, FLAP models pruned on different calibration sets show inconsistent performance across tasks. Specifically, FLAP (C4) lags behind FLAP (Mixed) by up to 9.90 in perplexity of LAMBADA and WikiText2 (sequence length 1024), but outperforms it by 1.39% on the average results of downstream tasks. In contrast, DIVE 1/8 achieves the best performance in both language modeling (perplexity) and downstream task (average accuracy), benefiting from the diversity of experts. This highlights the sensitivity of existing pruning methods and underscores the effectiveness of our approach.

A.3 Discussion on LLaMA-MoE

For Neuron-Sharing methods of LLaMA-MoE, which rely similarly on a pruning-based approach to construct experts and clusters data via sentence embeddings, they prune the model based solely on clustering results from the same dataset, without acknowledging the impact of different datasets. This limitation mirrors the issue we discuss with FLAP above, which can be summarized as a lack of diversity.

In contrast, our method focuses on the insight of the diversity of pruning outcomes, achieving better performance on various tasks. And we explore to reconstruct 1/8 MoE models and adjust the retraining phase to resolve the sparse gradients, especially for the top-1 routers.

B Retraining Details

B.1 Hyper-parameters

The training parameters for the sparse training stage are presented in Table 5. Our experiments are conducted on 4 A800 (80G) GPUs with a global batch size of 512, utilizing BFloat16 precision. We

Method	WikiText2 ↓	LAMBADA ↓	SciQ	PIQA	WinoGrande	ARC-e	ARC-c (25)
FLAP (C4)	26.32	70.27	<u>84.30</u>	60.94	<u>53.67</u>	40.40	23.12
FLAP (Mixed)	<u>22.67</u>	<u>60.37</u>	**85.20**	<u>62.40</u>	**53.99**	<u>44.28</u>	<u>23.21</u>
DIVE 1/8	**17.30**	**42.55**	83.60	**65.72**	52.96	**48.57**	**25.09**

Method	MathQA	HellaSwag (10)	LogiQA	BoolQ	OBQA	MMLU (5)	Average
FLAP (C4)	22.61	<u>33.66</u>	20.28	**62.29**	<u>17.00</u>	<u>23.01</u>	<u>40.12</u>
FLAP (Mixed)	**23.62**	32.83	<u>20.89</u>	39.91	16.80	22.92	38.73
DIVE 1/8	<u>23.15</u>	**35.71**	**22.43**	<u>51.50</u>	**17.60**	**26.16**	**41.14**

Table 4: Performance of FLAP pruned on C4 and mixed data, and DIVE 1/8 reconstructed without sparse training. Tasks marked with downward arrows (WikiText2 and LAMBADA) are evaluated by perplexity, which is inversely correlated with performance. Others are downstream tasks measured by accuracy, with averages shown in the last column. The best results are in **bold**, and the second-best results are <u>underlined</u>.

Training Parameter	Value
# GPUs	4
Sequence length	1024
Data type	Bfloat16
Learning rate	1e-4
Learning rate scheduler	Cosine
Cosine minimum ratio	0.1
Warmup ratio	0.03
Optimizer	AdamW
Global batch size	512
LoRA rank	8
LoRA alpha	16
LoRA dropout	0.1
DeepSpeed	Zero-2

Table 5: Training parameters used in the experiments.

Method	t
DIVE 1-8	0.05
DIVE 2-8	0.5
LLaMA-MoE 1-8	0.01
LLaMA-MoE 2-8	0.1

Table 6: Temperature coefficient(t) for MoE reconstruction methods used in dense training.

better optimize the different reconstructed modules (routers and experts).

1. **Dense training (for routers)**: Due to the retained abilities of reconstructed experts, the issues of diversity and generalization of pruning can be addressed by focusing sorely on dense router training, as discussed in Appendix A.2. Activating all experts can deal with the sparse gradients and non-differentiability of the top-k operation in routers.

2. **Sparse training (for experts)**: To further enhance the model's capabilities, we focus on training the experts and the associated normalization modules. With reasonable routing allocations already established, only the appropriate expert need to be activated during each forward and backward pass.

Based on the considerations above, we design a two-stage retraining approach of dense-then-sparse training, which is more suitable for our framework than an end-to-end training process.

