Neuronal Redundancy Reduction and Collaboration for Parameter-Efficient Mixture-of-Experts

Anonymous ACL submission

Abstract

Mixture of Experts (MoE) has great potential in scaling up the capacity of large models while maintaining low computational costs. Recent works have focused on reducing expert-level redundancy by designing various token allocation strategies within gating functions. Whereas, the intricate internal relationships between experts cause knowledge redundancy at the fine-grained neuron level, and research on collaboration among experts remains scarce. In this paper, we propose a Information Bottleneck based MoE (IBMoE) for parameter-efficient fine-tuning, which reduces neuron-level redundancy within each expert and fosters internal collaboration among all experts. Specifically, a sparse neuronal activation strategy is introduced to dynamically activate the relevant neurons while reducing the redundancy when processing different tasks. In addition, a diversity constraint is imposed among experts, which maximizes the knowledge difference to enable all experts cooperative more efficiently. Extensive experiments demonstrate the great advantages of our method. We achieve superior performance while reducing inference time by 63% and memory consumption by 48.5% compared to the recent baselines. Our code will be publicly accessible in the future.

1 Introduction

Large language models (LLMs) (OpenAI, 2023; Touvron et al., 2023a; Guo et al., 2025) demonstrate impressive results across various Natural Language Processing (NLP) tasks (Agarwal et al., 2025; Tang et al., 2025), benefiting from their outstanding capabilities in language comprehension and text generation. Nevertheless, the significant computational resources required for training and inference across diverse tasks present ongoing challenges. Thus, various parameter-efficient methods are proposed to reduce the computation costs, such as P-tuning (Liu et al., 2021), Low-Rank Adaption (LoRA) (Hu et al., 2021) and Mixture of Experts (MoE) (Shazeer et al., 2017; Lepikhin et al., 2020; Zadouri et al., 2023; Gao et al., 2024). MoE is a conditional computation framework that divides a dense model into multiple specialized experts. It employs a gating mechanism to dynamically select the relevant experts for computing inputs, which maintains a constant computational cost even as the model scales continuously grow in size.

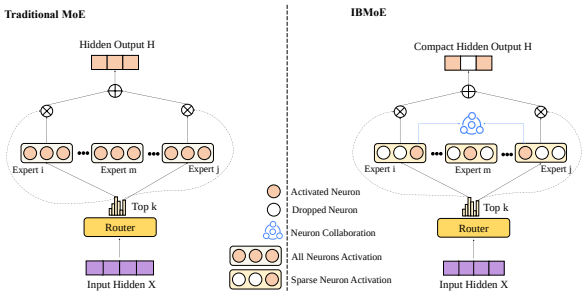

Figure 1: The differences between classical MoE and IBMoE. MoE focuses on expert-level specialization, while IBMoE enhances fine-grained, neuron-level specialization within each expert through sparse neuron activation and neuron collaboration.

One key challenge in MoE is how to make the experts specialized on certain tasks. Liu et al. (2024); Zhang et al. (2024) designed specialized routing mechanisms based on the task features for the gate unit. Zuo et al. (2022) and Zhu et al. (2024b) segmented the weights of dense models to obtain parameter separated experts, and invoke certain experts by measuring the correlation with tasks. However, these efforts struggle to address the issue of knowledge redundancy among experts that emerges during training. From the perspective of expert's training, Dai et al. (2024) devised a shared expert to preserve common knowledge and other experts to focus on specific knowledge. Do et al. (2025) reduced representation similarity to address representational collapse among experts, which helps alleviate knowledge redundancy. Whereas, these methods primarily focus on reducing the expert-

level redundancy, while the internal neuron-level redundancy of each expert remains rarely studied.

Another challenge lies in enabling all experts to collaborate efficiently (Cai et al., 2024). Most existing methods rely on routing mechanisms to activate the relevant experts with different weights to perform tasks collaboratively. For example, when tokens arrive, the router generates a probability distribution indicating each expert's contribution, thereby assigning tokens accordingly. Some approaches also incorporate balance loss to encourage collaborative behavior (Zhu et al., 2023). However, such mechanisms are limited to the expert level, ignoring the cooperation within each expert's internal neurons. Moreover, the knowledge difference among experts remains vague. How to explicitly model these differences for more efficient collaboration is still an open question.

To resolve the above challenges, we propose a parameter-efficient MoE method under the Information Bottleneck framework (Tishby et al., 2000; Alemi et al., 2016), namely IBMoE. As shown in Figure 1, IBMoE integrates sparse neuron activation and collaboration with IB theory, aiming to only retain the most relevant information in a complementary fashion for task completion. For each expert, a learnable query under the IB constraint is set to activate task-related neurons sparsely, compressing hidden representations. In this way, the knowledge characteristics of different experts are explicitly modeled by queries' distribution. Then, a neuron collaboration module is introduced to further diversify knowledge differences across experts, enhancing complementary specialization. This dual mechanism ensures parameter efficiency without sacrificing task performance.

