MoSE: Enhanced Multi-Task Learning with an Adaptive Mixture of Subspace Low-Rank Experts

Anonymous ACL submission

Abstract

Driven by the rapid progress Low-Rank Adaptation (LoRA), the mixture of low-rank experts framework has attracted significant interest from the multi-task learning community. However, existing expert design paradigms operating at multiple levels of granularity, including module, matrix, and subspace, fail to attain an optimal trade-off between parameter redundancy and effective load balance across subspaces. To address this limitation, we propose an adaptive **M**ixture **o**f **S**ubspace Low-Rank **E**xperts (MoSE) for multi-task scenarios. Specifically, we develop a cascading `Bi-Router` comprising two dedicated selectors, which sequentially achieve task-aware rank and subspace allocations. This novel design enables fine-grained subspace restructure across both the "which" (i.e., `capacity`) and "where" (i.e., `position`) dimensions, and thus facilitates uniform activation. To further improve the reliability of routing decisions, we provide more informative features for `Bi-Router`, augmented by a convergence-aware rank prioritization scheme. Moreover, a dynamic task scheduling strategy is introduced for a better-balanced learning. Our framework considerably strengthens the expressive capacity of the unified LoRA module through a more effective, flexible adaptation of experts. Extensive experiments on multiple upstream tasks demonstrate that MoSE outperforms existing methods while maintaining high parameter efficiency.

1 Introduction

The Mixture-of-Experts (MoE) framework (Jacobs et al., 1991; Shazeer et al., 2017) is structured around a set of sub-experts, each dedicated to specific task aspects, combined with a gating network that dynamically routes inputs to the most appropriate expert. The MoE paradigm has provided a powerful foundation for advancing multi-task learning (Zhang and Yang, 2022; Vandenhende et al., 2022), which enables a single model to handle multiple related tasks simultaneously. However, as the scale of large language models (LLMs) (Touvron et al., 2023; Yang et al., 2025a) increases, training and storing a full copy of the expert parameters becomes resource-intensive. To alleviate the computational burden, parameter-efficient variants of MoE (Mahabadi et al., 2021; Vu et al., 2022; Asai et al., 2022; Wang et al., 2023b) have been developed, among which LoRA-MoE (Wang et al., 2023a; Liu et al., 2023; Li et al., 2024a) framework has been the most extensively explored.

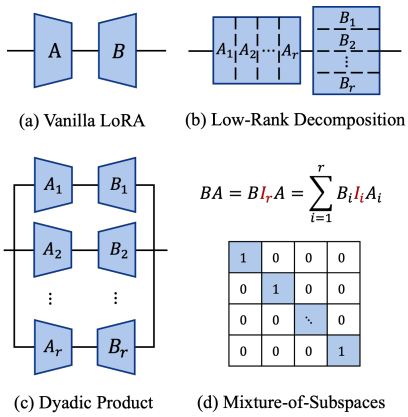


Figure 1: The process of decomposing standard LoRA into r subspaces and combining them can be viewed as the insertion of an identity matrix $I_r \in \mathbb{R}^{r \times r}$ between the matrices A and B.

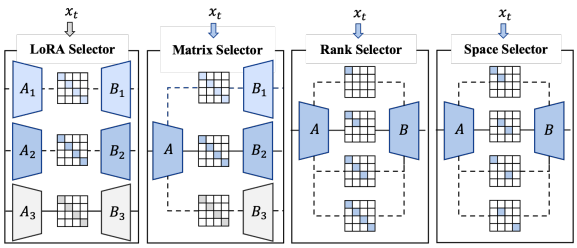
Recently, several studies (Yang et al., 2025b; Zhao et al., 2025; Ning et al., 2025; Liu et al., 2025) have focused on structural redundancy among experts and developed different architectures to achieve more effective multi-task generalization. Along this line, dynamic ranks offer a natural method for enhancing the adaptability of the LoRA model. For example, Valipour et al. (2023) introduces a technique that enables training of dynamic, search-free low-rank adaptation models efficiently over a spectrum of ranks. Zhang et al. (2023b) enable effective budget allocation to crucial param-

rank		MRPC	CoLA	RTE	SST-2	STSB
4	c_1	89.71 $\pm$ 0.12	58.39 $\pm$ 0.48	73.38 $\pm$ 0.40	**93.81** $\pm$ 0.11	**89.64** $\pm$ 0.26
	c_2	**90.20** $\pm$ 0.19	**59.45** $\pm$ 0.23	72.66 $\pm$ 0.19	**93.81** $\pm$ 0.34	89.46 $\pm$ 0.04
	c_3	89.71 $\pm$ 0.22	58.45 $\pm$ 0.45	73.38 $\pm$ 0.27	**93.81** $\pm$ 0.25	89.54 $\pm$ 0.17
	c_4	**90.20** $\pm$ 0.13	58.4 $\pm$ 0.35	**74.82** $\pm$ 0.27	93.58 $\pm$ 0.23	89.49 $\pm$ 0.17
8	c_1	90.20 $\pm$ 0.25	**60.52** $\pm$ 0.41	74.10 $\pm$ 0.11	93.12 $\pm$ 0.30	**89.57** $\pm$ 0.19
	c_2	89.22 $\pm$ 0.34	60.00 $\pm$ 0.45	74.10 $\pm$ 0.21	93.46 $\pm$ 0.16	89.38 $\pm$ 0.18
	c_3	**91.67** $\pm$ 0.21	59.45 $\pm$ 0.23	**75.54** $\pm$ 0.35	93.00 $\pm$ 0.07	89.37 $\pm$ 0.21
	c_4^*	90.69 $\pm$ 0.18	**60.52** $\pm$ 0.19	74.10 $\pm$ 0.21	**93.58** $\pm$ 0.20	89.42 $\pm$ 0.22
12	c_1^*	90.20 $\pm$ 0.26	60.00 $\pm$ 0.38	**74.82** $\pm$ 0.17	**93.69** $\pm$ 0.04	89.54 $\pm$ 0.24
	c_2	90.20 $\pm$ 0.14	**60.55** $\pm$ 0.16	73.38 $\pm$ 0.29	93.23 $\pm$ 0.28	89.17 $\pm$ 0.25
	c_3	**90.69** $\pm$ 0.32	60.06 $\pm$ 0.45	72.66 $\pm$ 0.24	93.12 $\pm$ 0.24	89.02 $\pm$ 0.09
	c_4	88.73 $\pm$ 0.16	58.98 $\pm$ 0.36	72.66 $\pm$ 0.30	93.46 $\pm$ 0.26	**89.63** $\pm$ 0.13

Table 1: Performance of *T5-base* with different ranks on 5 tasks. We explore four distinct low-rank subspace combinations (denoted as $\{c_1, c_2, c_3, c_4\}$) for each rank. For instance, the c_4^* corresponds to $[a_5b_5, a_6b_6, ..., a_{11}b_{11}, a_{12}b_{12}]$ for rank 8, the c_1^* denotes $[a_1b_1, a_2b_2, ..., a_{11}b_{11}, a_{12}b_{12}]$ for rank 12.

eters for enhanced fine-tuning, especially in low-resource cases. SoRA (Ding et al., 2023) employs a sparse gate for dynamic rank adaptation, compressing to an optimal LoRA module for efficient inference. Despite achieving improvements, these methods fail to account for the distinctions and interrelationships among different tasks in multi-task settings.

To investigate dynamic rank-space adaptation in multi-task learning, Zhang et al. (2025) introduce MoRE, a method that conceptualizes each LoRA rank as an expert and utilizes a rank selector to enable flexible, tailored parameter capacity allocation. The overlap and non-overlap part of a single LoRA unit are designed to respectively capture inter-task connections and task-specific distinctions. While MoRE shows considerable advantages, it continues to suffer from issues such as **over-activation of lower-indexed subspaces, inadequate learning of higher-indexed subspaces, and enforced updates of redundant subspaces**, which undermines the balance and efficiency of parameter learning. Specifically, building on the low-rank decomposition $\sum_{i=1}^{r} a_i b_i$ (Figure 1c), a hierarchical dependency exists among experts of different ranks, where rank-i experts is constrained to subspaces $a_j b_j$ $(j \leq i)$. Consequently, the nested structure causes subspaces with smaller j to activate more frequently, thus leading to under-utilization of higher-indexed parameters. Besides, if a latter-indexed subspace $a_j b_j$ is essential but preceding subspaces are redundant, this strategy forces the rank-j expert parameters to update at higher learning rates, reducing efficiency as shown in columns c_4^* and c_1^* of Table 1. Meanwhile, optimal performance demands task-specific subspace combinations, where under a rank-8[1] constraint, c_3 and c_1 yield the best results for MRPC and CoLA, respectively. These findings provide compelling evidence for a significant conclusion: **Effective LoRA-MoE for multi-task scenarios also requires adaptive subspace recomposition to facilitate balanced, flexible, and finer-grained expert activation and allocation.**



(a) Multi-LoRAs (b) Multi-Head (c) Mixture-of-Subspaces

Figure 2: Comparison among different LoRA-MoE architectural paradigms.

