On the Convergence Rate of LoRA Gradient Descent

Anonymous Authors[1]

Abstract

The low-rank adaptation (LoRA) algorithm for fine-tuning large models has grown popular in recent years due to its remarkable performance and low computational requirements. LoRA trains two "adapter" matrices that form a low-rank representation of the model parameters, thereby massively reducing the number of parameters that need to be updated at every step. Although LoRA is simple, its convergence is poorly understood due to the lack of Lipschitz smoothness, a key condition for classic convergence analyses. As a result, current theoretical results only consider asymptotic behavior or assume strong boundedness conditions which artificially enforce Lipschitz smoothness. In this work, we provide for the first time a non-asymptotic convergence analysis of the *original LoRA gradient descent* algorithm, which reflects widespread practice, without such assumptions. Our work relies on three key steps: i) reformulating the problem in terms of the outer product of the stacked adapter matrices, ii) a modified descent lemma for the "Lipschitz-like" reparametrized function, and iii) controlling the step size. With this approach, we prove that LoRA gradient descent converges to a stationary point at rate $O(\frac{1}{\log T})$, where T is the number of iterations. We conduct numerical experiments to validate our theoretical findings.

1. Introduction

Modern applications of large language models (LLMs) typically involve self-supervised pretraining of a large foundation model, which is later fine-tuned on a smaller task-specific dataset through supervised learning. Parameter efficient fine-tuning methods, which only train a small number of additional parameters, can efficiently adapt LLMs to a multitude of downstream tasks while maintaining comparable performance to fine-tuning of the full model parameters. One highly popular method is LoRA, or Low-Rank Adaptation (Hu et al., 2022), which trains a low-rank representation of the model parameters. Specifically, once the pretraining stage yields a model weight matrix W_0, LoRA only updates low-rank *adapter* matrices A, B, such that the final weights are $W_0 + BA$. The matrices A and B can be updated with any optimization method, including gradient descent. However, even this simple algorithm is challenging to analyze; as observed in (Sun et al., 2024) and (Malinovsky et al., 2024), even if the original loss function is Lipschitz smooth, the *new* loss function— reparametrized in terms of A and B— is not. This nonsmoothness is a major obstacle to applying classic convergence proof techniques of gradient methods to LoRA.

As a result, current theoretical analyses of LoRA fall into one of three categories. First, there are several works that examine LoRA in infinite regimes, such as its asymptotic convergence or its behavior on infinitely wide neural networks (Kim et al., 2025; Jang et al., 2024; Zeng & Lee, 2024; Yaras et al., 2024; Zhang & Pilanci, 2024; Hayou et al., 2024). However, these works do not establish a non-asymptotic convergence rate for finite models. Second, many works propose and analyze memory-efficient algorithms that resemble but do not address the original LoRA algorithm. These include GaLore (Zhao et al., 2024) and LoRA variants that only update a single adapter matrix at a time (Malinovsky et al., 2024; Sokolov et al., 2025). Third, some recent works analyze algorithmic frameworks that *do* extend to LoRA, and *do* derive a convergence rate (Jiang et al., 2024; Sun et al., 2024; Ghiasvand et al., 2025). However, these works require that the adapter matrices A and B are uniformly upper bounded by some constants which appear in the final convergence bound. These restrictive assumptions artificially enforce Lipschitz smoothness within the new parametrization, essentially reducing the problem to standard gradient descent with no substantially new proof techniques. Considering the existing literature, we seek to answer the question:

How fast does the original LoRA algorithm converge?

In this work, we close the above research gap by es-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

tablishing the first non-asymptotic convergence analysis of LoRA gradient descent, without requiring asynchronous updates, bounded adapter matrices, or Lipschitz smoothness of the reparametrized loss function. We only assume that the *original* loss function is Lipschitz smooth. Our novel proof approach involves stacking B and A into a single matrix V and deriving a modified "Lipschitz-like" descent lemma for gradient descent with respect to V. We prove that to achieve descent at each step, the algorithm requires a step size that is "small enough" with respect to the norm of the current parameters and gradient. This recursive relationship between the parameters, gradient, and step size introduces instability in the convergence, resulting in an $O(\frac{1}{\log T})$ convergence rate to a stationary point. If we further assume that the adapter norms are bounded, we can recover the classic $O(\frac{1}{T})$ convergence rate of gradient descent on Lipschitz smooth functions.

By analyzing LoRA under minimal assumptions, we can make rigorous observations about its behavior. First, the convergence rate of LoRA does not (and should not) depend on the chosen rank, since gradient descent is a dimension-free algorithm. Second, our results shed light onto the complex relationship between the parameter norm and the LoRA training dynamics, which has been alluded to in past work. We theoretically demonstrate that LoRA displays a "position dependency," in that the convergence is slowed if the iterates are moving away from the origin and accelerated if they are moving towards the origin. This is due to the major structural changes introduced by the LoRA reparametrization, including the creation of a stationary point at the origin *regardless* of the original loss function.

Motivated by our theoretical insights, we perform experiments to examine the impact of step size choice under the LoRA reparametrization in practice. We train logistic regression and ResNet-18 models on the CIFAR-10 images dataset with *adaptive* and *normalized* learning rates that directly stem from theory, and we compare their performance against constant learning rates of similar values. We find that adjusting the learning rate based on the gradient norm or parameter norm can accelerate and stabilize training by accounting for the unique structure induced by the low-rank reparametrization.

Our contributions can be summarized as follows:

- We prove for the first time that LoRA gradient descent converges to a stationary point at rate $O(\frac{1}{\log T})$, only assuming that the *original loss function* is Lipschitz smooth and lower bounded.
- We validate our theoretical results with experiments, and we propose and test practical new approaches for learning rate selection for LoRA.

In Section 2, we review the related work in detail and contextualize our contribution. In Section 3, we provide our convergence analysis, establishing preliminaries in Section 3.1, providing our main results in Section 3.2 and discussion in 3.3. Finally, in Section 4, we provide the results of our experiments.

2. Related Work

The LoRA algorithm, first proposed in (Hu et al., 2022), is motivated by the idea of a viable "low-rank representation" of deep neural networks (Oymak et al., 2019; Li et al., 2018). In particular, (Aghajanyan et al., 2021) argues that large overparametrized models reside on a lower "intrinsic dimension," and fine-tuning with a low dimension reparametrization, such as a random subset of the weight matrices, can yield strong performance while minimizing computational expenses.

LoRA is simple, popular, and empirically effective for fine-tuning large language models. Many variants have been proposed to improve the initialization, performance, memory efficiency, or privacy of the algorithm (Shen, 2025; Büyükakyüz, 2024; Li et al., 2025; Meng et al., 2024; Wang et al., 2024), including but not limited to federated LoRA (Yang et al., 2025), quantized LoRA (QLoRA) (Dettmers et al., 2023), and ReLoRA (Lialin et al., 2024). While LoRA performs well in practice, its behavior in comparison to full-rank training is poorly understood, with empirical studies showing that the two approaches can produce completely different solutions (Shuttleworth et al., 2025; Biderman et al., 2024). Consequently, the theoretic properties of LoRA are highly relevant for understanding the strengths and weakness of this method.

As highlighted in the introduction, the theory of LoRA convergence has been tackled from many angles. First, many works examine the behavior of LoRA in infinite regimes, whether by characterizing the kinds of solutions LoRA converges to at infinity or analyzing its convergence in the neural tangent kernel (NTK) regime, which defines the linearized training dynamics of an infinitely wide neural network (Malladi et al., 2023). For example, (Kim et al., 2025) argues that LoRA converges to a low-rank global minimum with high probability, assuming that one exists. The work (Zeng & Lee, 2024) studies the expressive power of the low-rank solutions achieved by LoRA. The work (Jang et al., 2024) shows that LoRA has no spurious local minima in the NTK regime. The LoRA+ (Hayou et al., 2024) algorithm, which updates the adapter matrices with differently scaled learning rates, is motivated by an NTK convergence analysis. Finally, the gradient flow dynamics of LoRA have been studied specifically for matrix factorization (Xu et al., 2025). These works do not establish a convergence rate for discrete-time LoRA for general functions.

Second, there are memory-efficient LoRA-like algorithms with concrete convergence analyses that do not apply to the original LoRA formulation. These include GaLore, an algorithm which projects the *gradient* into a low-rank subspace but maintains a full-rank update of the parameters (Zhao et al., 2024), LDAdam (Robert et al., 2025), which performs adaptive optimization steps within lower dimensional subspaces, and Randomized Subspace Optimization (RSO) (Chen et al., 2025). The LoRA variant LoRA-One incorporates information from the full gradient at initialization (Zhang et al., 2025), and it is analyzed only for Gaussian data and mean-squared loss. Finally, the variants RAC-LoRA (Malinovsky et al., 2024) and Bernoulli-LoRA (Sokolov et al., 2025) only train one adapter matrix while freezing the other as a fixed projection matrix, essentially preserving Lipschitz smoothness while failing to capture the complex training dynamics of actual LoRA implementations. In practice, the LoRA matrices are updated *simultaneously*, causing nonlinearity and nonsmoothness in the optimized variables. As the original LoRA formulation is much more commonly used in practice, these analyses do not address realistic implementations.

