GMoE: Empowering LLMs Fine-Tuning via MoE Graph Collaboration

Ting Bai
Beijing University of
Posts and Telecommunication
baiting@bupt.edu.cn

Yue Yu
Beijing University of
Posts and Telecommunication
loadingyy@bupt.edu.cn

Le Huang
Beijing University of
Posts and Telecommunication
lehuang@bupt.edu.cn

Zenan Xu
Tencent AI Lab
zenanxu@tencent.com

Zhe Zhao
Tencent AI Lab
nlpzhaozhe@tencent.com

Chuan Shi
Beijing University of
Posts and Telecommunication
shichuan@bupt.edu.cn

Abstract

The sparse Mixture-of-Experts (MoE) architecture of large language models (LLMs) confronts an inherent issue of load imbalance arising from the simplistic linear router strategy, which ultimately causes the instability and inefficient learning of LLMs. To address this challenge, we introduce a novel MoE graph-based framework **GMoE**, aimed at enhancing the collaboration among multiple experts. In GMoE, a graph router function is designed to capture the collaboration signals among experts. This enables all experts to dynamically allocate information derived from input data by sharing information with their neighboring experts. Moreover, we put forward two coordination strategies in GMoE: the *Poisson distribution-based distinction strategy* and the *Normal distribution-based balance strategy*, to further release the capacity of each expert and increase the model stability in the fine-tuning of LLMs. Specifically, we leverage a parameter-efficient fine-tuning technique, i.e., Low-Rank Adaptation (LoRA), to implement the graph MoE architecture. Extensive experiments on four real-world benchmark datasets demonstrate the effectiveness of GMoE, showing the benefits of facilitating collaborations of multiple experts in LLM fine-tuning. The code of experimental implementation is available at https://github.com/BAI-LAB/GMoE.

1 Introduction

The Mixture-of-Expert (MoE) architecture has emerged as a promising approach to enhance the overall performance of large language models (LLMs) under the scaling law theory. The application of MoE in the fine-tuning of LLMs has drawn significant attention due to the substantial improvements it brings to model performance in practical downstream applications. In a typical MoE setup, a simple linear function, acting as the "router function" or "gating function," assigns weights to each expert based on the information from input tokens, then the top-k experts with largest assigned weights are activated in the fine-tuning process. However, sparse MoE methods face significant challenges that impede their performance and introduce model instability, with a core issue being their linear router strategy that creates severe load imbalance, where a few experts are over-trained while others are underutilized (Fedus et al., 2022).

Recent studies have introduced load balance losses to constrain expert allocation frequencies, thereby alleviating MoE load imbalance (Fedus et al., 2022; Shazeer et al., 2016; Zoph et al., 2022; Luo et al., 2024; Dou et al., 2023a; Wang et al., 2022). However, these approaches primarily focus on regularization-based constraints for load balancing while failing to address a critical limitation: the absence of effective communication and collaborative mechanisms among experts during router allocation. Without expert collaboration, the router relies solely on individual input patterns instead of leveraging collective capabilities. This coordination gap exacerbates inefficiencies: even with balanced allocations, the lack of inter-expert communication prevents dynamic optimization to exploit each expert's unique strengths. As a result, experts remain underutilized, limiting the model's potential and leading to instability during LLM fine-tuning.

These challenges highlight a critical research gap: while load regularization alleviates imbalance, enhancing the stability and efficiency of LLM fine-tuning with MoE requires moving beyond standalone constraints to design architectures that enable effective collaboration among experts. To address this issue, we propose **GMoE**, a novel graph-based MoE framework that explicitly models collaboration among experts. Specifically, GMoE introduces a graph router to encode collaborative signals among experts via a MoE Graph integrating input tokens and expert nodes. Using graph neural networks (GNNs), the router aggregates information iteratively, capturing both token features and inter-expert interactions. This enables experts to jointly process inputs and dynamically share knowledge, leading to more informed, collaborative routing decisions.

To further empower the capability of each expert and enhance their collaboration, we propose a novel coordination strategy: the *Poisson distribution-based distinction strategy*. Specifically, the Poisson distribution loss function promotes specialization by encouraging experts to handle distinct input aspects, thereby stimulating their unique capabilities. Concurrently, the Normal distribution-based balance strategy regulates activation frequencies, naturally balancing the overall workload distribution. This synergistic combination enables experts to discriminatively process inputs while maintaining workload balance, fostering coordinated interaction and enhancing overall model efficiency.

We adopt the parameter-efficient fine-tuning (PEFT) technique, i,e., Low-Rank Adaptation (LoRA), to enable an efficient implementation of our graph MoE architecture for LLM fine-tuning. Through the expert collaboration mechanism in the graph router and two coordination strategies, our GMoE framework achieves state-of-the-art performance and remarkable stability during the fine-tuning of diverse benchmark datasets. Our contributions are summarized as follows:

- We propose a novel GNNs-based MoE framework GMoE to address the instability problem of MoE in LLMs fine-tuning. A graph router assigns weights to experts with consideration of their collaborative interactions on MoE Graph.

