Specialization without Sparsity: Efficient and Expressive Split-Path Experts for LLM Fine-Tuning

Ganghao Liu[1,2], Qin Zhou[1,2]*, Zhe Wang[1,2]*, Xuehan Lu[1,2], Haihua Huang[1,2], Yunfei Tong[1,2], Heng Tian[1,2]

[1]Key Laboratory of Smart Manufacturing in Energy Chemical Process
Ministry of Education, China
[2]Department of Computer Science and Engineering
East China University of Science and Technology, Shanghai, China

Abstract

Parameter-efficient fine-tuning (PEFT) enables low-cost adaptation of large language models but often suffers from limited representational flexibility. To address this, we incorporate a Mixture-of-Experts (MoE) design and propose Efficient and Expressive split-path experts that enhance specialization while maintaining low resource overhead. Split-Path Adaptive Representation Mixture-of-Experts (SparMoE) replaces discrete hard routing with a soft routing and fully-activated mixture, enabling stable optimization. Each expert is parameterized as a split-path modulation module, consisting of a scaling path that promotes expert specialization and a bias path that preserves expert-specific signals. This design significantly enhances expressive capacity while maintaining strict parameter efficiency and architectural compatibility with PEFT. Extensive evaluations on GLUE, GSM8K, MBPP, and a text rewriting task from SmolTalk show that our approach consistently outperforms or matches state-of-the-art PEFT methods under comparable parameter budgets, achieving a favorable trade-off between adaptability and efficiency.

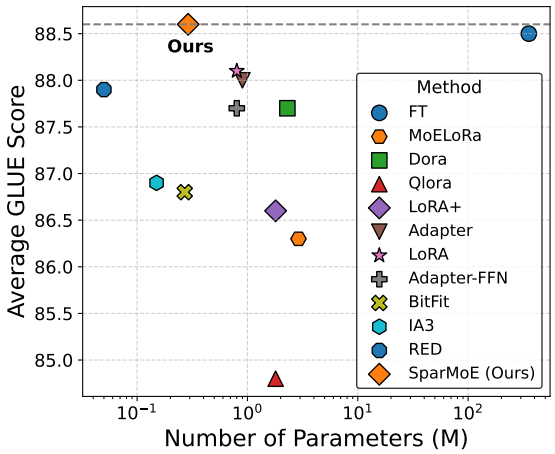

Figure 1: Comparison of SparMoE with other PEFT methods on the GLUE benchmark, using RoBERTa-large as the base model.

1 Introduction

In recent years, artificial intelligence systems have achieved remarkable success in a diverse array of domains, including visual recognition, continual learning, and domain-specific perception tasks (Xia et al., 2025; Yang and Wang, 2025). Large Language Models (LLMs) (Brown et al., 2020; Touvron et al., 2023a) have established a new paradigm in natural language processing, demonstrating state-of-the-art performance across diverse domains(Yang et al., 2025b).

However, as model parameters scale to unprecedented sizes, full fine-tuning becomes computationally prohibitive due to the massive overhead in gradient computation and weight storage (Strubell et al., 2019). Parameter-Efficient Fine-Tuning (PEFT) has emerged as a crucial alternative, adapting pretrained models by updating only a minimal fraction of parameters. Methods such as Adapters (Houlsby et al., 2019) and LoRA (Hu et al., 2021) have become standard practices, significantly reducing storage costs while maintaining competitive performance (Dettmers et al., 2023).

Despite their success, the constrained parameter space of existing PEFT methods often limits their representational capacity, particularly when tackling complex, multi-faceted tasks. To break this bottleneck, recent literature has explored the integration of Mixture-of-Experts (MoE) with PEFT. Contemporary efforts typically employ LoRA-style low-rank modules as experts (Gao et al., 2025; Dou et al., 2024), aiming to combine the parsimony of PEFT with the specialized modeling power of MoEs. However, we argue that a fundamental mismatch exists between conventional MoE formulations and the requirements of parameter-efficient

*Corresponding authors

Findings of the Association for Computational Linguistics: ACL 2026, pages 19515–19527

adaptation. First, LoRA is not inherently designed for multi-expert composition; extending it to MoE settings results in a parameter footprint that scales linearly with the number of experts, rapidly eroding the very efficiency advantage that makes PEFT attractive. Second, the combination of discrete, sparse routing and the use of low-rank experts often induces optimization challenges under data-scarce conditions. This often manifests as undertrained or inactive experts, highlighting that the high-variance nature of discrete selection is ill-suited for the highly restricted parameter space of PEFT. Consequently, such hybrid strategies frequently struggle to reconcile model expressivity with parameter efficiency.

To address these limitations, we propose Split-Path Adaptive Representation Mixture-of-Experts (SparMoE), a lightweight MoE framework explicitly tailored for stable and efficient adaptation. SparMoE departs from the discrete hard routing by employing a soft routing, fully activated mixture mechanism, which ensures stable, end-to-end gradient flow without the need for complex auxiliary load-balancing objectives. To maintain strict parameter efficiency, we introduce a split-path expert design: each expert consists of a scaling path for task-specific specialization and a bias path to preserve essential signals under regularization. By eschewing heavyweight feed-forward experts and unstable hard routing, SparMoE significantly enhances the model's representational power.

Our contributions are summarized as follows:

- We analyze and identify the fundamental limitations of existing PEFT and replication-based MoE methods in parameter- and data-constrained fine-tuning scenarios.
- We propose SparMoE, a lightweight MoE framework that combines a soft routing mechanism with a split-path expert design, enhancing representational power.
- We conduct extensive experiments across multiple LLM backbones, demonstrating that SparMoE achieves superior performance, improved robustness, and higher parameter efficiency compared to existing PEFT approaches.

2 Related Work

PEFT methods strive to adapt large pre-trained models by optimizing only a small subset of parameters, thereby mitigating the computational and memory burdens associated with full-model fine-tuning.