B.3 Temperature Coefficient

In this section, we explain the rationale behind our temperature coefficient settings. The original

set the warmup ratio to 3% and employ a cosine learning rate scheduler, decaying the learning rate from 1e-4 to 1e-5. Each model is retrained on 5B tokens. Our implementation is based on Huggingface Transformers (Wolf et al., 2020) and DeepSpeed (Rasley et al., 2020).

For the dense training stage of MoE reconstruction methods, we specifically set a constant learning rate of 1e-4, with a global batch size of 256. We adjust each temperature coefficient (t) based on their validation loss, as shown in Table 6. For all MoE models, we do not adopt the re-scaling factor in LLaMA-MoE.

B.2 Clarification of Two-Stage Retraining

As our models are reconstructed based on pruning results from pre-trained LLMs and the abilities of experts are partially retained, it is straightforward to separate the retraining process into two stages to

Figure 8: Training and validation loss for (a) DIVE 1/8 and DIVE 2/8 with and without temperature coefficient (t), trained on 0 to 0.5B tokens, and (b) DIVE 1/8 with t, trained on 0 to 1B tokens.

Method	WikiText2 ↓	LAMBADA ↓	SciQ	PIQA	WinoGrande	ARC-e	ARC-c (25)
LLaMA-MoE 1/8	18.60	73.96	77.40	61.15	52.81	43.56	22.10
DIVE 1/8	**13.23**	**24.15**	**83.00**	**68.01**	<u>54.70</u>	**52.53**	**24.40**
LLaMA-MoE 2/8	19.45	96.80	77.90	62.84	53.51	42.89	22.01
DIVE 2/8	<u>15.87</u>	<u>47.68</u>	<u>81.50</u>	<u>64.96</u>	**54.78**	<u>47.10</u>	<u>23.12</u>

Method	MathQA	HellaSwag (10)	LogiQA	BoolQ	OBQA	MMLU (5)	Average
LLaMA-MoE 1/8	21.07	<u>33.73</u>	20.58	<u>61.71</u>	16.60	25.47	39.65
DIVE 1/8	<u>22.55</u>	**37.17**	19.35	59.88	**20.20**	<u>25.88</u>	**42.52**
LLaMA-MoE 2/8	**23.08**	32.23	<u>21.51</u>	**61.77**	14.60	25.14	39.77
DIVE 2/8	22.58	33.27	**21.97**	58.47	<u>18.20</u>	**26.33**	<u>41.12</u>

Table 7: Performance of LLaMA-MoE and DIVE models further retrained on 15B tokens. Tasks marked with downward arrows (WikiText2 and LAMBADA) are evaluated by perplexity with a sequence length of 1024, and others are downstream tasks measured by accuracy. The last column shows the average accuracy of all downstream tasks. The best results are in **bold**, and the second-best results are <u>underlined</u>.

routing logits of a sparse MoE are defined as:

$$w_i(x) = \text{Softmax}(\text{TopK}(z(x)))_i, \quad (12)$$

which leads to a gap between the propagation during dense training and the sparse inference. To address this, we introduce a temperature coefficient t to sharpen the logits and approximate the top-k operation, ensuring alignment between the routing behaviors. The router output is modified as:

$$w_i(x) = \text{Softmax}\left(\frac{z(x)}{t}\right)_i. \quad (13)$$

Figure 8 (a) shows the training and validation loss throughout dense training of routers, from 0 to 0.5B tokens. For both DIVE 1/8 and DIVE 2/8, the models with the temperature coefficient consistently outperform those without it (validation loss of 2.82 vs 2.87, 3.65 vs 3.69), which aligns with our expectations. Furthermore, as shown in Figure 8 (b), the models converge around 0.5B tokens, which is the point we use to standardize the dense training stage.

C Further Retraining Details

C.1 Hyper-parameters

To further validate our approach, we retrain the DIVE and LLaMA-MoE models on an expanded training dataset. The experiments are conducted using 4 A800 (80GB) GPUs with a global batch size of 256 and a fixed learning rate of 1e-5. Each model is further trained on an additional 10B tokens, resulting in a total of 15B tokens including the initial retraining. Detailed parameters are provided in Table 8; unless otherwise specified, the settings are consistent with those in Appendix B.1.