- We propose a novel coordination strategies, i.e., the Poisson distribution-based distinction strategy. Along with the Normal distribution-based load balance strategy, our graph based MoE architecture, GMoE, fully unleash the capabilities of individual experts and significantly enhance their collaboration.

- Extensive experiments conducted on four real-world benchmark datasets with three typical base LLMs demonstrate the effectiveness of our GMoE in terms of model **Accuracy**, and **Stability**, showing the benefits of empowering collaborations of multiple experts of LLMs.

2 Related Work

2.1 Mixture-of-Expert (MoE)

Empowered by the collaboration of multiple experts, the Mixture-of-Expert (MoE) framework has achieved remarkable performance in many applications (Chen et al., 2024; Li et al., 2024a; Lin et al., 2024; Li et al., 2024c; Zadouri et al., 2023). Each expert in the MoE framework specializes in handling a subset of input information and different experts coordinate together to gain benefits for the overall MoE framework. According to the number of experts activated by the router function, the MoE methods can be classified into two categories, i.e., dense MoE and sparse MoE (Dou et al., 2023b; Li et al., 2024b). In dense MoE, all experts are activated in the learning process, which enhances the model capability but suffers from high computational costs. In sparse MoE, only a selected subset of experts are activated to use their specialized knowledge and achieve optimal results, thereby reducing computational overhead. The participation of each expert in the computation process is assigned by a router function (or gating function), ensuring an optimal blend of their specialized contributions (Zoph et al., 2022; Tang et al., 2025). Specifically, the router is usually a simple linear function that controls the engagement of expert computations. In the dense MoE, all experts are allocated to computation depending on the weights assigned by the router. In sparse MoE, only Top-K experts with the highest weights are selectively activated in the learning process (Shazeer et al., 2016).

For example, the typical sparse MoE approach Switch Transformers (Fedus et al., 2022) routes only a single expert for the input token to reduce the computation costs, but it faces the load imbalance problem due to lacking collaboration of multiple experts. Existing studies have incorporated auxiliary losses (Bengio et al., 2015; Fedus et al., 2022) to maintain a balanced load among experts in sparse MoE architectures. However, exploring an optimal collaboration mechanism among the activated experts to enhance their load distribution remains an ongoing challenge in this field. In our work, we propose a novel graph router to assign weights to experts with consideration of their collaborative interactions on the MoE Graph, so as to achieve better capability and stability in fine-tuning of LLMs.

2.2 PEFT with MoE

The aim of fine-tuning LLMs is to optimize the model's performance in specific downstream tasks. Due to the high computational costs of updating all parameters in the full fine-tuning approach, the Parameter-Efficient Fine-Tuning (PEFT) technique is proposed to address this problem (Hu et al., 2021; Houlsby et al., 2019; Li and Liang, 2021; Lester et al., 2021; Tian et al., 2024). In PEFT, only a small subset of parameters are updated of the base LLMs, making it more efficient and practical in real applications. The most widely used PEFT technique is LoRA (Hu et al., 2021), which uses a low-rank decomposition technique and updates the decomposed parameter matrix in the model training.

Recent studies have shown improvements of model performance by integrating the MoE framework with PEFT in LLMs fine-tuning literature (Wu et al., 2024; Zadouri et al., 2023). In the transformer block of LLMs, each expert is a LoRA module working on either the FFN layer (Dou et al., 2023b; Wang et al., 2022; Li et al., 2024a), attention layer (Luo et al., 2024; Gou et al., 2023; Zhu et al., 2023), the whole transformer block (Zadouri et al., 2023; Liu et al., 2023; Gao et al., 2024) and each layer (Wu et al., 2024). Then a router function of MoE is designed to blend the contributions of all experts. Among them, the adoption of LoRA in the FFN layer attracted the most attention due to its extensive applicability in various LLM tasks in recent years.

Our work focuses on exploring the effective MoE collaboration mechanism to enhance the PEFT of LLMs in downstream tasks. The MoE consists of multiple LoRA components that work on the FFN layer in the transformer block. Different from the conventional router function employed in existing MoE studies, where the weight assigned to each expert is solely dependent on its own input information, we design a novel graph router, which leverages graph neural networks to learn the collaboration information among all experts based on a MoE graph.

3 Preliminary

3.1 Low-Rank Adaptation (LoRA)

Low-rank adaptation (LoRA) (Hu et al., 2021) is a widely used parameter-efficient fine-tuning (PEFT) technique in pre-trained LLMs. LoRA adopts a low-rank decomposition technique to update the parameters of the decomposed parameter matrix to learn the data distribution of the specific downstream task. It works on the feed-forward layer (FFN layer) in a transformer block of LLMs. For a linear layer in FFN represented as $\mathbf{h} = \mathbf{W}\mathbf{x}$, the update of the decomposed parameter matrix of LoRA is defined as:

$$\mathbf{h} = \mathbf{W}\mathbf{x} + \Delta\mathbf{W}\mathbf{x} = \mathbf{W}\mathbf{x} + \frac{\alpha}{r}\mathbf{B}\mathbf{A}\mathbf{x}, \quad (1)$$

where $\mathbf{x} \in \mathbb{R}^I$ is the representation of input information, and $\mathbf{W} \in \mathbb{R}^{O \times I}$ is the pre-trained parameter matrix of LLMs. $\mathbf{A} \in \mathbb{R}^{r \times I}$ and $\mathbf{B} \in \mathbb{R}^{O \times r}$ are the low-rank matrix in LoRA with $r \ll min(I, O)$. α represents the magnitude of the changes in $\mathbf{W}$. Only decomposed matrices $\mathbf{A}$ and $\mathbf{B}$ are updated in the fine-tuning process.

