Retraction-free optimization over the Stiefel manifold with application to the LoRA fine-tuning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Optimization over the Stiefel manifold has played a significant role in various machine learning tasks. Many existing algorithms either use the retraction operator to keep each iterate staying on the manifold, or solve an unconstrained quadratic penalized problem. The retraction operator in the former corresponds to orthonormalization of matrices and can be computationally costly for large-scale matrices. The latter approach usually equips with an unknown large penalty parameter. To address the above issues, we propose a retraction-free and penalty parameter-free algorithm, which lands on the manifold. A key component of the analysis is the convex-like property of the quadratic penalty of the Stiefel manifold, which enables us to explicitly characterize the penalty parameter. As an application, we introduce a new algorithm, Manifold-LoRA, which employs the landing technique and a carefully designed step size strategy to accelerate low-rank adaptation (LoRA) in fine-tuning large language models. Numerical experiments on the benchmark datasets demonstrate the efficiency of our proposed method.

1 Introduction

Optimization over the Stiefel manifold has attracted considerable attention in the context of machine learning, e.g., RNN [3], batch normalization [10], and distributionally robust optimization [8]. The mathematical formulation of this class of problems is:

$$\min_{X\in\mathbb{R}^{d\times r}} \quad f(X) \quad \text{subject to} \quad X \in \mathrm{St}(d,r) := \{X \in \mathbb{R}^{d\times r} : X^\top X = I_d\}, \tag{1}$$

where $r \leq d$ and $f : \mathbb{R}^{d\times r} \to \mathbb{R}$ is a continuously differentiable function. The most popular methods for solving (1) are retraction-based algorithms, which have been extensively studied in the context of manifold optimization [2, 23, 6]. Recently, to alleviate the possible computational burden of the retraction operator, some retraction-free methods have been developed in [19, 18, 41, 1]. The ideas in these papers are based on a combination of the manifold geometry and a penalty function for the manifold constraint, which involves an unknown but sufficiently large penalty parameter. For large-scale machine learning applications, retraction-free algorithms are preferred. However, designing retraction-free algorithms with a known penalty parameter for solving (1) remains a challenge.

Another motivation for studying retraction-free methods arises from its application in the fine-tuning of large language models (LLMs). Recently, LLMs have revolutionized the field of natural language processing (NLP), achieving unprecedented performance across various applications [33, 32]. To tailor pretrained LLMs for specific downstream tasks, the most common approach is full fine-tuning, which requires prohibitively large computational resources due to the need to adapt all model weights, hindering the deployment of large models. As a result, parameter-efficient fine-tuning (PEFT) has gained widespread attention for requiring few trainable parameters while delivering comparable

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

or even superior results to full fine-tuning. This paradigm involves inserting learnable modules or designating only a small portion of weights as trainable, keeping the main model frozen [21, 26, 44]. Among fine-tuning methods, low-rank adaptation (LoRA) [22] has become the de facto standard among parameter-efficient fine-tuning techniques. It assumes that the change in weights lies in a "low intrinsic dimension", thereby modelling the update $\Delta W \in \mathbb{R}^{d\times m}$ by two low-rank (not greater than a small integer r) matrices $A \in \mathbb{R}^{r\times m}$ and $B \in \mathbb{R}^{d\times r}$, i.e., $\Delta W = BA$. Since $r \ll d$, the requirements on both storage and computation are significantly reduced. Due to its decompositional nature, there is redundancy in the representation of ΔW. Traditional optimization methods for LoRA do not exploit this redundancy, which consequently undermines model performance. Instead, we reformulate LoRA fine-tuning as an optimization problem over the product of Stiefel manifolds and Euclidean spaces. Therefore, we propose an algorithmic framework called Manifold-LoRA to accelerate the fine-tuning process and enhance model performance. Moreover, by exploiting projected gradients and incorporating a parameter-free penalty, the overhead that our method incurs is relatively negligible. Our contributions are as follows:

- We first prove the existence of explicit choice for the penalty parameter by establishing a strong convexity-like condition of the nonconvex penalty problem associated with the Stiefel manifold constraint. Furthermore, for the given penalty parameter, under mild conditions, we prove that the iterates of our proposed retraction-free gradient descent method eventually land on the Stiefel manifold and achieve the optimality of (1).
- Building upon the established landing theory of retraction-free and penalty parameter-free method and the AdamW framework, we proposed a new method, Manifold-LoRA, which employs a carefully designed step size strategy to accelerate the training process of fine-tuning. Compared with the conventional AdamW method, we use the penalized gradient instead of the usual gradient, and the computational overhead is negligible.
- Numerical experiments are conducted on a wide range of NLP tasks, demonstrating the efficiency of our algorithm. Specifically, compared to the vanilla LoRA, our Manifold-LoRA with half the trainable parameters not only delivers fast convergence but also yields improved generalization. In particular, Our method converges twice as fast as baseline methods on several typical datasets, including the SQuAD 2.0 dataset and the CoLA dataset.

1.1 Related Work

Optimization over the Stiefel manifold. Optimization over the Stiefel manifold has attracted lots of attention due to its broad applications. Through the use of retraction, known as the generalization of the exponential map, the Riemannian gradient descent is proposed [2, 6, 23], where all iterates lie on the manifold. When such retraction is computationally costly, the authors [19] develop a retraction-free algorithm based on the augmented Lagrangian method. More recently, by defining the constraint dissolving operator and adding a sufficiently large penalty term, the authors [41] convert the manifold constrained problem (1) into an unconstrained problem and then apply unconstrained optimization algorithms. In [1], motivated by the convergence of the Oja's flow, a landing flow, consisting of the projected gradient and the gradient of the penalty function, is developed to retraction-free method for the squared Stiefel manifold, i.e., $d = r$. All of these methods rely on an unknown penalty parameter to ensure the convergence. This motivates us to design penalty parameter-free algorithms, which could significantly reduce the need for tuning parameters in practical implementations.

LoRA. There are numerous variants of LoRA aiming to improve performance or reduce memory usage. AdaLoRA [46], a well-known successor, introduces the idea of adaptively adjusting the rank of different layers by incorporating an additional vector $\boldsymbol{g}$ to serve as the diagonal of a singular value matrix. This approach leverages a revised sensitivity-based importance measure to decide whether to disable entries in vector $\boldsymbol{g}$ and in matrices A and B. A similar work, SoRA [15], adopts the same model architecture as AdaLoRA, but proposes a different way to update vector $\boldsymbol{g}$ after training. This update rule is the proximal gradient of $\mathcal{L}_1$ loss, acting as a post-pruning method. Additionally, a recently emerged method called VeRA [25] significantly reduces memory overhead while maintaining competitive performance. Based on the idea that networks with random initialization contain subnetworks that are near-optimal or optimal [17], VeRA only uses two frozen low-rank matrices shared by all layers, training scaling vectors unique to each layer. Although LoRA has gained significant popularity and various variants have been developed, the potential for efficient training through leveraging the manifold geometry to reduce redundancy has not been well-explored.

1.2 Notation

For a matrix $X \in \mathbb{R}^{d\times r}$, we use $\|X\|$ to denote its Frobenius norm. For a squared matrix $A \in \mathbb{R}^{d\times d}$, we define $\mathrm{sym}(A) = \frac{A+A^\top}{2}$ and use $\mathrm{diag}(A) \in \mathbb{R}^d$ to denote its diagonal part. For two matrices $X, Y \in \mathbb{R}^{d\times r}$, we use $\langle X, Y\rangle := \sum_{i=1}^{d}\sum_{j=1}^{r} X_{ij}Y_{ij}$ to denote their Euclidean inner product. For a differential function $f : \mathbb{R}^{d\times r} \to d$, we use $\nabla f(X)$ to denote its Euclidean gradient at X.