Third, several works prove the convergence of LoRA under a strong boundedness assumption that artificially enforces Lipschitz smoothness. This includes analyses of federated LoRA algorithms (Sun et al., 2024; Park & Klabjan, 2025; Ghiasvand et al., 2025), for which single-device LoRA is a special case. These works assume that the adapter matrix norms are uniformly bounded by constants. In addition, (Jiang et al., 2024) analyzes convergence of LoRA and representation fine-tuning (ReFT) within the same framework, assuming that the singular values of A and B are uniformly upper bounded, which upper bounds the matrix norms by rank-dependent constants that appear in the final convergence result. They also require that the gradient is uniformly bounded. These assumptions are overly restrictive and unrealistic, preventing a robust understanding of LoRA.

Our work closes the gap in the existing literature by deriving the explicit convergence rate of the *original* LoRA algorithm for general models, without requiring bounded parameter norms or Lipschitz smoothness of the reparametrized loss function.

3. Convergence Analysis

3.1. Preliminaries

For real-valued matrices $M, N \in \mathbb{R}^{m\times n}$, we denote the Frobenius inner product $\langle\cdot,\cdot\rangle_F$ as

$$\langle M, N\rangle_F = \sum_{i,j} m_{ij}n_{ij} = Tr(M^T N),$$

and the Frobenius matrix norm $\|\cdot\|_F$ as

$$\|M\|_F = \sqrt{\sum_{i,j} |m_{ij}|^2} = \sqrt{Tr(M^T M)}.$$

The Frobenius inner product and norm for matrices are analogous to the Euclidean inner product and norm for vectors. To simplify notation, we also use $\|\cdot\|$ to denote the Frobenius matrix norm.

We can characterize fine-tuning as the model-agnostic minimization problem

$$\min_{W\in\mathbb{R}^{m\times n}} \ell(W_0 + W), \tag{1}$$

where $\ell : \mathbb{R}^{m\times n} \to \mathbb{R}$ represents the loss function, $W_0 \in \mathbb{R}^{m\times n}$ represents a frozen (pretrained) weight matrix, and $W \in \mathbb{R}^{m\times n}$ (sometimes denoted as ΔW) represents the update to the frozen weights after fine-tuning. In the LoRA algorithm, we parametrize $W = BA$, where $B \in \mathbb{R}^{m\times r}$, $A \in \mathbb{R}^{r\times n}$ are the low-rank *adapter* matrices with rank $r < \min\{m, n\}$. During finetuning, only A and B are optimized while W_0 remains fixed. To simplify notation, we can reformulate (1) with the function $\mathcal{L}(W) = \ell(W_0 + W)$ to obtain the LoRA minimization problem,

$$\min_{B\in\mathbb{R}^{m\times r}, A\in\mathbb{R}^{r\times n}} \mathcal{L}(BA). \tag{2}$$

We herein call $\mathcal{L} : \mathbb{R}^{m\times n} \to \mathbb{R}$ the *original loss function*. While any kind of optimization can be applied to (2), in this work we consider the prevailing case of LoRA gradient descent, where at time step t the matrices A, B are updated simultaneously as follows,

$$\begin{aligned} A_{t+1} &= A_t - \eta_t \nabla_A \mathcal{L}(B_t A_t), \\ B_{t+1} &= B_t - \eta_t \nabla_B \mathcal{L}(B_t A_t). \end{aligned} \tag{3}$$

We require the following standard assumptions for our analysis.

Assumption 3.1 (Lipschitz smoothness)**.** The original loss function $\mathcal{L} : \mathbb{R}^{m\times n} \to \mathbb{R}$ is differentiable, and there exists a constant $L \geq 1$ such that for all $W, W' \in \mathbb{R}^{m\times n}$,

$$\left\|\nabla\mathcal{L}(W) - \nabla\mathcal{L}(W')\right\|_F \leq L\left\|W - W'\right\|_F. \tag{4}$$

Equivalently, $\mathcal{L}$ satisfies the following *descent lemma*,

$$\mathcal{L}(W') \leq \mathcal{L}(W) + \langle\nabla\mathcal{L}(W), W' - W\rangle_F + \frac{L}{2}\|W - W'\|_F^2. \tag{5}$$

Assumption 3.2. The original loss function $\mathcal{L}$ is lower bounded by a constant $\mathcal{L}^*$ such that for all $W \in \mathbb{R}^{m\times n}$, $\mathcal{L}(W) \geq \mathcal{L}^*$.

Assumptions 3.1 and 3.2 are fundamental to the study of gradient descent convergence. Critically, even with these assumptions, $\mathcal{L}$ is *not* Lipschitz smooth in B or A, preventing the analysis of (3) via classic optimization techniques.

3.2. Results

In the following, we detail three steps for proving convergence of LoRA. First, we reformulate the problem (2) as optimization over a single variable V that contains the stacked matrices B and A^T. We can rewrite the loss function in terms of the outer product VV^T. Second, we derive a modified descent lemma, analogous to the descent lemma for Lipschitz smooth functions (5), that enables descent in one step as long as the step size is small enough with respect to the parameter norm $\|V\|$ and gradient norm. The last step of the proof is ensuring that the step size does not decrease too quickly, thereby guaranteeing convergence.

Step 1: Restructure problem into VV^T form. We stack the adapter matrices A^T, B, into a single variable $V \in \mathbb{R}^{(m+n)\times r}$. Then the outer product VV^T contains BA as follows,

$$V = \begin{bmatrix} B \\ A^T \end{bmatrix}, \qquad VV^T = \begin{bmatrix} BB^T & BA \\ A^TB^T & A^TA \end{bmatrix}.$$

To recover BA, we need to extract the top right block from VV^T. Let I_n represent the $n \times n$ identity matrix. Let $E_1 = \begin{bmatrix} I_m & 0_{m\times n} \end{bmatrix}$, $E_1 \in \mathbb{R}^{m\times(m+n)}$ represent the "extractor" matrix that extracts the top m rows from a $(m+n)\times(m+n)$ matrix when applied from the left, and let $E_2 = \begin{bmatrix} 0_{n\times m} & I_n \end{bmatrix}^T$, $E_2 \in \mathbb{R}^{(m+n)\times n}$ represent the matrix that extracts the right n columns when applied from the right. Then we have that

$$BA = E_1VV^TE_2.$$

We next define a new function $\mathcal{J} : \mathbb{R}^{(m+n)\times r} \to \mathbb{R}$ such that

$$\mathcal{J}(V) = \mathcal{L}(E_1VV^TE_2) = \mathcal{L}(BA). \tag{6}$$

By construction, LoRA gradient descent (3) is equivalent to gradient descent on $\mathcal{J}(V)$,

$$V_{t+1} = V_t - \eta_t \nabla \mathcal{J}(V_t),$$

since

$$\begin{aligned}\nabla\mathcal{J}(V) =& \nabla_V[\mathcal{L}(E_1VV^TE_2)] \\ =& \begin{bmatrix} \nabla_B\mathcal{L}(BA) \\ \nabla_{A^T}\mathcal{L}(BA) \end{bmatrix} = \begin{bmatrix} \nabla_B\mathcal{L}(BA) \\ (\nabla_A\mathcal{L}(BA))^T \end{bmatrix}.\end{aligned}$$

Step 2: Descent lemma. We consider the gradient $\nabla\mathcal{J}(V) \in \mathbb{R}^{(m+n)\times r}$. Let $g(V) = E_1VV^TE_2$, where $g : \mathbb{R}^{(m+n)\times r} \to \mathbb{R}^{m\times n}$. Then $\mathcal{J}(V) = \mathcal{L}(g(V))$. Let $G = \nabla\mathcal{L}(X)|_{X=E_1VV^TE_2}$ represent the gradient of $\mathcal{L}$ evaluated at $E_1VV^TE_2$. Then we have by the chain rule,

$$\begin{aligned}d\mathcal{J} =& \langle G, dg\rangle_F \\ =& \langle G, E_1(dVV^T + VdV^T)E_2\rangle_F \\ =& Tr(G^TE_1(dVV^T + VdV^T)E_2) \\ =& Tr(G^TE_1dVV^TE_2) + Tr(G^TE_1VdV^TE_2) \\ =& Tr(V^TE_2G^TE_1dV) + Tr(dV^TE_2G^TE_1V) \\ =& \langle E_1^TGE_2^TV, dV\rangle_F + \langle E_2G^TE_1V, dV\rangle_F \\ =& \langle 2Sym(E_1^TGE_2^T)V, dV\rangle_F,\end{aligned}$$

where $Sym(A) = \frac{A+A^T}{2}$. So we have

$$\nabla\mathcal{J}(V) = 2Sym(E_1^T\nabla\mathcal{L}(E_1VV^TE_2)E_2^T)V. \tag{7}$$

The function $\mathcal{J}$ is *not* Lipschitz smooth, due to the multiplicative factor of V in $\nabla\mathcal{J}(V)$. However, we can still obtain a descent lemma (Lemma 3.3) that is analogous to the classic descent lemma of Lipschitz smooth functions (5), but with more higher-order terms.

