Improved Sparse Upcycling for Instruction Tuning

Wangyi Jiang[1,2], **Yaojie Lu**[1], **Hongyu Lin**[1], **Xianpei Han**[1], **Le Sun**[1]
[1]Chinese Information Processing Laboratory, Institute of Software, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
{jiangwangyi2020,luyaojie,hongyu,xianpei,sunle}@iscas.ac.cn

Abstract

The Mixture-of-Experts (MoE) architecture has demonstrated significant potential in both large-scale pre-training and instruction tuning by offering increased parameter capacity without additional inference costs. However, developing MoE models faces challenges including training instability and the need for substantial high-quality training data. While efficient methodologies like sparse upcycling exist, they often lead to performance degradation in instruction tuning scenarios. We introduce representation-based sparse upcycling, a straightforward yet effective technique for converting dense language models into sparsely activated ones while maintaining similar computational costs. Unlike conventional sparse upcycling, our approach leverages intermediate representations from language models to initialize router weights. This strategy addresses the mismatch between randomly initialized and well-trained parameters while providing prior knowledge to guide expert specialization during training. Extensive experiments across diverse benchmarks demonstrate significant improvements in both model capabilities and routing consistency compared to existing approaches.

1 Introduction

Instruction tuning (Wei et al., 2022) has emerged as a pivotal technique for enhancing language models' capabilities by fine-tuning them on instruction-annotated datasets. Large Language Models (LLMs) that undergo instruction tuning demonstrate superior downstream performance on held-out tasks in both zero-shot and few-shot settings (Ouyang et al., 2022). Recent research indicates that increasing the diversity and quality of instruction tuning data yields substantially more significant improvements compared to merely expanding data quantity (Zhou et al., 2023a). Contemporary studies have focused on curating high-quality datasets through prompting advanced LLMs such as ChatGPT and GPT-4, subsequently training smaller models to emulate their reasoning and problem-solving processes (Wang et al., 2023b; Xu et al., 2024; Mukherjee et al., 2023). However, a substantial performance disparity persists between models of varying sizes. Smaller language models consistently encounter difficulties in complex reasoning scenarios, such as solving mathematical competition problems, primarily due to their limited parameter capacity constraining their achievable capabilities.

The Mixture-of-Experts (MoE) architecture (Shazeer et al., 2017) offers a promising solution by partitioning parameters into expert subsets and selectively activating only a fraction of these experts for each input during both training and inference. This architectural innovation enables MoE models to incorporate vast parameter counts while maintaining moderate computational requirements, frequently demonstrating superior capabilities compared to dense models with comparable inference costs (Fedus et al., 2022b; Du et al., 2022; Jiang et al., 2024). Nevertheless, MoE models commonly exhibit training instabilities (Fedus et al., 2022b; Du et al., 2022; Zoph et al., 2022), necessitating various mitigation techniques. However, validating these techniques on large-scale language models demands substantial computational resources. Consequently, constructing MoE models from pre-trained dense models presents a more resource-efficient alternative to training from scratch.

Komatsuzaki et al. (2023) introduced *sparse upcycling*, a methodology for converting existing dense models into larger, sparsely activated models by replicating MLP layers and randomly initializing router weights. While upcycled T5 (Raffel et al., 2020) models demonstrated performance improvements through continued pre-training, these gains diminished as base model size increased.

Proceedings of the 31st International Conference on Computational Linguistics, pages 9485–9498
January 19–24, 2025. ©2025 Association for Computational Linguistics

Moreover, when applying limited additional training, the original and upcycled models exhibited comparable performance. The homogeneity of replicated MLPs and randomly initialized routers impedes optimal training outcomes.

To address these limitations, we propose a representation-based sparse upcycling method. Based on our observation that internal representations of tokens from specific tasks tend to form distinct clusters within high-dimensional representation spaces in well-trained language models, we conceptualize expert routing behavior as a matching process between expert representations within the router and task- or context-aware token representations. This insight emphasizes the critical role of router weight initialization. By initializing routers with abstracted internal representations, we guide experts to focus on specific, semantically related, or task-related tokens, significantly mitigating random routing and training instability issues while facilitating expert specialization.

We validate our approach through comprehensive instruction tuning experiments, demonstrating superior downstream task performance and consistent routing behaviors.

We summarize our contributions as follows:

- We propose a novel representation-based sparse upcycling approach that improves upon existing sparse upcycling methods by initializing router weights with task or context-aware representations, thereby reducing training instability and enhancing expert specialization.
- We empirically demonstrate that intermediate representations in well-trained dense models exhibit inherent clustering tendencies, which we leverage to facilitate efficient expert routing in sparse models.
- We validate the effectiveness of our approach through extensive instruction tuning experiments across diverse benchmarks, showing significant improvements in downstream task performance and routing consistency.

Code will be available at https://github.com/icip-cas/sparse-upcycling.