The major contributions of our work can be summarized as follows:

- We propose a parameter-efficient MoE method under the information bottleneck framework, which reduces the neuron-level redundancy within each expert, and fosters internal collaboration among all experts.
- We present a sparse neuron activation mechanism to retain task-specific knowledge while reducing irrelevant information by masking out the unimportant neurons. Additionally, a neuron collaboration module is introduced to explicitly guide all experts to cooperate at the neuron-level with higher efficiency.
- We conduct extensive experiments to validate the effectiveness and superiority of our method. The experiments indicate that IBMoE achieves the state-of-the-art performance across various datasets, while reducing the inference time by 25% and the memory consumption by 48.5%.

2 Related work

2.1 Parameter Efficient Fine-Tuning

Parameter Efficient Fine-Tuning (PEFT) aims to reduce the expensive computational cost of training base models to adapt to various downstream tasks. These methods generally add trainable submodules, called adapters, to the dense model. The model is fine-tuned by updating the adaptor while keeping the base model frozen. Specifically, IA3 (Liu et al., 2022) adds three sets of learnable vectors in the key, value, and linear layers. Llama-Adapter (Zhang et al., 2023) introduces learnable adaptation prompts and zero-init attention to adaptively inject new instruction cues into Llama while effectively retaining its pre-trained knowledge. LoRA (Hu et al., 2021) uses two decomposed low-rank matrices to perform incremental updates. Adakron (Braga et al., 2024) leverages Kronecker products for improved efficiency and expressiveness. MELoRA (Ren et al., 2024) introduces a method that stacks multiple mini-LoRAs in parallel, effectively increasing the overall rank and enhancing performance potential. IBMoE utilizes the concept of PEFT to effectively reduce the training cost of traditional MoE methods while preserving the task specialization advantage of MoE.

2.2 Mixture of Experts

Mixture of Experts (MoE) is a framework that utilizes a gate unit to allocate different inputs to the most relevant expert modules, which reduces computational resources. Early efforts, such as GShard (Lepikhin et al., 2020) and Switch Transformer (Fedus et al., 2022), achieve this by replacing every other MLP layer with MoE layers and allocating tokens to experts using a top-k mechanism. DeepSeekMoE (Dai et al., 2024) introduces shared experts and smaller experts to achieve more fine-grained specialization. Llama-MoE (Zhu et al., 2024b) segments the weights of the dense model to initialize the experts by measuring their relevance to different tasks. MoLoRA (Zadouri et al., 2023) splits the LoRA adapters into multiple fine-grained

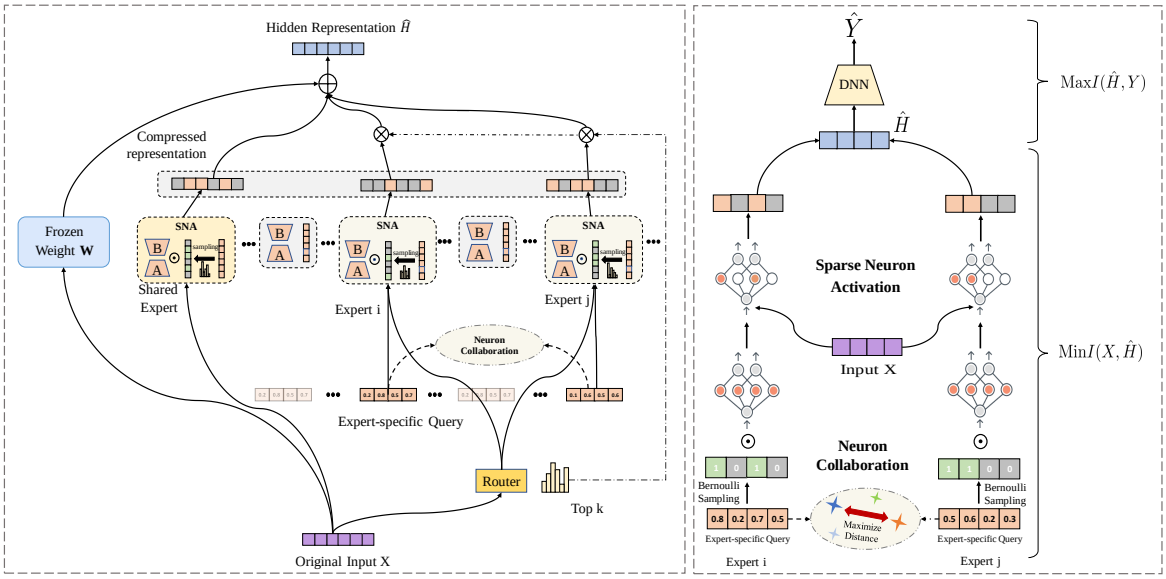

Figure 2: The overall framework of IBMoE consists of two components: Sparse Neuron Activation (SNA) and Neuron Collaboration (NC). SNA utilizes the IB to learn a neuron-level query vector for discarding irrelevant details. The NC strategy is then introduced to reduce redundant knowledge by modeling interactions among experts.

experts, allowing tiny experts to handle different tasks. Further, MoLA (Gao et al., 2024) introduces a method for layer-wise expert allocation to reduce knowledge redundancy. MoR1E (Liu et al., 2024) proposes a task-level routing mechanism to achieve expert specialization in rank-1 LoRA. HMoRA (Liao et al., 2025) introduces hybrid token- and task-level routing mechanism to capture features at varying levels of granularity, which improves expert specialization and generalization. However, research on neuron-level knowledge redundancy is still rare. Our IBMoE effectively measures the knowledge differences among experts from a feature perspective.

