MIXTURE-OF-CLUSTERED-EXPERTS: Advancing Expert Specialization and Generalization in Instruction Tuning

Sugyeong Eo[1], **Jungjun Lee**[2], **Chanjun Park**[3*], **Heuiseok Lim**[1*]
[1]Department of Computer Science and Engineering, Korea University, Republic of Korea
[2]KT, Republic of Korea
[3]School of Software, Soongsil University, Republic of Korea
{djtnrud,limhseok}@korea.ac.kr jungjun.lee@kt.com chanjun.park@ssu.ac.kr

Abstract

A sparse Mixture-of-Experts (MoE) architecture has emerged as a highly scalable solution by conditionally activating sub-modules without a proportional increase in computational costs. However, improving expert specialization to enhance performance and generalization remains a challenge for MoE, especially in instruction tuning scenarios characterized by significant input heterogeneity. In this work, we propose the Mixture-of-Clustered-Experts (MoCE) to address this limitation through a dual-stage routing mechanism. The first stage in the mechanism performs expert group routing based on sequence-level features, while the second stage activates the top-k experts within the group at the token level. This approach enables the effective partitioning of heterogeneous inputs based on their knowledge requirements, encouraging expert group specialization while maintaining the advantages of token-level routing. We evaluate MoCE across a comprehensive set of benchmarks, demonstrating its consistent superiority over strong baselines and its enhanced generalization capabilities. Detailed analysis further highlights the robustness and effectiveness of MoCE.

1 Introduction

A sparse Mixture-of-Experts (MoE) has garnered significant attention for its ability to scale parameters with minimal computational overhead (Shazeer et al., 2017; Dai et al., 2024; Jiang et al., 2024; Xue et al., 2024). MoE is a modified version of the Transformer architecture, replacing its feed-forward network (FFN) layer with an MoE layer that consists of multiple FFNs (experts). A gating function conditionally activates the most suitable experts for processing each token, enabling dynamic routing of input tokens among experts. Harnessing the unique characteristics of MoE, its integration with instruction tuning has outperformed dense counterparts, establishing itself as a compelling approach in the field (Komatsuzaki et al., 2023; Zadouri et al., 2024; Ostapenko et al., 2023; Zhu et al., 2024).

However, instruction tuning in the MoE architecture still leaves room for improvement. The input data for instruction tuning covers over a thousand NLP tasks and spans a wide range of domains (Longpre et al., 2023; Peng et al., 2023; Wang et al., 2022c). The inherent heterogeneity of this input data poses challenges to developing expert specialization, where each expert acquires focused knowledge without overlapping with others (Chen et al., 2023; Cai et al., 2024). In the absence of such specialization, overlapping knowledge among experts leads to redundancy, and the complexity of input data forces individual experts to handle dispersed knowledge (Dai et al., 2024).

Furthermore, the routing mechanism in standard MoE models operates exclusively at the token level, where input features derived from sequence-level information, such as domain and task, indirectly influence the routing process. This restricts the model's ability to fully manage the complexity of input data (Kudugunta et al., 2021; Zhu et al., 2024). These limitations highlight the need to distinctly partition and process inputs effectively based on their knowledge requirements to encourage specialization while enhancing generalization performance. Since instruction tuning serves as a regularization technique to alleviate the overfitting issue in MoE fine-tuning (Shen et al., 2024; Dou et al., 2024), the central challenge lies in enhancing the model's generalization capabilities while preserving these benefits.

In this work, we introduce the Mixture-of-Clustered-Experts (MoCE), an extension of the MoE architecture that incorporates a dual-stage routing mechanism leveraging both sequence-level and token-level information. In MoCE, we modify the MoE structure by organizing experts into

*Corresponding author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 14213–14224
November 4-9, 2025

groups, with each group containing multiple experts. MoCE activates experts in two stages: The first stage involves *sequence-level expert group allocation*, where an expert group is selected based on cluster information derived from sequence embeddings. To achieve this, we employ a k-means clustering algorithm to partition inputs, leveraging the latent relationships among data points for more multifaceted segmentation. In the subsequent *token-level expert allocation*, a group-specific gating function selectively activates a subset of experts from the activated expert group at the token level. *This approach enables specific expert groups to process specialized knowledge tailored to distinct input clusters while maintaining fine-grained expert activation optimized for handling individual tokens.*

Additionally, MoCE preserves the computational efficiency of MoE models by keeping the overall number of activated experts unchanged. However, MoCE still inherits a fundamental limitation of the MoE architecture: the VRAM-intensive requirement of loading all experts into memory. We adopt a lightweight adapter-based approach, as introduced by Wu et al. (2024a), making MoCE more practical for real-world applications. We conduct extensive evaluations across diverse benchmarks, including mathematical problem-solving, code generation, reasoning, and knowledge-based tasks. The experimental results show that MoCE demonstrates superior performance not only over general baselines but also over models individually optimized for specific domains. A comprehensive case study and analysis further validate the effectiveness and generalization capabilities of MoCE. Our contributions are threefold:

- We propose the Mixture-of-Clustered-Experts (MoCE) to effectively partition inputs based on their knowledge requirements, inducing expert specialization.
- We ensure efficiency by preserving the number of activated experts and leveraging a fast-converging k-means clustering algorithm for sequence-level routing.
- Extensive experiments and analysis demonstrate the superior performance of MoCE, highlighting its practical potential for a wide range of real-world applications.

