PACE: marrying generalization in PArameter-efficient fine-tuning with Consistency rEgularization

Yao Ni[†] **Shan Zhang**[‡,†] **Piotr Koniusz**[*,§,†]
[†]The Australian National University [§]Data61♥CSIRO
[‡]Australian Institute for Machine Learning, University of Adelaide
[†]yao.ni@anu.edu.au [‡]shan.zhang@adelaide.edu.au [§]piotr.koniusz@data61.csiro.au

Abstract

Parameter-Efficient Fine-Tuning (PEFT) effectively adapts pre-trained transformers to downstream tasks. However, the optimization for tasks performance often comes at the cost of generalizability in fine-tuned models. To address this issue, we theoretically connect smaller weight gradient norms during training and larger datasets to the improved model generalization. Motivated by this connection, we propose reducing gradient norms for enhanced generalization and aligning fine-tuned model with the pre-trained counterpart to retain knowledge from large-scale pre-training data. Yet, naive alignment does not guarantee gradient reduction and can potentially cause gradient explosion, complicating efforts to manage gradients. To address such issues, we propose PACE, marrying generalization of PArameter-efficient fine-tuning with Consistency rEgularization. We perturb features learned from the adapter with the multiplicative noise and ensure the fine-tuned model remains consistent for same sample under different perturbations. Theoretical analysis shows that PACE not only implicitly regularizes gradients for enhanced generalization, but also implicitly aligns the fine-tuned and pre-trained models to retain knowledge. Experimental evidence supports our theories. PACE surpasses existing PEFT methods in visual adaptation tasks (VTAB-1k, FGVC, few-shot learning, domain adaptation) and showing potential for resource-efficient fine-tuning. It also improves LoRA in text classification (GLUE) and mathematical reasoning (GSM-8K). Code will be available at MaxwellYaoNi/PACE.

1 Introduction

Transformers [67], with the self-attention mechanism [3] capturing long-range dependencies in data, have been successful in various deep learning tasks, including image classification (ViT [15]), multimodal learning (CLIP [54]), image synthesis (StableDiffusion [56]), semantic segmentation (SAM [32]) and text generation (LLaMA [64]). The success of transformers can be largely attributed to the availability of abundant data, such as ImageNet [11] and Laion5B [59], which has enabled researchers to scale up these models by training them with an enormous number of parameters.

Such huge models, with knowledge from large-scale pre-training [62], have become foundation models that can be easily adapted to various downstream tasks through full fine-tuning or linear probing [19], eliminating the need for task-specific model design [8]. However, full fine-tuning is storage-intensive and infeasible for maintaining separate model weights as the number of tasks grows, while linear probing, which only trains the last head layer, yields inferior adaptation performance.

To overcome these limitations, Parameter-Efficient Fine-Tuning (PEFT) [23] fine-tunes only a small subset of parameters, thereby reducing storage requirements while surpassing the performance of

[*]The corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

full fine-tuning and linear probing. These advantages have popularized PEFT and inspired the development of various PEFT methods for deep learning tasks, which can be categorized into two groups: those increasing inference cost and cost-efficient ones. The first group introduces additional learning branches, such as non-linear adapters [24, 8], or concatenates learnable parameters with input tokens, *e.g.*, visual prompts [27, 81, 51], increasing inference cost. The second group, focuses on cost-efficiency involving lower-rank adaptation in linear layers [7, 25], or affine transformations such as SSF [40] and RepAdapters [44], which can be reparameterized during inference for efficiency.

Despite the superiority and efficiency of PEFT, prioritizing optimization for downstream tasks compromises the generalizability of fine-tuned models, yielding suboptimal performance. Although some analyses have been conducted on PEFT [62, 26, 17, 71, 38], they fail to fully explain the generalization of PEFT, leading to ineffective strategies for improving generalization.