C.2 Results

Table 7 reports the performance of DIVE and LLaMA-MoE models after retraining on the extended token budget. The first two columns, marked with downward arrows, report perplexity on language modeling benchmarks, while the remaining columns present accuracy results on a suite of downstream tasks.

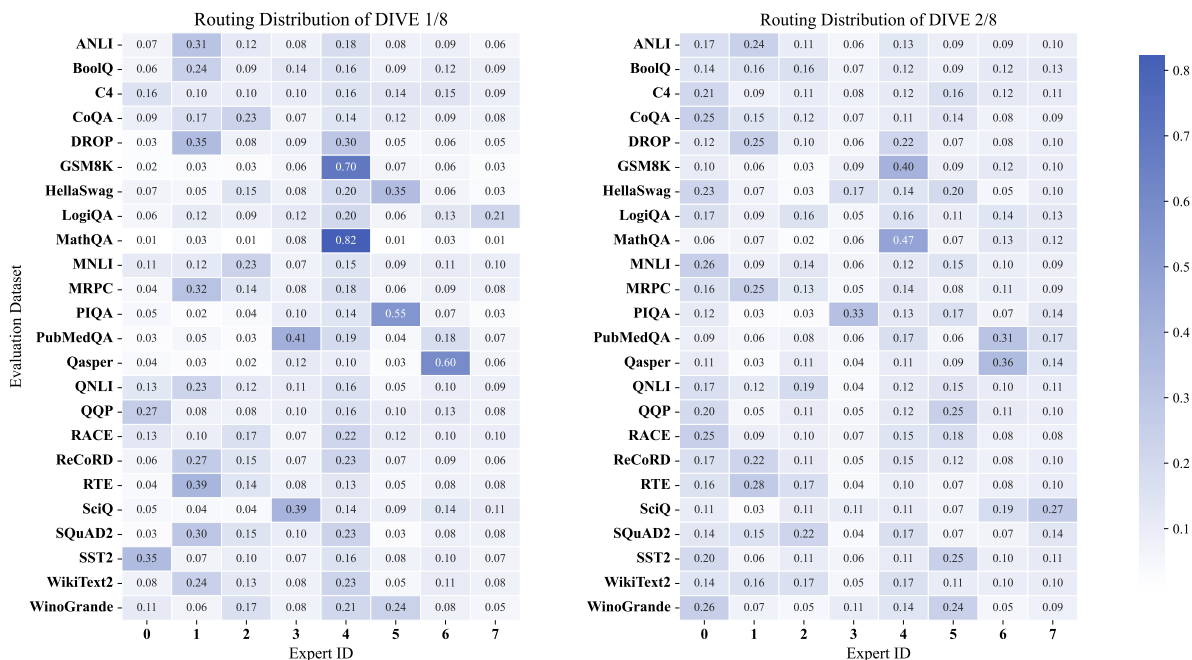Figure 9: Full routing distribution of DIVE 1/8 (left) and DIVE 2/8 (right), shown as activation ratios of the corresponding experts across all layers. The indexes of experts correspond to the domain clustering results in Appendix D.2.