2 Retraction-free and penalty parameter-free optimization over the Stiefel manifold

In this section, we focus on the design of retraction-free and penalty parameter-free algorithms for solving problem (1). We will first present the retraction-free algorithm and then show how the penalty parameter can be explicitly determined by characterizing the landscape of the penalty function.

2.1 Retraction-free algorithms

Inspired by the retraction-free algorithms [19, 41, 1], we consider the following retraction-free gradient descent method for problem (1):

$$X_{k+1} = X_k - \alpha \mathrm{grad} f(X_k) - \mu X_k(X_k^\top X_k - I_d), \tag{2}$$

where $\alpha, \mu > 0$ are step sizes and the projected gradient $\mathrm{grad} f(X_k) := \nabla f(X_k) - X_k \mathrm{sym}(X_k^\top \nabla f(X_k))$. Note that the tangent space of $\mathrm{St}(d, r)$ is $T_{X_k}\mathrm{St}(d, r) := \{\xi \in \mathbb{R}^{d\times r} : X_k^\top \xi + \xi^\top X_k = 0\}$. Then, for $X_k \in \mathrm{St}(d, r)$, $\mathrm{grad} f(X_k)$ is the projection of the Euclidean gradient $\nabla f(X_k)$ to the tangent space, i.e., $\mathrm{grad} f(X_k) = \mathcal{P}_{T_{X_k}\mathrm{St}(d,r)}(\nabla f(X_k))$. Note that the term $X_k(X_k^\top X_k - I_d)$ is exactly the gradient of the following quadratic penalty function

$$\varphi(X) := \frac{1}{4}\|X^\top X - I\|^2.$$

As will be shown in our theorem, the use of the projected gradient is essential for landing on the manifold. This differs with the usual penalty method, which optimizes $f(X) + \mu\varphi(X)$ using the update $X_{k+1} = X_k - \alpha\nabla f(X_k) - \mu X_k(X_k^\top X_k - I_d)$, needs $\mu \to \infty$ to guarantee the feasibility.

2.2 Explicit choice for the penalty parameter

It is known that a large penalty parameter yields better feasibility [29, Chapter 17]. To make the iterative scheme (2) be penalty parameter-free, we need a careful investigation on the landscape of the following optimization problem:

$$\min_{X\in\mathbb{R}^{d\times r}} \quad \varphi(X). \tag{3}$$

It can be easily verified that problem (3) is nonconvex and its the optimal solution set is $\mathrm{St}(d, r)$. The key of obtaining an explicit formula of μ is to establish certain strong convexity-type inequality and show the gradient descent method with step size μ has linear convergence.

For any $X \in \mathrm{St}(d, r)$, let us denote $\bar{X} := \mathcal{P}_{\mathrm{St}(d,r)}(X)$. Let $X = USV^\top$ be the singular value decomposition with orthogonal matrices $U \in \mathbb{R}^{d\times r}, V \in \mathbb{R}^{d\times d}$ and diagonal matrix $S \in \mathbb{R}^{d\times d}$, then $\bar{X} = UV^\top$. Building on these notations, we demonstrate that problem (3) satisfies the restrict secant inequality (RSI) [45], which serves as an alternative to the strong convexity in the linear convergence analysis of gradient-type methods.

Lemma 1. *For any $X \in \mathbb{R}^{d\times r}$ with $\|X - \bar{X}\| \le \frac{1}{8}$, we have*

$$\left\langle \nabla\varphi(X), X - \bar{X}\right\rangle \ge \|X - \bar{X}\|^2. \tag{4}$$

With the above RSI, we have the linear convergence of the gradient descent update for (3), i.e.,

$$X_{k+1} = X_k - \mu\nabla\varphi(X_k). \tag{5}$$

Lemma 2. *Let the sequence $\{X_k\}$ be generated by* (5) *with $\mu = \frac{1}{3}$. Suppose that $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. We have*

$$\|X_{k+1} - \bar{X}_{k+1}\|^2 \le \frac{2}{3}\|X_k - \bar{X}_k\|^2. \tag{6}$$

The proofs of Lemmas 1 and 2 can be found in Appendix B.

2.3 Landing on the Stiefel manifold

Building on the established linear convergence of gradient descent for problem (3), we are now able to show that the iterates generated by (2) will land on the Stiefel manifold eventually, and the limiting point is a stationary point of (1), i.e., $\mathrm{grad} f(X_\infty) = 0$.

Let us start with the Lipschitz continuity of $\mathrm{grad} f(X)$. For any $X \in \bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8})$, we define $\mathcal{P}_{T_X \mathrm{St}(d,r)}(U) = U - X\mathrm{sym}(X^\top U)$ for $U \in \mathbb{R}^{d\times r}$. We first have the following quadratic upper bound on f from its twice differentiability and the compactness of $\mathrm{St}(d,r)$.

Lemma 3. *There exists a constant $L > 0$ such that for any $X, Y \in \mathrm{St}(d,r)$, the following quadratic upper bound holds:*

$$f(Y) \le f(X) + \langle \mathrm{grad} f(X), Y - X\rangle + \frac{L}{2}\|Y - X\|^2. \tag{7}$$

In addition, there exists a constant $\hat{L} > 0$ such that for any $X \in \mathrm{St}(d,r), Y \in U_{\mathcal{M}}(\frac{1}{8})$,

$$\|\mathrm{grad} f(X) - \mathrm{grad} f(Y)\| \le \hat{L}\|X - Y\|. \tag{8}$$

By the linear convergence result in Lemma 2, we have the following bound on the feasibility error.

Lemma 4. *Let $\{X_k\}$ be the sequence generated by (2) with $\mu = \frac{1}{3}$ and $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. We have*

$$\|X_{k+1} - \bar{X}_{k+1}\| \le \frac{2}{3}\|X_k - \bar{X}_k\| + \alpha\|\mathrm{grad} f(X_k)\|. \tag{9}$$

The following one-step descent lemma on f is crucial in establishing the convergence.

Lemma 5. *Let $\{X_k\}$ be the sequence generated by (2) with $\mu = \frac{1}{3}$ and $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. We have*

$$\begin{aligned} f(\bar{X}_{k+1}) - f(\bar{X}_k) \le & -(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2)\|\mathrm{grad} f(X_k)\|^2 + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 \\ & + \frac{1}{2}\left(4\hat{D}_f + 16\hat{L}^2 + 16L + 3\right)\|X_k - \bar{X}_k\|^2. \end{aligned} \tag{10}$$

From the above lemma, the one-step descrease on f is related to both the gradient norm of f and the feasibility error. In terms of convergence, we need both $\mathrm{grad} f(X_k)$ and $\|X_k^\top X_k - I\|$ converge to 0. The following theorem demonstrates that the retraction-free and penalty parameter-free update (2) converges.

Theorem 1. *Let $\{X_k\}$ be the sequence generated by (2) with $\mu = \frac{1}{3}$ and $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. If the step size $\alpha < \frac{1}{2c_1}$ for some c_1 large enough, then we have*

$$\min_{k=0,\dots,K} \|\mathrm{grad} f(X_k)\|^2 \le \frac{1}{K}, \quad \min_{k=0,\dots,K} \|X_k^\top X_k - I\|^2 \le \frac{1}{K}. \tag{11}$$

The proofs of the above lemmas and theorem are presented in Appendix B.

