NORM-BOUNDED LOW-RANK ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

In this work, we propose norm-bounded low-rank adaptation (NB-LoRA) for parameter-efficient fine tuning. NB-LoRA is a novel parameterization of low-rank weight adaptations that admits explicit bounds on each singular value of the adaptation matrix, which can thereby satisfy any prescribed unitarily invariant norm bound, including the Schatten norms (e.g., nuclear, Frobenius, spectral norm). The proposed parameterization is unconstrained, smooth, and complete, i.e. it covers all matrices satisfying the prescribed rank and singular-value bounds. Natural language generation experiments show that NB-LoRA matches or surpasses performance of competing LoRA methods, while exhibiting stronger hyper-parameter robustness. Vision fine-tuning experiments show that NB-LoRA can avoid model catastrophic forgetting without minor cost on adaptation performance, and compared to existing approaches it is substantially more robust to a hyper-parameters such as including adaptation rank, learning rate and number of training epochs.

1 INTRODUCTION

Large pre-trained vision and language models have demonstrated impressive generalization capability across a wide variety of tasks; see, e.g. Achiam et al. (2023); Touvron et al. (2023). When a more specific target task is identified, however, it has been observed that parameter-efficient fine-tuning (PEFT) techniques, e.g. Houlsby et al. (2019); Hu et al. (2022), can improve performance via quick model adaption with low computation and data requirements. The primary goal for an effective PEFT method is to achieve good adaptation performance with high training efficiency, i.e., dramatically fewer trainable parameters and training epochs. Since training efficiency is the target, ideally such a method will be quite robust to hyperparameters. Alongside this primary goal, it is often also desirable to maintain the generalization performance of the original pre-trained model as much as possible, i.e. avoid "catastrophic forgetting" Qiu et al. (2023); Biderman et al. (2024).

Low-rank adaption (LoRA) (Hu et al., 2022) is a widely applied PEFT method, which parameterizes the update of pretrained weights $W_p \in \mathbb{R}^{m\times n}$ during finetuning as

$$y = (W_p + W)x = \left(W_p + \frac{\alpha}{r} B^\top A\right) x \tag{1}$$

where $A \in \mathbb{R}^{r\times n}, B \in \mathbb{R}^{r\times m}$ are the learnable matrices, α is a scaling factor, and $r \ll \min(m, n)$ is the rank budget of weight adaptation W. Matrix rank is a one way to quantify the "size" of a weight, corresponding the underlying dimensionality of its operation. But matrix norms – such as nuclear, Frobenius, or spectral norms – provide another notion of size, quantifying the magnitude of a matrix's elements and of its operation on vectors.

Recent works show that it is beneficial to control the rank and norm of the weight adaption. Jang et al. (2024); Kim et al. (2025) show that the global minimum of fine-tuning has low rank and small magnitude while spurious local minima (if they exist) have high rank and large magnitude. Moreover, bounding the magnitude of W can enhance training robustness (Bini et al., 2025). In Hu et al. (2025), LoRA training can achieve sub-quadratic time complexity under certain norm-bound conditions.

Motivated by those findings, we propose norm-bounded low-rank adaptation (NB-LoRA), a novel finetuning method that admits explicit bounds on both the rank *and* norm of weight update through matrix reparameterization (see Fig. 1). Our approach can control a family of matrix norms, called Schatten p-norms (i.e. p-norms of the singular value sequence), which include the nuclear norm, Frobenius norm, and spectral norm as special cases. We summarize our contributions as follows.

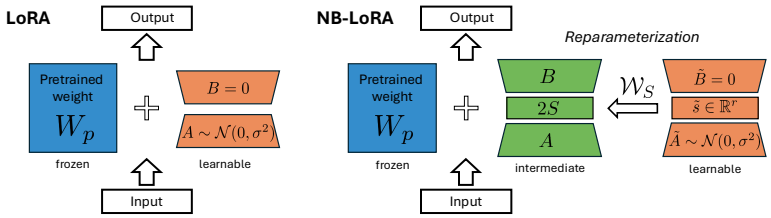


Figure 1: Visualization (Left) of the original LoRA Hu et al. (2022) and (Right) of our proposed method NB-LoRA, where bounded rank and norm are enforced by reparameterization $\mathcal{W}_S$.

- Our parameterization is a smooth map $W = \mathcal{W}_S(\tilde{A}, \tilde{B})$ which takes as argument two free matrix variables of the same size as A, B, but the resulting W automatically satisfies user-prescribed bounds on both rank and all individual singular values of W, which further allows any Schatten p-norm bound on W to be specified.
- Our parameterization is *complete*, i.e., for any $W \in \mathbb{R}^{m \times n}$ satisfying the prescribed bounds on singular values, there exists a (not necessarily unique) $\tilde{A}, \tilde{B}$ such that $W = \mathcal{W}_S(\tilde{A}, \tilde{B})$.
- LLM fine-tuning experiments show that NB-LoRA can substantially improve training stability, overall performance and robustness to learning rates.
- Vision transformer fine-tuning experiments illustrate that NB-LoRA can achieve similar adaptation performance to LoRA and other existing methods while exhibiting less "forgetting" of the source model. Also, norm bounds appear to significantly reduce sensitivity to hyperparameter variation.

2 Related Work

LoRA can be highly sensitive to learning rate (Bini et al., 2024; Biderman et al., 2024), model initialization (Hayou et al., 2024), and it is susceptible to over-training (Qiu et al., 2023). To mitigate these effects, several recent works have proposed regularization techniques for LoRA. For example, Gouk et al. (2021); Chen et al. (2023) propose an approach that preserves the Euclidean weight distances between pre-trained and fine-tuned models. In Liu et al. (2024), DoRA was proposed based on investigation of the vector-wise norm of the adaption matrix, and introduces an adaptive scaling of W. Bini et al. (2025) proposed DeLoRA - a PEFT method that decouples the angular learning from adaptation strength. VeRA is another method which learns a scaling vector for LoRA weights (Kopiczko et al., 2024b). Our method also contains a learnable scaling vector, which can be used to explicitly control bounds on each singular value of the weight adaptation.

Another line of LoRA methods are closely related to singular value decomposition (SVD). Meng et al. (2024) proposed a novel SVD-based LoRA initialization, called PiSSA, which can significantly speed up the training of LoRA. Zhang et al. (2023) proposed a dynamical rank allocation scheme, called AdaLoRA, which adaptively update the rank bound in each LoRA layer. In Lingam et al. (2024); Bałazy et al. (2024), the singular vectors of pretrained weights are re-used and a small square matrices are learned during fine-tuning. No explicit control of norm bounds or constraint on singular values were considered in these methods.

3 Motivating Analysis of LoRA

In this section we provide some brief analyses of LoRA that motivate consideration of alternative parameterizations of A and B.

Analysis of Gradients and Initialization. For LoRA the standard approach is to initialize with one of A or B equal to zero, so that $W = 0$ initially, and the other as a small random matrix to enable learning but avoid training instability (see Hayou et al. (2024) for a discussion of approaches). Based on Equation (1), we can obtain the gradients of the loss $\ell(\cdot)$ with respect to A and B as follows:

$$\frac{\partial \ell}{\partial A} = \frac{\alpha}{r} B \left(\frac{\partial \ell}{\partial y} \right) x^\top, \quad \frac{\partial \ell}{\partial B} = \frac{\alpha}{r} A x \left(\frac{\partial \ell}{\partial y} \right)^\top.$$

The component $\partial \ell / \partial y$ is typically not large, since the pre-trained base model has reasonable generalization capability over a wide range of tasks. Thus, at the beginning of fine-tuning, if $B = 0$ and A is a small random matrix, then the gradient $\partial \ell / \partial A = 0$ and $\partial \ell / \partial B$ is small and can be noisy. Therefore, both gradients could be small and uninformative for a large number of training steps, leading to slow convergence and poor performance since fine-tuning is often carried out for a few epochs. Large learning rate can help to speed up but it may cause training instability. The above phenomenon has been reported and analyzed in a recent work (Meng et al., 2024), which proposed PiSSA as an alternative initialization.

In this work, we provide a novel model reparameterization, and we note that this impacts training behavior since gradient descent is affected by changes of coordinates. In our approach, the low-rank matrices A, B are constructed from new free variables $\tilde{A}, \tilde{B}$ which share the same size as A and B, respectively. Under this reparameterization (detailed in Section 4), A and B cannot be both very small matrices. For example, if B is a zero matrix, then by construction A is a relatively large matrix, which in turn provides sufficiently large gradient for A to move away from zero in a few steps. However, the Frobenius norms of A, B are guaranteed to be bounded, thus the gradients of A, B are also bounded assuming that $x \left(\partial \ell / \partial y\right)^\top$ is bounded, and thus large gradient steps can be taken without training instability, i.e. we expect norm bounds to assist with robustness to learning rate. This informal argument is supported by experimental results in Section 5.