Adapter-based methods insert task-specific modules within Transformer layers. For example, the approaches proposed in (Stickland and Murray, 2019), (Rücklé et al., 2021), and (Mahabadi et al., 2021) introduce compact neural modules between existing layers, confining training efforts exclusively to these inserted components for effective task adaptation. Houlsby et al. (Houlsby et al., 2019) introduced bottleneck adapters using down-projection and up-projection layers. Reparameterization-based techniques optimize in low-rank subspaces, such as LoRA (Hu et al., 2021) which decomposes weight updates into trainable low-rank matrices. QLoRA drastically reduces fine-tuning memory usage. However, its quantization may introduce minor approximation errors, slightly impacting high-precision tasks (Dettmers et al., 2023). LoRA+ also cuts fine-tuning memory footprint (Hayou et al., 2024). Yet, it demands more complex hyperparameter tuning, with task performance sensitive to these choices. DoRA extends basic LoRA, enhancing pre-trained model adaptation to specific tasks (Liu et al., 2024). However the added complexity can lengthen training times and its effectiveness varies by base model architecture. IA^3 (Liu et al., 2022) which learns task-specific scaling vectors. BitFit (Zaken et al., 2022) achieves competitive task adaptation by updating only bias terms, minimizing computational costs while preserving pre-trained model performance. Despite achieving parameter reduction (Ding et al., 2023), these approaches face capacity limitations. He et al. (He et al., 2022) show that improved parameter efficiency may reduce representation diversity, particularly in complex reasoning tasks. Representation Editing (RED) (Wu et al., 2024) emphasizes the adjustment of hidden representations rather than weights, but it does not provide a strategy for handling multi-task scenarios, and the training efficacy of the model remains somewhat limited.

Recent attempts to integrate MoE into PEFT substantially increase both trainable parameters and hyperparameter complexity. Existing approaches typically construct MoE structures atop LoRA-based adaptation (e.g., LoRAMoE (Dou et al., 2024), MoELoRA (Gao et al., 2025; Luo et al., 2024; Liu et al., 2023), MoV (Zadouri et al., 2024)) by replicating LoRA modules as multiple experts. LoRA is

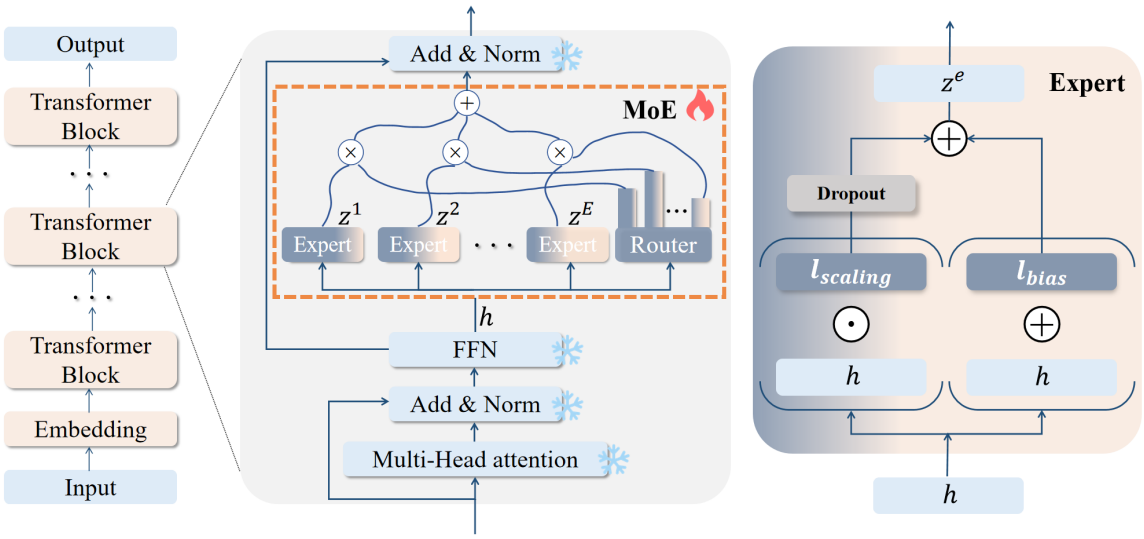

Figure 2: Architecture of the SparMoE module and its integration into a Transformer block. Inserted after the FFN within each Transformer block, SparMoE employs multiple experts, each with two parallel adaptation paths. Expert outputs are aggregated via soft routing, enabling full expert participation. During fine-tuning, only expert-specific parameters and gating weights are updated (backbone frozen).

not inherently designed for multi-expert composition and such designs are inherently inefficient for PEFT. LoRA itself introduces method-specific hyperparameters (low-rank dimension r and scaling factor α). Expanding LoRA to an MoE framework requires an additional hyperparameter for expert count. Moreover, traditional discrete routing mechanisms introduce auxiliary gating components and sensitive hyperparameters (e.g., top-k selection, load-balancing loss coefficients). This complexity is exacerbated in LoRAMoE (Dou et al., 2024), which necessitates explicit distinction between world and task-specific knowledge via dataset labels for expert assignment. This supervised routing not only heavily relies on prior domain knowledge but also restricts autonomous task adaptation, fundamentally deviating from PEFT's core philosophy.

In contrast, SparMoE is designed from a strictly PEFT-oriented perspective. It diverges from existing approaches that rely on replicating heavy adapter matrices (e.g., LoRA experts) or introducing complex routing networks. Instead, SparMoE utilizes a lightweight, vector-based expert structure requiring minimal modulation per expert. By avoiding explicit matrix multiplication within experts and eliminating sparse routing overhead, SparMoE enables high expert diversity and representational flexibility while preserving the training simplicity and computational efficiency of standard PEFT methods.

3 Methodology

3.1 SparMoE Architecture

We propose SparMoE, a parameter-efficient Mixture-of-Experts fine-tuning module designed to enhance adaptation capacity while preserving the lightweight property of PEFT. Unlike sparse MoE with dynamic routing, SparMoE adopts a fully-activated soft routing mechanism, where all experts are evaluated for each token and combined via static softmax weights, without auxiliary load-balancing losses or discrete expert selection.