3.2 Mixture-of-Expert (MoE-LoRA)

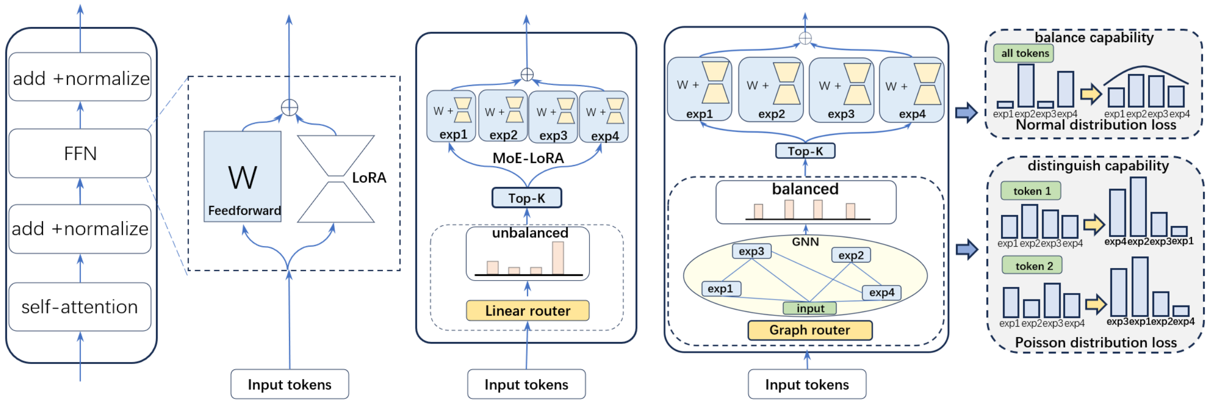
The illustration of MoE-LoRA is shown in Fig. 1 (a) and (b). Each expert network in MoE-LoRA can be represented as a LoRA module worked on the FFN layer in the transformer block. MoE uses a router function to assign the learning weights of each expert in the training process of LLMs. The coordinated weights of all experts are assigned by the router function in the feed-forward process, defined as:

$$\mathbf{h} = \mathbf{W}\mathbf{x} + \Delta\mathbf{W}\mathbf{x} = \mathbf{W}\mathbf{x} + \sum_{i=1}^{N} \mathcal{R}(\mathbf{x})_i \mathcal{E}_i(\mathbf{x}), \quad (2)$$

where $\mathcal{E}_i$ is the ith expert network and N is the number of experts. The router function $\mathcal{R}(\cdot)$ employs a linear function to produce a probability distribution from input information, which serves as the weights allocated to all experts. Following the sparse-gated strategy in MoE studies (Zoph et al., 2022; Li et al., 2024a), which had been proposed to address the computational overhead problem in LLMs. The top-K experts with the highest weights are selectively activated.

4 GMoE

The overview architecture of GMoE is shown in Fig. 1 (d). This section introduces the details of the graph router in GMoE and the propagation process in the FFN layer. Besides, to empower all experts to fully utilize their unique capabilities and keep the load balance, we use two coordination strategies: *load balance strategy* and *expert distinction strategy* to enhance the GMoE capability.



(a) LoRA (b) MoE-LoRA (c) GMoE

Figure 1: The overview of MoE architectures in the FFN layer. (a) The LoRA component is applied in the FFN layer of the transformer block. (b) The typical MoE architecture with LoRA in LLMs with a linear router function to assign weights. (c) Our proposed GMoE architecture with a graph router based on the MoE graph. For different input information (input1 and input2), the distinctive capability of experts is optimized by the Poisson distinction loss. For all input information, the activated frequency of each expert is balanced by the normal distribution loss.

4.1 Graph Router

In typical MoE-LoRA literature, all experts coordinate solely depending on the simple linear router function. The weight assigned to each expert solely depends on the input information, with no explicit collaboration or communication mechanisms among the experts. This may result in the overtraining of only a few experts and under-training of others stemming from its simplistic router strategy. The inefficient collaboration of all experts exacerbates the imbalance load problem of MoE in LLM fine-tuning.

