AuroRA: Breaking Low-Rank Bottleneck of LoRA with Nonlinear Mapping

Haonan Dong[1], **Wenhao Zhu**[1], **Guojie Song**[†1], **Liang Wang**[2]
[1]State Key Laboratory of General Artificial Intelligence,
School of Intelligence Science and Technology, Peking University,
[2]Alibaba Group, [†] Corresponding author
hndong25@stu.pku.edu.cn, gjsong@pku.edu.cn

Abstract

Low-Rank Adaptation (LoRA) is a widely adopted parameter-efficient fine-tuning (PEFT) method validated across NLP and CV domains. However, LoRA faces an inherent low-rank bottleneck: narrowing its performance gap with full fine-tuning requires increasing the rank of its parameter matrix, resulting in significant parameter overhead. Recent linear LoRA variants have attempted to enhance expressiveness by introducing additional linear mappings; however, their composition remains inherently linear and fails to fundamentally improve LoRA's representational capacity. To address this limitation, we propose **AuroRA**, which incorporates an Adaptive Nonlinear Layer (ANL) between two linear projectors to capture *fixed* and *learnable* nonlinearities. This combination forms an **MLP-like structure** with a compressed rank, enabling flexible and precise approximation of diverse target functions while theoretically guaranteeing lower approximation errors and bounded gradients. Extensive experiments on 22 datasets and 6 pretrained models demonstrate that **AuroRA**: **(I)** not only matches or surpasses full fine-tuning performance with only $6.18\% \sim 25\%$ of LoRA's parameters but also **(II)** outperforms competitive PEFT methods by up to 10.88% in both NLP and CV tasks, and **(III)** exhibits robust performance across various rank configurations.

1 Introduction

In recent years, pretrained models have demonstrated excellent generalization performance across numerous tasks in various domains [1, 2, 3, 4, 5, 6]. In practical applications, to further unleash their powerful capabilities on specific downstream tasks, these models often require relevant fine-tuning [7, 8, 9, 10, 11, 12]. However, the increasing size of their parameters poses a significant challenge to fine-tuning all parameters [13, 14]. To address this issue, the field of Parameter-Efficient Fine-Tuning (PEFT) has made substantial progress [13, 15, 16, 17, 18, 19, 20]. The core idea is to fine-tune only a small subset of the model's parameters while freezing the majority of the pretrained parameters, achieving performance comparable to full fine-tuning [21].

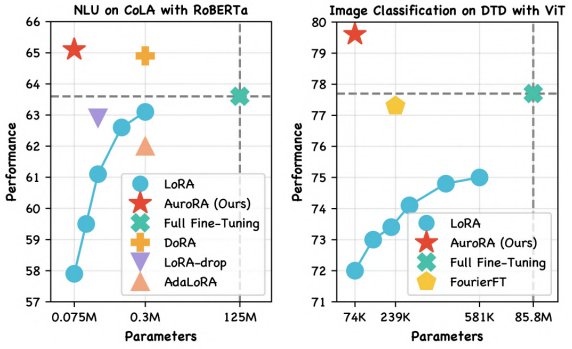

Figure 1: The trade-off between parameters and performance of various fine-tuning methods on NLP (left) and CV (right) tasks. (***Left***) In NLU, RoBERTa-Base is fine-tuned on COLA, with LoRA ranks $r = \{2, 3, 4, 6, 8\}$. (***Right***) In image classification, ViT-Base is fine-tuned on DTD, with LoRA ranks $r = \{2, 4, 6, 8, 12, 16\}$.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

https://doi.org/10.52202/085713-1241

LoRA is a commonly used state-of-the-art PEFT method [16]. Specifically, it assumes that the weight updates conform to a low-rank hypothesis and represents these updates using two low-rank matrices, i.e., $W_0 + \Delta W = W_0 + BA$. Its performance has been validated in fields such as natural language processing (NLP) [16, 22] and computer vision (CV) [23, 24]. Despite its significant success, LoRA still faces an inherent limitation, namely *the low-rank bottleneck*, as illustrated in Figure 1. As the rank of LoRA increases, the model's performance improves, thereby narrowing the gap with full fine-tuning [25]; however, the parameter cost grows proportionally with the rank, which weakens its parameter efficiency. This dilemma leads us to the first research question: ❶ ***Can we achieve a further balance between parameters and performance?***

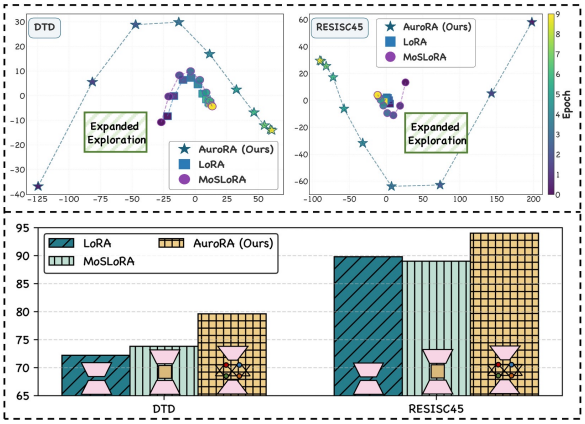

Figure 2: We evaluate LoRA, MoSLoRA, and our `AuroRA` on DTD and RESISC45 datasets, employing ViT-Base with a rank of $r = 2$. (***Upper***) We record the ΔW at the $\{0, 1, 2, ..., 9\}$-th epochs, and perform PCA visualization on these ΔW. We observe that `AuroRA` is capable of exploring a broader parameter space. (***Lower***) We present the accuracy results on both datasets.

Recently, several linear LoRA variants have emerged [26, 27, 28, 29]. They introduce an *additional matrix* between the B and A matrices of LoRA to weaken the correlation constraints between them, thereby enhancing LoRA's learning and expressive capabilities. Specifically, one approach involves the introduction of a *diagonal matrix* to facilitate singular value decomposition [26, 27], while another approach incorporates an *arbitrary matrix* to fuse subspaces [28, 29]. Nevertheless, LoRA's inherent linearity persists even when an additional matrix is introduced, preserving its fundamental structure as a linear mapping. As illustrated in Figure 2, when the rank is extremely low, MoSLoRA [28] (a linear variant that incorporates an arbitrary matrix) has only a *marginal* effect on expanding the exploration of ΔW, leading to a failure to further boost performance ($2.2\% \uparrow$ on DTD and $0.56\% \downarrow$ on RESISC45). The structural characteristics and resultant performance limitations of linear variants naturally prompt our second research question: ❷ ***Can we achieve more than marginal performance improvements by introducing a nonlinear transformation between LoRA's two linear layers?***

Motivated by the above two research questions, this paper focuses on introducing nonlinear mappings into LoRA and further compressing the rank to achieve a better balance between parameters and performance. To this end, we propose a method called `Activate Your Low-Rank Adaptation` (`AuroRA`). We revisit LoRA through the lens of linear mappings and identify two critical limitations: (**I**) *insufficient expressiveness* and (**II**) *limited training flexibility*. To fully harness LoRA's potential, `AuroRA` introduces an **A**daptive **N**onlinear **L**ayer (**ANL**) between the low-rank matrices, forming an **MLP-like structure**. ANL employs a hybrid design of *fixed* and *learnable* nonlinearities to enhance model expressivity within a more compressed rank while enabling flexible training strategies to expand the explorable parameter space (Figures 1 and 2). Theoretical analysis demonstrates that `AuroRA` not only achieves a strictly lower approximation error than LoRA but also preserves bounded gradient norms. Experiments across NLP and CV tasks confirm the *efficiency*, *generalizability*, and *robustness* of `AuroRA`. We further conduct ablation studies to dissect the contributions of fixed and learnable components, and evaluate its robustness against linear LoRA variants across multiple rank configurations. Our contributions can be summarized as follows:

❶ ***Perspective Shift.*** We systematically revisit two research lines of LoRA: *the low-rank bottleneck* and *linear LoRA variants*. By interpreting LoRA through the lens of linear mappings, we address both research questions within a unified framework, providing theoretical analyses.

❷ ***Nonlinear Proposal.*** We propose `AuroRA`, which introduces nonlinear mappings into LoRA and further compresses the rank, resulting in a superior balance between parameters and performance, paving the way for further unlocking the significant potential of LoRA.

❸ ***Experimental Validation.*** Extensive experiments on 22 datasets and 6 pretrained models showcase that `AuroRA`: **(I)** not only matches or surpasses full fine-tuning performance with only $6.18\% \sim 25\%$ of LoRA's parameters but also **(II)** outperforms competitive PEFT methods by up to 10.88% in NLP and CV tasks, and **(III)** exhibits robust performance across various rank configurations.

https://doi.org/10.52202/085713-1241

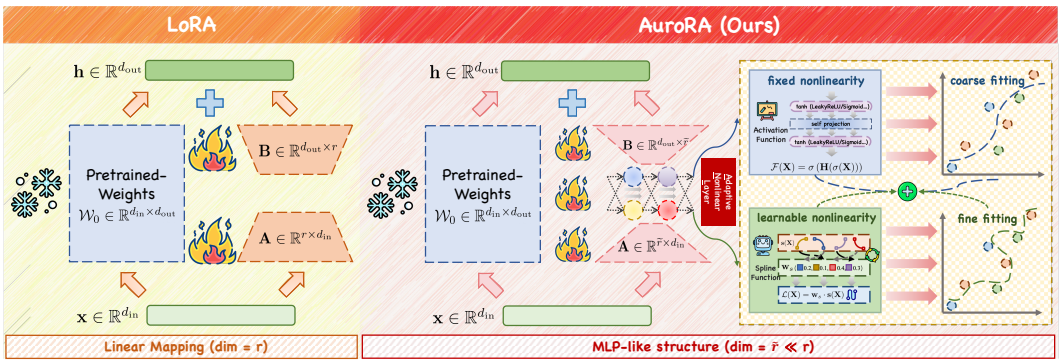

Figure 3: A general comparison of LoRA and our `AuroRA`. (***Left***) In LoRA, matrices $\mathbf{A}$ and $\mathbf{B}$ act as two linear projectors, forming a two-layer linear mapping with hidden dimension r. (***Right***) Our `AuroRA` extends LoRA by incorporating an adaptive nonlinear layer comprising fixed and learnable nonlinearities, forming an MLP-like structure with significantly reduced hidden dimension $\widetilde{r}$ ($\widetilde{r} \ll r$).

2 Methodology

As illustrated in Figure 3, we introduce `AuroRA`, an extension of LoRA that incorporates nonlinear mappings to overcome the inherent low-rank bottleneck. We reinterpret LoRA as a **two-layer linear mapping**, whereas our proposed `AuroRA` transforms it into an **MLP-like structure** by introducing an adaptive nonlinear layer.

2.1 LoRA: A Two-Layer Linear Mapping

In standard LoRA [16], the weight update $\Delta\mathcal{W}$ for a pre-trained weight matrix $\mathcal{W}_0$ is approximated as the product of two low-rank matrices:

$$\Delta\mathcal{W} = \mathbf{BA}, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{r\times d_{\text{in}}}$ and $\mathbf{B} \in \mathbb{R}^{d_{\text{out}}\times r}$, with the rank r satisfying $r \ll \min(d_{\text{in}}, d_{\text{out}})$. The forward propagation for an input vector $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$ is thus expressed as:

$$\mathbf{h} = \mathcal{W}_0\mathbf{x} + \Delta\mathcal{W}\mathbf{x} = \mathcal{W}_0\mathbf{x} + \mathbf{BAx}. \tag{2}$$

The above process can be interpreted as a two-layer linear mapping, where $\mathbf{A}$ serves as a downward projector $\mathcal{P}_{\text{down}}$ that maps the input $\mathbf{x}$ from a high-dimensional space $\mathbb{R}^{d_{\text{in}}}$ to a lower-dimensional hidden space $\mathbb{R}^{r}$, and $\mathbf{B}$ serves as an upward projector $\mathcal{P}_{\text{up}}$ that maps back to $\mathbb{R}^{d_{\text{out}}}$. However, we note that LoRA is constrained by its sequential linear mapping structure, leading to two significant shortcomings: ❶ **insufficient expressiveness**: being a purely linear structure, it requires increasing the hidden dimension to handle more complex incremental weights and improve performance; ❷ **limited training flexibility**: the direct low-rank decomposition induces strong interdependencies between the linear layers, imposing rigid structural constraints that reduce training flexibility [30].

