TASO: Task-Aligned Sparse Optimization for Parameter-Efficient Model Adaptation

Daiye Miao[1]*, **Yufang Liu**[1]*, **Jie Wang**[1], **Changzhi Sun**[1],
Yunke Zhang[2], **Demei Yan**[2], **Shaokang Dong**[2], **Qi Zhang**[3], **Yuanbin Wu**[1]
[1]East China Normal University [2]Honor Device Co., Ltd. [3]Fudan University
dymiao@stu.ecnu.edu.cn yfliu.antnlp@gmail.com ybwu@cs.ecnu.edu.cn

Abstract

LoRA has become one of the most widely used parameter-efficient fine-tuning methods due to its simplicity and effectiveness. However, numerous studies have shown that LoRA often introduces substantial parameter redundancy, which not only increases the number of trainable parameters but also hinders the effectiveness of fine-tuning. Since identifying redundant parameters in LoRA is inherently difficult, how to eliminate them efficiently and accurately remains a challenging problem. In this paper, we propose TASO, a redundancy reduction method that leverages importance information from the pretrained model's weights to mitigate LoRA redundancy. Specifically, we estimate parameter importance on downstream tasks and identify task-specific core regions based on the distribution of importance scores. The location information of these core regions is then used to determine the sparse structure of LoRA modules, enabling redundancy removal before fine-tuning. Our approach significantly reduces the number of trainable parameters required for task adaptation, while providing a novel task-aligned perspective for LoRA redundancy reduction. Experimental results demonstrate that, with a parameter budget comparable to LoRA with rank $r = 1$, TASO consistently outperforms standard LoRA across multiple tasks, achieving strong fine-tuning performance while effectively eliminating redundant parameters.

1 Introduction

With the rapid development of large-scale pretrained language models (Zhao et al., 2023; Zhou et al., 2024a; Minaee et al., 2024; Zhang et al., 2024a), efficiently adapting them to downstream tasks has become a central challenge(Ding et al., 2023; Han et al., 2024; Wang et al., 2024c). Although full fine-tuning remains highly effective, it requires storing a separate set of model weights for each task, leading to substantial storage and computational overhead. To address this, a variety of Parameter-Efficient Fine-Tuning (PEFT) methods(Zhou et al., 2024b; Prottasha et al., 2025) have been proposed, which aim to adapt models by updating only a small subset of parameters. Among them, Low-Rank Adaptation (LoRA) has gained significant popularity due to its simplicity and efficiency (Hu et al., 2022a). LoRA freezes the original model weights and injects trainable low-rank matrices, thereby avoiding additional inference overhead and becoming one of the most widely adopted PEFT techniques in practice.

However, recent studies have revealed substantial redundancy not only in the delta parameters generated during fine-tuning, but also within the LoRA modules themselves (Yu et al., 2024; Panda et al., 2024; Kopiczko et al., 2024a; Zhang et al., 2023), motivating further efforts to reduce the number of trainable parameters without sacrificing performance. Yet, most existing approaches still require training a complete LoRA module before gradually pruning unimportant weights during training, which fails to fundamentally reduce training costs.

Inspired by recent findings on the spatial concentration of task-relevant core regions in pretrained models (Zhang et al., 2024b), we propose a novel method to sparsify LoRA prior to fine-tuning. Specifically, we estimate the importance of pretrained weights by leveraging downstream task gradients to design a sparsified structure for the low-rank adaptation matrices. Building on the observed clustering patterns of high-importance weights, we pre-define zero positions for low-importance regions, thereby constructing sparse LoRA modules. During training, these modules dynamically aggregate updates only to the non-zero positions, ensuring efficient adaptation while preserving the predefined sparse structure.

* Equal contribution.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 22735–22747
November 4-9, 2025

Unlike methods that perform pruning during training, our approach eliminates redundancy in advance, based on the inherent sparsity of delta weights and the localized importance of core parameters. By identifying the subset of parameters to be trained before fine-tuning begins, we ensure that updates are concentrated on the most critical regions of the original model, thereby significantly reducing parameter overhead while maintaining model effectiveness.

Experimental results show that our method consistently outperforms standard high-rank LoRA on multiple benchmarks, even when using a parameter budget comparable to rank-1 LoRA. These findings confirm the effectiveness of proactively identifying and leveraging key parameter regions, enabling substantial reductions in fine-tuning cost without compromising generalization. The summary of our contributions is as follows:

- We introduce an importance-guided LoRA sparsification method that significantly reduces the number of trainable parameters without increasing inference cost, offering a new and practical solution for efficient adaptation of large language models.
- We introduce a sparsity-aware learning rate scaling strategy that ensures the retained parameters in pruned LoRA modules can be fully optimized, thereby preserving downstream adaptation performance.
- TASO consistently outperforms LoRA with significantly fewer trainable parameters across various NLP tasks and model architectures.

2 Related Work

2.1 Parameter-Efficient Fine-Tuning Methods

Parameter-efficient fine-tuning (PEFT) aims to adapt large pre-trained models by optimizing only a small subset of parameters while keeping the majority of the model frozen. Representative methods include Adapter tuning (Houlsby et al., 2019), Prefix and Prompt Tuning (Li and Liang, 2021a,b),BitFit (Ben Zaken et al., 2022),IA3 (Liu et al., 2022) which introduce lightweight modules or continuous prompts into the model to enable efficient task adaptation. Another direction focuses on identifying and updating specific subsets of the model's parameters (Hu et al., 2022b; Ben Zaken et al., 2022; Guo et al., 2021).