To enhance the collaboration of all experts, we propose a graph router to replace the simple router function. The graph router function in GMoE assigns weights to each expert in collaboration with other experts on the MoE graph. Specifically, given the MoE graph $G = (V, E)$. The node set $V = \{e_1, e_2, ...e_N, x\}$ consists of all expert nodes and the input-token information node x (after the self-attention layer and normalized layer in traditional transformer block). The input node is connected to all the expert nodes to ensure all experts can receive the input information equally. The edges between all expert nodes are randomly constructed and controlled by an edge density hype parameter β. For the input-token node x and its neighborhood expert nodes $N_e(x)$ in the MoE graph, we initialize their features using the commonly adopted Glorot uniform initialization. Subsequently, we use a graph neural network (GNN) to learn their interaction information, which not only implies the learning capability of each expert to the input-token node but also captures the collaboration information among all experts.

The feed-forward process in GMoE can be formulated as:

$$\mathbf{h} = \mathbf{W}\mathbf{x} + \Delta\mathbf{W}\mathbf{x} = \mathbf{W}\mathbf{x} + \sum_{i=1}^{N} \mathcal{R}_{GNN}(\mathbf{x})_i \mathcal{E}_i(\mathbf{x}), \tag{3}$$

where $\mathbf{x}$ is the representation of input node x, $\mathcal{E}_i$ is the ith expert network, and $\mathcal{R}_{GNN}(\cdot)$ is the graph router function in MoE graph, defined as:

$$\mathcal{R}_{GNN}(\mathbf{x})_i = \mathcal{R}(\mathcal{F}(GNN(e_i, N(e_i)))), \tag{4}$$

where $GNN(\cdot)$ is a two-layer graph neural network that learns the representation of expert node e_i using information from its neighbors $N(e_i)$ that contains other experts and input token node x. $\mathcal{F}(\cdot)$ is a projection function, and $\mathcal{R}(\cdot)$ is the Softmax function that assigns probability weights to all experts.

Then we use the Top-K experts in the feed-forward process of the FFN layer, defined as:

$$\mathbf{h} = \mathbf{W}\mathbf{x} + \sum_{i=1}^{K} \textit{Top-K}_{norm}(\mathcal{R}_{GNN}(\mathbf{x})_i \mathcal{E}_i(\mathbf{x})), \tag{5}$$

where the experts with the largest top-k assigned weights from Softmax function $\mathcal{R}$ are selected, and then normalized their weights for summarizing in feed-forward operation.

The router in GMoE takes full advantage of the collaborative information aggregations by GNNs from other expert nodes. This alleviates the load unbalance and localized convergence tendencies caused by a small number of experts activated in the existing MoE studies (Shazeer et al., 2016). Except for introducing a collaboration mechanism of all experts, we use two coordination strategies, i.e., the Poisson distribution-based expert distinction strategy and the normal distribution-based load balance strategy in GMoE, to further release the capacity of each expert and increase the model stability.

4.2 Expert Distinction Strategy

The key aspect of the MoE framework lies in leveraging the unique capabilities of each expert to collaborate effectively, thereby achieving enhanced overall performance. Hence, to elicit the distinct capabilities of different experts, we optimize the assigned weights by the graph router to approximate a Poisson distribution. Specifically, for the output vector of the graph router, denoted as $\mathbf{o}_r \in \mathbb{R}^N$. Each dimension of $\mathbf{o}_r$ represents the assigned weight for each expert to deal with the input information. As the order of experts has no practical meaning, we sort values of each dimension in the vector $\mathbf{o}_r$ in descending order to obtain a new vector $\mathbf{v}_r \Leftrightarrow \textit{sort}(\mathbf{o}_r)$. The distribution of $\mathbf{v}_r \in \mathbb{R}^N$ is optimized to approximate the Poisson distribution vector $\mathbf{v}_{\textit{poisson}(\lambda,i)}$. The Kullback–Leibler (KL) divergence distance is used to calculate the loss function, defined as:

$$\mathbf{v}_{\textit{poisson}(\lambda,i)} = L_{norm}(\frac{\lambda^i}{i!}e^{-\lambda}), \tag{6}$$

$$\textit{Loss-Poisson} = \sum_{i}^{N} \mathbf{v}_{\textit{poisson}(\lambda,i)} \log \frac{\mathbf{v}_{\textit{poisson}(\lambda,i)}}{\mathbf{v}_r}, \tag{7}$$

where $i = \{1, 2..., N\}$ and N is the number of experts. $\mathbf{v}_{\textit{poisson}(\lambda,i)}$ is Poisson distribution vector with L1-normalization operation (i.e., L_{norm}) and λ is the learning parameter.