2.2 `AuroRA`: An MLP-like Structure

To address these limitations, `AuroRA` introduces an Adaptive Nonlinear Layer (ANL) between $\mathbf{A}$ and $\mathbf{B}$, modifying the weight update as follows:

$$\Delta\mathcal{W} = \mathbf{B}\cdot\sigma(\mathbf{A}), \tag{3}$$

where σ is the element-wise ANL that maps from $\mathbb{R}^{\widetilde{r}}$ to $\mathbb{R}^{\widetilde{r}}$. Here, $\widetilde{r}$ denotes the compressed hidden dimension ($\widetilde{r} \ll r$), i.e., the low dimension to which the input is projected by $\mathcal{P}_{\text{down}}$. Formally, the forward propagation equation in the training phase is given by:

$$\mathbf{h} = \mathcal{W}_0\mathbf{x} + \mathbf{B}\cdot\sigma(\mathbf{Ax}). \tag{4}$$

The introduction of ANL enables `AuroRA` to form an MLP (Multilayer Perceptron)-like structure.

2.3 Adaptive Nonlinear Layer

Consider an arbitrary input vector $\mathbf{z}$. After projecting $\mathbf{z}$ into an $\widetilde{r}$-dimensional hidden layer, our objective is to introduce sufficient nonlinearity to capture as many complex relationships as possible

https://doi.org/10.52202/085713-1241

within this limited hidden space. To achieve this, we propose the following components: ❶ ***fixed nonlinearity*** ($\mathcal{F}$), which utilizes parameter-free nonlinear activation functions to activate neurons in the hidden space, thereby achieving *coarse fitting*; and ❷ ***learnable nonlinearity*** ($\mathcal{L}$), which employs parameterized nonlinear functions during the training process of the weight update increments, facilitating *fine fitting*. By combining ❶ and ❷, the Adaptive Nonlinear Layer (ANL) can be formally expressed as:

$$\sigma(\mathbf{Z}) = \mathcal{F}(\mathbf{Z}) + \mathcal{L}(\mathbf{Z}), \tag{5}$$

where $\mathcal{F}$ represents the fixed nonlinear activation, $\mathcal{L}$ denotes the learnable nonlinear function, and $\mathbf{Z}$ denotes the input to ANL. We provide a detailed comparison of fixed and learnable nonlinearity in Section 3.4.

For ❶, we adopt widely used activation functions in deep learning, such as ReLU [31], sigmoid, and tanh. A detailed comparison of different activation functions and their impact on **AuroRA**'s performance is provided in Section 3.4. Through our comparative evaluations, $\tanh$ emerges as the top-performing activation function, and theoretical analysis concurrently ensures its training stability. This preference is consistent with empirical findings in prior studies [32, 33] that demonstrate the robust performance of the $\tanh$ activation function for large-scale models, leading us to employ $\tanh$ in our implementation. The depth of the network influences the number of activation functions that can be introduced. Specifically, we introduce a *self-projection* $\mathcal{P}_{\text{self}} \in \mathbb{R}^{\widetilde{r} \times \widetilde{r}}$ between $\mathcal{P}_{\text{down}}$ and $\mathcal{P}_{\text{up}}$, which extends the depth of the standard LoRA structure. Subsequently, we introduce $\tanh$ activation functions between $\mathcal{P}_{\text{down}}$ and $\mathcal{P}_{\text{self}}$, and between $\mathcal{P}_{\text{self}}$ and $\mathcal{P}_{\text{up}}$. Formally, the fixed nonlinear component is defined as:

$$\mathcal{F}(\mathbf{Z}) = \tanh\left(\mathbf{H}\left(\tanh(\mathbf{Z})\right)\right), \tag{6}$$

where $\mathbf{H} \in \mathbb{R}^{\widetilde{r} \times \widetilde{r}}$ denotes $\mathcal{P}_{\text{self}}$.

To achieve ❷, we propose using spline functions to model complex relationships [34]. Numerous prior studies [35, 36, 37, 38] have demonstrated that splines are flexible, piecewise polynomial functions capable of approximating a wide range of nonlinear behaviors. Specifically, we employ B-spline basis functions to construct the learnable component. Formally, the learnable nonlinear component is defined as:

$$\mathcal{L}(\mathbf{Z}) = \mathbf{w}_s \cdot \mathbf{s}(\mathbf{Z}), \tag{7}$$

where $\mathbf{w}_s \in \mathbb{R}^{\widetilde{r}}$ is the spline weight vector, and $\mathbf{s}(\mathbf{Z}) = \sum_{i=1}^{\widetilde{r}} B(z_i)$ represents the spline basis functions applied to each dimension z_i of $\mathbf{Z}$. The learnable parameters in this component are the spline weights $\mathbf{w}_s$, which determine the contribution of each basis function $B(z_i)$ to the overall output of $\mathcal{L}(\mathbf{Z})$. During training, these weights are iteratively updated to minimize the task-specific loss function.

By introducing ❶ and ❷ in the hidden layer with dimension $\widetilde{r}$, ANL effectively captures complex relationships without significantly increasing the number of additional parameters. The combination of *coarse fitting* and *fine fitting* enhances the standard LoRA structure, improving its expressive capacity and training flexibility, achieving what we refer to as **`Activate Your Low-Rank Adaptation`**. The complete Adaptive Nonlinear Layer (ANL) developed in our work can then be formally represented as:

$$\sigma(\mathbf{Z}) = \tanh\left(\mathbf{H}\left(\tanh(\mathbf{Z})\right)\right) + \mathbf{w}_s \cdot \mathbf{s}(\mathbf{Z}). \tag{8}$$

Further details and the complete algorithmic workflow of **AuroRA** are provided in Appendix B.

2.4 Theoretical Analysis

In this subsection, we propose two theoretical propositions concerning **AuroRA** and analyze its parameter and computational cost. Additionally, we present an intuitive case in Appendix C to help better understand the role of nonlinearities.

Proposition 2.1 (Lower Approximation Error)**.** *Let* $M \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ *with* $\operatorname{rank}(M) > r$. *Define*

$$\varepsilon_r(M) \;=\; \inf_{U \in \mathbb{R}^{d_{\text{out}} \times r},\, V \in \mathbb{R}^{r \times d_{\text{in}}}} \| M - U\,V \|.$$

Then $\varepsilon_r(M) > 0$, *and for our proposed update of the form*

$$M_{\text{nonlinear}}(\mathbf{x}) \;=\; B\,\sigma\big(A\,\mathbf{x}\big), \quad A \in \mathbb{R}^{r \times d_{\text{in}}},\; B \in \mathbb{R}^{d_{\text{out}} \times r},$$

where σ *is our adaptive nonlinear layer, there exists a parameter set* (A^*, B^*, σ^*) *such that*

$$\left\| M - M_{\text{nonlinear}} \right\| \;\le\; c\,\varepsilon_r(M), \quad 0 < c < 1.$$

Hence, the approximation error is strictly below the linear rank-r limit $\varepsilon_r(M)$, using the same rank r.

► Proposition 2.1 indicates that, thanks to the introduction of nonlinear mappings, **AuroRA** achieves a strictly lower approximation error compared to LoRA at the same rank, meaning that the resulting weight updates are closer to the optimal solution. Furthermore, our empirical results demonstrate that this improvement persists even when further compressing the hidden dimensions of **AuroRA**. A rigorous proof, along with technical details and error bounds, is provided in Appendix D.

Proposition 2.2 (Gradient Boundedness)**.** *In the **AuroRA**, the use of the* $\tanh$ *activation function and B-spline basis functions results in bounded gradients with respect to both the inputs and the model parameters.*

► Proposition 2.2 posits that, despite the introduction of fixed and learnable nonlinearities, **AuroRA** maintains bounded gradients during training, thereby ensuring training stability. The corresponding proof is provided in Appendix E.

Parameter Cost In Section 1, we discussed the relationship between trainable parameters and rank in LoRA, where the number of introduced trainable parameters is $O(r(d_{\text{in}} + d_{\text{out}}))$. Here, d_{in} and d_{out} represent the input and output dimensions, respectively, i.e., PARAMS $\propto r$. In **AuroRA**, we aim to further compress the parameter count by setting the hidden layer dimension to $\widetilde{r} = r/k$, where k is a constant and r is the optimal rank setting of LoRA. This means that the number of trainable parameters in **AuroRA** is $1/k$ of that in LoRA. In this work, we set $\widetilde{r}$ to 2, corresponding to values of k such as 4 and 8. The additional parameters introduced in ANL are of the order $O(2\widetilde{r}^2)$, which, compared to the significant reduction in parameter count, can be considered negligible.

Computational Cost The computational complexity of **AuroRA**'s forward pass in the training phase, $\Delta\mathbf{h} = \mathbf{B}\sigma \cdot (\mathbf{A}\mathbf{x})$, is analyzed as follows. Let b denote the batch size, d_{in} and d_{out} the input/output feature dimensions, r the rank, and G the collective B-spline parameters (a small constant, $G = O(r)$). The linear projections by $\mathbf{A} \in \mathbb{R}^{r \times d_{in}}$ and $\mathbf{B} \in \mathbb{R}^{d_{out} \times r}$ incur complexities of $O(bd_{in}r)$ and $O(brd_{out})$, respectively. The intermediate fixed and learnable non-linearities, $\sigma(\cdot)$, each contribute an additional $O(br^2)$ term (with the learnable component's $O(brG)$ complexity simplifying due to $G = O(r)$). Consequently, the total complexity for **AuroRA** is $O(b(d_{in}r + 2r^2 + rd_{out}))$. Given the standard low-rank setting where $r \ll \min(d_{in}, d_{out})$, the quadratic overhead $O(br^2)$ introduced by the non-linearities is negligible compared to the dominant linear terms, thus maintaining a computational footprint comparable to that of LoRA.

3 Experiments

In this section, we conduct extensive experiments to answer the following research questions: ($\mathcal{RQ}1$) Can **AuroRA** effectively achieve efficiency in NLP tasks? ($\mathcal{RQ}2$) Can **AuroRA** effectively achieve efficiency in CV tasks? ($\mathcal{RQ}3$) What are the respective roles of fixed and learnable nonlinearity? ($\mathcal{RQ}4$) How do different activation functions in fixed nonlinearity affect performance? ($\mathcal{RQ}5$) How does **AuroRA**'s sensitivity to rank compare to that of linear LoRA variants? [1]

3.1 Experimental Setup

3.1.1 Datasets and Pre-Trained Models

Datasets For our experiments, we evaluate the ability of **AuroRA** to achieve parameter-efficient fine-tuning using four categories of datasets spanning both NLP and CV domains: ■ **Natural Language Understanding**: We employ GLUE (General Language Understanding Evaluation) [39], a widely used multi-task benchmark in NLU, which includes datasets such as SST-2, MRPC, CoLA, QNLI, RTE, and STS-B. The evaluation metrics are as follows: CoLA is assessed using Matthew's correlation coefficient, STS-B with Pearson's correlation coefficient, and accuracy is used for the other tasks. ■ **Commonsense Reasoning**: We use a collection of commonly used datasets, including BoolQ [40], PIQA [41], SocialIQA [42], HellaSwag [43], WinoGrande [44], ARC-e, ARC-c [45], and OpenBookQA [46]. For fair comparison, we follow the setup proposed by [28], fine-tuning the pretrained models on the Commonsense170K dataset, which serves as a mixture of the aforementioned benchmark datasets. We then evaluate using accuracy as the performance metric. ■ **Image Classification**: We use five datasets with small label spaces—OxfordPets [47], CIFAR-10 [48], DTD [49], EuroSAT [50], and RESISC45 [51], and three datasets with large label

[1] The source code is available at here.

https://doi.org/10.52202/085713-1241

Table 1: We report the performance of different fine-tuning methods on six datasets of the GLUE benchmark, using RoBERTa-Base and RoBERTa-Large models. For CoLA, we report the Matthew's Correlation Coefficient (MCC); for STS-B, we report the Pearson Correlation Coefficient (PCC); and for all other tasks, we report accuracy (Acc.). The reported results are the medians of five runs, each using a different random seed. * indicates numbers published in prior works. The best results are highlighted in **bold**, and the runners-up are underlined. For all six datasets, higher values are considered better for all metrics.