2 Related Work

The Sparse Mixture of Experts (MoE) architecture has become a foundational framework for scaling model parameters (Shazeer et al., 2017; Zoph et al., 2022; Ma et al., 2018; Jiang et al., 2024; Raposo et al., 2024). In particular, studies show that instruction tuning in MoE functions as a regularization technique that addresses the overfitting issue and outperforms dense counterparts, establishing it as a promising research direction (Shen et al., 2024; Komatsuzaki et al., 2023; Zadouri et al., 2024; Ostapenko et al., 2023).

With advancements in MoE, an increasing number of studies highlight the pivotal role of expert specialization (Chen et al., 2023; Cai et al., 2024; Shen et al., 2024). In the context of encouraging each expert to acquire distinct knowledge, studies have addressed the challenge of input heterogeneity by targeting specific criteria (Dai et al., 2022; Zhong et al., 2022; Ren et al., 2023; Sarkar et al., 2024; Kudugunta et al., 2021). For instance, Zhao et al. (2023) introduce linguistic-guided routing with language-specific expert allocation, while Gururangan et al. (2022) sparsely activate domain-specific experts depending on the input domain. Dai et al. (2022) define expert assignments for various input modalities. Unlike these approaches that apply explicit routing criteria, our method leverages a clustering algorithm to capture latent input features and partition inputs accordingly.

Additionally, studies have explored routing mechanisms at the sequence or task level (Gou et al., 2023; Kudugunta et al., 2021; Fan et al., 2024). While these studies focus on routing beyond the token level, our work employs a hierarchical routing strategy based on expert grouping. This enables the reflection of sequence-level information while preserving the benefits of selecting the most suitable experts to process individual tokens.

MoE has also been extended to parameter-efficient tuning techniques (Wang et al., 2022b; Diao et al., 2023), including the application of LoRA-based adaptations within the MoE framework (Zadouri et al., 2024; Wu et al., 2024b). Building on this direction, Wu et al. (2024a) presents the adapter-based upcycled model adaptation method. Unlike the LoRA-based approach, this structure minimizes additional memory usage caused by weight merging and enables parallel computation, leading us to adopt it for improved efficiency.

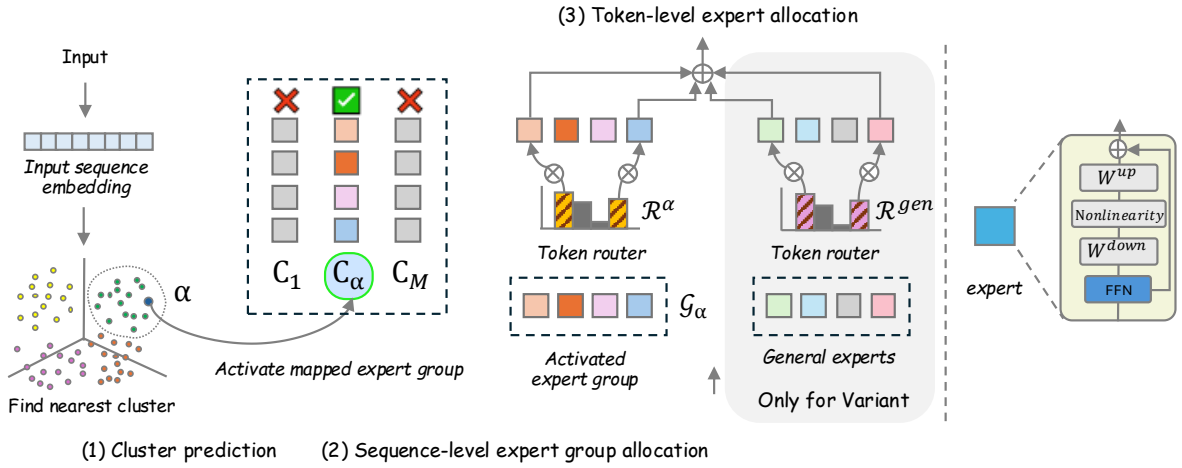


Figure 1: The Overall Architecture of MoCE. MoCE consists of two hierarchical stages: (1) Sequence-level expert group allocation, where an input's sequence embedding is applied to predict a cluster, activating a corresponding expert group $\mathcal{G}^\alpha$ with its gating function $\mathcal{R}^\alpha$. (2) Token-level expert allocation, where $\mathcal{R}^\alpha$ computes routing probabilities for each expert on a per-token basis, selecting the top-k experts to generate adapter outputs via weighted summation. An extended variant includes a general router $\mathcal{R}^{gen}$ and general experts, whose outputs are fused with the adapter outputs to form the final representation.

3 Mixture-of-Clustered-Experts

We propose the Mixture-of-Clustered-Experts (MoCE) architecture, which employs a hierarchical dual-stage routing mechanism that operates at both the sequence and token levels. In this section, we provide a detailed description of the MoCE.

3.1 Preliminaries

The MoE architecture replaces the dense feed-forward network (FFN) sub-layers within the Transformer block with MoE layers. Each MoE layer consists of a set of FFNs, denoted as $\{\mathcal{E}_i\}_{i=1}^N$. A gating function, also referred to as a router $\mathcal{R}(\cdot)$, sparsely activates the experts by routing the intermediate token representation x in the input sequence s to the most appropriate experts.