Lemma 3.3. *For $\mathcal{J}(V)$ defined in (6), we have for all $V_1, V_2 \in \mathbb{R}^{(m+n)\times r}$,*

$$\begin{aligned}\mathcal{J}(V_2) \le& \mathcal{J}(V_1) + \langle\nabla\mathcal{J}(V_1), V_2 - V_1\rangle_F \\ &+ \sqrt{2}L\|V_2 - V_1\|^2\|V_1\|^2 + \frac{\sqrt{2}L}{3}\|V_2 - V_1\|^3 \\ &+ \frac{2\sqrt{2}}{3}L\|V_2 - V_1\|^3\|V_1\| + \frac{\sqrt{2}L}{4}\|V_2 - V_1\|^4 \\ &+ \|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|V_2 - V_1\|^2.\end{aligned}$$

Proof. See Appendix A.2.

Step 3: Control step size to achieve convergence. Based on Lemma 3.3, we can pick η_t small enough to minimize the higher-order terms, thereby guaranteeing descent in one step as stated next.

Lemma 3.4. *For any $V_t \in \mathbb{R}^{(m+n)\times r}$ and $V_{t+1} = V_t - \eta_t\nabla\mathcal{J}(V_t)$, suppose that*

$$\eta_t = \min\{\frac{1}{5\sqrt{2}L(\|V_t\|^2 + \|\nabla\mathcal{L}(E_1V_tV_t^TE_2)\|)}, 1\}. \tag{8}$$

Then we have

$$\mathcal{J}(V_{t+1}) \le \mathcal{J}(V_t) - \frac{\eta_t}{5}\|\nabla\mathcal{J}(V_t)\|^2. \tag{9}$$

Proof. See Appendix A.3.

Lemma 3.4 states that to update V_{t+1}, we require the norm of the previous iterate $\|V_t\|$ and the gradient of the original loss function $\|\nabla\mathcal{L}(E_1V_tV_t^TE_2)\|$. By summing (9) over all

$t = 0, ..., T-1$ and telescoping sums, we determine that the minimum gradient norm over t is upper bounded as

$$\min_{t=0,\dots,T-1} \|\nabla \mathcal{J}(V_t)\|^2 \leq \frac{5(\mathcal{J}(V_0) - \mathcal{L}^*)}{\sum_{t=0}^{T-1} \eta_t}. \quad (10)$$

However, (10) is *not* sufficient to conclude convergence; we also need to show that the sum $\sum_{t=0}^{T-1} \eta_t$ *diverges* as T goes to infinity rather than converging to a finite value. Based on (8), this automatically holds if $\|V_t\|^2$ is uniformly upper bounded, leading to $\sum_{t=0}^{T-1} \eta_t = \Theta(T)$. However, if the parameter norm is not bounded, then $\|V_t\|^2$ may grow to infinity. In the proof of Theorem 3.5, we show that even in the worst case, we have $\|V_t\|^2 = O(t)$. The sum $\sum_{t=0}^{T-1} \eta_t$ is therefore lower bounded by a harmonic series which is $\Theta(\log(T))$, leading to Theorem 3.5.

Theorem 3.5. *Suppose Assumption 3.1 and 3.2 hold. We perform T steps of LoRA gradient descent (3), where at time step t the step size η_t is determined as*

$$\eta_t = \min\{\frac{1}{5\sqrt{2}L(\|V_t\|^2 + \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|)}, 1\}.$$

Then after T steps, we have

$$\min_{t=0,\dots,T-1} \|\nabla \mathcal{J}(V_t)\|^2 = O\left(\frac{1}{\log T}\right), \quad (11)$$

where the $O(\cdot)$ notation hides dependence on V_0, $\mathcal{J}(V_0)$, $\mathcal{L}^*$, and L. Moreover, if there exists a constant $C > 0$ such that for all V_t, $\|V_t\| \leq C$, then

$$\min_{t=0,\dots,T-1} \|\nabla \mathcal{J}(V_t)\|^2 = O\left(\frac{1}{T}\right), \quad (12)$$

where the $O(\cdot)$ notation hides dependence on $\mathcal{J}(V_0)$, $\mathcal{L}^*$, L, and C.

Proof. See Appendix A.4.

Theorem 3.5 states that LoRA gradient descent converges to a stationary point at $O(\frac{1}{\log T})$ rate. If $\|V_t\|$ is uniformly bounded, the rate improves to $O(1/T)$. This matches the standard convergence rate of gradient descent on Lipschitz smooth functions *and* the rates derived by existing works that analyze LoRA under the bounded parameter assumption, which essentially forces $\mathcal{J}$ to be Lipschitz smooth. Our results demonstrate that the behavior of LoRA is governed by complex interactions between η_t, $\|V_t\|$, and $\|\nabla \mathcal{J}(V_t)\|$, which slows convergence. However, even if $\|V_t\|$ grows and forces η_t to decrease at each iteration, the algorithm still converges; a simple explanation is that if $\|V_t\|$ grows, η_t decreases, which slows the growth of $\|V_t\|$.

3.3. Discussion

Our results yield several interesting insights into the behavior of LoRA. First, the choice of extractor matrices E_1, E_2, has little to no impact on the proof, suggesting that other subsets of the VV^T matrix could be extracted as alternate parameter-efficient reparametrizations to yield similar results. Moreover, the VV^T form is exactly the symmetric Burer-Monteiro parametrization (Burer & Monteiro, 2003), and the results of our analysis can be trivially extended to show that gradient descent with Burer-Monteiro also converges to stationary points for general Lipschitz smooth functions. Second, the convergence rate does not explicitly depend on the chosen LoRA rank r, except that it is $O(\frac{1}{\|V_0\|^2})$. This is expected because r just controls the dimension of the problem, but gradient descent is a *dimension-free* algorithm. Finally, due to the $\|V_t\|^2$ term in the denominator of η_t, the convergence of LoRA is slowed if the iterates are progressing away from the origin, and it is accelerated if they are moving towards the origin. This "position dependency" is not typical of gradient algorithms, and may be due to the fact that the LoRA reparametrization creates a stationary point at $V = 0$ *regardless* of the structure of the original function. Consequently, LoRA may converge to the origin even if the original global minima is arbitrarily far away, explaining the observed divergence in behavior between LoRA and full-rank fine-tuning (Shuttleworth et al., 2025; Biderman et al., 2024).

4. Experiments

We conduct experiments to investigate how the choice of step size and LoRA reparametrization affects training convergence in practice. We perform image classification on the CIFAR-10 dataset with 10 classes, using the cross-entropy loss function. We first consider the simple setting of logistic regression, which is appropriate because the loss function is known to be Lipschitz smooth. We train logistic models on both the original dataset as well as embeddings of the CIFAR-10 images, produced by a ResNet-18 model pre-trained on ImageNet. We then consider the more complex setting of directly training a ResNet-18 model on the CIFAR-10 dataset. We disable batch normalization layers because they can have complex interactions with the step size. For each experimental setting, we replace the model weight matrices with low-rank approximations ($r = 4$) and we train with mini-batch SGD with a large batch size ($b = 512$) to reflect our analysis of gradient descent. The model weights are first initialized randomly and frozen, and then the adapter matrices are initialized in the standard approach described in (Hu et al., 2022), with A as a random Gaussian matrix, and B as a zero matrix. For additional experimental details, see Appendix B.