Routing Distribution of DIVE 1/8

Evaluation Dataset	0	1	2	3	4	5	6	7
ANLI	0.07	0.31	0.12	0.08	0.18	0.08	0.09	0.06
BoolQ	0.06	0.24	0.09	0.14	0.16	0.09	0.12	0.09
C4	0.16	0.10	0.10	0.10	0.16	0.14	0.15	0.09
CoQA	0.09	0.17	0.23	0.07	0.14	0.12	0.09	0.08
DROP	0.03	0.35	0.08	0.09	0.30	0.05	0.06	0.05
GSM8K	0.02	0.03	0.03	0.06	0.70	0.07	0.06	0.03
HellaSwag	0.07	0.05	0.15	0.08	0.20	0.35	0.06	0.03
LogiQA	0.06	0.12	0.09	0.12	0.20	0.06	0.13	0.21
MathQA	0.01	0.03	0.01	0.08	0.82	0.01	0.03	0.01
MNLI	0.11	0.12	0.23	0.07	0.15	0.09	0.11	0.10
MRPC	0.04	0.32	0.14	0.08	0.18	0.06	0.09	0.08
PIQA	0.05	0.02	0.04	0.10	0.14	0.55	0.07	0.03
PubMedQA	0.03	0.05	0.03	0.41	0.19	0.04	0.18	0.07
Qasper	0.04	0.03	0.02	0.12	0.10	0.03	0.60	0.06
QNLI	0.13	0.23	0.12	0.11	0.16	0.05	0.10	0.09
QQP	0.27	0.08	0.08	0.10	0.16	0.10	0.13	0.08
RACE	0.13	0.10	0.17	0.07	0.22	0.12	0.10	0.10
ReCoRD	0.06	0.27	0.15	0.07	0.23	0.07	0.09	0.06
RTE	0.04	0.39	0.14	0.08	0.13	0.05	0.08	0.08
SciQ	0.05	0.04	0.04	0.39	0.14	0.09	0.14	0.11
SQuAD2	0.03	0.30	0.15	0.10	0.23	0.03	0.08	0.08
SST2	0.35	0.07	0.10	0.07	0.16	0.08	0.10	0.07
WikiText2	0.08	0.24	0.13	0.08	0.23	0.05	0.11	0.08
WinoGrande	0.11	0.06	0.17	0.08	0.21	0.24	0.08	0.05

Expert ID

Routing Distribution of DIVE 2/8

Evaluation Dataset	0	1	2	3	4	5	6	7
ANLI	0.17	0.24	0.11	0.06	0.13	0.09	0.09	0.10
BoolQ	0.14	0.16	0.16	0.07	0.12	0.09	0.12	0.13
C4	0.21	0.09	0.11	0.08	0.12	0.16	0.12	0.11
CoQA	0.25	0.15	0.12	0.07	0.11	0.14	0.08	0.09
DROP	0.12	0.25	0.10	0.06	0.22	0.07	0.08	0.10
GSM8K	0.10	0.06	0.03	0.09	0.40	0.09	0.12	0.10
HellaSwag	0.23	0.07	0.03	0.17	0.14	0.20	0.05	0.10
LogiQA	0.17	0.09	0.16	0.05	0.16	0.11	0.14	0.13
MathQA	0.06	0.07	0.02	0.06	0.47	0.07	0.13	0.12
MNLI	0.26	0.09	0.14	0.06	0.12	0.15	0.10	0.09
MRPC	0.16	0.25	0.13	0.05	0.14	0.08	0.11	0.09
PIQA	0.12	0.03	0.03	0.33	0.13	0.17	0.07	0.14
PubMedQA	0.09	0.06	0.08	0.06	0.17	0.06	0.31	0.17
Qasper	0.11	0.03	0.11	0.04	0.11	0.09	0.36	0.14
QNLI	0.17	0.12	0.19	0.04	0.12	0.15	0.10	0.11
QQP	0.20	0.05	0.11	0.05	0.12	0.25	0.11	0.10
RACE	0.25	0.09	0.10	0.07	0.15	0.18	0.08	0.08
ReCoRD	0.17	0.22	0.11	0.05	0.15	0.12	0.08	0.10
RTE	0.16	0.28	0.17	0.04	0.10	0.07	0.08	0.10
SciQ	0.11	0.03	0.11	0.11	0.11	0.07	0.19	0.27
SQuAD2	0.14	0.15	0.22	0.04	0.17	0.07	0.07	0.14
SST2	0.20	0.06	0.11	0.06	0.11	0.25	0.10	0.11
WikiText2	0.14	0.16	0.17	0.05	0.17	0.11	0.10	0.10
WinoGrande	0.26	0.07	0.05	0.11	0.14	0.24	0.05	0.09

Expert ID

Training Parameter	Value
# GPUs	4
Learning rate	1e-5
Learning rate scheduler	Constant
Global batch size	256

Table 8: Training parameters for the further retraining experiments (an additional 10B-token training phase, totaling 15B tokens), applied to DIVE and LLaMA-MoE models.