For the token representation, which is the output of the multi-head attention sub-layer, the router logit $h(x) = W_G^\top \cdot x$ is computed through a linear projection, where $W_G \in \mathbb{R}^{d_{model} \times N}$ denotes a trainable projection matrix. These scores are normalized via a softmax function over the N experts:

$$\mathcal{R}(x)_i = \frac{\exp\left(h(x)_i\right)}{\sum_{j=1}^{N} \exp\left(h(x)_j\right)}, \tag{1}$$

where the output of the gating function serves as the routing weight. The final output y is calculated as a weighted combination of the routing weights and the outputs of the activated experts:

$$y = \sum_{i=1}^{N} \text{TopK}(\mathcal{R}(x)_i, k) \cdot \mathcal{E}_i(x), \tag{2}$$

where the TopK function determines top-k experts to route the token x:

$$\text{TopK}(\mathcal{R}(x)_i, k) = \begin{cases} \mathcal{R}(x)_i & \text{if } i \text{ is in the} \\ & \text{top-}k \text{ of } \mathcal{R}(x), \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

To address the high VRAM demands of MoE, we leverage the Parameter-Efficient Sparsity Crafting (PESC) proposed by Wu et al. (2024a). We initialize model parameters from a pre-trained dense model (Komatsuzaki et al., 2023) and integrate adapters into each FFN. Therefore, the outputs in Equation (2) are modified as $y = \sum_{i=1}^{N} TopK(\mathcal{R}(x)_i, k) \cdot \mathcal{A}_i(x)$, where we denote each adapter $\mathcal{A}_i(x) = \sigma(W_i^{\text{down}} \cdot \mathcal{E}(x)) \cdot W_i^{\text{up}} + x$ as expert.

3.2 Expert Grouping and Clustering

Expert Grouping The MoCE layer is characterized by its grouping of experts, where multiple experts are combined to form an expert group. Each expert group $\mathcal{G}_j$ $(j = 1, ..., M)$ comprises N experts, denoted as $\{\mathcal{A}_i^{\mathcal{G}_j}\}_{i=1}^N$, alongside a group-specific gating function $\mathcal{R}^j$.

Clustering We opt for a k-means clustering mechanism to partition the input according to the features derived from the sequence-level. This allows for the input processing based on multidimensional attributes rather than relying solely on observable features. To enable cluster predictions for inputs during training and inference, we train a clustering model in advance.

Let $\mathcal{C}$ represent the set of clusters and $e \in E_\alpha$ denotes a sequence embedding assigned to cluster α. We utilize an encoder-based embedding model (Wang et al., 2022a) to generate these embeddings, capturing the contextual representation of the entire sequence. The clustering objective J minimizes the L^2 distance between each embedding e and its corresponding centroid μ_α, formulated as $J = \sum_{c=1}^{|\mathcal{C}|} \sum_{e \in E_\alpha} ||e - \mu_\alpha||^2$. Each sequence embedding is assigned to the nearest cluster, and centroids are updated by averaging the embeddings in each cluster, as follows:

$$E_\alpha = \{e : ||e-\mu_\alpha||^2 \leq ||e-\mu_c||^2, \forall c = 1, ..., |\mathcal{C}|\},$$
$$\mu_\alpha = \frac{1}{|E_\alpha|} \sum_{e \in E_\alpha} e. \quad (4)$$

This iterative process continues until cluster assignments converge. Instead of manually setting the cluster count, we use the elbow method to determine the optimal number of clusters. We incrementally increase the cluster count and identify the point where the reduction in the sum of L^2 distances distinctly decelerates. Details about the elbow method are provided in the Appendix B. While k-means clustering is often regarded as outdated, we prioritize its rapid convergence without hindering the efficiency of the MoE. Its linear complexity relative to the number of data points ensures scalability to larger datasets.

3.3 Routing Mechanism

Building upon the grouped expert structure and clustering model, MoCE introduces a hierarchical dual-stage routing mechanism: *sequence-level expert group allocation* followed by *token-level expert allocation*. Figure 1 provides a detailed overview of the MoCE architecture.

Sequence-Level Expert Group Allocation We begin by performing sequence embeddings on the input samples. As noted earlier, we employ an encoder-based embedding model, which operates independently of the MoE's input token representation. This approach addresses the limitations of decoder-based models that rely solely on preceding tokens (BehnamGhader et al., 2024). Each sequence embedding is then assigned a cluster number through a k-means clustering model, which identifies the nearest centroid relative to the input.

We then map the predicted cluster to a corresponding expert group. For example, if the assigned cluster number for input is α, we map this to the expert group $\mathcal{G}_\alpha$. This process involves activating different expert groups based on the distinct characteristics of each cluster, with a one-to-one mapping, i.e., $|\mathcal{C}| = |\mathcal{G}|$. Accordingly, the input is initially routed to the designated expert group corresponding to the assigned cluster number. Note that all token representations in the sequence are routed to their corresponding expert group, and only the group is activated to encourage specialization.

Token-Level Expert Allocation The second stage involves routing individual token representation to specific experts, maintaining the advantage of MoE in selecting experts best suited for processing the current token. This token-level routing follows an equivalent approach to that of standard MoE models. One distinction is that this token-level routing is applied exclusively to the activated expert group. The outputs from the selected experts are computed as follows:

$$y = \sum_{i=1}^{N} \text{TopK}(\mathcal{R}^\alpha(x)_i, k) \cdot \mathcal{A}_i^{\mathcal{G}_\alpha}(x). \quad (5)$$

Selecting top-k experts is mainly based on $k = 2$, with soft merging additionally investigated as a complementary technique. (Zadouri et al., 2024; Muqeeth et al., 2024). In the case of soft merging, the outputs from each adapter are aggregated through a weighted combination, with the weights generated by the group-specific gating function $\mathcal{R}^\alpha$ (Puigcerver et al., 2024). This formulation enables exact gradient computation based on estimated gradients.