In this work, we introduce MoSE, a novel adaptive mixture of subspace low-rank experts designed for multi-task scenarios. As shown in Figure 1d, we first formulate MoRE as inserting different I_r between matrices A and B, where the first $\hat{r}$ diagonal entries are 1 and the rest are 0. Based on this foundation, MoSE differs from MoRE by explicitly establishing essential connections between subspaces and specific tasks, as illustrated in Figure 2c. Specifically, we propose a `Bi-Router` consisting of two specialized selectors to tailor experts. Furthermore, we enhance the gating mechanism by integrating prior knowledge during the initializa-

[1]In this paper, the rank is defined as the number of subspaces.

tion of task embeddings, thereby provides richer task characteristics. Moreover, to improve the precision of rank selector, a convergence-aware optimization term is designed to achieve rank targeted regulation for current task. Furthermore, we develop a dynamic task scheduling strategy to assigns a higher priority to more challenging tasks.

In summary, our key contributions are threefold:

- We propose MoSE, which achieves fine-grained allocation, balanced activation, and flexible recomposition through a cascading `Bi-Router` module.
- We achieve enhanced optimization by providing informative initialization, leveraging loss convergence patterns, and integrating adaptive task scheduling.
- Through extensive evaluation on multi-task benchmarks, MoSE demonstrates significant effectiveness.

2 Related Work

2.1 Low-Rank Adaptation (LoRA)

To reduce the computational cost of fine-tuning LLMs, parameter-efficient fine-tuning (PEFT) methods such as Prefix-Tuning (Li and Liang, 2021), Prompt-Tuning (Lester et al., 2021), and LoRA (Hu et al., 2022) have been widely adopted. Studies of LoRA variants focus on structural imbalance, redundancy, and flexibility. (1) To mitigate structural imbalance, methods such as LoRA+ (Hayou et al., 2024), Asymmetric LoRA (Zhu et al., 2024), and LoRA-FA (Zhang et al., 2023a) employ asymmetric fine-tuning strategies for LoRA modules. (2) To reduce parameter redundancy, VeRA (Kopiczko et al., 2024) and VB-LoRA (Li et al., 2024b) utilize shared parameters across modules. (3) Moving beyond manual rank selection, methods such as DyLoRA (Valipour et al., 2023), AdaLoRA (Zhang et al., 2023b), and SoRA (Ding et al., 2023) dynamically configure LoRA ranks to improve flexibility and efficiency.

2.2 LoRA-based Multi-Task Learning

Multi-Task Learning (MTL) (Zhang and Yang, 2022; Vandenhende et al., 2022) is an extensively studied paradigm that tackles multiple related tasks with a single model. To this end, recent work has pursued three prominent strategies: parallel modules, asymmetric LoRA, and unified adapter, to better facilitate parameter efficiency.

Parallel Modules. Some studies treat parallel LoRA modules to handle different tasks. Early approaches, such as Huang et al. (2023) and Feng et al. (2024), pioneered the pre-training of multiple LoRA weights followed by gating mechanisms. Building on this, subsequent works have achieved finer-grained control over the combination weights by grouping (Dou et al., 2024) and layering (Wu et al., 2024b) LoRA experts. This evolution culminates in the standard LoRA-MoE framework (Wang et al., 2023a; Li et al., 2024a; Luo et al., 2024), which co-learns experts and routers to manage task-shared and task-specific knowledge.

Asymmetric LoRA. Building on the key insight that down-projection matrix A captures redundant, task-general knowledge whereas the up-projection matrix B encodes diverse, task-specific features, works such as Zhu et al. (2024); DeepSeek-AI (2024); Guo et al. (2025) leverage this inherent asymmetry to construct structurally reorganized matrices. Furthermore, MTL-LoRA (Yang et al., 2025b) employs task-specific transformation matrices alongside multiple up-projection matrices to model both task-relevant and shared information. In addition, MoSLD (Zhao et al., 2025) applies the dropout operation on the general-feature parameter matrix A to alleviate the imbalance and over-fitting limitations. To achieve more tailored refinement, Wang et al. (2025) propose an asymmetric rank approach that reallocates redundant parameters from matrix A to the more critical matrix B.

Unified Adapter. Recently, Liu et al. (2025) demonstrate that the architectural complexity of multi-adapter and multi-head designs is superfluous to achieve strong multi-task generalization. Additionally, Zhang et al. (2025) introduced MoRE, a framework with a unified LoRA unit that pioneers adaptive rank selection for multi-task learning. However, its performance is constrained by imbalanced expert activation, inefficient parameter updates in redundant subspaces, and inflexible expert adaptation.

3 Preliminary

Vanilla LoRA LoRA (Figure 1a) enhances a feed-forward layer with input $x \in \mathbb{R}^{1\times m}$ and weight $W_0 \in \mathbb{R}^{m\times n}$ by incorporating a down-projection matrix $A \in \mathbb{R}^{m\times r}$ and an up-projection matrix $B \in \mathbb{R}^{n\times r}$, producing the output $y = x(W_0 + AB^T) \in \mathbb{R}^{1\times n}$.

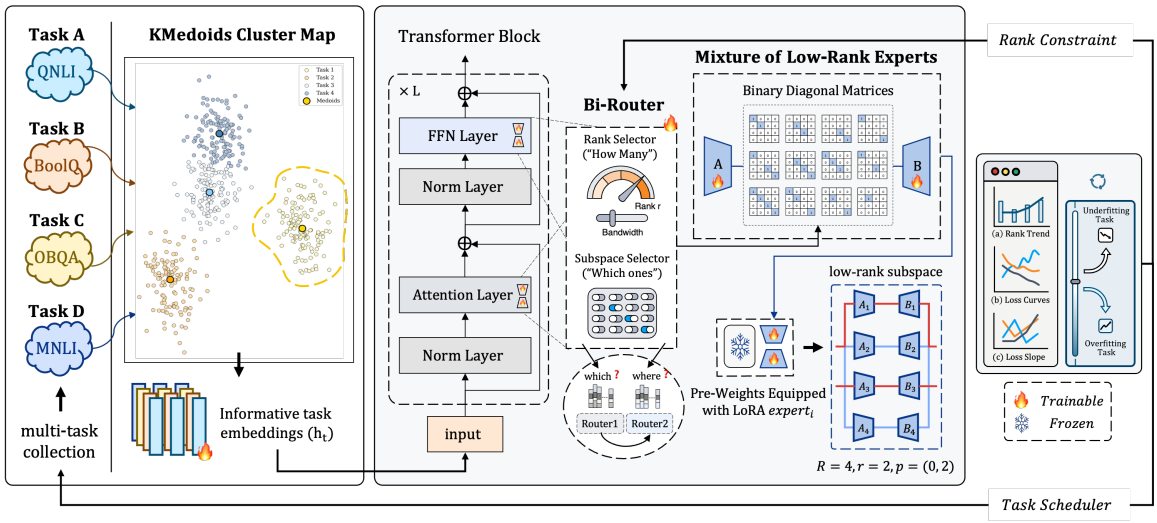


Figure 3: Overview of the MoSE framework.