3 Accelerate LoRA fine-tuning with landing

In this section, we will first clarify where the Stiefel manifold constraint comes from in the LoRA fine-tuning. Then, we will apply the above developed retraction-free and penalty parameter-free method to enhance LoRA fine-tuning.

3.1 Manifold optimization formulation of LoRA fine-tuning

In neural networks, the dense layers perform matrix multiplication, and the weight matrices in these layers usually have a full rank. However, when adapting to a specific task, pre-trained language models have been shown to have a low intrinsic dimension, allowing them to learn efficiently even with a random projection to a smaller subspace. One possible drawback in the current LoRA fine-tuning framework is that the low-rank decomposition ΔW into product BA is not unique. Specifically, for any invertible matrix C, it holds that $BA = (BC)(C^{-1}A)$. Note that BC shares the same

column space with B. This suggests us optimizing the subspace generated by B instead of B itself. Numerous studies in the field of low-rank optimization, e.g., [7, 13, 12], investigate the manifold geometry of the low-rank decomposition and develop efficient algorithms. However, such geometry has not been explored in the LoRA fine-tuning.

To address such redundancy (i.e., the non-uniqueness of BA representations), we regard B as the basis through the manifold constraint and A as the coordinate of ΔW under B. Hence, the optimization problem can be formulated as

$$\min_{A,B} \quad L(BA), \quad \text{subject to} \quad B \in \mathrm{St}(d,r) \text{ or } B \in \mathrm{Ob}(d,r), \tag{12}$$

where $\mathrm{Ob}(d,r) := \{B \in \mathbb{R}^{d\times r} : \mathrm{diag}(B^\top B) = \mathbf{1}\}$. Compared to the Stiefel manifold $\mathrm{St}(d,r)$, the oblique manifold $\mathrm{Ob}(d,r)$ necessitates that the matrix B has unit norms in its columns, without imposing requirements for orthogonality between the columns. Problem (12) is an optimization problem over the product of manifolds and Euclidean spaces.

3.2 Manifold-LoRA

The retraction-free method is well-suited to address (12), simultaneously minimizing the loss function $L(BA)$ and constraint violation. To control the constraint violation, we use the quadratic penalties $R_s(B) := \|B^\top B - I\|^2$ and $R_o(B) := \|\mathrm{diag}(B^\top B) - 1\|^2$ for the Stiefel manifold and oblique manifold, respectively. As shown in the landing theory in Section 2, we shall use the projected gradient of the loss part instead of the Euclidean gradient. For the Stiefel manifold and the oblique manifold, the respective projected gradients are

$$\mathrm{grad}_B L(BA) = \nabla_B L(BA) - B\mathrm{sym}(B^\top \nabla_B L(BA)) \tag{13}$$

and

$$\mathrm{grad}_B L(BA) = \nabla_B L(BA) - B\mathrm{diag}(\mathrm{diag}(B^\top \nabla_B L(BA))), \tag{14}$$

where $\mathrm{sym}(X) := (X + X^\top)/2$. Thus, the gradients of our retraction-free method for A and B are $\nabla_A L(BA)$ and $\mathrm{grad}_B L(BA) + \mu\nabla R_s(B)$(or $\nabla R_o(B)$).

Note that B and A represent the basis and the coordinate of ΔW, respectively. This results in different magnitudes and different Lipschitz constants of their gradient function. In fact, let $X = BA$. It follows

$$\nabla_A L(BA) = B^\top \nabla_X L(X), \quad \nabla_B L(BA) = \nabla_X L(X) A^\top.$$

Then,

$$\|\nabla_A L(BA_1) - \nabla L(BA_2)\| \le \|B\|_2 L_g \|A_1 - A_2\|,$$
$$\|\nabla_B L(B_1A) - \nabla L(B_2A)\| \le \|A\|_2 L_g \|B_1 - B_2\|,$$

where L_g is the Lipschitz constant of $\nabla_X L(X)$ and $\|\cdot\|_2$ represent the matrix ℓ_2 norm (i.e., the largest singular value). Note that the step size generally should be propositional to the reciprocal of Lipschitz constant for the gradient type algorithms [29, 5]. Hence, we schedule the learning rates for the two matrices based on their respective ℓ_2 norms. Having prepared the above, we incorporate the AdamW optimizer [28] with our manifold-accelerated technique to enhance the LoRA fine-tuning, as presented in Algorithm 1.

4 Experiments

In this section, we delve into the experimental results and their detailed analysis. This discussion is structured around two principal areas: (1) the performance gain compared to other mainstream fine-tuning methods and accelerated convergence achieved through our manifold-constrained optimization approach; (2) the convergence of matrix B onto the manifold, illustrated by the heat map of $B^\top B$.

Baselines We compare our approach against several baseline methods, including full fine-tuning, Adapter [21], BitFit [44] and LoRA [22]. The variants of the Adapter method are excluded from the baselines, as their performance are relatively similar.

Implementation Details Our code is based on Pytorch [31], Huggingface Transformers [40] and an open-source plug-and-play library for parameter-efficient fine-tuning opendelta [24]. The bottleneck dimension for the Adapter is set to 16 or 32, ensuring that the number of trainable parameters aligns

Algorithm 1: Manifold-LoRA

Input: Initial point A_0, B_0, $\mu \in \mathbb{R}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $upper\_bound \geq lower\_bound > 0$, $\epsilon = 10^{-8}$, $\gamma > 0$, $\lambda \in \mathbb{R}$, and $k = 0$.

while *Stopping conditions not met* **do**
  for $C \in \{A, B\}$ **do**
    if $C = B$ **then**
      Set $g(C_k)$ according to (13) or (14) using the stochastic estimate of $\nabla_B L(B_k A_k)$
      `// Projected gradient for matrix` B
    else
      Set $g(C_k)$ to be the stochastic estimate of $\nabla_A L(B_k A_k)$
    end
  end
  $m(C_k) \leftarrow \beta_1 m(C_k) + (1-\beta_1) g(C_k)$
  $v(C_k) \leftarrow \beta_2 v(C_k) + (1-\beta_2) g_t^2(C_k)$
  $\hat{m}(C_k) \leftarrow \frac{m(C_k)}{1-\beta_1^t}$
  $\hat{v}(C_k) \leftarrow \frac{v(C_k)}{1-\beta_2^t}$
  $\eta(C_k) \leftarrow clip(\text{norm}_{C_k}, upper\_bound, lower\_bound)$
  `// Scheduling step size of matrix A and B`
  $C_k \leftarrow C_{k-1} - \eta_t(C_k)\left(\hat{m}_t(C_k)/\left(\sqrt{\hat{v}_t(C_k)} + \epsilon\right)\right) - \lambda C_{k-1}$
  if $C = B$ **then**
    $C_k \leftarrow C_k - \mu \nabla R_s(C_k)(\text{ or } \nabla R_o(C_k))$ `// Apply penalty gradient for matrix` B
  end
end
$k \leftarrow k + 1$
end

closely with that of the LoRA method and the new layers are inserted into the attention layer and feed-forward layer. The update of LoRA is scaled by a hyper-parameter α. This value is typically left unmodified, as it is usually set as 16 or 32 and never tuned [22, 43]. The exponential moving average parameters β_1 and β_2 of AdamW [27] are set to their default values of 0.9 and 0.999, respectively. All the experiments are conducted on NVIDIA A800 GPUs. More details are presented in Appendix C.