2.3 Information Bottleneck

Previous studies have shown that controlling the unnecessary information in the hidden layer through the information bottleneck can effectively control the generalization error in deep learning (Kawaguchi et al., 2023). IB-related work on deep neural networks (DNNs) dates back to 2015, when Tishby et al. demonstrated the feasibility of IB application in deep neural networks using information plane visualization (Tishby and Zaslavsky, 2015). However, optimizing IB in DNNs poses challenges due to the non-convex and intractable of the conventional IB. DIB (Strouse and Schwab, 2017) optimizes the compression term in conventional IB, significantly improving computational efficiency. Currently, several works based on DeepVIB have achieved convincing results in various domains, such as graph representation learning (Yu et al., 2021; Zhang et al., 2025), adversarial training (Chen et al., 2021), network compression (Lorenzen et al., 2021), and representation compression (Sheng et al., 2022), etc. For example, Zhu et al. introduced IB to suppress noise by maximizing the mutual information with the target output while minimizing that with the retrieved content to improve answer accuracy and conciseness by suppressing noise (Zhu et al., 2024a).

3 Preliminaries

Mixture of Experts: In general, the MoE architecture consists of N feed-forward neural networks, denoted as $\{E\}_{i=1}^{N}$, which process various inputs through a gating mechanism. Specifically, the gating mechanism is a function equipped with the trainable parameters $W_r \in \mathbb{R}^{d_{in} \times N}$. For each input $x \in \mathbb{R}^{d_{in} \times 1}$, the gate function maps it to an N-dimensional discrete probability vector, which represents the contribution of each expert. Then, a routing algorithm, such as the top-k or soft routing method, is used to allocate the input to the experts. The output y_i at i-th token is calculated as:

$$G_i = TopK(softmax(W_r^\top x_i)) \quad (1)$$

$$y_i = \sum_{e=1}^{N} G_{i,e} E_e(x_i) \quad (2)$$

where $G_{i,e}$ represents the contribution of e-th expert at i-th token.

Low-Rank Adaptation: Inspired by the natural low intrinsic dimension, LoRA introduces low-rank decomposition for optimizing the weight matrix of pre-trained LLMs to reduce the number of training parameters. Let the weight matrix of a pre-trained linear layer be $W_0 \in \mathbb{R}^{d_{in} \times d_{out}}$. LoRA learns two low-rank matrices $A \in \mathbb{R}^{r \times d_{in}}$ and $B \in \mathbb{R}^{d_{out} \times r}$ to update W_0, where $r \ll \min(d_{in}, d_{out})$. During training, W_0 is frozen and only matrices A and B are updated. Given an input x_i, the output of the LoRA layer can be expressed as:

$$\begin{aligned} h_i &= W_0^\top x_i + \frac{\alpha}{r} BAx_i \\ &= W_0^\top x_i + \triangle W x_i \end{aligned} \tag{3}$$

where α is a scale hyper-parameter.

MoE and LoRA: Some studies integrate the advantages of LoRA and MoE by constructing multiple LoRA adapters as experts, known as LoRA-MoE, to achieve expert specialization on different tasks. Combining Equations 2 and 3, we have:

$$h_i = W_0^\top x_i + \sum_{e=1}^{N} G_{i,e} \triangle W_e x_i \tag{4}$$

where $\triangle W_e$ is the weight matrix of e-th LoRA expert.

4 Our Proposed Approach

In this section, we describe the details of the IB-MoE framework with Sparse Neuron Activation (SNA) and Neuron Collaboration (NC) (Figure 2). Specifically, SNA first introduces the IB into the MoE framework to improve expert specialization. Then, we further design NC to reduce redundant information among multiple experts by maximizing their knowledge diversity.

4.1 Sparse Neuron Activation

Similar to the idea of expert-level sparse activation that only partial knowledge of LLM is utilized to process a task. Not all neurons in an expert contribute equally to a task, so we expect each expert to activate only the most relevant neurons. In this way, SNA not only enhances fine-grained specialization but also captures knowledge characteristics by measuring the distribution differences in neuron activations across experts. Following the mutual information theory of IB. Let the inputs and labels be denoted as X and Y, respectively. The standard objective function is expressed as:

$$\begin{aligned} L_{ib} &= \beta I(X, \hat{H}) - I(\hat{H}, Y) \\ &= \mathop{\mathbb{E}}_{(x,\hat{h}) \sim p(x,\hat{h})} \beta \log \frac{p(\hat{h}|x)}{p(\hat{h})} \\ &\quad - \mathop{\mathbb{E}}_{(\hat{h},y) \sim p(\hat{h},y)} \log \frac{p(y|\hat{h})}{p(y)} \end{aligned} \tag{5}$$

where $\hat{H}$ is the compressed representation of X, $I(.,.)$ denotes mutual information, and β is a hyper-parameter that used to controls the compression of X. The experts are required to discard information from X that is not crucial for their tasks, while retaining sufficient task-related information to predict the label Y.

