

GSR-LoRA: Group-Sparse Routing-aware Low-Rank Adapter Compression for Multi-Task Mixture-of-Experts Fine-Tuning

Weimin Cheng¹, Chengyu Zhao¹, Xuanjing Huang², Yatian Shen^{1,*}

¹School of Computer and Information Engineering, Henan University

²College of Computer Science and Artificial Intelligence, Fudan University

{cwm, zhaochengyu, ytshen}@henu.edu.cn, xjhuang@fudan.edu.cn

*Corresponding author

Abstract

In multi-task mixture-of-experts parameter-efficient fine-tuning, fixed-rank Low-Rank Adaptation (LoRA) adapters often struggle to accommodate the heterogeneous representation requirements across different tasks and network layers. Existing methods mainly focus on task-level expert selection, layer-wise rank allocation, or post-training parameter compression, making it difficult to jointly characterize low-rank capacity differences across tasks, layers, and modules within a unified framework. To address this issue, we propose Group-Sparse Routing-aware Low-Rank Adaptation (GSR-LoRA), a routing-aware structured rank compression method for multi-task parameter-efficient fine-tuning (PEFT). Specifically, we model low-rank components as selectable rank groups and use a task-dependent router to allocate rank capacity across tasks. Furthermore, we introduce a Group Lasso structured sparsity constraint to suppress redundant rank groups during training, forming a prunable adapter structure. During compression, we select rank groups by jointly considering parameter importance and routing usage signals, and map the router to the compressed rank space to obtain compact LoRA adapters that remain consistent with task routing. Experiments on GLUE show that GSR-LoRA reduces the average adapter rank from 16 to 5.5, removing 65.6% of rank components with only a 0.1-point performance degradation. These results demonstrate its ability to learn compact task-adaptive LoRA adapters while preserving multi-task performance. The code is available at <https://github.com/HNLP93/GSR-LoRA>.

1 Introduction

Multi-task learning aims to improve model generalization through knowledge sharing across tasks (Caruana, 1997; Zhang et al., 2023; Zhang and Yang, 2022; Chen et al., 2024; Yu et al., 2024).

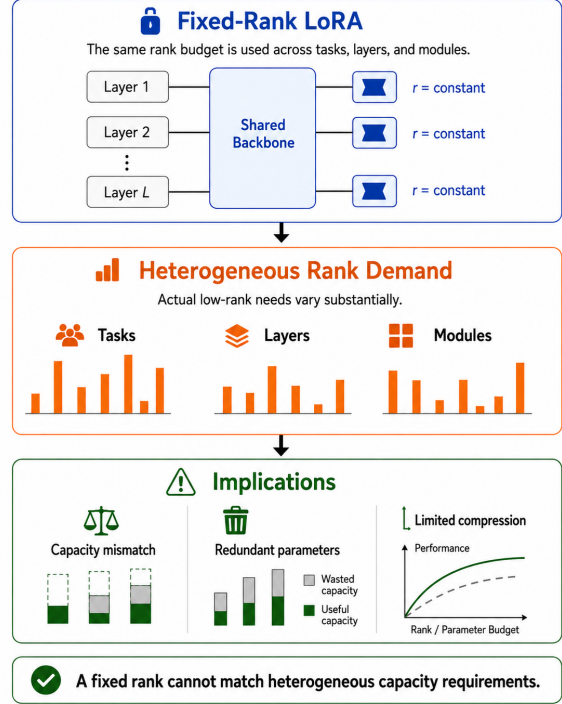


Figure 1: Standard LoRA assigns the same rank budget across tasks, layers, and modules, while the actual low-rank capacity demands may vary substantially. This mismatch can lead to rigid rank allocation, redundant parameters, and limited compression potential.

However, substantial semantic differences, objective discrepancies, and variations in optimization difficulty commonly exist among tasks, making it challenging for a unified parameter adaptation structure to balance expressive capacity and parameter efficiency. Mixture-of-experts models dynamically select substructures that provide different tasks with more appropriate parameter paths, thereby modeling task heterogeneity and achieving a better balance between shared and task-specific modeling (Ma et al., 2018; Cai et al., 2025; Masoudnia and Ebrahimpour, 2014; Liu et al., 2026; Pan et al., 2026). Meanwhile, parameter-efficient fine-tuning significantly reduces the adaptation cost

of large models by updating only a small number of additional parameters (Han et al., 2024; Houlsby et al., 2019; Ding et al., 2023b; Wang et al., 2025a). Among these methods, LoRA achieves a favorable trade-off between efficiency and performance through low-rank parameterization, and has become a representative approach in parameter-efficient fine-tuning (Hu et al., 2022; Mao et al., 2025; Zhu and Nguyen, 2025).

However, standard LoRA typically adopts a pre-defined fixed-rank configuration, where different tasks, layers, and modules share the same low-rank budget. As illustrated in Figure 1, although this design simplifies implementation, it also introduces clear limitations. In multi-task MoE fine-tuning, different tasks may exhibit diverse semantic objectives, optimization difficulties, and representation requirements, while different layers and modules also play distinct roles in feature transformation. Therefore, the fixed-rank LoRA can hardly match the heterogeneous low-rank adaptation demands across tasks, layers, and modules.

Existing studies have improved fixed-rank LoRA along three main directions. Multi-task low-rank expert methods, such as MoRE (Zhang et al., 2025), MoDE (Ning et al., 2025), and MALoRA (Wang et al., 2025b), enhance adaptation flexibility through task-aware expert selection or low-rank substructure composition, but mainly focus on task-level differences. Adaptive-rank methods, including ALoRA (Liu et al., 2024), AutoLoRA (Zhang et al., 2024b), DyLoRA (Valipour et al., 2023), SalientLoRA (Ke et al., 2024), AROMA (Sheng et al., 2025), and GeLoRA (Ed-dib et al., 2025), adjust rank budgets based on layer importance or component contribution, yet seldom model task-specific capacity demands. Sparsification and compression methods, such as SoRA (Ding et al., 2023a), RoseLoRA (Wang et al., 2024), LoRAPrune (Zhang et al., 2024a), and SVD-based LoRA compression (Chen and Zheng, 2026), reduce adapter redundancy but typically treat compression independently from multi-task routing. Thus, existing methods rarely capture the coupled heterogeneity of low-rank demands across tasks, layers, and modules within a unified multi-task PEFT framework.

Based on the above issue, we propose GSR-LoRA, a routing-aware structured LoRA rank compression method for multi-task PEFT. GSR-LoRA views LoRA rank groups as lightweight low-rank experts within a shared adapter space, enabling

task-dependent capacity allocation without introducing separate expert modules. Different from prefix-based dynamic-rank methods, which assume that important rank components appear in earlier positions, GSR-LoRA allows non-prefix rank groups to be retained during compression. Specifically, we use a task-dependent router to allocate rank groups across tasks and introduce Group Lasso structured sparsity to suppress redundant groups during training. During compression, rank groups are selected according to both parameter importance and router usage, and the router is remapped to the compressed rank space for consistent task-specific routing.

Experimental results on the GLUE benchmark show that the proposed method improves low-rank parameter utilization while preserving multi-task adaptability, and learns compact adapter structures with heterogeneous rank distributions. Specifically, under an initial rank setting of 16, GSR-LoRA compresses the average retained rank to 5.5, removing approximately 65.6% of rank components with only a 0.1-point drop in average performance. In addition, layer-wise and module-wise analysis shows that the retained ranks are not reduced uniformly, but exhibit differentiated patterns across attention and feed-forward modules. This observation verifies the structural heterogeneity of low-rank demands in multi-task settings and indicates that the proposed method can adapt more finely to the representational needs of different substructures.

The main contributions of this work are summarized as follows:

- We propose GSR-LoRA, a routing-aware structured rank compression framework that adaptively allocates LoRA capacity across tasks, layers, and modules for multi-task PEFT.
- We combine task-dependent rank routing with Group Lasso structured sparsity, enabling redundant rank groups to be learned and pruned in a unified training-to-compression pipeline.
- We design a router-aware group selection strategy based on parameter importance and router usage, with router remapping to the compressed rank space. On GLUE benchmark, GSR-LoRA removes 65.6% of rank components with only a 0.1-point performance degradation.

2 Related Work

2.1 Multi-task PEFT and Low-rank Expert Modeling

Recent studies have improved fixed-rank LoRA for multi-task PEFT by introducing task-aware low-rank experts or selection mechanisms. Representative methods include MoRE, which formulates multi-task adaptation as a mixture of low-rank experts with adaptive rank selection; MoDE, which composes finer-grained dyadic experts; and MALoRA, which improves multi-task adaptation through asymmetric low-rank experts (Zhang et al., 2025; Ning et al., 2025; Wang et al., 2025b). These methods show that expert selection and task-aware low-rank routing can enhance the flexibility of multi-task PEFT. However, they mainly focus on task-level expert allocation or low-rank substructure composition, while paying less attention to layer-level and module-level differences in low-rank capacity requirements.