3.4 MoCE Variant

We introduce a variant of MoCE that includes additional general experts, $\{\mathcal{A}_j^{gen}\}_{j=1}^{N}$, and a general router, $\mathcal{R}^{gen}$. These processes all sequences, similar to the standard MoE model. As shown in Figure 1, this operates alongside the sentence-level and token-level routing, aiming to integrate knowledge obtained from all inputs. The output from the general experts is combined with the output

Method	Code				Mathematics	
	HumanEval(@1)	HumanEval(@10)	MBPP(@1)	MBPP(@10)	MathQA	GSM8K
Vanilla LLaMA	13.57	17.72	15.57	19.67	28.81	23.35
LLaMA (A)	14.90	18.52	16.50	19.35	28.91	21.91
BTX (A)	13.00	17.52	17.53	20.24	28.71	22.67
PESC	16.00	23.21	20.97	25.77	29.25	33.21
Specialized-Math	14.28	19.46	15.28	21.83	28.17	40.11
Specialized-Code	18.39	24.57	22.91	27.15	29.35	22.37
Specialized-R&K	15.19	20.87	17.73	22.73	29.28	26.54
MoCE-E5	19.28	30.05	22.11	27.69	30.35	**41.93**
MoCE-instructor	20.08	28.48	**23.55**	**28.11**	30.12	36.69
MoCE-instructor(V)	**21.75**	**31.90**	23.12	27.21	**30.75**	37.15

Method	General		Knowledge & Reasoning			Average
	BBH	MMLU-Pro	Winogrande	ARC(Easy)	ARC(Challenge)	
Vanilla LLaMA	36.76	20.02	66.46	69.70	44.28	32.36
LLaMA (A)	36.96	19.03	66.38	68.90	43.52	32.26
BTX (A)	37.13	20.06	66.22	69.61	44.20	32.44
PESC	37.94	20.81	67.56	71.59	45.56	35.62
Specialized-Math	38.29	19.27	67.80	69.49	43.26	34.29
Specialized-Code	37.58	19.55	67.01	70.20	44.97	34.91
Specialized-R&K	38.80	21.00	68.03	70.96	**46.84**	34.34
MoCE-E5	**39.56**	20.21	**68.59**	71.80	46.33	**37.99**
MoCE-instructor	39.09	**21.10**	67.80	**72.26**	46.25	37.61
MoCE-instructor (V)	38.66	20.22	68.43	71.00	46.33	37.87

Table 1: Performance of MoCE across four evaluation categories

from experts activated by the group-specific gating function, which is computed as:

$$y = \sum_{i=1}^{N} \text{TopK}\left(\mathcal{R}^{\alpha}(x)_i, k\right) \cdot \mathcal{A}_i^{\mathcal{G}_\alpha}(x) + \sum_{j=1}^{N} \text{TopK}\left(\mathcal{R}^{gen}(x)_j, k\right) \cdot \mathcal{A}_j^{gen}(x). \quad (6)$$

By combining both outputs, we merge the knowledge acquired from the entire dataset with specialized knowledge.

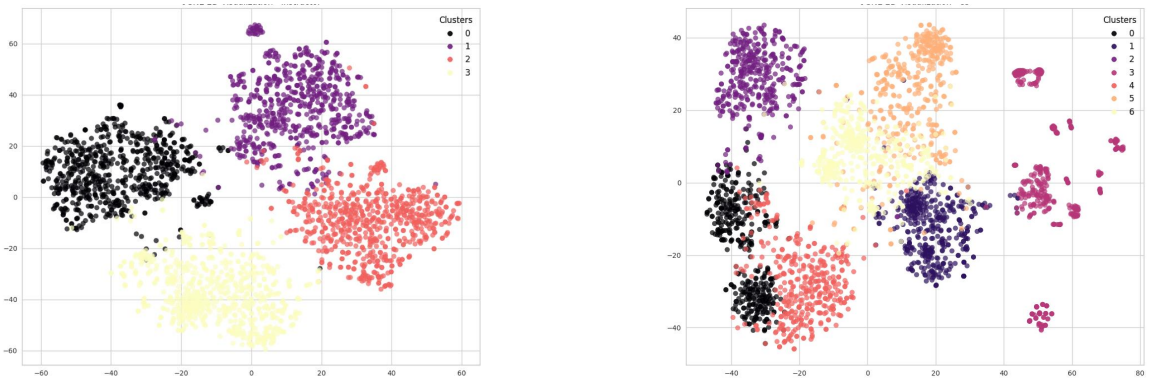
4 Experiments

4.1 Experimental Setup

Datasets. We conduct instruction tuning on MoCE using three distinct datasets. These include SlimOrca (Lian et al., 2023), a curated version of the OpenOrca dataset comprising 518K multi-task examples; Magicoder (Wei et al., 2024), which contains 110K coding problems, providing a rich source for programming and algorithmic tasks; and MetaMathQA (Zhong et al., 2022), consisting of 395K mathematical questions. In total, approximately 1M data points are used to train the models in our experiments.

Models. To mitigate the limitations of decoder-only models, we opt for Instructor (Su et al., 2023) and E5 (Wang et al., 2022a), an encoder-based sequence embedding model. By applying the elbow method, we determine the optimal number of clusters and corresponding expert groups to be four and seven, with each group comprising four experts. For the backbone model, we train the LLaMA2 (Touvron et al., 2023) model with 7 billion parameters, selected for its wide applicability. To prevent expert collapse, we apply a load-balancing loss throughout the experiments. Further implementation details are provided in Appendix A.