However, computing $p(\hat{h})$ and $p(y|\hat{h})$ directly is intractable. To solve this, with the concept of Variational Information Bottleneck (VIB) (Alemi et al., 2016), we use two variational distributions $r(\hat{h})$ and $q(y|\hat{h})$ to approximate $p(\hat{h})$ and $p(y|\hat{h})$, which allows us to optimize the upper bound of L_{ib}. Based on the fact that $KL[p(\hat{h})||r(\hat{h})] \geq 0$, we have:

$$\mathop{\mathbb{E}}_{\hat{h} \sim p(\hat{h})} - \log \frac{p(\hat{h})}{r(\hat{h})} \leq 0 \tag{6}$$

For the former term in Equation 5, the upper bound is calculated as:

$$\begin{aligned} \mathop{\mathbb{E}}_{(x,\hat{h}) \sim p(x,\hat{h})} \log \frac{p(\hat{h}|x)}{p(\hat{h})} &\leq \mathop{\mathbb{E}}_{(x,\hat{h}) \sim p(x,\hat{h})} \log \frac{p(\hat{h}|x)}{r(\hat{h})} \\ &\leq \mathop{\mathbb{E}}_{x \sim p(x)} KL[p(\hat{h}|x)||r(\hat{h})] \end{aligned} \tag{7}$$

Similarly, for the latter term in Equation 5 and based on the fact that $KL[p(y|\hat{h})||q(y|\hat{h})] \geq 0$, the lower bound is calculated as:

$$\begin{aligned} \mathop{\mathbb{E}}_{(\hat{h},y) \sim p(\hat{h},y)} \log \frac{p(y|\hat{h})}{p(y)} &\geq \mathop{\mathbb{E}}_{(\hat{h},y) \sim p(\hat{h},y)} \log \frac{q(y|\hat{h})}{p(y)} \\ &\geq \mathop{\mathbb{E}}_{(\hat{h},y) \sim p(\hat{h},y)} \log q(y|\hat{h}) + C \end{aligned} \tag{8}$$

where C is a constant that corresponds to the entropy of Y.

Then, for optimizing Equation 7, we first add a task-specific query $V_i \in \mathbb{R}^{d_{out}}$ for E_i to represent the importance of the neurons in $\triangle W_i$. Before the forward process, E_i uses V_i to generate a binary mask $M_i = [m_1, m_2, ..., m_{d_{out}}]^\top \in \{0, 1\}^{d_{out}}$ via a Bernoulli-based importance sampling, where M_i

indicates which neurons should be dropped. M_i is generated as follows:

$$M_i = Ber(\frac{V_i - \min(V_i)}{\max(V_i) - \min(V_i)}) = Ber(\hat{V}) \tag{9}$$

where $Ber(.)$ is a function used for Bernoulli sampling. Then, the compressed hidden representation can be expressed as:

$$\hat{h_i} = W_0^\top x_i + \sum_{e=1}^{N} \frac{d_{out} G_{i,e} \triangle \hat{W}_e x_i}{sum(M_e)} \tag{10}$$

where $\triangle \hat{W}_e = M_e \circ \triangle W_e$, $\circ$ denotes element-wise multiplication at the column level, and $sum(.)$ is a summation function.

In this paper, Bernoulli sampling is used as the prior distribution, denoted as $r(\hat{h}) = Ber(\pi)$. We aim for the expected number of activated neurons to follow a Bernoulli distribution with a probability of π, which enables controllable compression of invalid information. Compared with h, the computation of $\hat{h}$ additionally depends on the query V of each expert, where V is independent of x. Consequently, we have $p(\hat{h}|x) = p(M|V)$ and $r(\hat{h}) = r(M)$. Therefore, a sparsity constraint is designed to minimize $KL[p(\hat{h}|x)||r(\hat{h})]$:

$$L_s = \sum_{i=1}^{N} [\bar{V_i} \cdot \log \frac{\bar{V_i}}{\pi} + (1 - \bar{V_i}) \cdot \log \frac{1 - \bar{V_i}}{1 - \pi}] \tag{11}$$

where $\bar{V_i}$ represents the mean value of $\hat{V_i}$.

In practical, considering the non-differentiability of Bernoulli sampling during training, we use Gumbel-Softmax reparameterization trick to approximate the Bernoulli distribution (Jang et al., 2017). The sampling process for $\hat{V_i}$ is computed as follows:

$$M_i = \sigma(\frac{log\hat{V_i} + g}{\tau}) \tag{12}$$

where $g = -log(-log(u)), u \sim U(0,1)$, $\tau = 0.5$ is a hyper-parameter for controlling sampling randomization, and $\sigma(.)$ is sigmoid function.

Finally, for Equation 8, $q(y|\hat{h})$ is typically achieved by a network $q_\theta(\hat{h})$ used for the next token prediction task, with the loss function being cross-entropy. The loss function is defined as:

$$L_p = \sum_{x_i \in X} -y_i \log(q_\theta(p_\phi(x_i))) \tag{13}$$

where $p_\phi(.)$ represents a learnable neural network.

4.2 Neuron Collaboration

In this section, we introduce Neuron Collaboration (NC) to enhance the knowledge diversity among experts. Traditional MoE frameworks suffer from knowledge redundancy due to task similarity, but in IBMoE, we tackle this by applying a feature-level diversity loss. It controls the knowledge differences among experts, promoting task specialization. Specifically, a query's distribution represents the knowledge characteristics of an expert. For two queries $\hat{V_i}$ and $\hat{V_j}$, we minimize their similarity, encouraging experts to focus on different feature dimensions through diverse query distributions. The diversity constraint is expressed as:

$$L_d = \sum_{i,j}^{N} cosine(\hat{V_i}, \hat{V_j}) \tag{14}$$

where $cosine(.,.)$ is a function for computing the cosine similarity between two vectors.