Among the four models evaluated, DIVE 1/8 achieves the best overall performance, with the lowest perplexity scores on both WikiText2 and LAMBADA, and the highest average accuracy across downstream tasks. In particular, it reduces the perplexity of the LLaMA-MoE models by at least 5.37 on WikiText2 and 49.81 on LAMBADA, and improves the best average downstream accuracy of LLaMA-MoE by 2.75%. It shows strong performance on SciQ, PIQA, ARC-e, ARC-c, HellaSwag, and OBQA. DIVE 2/8 achieves the second-best overall results, with notable improvements in WinoGrande, LogiQA, and MMLU. These results demonstrate that our approach consistently outperforms LLaMA-MoE when retrained with more

data, highlighting the robustness of our method.

D Domain Clustering Details

D.1 Domain Affinity Mining Datasets

In domain affinity mining, we conduct a large-scale empirical study on the relationship between 24 calibration datasets used for pruning and the performance of the pruned models. Our dataset selection is inspired by SPoT (Vu et al., 2022), a transfer learning approach using a learned prompt in source tasks and then initializing the prompt for a target task, which gains superior performance in a variety of downstream tasks, demonstrating its generalization. Considering generalization and an exploration of specialization, we use the source task groups in SPoT and extend them by specialized tasks, using their training sets as calibration datasets and test sets as the evaluation datasets.

These datasets include language modeling (C4 (Raffel et al., 2020), WikiText2 (Merity et al., 2017)), natural language inference (ANLI (Nie et al., 2020), MNLI (Williams et al., 2018), QNLI (Zhang et al., 2015), RTE (Wang et al., 2019)), question answering (SQuAD2 (Rajpurkar et al., 2018), BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), DROP (Dua et al.,

19390

Method	WikiText2 ↓	LAMBADA ↓	SciQ	PIQA	WinoGrande	MathQA	HellaSwag
DIVE 1/8	**13.90**	**27.09**	**82.90**	**67.19**	**55.01**	**22.55**	**36.97**
- D.A.M	14.25	31.08	81.60	65.40	54.62	22.35	35.84
DIVE 2/8	**18.09**	**63.45**	**81.50**	**63.98**	**52.96**	**22.65**	**32.26**
- D.A.M	20.02	76.03	79.90	60.72	52.33	22.48	30.40

Method	LogiQA	BoolQ	ARC-e	ARC-c (25)	OBQA	MMLU (5)	Average
DIVE 1/8	18.59	59.54	**51.56**	25.94	18.80	**25.10**	**42.20**
- D.A.M	**20.74**	**61.31**	50.67	**26.11**	**20.60**	23.86	42.10
DIVE 2/8	21.66	60.18	**44.87**	**22.18**	16.60	**27.03**	**40.53**
- D.A.M	**21.81**	**61.31**	40.74	21.67	**17.20**	24.98	39.47

Table 9: Performance of DIVE with and without D.A.M, retrained on 1B tokens. Tasks marked with downward arrows (WikiText2 and LAMBADA) are evaluated by perplexity with a sequence length of 1024, and others are downstream tasks measured by accuracy. The last column shows the average accuracy of all downstream tasks. The best results are in **bold**.

2019), CoQA (Reddy et al., 2019), LogiQA (Liu et al., 2020), ReCoRD (Zhang et al., 2018)), math problems (GSM8K (Cobbe et al., 2021), MathQA (Amini et al., 2019)), specialized knowledge question answering (SciQ (Welbl et al., 2017), Qasper (Dasigi et al., 2021), PubMedQA (Jin et al., 2019), RACE (Lai et al., 2017)), commonsense reasoning (WinoGrande (Sakaguchi et al., 2020), HellaSwag (Zellers et al., 2019)), paraphrase detection (QQP (Wang et al., 2019), MRPC (Dolan and Brockett, 2005)), sentiment analysis (SST-2 (Socher et al., 2013)).

D.2 Domain Clustering Results

We apply our domain-specific data selection to various dense LLM backbones with different activation functions, including LLaMA2-7B, TinyLlama-1.1B, OPT-6.7B and Qwen2.5-7B, resulting in 8 hierarchical clusters on each LLM with FFN pruning ratios of 50% and 75%. These clusters are used to define the domains of experts in the reconstruction phase. The clustering results are presented in Table 10. Each pruning ratio is inversely related to the size of a single FFN, where FFN Size = 1 − FFN Pruning Ratio.