4.1 Natural language understanding

We first evaluate our backbone model DeBERTaV3-base [20] on GLUE [37] benchmark containing nine sub datasets, including MNLI [39], SST-2 [36], CoLA [38], QQP [37], QNLI [35], RTE [4], MRPC [16], and STS-B [37].

Experimental results of the GLUE dataset are recorded in Table 1. It can be seen that our method is consistently superior to other baselines. Notably, for RTE and STS-B datasets, both sphere-constrained (i.e., oblique manifold-constrained) and Stiefel-constrained have an obvious performance gain even with only half the trainable parameters compared to the LoRA baseline, i.e., $\text{Sphere}_{r=8}$ and $\text{Stiefel}_{r=8}$ beat $\text{LoRA}_{r=16}$. In addition, with the help of manifold geometry, the fine-tuning process can be significantly accelerated compared to the vanilla AdamW optimizer, achieving a lower training loss, as shown in Figure 1. Particularly on the CoLA dataset presented in Figure 1a, our approach achieves the same training loss as the standard Adam optimizer but requires nearly half the number of epochs.

4.2 Question Answering

We conduct an evaluation on two question answering datasets: SQuAD v1.1 [35] and SQuADv2.0 [34]. Manifold-LoRA is used to fine-tune DeBERTaV3-base for these tasks, which are treated as sequence labeling problems predicting the probability of each token as the start or end of an answer span.

The main experimental results are presented in Table 2. For LoRA and our algorithms, new layers are inserted into $W_q, W_k, W_v, W_o, FC_1, FC_2$. Notably, both manifold-regularized LoRA variants consistently outperform all fine-tuning methods. Additionally, we plot the training loss, evaluation

Table 1: Results with DeBERTaV3-base on GLUE benchmark. We denote the best results in **bold**.

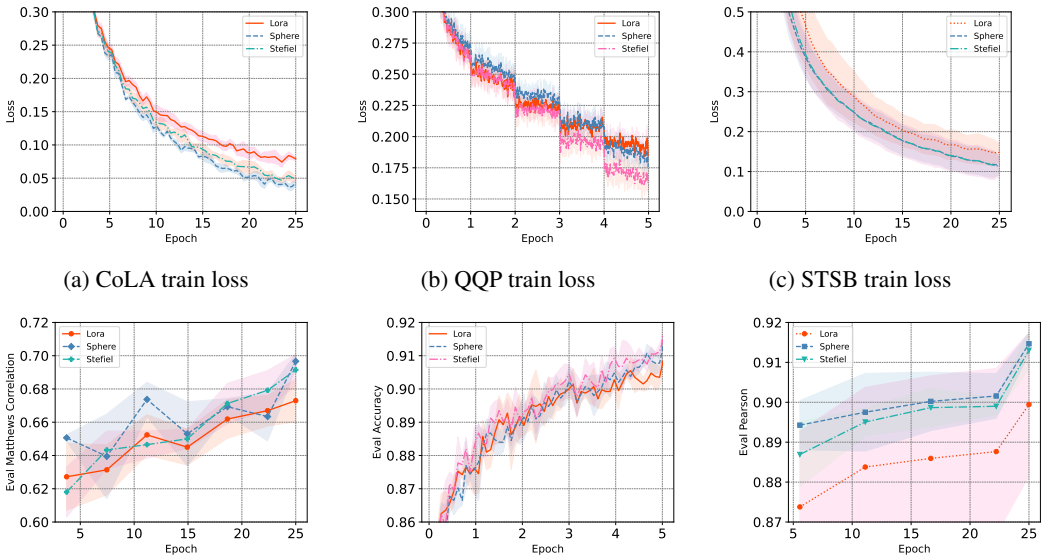
Method	# Params	MNLI m / mm	SST-2 Acc	CoLA Mcc	QQP Acc / F1	QNLI Acc	RTE Acc	MRPC Acc	STS-B Corr	All Ave.
Full FT	184.42M	90.45/**90.60**	95.48	68.17	**91.99/89.12**	93.60	79.28	88.93	90.92	87.85
Adapter	0.61M	90.13/90.16	94.86	69.37	91.38/88.46	93.54	81.87	89.12	91.52	88.06
BitFit	0.06M	87.08/86.39	94.88	69.11	87.96/84.35	92.19	76.52	87.06	90.96	85.65
LoRA$_{r=8}$	0.30M	90.20/90.08	94.93	68.14	90.78/87.68	93.85	80.15	90.40	90.29	87.60
LoRA$_{r=16}$	0.59M	90.44/90.12	95.41	68.19	90.92/87.77	94.00	80.58	90.20	90.34	87.74
Sphere$_{r=8}$	0.30M	90.37/90.09	95.48	69.55	91.25/88.34	94.02	82.44	91.55	91.26	88.44
Sphere$_{r=16}$	0.59M	**90.52**/90.19	95.64	**70.14**	91.46/88.65	**94.29**	82.16	**91.67**	**91.59**	**88.63**
Stiefel$_{r=8}$	0.30M	90.25/89.99	95.46	69.85	91.44/88.60	94.09	**83.16**	91.18	91.22	88.52
Stiefel$_{r=16}$	0.59M	90.26/90.28	**95.76**	68.92	91.71/89.00	94.10	82.16	91.10	91.51	88.48

exact match, and evaluation F1 scores against epochs in Figure 2. We conclude that the proposed Manifold-LoRA method achieves a 2x speed-up in training epochs compared to AdamW, while simultaneously improving model performance. We also illustrate the heat map of $B^\top B$ in Figure 3, which indicates that the matrix B lands on the manifold eventually. This supports our assertion that landing on manifold enhances the performance of LoRA.

4.3 Natural Language Generation

The E2E NLG Challenge[30], as introduced by Novikova, provides a dataset for training end-to-end, data-driven natural language generation systems, widely used in data-to-text evaluations. The E2E dataset comprises approximately 42,000 training examples, 4,600 validation examples, and 4,600 test examples, all from the restaurant domain. We test our method on the E2E dataset using GPT-2 Medium and Large models, following the experimental setup outlined by LoRA [22]. For LoRA, we set the hyperparameters to match those specified in the original paper.

The results from the E2E dataset are recorded in Table 3, where we focus on comparing LoRA and Manifold-LoRA. The results clearly indicate that our proposed algorithm outperforms the established baselines. Also, as shown in Figure 4, the matrix B resides on the manifold even at the early training stage, validating the feasibility of our method.



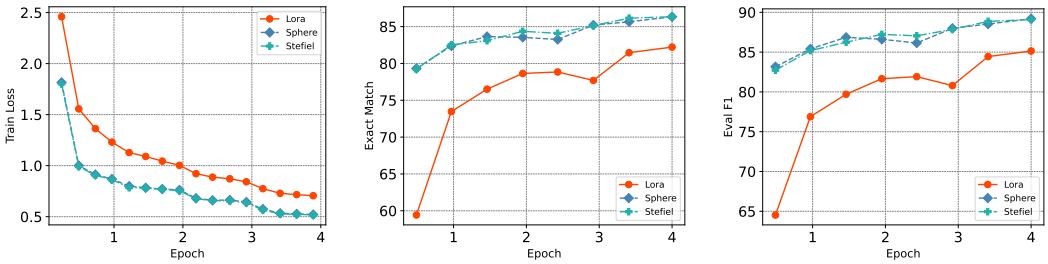
(a) CoLA train loss (b) QQP train loss (c) STSB train loss

(d) CoLA evaluation matthews correlation (e) QQP evaluation accuracy (f) STSB evaluation pearson

Figure 1: The figures illustrate that both sphere constrained and Stiefel constrained manifold-LoRA achieve a faster convergence rate and attain a lower training loss within same optimization steps compared to LoRA method on three distinct datasets CoLA, QQP, STSB.