Mixture-of-Subspaces in LoRA Following Liu et al. (2024); Wu et al. (2024a), we decompose the LoRA module (Figure 1b) into the multiplication of two vectors (Figure 1c), leading to an expression of the LoRA update $\Delta W = AB^T$ as a sum of dyadic products by factorizing A and B into their column vectors.

$$\Delta W = [a_1, ..., a_r] * [b_1, ..., b_r]^T = \sum_{i=1}^{r} (a_i \otimes b_i) \quad (1)$$

where a_i and b_i are the i-th columns of A and B, respectively. As illustrated in Figure 1d, the forward operation can be formally represented by Eq.(2), which incorporates a frozen identity matrix $I \in \mathbb{R}^{r \times r}$.

$$y = x(W_0 + AIB^T) \quad (2)$$

LoRA-MoE Figure 2a shows a vanilla LoRA-MoE framework (Zadouri et al., 2024; Li et al., 2024a; Luo et al., 2024), where k LoRA experts ($E_i = A_i B_i^T$, $i \in \{1, ..., m\}$) are added to each layer. A router $\mathcal{R}$, parameterized by the weight matrix $W_{\mathcal{R}} \in \mathbb{R}^{m \times k}$, governs the selection of an expert, with the output given by:

$$y = xW_0 + \sum_{i=1}^{k} \mathcal{R}_i(x)(xA_iB_i^T) \quad (3)$$

where $\mathcal{R}_i(x)$ represents the routing probability for expert E_i. Particularly, some works (Yang et al., 2025b; Wang et al., 2025) define each expert E_i as AB_i^T (Figure 2b) while others regard as AB^T (Liu et al., 2025) or $A_{[:i,:]}B_{[:i,:]}{}^T$ (Zhang et al., 2025) (Figure 2c).

4 Methodology

Task. Given a set of $\mathcal{K}$ tasks $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_{\mathcal{K}}\}$, our objective is to learn a unified model $\mathcal{M}$ that satisfies the requirements of all tasks. For t-th task, the associated dataset $\mathcal{D}_t = \{(x_i^t, y_i^t)\}_{i=1}^{|\mathcal{D}_t|}$ consists of $|\mathcal{D}_t|$ input-output pairs.[2]

Overview. To address the challenges outlined in Section 1, we propose MoSE, a novel framework designed to enable fine-grained and balanced activation, as illustrated in Figure 3. Our framework begins with an informative vector initialization (Section 4.1), feeding into the `Bi-Router` (Section 4.2) scheme, followed by the design of an additional convergence-aware rank prioritization (Section 4.3) and a dynamic task-scheduling strategy (Section 4.4), culminating in a training objective (Section 4.5). The subsequent sections provide a detailed description of each component.

4.1 Informative Initial Task Vector

We propose a novel task embedding initialization mechanism based on clustering algorithms, designed to inject meaningful prior knowledge into the learning trajectory of the router. Specifically, given a dataset $\mathcal{D}_t$ for task t, we construct i-th input sequence as $s_i^t = x_i^t \oplus$ "answer: " $\oplus\, y_i^t$, which is then encoded into an embedding representation h_i^t. Then, we form the embedding set $\{h_1^t, h_2^t, ..., h_{|\mathcal{D}_t|}^t\}$ and apply *k-medoids clustering* offline to extract the centroid embedding h_*^t, which

[2]Here, x_i^t represents the input data and y_i^t denotes the label or output for task $\mathcal{T}_t$.

serves as the initialized task vector $\mathbf{v}_t^{(0)}$ for task t. Since $\mathbf{v}_t^{(0)}$ already carries rich semantic information, we introduce a linear scaling coefficient μ to reduce the learning rate of $\mathbf{v}_t$, thereby preserving the beneficial initialized features and stabilizing task representation learning.

4.2 Cascading Bi-Router for Expert Allocation

We abstract the expert allocation mechanism in MoRE as follows: each selection of $\hat{r}$ (where $\hat{r} \leq$ the rank r) corresponds to keeping the first $\hat{r}$ diagonal elements of the matrix I in (2) as 1 and setting the rest to 0, thereby enabling on-the-fly activation of rank-truncated LoRA modules. However, this strategy suffers from activation imbalance and redundant updates across subspaces, ultimately resulting in suboptimal expert allocation modeling with limited flexibility.

Accordingly, we propose a `Bi-Router` unit consists of two selectors that respectively determine the rank and subspace to be utilized. This design decouples expert activation into two dimensions: "which" rank and "where" subspace to select, enabling fine-grained subspace composition that liberates the model from the constraints of an inherently nested configuration. Specifically, we construct a rank selector $G_r(\cdot)$ that determines the number of subspaces $\hat{r}_t + 1 \in [1, r]$ by applying an $\arg\max$ operation.

$$\begin{aligned} G_r(\mathbf{v}_t) &= \text{Softmax}(W_r \mathbf{v}_t + b_r) \\ \hat{r}_t &= \arg\max G_r(\mathbf{v}_t) \end{aligned} \tag{4}$$

Subsequently, we introduce an additional gating mechanism $G_s(\cdot)$ for selecting the specific subspace index set $S_t = \{i_1, i_2, ..., i_{\hat{r}_t+1}\}_t$. Since the probabilities at each index position are independent, we replace the softmax in Eq.(4) with a sigmoid function.

$$\begin{aligned} G_s(\mathbf{v}_t) &= \text{Sigmoid}(W_s \mathbf{v}_t + b_s) \\ S_t &= \arg top(\hat{r}_t + 1)(G_s(\mathbf{v}_t)) \end{aligned} \tag{5}$$

where $\arg top(i)(j)$ stands for the indices of the top i values in j. Notably, the subspace selector relies on the result of the rank selector, forming a cascaded architecture within the linear layer. One step further, the S_t then yields the recomposed LoRA expert $\Delta W_t = \mathrm{B}_t \mathrm{A}_t$[3]:

$$\mathrm{A}_t = A[S_t, :], \quad \mathrm{B}_t = B[:, S_t] \tag{6}$$

[3]where $A \in \mathbb{R}^{r \times m}, B \in \mathbb{R}^{n \times r}$

In line with Zhang et al. (2025), we incorporate Straight-Through Estimator (STE) (Bengio et al., 2013) technique for gradient approximation, enabling effective backpropagation to $G_r(\cdot)$ and $G_s(\cdot)$. Consequently, the forward pass in Eq.(6) is modified as follows:

$$\begin{aligned} \Delta W_t &= \text{Ste}(G_r)[\hat{r}_t] \cdot \text{Ste}(G_s)[S_t] \cdot \mathrm{B}_t \mathrm{A}_t \\ \text{Ste}(\alpha) &= \alpha + sg[\text{SparseTopk}(\alpha) - \alpha] \end{aligned} \tag{7}$$

where $sg(\cdot)$ denotes the stop-gradient operator. If α is $G_r(\cdot)$, applying SparseTopk with k = 1 yields a one-hot vector; if α indicates $G_s(\cdot)$, applying SparseTopk with $k > 1$ yields a k-hot vector.

4.3 Convergence-Aware Rank Prioritization

Complex tasks are inherently associated with sophisticated data distributions and nonlinear decision boundaries. Therefore, approximating these rich input-output mappings requires high-rank models with sufficient capacity. We posit that the slow convergence and elevated initial loss observed in such tasks are indicative of this complexity, thereby necessitating higher ranks for adequate learning.

To achieve this, we record and update the validation loss history for task i as $\{\mathcal{L}_{i,1}, \mathcal{L}_{i,2}, \mathcal{L}_{i,3}, \mathcal{L}_{i,4}\}$ with a window size $w = 4$, where $\mathcal{L}_{i,4}$ is the latest loss. Based on these, we compute three signals $\{\alpha_i, \beta_i, d_i\}$ to further regulate the probability distribution of G_r over different ranks in Eq.(9).

1. Estimation of Rank Trend. The slope of the rank distribution (α_i) across all training steps is estimated through the least squares method, with details provided in Appendix A.2.

2. Convergence Slope. The convergence slope (β_i) of the loss set $\{\mathcal{L}_{i,1}, \mathcal{L}_{i,2}, \mathcal{L}_{i,3}, \mathcal{L}_{i,4}\}$ is captured based on normalization and linear regression, as detailed in Appendix A.2.

3. Loss Absolute Deviation. We define the difference (d_i) as the deviation between the latest loss of task i and the average loss across all $\mathcal{K}$ tasks.

$$d_i = \mathcal{L}_{i,4} - \frac{1}{\mathcal{K}} \sum_{j=1}^{\mathcal{K}} \mathcal{L}_{j,4} \tag{8}$$

Specifically, whenever a significant decline in the task-fitting slope ($\beta_i < 0$) is combined with a high loss value ($d_i > 0$) and accompanied by a negative expert signal ($\alpha_i < 0$), a penalty will be triggered

to prevent premature convergence.

$$\mathcal{L}_i = -\sum_{j=1}^{r} j \cdot (G_r(\mathbf{v}_i))_*, \quad \mathcal{L}_{\text{add}} = \sum_{i=1}^{\mathcal{K}} \mathcal{L}_i \tag{9}$$

Here, $(G_r)_*$ denotes the gating weight for $*$-th rank selection. When $\alpha_i < 0$, $d_i > 0$, and $\beta_i < 0$, the gate assigns a higher probability to experts with larger ranks (by setting $* = j$), thereby promoting learning. Conversely, when $\alpha_i > 0$, $d_i < 0$, and $\beta_i > 0$, it reduces the selection probability for higher-rank experts (by setting $* = r + 1 - j$), thus mitigating overfitting.