Baselines. Given that MoCE adopts adapter-based training, we compare it against the following baselines: (1) LLaMA-Adapter, denoted as LLaMA (A), an adapter-based model designed for efficient parameter adaptation (Houlsby et al., 2019), and (2) BTX (Sukhbaatar et al., 2024), an MoE architecture that integrates the FFNs of independently trained domain-specialized LLMs, and subsequently learns a gating function to route inputs to the appropriate experts. To align with our experimental setting, we utilize its adapter-based implementation, denoted as BTX (A). (4) Parameter Efficient Sparsity Crafting (PESC) (Wu et al., 2024a), which integrates MoE architecture with adapters to enhance computational efficiency. To further evaluate MoCE's



(1) Clustering results using Instructor **(2) Clustering results using E5**

Figure 2: K-means clustering results based on sequence embeddings from Instructor and E5 models

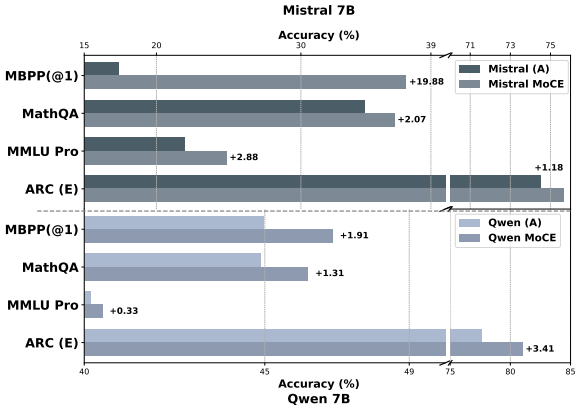


Figure 3: Performance of MoCE applied to Mistral and Qwen language models

ability to balance specialization and generalization, we include domain-specific PESC models as additional baselines, each trained exclusively on tasks from mathematics, code, and reasoning & knowledge domains.

Evaluation Benchmarks. To rigorously assess the effectiveness of MoCE, we employ a comprehensive benchmark spanning four distinct task categories. **Mathematics:** This category evaluates the model's ability to solve multi-step arithmetic and algebraic problems, using MathQA (Amini et al., 2019) and GSM8K (Cobbe et al., 2021). **Code:** Coding proficiency and algorithmic reasoning are assessed through HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021). **Knowledge:** General knowledge and domain-specific understanding are evaluated leveraging BBH (Suzgun et al., 2022) and MMLU-Pro (Wang et al., 2024), which spans a wide range of subjects. **Reasoning:** The model's capacity for commonsense and logical reasoning is measured using Winogrande (Sakaguchi et al., 2021) and ARC (easy and challenge subsets) (Clark et al., 2018). All evaluations are performed under the same settings using lm-evaluation-harness (Gao et al., 2024) and bigcode-evaluation-harness (Ben Allal et al., 2022). The few-shot examples are sourced from lm-evaluation-harness.

5 Results and Discussion

5.1 Main Results

We compare MoCE against eight benchmark datasets spanning four task categories. Figure 2 presents the embedding results, which we detail in Appendix C. As shown in Table 1, MoCE consistently outperforms all baseline models on average. MoCE achieves an average score of 37.99 with MoCE-E5 and 37.61 with MoCE-Instructor, outperforming the baseline models LLaMA (A), which scores 32.26, BTX (A) at 32.44, and PESC at 35.62. MoCE exhibits particularly strong performance in the mathematics and code domains, while also maintaining high accuracy across the majority of evaluated benchmarks. Specifically, MoCE-E5 demonstrates superior performance over PESC, with scores improving from 33.21 to 41.93 on GSM8K and from 16.00 to 19.28 on HumanEval.

Compared to domain-specialized models trained on a single task category, MoCE demonstrates a clear advantage. Although specialized models achieve strong performance within their respective domains, the performance diminishes on out-of-domain tasks. In contrast, MoCE not only matches or surpasses in-domain performance but also generalizes effectively across diverse task categories.

Model	MBPP(@1)	GSM8K	BBH	ARC(E)	Avg
Qwen7B-Chat	37.40	54.12	46.05	63.43	50.25
DeepSeek7B-Chat	39.00	16.60	34.87	70.79	40.32
Mistral7B-Inst	36.09	42.76	45.88	73.40	49.53
QwenMoE	36.60	51.71	41.69	68.31	49.58
DeepSeekMoE	39.20	16.91	33.80	73.15	40.77
Mistral-MoCE	**41.05**	**54.97**	**48.43**	**75.67**	**55.03**

Table 2: Comparative evaluation results with dense and MoE models

These findings reveal that our dual-stage routing mechanism enables expert specialization by successfully managing heterogeneous inputs while preserving the regularization benefits of instruction tuning. Furthermore, the variant of MoCE demonstrates higher average performance compared to the standard MoCE. This indicates that the inclusion of general experts in MoCE improves performance by enabling effective collaboration with specialized experts.

5.2 Application of MoCE Across Different Model Families

As illustrated in Figure 3, MoCE consistently outperforms the baseline across both the Mistral and Qwen model families. Notably, it achieves substantial improvements on domain-specific benchmarks, while also delivering consistent gains in general and reasoning tasks. On average, MoCE improves accuracy by +6.50 points in Mistral 7B and by +1.74 points in Qwen 7B.

These results indicate that MoCE enhances performance across diverse domains without exhibiting signs of overfitting to any particular training domain. We attribute these gains to MoCE's structured routing mechanism, which facilitates targeted expert activation and promotes effective expert specialization, enabling robust adaptation to heterogeneous task distributions. These findings indicate that MoCE serves as a robust and transferable solution across model families beyond LLaMA.

5.3 Evaluating MoCE Against Dense and MoE Models

To assess the practical utility of MoCE, we conduct a comparative analysis with publicly released dense and MoE-based models. Table 2 provides a comparative evaluation of the Mistral-MoCE model with respect to two categories of LLMs: dense models of similar size and MoE-based models. Compared to the dense Mistral 7B model, MoCE achieves substantial performance improvements, increasing GSM8K accuracy from 42.76 to 54.97 and MBPP from 36.09 to 41.05, along with consistent gains on other evaluation benchmarks. Mistral-MoCE achieves consistently superior performance over both Qwen1.5-MoE and DeepSeekMoE, indicating a clear performance margin. These findings suggest that MoCE offers an effective solution for expert specialization and generalization, while retaining computational efficiency.