Let $\mathbf{h} \in \mathbb{R}^{S\times H}$ denote the hidden representation produced by the feed-forward network (FFN), where S is the sequence length and H is the hidden dimension. SparMoE inserts an MoE adaptation layer after the FFN, consisting of E parallel experts. Each expert applies a lightweight *split-path* transformation to $\mathbf{h}$, producing an expert-specific output $\mathbf{z}^{(e)}$:

$$\mathbf{z}^{(e)} = \mathbf{z}^{(e)}_{\text{scaling}} + \mathbf{z}^{(e)}_{\text{bias}}, \quad e = 1, \ldots, E. \tag{1}$$

To aggregate expert outputs, a soft routing function $G : \mathbb{R}^H \to \mathbb{R}^E$ produces token-wise mixture weights:

$$\mathbf{p} = \text{softmax}(G(\mathbf{h})) \in \mathbb{R}^{S\times E}. \tag{2}$$

The final adapted representation is obtained as a weighted sum over all experts:

$$\mathbf{h}' = \sum_{e=1}^{E} \mathbf{p}_{:,e} \odot \mathbf{z}^{(e)}. \tag{3}$$

During fine-tuning, only the expert parameters and the routing function G are updated, while the pretrained backbone remains frozen, ensuring strict parameter efficiency. The aggregated output $\mathbf{h}'$ is subsequently passed to the residual Add & Norm layer.

3.2 Split-Path Expert Design

Each expert in SparMoE employs a split-path structure composed of a *scaling path* and a *bias path*, both parameterized by learnable vectors in $\mathbb{R}^H$. The scaling path modulates feature dimensions multiplicatively and is followed by dropout to induce expert diversity, while the bias path introduces additive expert-specific shifts that remain stable under normalization and stochastic perturbations:

$$\mathbf{z}^{(e)}_{\text{scaling}} = \frac{1}{1-\rho}\left(\mathbf{h} \odot \mathbf{l}^{(e)}_{\text{scaling}} \odot \mathbf{m}\right), \quad (4)$$

$$\mathbf{z}^{(e)}_{\text{bias}} = \mathbf{h} + \mathbf{l}^{(e)}_{\text{bias}}, \quad (5)$$

where $\mathbf{l}^{(e)}_{\text{scaling}}, \mathbf{l}^{(e)}_{\text{bias}} \in \mathbb{R}^H$ are expert-specific parameters, and $\mathbf{m} \sim \text{Bernoulli}(1-\rho)$ is a dropout mask.

The combination of stochastic scaling and deterministic bias enables experts to diverge in both random and structured manners. Specifically, the expected squared distance between the outputs of two experts e and e' can be decomposed as:

$$\mathbb{E}\left\|\mathbf{z}^{(e)} - \mathbf{z}^{(e')}\right\|^2 \approx \mathbb{E}\left\|\mathbf{z}^{(e)}_{\text{scaling}} - \mathbf{z}^{(e')}_{\text{scaling}}\right\|^2 + \left\|\mathbf{l}^{(e)}_{\text{bias}} - \mathbf{l}^{(e')}_{\text{bias}}\right\|^2. \quad (6)$$

where the approximation follows from the independence between dropout-induced noise and bias parameters, as well as the near-zero-mean property of normalized hidden representations. This formulation highlights that expert diversity arises jointly from stochastic scaling perturbations and deterministic bias offsets. Moreover, the bias path provides a robust signal when dropout suppresses the scaling path. For each hidden dimension i, the expert output satisfies:

$$z^{(e)}_i = \begin{cases} \dfrac{z^{(e)}_{\text{scaling},i}}{1-\rho} + z^{(e)}_{\text{bias},i}, & m_i = 1, \\ z^{(e)}_{\text{bias},i}, & m_i = 0, \end{cases} \quad (7)$$

ensuring that expert-specific information is preserved even when multiplicative signals are dropped.

4 Experiments

In this section, we assess the performance of SparMoE across four representative benchmarks spanning diverse tasks: GLUE for natural language understanding, GSM8K for arithmetic reasoning, MBPP for program synthesis, and the explore-instruct-rewriting subset of the SmolTalk dataset. Detailed specifications of these datasets are provided in Appendix A. To strike a pragmatic balance between performance and parameter efficiency, we set the number of experts to 4 for RoBERTa and LLaMA models, whereas 8 experts are employed for Qwen models. All experiments are implemented on NVIDIA 4090 and L20 GPUs.

4.1 Experimental Setup

GLUE. Experiments are conducted on the GLUE benchmark (Wang et al., 2019). RoBERTa-base and RoBERTa-large serve as the backbone models. The training protocol, including the division of validation and test sets and checkpoint selection, follows the guidelines outlined in (Wu et al., 2024).

GSM8K. For evaluating mathematical reasoning capabilities, we utilize GSM8K (Cobbe et al., 2021), a benchmark dataset comprising multi-step arithmetic word problems. The LLaMA2-7B-base model (Touvron et al., 2023b) is fine-tuned on the training split, and evaluation is conducted on the test set using the OpenCompass.

MBPP. We train on the official MBPP (Austin et al., 2021) training split and perform evaluation on the sanitized MBPP test set following the OpenCompass evaluation framework. The underlying base model is LLaMA2-13B-base (Touvron et al., 2023b).

SmolTalk We conduct experiments on the `explore-instruct-rewriting` subset of the SmolTalk dataset (Allal et al., 2025). This dataset consists of high-quality instruction–response pairs focusing on rewriting, paraphrasing, and controlled text transformation tasks. We fine-tune Qwen-series models (Yang et al., 2025a) on the official training split and evaluate performance on the corresponding validation set following the standard evaluation protocol.