Our CoR mechanism adapts to complex tasks not by forcibly activating higher-rank components, but by increasing the activation probability of already high-rank experts. This is guided by an adaptive loss term that is only triggered when a task exhibits three conditions simultaneously: high validation loss, continuous convergence, and a declining rank trend. The loss remains inactive in all other cases.

4.4 Adaptive Task Scheduling

At each training iteration, a task is sampled according to a sampling probability ϕ_t, and samples from selected task are randomly extracted for learning. Since learning balance depends on ϕ_t, we propose a novel task scheduling strategy to avoid prioritizing overfitting tasks while ensuring sufficient attention to challenging ones. First, the initial sampling weight ϕ_t are derived following Zhang et al. (2025), based on the dataset size $|\mathcal{D}_t|$. After each evaluation, ϕ_t is dynamically updated to $\hat{\phi}_t$ based on the trajectory of the validation loss, with the adjustment quantified by δ_t.

$$\begin{aligned} \hat{\phi}_t &= \text{Softmax}([\phi_1 + \delta_1, ..., \phi_{\mathcal{K}} + \delta_{\mathcal{K}}])_t \\ \delta_t &= \eta \cdot [(\mathcal{L}_t)_{\text{-1}} - \bar{\mathcal{L}} \cdot (1 + \mathbb{I}_t \cdot \kappa)] \end{aligned} \tag{10}$$

where $(\mathcal{L}_t)_{\text{-1}}$ represents the latest validation loss of task t. The average loss across all $\mathcal{K}$ tasks is defined as $\bar{\mathcal{L}} = \frac{1}{\mathcal{K}} \sum_{j=1}^{\mathcal{K}} (\mathcal{L}_j)_{\text{-1}}$. The weight coefficient is η, and κ serves as an amplification factor. The indicator $\mathbb{I}_t$ is activated when the loss exhibits an increase over three consecutive evaluation steps while remaining below the average loss. Upon activation, it amplifies the adjustment term by a factor of κ to accelerate the increase of δ_t.

4.5 Optimization Objective

Our learning objective integrates a contrastive loss to promote discriminative task embeddings, following the practice of Zhang et al. (2025). The overall loss function combines the primary generation loss with the contrastive loss (as details in Appendix A.3) and an additional term, balanced by two hyperparameters λ_1 and λ_2. Formally, the total objective for MoSE is defined as:

$$\begin{aligned} \mathcal{L}_{\text{gen}} &= -\sum_{t} y_t \log \hat{y}_t \\ \mathcal{L}_{\text{total}} &= \mathcal{L}_{\text{gen}} + \lambda_1 \mathcal{L}_{\text{con}} + \lambda_2 \mathcal{L}_{\text{add}} \end{aligned} \tag{11}$$

where y_t and $\hat{y}_t$ refer to the t-th token in the target and generated sequences, respectively, and $\mathcal{L}_{\text{gen}}$ is the standard cross-entropy loss for generation.

5 Experiments

5.1 Experimental Setup

Datasets and Tasks. We evaluate MoSE on the GLUE benchmark (Wang et al., 2018) and datasets including BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), OBQA (Mihaylov et al., 2018), and ARC (Clark et al., 2018) to assess its NLU and commonsense reasoning abilities[4]. Meanwhile, we employ the SciTail (Khot et al., 2018) and CB (de Marneffe et al., 2019) datasets for assessing the robustness and generalization of MoSE in low-resource settings.

Compared Methods. We compare MoSE with state-of-the-art (SoTA) LoRA-MoE methods, including: **MultiLoRA** (Wang et al., 2023a), employing parallel adapters for multi-task learning; **MixLoRA** (Li et al., 2024a), which integrates a gating mechanism and restricts LoRA to FFN layers; **MoELoRA** (Liu et al., 2023), utilizing parallel LoRA modules with a reduced rank of r/n alongside a task embedding module; **MTL-LoRA** (Yang et al., 2025b), designed to learn task-relevant and shared representations through specialized transformations and adaptive weights; and **MoRE** (Zhang et al., 2025), which features a novel selector that enables dynamic rank adaptation in multi-task learning.

Implementation. Following previous works, we employ publicly accessible pre-trained T5[5](Raffel et al., 2020) and LLaMA2[6](Touvron et al., 2023) from HuggingFace as the backbone. All experimental results reported in this paper were obtained

[4]Detailed dataset statistics are provided in Appendix B.
[5]https://huggingface.co/google-t5
[6]https://huggingface.co/meta-llama

by averaging the performance over three independent runs. Implementation details are provided in Appendix C.

5.2 Overall Performance

As summarized in Table 2, MoSE establishes new state-of-the-art (SoTA), achieving the highest average performance across 13 tasks and demonstrating consistent gains on diverse benchmarks. Compared to structurally sophisticated approaches, simpler methods (MoELoRA, MTL-LoRA, MoRE) offer more competitive results. However, they still faces distinct challenges. Specifically, MoELoRA fails to incorporate task-aware mechanisms, and MTL-LoRA does not implement tailored rank strategies, which ultimately restricts their effectiveness. While MoRE employs only a single LoRA module and exhibits an superior balance between efficiency and performance, its effectiveness is limited by suboptimal expert allocation, a constraint inherent to its nested architecture. Among them, MoSE introduces a `Bi-Router` module enriched with informative task characteristics, which judiciously **assigns more suitable experts to each input** and consequently achieves optimal results. Besides, MoSE demonstrates particularly notable improvements on challenging tasks, with gains of +6.4% on CoLA, +15.1% on ARC-C, and +9.1% on ARC-E. These results validate the effectiveness of the proposed adaptive rank prioritization and task scheduling strategy, which enables the model to **focus its capacity on more challenging tasks**. However, our method underperforms on large-scale, simpler datasets like MNLI, QNLI, and QPP, primarily because the training data was limited to 20,000 samples. Consequently, our strategy, which prioritizes complex tasks, therefore allocated fewer resources to these tasks within the data constraint. Nevertheless, it remains competitive with MultiLoRA, MoELoRA and MixLoRA with a smaller number of trainable parameters.

As shown in Table 3, we evaluate the few-shot transfer ability of MoSE[7]on three datasets. MoSE shows a pronounced advantage in low-resource settings, significantly outperforming all previous baselines. This suggests that MoSE learns a more informative task vector, which provides more precise input representations for held-out but related tasks, thereby improving routing decisions and generalization. Furthermore, our strategy promotes balanced learning, mitigating overfitting by enabling dynamic scheduling and rank allocation based on performance, leading to more effective learning from limited samples.

[7]The model is fine-tuned only on the GLUE benchmark.

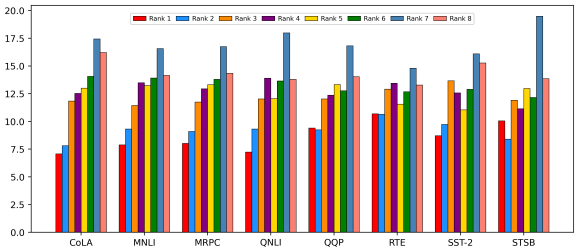


Figure 4: Distribution of rank loadings in MoSE.

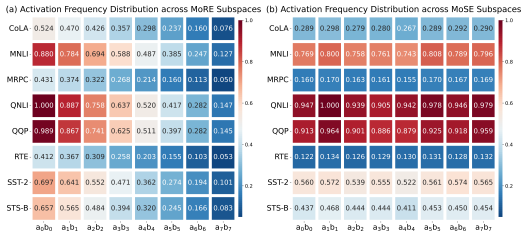


Figure 5: Subspace allocation in MoRE and MoSE.