2.2 Adaptive Rank Allocation for LoRA

Another line of work explores adaptive rank allocation to overcome the rigidity of manually specified LoRA ranks. DyLoRA learns ordered and nested low-rank representations that support multiple prefix rank budgets (Valipour et al., 2023). ALoRA reallocates LoRA budgets according to rank importance (Liu et al., 2024), while AutoLoRA formulates rank selection as a meta-learning problem (Zhang et al., 2024b). Other methods further study adaptive rank construction from different perspectives, such as salience-based intrinsic rank analysis in SalientLoRA, rank-one component composition in AROMA, and geometry-guided rank selection in GeLoRA (Ke et al., 2024; Sheng et al., 2025; Ed-dib et al., 2025). Although these methods improve rank flexibility, they mainly estimate rank budgets from layer-wise or component-wise importance and seldom model task-specific capacity demands in multi-task settings.

2.3 Sparsification, Pruning, and Compression of LoRA Adapters

LoRA sparsification and compression methods aim to reduce adapter redundancy for more efficient deployment. SoRA introduces sparse low-rank adaptation with dynamic rank control, RoseLoRA imposes row- and column-wise sparsity on low-rank updates, and LoRAPrune combines structured pruning with parameter-efficient fine-tuning (Ding et al.,

2023a; Wang et al., 2024; Zhang et al., 2024a). Recent post-training methods further compress trained LoRA adapters through singular value decomposition based reconstruction (Chen and Zheng, 2026). These studies demonstrate that LoRA adapters contain substantial structural redundancy. However, most of them focus on parameter removal or post-training reconstruction, while providing limited modeling of how redundant low-rank components emerge during multi-task training and how compression should interact with task-specific capacity requirements.

3 Methodology

GSR-LoRA follows a two-stage design. During training, GSR-LoRA learns task-dependent effective ranks through prefix routing and induces structured sparsity over LoRA rank groups. After training, GSR-LoRA converts the learned sparsity into real compression through router-aware non-prefix group selection.

3.1 Problem Setup

We consider multi-task PEFT with LoRA adapters. Given a frozen linear weight $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, LoRA introduces a low-rank update:

$$\Delta W = BA, \quad A \in \mathbb{R}^{r_{\text{max}} \times d_{\text{in}}}, \quad B \in \mathbb{R}^{d_{\text{out}} \times r_{\text{max}}}, \quad (1)$$

where r_{max} is the initial maximum rank. Standard LoRA assigns the same rank budget to all tasks, layers, and modules, which may be inefficient under heterogeneous multi-task demands.

We decompose each LoRA adapter into rank groups and view each group as a lightweight low-rank expert. For layer l , the k -th rank group is:

$$g_k^{(l)} = \{A_{[k,:]}^{(l)}, B_{[:,k]}^{(l)}\}. \quad (2)$$

Accordingly, the LoRA update is:

$$\Delta W^{(l)} = \sum_{k=1}^{r_{\text{max}}} B_{[:,k]}^{(l)} A_{[k,:]}^{(l)}. \quad (3)$$

3.2 Task-dependent Prefix Routing

To allocate different effective ranks to different tasks, we introduce a task-dependent router for each LoRA module. Given a task embedding e_t , the router predicts a distribution over candidate prefix ranks:

$$\mathbf{p}_t^{(l)} = \text{softmax}\left(G^{(l)}(e_t)\right), \quad \mathbf{p}_t^{(l)} \in \mathbb{R}^{r_{\text{max}}}. \quad (4)$$

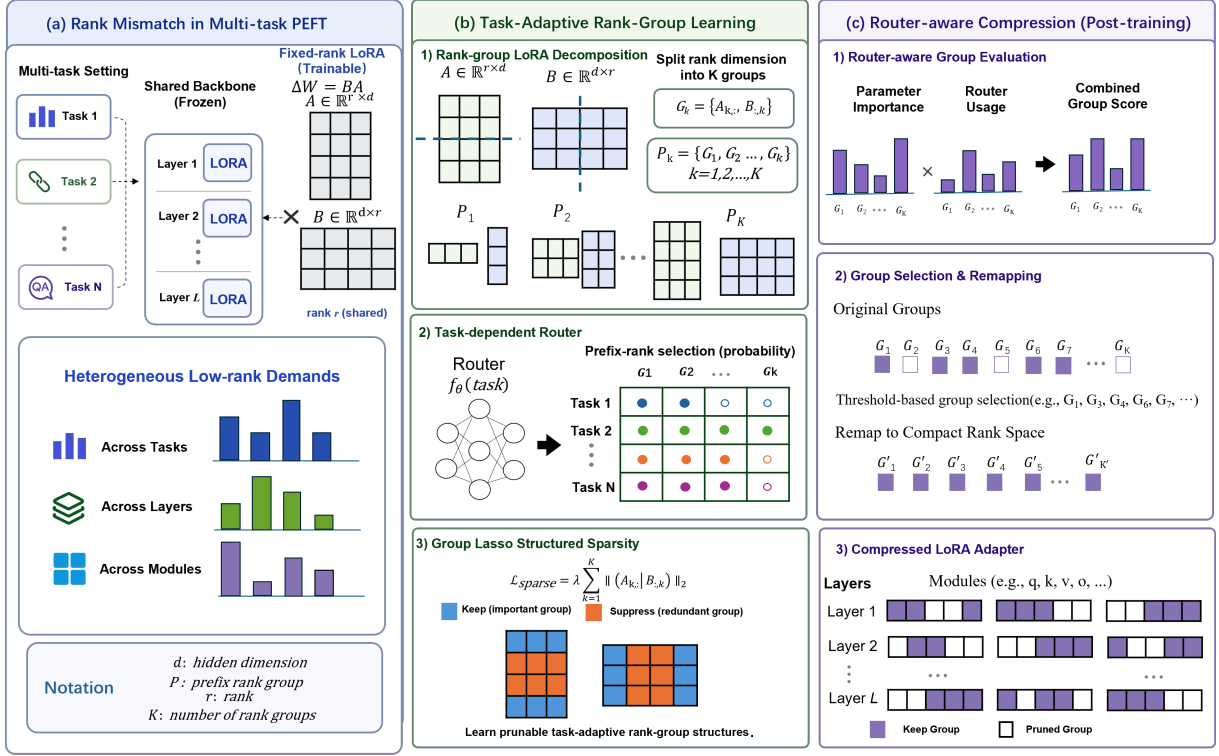


Figure 2: Overview of GSR-LoRA. (a) Fixed-rank LoRA assigns the same rank budget to all tasks, layers and modules. (b) During training, GSR-LoRA treats LoRA rank groups as lightweight low-rank experts and learns task-dependent prefix routing under Group Lasso sparsity. (c) During compression, router-aware non-prefix group selection removes redundant rank groups and remaps the prefix router to the compressed rank space.

During training, we use a straight-through argmax operation to obtain a prefix-rank decision:

$$\mathbf{a}_t^{(l)} = \text{STArgmax}(\mathbf{p}_t^{(l)}), \quad (5)$$

where $a_{t,r}^{(l)} = 1$ means that task t selects prefix rank r at layer l . The corresponding prefix mask over rank groups is:

$$m_{t,k}^{(l)} = \sum_{r=k}^{r_{\max}} a_{t,r}^{(l)}, \quad m_{t,k}^{(l)} \in \{0, 1\}. \quad (6)$$

Thus, selecting prefix rank r activates rank groups $1, \dots, r$ and masks out the remaining groups. The task-specific LoRA update is:

$$\Delta W_t^{(l)} = \sum_{k=1}^{r_{\max}} m_{t,k}^{(l)} B_{[:,k]}^{(l)} A_{[k,:]}^{(l)}. \quad (7)$$

3.3 Group-sparse Multi-task Training

During multi-task training, we first sample a task t according to the task-sampling distribution π and then draw a mini-batch $\mathcal{B}_t$ from its dataset $\mathcal{D}_t$. All examples in a mini-batch therefore belong to the same task.

For the sampled task, the complete training objective is:

$$\mathcal{L} = \mathcal{L}_{\text{s2s}} + \lambda_{\text{cl}} \mathcal{L}_{\text{cl}} + \lambda_{\text{gl}} \mathcal{L}_{\text{gl}} + \lambda_{\text{ent}} \mathcal{L}_{\text{ent}} + \lambda_{\text{rank}} \mathcal{L}_{\text{rank}}, \quad (8)$$

where $\mathcal{L}_{\text{s2s}}$ is the token-level sequence-to-sequence cross-entropy loss of T5.