Table 2: Results with DeBERTaV3-base on SQuAD v1.1 and SQuADv2.0. We report EM/F1. The best results in each setting are shown in **bold**.

Methods	Params	SQuADv1.1	SQuADv2.0
Full FT	184M	86.30 / 92.85	84.30 / 87.58
Adapter$_{r=16}$	0.61M	87.46 / 93.41	85.30 / 88.23
Adapter$_{r=32}$	1.22M	87.53 / 93.51	85.42 / 88.36
Bitfit	0.07M	80.26 / 88.79	74.21 / 87.19
LoRA$_{r=8}$	1.33M	87.90 / 93.88	85.56 / 88.52
LoRA$_{r=16}$	2.65M	87.94 / 93.75	85.90 / 88.81
Sphere$_{r=8}$	1.33M	88.51 / **94.25**	86.33 / 89.20
Sphere$_{r=16}$	2.65M	88.32 / 94.03	86.15 / 89.03
Stiefel$_{r=8}$	1.33M	**88.68** / 94.23	86.35 / 89.09
Stiefel$_{r=16}$	2.65M	88.25 / 94.04	**86.41** / **89.22**



(a) SQuADv2.0 Train Loss (b) SQuADv2.0 Eval Exact Match (c) SQuADv2.0 Eval F1

Figure 2: The figures compare the training loss, evaluation exact match, and evaluation F1 metrics against the number of epochs for the SQuADv2.0 dataset.

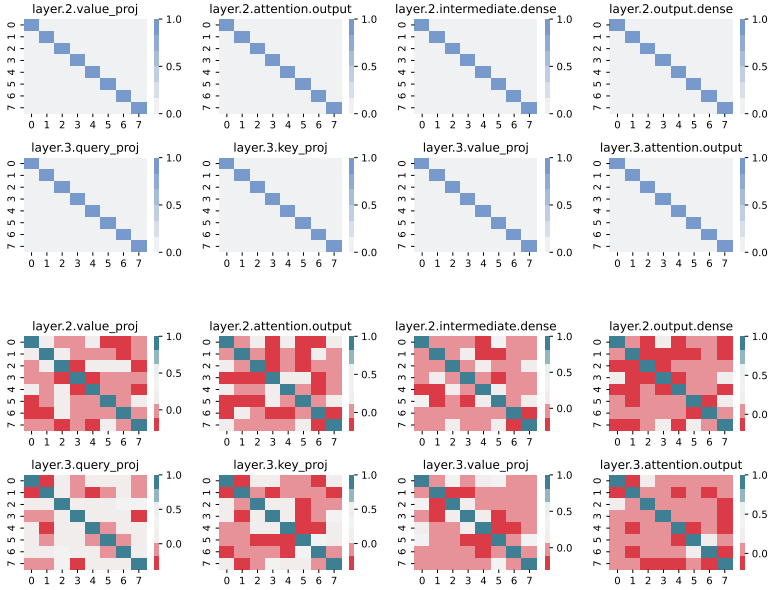


Figure 3: The heat map of $B^\top B$ with the Stiefel manifold (the first and second rows) and the oblique manifold (the third and fourth rows) at the end of training on SQuADv2.0 dataset.

Table 3: GPT-2 medium (M) and large (L) models were evaluated on the E2E NLG Challenge. * denotes results from previously published works.

Model	Parameters	BLEU	NIST	MET	ROUGE-L	CIDEr
GPT-2 M (FT)*	354.92M	68.2	8.62	46.2	71.0	2.47
GPT-2 M (Adapter$^{\text{L}}$)*	0.37M	66.3	8.41	45.0	69.8	2.40
GPT-2 M (Adapter$^{\text{L}}$)*	11.09M	68.9	8.71	46.1	71.3	2.47
GPT-2 M (Adapter$^{\text{H}}$)*	11.09M	$67.3_{\pm.6}$	$8.50_{\pm.07}$	$46.0_{\pm.2}$	$70.7_{\pm.2}$	$2.44_{\pm.01}$
GPT-2 M (FT$^{\text{Top2}}$)*	25.19M	68.1	8.59	46.0	70.8	2.41
GPT-2 M (PreLayer)*	0.35M	69.7	8.81	46.1	71.4	2.49
GPT-2 M (LoRA)	0.35M	68.9	8.69	46.5	71.5	2.51
GPT-2 M(Stiefel)	0.35M	70.1	8.82	**46.8**	**71.7**	**2.53**
GPT-2 M(Sphere)	0.35M	**70.3**	**8.83**	46.7	**71.7**	2.52
GPT-2 L (FT)*	774.03M	68.5	8.78	46.0	69.9	2.45
GPT-2 L (Adapter$^{\text{L}}$)*	0.88M	$69.1_{\pm.1}$	$8.68_{\pm.03}$	$46.3_{\pm.0}$	$71.4_{\pm.2}$	$2.49_{\pm.0}$
GPT-2 L (Adapter$^{\text{L}}$)*	23.00M	$68.9_{\pm.3}$	$8.70_{\pm.04}$	$46.1_{\pm.1}$	$71.3_{\pm.2}$	$2.45_{\pm.02}$
GPT-2 L (PreLayer)*	0.77M	70.3	8.85	46.2	71.7	2.47
GPT-2 L (LoRA)	0.77M	70.1	8.82	46.7	72.0	2.53
GPT-2 L(Stiefel)	0.77M	70.4	8.86	**46.8**	72.1	2.53
GPT-2 L(Sphere)	0.77M	**70.9**	**8.92**	**46.8**	**72.5**	**2.55**

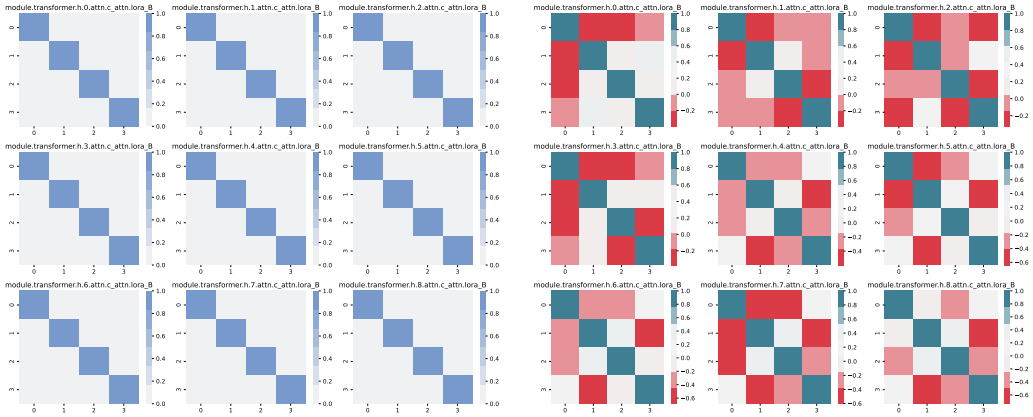


Figure 4: The heat map of $B^\top B$ with the Stiefel manifold (left) and the oblique manifold (right) on E2E dataset.