5.3 Expert Allocation Analysis

Analysis of the expert distribution across layers (Figure 4) indicates a generally balanced pattern, with slightly higher activation for ranks 7 and 8. A detailed view of subspace activation frequencies shows that MoRE presents a strict inverse correlation between subspace index and activation frequency (Figure 5a), while MoSE displays no consistent preference for a single configuration, resulting in more uniform activation (Figure 5b). This confirms the advantage of our subspace selector in enabling balanced, fine-grained, and flexible expert allocation.

5.4 Visualization of Task Embeddings

Task embeddings serve as conditional signals guiding the downstream module, making their representational quality critical. Therefore, we conduct a visual comparative analysis through PCA to examine embedding variations across training stages in both MoRE (Figure 6) and MoSE (Figure 7). Our findings are summarized as follows: (1) MoSE produces well-initialized representations that reflect task similarity and distinction, with related tasks placed closer (RTE and QNLI) and unrelated tasks positioned farther apart (STSB and PIQA), implicitly encoding task-specific semantics. This contributes to **more coordinated, stable, and efficient**

Method	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	ARC-C	ARC-E	BoolQ	OBQA	PIQA	Avg.
	mcc.	acc	acc	acc	acc	acc	acc	pea.	acc	acc	acc	acc	acc	
MultiLoRA†	72.4	87.6	88.7	93.4	85.0	87.8	96.7	92.2	61.9	81.6	76.5	68.2	72.9	80.1
MixLoRA†	73.2	86.8	88.2	93.6	88.1	87.1	96.0	91.3	76.3	86.8	84.3	82.6	79.5	84.9
MoELoRA†	64.4	87.0	91.2	91.4	87.6	87.8	96.3	92.4	77.3	86.8	84.0	81.8	79.9	84.6
MTL-LoRA†	68.0	**91.4**	90.2	**96.3**	**91.4**	92.4	**97.1**	93.3	73.4	87.0	71.0	82.6	84.4	86.0
MoRE†	66.9	89.4	89.2	94.4	89.0	92.1	96.9	92.2	74.2	86.7	87.2	83.0	82.3	85.7
MoSE	**73.3**	86.8	**92.2**	94.3	87.0	**93.5**	96.1	**93.5**	**89.3**	**95.8**	**88.5**	**89.8**	**87.9**	**89.8**

Table 2: Performance comparison of baselines and MoSE on GLUE and five commonsense reasoning tasks (LLaMA2-7B backbone). For STS-B, we report pearson correlation coefficients; for CoLA, matthews correlation; and for all other tasks, accuracy. The best results are highlighted in **bold**, the second-best are underlined, and results copied from the MoRE paper are marked with †.

Task	BoolQ(4/16/32)	CB(4/16/32)	SciTail(4/16/32)
Finetuning	50.5/56.5/58.4	57.7/77.0/80.0	79.6/80.0/81.9
LoRA	64.2/66.1/67.4	84.3/85.7/87.1	80.8/84.0/85.3
HyperFormer	48.0/50.2/58.3	60.7/76.3/81.4	82.0/86.5/85.8
MPT	62.2/63.3/68.9	73.6/78.6/82.1	80.2/87.3/86.3
MultiLoRA	65.2/65.8/67.6	85.0/85.7/86.6	78.1/81.7/83.6
MixLoRA	62.8/64.4/66.2	86.6/86.4/89.3	77.5/82.4/83.3
MoELoRA	64.0/64.8/65.7	85.4/86.3/88.3	80.4/83.1/84.5
MoRE	64.6/66.2/67.9	85.7/86.4/88.6	**83.8**/86.7/87.4
MoSE	**68.5/69.7/71.1**	**89.3/89.3/92.9**	83.7/**86.9/88.5**

Table 3: Results summarize the few-shot domain transfer accuracy achieved by T5-base model following finetuning on GLUE benchmark. **Bold** denotes the best results.

Scheme	ARC-C	ARC-E	BoolQ	OBQA	PIQA	Avg.
MoSE	**65.53**	**77.86**	84.89	**74.40**	**80.47**	**84.41**
w/o `InfoVec`	62.29	76.81	86.61	73.00	78.94	83.68
w/o `Bi-Router`	62.29	75.13	**86.79**	71.40	77.69	83.03
w/o `ATS`	59.64	74.37	86.42	68.00	79.05	82.61
w/o `CoR`	59.81	74.83	86.42	68.80	75.95	82.33
MoRE	51.28	69.15	85.50	63.00	73.83	80.73

Table 4: Ablation results validate the contribution of each component in MoSE, with average scores reported over 13 tasks using the T5-3B backbone. "w/o `InfoVec`" represents the variant that removing the design in Section 4.1, "w/o `Bi-Router`" in Section 4.2, "w/o `CoR`" in Section 4.3, and "w/o `ATS`" in Section 4.4.

multi-task learning. (2) In contrast, the initial embeddings in MoRE exhibit a more random distribution, necessitating that the model learn task distinctions from scratch. This not only consumes additional training resources but also leads to instability and oscillations during the learning process. (3) Furthermore, the embeddings of MoSE remain highly consistent before and after training, enabling stable fine-tuning and better preservation of inter-task relationships, which **promotes faster convergence and improved generalization.**

5.5 Ablation Study

Ablation results for MoSE are presented in Table 4. The study emphasizes that removing the rank prioritization (w/o `CoR`) causes a performance drop of 2.08%, underscoring its critical role in maintaining model capacity. Meanwhile, instead of adaptive scheduling, using a fixed sampling probability (w/o `ATS`) leads to a 1.8% decrease, highlighting the necessity of dynamically allocating resources based on task difficulty. Besides, ablating the cascaded routing strategy (w/o `Bi-Router`) and relying only on the rank selector, results in a 1.38% drop, confirming that the subspace selector is essential for fine-grained expert construction. Finally, random initialization based on Kaiming distribution (w/o `InfoVec`) also yields suboptimal results, indicating that task vectors infused with prior knowledge are crucial for learning.

6 Conclusion

In this paper, we present MoSE, a parameter-efficient MoE framework featuring a cascaded `Bi-Router` that performs hierarchical rank allocation and subspace recomposition for fine-grained, task-aware, and balanced expert selection. This design decouples expert activation along two orthogonal dimensions: "which" rank and "where" subspace, thereby liberating the model from inherently nested architectural constraints. Besides, we introduce an informative task vector for improved initialization, enhanced by convergence-aware rank prioritization and dynamic task scheduling to enable adaptive multi-task learning. Empirical results on supervised, low-resource, and ablation experiments show that MoSE outperforms existing methods across diverse benchmarks while maintaining high parameter efficiency, offering a practical solution for scalable LoRA-MoE.

Limitations

One limitation of this work is that evaluations are confined to standard NLP tasks, which are adopted for their well-established research landscape and mature benchmark suites, while further validation on multimodal scenarios remains unexplored. This is primarily because existing baseline methods have not released results in such cross-domain settings. Performing additional retraining would inevitably introduce confounding variables, which would contaminate the validation of our core contribution. Moreover, a direct and head-to-head comparison is both feasible and fair, as all baseline approaches cited in this paper have been exclusively evaluated on NLP benchmarks. MoSE establishes a new state-of-the-art and corroborates the effectiveness of its foundational design, while leaving extensions to multimodal scenarios for future work.

References

Akari Asai, Mohammadreza Salehi, Matthew E Peters, and Hannaneh Hajishirzi. 2022. Attempt: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6655–6672.

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, pages 7432–7439.

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 the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 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. *CoRR*, abs/1803.05457.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *Proceedings of Sinn und Bedeutung 23*.

DeepSeek-AI. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *CoRR*, abs/2405.04434.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. 2023. Sparse low-rank adaptation of pre-trained language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4133–4145.

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.

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. 2024. Mixture-of-loras: An efficient multitask tuning method for large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 11371–11380.

Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. 2025. Selective aggregation for low-rank adaptation in federated learning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2023. Lorahub: Efficient cross-task generalization via dynamic lora composition. *CoRR*, abs/2307.13269.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Comput.*, 3(1):79–87.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the AAAI conference on artificial intelligence*.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. 2024. Vera: Vector-based random matrix adaptation. In *The Twelfth 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. *CoRR*, abs/2404.15159.

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.

Yang Li, Shaobo Han, and Shihao Ji. 2024b. Vb-lora: Extreme parameter efficient fine-tuning with vector banks. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Jinda Liu, Bo Cheng, Yi Chang, and Yuan Wu. 2025. Align, don't divide: Revisiting the lora architecture in multi-task learning. *CoRR*, abs/2508.05078.