Analysis of Model Forgetting. Let $\{(x_i^s, y_i^s)\}_{1 \leqslant i \leqslant M}$ be the pretrained input-output pair of equation 1 under the source training dataset $\mathcal{D}_S$. After fine-tuning the adaption weight W based on some target dataset $\mathcal{D}_T$, we can approximate the loss changes on $\mathcal{D}_S$ by

$$\ell_{\mathcal{D}_s}(W_p + W) - \ell_{\mathcal{D}_s}(W_p) \approx \frac{1}{M} \sum_{i=1}^{M} \left(\frac{\partial \ell}{\partial y_i^s} \right)^\top W x_i^s.$$

To prevent catastrophic forgetting we need to bound the left-hand side. Since x_s and $\frac{\partial \ell}{\partial y_i^s}$ are fixed, we argue that constraining the norm of W is a natural approach. If we have access to $\mathcal{D}_S$ during finetuning, one could incorporate source loss into the training to mitigate forgetting. However, $\mathcal{D}_S$ is often not available in fine-tuning applications.

4 NB-LoRA

In this section we present our main contribution: a parameterization of low-rank matrices that admits bounds on each individual singular value, and hence on any unitarily invariant matrix norm.

4.1 Preliminaries and Problem Formulation

The problem we are interested in can be formalized as follows:

$$\min \quad \ell(W) \quad \text{s.t.} \quad \operatorname{rank}(W) \leqslant r, \ \|W\|_{S_p} \leqslant \delta \tag{2}$$

where ℓ is some training loss and $\|W\|_{S_p} = \left(\sum_{i=1}^{r} \sigma_i^p\right)^{1/p}$ for $p \in [1, \infty)$ and $\|W\|_{S_\infty} = \sigma_1$, where $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_r \geqslant 0$ are the singular values of W. Since Schatten p-norm is the vector p-norm of the singular value sequence, it is unitarily invariant, i.e., $\|W\|_{S_p} = \|UWV\|_{S_p}$ for any orthogonal matrices U, V.

We first define some notation. Since our approach involves comparing singular values of matrices of potentially different ranks and sizes, for convenience we define $\sigma_j(W) = 0$ if $j > \operatorname{rank}(W)$. We now introduce the relation $\preceq_\sigma$.

Definition 4.1. Let X, Y be two matrices. We say $X \preceq_\sigma Y$ if $\sigma_j(X) \leqslant \sigma_j(Y), \ \forall j \in \mathbb{N}$.

Note the $\preceq_\sigma$ is reflexive ($X \preceq_\sigma X$) and transitive ($X \preceq_\sigma Y, Y \preceq_\sigma Z \Rightarrow X \preceq_\sigma Z$). But it is not antisymmetric, i.e., $X \preceq_\sigma Y, Y \preceq_\sigma X \nRightarrow X = Y$, e.g., when X, Y are distinct orthogonal matrices. Most importantly for our purposes: if $X \preceq_\sigma Y$, then $\|X\|_{S_p} \leqslant \|Y\|_{S_p}$ for all $p \in [1, \infty]$.

Let $s \in \mathbb{R}_+^r$, where $\mathbb{R}_+ = [0, \infty)$, and $S = \operatorname{diag}(s)$ be the diagonal matrix with $S_{jj} = s_j$. We define the set of matrices whose singular values are bounded by S by

$$\mathbb{W}_S := \{W \in \mathbb{R}^{m \times n} \mid W \preceq_\sigma S\}.$$

Note that for any $W \in \mathbb{W}_S$, we have $\operatorname{rank}(W) \leqslant \operatorname{rank}(S) = r$ and $\|W\|_{S_p} \leqslant \|S\|_{S_p}$.

4.2 NB-LoRA Parameterization

We now present so-called *direct* parameterization of $\mathbb{W}_S$, a smooth mapping $\mathcal{W}_S$ from free matrix variables to W which maps onto the entire set $\mathbb{W}_S$. Then, we can transform (2) into an unconstrained problem by further parameterizing the positive diagonal matrix S such that $\|S\|_{S_p} = \delta$.

Our parameterization takes $\tilde{A} \in \mathbb{R}^{r\times n}, \tilde{B} \in \mathbb{R}^{r\times m}$ as the free parameters and produces W via

$$W = \mathcal{W}_S(\tilde{A}, \tilde{B}) := 2B^\top SA, \text{ where } \begin{bmatrix} A^\top \\ B^\top \end{bmatrix} = \operatorname{Cayley}\left(\begin{bmatrix} \tilde{A}^\top \\ \tilde{B}^\top \end{bmatrix}\right). \tag{3}$$

Here the Cayley transformation for a tall matrix $\begin{bmatrix} X \\ Y \end{bmatrix}$ with $X \in \mathbb{R}^{r\times r}$ and $Y \in \mathbb{R}^{q\times r}$ is defined by

$$\operatorname{Cayley}\left(\begin{bmatrix} X \\ Y \end{bmatrix}\right) := \begin{bmatrix} (I - Z)(I + Z)^{-1} \\ -2Y(I + Z)^{-1} \end{bmatrix}, \text{ where } Z = X - X^\top + Y^\top Y. \tag{4}$$

Note that $G = \operatorname{Cayley}(F)$ is a semi-orthogonal matrix, i.e., $G^\top G = I$ for any tall matrix F (Golub & Van Loan, 2013), however it is not by itself a complete parameterization for the set of semi-orthogonal matrices, e.g., there does not exist an F such that $\operatorname{Cayley}(F) = -I$. Despite this, we have the following, which is the main theoretical result of the paper.

Theorem 4.2. *The NB-LoRA parameterization in (3) is a direct (smooth and complete) parameterization of $\mathbb{W}_S$, i.e. $\mathcal{W}_S$ is differentiable and $\mathcal{W}_S(\mathbb{R}^N) = \mathbb{W}_S$.*

Remark 4.3. A special case of the above theorem is $S = I$, which is a complete parameterization of all 1-Lipschitz linear layer, i.e. $f(x) = Wx$ with $\|W\|_{S_\infty} \leqslant 1$, see Proposition 3.3 of Wang & Manchester (2023). One can further extend it to a nonlinear layer with low-rank and norm-bounded Jacobian. Specifically, we take a nonlinear layer of the form $f(x) = 2B^\top D_1 \phi(D_2 Ax)$ where A, B are constructed from (3), D_1, D_2 are diagonal matrices satisfying $0 \preceq D_1 D_2 \preceq S$ and ϕ is a scalar activation with slope-restricted in $[0, 1]$. Then, we have $\partial f/\partial x \in \mathbb{W}_S$ for all $x \in \mathbb{R}^n$.

Imposing the Norm Bound on W. From Theorem 4.2, if we construct a complete parameterization for the set of singular bound vector $s \in \mathbb{R}^r_+$ such that $\|s\|_p = \delta$, then the proposed NB-LoRA (3) covers all adaptation matrices W satisfying the prescribed rank and norm bounds. For $p = \infty$, we simply take $s = (\delta, \delta, \ldots, \delta)$. For $p \in [1, \infty)$, one approach is $s = \delta|\tilde{s}|/\|\tilde{s}\|_p$, where $\tilde{s} \in \mathbb{R}^r$ is a free non-zero vector. However, this parameterization is not smooth at $\tilde{s} = 0$. Instead, we use the following parameterization in our experiments:

$$s = \delta \left[\operatorname{Softmax}\left(\tilde{s}/\sqrt{r}\right)\right]^{1/p}.$$

Technically, the above parameterization omits some boundary cases with $\|W\|_{S_p} = \delta$ and $\sigma_r(W) = 0$ since softmax has strictly positive outputs. However, since it covers the interior of the feasible set and can approximate the boundary, there is no practical impact on optimization performance.

Model Initialization and Gradient analysis. Here we return to the motivating analysis from Section 3 and show why NB-LoRA helps resolve the issue of small gradients. We can adapt the standard LoRA initialization to NB-LoRA's free parameters: sampling $\tilde{A}$ as a small random matrix and setting $\tilde{B} = 0$. After applying the Cayley transformation, we have $AA^\top = I$ and $B = 0$, yielding a zero initialization for W. The gradients of A, B can be written as

$$\frac{\partial \ell}{\partial A} = 2SB\left(\frac{\partial \ell}{\partial y}\right)x^\top = 2\hat{B}\left(\frac{\partial \ell}{\partial y}\right)x^\top, \quad \frac{\partial \ell}{\partial B} = 2SAx\left(\frac{\partial \ell}{\partial y}\right)^\top = 2\hat{A}x\left(\frac{\partial \ell}{\partial y}\right)^\top$$

where $\hat{A}, \hat{B}$ satisfy

$$\hat{A}\hat{A}^\top + \hat{B}\hat{B}^\top = S(AA^\top + BB^\top)S = S^2$$

with $\|S\|_{S_p} = \delta > 0$. Hence $\hat{A}, \hat{B}$ cannot be both arbitrarily small matrices, implying that $\partial \ell/\partial A$ and $\partial \ell/\partial B$ cannot be both arbitrarily small initially. On the other hand, the Frobenius norms of $\hat{A}, \hat{B}$ are also bounded by $c\delta$ where c is the constant satisfying $\|S\|_{S_2} \leqslant c\|S\|_{S_p}$ for all S. Thus, if $x(\partial \ell/\partial y)^\top$ remains bounded, then $\partial \ell/\partial A$ and $\partial \ell/\partial B$ are bounded, allowing stable training for a wider range of learning rates than LoRA.