2 Related Work

2.1 Mixture-of-Experts

Mixture-of-Experts (MoE) are a variant of sparse expert models, in which a part of the parameters are partitioned into individual experts (Fedus et al., 2022a). During both training and inference, only a subset of these experts is selectively activated based on the input features. This selective activation allows each expert to specialize in specific tasks, significantly boosting performance metrics across a variety of applications. By assigning particular tasks to the most suitable experts, the model effectively harnesses their specialized knowledge, leading to substantial improvements in performance.

Shazeer et al. (2017) applies MoE layers between stacked LSTM layers (Hochreiter and Schmidhuber, 1997), resulting in the creation of the largest model at that time, which achieves state-of-the-art performance on language modeling and machine translation. Subsequent research has focused on unleashing the potential of the Transformer architecture (Vaswani et al., 2017) and achieve substantial advancements on both language and vision tasks (Lepikhin et al., 2021; Fedus et al., 2022b; Jiang et al., 2024; Ruiz et al., 2021; Wu et al., 2022; Puigcerver et al., 2023).

Nonetheless, much of the existing work has concentrated on training sparse models from scratch, with training stability emerging as a major research focus. Zoph et al. (2022) conducts a large-scale stability study of sparse models, investigating how factors such as multiplicative interactions, noise injection during training, auxiliary router loss, and training precision contribute to improving model stability. Dai et al. (2022) identified the routing fluctuation problem in previous MoE methods and proposed a balanced and cohesive routing strategy to address this issue.

Sparse Upcycling (Komatsuzaki et al., 2023), outlines a methodology for transitioning from a well-trained dense model to a sparse model, rather than to a larger dense model. This technique leverages the additional capacity provided by increased parameters while maintaining inference costs through computational sparsity. We adopt this approach to create sparsely activated models in our research.

2.2 Instruction Tuning

Instruction tuning (Wei et al., 2022) involves fine-tuning pre-trained language models on datasets comprising instruction-output pairs. This process enhances the models' ability to understand and execute human instructions effectively.

The success of instruction tuning largely depends on the creation of high-quality datasets. Both manually annotated data (Ouyang et al., 2022) and synthetically generated data through distillation (Wang et al., 2023b; Taori et al., 2023) are employed to boost language models' performance in areas such as general reasoning (Xu et al., 2024), code generation (Luo et al., 2023b; Wei et al., 2024; Yu et al., 2024b), and mathematical problem solving (Luo et al., 2023a; Yu et al., 2024a; Yue et al., 2024a,b).

Research into efficient and effective instruction tuning techniques is an important complementary direction. NEFTune (Jain et al., 2024) enhances the conversational capabilities of instruction-tuned models by fine-tuning with noisy embeddings. LoRA (Hu et al., 2022), along with its sparse variations (Wu et al., 2024b; Gou et al., 2024; Wu et al., 2024a), focuses on adapting language models to downstream tasks by optimizing only a subset of parameters, thus minimizing performance loss.

3 Preliminary

3.1 Sparsely Activated Mixture-of-Experts

In Transformer based MoE models, a prevalent approach involves substituting the Feed-Forward Networks (FFNs) within certain Transformer blocks with specialized experts, which are collections of inherently independent FFNs. Additionally, a routing network is incorporated to allocate the appropriate experts for each input feature.

In this work, we primarily focus on Top-k routing (Shazeer et al., 2017; Fedus et al., 2022b; Du et al., 2022). The router takes a token representation as input and routes it to the best k experts, selected from a set of N experts $\{E\}_{i=1}^{N}$. Specifically, the router weights, $W_r \in \mathbb{R}^{N\times d}$, where each row $r_i \in \mathbb{R}^d$ represents an expert embedding, produce logits $r(x) = W_r \cdot x$. The logits are then normalized via the softmax function, yielding a distribution over the experts,

$$p_i(x) = \frac{e^{r(x)_i}}{\sum_j^N e^{r(x)_j}}. \tag{1}$$

The input then selects the k experts with highest probabilities, where the indices of the selected experts constitute a set $\mathcal{K}$. The output of the MoE layer is computed as a linear combination of the output of the selected experts,

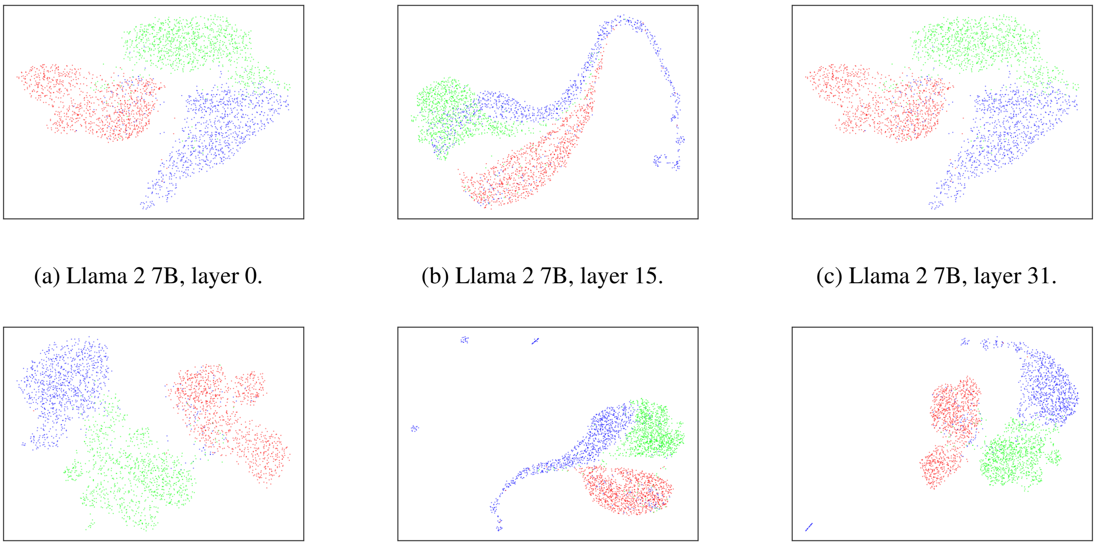
$$y = \sum_{i\in\mathcal{K}} p_i(x)E_i(x). \tag{2}$$