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*, abs/2310.18339.

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. *CoRR*, abs/2404.08985.

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. *CoRR*, abs/2402.12851.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. 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 565–576.

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 *EMNLP*.

Lin Ning, Harsh Lara, Meiqi Guo, and Abhinav Rastogi. 2025. Mode: Effective multi-task parameter efficient fine-tuning with a mixture of dyadic experts. In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 8233–8246.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

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

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

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2023. Dylora: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3274–3287.

Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. 2022. Multi-task learning for dense prediction tasks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(7):3614–3633.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2022. Spot: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Xujia Wang, Haiyan Zhao, Shuo Wang, Hanqing Wang, and Zhiyuan Liu. 2025. Malora: Mixture of asymmetric low-rank adaptation for enhanced multi-task learning. In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 5609–5626.

Yiming Wang, Yu Lin, Xiaodong Zeng, and Guannan Zhang. 2023a. Multilora: Democratizing lora for better multi-task learning. *CoRR*, abs/2311.11501.

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. 2023b. Multitask prompt tuning enables parameter-efficient transfer learning. In *The Eleventh International Conference on Learning Representations*.

Taiqiang Wu, Jiahao Wang, Zhe Zhao, and Ngai Wong. 2024a. Mixture-of-subspaces in low-rank adaptation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

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

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, and other. 2025a. Qwen3 technical report. *CoRR*, abs/2505.09388.

Yaming Yang, Dilxat Muhtar, Yelong Shen, Yuefeng Zhan, Jianfeng Liu, Yujing Wang, Hao Sun, Weiwei Deng, Feng Sun, Qi Zhang, Weizhu Chen, and Yunhai Tong. 2025b. Mtl-lora: Low-rank adaptation for multi-task learning. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 22010–22018.

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

Dacao Zhang, Kun Zhang, Shimao Chu, Le Wu, Xin Li, and Si Wei. 2025. MoRE: A mixture of low-rank experts for adaptive multi-task learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1311–1324.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. 2023a. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning. *CoRR*, abs/2308.03303.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023b. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*.

Yu Zhang and Qiang Yang. 2022. A survey on multi-task learning. *IEEE Trans. Knowl. Data Eng.*, 34(12):5586–5609.

Lulu Zhao, Weihao Zeng, Xiaofeng Shi, and Hua Zhou. 2025. Mosld: An extremely parameter-efficient mixture-of-shared loras for multi-task learning. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 1647–1659.

Jiacheng Zhu, Kristjan H. Greenewald, Kimia Nadjahi, Haitz Sáez de Ocáriz Borde, Rickard Brüel Gabrielsson, Leshem Choshen, Marzyeh Ghassemi, Mikhail Yurochkin, and Justin Solomon. 2024. Asymmetry in low-rank adapters of foundation models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*.

A Details of MoSE Framework

A.1 Gating Router Design

The gating functions in Eq.(4) and (5) are two structurally identical, independent multi-class probability routers (G_r, G_s). Each applies a Kaiming-initialized linear layer (with parameters W, b) and Softmax to maps the input to raw expert scores. Since the routing decisions are non-differentiable, we define $sg(\cdot)$ as the stop gradient operation. During forward pass, output equals y (one-hot for G_r or multi-hot vector for G_s). During backward pass, gradients flow through logits: output $=$ logits $-$ $sg(\text{logits}) + sg(y)$.

A.2 Rank and Convergence Slope Estimation

We estimate the trend of rank selection in the gating network $G_r(\cdot)$ and the convergence slope by performing linear regression through the least squares method.

Rank Trend Specifically, over a consecutive sequence of m training steps, we record the expert selection probability distribution $P_i \in \mathbb{R}^{m \times n}$ for task i. The expected expert rank $\hat{y}_{i,t}$ of task i at each time step t is then computed as $\hat{y}_{i,t} = \sum k \times P_i(t,k)$, where k denotes the expert index. Then, we fit a linear model $\hat{y}_{i,t} = \alpha_0 + \alpha_i t$ through least squares regression, yielding the estimated slope α_i. This slope reflects the directional trend of expert rank selection over time.

$$\alpha_i = \frac{m \sum_{t=1}^{m} (t \times \hat{y}_{i,t}) - \sum_{t=1}^{m} t \times \sum_{t=1}^{m} \hat{y}_{i,t}}{m \sum_{t=1}^{m} t^2 - (\sum_{t=1}^{m} t)^2} \tag{12}$$

Convergence Slope We quantify the convergence speed of each task by analyzing the slope of its validation loss curve. To facilitate a fair comparison across tasks, the validation losses are first normalized. Specifically, we define the normalized loss ratio:

$$\bar{\mathcal{L}}_i(\theta; t) = \frac{\mathcal{L}_i(\theta; t)}{\mathcal{L}_i(\theta; 0)} \tag{13}$$

where $\mathcal{L}_i(\theta; 0)$ is the initial validation loss for task i. A linear model $\beta_i t + \gamma_i$ is then fitted to the normalized loss trajectory over a recent window of w steps. The slope β_i serves as an estimate of the local convergence rate. Formally, at step t we collect the recent data points starting from $s_0 = \max(0, t - w + 1)$. Finally, the slope β_i is calculated using Eq.(14).

A.3 Contrastive Learning

The objective of $\mathcal{L}_{\text{con}}$ is to bring sample representations closer to their corresponding task embedding and push them farther from those of other tasks. For a sample set of size N, the softmax function is applied to each sample representation h_i over its similarity scores with the full set of task embeddings $\{e_k\}_{k=1}^{T}$. The numerator measures the similarity between h_i and its corresponding task embedding e_i, whereas the denominator sums the exponentiated similarities to all task embeddings, scaled by the temperature parameter τ.

$$\mathcal{L}_{\text{con}} = -\frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{\exp\left(\text{sim}(h_i, e_i)/\tau\right)}{\sum_{k=1}^{T} \exp\left(\text{sim}(h_i, e_k)/\tau\right)} \right) \tag{14}$$

B Datasets Statistics

For a comprehensive evaluation, we employed the standard GLUE benchmark (Wang et al., 2018) alongside more challenging datasets, including BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), OBQA (Mihaylov et al., 2018), and ARC (Clark et al., 2018), covering a wide range of tasks. For the supervised learning experiments, training was conducted utilizing a maximum of 20,000 samples for each dataset. To assess robustness in low-resource settings, we further included SciTail (Khot et al., 2018), BoolQ, and CB (de Marneffe et al., 2019) datasets. The details of data statistics are provided in Table 5.

C Implementation Details

We employ LLaMA2-7B, T5-base, T5-3B, and Qwen3-4B[8](Yang et al., 2025a) as backbone models, optimizing them with AdamW. LoRA is applied to the query, key, value, output, and feed-forward layers in T5, and to the query, key, up-projection, and down-projection layers in Qwen3 and LLaMA2. Training was conducted utilizing four NVIDIA A100 GPUs for T5 and eight NVIDIA H20 GPUs for LLaMA2 and Qwen3. In the main and few-shot experiments (Tables 2, 3), the original publications only present a single best result without standard deviations, and do not disclose the detailed hyperparameters necessary for exact reproduction. Consequently, we report the best performance obtained across three independent runs. For prior and ablation studies (Tables 1, 4, 7), all experiments are performed with fixed

[8]https://huggingface.co/collections/Qwen/qwen3

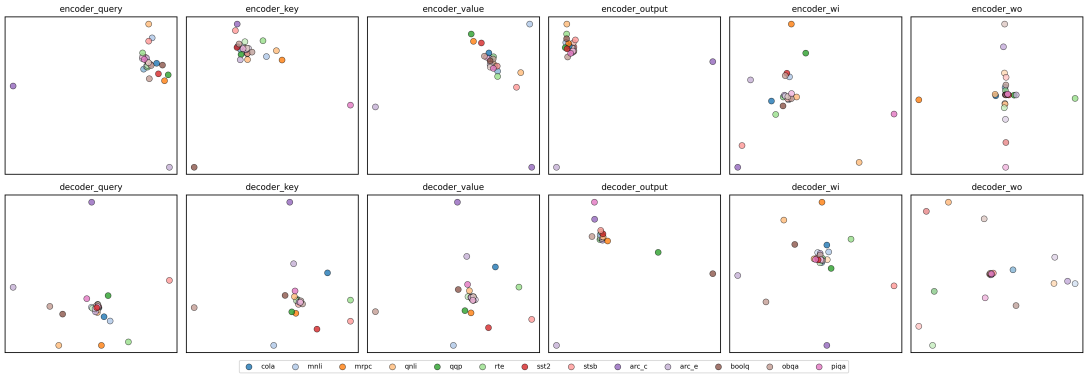


Figure 6: PCA visualization of task embeddings from layer 6 in the MoRE framework. Lighter and darker shades correspond to the embedding distributions before and after training, respectively.