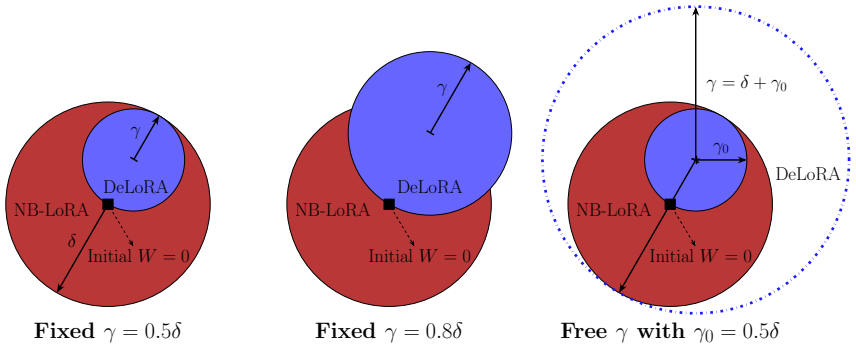


Figure 2: Visualization of the reachable sets $\mathbb{W}_{\texttt{NBLoRA}}$ (red) and $\mathbb{W}_{\texttt{DeLoRA}}$ (blue). (Left) With frozen scaling factor $\gamma = 0.5\delta$, DeLoRA provides the same certified norm bound as NB-LoRA while $\mathbb{W}_{\texttt{DeLoRA}}$ is much smaller than $\mathbb{W}_{\texttt{NBLoRA}}$. (Middle) Further increasing the fixed γ can enlarge $\mathbb{W}_{\texttt{DeLoRA}}$ but it does not cover $\mathbb{W}_{\texttt{NBLoRA}}$. (Right) DeLoRA can cover $\mathbb{W}_{\texttt{NBLoRA}}$ if γ is free and sufficiently large, i.e., $\gamma \geqslant \delta + \gamma_0$. However, its norm bound $\delta + 2\gamma_0$ is much larger than NB-LoRA.

Computational cost of Cayley Transformation. Due to the low-rank nature (r is often less than 256), computing the inverse of an $r \times r$ matrix is not overly expensive. While matrix inversion is one part of the total training cost, another computationally intensive part is the backward pass for the Cayley transformation (4). We provide an efficient custom backward step in Appendix C.

DeLoRA vs NB-LoRA. Similarly to our method, DeLoRA (Bini et al., 2025) can also control the Frobenius norm bound of weight adaption based on the following parameterization:

$$W = \frac{\gamma}{r} B^\top \Xi A - \frac{\gamma_0}{r} B_0^\top \Xi_0 A_0 \tag{5}$$

where the scaling factor γ and weight parameter A, B are initialized as γ_0 and A_0, B_0. Ξ is a diagonal matrix that normalizes each row of A, B. Similar to PiSSA Meng et al. (2024), the second term in (5) can be absorbed into the pretrained weight W_p. Note that our method can control other Schatten norms (e.g. nuclear, spectral), which were not considered in Bini et al. (2025).

As shown in Appendix D, both DeLoRA and NB-LoRA can be represented as sum of NB-LoRA matrices with both rank and norm bound of 1. The main difference comes from their reachable sets $\mathbb{W}_{\texttt{NBLoRA}}$ and $\mathbb{W}_{\texttt{DeLoRA}}$ when an explicit norm bound is specified. From Theorem 4.2 we have that $\mathbb{W}_{\texttt{NBLoRA}}$ covers the feasible region of W with norm bound of δ. Moreover, the initial point $W = 0$ of NB-LoRA lies at the center of the feasible region, allowing searching for all directions, see Figure 2. Due to the residual type initialization, DeLoRA requires a fixed $\gamma = 0.5\delta$ to ensure the same norm bound guarantee, see the left of Figure 2. Since its initial W lies at the boundary of $\mathbb{W}_{\texttt{DeLoRA}}$, the searching directions of DeLoRA are constrained in certain ranges that depend on the random initial guess. Although these issues can be resolved by making γ learnable, DeLoRA allows an unbounded Frobenius norm and needs a larger bound to cover the range of NB-LoRA, see the right of Figure 2.

PiSSA vs NB-LoRA. PiSSA (Meng et al., 2024) addresses the small initial gradient issue of LoRA via a residual-type initialization, i.e., $W = \frac{\alpha}{r}(B^\top A - B_0^\top A_0)$ where A_0, B_0 are the initial values of A, B, respectively. Since the term $\frac{\alpha}{r} B_0^\top A_0$ can be absorbed into the pretrained weight W_p, it does not cause any extra computation cost compared with LoRA. Moreover, PiSSA has much lager initial gradients than LoRA by constructing A_0, B_0 from the reduced SVD of W_p. Thus, PiSSA can speed up the fine-tuning process, however, its performance might be sensitive to learning rate as A, B are unbounded. Different from PiSSA, the proposed NB-LoRA approach address the small initial gradient issue through reparameterization. Since A, B live on a compact manifold by construction, thus NB-LoRA allows for a wide range choice of learning rates.

DoRA vs NB-LoRA. DoRA (Liu et al., 2024) decouples angular and magnitude components of weight adaptation via $W = \underline{m} \frac{(W_p + B^\top A)}{\|W_p + B^\top A\|_c}$ with $\|\cdot\|_C$ as the column-wise vector norm. Note that the normalization vector $\|W_p + B^\top A\|_c$ requires computing $B^\top A \in \mathbb{R}^{m \times n}$, whose forward computation time could be much larger than $r \times r$-matrix inverse, see Table 7 of Appendix E.

5 LLM FINE-TUNING EXPERIMENTS

In this section, we evaluate our proposed NB-LoRA method for natural language generation (NLG) tasks. Our main objectives are as follows: i) NB-LoRA can avoid small initial gradients while still maintain training stability for a wide range of learning rates; ii) Controlling the norm is beneficial for robust performance; iii) Due to the ability of tight bound control, our method can outperform existing approaches with the same certified norm bound.

NLG Tasks. We start with comparing LoRA, DoRA and PiSSA on NLG tasks. We fine-tuned the LLaMA model family (Touvron et al., 2023) and Mistral-7B-v0.1 (Jiang et al., 2023) on the MetaMathQA dataset (Yu et al., 2023) to evaluate their mathematical problem-solving capability on the GSM8K Cobbe et al. (2021) and MATH Hendrycks et al. (2021) test datasets. We also fine-tuned the models on the the CodeFeedback dataset Zheng et al. (2024) and evaluated for coding proficiency using the HumanEval Chen et al. (2021) and MBPP Austin et al. (2021). We adopt the implementation strategy from Taori et al. (2023). We follow the training setup in Meng et al. (2024) with default rank budget of $r = 128$ and scaling parameter $\alpha = r$ for LoRa and PiSSA. For the proposed NB-LoRA method, since the scaling components in S of (3) are initialized close to $1/r$, we then choose the nuclear norm bound of $\delta = r$, leading to the same scaling factor as LoRA and PiSSA. More training details can be found in Appendix E.

Large Initial Gradients and Training Stability. The analysis in Section 3 suggested that norm bounds may improve robustness to learning rates by mitigating the effect of small initial gradients in LoRA. We conducted experiments on LLaMA-2-7B fine-tuning across a wide range of learning rates from 5e-5 to 1e-3 on math and python coding datasets. Figure 3 (a) and (b) show that LoRA and DoRA both suffer from poor performance with small learning rates, due to the small gradients problem. Increasing the learning rate helps up to a point but then training goes unstable. In contrast, NB-LoRA achieves good performance for a wide range of learning rates, outperforming all other models on most tasks. PiSSA outperforms NB-LoRA in terms of peak performance on GSM8K, but under-performs on other tasks and is more sensitive to learning rate.

Figure 3 (c) and (d) show loss and gradient norm vs training steps. It can be seen that with a small learning rate (5e-5) NB-LoRA (and PiSSA) train similarly, both faster than LoRA and DoRA and with larger gradient norms. With a larger learning rate (1e-3) LoRA and DoRA were unstable, and NB-LoRA trains fastest. Note that PiSSA has a larger gradient norm but slower training: since the parameterizations are different the gradient norms are not directly comparable.

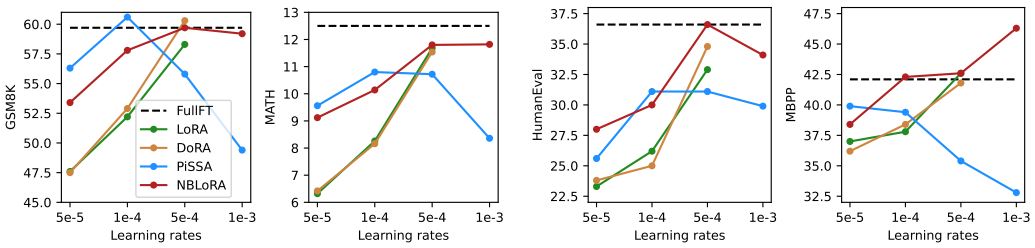
Hyperparameter Robustness. Table 2 compiles the results of a comprehensive sweep across tasks, base models and learning rates, comparing NB-LoRA to LoRA, DoRA, and PiSSA in terms of their robustness to these variations (see table caption for details). While different methods were competitive for different particular scenarios, when averaging across models and tasks NB-LoRA is clearly superior.