We test three learning rate schemes: constant, *adaptive*, and *normalized*. We define the adaptive learning rate η^{adapt} as

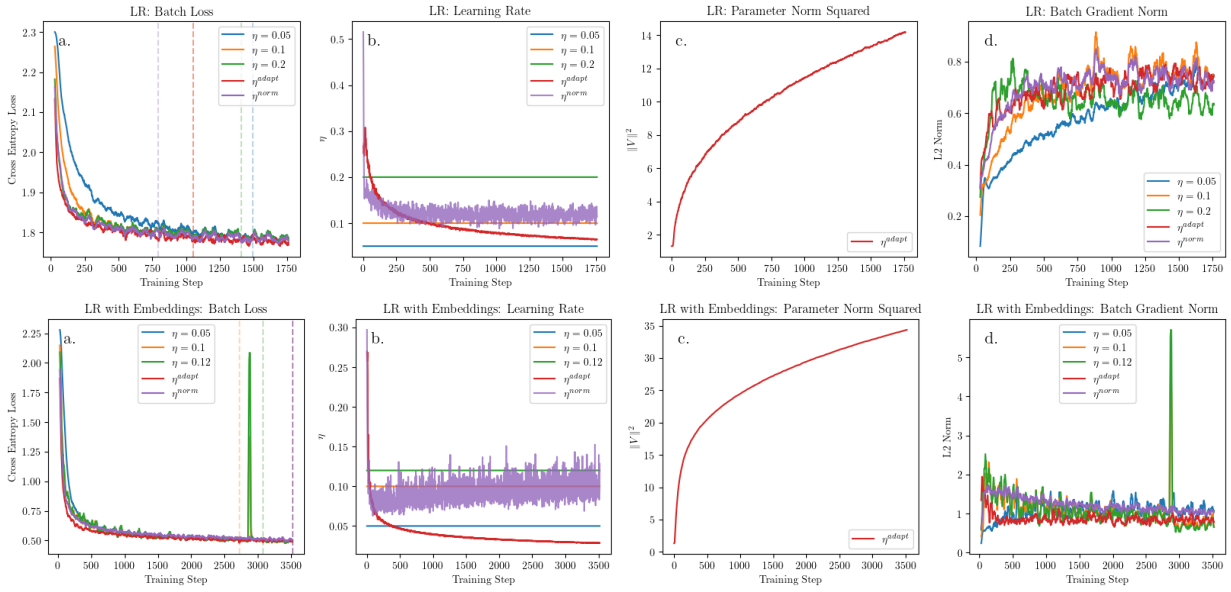

Figure 1. Training logistic regression model on CIFAR-10 with constant, adaptive, and normalized learning rates. The top row displays the results on the original dataset, and the bottom row displays the results on the data embeddings generated by a ResNet-18 model pretrained on ImageNet. Vertical lines indicate the time step of the minimum validation loss. For clarity, the moving averages (window size 30) of the batch loss and gradient norm are plotted (Figures a, d, e, and f).

follows:

$$\eta_t^{adapt} = \frac{\alpha}{\|V_t\|^2 + \|\nabla\mathcal{L}(E_1 V_t V_t^T E_2)\|}. \tag{13}$$

The adaptive learning rate mirrors the relationship between η_t, the parameter norm $\|V_t\|$ and the intermediate gradient norm $\|\nabla\mathcal{L}(E_1 V_t V_t^T E_2)\|$ derived in (8), multiplied by a scaling factor α that can be treated as a hyperparameter. While η^{adapt} closely reflects theory, it has two drawbacks. First, while theoretically the quantity $\|\nabla\mathcal{L}(E_1 V_t V_t^T E_2)\|$ is computed as a byproduct of the chain rule, in practice it saves computation to calculate $xA^T B^T$ instead of $x(BA)^T$ for a row vector input x, which does not automatically produce $\|\nabla\mathcal{L}(E_1 V_t V_t^T E_2)\|$. Second, the quantity $\|V_t\|$ can be prohibitively large for large models with millions of parameters, which would require a large scaling factor that can introduce numerical instability. As such, we also consider a more computationally tractable *normalized* learning rate η_{norm}, defined as follows.

$$\eta_t^{norm} = \frac{\alpha}{\|\nabla\mathcal{L}(V_t)\|^{1/2}}. \tag{14}$$

This inverse relationship between η_t and $\|V_t\|^{1/2}$ is required for minimizing higher order terms in the descent lemma, as stated in (23) and (25) in the Appendix. In practice, we compute η_t^{adapt} and η_t^{norm} using the *batch* gradient at training step t.

Figure 1 displays the empirical results of both logistic regression settings. Figures 1a and 1e demonstrate that training with either η_t^{adapt} or η_t^{norm} exhibits faster and more stable convergence than training with a constant learning rate in the same range of values. If the constant learning rate is too high training becomes unstable, but if it is too low convergence slows. Figures 1b and 1f display η^{adapt}, η^{norm}, and various constant learning rates, demonstrating that η_t^{norm} and η_t^{adapt} are closely correlated early on, but diverge as the model converges, with η_t^{norm} displaying more unstable behavior. Figures 1c and 1g display the progress of $\|V_t\|^2$ when the model is trained with η^{adapt}, which in combination with Figures 1d and 1h, suggest that the model is initialized in a flat region close to the origin, where the gradient norm is small. This aligns with the fact that LoRA always creates a stationary point at the origin.

Figure 2 displays the results of training ResNet-18 on CIFAR-10 using constant and normalized learning rates. The normalized learning rate significantly stabilizes training compared to constant learning rate while maintaining fast convergence. Therefore, η^{norm} can be a computationally efficient method for improving LoRA convergence on large neural networks.

Ultimately, our results demonstrate that η^{adapt} and η^{norm} can improve LoRA training by taking advantage of the structural changes introduced by the low-rank reparametrization. Both learning rate schemes start at higher values when the gradient norm is low and decrease over time, thereby accelerating training through the plateau around initialization and stabilizing it when the gradient norm is large.

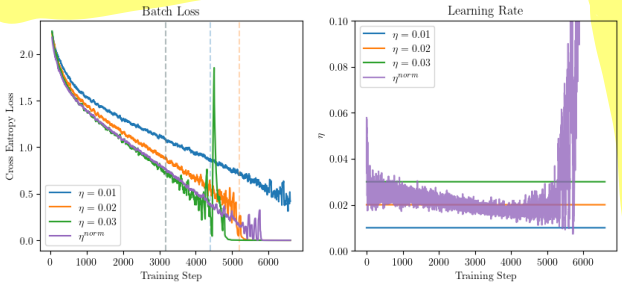

Figure 2. Training ResNet-18 on CIFAR-10 with constant and normalized learning rates. For clarity, the moving average (window size 50) of the batch loss is plotted. Vertical lines indicate the time step of the minimum validation loss.

5. Conclusion

In this work, we show for first time that the original LoRA gradient descent algorithm achieves a convergence rate of $O(\frac{1}{\log T})$, without requiring bounded adapter matrices or Lipschitz smoothness of the reparametrized loss function.

Future research directions may include determining convergence rates on convex or strongly convex loss functions, whose geometric properties may change under the LoRA reparametrization, as well as analyzing the *stochastic* gradient descent setting. A major challenge of this setting is deriving a fourth-moment bound on the noise.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Aghajanyan, A., Gupta, S., and Zettlemoyer, L. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *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)*, pp. 7319–7328, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

Ahmed, S. How to choose the right learning rate in deep learning (with PyTorch), February 2025. URL https://medium.com/@sahin.samia/how-to-choose-the-right-learning-rate-in-deep-learning-with-pytorch-690de782b405.

Biderman, D., Portes, J., Ortiz, J. J. G., Paul, M., Greengard, P., Jennings, C., King, D., Havens, S., Chiley, V., Frankle, J., Blakeney, C., and Cunningham, J. P. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=aloEru2qCG. Featured Certification.

Burer, S. and Monteiro, R. D. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, February 2003. ISSN 1436-4646. doi: 10.1007/s10107-002-0352-8. URL https://doi.org/10.1007/s10107-002-0352-8.

Büyükakyüz, K. Olora: Orthonormal low-rank adaptation of large language models, 2024. URL https://arxiv.org/abs/2406.01775.

Chen, Y., Zhang, Y., Liu, Y., Yuan, K., and Wen, Z. A memory efficient randomized subspace optimization method for training large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=XuCf87V8OF.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1feb87871436031bdc0f2beaa62a049b-Paper-Conference.pdf.

Garrigos, G. and Gower, R. M. Handbook of convergence theorems for (stochastic) gradient methods, 2024. URL https://arxiv.org/abs/2301.11235.

Ghiasvand, S., Alizadeh, M., and Pedarsani, R. Decentralized low-rank fine-tuning of large language models, 2025. URL https://arxiv.org/abs/2501.15361.

Hardy, G., Littlewood, J., and Polya, G. *Inequalities*. Cambridge University Press, 1934. URL https://mathematicalolympiads.wordpress.com/wp-content/uploads/2012/08/inequalities-hardy-littlewood-polya.pdf.

Hayou, S., Ghosh, N., and Yu, B. Lora+: Efficient low rank adaptation of large models, 2024. URL https://arxiv.org/abs/2402.12354.

Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jang, U., Lee, J. D., and Ryu, E. K. LoRA training in the NTK regime has no spurious local minima. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=s1sdx6vNsU.