5 Conclusion

Optimization over the Stiefel manifold has been widely used in machine learning tasks. In this work, we develop a retraction-free and penalty parameter-free gradient method, and prove that the generated iterates eventually land on the manifold and achieve the optimality simultaneously. We then apply this landing theory to avoid the possible redundancy of LoRA fine-tuning in LLMs. Specifically, we reformulate the LoRA fine-tuning as an optimization problem over the Stiefel manifold, and propose a new algorithm, Manifold-LoRA, which incorporates a careful analysis of step sizes to enable fast training using the landing properties. Extensive experimental results demonstrate that our approach not only accelerates the training process but also yields significant performance improvements.

Our study suggests several potential directions for future research. Although the established landing theory focuses on the Stiefel manifold, extending this theory to general manifolds is one potential direction. Additionally, evaluating the performance of Manifold-LoRA on LLMs with billions of parameters would be valuable. Due to the heterogeneity of different layers, incorporating adaptive ranks for ΔW across different layers is another possible direction. This may be achievable by adding sparsity regularization to the coordinate matrix A.

References

[1] Pierre Ablin and Gabriel Peyré. Fast and accurate optimization on the orthogonal manifold without retraction. In *International Conference on Artificial Intelligence and Statistics*, pages 5636–5657. PMLR, 2022.

[2] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

[3] Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In *International conference on machine learning*, pages 1120–1128. PMLR, 2016.

[4] Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1, 2009.

[5] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

[6] Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

[7] Nicolas Boumal and Pierre-antoine Absil. Rtrmc: A riemannian trust-region method for low-rank matrix completion. *Advances in neural information processing systems*, 24, 2011.

[8] Robert S Chen, Brendan Lucier, Yaron Singer, and Vasilis Syrgkanis. Robust optimization for non-convex objectives. *Advances in Neural Information Processing Systems*, 30, 2017.

[9] Shixiang Chen, Alfredo Garcia, Mingyi Hong, and Shahin Shahrampour. Decentralized Riemannian gradient descent on the Stiefel manifold. In *International Conference on Machine Learning*, pages 1594–1605. PMLR, 2021.

[10] Minhyung Cho and Jaehyung Lee. Riemannian approach to batch normalization. *Advances in Neural Information Processing Systems*, 30, 2017.

[11] Francis H Clarke, Ronald J Stern, and Peter R Wolenski. Proximal smoothness and the lower-C2 property. *Journal of Convex Analysis*, 2(1-2):117–144, 1995.

[12] Wei Dai, Ely Kerman, and Olgica Milenkovic. A geometric approach to low-rank matrix completion. *IEEE Transactions on Information Theory*, 58(1):237–247, 2012.

[13] Wei Dai, Olgica Milenkovic, and Ely Kerman. Subspace evolution and transfer (set) for low-rank matrix completion. *IEEE Transactions on Signal Processing*, 59(7):3120–3132, 2011.

[14] Kangkang Deng and Jiang Hu. Decentralized projected riemannian gradient method for smooth optimization on compact submanifolds. *arXiv preprint arXiv:2304.08241*, 2023.

[15] Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. *arXiv preprint arXiv:2311.11696*, 2023.

[16] Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*, 2005.

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

[18] Bin Gao, Guanghui Hu, Yang Kuang, and Xin Liu. An orthogonalization-free parallelizable framework for all-electron calculations in density functional theory. *SIAM Journal on Scientific Computing*, 44(3):B723–B745, 2022.

[19] Bin Gao, Xin Liu, Xiaojun Chen, and Ya-xiang Yuan. A new first-order algorithmic framework for optimization problems with orthogonality constraints. *SIAM Journal on Optimization*, 28(1):302–332, 2018.

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

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

[22] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[23] Jiang Hu, Xin Liu, Zai-Wen Wen, and Ya-Xiang Yuan. A brief introduction to manifold optimization. *Journal of the Operations Research Society of China*, 8:199–248, 2020.

[24] Shengding Hu, Ning Ding, Weilin Zhao, Xingtai Lv, Zhen Zhang, Zhiyuan Liu, and Maosong Sun. Opendelta: A plug-and-play library for parameter-efficient adaptation of pre-trained models. *arXiv preprint arXiv:2307.03084*, 2023.

[25] Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki Markus Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

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

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[29] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

[30] Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. The e2e dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*, 2017.

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[32] Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*, 2023.

[33] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[34] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[35] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[36] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[37] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

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

[39] Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

[40] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[41] Nachuan Xiao, Xin Liu, and Kim-Chuan Toh. Dissolving constraints for riemannian optimization. *Mathematics of Operations Research*, 49(1):366–397, 2024.

[42] Jinming Xu, Shanying Zhu, Yeng Chai Soh, and Lihua Xie. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 2055–2060. IEEE, 2015.

[43] Greg Yang and Edward J Hu. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*, 2020.

[44] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

[45] Hui Zhang and Wotao Yin. Gradient methods for convex minimization: better rates under weaker conditions. *arXiv preprint arXiv:1303.4645*, 2013.

[46] 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.

A Proximal smoothness

The notion of proximal smoothness, as introduced by [11], refers to the characteristic of a closed set whereby the nearest-point projection becomes a singleton when the point is in close enough to the set. This property facilitates algorithmic and theoretical advancements by endowing nonconvex sets with convex-like structural attributes. Specifically, for any positive real number γ, we define the γ-tube around $\mathcal{M}$ as $U_{\mathcal{M}}(\gamma) := \{x : \mathrm{dist}(x, \mathcal{M}) < \gamma\}$. We say a closed set $\mathcal{M}$ is γ-proximally smooth if the projection operator $\mathcal{P}_{\mathcal{M}}(x) := \mathrm{argmin}_{y\in\mathcal{M}} \|y - x\|^2$ is a singleton whenever $x \in U_{\mathcal{M}}(\gamma)$.

Obviously, any closed and convex set is proximally smooth for arbitrary $\gamma \in (0, \infty)$. According to [11, Corollary 4.6], a closed set $\mathcal{M}$ is convex if and only if it is proximally smooth with a radius of γ for every $\gamma > 0$. It is worth noting that the Stiefel manifold is 1-proximally smooth. By following the proof in [11, Theorem 4.8],

$$\left\|\mathcal{P}_{\mathrm{St}(d,r)}(x) - \mathcal{P}_{\mathrm{St}(d,r)}(y)\right\| \le 2\|x - y\|, \ \ \forall x, y \in \bar{U}_{\mathrm{St}(d,r)}(\frac{1}{2}), \tag{15}$$

where $\bar{U}_{\mathrm{St}(d,r)}(\frac{1}{2}) := \{x : \mathrm{dist}(x, \mathrm{St}(d, r)) \le \frac{1}{2}\}$ is the closure of $U_{\mathrm{St}(d,r)}(\frac{1}{2})$. It is worth noting that for any closed convex set $\mathcal{M} \subset \mathbb{R}^{d\times r}$, the projection operator $\mathcal{P}_{\mathcal{M}}$ is 1-Lipschitz continuous over $\mathbb{R}^{d\times r}$.