E Diversity Analysis Details

E.1 Routing Distribution

As described in Section 4.3, we analyze routing distributions across 24 pruning datasets to assess expert diversity. The complete heatmaps are shown in Figure 9, where each value indicates the normalized activation ratio of tokens routed to a specific expert. Expert indices are aligned with domain clusters. These results demonstrate the effectiveness of

our retraining setup and the successful realization of diverse expert routing.

E.2 Case Study

We do a case study on routing distribution to further analyze the diversity, specifically on DIVE 1/8. In Table 11, we select data from the test set of four datasets: C4, MathQA, GSM8K and PubMedQA, and analyze the routing results. The original sentences and expertised ones, which are tokenized and routed, are shown. The corresponding expert is determined by the expert with the highest frequency of the current token across all 22 layers of TinyLlama-1.1B.

As expected, the tokens in the C4 dataset is routed relatively evenly across experts, while the data in MathQA, GSM8K and PubMedQA tends to be more concentrated in their corresponding domain-specific experts. Specifically, many technical terms, such as "Neuronal" and all numerical expressions, are routed to the appropriate experts (Expert 3 and Expert 4). This confirms the success of our diversity approach and routing distribution.

F Ablation Details

In Section 4.4.1, to intuitively demonstrate the contribution of domain affinity mining (D.A.M) to DIVE, downstream tasks are categorized into in-distribution (ID) and out-of-distribution (OOD) groups, based on whether they belong to the 24 datasets used for pruning. Specifically, ID tasks include SciQ, PIQA, WinoGrande, MathQA, HellaSwag, LogiQA, and BoolQ, while OOD tasks include ARC-e, ARC-c, OBQA, and MMLU.

Besides the ablation study on DIVE 2/8, we also

conduct one on DIVE 1/8 with and without D.A.M. The full results are shown in Table 9. These results indicate that incorporating D.A.M into DIVE 1/8 reduces perplexity on WikiText2 and LAMBADA by 0.35 and 3.99, respectively. Furthermore, downstream task performance of DIVE 1/8 consistently improves with D.A.M, highlighting its significant role and impact on overall model effectiveness.

Dense Model	FFN Pruning Ratio	Cluster Index	Calibration Datasets
LLaMA2-7B	50%	0	C4, MNLI, ANLI, RACE, CoQA
		1	QNLI, SQuAD2, BoolQ
		2	WikiText2, RTE, DROP, ReCoRD, MRPC
		3	SciQ, Qasper, PubMedQA
		4	GSM8K, MathQA
		5	PIQA, HellaSwag
		6	WinoGrande, SST-2
		7	LogiQA, QQP
LLaMA2-7B	75%	0	C4, MNLI, ANLI, CoQA
		1	QNLI, SQuAD2, BoolQ
		2	WikiText2, RTE, DROP, ReCoRD, MRPC
		3	SciQ, Qasper, PubMedQA
		4	GSM8K, MathQA
		5	PIQA, WinoGrande, HellaSwag
		6	RACE, QQP, SST-2
		7	LogiQA
TinyLlama-1.1B	50%	0	C4, RACE, QQP, SST-2
		1	WikiText2, QNLI, RTE, SQuAD2, BoolQ, DROP, ReCoRD, MRPC
		2	MNLI, ANLI, CoQA
		3	SciQ, PubMedQA
		4	GSM8K, MathQA
		5	PIQA, WinoGrande, HellaSwag
		6	Qasper
		7	LogiQA
TinyLlama-1.1B	75%	0	C4, MNLI, CoQA, RACE
		1	WikiText2, RTE, ANLI, BoolQ, DROP, ReCoRD, MRPC
		2	QNLI, SQuAD2, LogiQA
		3	PIQA
		4	GSM8K, MathQA
		5	WinoGrande, HellaSwag, QQP, SST2
		6	Qasper, PubMedQA
		7	SciQ
OPT-6.7B	50%	0	C4, LogiQA
		1	WikiText2, QNLI, SQuAD2, BoolQ
		2	MNLI, ANLI, CoQA
		3	RTE, DROP, ReCoRD, MRPC
		4	GSM8K, MathQA, HellaSwag, PIQA
		5	RACE, WinoGrande, SST2
		6	QQP
		7	SciQ, Qasper, PubMedQA
OPT-6.7B	75%	0	C4, QNLI
		1	WikiText2, SQuAD2, BoolQ
		2	ANLI, CoQA
		3	RTE, DROP, ReCoRD, MRPC
		4	GSM8K, MathQA, HellaSwag, QQP, PIQA
		5	WinoGrande, SST2
		6	MNLI, RACE
		7	LogiQA, SciQ, Qasper, PubMedQA
Qwen2.5-7B	50%	0	C4, RACE, SST2
		1	WikiText2, QNLI, SQuAD2, BoolQ, DROP
		2	MNLI, RTE, ANLI, CoQA, ReCoRD, MRPC
		3	PIQA, HellaSwag
		4	GSM8K, MathQA
		5	WinoGrande
		6	SciQ, Qasper, PubMedQA
		7	LogiQA, QQP