4.2 Results on GLUE with RoBERTa

Results on RoBERTa-base are reported in Table 1. With only 0.11M trainable parameters, SparMoE

Method	#param	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
RoBERTa-Base										
FT (Full fine-tuning)	125M	87.3	94.4	87.9	62.4	92.5	**91.7**	78.3	90.6	85.6
MoELoRa	1.7M	84.8	91.3	91.0	62.2	92.6	86.3	76.4	90.9	84.4
DoRA	1.1M	**87.4**	93.8	90.3	58.1	**93.1**	89.1	73.4	91.1	84.5
QLoRA	1.0M	83.1	91.5	91.1	54.5	91.5	76.3	71.9	90.9	81.4
LoRA+	1.0M	86.9	93.6	90.1	55.6	92.7	89.1	74.8	90.6	84.2
MoV	0.71M	85.2	93.6	91.4	–	89.1	86.5	60.7	84.2	–
Adapter	0.4M	87.0	93.3	88.4	60.9	92.5	90.5	76.5	90.5	85.0
LoRA	0.3M	86.6	93.9	88.7	59.7	92.6	90.4	75.3	90.3	84.7
Adapter-FFN	0.3M	87.1	93.0	88.8	58.5	92.0	90.2	77.7	90.4	84.7
BitFit	0.1M	84.7	94.0	88.1	54.0	91.0	87.3	69.8	89.5	82.3
IA3	0.06M	85.4	93.4	86.4	57.8	91.1	88.5	73.5	88.5	83.1
RED	0.02M	83.9	93.9	89.2	61.0	90.7	87.2	78.0	90.4	84.3
SparMoE (Ours)	**0.11M**	86.2	**94.7**	**92.7**	**63.5**	92.4	87.7	**80.3**	**91.4**	**86.1**
RoBERTa-Large										
FT (Full fine-tuning)	355M	88.8	96.0	91.7	68.2	93.8	**91.5**	85.8	**92.6**	88.5
MoELoRa	2.9M	**90.5**	93.6	89.0	61.3	94.4	89.8	83.7	87.8	86.3
DoRA	2.3M	88.4	94.3	92.3	67.4	94.3	90.0	83.2	91.8	87.7
QLoRA	1.8M	89.9	95.4	87.3	58.2	93.6	88.5	75.5	90.1	84.8
LoRA+	1.8M	89.8	95.2	92.3	61.3	94.0	88.3	82.4	89.6	86.6
MoV	1.4M	89.0	93.6	78.4	52.0	–	88.2	–	88.5	–
Adapter	0.9M	90.1	95.2	90.5	65.4	94.6	91.4	85.3	91.5	88.0
LoRA	0.8M	90.2	96.0	89.8	65.5	**94.7**	90.7	**86.3**	91.7	88.1
Adapter-FFN	0.8M	90.3	**96.1**	90.5	64.4	94.3	91.3	84.8	90.2	87.7
BitFit	0.27M	90.0	94.3	91.0	65.9	94.4	87.7	80.9	89.8	86.8
IA3	0.15M	90.1	94.5	87.1	63.2	93.9	89.3	85.3	91.5	86.9
RED	0.05M	89.5	96.0	90.3	68.1	93.5	88.8	86.2	91.3	87.9
SparMoE (Ours)	**0.29M**	90.2	**96.1**	**93.5**	**68.3**	94.4	88.7	85.6	92.2	**88.6**

Table 1: Performance comparison of RoBERTa fine-tuned by SparMoE and other PEFT methods on the GLUE benchmark.

achieves an average GLUE score of 86.1, outperforming all PEFT baselines and surpassing full fine-tuning (85.6) by 0.5 points. Notably, SparMoE attains the best PEFT performance on SST-2 (94.7), MRPC (92.7), CoLA (63.5), RTE (80.3), and STS-B (91.4), covering a diverse range of task types including sentiment analysis, paraphrase identification, linguistic acceptability, and semantic similarity.

We note that MoV exhibit missing results on certain tasks. This is primarily due to the increased parameter and hyperparameter complexity introduced by extending LoRA into an MoE formulation. In practice, these methods require tuning both LoRA-specific hyperparameters (e.g., rank r and scaling factor α) and additional routing-related configurations, which makes stable reproduction across all GLUE tasks challenging. In contrast, SparMoE avoids explicit LoRA replication and gating mechanisms, resulting in more stable and consistent performance across tasks.

Results for RoBERTa-large are also presented in Table 1. Under a larger backbone, SparMoE achieves the highest average GLUE score of 88.6 among all evaluated methods, while introducing only 0.29M trainable parameters, slightly exceeding full fine-tuning (88.5). SparMoE obtains the best performance on SST-2 (96.1), MRPC (93.5), and CoLA (68.3), demonstrating its effectiveness in modeling both semantic relations and syntactic acceptability under parameter-efficient constraints.

Compared with Adapter, LoRA, and QLoRA, SparMoE achieves comparable or superior performance with substantially fewer trainable parameters. This advantage stems from its split-path expert design, which enhances representational diversity without incurring the parameter growth and hyperparameter burden commonly associated with LoRA-based MoE extensions. Meanwhile, ultra-lightweight methods such as IA3 and BitFit, although highly parameter-efficient, consistently underperform across most tasks, indicating a clear

Method	LLaMA2-7B (GSM8K)			LLaMA2-13B (MBPP)		
	#Param	Peak Memory (GB)	Training Time (min)	#Param	Peak Memory (GB)	Training Time (s)
LoRA (r=8)	8.4M	25.3	18.2	–	–	–
LoRA (r=4)	4.2M	19.9	18.2	–	–	–
LoRA	2.1M	16.4	16.2	3.3M	28.5	57.6
DoRA (r=8)	8.9M	25.2	35.7	–	–	–
DoRA (r=4)	4.8M	25.0	35.8	–	–	–
DoRA	2.6M	20.8	35.7	4.1M	46.2	181.9
LoRA+	2.1M	16.4	16.3	3.3M	28.5	57.5
QLoRA	2.1M	**9.8**	49.2	3.3M	**13.7**	202.0
IA^3	1.2M	16.6	**14.6**	1.9M	28.8	**50.7**
SparMoE (Ours)	1.6M	16.3	14.7	2.5M	28.1	57.3

Table 2: Resource efficiency comparison of SparMoE and representative PEFT methods on GSM8K with LLaMA2-7B and MBPP with LLaMA2-13B.