Scalability to Larger Models. We trained NB-LoRA to LoRA and PiSSA on the LLaMA-3-70B model for GSM8K and compared them in terms of computational resources, accuracy, and learning-rate robustness. In Table 1 it can be seen that NB-LoRA achieved the highest accuracy overall. It uniformly outperformed PiSSA, while standard LoRA achieved good performance for low learning rates but was unstable for larger learning rates. NB-LoRA required slightly more computational resources than LoRA and PiSSA: $\sim$6% more memory and $\sim$9% longer training time.

Table 1: **Scalability to larger models:** Comparison of LoRA, PiSSA and NB-LoRA on LLaMA-3-70B with learning rates from 5e-5 to 5e-4.

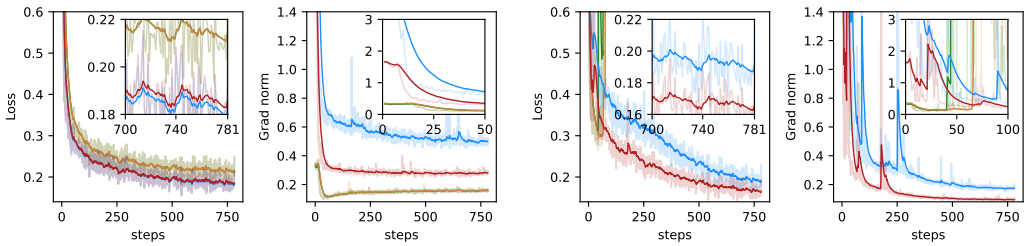
Method	Learning Rate			Computation	
	5e-5	1e-4	5e-4	GPU Mem.	Train Time
LoRA	86.2	86.2	failed	65.57GB	169m
PiSSA	83.6	79.0	41.8	65.57GB	170m
NB-LoRA	**87.1**	85.4	83.3	69.15GB	185m

Comparison with DeLoRA. Figure 4 compares NB-LoRA with DeLoRA Bini et al. (2025) with δ set to 10, 20, and free (see Section 4 and Figure 2 for discussion). Firstly, in panel (a) we see that NB-LoRA achieves the highest test accuracy, outperforming DeLoRA with an equivalent norm



(a) Test accuracy for math problem

(b) Test accuracy for python coding



(c) Learning rate of 5e-5

(d) Learning rate of 1e-3

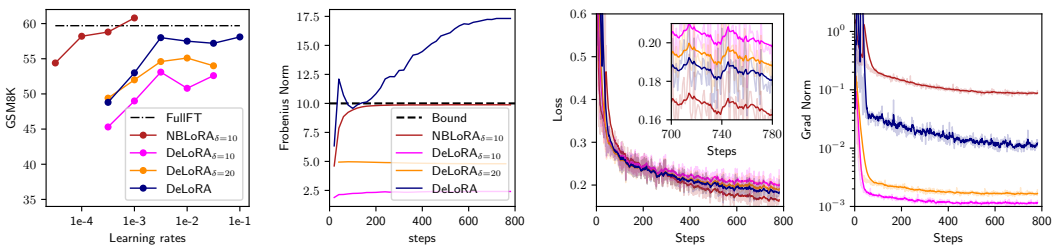
Figure 3: (Top) the evaluation accuracy over a range of learning rates and (Bottom) the loss and grad norm over the training steps for two learning rates.

Table 2: Fine-tuning three base models based on LoRA (Lo), DoRA (Do), PiSSA (Pi) and NB-LoRA (NB) over different learning rates ({1e-5, 5e-5,1e-4} for Mistral and {5e-5, 1e-4, 5e-4} for LLaMA). We report the minimum, maximum and averaged test results, where the metrics for math and coding are $\frac{1}{2}(\text{GSM8K} + \text{MATH})$ and $\frac{1}{2}(\text{HumanEval} + \text{MBPP})$, respectively.

Base Model		Mistral-7B-v0.1				LLaMA-3-8B				LLaMA-2-13B				Model Avg.			
Method		Lo	Do	Pi	NB	Lo	Do	Pi	NB	Lo	Do	Pi	NB	Lo	Do	Pi	NB
Math	min	43.8	44.2	45.9	46.7	49.6	50.4	41.6	48.5	35.1	35.1	38.2	38.1	42.8	43.2	41.9	**44.4**
	max	48.2	48.2	47.0	48.0	51.5	51.8	52.0	52.9	41.7	41.0	40.5	41.3	47.1	47.0	46.5	**47.4**
	avg	46.6	46.4	46.5	47.4	50.7	51.3	48.2	51.2	38.1	38.0	39.3	39.7	45.1	45.2	44.7	**46.1**
Code	min	52.4	53.7	55.6	57.8	59.5	61.1	44.4	59.8	42.5	42.5	44.4	44.0	51.5	52.4	48.1	**53.9**
	max	58.3	59.2	59.0	59.7	63.2	62.6	63.0	63.1	46.6	45.9	45.6	48.8	56.0	55.9	55.9	**57.2**
	avg	56.2	56.5	57.0	58.8	61.6	61.8	55.0	61.9	44.7	44.3	45.2	47.0	54.2	54.2	52.4	**55.9**
Task Avg.	min	48.1	49.0	50.8	**52.2**	54.5	**55.8**	43.0	54.1	38.8	38.8	**41.3**	41.0	47.1	47.8	45.0	**49.1**
	max	53.2	53.7	53.0	**53.9**	57.4	57.2	57.5	**58.0**	44.2	43.5	43.0	**45.0**	51.6	51.5	51.2	**52.3**
	avg	51.4	51.5	51.8	**53.1**	56.2	**56.5**	51.6	**56.5**	41.4	41.1	42.2	**43.4**	49.7	49.7	48.5	**51.0**

bound $\delta = 10$ by about 10%. Secondly, in (b) we see that the NB-LoRA parameterization achieves very tight norm bounds, whereas with DeLoRA they are quite loose: with $\delta = 10$, the observed Frobenius norm only reached around 2.5. With δ free, the Frobenius norm grew beyond the norm bound. Panel (c) shows that NB-LoRA achieves lower loss and higher gradient norm than DeLoRA. Note that the higher gradient norm explains the apparent "offset" in panel (a) w.r.t. learning rate.

Computation Cost. Table 3 compares computational costs against two methods, PiSSA and DoRA, across different ranks on LLaMA 2-7B. Due to the extra reparameterization layer, NB-LoRA takes slightly more GPU memory and training time. NB-LoRA with the largest rank $r = 256$ still takes less computational resources than DoRA with the smallest rank $r = 2$. The main reason is that DoRA requires explicit calculation of the full adaptation matrix, which can be avoided with LoRA and NB-LoRA.



(a) Test accuracy (b) $\|W\|_F$ (c) Loss and grad norm

Figure 4: (a) Comparison of DeLoRA and NB-LoRA over a range of learning rates; (b) Frobenius norm bound (maximized over all adaptation modules); (c) loss and grad norm vs training steps.

Table 3: Computation comparison of DoRA, PiSSA and NB-LoRA with rank choice from 2 to 256. Experiments are conducted with 4 H200 GPUs.

Rank		2	4	8	16	32	64	128	256
Training Time	DoRA	24m46s	24m33s	24m03s	24m04s	24m05s	24m02s	24m18s	24m53s
	PiSSA	17m44s	17m35s	17m09s	17m10s	17m12s	17m15s	17m32s	18m09s
	NB-LoRA	18m53s	18m55s	18m26s	18m38s	18m51s	19m13s	20m15s	22m40s
Peak GPU Mem. (GB)	DoRA	102.41	102.44	102.51	102.64	102.90	103.41	104.47	106.53
	PiSSA	60.92	60.96	61.02	61.15	61.41	61.93	62.96	65.04
	NB-LoRA	60.94	60.99	61.08	61.28	61.67	62.45	64.05	67.19

6 ViT Experiments

To further explore the potential benefits of NB-LoRA, we conducted experiments in fine-tuning a vision transformer (ViT) model. In particular, we explore adaptation performance to a target dataset vs forgetting of the source (pretraining) dataset as well as hyperparameter robustness. We compare to standard LoRA and several recently-proposed methods.

Adaptation vs Forgetting The main goal of this experiment is to explore the utility of norm bounds in preventing catastrophic model forgetting McCloskey & Cohen (1989); French (1999); Wang et al. (2024). Our hypothesis is that tight control of the adaption norm will prevent loss of performance on the pre-trained model as per the analysis in Section 3, while still enabling good adaptation performance. We perform experiments (Bafghi et al., 2024) on ViT-B/16 model (Dosovitskiy et al., 2020), which is pre-trained on ImageNet-21k (Deng et al., 2009) and then fine-tuned to ImageNet-1k. For the proposed NB-LoRA, we choose the norm bound as $\delta = \gamma \|W_p\|_{S_p}$, where the ratio γ is a hyper-parameter. Similar to the setup in Kopiczko et al. (2024a), we adapt Q, V matrices and learn the classification head for the Street View House Number (SVHN) dataset. Here we report the results for NB-LoRA using nuclear norm with bound ratio of γ between 0.1 and 1.6, see Appendix F for additional results with different setups and datasets including CIFAR-100 Krizhevsky et al. (2009) and Food-101 Bossard et al. (2014).

