LeLoRA: Learnable Low-Rank Adaptation of Large Language Models

Xiaoling Zhou[1*], **Mingjie Zhang**[1*], **Zhemg Lee**[2*], **Wei Ye**[1†] ,**Shikun Zhang**[1†]
[1]Peking University, [2]Tianjin University
{xiaolingzhou, mjzhang}@stu.pku.edu.cn, zhemglee@tju.edu.cn,
{wye,zhangsk}@pku.edu.cn

Abstract

Fine-tuning large language models (LLMs) is an effective approach to enhancing their performance on specialized downstream tasks. Among the various techniques, low-rank adaptation has garnered significant attention due to its ability to maintain the full performance of fine-tuning while enhancing computational efficiency. However, existing approaches often rely on manually specified and fixed hyperparameters to identify the trainable components within weight matrices, resulting in suboptimal performance and low parameter efficiency. This paper presents a novel **Le**arnable **Lo**w-**R**ank **A**daptation (LeLoRA) framework that utilizes dynamically learned fine-tuning strategies to facilitate the effective adaptation of LLMs. Our framework integrates an LLM with a policy network that automatically and adaptively generates matrix-specific adaptation strategies to identify the trainable components of each weight matrix, taking into account their unique characteristics, such as singular values and matrix norms. A reinforcement learning-based optimization algorithm is then employed to iteratively update the LLM and the policy network, ensuring that the generated strategies adapt in real time to the evolving states of the LLM. Extensive experiments have been conducted across various natural language processing tasks. The results across ten different LLMs, ranging from 125M to 70B parameters, provide compelling evidence that LeLoRA consistently outperforms existing baselines in adapting LLMs.

1 Introduction

Recent advances in natural language processing (NLP) have positioned fine-tuning pre-trained large language models (LLMs) for specific tasks as one of the most promising techniques (Xiang et al., 2025; Yang et al., 2026; Xu et al., 2025; Ma et al., 2026; Li et al., 2026a; Zheng et al., 2025b; Chu et al., 2025). However, fully fine-tuning all parameters in LLMs demands significant computational resources and time, rendering it impractical in resource-constrained environments (Pan et al., 2024; Jiang et al., 2025a; Zhang et al., 2026; Zhou et al., 2024; Jiang et al., 2025b). To address this challenge, various parameter-efficient fine-tuning (PEFT) methods have been proposed to reduce the number of parameters that require tuning (Weyssow et al., 2025; Albert et al., 2025; Tian et al., 2025; Zhang et al., 2025a). Among these techniques, low-rank adaptation (LoRA) (Hu et al., 2022) has gained significant attention as an effective PEFT method, as it preserves the performance of full fine-tuning while mitigating the increase in inference latency (Zhou et al., 2025a; Liu et al., 2024).

A variety of enhancements to the LoRA framework have been proposed, leading to notable advancements in the adaptation performance of LLMs (Mu et al., 2025; Hu et al., 2025; Zhou et al., 2025d,c; Ren et al., 2024). For example, LoRA+ (Hayou et al., 2024) enhances training efficiency by applying distinct learning rates to the two low-rank matrices. Meanwhile, DoRA (Liu et al., 2024) decomposes the weights into two independent components, direction and magnitude, thereby enabling more refined fine-tuning. To mitigate slow convergence resulting from improper initialization and further reduce parameter budget, several studies have employed singular value decomposition (SVD) to parameterize incremental updates or weight matrices (Meng et al., 2024; Wang et al., 2024; Zhang et al., 2023b). However, identifying the specific components of the matrices that need to be tuned remains a substantial challenge. For example, PiSSA (Meng et al., 2024) and Laser (Sharma et al., 2024) focus on updating the principal components, while MiLoRA (Wang et al., 2024) constrains the optimization to the minor singular parts. Actually, adaptation strategies

*Equal contribution.
*Corresponding authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 35255–35273
July 2-7, 2026 ©2026 Association for Computational Linguistics

for selecting trainable components should be customized for each matrix, given the distinct roles of the various parameters (Zhang et al., 2023b; Zhou et al., 2025a; Xie et al., 2026). Although some studies have allocated varying budgets across weight matrices, they rely on hand-crafted heuristics, such as gradient- or Hessian-based rules, and control only the rank, without explicitly specifying which positions to update. Consequently, a more flexible strategy is required, one that operates at the matrix level and jointly determines both the allocation budget and the precise locations of the parameters to be updated.

To address the aforementioned challenges, this study introduces a novel **Le**arnable **Lo**w-**R**ank **A**daptation framework, termed LeLoRA, which utilizes automatically learned, matrix-specific fine-tuning strategies for the effective adaptation of LLMs. Our framework integrates an LLM with a policy network, where the LLM is fine-tuned according to the strategies generated by the policy network. Unlike previous approaches that rely on manually defined indicators or fixed hyperparameters (e.g., rank) to allocate parameter budgets across different weight matrices, our framework leverages a policy network that dynamically and adaptively generates matrix-specific adaptation strategies. These strategies determine the trainable components of each weight matrix, tailored to its unique characteristics, such as singular values and matrix norm. The LLM and policy network are updated alternately using a reinforcement learning-based optimization algorithm. Since the updates to the policy network are driven by the performance of the LLM, the resulting strategies are continuously refined to more effectively align with the evolving learning dynamics of the LLM.

Extensive experiments have been conducted across various tasks, including natural language understanding (NLU), commonsense reasoning, math reasoning, code generation, and instruction following. The results demonstrate that LeLoRA consistently achieves state-of-the-art performance across diverse tasks by dynamically selecting the most beneficial components for task-specific updating. Moreover, the learned adaptation strategies are carefully analyzed, offering valuable insights to guide future research in PEFT.

Overall, our main contributions are as follows:

- We introduce LeLoRA, a novel learnable low-rank adaptation framework, in which a policy network dynamically generates adaptation strategies tailored to the unique characteristics of each weight matrix.
- We present a reinforcement learning-based algorithm that alternately updates the LLM and the policy network within our framework. The training of the policy network is guided by the performance of the LLM, ensuring that the generated strategies adaptively adjust to the evolving training dynamics in real time.
- We conduct extensive comparative and analytical experiments that provide conclusive evidence for the effectiveness of LeLoRA in adapting LLMs. Additionally, several insightful findings are obtained, offering valuable directions for future PEFT research.

2 Related Work

Parameter-Efficient Fine-Tuning With the widespread application of LLMs, PEFT has become increasingly popular (Chen et al., 2025; Ye et al., 2026; Zheng et al., 2025a,c; Jiang and Ferraro, 2026a,b). PEFT techniques reduce the overhead of fully fine-tuning LLMs by updating only a small subset of model parameters, thereby enabling efficient task adaptation (Zhou et al., 2025b,a; Wei et al., 2024; Tian et al., 2025; Li et al., 2026b; Xiao et al., 2026; Zhao et al., 2025). Existing methods fall into three main categories. Adapter-based approaches insert trainable modules into frozen backbones (He et al., 2022; Wu et al., 2024; Zhang et al., 2024, 2025b), either sequentially (Karimi Mahabadi et al., 2021) or in parallel (He et al., 2022). Prompt-based methods prepend learnable prompts or small parameter sets to the inputs (Masoud et al., 2024; Soylu et al., 2024), but are sensitive to initialization and prone to overfitting (Liu et al., 2022b; Meng et al., 2024). Both categories, however, often incur additional inference latency compared to baseline models (Chen et al., 2023; Zhang et al., 2025b; Wang et al., 2024; Xie, 2026).

Low-Rank Adaptation The third category of PEFT, exemplified by LoRA (Hu et al., 2022), represents low-rank adaptations of weight updates, significantly reducing tunable parameters without introducing additional inference overhead (Hayou et al., 2024; Mu et al., 2025; Hu et al., 2025; Zhou et al., 2025a). Several improvements have been proposed: DoRA (Liu et al., 2024) decomposes weights into direction and magnitude for more granular updates; AdaLoRA (Zhang et al., 2023b) and LoRA-drop (Zhou et al., 2025a) allocate parameter budgets using predefined importance scores; PiSSA (Meng et al., 2024) focuses on essential components while keeping residuals fixed; and MiLoRA (Wang et al., 2024) emphasizes minor singular components to preserve pre-trained knowledge. Despite their effectiveness, identifying which components to fine-tune remains challenging (Meng et al., 2024; Mu et al., 2025; Zhou et al., 2022; Wang et al., 2024). Although some studies allocate varying budgets across weight matrices, they rely on hand-crafted heuristics and control only the rank, without explicitly specifying which parameter positions to update. In contrast, our approach learns a data-driven policy that jointly determines both the budget and locations of parameter updates, providing greater flexibility and adaptivity through reinforcement learning guided by feedback from LLM performance.

Singular Value Decomposition Given a weight matrix $\boldsymbol{W} \in \mathbb{R}^{m\times n}$, its SVD (Eckart and Young, 1936) is expressed as $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_m] \in \mathbb{R}^{m\times m}$, with each column representing a left singular vector; and $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n] \in \mathbb{R}^{n\times n}$, with each column representing a right singular vector. The matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times n}$ is a diagonal matrix containing the singular values σ_i of $\boldsymbol{W}$, arranged in descending order (Bhatnagar et al., 2013; Liu et al., 2022a). For simplicity, we assume without loss of generality that $m \le n$. Thus, the SVD of $\boldsymbol{W}$ can be formulated as $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \sum_{i=1}^{m} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top$.