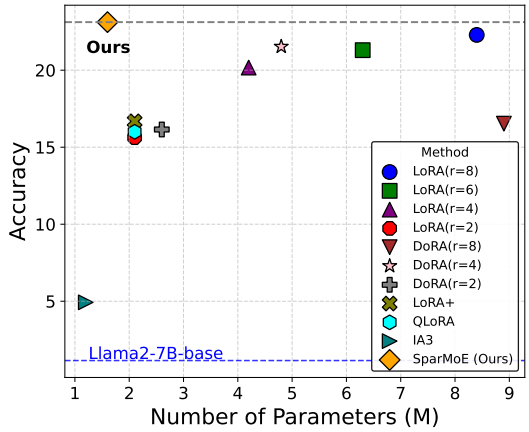

Figure 3: Performance comparison of PEFT methods on GSM8K using LLaMA2-7B.

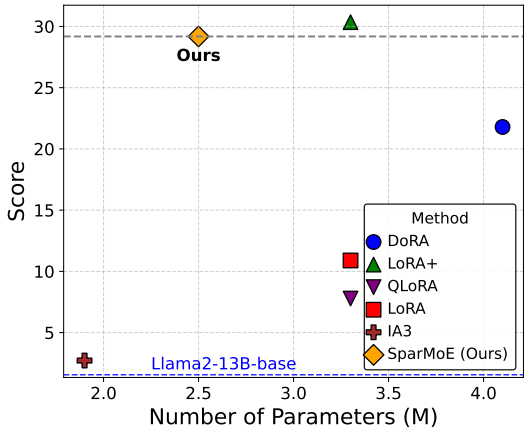

Figure 4: Performance comparison of PEFT methods on MBPP using LLaMA2-13B.

performance ceiling.

4.3 Experimental Results with LLaMA2

4.3.1 Performance with LLaMA2

Table 2 and Figures 3–4 report the performance of various PEFT methods on the GSM8K and MBPP benchmarks. Due to GPU resource constraints, we were unable to successfully run experiments with larger values of r or other MoE-based methods, as these configurations consistently resulted in out-of-memory errors. On GSM8K with LLaMA2-7B (Touvron et al., 2023b), SparMoE achieves the highest accuracy of 23.12% using only 1.6M trainable parameters, outperforming the best LoRA (r=8, 22.29%, 8.4M parameters) by 0.8 points while reducing parameter usage by ∼80%. In comparison, DoRA (r=8) and LoRA+ attain 16.53% (8.9M) and 16.68% (2.1M), respectively, and IA3 (1.2M) reaches only 4.93%, highlighting SparMoE's superior efficiency-performance trade-off in complex arithmetic reasoning.

On MBPP with LLaMA2-13B (Touvron et al., 2023b), SparMoE attains 29.18 with 2.5M parameters. While LoRA+ achieves a slightly higher score (30.35) using 3.3M parameters, it introduces additional hyperparameters and tuning complexity, including separate learning rate adjustments for the A and B projection matrices. Other baselines perform substantially worse, confirming that SparMoE's split-path expert design effectively enhances programmatic reasoning with strong parameter efficiency.

4.3.2 Efficiency Analysis

SparMoE demonstrates superior efficiency not only in parameter count but also in memory usage and training time. On GSM8K with LLaMA2-7B, it consumes 16.25 GB of memory. Its training duration (14.7 m) is substantially shorter than LoRA and LoRA+. On MBPP with LLaMA2-13B, SparMoE maintains efficient memory usage (28.05 GB) and achieves the second-best results in both memory footprint and training time (57.3 s). These results confirm that SparMoE effectively

Model	Method	#Param	Ratio	BLEU↑	ROUGE-1↑	ROUGE-2↑	ROUGE-L↑	METEOR↑
Qwen2.5-0.5B-Instruct	SparMoE	**0.51M**	**0.10%**	**4.81**	**24.72**	**9.56**	**19.11**	**31.26**
	LoRA	0.54M	0.11%	3.85	22.90	8.08	17.25	29.30
	MoELoRA	9.8M	1.98%	4.45	23.59	8.95	18.17	30.52
Qwen3-0.6B-Base	SparMoE	**0.69M**	**0.12%**	**4.69**	**24.43**	**10.03**	**19.22**	**29.79**
	LoRA	1.15M	0.19%	3.50	21.57	7.58	16.46	27.46
	MoELoRA	11.24M	1.89%	3.54	22.72	7.85	17.83	26.23
Qwen3-8B-Base	SparMoE	**3.54M**	**0.04%**	**3.13**	**21.36**	**7.90**	**16.54**	**23.19**
	LoRA	3.83M	0.047%	0.99	14.67	2.93	10.68	17.40
Qwen3-14B-Base	SparMoE	**4.92M**	**0.03%**	**7.25**	**32.05**	**16.21**	**27.13**	**32.16**
	LoRA	5.24M	0.036%	3.11	19.90	7.13	15.60	22.97

Table 3: Parameter efficiency and performance comparison of SparMoE, LoRA, and MoELoRA. ↑ indicates that higher values are better.

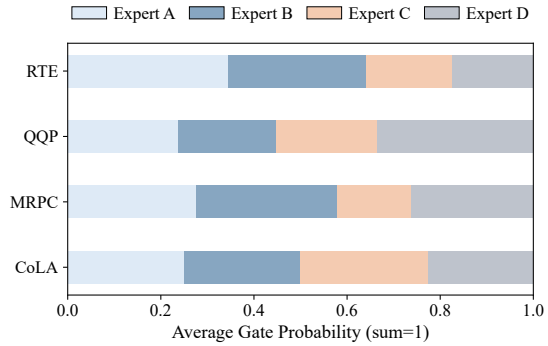

Figure 5: Distribution of expert activations across tasks for SparMoE based on RoBERTa-large.

balances representational power and computational efficiency, positioning it as a state-of-the-art PEFT approach for complex reasoning and programming tasks.