3.2 Specialization

Sparse Models Expert specialization within MoE models is critical for fostering diversity among the experts. This diversity is essential because if all experts converge towards homogeneity, the MoE model effectively becomes a conventional dense model, thereby diminishing its intended benefits. The phenomenon known as *representation collapse*, where experts fail to maintain distinct knowledge or skills, poses a significant challenge to the effectiveness of sparse MoE models. Chi et al. (2022) provides a theoretical examination of this issue, suggesting that addressing representation collapse can lead to substantial performance improvements across various tasks.

The significance of expert specialization is further underscored by empirical observations across numerous studies. For instance, research by Lewis et al. (2021) demonstrates that the assignment of inputs to experts is influenced by local syntactic information, indicating a form of specialization based on the nature of the input data. Moreover, in the context of sparse encoder-decoder Transformers, Zoph et al. (2022) observed distinct specializations among encoder experts, with certain experts focusing predominantly on specific linguistic elements such as punctuation, verbs, proper nouns, and numerical data. This specialization is not limited to linguistic tasks. In multimodal MoE models, both modality-specific and multimodal experts exhibit specialization, effectively enhancing the model's performance across diverse datasets and tasks, as illustrated by Mustafa et al. (2022).

Dense Models We perform an empirical analysis on the inherent domain specialization of dense language models. In particular, we are interested whether there exists a fascinating pattern regarding the layer-wise representation of data points from specific domains (e.g. code, mathematics, etc.). We sample $1,000$ textual examples each from general reasoning datasets, code generation datasets, and mathematical reasoning datasets. Layer-wise representations before and after FFNs generated by language models are taken into consideration, and subsequently projected using Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2020) for spatial structure visualization in hyperbolic space. Figure 1 shows that, for both Llama 2 7B (Touvron et al., 2023) and Llama 3 8B (Dubey et al., 2024), the internal representation of the models originating from homogeneous do-



(a) Llama 2 7B, layer 0. (b) Llama 2 7B, layer 15. (c) Llama 2 7B, layer 31.

(d) Llama 3 8B, layer 0. (e) Llama 3 8B, layer 15. (f) Llama 3 8B, layer 31.

Figure 1: Representations generated by the self-attention modules of Llama 2 7B and Llama 3 8B model. Red, blue, and green points are samples from datasets specialized in text, code, and math, respectively.

mains tend to aggregate into distinct clusters within high-dimensional space.

4 Methodology

This section presents our novel approaches for transforming dense language models into sparse architectures. We introduce two distinct methodologies: *task-representation based sparse upcycling* (TRSU), and *context-representation based sparse upcycling* (CRSU). Following the transformation, all models undergo instruction tuning to ensure alignment with target tasks.

4.1 Problem Formulation

Our preliminary experiments revealed that sparse upcycled MoE models frequently underperform compared to their dense counterparts on downstream tasks when instruction-tuned on datasets ranging from thousands to millions of instances. This observation is particularly noteworthy given that sparse upcycled models and dense models possess equivalent capabilities prior to training. While Komatsuzaki et al. (2023) demonstrated that upcycled models exhibit performance advantages after training on large-scale datasets such as C4 (Raffel et al., 2020), we hypothesize that limited training data constrains the sparse models' ability to develop effective and consistent routing behavior.

The random initialization of router weights results in a highly entropic initial state during training, leading to limited and potentially noisy knowledge acquisition by each expert. Therefore, providing a priori guidance for routing behavior becomes crucial, serving as an entropy reduction mechanism to ensure more consistent and robust performance during subsequent training and inference phases.

4.2 Task Aware Sparse Upcycling

Contemporary large language models demonstrate proficiency across a spectrum of tasks, from basic to complex. We focus on scenarios where models undergo instruction tuning on diverse task sets to function as effective assistants.

Given a set of tasks $\mathcal{T} = \{t_i\}_{i=1}^{T}$, our objective is to upcycle a language model of limited size to a sparse architecture with T experts. To mitigate the inherent randomness in sparse model training and provide task-oriented guidance for each expert – specifically, designating each expert to primarily handle tokens with high representation similarity – we heuristically employ representative task representations as the initialization parameters for router weights.