Jiang, Z., Saadati, N., Balu, A., Pham, M., Waite, J. R., Saleem, N., Hegde, C., and Sarkar, S. A unified convergence theory for large language model efficient fine-tuning. In *OPT 2024: Optimization for Machine Learning*, 2024. URL https://openreview.net/forum?id=f0lq26eITJ.

Kim, J., Kim, J., and Ryu, E. K. LoRA training provably converges to a low-rank global minimum or it fails loudly (but it probably won't fail). In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=o9zDYV4Ism.

Li, C., Farkhoor, H., Liu, R., and Yosinski, J. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryup8-WCW.

Li, T., He, Z., Li, Y., Wang, Y., Shang, L., and Huang, X. Flat-loRA: Low-rank adaptation over a flat loss landscape. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=3Qj3xSwN2I.

Lialin, V., Muckatira, S., Shivagunde, N., and Rumshisky, A. ReloRA: High-rank training through low-rank updates. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=DLJznSp6X3.

Malinovsky, G., Michieli, U., Hammoud, H. A. A. K., Ceritli, T., Elesedy, H., Ozay, M., and Richtárik, P. Randomized asymmetric chain of lora: The first meaningful theoretical framework for low-rank adaptation, 2024. URL https://arxiv.org/abs/2410.08305.

Malladi, S., Wettig, A., Yu, D., Chen, D., and Arora, S. A kernel-based view of language model fine-tuning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23610–23641. PMLR, 7 2023. URL https://proceedings.mlr.press/v202/malladi23a.html.

Meng, F., Wang, Z., and Zhang, M. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=6ZBHIEtdP4.

Oymak, S., Fabian, Z., Li, M., and Soltanolkotabi, M. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *CoRR*, abs/1906.05392, 2019. URL http://arxiv.org/abs/1906.05392.

Park, H. and Klabjan, D. Communication-efficient federated low-rank update algorithm and its connection to implicit regularization, 2025. URL https://arxiv.org/abs/2409.12371.

Robert, T., Safaryan, M., Modoranu, I.-V., and Alistarh, D. LDAdam: Adaptive optimization from low-dimensional gradient statistics. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Zkp1GuHerF.

Shen, Y. Kronecker-lora: hybrid kronecker-lora adapters for scalable, sustainable fine-tuning, 2025. URL https://arxiv.org/abs/2508.01961.

Shuttleworth, R., Andreas, J., Torralba, A., and Sharma, P. Lora vs full fine-tuning: An illusion of equivalence, 2025. URL https://arxiv.org/abs/2410.21228.

Smith, L. N. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 464–472, 2017. doi: 10.1109/WACV.2017.58.

Sokolov, I., Sadiev, A., Demidovich, Y., Al-Qahtani, F. S., and Richtárik, P. Bernoulli-lora: A theoretical framework for randomized low-rank adaptation, 2025. URL https://arxiv.org/abs/2508.03820.

Sun, Y., Li, Z., Li, Y., and Ding, B. Improving loRA in privacy-preserving federated learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NLPzL6HWNl.

Wang, S., Yu, L., and Li, J. LoRA-GA: Low-rank adaptation with gradient approximation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=VaLAWrLHJv.

Xu, Z., Min, H., Luo, J., MacDonald, L. E., Tarmoun, S., Mallada, E., and Vidal, R. Understanding the learning dynamics of loRA: A gradient flow perspective on low-rank

adaptation in matrix factorization. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL https://openreview.net/forum?id=hphdX8WlcT.

Yang, Y., Long, G., Lu, Q., Zhu, L., Jiang, J., and Zhang, C. Federated low-rank adaptation for foundation models: a survey. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, IJCAI '25, 2025. ISBN 978-1-956792-06-5. doi: 10.24963/ijcai.2025/1196. URL https://doi.org/10.24963/ijcai.2025/1196.

Yaras, C., Wang, P., Balzano, L., and Qu, Q. Compressible dynamics in deep overparameterized low-rank learning & adaptation. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=uDkXoZMzBv.

Zeng, Y. and Lee, K. The expressive power of low-rank adaptation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=likXVjmh3E.

Zhang, F. and Pilanci, M. Riemannian preconditioned lora for fine-tuning foundation models, 2024. URL https://arxiv.org/abs/2402.02347.

Zhang, Y., Liu, F., and Chen, Y. LoRA-one: One-step full gradient could suffice for fine-tuning large language models, provably and efficiently. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=KwIlvmLDLm.

Zhao, J., Zhang, Z., Chen, B., Wang, Z., Anandkumar, A., and Tian, Y. Galore: Memory-efficient LLM training by gradient low-rank projection. In *5th Workshop on practical ML for limited/low resource settings*, 2024. URL https://openreview.net/forum?id=AzqPyO22zt.

A. Proofs

A.1. Helper Lemmas

Lemma A.1. *(Weighted) AM-GM inequality. Given the nonnegative values x_1, x_2, w_1, w_2, we have*

$$\frac{w_1 x_1 + w_2 x_2}{w_1 + w_2} \geq (x_1^{w_1} x_2^{w_2})^{\frac{1}{w_1+w_2}}. \tag{15}$$

In particular, for $w_1 = w_2 = 1$, we have

$$x_1 + x_2 \geq 2\sqrt{x_1 x_2}. \tag{16}$$

For the proof of Lemma A.1, see Section 2.5 of (Hardy et al., 1934).

Lemma A.2. *For any matrix $A \in \mathbb{R}^{(m+n)\times(m+n)}$ and the extractor matrices $E_1 = \begin{bmatrix} I_m & 0_{m\times n} \end{bmatrix}$, $E_1 \in \mathbb{R}^{m\times(m+n)}$, and $E_2 = \begin{bmatrix} 0_{n\times m} & I_n \end{bmatrix}^T$, $E_2 \in \mathbb{R}^{(m+n)\times n}$*

$$\|E_1 A E_2\| \leq \|A\|.$$

Moreover, for any matrix $B \in \mathbb{R}^{m\times n}$,

$$\|E_1^T B E_2^T\| = \|B\|.$$

Finally, if $A \in \mathbb{R}^{(m+n)\times(m+n)}$ is symmetric,

$$\|E_1 A E_2\| \leq \frac{1}{\sqrt{2}}\|A\|. \tag{17}$$

Proof. The first statement is true because $E_1 A E_2$ contains a subset of the entries of A. The second statement is true because $E_1^T B E_2^T$ produces a matrix that only contains the entries of B and zeros. The last statement can be shown by observing that $E_1 A E_2$ extracts the upper right $m \times n$ block from A, specifically the elements at i, j where $1 \leq i \leq m$ and $m + 1 \leq j \leq m + n$. So if we have

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix},$$

Then $\|A\|^2 = \|A_{11}\|^2 + \|A_{22}\|^2 + 2\|A_{12}\|^2$ and $\|A_{12}\| \leq \frac{1}{\sqrt{2}}\|A\|$. □

Lemma A.3. *Let $\{a_n\}_{n\geq 0}$ be a real, nonnegative sequence, and let $c > 0$ be a fixed constant. If the series $\sum_{n=1}^{\infty} \min(a_n, c)$ converges, then the series $\sum_{n=1}^{\infty} a_n$ also converges.*

Proof. Let $b_n = \min(a_n, c)$. Since $\sum_{n=1}^{\infty} b_n$ converges, the terms b_n must go to zero, so there exists $N > 0$ such that for $n \geq N$,

$$b_n \leq \frac{c}{2}.$$

This also implies that for $n \geq N$, $a_n \leq \frac{c}{2} < c$. So we have that $\sum_{n=N}^{\infty} a_n = \sum_{n=N}^{\infty} b_n$, and both these tail sums must converge. Then our original series can be written in terms of a finite sum and the tail sum, both of which are finite.

$$\sum_{n=1}^{\infty} a_n = \sum_{n=1}^{N-1} a_n + \sum_{n=N}^{\infty} a_n = \sum_{n=1}^{N-1} a_n + \sum_{n=N}^{\infty} b_n.$$

□

A.2. Proof of Lemma 3.3

Proof. Based on the expression for the gradient $\nabla \mathcal{J}(V)$ in (7), we have

$$\begin{aligned}
\nabla \mathcal{J}(V_2) - \nabla \mathcal{J}(V_1) =& 2\big[Sym(E_1^T \nabla \mathcal{L}(E_1 V_2 V_2^T E_2) E_2^T) V_2 - Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T) V_1\big], \\
=& 2\big[Sym(E_1^T \nabla \mathcal{L}(E_1 V_2 V_2^T E_2) E_2^T) V_2 - Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T) V_2 \\
&+ Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T) V_2 - Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T) V_1\big], \\
=& 2\big[(Sym(E_1^T \nabla \mathcal{L}(E_1 V_2 V_2^T E_2) E_2^T) - Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T)) V_2 \\
&+ Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T)(V_2 - V_1)\big], \\
=& 2\big[Sym(E_1^T \nabla \mathcal{L}(E_1 V_2 V_2^T E_2) E_2^T - E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T) V_2 \\
&+ Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T)(V_2 - V_1)\big], \\
=& 2\big[Sym(E_1^T (\nabla \mathcal{L}(E_1 V_2 V_2^T E_2) - \nabla \mathcal{L}(E_1 V_1 V_1^T E_2)) E_2^T) V_2 \\
&+ Sym(E_1^T \nabla \mathcal{L}(E_1 V_1 V_1^T E_2) E_2^T)(V_2 - V_1)\big]. \qquad (18)
\end{aligned}$$