3 Methodology

This study introduces a novel LeLoRA framework building upon the concept of "learnable adaptation strategy." We first introduce our new formulation of adaptive parameter allocation, where the trainable components of each weight matrix are determined based on its unique characteristics. Subsequently, we present the entire LeLoRA framework, which integrates an LLM with a policy network. Finally, we introduce a reinforcement learning-based optimization algorithm that alternately updates the LLM and the policy network.

3.1 Adaptive Parameter Allocation

Given that incremental decomposition methods are susceptible to noise and that reliance on zero initialization can impede early convergence (Meng et al., 2024; Wang et al., 2024), we directly parameterize the pre-trained weight matrices $\boldsymbol{W}$ in LLMs using SVD, such that $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ represent the left and right singular vectors, respectively, and $\boldsymbol{\Sigma}$ is a diagonal matrix containing the singular values of $\boldsymbol{W}$. Similar to PiSSA (Meng et al., 2024), SVD is performed only once prior to the training process to initialize the indices of components to be updated.

Unlike previous approaches that either apply a uniform fine-tuning strategy across all matrices or rely on manually defined functions to allocate budgets, our method selectively prioritizes the fine-tuning of the most impactful components based on learned strategies, enabling more targeted and parameter-efficient optimization. Specifically, the volume and position of the trainable components within each matrix are determined by a policy network, which takes into account the matrix's unique characteristics, such as singular values and matrix norms, as input. The details of this process are provided in the following subsection. In this context, each weight matrix is decomposed into two segments: the trainable component $\boldsymbol{W}_{\mathrm{T}}$, and the frozen component $\boldsymbol{W}_{\mathrm{F}}$, as follows:

$$\begin{aligned}\boldsymbol{W} &= \boldsymbol{W}_{\mathrm{T}} + \boldsymbol{W}_{\mathrm{F}} \\ &= \boldsymbol{U}_{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{T}}\boldsymbol{V}_{\mathrm{T}}^\top + \boldsymbol{U}_{\mathrm{F}}\boldsymbol{\Sigma}_{\mathrm{F}}\boldsymbol{V}_{\mathrm{F}}^\top \\ &= \sum\nolimits_{i=1}^{r} \sigma_i^{\mathrm{T}} \boldsymbol{u}_i^{\mathrm{T}} (\boldsymbol{v}_i^{\mathrm{T}})^\top + \sum\nolimits_{j=1}^{m-r} \sigma_j^{\mathrm{F}} \boldsymbol{u}_j^{\mathrm{F}} (\boldsymbol{v}_j^{\mathrm{F}})^\top,\end{aligned} \tag{1}$$

where $\boldsymbol{U}_{\mathrm{T}} = [\boldsymbol{u}_1^{\mathrm{T}}, \ldots, \boldsymbol{u}_r^{\mathrm{T}}]$ and $\boldsymbol{U}_{\mathrm{F}} = [\boldsymbol{u}_1^{\mathrm{F}}, \ldots, \boldsymbol{u}_{m-r}^{\mathrm{F}}]$. This formulation is similarly applied to the trainable and frozen components of $\boldsymbol{V}$ and $\boldsymbol{\Sigma}$. To determine the volume and position of the trainable components within each matrix, the policy network generates two matrix-specific outputs: the effective dimension r, which defines the size of the trainable components, and the location index s, which specifies their position. For instance, if $r = 8$ and $s = 2048$, then the trainable components of $\boldsymbol{U}$ are represented as $\boldsymbol{U}_{\mathrm{T}} = [\boldsymbol{u}_{2048}, \cdots, \boldsymbol{u}_{2055}]$, while the frozen components are $\boldsymbol{U}_{\mathrm{F}} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_{2047}, \boldsymbol{u}_{2056}, \cdots, \boldsymbol{u}_m]$. The matrices $\boldsymbol{V}$ and $\boldsymbol{\Sigma}$ can be similarly restructured in a comparable fashion.

3.2 Learnable Low-Rank Adaptation

Fig. 1 illustrates the proposed LeLoRA framework, which integrates an LLM with a policy network. The policy network, implemented as a multi-layer

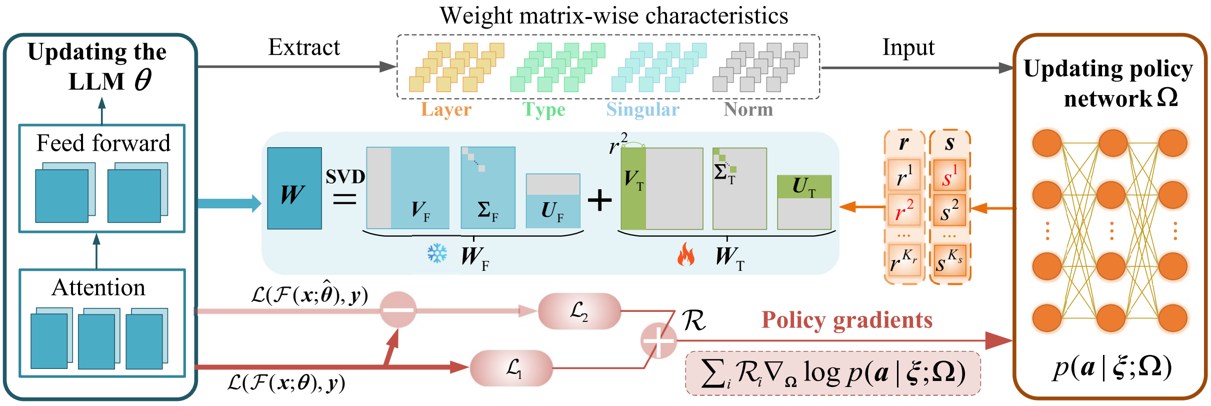

Figure 1: Diagram of the LeLoRA framework. The adaptation strategies for each weight matrix are dynamically generated by a policy network based on its unique characteristics: singular values, norm, layer index, and type. A reinforcement learning algorithm is then utilized to iteratively optimize both the LLM and the policy network.

perceptron (MLP)[1], processes a series of characteristics of the weight matrices to generate matrix-wise adaptation strategies. The extracted characteristics for each matrix include its layer index, type (e.g., query, key, value, etc.), Frobenius norm (Böttcher and Wenzel, 2008), and singular values, which collectively provide insights into the matrix's role and position within the model. Moreover, the policy network outputs two elements: the effective dimension r and the location index s. The structure of the policy network, implemented as an MLP, is detailed in Appendix B[2].

Let $\boldsymbol{a} = \{r, s\} \in \mathcal{A}$ denote an adaptation strategy, where $\mathcal{A}$ represents the set of all possible adaptation strategies. Each parameter has a set of possible choices, which are represented using one-hot encoding. For example, $r \in \{r^1, r^2, \ldots, r^{K_r}\}$ has K_r possible values, and $s \in \{s^1, s^2, \ldots, s^{K_s}\}$ has K_s possible choices. In our framework, the possible values for the effective dimension are drawn from a series of integers, such as {0, 2, 4, 8}, while the location index consists of three distinct options: 0, $m/2$, or m, corresponding to the beginning, middle, and end of the matrix, respectively[3]. A specific adaptation strategy is defined by a particular combination of the selected values of these two parameters. Accordingly, the policy network models the conditional distribution of the adaptation strategy $\boldsymbol{a}$ given the characteristics of the weight matrix $\boldsymbol{\xi}$, formulated as $p(\boldsymbol{a}|\boldsymbol{\xi};\boldsymbol{\Omega})$, where $\boldsymbol{\Omega}$ denotes the parameters of the policy network.

To guide the learning of the policy network, we introduce two metrics to assess both the LLM performance and the effectiveness of the adaptation strategy. The first metric is the evaluated loss $\mathcal{L}$ of the LLM ($\mathcal{F}$), denoted as

$$\mathcal{L}_1 = \mathcal{L}(\mathcal{F}(\boldsymbol{x};\boldsymbol{\theta}), \boldsymbol{y}), \tag{2}$$

where $\boldsymbol{\theta}$ denotes the parameters of the LLM. The second metric quantifies the performance variation of the LLM due to adaptation and is defined as

$$\mathcal{L}_2 = \mathcal{L}(\mathcal{F}(\boldsymbol{x};\boldsymbol{\theta}), \boldsymbol{y}) - \mathcal{L}(\mathcal{F}(\boldsymbol{x};\hat{\boldsymbol{\theta}}), \boldsymbol{y}), \tag{3}$$

where $\hat{\boldsymbol{\theta}}$ denotes the parameters of the LLM prior to updating in a given iteration. The incorporation of the relative performance changes helps stabilize and smooth the gradient signal, thereby reducing variance. These two metrics are computed using a small validation set.