As demonstrated in Section 3.2, the internal representations generated by language models from high-quality task data naturally cluster within high-dimensional space. These representations interact with expert representations within the routers to determine routing behavior. We leverage this phenomenon by: (1) Sampling representative task data;

(2) Clustering the generated attention representations; (3) Utilizing resulting vectors as initialization parameters for specific experts. This approach facilitates matching task-relevant data to corresponding experts while delegating task-irrelevant data to alternative experts.

4.3 Context Aware Sparse Upcycling

While the task-aware approach represents an advancement over conventional sparse upcycling, it presents certain limitations: (1) The number of experts is constrained to match the number of tasks; (2) The clustering of task data representations relies on empirical observation rather than rigorous theoretical foundation; (3) As task quantity increases or task boundaries become less distinct, clusters may overlap, potentially compromising routing effectiveness.

To enhance practical applicability, we propose a more generalized methodology. Given a dataset consisting single or multiple tasks, we sample a representative subset of the data and perform clustering on their representations generated by language models in high-dimensional space, using a pre-defined number K (e.g., via K-means clustering (MacQueen et al., 1967)). This process yields K representative directions within the contextual embedding space, which are then assigned to K experts as initial router representations.This approach removes the constraint on expert quantity imposed by the task-aware method. The resulting vectors may represent more general token categories beyond task-specific representations, such as numerical text, code, or punctuation, aligning with observations by Zoph et al. (2022).

4.4 Design Decisions

MoE models share several significant configurations, including router type, number of sparse layers, number of experts per layer, number of experts to activate, etc., which exert influence on computation budget, model size, and model performance.

Router type We mainly focus on the classic switch routing (Top-1 routing) (Fedus et al., 2022b). Despite the augmentation of total parameters, the MoE layer exhibits comparable computational efficiency to dense ones.

Number of sparse layers Incorporating more sparse layers is benefit to enhance the capacity of models, albeit with a concomitant escalation in computational and resource expenditure. Owing to the recent success of Mixtral (Jiang et al., 2024) and DeepSeekMoE (Dai et al., 2024), we transform all transformer layers into sparse ones.

Load balance Following Fedus et al. (2022b), we adopt a differentiable load balancing loss to encourage uniform routing over experts. Specifically, given N experts and a batch of $B \times L$ tokens, the auxiliary loss is computed as the inner product of vectors f and P,

$$\text{loss} = \alpha \cdot N \cdot \sum_{i=1}^{N} f_i \cdot P_i \tag{3}$$

where f_i is the fraction of the tokens dispatched to expert i, and P_i is the fraction of the router probability allocated for expert i,

$$\begin{aligned} f_i &= \frac{1}{L} \sum_{x \in \mathcal{B}} \mathbb{1}\{\text{argmax} p(x) = i\} \\ P_i &= \frac{1}{L} \sum_{x \in \mathcal{B}} p_i(x) \end{aligned} \tag{4}$$

a hyper-parameter α is a multiplicative coefficient for the load balancing loss.

5 Experiments

This section elucidates our exploration of language model sparsity through several methodologies. We employ three distinct approaches to transform dense language models into their sparse counterparts: sparse upcycling (SU), task-representation based sparse upcycling (TRSU), and context-representation based sparse upcycling (CRSU). Subsequently, all models, both dense and sparse, undergo instruction tuning to ensure alignment. We then conduct comprehensive evaluations to assess and compare the performance and characteristics of these models.

5.1 Experimental Setup

5.1.1 Training Datasets

We evaluate TRSU's effectiveness through a comprehensive multi-task learning framework. We focus on three fundamental domains: natural language (text), programming (code), and mathematical reasoning (math), following Ding et al. (2024). While these domains are essential for demonstrating the capabilities of LLMs, their integration into a unified model presents significant challenges. The training process leverages a curated dataset from established open-source resources: OpenOrca

Model	Size	**MMLU** Acc (%)	**HellaSwag** Acc (%)	**HumanEval** Pass@1 (%)	**MBPP** Pass@1 (%)	**GSM8K** Acc (%)	**MATH** Acc (%)	**Average** (%)
Danube 2	1.8B	39.7	**70.9**	34.5	30.2	49.2	13.7	39.7
	$2\times1.8\mathrm{B}_{\mathrm{SU}}$	38.8	70.1	32.8	28.6	48.7	12.8	38.6
	$2\times1.8\mathrm{B}_{\mathrm{TRSU}}$	**40.2**	70.4	**36.2**	**30.7**	**49.8**	**14.0**	**40.2**
Llama 2	7B	47.4	**75.2**	47.6	53.7	59.2	17.3	50.0
	$2\times7\mathrm{B}_{\mathrm{SU}}$	47.1	73.9	44.3	51.4	58.5	16.8	48.6
	$2\times7\mathrm{B}_{\mathrm{TRSU}}$	**49.5**	74.8	**48.4**	**54.1**	**59.9**	**18.0**	**50.7**

Table 1: Overall TRSU results of dense and sparse models across benchmarks.