LoRA remains one of the most widely used fine-tuning methods due to its simplicity and effectiveness. Building on its success, many studies propose extensions to improve its flexibility or generalization. Some works explore combining LoRA modules across tasks or styles (Luo et al., 2024; Huang et al., 2023; Shah et al., 2024; Hu et al., 2022b). Others focus on modifying its internal structure (Tian et al., 2024; Kopiczko et al., 2024b; Balazy et al., 2024). These works demonstrate that while LoRA is highly efficient, further improvements can be achieved through better module design and composition strategies.

2.2 Redundancy in LoRA Parameters

Although LoRA significantly reduces the number of trainable parameters, recent studies have shown that it can still introduce redundancy. For example, Yu et al. (2024) show that LoRA's delta parameters still contain substantial redundancy, and demonstrate that techniques originally developed for other modules remain effective when applied to LoRA.

To address this, methods like AdaLoRA (Zhang et al., 2023) and DyLoRA (Valipour et al., 2022) dynamically allocate rank across layers by pruning unimportant directions. Other approaches impose structural sparsity on LoRA matrices. SoRA (Wang et al., 2024b) introduces trainable gates to zero out redundant components during training, while RoseLoRA (Wang et al., 2024a) enforces row- and column-wise sparsity for more targeted adaptation.

These works demonstrate that reducing LoRA redundancy—either by pruning or structured sparsity—can maintain or even improve performance while further lowering the adaptation cost. Our method continues along this line, proposing a more effective strategy for eliminating unnecessary parameters in LoRA modules.

3 Background

Low-rank Adaptation Before introducing our approach, we first briefly recap low-rank adaptation (LoRA), which operates by freezing the pre-trained weight matrices $\mathbf{W}_0 \in \mathbb{R}^{p\times q}$ while introducing trainable low-rank decomposition matrices to model weight updates. Specifically, the weight update $\Delta\mathbf{W}$ is factorized into two low-rank matrices $\mathbf{A} \in \mathbb{R}^{r\times q}$ and $\mathbf{B} \in \mathbb{R}^{p\times r}$ such that $\Delta\mathbf{W} = \mathbf{BA} \in \mathbb{R}^{p\times q}$, where $r \ll \min(p, q)$ represents the intrinsic rank dimension controlling the adaptation capacity.

The modified forward pass of a layer can be

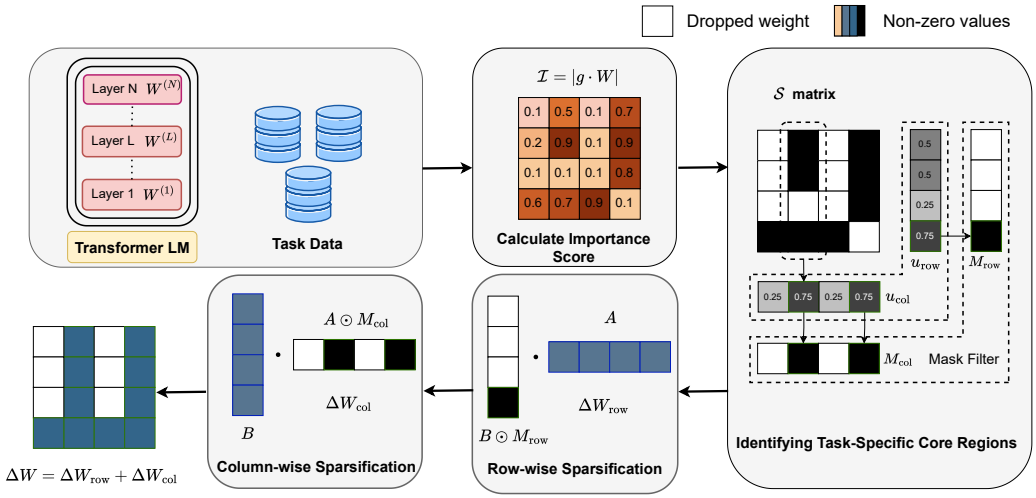

Figure 1: Overview of TASO. We compute task-specific importance from SFT loss to determine core regions, perform structured pruning of LoRA, and scale learning rate based on sparsity level.

expressed as:

$$\mathbf{y} \leftarrow \mathbf{W}_0\mathbf{x} + \mathbf{BAx}, \tag{1}$$

where $\mathbf{x}$ is the input and $\mathbf{y}$ is the output of the layer. The key computational advantage stems from the low-rank projection $\mathbf{z} \leftarrow \mathbf{BAx}$, which enables efficient adaptation while maintaining the original model structure. The hyperparameter r governs both the number of trainable parameters and the expressiveness of the adaptation. Existing approaches (Wang et al., 2024b; Zhang et al., 2023) often train full-rank LoRA parameters and subsequently prune or reduce them, which may incur unnecessary training overhead. In contrast, we aim to explore a more efficient alternative that reduces adaptation cost from the outset.

Parameter Importance Estimation Quantifying the relative importance of model parameters constitutes a fundamental challenge in neural network analysis, with broad implications for understanding and optimizing model behavior (Dai et al., 2022; Yao et al., 2024). By identifying parameters that are critical for maintaining task performance, this analysis enables more principled approaches to model adaptation and refinement (Wang et al., 2024a). A theoretically grounded framework for parameter importance estimation derives from sensitivity analysis, where importance is defined as the element-wise product of a parameter's value and its corresponding loss gradient. Formally, for each parameter θ_i, the importance metric is given by:

$$\mathcal{I}_i(\theta) = \left| \theta_i \cdot \frac{\partial \mathcal{L}}{\partial \theta_i} \right|.$$

This formulation simultaneously captures a parameter's magnitude and its influence on the loss landscape. High-scoring parameters are essential to model performance, while low-scoring ones can be modified with minimal impact. The method provides an efficient way to evaluate parameter salience across learning scenarios.