Model	Method	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	Avg.	Params.
RoBERTa-Base	Full Fine-Tuning*	94.8	90.2	63.6	92.8	78.7	91.2	85.2	125M
	BitFit*	93.7$_{\downarrow 1.1}$	**92.7**$_{\uparrow 2.5}$	62.0$_{\downarrow 1.6}$	91.8$_{\downarrow 1.0}$	81.5$_{\uparrow 2.8}$	90.8$_{\downarrow 0.4}$	85.4$_{\uparrow 0.2}$	0.1M
	Adapter^{D}*	94.7$_{\downarrow 0.1}$	88.4$_{\downarrow 1.8}$	62.6$_{\downarrow 1.0}$	93.0$_{\uparrow 0.2}$	75.9$_{\downarrow 2.8}$	90.3$_{\downarrow 0.9}$	84.2$_{\downarrow 1.0}$	0.9M
	LoRA*	95.1$_{\uparrow 0.3}$	89.7$_{\downarrow 0.5}$	63.4$_{\downarrow 0.2}$	93.3$_{\uparrow 0.5}$	78.4$_{\downarrow 0.3}$	**91.5**$_{\uparrow 0.3}$	85.2$_{\uparrow 0.0}$	0.3M
	AdaLoRA*	94.5$_{\downarrow 0.3}$	88.7$_{\downarrow 1.5}$	62.0$_{\downarrow 1.6}$	93.1$_{\uparrow 0.3}$	81.0$_{\uparrow 2.3}$	90.5$_{\downarrow 0.7}$	85.0$_{\downarrow 0.2}$	0.3M
	DyLoRA*	94.3$_{\downarrow 0.5}$	89.5$_{\downarrow 0.7}$	61.1$_{\downarrow 2.5}$	92.2$_{\downarrow 0.6}$	78.7$_{\uparrow 0.0}$	91.1$_{\downarrow 0.1}$	84.5$_{\downarrow 1.3}$	0.3M
	FourierFT*	94.2$_{\downarrow 0.6}$	90.0$_{\downarrow 0.2}$	63.8$_{\uparrow 0.2}$	92.2$_{\downarrow 0.6}$	79.1$_{\uparrow 0.4}$	90.8$_{\downarrow 0.4}$	85.0$_{\downarrow 0.2}$	0.024M
	LoRA-drop*	94.5$_{\downarrow 0.3}$	89.5$_{\downarrow 0.7}$	62.9$_{\downarrow 0.7}$	93.1$_{\uparrow 0.3}$	81.4$_{\uparrow 2.7}$	91.0$_{\downarrow 0.2}$	85.4$_{\uparrow 0.2}$	0.15M
	DoRA*	95.0$_{\uparrow 0.2}$	89.7$_{\uparrow 0.5}$	64.9$_{\uparrow 1.3}$	92.9$_{\uparrow 0.1}$	79.2$_{\uparrow 0.5}$	91.3$_{\uparrow 0.1}$	85.5$_{\uparrow 0.3}$	0.3M
	`AuroRA`	**95.2**$_{\uparrow 0.4}$	91.9$_{\uparrow 1.7}$	**65.1**$_{\uparrow 1.5}$	**93.4**$_{\uparrow 0.6}$	**85.2**$_{\uparrow 6.5}$	**91.5**$_{\uparrow 0.3}$	**87.1**$_{\uparrow 1.7}$	0.075M
RoBERTa-Large	Full Fine-Tuning*	96.4	90.9	68.0	94.7	86.6	92.4	88.2	356M
	Adapter^{P}*	96.1$_{\downarrow 0.3}$	90.2$_{\downarrow 0.7}$	68.3$_{\uparrow 0.3}$	94.8$_{\uparrow 0.1}$	83.8$_{\downarrow 2.8}$	92.1$_{\downarrow 0.3}$	87.6$_{\downarrow 0.6}$	3M
	Adapter^{H}*	96.2$_{\downarrow 0.2}$	88.7$_{\downarrow 2.2}$	66.5$_{\downarrow 1.5}$	94.7$_{\uparrow 0.0}$	83.4$_{\downarrow 3.2}$	91.0$_{\downarrow 1.2}$	86.8$_{\downarrow 1.4}$	6M
	LoRA*	96.2$_{\downarrow 0.2}$	90.2$_{\downarrow 0.7}$	68.2$_{\uparrow 0.2}$	94.8$_{\uparrow 0.1}$	85.2$_{\downarrow 1.4}$	92.3$_{\downarrow 0.1}$	87.8$_{\downarrow 0.4}$	0.8M
	FourierFT*	96.0$_{\downarrow 0.4}$	90.9$_{\uparrow 0.0}$	67.1$_{\downarrow 0.9}$	94.4$_{\downarrow 0.3}$	87.4$_{\uparrow 0.8}$	91.9$_{\downarrow 0.5}$	88.0$_{\downarrow 0.2}$	0.048M
	`AuroRA`	**96.6**$_{\uparrow 0.2}$	**91.2**$_{\uparrow 0.3}$	**69.2**$_{\uparrow 1.2}$	**95.0**$_{\uparrow 0.3}$	**89.9**$_{\uparrow 3.3}$	**92.5**$_{\uparrow 0.1}$	**89.1**$_{\uparrow 0.9}$	0.2M

Table 2: Commonsense reasoning evaluation results for LLaMA3-8B on eight tasks. * indicates numbers taken from [57]. The best results are highlighted in **bold**, and the runners-up are underlined. For all eight tasks, higher values are considered better.

Method	Params.	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
LoRA*	56.6M	70.8	85.2	**79.9**	91.7	84.3	84.2	71.2	79.0	80.8
PiSSA*	83.8M	67.1$_{\downarrow 3.7}$	81.1$_{\downarrow 4.1}$	77.2$_{\downarrow 2.7}$	83.6$_{\downarrow 8.1}$	78.9$_{\downarrow 5.4}$	77.7$_{\downarrow 6.5}$	63.2$_{\downarrow 8.0}$	74.6$_{\downarrow 5.4}$	75.4$_{\downarrow 5.4}$
MiLoRA*	56.6M	68.8$_{\downarrow 2.0}$	86.7$_{\uparrow 1.5}$	77.2$_{\downarrow 2.7}$	92.9$_{\uparrow 1.2}$	**85.6**$_{\uparrow 1.3}$	86.8$_{\uparrow 2.6}$	75.5$_{\uparrow 4.3}$	81.8$_{\uparrow 2.8}$	81.9$_{\uparrow 1.1}$
`AuroRA`	3.5M	**72.5**$_{\uparrow 1.7}$	**87.4**$_{\uparrow 2.2}$	79.0$_{\downarrow 0.9}$	**94.2**$_{\uparrow 2.5}$	83.0$_{\downarrow 1.3}$	**89.3**$_{\uparrow 5.1}$	**78.8**$_{\uparrow 7.6}$	**84.8**$_{\uparrow 5.8}$	**83.6**$_{\uparrow 2.8}$

spaces, namely StanfordCars [52], FGVC [53], and CIFAR-100 [48]. ■ **Subject-Driven Generation**: Following [54], we use the DreamBooth dataset. More detailed descriptions of the datasets can be found in Appendix F.1, F.2, F.3.

Pre-Trained Models We focus on a selection of representative pretrained models, including RoBERTa (Base & Large) [1], LLAMA3-8B [3], ViT (Base & Large) [55] and SDXL [56].

3.1.2 Baselines

In the baseline evaluation, we adopt a range of representative and competitive fine-tuning methods, categorized into three groups: Full Fine-Tuning, PEFT methods, and LoRA variants. The PEFT methods we use include BitFit [20], Adapter^{H} [32], Adapter^{D} [58], Adapter^{P} [59], and LoRA [16]. For the LoRA variants, we consider AdaLoRA [26], DyLoRA [60], FourierFT [61], LoRA-drop [62], DoRA [63], MoSLoRA [28], PiSSA [64] and MiLoRA [57].

3.2 `AuroRA` Achieves Efficiency in NLP Tasks ($\mathcal{RQ}1$)

To answer $\mathcal{RQ}1$, we design two tasks: Natural Language Understanding (NLU) and Commonsense Reasoning. In the NLU task, we select RoBERTa-Base and RoBERTa-Large [1] as pretrained models and compare `AuroRA` with **ten** other widely-used fine-tuning methods across all six datasets of the GLUE benchmark [39]. The results of this extensive comparison are shown in Table 1, with additional hyperparameter configuration details provided in Appendix G.1. Following [16], we fine-tune only the query and value weights of each transformer block, while fully fine-tuning the classification head. For the commonsense reasoning task, we select LLaMA3-8B [3] as the base model and compare `AuroRA` with LoRA and two other LoRA variants (PiSSA [64] and MiLoRA [57]). The results are shown in Table 2. The relevant hyperparameters are listed in Appendix G.2. Our observations can be summarized as follows:

Table 3: Fine-tuning results with ViT Base and Large models on different image classification datasets. We report the accuracy (%) after 10 epochs. Avg. represents the average accuracy across all datasets for each method. * indicates numbers taken from [61]. The best results are highlighted in **bold**, and the runners-up are underlined (excluding full fine-tuning).

Model	Method	Params.	OxfordPets	StanfordCars	CIFAR10	DTD	EuroSAT	FGVC	RESISC45	CIFAR100	Avg.
ViT-Base	Full Fine-Tuning*	85.8M	93.1	79.8	98.9	77.7	99.1	54.8	96.1	92.4	86.5
	Linear Probing*	-	90.3$_{\downarrow 2.8}$	25.8$_{\downarrow 54.0}$	96.4$_{\downarrow 2.5}$	69.8$_{\downarrow 7.9}$	88.7$_{\downarrow 10.4}$	17.4$_{\downarrow 37.4}$	74.2$_{\downarrow 21.9}$	84.3$_{\downarrow 8.1}$	68.4$_{\downarrow 18.1}$
	LoRA*	581K	93.2$_{\uparrow 0.1}$	45.4$_{\downarrow 34.4}$	**98.8**$_{\downarrow 0.1}$	75.0$_{\downarrow 2.7}$	98.4$_{\downarrow 0.7}$	25.2$_{\downarrow 29.6}$	92.7$_{\downarrow 3.4}$	**92.0**$_{\downarrow 0.4}$	77.6$_{\downarrow 8.9}$
	FourierFT*	239K	93.1$_{\uparrow 0.0}$	56.4$_{\downarrow 23.4}$	98.7$_{\downarrow 0.2}$	77.3$_{\downarrow 0.4}$	**98.8**$_{\downarrow 0.3}$	32.4$_{\downarrow 22.4}$	**94.3**$_{\downarrow 1.8}$	91.5$_{\downarrow 0.9}$	80.3$_{\downarrow 6.2}$
	AuroRA	74K	**93.9**$_{\uparrow 0.8}$	**75.7**$_{\downarrow 4.1}$	**98.8**$_{\downarrow 0.1}$	**79.6**$_{\uparrow 1.9}$	**98.8**$_{\downarrow 0.3}$	**48.2**$_{\downarrow 6.6}$	93.6$_{\downarrow 2.5}$	**92.0**$_{\downarrow 0.4}$	**85.1**$_{\downarrow 1.4}$
ViT-Large	Full Fine-Tuning*	303.3M	94.4	88.9	99.2	81.8	99.0	68.3	96.4	93.6	90.2
	Linear Probing*	-	91.1$_{\downarrow 3.3}$	37.9$_{\downarrow 51.0}$	97.8$_{\downarrow 1.4}$	73.3$_{\downarrow 8.5}$	92.6$_{\downarrow 6.4}$	24.6$_{\downarrow 43.7}$	82.0$_{\downarrow 14.4}$	84.3$_{\downarrow 9.3}$	73.0$_{\downarrow 17.2}$
	LoRA*	1.57M	94.8$_{\uparrow 0.4}$	73.3$_{\downarrow 15.6}$	**99.1**$_{\downarrow 0.1}$	81.8$_{\uparrow 0.0}$	98.6$_{\downarrow 0.4}$	42.3$_{\downarrow 26.0}$	94.7$_{\downarrow 1.7}$	**94.9**$_{\uparrow 1.3}$	84.9$_{\downarrow 5.3}$
	FourierFT*	480K	94.8$_{\uparrow 0.4}$	79.1$_{\downarrow 9.8}$	**99.1**$_{\downarrow 0.1}$	81.9$_{\uparrow 0.1}$	98.7$_{\downarrow 0.3}$	51.3$_{\downarrow 17.0}$	**95.2**$_{\downarrow 1.2}$	93.4$_{\downarrow 0.2}$	86.7$_{\downarrow 3.5}$
	AuroRA	197K	**94.9**$_{\uparrow 0.5}$	**82.5**$_{\downarrow 6.4}$	**99.1**$_{\downarrow 0.1}$	**82.1**$_{\uparrow 0.3}$	**98.9**$_{\downarrow 0.1}$	**59.8**$_{\downarrow 8.5}$	94.9$_{\downarrow 1.5}$	93.3$_{\downarrow 0.3}$	**88.2**$_{\downarrow 2.0}$