B Proofs

Proof of Lemma 1

Proof. Denote the SVD of X by $X = USV^\top$. Then, it holds that $\mathrm{dist}(X, \mathrm{St}(d, r)) = \|X - \bar{X}\| = \|s - 1\|_2$, where $s = \mathrm{diag}(S)$. Furthermore, we have

$$\begin{aligned}
\left\langle \nabla\varphi(X), X - \bar{X}\right\rangle &= \left\langle USV^\top(VS^2V^\top - I), USV^\top - UV^\top\right\rangle \\
&= \left\langle U(S^3 - S)V^\top, U(S - I)V^\top\right\rangle \\
&= \mathrm{tr}((S^3 - S)(S - I)) \\
&\ge \frac{3}{2}\|s - 1\|_2^2 = \frac{3}{2}\|X - \bar{X}\|^2,
\end{aligned}$$

where the last inequality is from $\min_i s_i(s_i + 1) \ge \frac{105}{64} \ge \frac{3}{2}$. This completes the proof. □

Proof of Lemma 2

Proof. Assume that $\|X_k - \bar{X}_k\| \le \frac{1}{8}$. Denote the SVD of X_k by $USV^\top$. Let $s = \mathrm{diag}(S)$. Then, we have $\frac{7}{8} \le s_i \le \frac{9}{8}$ for any i. This implies

$$\|\nabla\varphi(X_k)\|^2 = \mathrm{tr}((S^3 - S)^2) \le 6\|X_k - \bar{X}_k\|^2. \tag{16}$$

Hence, we have

$$\begin{aligned}
\|X_{k+1} - \bar{X}_{k+1}\|^2 &\le \|X_{k+1} - \bar{X}_k\|^2 \\
&= \|X_k - \frac{1}{3}\nabla\varphi(X_k) - \bar{X}_k\|^2 \\
&= \|X_k - \bar{X}_k\|^2 - \frac{2}{3}\left\langle X_k - \bar{X}_k, \nabla\varphi(X_k)\right\rangle + \frac{1}{9}\|\nabla\varphi(X_k)\|^2 \\
&\le (1 - 1 + \frac{2}{3})\|X_k - \bar{X}_k\|^2 \\
&= \frac{2}{3}\|X_k - \bar{X}_k\|^2,
\end{aligned}$$

where the first inequality is from $\bar{X}_{k+1} = \mathrm{argmin}_{X\in\mathrm{St}(d,r)} \|X - X_k\|^2$ and the second inequality is due to Lemma 1 and (16). □

Proof of Lemma 3

Proof. Due to the twice differentiability of f and the compactness of $\mathrm{St}(d,r)$, the inequality (7) directly follows from [9, Lemma 2.4] and [14, Lemma 4.2], where $L := L_f + D_f$ with L_f being the Lipschitz constant of $\nabla f(X)$ over $\mathrm{St}(d,r)$ and $D_f := \max_{X\in\mathrm{St}(d,r)} \|\nabla f(X)\|$.

For the second argument, we have

$$
\begin{aligned}
&\|\mathrm{grad} f(X) - \mathrm{grad} f(Y)\| \\
\leq &\|\mathcal{P}_{T_X\mathrm{St}(d,r)}(\nabla f(X)) - \mathcal{P}_{T_X\mathrm{St}(d,r)}(\nabla f(Y))\| + \|\mathcal{P}_{T_X\mathrm{St}(d,r)}(\nabla f(Y)) - \mathrm{grad} f(Y)\| \\
\leq &L_f\|X-Y\| + \frac{1}{2}\|X(X^\top\nabla f(Y) + \nabla f(Y)^\top X) - Y(Y^\top\nabla f(Y) + \nabla f(Y)^\top Y)\| \\
\leq &L_f\|X-Y\| + \frac{1}{2}\|X((X-Y)^\top\nabla f(Y) + \nabla f(Y)^\top (X-Y))\| \\
&+ \frac{1}{2}\|(X-Y)(Y^\top\nabla f(Y) + \nabla f(Y)^\top Y)\| \\
\leq &L_f\|X-Y\| + \frac{1}{2}(2\hat{D}_f + 3\hat{D}_f)\|X-Y\| \\
= &(L_f + \frac{5}{2}\hat{D}_f)\|X-Y\|,
\end{aligned}
$$

where $\hat{D}_f := \max_{X\in\bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8})} \|\nabla f(X)\|$, the second inequality is due to the contractive property of $\mathcal{P}_{T_X\mathrm{St}(d,r)}$, and the last inequality is from the fact that $\|Y\|_2 \leq \frac{3}{2}$. By setting $\hat{L} = L_f + \frac{5}{2}\hat{D}_f$, we complete the proof. □

Proof of Lemma 4

Proof. It follows that

$$
\begin{aligned}
\|X_{k+1} - \bar{X}_{k+1}\| &\leq \|X_{k+1} - \bar{X}_k\| \\
&\leq \|X_k - \mu\varphi(X_k) - \bar{X}_k\| + \alpha\|\mathrm{grad} f(X_k)\| \\
&\leq \frac{2}{3}\|X_k - \bar{X}_k\| + \alpha\|\mathrm{grad} f(X_k)\|.
\end{aligned}
$$

We complete the proof. □

Proof of Lemma 5

Proof. It follows from (7) that

$$
\begin{aligned}
&f(\bar{X}_{k+1}) - f(\bar{X}_k) \le \left\langle \mathrm{grad} f(\bar{X}_k), \bar{X}_{k+1} - \bar{X}_k \right\rangle + \frac{L}{2}\|\bar{X}_{k+1} - \bar{X}_k\|^2 \\
\le& \left\langle \mathrm{grad} f(\bar{X}_k), \bar{X}_{k+1} - X_{k+1} + X_k - \bar{X}_k \right\rangle + \left\langle \mathrm{grad} f(\bar{X}_k), X_{k+1} - X_k \right\rangle \\
&+ 2L\|X_{k+1} - X_k\|^2 \\
\le& \left\langle \mathrm{grad} f(\bar{X}_k), \bar{X}_{k+1} - X_{k+1} \right\rangle + \left\langle \mathrm{grad} f(\bar{X}_k), X_{k+1} - X_k \right\rangle \\
&+ 4L(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
=& \left\langle \mathrm{grad} f(\bar{X}_k) - \mathrm{grad} f(\bar{X}_{k+1}), \bar{X}_{k+1} - X_{k+1} \right\rangle + \langle \mathrm{grad} f(X_k), X_{k+1} - X_k \rangle \\
&+ \left\langle \mathrm{grad} f(\bar{X}_k) - \mathrm{grad} f(X_k), X_{k+1} - X_k \right\rangle \\
&+ 4L(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
\le& 2\hat{L}^2\|X_{k+1} - X_k\|^2 + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 - \alpha\|\mathrm{grad} f(X_k)\|^2 \\
&- \mu \left\langle \mathrm{grad} f(X_k), \nabla\varphi(X_k) \right\rangle + \frac{1}{2}(\hat{L}^2\|X_k - \bar{X}_k\|^2 + \|X_{k+1} - X_k\|^2) \\
&+ 4L(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
\le& -\alpha\|\mathrm{grad} f(X_k)\|^2 - \mu\left\langle \nabla f(X_k), \mathcal{P}_{T_{X_k}\mathrm{St}(d,r)}(\nabla\varphi(X_k)) \right\rangle + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 \\
&+ \frac{1}{2}\|X_k - \bar{X}_k\|^2 + (4\hat{L}^2 + 4L + 1)(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
\le& -(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2)\|\mathrm{grad} f(X_k)\|^2 + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 \\
&+ (6\mu\hat{D}_f + \frac{1}{2} + 16(4\hat{L}^2 + 4L + 1)\mu^2)\|X_k - \bar{X}_k\|^2,
\end{aligned} \tag{17}
$$

where the second inequality is from the 2-Lipschitz continuity of $\mathcal{P}_{\mathrm{St}(d,r)}$ over $\bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8})$, the third inequality is due to the facts that $X_k - \bar{X}_k \in N_{\bar{X}_k}\mathrm{St}(d,r)$ and $\langle A, B\rangle \le \frac{1}{2}(\|A\|^2 + \|B\|^2)$ for any $A, B \in \mathbb{R}^{n\times d}$, and the last inequality comes from

$$
\|\mathcal{P}_{T_{X_k}\mathrm{St}(d,r)}(\nabla\varphi(X_k))\| = \|X_k(X_k^\top X_k - I)^2\| \le 6\|X_k - \bar{X}_k\|^2.
$$

Plugging $\mu = \frac{1}{3}$ into (17) gives (10). □

Proof of Theorem.