The **metric for model forgetting** is the test accuracy of the fine-tuned model on the source dataset: ImageNet-1k, which can be compared against performance on the target dataset. As shown in Figure 5(Left), the linear adapter (i.e. just learning the classification head) avoids forgetting of the source but has poor performance on the target set. In contrast, LoRA, DoRA and PiSSA achieve high adaptation performance to the target data set, but with a dramatic loss of performance on the source data set (from around 80% to less than 10%). AdaLoRA achieves good target adaptation with less severe but still significant forgetting. VeRA and NB-LoRA can both achieve a good balance of both, but NB-LoRA outperforms in terms of both source and target performance. It can also be seen that tuning of γ allows a trade-off between source and target performance.

The middle panels of Figure 5 show the evolution of source and target accuracy vs training steps. All models (except linear) perform quite similarly in terms of adaptation to the target, whereas on the source dataset NB-LoRA (shown with $\gamma = 0.1$) maintains high accuracy throughout training,

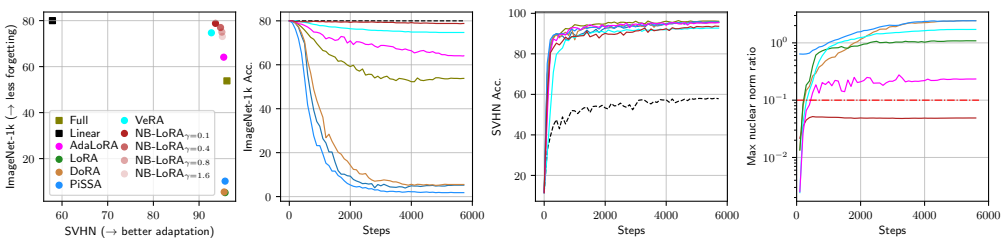


Figure 5: Analysis of Adaptation to a target data set vs forgetting of a source dataset. Left: final results for a variety of methods. Middle-Left: forgetting of the source dataset (ImageNet-1k). Middle-Right: adaptation to the target dataset (SVHN). Right: maximum nuclear norm ratio observed during training.

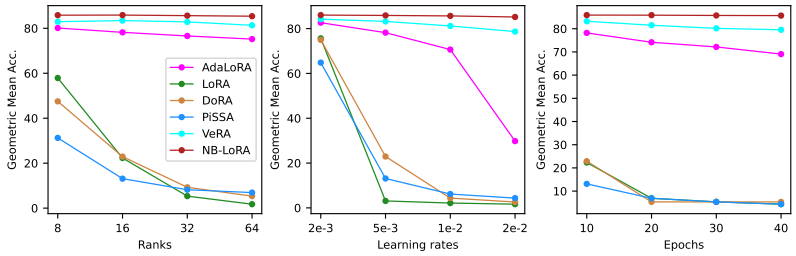


Figure 6: Analysis of hyperparameter robustness of different methods in terms of geometric mean of source (ImageNet-1k) and target (SVHN) dataset accuracies.

while most other methods quickly forget source performance. For PiSSA, DoRA, and LoRA the source performance drops significantly before target accuracy has converged, so early stopping can not solve the problem. Figure 5 (Right) plots the maximum nuclear norm ratio of the models. NB-LoRA remains below the bound, while several others are more than an order of magnitude larger.

Figure 6 shows an analysis of hyperparameter robustness of the different models, in terms of geometric mean of source and target accuracy. It can be seen that NB-LoRA is almost invariant with respect to these hyperparameters, while most other methods (except VeRA) are highly sensitive to some or all of them.

7 Limitations

Compared to LoRA, the proposed method incurrs slightly higher computational cost (peak memory and training time). In addition, it requires selection of an additional hyperparameter (the norm bound), although our experiments indicate that it improves robustness to other hyperparameters making their selection less critical.

8 Conclusion

In this paper we propose a norm-bounded low-rank adaptation (NB-LoRA) for model fine tuning. In particular, we introduce a new parameterization which is smooth and complete, i.e. it covers all matrices of a specified rank and singular value bounds, which can then be used to impose a Schatten p-norm bound (e.g. Frobenius, nuclear, spectral norm).

We argue that the proposed parameterization mitigates addresses some challenges related to the initialization of LoRA and its impact on learning rate, and can also mitigate the tendency of LoRA to forget source model performance. In experiments on fine tuning of language models, we compare to standard LoRA and other existing methods and demonstrate that NB-LoRA can substantially improve overall performance and robustness to learning rate. We showed that NB-LoRA is scalable to larger models (LLaMa-3-70B) with only a moderate computational penalty relative to standard LoRA.

REFERENCES

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

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

Reza Akbarian Bafghi, Nidhin Harilal, Claire Monteleoni, and Maziar Raissi. Parameter efficient fine-tuning of self-supervised vits without catastrophic forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3679–3684, 2024.

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

Rajendra Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.

Rajendra Bhatia and Fuad Kittaneh. On the singular values of a product of operators. *SIAM Journal on Matrix Analysis and Applications*, 11(2):272–277, 1990.

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.

Massimo Bini, Karsten Roth, Zeynep Akata, and Anna Khoreva. Ether: Efficient finetuning of large-scale models with hyperplane reflections. In *ICML*, 2024.

Massimo Bini, Leander Girrbach, and Zeynep Akata. Delora: Decoupling angles and strength in low-rank adaptation. In *International Conference on Learning Representations*, 2025.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pp. 446–461. Springer, 2014.

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. IEEE, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

Henry Gouk, Timothy Hospedales, et al. Distance-based regularisation of deep networks for fine-tuning. In *International Conference on Learning Representations*, 2021.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. The impact of initialization on loRA finetuning dynamics. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=sn3UrYRItk.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.

Jerry Yao-Chieh Hu, Maojiang Su, En-Jui Kuo, Zhao Song, and Han Liu. Computational limits of low-rank adaptation (lora) fine-tuning for transformer models. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2025.

Uijeong Jang, Jason D Lee, and Ernest K Ryu. Lora training in the ntk regime has no spurious local minima. In *International Conference on Machine Learning*, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Junsu Kim, Jaeyeon Kim, and Ernest K Ryu. Lora training provably converges to a low-rank global minimum or it fails loudly (but it probably won't fail). In *International Conference on Machine Learning*, 2025.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Elora: Efficient low-rank adaptation with random matrices. In *The Twelfth International Conference on Learning Representations*, 2024a.

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

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Vijay Lingam, Atula Tejaswi Neerkaje, Aditya Vavre, Aneesh Shetty, Gautham Krishna Gudur, Joydeep Ghosh, Eunsol Choi, Alex Dimakis, Aleksandar Bojchevski, and sujay sanghavi. SVFT: Parameter-efficient fine-tuning with singular vectors. In *2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ICML 2024)*, 2024.

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.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072, 2024.

Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. *Advances in Neural Information Processing Systems*, 36:79320–79362, 2023.

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.

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

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Ruigang Wang and Ian Manchester. Direct parameterization of lipschitz-bounded deep networks. In *ICML*, 2023.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

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.

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

A Key Technical Lemmas

Here we present some key lemmas which are used in our proofs later.

Lemma A.1. *For any $Q \in \mathbb{R}^{n\times n}$, there exists a diagonal matrix P with $P_{jj} \in \{-1, 1\}$ such that $I + PQ^\top$ is invertible.*

Proof. Let e_k, q_k be the kth column of I and Q, respectively. We construct A_k via

$$A_k^{-1} = A_{k-1}^{-1} - \frac{s_k A_{k-1}^{-1} e_k q_k^\top A_{k-1}^{-1}}{1 + s_k q_k^\top A_{k-1}^{-1} e_k}, \tag{6}$$

where $A_0 = I$ and $s_k = \operatorname{sign}\big(v_k^\top A_{k-1}^{-1} e_k\big)$ with $\operatorname{sign}(0) = 1$. From Sherman-Morrison formula, A_k is well-defined (i.e., invertible) and satisfies $A_k = A_{k-1} + s_k e_k q_k^\top$. By taking $P = \operatorname{diag}(s_1, \ldots, s_n)$, we have $A_n = I + \sum_{k=1}^n s_k e_k q_k^\top = I + PQ^\top$ is also invertible. □

Lemma A.2. *Let $G \in \mathbb{R}^{r\times r}$ and $H \in \mathbb{R}^{s\times r}$ such that $G^\top G + H^\top H = I$. Then,*

$$\begin{bmatrix} G \\ H \end{bmatrix} = \operatorname{Cayley}\left(\begin{bmatrix} X \\ Y \end{bmatrix}\right) = \begin{bmatrix} (I - Z)(I + Z)^{-1} \\ -2Y(I + Z)^{-1} \end{bmatrix} \tag{7}$$

for some $X \in \mathbb{R}^{r\times r}$ and $Y \in \mathbb{R}^{s\times r}$ if and only if $I + G$ is invertible.