Task-embedding contrastive loss. Let $\mathbf{h}_b^{(l)}$ denote the representation of example b at LoRA module l . We define:

$$z_{b,j}^{(l)} = \frac{\cos(\mathbf{h}_b^{(l)}, \mathbf{e}_j)}{\tau_{\text{cl}}}, \quad (9)$$

and:

$$\mathcal{L}_{\text{cl}} = -\frac{1}{L|\mathcal{B}_t|} \sum_{l=1}^L \sum_{b \in \mathcal{B}_t} \log \frac{\exp(z_{b,t}^{(l)})}{\sum_{j=1}^T \exp(z_{b,j}^{(l)})}. \quad (10)$$

This objective encourages representations from the sampled task to align with the corresponding task embedding while remaining distinguishable from other task embeddings.

Group Lasso regularization. For each rank group, we define:

$$s_k^{(l)} = \sqrt{\|A_{[k,:]}^{(l)}\|_2^2 + \|B_{[:,k]}^{(l)}\|_2^2} + \epsilon, \quad (11)$$

and impose:

$$\mathcal{L}_{\text{gl}} = \sum_{l=1}^L \sum_{k=1}^{r_{\max}} s_k^{(l)}. \quad (12)$$

This regularizer promotes structured sparsity at the rank-group level by jointly shrinking the corresponding row of A and column of B , providing a structured basis for subsequent rank-group compression.

Router regularization. Using the soft prefix-rank probabilities $\mathbf{p}_t^{(l)}$ defined in Sec. 3.2, we minimize the normalized router entropy:

$$\mathcal{L}_{\text{ent}} = -\frac{1}{L} \sum_{l=1}^L \frac{\sum_{k=1}^{r_{\max}} p_{t,k}^{(l)} \log p_{t,k}^{(l)}}{\log r_{\max}}, \quad (13)$$

and the expected normalized rank:

$$\mathcal{L}_{\text{rank}} = \frac{1}{L} \sum_{l=1}^L \sum_{k=1}^{r_{\max}} p_{t,k}^{(l)} \frac{k}{r_{\max}}. \quad (14)$$

The entropy regularizer encourages more decisive prefix-rank distributions, whereas the rank regularizer biases the router toward smaller effective ranks.

3.4 Router-aware Non-prefix Compression

Although training uses prefix routing for stable capacity control, important rank groups are not necessarily concentrated in prefix positions. Therefore, after training, GSR-LoRA performs router-aware non-prefix group selection.

We first compute the router usage of each rank group from the prefix masks:

$$u_k^{(l)} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} m_{t,k}^{(l)}. \quad (15)$$

Then, we combine parameter magnitude and router usage to score each group:

$$q_k^{(l)} = s_k^{(l)} \cdot \left(u_k^{(l)}\right)^\gamma, \quad (16)$$

where γ controls the strength of router-aware selection. When $\gamma = 0$, the score reduces to magnitude-only pruning.

Given a threshold τ , the retained group set is:

$$\mathcal{K}^{(l)} = \left\{ k \mid \frac{q_k^{(l)}}{\max_j q_j^{(l)}} \geq \tau \right\}. \quad (17)$$

A minimum retained rank is applied to avoid over-compression. Let $\mathcal{K}^{(l)} = \{k_1, \dots, k_{r'_l}\}$ be the ordered retained indices. The compressed LoRA factors are:

$$A'_{[j,:]}^{(l)} = A_{[k_j,:]}^{(l)}, \quad B'_{[:,j]}^{(l)} = B_{[:,k_j]}^{(l)}, \quad j = 1, \dots, r'_l. \quad (18)$$

Since non-prefix selection changes the rank index space, the prefix router is remapped to the compressed adapter. For the original prefix-rank decision $r_t^{(l)} = \sum_{r=1}^{r_{\max}} r a_{t,r}^{(l)}$, the compressed mask is:

$$m'_{t,j} = \mathbb{I} \left[k_j \leq r_t^{(l)} \right], \quad j = 1, \dots, r'_l. \quad (19)$$

The final adapter consists of the retained LoRA factors and the remapped router, enabling task-specific rank usage without storing pruned rank groups.

4 Experiment

4.1 Datasets

We evaluate GSR-LoRA on the GLUE benchmark, a standard multi-task benchmark for natural language understanding (Wang et al., 2018). We use eight tasks: CoLA, SST-2, MRPC, STS-B, QQP, MNLI, QNLI, and RTE, covering grammatical acceptability, sentiment classification, semantic equivalence, semantic similarity, and natural language inference. The diversity in task objectives, data scales, and optimization difficulty makes GLUE suitable for evaluating multi-task PEFT and analyzing heterogeneous low-rank capacity demands.

4.2 Evaluation Metrics

We evaluate GSR-LoRA in terms of task performance and compression effectiveness. Following the standard GLUE protocol, we use MCC for CoLA, Pearson correlation for STS-B, and Accuracy for the remaining tasks. The arithmetic mean over the eight tasks is reported as Avg. To measure adapter compactness, we report Avg Rank, defined as the average number of LoRA rank groups used by each LoRA-adapted module during evaluation.

Method	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	Avg	Avg Rank
T5-base(220M)										
Fixed-rank LoRA	58.4	93.1	89.2	90.2	89.5	86.3	92.6	71.4	83.8	16.0
MoRE	61.3	<u>94.1</u>	86.8	<u>90.5</u>	90.2	85.9	<u>92.9</u>	72.7	<u>84.3</u>	16.0
MoDE	58.5	93.7	87.9	90.4	88.9	86.2	91.8	75.6	84.1	16.0
MALoRA	57.7	93.3	88.4	90.1	90.2	85.8	92.6	73.4	83.9	16.0
ALoRA	<u>59.9</u>	93.8	88.1	89.9	90.7	86.1	92.1	72.3	84.1	16.0
AutoLoRA	58.2	93.7	88.3	90.7	89.7	86.8	92.7	71.4	83.9	<u>8.3</u>
SoRA	57.9	93.9	89.9	89.7	<u>90.4</u>	86.8	91.7	73.5	84.2	<u>8.7</u>
GSR-LoRA	58.1	94.2	88.7	90.2	90.3	<u>86.3</u>	93.2	<u>75.5</u>	84.6	5.5
T5-Large(770M)										
Fixed-rank LoRA	60.9	95.8	<u>89.7</u>	91.4	90.6	88.2	94.3	81.2	86.5	16.0
MoRE	59.5	96.3	90.1	91.4	<u>91.0</u>	89.3	94.0	82.7	86.8	16.0
MoDE	62.6	95.5	88.8	<u>91.5</u>	91.1	89.3	<u>94.1</u>	81.2	86.8	16.0
MALoRA	64.3	95.6	89.0	90.8	90.1	89.2	93.9	82.7	87.0	16.0
ALoRA	62.7	<u>96.1</u>	89.3	91.2	<u>91.0</u>	<u>89.5</u>	<u>94.1</u>	83.1	<u>87.1</u>	16.0
AutoLoRA	62.7	96.0	89.2	91.2	90.2	89.4	<u>94.1</u>	82.3	86.9	7.3
SoRA	62.2	95.9	88.0	<u>91.5</u>	90.8	89.6	93.6	<u>83.4</u>	86.9	<u>6.9</u>
GSR-LoRA	<u>63.1</u>	95.5	89.5	91.7	90.9	<u>89.5</u>	94.0	84.1	87.3	4.2

Table 1: Results are averaged over three random seeds. The best results are shown in bold, and the second-best results are underlined. Statistical significance of GSR-LoRA over the strongest baseline is evaluated using a paired t-test, with $p < 0.05$.

4.3 Experimental Setup

All experiments are conducted on $2 \times$ Ascend NPU 910B3 devices. We use T5 (Raffel et al., 2020) as the backbone and apply LoRA to q, k, v, o, wi, and wo modules, with initial rank $r = 16$, scaling factor $\alpha = 32$, and dropout rate 0.1. Models are trained for 10 epochs using AdamW with a learning rate of 3×10^{-4} , weight decay of 0.01, and 500 warmup steps. For GSR-LoRA, we set the contrastive temperature $\tau_{cl} = 0.05$, the numerical stability constant $\epsilon = 10^{-8}$, and the loss coefficients $\lambda_{cl} = 0.1$, $\lambda_{gl} = 10^{-5}$, $\lambda_{ent} = 10^{-3}$, and $\lambda_{rank} = 10^{-4}$. These hyperparameters are shared across all tasks and model scales.

4.4 Baselines

We compare GSR-LoRA with three groups of baselines.

Fixed-rank PEFT. We include fixed-rank LoRA as the fundamental PEFT baseline under a uniform rank budget.

Multi-task low-rank expert methods. We compare with representative multi-task PEFT methods, including MoRE, MoDE, and MALoRA (Zhang et al., 2025; Ning et al., 2025; Wang et al., 2025b). These methods improve multi-task adaptation through low-rank expert selection, dyadic expert composition, or asymmetric low-rank expert design, making them closely related to task-aware low-rank modeling.