Let $\mathcal{R} = \mathcal{L}_1 + \alpha\mathcal{L}_2$, where α is the modulation factor that controls the relative strengths of the two loss terms. The objective function of the entire framework is as follows:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \left[\min_{\boldsymbol{\Omega}} \mathbb{E}_{\boldsymbol{a}\sim p(\boldsymbol{a}|\boldsymbol{\xi};\boldsymbol{\Omega})} \left[\mathcal{L}(\boldsymbol{\theta}) + \mathcal{R}(\boldsymbol{\Omega})\right] \right], \tag{4}$$

where $\{(\boldsymbol{x}, \boldsymbol{y})\}$ denotes the training dataset, and $\mathcal{L}(\boldsymbol{\theta})$ represents the training loss of the LLM.

3.3 Optimization via Reinforcement Learning

This section elaborates on the optimization process of the LeLoRA framework. Given the parameters of the policy network, $\boldsymbol{\Omega}$, the optimization problem for the LLM can be expressed as follows:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \mathbb{E}_{\boldsymbol{a}\sim p(\boldsymbol{a}|\boldsymbol{\xi};\boldsymbol{\Omega})} \left[\mathcal{L}(\boldsymbol{\theta})\right]. \tag{5}$$

[1]Further discussions on alternative architectures for the policy network are provided in Appendix J.

[2]The introduced policy network is lightweight, constituting less than 0.1% of the parameters in the LLM.

[3]When 0, $m/2$, or m is selected, it indicates that the trainable component is located within the primary, medium, or minor components of the matrix, respectively.

Method	Rank	MRPC	RTE	CoLA	STS-B	SST-2	QQP	QNLI	MNLI	Avg.
Full FT†	-	88.2	84.1	64.6	90.6	94.3	92.0	92.7	87.5	86.7
LoRA†	8	89.9	85.9	62.4	91.4	94.4	90.8	92.6	86.9	86.8
PiSSA	8	90.9	86.4	63.0	92.1	95.3	91.7	93.5	87.9	87.6
MiLoRA	8	90.5	86.1	62.5	91.6	95.2	91.5	93.1	87.6	87.3
AdaLoRA†	8	90.2	85.2	61.6	91.2	94.5	90.1	93.1	87.3	86.7
MELoRA†	8	90.9	86.6	64.1	91.9	95.4	90.8	93.2	87.2	87.5
LoRA-XS	8	90.9	86.0	63.8	92.5	95.4	91.1	93.3	87.5	87.6
DoRA	8	91.5	87.2	64.8	91.9	95.6	92.2	93.5	88.3	88.1
NNI-TPE	8	90.4	86.3	63.0	91.1	94.9	92.0	93.2	87.6	87.3
NOMAD	8	90.2	86.3	63.8	91.7	95.1	91.2	93.1	87.2	87.3
LeLoRA	8	<u>92.5</u>	<u>88.7</u>	<u>65.9</u>	<u>93.2</u>	<u>96.0</u>	<u>92.9</u>	<u>94.2</u>	<u>89.4</u>	<u>89.1</u>
LeLoRA	16	**93.1**	**89.5**	**66.7**	**93.6**	**96.3**	**93.4**	**94.5**	**90.0**	**89.6**

Table 1: Evaluation results on the GLUE benchmark using RoBERTa-base. The rank values for LeLoRA (NNI-TPE and NOMAD) correspond to the largest effective dimension (rank). When LeLoRA's maximum effective dimension is set to match the baseline ranks, all methods share the same latent capacity upper bound. In practice, however, LeLoRA updates far fewer parameters by often selecting smaller dimensions (Table 4). Results marked with † are taken from MELoRA (Ren et al., 2024). The highest and second-highest scores are in **bold** and <u>underlined</u>.

Accordingly, at each iteration, the policy network generates a distribution of adaptation strategies based on the extracted matrix-wise characteristics, $p(\boldsymbol{a} \mid \boldsymbol{\xi}; \boldsymbol{\Omega})$. An adaptation strategy for each weight matrix is then sampled from this distribution. Subsequently, the parameters of the LLM are updated using gradient descent as follows:

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta_1 \frac{1}{n}[\sum\nolimits_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \mathcal{L}\left(\mathcal{F}_{\boldsymbol{\theta}}\left(\boldsymbol{x}_i\right), \boldsymbol{y}_i\right)], \tag{6}$$

where n denotes the batch size, and η_1 represents the learning rate. Notably, based on the sampled strategies, only the trainable components of the matrices $\boldsymbol{W}$ within $\boldsymbol{\theta}$ are updated in each iteration.

Moreover, given $\boldsymbol{\theta}$, the optimization subproblem for the policy network parameters $\boldsymbol{\Omega}$ can be formulated as follows:

$$\min_{\boldsymbol{\Omega}} \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \mathbb{E}_{\boldsymbol{a}\sim p(\boldsymbol{a}\mid\boldsymbol{\xi};\boldsymbol{\Omega})}[\mathcal{R}(\boldsymbol{\Omega})]. \tag{7}$$

Due to the presence of non-differentiable operations, specifically the selection of the effective dimension and location index from discrete sets, which impede gradient backpropagation to the policy network, we utilize the simple and straightforward algorithm, REINFORCE (Williams, 1992) for its optimization. A value baseline is used, with the running mean of recent rewards subtracted to reduce gradient variance.

Define $Q(\boldsymbol{\Omega}) = \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \mathbb{E}_{\boldsymbol{a}\sim p(\boldsymbol{a}\mid\boldsymbol{\xi};\boldsymbol{\Omega})}[\mathcal{R}]$. The gradient of the objective function with respect to $\boldsymbol{\Omega}$, termed $\nabla_{\boldsymbol{\Omega}} Q(\boldsymbol{\Omega})$, can be computed as follows:

$$\begin{aligned}
\nabla_{\boldsymbol{\Omega}} Q(\boldsymbol{\Omega}) &= \nabla_{\boldsymbol{\Omega}} \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \mathbb{E}_{\boldsymbol{a}\sim p(\boldsymbol{a}\mid\boldsymbol{\xi};\boldsymbol{\Omega})}[\mathcal{R}] \\
&= \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \int_{\boldsymbol{a}} \mathcal{R} \nabla_{\boldsymbol{\Omega}} p(\boldsymbol{a} \mid \boldsymbol{\xi}; \boldsymbol{\Omega}) d\boldsymbol{a} \\
&= \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \int_{\boldsymbol{a}} \mathcal{R} p(\boldsymbol{a} \mid \boldsymbol{\xi}; \boldsymbol{\Omega}) \nabla_{\boldsymbol{\Omega}} \log p(\boldsymbol{a} \mid \boldsymbol{\xi}; \boldsymbol{\Omega}) d\boldsymbol{a} \\
&= \mathbb{E}_{\{(\boldsymbol{x},\boldsymbol{y})\}} \mathbb{E}_{\boldsymbol{a}\sim p(\boldsymbol{a}\mid\boldsymbol{\xi};\boldsymbol{\Omega})} \left[\mathcal{R} \nabla_{\boldsymbol{\Omega}} \log p(\boldsymbol{a} \mid \boldsymbol{\xi}; \boldsymbol{\Omega})\right].
\end{aligned} \tag{8}$$

Similar to solving Eq. (5), we sample the adaptation strategy from the conditional distribution to identify the trainable components of the weight matrices. The gradient concerning the parameters of the policy network can be approximated as

$$\nabla_{\boldsymbol{\Omega}} Q(\boldsymbol{\Omega}) \approx \frac{1}{n} \sum\nolimits_{i=1}^{n} \mathcal{R}_i \cdot \nabla_{\boldsymbol{\Omega}} \log p\left(\boldsymbol{a}_i \mid \boldsymbol{\xi}_i; \boldsymbol{\Omega}\right). \tag{9}$$

The policy network can then be updated using gradient descent, as follows:

$$\boldsymbol{\Omega}^{t+1} = \boldsymbol{\Omega}^t - \eta_2 \nabla_{\boldsymbol{\Omega}} Q\left(\boldsymbol{\Omega}^t\right), \tag{10}$$

where η_2 denotes the learning rate. Consequently, the LLM and the policy network are updated alternately using Eqs. (6) and(10), respectively.

4 Experiments

We conduct extensive experiments across various tasks, including NLU, commonsense reasoning, math reasoning, code generation, and instruction following. Ten LLMs ranging from 125M to 70B parameters are involved. Due to space constraints, experiments on code generation are provided in Appendix F.

Method	Rank	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
LoRA‡	32	70.8	85.2	79.9	91.7	84.3	84.2	71.2	79.0	80.8
PiSSA	32	73.0	86.9	78.3	89.8	84.6	83.9	74.3	80.6	81.4
MiLoRA‡	32	68.8	86.7	77.2	92.9	85.6	86.8	75.5	81.8	81.9
AdaLoRA	32	74.8	86.8	80.0	84.0	81.8	87.7	74.9	81.5	81.4
MELoRA	32	75.0	87.7	79.9	84.1	82.9	87.6	75.7	81.8	81.8
LoRA-XS*	32	66.6	85.8	79.4	90.1	85.2	87.0	76.5	81.8	81.6
DoRA*	32	74.6	89.3	79.9	95.5	85.6	90.5	80.4	85.8	85.2
NNI-TPE	32	71.1	85.8	80.2	92.2	84.6	84.9	72.0	80.3	81.4
NOMAD	32	71.5	86.0	80.2	92.1	84.9	85.0	72.3	81.4	81.7
LeLoRA	32	<u>75.7</u>	<u>90.1</u>	<u>81.0</u>	<u>96.0</u>	<u>86.9</u>	<u>91.5</u>	<u>81.3</u>	<u>86.1</u>	<u>86.1</u>
LeLoRA	64	**75.9**	**90.4**	**81.7**	**96.2**	**87.3**	**91.7**	**82.2**	**86.8**	**86.5**

Table 2: Evaluation results for commonsense reasoning tasks using the LLaMA3-8B model. The symbol ‡ denotes MiLoRA results (Wang et al., 2024), and * indicates values reported in the original paper.