4.3 Load Balance Strategy

Apart from the distinct capabilities of different experts, one important factor in MoE training is keeping the load balance of all experts. Otherwise, greater routing weights on a small number of experts in the early stages of the fine-tuning process will result in a rapid localized optimization problem. In GMoE, we propose the Normal distribution-based load balance strategy. Specifically, as only Top-K experts are activated in dealing with input information, we calculate the cumulative weights of each expert to represent its activation frequency. Then construct the activation frequency vector of all experts by normalizing the cumulative weights, denoted as $\mathbf{v}_a$ in the next feed-forward step. Our aim is to make the activation frequency vector $\mathbf{v}_a$ follow a natural normal distribution rather than the absolute equality in existing MoE literature (Li et al., 2024a; Zoph et al., 2022). Formally, the Normal distribution-based load balance loss function can be defined as:

$$\mathbf{v}_{\textit{normal}(\mu,\sigma,i)} = L_{norm}(\frac{1}{\sqrt{2\pi\sigma}}\exp(-\frac{(i-u)^2}{2\sigma^2})), \tag{8}$$

$$\textit{Loss-Normal} = \sum_{i}^{N} \mathbf{v}_{\textit{normal}(\mu,\sigma,i)} \log \frac{\mathbf{v}_{\textit{normal}(\mu,\sigma,i)}}{\mathbf{v}_a}, \tag{9}$$

where $i = \{1, 2..., N\}$ and N is the number of experts. $\mathbf{v}_{\textit{normal}(\mu,\sigma,i)}$ is the normalized vector that follows the Normal distribution, $\mu = \frac{N}{2}$ is the mean of all samples, and σ is learning parameter optimized in the fine-tuning process.

5 Experiments

5.1 Experimental Settings

Datasets. To evaluate the performance of our framework, we use four representative public datasets, including the question-answering task, i.e., ARC-Challenge (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018), SIQA (Sap et al., 2019), and task classification task in BoolQ (Clark et al., 2019) dataset. These datasets cover the evaluations on different domains of LLMs, such as factual knowledge from Wikipedia, natural science, science facts, and social interactions.

Baseline Methods. To verify the effectiveness of GMoE, we compare the performance of typical PEFT MoE methods on three popular open-source LLMs, i.e., Llama3-8B[1], Qwen2-7B (Yang et al., 2024), and Yi-1.5-9B (Young et al., 2024).

LoRAMoE (Dou et al., 2023b). It is a representative dense MoE model. A localized balancing con-

[1]https://github.com/meta-llama/llama3

Table 1: The comparisons of model **Accuracy** and **Stability** measured by standard deviation (Std). The smaller the standard deviation, the greater the stability of the model. The underline represents the SOTA baseline method. The best results on Accuracy and Stability are highlighted in bold. For Stability evaluation, the superior outcomes of our approach compared with sparse MoE methods (excluding LoRAMoE) are emphasized in italics.

LLMs	Method	Accuracy Evalution (↑)					Stability Evaluation (Std↓)				
		ARC-C	BoolQ	OBQA	SIQA	Avg.	ARC-C	BoolQ	OBQA	SIQA	Avg.
Llama3	LoRAMoE	76.77	74.48	87.33	79.66	79.56	0.25	0.66	1.01	0.39	0.58
	MING-MoE	77.28	72.95	86.47	79.48	79.05	0.91	0.91	1.42	0.31	0.89
	MoLA	76.74	73.50	84.00	78.70	78.24	0.47	0.34	1.44	0.90	0.79
	MixLoRA	77.36	75.43	87.20	79.53	79.88	0.52	0.99	1.13	**0.15**	0.70
	GMoE	**77.56**	**75.90**	**88.13**	**80.48**	**80.52**	***0.26***	**0.17**	**0.98**	0.37	**0.45**
Qwen2	LoRAMoE	83.50	74.80	90.53	80.54	82.34	0.44	0.03	0.23	1.15	0.46
	MING-MoE	83.99	74.21	91.00	80.67	82.47	1.51	0.62	2.00	1.12	1.31
	MoLA	83.19	74.48	89.93	80.59	82.05	1.34	0.75	1.79	0.83	1.18
	MixLoRA	84.41	74.77	90.00	80.31	82.37	0.69	0.90	0.69	1.90	1.04
	GMoE	**85.10**	**75.40**	**91.67**	**80.98**	**83.29**	***0.48***	***0.57***	***0.64***	**0.24**	**0.48**
Yi-1.5	LoRAMoE	84.33	72.89	91.73	80.94	82.47	3.09	0.38	0.42	0.36	1.06
	MING-MoE	84.58	73.32	90.07	81.56	82.38	0.61	0.58	0.50	0.39	0.52
	MoLA	84.47	72.26	90.07	81.25	82.01	0.61	**0.21**	1.03	0.31	0.54
	MixLoRA	84.36	73.32	**91.80**	81.89	82.84	1.75	0.39	0.40	0.31	0.71
	GMoE	**85.32**	**74.23**	91.33	**82.24**	**83.28**	**0.52**	0.32	**0.40**	**0.28**	**0.38**

straint is used to alleviate the knowledge-forgetting problem in the model updating process.

MING-MoE (Liao et al., 2024). It is a typical sparse MoE architecture without constant loss functions and designed for medical multi-task learning.

MoLA (Gao et al., 2024). It is a recently proposed MoE-based PEFT model that applies different numbers of experts in router functions in different layers. More experts are used at higher layers of the transformer block.

MixLoRA (Li et al., 2024a). It is the SOTA method of PEFT in MoE literature. It adopts the MoE-LoRA architecture in the FFN layer of LLMs. To address the imbalance load problem, it uses an average auxiliary load balance loss.