Adaptive-rank and sparsity-based LoRA methods.

We further compare with ALoRA, AutoLoRA, and SoRA (Liu et al., 2024; Zhang et al., 2024b; Ding et al., 2023a). ALoRA and AutoLoRA adapt LoRA ranks based on component or layer importance, while SoRA introduces sparsity-aware dynamic rank control. These baselines evaluate whether GSR-LoRA improves over existing adaptive-rank and sparsification strategies.

All baselines are implemented under the same T5 backbone and use identical optimization hyperparameters. Except for the adaptation and rank-allocation mechanisms, all training settings are kept the same to ensure a fair comparison.

4.5 Main Results

GSR-LoRA achieves the best average performance on both T5-base and T5-Large while using the lowest average rank, demonstrating a strong performance-compression trade-off.

Table 1 reports the main results on GLUE. For T5-base, GSR-LoRA obtains an average score of 84.6, outperforming the strongest baseline MoRE by 0.3 points and fixed-rank LoRA by 0.8 points. At the same time, it reduces the average rank from 16.0 to 5.5. Compared with AutoLoRA and SoRA, GSR-LoRA achieves higher average performance with fewer retained rank components, suggesting that task-dependent rank routing is more effective than purely layer or sparsity-based rank reduction in multi-task PEFT.

Variant	T5-base		T5-Large	
	GLUE Avg	Avg Rank	GLUE Avg	Avg Rank
<i>Rank-matched ablations</i>				
w/ Random Selection	83.1	5.5	86.3	4.2
w/o Task Router	83.5	5.5	86.8	4.2
w/o Group Lasso	83.7	5.5	86.6	4.2
GSR-LoRA	84.6	5.5	87.3	4.2
<i>Compression-related references</i>				
w/o Non-prefix Selection	84.6	15.2	87.2	14.6
w/o Post-training Pruning	84.7	16.0	87.6	16.0

Table 2: Ablation study of GSR-LoRA on the GLUE benchmark. Rank-matched ablations use the same average retained rank as GSR-LoRA and are ordered by T5-base GLUE Avg. The w/o Non-prefix Selection variant evaluates the compression efficiency of non-prefix group selection, while w/o Post-training Pruning serves as an uncompressed upper reference.

For T5-Large, GSR-LoRA reaches the highest average score of 87.3, surpassing ALoRA by 0.2 points, while further reducing the average rank to 4.2. This consistent improvement across model scales indicates that the proposed method does not simply benefit from over-parameterization in small adapters, but can also identify and remove redundant low-rank groups in larger backbones.

Although GSR-LoRA is not the best on every individual task, it achieves the strongest overall average. This is desirable in multi-task learning, where the key challenge is to balance heterogeneous task requirements rather than optimize a single task. The task-level results show that GSR-LoRA remains competitive across classification, similarity, and inference tasks, indicating that routing-aware rank selection preserves useful task-specific capacity under a compact adapter structure.

Overall, these results demonstrate that GSR-LoRA provides a favorable balance between task performance and rank compression. Rather than simply reducing rank uniformly, GSR-LoRA learns task-dependent rank usage and removes redundant rank groups in a structured manner. This enables the model to preserve useful low-rank capacity for heterogeneous tasks while achieving a much smaller effective rank.

Additional multi-domain results and retained-rank analyses on LLaMA-2 7B are provided in Appendix E.

4.6 Ablation Studies

We further examine how each component contributes to the performance-compression trade-off of GSR-LoRA. For rank-matched variants, the average retained rank is kept identical to GSR-LoRA, so that the comparison mainly reflects the quality

of routing and rank-group selection rather than differences in compression budget. The results are reported in Table 2.

Additional ablations of the individual loss components are provided in Appendix F.

Removing the task router consistently degrades performance on both backbones, indicating that a task-agnostic rank allocation cannot sufficiently capture the heterogeneous capacity demands across tasks. Similarly, removing Group Lasso leads to lower GLUE averages, suggesting that structured sparsity is important for making redundant rank groups identifiable before pruning. Random group selection performs the worst among the rank-matched variants, which further shows that the effectiveness of GSR-LoRA does not come from retaining a fixed number of rank groups alone, but from preserving informative low-rank components.

The compression-related variants reveal the source of parameter efficiency. Without non-prefix selection, the model can maintain comparable performance, but only by retaining substantially higher average ranks. This indicates that important rank groups are not always located in early prefix positions, and prefix-constrained retention introduces considerable redundancy. The uncompressed variant achieves slightly higher performance, but keeps the full rank of 16. After post-training pruning, GSR-LoRA preserves nearly the same performance while reducing the average retained rank by more than two-thirds.

Overall, these results demonstrate that the strong trade-off of GSR-LoRA is jointly enabled by task-dependent routing, Group Lasso sparsity, and router-aware non-prefix group selection, rather than by naive rank reduction.

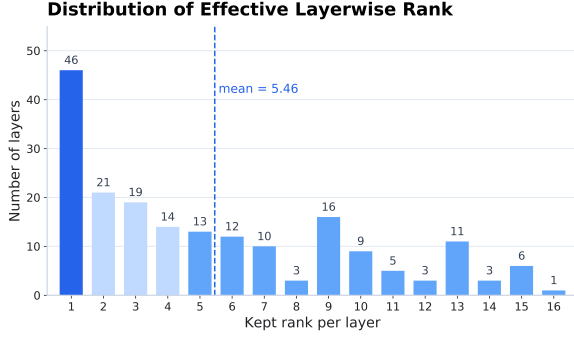


Figure 3: Distribution of effective layer-wise retained ranks after compression (T5-base).

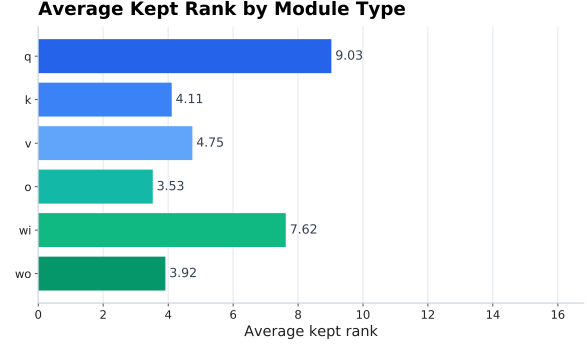


Figure 4: Average retained rank for different module types (T5-base).

4.7 Compression and Structural Analysis

To further examine how GSR-LoRA compresses low-rank adapters, we analyze the retained-rank distributions on T5-base across layers, modules, and Transformer blocks. The corresponding analysis on T5-Large is provided in Appendix A.

4.7.1 Overall Distribution of Effective Retained Ranks

The retained-rank distribution shows that GSR-LoRA performs selective rather than uniform compression. Instead of reducing all layers to a similar compact rank, GSR-LoRA learns heterogeneous retained-rank patterns across layers, with effective ranks spanning from 1 to 16. As shown in Figure 3, many layers are compressed to very small ranks, with rank 1 appearing most frequently, indicating substantial redundancy in many LoRA adapters. At the same time, several layers still preserve high ranks, including ranks above 10 and one full-rank layer, suggesting that certain substructures remain capacity-sensitive after multi-task fine-tuning.

Overall, the average retained rank is reduced from 16 to 5.5, corresponding to a 65.6% rank reduction. The coexistence of highly compressed and high-rank layers indicates that low-rank capacity demand is strongly layer-dependent. This finding supports the non-prefix group selection design of GSR-LoRA, as important rank groups are not necessarily concentrated in early prefix positions and can be more effectively preserved through selective retention than through uniform or purely prefix-constrained compression.

4.7.2 Module-wise retained-rank heterogeneity

GSR-LoRA also exhibits clear module-dependent compression behavior. As shown in Figure 4, differ-

ent projection modules retain substantially different average ranks, indicating heterogeneous low-rank capacity demands across functional components. Among the six module types, *q* retains the highest average rank of 9.03, followed by *wi* with 7.62. In contrast, *o*, *wo*, and *k* are compressed more aggressively, with average retained ranks of 3.53, 3.92, and 4.11, respectively, while *v* remains in the middle with 4.75.

This pattern suggests that query projection and input-side feed-forward transformation require more low-rank capacity for preserving task-adaptive representations. The high retained rank of *q* indicates the importance of task-dependent attention query formation, while the relatively high rank of *wi* reflects the role of feed-forward feature expansion. By contrast, the lower ranks of *o*, *wo*, and *k* suggest that these modules contain more compressible low-rank redundancy.

Overall, the module-wise imbalance supports adaptive rank allocation. Unlike fixed-rank LoRA, which assigns the same rank budget to all modules, GSR-LoRA preserves capacity where it is structurally more needed and removes redundant low-rank dimensions elsewhere, leading to more efficient parameter utilization.