4.4 Experimental Results with Qwen

Table 3 compares SparMoE with representative PEFT methods across various Qwen backbones, including Qwen2.5-0.5B-Instruct and the Qwen3 series (Yang et al., 2025a). Across all models, SparMoE consistently achieves superior generation performance while updating a smaller fraction of parameters, demonstrating strong robustness to architectural variations. On Qwen2.5-0.5B-Instruct, SparMoE uniformly outperforms LoRA under comparable trainable budgets, indicating more effective parameter utilization. Notably, SparMoE also surpasses MoELoRA despite using an order of magnitude fewer trainable parameters, highlighting the limitations of naïvely combining MoE with LoRA. On Qwen3 models ranging from 0.6B to 14B parameters, the performance gap between SparMoE and LoRA further widens as model scale increases, suggesting improved scalability. Even under extremely low trainable ratios, SparMoE consistently delivers substantial gains across BLEU, ROUGE, and METEOR. In conclusion, these results demonstrate that SparMoE consistently outperforms existing PEFT methods, proving highly effective for both the continuous optimization of instruction-tuned models and the elicitation of instruction-following capabilities in base models. Representative generation examples and qualitative case studies are deferred to Appendix C.

4.5 In-depth Analysis

4.5.1 Analysis of Expert Specialization Patterns

To assess expert specialization in SparMoE, we examine the learned expert weight distributions across four GLUE tasks using separately fine-tuned models. As shown in Figure 5, different tasks consistently prioritize distinct experts: RTE relies on Experts A and B for entailment reasoning; QQP assigns higher weight to Expert D for paraphrase detection; MRPC emphasizes Expert B for sentence similarity; and CoLA favors Expert C, reflecting syntactic processing. These results indicate that, even without dynamic routing or task-specific conditioning, SparMoE naturally induces functional specialization, supporting diverse task competencies.

4.5.2 Ablation Study

We conduct ablation studies to validate key SparMoE components (dropout, scaling/bias paths, router) on 8 GLUE tasks with RoBERTa-base (Table 4). Abolishing dropout causes consistent performance drops (e.g., 1.6% SST-2, 1.4% RTE), confirming its role in fostering expert diversity

Method	#Param	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B
SparMoE (full)	0.11M	86.2	94.7	92.7	63.5	92.4	87.7	80.3	91.4
w/o dropout	0.11M	85.7	93.1	89.6	62.2	92.1	86.9	78.9	90.2
w/o scaling	0.07M	86.2	92.9	89.6	60.5	92.4	87.5	77.5	90.7
w/o bias	0.07M	82.6	92.0	88.4	51.0	89.2	85.9	74.0	88.0
w/o router (expert=1)	0.02M	84.2	94.5	90.2	62.3	91.1	85.4	78.4	90.4

Table 4: Results of the ablation studies on different datasets from the GLUE benchmark using the RoBERTa-base.

Model	Method	#Param	Ratio	BLEU$_\uparrow$	ROUGE-1$_\uparrow$	ROUGE-2$_\uparrow$	ROUGE-L$_\uparrow$	METEOR$_\uparrow$
Qwen2.5-0.5B-Instruct	SparMoE (n=4)	258K	0.05%	2.94	20.23	6.71	15.52	23.76
	SparMoE (n=8)	516K	0.10%	4.81	24.72	9.56	19.11	31.26

Table 5: Effect of the number of experts on Qwen2.5-0.5B-Instruct.

#Experts	#Params	CoLA	RTE	MRPC
2	0.06	60.7	79.1	90.7
4	0.11	63.5	80.3	92.7
6	0.16	62.5	80.4	92.6
8	0.22	63.3	80.7	92.8

Table 6: Effect of the number of experts on performance across GLUE tasks.

and mitigating overfitting. Removing the scaling path induces significant losses on low-resource tasks (4.0% CoLA, 2.8% RTE), underscoring task-adaptive activation. Eliminating the bias path leads to severe deterioration (3.6% MNLI, 12.5% CoLA), verifying its necessity for bias signal preservation and representation stabilization. Router removal impairs performance, highlighting expert selection value.

4.5.3 Impact of Expert Number

To analyze the scaling behavior of SparMoE, we vary the number of experts $N_e \in 2, 4, 6, 8$. On the GLUE benchmark (Table 6), increasing N_e from 2 to 4 yields consistent performance gains, particularly on RTE and MRPC, indicating that moderate expert expansion improves task-specific specialization. Further increasing N_e leads to marginal or task-dependent improvements, with non-monotonic behavior observed on CoLA, likely due to limited training data and higher optimization variance. In contrast, on the Qwen2.5-0.5B-Instruct backbone (Table 5), scaling the number of experts from $N_e = 4$ to $N_e = 8$ results in substantial and consistent improvements across all generation metrics, demonstrating that larger expert pools can be beneficial for instruction-tuned generative models.

4.5.4 Hyperparameter Sensitivity Analysis

We evaluate SparMoE's sensitivity to learning rate and dropout on CoLA and MRPC tasks. On CoLA, performance remains stable for moderate learning rates $[0.007, 0.01]$. On MRPC, consistent high scores across a wide range of dropout values (0.01 to 0.5) indicate robust performance without over-masking. Overall, SparMoE exhibits robustness to hyperparameter variations, where moderate dropout promotes expert specialization. Detailed hyperparameter analysis results are presented in Appendix B.

5 Conclusion

We propose SparMoE, a PEFT framework with split-path expert design and soft routing. By incorporating scaling and bias paths, it encourages expert specialization while preserving optimization stability. Its fully activated, lightweight expert modules introduce only a very small number of additional parameters, resulting in a parameter footprint comparable to that of existing PEFT approaches. Extensive evaluations on RoBERTa (for GLUE tasks), LLaMA2 (for math and code reasoning), and Qwen-series models (for instruction-following generation) consistently demonstrate that SparMoE achieves state-of-the-art or competitive performance compared with strong PEFT baselines. Notably, experiments attest to its excellent scalability and architecture agnosticism. Moreover, SparMoE maintains favorable memory efficiency and training speed, rendering it a practical and effective PEFT solution for diverse model families and downstream tasks.

Limitations

This work primarily focuses on the design and evaluation of the SparMoE framework. We have not addressed the forgetting of pre-existing knowledge in instruction-tuned models, nor have we conducted extensive robustness testing. Due to limited resources, experiments were not performed on models larger than 30B. These aspects will be explored in future work.