Proof. Applying [42, Lemma 2] to (9) yields

$$
\sum_{k=0}^{K}\|X_k - \bar{X}_k\|^2 \le 18\alpha^2\sum_{k=0}^{K}\|\mathrm{grad} f(\bar{X}_k)\|^2 + 4. \tag{18}
$$

Then, summing (10) over $k = 0, \ldots, K$ gives

$$
\begin{aligned}
&f(\bar{X}_{k+1}) - f(\bar{X}_0) \\
\le& -(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2)\sum_{k=0}^{K}\|\mathrm{grad} f(X_k)\|^2 \\
&+ \frac{1}{2}\left(4\hat{D}_f + 16\hat{L}^2 + 16L + 3\right)\sum_{k=0}^{K+1}\|X_k - \bar{X}_k\|^2 \\
\le& -(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2 + 9(4\hat{D}_f + 16\hat{L}^2 + 16L + 3)\alpha^2)\sum_{k=0}^{K}\|\mathrm{grad} f(X_k)\|^2 \\
&+ \frac{1}{2}\left(4\hat{D}_f + 16\hat{L}^2 + 16L + 3\right)(18\alpha^2\|\mathrm{grad} f(X_{k+1})\|^2 + 4).
\end{aligned} \tag{19}
$$

Define $c_1 = 148\hat{L}^2 + 148L + 36\hat{D}_f + 28$ and $c_2 = (9\hat{D}_f^2 + 2)(4\hat{D}_f + 16\hat{L}^2 + 16L + 4)$. Then, we have

$$\alpha(1 - c_1\alpha)\sum_{k=0}^{K} \|\mathrm{grad} f(X_k)\|^2 \leq f(\bar{X}_0) - f(\bar{X}_{k+1}) + c_2.$$

Therefore, for any $\alpha \leq \frac{1}{2c_1}$, taking $K \to \infty$ gives $\sum_{k=0}^{\infty} \|\mathrm{grad} f(X_k)\|^2 < \infty$. Then by (11), $\sum_{k=0}^{\infty} \|X_k - \bar{X}_k\|^2 < \infty$. These lead to (11). □

C Hyperparameters

Table 4: Hyperparameter setup of Manifold-LoRA for question answering tasks.

Method	Hyperparamter	SQuADv1.1	SQuADv2.0
	Warmup Ratio	0.06	
	LR Schedule	Linear	
	Weight Decay	0.1	
	β_1	0.9	
	β_2	0.999	
	Batch Size	64	
	Learning Rate	3e-3	
	Epochs	4	
Sphere(r=8)	μ	0.85	0.85
	Lower	0.25	0.25
	Upper	0.75	0.5
Sphere(r=16)	μ	0.9	0.85
	Lower	0.25	0.25
	Upper	0.5	0.5
Stiefel(r=8)	μ	0.85	0.85
	Lower	0.25	0.25
	Upper	0.5	0.5
Stiefel(r=16)	μ	0.9	0.85
	Lower	0.25	0.25
	Upper	0.5	0.5

Table 5: Hyperparameter configurations of Manifold-LoRA for GLUE benchmark

Method	Hyperparameter	MNLI	SST-2	CoLA	QQP	QNLI	RTE	MRPC	STS-B
	Warmup Ratio	0.06							
	LR Schedule	Linear							
	Max Sequence Length	256							
	Weight Decay	0.1							
	β_1	0.9							
	β_2	0.999							
	Batch Size	32							
	LoRA Layer	W_q, W_v							
	Epochs	7	24	25	5	5	50	30	25
	Learning rate	5e-4	8e-4	5e-4	5e-4	1.2e-3	1.2e-3	1e-3	2.2e-3
Sphere(r=16)	μ	1	0.9	0.8	0.9	0.95	1.2	0.85	0.9
	Lower	0.25	0.25	0.5	0.5	0.5	0.5	1	1
	Upper	2	2	2	4	2	2	4	4
Sphere(r=8)	μ	0.95	0.95	1	0.9	1	0.9	0.85	1
	Lower	2	0.5	1	0.5	0.5	0.25	2	1
	Upper	8	2	8	2	2	0.5	4	8
Stiefel(r=16)	μ	0.8	0.85	0.95	0.9	0.95	1.2	0.8	1
	Lower	2	0.5	2	0.5	0.5	0.5	1	1
	Upper	8	1	8	4	1	2	4	16
Stiefel(r=8)	μ	0.8	0.95	0.95	0.9	0.85	0.9	1	1
	Lower	2	0.5	2	0.5	0.5	0.25	1	1
	Upper	8	2	8	2	2	1	4	16

Table 6: Hyperparameter setup of Manifold-LoRA for E2E benchmark.

Method	Hyperparamter	GPT-2(M)	GPT-2(L)
	Warmup Steps	500	
	LR Schedule	Linear	
	Weight Decay	0.01	
	β_1	0.9	
	β_2	0.999	
	LoRA dropout	0	
	Batch Size	8	
	Learning Rate	2e-4	
	Epochs	5	
Sphere(r=4)	μ	1	0.9
	Lower	0.5	0.5
	Upper	2	2
Stiefel(r=4)	μ	1	1.1
	Lower	0.5	0.5
	Upper	4	2

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Our empirical results in Section 4 justify ours claims.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss our limitations in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

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

   Answer: [Yes]

Justification: We provide complete proofs in Appendix B and full set of assumptions in Section 2

Guidelines:

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

4. **Experimental Result Reproducibility**

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

Answer: [Yes]

Justification: We specify the training details in Section 4 and Appendix C

Guidelines:

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

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

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

Answer: [Yes]

Justification: We specify the code and dataset in Section 4.

Guidelines:

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

6. **Experimental Setting/Details**

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

Answer: [Yes]

Justification: We specify training details in Section 4 and hyperparameters in Appendix C.

Guidelines:

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

7. **Experiment Statistical Significance**

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

Answer: [Yes]

Justification: We specify these in our Section 4.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

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

   Answer: [Yes]

   Justification: We use the Hugging face and opendelta as our base code and make some modifications. We use GLUE, E2E, and Suqad three dataset.

   Guidelines:

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

9. **Code Of Ethics**

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

   Answer: [Yes]

   Justification: Our research is compatible with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader Impacts**

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

    Answer: [NA]

    Justification: There is no societal impact of our work performed

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our work poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We correctly cite the code and datasets we used in Section 4.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

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

    Answer: [NA]

    Justification: Our paper does not release new asset.

    Guidelines:

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

14. **Crowdsourcing and Research with Human Subjects**

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

    Answer: [NA]

    Justification: Our study does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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