4 Method

The proposed methodology is illustrated in Figure 1. Our approach consists of three key stages: First, we identify task-specific core regions within the model by estimating the importance of individual weights with respect to the target task. Subsequently, through analysis of the distribution patterns among highly important weights, we observe distinct row- and column-wise clustering characteristics. This empirical observation informs the design of an optimized sparsified LoRA structure. Finally, we perform fine-tuning based on this task-aligned sparse architecture to enhance parameter efficiency while maintaining model performance.

4.1 Identifying Task-Specific Core Regions

Recent studies (Zhang et al., 2024b; Huang et al., 2025) have demonstrated that parameter contributions in pre-trained models are highly task-dependent. Notably, Zhang et al. (2024b) discover that specific model capabilities are governed by sparse subsets of weights (termed "linguistic regions"), while the majority of parameters remain inactive. This observation motivates our hypothesis that LoRA updates could achieve greater effi-

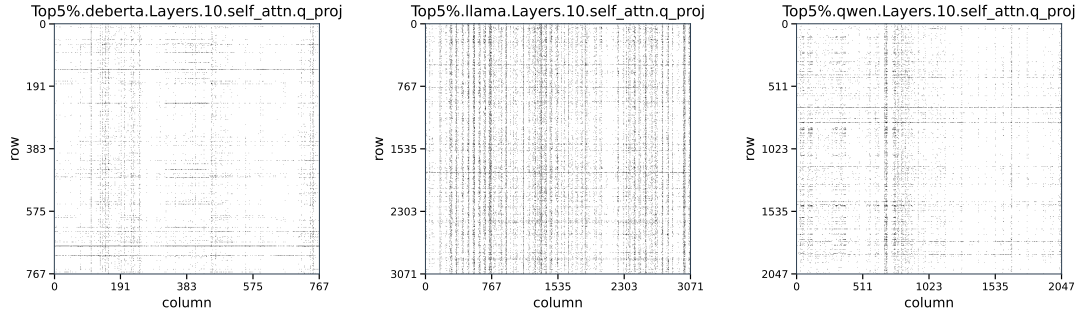

Figure 2: Visualization of task-specific important weights identified by sensitivity analysis. Heatmaps show the top-5% important parameters in the self-attention query matrices of DeBERTa-v3 on RTE, LLaMA3.2 3B on GSM8K, and Qwen2.5 3B on GSM8K. The important weights concentrate in specific rows and columns of the matrices.

ciency by concentrating on task-relevant parameter subspaces. Given a training task, we compute the importance score $\mathcal{I}_i$ for each parameter θ_i. We then define a binary mask $\mathcal{S}$, where:

$$\mathcal{S}_i = \begin{cases} 1, & \text{if } \mathcal{I}_i \text{ ranks in top-}k\% \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Here, $\mathcal{S}_i$ indicates whether parameter θ_i is among the top-$k\%$ most important parameters for the task.

Figure 2 highlights the top-5% most important parameters in the self-attention query matrices of three models: DeBERTa-v3 (on RTE), LLaMA3.2 3B (on GSM8K), and Qwen2.5 3B (on GSM8K). The analysis reveals two key findings: (1) Important parameters exhibit non-uniform, clustered distributions with clear row-wise and column-wise patterns; (2) These structural regularities persist across diverse architectures and tasks, suggesting an inherent organization of parameter importance in pre-trained models. The observed patterns motivate our pruning strategy for LoRA parameters, where we selectively preserve these structurally important dimensions while eliminating redundant ones before training, thereby significantly reducing both computational overhead and memory requirements during adaptation.

To unify the importance of rows and columns, we compute average importance scores for each row i and column j from the binary mask $\mathcal{S}$:

$$u_i^{\text{row}} = \frac{1}{m}\sum_{j=1}^{m} \mathcal{S}_{i,j}, \quad u_j^{\text{col}} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{S}_{i,j}$$

which represent the density of important parameters in each dimension. We then perform a global ranking based on importance scores u, and select the top-$p\%$ entries as the task-specific core region. Among them, we define $\mathcal{R}_{\text{core}}$ as the set of row indices and $\mathcal{C}_{\text{core}}$ as the set of column indices. These core regions will guide structured pruning in LoRA modules.

4.2 Structured LoRA Pruning Guided by Core Regions

Building upon the identified core regions $\mathcal{R}_{\text{core}}$ and $\mathcal{C}_{\text{core}}$, we implement a structured pruning strategy for LoRA parameters that optimizes training efficiency. To minimize training overhead, we set the rank $r = 1$. The approach combines rank-constrained decomposition with mask-based sparsification, which includes two sequential stages: row sparsification followed by column sparsification.

Row-wise Sparsification First, we apply row-wise sparsification to the LoRA matrix A. This is implemented by masking all rows not contained in $\mathcal{R}_{\text{core}}$, effectively preserving only the parameters from the identified important rows. The training objective during this phase can be expressed as:

$$\Delta\mathbf{W}_{\text{row}} = \mathbf{B} \cdot (\mathbf{A} \odot \mathbf{M}_{\text{row}}),$$

where $\mathbf{M}_{\text{row}}$ is the row mask matrix defined by $\mathbf{M}_{\text{row}}(i,:) = \mathbb{I}(i \in \mathcal{R}_{\text{core}})$ for each row i, and $\odot$ denotes element-wise multiplication. Row-wise sparsification restricts updates to rows in $\mathcal{R}_{\text{core}}$ while preserving global column propagation. Following row-wise training, we update the original parameters by accumulating the learned delta weights:

$$\mathbf{W}_{\text{updated}} = \mathbf{W}_0 + \Delta\mathbf{W}_{\text{row}}.$$

Column-wise Sparsification Subsequently, we perform column-wise sparsification on matrix B using the same approach. The column mask M_{col}

preserves only columns from $\mathcal{C}_{\text{core}}$, with the training operation formulated as:

$$\Delta \mathbf{W}_{\text{col}} = (\mathbf{B} \odot \mathbf{M}_{\text{col}}) \cdot \mathbf{A}.$$

Similarly, column-wise sparsification restricts updates to columns in $\mathcal{C}_{\text{core}}$ while preserving global row projection.