Model	Size	**MMLU** Acc (%)	**MMLU-Pro** Acc (%)	**IFEval** Acc (%)	**HumanEval** Pass@10 (%)	**MATH** Acc (%)	**Average** (%)
Danube 2	1.8B	44.2	14.8	26.6	19.6	**4.9**	22.0
	$2\times1.8\mathrm{B}_{\mathrm{SU}}$	43.6	15.1	25.8	19.5	4.1	21.6
	$2\times1.8\mathrm{B}_{\mathrm{TRSU}}$	**44.5**	**15.8**	**27.2**	**20.2**	4.7	**22.5**
Llama 2	7B	49.7	19.7	35.2	25.0	**7.4**	27.4
	$2\times7\mathrm{B}_{\mathrm{SU}}$	48.9	19.9	33.7	25.0	6.7	26.8
	$2\times7\mathrm{B}_{\mathrm{TRSU}}$	**50.1**	**20.5**	**37.1**	**27.1**	7.1	**28.4**

Table 2: Overall CRSU results of dense and sparse models across benchmarks.

dataset (Lian et al., 2023a), Magicoder Evol Instruct (Luo et al., 2023b), Magicoder OSS Instruct (Wei et al., 2024), and MetaMathQA (Yu et al., 2024a), following filtration and sampling procedures.

To assess CRSU's domain-agnostic capabilities, we employ SlimOrca (Lian et al., 2023b), a curated subset of OpenOrca (Lian et al., 2023a). This dataset extends the FLAN Collection (Longpre et al., 2023) by incorporating step-by-step reasoning patterns derived from GPT-3.5 and GPT-4.

5.1.2 Implementation Details

We implement our approach using Mergekit (Goddard et al., 2024) to transform dense language models into sparse architectures. Unlike conventional sparse upcycling that randomly initializes expert routers, our method derives task and context representations from both the base model and training data. We then apply K-means clustering to generate representative router parameters (see Appendix B).

Our experiments utilize two base models: H2O-Danube2-1.8B (Singer et al., 2024) (hereinafter Danube 2 1.8B) and Llama 2 7B (Touvron et al., 2023). Training proceeds for one epoch using an 8×80GB A100 GPU cluster, with a batch size of 128 and maximum sequence length of 4096. Model optimization employs the AdamW optimizer (Loshchilov and Hutter, 2019) with learning rates of 1×10^{-5} and 2×10^{-5} for the Danube 2 1.8B series and Llama 2 7B series, respectively.

For TRSU experiments, we convert the base models into MoE architectures, incorporating two domain-specific experts (code and math) in each Transformer block. We benchmark both SU and TRSU models against the original dense models. The CRSU experiments extend this framework with four experts per Transformer block, comparing both SU and CRSU performance.

5.1.3 Evaluation

We conduct comprehensive evaluations of both dense and sparse models across diverse benchmarks to assess their capabilities after fine-tuning. Our evaluation strategy is tailored to the specific training objectives:

- For multi-task trained models, we focus on assessing their ability to simultaneously handle multiple complex tasks. The evaluation spans three key domains: MMLU (Hendrycks et al., 2021a) and HellaSwag (Zellers et al., 2019) for text, HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) for code, and GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021b) for math.

- For models trained on SlimOrca, which encompasses a broader distribution of general instructions, we evaluate their performance across: MMLU (Hendrycks et al., 2021a) and MMLU-Pro (Wang et al., 2024) for general knowledge and reasoning, IFEval (Zhou et al., 2023b) for instruction understanding and execution, HumanEval (Chen et al., 2021) and MATH (Hendrycks et al., 2021b) for technical problem-solving.

To ensure consistency and facilitate fair comparisons, all evaluations adhere to a standardized instruction-following paradigm. Test instances are transformed into a uniform chat template format, maintaining consistent interaction patterns across all benchmarks. Detailed evaluation protocols, metrics, and implementation specifics are provided in Appendix C.

5.2 Main Results

Tables 1 and 2 present comprehensive performance comparisons across different model configurations. Several key findings emerge from our experiments:

Effectiveness of Task-Representation Sparse Upcycling The TRSU approach demonstrates superior performance compared to both dense models and traditional sparse upcycling (SU) across the majority of benchmarks. In the Danube 2 1.8B series, TRSU yields an average performance gain of 0.5 percentage points relative to the dense baseline (40.2% vs. 39.7%) and a more substantial improvement of 1.6 points compared to SU (40.2% vs. 38.6%). The Llama 2 7B series exhibits even more pronounced improvements, with TRSU surpassing the dense model by 0.7 points (50.7% vs. 50.0%) and SU by 2.1 points (50.7% vs. 48.6%).

Context-Representation Benefits The empirical results in Table 2 demonstrate that CRSU effectively captures contextual patterns, particularly in reasoning-intensive tasks. This advantage manifests most notably in MMLU-Pro and IFEval performance metrics. The Llama 2 7B CRSU variant exhibits substantial improvements in both IFEval (37.1% vs. 35.2% dense) and MMLU-Pro (20.5% vs. 19.7% dense), suggesting enhanced capabilities in context-dependent reasoning tasks.