We add an auxiliary loss for the router to ensure differentiated token allocation. Let P_r represent the probability distribution of tokens assigned to the experts. The objective function is defined as:

$$L_b = e^{[std(mean_t(P_r)) - mean(std_e(P_r))]} \tag{15}$$

where $mean_t(.)$ represents the token-wise mean value, and $std_e(.)$ represents the expert-wise standard deviation. Combining Equations 11, 13, 14 and 15, the finally loss function is expressed as:

$$L = L_p + \beta L_s + \gamma L_d + \lambda L_b \tag{16}$$

5 Experimental Setup

In this section, we conduct extensive experiments to evaluate the superiority and effectiveness of our method on various NLP tasks. Additionally, we also perform further analyses to investigate the influence of various details in our method.

5.1 Datasets

The General Language Understanding Evaluation (GLUE) is a popular benchmark comprising three types of NLP tasks: single-sentence tasks, similarity and paraphrase tasks, and inference tasks. In this paper, we select one task from each category to evaluate the effectiveness of IBMoE, includes **Recognizing Textual Entailment (RTE)**(Wang et al., 2019), **Corpus of Linguistic Acceptability (CoLA)**(Warstadt et al., 2019), **Microsoft Research Paraphrase Corpus (MRPC)**(Dolan and

Models	Methods	Act Params ↓	Datasets (Acc %↑)						Average
			MRPC	CoLA	RTE	SQA	OBQA	CSQA	
LLaMA	LoRA	19.9M	83.13	86.29	85.92	91.01	75.51	77.00	83.14
	MoLoRA	107.8M	86.13	86.19	87.00	92.03	82.00	79.52	85.47
	MoLA†	43.12M	83.48	86.87	86.28	92.36	78.95	79.60	84.59
	AlphaMoLA†	43.12M	84.23	86.67	87.36	92.71	80.80	78.05	84.97
	AdaMoLE	dynamic	83.88	86.52	87.73	91.00	82.40	78.71	85.04
	HMoRA	43.85M	84.63	86.56	87.36	**93.61**	79.40	79.36	85.15
	IBMoE	7.89M	**87.25**	**87.15**	**88.43**	93.34	**82.40**	**79.77**	**86.39**
Mistral	LoRA	20.50M	85.86	86.23	90.97	94.86	86.2	79.85	87.32
	MoLoRA	111.1M	86.14	87.53	89.16	94.96	86.8	82.15	87.79
	MoLA†	43.40M	86.95	87.44	88.80	95.14	88.20	83.37	88.32
	AlphaMoLA†	43.40M	87.13	**87.91**	91.70	95.00	89.20	**84.00**	89.16
	AdaMoLE	dynamic	85.91	86.10	92.05	94.90	87.80	82.55	88.22
	HMoRA	44.20M	87.48	87.53	90.28	94.60	88.60	82.64	88.52
	IBMoE	8.22M	**88.58**	87.53	**92.77**	**95.68**	**90.40**	83.05	**89.67**

Table 1: Comparison of performance and parameter efficiency with other SOTA methods across different model architectures. Average accuracy across six datasets (**Average**) is reported to evaluate overall performance. Best results are highlighted in **bold**, † denotes results reported by the original methods.

Brockett, 2005). We also use three common Question Answering (QA) tasks to evaluate our IBMoE, includes **ScienceQA (SQA)**(Lu et al., 2022), **OpenbookQA (OBQA)**(Mihaylov et al., 2018), **CommonSenseQA (CSQA)**(Talmor et al., 2019).

5.2 Base models and Baselines

In the experiment, we use LLaMA2-7B (Touvron et al., 2023b) and Mistral-7B-v0.1 (Jiang et al., 2023) models as our base models. To evaluate the effectiveness of our model, we compare our IBMoE with recent parameter-efficient fine-tuning (PEFT)-based and MoE-based methods. For PEFT-based methods, we select **LoRA** (Hu et al., 2021). For MoE-based methods, we employ **MoLoRA** (Zadouri et al., 2023), **MoLA** (Gao et al., 2024), **AlphaMoLA** (Qing et al., 2024), **AdaMoLE** (Liu and Luo, 2024) and **HMoRA** (Liao et al., 2025).

5.3 Evaluation Metrics

We use multiple metrics to comprehensively evaluate IBMoE's performance. For traditional NLP tasks, we assess accuracy (**ACC**) and activated parameters (**Act Params**) to evaluate generalization performance and parameter usage of the adapters. Additionally, we measure the model's inference efficiency by assessing **Time** and **Peak Memory** for the memory consumption of the added adapters.

5.4 Implementation Details

The hyper-parameters are set as follows: (1) **Training Parameters**: We use the Adafactor optimizer with a learning rate of $2e^{-4}$, a batch size of 128. The number of epochs is set to 10. (2) **LoRA adapter**: We adopt the default LoRA configuration, with the intrinsic LoRA rank of experts set to 8 for all the baselines. (3) **IBMoE Parameters**: We apply 5 LoRA experts to every other MLP layer, with one shared expert handling all inputs. During forward processing, the top-k is set to 2, selecting one specific expert and the shared expert for each token. We set $\pi = 0.6$, $\beta = 0.1$, $\gamma = 0.1$, and $\lambda = 0.01$. All experiments are conducted on the NVIDIA A800 GPU.