The complete parameter update then becomes:

$$\mathbf{W}_{\text{final}} = \mathbf{W}_{\text{updated}} + \Delta \mathbf{W}_{\text{col}}.$$

This formulation ensures that $\Delta \mathbf{W}$ introduces non-zero updates exclusively within the task-specific core region, achieving targeted sparsification while maintaining the expressiveness of the rank-1 LoRA adaptation.

4.3 Scaling Learning Rates under Structured Sparsity

Structured pruning significantly reduces the number of trainable parameters in LoRA modules, thereby impairing the model's adaptation capability during fine-tuning. To compensate for this sparsity, the DARE method (Yu et al., 2024) proposes parameter rescaling:

$$\Delta \mathbf{W}_{\text{rescaled}} = \frac{1}{1-\rho} \cdot \Delta \mathbf{W}_{\text{pruned}} \tag{3}$$

where $\Delta \mathbf{W}_{\text{pruned}}$ denotes the parameter matrix after structured pruning (with some elements zeroed out), and $\rho \in [0, 1)$ represents the pruning ratio (fraction of zeroed elements [1]). The remaining non-zero parameters are amplified by a factor of $1/(1-\rho)$ to maintain the original magnitude scale.

Unlike conventional post-hoc rescaling of $\Delta \mathbf{W}$, our approach mitigates the effects of sparsity by adjusting the learning rate. This avoids potential numerical instability caused by parameter scaling. Specifically, we scale the LoRA factors as:

$$\Delta \mathbf{W} = \left(\sqrt{\tfrac{1}{1-\rho}} \cdot \mathbf{A}\right)\left(\sqrt{\tfrac{1}{1-\rho}} \cdot \mathbf{B}\right) = \tfrac{1}{1-\rho} \cdot \mathbf{AB}, \tag{4}$$

which corresponds to scaling the learning rate as:

$$\text{lr}_{\text{scaled}} = \sqrt{\tfrac{1}{1-\rho}} \cdot \text{lr}, \tag{5}$$

here the scaling factor compensates for gradient attenuation due to sparsity.

This approach preserves the desired effect of sparsity compensation, while providing finer control over LoRA's impact on the model's overall behavior. It also avoids abrupt changes to $\Delta \mathbf{W}$, leading to more stable fine-tuning.

[1] The value of ρ is determined by calculating the proportion of non-zero elements in the ΔW.

5 Results

Datasets and Models We conduct comprehensive evaluations across two major categories of language models. For **decoder-only** language models, we evaluate multiple benchmarks: mathematical reasoning (GSM8K (Cobbe et al., 2021)), question answering (BoolQ (Clark et al., 2019)), word sense disambiguation (WiC (Pilehvar and Camacho-Collados, 2018)), and scientific question answering (ARC (Clark et al., 2018), including ARC-e and ARC-c). We use Qwen2.5-3B (Yang et al., 2024) and LLaMA3.2-3B (Dubey et al., 2024) as backbone models, splitting each task's validation set into development and test sets (2:8 ratio). For **encoder-based** models, we evaluate on GLUE (Wang et al., 2018), covering tasks like grammatical acceptability (CoLA (Warstadt et al., 2019)), sentiment analysis (SST-2 (Socher et al., 2013)), paraphrase identification (MRPC (Dolan and Brockett, 2005), QQP (Quora, 2017)), sentence similarity (STS-B (Cer et al., 2017)), and natural language inference (MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2005)). DeBERTaV3-base (He et al., 2023) serves as the backbone model.

Baselines We compare with the following baselines:

- **Fine-tune**: Updates all model parameters during task-specific training, serving as the conventional upper-bound baseline.
- **BitFit**: A highly parameter-efficient approach that optimizes only the bias terms while freezing other parameters.
- **Adapter**: Inserts lightweight bottleneck modules between transformer layers, preserving original model weights through complete freezing.
- **LoRA**: Performs low-rank decomposition of weight matrices, injecting trainable rank-defficient matrices for efficient adaptation.
- **IA3**: Implements learnable scaling vectors on attention activations and feed-forward network outputs for localized adaptation.
- **DoRA**: Enhances LoRA by decoupling weight updates into directional and magnitude components, improving both expressiveness and training stability.

All baselines are implemented with standard configurations to ensure fair comparison.

Method	Model	#Param.	GSM8K	BoolQ	WiC	ARC-e	ARC-c	Avg.
Fine-tune	Qwen2.5 3B	3151.91 M	67.20	87.76	72.35	93.74	82.72	80.75
IA3		1.35 M	64.64	88.26	72.54	93.16	81.76	80.07
Adapter		67.10 M	66.44	88.22	71.56	93.84	81.98	80.41
LoRA (r=8)		14.97 M	68.83	87.50	73.13	94.58	82.08	81.22
LoRA (r=32)		59.87 M	68.34	87.95	72.94	94.63	82.83	81.34
DoRA (r=32)		65.98 M	68.53	88.37	**74.11**	94.73	83.26	81.80
AdaLoRA (r=32)		61.32 M	68.72	**89.14**	73.31	94.63	81.66	81.09
VERA (r=1024)		1.42 M	68.90	85.43	71.56	94.05	81.02	80.19
LoRA-XS (r=128)		4.13 M	69.09	82.07	73.52	92.89	84.00	80.31
TASO		2.06 M	**70.04**	88.64	73.33	**94.84**	**85.50**	**82.47**
Fine-tune	LLaMA3.2 3B	3266.58 M	**45.11**	80.73	74.21	85.11	70.46	71.12
IA3		0.91 M	32.89	84.51	70.98	84.79	67.59	68.15
Adapter		50.32 M	36.39	85.28	72.54	86.53	66.63	69.47
LoRA (r=8)		12.16 M	35.45	80.73	71.76	85.21	64.60	67.55
LoRA (r=32)		48.63 M	39.60	82.14	71.96	86.53	67.48	69.54
DoRA (r=32)		53.73 M	40.00	84.63	70.58	86.32	68.44	70.00
AdaLoRA (r=32)		49.51 M	41.06	87.80	73.92	85.58	68.76	71.02
VERA (r=1024)		1.09 M	38.00	85.85	**74.50**	84.27	66.73	69.87
LoRA-XS (r=128)		3.21 M	33.74	79.89	72.94	84.06	67.37	67.20
TASO		1.67 M	39.81	**88.53**	74.50	**86.74**	**68.86**	**71.82**