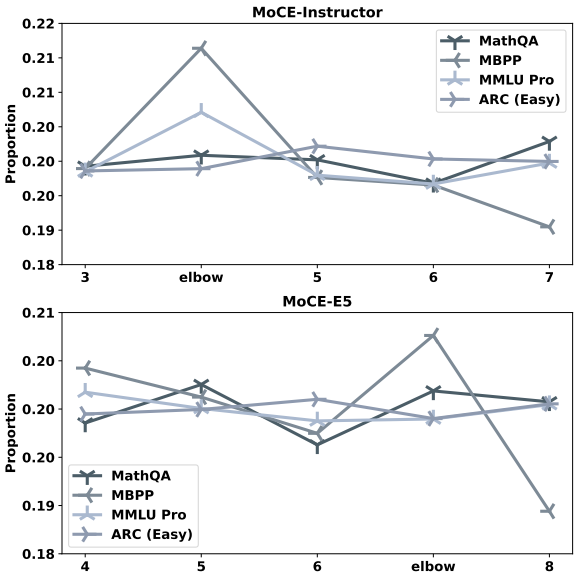


Figure 4: Performance variations across different numbers of expert groups, which correspond to the number of clusters

# Experts	MBPP(@1)	MathQA	MMLU-Pro	ARC-Easy	Average
4 experts (4*1)	20.62	29.65	20.05	71.89	35.55
8 experts (4*2)	21.90	29.61	20.46	71.63	35.90
16 experts (4*4)	**23.55**	**30.12**	**21.10**	**72.26**	**36.76**

Table 3: Comparison of configurations with 1, 2, and 4 experts per cluster in a four-cluster model

5.4 Discussion

The elbow method successfully identifies the optimal number of expert groups. We apply the elbow method to determine the optimal number of clusters for MoCE and further analyze performance variations under different cluster configurations. Figure 4 illustrates the performance trends as the number of clusters changes. Note that the number of clusters is equal to the number of expert groups. The elbow method identifies four and seven as the optimal cluster counts for Instructor and E5 embeddings, respectively, both yielding the best overall performance. These findings validate the elbow method as a reliable approach for selecting

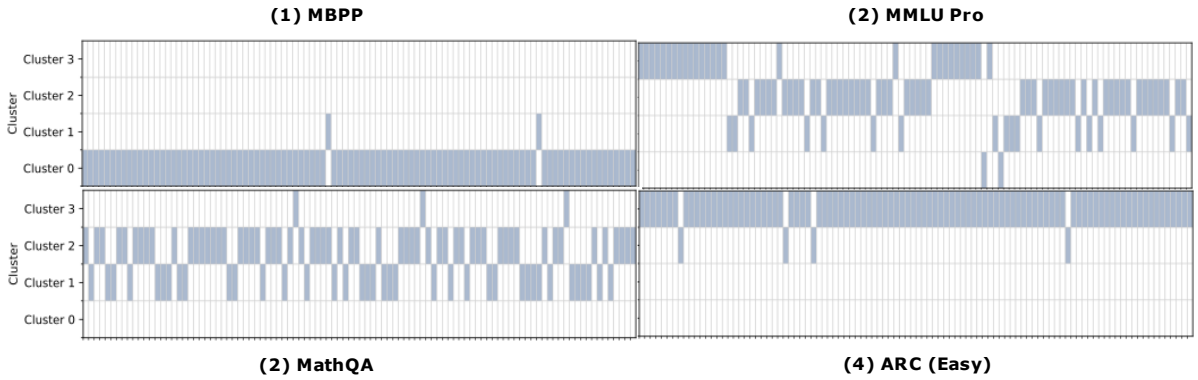


Figure 5: Cluster prediction results on four evaluation benchmarks, based on 100 sampled instances per benchmark. Sequence embeddings are obtained using the Instructor model, and k-means clustering is performed with four clusters.

the optimal number of expert groups.

The number of experts within each expert group exhibits scalability. One of the key advantages of the MoE architecture is its scalability (Zadouri et al., 2024). To evaluate this property, we conduct experiments by progressively increasing the number of experts within each expert group while keeping the number of clusters fixed. Table 3 presents the results with expert counts ranging from 4 (four clusters with one expert each) to 16 (four clusters with four experts each) per cluster. As shown in the table, increasing the number of experts consistently improves model performance. In particular, the average score increases steadily from 35.55 to 36.76 as the expert count scales up, demonstrating that MoCE effectively leverages additional experts to achieve higher performance.

Assigning inputs to distinct expert groups is essential for achieving effective expert specialization. Figure 5 presents the distribution of cluster predictions across the four domains. The results reveal distinct cluster preferences depending on the domain. For example, MBPP inputs are predominantly assigned to cluster 0, while ARC samples are mostly associated with cluster 3. Mathematics examples are largely concentrated in clusters 2 and 3. In the case of MMLU, the cluster predictions are more evenly distributed, which aligns with the dataset's diverse coverage of world knowledge. These findings suggest that expert specialization in MoCE successfully emerges from assigning inputs to specific expert groups based on their sequence-level features.