4.7.3 Block-wise and Encoder-Decoder Retained-rank Patterns

The block-wise retained ranks show that GSR-LoRA learns depth- and stack-dependent compression patterns. As shown in Figure 6, the average retained rank varies substantially across Transformer blocks rather than following a flat profile. Early encoder blocks generally retain higher ranks, while several middle and late decoder blocks are compressed more aggressively, suggesting that encoder-side representation construction requires more low-

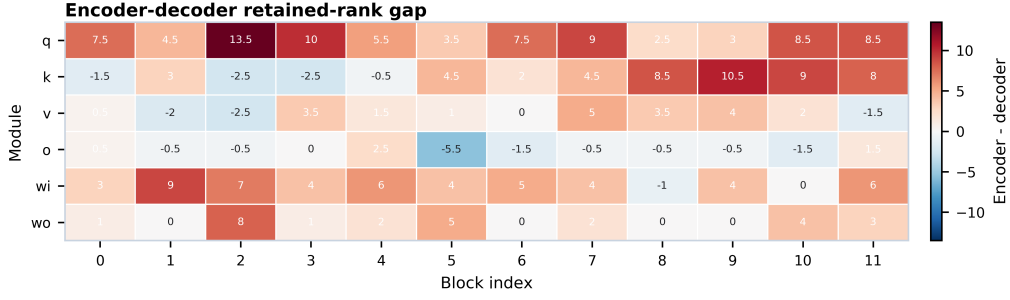


Figure 5: Encoder–decoder retained-rank gap across Transformer blocks and module types. Each value denotes the retained-rank difference between the encoder and decoder at the same block index and module type.

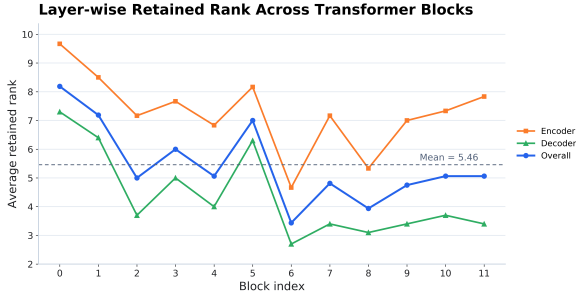


Figure 6: Layer-wise average retained-rank patterns across Transformer blocks (T5-base).

rank capacity.

Figure 5 further shows clear encoder–decoder asymmetry. The largest gaps appear in the q and wi modules, where the encoder usually retains more capacity than the decoder. This indicates that query formation and input-side feed-forward transformation are more sensitive to compression. Other modules, such as o, k, v, and wo, exhibit more mixed patterns, showing that compression cannot be explained by a single global rule.

Overall, these results confirm that low-rank capacity demand is depth-, stack-, and module-dependent. GSR-LoRA therefore preserves capacity in structurally important components while aggressively compressing redundant ones.

5 Conclusion

We proposed GSR-LoRA, a routing-aware structured rank compression method for multi-task PEFT. By integrating task-dependent rank routing, Group Lasso sparsity, and router-aware non-prefix selection, GSR-LoRA learns compact task-adaptive LoRA adapters under heterogeneous multi-task demands. Our analysis highlights the benefits of adaptive, structure-aware rank allocation across tasks, LoRA modules, network depth,

and encoder–decoder stacks.

Our main experiments on T5-base and T5-Large demonstrate that GSR-LoRA can achieve a favorable performance–compression trade-off across different model scales. Additional multi-domain results on LLaMA-2 7B in Appendix E further provide evidence of its applicability to larger models. Future work will explore broader model families, larger-scale instruction-tuning settings, and finer-grained compression guided by task and routing dynamics.

Limitations

This work has several limitations. First, our main experiments remain concentrated on GLUE with T5-base and T5-Large. Although additional results on LLaMA-2 7B are provided in Appendix E, broader evaluations across more diverse tasks, model families, and instruction-tuning settings are still needed.

Second, our system-level evaluation is preliminary. Adapter compression substantially reduces storage, whereas end-to-end latency, memory, and throughput gains are more modest and depend on hardware and implementation. Further evaluation on different platforms and optimized kernels is therefore needed.

Finally, GSR-LoRA focuses on structured LoRA rank-group compression, while interactions among task characteristics, routing behavior, and rank utilization remain underexplored. Extending the framework to broader multi-task and MoE settings is left for future work.

References

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2025. A survey on mixture of experts in large language models. *IEEE*

- Transactions on Knowledge and Data Engineering*, 37(7):3896–3915.
- Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.
- Shangyu Chen and Hangfei Zheng. 2026. [Post-hoc compression of loRA adapters via singular value decomposition](#). Submitted to ICLR 2026.
- Shijie Chen, Yu Zhang, and Qiang Yang. 2024. Multi-task learning in natural language processing: An overview. *ACM Computing Surveys*, 56(12):1–32.
- Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. 2023a. [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, Singapore. Association for Computational Linguistics.
- Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2023b. [Parameter-efficient fine-tuning of large-scale pre-trained language models](#). *Nature Machine Intelligence*, 5(3):220–235.
- Abdessalam Ed-dib, Zhanibek Datbayev, and Amine M. Aboussalah. 2025. [GeLoRA: Geometric adaptive ranks for efficient LoRA fine-tuning](#). In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 25174–25196, Suzhou, China. Association for Computational Linguistics.
- Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. [Parameter-efficient fine-tuning for large models: A comprehensive survey](#). *Transactions on Machine Learning Research*.
- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799.
- Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.
- Wenjun Ke, Jiahao Wang, Peng Wang, Jiajun Liu, Dong Nie, Guozheng Li, and Yining Li. 2024. [Unveiling lora intrinsic ranks via salience analysis](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 131575–131595. Curran Associates, Inc.
- Jiacheng Liu, Peng Tang, Wenfeng Wang, Yuhang Ren, Xiaofeng Hou, Pheng Ann Heng, Minyi Guo, and Chao Li. 2026. A survey on inference optimization techniques for mixture of experts models. *ACM Computing Surveys*, 58(10):1–37.
- Zequan Liu, Jiawen Lyn, Wei Zhu, Xing Tian, and Yvette Graham. 2024. [ALoRA: Allocating low-rank adaptation for fine-tuning large language models](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 622–641, Mexico City, Mexico. Association for Computational Linguistics.
- Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. 2018. [Modeling task relationships in multi-task learning with multi-gate mixture-of-experts](#). In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1930–1939.
- Yuren Mao, Yuhang Ge, Yijiang Fan, Wenyi Xu, Yu Mi, Zhonghao Hu, and Yunjun Gao. 2025. A survey on lora of large language models. *Frontiers of Computer Science*, 19(7):197605.
- Saeed Masoudnia and Reza Ebrahimpour. 2014. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293.
- Lin Ning, Harsh Lara, Meiqi Guo, and Abhinav Ras-togi. 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*, pages 8248–8261, Albuquerque, New Mexico. Association for Computational Linguistics.
- Dong Pan, Bingtao Li, Yongsheng Zheng, Jiren Ma, and Victor Fei. 2026. The rise of sparse mixture-of-experts: A survey from algorithmic foundations to decentralized architectures and vertical domain applications. *arXiv preprint arXiv:2602.08019*.
- 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. *Journal of machine learning research*, 21(140):1–67.
- Hao Nan Sheng, Zhi-Yong Wang, Hing Cheung So, and Mingrui Yang. 2025. [AROMA: Autonomous rank-one matrix adaptation](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3443–3459, Suzhou, China. Association for Computational Linguistics.
- Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobzyev, 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, Dubrovnik, Croatia. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. [GLUE: A multi-task benchmark and analysis platform for natural language understanding](#). In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355. Association for Computational Linguistics.

Haoyu Wang, Tianci Liu, Ruirui Li, Monica Xiao Cheng, Tuo Zhao, and Jing Gao. 2024. [RoseLoRA: Row and column-wise sparse low-rank adaptation of pre-trained language model for knowledge editing and fine-tuning](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 996–1008, Miami, Florida, USA. Association for Computational Linguistics.

Luping Wang, Sheng Chen, Linnan Jiang, Shu Pan, Runze Cai, Sen Yang, and Fei Yang. 2025a. Parameter-efficient fine-tuning in large language models: a survey of methodologies. *Artificial Intelligence Review*, 58(8):227.

Xujia Wang, Haiyan Zhao, Shuo Wang, Hanqing Wang, and Zhiyuan Liu. 2025b. [MALoRA: Mixture of asymmetric low-rank adaptation for enhanced multi-task learning](#). In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5624–5641, Albuquerque, New Mexico. Association for Computational Linguistics.

Jun Yu, Yutong Dai, Xiaokang Liu, Jin Huang, Yishan Shen, Ke Zhang, Rong Zhou, Eashan Adhikarla, Wenxuan Ye, Yixin Liu, Zhaoming Kong, Kai Zhang, Yilong Yin, Vinod Namboodiri, Brian D. Davison, Jason H. Moore, and Yong Chen. 2024. [Unleashing the power of multi-task learning: A comprehensive survey spanning traditional, deep, and pretrained foundation model eras](#). *Preprint*, arXiv:2404.18961.