6 Experimental Results

6.1 Main Results

Table 1 compares the performance of IBMoE with other PEFT and MoE baselines. The table shows that IBMoE achieves the best performance across all models, while using the fewest activated parameters. MoLA and AlphaMoLA reduce knowledge redundancy by assigning fewer experts to lower layers and more to higher layers for better specialization, but its performance is limited without explicit specialization constraints. Although HMoRA introduces features at varying levels of granularity, its performance remains suboptimal because it fails to

account for knowledge differences among experts. Moreover, IBMoE enhances performance by leveraging both task-level and neuron-level sparse activation, significantly reducing computational costs, as indicated by **Act Params** in Table 1.

6.2 Ablation Studies

We conduct ablation studies (Table 2) to evaluate the effectiveness of IBMoE's components. First, we experiment without L_b. We find that imbalanced routing leads to some experts handling disproportionate amounts of data by activating more neurons, which increases the **Act Params**. Next, we remove the sparsity constraint, which makes the experts fail to focus on the most relevant information, and unstable neuron sampling further reduces performance. Finally, we remove the diversity loss, which worsens IBMoE's performance. Without L_d, knowledge redundancy occurs among the experts. Additionally, without any loss constraints, the number of activated parameters in IBMoE approximately doubles because the inactive SNA module causes the model to activate all neurons to complete the tasks.

Model	RTE	
	Acc (%↑)	Act Params ↓
IBMoE	88.43	7.89M
$-L_b$	87.95	7.91M
$-L_b - L_s$	87.06	7.07M
$-L_b - L_s - L_d$	86.29	13.16M

Table 2: Ablation results of IBMoE on the RTE dataset.

6.3 Further Analyses

Analysis of Compression Ratio. In IBMoE, the parameter π controls the proportion of activated neurons. The results in Figure 3 illustrate the model's performance under various compression ratios across multiple datasets. We observe that model performance generally improves as more neurons are activated. However, this improvement gradually diminishes and may eventually reverse as π continues to increase. This observation suggests the presence of neuron redundancy in conventional MoE methods, which can potentially degrade model performance by activating task-irrelevant neurons. Moreover, the model's performance on the RTE dataset drops by approximately 3% when π is set to 0.2, indicating that an excessively low activation ratio may limit the model's capacity to capture task-relevant information.

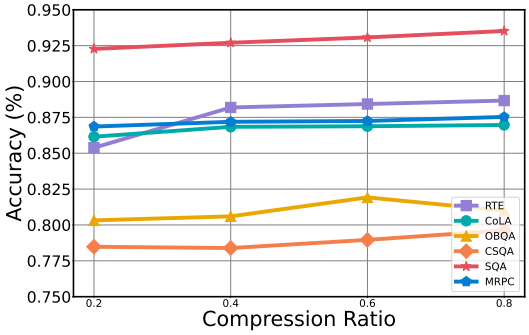

Figure 3: Compression Ratio π with different values.

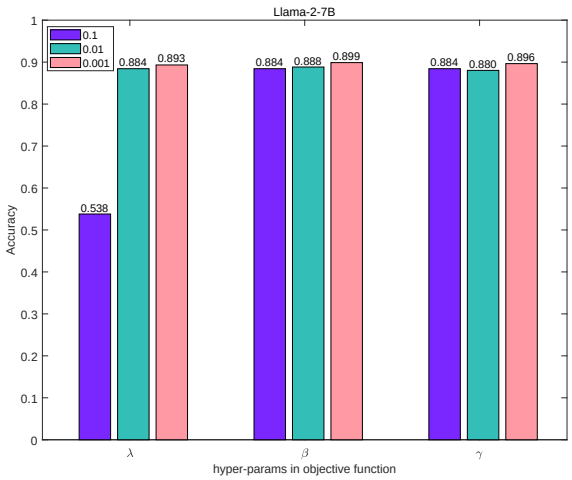

Figure 4: Sensitivity of hyper-parameters.

Analysis of hyper-parameters. The IBMoE framework introduces three hyper-parameters (λ, β, and γ) for the components of the loss function. To analyze the sensitivity of these hyper-parameters, we conduct experiments by varying their values within the range of [0, 0.1, 0.01, 0.001]. Specifically, we adjust one parameter (e.g., λ) while fixing the others to their default values. This approach eliminates potential confounding effects from other factors. The same procedure is repeated for β and γ. As shown in Figure 4, the proposed method exhibits strong robustness to hyper-parameter variations. Notably, setting lower values (e.g., 0.01 or 0.001) leads to a slight improvement in model performance. However, when λ is set to 0.1, the model fails to converge. This instability arises because the scaling of the balance loss L_b dominates the primary task loss L_p.

Analysis of the Greedy Sampling. As mentioned above, each expert drops a large number of neurons by performing Bernoulli sampling on V.