4.1 Main Comparative Experiments

4.1.1 Natural Language Understanding Tasks

We begin by conducting experiments on the GLUE benchmark, which offers a set of NLU tasks, including MNLI (Williams et al., 2018), SST-2 (Socher et al., 2013), MRPC (Dolan and Brockett, 2005), CoLA (Warstadt et al., 2019), QNLI (Rajpurkar et al., 2018), QQP (Wang et al., 2017), RTE (Bentivogli et al., 2017), and STS-B (Cer et al., 2017). The training configurations and evaluation metrics are consistent with those utilized in prior studies (Ren et al., 2024; Meng et al., 2024), and we employ the RoBERTa-base (Liu et al., 2019) and RoBERTa-large models as the backbones, with the rank for the compared baselines set to 8. Matthew's correlation is reported for CoLA, Pearson correlation for STS-B, and accuracy for the remaining tasks. Moreover, the value of α, which controls the relative strengths of the two loss terms, is selected from the set $\{0.5, 0.8, 1\}$. Further details on the hyperparameter settings and the baseline methods are provided in Appendices B and C.

Comparative results for RoBERTa-base are shown in Table 1, and those for RoBERTa-large are in Appendix D. Although our comparison setup, in which LeLoRA updates fewer parameters than the baselines, is less favorable for downstream adaptation, results show that LeLoRA still significantly outperforms them, owing to its selective updates of the most impactful regions of the parameter matrices. Specifically, when the maximum effective dimension in LeLoRA matches the rank of the baselines, it achieves an average improvement of 2.3% over vanilla LoRA and exceeds the best-performing baseline by 1.0%. Moreover, increasing the maximum effective dimension can further improve the performance of LeLoRA, as it effectively broadens the scope of policy selection.

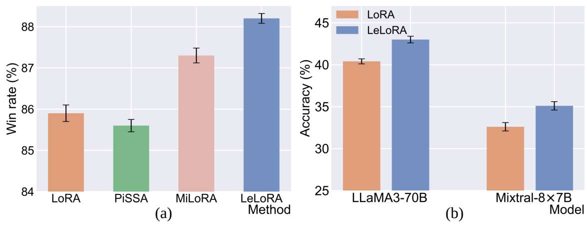

Figure 2: (a) Instruction tuning results. Win rate denotes the percentage of cases where fine-tuned LLM outputs are preferred over text-davinci-003. (b) MATH performance of scaled models.

4.1.2 Commonsense Reasoning Tasks

We fine-tune the LLaMA3-8B (Dubey et al., 2024), LLaMA-13B (Touvron et al., 2023a), and LLaMA2-13B (Touvron et al., 2023b) models on the Commonsense170K dataset (Hu et al., 2023). For evaluation, we utilize eight commonsense reasoning datasets, including BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC-e, ARC-c (Clark et al., 2018), and OBQA (Mihaylov et al., 2018). The tasks are formulated as multiple-choice questions, and we report the accuracy across all datasets. For all compared baselines, the rank is set to 32, with additional settings provided in Table 7 of the Appendix.

Table 2 presents results for LLaMA3-8B, with additional results for the other three models in Appendix E. LeLoRA consistently outperforms previous baselines across various datasets and LLMs.

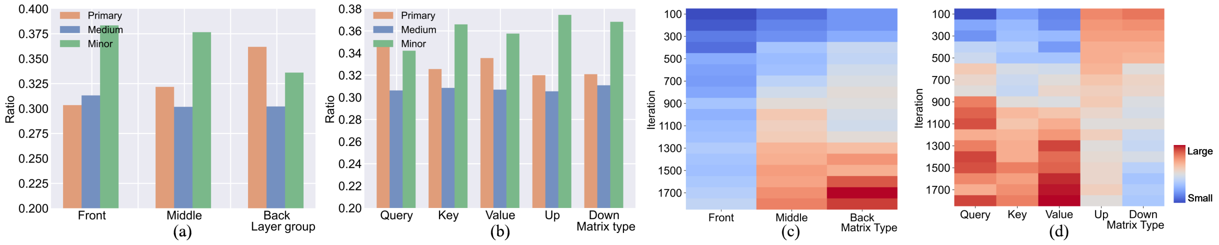

Figure 3: Analysis of the positions of tunable parameters across matrices in various layers (a) and across different matrix types (b). Average dimensions of matrices in various layers (c) and of different matrix types (d) throughout training. These analyses are conducted on commonsense reasoning tasks utilizing the LLaMA2-7B model.

Method	GSM8K	MATH	Avg.
Mistral-7B			
Full FT	67.08	18.72	42.90
LoRA	67.56	19.83	43.70
PiSSA	72.79	21.62	47.21
MiLoRA	72.68	21.03	46.86
LeLoRA	**74.67**	**23.61**	**49.14**
Gemma-7B			
Full FT	71.35	22.81	47.08
LoRA	74.93	31.06	53.00
PiSSA	77.98	31.87	54.93
MiLoRA	78.02	31.54	54.78
LeLoRA	**79.75**	**33.50**	**56.63**

Table 3: Math reasoning results on GSM8K and MATH, with the value of rank set to 64.

Specifically, it achieves average improvements of 3.2% over LoRA across the four LLMs. Moreover, LeLoRA outperforms the best baselines by 1.4% on average. Notably, LeLoRA consistently outperforms PEFT methods with optimized hyperparameters, such as NNI-TPE and NOMAD, due to its fine-grained, matrix-wise adaptive strategy, while the baselines optimize only global hyperparameters shared across matrices.

4.1.3 Instruction Following Tasks

The experimental settings follow those outlined by Wang et al. (2024), wherein we fine-tune LLaMA2-7B using Ultrafeedback (Cui et al., 2024). Consistent with previous studies, we utilize AlpacaEval v1.0 (Li et al., 2023) for evaluation, reporting the win rate relative to text-davinci-003, with GPT4-0613 serving as the annotator (Achiam et al., 2023). The comparison results among LoRA, PiSSA, MiLoRA, and LeLoRA are presented in Fig. 2(a), where LeLoRA significantly outperforms previous baselines.

4.1.4 Math Reasoning Tasks

We fine-tune two LLMs (i.e., Mistral-7B (Jiang et al., 2023), and Gemma-7B (Team et al., 2024)) on MetaMathQA (Yu et al., 2024), which consists of 395K samples drawn from the training sets of GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021). For evaluation, we utilize the test sets from GSM8K and MATH, reporting the exact match ratio against the ground truth for each set. The baselines include LoRA, PiSSA, and MiLoRA, with the rank set to 64. Results for the Mistral-7B and Gemma-7B models are presented in Table 3. LeLoRA consistently outperforms previous PEFT methods across various LLMs. Additionally, we evaluate the performance of LeLoRA on larger-scale models: LLaMA3-70B (Dubey et al., 2024) and Mixtral-8×7B (Jiang et al., 2024a). Due to computational constraints, we set the rank parameter for LoRA and the maximum dimension for LeLoRA to 4. Fig. 2(b) demonstrates that LeLoRA significantly outperforms LoRA, even for larger-scale LLMs.

4.2 Analytical Experiments

Various analytical experiments are conducted to further understand the LeLoRA framework.

4.2.1 Adaptation Strategy Analysis

The strategies generated during the training procedure have been thoroughly analyzed. For clarity, we divide the LLM layers into front, middle, and back segments. As shown in Fig. 3(a), minor components are predominantly fine-tuned in the front and middle layers, while back layers primarily adjust primary components, reflecting that front layers encode general features and back layers capture task-specific information. Fig. 3(b) reveals distinct allocation patterns of primary, medium, and minor components across different matrix types. Fig. 3(c) demonstrates that the average adaptation dimen-

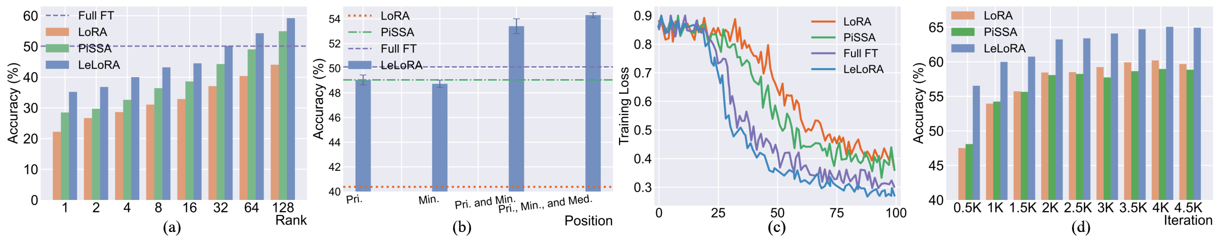

Figure 4: (a) Performance variation regarding different volumes of tunable parameters. (b) Performance variation regarding the tuning of parameters across various positions (primary, minor, and medium). (c) and (d) Comparison of loss (first 100 steps) and accuracy values among LoRA, PiSSA, and LeLoRA during the training process.