Obs. ❶ AuroRA demonstrates strong efficiency in NLP tasks. It is evident that AuroRA outperforms the baseline across all datasets and pretrained models in both tasks. Compared to Full Fine-Tuning, AuroRA achieves a performance improvement ranging from $0.1\% \sim 8.3\%$ while using only $0.04\% \sim 0.06\%$ of the total parameters. Compared to PEFT baselines, including LoRA, AuroRA achieves a performance improvement of up to 24.7% and an average improvement of $1.25\% \sim 10.88\%$, using only $6.25\% \sim 25\%$ of the parameters. Specifically, in the commonsense reasoning task using LLaMA3-8B as the pretrained model, AuroRA achieves a significant 10.7% performance boost on ARC-C with just 6.25% of LoRA's parameter budget. In the NLU task, although AuroRA uses more parameters than FourierFT, it demonstrates significant performance gains across all pretrained models and datasets. For instance, using RoBERTa-Base, AuroRA improves performance by 7.7% on RTE.

Obs. ❷ AuroRA can be scaled up to fine-tune large pretrained models. AuroRA scales effectively to fine-tuning larger pretrained models. In the NLU task, when the pretrained model changes to RoBERTa-Large from Base, nearly all PEFT methods show a performance drop compared to Full Fine-Tuning, with the largest decrease reaching 3.7%. In contrast, AuroRA still achieves performance improvements of $0.1\% \sim 3.8\%$ across all datasets. In the commonsense reasoning task, when the model size increases to 8B, AuroRA continues to outperform LoRA by $2.4\% \sim 10.7\%$.

3.3 AuroRA Achieves Efficiency in CV Tasks ($\mathcal{RQ}2$)

To answer $\mathcal{RQ}2$, we design two tasks: Image Classification and Subject-Driven Image Generation. In the image classification task, following [61], we select ViT-Base and ViT-Large [55], two popular CV foundation models, which are pretrained on the ImageNet-21K [65] dataset. We then compare AuroRA with Full Fine-Tuning, Linear Probing (fine-tuning only the classification head), LoRA, and FourierFT. The results are presented in Table 3, with more implementation details available in Appendix G.3. In the subject-driven image generation task [54], following [61] and [28], we use the SDXL model [56] as our backbone, and then fine-tune it using both LoRA and AuroRA. The objective is to generate images based on specified prompts for a particular subject, which is defined using a set of reference images. Initially, we fine-tune a text-to-image model by pairing the input images with text prompts that include a unique identifier (e.g., "A photo of a [V] dog"). Subsequently, the model can generate images corresponding to other prompts that incorporate the same unique identifier, thereby producing images of the defined subject. The results are presented in Figure 4, and more generated cases are in Appendix H. Our observations can be summarized as follows:

Obs. ❸ AuroRA achieves the best performance, excluding Full Fine-Tuning, with the least number of parameters. It is evident that AuroRA outperforms all other PEFT baseline methods across all eight datasets with the lowest parameter count (12.7% of LoRA and 31.0% of FourierFT) when using both the Base and Large models. Compared to Full Fine-Tuning, AuroRA uses only 0.086% of the parameters and achieves a performance improvement of $0.4\% \sim 2.4\%$ on some datasets, with only a $1.6\% \sim 2.2\%$ gap in average performance. When using ViT-Base on STANFORDCARS, other baselines show a significant performance drop of $29.3\% \sim 67.7\%$ compared to Full Fine-Tuning. In contrast, AuroRA only experiences a moderate drop of 5.1%. Compared to PEFT methods, AuroRA achieves an average performance improvement of $1.73\% \sim 9.66\%$.

Obs. ❹ AuroRA demonstrates stronger adaptability in the text-to-image domain. We observe that in the subject-driven image generation task, AuroRA aligns better with the environment specified in the prompt. Specifically, when given the prompt "A [V] bear plushie on top of green grass with

https://doi.org/10.52202/085713-1241

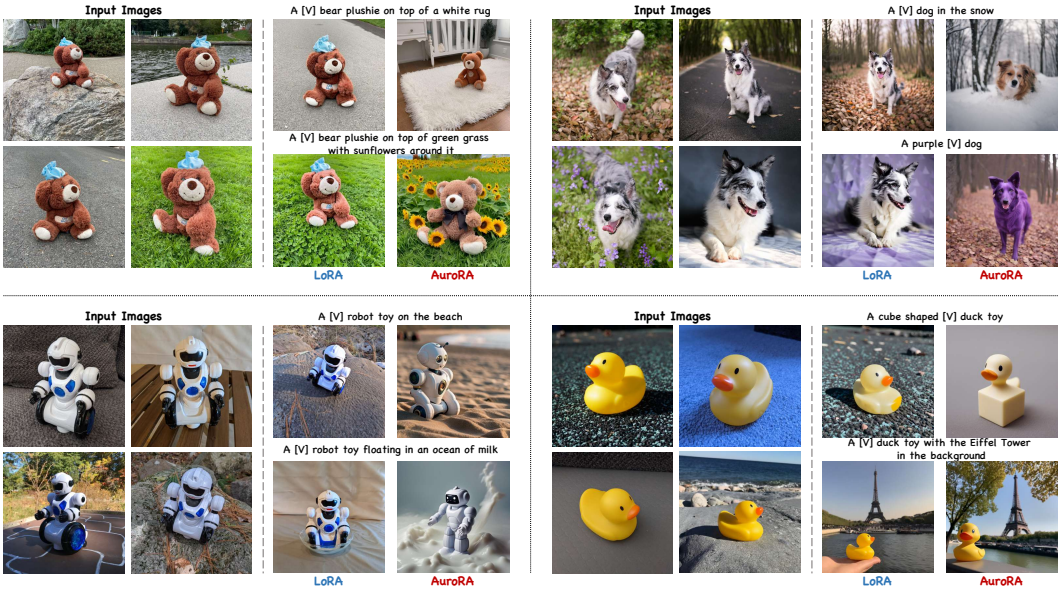

Figure 4: Results of LoRA and **AuroRA** in the subject-driven image generation task. **AuroRA** aligns better with the prompt.

sunflowers around it", LoRA generates an environment with only green grass but no sunflowers. In contrast, **AuroRA** successfully generates green grass with sunflowers.

3.4 Study

Ablation Study ($\mathcal{RQ}3$) To evaluate the contribution of different modules in **AuroRA**, we introduce two variants: (1) **AuroRA** w/o $\mathcal{F}$, and (2) **AuroRA** w/o $\mathcal{L}$, which correspond to the removal of the fixed and learnable nonlinearity in **AuroRA**, respectively. We compare these two variants with **AuroRA** by fine-tuning ViT-Base on OXFORDPETS, CIFAR10, DTD, and EUROSAT in the image classification task. From Table 4, we observe that: ❶ removing any component results in a performance drop for **AuroRA**; ❷ **AuroRA** w/o $\mathcal{L}$ consistently underperforms across all datasets, indicating that the learnable nonlinearity plays a more crucial role in the success of our method. Specifically, the learnable nonlinearity enables fine fitting, while the fixed nonlinearity contributes to coarse fitting.

Table 4: Comparison of different settings.

Setting	OxfordPets	CIFAR10	DTD	EuroSAT
AuroRA	93.9	98.8	79.6	98.8
AuroRA w/o $\mathcal{F}$	93.3 $_{\downarrow 0.6}$	98.4 $_{\downarrow 0.4}$	78.9 $_{\downarrow 0.7}$	98.3 $_{\downarrow 0.5}$
AuroRA w/o $\mathcal{L}$	93.1 $_{\downarrow 0.8}$	98.2 $_{\downarrow 0.6}$	77.8 $_{\downarrow 1.8}$	98.0 $_{\downarrow 0.8}$

Table 5: Comparison of different activation functions.

Setting	StanfordCars	FGVC	RESISC45	CIFAR100
AuroRA	75.7	48.2	93.6	92.0
AuroRA-lr	75.6 $_{\downarrow 0.1}$	47.8 $_{\downarrow 0.4}$	93.4 $_{\downarrow 0.2}$	91.9 $_{\downarrow 0.1}$
AuroRA-sm	75.2 $_{\downarrow 0.5}$	47.7 $_{\downarrow 0.5}$	92.9 $_{\downarrow 0.7}$	91.7 $_{\downarrow 0.3}$

Effect of Activation Function ($\mathcal{RQ}4$) We investigate the impact of the choice of activation function in the fixed nonlinearity on **AuroRA**'s performance. Specifically, we introduce two variants: (1) **AuroRA**-lr, and (2) **AuroRA**-sm, which correspond to replacing the activation function in the fixed nonlinearity ($\tanh$) with LeakyReLU and Sigmoid, respectively. We compare these variants with **AuroRA** by fine-tuning the ViT-Base model on STANFORDCARS, FGVC, RESISC45, and CIFAR100 in the image classification task. From Table 5, we observe that Sigmoid results in the lowest performance, while $\tanh$ achieves the highest performance. Therefore, we choose $\tanh$ as the activation function for fixed nonlinearity in all our experiments.

Sensitivity to Rank & Comparison with Linear LoRA Variants ($\mathcal{RQ}5$) To further investigate the impact of introducing nonlinearity, we examine its sensitivity to rank and compare it with the linear LoRA variant under identical experimental settings. Specifically, we select LLaMA3-8B as the pretrained model and fine-tune it using **AuroRA**, MoSLoRA, and LoRA, varying the rank among $\{2, 4, 8, 16\}$. We evaluate their performance across four datasets. From Figure 5, we observe the following: ❶ the introduction of nonlinearity results in smaller performance fluctuations as the rank varies, i.e., more robustness to rank; ❷ **AuroRA** consistently outperforms across almost all rank settings and datasets, indicating that incorporating nonlinearity further enhances the model's expressiveness compared to linear approaches.

https://doi.org/10.52202/085713-1241

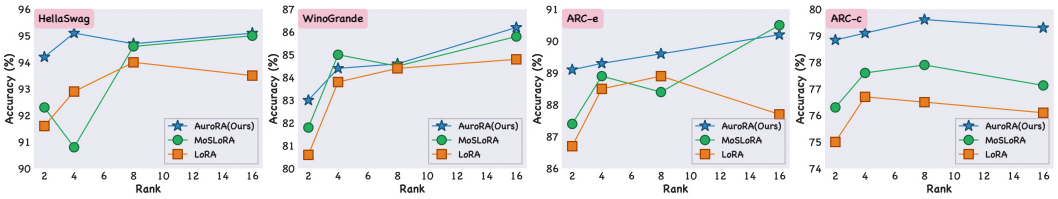

Figure 5: Performance comparison of different methods with varying ranks. We use LLaMA 3-8B as the pretrained model and fine-tune it using `AuroRA`, MoSLoRA, and LoRA methods on the HELLASWAG, WINOGRANDE, ARC-E and ARC-C datasets, with ranks $\{2, 4, 8, 16\}$.