Table 1: Comparison of full fine-tuning and PEFT methods on decoder-only models Qwen2.5-3B and LLaMA3.2-3B. #Params. denotes the number of trainable parameters used during task-specific tuning. We evaluate methods across diverse task types, including reasoning, classification, and understanding. Avg. indicates the average accuracy across all evaluated tasks.

Evaluation Metrics For decoder-based models, we report zero-shot accuracy on GSM8K, BoolQ, WiC, and ARC.For benchmarks that only provide validation sets , we split the validation set into 20% development and 80% testing. The development split is used for hyperparameter tuning, while the testing split is reserved exclusively for final evaluation For encoder-based models, we follow GLUE standard metrics: Matthews correlation coefficient for CoLA, accuracy for SST-2, MRPC, QNLI, and RTE, matched accuracy for MNLI, accuracy and F1 for QQP, and Pearson and Spearman correlations for STS-B.

Implementation Details Encoder experiments are conducted on NVIDIA RTX 3090 GPUs, while decoder experiments are run on NVIDIA A100 GPUs. We provide the learning rate settings for all methods, including LoRA (decoder and encoder variants), in Appendix C.1. For task-specific importance estimation, we set $k = 5\%$, selecting the top 5% of parameters based on their importance scores. To identify the task-specific core region, we compute row- and column-wise retention ratios, jointly rank them, and set $p = 5\%$ to extract the top 10% entries.

5.1 Results

The experimental results of parameter-efficient fine-tuning methods across different model architectures are systematically presented in Tables 1 and 2. Table 1 provides a comprehensive comparison of full fine-tuning versus PEFT approaches on two state-of-the-art decoder-only language models: Qwen2.5-3B and LLaMA3.2-3B. Table 2 extends this analysis to the encoder-only architecture using DeBERTa-v3-base. From the results, we could find:

- First, the proposed TASO method demonstrates remarkable parameter efficiency in decoder architectures. TASO achieves the highest average accuracy with only 2.06M and 1.67M trainable parameters on Qwen2.5-3B and LLaMA3.2-3B, respectively—over 30 $\times$ fewer than LoRA (r=32) and DoRA (r=32). On Qwen2.5-3B, TASO outperforms all baselines on GSM8K and BoolQ, while maintaining strong results on WiC and ARC. On LLaMA3.2-3B, TASO achieves top performance on BoolQ and WiC, highlighting its parameter efficiency in decoder-only settings.
- Second, In encoder architectures, TASO's efficiency advantage becomes even more pronounced. Requiring only 0.18M parameters

Method	#Param.	CoLA	MRPC	QNLI	QQP	RTE	SST-2	STS-B	MNLI	Avg.
Fine-Tune	184 M	69.21	89.22	93.78	92.05/89.31	82.49	95.64	91.59	89.98/89.95	87.82
Adapter	1.41 M	69.00	89.90	93.79	91.45/88.88	82.44	95.16	91.45	90.11/90.11	87.85
Bitfit	0.1 M	68.70	87.16	91.90	87.86/84.20	76.12	94.38	89.71	87.45/87.45	85.18
LoRA (r=8)	1.33 M	69.73	89.71	93.76	91.95/89.26	85.32	95.57	91.86	90.47/90.46	88.38
LoRA (r=16)	2.65 M	69.87	89.91	93.46	92.22/89.63	87.05	95.53	91.79	90.55/90.31	88.62
SoRA	1.33 M	71.48	**91.98**	**94.28**	**92.39/89.87**	87.77	95.64	92.22	90.35/90.38	89.36
TASO	0.18 M	69.86	90.03	93.72	91.30/88.57	88.01	95.60	92.30	90.36/90.44	88.73
TASO†	0.75 M	**71.69**	91.83	94.17	91.60/88.94	**88.49**	**95.87**	**92.40**	**90.61/90.89**	**89.43**

Table 2: Comparison of full fine-tuning and PEFT methods on the encoder-only model DeBERTa-v3-base. Total Params indicates the number of trainable parameters during fine-tuning. We evaluate performance across multiple GLUE tasks of various types. Avg. denotes the average score over all tasks. TASO† performs 4 rounds of iterative fine-tuning, where the incrementally trained weights from each round are merged into the model.

(15× fewer than LoRA with r=16), TASO establishes new state-of-the-art results on the GLUE benchmark. The performance improvements are particularly notable in three critical NLP tasks: textual entailment (RTE: 88.01 vs. LoRA's 87.54), paraphrase detection (MRPC: 90.03 vs. 89.41), and semantic textual similarity (STS-B: 92.30 vs. 91.82). This consistent outperformance across diverse tasks underscores TASO's enhanced generalization capabilities.

In summary, these findings confirm that our task-specific sparsification strategy significantly reduces redundancy in LoRA modules while preserving or enhancing performance across a broad range of tasks and model architectures.