To address this gap in understanding generalization in PEFT, we establish a theoretical connection from generalization theory: smaller weight gradient norms and larger data volumes contribute to better generalization. Motivated by this, we propose reducing weight gradient norms and aligning output space of the fine-tuned model with the pre-trained one to retain knowledge captured from large pre-training data. Yet, theoretical analyses reveal this naive alignment dose not guarantee gradient regularization and can even cause gradient explosion, complicating efforts for gradient management. To address this issue, we propose perturbing features learned from the adapter with multiplicative noise and constraining the network output to be consistent across different perturbations.

We call our method PACE. It marries generalization of PArameter-efficient fine-tuning with Consistency rEgularization. The name reflects our goal of keeping the output behavior of the fine-tuned model *in pace with* the pre-trained one. Despite its simplicity, theoretical analysis confirms that PACE not only implicitly regularizes weight gradients for better generalization but also implicitly aligns the fine-tuned model with the pre-trained counterpart to retain knowledge from large-scale pre-training data. Experimental evidence supports our theories. PACE improves existing PEFT methods, achieving superior results across six adaptation benchmarks. Our key contributions are:

i. We establish a theory connecting smaller weight gradient norms and larger datasets with enhanced generalization, motivating gradient reduction and model alignment for fine-tuning.
ii. We propose PACE, a simple yet effective method perturbing features from adapters with multiplicative noise and constraining output of fine-tuned model to be consistent across perturbations.
iii. Our theoretical and empirical evidence confirms that PACE implicitly regularizes gradients and aligns the fine-tuned model with the pre-trained one. PACE excels on 4 visual adaptation tasks.
iv. We provide novel theoretical explanations for how gradient penalization and consistency regularization benefit generalization, offering fundamental insights applicable across deep learning.

2 Related work

Parameter-Efficient Fine-Tuning (PEFT). LoRA [25] uses low-rank decomposition to reduce parameters and treats adapters as side paths. SSF [40] proposes affine transformations on latent features. FacT [29] decomposes and reassembles parameter matrices in ViT. Surgical fine-tuning [35] different network parts results in different performance for different datasets. FLoRA [73] aims at real-time global service. GLoRA [7] unifies cost-efficient PEFT methods. NOAH [81] uses parameter search on neural prompts. ARC [13] leverages cross-layer ViT similarity, parameter-sharing adapter and scaling factors for lower fine-tuning cost. RLRR [14] incorporates a residual term for flexibility while preserving pre-trained representation. RepAdapter [44] reparameterizes adapters for efficient inference. Res-tuning [28] unbinds tuners from the backbone for memory efficiency. Zhao *et al*. [82] show impressive fine-tuning results by tuning only the attention layer normalization. OFT [53] and BOFT [41] propose orthogonal fine-tuning to preserve hypersphere energy between neurons.

Consistency Regularization. Fixmatch [60] applies consistency regularization over augmented images for semi-supervised learning. Openmatch [58] utilizes it on outlier predictions for open-set semi-supervised learning. R-Drop [75] applies it to transformers [67] with dropout for NLP tasks. CR [78] applies it over augmented real and fake images for GAN training. CAGAN [49] enforces consistency on discriminators with dropout for GAN training. Despite the empirical success of consistency regularization demonstrated by previous works, theoretical analysis is lacking. While NICE [47] demonstrates that consistency regularization lowers latent feature gradients for stable GAN training, it fails to reveal reduced weight gradient for enhanced generalization. Our study goes

beyond prior works by providing a theoretical link between smaller weight gradients and improved generalization, effectively marrying generalization of PEFT with consistency regularization.