4 Related Work

4.1 Parameter-Efficient Fine-Tuning

Parameter-Efficient Fine-Tuning (PEFT) has emerged as a pivotal strategy for addressing the computational challenges associated with fine-tuning large-scale pretrained models. PEFT methodologies can be broadly categorized into: ❶ **Additive PEFT** approaches introduce new, trainable modules to a frozen base model [18, 66, 32, 13, 67, 68, 15]. Common strategies include adapter-based techniques, such as AdapterFusion [59] and Hyperformer [69]; prompt-based methods, like Prefix-tuning [17] and p-tuning v2 [70]. ❷ **Selective PEFT** methods optimize a chosen subset of a pretrained model's parameters while keeping the majority frozen [71, 72, 73, 74, 75, 76]. This selection is often achieved through unstructured masking based on criteria like parameter significance, as seen in FishMask [77] and Child-tuning [78], or via structured techniques that group parameters, such as Bitfit [20] and SPT [79]. ❸ **Reparameterized PEFT** techniques transform model weights into more efficient, often low-rank, representations during fine-tuning, without altering the core architecture for inference [80, 60, 81, 82, 83]. A prominent example is LoRA [16], which introduces low-rank matrices for updates. ❹ **Memory-Efficient PEFT** methods focus on reducing the memory footprint of fine-tuning by optimizing the training dynamics rather than the model architecture [84, 85, 86, 87]. A representative example is GaLore [87], which projects gradients into low-rank subspaces to lower optimizer-state memory while preserving full-parameter adaptability. ❺ **Hybrid PEFT** methods integrate multiple strategies from different PEFT categories to capitalize on their respective advantages [19, 88, 89]. For instance, NOAH [90] and AUTOPEFT [91], leverage neural architecture search to identify effective PEFT combinations for specific tasks. In this paper, we primarily focus on LoRA, a reparameterized PEFT method.

4.2 LoRA and its Variants

The core idea of LoRA [16] is to approximate weight updates using mergeable, low-rank matrix pathways. Its variants can be broadly categorized into several types: ❶ **Novel Branch Designs** primarily focus on remodeling or reformulating the original low-rank matrix approximation pathway, with notable examples including VeRA [80], FourierFT [61], PiSSA [64], and DoRA [92]. ❷ **Multi-Task Variants**, exemplified by MoELoRA [81], MoA [82], CA-LoRA [93], and HydraLoRA [94], are engineered to enhance cross-task generalization—particularly in scenarios such as multi-task learning, domain adaptation, and continual learning—often through the strategic employment of LoRA module mixtures or ensembles. ❸ **Linear Variants**, including AdaLoRA [26], SaLoRA [27], MoSLoRA [28], and FLoRA [29], typically augment the LoRA framework by incorporating an additional linear matrix between the two original low-rank factors, thereby bolstering information capture during the training phase. Beyond these, several nonlinear LoRA variants have recently emerged, including LoRAN [95], SineLoRA [96], LoDA [97], NEAT [98], and CoLA [99]. However, these recent nonlinear variants do not resolve inherent low-rank bottleneck in LoRA. In contrast, our method pairs nonlinearities with a focus on LoRA's fundamental structural limitations, achieving a superior balance between performance and parameter efficiency.

5 Conclusion

In this paper, we revisit LoRA from the perspective of linear mappings and introduce nonlinearity into LoRA by proposing `AuroRA`, an MLP-like structure. `AuroRA` incorporates an adaptive nonlinear layer that includes both fixed and learnable nonlinearities between the two low-rank matrices. `AuroRA` achieves a superior balance between performance and parameters across tasks in both the NLP and CV domains. We hope that `AuroRA` will inspire further exploration of nonlinear extensions to LoRA.

https://doi.org/10.52202/085713-1241

Limitation A potential limitation is that, due to limited computational resources, we do not evaluate performance on larger pretrained models in this study, leaving this exploration for future work.

Broader Impact As a novel nonlinear method, `AuroRA` is envisioned for broad future applications in key sectors such as healthcare and finance. It is anticipated to deliver more accurate and reliable services while significantly reducing resource consumption, thereby better serving human society.

Acknowledgments and Disclosure of Funding

This work is supported by the State Key Laboratory of General Artificial Intelligence; and the National Natural Science Foundation of China (Grant No. 62276006).

References

[1] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364, 2019.

[2] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*, 2021.

[3] AI@Meta. Llama 3 model card. 2024.

[4] Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

[5] Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*, 2024.

[6] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[7] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[8] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[10] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[11] Haonan Dong, Haoran Ye, Wenhao Zhu, Kehan Jiang, and Guojie Song. Meta-r1: Empowering large reasoning models with metacognition, 2025.

[12] Haoran Ye, Yuhang Xie, Yuanyi Ren, Hanjun Fang, Xin Zhang, and Guojie Song. Measuring human and ai values based on generative psychometrics with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 2025.

[13] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[14] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[15] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *AI Open*, 5:208–215, 2024.

[16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

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

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

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

[20] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1–9. Association for Computational Linguistics, 2022.

[21] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.

[22] Simeng Sun, Dhawal Gupta, and Mohit Iyyer. Exploring the impact of low-rank adaptation on the performance, efficiency, and regularization of rlhf. *arXiv preprint arXiv:2309.09055*, 2023.

[23] Zihan Zhong, Zhiqiang Tang, Tong He, Haoyang Fang, and Chun Yuan. Convolution meets loRA: Parameter efficient finetuning for segment anything model. In *The Twelfth International Conference on Learning Representations*, 2024.

[24] Wei Dong, Xing Zhang, Bihui Chen, Dawei Yan, Zhijun Lin, Qingsen Yan, Peng Wang, and Yang Yang. Low-rank rescaled vision transformer fine-tuning: A residual design approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16101–16110, 2024.

[25] Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John Patrick Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024. Featured Certification.

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

[27] Yahao Hu, Yifei Xie, Tianfeng Wang, Man Chen, and Zhisong Pan. Structure-aware low-rank adaptation for parameter-efficient fine-tuning. *Mathematics*, 11(20):4317, 2023.

[28] Taiqiang Wu, Jiahao Wang, Zhe Zhao, and Ngai Wong. Mixture-of-subspaces in low-rank adaptation. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7880–7899, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

 https://doi.org/10.52202/085713-1241

[29] Chongjie Si, Xuehui Wang, Xue Yang, Zhengqin Xu, Qingyun Li, Jifeng Dai, Yu Qiao, Xiaokang Yang, and Wei Shen. Flora: Low-rank core space for n-dimension. *arXiv preprint arXiv:2405.14739*, 2024.

[30] Chongjie Si, Xiaokang Yang, and Wei Shen. See further for parameter efficient fine-tuning by standing on the shoulders of decomposition. *arXiv preprint arXiv:2407.05417*, 2024.

[31] AF Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

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

[33] Jiachen Zhu, Xinlei Chen, Kaiming He, Yann LeCun, and Zhuang Liu. Transformers without normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[34] Carl de Boor. A practical guide to splines. In *Applied Mathematical Sciences*, 1978.

[35] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Y. Hou, and Max Tegmark. KAN: Kolmogorov–arnold networks. In *The Thirteenth International Conference on Learning Representations*, 2025.

[36] Shayan Aziznejad and Michael Unser. Deep spline networks with control of lipschitz regularity. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3242–3246. IEEE, 2019.

[37] Daniele Fakhoury, Emanuele Fakhoury, and Hendrik Speleers. Exsplinet: An interpretable and expressive spline-based neural network. *Neural Networks*, 152:332–346, 2022.

[38] Pakshal Bohra, Joaquim Campos, Harshit Gupta, Shayan Aziznejad, and Michael Unser. Learning activation functions in deep (spline) neural networks. *IEEE Open Journal of Signal Processing*, 1:295–309, 2020.

[39] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[40] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *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, June 2019. Association for Computational Linguistics.

[41] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

[42] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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, November 2019. Association for Computational Linguistics.

[43] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics.

[44] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press, 2020.

[45] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

[46] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[47] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[48] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[49] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[50] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[51] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

[52] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[53] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[54] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 22500–22510. IEEE, 2023.

[55] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[56] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[57] Hanqing Wang, Yixia Li, Shuo Wang, Guanhua Chen, and Yun Chen. MiLoRA: Harnessing minor singular components for parameter-efficient LLM finetuning. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language*

 https://doi.org/10.52202/085713-1241

Technologies (Volume 1: Long Papers), pages 4823–4836, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics.

[58] Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. AdapterDrop: On the efficiency of adapters in transformers. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[59] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online, April 2021. Association for Computational Linguistics.

[60] Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. DyLoRA: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3274–3287, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

[61] Ziqi Gao, Qichao Wang, Aochuan Chen, Zijing Liu, Bingzhe Wu, Liang Chen, and Jia Li. Parameter-efficient fine-tuning with discrete fourier transform. In *Forty-first International Conference on Machine Learning*, 2024.

[62] Hongyun Zhou, Xiangyu Lu, Wang Xu, Conghui Zhu, Tiejun Zhao, and Muyun Yang. LoRA-drop: Efficient LoRA parameter pruning based on output evaluation. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5530–5543, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics.

[63] Yulong Mao, Kaiyu Huang, Changhao Guan, Ganglin Bao, Fengran Mo, and Jinan Xu. DoRA: Enhancing parameter-efficient fine-tuning with dynamic rank distribution. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11662–11675, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[64] Fanxu Meng, Zhaohui Wang, and Muhan Zhang. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[65] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[66] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[67] Yaoming Zhu, Jiangtao Feng, Chengqi Zhao, Mingxuan Wang, and Lei Li. Counter-interference adapter for multilingual machine translation. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2812–2823, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[68] Tao Lei, Junwen Bai, Siddhartha Brahma, Joshua Ainslie, Kenton Lee, Yanqi Zhou, Nan Du, Vincent Y Zhao, Yuexin Wu, Bo Li, Yu Zhang, and Ming-Wei Chang. Conditional adapters: Parameter-efficient transfer learning with fast inference. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[69] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *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, August 2021. Association for Computational Linguistics.

[70] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *CoRR*, abs/2110.07602, 2021.

[71] Demi Guo, Alexander Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *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, August 2021. Association for Computational Linguistics.

[72] Neal Lawton, Anoop Kumar, Govind Thattai, Aram Galstyan, and Greg Ver Steeg. Neural architecture search for parameter-efficient fine-tuning of large pre-trained language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8506–8515, Toronto, Canada, July 2023. Association for Computational Linguistics.

[73] Baohao Liao, Yan Meng, and Christof Monz. Parameter-efficient fine-tuning without introducing new latency. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4242–4260, Toronto, Canada, July 2023. Association for Computational Linguistics.

[74] Sarkar Snigdha Sarathi Das, Ranran Haoran Zhang, Peng Shi, Wenpeng Yin, and Rui Zhang. Unified low-resource sequence labeling by sample-aware dynamic sparse finetuning. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6998–7010, Singapore, December 2023. Association for Computational Linguistics.

[75] Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. Composable sparse fine-tuning for cross-lingual transfer. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1778–1796, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[76] Tingfeng Hui, Zhenyu Zhang, Shuohuan Wang, Weiran Xu, Yu Sun, and Hua Wu. HFT: half fine-tuning for large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 12791–12819. Association for Computational Linguistics, 2025.

[77] Yi-Lin Sung, Varun Nair, and Colin Raffel. Training neural networks with fixed sparse masks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[78] Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9514–9528, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[79] Haoyu He, Jianfei Cai, Jing Zhang, Dacheng Tao, and Bohan Zhuang. Sensitivity-aware visual parameter-efficient fine-tuning. In *ICCV*, 2023.

 https://doi.org/10.52202/085713-1241

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

[81] Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. When moe meets llms: Parameter efficient fine-tuning for multi-task medical applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1104–1114, 2024.

[82] Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. Mixture-of-LoRAs: An efficient multitask tuning method for large language models. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 11371–11380, Torino, Italia, May 2024. ELRA and ICCL.

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

[84] Qijun Luo, Hengxu Yu, and Xiao Li. Badam: A memory efficient full parameter optimization method for large language models. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[85] Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. LISA: layerwise importance sampling for memory-efficient large language model fine-tuning. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[86] Pengxiang Li, Lu Yin, Xiaowei Gao, and Shiwei Liu. Outlier-weighed layerwise sampling for LLM fine-tuning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 19460–19473. Association for Computational Linguistics, 2025.