Acknowledgments

This work was supported by the Natural Science Foundation of China under Grant No. 62476087 and 62306115, Shanghai Municipal Education Commission's Initiative on Artificial Intelligence-Driven Reform of Scientific Research Paradigms and Empowerment of Discipline Leapfrogging, and the Natural Science Foundation of Shanghai under the 2024 Shanghai Action Plan for Science, Technology and Innovation (No. 24ZR1416800).

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, and 3 others. 2025. Smollm2: When smol goes big – data-centric training of a small language model. *Preprint*, arXiv:2502.02737.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Daniel Matthew Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *International Workshop on Semantic Evaluation*.

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. *arXiv preprint arXiv:2110.14168*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *ArXiv*, abs/1809.02922.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *Conference on Neural Information Processing Systems*.

Ning Ding, Yujia Qin, Guang Yang, Fu Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Haitao Zheng, Jianfei Chen, Y. Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5:220–235.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *International Joint Conference on Natural Language Processing*.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, Shiliang Pu, Jiang Zhu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. 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, Bangkok, Thailand. Association for Computational Linguistics.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Ruibo Liu, Baochen Sun, Yawen Zhang, Daiyi Peng, Xiaoyuan Guo, and Vs Subrahmanian. 2025. MoLA: MoE LoRA with layer-wise expert allocation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5097–5112, Albuquerque, New Mexico. Association for Computational Linguistics.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. In *ICML*. OpenReview.net.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*. OpenReview.net.

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.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *NeurIPS*.

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. 2023. When moe meets llms: Parameter efficient fine-tuning for multi-task medical applications. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: Weight-decomposed low-rank adaptation. In *ICML*. OpenReview.net.

Tongxu Luo, Jiahe Lei, Fangyu Lei, Weihao Liu, Shizhu He, Jun Zhao, and Kang Liu. 2024. Moelora: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. *ArXiv*, abs/2402.12851.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. In *Conference on Neural Information Processing Systems*, pages 1022–1035.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. Adapterdrop: On the efficiency of adapters in transformers. In *Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, A. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3645–3650. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023a. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Niko lay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2018. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. In *North American Chapter of the Association for Computational Linguistics*.

Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Jianhao Zhu, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. 2024. Advancing parameter efficiency in fine-tuning via representation editing. In *Annual Meeting of the Association for Computational Linguistics*, pages 13445–13464.

Yu Xia, Subhojyoti Mukherjee, Zhouhang Xie, Junda Wu, Xintong Li, Ryan Aponte, Hanjia Lyu, Joe Barrow, Hongjie Chen, Franck Dernoncourt, Branislav Kveton, Tong Yu, Ruiyi Zhang, Jiuxiang Gu, Nesreen K. Ahmed, Yu Wang, Xiang Chen, Hanieh Deilamsalehy, Sungchul Kim, and 15 others. 2025. From selection to generation: A survey of llm-based active learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 14552–14569. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *ArXiv*, abs/2505.09388.

Hai Yang, Zhenbei Yang, Lei Zhang, Yijing Yang, Dan Zhou, Dongdong Li, Jing Zhang, and Zhe Wang. 2025b. Trans-driver: A deep learning approach for cancer driver gene discovery with multi-omics data. *IEEE Trans. Comput. Biol. Bioinform.*, 22(6):3065–3076.

Mengping Yang and Zhe Wang. 2025. Image synthesis under limited data: A survey and taxonomy. *International Journal of Computer Vision*, 133(6):3689–3726.

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 *ICLR*. OpenReview.net.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Annual Meeting of the Association for Computational Linguistics*, pages 1–9.

A Datasets

A.1 GLUE Benchmark

The General Language Understanding Evaluation benchmark comprises a variety of datasets, including CoLA (Warstadt et al., 2018), SST-2 (Socher et al., 2013), MRPC (Dolan and Brockett, 2005), QQP (Wang et al., 2019), STS-B (Cer et al., 2017), MNLI (Williams et al., 2017), QNLI (Demszky et al., 2018), and RTE (Bar-Haim et al., 2006).The division of the datasets is presented in Table 7.

A.2 GSM8K

The GSM8K benchmark (Cobbe et al., 2021) is a high-quality dataset of school-level math word problems designed to evaluate the arithmetic reasoning capabilities of language models. Each problem is presented in natural language and requires multi-step reasoning to derive the final answer. The dataset is split into 7,473 training examples and 1,319 test examples. GSM8K emphasizes not just numerical computation, but also the ability to interpret and reason through natural language descriptions of mathematical tasks. Our approach is evaluated using the OpenCompass framework (Contributors, 2023), which provides a standardized and reproducible environment for benchmarking large language models across a wide range of tasks. For the GSM8K benchmark, we adopt the built-in evaluation pipeline with the GSM8K_0shot configuration.

A.3 MBPP

The MBPP benchmark (Austin et al., 2021) comprises a set of Python programming problems collected via crowdsourcing. Each task includes a natural language description, a function signature, and multiple test cases, targeting fundamental algorithmic skills suitable for entry-level programmers. In our experiments, models are trained on the full version of the MBPP dataset and evaluated on the sanitized version. Our approach is evaluated using the OpenCompass framework. For the MBPP benchmark, we adopt the sanitized_mbpp_0shot configuration.

A.4 SmolTalk

SmolTalk is a synthetic dataset designed for supervised fine-tuning of large language models, containing 1M instruction-following samples (Allal et al., 2025). It was developed to improve over public SFT datasets, covering diverse tasks including text

editing, rewriting, summarization, and reasoning. During the development of SmolLM2, data ablations at the 1.7B scale guided the incorporation of additional public datasets to enhance capabilities such as mathematics, coding, system prompt compliance, and long-context understanding.

Dataset	#Train	#Validation	#Test	Metric
CoLA	8.5K	522	521	MCC
SST-2	67K	436	436	ACC
MRPC	3.7K	204	204	ACC
QQP	364K	1K	39K	ACC
STS-B	5.7K	750	750	CORR
MNLI	393K	1K	8K	ACC
QNLI	105K	1K	4.5K	ACC
RTE	2.5K	139	138	ACC

Table 7: The sizes of the training, validation, and test sets in the GLUE benchmark.