Generalization of Fine-Tuning. Li *et al*. [37] constrain the fine-tuned model's closeness to the pre-trained model in weight space. Fu *et al*. [17] induce sparsity on PEFT methods for enhanced generalization. Wang *et al*. [71] finds PEFT methods improve generalization on fine-tuning graph neural network. Recent works, including VioLET [72], PromptSRC [30], CoPrompt [57], propose aligning the fine-tuned model with the pre-trained one for enhanced generalization or avoiding forgetting, which can be seen as our naive alignment. Additionally, L2SP [76], DELTA [39], and FTP [63] aim to retain pre-trained knowledge by aligning finetuned models with pre-trained ones, reducing distance in weight space, feature space and using projected gradient descent, respectively. However, they fail to provide a theoretical analysis for this alignment. Our study goes beyond understanding generalization of PEFT by discovering the benefits of gradient regularization and model alignment. We propose PACE to match both requirements, paving a comprehensive understanding for PEFT.

Gradient regularization. Previous studies have empirically shown that gradient regularization improves neural network performance [66, 83, 46, 48]. However, they failed to theoretically establish the connection between smaller gradient norms and better generalization [16, 80, 6]. Our work bridges this gap by establishing a fundamental theory between reduced gradient norms and improved generalization, providing a solid foundation for future research on enhancing generalization.

3 Approach

We begin with a unified perspective on cost-efficient PEFT based on GLoRA [7], linking generalization with gradients and large-scale data and motivating the alignment of the fine-tuned model with the pre-trained model to leverage its knowledge. We identify limitations of naive alignment in gradient regularization and introduce PACE, which implicitly enhances gradient regularization and model alignment. We conclude with theoretical justification and efficient implementations.

3.1 A unified perspective on cost-efficient PEFT methods

The transformer architectures [67, 15] have excelled in natural language processing and computer vision tasks through their powerful sequential modeling capabilities. This success stems from their ability to process text/image tokens through L transformer blocks, where each block contains self-attention and MLP modules primarily composed of linear layers. These linear layers enable the self-attention mechanism to capture long-range dependencies, allowing transformers to achieve superior performance when scaled to huge parameters and trained on extensive datasets.

With massive parameters pretrained on large-scale data, transformers serve as foundation models that can be fine-tuned for downstream tasks using limited data. However, fully fine-tuning all parameters for various downstream tasks requires substantial memory and can lead the forgetting of pretrained knowledge. To alleviate this without increasing inference cost, adapters with lightweight parameters are often preferred for fine-tuning. Let $\bar{h}_0(\cdot)$ be a transformation within the pre-trained transformer. Current adapters can be unified as introducing a residual branch $\Delta\bar{h}$ to form a new transformation $\bar{h}$:

$$\bar{h}(\boldsymbol{a}) = \bar{h}_0(\boldsymbol{a}) + \Delta\bar{h}(\boldsymbol{a}). \tag{1}$$

Here, $\boldsymbol{a}$ is the input and $\bar{h}_0$ can represent MLP modules, as in Adapter [24] and AdaptFormer [8], or linear layers in self-attention and MLP modules, as in [25, 7, 12, 33]. In SSF [40], $\bar{h}_0$ is the identity mapping and $\Delta\bar{h}(a) = \boldsymbol{a} \odot (\boldsymbol{\gamma} - \mathbf{1}) + \boldsymbol{\beta}$ with $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ as affine transformation parameters.

Given that linear layers are key components in transformer, tuning them offers a flexible and effective way to adapt models to downstream tasks. This work focuses on methods that tune the linear layer without increasing inference cost. Let $(\boldsymbol{W}_0, \boldsymbol{b}_0)$, $(\Delta\boldsymbol{W}, \Delta\boldsymbol{b})$, and $(\boldsymbol{W}, \boldsymbol{b})$ be the parameters of pretrained model, adapter and finetuned model, respectively, where $\boldsymbol{W}_0, \Delta\boldsymbol{W}, \boldsymbol{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ and $\boldsymbol{b}_0, \Delta\boldsymbol{b}, \boldsymbol{b} \in \mathbb{R}^{d_{\text{out}}}$, finetuning a linear layer in self-attention or MLP module can be formed as:

$$\begin{aligned} h(\boldsymbol{a}) &= \boldsymbol{W}\boldsymbol{a} + \boldsymbol{b} = (\boldsymbol{W}_0 + \Delta\boldsymbol{W})\boldsymbol{a} + (\boldsymbol{b}_0 + \Delta\boldsymbol{b}) \\ &= h_0(\boldsymbol{a}) + \Delta h(\boldsymbol{a}) = (\boldsymbol{W}_0\boldsymbol{a} + \boldsymbol{b}_0) + (\Delta\boldsymbol{W}\boldsymbol{a} + \Delta\boldsymbol{b}). \end{aligned} \tag{2}$$