Proof. From the Cayley transformation (4) we have the following relationships:

$$G = (I - Z)(I + Z)^{-1}, \quad H = -2Y(I + Z)^{-1}, \quad Z = X - X^\top + Y^\top Y. \tag{8}$$

(**if**). From the above equation we have $I + G = (I + Z)^{-1}$ invertible.

(**only if**). The proof is constructive, i.e., finding $X, Z \in \mathbb{R}^{r\times r}$ and $Y \in \mathbb{R}^{s\times r}$ satisfying (8). We consider a candidate solution as follows:

$$Z = (I + G)^{-1}(I - G), \quad Y = -\frac{1}{2}H(I + Z), \quad X = \frac{1}{2}Z. \tag{9}$$

It is easy to check that the above solution satisfies the first two equations in (8). We now verify the last equation as follows:

$$\begin{aligned}
&Z + X^\top - X - Y^\top Y = \frac{1}{2}(Z + Z^\top) - Y^\top Y \\
=&\frac{1}{2}[(I + G)^{-1}(I - G) + (I - G^\top)(I + G^\top)^{-1}] - (I + G^\top)^{-1}H^\top H(I + G)^{-1} \\
=&\frac{1}{2}[(I - G)(I + G)^{-1} + (I + G^\top)^{-1}(I - G^\top)] - (I + G^\top)^{-1}H^\top H(I + G)^{-1} \\
=&\frac{1}{2}(I + G^\top)^{-1}[(I + G^\top)(I - G) + (I - G^\top)(I + G) - 2H^\top H](I + G)^{-1} \\
=&(I + G^\top)^{-1}[I - G^\top G - H^\top H](I + G)^{-1} = 0,
\end{aligned}$$

where the second line is due to that $(I + G)^{-1}$ and $(I - G)$ are commutative. □

Lemma A.3. *Let $A \in \mathbb{R}^{r\times m}$ and $B \in \mathbb{R}^{r\times n}$ with $AA^\top + BB^\top = I$. Then, there exist a diagonal matrix $P \in \mathbb{R}^{r\times r}$ with $P_{jj} \in \{-1, 1\}$ and $\tilde{A} \in \mathbb{R}^{r\times m}, \tilde{B} \in \mathbb{R}^{r\times n}$ satisfying*

$$[PA \quad PB]^\top = \text{Cayley}\left(\begin{bmatrix}\tilde{A} & \tilde{B}\end{bmatrix}^\top\right). \tag{10}$$

Proof. From the assumption we have that $\begin{bmatrix}A^\top \\ B^\top\end{bmatrix}$ is a tall matrix, i.e., $r \leqslant m + n$. We then take the partition $\begin{bmatrix}A^\top \\ B^\top\end{bmatrix} = \begin{bmatrix}\bar{G} \\ \bar{H}\end{bmatrix}$ with $\bar{G} \in \mathbb{R}^{r\times r}$ and $\bar{H} \in \mathbb{R}^{(m+n-r)\times r}$. We introduce $G = \bar{G}P$ and $H = \bar{H}P$, where P is a diagonal matrix with $P_{jj} \in \{-1, 1\}$. Then, we can obtain

$$G^\top G + H^\top H = P(\bar{G}^\top\bar{G} + \bar{H}^\top\bar{H})P = P(AA^\top + BB^\top)P = P^2 = I$$

for all diagonal such P. From Theorem A.1, we can pick a particular P such that $I + G = I + \bar{G}P$ is invertible. We then follow Theorem A.2 to compute $X \in \mathbb{R}^{r\times r}$ and $Y \in \mathbb{R}(m + n - r) \times r$ satisfying (7). Finally, we take the partition $\begin{bmatrix}X^\top & Y^\top\end{bmatrix} = \begin{bmatrix}\tilde{A} & \tilde{B}\end{bmatrix}$. □

B Proof of Theorem 4.2

The proof includes two parts: I) $W = \mathcal{W}(\tilde{A}, \tilde{B}) \in \mathbb{W}_S$ for any $\tilde{A} \in \mathbb{R}^{r\times m}$ and $\tilde{B} \in \mathbb{R}^{r\times n}$; II) for any $W \in \mathbb{W}_S$, there exists a pair of $\tilde{A} \in \mathbb{R}^{r\times m}$ and $\tilde{B} \in \mathbb{R}^{r\times n}$ such that $W = \mathcal{W}(\tilde{A}, \tilde{B})$.

Part I It is obvious that $\text{rank}(W) \leqslant r$. The jth singular value of W satisfies

$$\sigma_j(W) = 2\sigma_j(\underbrace{B^\top S^{\frac{1}{2}}}_{Q^\top}\underbrace{S^{\frac{1}{2}}A}_{K}) \leqslant \sigma_j\left(QQ^\top + KK^\top\right) = \sigma_j(S^{\frac{1}{2}}(\underbrace{AA^\top + BB^\top}_{I})S^{\frac{1}{2}}) = \sigma_j(S) \tag{11}$$

where the inequality is the matrix arithmetic-geometric mean inequality (Bhatia & Kittaneh, 1990; Bhatia, 2013), and the last equality follows by the Cayley transformation.

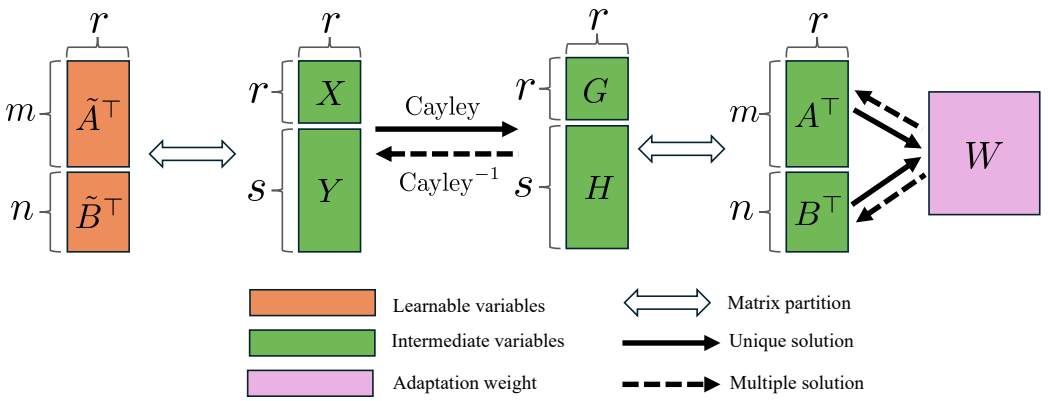


Figure 7: Diagram of NB-LoRA parameterization.

Part II Without loss of generality, we assume that the diagonal elements of S is in descending order, i.e., $\sigma_j(S) = S_{jj}$ for $j = 1, \ldots, r$. Since W has maximally r non-zero singular values, we can take the reduced SVD decomposition $W = U_w \Sigma_w V_w^\top$ where $U_w \in \mathbb{R}^{m\times r}, V_w \in \mathbb{R}^{n\times r}$ are semi-orthogonal, and the positive diagonal matrix $S_w \in \mathbb{R}^{r\times r}$. We now consider the following candidates for A, B:

$$A = P\Sigma_a V_w^\top, \quad B = P\Sigma_b U_w^\top, \tag{12}$$

where $P \in \mathbb{R}^{r\times r}$ is a diagonal matrix with $P_{jj} \in \{-1, 1\}$, and $\Sigma_a, \Sigma_b \in \mathbb{R}^{r\times r}$ are positive diagonal matrices. The first constraint for A and B is that $[A \quad B]^\top$ is semi-orthogonal since it is an output of the Cayley transformation. Thus, we have

$$I = AA^\top + BB^\top = P(\Sigma_a^2 + \Sigma_b^2)P^\top \implies \Sigma_a^2 + \Sigma_b^2 = I. \tag{13}$$

The second constraint for A, B is $W = 2B^\top SA$, which implies

$$U_w\Sigma_w V_w^\top = 2U_w\Sigma_a P^\top S P \Sigma_b V_w^\top = U_w(2\Sigma_a\Sigma_b S)V_w^\top \implies 2\Sigma_a\Sigma_b = \Sigma_w S^{-1} \tag{14}$$

Eq. (13) and (14) yield a solution of

$$\Sigma_a = \frac{\sqrt{I+J} + \sqrt{I-J}}{2}, \ \Sigma_b = \frac{\sqrt{I+J} - \sqrt{I-J}}{2}. \tag{15}$$

where $J = \Sigma_w S^{-1}$ satisfies $0 \preceq J \preceq I$ since $S_w \preceq S$ for $W \in \mathbb{W}_S$. Note that we need to deal with the case where S is not full rank, i.e., there exists an $k < r$ such that $S_{kk} = 0$ and $S_{k-1,k-1} > 0$. Since $0 \preceq \Sigma_w \preceq S$, we have $\Sigma_{ii} = 0$ for all $i \geqslant k$ and simply take $J_{ii} = 1$. It is easy to verify that Equations (13) - (15) still hold. Finally, Theorem A.3 shows that we can recover $\tilde{A}, \tilde{B}$ from A, B by picking a proper P in (12) based on Theorem A.1.