Rank	32		64	
Method	**# Params.**	**Acc.**	**# Params.**	**Acc.**
LoRA	56M	77.6	112M	77.5
PiSSA	56M	73.8	112M	78.4
DoRA	57M	79.7	113M	79.4
AdaLoRA	54M	78.2	106M	78.5
LeLoRA	**25M**	**81.0**	**39M**	**81.6**

Table 4: Comparison of the number of tunable parameters and performance across various approaches on the commonsense reasoning tasks.

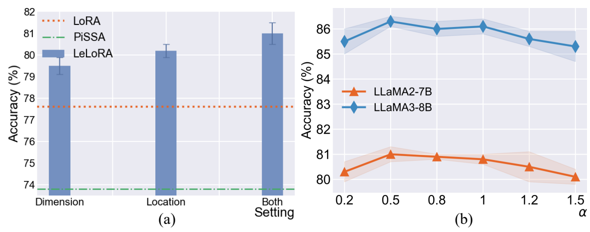

Figure 5: (a) Ablations of the adaptation strategies. (b) Sensitivity tests of the modulating factor α, which controls the relative strengths of the two loss terms.

sion grows over time, with deeper layers generally exceeding shallower ones. Moreover, Fig. 3(d) indicates that feed-forward matrices (Up/Down) are tuned more in early phases, whereas attention matrices dominate later.

4.2.2 Varying Strategy Sets

Given the importance of candidate set selection, we examine performance across different settings of effective dimensions and location indices. The model is trained for one epoch using the MetaMathQA-100K dataset (Meng et al., 2024). Results on GSM8K are shown in Figs. 4(a) and (b), with others provided in Appendix H. As shown in Fig. 4(a), expanding the effective dimension set leads to enhanced performance. Moreover, LeLoRA consistently outperforms the compared baselines across varying amounts of tunable parameters. Additionally, Fig. 4(b) shows that adaptively adjusting the positions of the trainable parameters yields significantly better performance compared to approaches that rely on predefined positions.

4.2.3 Training Analysis

We analyze the adaptation process of LeLoRA by fine-tuning LLaMA2-7B on a single NVIDIA A800 GPU. As shown in Table 4, LeLoRA, which dynamically selects the most beneficial components of the weight matrices for fine-tuning, significantly reduces the number of tunable parameters and outperforms all baselines, thereby achieving a better balance between preserving the pre-trained knowledge and adapting to new tasks. Moreover, although LeLoRA incurs additional overhead from components like REINFORCE-based policy network training, it achieves a superior trade-off between effectiveness and efficiency compared to existing methods. We also compare LeLoRA with previous baselines under roughly equal amounts of updated parameters. As shown in Table 12, LeLoRA achieves substantial performance gains on this occasion. Furthermore, a comparison of the loss and accuracy values during the training process is provided in Figs. 4(c) and (d). LeLoRA converges faster and continuously outperforms the compared baselines. Since the policy network's reward is directly based on the LLM loss, the results also reflect its training stability.

4.2.4 Ablation and Sensitivity Studies

We evaluate LeLoRA's performance by separately examining the effects of the dimension and position of the tunable parameters. When considering only the dimension, adaptation is restricted to the primary components. Moreover, when considering only the location, all matrices are fine-tuned with a dimension of 32. Fig. 5(a) shows the performance on commonsense reasoning tasks using LLaMA2-

7B. The results clearly highlight the importance of both dynamic and adaptive adjustments to the volume and position of trainable components in enhancing adaptation performance. Furthermore, a sensitivity analysis of the hyperparameter α is conducted. From Fig. 5(b), the performance remains stable for $\alpha \in [0.5, 1]$.

5 Conclusion

This study presents a novel learnable low-rank adaptation framework, termed LeLoRA, which is designed for the effective and parameter-efficient adaptation of LLMs. Within this framework, a policy network dynamically and adaptively generates matrix-specific adaptation strategies, specifying the components to be tuned. The LLM and the policy network are alternately updated using a reinforcement learning-based optimization algorithm, allowing the generated strategies to adapt to the evolving dynamics of the training process. Extensive experiments across various tasks and LLMs conclusively demonstrate the efficacy of the LeLoRA approach.

Limitations

While LeLoRA demonstrates strong performance in fine-tuning LLMs, it has certain limitations that open up avenues for future research. First, although our approach allows adaptive and dynamic adjustment of fine-tuning strategies for each weight matrix, the integration of REINFORCE-based adaptive parameter allocation inevitably introduces computational overhead. Future work could investigate more efficient optimization algorithms to further enhance the training of our framework. Second, our approach considers four main characteristics of each weight matrix, such as singular values and matrix norms, to determine its adaptation strategy. While these indicators provide a comprehensive understanding of the role and position of matrices within LLMs, the set of metrics remains limited. An avenue for future research is to incorporate additional task-specific and training-dynamics-related indicators, thereby enabling a more comprehensive characterization of each weight matrix. Moreover, our approach considers only two adaptation strategies, the effective dimension and the location index, which govern the trainable components of the weight matrices. An extension of our LeLoRA framework in future research could involve integrating other fine-tuning strategies, such as step size and weight decay, to further refine the adaptation process of LLMs.

Acknowledgments

This work was supported by the NSFC under Grant 625B2009.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Paul Albert, Frederic Z. Zhang, Hemanth Saratchandran, Cristian Rodriguez-Opazo, Anton van den Hengel, and Ehsan Abbasnejad. 2025. Randlora: Full rank parameter-efficient fine-tuning of large models. In *The Thirteenth International Conference on Learning Representations*.

Sercan Ö Arik and Tomas Pfister. 2021. Tabnet: Attentive interpretable tabular learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6679–6687.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. 2024. Lora-xs: Low-rank adaptation with extremely small number of parameters. *arXiv preprint arXiv:2405.17604*.

Luisa Bentivogli, Ido Dagan, and Bernardo Magnini. 2017. The recognizing textual entailment challenges: Datasets and methodologies. *Handbook of Linguistic Annotation*, pages 1119–1147.

Gaurav Bhatnagar, QM Jonathan Wu, and Pradeep K Atrey. 2013. Secure randomized image watermarking based on singular value decomposition. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 10(1):1–21.

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

Albrecht Böttcher and David Wenzel. 2008. The frobenius norm and the commutator. *Linear Algebra and Its Applications*, 429(8-9):1864–1885.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Yuyan Chen, Qiang Fu, Ge Fan, Lun Du, Jian-Guang Lou, Shi Han, Dongmei Zhang, Zhixu Li, and Yanghua Xiao. 2023. Hadamard adapter: An extreme parameter-efficient adapter tuning method for pre-trained language models. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 276–285.

Zhiwei Chen, Yupeng Hu, Zixu Li, Zhiheng Fu, Xuemeng Song, and Liqiang Nie. 2025. Offset: Segmentation-based focus shift revision for composed image retrieval. In *Proceedings of the ACM International Conference on Multimedia*, page 6113–6122.

Kexin Chu, Zixu Shen, Dawei Xiang, and Wei Zhang. 2025. Safekv: Safe kv-cache sharing in llm serving. In *Machine Learning for Computer Architecture and Systems*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

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*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. Ultrafeedback: Boosting language models with scaled AI feedback. In *Proceedings of the 41st International Conference on Machine Learning*, pages 9722–9744.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Carl Eckart and Gale Young. 1936. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 17783–17806.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *Proceedings of the 10th International Conference on Learning Representations*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

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 *Proceedings of 10th International Conference on Learning Representations*.

Yuxuan Hu, Jing Zhang, Xiaodong Chen, Zhe Zhao, Cuiping Li, and Hong Chen. 2025. Lors: Efficient low-rank adaptation for sparse large language model. *arXiv preprint arXiv:2501.08582*.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024a. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Jiantong Jiang, Peiyu Yang, Rui Zhang, and Feng Liu. 2025a. Towards efficient large language model serving: A survey on system-aware kv cache optimization. *Authorea Preprints*.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and 1 others. 2024b. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*.

Yuxuan Jiang and Francis Ferraro. 2026a. Beyond math: Stories as a testbed for memorization-constrained reasoning in llms. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5590–5607.

Yuxuan Jiang and Francis Ferraro. 2026b. Scribe: Structured mid-level supervision for tool-uing language models. *arXiv preprint arXiv:2601.03555*.

Yuxuan Jiang, Dawei Li, and Frank Ferraro. 2025b. Drp: Distilled reasoning pruning with skill-aware step decomposition for efficient large reasoning models. *arXiv preprint arXiv:2505.13975*.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 565–576, Online. Association for Computational Linguistics.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. 2024. Vera: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models.

Zixu Li, Yupeng Hu, Zhiwei Chen, Qinlei Huang, Guozhi Qiu, Zhiheng Fu, and Meng Liu. 2026a. Retrack: Evidence-driven dual-stream directional anchor calibration network for composed video retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 23373–23381.

Zixu Li, Yupeng Hu, Zhiwei Chen, Shiqi Zhang, Qinlei Huang, Zhiheng Fu, and Yinwei Wei. 2026b. Habit: Chrono-synergia robust progressive learning framework for composed image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6762–6770.