[87] Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient LLM training by gradient low-rank projection. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[88] Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Scott Yih, and Madian Khabsa. UniPELT: A unified framework for parameter-efficient language model tuning. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6253–6264, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[89] Jiaao Chen, Aston Zhang, Xingjian Shi, Mu Li, Alex Smola, and Diyi Yang. Parameter-efficient fine-tuning design spaces. In *The Eleventh International Conference on Learning Representations*, 2023.

[90] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–14, 2024.

[91] Han Zhou, Xingchen Wan, Ivan Vulić, and Anna Korhonen. AutoPEFT: Automatic configuration search for parameter-efficient fine-tuning. *Transactions of the Association for Computational Linguistics*, 12:525–542, 2024.

[92] Shih yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

[93] Weilin Zhao, Yuxiang Huang, Xu Han, Zhiyuan Liu, Zhengyan Zhang, Kuai Li, Chen Chen, TAO YANG, and Maosong Sun. CA-loRA: Adapting existing loRA for compressed LLMs to enable efficient multi-tasking on personal devices. In *First Conference on Language Modeling*, 2024.

[94] Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. Hydralora: An asymmetric lora architecture for efficient fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[95] Yinqiao Li, Linqi Song, and Hanxu Hou. LoRAN: Improved low-rank adaptation by a non-linear transformation. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3134–3143, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[96] Yiping Ji, Hemanth Saratchandran, Cameron Gordon, Zeyu Zhang, and Simon Lucey. Efficient learning with sine-activated low-rank matrices. In *The Thirteenth International Conference on Learning Representations*, 2025.

[97] Jing Liu, Toshiaki Koike-Akino, Pu Wang, Matthew Brand, Ye Wang, and Kieran Parsons. Loda: Low-dimensional adaptation of large language models. In *NeurIPS'23 Workshop on on Efficient Natural Language and Speech Processing*, 2023.

[98] Yibo Zhong, Haoxiang Jiang, Lincan Li, Ryumei Nakada, Tianci Liu, Linjun Zhang, Huaxiu Yao, and Haoyu Wang. Neat: Nonlinear parameter-efficient adaptation of pre-trained models, 2025.

[99] Ziyue Liu, Ruijie Zhang, Zhengyang Wang, Zi Yang, Paul D. Hovland, Bogdan Nicolae, Franck Cappello, and Zheng Zhang. Cola: Compute-efficient pre-training of llms via low-rank activation. *CoRR*, abs/2502.10940, 2025.

[100] Roger A Horn and Charles R Johnson. *Topics in matrix analysis*. Cambridge university press, 1994.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the

 https://doi.org/10.52202/085713-1241

supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In this paper, we introduce a novel non-linear parameter-efficient fine-tuning method and we claim the contributions and scope in the abstract and introduction sections (See Abstract and Introduction Section).

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In this work, we systematically discuss the limitations of our research and outline directions for future work (See Conclusion Section).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: In this work, we analyze our proposed method from a theoretical perspective and provide complete and detailed proofs (See Method Section and Appendix).

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the code necessary for replicating the studies described in this paper via an anonymous link, and we detail the experimental setup for the replication in the article itself (See Appendix).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

 https://doi.org/10.52202/085713-1241

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: For the datasets disclosed in the article, we have provided information regarding their sources and origins (See Appendix).

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have specified all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results (See Appendix).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Experimental results are tested multiple times to ensure stability and reliability (See Experiments).

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In this paper, we provide detailed information about the experimental resources, including GPU configurations used in our studies (See Appendix).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The study presented in this paper conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

https://doi.org/10.52202/085713-1241

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have provided the societal impacts of the work (See Conclusion).

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper does not address issues related to this aspect.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All creators and original owners of the assets used in our paper, such as code, data, and models, have been properly credited. We have explicitly mentioned the licenses and terms of use for each asset and have ensured full compliance with these terms throughout our research.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The research presented in this paper is not concerned with new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve experiments or research related to human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

https://doi.org/10.52202/085713-1241

Justification: This paper does not address potential risks incurred by study participants.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Notations

We summarize the notations used throughout the manuscript in Table 6.

Table 6: Notations commonly used in the **AuroRA** method.

Notation	Definition
$\mathcal{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$	Pretrained weight matrix
$\Delta\mathcal{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$	Weight update matrix
$\mathbf{A} \in \mathbb{R}^{\widetilde{r} \times d_{\text{in}}}$	Downward projector (low-rank matrix)
$\mathbf{B} \in \mathbb{R}^{d_{\text{out}} \times \widetilde{r}}$	Upward projector (low-rank matrix)
r	Original LoRA rank ($r \ll \min(d_{\text{in}}, d_{\text{out}})$)
$\widetilde{r}$	Compressed hidden dimension ($\widetilde{r} \ll r$)
$\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$	Input vector
$\mathbf{h} \in \mathbb{R}^{d_{\text{out}}}$	Output vector
$\sigma(\cdot)$	Adaptive Nonlinear Layer (ANL) mapping $\mathbb{R}^{\widetilde{r}} \to \mathbb{R}^{\widetilde{r}}$
$\mathcal{P}_{\text{down}}$	Projection onto $\widetilde{r}$-dim hidden space (matrix $\mathbf{A}$)
$\mathcal{P}_{\text{self}}$	Self-projection in hidden space (matrix $\mathbf{H} \in \mathbb{R}^{\widetilde{r} \times \widetilde{r}}$)
$\mathcal{P}_{\text{up}}$	Projection back to d_{out} (matrix $\mathbf{B}$)
$\mathcal{F}(\cdot)$	Fixed nonlinearity (e.g. $\tanh$)
$\mathcal{L}(\cdot)$	Learnable nonlinearity (B-spline based)
$\mathbf{w}_s \in \mathbb{R}^{\widetilde{r}}$	Spline weight vector in $\mathcal{L}$
$\mathbf{s}(\mathbf{Z})$	Spline basis functions applied to each component of $\mathbf{Z}$
T	Number of training epochs

B Complete Process

In this section, we first provide further details on both the static weight merging operation performed after the training phase and the actual process at inference time. We then offer empirical validation for our approach. Finally, the complete algorithm workflow of our **AuroRA** is presented in Algo. 1.

B.1 Weights Merging & Inference Phase

During the training phase, to enhance training flexibility, **AuroRA** utilizes a *dynamic*, input-dependent update mechanism formulated as $\mathbf{B} \cdot \sigma(\mathbf{A}\mathbf{x})$. After the training phase, once all learnable parameters are fixed and considered optimized, **AuroRA** transitions to a *static* form for the inference stage. This static form, given by $\Delta\mathcal{W} = \mathbf{B} \cdot \sigma(\mathbf{A})$, facilitates the seamless integration of **AuroRA** into the pre-trained weight $\mathcal{W}_0$, consistent with standard LoRA. Therefore, after the weights are merged, the effective forward propagation process at inference time is formally given by:

$$\mathbf{h} = \mathcal{W}_0\mathbf{x} + \Delta\mathcal{W}\mathbf{x} = \mathcal{W}_0\mathbf{x} + \mathbf{B} \cdot \sigma(\mathbf{A})\mathbf{x}. \tag{9}$$

Such an approach eliminates any additional computational overhead during inference, while concurrently preserving superior performance.

Why is This Strategy Effective? To validate our strategy of *dynamic* training combined with *static* inference, we empirically compare it against both *fully dynamic* and *fully static* approaches. For this purpose, we introduce two comparative variants: (1) **AuroRA**-$\mathcal{D}$, which maintains dynamic processing throughout both training and inference (i.e., its forward pass is consistently $\mathbf{h} = \mathcal{W}_0\mathbf{x} + \mathbf{B} \cdot \sigma(\mathbf{A}\mathbf{x})$), and (2) **AuroRA**-$\mathcal{S}$, which consistently employs a static form for both phases (i.e., $\mathbf{h} = \mathcal{W}_0\mathbf{x} + \mathbf{B} \cdot \sigma(\mathbf{A})\mathbf{x}$). We evaluate **AuroRA**, **AuroRA**-$\mathcal{D}$, and **AuroRA**-$\mathcal{S}$ by fine-tuning LLaMA3-8B for Commonsense Reasoning on datasets including ARC-E, OBQA, SIQA, and ARC-C, and record both accuracy and total training and inference time. From Table 7, we observe that: ❶ **AuroRA** exhibits nearly identical performance to **AuroRA**-$\mathcal{D}$ and significantly outperforms **AuroRA**-$\mathcal{S}$ in

https://doi.org/10.52202/085713-1241

Table 7: Comparison of accuracy and total time consumption for different settings on eight Commonsense Reasoning datasets, using LLaMA3-8B as pre-trained model.

Setting	Time	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
LoRA	15.05 h	70.8	85.2	79.9	91.7	84.3	84.2	71.2	79.0	80.8
AuroRA-$\mathcal{S}$	15.18 h	71.4	86.9	78.1	93.5	81.7	88.5	78.1	83.9	82.8
AuroRA-$\mathcal{D}$	15.60 h	72.6	87.5	79.2	94.3	83.0	89.5	78.9	85.0	83.8
AuroRA	15.28 h	72.5	87.4	79.0	94.2	83.0	89.3	78.8	84.8	83.6

practice; ❷ AuroRA achieves a runtime comparable to that of AuroRA-$\mathcal{S}$ and is markedly faster than AuroRA-$\mathcal{D}$. Therefore, our practical implementation adopts this dynamic training with static inference strategy, which can be seamlessly merged into pre-trained weights after the training phase (consistent with standard LoRA), and thereby also achieves an effective trade-off between performance and computational cost.

B.2 Algorithm Workflow

The algorithm framework is presented in Algo. 1.

Algorithm 1: Algorithm workflow of AuroRA

Input : Pretrained weight $\mathcal{W}_0$, low-rank factors $\mathbf{A} \in \mathbb{R}^{\tilde{r}\times d}$ and $\mathbf{B} \in \mathbb{R}^{d\times\tilde{r}}$, ANL parameters, training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, number of epochs T

```
/* Training Phase (dynamic update: B σ(Ax))                    */
for epoch t ← 1 to T do
    for each minibatch {x, y} in training data do
        /* Forward pass with ANL on input                      */
        h ← W_0 x + B · σ(A x) ;                    ▷ Eq. 4
        Compute loss L(h, y)
        /* Backpropagate through A, B, and ANL parameters      */
        Backpropagate and update {A, B, ANL}
    end
end
/* Inference Preparation (static merge: B σ(A))                */
Function MergeWeights():
    /* Compute element-wise ANL on matrix A                    */
      Ã ← σ(A)
    /* Form the effective weight update                        */
      ΔW ← B Ã ;                                    ▷ Eq. 3
    /* Merge into pretrained weights                           */
      W ← W_0 + ΔW
    return W
/* Inference Phase (static forward: no extra ANL)              */
for each test sample x do
      W ← MergeWeights()
      h ← W x ;                                     ▷ Eq. 9
    /* Use h for downstream prediction                         */
end
```

C Intuitive Case

To intuitively and concisely illustrate the impact of nonlinear mapping on the matrices, we first consider a simple scenario where $\mathbf{A} \in \mathbb{R}^{1\times 2}$ and $\mathbf{B} \in \mathbb{R}^{2\times 1}$:

$$\mathbf{A} = [a_1 \quad a_2], \quad \mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{10}$$

We then introduce the LeakyReLU activation function as the nonlinear mapping between the two low-rank matrices. Depending on the elements of matrix $\mathbf{A}$, the resulting weight update comprises the following four matrix structures:

$$\Delta\mathcal{W} = \begin{cases} \begin{bmatrix} b_1a_1 & b_2a_1 \\ b_1a_2 & b_2a_2 \end{bmatrix}, & \text{if } a_1 > 0 \text{ and } a_2 > 0, \\ \begin{bmatrix} b_1(\alpha a_1) & b_2(\alpha a_1) \\ b_1a_2 & b_2a_2 \end{bmatrix}, & \text{if } a_1 \leq 0 \text{ and } a_2 > 0, \\ \begin{bmatrix} b_1a_1 & b_2a_1 \\ b_1(\alpha a_2) & b_2(\alpha a_2) \end{bmatrix}, & \text{if } a_1 > 0 \text{ and } a_2 \leq 0, \\ \begin{bmatrix} b_1(\alpha a_1) & b_2(\alpha a_1) \\ b_1(\alpha a_2) & b_2(\alpha a_2) \end{bmatrix}, & \text{if } a_1 \leq 0 \text{ and } a_2 \leq 0. \end{cases} \tag{11}$$

where α is a hyperparameter of LeakyReLU, usually a positive number less than 1. Meanwhile, LoRA can only produce:

$$\Delta\mathcal{W} = \begin{bmatrix} b_1a_1 & b_2a_1 \\ b_1a_2 & b_2a_2 \end{bmatrix} \tag{12}$$

Under the LeakyReLU activation function, each negative component of $\mathbf{A}$ is scaled by the factor α, while positive components remain unchanged. This piecewise linear mapping disrupts the uniformity of low-rank multiplication, causing the final weight update $\Delta\mathcal{W}$ to depend not only on the product of $\mathbf{A}$ and $\mathbf{B}$ but also on the local behavior of each element in $\mathbf{A}$. Consequently, even under tight rank constraints, the model benefits from a richer set of possible weight updates, enhancing its adaptability to varying input distributions.