C Custom Backward for Cayley Transformation

We first rewrite the forward computation of Cayley transformation $(X, Y) \to (G, H)$ as follows:

$$Z = X - X^\top + Y^\top Y, \quad M = I + Z, \quad W = M^{-1}, \quad G = (I - Z)W, \quad H = -2YW \tag{16}$$

where $G, X, Z, M, W \in \mathbb{R}^{r\times r}$ and $H, Y \in \mathbb{R}^{s\times r}$. We provide a custom backward $(\nabla_G, \nabla_H) \to (\nabla_X, \nabla_Y)$ with $\nabla_A = (\partial\ell/\partial A)^\top$ for (16) as follows:

$$\begin{aligned} \begin{bmatrix}\tilde{\nabla}_G \\ \tilde{\nabla}_H\end{bmatrix} &= \begin{bmatrix}\nabla_G \\ \nabla_H\end{bmatrix} W^\top, \quad S_Z = \begin{bmatrix}I+G \\ H\end{bmatrix}^\top \begin{bmatrix}\tilde{\nabla}_G \\ \tilde{\nabla}_H\end{bmatrix}, \\ \nabla_X &= S_Z^\top - S_Z, \quad \nabla_Y = -\frac{1}{2}HM(S_Z^\top + S_Z) - 2\tilde{\nabla}_H, \end{aligned} \tag{17}$$

where W, M, G, H can be reused from the Cayley forward step. Note that when applying AutoDiff to (16), it is necessary to store the input X, Y, output G, H as well as some intermediate steps, which

Table 4: Computation comparison for training NB-LoRA with AutoDiff and custom backward step.

Method	Peak Mem.	Train Time	GSM8K Acc.
AutoDiff	69.52GB	23m51s	58.0
Custom	67.19GB	22m40s	57.8

requires more memory than our custom backward step (17) since $Y \in \mathbb{R}^{s\times r}$ is much larger than $W, M \in \mathbb{R}^{r\times r}$. In our approach, we can recover Y from other stored variables, i.e., $Y = -\frac{1}{2}HM$. To give detail derivation for (17), we first differentiate the forward step (16):

$$\begin{aligned} &\mathrm{d}Z = \mathrm{d}X - \mathrm{d}X^\top + Y^\top \mathrm{d}Y + \mathrm{d}Y^\top Y, \quad \mathrm{d}W = -W\mathrm{d}ZW, \\ &\mathrm{d}G = -\mathrm{d}ZW + (I - Z)\mathrm{d}W, \quad \mathrm{d}H = -2\mathrm{d}YW - 2Y\mathrm{d}W. \end{aligned} \tag{18}$$

The differential of loss function $\mathrm{d}\ell$ satisfies

$$\mathrm{d}\ell = \mathrm{Tr}\left(\nabla_X^\top \mathrm{d}X\right) + \mathrm{Tr}\left(\nabla_Y^\top \mathrm{d}Y\right) = \mathrm{Tr}\left(\nabla_G^\top \mathrm{d}G\right) + \mathrm{Tr}\big(\nabla_H^\top \mathrm{d}H\big). \tag{19}$$

By further substituting (18) into (19), we have

$$\begin{aligned} &\mathrm{Tr}\big(\nabla_G^\top \mathrm{d}G\big) + \mathrm{Tr}\big(\nabla_H^\top \mathrm{d}H\big) \\ &\quad = -\mathrm{Tr}\big(\nabla_G^\top \mathrm{d}ZW + \nabla_G^\top (I - Z)W\mathrm{d}ZW\big) - 2\,\mathrm{Tr}\big(\nabla_H^\top \mathrm{d}YW - \nabla_H^\top YW\mathrm{d}ZW\big) \\ &\quad = -\mathrm{Tr}\big(W(\nabla_G^\top + \nabla_G^\top (I - Z)W - 2\nabla_H^\top YW)\mathrm{d}Z\big) - \mathrm{Tr}\big(2W\nabla_H^\top \mathrm{d}Y\big) \\ &\quad = -\mathrm{Tr}\big(W(\nabla_G^\top + \nabla_G^\top G + \nabla_H^\top H)\mathrm{d}Z\big) - \mathrm{Tr}\big(2W\nabla_H^\top \mathrm{d}Y\big) \\ &\quad = -\mathrm{Tr}\big(S_Z^\top \mathrm{d}Z\big) - \mathrm{Tr}\big(2W\nabla_H^\top \mathrm{d}Y\big) \\ &\quad = -\mathrm{Tr}\big((S_Z^\top - S_Z)dX\big) - \mathrm{Tr}\big(((S_Z + S_Z^\top)Y^\top + 2W\nabla_H^\top)\mathrm{d}Y\big) = \mathrm{Tr}\left(\nabla_X^\top \mathrm{d}X\right) + \mathrm{Tr}\left(\nabla_Y^\top \mathrm{d}Y\right) \end{aligned}$$

which yields the custom backward step (17) by substituting $Y = -\frac{1}{2}HM$. As shown in Table 4, the custom backward pass can save both GPU memory and training time.

D Connections between DeLoRA and NB-LoRA

DeLoRA (Bini et al., 2025) is a fine-tuning method which can control both rank and Frobenius norm bound of weight adaptation W. Specifically, DeLoRA takes the form of

$$W = \frac{\delta}{r} B^\top \Xi A := \frac{\delta}{r} B^\top \,\mathrm{diag}(|b_i|_2 \cdot |a_i|_2) A \tag{20}$$

where a_i, b_i are the ith row of $A \in \mathbb{R}^{r\times n}$ and $B \in \mathbb{R}^{r\times m}$, respectively. The above parameterization can be rewritten as sum of NB-LoRA matrices with both rank and norm bound of 1:

$$W = \frac{\delta}{r} \sum_{i=1}^{r} 2 \left(\frac{b_i}{\sqrt{2}|b_i|_2}\right)^\top \left(\frac{a_i}{\sqrt{2}|a_i|_2}\right) = \frac{\delta}{r} \sum_{i=1}^{r} 2\bar{b}_i^\top \bar{a}_i = \frac{\delta}{r} \sum_{i=1}^{r} \bar{W}_i,$$

where $[\,\bar{a}_i\ \bar{b}_i\,]$ is a set of of decoupled unit vectors. By Theorem 4.2 we have that $\|\bar{W}_i\|_F \leqslant 1$ and $\|W\|_F \leqslant \delta/r \sum_{i=1}^{r} \|\bar{W}_i\|_F \leqslant \delta$. NB-LoRA in (3) also has a similar representation:

$$W = 2B^\top SA = \sum_{i=1}^{r} s_i (2\hat{b}_i^\top \hat{a}_i) = \sum_{i=1}^{r} s_i \hat{W}_i.$$

Different from DeLoRA, $[\,\hat{a}_i\ \hat{b}_i\,]$ is a set of coupled unit vectors as they are orthogonal to each other. This coupling behavior allows us to specify the bound for each singular value of W, providing tight control of a wide family of matrix norms. Another main difference is model initialization. Since it is not straightforward to initialize A, B satisfying $W = 0$ for (20), the residual-type initialization (Meng et al., 2024) is adopted, resulting a smaller reachable set than NB-LoRA when an explicit bound is specified, see detailed discussion in Section 4.

Table 5: Summary of incremental design choices from LoRA to NB-LoRA.

Design choice	Method	W formulation
	LoRA	$\frac{\alpha}{r} B^\top A$
+(Cayley transform)	NB-LoRA with $\|W\|_{S_\infty} \leqslant \delta$	$2\delta B^\top A$ with $(A, B) = \text{Cayley}(\tilde{A}, \tilde{B})$
+(learnable scaling)	NB-LoRA with $\|W\|_{S_p} \leqslant \delta$	$2\delta B^\top SA$ with $S = \text{diag}(s)$ and $\|s\|_p \leqslant \delta$

Table 6: We report the GSM8K accuracy for ablation of NB-LoRA on fine-tuning LLaMA-2-7B models with different ranks and learning rates.

Rank	Method	Learning Rate		
		1e-4	5e-4	1e-3
128	LoRA	52.8	58.3	failed
	NB-LoRA (spectral)	57.7	**60.5**	60.0
	NB-LoRA (nuclear)	57.8	59.7	59.2
16	LoRA	43.2	55.8	57.5
	NB-LoRA (spectral)	47.9	55.3	56.5
	NB-LoRA (nuclear)	49.4	**56.8**	55.6

E LLM Experimental Details and Additional Results

Training Details. In our LLM experiments, we use the same training setup as Meng et al. (2024); Taori et al. (2023), i.e., AdamW Loshchilov & Hutter (2019) with no weight decay. We use the cosine annealing scheduler with a warm-up ratio of 0.03. The default batch size is 128. We ensure $\alpha = r$ for all adapters, although NB-LoRA does not use this parameter. We choose the norm bound of $\delta = r$ with nuclear norm, which results in the same scaling factor as the other adapters. When Frobenius or spectral norm is used, we set the default bound as $\delta = \sqrt{r}$ and $\delta = 1$, respectively, which also results in the same scaling factor as other adapters. We set `lora_dropout` to 0, and insert the adapters into all linear layers of the base model. We use BFloat16 for both the base model and the adapters.