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, Vienna, Austria. Association for Computational Linguistics.

Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. 2024a. [LoRAPrune: Structured pruning meets low-rank parameter-efficient fine-tuning](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3013–3026, Bangkok, Thailand. Association for Computational Linguistics.

Ruiyi Zhang, Rushi Qiang, Sai Ashish Somayajula, and Pengtao Xie. 2024b. [AutoLoRA: Automatically tuning matrix ranks in low-rank adaptation based on meta learning](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages

5048–5060, Mexico City, Mexico. Association for Computational Linguistics.

Yu Zhang and Qiang Yang. 2022. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, 34(12):5586–5609.

Zhihan Zhang, Wenhao Yu, Mengxia Yu, Zhichun Guo, and Meng Jiang. 2023. A survey of multi-task learning in natural language processing: Regarding task relatedness and training methods. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 943–956.

Mengyao Zhu and Phuc Huu Nguyen. 2025. A survey of lora algorithm variations for language models. In *International Conference on Applications of Natural Language to Information Systems*, pages 275–290. Springer.

A Additional Rank Analysis on T5-Large

To complement the analysis on T5-base, we further examine the retained-rank patterns of GSR-LoRA on T5-Large. The results show similar structural trends, but with stronger compression: the average retained rank is reduced to 4.2, indicating that larger backbones contain substantial low-rank redundancy that can be removed without sacrificing multi-task performance.

A.1 Module-wise rank allocation

GSR-LoRA learns clearly non-uniform rank budgets across module types. As shown in Figure 7, the q projection retains the highest average rank, while wi also preserves relatively larger capacity. In contrast, o, wo, v, and k are compressed more aggressively. This pattern suggests that query formation and feed-forward feature expansion remain more capacity-sensitive, whereas several output and key/value-related projections contain more compressible low-rank redundancy.

More importantly, the selected prefix span is consistently larger than the final retained rank. For example, q has an average prefix span of 10.7 but an average retained rank of 8.7, while wi decreases from 7.2 to 4.7. This gap shows that non-prefix selection further removes redundant groups inside the selected prefix region, rather than simply keeping a contiguous prefix.

A.2 Prefix budget and compression source

Figure 8 further explains where the compression comes from. The total rank budget decreases from 6144 original rank groups to 2526 groups under prefix selection, and is further reduced to 1623 groups

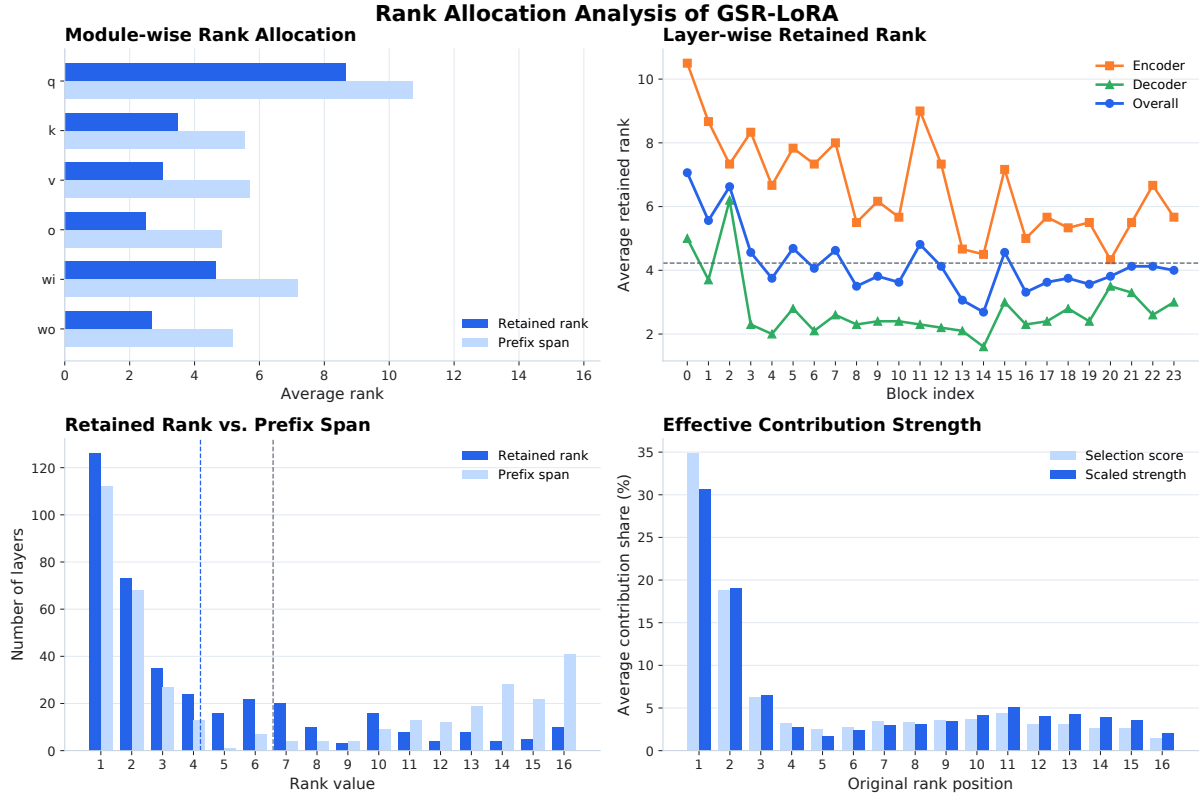


Figure 7: Rank allocation analysis of GSR-LoRA on T5-Large.

after non-prefix retention. Thus, the final compressed adapter keeps only 26.4% of the original rank groups. At the module level, the rank saving is highly uneven: *q* is compressed less aggressively, while *o*, *wo*, *v*, and *k* achieve much larger savings. This confirms that GSR-LoRA does not apply a uniform compression ratio, but allocates rank capacity according to module-specific importance.

A.3 Depth-wise and encoder-decoder patterns

The heatmaps in Figure 9 show that retained ranks are also strongly depth- and stack-dependent. Encoder-side *q* retains high ranks across many blocks, often close to the maximum rank, while decoder-side modules are generally compressed more aggressively. The *wi* module also exhibits several high-rank regions in the encoder, suggesting that encoder-side feed-forward expansion remains important for preserving shared multi-task representations. By contrast, decoder-side *o*, *wo*, *k*, and *v* often retain much smaller ranks, indicating larger redundancy in these substructures.

The prefix-depth map in Figure 10 provides a complementary view. Some encoder modules select deep prefixes even when the final retained

rank is much smaller, implying that useful rank groups may appear at later rank positions. This supports the non-prefix compression design: important groups should be selected by their learned contribution rather than by their prefix order alone.

A.4 Group selection behavior

Figure 11 shows that router usage is strongly biased toward early rank positions, but the effective contribution strength is not limited to the earliest ranks. Although the first few rank groups dominate the total contribution, several later positions still retain non-negligible strength. This explains why pure prefix retention can be inefficient: it may need to keep many intermediate groups to preserve useful high-index components. Router-aware non-prefix selection avoids this by directly retaining informative groups and removing skipped redundant groups.

Overall, the T5-Large analysis reinforces the main findings from T5-base. Low-rank capacity demand is not only task-dependent, but also module-, depth-, and stack-dependent. GSR-LoRA captures this heterogeneity by combining prefix routing with non-prefix group selection, enabling stronger compression while preserving structurally important