If we denote $d = V_2 - V_1$, and parametrize $\mathcal{J}$ between V_1 and V_2 such that for $t \in [0, 1]$, then

$$\begin{aligned}
\phi(t) =& \mathcal{J}(V_1 + td), \\
\phi'(t) =& \langle \nabla \mathcal{J}(V_1 + td), d \rangle_F.
\end{aligned}$$

By the fundamental theorem of calculus, we have

$$\begin{aligned}
\mathcal{J}(V_2) - \mathcal{J}(V_1) = \phi(1) - \phi(0) &= \int_0^1 \phi'(t) dt, \\
&= \int_0^1 \langle \nabla \mathcal{J}(V_1), d \rangle_F \, dt + \int_0^1 \langle \nabla \mathcal{J}(V_1 + td) - \nabla \mathcal{J}(V_1), d \rangle_F \, dt, \\
&= \langle \nabla \mathcal{J}(V_1), d \rangle_F + \int_0^1 \langle \nabla \mathcal{J}(V_1 + td) - \nabla \mathcal{J}(V_1), d \rangle_F \, dt, \\
&\leq \langle \nabla \mathcal{J}(V_1), d \rangle + \int_0^1 \|\nabla \mathcal{J}(V_1 + td) - \nabla \mathcal{J}(V_1)\| \|d\| dt, \qquad (19)
\end{aligned}$$

where in the last step we use the Cauchy-Schwarz inequality and the fact that integrals preserve inequalities. From (18), we have

$$
\begin{aligned}
\|\nabla\mathcal{J}(V_1+td)-\nabla\mathcal{J}(V_1)\| =& 2\|Sym(E_1^T(\nabla\mathcal{L}(E_1(V_1+td)(V_1+td)^TE_2)-\nabla\mathcal{L}(E_1V_1V_1^TE_2))E_2^T)(V_1+td)\\
&+Sym(E_1^T\nabla\mathcal{L}(E_1V_1V_1^TE_2)E_2^T)td\|,\\
\overset{\text{Cauchy-Schwarz}}{\leq}& 2\|Sym(E_1^T(\nabla\mathcal{L}(E_1(V_1+td)(V_1+td)^TE_2)-\nabla\mathcal{L}(E_1V_1V_1^TE_2))E_2^T)\|\|V_1+td\|\\
&+2\|Sym(E_1^T\nabla\mathcal{L}(E_1V_1V_1^TE_2)E_2^T)\|\|td\|,\\
\overset{\|Sym(A)\|\leq\|A\|}{\leq}& 2\|E_1^T(\nabla\mathcal{L}(E_1(V_1+td)(V_1+td)^TE_2)-\nabla\mathcal{L}(E_1V_1V_1^TE_2))E_2^T\|\|V_1+td\|\\
&+2\|E_1^T\nabla\mathcal{L}(E_1V_1V_1^TE_2)E_2^T\|\|td\|,\\
\overset{\text{Lemma A.2}}{\leq}& 2\|\nabla\mathcal{L}(E_1(V_1+td)(V_1+td)^TE_2)-\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|V_1+td\|\\
&+2t\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|,\\
\overset{\text{Assumption 3.1}}{\leq}& 2L\|E_1(V_1+td)(V_1+td)^TE_2-E_1V_1V_1^TE_2\|\|V_1+td\|+2t\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|,\\
\overset{(17)}{\leq}& \sqrt{2}L\|(V_1+td)(V_1+td)^T-V_1V_1^T\|\|V_1+td\|+2t\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|,\\
=&\sqrt{2}L\|(V_1+td)(td)^T+tdV_1^T\|\|V_1+td\|+2t\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|,\\
\leq&\sqrt{2}Lt\|d\|\|V_1+td\|^2+\sqrt{2}Lt\|d\|\|V_1\|\|V_1+td\|+2t\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|,\\
\leq&\sqrt{2}Lt\|d\|\|V_1+td\|^2+\sqrt{2}Lt\|d\|\|V_1\|^2+\sqrt{2}Lt^2\|d\|^2+2t\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|.
\end{aligned}
$$

We plug this expression into the integral in (19) and pull out terms that do not depend on t to obtain

$$
\begin{aligned}
\int_0^1\|\nabla\mathcal{J}(V_1+td)-\nabla\mathcal{J}(V_1)\|\|d\|dt \leq&\sqrt{2}L\|d\|^2\int_0^1 t\|V_1+td\|^2dt+\sqrt{2}L\|d\|^2\|V_1\|^2\int_0^1 tdt\\
&+\sqrt{2}L\|d\|^3\int_0^1 t^2dt+2\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|^2\int_0^1 tdt,\\
=&\sqrt{2}L\|d\|^2(\frac{1}{2}\|V_1\|^2+\frac{2}{3}\langle V_1,d\rangle_F+\frac{1}{4}\|d\|^2)+\frac{\sqrt{2}}{2}L\|d\|^2\|V_1\|^2\\
&+L\frac{\sqrt{2}}{3}\|d\|^3+\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|^2,\\
\overset{\text{Cauchy-Schwarz}}{\leq}&\frac{\sqrt{2}}{2}L\|d\|^2\|V_1\|^2+\frac{2\sqrt{2}}{3}L\|d\|^3\|V_1\|+\frac{\sqrt{2}}{4}L\|d\|^4+\frac{\sqrt{2}}{2}L\|d\|^2\|V_1\|^2\\
&+L\frac{\sqrt{2}}{3}\|d\|^3+\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|^2,\\
=&\frac{2\sqrt{2}}{3}L\|d\|^3\|V_1\|+\frac{\sqrt{2}}{4}L\|d\|^4+\sqrt{2}L\|d\|^2\|V_1\|^2+L\frac{\sqrt{2}}{3}\|d\|^3+\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|d\|^2.
\end{aligned}
$$

We therefore have for all V_1, V_2,

$$
\begin{aligned}
\mathcal{J}(V_2)\leq&\mathcal{J}(V_1)+\langle\nabla\mathcal{J}(V_1),V_2-V_1\rangle_F+\frac{2\sqrt{2}}{3}L\|V_2-V_1\|^3\|V_1\|+\sqrt{2}L\|V_2-V_1\|^2\|V_1\|^2+L\frac{\sqrt{2}}{3}\|V_2-V_1\|^3\\
&+\frac{\sqrt{2}}{4}L\|V_2-V_1\|^4+\|\nabla\mathcal{L}(E_1V_1V_1^TE_2)\|\|V_2-V_1\|^2.
\end{aligned}
$$

□

A.3. Proof of Lemma 3.4

Proof. We apply the descent lemma (Lemma 3.3) with $V_1 = V_t$, $V_2 = V_{t+1} = V_t - \eta_t \nabla \mathcal{J}(V_t)$, which yields

$$\begin{aligned}\mathcal{J}(V_{t+1}) \leq & \mathcal{J}(V_t) - \eta_t \langle \nabla \mathcal{J}(V_t), \nabla \mathcal{J}(V_t) \rangle_F + \frac{2\sqrt{2}}{3} L \eta_t^3 \|\nabla \mathcal{J}(V_t)\|^3 \|V_t\| + \sqrt{2} L \eta_t^2 \|\nabla \mathcal{J}(V_t)\|^2 \|V_t\|^2 + \frac{\sqrt{2}L}{3} \eta_t^3 \|\nabla \mathcal{J}(V_t)\|^3 \\ & + \frac{\sqrt{2}}{4} \eta_t^4 L \|\nabla \mathcal{J}(V_t)\|^4 + \eta_t^2 \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\| \|\nabla \mathcal{J}(V_t)\|^2, \\ = & \mathcal{J}(V_t) - \eta_t \|\nabla \mathcal{J}(V_t)\|^2 + \frac{2\sqrt{2}}{3} L \eta_t^3 \|\nabla \mathcal{J}(V_t)\|^3 \|V_t\| + \eta_t^2 \|\nabla \mathcal{J}(V_t)\|^2 (\sqrt{2} L \|V_t\|^2 + \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|) \\ & + \frac{\sqrt{2}L}{3} \eta_t^3 \|\nabla \mathcal{J}(V_t)\|^3 + \frac{\sqrt{2}L}{4} \eta_t^4 \|\nabla \mathcal{J}(V_t)\|^4.\end{aligned}$$