Method	MBPP(@1)	MathQA	MMLU-Pro	ARC(E)
routing strategy				
w/ Top-2	**23.55**	30.12	**21.10**	72.26
w/ Top-1	22.33	29.75	20.34	70.96
w/ Soft Merging	22.93	**30.39**	20.64	**72.31**
top-k token routing strategy				
MoCE	**23.55**	**30.12**	**21.10**	**72.26**
w/o Clustering	21.72	29.01	20.68	71.13
w/o Token routing	20.62	29.65	20.05	71.63

Table 4: Ablation study on routing strategies in MoCE

Dual-stage routing facilitates effective expert specialization in MoCE. Each component of the MoCE framework plays an important role in enabling effective expert specialization. We validate this via an ablation study shown in Table 4. We begin by analyzing the impact of varying the value of k in the Top-k token routing mechanism. Among the configurations evaluated, Top-2 routing and soft merging yield higher performance. These results suggest that selecting multiple experts per token, rather than relying solely on the most confident single expert, contributes to more effective expert utilization in MoCE.

To gain a deeper understanding of the effectiveness of the hierarchical dual-stage routing structure, we selectively disable each stage. By holding all other variables constant, we isolate the effect of each architectural component and observe that ablating either token-level routing or sequence-level clustering leads to a consistent performance degradation relative to the full dual-stage configuration. For example, MBPP accuracy decreases from 23.55 to 21.72 without clustering and to 20.62 without token routing. The degradation in the sequence-level-only routing setup stems from the lack of token-level granularity, as all inputs are routed to a single

expert based solely on coarse-grained features. In contrast, the token-only setup lacks global contextual signals, limiting the model's ability to capture sequence-level semantics and leading to suboptimal expert assignment. These findings highlight the necessity of a routing hierarchy that combines sequence- and token-level mechanisms to enable effective expert specialization.

6 Conclusion

In this study, we introduce the Mixture-of-Clustered-Experts (MoCE), which incorporates a dual-stage routing mechanism that effectively leverages both sequence-level and token-level features. By incorporating a sequence-level expert group allocation and a token-level expert allocation, MoCE effectively manages inputs while maintaining computational efficiency. Our experimental results consistently show that MoCE outperforms baseline models, highlighting its ability to promote expert specialization and enhance generalization capability. Comprehensive evaluations further validated the effectiveness of our approach, establishing MoCE as a promising framework for managing complex inputs by encouraging expert specialization.

Limitations

While MoCE introduces a novel dual-stage routing mechanism that improves model performance across various tasks, several limitations remain. First, due to computational resource constraints, we applied MoCE to adapter-based models to maintain efficiency. Applying this method directly to the full feed-forward network (FFN) layers for instruction tuning is left as future work.

Second, interpretability is an ongoing challenge in MoE-based architectures. Although we employed k-means clustering, this method still does not address the explainability issues inherent in MoE systems. Even the interpretability of k-means clustering results is limited to a subset of distinctive clusters, such as mathematics and code.

Lastly, while MoCE has demonstrated strong results across tasks in mathematics, coding, and general knowledge, expanding the framework to more languages and specialized domains remains an open area for future work. Investigating its effectiveness in multilingual and domain-specific settings will be essential for broadening the applicability of MoCE.

Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (RS-2024-00398115, Research on the reliability and coherence of outcomes produced by Generative AI) and this research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2021R1A6A1A03045425) and this work was supported by the Commercialization Promotion Agency for R&D Outcomes(COMPA) grant funded by the Korea government(Ministry of Science and ICT)(2710086166).

References

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.

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

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*.

Loubna Ben Allal, Niklas Muennighoff, Logesh Kumar Umapathi, Ben Lipkin, and Leandro von Werra. 2022. A framework for the evaluation of code generation models. `https://github.com/bigcode-project/bigcode-evaluation-harness`.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2024. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Tianlong Chen, Zhenyu Zhang, AJAY KUMAR JAISWAL, Shiwei Liu, and Zhangyang Wang. 2023. Sparse moe as the new dropout: Scaling dense and self-slimmable transformers. In *The Eleventh International Conference on Learning Representations*.

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

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 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. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1280–1297, Bangkok, Thailand. Association for Computational Linguistics.

Yong Dai, Duyu Tang, Liangxin Liu, Minghuan Tan, Cong Zhou, Jingquan Wang, Zhangyin Feng, Fan Zhang, Xueyu Hu, and Shuming Shi. 2022. One model, multiple modalities: A sparsely activated approach for text, sound, image, video and code. *arXiv preprint arXiv:2205.06126*.

Shizhe Diao, Tianyang Xu, Ruijia Xu, Jiawei Wang, and Tong Zhang. 2023. Mixture-of-domain-adapters: Decoupling and injecting domain knowledge to pretrained language models' memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5113–5129, Toronto, Canada. Association for Computational Linguistics.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, et al. 2024. Loramoe: Alleviating world knowledge forgetting in large language models via moe-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1932–1945.

Dongyang Fan, Bettina Messmer, and Martin Jaggi. 2024. TOWARDS AN EMPIRICAL UNDERSTANDING OF MOE DESIGN CHOICES. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Aoxue Li, Dit-Yan Yeung, James T Kwok, and Yu Zhang. 2023. Mixture of cluster-conditional lora experts for vision-language instruction tuning. *arXiv preprint arXiv:2312.12379*.

Suchin Gururangan, Mike Lewis, Ari Holtzman, Noah A. Smith, and Luke Zettlemoyer. 2022. DEMix layers: Disentangling domains for modular language modeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5557–5576, Seattle, United States. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

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*.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021. Beyond distillation: Task-level mixture-of-experts for efficient inference. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3577–3599, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wing Lian, Guan Wang, Bleys Goodson, Eugene Pentland, Austin Cook, and Chanvichet Vong. 2023. Slimorca: An open dataset of gpt-4 augmented flan reasoning traces, with verification. *Slimorca: An open dataset of gpt-4 augmented flan reasoning traces, with verification*, 5.