Ablation of NB-LoRA Design Choice. We summarize the incremental design choices that transform LoRA into NB-LoRA in Table 5. We conduct an ablation study on LLaMA-2-7B fine-tuning in Table 6. For a large rank ($r = 128$), NB-LoRA with spectral norm bound achieves slightly better performance, whereas the nuclear norm performs better under a low-rank budget. Both methods yield more robust performance compared to LoRA.

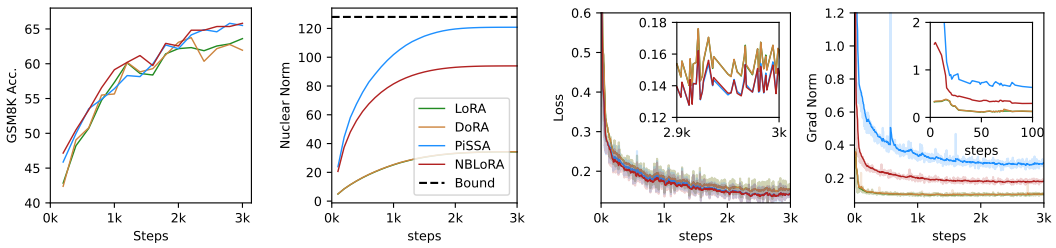
Robust Performance for Prolong Training. We conduct a full epoch training of LLaMA-2-7B on the MetaMathQA dataset. The learning rates are chosen to be 1e-4 and 5e-4, which achieve good performance for different adapters in short horizon training (see Figure 3). We can observe that NB-LoRA consistently outperforms other methods. In particular, NB-LoRA is more stable for the large learning rate, due to the norm saturation on weight adaptation. Meanwhile, DoRA depicts unstable training and PiSSA has poor performance due to excessive increase in weight norm.

Experiments on Various Ranks. Figure 9 explores the impact of rank on LoRA, DoRA, PiSSA and NB-LoRA with learning rate of 1e-4. Under the setup, PiSSA achieves the best GSM8K accuracy. As the rank decrease, the gap between NB-LoRA and PiSSA narrows. And NB-LoRA outperforms PiSSA for low ranks when $r < 16$. NB-LoRA outperforms LoRA and DoRA by approximately 5% across all ranks. We also examine the effect of varying learning rates at rank 16, demonstrating robustness to learning rate choices across different ranks.

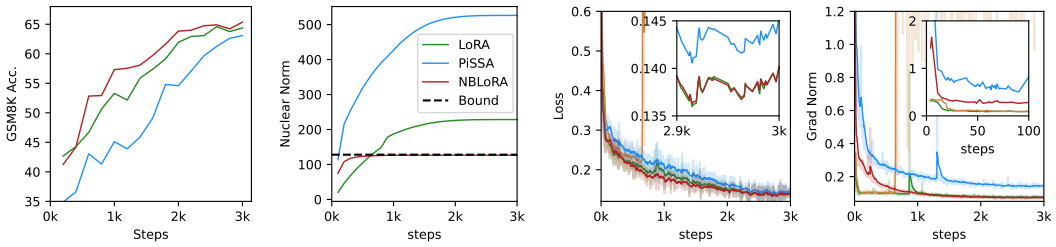
Addition Computation Comparison between DoRA and NB-LoRA. We first report the forward computation time of key operations in DoRA and NB-LoRA in Table 7, showing that inverting a small low rank matrix is much computationally cheaper than computing a large low-ran weight matrix.

Table 7: Computation time (μs) of the rank-r matrix $B^\top A \in \mathbb{R}^{m\times n}$ in DoRA and $M^{-1} \in \mathbb{R}^{r\times r}$ in NB-LoRA. We use $m = 4096$, $n = 4m$ and rank r from 2 to 256. Computation time is measured based on 500 samples with 500 warm-up steps on RTX4090.

Matrix Operation	2	4	8	16	32	64	128	256
$B^\top A \in \mathbb{R}^{m\times n}$	314.1±2.5	311.9±1.6	312.1±2.4	310.7±2.0	288.3±2.0	323.9±1.9	421.9±2.0	792.6±1.1
$M^{-1} \in \mathbb{R}^{r\times r}$	29.2±0.9	30.7±0.9	35.1±1.4	48.3±0.9	72.9±1.1	97.0±2.0	169.7±1.4	361.3±1.7



(a) Test accuracy (lr=1e-4) (b) $\|W\|_{S_1}$ (lr=1e-4) (c) Loss and grad norm (lr=1e-4)



(d) Test accuracy (lr=5e-4) (e) $\|W\|_{S_1}$ (lr=5e-4) (f) Loss and grad norm (lr=5e-4)

Figure 8: The evaluation accuracy, the nuclear norm bound, loss and grad norm over a full training epoch on MetaMathQA. The norm bound is computed by maximizing over all adaptation modules.

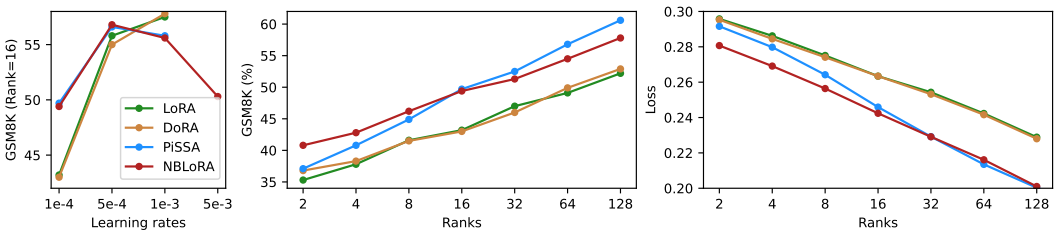


Figure 9: Comparison among LoRA, DoRA, PiSSA and NB-LoRA across ranks from 2 to 128. We also test the learning rate robustness for the case with $r = 16$.

F ViT Experiments

Training Details. A similar ViT fine-tuning experiments for the model forgetting issue can be found in Bafghi et al. (2024). We take the ViT-B/16 model Dosovitskiy et al. (2020) and insert adaption blocks into the Q, V matrices Kopiczko et al. (2024a). We choose AdamW Loshchilov & Hutter (2019) as the optimizer with default learning rate of 5e-3 and weight decay of 0.01. For the full fine-tuning, we reduce the learning rate to 5e-4. We take one-cycle learning rate scheduler with warm-up ratio of 0.1. We use batch size of 128 for SVHN dataset and 256 for CIFAR-100 and Food-101 dataset.

Extra results. We report the ViT examples with different target datasets: CIFAR-100 and Food-101 in Figure 10. A similar conclusion as the SVHN experiment can be drawn from two datasets.

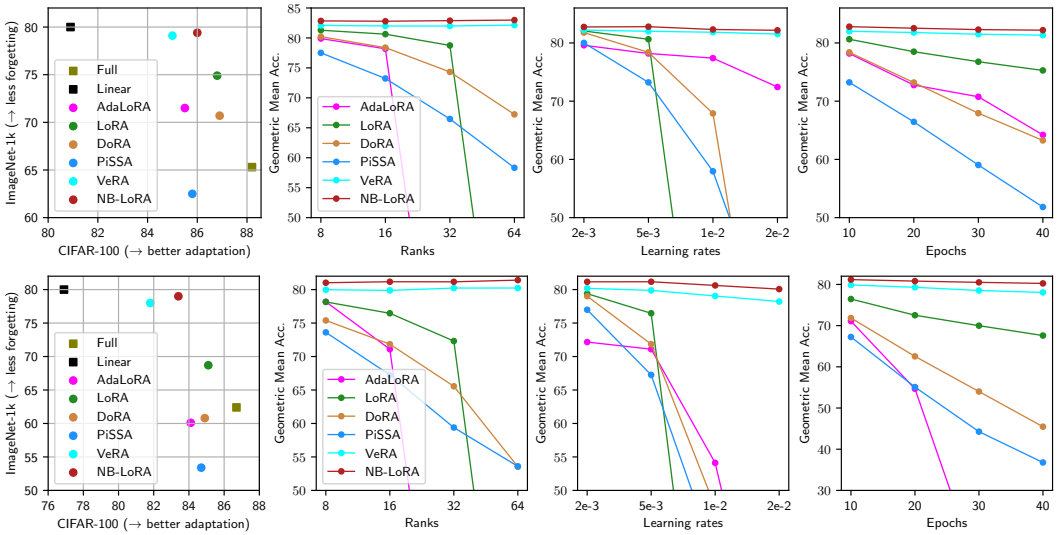


Figure 10: Geometric mean of CIFAR-100 (top) and Food-101 (bottom) with different adapters on various of hyper-parameter setup.