Trade-offs in Mathematical Tasks While our sparse approaches generally enhance performance across most benchmarks, we observe distinct patterns in mathematical reasoning tasks. In the CRSU configuration, both model series exhibit marginal performance degradation on MATH compared to their dense counterparts. This phenomenon may be attributed to the relative underrepresentation of mathematical content in the training corpus, resulting in suboptimal mathematical context specialization during the construction of contextual representations. The limited exposure to mathematical patterns and reasoning structures potentially impedes the development of mathematics-specific routing capabilities, subsequently affecting the models' mathematical problem-solving proficiency.

Model	Size	Text	Code	Math
Danube 2	$2\times1.8\text{B}_{\text{TRSU}}$	55.3	33.4	31.9
	$3\times1.8\text{B}_{\text{TRSU}}$	54.8	33.2	31.3

Table 3: Evaluation on TRSU models with different number of experts in each sparse layer.

Model	Size	MMLU	MMLU-Pro
Danube 2	$2\times1.8\text{B}_{\text{CRSU}}$	42.7	14.1
	$4\times1.8\text{B}_{\text{CRSU}}$	44.5	15.8
	$8\times1.8\text{B}_{\text{CRSU}}$	44.7	15.9

Table 4: Evaluation on CRSU models with different number of experts in each sparse layer.

5.3 Ablations and Analysis

5.3.1 Number of experts

In the TRSU experiments, we transform dense models into MoE architectures by implementing dual experts within each Transformer block. This design aims to enable expert specialization across code and math domains, while treating text processing as a shared responsibility between experts. We further investigate whether designating text as a distinct domain would enhance performance. Table 3 shows the performance comparison between two-expert and three-expert configurations, evaluated on the same benchmarks as Table 1. The slight performance decline from Danube 2 2×1.8B to 3×1.8B suggests that dedicating a specific expert to text processing offers no substantial benefit. This finding indicates that MoE architectures may be more effective for tasks with clear domain boundaries rather than general language tasks that require integrated world knowledge.

For CRSU experiments, we explore the scalability benefits of increasing expert count. Resource limitations constrained our investigation to 2-, 4-,

Training Process	25%	50%	75%	100%
Performance Diff.	+1.9%	+1.4%	+1.1%	+0.9%

Table 5: Performance difference between Danube 2 $4\times1.8\text{B}_{\text{CRSU}}$ and Danube 2 $4\times1.8\text{B}_{\text{SU}}$ during the training process.

Model	**Size**	**Consistency**
Danube 2	$2\times1.8\text{B}_{\text{SU}}$	0.69
	$2\times1.8\text{B}_{\text{CRSU}}$	0.83
Danube 2	$4\times1.8\text{B}_{\text{SU}}$	0.61
	$4\times1.8\text{B}_{\text{CRSU}}$	0.76
Danube 2	$8\times1.8\text{B}_{\text{SU}}$	0.54
	$8\times1.8\text{B}_{\text{CRSU}}$	0.70

Table 6: Routing consistency scores for different expert configurations.

and 8-expert configurations, where context representations are clustered accordingly to determine routing parameters. Table 4 demonstrates that expanding from 2 to 4 experts yields meaningful performance improvements in general language tasks. However, further expansion to 8 experts shows diminishing returns, suggesting an optimal balance point in the trade-off between model complexity and performance gains.

5.3.2 Data Efficiency

We analyze the performance trajectories of Danube 2 4×1.8B models using SU and CRSU approaches at four training milestones: 25%, 50%, 75%, and 100%. Table 5 shows that CRSU consistently outperforms SU, with the largest gap (+1.9%) observed at 25% of training. The performance difference gradually decreases to +0.9% at completion.

This pattern indicates that while standard SU eventually learns effective routing patterns, the representation-guided initialization in CRSU provides a better starting point for expert specialization. The clustering-based approach requires fewer training examples to discover meaningful specialization patterns, making it particularly valuable in scenarios with limited training data or computational resources.

5.3.3 Routing Analysis

This section analyzes the routing dynamics of sparsity-crafted models during training and inference.

Model	**Expert**	**Code**	**Math**
Danube 2 $2\times1.8\text{B}_{\text{SU}}$	0	54.1%	47.6%
	1	45.9%	52.4%
Danube 2 $2\times1.8\text{B}_{\text{TRSU}}$	0	40.7%	62.0%
	1	59.3%	38.0%

Table 7: Proportions of tokens assigned to each expert on text data from code and math domains. Values are reported as the average of layers.

Training. To evaluate routing consistency in MoE models developed through SU and CRSU, we analyze four checkpoints: initial state, one-third, two-thirds, and final state of training. We assess model performance on a held-out test set of 1,000 samples. The Jaccard similarity index quantifies routing consistency by measuring the overlap of activated experts for each input token across checkpoints. Table 6 shows that CRSU achieves higher routing consistency than SU across all expert configurations. This indicates that enhanced router initialization promotes stable routing behavior regardless of expert count, enabling more efficient optimization under limited training resources.