We want to select η_t to minimize the last four terms, such that they sum to a value smaller than $\eta_t \|\nabla \mathcal{J}(V_t)\|^2$. This will guarantee descent in function value $\mathcal{J}$. Let

$$\eta_t = \min\{\frac{1}{5\sqrt{2}L(\|V_t\|^2 + \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|)}, 1\}.$$

In the following, we prove that η_t is smaller than various upper bounds that will allow us to achieve a clean descent result. First, since $L \geq 1$, we have the following bound,

$$\eta_t \leq \frac{1}{5(\sqrt{2}L\|V_t\|^2 + \sqrt{2}L\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|)} \leq \frac{1}{5(\sqrt{2}L\|V_t\|^2 + \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|)}. \tag{20}$$

For the next bound, we have from Cauchy-Schwarz and (7) that

$$\|\nabla \mathcal{J}(V)\| \leq 2\|\nabla \mathcal{L}(E_1 V V^T E_2)\| \|V\|. \tag{21}$$

Therefore, from (16) of Lemma A.1, we have $\frac{1}{a+b} \leq \frac{1}{2\sqrt{ab}}$ for $a, b > 0$, such that η_t satisfies the following bound,

$$\begin{aligned}\eta_t \leq \frac{1}{5\sqrt{2}L\|V_t\|^2 + 5\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|} & \leq \left(\frac{1}{100\sqrt{2}L\|V_t\|^2 \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|}\right)^{1/2}, \\ & \overset{(21)}{\leq} \left(\frac{1}{50\sqrt{2}L\|V_t\| \|\nabla \mathcal{J}(V_t)\|}\right)^{1/2}, \qquad (22) \\ & \leq \left(\frac{3}{10\sqrt{2}L\|\nabla \mathcal{J}(V_t)\| \|V_t\|}\right)^{1/2}. \qquad (23)\end{aligned}$$

For the next bound, by (15) of Lemma A.1 with $w_1 = 1$, $w_2 = 2$, $x_1 = 5\sqrt{2}\|V_t\|^2$, $x_2 = \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|$, we have

$$\begin{aligned}5\sqrt{2}L\|V_t\|^2 + 2\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\| \geq & 3\left(5\sqrt{2}L\|V_t\|^2 \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|^2\right)^{1/3}, \\ \overset{(21)}{\geq} & 3\left(\frac{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|^2}{4}\right)^{1/3}.\end{aligned}$$

So η_t also satisfies the bound

$$\begin{aligned}\eta_t \leq & \frac{1}{5\sqrt{2}L\|V_t\|^2 + 5\sqrt{2}L\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|}, \\ \leq & \frac{1}{5\sqrt{2}L\|V_t\|^2 + 2\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|}, \\ \leq & \left(\frac{4}{135\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|^2}\right)^{1/3}, \\ \leq & \left(\frac{4}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|^2}\right)^{1/3}. \qquad (24)\end{aligned}$$

Finally, we also want to show

$$\eta_t \leq \left(\frac{3}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}\right)^{1/2}. \tag{25}$$

Let us assume that $\left(\frac{3}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}\right)^{1/2} < 1$, because if this is not true, then the result is trivial. We can break down the casework as follows.

- Case 1: $\|V_t\| = 0$. In this case, we have by (7) that $\|\nabla \mathcal{J}(V_t)\| = 0$. So the result is trivially shown.
- Case 2: $0 < \|V_t\| \leq 1$. In this case, we have from (21)

$$\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\| \geq \frac{\|\nabla \mathcal{J}(V_t)\|}{2\|V_t\|}.$$

  Then we have

$$\begin{aligned}
\eta_t \leq & \frac{1}{5\sqrt{2}L\|V_t\|^2 + 5L\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|}, \\
\leq & \frac{1}{5\sqrt{2}L\|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|}, \\
\leq & \frac{2\|V_t\|}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}, \\
\leq & \frac{2}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}, \\
\leq & \frac{3}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|} \leq \left(\frac{3}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}\right)^{1/2},
\end{aligned}$$

  where in the last step we use the fact that $\left(\frac{3}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}\right)^{1/2} < 1$.

- Case 3: $\|V_t\| > 1$. From (22), we have

$$\begin{aligned}
\eta_t \leq & \left(\frac{1}{50\sqrt{2}L\|V_t\|\|\nabla \mathcal{J}(V_t)\|}\right)^{1/2}, \\
\leq & \left(\frac{1}{50\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}\right)^{1/2}, \\
\leq & \left(\frac{3}{5\sqrt{2}L\|\nabla \mathcal{J}(V_t)\|}\right)^{1/2}.
\end{aligned}$$

In conclusion, with the choice of η_t in (8), we can substitute the derived bounds on η_t into Lemma 3.3, achieving descent in one step.

$$\begin{aligned}
\mathcal{J}(V_{t+1}) \leq & \mathcal{J}(V_t) - \eta_t\|\nabla \mathcal{J}(V_t)\|^2 + \overbrace{\frac{2\sqrt{2}L}{3}\eta_t^3\|\nabla \mathcal{J}(V_t)\|^3\|V_t\|}^{(23)} + \overbrace{\eta_t^2\|\nabla \mathcal{J}(V_t)\|^2(\sqrt{2}L\|V_t\|^2 + \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|)}^{(20)} \\
& + \underbrace{\frac{\sqrt{2}L}{3}\eta_t^3\|\nabla \mathcal{J}(V_t)\|^3}_{(25)} + \underbrace{\frac{\sqrt{2}L}{4}\eta_t^4\|\nabla \mathcal{J}(V_t)\|^4}_{(24)} \\
\mathcal{J}(V_{t+1}) \leq & \mathcal{J}(V_t) - \eta_t\|\nabla \mathcal{J}(V_t)\|^2 + \frac{\eta_t}{5}\|\nabla \mathcal{J}(V_t)\|^2 + \frac{\eta_t}{5}\|\nabla \mathcal{J}(V_t)\|^2 + \frac{\eta_t}{5}\|\nabla \mathcal{J}(V_t)\|^2 + \frac{\eta_t}{5}\|\nabla \mathcal{J}(V_t)\|^2 \\
\leq & \mathcal{J}(V_t) - \frac{\eta_t}{5}\|\nabla \mathcal{J}(V_t)\|^2.
\end{aligned}$$

□

A.4. Proof of Theorem 3.5

Proof. From Lemma 3.4, we know that if

$$\eta_t = \min\{\frac{1}{5\sqrt{2}L(\|V_t\|^2 + \|\nabla\mathcal{L}(E_1V_tV_t^TE_2)\|)}, 1\},$$

then we have descent in one step

$$\mathcal{J}(V_{t+1}) - \mathcal{J}(V_t) \leq -\frac{\eta_t}{5}\|\nabla\mathcal{J}(V_t)\|^2.$$

Rearranging this equation and summing on both sides yields

$$\begin{aligned}
\frac{\eta_t}{5}\|\nabla\mathcal{J}(V_t)\|^2 \leq& \mathcal{J}(V_t) - \mathcal{J}(V_{t+1}),\\
\sum_{t=0}^{T-1}\frac{\eta_t}{5}\|\nabla\mathcal{J}(V_t)\|^2 \leq& \sum_{t=0}^{T-1}\mathcal{J}(V_t) - \mathcal{J}(V_{t+1}),\\
=& \mathcal{J}(V_0) - \mathcal{J}(V_T),\\
\overset{\text{Assumption 3.2}}{\leq}& \mathcal{J}(V_0) - \mathcal{L}^*,\\
\min_{t=0,\dots,T-1}\|\nabla\mathcal{J}(V_t)\|^2\sum_{t=0}^{T-1}\eta_t \leq& 5(\mathcal{J}(V_0) - \mathcal{L}^*),\\
\min_{t=0,\dots,T-1}\|\nabla\mathcal{J}(V_t)\|^2 \leq& \frac{5(\mathcal{J}(V_0) - \mathcal{L}^*)}{\sum_{t=0}^{T-1}\eta_t}.
\end{aligned}$$

So to show convergence to a stationary point, we need to show that the sum of learning rates $\sum_{t=0}^{\infty}\eta_t$ diverges. From Lipschitz smoothness of $\mathcal{L}$, we have for all $W \in \mathbb{R}^{m\times n}$ (See Lemma 2.28 in (Garrigos & Gower, 2024)),

$$\begin{aligned}
\|\nabla\mathcal{L}(W)\|^2 \leq& 2L(\mathcal{L}(W) - \mathcal{L}^*)\\
\|\nabla\mathcal{L}(W)\| \leq& (2L(\mathcal{L}(W) - \mathcal{L}^*))^{1/2}\\
\|\nabla\mathcal{L}(E_1VV^TE_2)\| \leq& (2L(\mathcal{L}(E_1VV^TE_2) - \mathcal{L}^*))^{1/2} = (2L(\mathcal{J}(V) - \mathcal{L}^*))^{1/2}.
\end{aligned}$$