5.2 Connections with Lottery Ticket in LoRA

The *lottery ticket hypothesis* (Frankle and Carbin, 2018) posits that dense neural networks contain sparse, trainable subnetworks capable of matching the performance of the original network when trained in isolation. This phenomenon is typically identified through the Iterative Magnitude Pruning (IMP) algorithm, which progressively removes low-magnitude weights and retrains the remaining subnetwork. Our application of IMP to LoRA modules (r=8) in DeBERTa-v3-base (RTE task) reveals that 99% sparsity can be achieved while maintaining performance (left figure in Figure 3)- a significant improvement over existing sparse LoRA studies (Wang et al., 2024b,a) that typically achieve only 30% sparsity. This remarkable sparsification potential directly motivates TASO's core design: if such high sparsity is achievable in standard LoRA, then an optimally pruned r=1 LoRA could potentially match the performance while being dramatically more parameter-efficient.

Notably, IMP's surviving weights exhibit column-wise clustering patterns (middle figure in Figure 3) that align with TASO's task-specific core regions (Figure 2), suggesting both methods converge to similar sparse configurations through distinct mechanisms.

However, IMP method requires computationally expensive iterative pruning and retraining cycles. In contrast, TASO achieves comparable sparsity levels in two iterative training steps through gradient-informed core region identification. The right panel of Figure 3 compares the computational time between TASO and IMP algorithms. Notably, when achieving comparable sparsity levels to TASO, IMP demonstrates approximately an order of magnitude higher computational cost (requiring 171 minutes versus 12 minutes for the RTE task). Furthermore, this performance gap exhibits a pronounced expansion with increasing task complexity. Therefore, it is noteworthy that TASO maintains 90% sparsity with remarkably low computational overhead.

5.3 Discussions

To further inspect the proposed method, we investigate the following research questions.

Does scaled learning rate help sparse training? To evaluate the necessity of sparsity-aware learning rate scaling, we compare the full TASO framework against a variant with fixed baseline learning rates (*w/o LR scale*) across three model architectures. As shown in Table 3:

- The impact varies substantially by model architecture, with the LLaMA3.2 3B showing the most dramatic sensitivity: removing learning rate scaling causes performance drops of 10.81, 8.03, and 17.45 points on GSM8K, BoolQ, and WiC re-

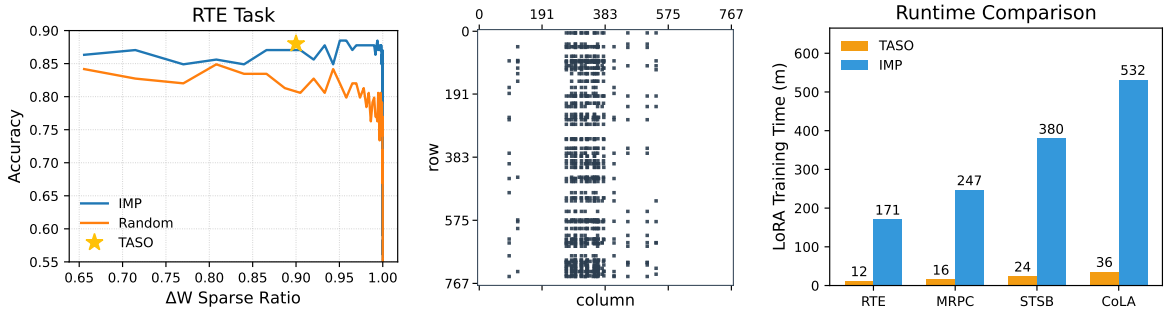

Figure 3: **Left:** Accuracy vs. sparsity curve with TASO highlighted. **Middle:** Visualization of key mask sparsity. **Right:** LoRA training runtime for TASO vs. IMP on four tasks.

Dataset	TASO	w/o LR scale	w/o core region
(a) LLaMA3.2 3B model			
GSM8K	**39.81**	29.00$_{-10.81}$	37.63$_{-2.18}$
BoolQ	**88.53**	80.50$_{-8.03}$	82.14$_{-6.39}$
WiC	**74.50**	57.05$_{-17.45}$	61.37$_{-13.13}$
(b) Qwen2.5 3B model			
GSM8K	**70.04**	68.53$_{-1.51}$	67.58$_{-2.46}$
BoolQ	**88.64**	86.39$_{-2.25}$	87.19$_{-1.45}$
WiC	**73.33**	72.94$_{-0.39}$	72.35$_{-0.98}$
(c) DeBERTa-v3-base model			
RTE	**88.01**	84.17$_{-3.84}$	81.77$_{-6.24}$
CoLA	**69.86**	68.33$_{-1.53}$	67.88$_{-1.98}$
MRPC	**90.03**	89.71$_{-0.32}$	89.56$_{-0.47}$
STS-B	**92.30**	91.93$_{-0.37}$	91.53$_{-0.77}$

Table 3: Ablation results across different models and tasks. Both learning rate scaling and core region-based pruning contribute significantly to performance.

spectively.

- The technique proves universally beneficial, with measurable gains across all three model families (LLaMA, Qwen, and DeBERTa) and all seven evaluation tasks, though the magnitude of improvement is task-dependent.

These findings suggest that adaptive learning rate adjustment is crucial for effective sparse training, as it helps balance the information flow through the remaining active connections.

Does the identified core region contain critical information? We analyze the importance of structured pruning through task-specific core regions by comparing TASO against a variant with randomly selected rows and columns results (*w/o core region*):

- Pruning non-core parameters retains >90% of original performance in most cases (e.g., LLaMA3.2 on GSM8K: 39.81%→37.63%), indicating core regions encode compact but sufficient task representations.
- The effectiveness holds for both generative (Qwen2.5) and discriminative (DeBERTa) models, suggesting these regions capture architecture-agnostic functional units.