Inference. We analyze the routing patterns of models upcycled through TRSU to determine whether representation-based initialization during training leads to task-specific expert specialization. Using the GSM8K (Cobbe et al., 2021) and MBPP (Austin et al., 2021) datasets, we measure expert selection distribution in TRSU models. Table 7 reveals that models with task-based router weights exhibit clear task-specific expert allocation patterns, effectively matching experts to their specialized domains. In contrast, models with random router weight initialization show more uniform expert allocation across tasks, primarily due to the auxiliary load balancing loss constraints.

6 Conclusion

This paper presents representation-based sparse upcycling, an innovative approach for transforming dense language models into efficient sparsely activated architectures. Our method improves upon existing techniques by leveraging task- and context-aware representations to initialize router weights, enabling more effective expert specialization while maintaining computational efficiency. Through comprehensive evaluation in both multi-task and general instruction-following scenarios, we demonstrate consistent performance improvements over conventional sparse upcycling and dense models across diverse benchmarks. The superior routing consistency and task specialization achieved by our approach highlights the importance of informed initialization in sparse architectures and establishes a promising direction for developing more efficient and capable language models. The success of our method suggests that representation-guided routing could be a key component in advancing the development of specialized sparse models, though further theoretical investigation is warranted to fully understand its mechanisms and potential applications.

7 Limitations

While our representation-based sparse upcycling method effectively mitigates performance degradation and shows improvements over dense models, it has notable limitations. The method's heavy reliance on training data sampling may make it infeasible in scenarios with limited high-quality data. Additionally, further theoretical investigation is needed to fully understand the mechanisms behind its effectiveness and establish performance guarantees. These limitations suggest important directions for future research in developing more robust and theoretically grounded sparse architectures.

Acknowledgements

We sincerely thank the reviewers for their insightful comments and valuable suggestions. This work was supported by Beijing Natural Science Foundation (L243006), the Natural Science Foundation of China (No. 62122077, 62306303, 62106251) and Beijing Municipal Science and Technology Project (Nos. Z231100010323002).

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Zewen Chi, Li Dong, Shaohan Huang, Damai Dai, Shuming Ma, Barun Patra, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, Heyan Huang, and Furu Wei. 2022. On the representation collapse of sparse mixture of experts. In *Advances in Neural Information Processing Systems*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

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. *Preprint*, arXiv:2401.06066.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. StableMoE: Stable routing strategy for mixture of experts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7085–7095, Dublin, Ireland. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Ganqu Cui, Xingtai Lv, Ruobing Xie, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. 2024. Mastering text, code and math simultaneously via fusing highly specialized language models. *arXiv preprint arXiv:2403.08281*.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. 2022. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

William Fedus, Jeff Dean, and Barret Zoph. 2022a. A review of sparse expert models in deep learning. *Preprint*, arXiv:2209.01667.

William Fedus, Barret Zoph, and Noam Shazeer. 2022b. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23(1).

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.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's MergeKit: A toolkit for merging large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 477–485, Miami, Florida, US. Association for Computational Linguistics.

Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Aoxue Li, Dit-Yan Yeung, James T. Kwok, and Yu Zhang. 2024. Mixture of cluster-conditional lora experts for vision-language instruction tuning. *Preprint*, arXiv:2312.12379.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

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 *International Conference on Learning Representations*.

Neel Jain, Ping yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R. Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. NEFTune: Noisy embeddings improve instruction finetuning. In *The Twelfth International Conference on Learning Representations*.

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. *Preprint*, arXiv:2401.04088.

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*.

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

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. Base layers:

Simplifying training of large, sparse models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6265–6274. PMLR.

Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023a. Openorca: An open dataset of gpt augmented flan reasoning traces. `https://https://huggingface.co/Open-Orca/OpenOrca`.

Wing Lian, Guan Wang, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023b. Slimorca: An open dataset of gpt-4 augmented flan reasoning traces, with verification.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. *Preprint*, arXiv:2301.13688.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023a. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *Preprint*, arXiv:2308.09583.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023b. Wizardcoder: Empowering code large language models with evol-instruct. *Preprint*, arXiv:2306.08568.

James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 14, pages 281–297. Oakland, CA, USA.

Leland McInnes, John Healy, and James Melville. 2020. Umap: Uniform manifold approximation and projection for dimension reduction. *Preprint*, arXiv:1802.03426.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *Preprint*, arXiv:2306.02707.

Basil Mustafa, Carlos Riquelme Ruiz, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. 2022. Multimodal contrastive learning with LIMoe: the language-image mixture of experts. In *Advances in Neural Information Processing Systems*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Joan Puigcerver, Carlos Riquelme, Basil Mustafa, and Neil Houlsby. 2023. From sparse to soft mixtures of experts. *Preprint*, arXiv:2308.00951.

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(1).

Carlos Riquelme Ruiz, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. In *Advances in Neural Information Processing Systems*.

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

Philipp Singer, Pascal Pfeiffer, Yauhen Babakhin, Maximilian Jeblick, Nischay Dhankhar, Gabor Fodor, and Sri Satish Ambati. 2024. H2o-danube-1.8b technical report. *Preprint*, arXiv:2401.16818.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas

Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023a. How far can camels go? exploring the state of instruction tuning on open resources. *Preprint*, arXiv:2306.04751.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and LINGMING ZHANG. 2024. Magicoder: Empowering code generation with OSS-instruct. In *Forty-first International Conference on Machine Learning*.