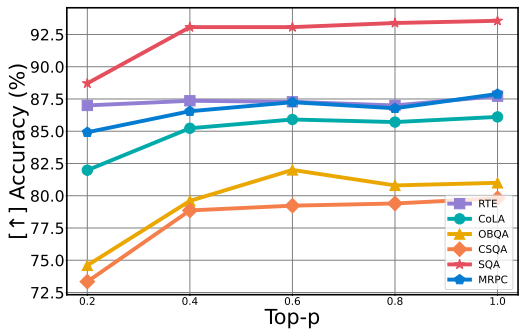

Figure 5: Impact of p on the different tasks.

In this section, we activate neurons using greedy sampling to analyze the impact of neuron activation. Similar to the gating mechanism, we apply top-p sampling on V to activate the most task-related neurons for forward processing. Figure 5 shows the performance of IBMoE$_{\pi=0.2}$ with greedy sampling during inference, where top-$p = 0.2$ denotes that the top 20% of neurons are activated. We observe that the performance of the greedy sampling is lower than that of Bernoulli-based importance sampling at an equal activation scale. This is because importance sampling inherently combines neurons of varying importance levels, enabling a global planning of feature selection, whereas greedy sampling focuses on partial information of the feature. As p increases, greedy sampling achieves a comparable performance to importance sampling, indicating that importance sampling effectively balances performance and computational cost.

Analysis of the Inference Efficiency. We also compare the inference performance of IBMoE with other sparse MoE methods, including MoLA, AdaMoLE and HMoRA, to evaluate its efficiency. These methods are selected because they are designed for more efficient parameter allocation or expert specialization. Specifically, we use one NVIDIA A800 GPU with 80GB of memory to perform inference on 1,000 randomly sampled test set examples, measuring the inference time and peak memory usage of the adapters with a batch size of 16. In Table 3, our proposed method achieves the best performance in both time and memory efficiency, with improvements of 63% in time and 48.5% in memory efficiency compared to the SOTA methods. We attribute this advantage to the fact that the trained IBMoE significantly reduces computational cost and memory consumption by compressing the weight matrices of LoRA experts.

Model	**Inference with 1,000 Samples**	
	Time (s) ↓	Peak Memory (MB) ↓
MoLA	739.3	1452.6
AdaMoLE	329.8	146.2
HMoRA	295.6	842.1
IBMoE	109.3	75.7

Table 3: Inference efficiency of IBMoE.

Analysis of the Training Complexity. Considering that IBMoE introduces additional complexity compared to vanilla MoE methods, we also provide a brief theoretical analysis of its computational cost. Specifically, let the size of a weight matrix be $N \times M$ (e.g., 11,048 and 4,096 in LLaMA2-7B), where $N > M$. The number of experts be K, and the rank of a LoRA expert be r (e.g., $r = 64$). The computational complexity of a vanilla LoRA expert is $O(NrM)$. In IBMoE, the dimensionality of each expert's learnable parameters is bounded by $V \in \mathbb{R}^{\mathbb{N}}$, and two novel loss terms, L_d and L_s, must be computed. To compute L_d, we calculate the cosine similarity for each pair of V, which involves computing the L2 norm for each V and the dot product for every pair. This results in a complexity of L_d is $O(KN)+O(K(K-1)N/2)$. For L_s, we needs to compute mean value of each V, and the complexity is $O(KN)$. Therefore, the total complexity is $O(2KN) + O(K(K-1)N/2) < O(2KN)+O(K^2N) = O(K(K+2)N)$. In practice, K is generally much smaller than M, we have $O(K(K+2)N) \ll O(MrN)$. This indicates that the additional computation introduced by the loss functions is less than that required by a LoRA adapter. Hence, our method achieves a favorable trade-off between performance improvement and computational overhead.

7 Conclusions and Future Work

In this paper, we propose a novel parameter-efficient MoE method that introduces an IB framework and multiple constraints to achieve fine-grained reduction of neuronal redundancy, fostering closer collaboration among experts. Extensive experiments show that our method achieves superior performance comparable to state-of-the-art methods, and reduces inference time by 63% and memory consumption by 48.5%. Furthermore, ablation studies validate the effectiveness of each IBMoE component. In future work, we will explore applications in continual learning by capitalizing on the knowledge specialization inherent in MoE.

8 Limitations

For the sparsity constraint, we simply set the compression ratio $\pi = 0.6$ uniformly for all experts. However, experts in different layers typically need to perform varying levels of feature abstraction on the inputs, which leads to unstable model performance in complex tasks. A potential solution is to design an adaptive compression strategy for each expert, allowing dynamic adjustment of the number of activated neurons, or to vary the number of experts across layers.

References

Ankush Agarwal, Chaitanya Devaguptapu, and Ganesh S. 2025. Hybrid graphs for table-and-text based question answering using LLMs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 858–875.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2016. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, arXiv:1612.00410.

Marco Braga, Alessandro Raganato, Gabriella Pasi, and 1 others. 2024. Adakron: An adapter-based parameter efficient model tuning with kronecker product. In *2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC-COLING 2024-Main Conference Proceedings*, pages 350–357.

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*, arXiv:2407.06204.

Jiawei Chen, Ziqi Zhang, Xinpeng Xie, Yuexiang Li, Tao Xu, Kai Ma, and Yefeng Zheng. 2021. Beyond mutual information: Generative adversarial network for domain adaptation using information bottleneck constraint. *IEEE Transactions on Medical Imaging*, 41(3):595–607.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, and 1 others. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, arXiv:2401.06066.