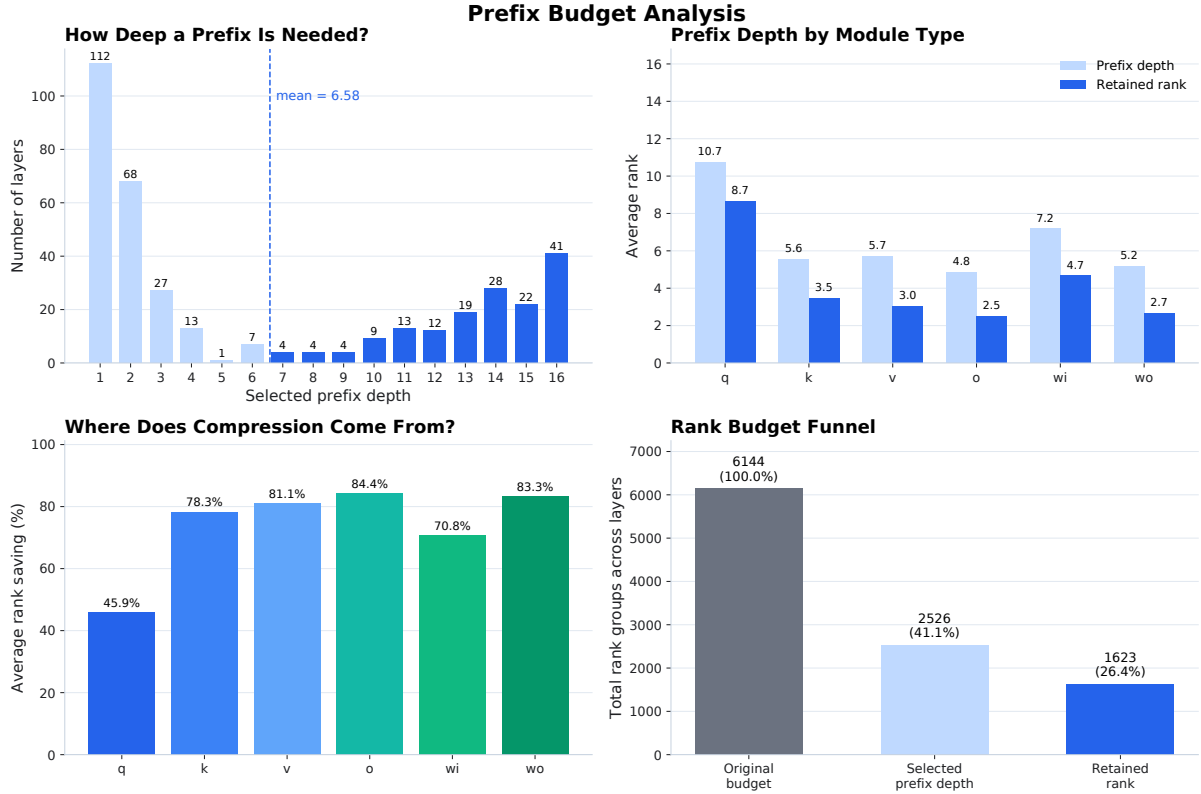


Figure 8: Prefix budget analysis of GSR-LoRA on T5-Large.

low-rank capacity.

B Artifact Licenses

We use publicly available datasets and pretrained models in accordance with their respective licenses and terms of use. These resources include the GLUE benchmark, T5 checkpoints, LLaMA-2 7B, and the additional evaluation benchmarks used in our multi-domain experiments. Our released code will include a license file specifying its terms of use.

C Potential Risks

GSR-LoRA reduces the cost of multi-task PEFT, but may also facilitate unintended model customization. Compressed adapters may inherit model or data biases, so safety and robustness should be evaluated before high-stakes deployment.

D Inference Efficiency on T5-Large

We further evaluate the practical efficiency of GSR-LoRA on T5-Large. To distinguish adapter-level compression from end-to-end system efficiency, we report the performance-compression trade-off and inference measurements separately.

Method	Rank	Param. Red.	GLUE	Storage
Uncompressed	16.00	0.0%	87.60	116.00 MB
$\tau = 0.01$	4.23	63.3%	87.27	42.62 MB
$\tau = 0.05$	4.17	63.5%	87.29	42.34 MB
$\tau = 0.10$	4.08	63.9%	87.22	41.87 MB
$\tau = 0.30$	3.45	66.8%	87.24	38.55 MB
$\tau = 0.50$	2.85	69.5%	87.15	35.38 MB
$\tau = 0.70$	2.13	73.1%	86.97	31.18 MB
$\tau = 1.00$	1.00	79.2%	86.41	24.16 MB

Table 3: Performance-compression trade-off of GSR-LoRA on T5-Large. Param. Red. denotes adapter parameter reduction, and Storage refers to adapter storage.

As shown in Table 3, GSR-LoRA substantially reduces adapter-level storage while largely preserving task performance. Across different compression thresholds, the average retained rank decreases from 16.00 to 4.23–1.00, while adapter storage is reduced from 116.00 MB to 42.62–24.16 MB, corresponding to a 63.3%–79.2% reduction in adapter parameters. The results demonstrate a clear trade-off between compression ratio and task performance as the threshold increases.

Table 4 reports the corresponding end-to-end inference measurements. Compression reduces peak memory by approximately 56–69 MB across the evaluated settings. Batch-1 latency is generally

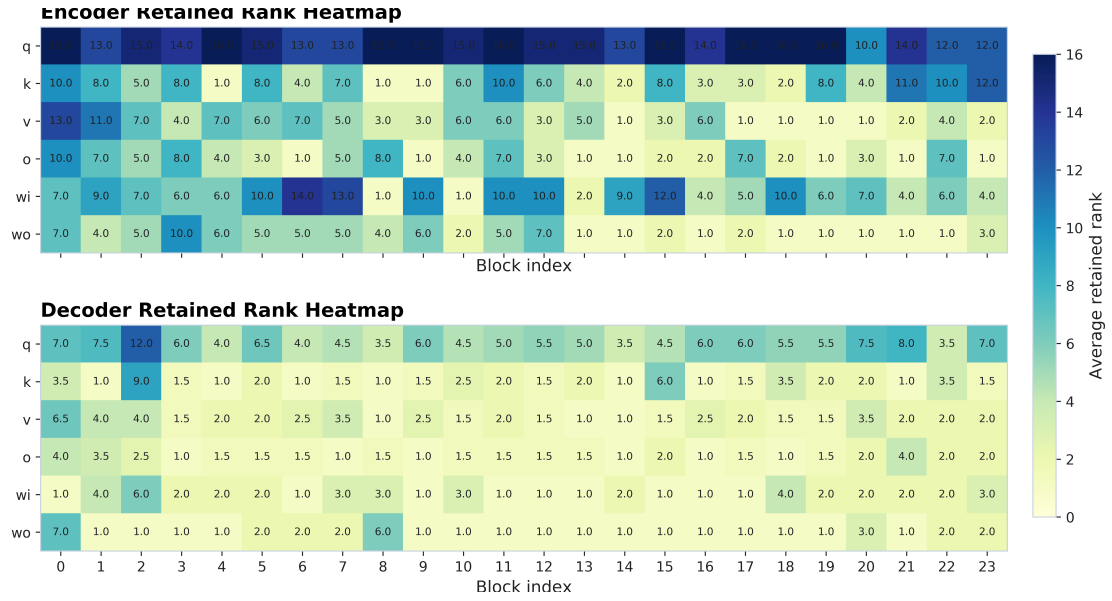


Figure 9: Encoder and decoder retained-rank heatmaps on T5-Large.

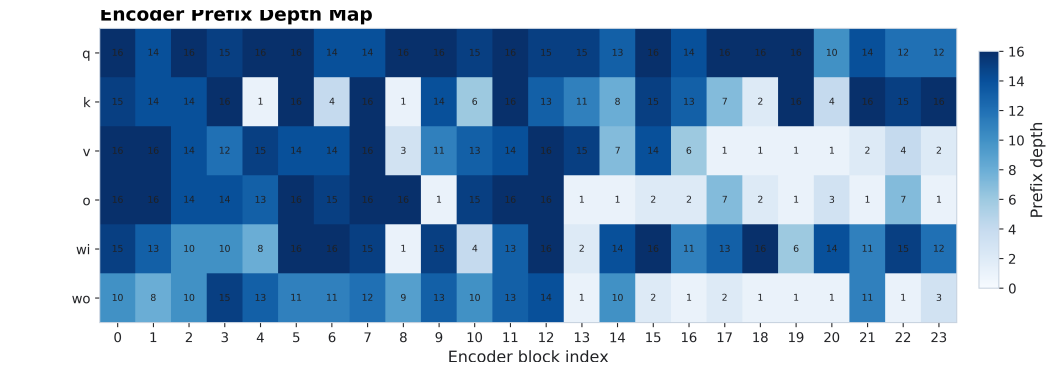


Figure 10: Encoder prefix-depth map on T5-Large.