Based on GLoRA [7], cost-efficient PEFT methods for linear layers vary in the form of $\Delta\boldsymbol{W}, \Delta\boldsymbol{b}$:

LoRA$_{\text{add}}$: $\Delta\boldsymbol{W} = \boldsymbol{W}_{\text{d}}\boldsymbol{W}_{\text{u}}, \Delta\boldsymbol{b} = \boldsymbol{b}_{\text{lora}}$ where $\boldsymbol{W}_{\text{d}} \in \mathbb{R}^{d_{\text{out}} \times r}$, $\boldsymbol{W}_{\text{u}} \in \mathbb{R}^{r \times d_{\text{in}}}$, and r is the rank.

LoRA$_{\text{mul}}$: $\Delta \boldsymbol{W} = \boldsymbol{W}_0 \odot (\boldsymbol{W}_{\text{d}} \boldsymbol{W}_{\text{u}})$, $\Delta \boldsymbol{b} = \boldsymbol{b}_0 \odot \boldsymbol{b}_{\text{lora}}$, including RepAdapter [44] via reparameterization.

VPT$_{\text{add}}$: $\Delta \boldsymbol{W}$ is zero, $\Delta \boldsymbol{b} = \boldsymbol{W}_0 \boldsymbol{P}$, with learnable $\boldsymbol{P} \in \mathbb{R}^{d_{\text{in}} \times 1}$ as layer-wise visual prompt. We use VPT$_{\text{add}}$ to differentiate from VPT [27], which concatenates $\boldsymbol{P}$ with tokens, increasing inference cost.

3.2 Generalization of deep neural networks

Having established a unified perspective on cost-efficient PEFT, we now motivate our method from a perspective on improving generalization of neural networks to enhance performance on unseen data. Consider a network $f := \phi(g(x))$ with l layers, where g is feature extractor and ϕ is the classification head. Let $\boldsymbol{\theta} := \{(\boldsymbol{W}^{(i)}, \boldsymbol{b}^{(i)})\}_{i=1}^{l}$ be the parameter set with dimension d and $\mathcal{D}^n := \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$ be the training set of size n drawn i.i.d. from distribution $\mathscr{D}$, which contains infinite data. The following lemma from [16] builds a relationship between the empirical and population loss.

Lemma 1 *(Theorem 1 from [16]) Let $\mathcal{L}_{\mathcal{D}^n}(\boldsymbol{\theta})$ be the empirical loss function over f on training set $\mathcal{D}^n$ and $\mathcal{L}_{\mathscr{D}}(\boldsymbol{\theta})$ be the population loss. For any $\rho > 0$, with high probability over $\mathcal{D}^n \sim \mathscr{D}$, we have*

$$\mathcal{L}_{\mathscr{D}}(\boldsymbol{\theta}) \leq \max_{\|\boldsymbol{\epsilon}\|_2 \leq \rho} \mathcal{L}_{\mathcal{D}^n}(\boldsymbol{\theta} + \boldsymbol{\epsilon}) + R\Big(\frac{\|\boldsymbol{\theta}\|_2^2}{\rho^2}, \frac{1}{n}\Big), \tag{3}$$

where $R : (\mathbb{R}_+, \mathbb{R}_+) \to \mathbb{R}_+$ is a increasing function (under some conditions on $\mathcal{L}_{\mathscr{D}}(\boldsymbol{\theta})$ and n).