Giang Do, Hung Le, and Truyen Tran. 2025. Simsmoe: Toward efficient training mixture of experts via solving representational collapse. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 2012–2025.

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing*, pages 9–16.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The International Journal of Man-Machine Studies*, 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*, arXiv:2402.08562.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, arXiv:2501.12948.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, arXiv:2106.09685.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical Reparameterization with Gumbel-Softmax. In *Proceedings of 5th International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*, arXiv:2310.06825.

Kenji Kawaguchi, Zhun Deng, Xu Ji, and Jiaoyang Huang. 2023. How does information bottleneck help deep learning? In *Proceedings of the 40th International Conference on Machine Learning*, pages 16049–16096.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, arXiv:2006.16668.

Mengqi Liao, Wei Chen, Junfeng Shen, Shengnan Guo, and Huaiyu Wan. 2025. Hmora: Making llms more effective with hierarchical mixture of lora experts. In *The Thirteenth International Conference on Learning Representations*.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning

is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35(120):1950–1965.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, arXiv:2110.07602.

Yijiang Liu, Rongyu Zhang, Huanrui Yang, Kurt Keutzer, Yuan Du, Li Du, and Shanghang Zhang. 2024. Intuition-aware mixture-of-rank-1-experts for parameter efficient finetuning. *arXiv preprint arXiv:2404.08985*, arXiv:2404.08985.

Zefang Liu and Jiahua Luo. 2024. Adamole: Fine-tuning large language models with adaptive mixture of low-rank adaptation experts. *arXiv preprint arXiv:2405.00361*, arXiv:2405.00361.

Stephan Sloth Lorenzen, Christian Igel, and Mads Nielsen. 2021. Information bottleneck: Exact analysis of (quantized) neural networks. *arXiv preprint arXiv:2106.12912*, arXiv:2106.12912.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521.

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.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, arXiv:2303.08774.

Peijun Qing, Chongyang Gao, Yefan Zhou, Xingjian Diao, Yaoqing Yang, and Soroush Vosoughi. 2024. Alphalora: Assigning lora experts based on layer training quality. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20511–20523.

Pengjie Ren, Chengshun Shi, Shiguang Wu, Mengqi Zhang, Zhaochun Ren, Maarten Rijke, Zhumin Chen, and Jiahuan Pei. 2024. Melora: Mini-ensemble low-rank adapters for parameter-efficient fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3052–3064.

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. *arXiv preprint arXiv:1701.06538*, arXiv:1701.06538.

Changchong Sheng, Li Liu, Wanxia Deng, Liang Bai, Zhong Liu, Songyang Lao, Gangyao Kuang, and Matti Pietikäinen. 2022. Importance-aware information bottleneck learning paradigm for lip reading. *IEEE Transactions on Multimedia*, 25:6563–6574.

DJ Strouse and David J Schwab. 2017. The deterministic information bottleneck. *Neural computation*, 6(120):1611–1630.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4149–4158.

Shaopeng Tang, Lin Li, Xiaohui Tao, Leqi Zhong, and Qing Xie. 2025. MATO: A model-agnostic training optimization for aspect sentiment triplet extraction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1648–1662.

Naftali Tishby, Fernando C Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint physics/0004057*, physics/0004057.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pages 1–5.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, arXiv:2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, arXiv: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 *Proceedings of 7th International Conference on Learning Representations*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. 2021. Recognizing predictive substructures with subgraph information bottleneck. *IEEE transactions on pattern analysis and machine intelligence*, 46(3):1650–1663.

Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermiş, Acyr Locatelli, and Sara Hooker. 2023. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. *arXiv preprint arXiv:2309.05444*, arXiv:2309.05444.

Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. 2023. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, arXiv:2303.16199.

Rongyu Zhang, Yulin Luo, Jiaming Liu, Huanrui Yang, Zhen Dong, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, Yuan Du, and 1 others. 2024. Efficient Deweahter Mixture-of-Experts with Uncertainty-Aware Feature-Wise Linear Modulation. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, pages 16812–16820.

Shuai Zhang, Junfeng Fang, Xuqiang Li, ALAN XIA, Ye Wei, Wenjie Du, Yang Wang, and 1 others. 2025. Iterative substructure extraction for molecular relational learning with interactive graph information bottleneck. In *The Thirteenth International Conference on Learning Representations*.

Kun Zhu, Xiaocheng Feng, Xiyuan Du, Yuxuan Gu, Weijiang Yu, Haotian Wang, Qianglong Chen, Zheng Chu, Jingchang Chen, and Bing Qin. 2024a. An information bottleneck perspective for effective noise filtering on retrieval-augmented generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1044–1069.

Tong Zhu, Xiaoye Qu, Daize Dong, Jiacheng Ruan, Jingqi Tong, Conghui He, and Yu Cheng. 2024b. Llama-moe: Building mixture-of-experts from llama with continual pre-training. *arXiv preprint arXiv:2406.16554*, arXiv:2406.16554.

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*, arXiv:2311.09179.

Simiao Zuo, Qingru Zhang, Chen Liang, Pengcheng He, Tuo Zhao, and Weizhu Chen. 2022. Moebert: from bert to mixture-of-experts via importance-guided adaptation. *arXiv preprint arXiv:2204.07675*, arXiv:2204.07675.