D Proof of Proposition 2.1

In this appendix, we provide the complete theoretical analysis and proof of Proposition 2.1. We first restate the problem setup and then present the necessary lemmas, followed by the main proof.

Definition D.1 (Best Linear Rank-r Error)**.** For $M \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, define

$$\varepsilon_r(M) := \inf_{U \in \mathbb{R}^{d_{\text{out}} \times r},\, V \in \mathbb{R}^{r \times d_{\text{in}}}} \| M - U\,V \|.$$

If $\text{rank}(M) > r$, then $\varepsilon_r(M) > 0$ [100].

Assumption D.2 (Bounded Input Domain)**.** We assume $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^{d_{\text{in}}}$ satisfies $\|\mathbf{x}\| \leq X_{\max}$. Then for $A \in \mathbb{R}^{r \times d_{\text{in}}}$ with $\|A\| \leq A_{\max}$, the vector $\mathbf{z} = A\mathbf{x}$ remains in a bounded set $\Omega \subset \mathbb{R}^r$ (compact).

Definition D.3 (Nonlinear Low-Rank Update)**.** Let

$$M_{\text{nonlinear}}(\mathbf{x}) \;=\; B\,\sigma\big(A\,\mathbf{x}\big),$$

with $A \in \mathbb{R}^{r \times d_{\text{in}}}$, $B \in \mathbb{R}^{d_{\text{out}} \times r}$. The map $\sigma : \mathbb{R}^r \to \mathbb{R}^r$ is given by

$$\sigma(\mathbf{z}) \;=\; \mathcal{F}(\mathbf{z}) \;+\; \mathbf{w}_s \cdot \mathbf{s}(\mathbf{z}),$$

where $\mathcal{F}$ is a fixed bounded function (e.g. $\tanh$-based) and $\mathbf{s}(\mathbf{z})$ denotes B-spline basis functions in $\mathbb{R}^r$.

Lemma D.4 (Piecewise Polynomial Approximation)**.** *Consider $f : \Omega \to \mathbb{R}^m$ with $f \in C^k(\Omega)$ on a bounded domain $\Omega \subset \mathbb{R}^r$. Let $\Delta > 0$ be the subdivision size in each coordinate axis for constructing a tensor-product B-spline. Then there exists a B-spline $g(\mathbf{z})$ such that*

$$\sup_{\mathbf{z} \in \Omega} \|f(\mathbf{z}) - g(\mathbf{z})\| \;\leq\; C_f(\Delta)^k,$$

where $C_f > 0$ is a constant depending on f's k-th order partial derivatives and the geometry of Ω [34].

https://doi.org/10.52202/085713-1241

Lemma D.5 (Combining Fixed and Learnable Nonlinearities). *Let $\Omega \subset \mathbb{R}^r$ be compact. Assume $\mathcal{F} : \Omega \to \mathbb{R}^r$ is fixed, bounded, and C^1, and let $h(\mathbf{z}) \in C^k(\Omega)$ be the target. Define*

$$\sigma(\mathbf{z}) \;=\; \mathcal{F}(\mathbf{z}) \;+\; \mathbf{w}_s \cdot \mathbf{s}(\mathbf{z}),$$

where $\mathbf{s}(\mathbf{z})$ is a B-spline basis. Then, for any $\epsilon > 0$, one can choose $\Delta > 0$ and $\mathbf{w}_s$ such that

$$\sup_{\mathbf{z}\in\Omega} \big\| h(\mathbf{z}) - \big[\mathcal{F}(\mathbf{z}) + \mathbf{w}_s \cdot \mathbf{s}(\mathbf{z})\big] \big\| \;\le\; \epsilon.$$

Furthermore, the error decays like $O\big((\Delta)^k\big)$ as $\Delta \to 0$.

Proof of Lemma D.5. Let $r(\mathbf{z}) = h(\mathbf{z}) - \mathcal{F}(\mathbf{z})$. Since $h \in C^k(\Omega)$ and $\mathcal{F}$ is fixed and C^1, $r(\mathbf{z})$ remains C^k. Applying Lemma D.4 to $r(\mathbf{z})$ yields a B-spline $g(\mathbf{z})$ with $\|r(\mathbf{z}) - g(\mathbf{z})\| \le C\,(\Delta)^k$. Hence $\|h(\mathbf{z}) - [\mathcal{F}(\mathbf{z}) + g(\mathbf{z})]\| \le C\,(\Delta)^k$, completing the proof. □

We restate Proposition 2.1 here for completeness:

Proposition 2.1 (Lower Approximation Error) *Let $M \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ with $\operatorname{rank}(M) > r$. Then*

$$\varepsilon_r(M) \;:=\; \inf_{U,V} \|M - U\,V\| \;>\; 0.$$

Under Definition D.3, there exist A^*, B^* and a B-spline parameter set $(\mathbf{w}_s^*)$ such that

$$\|\,M - M_{\text{nonlinear}}\| \;\le\; c\,\varepsilon_r(M), \quad 0 < c < 1.$$

Proof. Let $M^* = U^*V^*$ be the best linear rank-r approximation of M, so $\|M - M^*\| = \varepsilon_r(M)$. Denote the residual $R = M - M^*$, and we have $\|R\| = \varepsilon_r(M)$.

By Assumption D.2, for $\|\mathbf{x}\| \le X_{\max}$, let $\mathbf{z} = A^*\mathbf{x} \in \Omega \subset \mathbb{R}^r$, with $\|A^*\| \le A_{\max}$. Thus $\mathbf{z}$ lies in a compact Ω. Consider $R(\mathbf{x})$ as a function $h(\mathbf{z}) = R(\mathbf{x})$. Since R is linear (hence C^∞), $h(\mathbf{z})$ is at least C^1 in $\mathbf{z}$.

From Lemma D.5, there is a B-spline $\mathbf{w}_s^* \cdot \mathbf{s}(\mathbf{z})$ approximating $h(\mathbf{z}) - \mathcal{F}(\mathbf{z})$ within $\gamma\,\|R\|$ for some $0 < \gamma < 1$. Define

$$\widehat{M}(\mathbf{x}) := M^*(\mathbf{x}) \;+\; B^*\Big[\mathcal{F}(A^*\mathbf{x}) + \mathbf{w}_s^* \cdot \mathbf{s}(A^*\mathbf{x})\Big].$$

Then

$$\|\widehat{M} - M\| \;=\; \|\big(M^* + B^*[\ldots]\big) - (M^* + R)\| \;=\; \big\|\,B^*\big[\mathcal{F}(\cdot) + \mathbf{w}_s^* \cdot \mathbf{s}(\cdot)\big] - R\big\| \;\le\; \gamma\,\|R\| = \gamma\,\varepsilon_r(M).$$

Since $\gamma < 1$, we obtain $\|\widehat{M} - M\| < \varepsilon_r(M)$, which strictly improves upon the LoRA limit. Setting $\widehat{M} \equiv M_{\text{nonlinear}}$ completes the proof. □

E Proof of Proposition 2.2

Proposition 2.2. *In the* `AuroRA`*, the use of the* tanh *activation function and B-spline basis functions results in bounded gradients with respect to both the inputs and the model parameters.*

The loss function L for a single data point (x, y) is defined as $L(x, y) = \frac{1}{2}\,\|f_{\texttt{AuroRA}}(x) - y\|^2$, where $y \in \mathbb{R}^{d_{\text{out}}}$ is the target output. We will compute and bound the gradients of the loss function with respect to W_b, w_s and the input x.

Lemma E.1. *The gradients of the loss function with respect to W_b is bounded.*

Proof. Compute the gradient $\frac{\partial L}{\partial W_b}$:

$$\frac{\partial L}{\partial W_b} = \left(f_{\texttt{AuroRA}}(x) - y\right)^\top B \cdot \frac{\partial \text{ANL}(A^\top x)}{\partial W_b}.$$

Compute $\frac{\partial \text{ANL}(z)}{\partial W_b}$:

$$\frac{\partial \text{ANL}(z)}{\partial W_b} = \frac{\partial \phi(W_b\phi(z))}{\partial W_b} = \operatorname{diag}\left(\phi'\left(W_b\phi(z)\right)\right) \cdot \phi(z)^\top,$$

where $\phi'(u) = 1 - \tanh^2(u)$ is the derivative of Tanh, $\mathrm{diag}(v)$ denotes a diagonal matrix with vector v on the diagonal. It is not difficult to deduce that $\phi(z) \in (-1, 1)$ since Tanh outputs are bounded, and $\phi'(u) \in (0, 1]$ because $1 - \tanh^2(u) \leq 1$. So $\frac{\partial \mathrm{ANL}(z)}{\partial W_b}$ is bounded. Consequently, $\frac{\partial L}{\partial W_b}$ is bounded as it is a product of bounded terms. □

Lemma E.2. *The gradients of the loss function with respect to w_s is bounded.*

Proof. Compute the gradient $\frac{\partial L}{\partial w_s}$:

$$\frac{\partial L}{\partial w_s} = \left(f_{\texttt{AuroRA}}(x) - y\right)^\top B \cdot \frac{\partial \mathrm{ANL}(A^\top x)}{\partial w_s}.$$

Compute $\frac{\partial \mathrm{ANL}(z)}{\partial w_s}$:

$$\frac{\partial \mathrm{ANL}(z)}{\partial w_s} = s(z),$$

since $\mathrm{ANL}(z)$ is linear in w_s. B-spline basis functions $B(z_i)$ are smooth and have compact support, and the outputs of $B(z_i)$ are bounded. Therefore, $s(z)$ is bounded, and thus $\frac{\partial L}{\partial w_s}$ is bounded. □

Lemma E.3. *The gradients of the loss function with respect to the input x is bounded.*

Proof. Compute the gradient $\frac{\partial L}{\partial x}$:

$$\frac{\partial L}{\partial x} = \left(f_{\texttt{AuroRA}}(x) - y\right)^\top \left(W + B \cdot \frac{\partial \mathrm{ANL}(A^\top x)}{\partial x}\right).$$

Compute $\frac{\partial \mathrm{ANL}(A^\top x)}{\partial x}$:

$$\frac{\partial \mathrm{ANL}(A^\top x)}{\partial x} = \frac{\partial \mathrm{ANL}(z)}{\partial z} \cdot A^\top,$$

where $z = A^\top x$. Compute $\frac{\partial \mathrm{ANL}(z)}{\partial z}$:

$$\frac{\partial \mathrm{ANL}(z)}{\partial z} = \frac{\partial \phi(W_b \phi(z))}{\partial z} + w_s \cdot \frac{\partial s(z)}{\partial z}.$$

Compute $\frac{\partial \phi(W_b \phi(z))}{\partial z}$:

$$\frac{\partial \phi(W_b \phi(z))}{\partial z} = \mathrm{diag}\left(\phi'\left(W_b \phi(z)\right)\right) W_b \, \mathrm{diag}\left(\phi'(z)\right).$$

$\phi'(z)$ and $\phi'\left(W_b \phi(z)\right)$ are bounded in $(0, 1]$. Entries of W_b are finite. So $\frac{\partial \phi(W_b \phi(z))}{\partial z}$ is bounded.