Lemma 1 bounds the population loss by the empirical loss with perturbed weights, indicating that minimal empirical loss increase from small weight perturbations implies low population loss.

By observing that the maximum of $\mathcal{L}_{\mathcal{D}^n}$ is achieved at $\boldsymbol{\epsilon} = \frac{\rho \boldsymbol{\nabla}_{\boldsymbol{\theta}}}{\|\boldsymbol{\nabla}_{\boldsymbol{\theta}}\|_2}$, where $\boldsymbol{\nabla}_{\boldsymbol{\theta}}$ is the gradient of $\mathcal{L}_{\mathcal{D}^n}$ at $\boldsymbol{\theta}$, and performing a Taylor expansion of $\mathcal{L}_{\mathcal{D}^n}$ around $\boldsymbol{\theta}$, we formulate the following theorem:

Theorem 1 *Denote $\boldsymbol{\nabla}_{\boldsymbol{\theta}}$ as the gradient and $\lambda_{max}^{\boldsymbol{H}}$ as the largest eigenvalues of the Hessian matrix $\boldsymbol{H}_{\boldsymbol{\theta}}$ of $\mathcal{L}_{\mathcal{D}^n}$ at $\boldsymbol{\theta}$. For any $\rho > 0$, with high probability over training set $\mathcal{D}^n \sim \mathscr{D}$, we have*

$$\mathcal{L}_{\mathscr{D}}(\boldsymbol{\theta}) \leq \mathcal{L}_{\mathcal{D}^n}(\boldsymbol{\theta}) + \rho \|\boldsymbol{\nabla}_{\boldsymbol{\theta}}\|_2 + \frac{\rho^2}{2} \lambda_{max}^{\boldsymbol{H}} + R\Big(\frac{\|\boldsymbol{\theta}\|_2^2}{\rho^2}, \frac{1}{n}\Big). \tag{4}$$

Here, higher-order terms from the Taylor expansion are incorporated into $R\Big(\frac{\|\boldsymbol{\theta}\|_2^2}{\rho^2}, \frac{1}{n}\Big)$, which is related to weights norm and inversely related to the training data size n.

Theorem 1 (proof in §B.1) outlines strategies for enhancing generalization. These involve regularizing weight norms and the largest Hessian eigenvalues, and crucially, increasing data size n and reducing the weight gradient norms (illustrated in Figure 1). However, excessive reduction should be avoided as it could impair network's representation capacity, yielding higher empirical and population loss.

3.3 Motivation and limitation of aligning the fine-tuned model with the pre-trained model

Theorem 1 emphasizes that large-scale data and smaller gradient magnitudes are essential for better generalization in neural network training. Therefore, aligning the fine-tuned model with the pre-trained one is crucial, as it ensures retention of knowledge developed from large-scale data, preserving generalizability. PEFT methods achieve this alignment by limiting the number of trainable parameters, restricting model's capacity to deviate from the pre-trained one and often outperforming full fine-tuning. However, the training objective prioritizes downstream task performance, compromising alignment with pre-trained knowledge. While sparsity regularization [17] and weight decay on adapter weights help, they do not ensure alignment, as even small weight changes can lead to significant divergence in output space [74, 20, 16]. Therefore, we propose to achieve the alignment by reducing the FP-distance (output distance between fine-tuned and pre-trained models on training samples):

$$D^{\text{fp}}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \|f(\boldsymbol{x}_i; \boldsymbol{\theta}) - f(\boldsymbol{x}_i; \boldsymbol{\theta}_0)\|_2^2, \quad \boldsymbol{\theta} = \boldsymbol{\theta}_0 + \Delta \boldsymbol{\theta}, \tag{5}$$

where $\boldsymbol{\theta}, \boldsymbol{\theta}_0, \Delta \boldsymbol{\theta} \in \mathbb{R}^d$ are parameters for the fine-tuned model, pre-trained model and the adapter.