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. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 22631–22648. PMLR.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1930–1939.

Mohammed Muqeeth, Haokun Liu, and Colin Raffel. 2024. Soft merging of experts with adaptive routing.

Oleksiy Ostapenko, Lucas Caccia, Zhan Su, Nicolas Le Roux, Laurent Charlin, and Alessandro Sordoni. 2023. A case study of instruction tuning with mixture of parameter-efficient experts. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Joan Puigcerver, Carlos Riquelme Ruiz, Basil Mustafa, and Neil Houlsby. 2024. From sparse to soft mixtures of experts. In *The Twelfth International Conference on Learning Representations*.

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 arXiv:2404.02258*.

Xiaozhe Ren, Pingyi Zhou, Xinfan Meng, Xinjing Huang, Yadao Wang, Weichao Wang, Pengfei Li, Xiaoda Zhang, Alexander Podolskiy, Grigory Arshinov, et al. 2023. Pangu-{\Sigma}: Towards trillion parameter language model with sparse heterogeneous computing. *arXiv preprint arXiv:2303.10845*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

Soumajyoti Sarkar, Leonard Lausen, Volkan Cevher, Sheng Zha, Thomas Brox, and George Karypis. 2024. Revisiting smoe language models by evaluating inefficiencies with task specific expert pruning. *arXiv preprint arXiv:2409.01483*.

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*.

Sheng Shen, Le Hou, Yanqi Zhou, Nan Du, Shayne Longpre, Jason Wei, Hyung Won Chung, Barret Zoph, William Fedus, Xinyun Chen, Tu Vu, Yuexin Wu, Wuyang Chen, Albert Webson, Yunxuan Li, Vincent Y Zhao, Hongkun Yu, Kurt Keutzer, Trevor Darrell, and Denny Zhou. 2024. Mixture-of-experts meets instruction tuning: A winning combination for large language models. In *The Twelfth International Conference on Learning Representations*.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada. Association for Computational Linguistics.

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Roziere, Jacob Kahn, Shang-Wen Li, Wen tau Yih, Jason E Weston, and Xian Li. 2024. Branch-train-mix: Mixing expert LLMs into a mixture-of-experts LLM. In *First Conference on Language Modeling*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *Preprint*, arXiv:2210.09261.

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

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022a. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. 2022b. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5744–5760, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022c. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. 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. *Preprint*, arXiv:2406.01574.

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, and Bei Yu. 2024a. Parameter-efficient sparsity crafting from dense to mixture-of-experts for instruction tuning on general tasks. *arXiv preprint arXiv:2401.02731*.

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

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. 2024. Openmoe: An early effort on open mixture-of-experts language models. *arXiv preprint arXiv:2402.01739*.

Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. 2024. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Xinyu Zhao, Xuxi Chen, Yu Cheng, and Tianlong Chen. 2023. Sparse moe with language guided routing for multilingual machine translation. In *The Twelfth International Conference on Learning Representations*.

Tao Zhong, Zhixiang Chi, Li Gu, Yang Wang, Yuanhao Yu, and Jin Tang. 2022. Meta-dmoe: Adapting to domain shift by meta-distillation from mixture-of-experts. *Advances in Neural Information Processing Systems*, 35:22243–22257.

Tong Zhu, Daize Dong, Xiaoye Qu, Jiacheng Ruan, Wenliang Chen, and Yu Cheng. 2024. Dynamic data mixing maximizes instruction tuning for mixture-of-experts. *arXiv preprint arXiv:2406.11256*.

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. *arXiv preprint arXiv:2202.08906*.

A Implementation Details

The hyperparameters used in our experiments are as follows: the adapter dimension is set to 64, with an MoE scaling factor of 1, and the number of experts is set to 4 per cluster. The maximum sequence length is 512, and we apply top-2 expert selection for expert routing. The learning rate is set to 2×10^{-4}, and a batch size of 32 is used. All models are trained for 1 epoch on eight 80G A100 GPUs.

Two embedding models are used for clustering: E5 (`intfloat/e5-large-v2`) and Instructor (`hkunlp/instructor-xl`), both initialized from Huggingface checkpoints. Similarly, three models are used for instruction tuning: LLaMA2 (`meta-llama/Llama-2-7b-chat-hf`), Mistral (`mistralai/Mistral-7B-Instruct-v0`), and Qwen (`Qwen/Qwen2-7B-Instruct`), also initialized from Huggingface checkpoints.

B Details of the Elbow Method

We employ the elbow method to minimize manual configuration and mitigate overfitting when clustering instruction tuning datasets with high variance. The optimal number of clusters k is determined through the following steps:

- Step 1: We iteratively train k-means clustering models, gradually increasing the number of clusters. In our experiment, the number of clusters is incremented from 1 to 10.
- Step 2: We measure and record the within-cluster sum of squared errors (SSE) for each cluster count. As the number of clusters increases, SSE progressively decreases.
- Step 3: We determine the point at which the decrease in SSE clearly begins to plateau. This indicates the elbow point, which is an optimal number of clusters.

C Embedding Results

We present a visualization of the clustering results obtained using both E5 and Instructor embeddings, with dimensionality reduction performed via t-SNE. The resulting visualizations are shown in Figure 2. In both cases, the embeddings exhibit strong convergence around well-defined centroids. The clear separation among clusters suggests that the model effectively captures latent semantic distinctions across input types, facilitating more targeted expert activation. These results provide qualitative support for the expert specialization mechanism in MoCE.