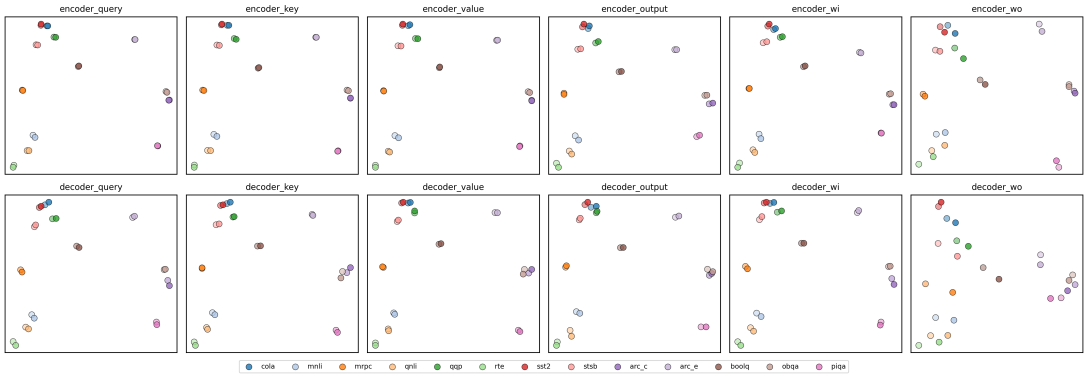


Figure 7: PCA visualization of layer-6 task embeddings in the MoSE framework, illustrating the evolution of the distribution from pre-training (lighter shades) to post-training (darker shades).

hyperparameters over three random seeds. This design guarantees fair and reproducible comparisons, and further validates the robustness of our results.

D Supplementary Experiments

D.1 Cross-Architecture Evaluation

To ensure a comprehensive evaluation and robust conclusions, we conduct extensive experiments across multiple model architectures. As summarized in Table 7, MoSE achieves superior average performance, with particularly pronounced gains on challenging tasks such as ARC and PIQA. These results consistently validate the effectiveness of our approach and highlight its **strong scalability and architectural adaptability**. Furthermore, we observe that decoder-only models are better suited for commonsense reasoning, while encoder-decoder models excel at language understanding, offering insights for matching model biases to tasks.

D.2 Comparison of Routing Strategies

To enable a rigorous comparison between the two routing mechanisms, we performed a preliminary experiment with fixed hyperparameters and setup, yielding the two variants under study. The experimental results in Table 8 indicate that the hard top-k routing strategy yields an average performance of 85.22%, whereas the soft (weighted) routing variant (applied exclusively to subspace selection) achieves 84.30%. On straightforward tasks such as MRPC, SST-2, and QNLI, hard and soft routing exhibit comparable performance, as well-defined task patterns diminish the need for selective routing. In contrast, on more challenging tasks including RTE and BoolQ, **hard routing demonstrates a distinct advantage by selecting high-quality, complementary subspaces more effectively, rather than uniformly utilizing all available subspaces**. Furthermore, the mere 0.92% performance gap between the two routing strategies is considerably smaller than the overall 3.2% improvement brought by `Bi-Router` over the baseline. In conclusion, our core contribution lies in fine-grained low-rank subspace selection. This design **alleviates the subspace activation imbalance and routing inflexibility** inherent in conventional LoRA-MoE models. The selection between hard top-k and soft routing is merely an **implementation detail** and does not

Corpus	Domain	Task	# Train	# Dev	# Test
BoolQ	Wikipedia	Text Classification	9427	3270	3245
ARC-E	Natural Science	Question Answering	2251	570	2376
ARC-C	Natural Science	Question Answering	1119	299	1172
OBQA	Science Facts	Question Answering	4957	500	500
PIQA	Physical Interaction	Question Answering	16113	1838	3084
CoLA	Music	Linguistic Acceptability	8551	1043	1063
SST-2	Movie Reviews	Sentiment Classification	67349	872	1821
MRPC	News	Paraphrase Detection	3668	408	1725
STS-B	Music	Sentence Similarity	5749	1500	1379
QQP	Social QA	Paraphrase Detection	363846	40430	390965
MNLI	Music	Natural Language Inference	392702	9815/9832	9796/9847
QNLI	Wikipedia	Natural Language Inference	104743	5463	5463
RTE	News, Wikipedia	Natural Language Inference	2490	277	3000
CB	Commitment	Natural Language Inference	250	56	250
SciTail	Natural Science	Natural Language Inference	23,097	1,304	2126

Table 5: Statistics of the benchmark datasets used in the experiments. The table lists the corpus, domain, task type, and the number of samples for training, development, and test splits. For MNLI, the development and test sets are reported for the matched and mismatched subsets separately.

param	η ($\mu = 0.01, \kappa = 1.5$)						μ ($\eta = 0.4, \kappa = 1.0$)					
	0.1	0.2	0.4	0.6	0.8	1.0	0.01	0.02	0.04	0.06	0.08	0.1
MNLI	**85.49**	83.48	85.28	84.18	84.33	83.53	85.34	85.34	**85.63**	85.10	85.24	84.97
MRPC	82.35	88.73	**91.18**	86.76	**91.18**	90.69	89.71	**90.69**	89.71	89.71	88.24	89.22
QNLI	92.53	92.48	**92.57**	91.25	92.40	90.96	91.80	92.35	92.26	92.31	**92.51**	92.31
RTE	74.10	69.06	74.82	**75.54**	74.10	69.78	76.26	76.98	**80.58**	79.14	74.82	74.10
Avg.	81.74	80.87	**83.34**	82.00	82.45	82.38	84.52	84.69	**84.73**	84.39	83.96	83.35

Table 6: Average performance based on T5-base under varying η (Section 4.4) and μ (Section 4.1). The best results are highlighted in **bold**, the second-best are underlined, and "Avg." indicates the average performance of GLUE benchmark.

alter the core novelty of our proposed method.

D.3 Quantitative Validation

Activation Frequency. In the MoRE framework, once expert E_i is selected, all of its subspaces are updated, including low-index subspaces (e.g., $\{a_1b_1, a_2b_2, ...\}$) that may be redundant to the current task. On the STSB task, MoSE **reduces the standard deviation of subspace activation frequency from 19.85% to 1.66%**. This represents a 91.6% decrease, and detailed results are presented in Table 9. This reduction further results in a nearly uniform activation distribution across subspaces.

Flexibility. The perform routing of MoRE is confined to the rank level. It activates a fixed set of subspaces and lacks flexible control over expert allocation. In contrast, MoSE supports fine-grained routing by selectively activating $\hat{r}$ out of r total subspaces where $\hat{r} \leq r$. This setting yields $\binom{r}{\hat{r}}$ possible subspace combinations and substantially improves representation diversity and routing flexibility. For a typical LoRA module with $r = 8$, MoSE **offers 8× to 70× more candidate combinations** than MoRE.

D.4 Training and Inference Overhead

We recognize the need to profile training and inference overhead for deployment. Since the original MoRE paper lacks absolute runtime data, we compare our `Bi-Router` with a reproduced MoRE baseline under identical hardware and setup.

Training. As shown in Table 10, MoSE delivers faster and more efficient training, achieving **a speedup of 1.1×** over the baseline. Training time generally scales with the number of updated parameters. Compared with MoRE, MoSE introduces merely $6Lhr$[9] additional parameters, leading to negligible computational overhead in both forward and backward propagation. The primary bottleneck of training efficiency arises from routing load imbalance. Excessive activation of certain subspaces while others remain idle wastes computational resources and incurs unnecessary synchronization delays. Our `Bi-Router` alleviates this issue by activating only the required subspaces. Unlike conventional nested architectures that redundantly activate low-index subspaces, our precise selection strategy improves the utilization of high-index subspaces, thereby boosting overall training efficiency. Furthermore, our method is only around 1.4× slower than standard LoRA. In contrast, other mixture-based adapter methods exhibit considerably higher latency. MultiLoRA (2.9×), MoELoRA (3.6×), and MixLoRA (4.2×) suffer from heavy expert management overhead and GPU control-flow divergence, resulting in substantially degraded training speed.