B Hyperparameter Sensitivity Analysis

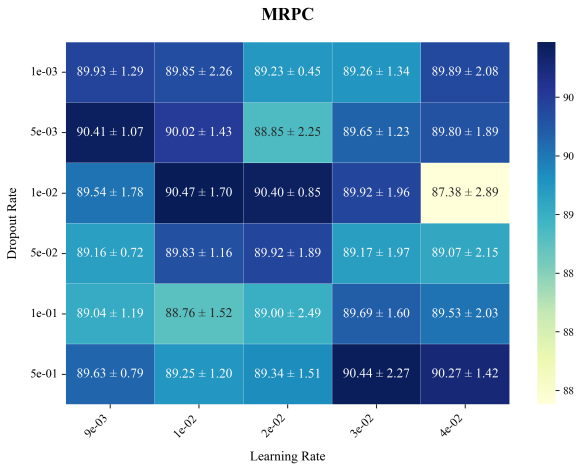

Figure 6: Sensitivity of SparMoE on MRPC with varying learning rates (x-axis) and dropout probabilities (y-axis).

As shown as Figure 7 and Figure 6 We evaluate SparMoE's sensitivity to learning rate and dropout on CoLA and MRPC tasks. On CoLA, performance remains stable for moderate learning rates $[0.007, 0.01]$ and high dropout rates (> 0.1), while a lower dropout rate limits the adaptability to low-resource tasks. The optimal point at $(\text{lr} = 0.009, \text{dropout} = 0.5)$ balances effective feature utilization and stable convergence. On MRPC, consistent high scores across a wide range of dropout values (0.01 to 0.5) indicate robust performance without over-masking. Learning rates between 0.009 and 0.03 show minor effects, demonstrating a wide stable training range. Overall, SparMoE exhibits robustness to hyperparameter variations, where moderate dropout promotes expert specialization.

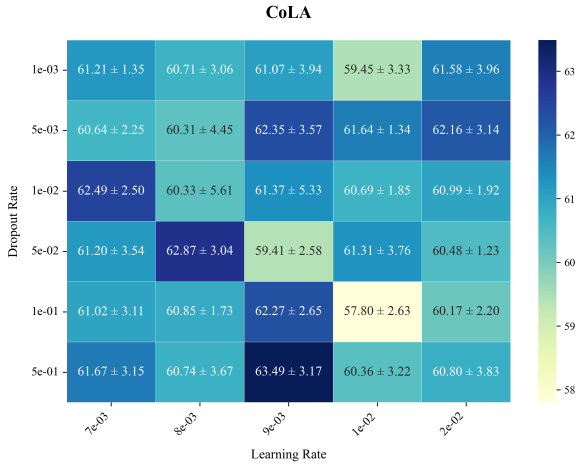

Figure 7: Sensitivity of SparMoE on CoLA with varying learning rates (x-axis) and dropout probabilities (y-axis).

C Case-Based Performance Comparison of Rewriting

As shown as Figure 8, we first examine a sentence rewriting task (starting with a participial phrase): the reference expects a participial phrase opener, while SparMoE produces a clause with redundant "despite that fact" (weaker but retains core meaning), LoRA/MoE-LoRA deviate entirely from the original context (shifting to Sarah's airport or grocery scenarios). For the tense-conversion task: the reference requires full present-tense verbs, SparMoE only adjusts partial details (e.g., "it was out of stock") but keeps most content aligned; LoRA/MoE-LoRA again drift from the original (e.g., adding a flower shop detour or a home return plot). Across both cases, SparMoE consistently maintains the core content and task constraints, unlike LoRA/MoE-LoRA's context deviation. This demonstrates stronger adherence to user instructions, and it outperforms the comparative methods in task alignment and content fidelity.

"prompt": <System>\nYou are an AI rewriting assistant. You will be provided with a text and you need to rewrite it according to the user's instructions.\n</System>\n<User>\nRewrite this sentence to start with a participial phrase: The child, who was already late for school, struggled to tie his shoes:\nThe child struggled to tie his shoes although he was already late for school.\n</User>\n<Assistant>\n

"reference": "Already late for school, the child struggled to tie his shoes."

[SparMoE] Although the child was already late for school, he struggled to tie his shoes despite that fact.

[LoRA] Although she had been waiting in line all morning, Sarah finally made her way through the crowd of people at the airport without any issues. She arrived just as the plane took off, feeling relieved that everything would go smoothly from there on out.

[MoELoRA] Although she had been waiting in line all morning, Sarah finally made her way through the crowd of people at the grocery store. She found what she needed quickly before heading home.

"prompt": "<System>\nYou are an AI rewriting assistant. You rewrite the input text following the user's instructions. Do not include analysis or thinking process, and provide the answer directly.\n</System>\n<User>\nPlease change the verbs in each sentence within this passage to be written in the present tense:\nI went to the store to buy some bread, but they were out. Next, I walked to the bakery down the street. The air smelled sweet and I found myself looking at all the treats in the window. I always go to this bakery because their pastries are the best.\n</User>\n<Assistant>\n",

"reference": "I go to the store to buy some bread, but they are out. Next, I walk to the bakery down the street. The air smells sweet and I find myself looking at all the treats in the window. I always go to this bakery because their pastries are the best."

[SparMoE] I went to the store to buy some bread, but it was out of stock. Next, I walked to the bakery down the street. The air smelled sweet and I found myself looking at all the treats in the window. I always go to this bakery because its pastries are the best.

[LoRA] I headed to the grocery store with my groceries when I realized that there was no bread left. Instead of walking towards a nearby bakery on the corner, however, I decided to take a detour into the neighboring park where I discovered a charming flower shop selling fresh flowers for sale.

[MoELoRA] I went to the store to buy some bread, so it was there when I arrived. Next, I walked to the bakery down the street where the air smelled sweet and I saw a variety of delicious pastries on display. Finally, after enjoying my treat, I returned home with a satisfied smile.

Figure 8: Comparison of rewriting by Qwen2.5-0.5B-Instruct.