Debin Liu, Laurence T Yang, Puming Wang, Ruonan Zhao, and Qingchen Zhang. 2022a. Tt-tsvd: A multimodal tensor train decomposition with its application in convolutional neural networks for smart healthcare. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 18(1s):1–17.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, pages 32100–32121.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022b. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xueqi Ma, Shuo Yang, Yanbei Jiang, Shu Liu, Zhenzhen Liu, Jiayang Ao, Xingjun Ma, Sarah Monazam Erfani, and James Bailey. 2026. Attention in space: Functional roles of vlm heads for spatial reasoning. *arXiv preprint arXiv:2603.20662*.

Reem I Masoud, Martin Feriancc, Philip Colin Treleaven, and Miguel RD Rodrigues. 2024. Llm alignment using soft prompt tuning: The case of cultural alignment. In *Workshop on Socially Responsible Language Modelling Research*.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2024. Pissa: Principal singular values and singular vectors adaptation of large language models. In *The 38th Annual Conference on Neural Information Processing Systems*.

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

Lin Mu, Xiaoyu Wang, Li Ni, Yang Li, Zhize Wu, Peiquan Jin, and Yiwen Zhang. 2025. Denselora: Dense low-rank adaptation of large language models. *arXiv preprint arXiv:2505.23808*.

Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. 2024. Lisa: Layerwise importance sampling for memory-efficient large language model fine-tuning. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, pages 57018–57049.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789. Association for Computational Linguistics.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

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

Pratyusha Sharma, Jordan T. Ash, and Dipendra Misra. 2024. The truth is in there: Improving reasoning in language models with layer-selective rank reduction. In *Proceedings of the 12th International Conference on Learning Representations*.

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, Seattle, Washington, USA. Association for Computational Linguistics.

Dilara Soylu, Christopher Potts, and Omar Khattab. 2024. Fine-tuning and prompt optimization: Two great steps that work better together. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10696–10710.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, and 1 others. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Zichen Tian, Yaoyao Liu, and Qianru Sun. 2025. Meta-learning hyperparameters for parameter efficient fine-tuning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 23037–23047.

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

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

Christophe Tribes, Sacha Benarroch-Lelong, Peng Lu, and Ivan Kobyzev. 2023. Hyperparameter optimization for large language model instruction-tuning. *arXiv preprint arXiv:2312.00949*.

Eileen Wang, Hiba Arnaout, Dhita Pratama, Shuo Yang, Dangyang Liu, Jie Yang, Josiah Poon, Jeff Pan, and Caren Han. 2026. Mmcomet: A large-scale multimodal commonsense knowledge graph for contextual reasoning. *arXiv preprint arXiv:2603.01055*.

Hanqing Wang, Zeguan Xiao, Yixia Li, Shuo Wang, Guanhua Chen, and Yun Chen. 2024. Milora: Harnessing minor singular components for parameter-efficient llm finetuning. *arXiv preprint arXiv:2406.09044*.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, page 4144–4150.

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

Jingyu Wei, Yi Su, Kele Xu, Lingbin Zeng, Bo Liu, and Huaimin Wang. 2024. Demonstrative instruction following in multimodal llms via integrating low-rank adaptation with ensemble learning. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 11435–11441.

Martin Weyssow, Xin Zhou, Kisub Kim, David Lo, and Houari Sahraoui. 2025. Exploring parameter-efficient fine-tuning techniques for code generation with large language models. *ACM Transactions on Software Engineering and Methodology*, 34(7):1–25.

Adina Williams, Nikita Nangia, and Samuel 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, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256.

Bingyang Wu, Ruidong Zhu, Zili Zhang, Peng Sun, Xuanzhe Liu, and Xin Jin. 2024. Dlora: Dynamically orchestrating requests and adapters for lora llm serving. In *Proceedings of the 18th USENIX Symposium on Operating Systems Design and Implementation*, pages 911–927.

Dawei Xiang, Wenyan Xu, Kexin Chu, Tianqi Ding, Zixu Shen, Yiming Zeng, Jianchang Su, and Wei Zhang. 2025. Promptsculptor: Multi-agent based text-to-image prompt optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 774–786.

ShiLin Xiao, Tianxiang Xu, Canran Xiao, Weihao Luo, Liwei Hou, and Chuangxin Zhao. 2026. Meta-ucf: Unified task-conditioned lora generation for continual learning in large language models. In *The Fourteenth International Conference on Learning Representations*.

Zequn Xie. 2026. Conquer: Context-aware representation with query enhancement for text-based person search. *arXiv preprint arXiv:2601.18625*.

Zequn Xie, Xin Liu, Boyun Zhang, Yuxiao Lin, Sihang Cai, and Tao Jin. 2026. Hvd: Human vision-driven video representation learning for text-video retrieval. *arXiv preprint arXiv:2601.16155*.

Zhipei Xu, Xuanyu Zhang, Qing Huang, Xing Zhou, and Jian Zhang. 2025. Avatarshield: Visual reinforcement learning for human-centric synthetic video detection. *arXiv preprint arXiv:2505.15173*.

Shuo Yang, Caren Han, Yihao Ding, Shuhe Wang, and Eduard Hovy. 2026. Tooltree: Efficient llm tool planning via dual-feedback monte carlo tree search and bidirectional pruning. In *The Fourteenth International Conference on Learning Representations*.

Hua Ye, Siyuan Chen, Ziqi Zhong, Canran Xiao, Haoliang Zhang, Yuhan Wu, and Fei Shen. 2026. Seeing through the conflict: Transparent knowledge conflict handling in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 34423–34431.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *Proceedings of the 12th International Conference on Learning Representations*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

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

Mingjie Zhang, Xiaoling Zhou, Yuxiao Luo, Yiyu Liu, Shikun Zhang, and Wei Ye. 2026. Askd: Reinforcement learning-style knowledge distillation with quality-adaptive skewness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 34781–34789.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023b. Adaptive budget allocation for parameter-efficient fine-tuning. In *Proceedings of the 11th International Conference on Learning Representations*.

Renrui Zhang, Jiaming Han, Chris Liu, Aojun Zhou, Pan Lu, Yu Qiao, Hongsheng Li, and Peng Gao. 2024. Llama-adapter: Efficient fine-tuning of large language models with zero-initialized attention. In *The Twelfth International Conference on Learning Representations*.

Xueyan Zhang, Jinman Zhao, Zhifei Yang, Yibo Zhong, Shuhao Guan, Linbo Cao, and Yining Wang. 2025a. Uora: Uniform orthogonal reinitialization adaptation in parameter efficient fine-tuning of large mdels. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11709–11728.

Yue Zhang, Hehe Fan, Wei Ji, Yongkang Wong, Roger Zimmermann, and Yi Yang. 2025b. Prompt-aware adapter: Learning adaptive visual tokens for multimodal large language models. *IEEE Transactions on Artificial Intelligence*.

Jinman Zhao, Xueyan Zhang, Jiaru Li, Jingcheng Niu, Yulan Hu, Erxue Min, and Gerald Penn. 2025. Tiny budgets, big gains: Parameter placement strategy in parameter super-efficient fine-tuning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 6315–6333.

Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. 2024. Opencodeinterpreter: Integrating code generation with execution and refinement. *arXiv preprint arXiv:2402.14658*.

Xiaofan Zheng, Minnan Luo, and Xinghao Wang. 2025a. Unveiling fake news with adversarial arguments generated by multimodal large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7862–7869.

Xiaofan Zheng, Zinan Zeng, Heng Wang, Yuyang Bai, Yuhan Liu, and Minnan Luo. 2025b. From predictions to analyses: Rationale-augmented fake news detection with large vision-language models. In *Proceedings of the ACM on Web Conference 2025*, page 5364–5375.

Xiaofan Zheng, Huixuan Zhang, and Xiaojun Wan. 2025c. Tracing training footprints: A calibration approach for membership inference attacks against multimodal large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 17179–17191.

Hongyun Zhou, Xiangyu Lu, Wang Xu, Conghui Zhu, and Tiejun Zhao. 2025a. Lora-drop: Efficient lora parameter pruning based on output evaluation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5530–5543.

Xiaoling Zhou, Ou Wu, Weiyao Zhu, and Ziyang Liang. 2022. Understanding difficulty-based sample weighting with a universal difficulty measure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 68–84. Springer.

Xiaoling Zhou, Wei Ye, Zhemg Lee, Lei Zou, and Shikun Zhang. 2025b. Valuing training data via causal inference for in-context learning. *IEEE Transactions on Knowledge and Data Engineering*.

Xiaoling Zhou, Wei Ye, Yidong Wang, Chaoya Jiang, Zhemg Lee, Rui Xie, and Shikun Zhang. 2024. Enhancing in-context learning via implicit demonstration augmentation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2810–2828.

Xiaoling Zhou, Mingjie Zhang, Zhemg Lee, Yuncheng Hua, Wei Ye, Flora D Salim, Shikun Zhang, and 1 others. 2025c. Boosting resilience of large language models through causality-driven robust optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Xiaoling Zhou, Mingjie Zhang, Zhemg Lee, Wei Ye, and Shikun Zhang. 2025d. Hademif: Hallucination detection and mitigation in large language models. In *The Thirteenth International Conference on Learning Representations*.