We can obtain the following lower bound, using the fact that $\mathcal{J}(V_t)$ is decreasing with t,

$$\begin{aligned}
\frac{1}{5\sqrt{2}L(\|V_t\|^2 + \|\nabla\mathcal{L}(E_1V_tV_t^TE_2)\|)} \geq& \frac{1}{5\sqrt{2}L(\|V_t\|^2 + (2L(\mathcal{J}(V_t) - \mathcal{L}^*))^{1/2})},\\
\geq& \frac{1}{5\sqrt{2}L(\|V_t\|^2 + (2L(\mathcal{J}(V_0) - \mathcal{L}^*))^{1/2})}.
\end{aligned}$$

Let $D_t = 5\sqrt{2}L(\|V_t\|^2 + (2L(\mathcal{J}(V_0) - \mathcal{L}^*))^{1/2})$, such that we can lower bound η_t as follows,

$$\eta_t = \min\{\frac{1}{5\sqrt{2}L(\|V_t\|^2 + \|\nabla\mathcal{L}(E_1V_tV_t^TE_2)\|)}, 1\} \geq \min\{\frac{1}{D_t}, 1\}.$$

To show the divergence of $\sum_{t=0}^{\infty}\eta_t$, it is sufficient to show the divergence of the series $\sum_{t=0}^{\infty}\min\{\frac{1}{D_t}, 1\}$. Moreover, by the contrapositive of Lemma A.3, we only need to show divergence of $\sum_{t=0}^{\infty}\frac{1}{D_t}$. We first show that we achieve divergence if the adapter norms are bounded. If $\|V_t\| \leq C$ for all t, then η_t is lower bounded by a nonzero constant $\eta > 0$

$$\eta := \min\{\frac{1}{5\sqrt{2}L(C^2 + (2L(\mathcal{J}(V_0) - \mathcal{L}^*))^{1/2})}, 1\} \leq \eta_t,$$

and we prove (12) of Theorem 3.5 as follows

$$\min_{t=0,\dots,T-1}\|\nabla\mathcal{J}(V_t)\|^2 \leq \frac{5(\mathcal{J}(V_0) - \mathcal{L}^*)}{\eta T} = O(1/T).$$

Now we consider the general case where the adapter norms are not necessarily bounded. In the following, we show that $D_t = O(t)$. We first control the growth of $\|V_t\|^2$, showing that it increases at most linearly with t. We have

$$\begin{aligned}
\|V_{t+1}\|^2 =& \|V_t - \eta_t \nabla \mathcal{J}(V_t)\|^2, \\
\leq& \|V_t\|^2 + 2\eta_t |\langle V_t, \nabla \mathcal{J}(V_t)\rangle_F| + \eta_t^2 \|\nabla \mathcal{J}(V_t)\|^2, \\
\overset{\eta_t \leq 1}{\leq}& \|V_t\|^2 + 2\eta_t |\langle V_t, \nabla \mathcal{J}(V_t)\rangle_F| + \eta_t \|\nabla \mathcal{J}(V_t)\|^2, \\
\leq& \|V_t\|^2 + \eta_t(\|V_t\|^2 + \|\nabla \mathcal{J}(V_t)\|^2) + \eta_t \|\nabla \mathcal{J}(V_t)\|^2, \\
\leq& \|V_t\|^2 + \frac{\|V_t\|^2}{5\sqrt{2}L(\|V_t\|^2 + \|\nabla \mathcal{L}(E_1 V_t V_t^T E_2)\|)} + 2\eta_t \|\nabla \mathcal{J}(V_t)\|^2, \\
\leq& \|V_t\|^2 + \frac{1}{5\sqrt{2}L} + 2\eta_t \|\nabla \mathcal{J}(V_t)\|^2, \\
\|V_{t+1}\|^2 - \|V_t\|^2 \leq& \frac{1}{5\sqrt{2}L} + 2\eta_t \|\nabla \mathcal{J}(V_t)\|^2, \\
\sum_{t=0}^{T-1} \|V_{t+1}\|^2 - \|V_t\|^2 \leq& \frac{T}{5\sqrt{2}L} + 2\sum_{t=0}^{T-1} \eta_t \|\nabla \mathcal{J}(V_t)\|^2, \\
\leq& \frac{T}{5\sqrt{2}L} + 10(\mathcal{J}(V_0) - \mathcal{L}^*), \\
\|V_T\|^2 \leq& \|V_0\|^2 + \frac{T}{5\sqrt{2}L} + 10(\mathcal{J}(V_0) - \mathcal{L}^*).
\end{aligned}$$

We have that D_t is upper bounded by a linear term,

$$D_t = 5\sqrt{2}L(\|V_t\|^2 + (2L(\mathcal{J}(V_0) - \mathcal{L}^*))^{1/2}) \leq t + 5\sqrt{2}L(\|V_0\|^2 + 10(\mathcal{J}(V_0) - \mathcal{L}^*) + (2L(\mathcal{J}(V_0) - \mathcal{L}^*))^{1/2}).$$

We can therefore lower bound the series with a harmonic series as follows,

$$\begin{aligned}
\sum_{t=0}^{T-1} \frac{1}{D_t} \geq& \sum_{t=0}^{T-1} \frac{1}{t + 5\sqrt{2}L(\|V_0\|^2 + 10(\mathcal{J}(V_0) - \mathcal{L}^*) + (2L(\mathcal{J}(V_0) - \mathcal{L}^*))^{1/2})}, \\
=& \Theta(\log(T)).
\end{aligned}$$

Finally,

$$\min_{t=0,\ldots,T-1} \|\nabla \mathcal{J}(V_t)\|^2 \leq \frac{5(\mathcal{J}(V_0) - \mathcal{L}^*)}{\sum_{t=0}^{T-1} \eta_t} = O\left(\frac{1}{\log T}\right).$$

□

Table 1. Experiment hyperparameters.

Hyperparameter	Logistic regression	Logistic regression with ResNet-18 embeddings	ResNet-18
Batch size	512	512	512
LoRA rank	4	4	4
Data input dimension	3072	512	$32 \times 32 \times 3$
Epochs	20	40	75
α^{adapt}	1	1	
α^{norm}	0.1	0.1	0.05
Constant learning rate range	[0.05, 0.1, 0.2]	[0.05, 0.1, 0.12] (Higher values cause exploding loss)	[0.01, 0.02, 0.03]

B. Experimental Details

In the following, we provide additional details about our experimental setup. Table 1 displays the hyperparameters used in our experimental results.

Learning rate. While standard LoRA implementations tend to scale by both a scaling factor and $\frac{1}{r}$, for simplicity we just set the learning rate as a whole. To determine the range of learning rates to test in our experiments, we use a cyclical learning rate finder as introduced in (Smith, 2017) and further described in (Ahmed, 2025). We start with a small learning rate and train the model for a single epoch, repeating this with exponentially increasing learning rates. We plot the loss over the learning rates and pick learning rates in the range where the loss decreases the fastest. The resulting plots are available in Figure 3.

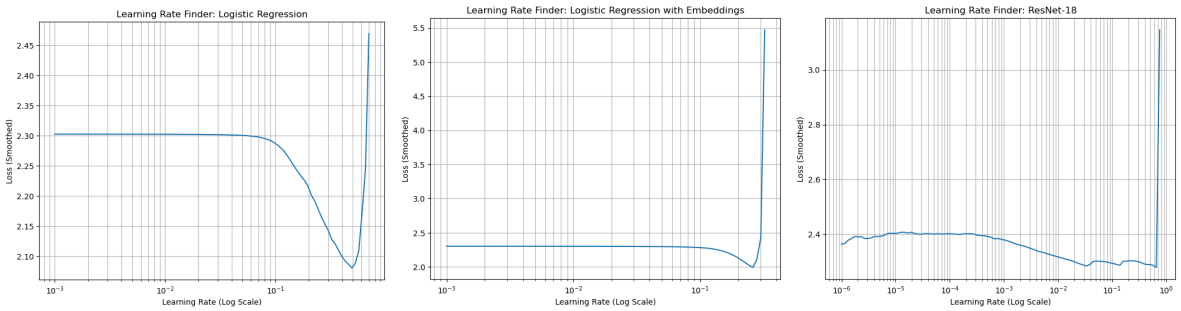

Figure 3. Learning rate selection using a cyclical finder for all three experimental settings.

Hardware and software. All experiments were run using PyTorch 2.5.0 and CUDA 12.1, on an Intel(R) Core(TM) i7-6850K CPU (3.60GHz) with an NVIDIA GeForce GTX 1080 GPU (8 GB VRAM).