GMoE. Different from MixLoRA, we propose a novel graph router. It takes advantage of the collaboration of all experts learned by graph neural networks. Besides, two coordination strategies, i.e., the expert distinct strategy and load balance strategy, are proposed to enhance the capabilities of all experts.

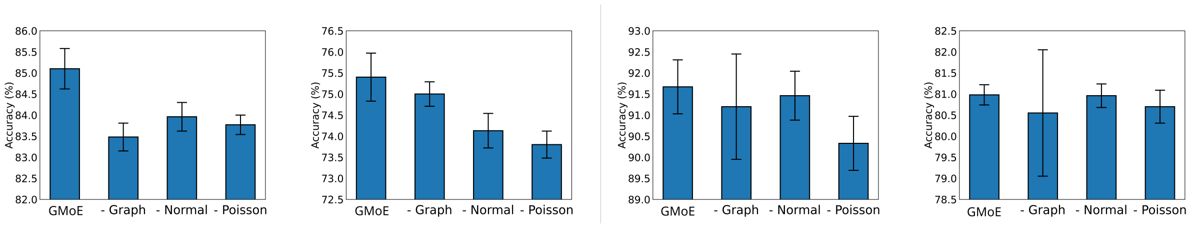
Parameter Settings. All methods are implemented in PyTorch with an A800-80G. For each baseline method, a grid search is applied to find the optimal settings. These include learning rate from $\{0.1, 0.01, 0.001, 0.0001, 0.00001\}$, number of experts from $\{4, 8, 12, 16\}$, rank of LoRA from $\{1, 2, 4, 8, 16\}$, Top-K experts from $\{1, 2, 3, 4, 5\}$. We report the results of each method with its optimal hyperparameter settings on the validation data. In our model, we adopt GCN as the graph neural network (GNN). The number of aggregation layers in GCN is 2, and the hidden dimension of GCN layers is 256. The edge density β to construct the MoE graph is 0.1. The α in Eq. 1 is 4 in the LoRA component. The coefficients of the Poisson distribution loss function and Normal distribution loss function are set to 0.005 and 8 in the final loss function in LLM fine-tuning.

5.2 Main Results

We make comprehensive evaluations of different MoE methods from both **Accuracy** and **Stability** evaluations. We repeat each experiment under five different random seeds, and report the mean value of accuracy. The standard deviation of each model is used to measure the stability of different models. As shown in Table 1. The smaller the standard deviation, the greater the stability of the model. We can see that:

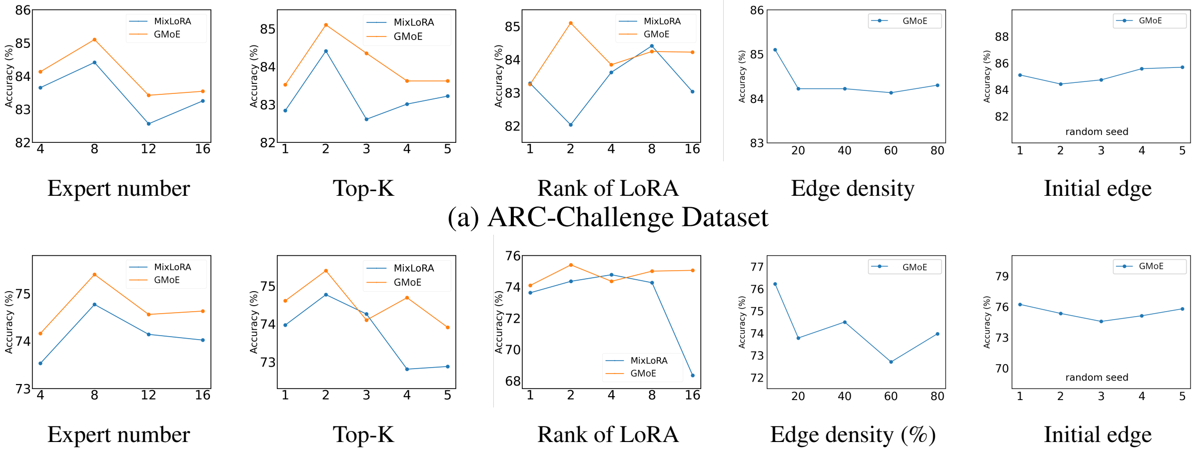
(1) The sparse MoE architecture with an auxiliary load balance loss in MixLoRA performs better than both the sparse MoE (i.e., MING-MoE, MoLA) and dense MoE methods (i.e., LoRAMoE) in many cases, showing the importance of enhancing the coordination of experts.

(2) The performance of the methods varies across different datasets. For example, for the easier task with higher model accuracy in the OpenBookQA dataset, the dense MoE method (i.e., Lo-



(a) ARC-Challenge (b) BoolQ (c) OpenBookQA (d) SIQA

Figure 2: The model performance of degradation variants of GMoE on Qwen2-7B.