Inference. As shown in Table 11, our inference overhead is approximately 8-10%, which mainly comes from one additional forward pass per layer in the cascaded `Bi-Router`. Nevertheless, this overhead is still lower than that of standard MoE routing, which incurs 20-30% extra cost. Benefiting from subspace-level routing, our method avoids expensive all-to-all communication. In summary, MoSE **achieves competitive performance while introducing only marginal inference overhead**.

D.5 Parameter Efficiency

The evaluation of model complexity is based on the count of trainable parameters. We use L to denote the number of layers, r for the LoRA rank, m and d for the model dimensions, n for the number of experts, T for the number of tasks, and h for the task embedding dimension. The total parameter count for MoSE yields $6Lr(m+d)+6Lh(2r+T)$. The overhead versus standard LoRA is $6Lh(2r+T)$, which constitutes task embeddings Th and dual selector modules $2rh$ for each expert, resulting in a relatively small parameter increase.

D.6 Sensitivity Analysis

For common parameters such as rank, learning rate, and batch size, we scale their values according to model size (e.g., smaller batch size for larger models), following standard practice. Details are provided in Table 12. As shown in Appendix Table 6, initializing the adjustment coefficient η with excessively large values causes the dynamic sampling mechanism to fail and degrades overall performance. In contrast, stable and optimal performance is achieved at $\eta = 0.4$. The learning rate scaling factor μ exhibits a clear performance peak: accuracy improves with increasing μ up to an optimal threshold ($\mu = 0.04$) and then declines. This is because small values of μ lead to insufficient adaptation of task vectors, while large values disrupt pre-trained initialization. Additionally, the value of λ_1 is chosen in accordance with prior studies to facilitate a fair comparison. The window size w

[9] Additional details are provided in Section D.5.

Method	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2
T5-3B							
MoRE	66.31 $\pm$ 0.29	89.42 $\pm$ 0.43	87.25 $\pm$ 0.35	**95.15** $\pm$ 0.05	**89.67** $\pm$ 0.41	92.09 $\pm$ 0.26	95.41 $\pm$ 0.08
MoSE	**69.04** $\pm$ 0.24	**90.39** $\pm$ 0.36	**91.67** $\pm$ 0.27	94.16 $\pm$ 0.02	88.60 $\pm$ 0.17	**93.53** $\pm$ 0.24	**95.76** $\pm$ 0.18
	STSB	ARC-C	ARC-E	BoolQ	OBQA	PIQA	**Avg**.
T5-3B							
MoRE	**91.41** $\pm$ 0.22	51.28 $\pm$ 0.43	69.15 $\pm$ 0.34	**85.50** $\pm$ 0.26	63.00 $\pm$ 0.31	73.83 $\pm$ 0.25	80.73 $\pm$ 0.26
MoSE	91.03 $\pm$ 0.26	**65.53** $\pm$ 0.35	**77.86** $\pm$ 0.25	84.89 $\pm$ 0.29	**74.40** $\pm$ 0.17	**80.47** $\pm$ 0.23	**84.41** $\pm$ 0.12
	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2
Qwen3-4B							
MoRE	65.19 $\pm$ 0.02	81.57 $\pm$ 0.10	87.75 $\pm$ 0.29	90.65 $\pm$ 0.13	84.80 $\pm$ 0.07	**92.09** $\pm$ 0.13	93.92 $\pm$ 0.15
MoSE	**72.58** $\pm$ 0.13	**86.36** $\pm$ 0.14	**88.73** $\pm$ 0.09	**92.49** $\pm$ 0.12	**86.06** $\pm$ 0.20	89.93 $\pm$ 0.12	**95.87** $\pm$ 0.14
	STSB	ARC-C	ARC-E	BoolQ	OBQA	PIQA	**Avg.**
Qwen3-4B							
MoRE	92.00 $\pm$ 0.25	87.29 $\pm$ 0.17	94.02 $\pm$ 0.16	84.34 $\pm$ 0.23	86.40 $\pm$ 0.20	83.13 $\pm$ 0.26	86.40 $\pm$ 0.08
MoSE	**93.87** $\pm$ 0.32	**89.42** $\pm$ 0.31	**95.88** $\pm$ 0.28	**87.40** $\pm$ 0.17	**87.60** $\pm$ 0.16	**86.07** $\pm$ 0.09	**88.63** $\pm$ 0.15

Table 7: Comparative performance of MoRE and MoSE on 13 tasks, based on *T5-3B* and *Qwen3-4B* backbones. The best results are denoted in **bold**.

Task	**Hard Top-K**	**Soft Routing**
MRPC	89.71 $\pm$ 0.69	**90.04** $\pm$ 0.28
RTE	**73.86** $\pm$ 0.68	71.22 $\pm$ 0.72
SST-2	**93.08** $\pm$ 0.47	93.01 $\pm$ 0.39
BoolQ	**77.37** $\pm$ 0.13	74.72 $\pm$ 1.56
QNLI	92.11 $\pm$ 0.40	**92.54** $\pm$ 0.08
Avg. (%)	**85.22** $\pm$ 0.15	84.30 $\pm$ 0.24

Table 8: Empirical performance of the two MoSE routing variants.

is set to 4 after sensitivity analysis: while configurations with $w = 6$ and $w = 8$ yield comparable performance, they entail significantly higher computational overhead. Parameters such as λ_2 and κ are activated only during specific training steps, resulting in negligible disturbance to both the loss function and gradient flow. Preliminary results indicate that the performance variance induced by these parameters is minimal (standard deviation of 0.04%), which is substantially smaller than our performance gains (1.48$\sim$1.94%) over MoRE. This confirms that the observed improvements are driven by our novel architectural design, rather than meticulous hyperparameter optimization.

Index	MoRE Fq. (%)	**MoSE Fq. (%)**	Δ (%)
0	65.7	43.7	-22.0
1	56.5	46.8	-9.7
2	48.4	44.4	-4.0
3	39.4	44.4	+5.0
4	32.0	41.1	+9.1
5	24.5	45.3	+20.8
6	16.6	45.0	+28.4
7	8.3	45.4	+37.1
Std.	19.85	1.66	**$\downarrow$91.6**

Table 9: A comparison of subspace activation frequencies between MoRE and MoSE. Here, "Index" denotes the index of the subspace.

Method	Train Time	Samples	Steps	Speedup
MoRE	100,077.30	4.00	0.25	1.0×
Bi-Router + STE	90177.58	4.43	1.26	**1.1×**

Table 10: Training speed comparison of different methods. "Samples" denote the number of samples processed per second, while "steps" represent the number of forward passes executed per second.

Task	MoRE	**Bi-Router + STE**	Δ (%)
MNLI	5,615	6,122	+9.0
QQP	11,941	13,170	+10.3
QNLI	1,615	1,773	+9.8
SST-2	227	246	+8.4
CoLA	140	151	+7.8
STS-B	202	220	+8.9
MRPC	63	69	+9.5
RTE	44	48	+9.8
BoolQ	424	458	+8.0
ARC-C	303	327	+7.9
ARC-E	615	667	+8.5
OBQA	130	141	+8.5
PIQA	494	538	+8.9
Avg.	-	-	**+8.9**

Table 11: Inference runtime comparison between MoRE and `Bi-Router` + STE across different tasks.

# Params	**T5-base**	**T5-3B**	**Qwen3-4B**	**LLaMA2-7B**
Learning Rate	1e-3	3e-04	2e-05	5e-6
Batch size	32	8	6	1
Accum. Steps	4	4	8	8
# Epochs	3	3	3	5
Rank r	16	8	8	8
# Experts	16	8	8	8

Table 12: Hyperparameter configurations for fine-tuning MoSE on T5-base, T5-3B, LLaMA2-7B, and Qwen3-4B.

# Params	**Value**
Cutoff Length	512
Optimizer	AdamW
Dropout	0.1
LoRA Alpha α	32
Weight λ_1 ($\mathcal{L}_{\text{con}}$)	0.1
Weight λ_2 ($\mathcal{L}_{\text{add}}$)	0.5
Adjustment Intensity η	0.4
Amplification Factor κ	1.0
Scaling Coefficient μ	0.04
# Window Size w	4

Table 13: General hyperparameters for all models.