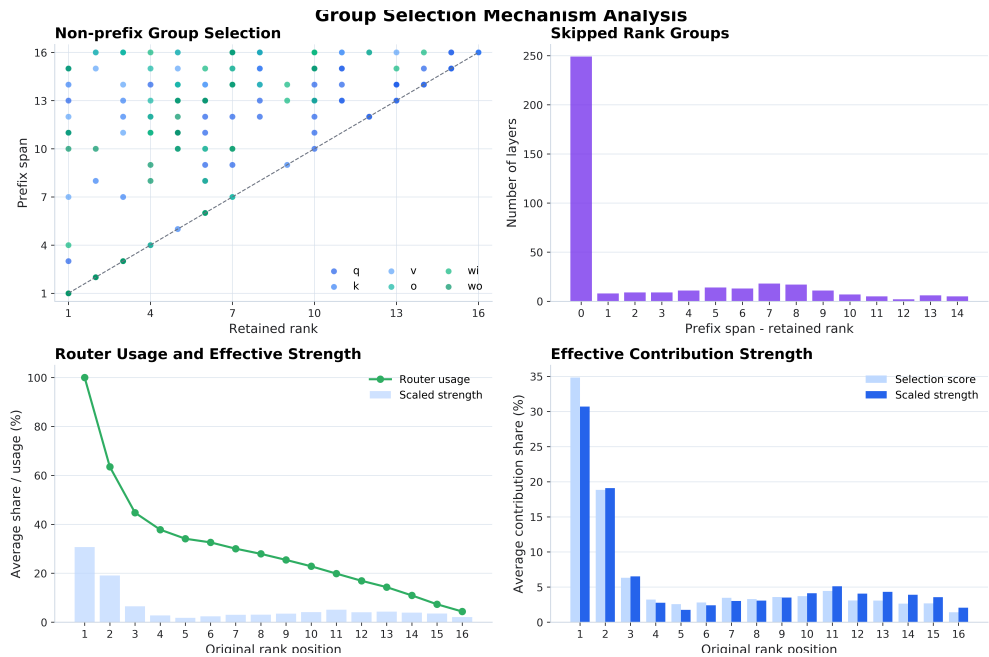


Figure 11: Group selection mechanism analysis of GSR-LoRA on T5-Large.

lower than that of the uncompressed model, but the reduction is modest and does not vary monotonically with the retained rank. At batch size 304, the measured throughput remains comparable but does

lower than that of the uncompressed model, but the reduction is modest and does not vary monotonically with the retained rank. At batch size 304, the measured throughput remains comparable but does

Method	Mem ₁	Lat. ₁	Mem ₃₀₄	Thr. ₃₀₄
Uncompressed	1898.29	3672.13	6030.33	57.13
$\tau = 0.01$	1842.01	3469.09	5974.33	55.19
$\tau = 0.05$	1841.82	3552.50	5974.13	54.81
$\tau = 0.10$	1841.50	3536.88	5973.81	56.29
$\tau = 0.30$	1839.24	3619.94	5971.55	55.57
$\tau = 0.50$	1837.00	3585.30	5969.33	55.79
$\tau = 0.70$	1834.08	3573.49	5966.41	53.56
$\tau = 1.00$	1829.10	3601.13	5961.43	54.24

Table 4: End-to-end inference efficiency on T5-Large. Mem₁ and Mem₃₀₄ denote peak memory (MB) at batch sizes 1 and 304, respectively; Lat.₁ denotes batch-1 latency (ms), and Thr.₃₀₄ denotes throughput (samples/s) at batch size 304.

Method	#Params	Latency	MedMCQA	HumanEval	GSM8K
LLaMA-2 7B [†]	—	—	34.2	14.6	13.2
LoRA [†]	2.10%	833.3	43.2	34.8	34.0
AsyLoRA [†]	2.20%	—	43.3	31.1	27.6
DoRA [†]	2.10%	1020.9	41.3	32.3	32.6
MoAsyLoRA [†]	2.30%	—	40.6	24.4	23.9
MoLoRA [†]	2.20%	1072.3	42.1	29.3	33.1
MoLA [†]	1.50%	—	41.0	24.4	33.4
MALoRA-Small [†]	1.60%	—	41.0	29.9	34.0
MALoRA [‡]	2.30%	1173.5	40.8	32.2	32.6
GSR-LoRA[‡]	0.99%	1042.7	40.9	30.9	33.7

Table 5: Additional evaluation on LLaMA-2 7B. Latency is reported in μ s.

not exceed the uncompressed baseline of 57.13 samples/s.

These results indicate that the primary efficiency benefit of GSR-LoRA lies in substantial adapter-level storage reduction and modest peak-memory savings, rather than substantial end-to-end acceleration. The limited wall-clock improvement is expected because the frozen T5-Large backbone and dense NPU kernels still dominate the overall inference cost.

E Additional Evaluation on LLaMA-2 7B

To further evaluate the scalability and generalization of GSR-LoRA, we conduct additional experiments on LLaMA-2 7B. We report multi-domain evaluation results together with a detailed analysis of the retained-rank distributions across different projection modules.

E.1 Multi-Domain Evaluation

Tables 5 and 6 report the results on seven benchmarks covering medical reasoning, code generation, mathematical reasoning, financial sentiment analysis, commonsense reasoning, and data-to-text generation.

Except for the hardware platform, all training configurations, evaluation protocols, and hyperpa-

Method	PhraseBank	ARC-C	ARC-E	E2E	AVG.
LLaMA-2 7B [†]	57.6	43.1	75.5	21.7	37.1
LoRA [†]	69.6	59.1	76.5	65.9	54.7
AsyLoRA [†]	74.2	59.8	77.8	65.5	54.2
DoRA [†]	74.0	60.0	75.6	66.5	54.6
MoAsyLoRA [†]	73.0	58.5	76.0	65.8	51.7
MoLoRA [†]	78.1	61.1	78.1	65.6	55.3
MoLA [†]	76.1	59.1	77.9	66.3	54.1
MALoRA-Small [†]	80.3	59.4	77.2	65.7	55.4
MALoRA [‡]	81.4	61.7	74.8	66.1	55.7
GSR-LoRA[‡]	82.2	62.3	75.1	66.6	56.0

Table 6: Additional task-wise results on LLaMA-2 7B.

[†] Results are taken from the original MALoRA paper. [‡] MALoRA and GSR-LoRA are evaluated using our implementation, with task scores averaged over two random seeds.

Module	Mean Rank	Median Rank	Compression	≤ 4	≥ 12
q_proj	8.00	6.5	50.0%	8	9
k_proj	9.16	9.5	42.8%	7	9
v_proj	8.97	10.0	43.9%	8	13
o_proj	8.47	8.0	47.1%	12	11
gate_proj	4.78	2.5	70.1%	20	4
up_proj	4.38	4.5	72.7%	16	0
down_proj	5.28	4.5	67.0%	16	2

Table 7: Module-wise retained-rank statistics on LLaMA-2 7B. Compression is measured relative to the initial rank $r = 16$. The last two columns report the numbers of layers with retained rank no greater than 4 and no smaller than 12, respectively.

rameter settings follow MALoRA. The latency of MALoRA and GSR-LoRA is measured under the same hardware and evaluation environment.

GSR-LoRA achieves the highest average score of 56.0 while using only 0.99% trainable parameters. Compared with MALoRA, GSR-LoRA reduces the trainable parameter ratio from 2.30% to 0.99% while achieving a slightly higher average score. It also obtains the best results on PhraseBank, ARC-C, and E2E. The latency remains comparable to the evaluated multi-task LoRA variants, indicating that the main efficiency benefit comes from adapter parameter reduction rather than substantial end-to-end acceleration.

E.2 Module-wise Retained-Rank Analysis

Starting from an initial rank of $r = 16$, GSR-LoRA learns non-uniform retained-rank distributions across different projection modules, as summarized in Table 7.

The results reveal clear module-level heterogeneity. The attention projection modules retain an average rank of 8.65, whereas the feed-forward pro-

Variant	GLUE Avg	Avg Rank
w/o $\mathcal{L}_{cl}$	84.0	5.7
w/o $\mathcal{L}_{ent}$	84.3	6.2
w/o $\mathcal{L}_{rank}$	84.7	7.3
Full	84.6	5.5

Table 8: Ablation of individual auxiliary loss components on T5-base. The Group Lasso ablation is reported separately in Table 2.

jection modules retain an average rank of only 4.81. In particular, k_proj and v_proj retain relatively high average ranks, and 13 v_proj layers retain ranks of at least 12. In contrast, the feed-forward modules show substantially greater compression potential: $gate_proj$, up_proj , and $down_proj$ achieve compression ratios of 70.1%, 72.7%, and 67.0%, respectively. Notably, no up_proj layer retains a rank of at least 12, while 20 $gate_proj$ layers retain ranks no greater than 4.

Further inspection of the layer-wise retained-rank patterns suggests that rank allocation also varies with network depth. Some shallow $gate_proj$ modules retain relatively high ranks, while middle-layer k_proj and v_proj modules tend to preserve more capacity. Some deeper o_proj modules also retain relatively high ranks. These observations indicate that low-rank capacity requirements vary jointly across network depth and module type.

These functional interpretations are based on the observed rank-allocation patterns rather than direct causal evidence of module functionality. Nevertheless, the results provide additional empirical support for heterogeneous rather than uniform rank allocation in large language models.

F Additional Loss-component Ablations

We further examine the individual contributions of the auxiliary loss components in the complete training objective. The results on T5-base are reported in Table 8.

Removing $\mathcal{L}_{cl}$ mainly degrades task performance, while removing $\mathcal{L}_{ent}$ moderately increases the retained rank. Without $\mathcal{L}_{rank}$, the GLUE score remains comparable, but the average rank increases from 5.5 to 7.3, confirming its role in encouraging compact rank allocation. Overall, the auxiliary objectives provide complementary supervision for task-aware routing and rank compression.