Haoyuan Wu, Haisheng Zheng, Zhuolun He, and Bei Yu. 2024a. Parameter-efficient sparsity crafting from dense to mixture-of-experts for instruction tuning on general tasks. *Preprint*, arXiv:2401.02731.

Lemeng Wu, Mengchen Liu, Yinpeng Chen, Dongdong Chen, Xiyang Dai, and Lu Yuan. 2022. Residual mixture of experts. *Preprint*, arXiv:2204.09636.

Xun Wu, Shaohan Huang, and Furu Wei. 2024b. Mixture of loRA experts. In *The Twelfth International Conference on Learning Representations*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024a. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Zhaojian Yu, Xin Zhang, Ning Shang, Yangyu Huang, Can Xu, Yishujie Zhao, Wenxiang Hu, and Qiufeng Yin. 2024b. Wavecoder: Widespread and versatile enhanced instruction tuning with refined data generation. *Preprint*, arXiv:2312.14187.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024a. MAmmoTH: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhu Chen. 2024b. Mammoth2: Scaling instructions from the web. *Preprint*, arXiv:2405.03548.

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.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. LIMA: Less is more for alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023b. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. St-moe: Designing stable and transferable sparse expert models. *Preprint*, arXiv:2202.08906.

A Dataset Details

We show the composition of the multi-task training dataset in Table 8.

B Model Creation Details

Our approach begins with a dense model $\mathcal{M}_d$, which we transform into a mixture-of-experts model $\mathcal{M}_s$ through conventional sparse upcycling. The resulting $\mathcal{M}_s$ contains N experts per Transformer block.

In TRSU, we process a training dataset $\mathcal{D}$ containing multiple tasks $\mathcal{T}_i$. For each task, we measure

OpenOrca	Evol Instruct	OSS Instruct	MetaMathQA
200K	100K	50K	200K

Table 8: The composition of the multi-task training dataset.

instance-level perplexity using $\mathcal{M}_d$ and identify the top 5% of instances exhibiting the highest perplexity scores. For each selected instance $\mathcal{T}_{i,j}$, we calculate layer-wise attention outputs per token from $\mathcal{M}_d$, generating task-specific vector sets. These vectors undergo clustering analysis, with the resulting cluster centroids serving as routing weights for the experts within each Transformer block.

In CRSU, we employ a similar methodology but modify the selection criterion. Rather than task-specific sampling, we randomly select 1% of tokens from the aggregated training dataset. We then perform clustering on the layer-wise attention outputs to form N distinct groups. The centroid of each cluster determines the routing weights for the corresponding expert in each Transformer block.

C Evaluation Details

We evaluate our models across diverse tasks spanning mathematical reasoning, commonsense inference, coding ability, instruction following, and domain-specific knowledge. The evaluation framework is supported by Language Model Evaluation Harness (Gao et al., 2023) and Open Instruct (Wang et al., 2023a). Below are the detailed evaluation protocols for each benchmark:

- **GSM8K:** We evaluate mathematical reasoning abilities using the test set of Grade School Math 8K (Cobbe et al., 2021). Using 8 few-shot examples as demonstrations, we report the exact-match accuracy where both the final answer and solution steps must match the reference.

- **HellaSwag:** We assess commonsense inference capabilities using 10-shot examples. This benchmark tests the model's ability to complete situational narratives with plausible endings, focusing on grounded common sense in specific scenarios.

- **HumanEval:** We test Python code generation capabilities using the HumanEval benchmark (Chen et al., 2021). The evaluation uses 0-shot prompting, where models must generate functionally correct Python code based on docstring descriptions. We report unbiased estimates of pass@k and solutions are sampled with a temperature of 0.8.

- **IFEval:** The ability to follow explicit instructions is evaluated using 0-shot examples. We report instruction-level strict accuracy

- **MATH:** We utilize the MATH benchmark (Hendrycks et al., 2021b) to assess advanced mathematical problem-solving capabilities across various topics including algebra, geometry, and calculus. The evaluation employs 4-shot examples, and we report both the solution accuracy.

- **MBPP:** The Multiple Python Programming Problems (MBPP) benchmark (Austin et al., 2021) is used to evaluate Python programming proficiency. Using 0-shot examples, we assess the model's ability to generate code that passes all provided test cases.

- **MMLU:** We measure the multi-task accuracy with 5-shot examples. The results are reported as the average accuracy across all test instances, covering multiple subjects ranging from STEM fields to humanities.

- **MMLU-Pro:** We evaluate models on an enhanced version of MMLU featuring higher-quality and more challenging questions. The assessment includes 5 few-shot examples as in-context demonstrations, and we report the average accuracy across all subjects.

All evaluations are conducted using standardized prompting templates and scoring criteria to ensure consistency and reproducibility. For tasks requiring code execution or mathematical verification, we employ isolated environments to maintain security and deterministic behavior.