The consistent preservation of functionality confirms that core regions constitute minimal sufficient parameter sets for task execution.

Does Sparse LoRA Improve Cross-Task Compositionality? We investigate whether sparsified LoRA modules enhance compositional generalization compared to dense counterparts. The experiment combines: (1) A base dense LoRA module (fixed, row tasks in Table 4), and (2) Either a standard dense LoRA or our pruned LoRA (column tasks), evaluated through task-pair accuracy averaging. Key observations reveal:

- Pruned modules achieve higher composition accuracy in most task pairs (e.g., CoLA+RTE: 50.40% vs 41.01%), demonstrating better cross-task compatibility.
- Largest improvements occur when combining semantically distant tasks (MRPC+CoLA: +18.6%), suggesting sparsity helps isolate task-specific features.

These results confirm that sparsification not only reduces trainable parameters but also enhances LoRA's compositional properties, providing a principled solution for multi-task adaptation.

Beyond these empirical findings, TASO also offers a broader design perspective for parameter-efficient fine-tuning. Whereas most existing approaches adopt a uniform update strategy across all modules, TASO explicitly leverages gradient-informed importance distributions to identify task-relevant core regions prior to training, and restricts updates to these structures using low-rank LoRA.

Task	Dense			Pruned (TASO)		
	CoLA	RTE	MRPC	CoLA	RTE	MRPC
CoLA	–	41.01	48.49	–	**50.40**	**52.87**
RTE	41.01	–	65.02	**60.23**	–	63.82
MRPC	48.49	67.04	–	**67.04**	59.62	–

Table 4: Cross-task composition accuracy (%). Each row denotes a base task whose dense LoRA module is fixed. Each column corresponds to a second LoRA module trained on a different task, which is either a standard dense module (left) or a pruned module obtained by our method (right).

This task-sensitive and structure-aware strategy not only improves parameter efficiency but also avoids redundant updates to less important components.

Furthermore, our analysis suggests a new interpretation of the role of rank in LoRA: even with extremely low ranks (e.g., r = 1), strong performance can be achieved provided that the mask aligns with the core region. This view goes beyond the intrinsic rank hypothesis, offering new insights for theoretical modeling in PEFT. Importantly, TASO is not mutually exclusive with other approaches; rather, it represents a sparsity-guided and structure-aware design principle that can be combined with complementary techniques. We believe this perspective provides a scalable and composable foundation for future research on extreme parameter-efficient tuning.

6 Conclusion

By leveraging the positional characteristics of task-specific core regions and the structural properties of LoRA, TASO enables highly efficient fine-tuning with a parameter budget close to LoRA rank $r = 1$. It not only reduces the number of trainable parameters, but also consistently outperforms standard LoRA across a wide range of tasks and model architectures. In future work, we aim to explore the generalization ability of TASO beyond language models, particularly in the domains of vision and audio.

7 Limitations

While TASO demonstrates promising results, our study has several limitations. This work evaluates TASO solely on standard NLP tasks, and its applicability to other modalities—such as vision or multimodal models—remains an open question. Further research is needed to assess whether TASO can generalize as effectively as other PEFT techniques across diverse settings. Moreover, although TASO offers new insights into parameter-efficient fine-tuning, each component of the method still leaves room for further optimization.

Acknowledgments

The authors wish to thank all reviewers for their helpful comments and suggestions. The corresponding author is Yuanbin Wu. This research was (partially) supported by National Key R&D Program of China (2024YFC3308103).

References

Klaudia Balazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. 2024. Lora-xs: Low-rank adaptation with extremely small number of parameters. *CoRR*, abs/2405.17604.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity—multilingual and cross-lingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

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. *arXiv preprint arXiv:1905.10044*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the*

60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 8493–8502. 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, and 1 others. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Demi Guo, Alexander Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. 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 4884–4896, Online. 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. *arXiv preprint arXiv:2403.14608*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022a. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Shengding Hu, Zhen Zhang, Ning Ding, Yadao Wang, Yasheng Wang, Zhiyuan Liu, and Maosong Sun. 2022b. Sparse structure search for parameter-efficient tuning. *arXiv preprint arXiv:2206.07382*.

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

Weiyu Huang, Yuezhou Hu, Guohao Jian, Jun Zhu, and Jianfei Chen. 2025. Pruning large language models with semi-structural adaptive sparse training. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 24167–24175. AAAI Press.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. 2024a. Vera: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. 2024b. Vera: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Xiang Lisa Li and Percy Liang. 2021a. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xiang Lisa Li and Percy Liang. 2021b. 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, Online. Association for Computational Linguistics.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

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

Shervin Minaee, Tomás Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *CoRR*, abs/2402.06196.

Ashwinee Panda, Berivan Isik, Xiangyu Qi, Sanmi Koyejo, Tsachy Weissman, and Prateek Mittal. 2024. Lottery ticket adaptation: Mitigating destructive interference in llms. *arXiv preprint arXiv:2406.16797*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2018. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*.

Nusrat Jahan Prottasha, Upama Roy Chowdhury, Shetu Mohanto, Tasfia Nuzhat, Abdullah As Sami, Md Shamol Ali, Md Shohanur Islam Sobuj, Hafijur Raman, Md Kowsher, and Ozlem Ozmen Garibay. 2025. Peft a2z: Parameter-efficient fine-tuning survey for large language and vision models. *arXiv preprint arXiv:2504.14117*.

Quora. 2017. Quora question pairs. https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Viraj Shah, Nataniel Ruiz, Forrester Cole, Erika Lu, Svetlana Lazebnik, Yuanzhen Li, and Varun Jampani. 2024. Ziplora: Any subject in any style by effectively merging loras. In *European Conference on Computer Vision*, pages 422–438. Springer.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

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

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2022. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*.