Bojia Zi, Xianbiao Qi, Lingzhi Wang, Jianan Wang, Kam-Fai Wong, and Lei Zhang. 2023. Delta-lora: Fine-tuning high-rank parameters with the delta of low-rank matrices. *arXiv preprint arXiv:2309.02411*.

A Ethical Considerations

In our experiments, we have utilized publicly available pre-trained models, as well as publicly accessible datasets. All models and datasets used in this study have been meticulously processed and curated by their respective publishers to mitigate ethical issues.

B Detailed Experimental Configurations

The settings for the GLUE benchmark using the RoBERTa-base (Liu et al., 2019) and RoBERTa-large models are provided in Tables 5 and 6, respectively. Moreover, the hyperparameter configurations for commonsense reasoning, math reasoning, code generation, and instruction following tasks are summarized in Table 7. The learning rate for the policy network is set to 1×10^{-3}, while the learning rates for the LLMs correspond to the values specified in Tables 5, 6, and 7. The validation set consists of 32 samples, which are randomly drawn from the training data. As shown in these tables, the set of possible location indexes includes three options: 0, $m/2$, and m, which correspond to the head, middle, and tail of the matrix, respectively. Alternatively, when 0, $m/2$, or m is selected, the primary, medium, or minor components of the weight matrix will be optimized. Moreover, if 0 is selected from the set of effective dimensions, it indicates that the matrix does not require fine-tuning.

To handle the two categorical variables, namely the layer index and matrix type, they are first transformed into dense vectors using two embedding layers. These vectors are then concatenated with the matrix norm and singular values to form a unified representation for each weight matrix. For all tasks, the layer index and matrix type are embedded into 200-dimensional dense vectors. The policy network is implemented as an MLP with three hidden layers, which transform the input feature dimensions into 1,024, 256, and 64, respectively. Additionally, the network includes two output layers, each employing the Softmax function, which is responsible for predicting the effective dimension and the location index. The entire policy network is initialized using Kaiming initialization with a uniform distribution (He et al., 2015).

C Compared Baselines

A series of prevalent and advanced PEFT baselines are involved in our experiments, as detailed below:

- LoRA (Hu et al., 2022) freezes the pre-trained model weights while introducing trainable low-rank decomposition matrices into each layer of the Transformer architecture.

- PiSSA (Meng et al., 2024) fine-tunes only the primary components of weight matrices while keeping the minor components fixed.

- MiLoRA (Wang et al., 2024) focuses on adapting the minor singular components to retain the pre-trained knowledge.

- AdaLoRA (Zhang et al., 2023b) allocates parameter budgets across different weight matrices based on their importance scores.

Hyperparameters	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B
Effective dimension set	{0, 2, 4, 8} or {0, 2, 4, 8, 16}							
Location index set	$\{0, m/2, m\}$							
Optimizer	AdamW							
Learning rate	5×10^{-4}	5×10^{-4}	4×10^{-4}	4×10^{-4}	4×10^{-4}	4×10^{-4}	4×10^{-4}	4×10^{-4}
Scheduler	Linear							
Batch size	128	128	128	64	128	128	128	128
Warmup ratio	0.06							
Weight decay	0.1							
Epochs	30	60	30	80	25	25	80	40
Learning rate for the policy network	1×10^{-3}							
Placement	query, value							
Evaluation metrics	Accuracy	Accuracy	Accuracy	Matthews correlation	Accuracy	Accuracy	Accuracy	Pearson correlation

Table 5: Hyperparameter settings for the GLUE benchmark using the RoBERTa-base model. For a fair comparison, we adopt the configurations outlined by Ren et al. (2024), with the exception of the hyperparameters specific to our approach. In this context, m denotes the maximum dimensionality of the singular values.

Hyperparameters	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B
Effective dimension set	{0, 2, 4, 8} or {0, 2, 4, 8, 16}							
Location index set	$\{0, m/2, m\}$							
Optimizer	AdamW							
Learning rate	1×10^{-4}	2×10^{-4}	6×10^{-4}	4×10^{-4}	1×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Scheduler	Linear							
Batch size	32	32	16	16	6	32	16	16
Warmup ratio	0.03							
Epochs	10	10	20	20	10	20	20	30
Learning rate for the policy network	1×10^{-3}							
Placement	query, value							
Evaluation metrics	Accuracy	Accuracy	Accuracy	Matthews correlation	Accuracy	Accuracy	Accuracy	Pearson correlation

Table 6: Hyperparameter settings for the GLUE benchmark using RoBERTa-large. To ensure a fair comparison, we adhere to the configurations specified by Meng et al. (2024), with the exception of the unique hyperparameters introduced in our approach.

- MELoRA (Ren et al., 2024) freezes the original model weights and trains a series of mini LoRAs during training.
- LoRA-XS (Bałazy et al., 2024) trains an additional small matrix inserted between frozen low-rank matrices.
- DoRA (Liu et al., 2024) decomposes the pretrained weights into two components, including direction and magnitude, to facilitate more nuanced updates.
- Tribes et al. (2023) investigated the selection of hyperparameters in previous PEFT approaches using two black-box optimization techniques, namely NOMAD and NNI. Based on these techniques, two corresponding PEFT methods, referred to as NOMAD and NNI-TPE, are derived.

Additionally, we include full fine-tuning in our comparisons, following the protocol outlined by Zi et al. (2023).

D More Results for GLUE Benchmark

As mentioned in the main text, we conduct experiments on the GLUE benchmark to evaluate the performance of LeLoRA on NLU tasks, using both the RoBERTa-base (Liu et al., 2019) and RoBERTa-large models. The experimental results for RoBERTa-large are presented in Table 8, while those for RoBERTa-base are provided in the main text. Our proposed LeLoRA approach consistently outperforms previous baselines, owing to its ability to dynamically and adaptively adjust the tunable components of each weight matrix.

E More Results for Commonsense Reasoning Tasks

Table 9 presents a comparative analysis of the performance of the LLaMA-13B (Touvron et al., 2023a), and LLaMA2-13B (Touvron et al., 2023b) models on commonsense reasoning (Wang et al.,

Hyperparameters	**Settings**
Effective dimension set	{0, 2, 4, 8, 16, 32} or {0, 2, 4, 8, 16, 32, 64}
Location index set	$\{0, m/2, m\}$
Dropout	0.05
Optimizer	AdamW
Learning rate	3×10^{-4}
Scheduler	Linear
Batch size	64
Warmup steps	100
Epochs	3
Learning rate for the policy network	1×10^{-3}
Placement	query, key, value, up, down

Table 7: Hyperparameter settings for commonsense reasoning, math reasoning, code generation, and instruction following tasks. To ensure a fair comparison, we follow the configurations employed by Wang et al. (2024) and Meng et al. (2024), with the exception of the unique hyperparameters specific to our approach.

Method	**Rank**	**MRPC**	**RTE**	**CoLA**	**STS-B**	**SST-2**	**QQP**	**QNLI**	**MNLI**	**Avg.**
Full FT[‡]	-	90.2	96.4	90.9	68.0	94.7	92.2	86.6	91.5	88.8
LoRA[‡]	8	90.6	96.2	90.9	68.2	94.9	91.6	87.4	92.6	89.1
PiSSA[‡]	8	90.7	<u>96.7</u>	91.9	69.0	95.1	91.6	91.0	92.9	89.9
LeLoRA	8	<u>91.8</u>	**97.6**	<u>93.1</u>	<u>70.2</u>	<u>96.2</u>	<u>92.7</u>	<u>92.4</u>	<u>94.1</u>	<u>91.0</u>
LeLoRA	16	**92.4**	**97.6**	**93.5**	**70.8**	**96.6**	**93.2**	**93.0**	**94.2**	**91.4**

Table 8: More evaluation results on the GLUE benchmark for the RoBERTa-large model. Results marked with ‡ are taken from PiSSA (Meng et al., 2024).

2026) tasks. The proposed LeLoRA approach consistently outperforms all compared baselines across various models and tasks, achieving the best overall performance.

F Code Generation Tasks

We fine-tune three models, including LLaMA2-7B, Mistral-7B (Jiang et al., 2023), and Gemma-7B (Team et al., 2024) on the CodeFeedback dataset (Zheng et al., 2024). Subsequently, the adapted LLMs are evaluated for coding proficiency using the HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) datasets. As shown in Table 10, LeLoRA demonstrates significantly better performance than the baselines across diverse code generation tasks. Specifically, LeLoRA outperforms LoRA and PiSSA by an average of 4.7% and 2.4%, respectively.

G Comparison with Methods Mitigating Manual Hyperparameter Tuning

To demonstrate the effectiveness of LeLoRA in allocating parameters across matrices compared to prior approaches, we compare it with previous methods that challenge manual parameter design, including LoRA-FA (Zhang et al., 2023a), VeRA (Kopiczko et al., 2024), and MoRA (Jiang et al., 2024b). The comparison results are presented in Table 11. From the results, LeLoRA demonstrates significantly superior performance compared to the baseline methods. This improvement can be attributed to its use of a dynamically learned function during training, which determines the update strategy for each weight matrix based on its unique training dynamics, jointly specifying both the rank and the locations of the updates.