(a) ARC-Challenge Dataset

(b) BoolQ Dataset

Figure 3: The hyper-parameters analysis in ARC-Challenge and BoolQ dataset based on Qwen2-7B.

RAMoE) performs better in Llama3 and Yi-1.5. This indicates that localized optimization of sparse MoE may reduce the model's capability and lose its advantages in easy task learning.

(3) GMoE achieves the best performance in most cases on four datasets, showing the effectiveness of the collaborations of multiple experts learned on the MoE graph.

(4) Among sparse MoE methods (MING-MoE, MoLA, MixLoRA), GMoE exhibits the smallest standard deviation in most cases, achieving stability comparable to LoRAMoE, which is a dense MoE activating all experts for balance but with high computational costs.

5.3 Ablation Studies

We perform ablation studies to demonstrate the effectiveness of the graph router and the two coordination strategies used in GMoE. To verify the utility of each component, we individually remove the graph router, Poisson distinct loss (see in Eq. 7), and Normal balance loss (see in Eq. 9). The performance of the degraded variants, namely (-Graph), (-Normal), and (-Poisson), across four datasets is illustrated in Figure 2. We can see that all three components make contributions to the model's performance. The degradation impact of removing the graph router (-Graph) is most obvious in ARC-Challenge and SIQA datasets in both model accuracy and stability aspects, indicating the importance of using the MoE graph to enhance expert collaborations. Moreover, the removal of the Poisson distribution loss (-Poisson) results in the most pronounced decrease in model performance on the BoolQ and OpenBookQA datasets, showing the essential role of maintaining the distinct capability of each expert. The normal distribution loss contributes to keeping the load balance of all experts and substantially improves the model performance.

5.4 Hyper-Parameter Analysis

The performance of GMoE is affected by many hyper-parameters. We conduct analysis experiments to show the effects of hyper-parameters and present the results of ARC-Challenge and BoolQ datasets on Qwen2-7B in Fig. 3.

The number of experts. The number of experts varies from $\{4, 8, 12, 16, 32\}$. We can see that our model achieves the best performance with 8 experts. The increase in the number of experts may

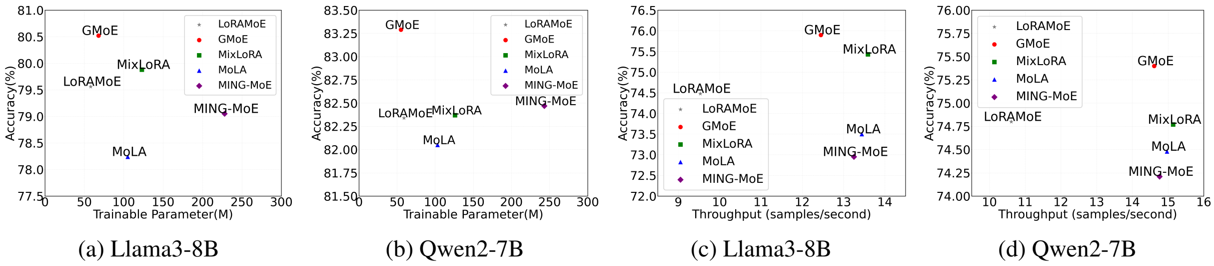


Figure 4: Efficiency comparisons across MoE methods. Illustrated with Trainable parameters and Throughput (samples/second) on the x-axis and average accuracy across four datasets on the y-axis.

potentially complicate the collaboration process of experts and enlarge the imbalance problem.

Top-K experts. The number of experts activated by the router function with the Top-K largest assigned weights. The K varies from $\{1, 2, 3, 4, 5\}$ in our experiments. We find our model performs the best with $K = 2$.

Rank of LoRA. The hyper-parameter rank r in LoRA controls the number of parameters that are updated in the fine-tuning process. The rank r varies from $\{1, 2, 4, 8, 16, 32\}$. Our model with $r = 2$ achieves the best model performance which is less than the rank $r = 4$ and $r = 8$ required in MixLoRA in two datasets.

Edge density. The edge density controls the connection among expert nodes. The edge density is computed as the ratio of the number of edges in the MoE graph compared to the edges in the fully connected graph. The edge density β varies from $\{10\%, 20\%, 40\%, 60\%, 80\%\}$. We find that with only 10% connected edges, the collaboration information shared among experts facilitates our model to achieve the best results.

Initial edge connection. In our experiments, the edges between all expert nodes are randomly constructed. To investigate whether the edge connections among experts in the initial graph affect the model's performance, we repeat experiments under the same edge density 10% with five random edge-connection settings. We can see that the initial connections among expert nodes have minimal influence on the performance of our model. Initially, all expert nodes are considered equal to obtain the input information. As training progresses, each expert node is automatically assigned corresponding weights based on the learning tasks to achieve optimal allocation effects.