Compute $\frac{\partial s(z)}{\partial z}$:

$$\frac{\partial s(z)}{\partial z} = \left[B'(z_1), B'(z_2), \ldots, B'(z_r)\right]^\top.$$

Derivatives $B'(z_i)$ of B-spline functions are bounded due to their polynomial nature and compact support. So $\frac{\partial s(z)}{\partial z}$ is bounded. Therefore, $\frac{\partial \mathrm{ANL}(z)}{\partial z}$ is bounded, leading to $\frac{\partial \mathrm{ANL}(A^\top x)}{\partial x}$ being bounded. Consequently, $\frac{\partial L}{\partial x}$ is bounded. □

F Dataset

F.1 GLUE Benchmark

The GLUE (General Language Understanding Evaluation), as introduced in [39], is a widely adopted benchmark in the field of Natural Language Processing (NLP). GLUE encompasses a collection of eight diverse NLP tasks: MNLI (natural language inference), SST-2 (sentiment analysis), MRPC (paraphrase detection), CoLA (linguistic acceptability), QNLI (natural language inference), QQP (question answering), RTE (recognizing textual entailment), and STS-B (textual similarity). The statistical details of these datasets are summarized in Table 8.

https://doi.org/10.52202/085713-1241

Table 8: Detailed task descriptions and dataset statistics for the GLUE benchmark. STS-B is categorized as a regression task, while all other tasks involve single-sentence or sentence-pair classification.

Corpus	Task	Metrics	# Labels	# Train	# Val	# Test	Domain
			Single-Sentence Tasks				
CoLA	Acceptability	Matthews Corr.	2	8.55k	1.04k	1.06k	misc.
SST-2	Sentiment	Accuracy	2	67.3k	872	1.82k	Movie reviews
			Similarity and Paraphrase Tasks				
MRPC	Paraphrase	Accuracy/F1	2	3.67	408	1.73k	News
STS-B	Sentence similarity	Pearson/Spearman Corr.	1	5.75k	1.5k	1.38k	misc.
QQP	Paraphrase	Accuracy/F1	2	364k	40.4k	391k	Social QA
			Inference Tasks				
MNLI	NLI	Accuracy	3	393k	19.65k	19.65k	misc.
QNLI	QA/NLI	Accuracy	2	105k	5.46k	5.46k	Wikipedia
RTE	NLI	Accuracy	2	2.49k	277	3k	News & Wikipedia

F.2 Commonsense Reasoning

Following [19], we use eight datasets in Commonsense Reasoning task. (1) The BoolQ [40] dataset is a question-answering benchmark consisting of 15,942 examples, where the questions are naturally occurring and generated in unprompted and unconstrained settings, requiring yes/no answers. (2) The PIQA [41] dataset presents questions with two potential solutions, demanding physical commonsense reasoning to identify the correct answer. (3) The SIQA [42] dataset focuses on reasoning about human actions and their social implications. (4) The HellaSwag [43] dataset is designed for commonsense natural language inference (NLI) tasks, where each question includes a context and several potential endings, from which the correct continuation must be selected. (5) The WinoGrande [44] dataset is a fill-in-the-blank task with binary options, where the goal is to select the most plausible option for a given sentence requiring commonsense reasoning. (6) The ARC-c and (7) ARC-e [45] datasets refer to the Challenge and Easy sets, respectively, of the ARC dataset, which consists of multiple-choice science questions designed at a grade-school level, with the former being more challenging than the latter. (8) The OBQA [46] dataset focuses on questions that necessitate multi-step reasoning, integration of external common knowledge, and in-depth text comprehension. Statistical details are shown in Table 9.

Table 9: Details of datasets being evaluated in commonsense reasoning task.

Dataset	#Train	#Test	Answer
BoolQ [40]	9.4K	3270	Yes/No
PIQA [41]	16.1K	1830	Option
SIQA [42]	33.4K	1954	Option
HellaSwag [43]	39.9K	10042	Option
WinoGrande [44]	63.2K	1267	Option
ARC-e [45]	1.1K	2376	Option
ARC-c [45]	2.3K	1172	Option
OBQA [46]	5.0K	500	Option

F.3 Image Classification

We show the details of the datasets in Image Classification task in Table 10.

G Hyperparameters

To ensure the reproducibility of our experimental results, we provide the detailed hyperparameter settings used in our experiments. In all of our experiments, to achieve a better balance between parameter count and performance, we set the hidden layer dimension (Rank $\widetilde{r}$) of `AuroRA` to 2. Correspondingly, we set the hyperparameter α of `AuroRA` to 4. Natural Language Understanding and

Table 10: Details of the datasets for the Image Classification task.

Dataset	#Class	#Train	#Val	#Test	Rescaled resolution
OxfordPets [47]	37	3,312	368	3,669	224×224
StandfordCars [52]	196	7,329	815	8,041	
CIFAR10 [48]	10	45,000	5,000	10,000	
DTD [49]	47	4,060	452	1,128	
EuroSAT [50]	10	16,200	5,400	5,400	
FGVC [53]	100	3,000	334	3,333	
RESISC45 [51]	45	18,900	6,300	6,300	
CIFAR100 [48]	100	45,000	5,000	10,000	

image classification tasks run on four NVIDIA GeForce RTX 4090 (24GB) GPUs. Commonsense reasoning and subject-driven generation tasks run on NVIDIA L20 (48GB).

G.1 Natural Language Understanding

We provide the hyperparameters used for the GLUE benchmark in natural language understanding experiments in Table 11. To facilitate reproducibility, we fix the random seed to 0. We tune the learning rate, while all other settings follow those used in LoRA [16] and FourierFT [61].

Table 11: Hyperparameter setup of **AuroRA** for the GLUE benchmark.

Model	Hyperparameter	STS-B	RTE	MRPC	CoLA	SST-2	QNLI
Both	Optimizer	AdamW					
	LR Schedule	Linear					
	Warmup Ratio	0.06					
	Rank $\widetilde{r}$	2					
	α	4					
Base	Epochs	30	80	30	90	30	80
	Learning Rate	6E-4	5E-4	8E-4	5E-3	8E-4	5E-3
	Max Seq. Len	512	512	512	512	512	512
	Batch Size	64	16	64	32	32	32
Large	Epochs	20	10	30	50	40	20
	Learning Rate	3E-4	4E-4	1E-3	5E-4	8E-4	4E-4
	Max Seq. Len	512	512	512	256	128	512
	Batch Size	16	32	16	16	16	16

G.2 Commonsense Reasoning

We provide the detailed hyperparameters for fine-tuning LLaMA3-8B in the commonsense reasoning task in Table 12.

Table 12: Hyperparameter setup of **AuroRA** for Commonsense Reasoning.

Hyperparameter	Commonsense Reasoning
Rank $\widetilde{r}$	2
α	4
Dropout	0.05
Batch Size	16
Optimizer	Adam W
Learning Rate	3e-4
Warmup Steps	100
Epochs	3
Target module	q,k,v,up,down

https://doi.org/10.52202/085713-1241

G.3 Image Classification

We provide the detailed hyperparameters for the image classification in Table 13. We tune the learning rate, while the weight decay value follows the settings used in FourierFT [61] without tuning.

Table 13: Hyperparameter setup of **AuroRA** for the image classification.

Model	Hyperparameter	OxfordPets	StanfordCars	CIFAR10	DTD	EuroSAT	FGVC	RESISC45	CIFAR100
Both	Optimizer	AdamW							
	LR Schedule	Linear							
	Epochs	10							
	Rank $\widetilde{r}$	2							
	α	4							
Base	Learning Rate (**AuroRA**)	5e-3	1e-2	1e-2	1e-2	5e-3	1e-2	8e-3	8e-3
	Learning Rate (Head)	5E-3	1e-2	3e-2	8E-3	8E-3	1e-2	1e-2	5e-3
	Weight Decay	8E-4	4E-5	9E-5	7E-5	3E-4	7E-5	3E-4	1E-4
Large	Learning Rate (**AuroRA**)	5e-3	9e-3	8e-3	8e-3	4e-3	1.5e-2	7.5e-3	1.5e-2
	Learning Rate (Head)	4e-3	8e-3	4e-2	9e-3	8e-3	1e-2	1.5e-2	5e-3
	Weight Decay	8E-4	4E-5	9E-5	7E-5	3E-4	7E-5	3E-4	1E-4

H More Cases of Generated Images

In Figure 6, we present more results of subject-driven generation using both LoRA and **AuroRA**.

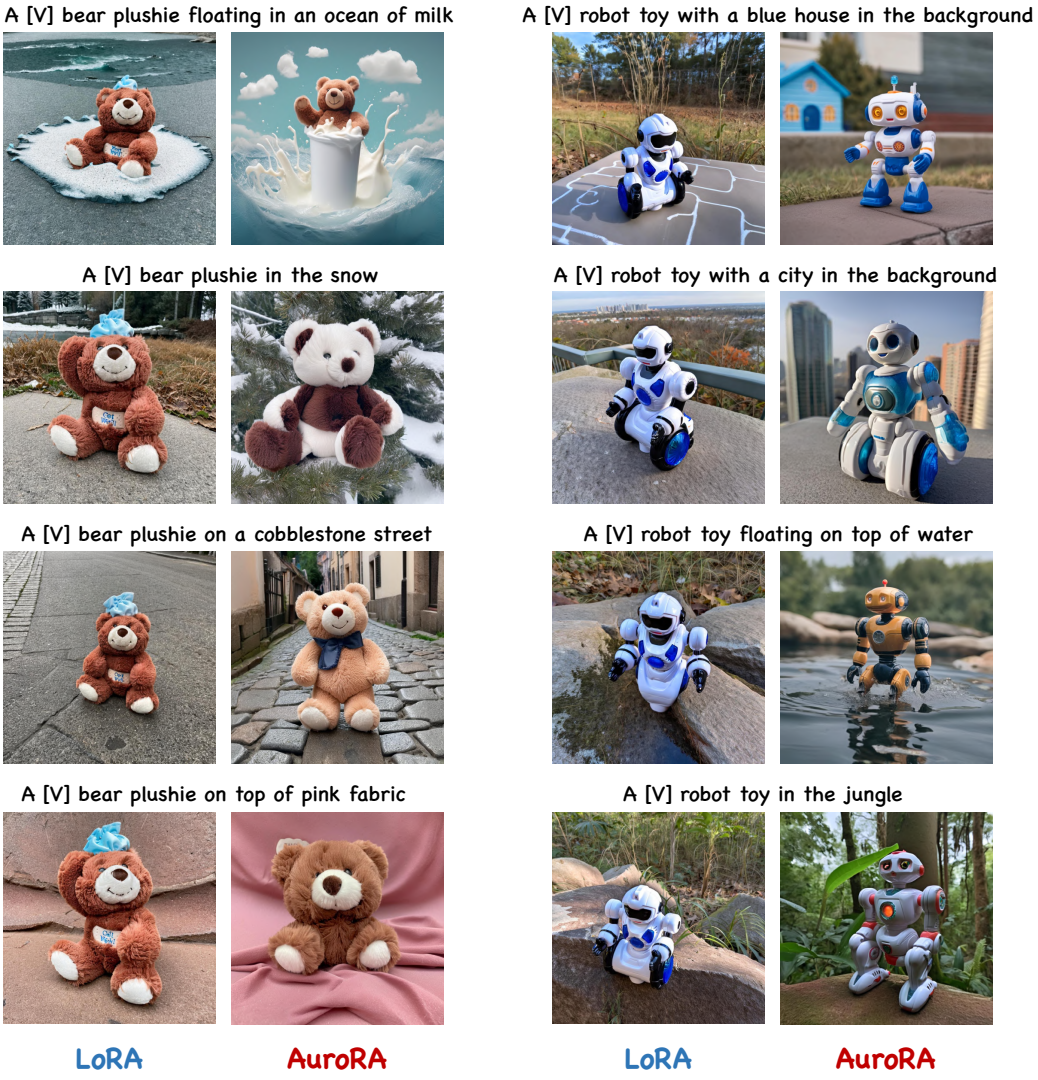

Figure 6: More generated images in the subject-driven generation task.

https://doi.org/10.52202/085713-1241