Table 10: Clustering results of LLaMA2-7B, TinyLlama-1.1B, OPT-6.7B and Qwen2.5-7B for FFN pruning ratios of 50% and 75%.

Data Source	Sentence
C4 (Original)	Who am I? What's this all About? You could say I grew up traveling, when I was young I lived in Lagos with my family. Over the years I spent there we traveled to some of the countries you see highlighted in green.
C4 (Expertised)	\<s\> _Who _am _I ? _What ' s _this _all _About ? _You _could _say _I _grew _up _travel _ing "," _when _I _was _young _I _lived _in _Lag os _with _my _family . _Over _the _years _I _spent _there _we _tra ve led _to _some _of _the _countries _you _see _hightlight ed _in _green .
MathQA (Original)	a multiple choice test consists of 4 questions, and each question has 5 answer choices. in how many r ways can the test be completed if every question is unanswered?
MathQA (Expertised)	\<s\> _a _multiple _choice _test _consists _of _ 4 _questions "," _and _each _question _has _ 5 _answer _choices _ . _in _how _many _r _ways _can _the _test _be _completed _if _every _question _is _un answer ed _?
GSM8K (Original)	To make the pizza, Kimber half as many teaspoons of salt as the number of cups of flour, meaning she needs 1/2*16 = «16*1/2=8»8 teaspoons of salt. The total number of cups of flour and teaspoons of salt she needs is 8+16 = «8+16=24»24 She also needs 10 cups of water, which means the total number of cups of water and flour and teaspoons of salt she needs is 24+10 = «24+10=34»34 #### 34
GSM8K (Expertised)	\<s\> _To _make _the _p izza "," _Kim ber _half _as _many _te as po ons _of _salt _as _the _number _of _cu ps _of _fl our , _meaning _she _needs _ 1 / 2 * 1 6 _= _« 1 6 * 1 / 2 = 8 » 8 _te as po ons _of _salt . _The _total _number _of _cu ps _of _fl our _and _te as po ons _of _salt _she _needs _is _ 8 + 1 6 _= _« 8 + 1 6 = 2 4 » 2 4 _She _also _needs _ 1 0 _cu ps of _water "," _which _means _the _total _number _of _cu ps _of _water _and _fl our _and _te as po ons _of _salt _she _needs _is _ 2 4 + 1 0 _= _« 2 4 + 1 0 = 3 4 _#### _ 3 4
PubMedQA (Original)	Neuronal signals via the hepatic vagus nerve contribute to the development of steatohepatitis and protection against obesity in HFD fed Pemt(-/-) mice.
PubMedQA (Expertised)	\<s\> _Ne ur onal _signals _via _the _he p _atic _v ag us _n erve _contribute _to _the _development _of _ste ato he pat itis _and _protection _against _ob es ity _in _H FD _fed _P em t (- / -) _m ice .

Table 11: Case study on DIVE 1/8. (Original) refers to the original data, while the (Expertised) ones denote the routing results to experts of tokenized sentences and are marked by colors. Purple, green, yellow, orange, red, magenta, cyan and blue respectively denote the Expert IDs of 0-7.