H More Experiments for Varying Strategy Sets

We present additional experimental results that evaluate the performance of our proposed LeLoRA

Method	Rank	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
LLaMA-13B										
LoRA	32	72.1	83.5	80.5	90.5	83.7	82.8	68.3	82.4	80.5
DoRA*	32	72.4	84.9	81.5	92.4	84.2	84.2	69.6	82.8	81.5
LeLoRA	32	73.5	86.2	82.7	93.9	85.8	85.4	70.9	84.0	82.8
LeLoRA	64	**74.2**	**86.3**	**82.9**	**94.1**	**86.2**	**85.6**	**71.0**	**84.4**	**83.1**
LLaMA2-13B										
LoRA	32	74.4	85.8	80.5	92.1	85.8	86.8	75.2	85.0	83.2
DoRA	32	74.8	84.5	81.4	93.5	86.7	87.0	74.9	84.3	83.4
LeLoRA	32	76.2	87.8	82.4	94.4	87.9	89.1	77.0	87.2	85.3
LeLoRA	64	**76.7**	**88.8**	**83.4**	**95.3**	**88.5**	**89.9**	**77.9**	**88.1**	**86.1**

Table 9: Evaluation results for commonsense reasoning tasks using the LLaMA2-7B, LLaMA-13B, and LLaMA2-13B models. The symbol * indicates results reported in the original paper.

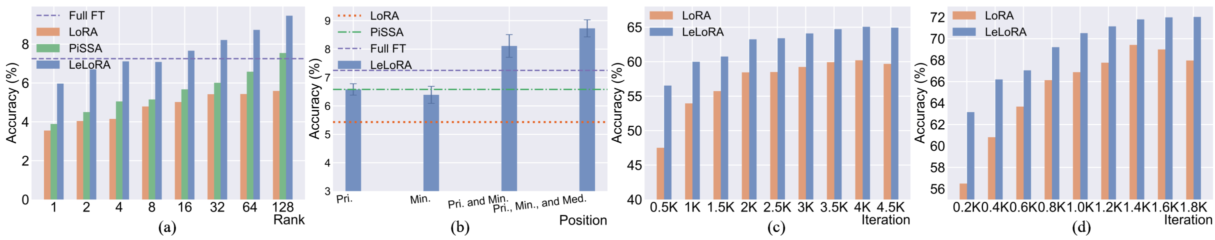

Figure 6: (a) Performance variation across different volumes of tunable parameters on the MATH dataset. (b) Performance variation across different positions (including primary, minor, and medium) of tunable parameters on the MATH dataset. The rank and the maximum dimension in our effective dimension set are both set to 64. (c) and (d) Accuracy progression over iterations for LoRA and LeLoRA on the GSM8K and BoolQ datasets.

framework across various configurations of adaptation strategies, including different sets of effective dimensions and location indices. The performance variation across different volumes of tunable parameters, as well as across different positions of these parameters, on the MATH (Hendrycks et al., 2021) dataset is shown in Figs. 6(a) and (b). The results demonstrate that expanding the set of tunable dimensions contributes to performance gains. Furthermore, when the maximum dimension in our effective dimension set matches the rank of the compared baselines, our approach consistently outperforms the compared baselines across various settings of tunable dimensions. Notably, in these cases, our approach requires significantly fewer tunable parameters compared to other methods. Additionally, incorporating trainable parameters at primary, minor, and medium positions leads to substantial performance improvements compared to methods that focus exclusively on fine-tuning parameters at a single position.

I Additional Training Analysis

Table 4 compares the number of tunable parameters and performance of various PEFT methods using the LLaMA2-7B model. The results indicate that LeLoRA, which dynamically identifies and fine-tunes the most beneficial components of the weight matrices, substantially reduces the number of tunable parameters while consistently outperforming all baseline methods. This demonstrates its superior ability to balance the retention of pre-trained knowledge with effective task-specific adaptation. Notably, similar to LoRA, our method does not introduce any additional inference latency.

Our experiments reveal that when the rank of LoRA is set to 32, and the maximum effective dimension in LeLoRA is set to 128, the number of trainable parameters for both methods is approximately the same. Table 12 further compares the performance of LeLoRA and other compared baselines under the condition of comparable trainable parameters. As observed, LeLoRA significantly outperforms previous baselines, providing additional evidence of the superiority of our proposed LeLoRA approach.

Method	HumanEval	MBPP	Avg.
LLaMA2-7B			
Full FT	21.45	35.61	28.53
LoRA	18.18	35.40	26.79
PiSSA	22.06	37.15	29.61
LeLoRA	**25.07**	**40.58**	**32.83**
Mistral-7B			
Full FT	45.21	51.43	48.32
LoRA	43.87	58.28	51.08
PiSSA	46.97	62.72	54.85
LeLoRA	**48.22**	**65.13**	**56.68**
Gemma-7B			
Full FT	46.98	55.70	51.34
LoRA	53.71	65.55	59.63
PiSSA	54.34	66.21	60.28
LeLoRA	**56.16**	**68.05**	**62.11**

Table 10: Evaluation results for coding proficiency using the HumanEval and MBPP datasets, with the value of rank setting to 64.

Moreover, from the results presented in Figs. 6(c) and (d), the accuracy of LeLoRA is consistently higher than that of LoRA during the training process, further highlighting its superiority. Furthermore, as reinforcement learning is used solely to train the policy network, while the target LLM is optimized via standard gradient descent, the overall training process remains stable.

J More Ablation Studies

We investigate the performance of LeLoRA when utilizing a single loss term to guide the learning of the policy network. The first loss term $\mathcal{L}_1$ reflects the performance of the LLM, while the second term, $\mathcal{L}_2$, captures the performance variation before and after the update in a given iteration. The results, as shown in Fig. 7, indicate that both loss terms (i.e., $\mathcal{L}_1$ and $\mathcal{L}_2$) effectively contribute to guiding the learning of the policy network.

We further investigate the impact of the policy network's architectural design on the performance of the LeLoRA framework. Specifically, we evaluate three network backbones: an MLP, ResNet-32 (He et al., 2016), and TabNet (Arik and Pfister, 2021). The comparative results, summarized in Tables 13 and 14 indicate that MLP and TabNet outperform the ResNet architecture. This can be attributed to the nature of the policy network inputs,

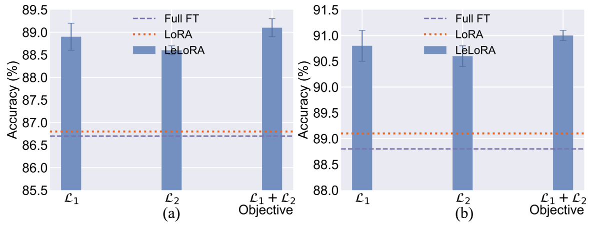

Figure 7: Ablation studies on the loss terms of the policy network in our framework using the RoBERTa-base (a) and RoBERTa-large (b) models.

numerical features including layer indices, matrix types, Frobenius norms, and singular values, which are inherently more compatible with MLP and TabNet than with convolutional networks. Considering the trade-off between computational efficiency and performance, we adopt the MLP as the default policy network in our experiments.

Method	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	Avg.
LoRA-FA	94.7	89.9	63.7	92.7	67.8	89.8	83.1
VeRA	94.8	89.3	<u>65.3</u>	92.1	78.5	90.9	85.2
MoRA	<u>95.0</u>	<u>90.4</u>	64.2	<u>92.8</u>	<u>80.1</u>	<u>91.2</u>	<u>85.6</u>
LeLoRA	**96.0**	**92.5**	**65.9**	**94.2**	**88.7**	**93.2**	**88.4**

Table 11: Comparison results between LeLoRA and other methods that challenge the reliance on manual hyperparameter tuning, evaluated using the RoBERTa-base model.

Method	Rank	# Params.	Avg.
LLaMA2-7B			
LoRA	32	56M	77.6
PiSSA	32	56M	73.8
MiLoRA	32	56M	<u>79.2</u>
LeLoRA	128	58M	**81.8**
LLaMA3-8B			
LoRA	32	57M	80.8
PiSSA	32	57M	81.4
MiLoRA	32	57M	<u>81.9</u>
LeLoRA	128	56M	**86.9**

Table 12: Performance comparison between LoRA and LeLoRA with comparable size of tunable parameters.

Method	MRPC	RTE	CoLA	SST-2
MLP	**92.5**	**88.7**	**65.9**	**96.0**
ResNet	90.6	86.5	<u>64.7</u>	94.9
TabNet	<u>92.0</u>	<u>86.9</u>	64.0	<u>95.2</u>

Table 13: Ablation studies for the structure of the policy network across various NLU tasks using the RoBERTa-base model.

Method	OBQA	PIQA	HellaSwag	HumanEval
LLaMA3-8B				
MLP	**86.1**	**90.1**	**96.0**	**42.2**
ResNet	85.2	88.8	<u>95.2</u>	41.6
TabNet	<u>85.8</u>	<u>89.5</u>	**96.0**	<u>42.0</u>
LLaMA2-13B				
MLP	**87.2**	**87.8**	**94.4**	**33.6**
ResNet	86.5	<u>87.2</u>	93.1	32.5
TabNet	<u>86.8</u>	86.5	<u>93.7</u>	<u>33.2</u>

Table 14: Ablation studies for the structure of the policy network across various commonsense reasoning and code generation tasks.