5.4.1 Efficiency Analysis

As shown in Fig. 4, we quantify architectural efficiency through the number of **Trainable parameters** and **Throughput (samples/second)**. Trainable parameters can be used as a critical metric for memory and storage costs, and Throughput can serve as a metric for computational speed that reflects the data processing rate during inference stages. These two metrics reflect the computational efficiency of the training and inference stages of our method. We can observe that our GMoE achieves the best model performance with the least number of trainable parameters. This phenomenon is primarily attributed to the much lower rank of LoRA required in our framework. The effective expert cooperation mechanism in our graph router reduces the number of parameters required by each expert, enabling our method to achieve both effectiveness and efficiency. In terms of model throughput, although our model does not have the highest throughput, it is comparable to other models while achieving the best model performance.

5.5 Conclusion

We propose GMoE, a novel framework to address the instability of LLMs caused by MoE load imbalance. GMoE enhances expert collaboration in parameter-efficient fine-tuning through effective information sharing on graph neural networks. GMoE introduces a novel Poisson-based expert distinction strategy to promote expert specialization while employing a normal-based load balance strategy to regulate workload distribution. The comprehensive collaboration among our sparse graph MoE architecture provides a solution to make trade-offs among the model performance, stability, and resource overhead.

5.6 Limitation

We propose GMoE, a graph-based Mixture-of-Experts framework that addresses load imbalance and instability in LLM fine-tuning by enabling expert collaboration via a GNN-powered graph router. Despite its advancements, GMoE has several notable limitations: (1) due to computational resource constraints, the framework's effectiveness and multi-expert balance mechanism have only been validated in downstream task fine-tuning for LLMs with under 10B parameters, leaving their applicability to larger-scale pre-training settings (e.g., 100B+ parameters) untested; (2) while initial random expert connections in the MoE Graph show minimal impact on performance, the current static graph topology does not adaptively optimize expert interactions, potentially limiting the full realization of collaborative potential and requiring future exploration of dynamic structure learning. These gaps highlight opportunities for future research to scale GMoE to larger models, dynamic graph architectures, and expanded application domains.

References

Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. 2015. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*.

Shaoxiang Chen, Zequn Jie, and Lin Ma. 2024. Llava-mole: Sparse mixture of lora experts for mitigating data conflicts in instruction finetuning mllms. *arXiv preprint arXiv:2401.16160*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, pages 2924–2936.

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*.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, and 1 others. 2023a. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, and 1 others. 2023b. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 4(7).

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Baochen Sun, Ruibo Liu, Daiyi Peng, Yawen Zhang, Xiaoyuan Guo, Jie Yang, and VS Subrahmanian. 2024. Higher layers need more lora experts. *arXiv preprint arXiv:2402.08562*.

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*.

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.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

Dengchun Li, Yingzi Ma, Naizheng Wang, Zhiyuan Cheng, Lei Duan, Jie Zuo, Cal Yang, and Mingjie Tang. 2024a. Mixlora: Enhancing large language models fine-tuning with lora based mixture of experts. *arXiv preprint arXiv:2404.15159*.

Jing Li, Zhijie Sun, Xuan He, Li Zeng, Yi Lin, Entong Li, Binfan Zheng, Rongqian Zhao, and Xin Chen. 2024b. Locmoe: A low-overhead moe for large language model training. *arXiv preprint arXiv:2401.13920*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Yunxin Li, Shenyuan Jiang, Baotian Hu, Longyue Wang, Wanqi Zhong, Wenhan Luo, Lin Ma, and Min Zhang. 2024c. Uni-moe: Scaling unified multimodal llms with mixture of experts. *arXiv preprint arXiv:2405.11273*.

Yusheng Liao, Shuyang Jiang, Yu Wang, and Yanfeng Wang. 2024. Ming-moe: Enhancing medical multi-task learning in large language models with sparse mixture of low-rank adapter experts. *arXiv preprint arXiv:2404.09027*.

Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Junwu Zhang, Munan Ning, and Li Yuan. 2024. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*.

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. 2023. Moelora: An moe-based parameter efficient fine-tuning method for multi-task medical applications. *CoRR*.

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 preprint arXiv:2402.12851*.

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

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2016. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Chen Tang, Bo Lv, Zifan Zheng, Bohao Yang, Kun Zhao, Ning Liao, Xiaoxing Wang, Feiyu Xiong, Zhiyu Li, Nayu Liu, and 1 others. 2025. Graphmoe: Amplifying cognitive depth of mixture-of-experts network via introducing self-rethinking mechanism. *arXiv preprint arXiv:2501.07890*.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Cheng-Zhong Xu. 2024. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *Advances in Neural Information Processing Systems*, 37:9565–9584.

Yaqing Wang, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. 2022. Adamix: Mixture-of-adapter for parameter-efficient tuning of large language models. *arXiv preprint arXiv:2205.12410*, 1(2):4.

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

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, and 1 others. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. 2023. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Yun Zhu, Nevan Wichers, Chu-Cheng Lin, Xinyi Wang, Tianlong Chen, Lei Shu, Han Lu, Canoee Liu, Liangchen Luo, Jindong Chen, and 1 others. 2023. Sira: Sparse mixture of low rank adaptation. *arXiv preprint arXiv:2311.09179*.

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*.