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. *arXiv preprint arXiv:1804.07461*.

Haoyu Wang, Tianci Liu, Ruirui Li, Monica Xiao Cheng, Tuo Zhao, and Jing Gao. 2024a. 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, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 996–1008. Association for Computational Linguistics.

Haoyu Wang, Tianci Liu, Tuo Zhao, and Jing Gao. 2024b. Roselora: Row and column-wise sparse low-rank adaptation of pre-trained language model for knowledge editing and fine-tuning. *CoRR*, abs/2406.10777.

Luping Wang, Sheng Chen, Linnan Jiang, Shu Pan, Runze Cai, Sen Yang, and Fei Yang. 2024c. Parameter-efficient fine-tuning in large models: A survey of methodologies. *arXiv preprint arXiv:2410.19878*.

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

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1112–1122.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Kai Yao, Penglei Gao, Lichun Li, Yuan Zhao, Xiaofeng Wang, Wei Wang, and Jianke Zhu. 2024. Layer-wise importance matters: Less memory for better performance in parameter-efficient fine-tuning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 1977–1992. Association for Computational Linguistics.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*.

Yu Zhang, Xiusi Chen, Bowen Jin, Sheng Wang, Shuiwang Ji, Wei Wang, and Jiawei Han. 2024a. A comprehensive survey of scientific large language models and their applications in scientific discovery. *arXiv preprint arXiv:2406.10833*.

Zhihao Zhang, Jun Zhao, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024b. Unveiling linguistic regions in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6228–6247, Bangkok, Thailand. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Method	Model	GSM8K	BoolQ	WiC
sensitivity	Qwen2.5	**70.04**	**88.64**	73.33
gradient		67.67$_{-2.37}$	88.53$_{-0.11}$	73.33$_{-0.00}$
sensitivity	LLaMA3.2	**39.81**	**88.53**	**74.50**
gradient		39.43$_{-0.38}$	87.00$_{-1.53}$	69.01$_{-5.49}$

Table 5: Comparison of importance metrics for TASO. sensitivity uses $|\theta_j \cdot g_j|$, while gradient uses $|g_j|$ only.

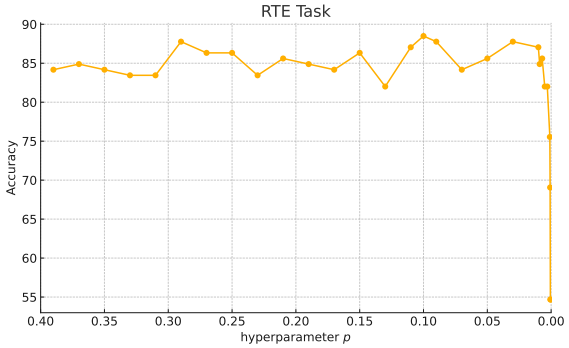

Figure 4: Accuracy on the RTE task as a function of the pruning hyperparameter p, which indicates the fraction of non-zero values after pruned. The x-axis is shown from 0.40 (left) to 0 (right) to highlight performance under increasing sparsity.

Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, and 1 others. 2024a. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *International Journal of Machine Learning and Cybernetics*, pages 1–65.

Xiongtao Zhou, Jie He, Yuhua Ke, Guangyao Zhu, Victor Gutierrez Basulto, and Jeff Pan. 2024b. An empirical study on parameter-efficient fine-tuning for MultiModal large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10057–10084, Bangkok, Thailand. Association for Computational Linguistics.

A Comparison with different importance scores

We evaluate how different parameter importance metrics affect model performance by comparing TASO's sensitivity scoring $|\theta_j \cdot g_j|$ against gradient-only scoring $|g_j|$. Experiments are conducted on Qwen2.5 3B and LLaMA3.2 3B across three reasoning tasks (GSM8K, BoolQ, WiC). Results (in Table 5) show that the joint metric performs matches or outperforms than gradient-only scoring, with particularly strong gains on LLaMA3.2's WiC task (+5.49%). These results demonstrate that incorporating parameter magnitude alongside gradients provides more reliable importance estimates.

B Impact of sparsity ratio on Performance

We conduct a systematic analysis of the sparsity hyperparameter p (fraction of retained parameters) using the RTE task, with results visualized in Figure 4. Our experiments reveal three distinct operational regimes:

- Remarkably maintains performance despite retaining only 2-5% parameters, demonstrating TASO's ability to preserve critical information under extreme sparsity conditions. This suggests our importance scoring effectively identifies and protects task-essential parameters.
- Across a wide sparsity range (from 5% to 40% retained parameters), the method sustains over 90% of peak accuracy, confirming that most parameters in standard LoRA modules are indeed redundant for task adaptation.
- The sweet spot ($p = 0.1$) achieves peak performance (approximately 88%) while reducing parameters by 90%, striking the ideal balance between compactness and capability.

These findings demonstrate that TASO's structured pruning approach effectively identifies and preserves task-essential parameters, enabling radical parameter reduction without compromising model effectiveness. The consistent performance across sparsity levels suggests our method's suitability for both memory-constrained and high-performance scenarios.

C Additional Experimental Details

C.1 Learning Rate Settings

For Qwen2.5-3B, the learning rate configurations of different fine-tuning methods are summarized in Table 6.

Method	Learning Rate
TASO / LoRA	5×10^{-5}
Full Finetune	5×10^{-6}
DoRA	5×10^{-5}
Adapter	5×10^{-5}
BitFit	3×10^{-3}
IA^3	3×10^{-3}
LoRA-XS	1×10^{-4}
VeRA	1×10^{-2}

Table 6: Learning rate configurations for different fine-tuning methods on Qwen2.5-3B.

Note. In practice, we initialize the learning rate

with the same setting as LoRA, but during training the code dynamically adjusts it according to the method described in Section 4.3.