While reducing FP-distance keeps the fine-tuned model close to the pre-trained model, thus preserving its knowledge, it does not ensure reduced gradient magnitudes, leading to suboptimal generalization. To understand the gradient-related limitations in this alignment, we assume $\Delta \boldsymbol{\theta}$ is small enough for a

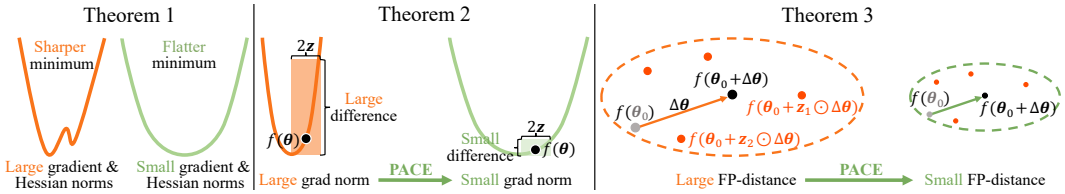


Figure 1: Thm. 1: A flatter minimum has smaller gradient and Hessian norms, yielding better generalization. Thm. 2: Large gradient norms indicate large differences among perturbations. PACE minimizes these differences, reducing gradient norms. Thm. 3: Minimizing all pairs of distances between $f(\boldsymbol{\theta}_0+\boldsymbol{z}_1\odot\Delta\boldsymbol{\theta})$ and $f(\boldsymbol{\theta}_0+\boldsymbol{z}_2\odot\Delta\boldsymbol{\theta})$ where $\boldsymbol{z}_1,\boldsymbol{z}_2\sim\mathcal{N}(\mathbf{1},\sigma^2\boldsymbol{I})$ also reduces FP-distance (between fine-tuned $f(\boldsymbol{\theta}_0+\Delta\boldsymbol{\theta})$ and pre-trained $f(\boldsymbol{\theta}_0)$), especially when $\boldsymbol{z}_1=\mathbf{1}$, $\boldsymbol{z}_2=\mathbf{0}$ or vice versa.

Taylor expansion approximation. Following standard practices [16, 79, 2], we perform the expansion up to the second-order terms. Given the independence between elements in squared L_2 distances (§B.4) and to simplify our theories, we analyze a one-dimensional output for a single i.i.d. sample, which leads us to the following proposition.

Proposition 1 *Assuming $\Delta\boldsymbol{\theta}$ is small, denote $f(\boldsymbol{\theta})\in\mathbb{R}$ as the one-dimensional output for $\boldsymbol{x}$, with $\boldsymbol{\nabla}$ and $\boldsymbol{H}$ as its gradient and Hessian at $\boldsymbol{\theta}$. FP-distance over $\boldsymbol{x}$ can be decomposed as follows:*

$$[f(\boldsymbol{\theta})-f(\boldsymbol{\theta}_0)]^2=[f(\boldsymbol{\theta})-f(\boldsymbol{\theta}-\Delta\boldsymbol{\theta})]^2\approx\left[f(\boldsymbol{\theta})-[f(\boldsymbol{\theta})-\Delta\boldsymbol{\theta}^T\boldsymbol{\nabla}+\frac{1}{2}\Delta\boldsymbol{\theta}^T\boldsymbol{H}\Delta\boldsymbol{\theta}]\right]^2$$
$$\approx[\Delta\boldsymbol{\theta}^T\boldsymbol{\nabla}-\frac{1}{2}\Delta\boldsymbol{\theta}^T\boldsymbol{H}\Delta\boldsymbol{\theta}]^2. \tag{6}$$

Prop. 1 establishes the relationship between weight gradients, adapter weights, and FP-distance. However, it remains unclear if it regulates gradients. Our experiments show that minimizing FP-distance can sometimes increase gradient magnitude, complicating efforts for managing gradient.

3.4 Consistency regularization

To achieve better generalization by both regularizing gradients and aligning the fine-tuned model with the pre-trined model, we propose a consistency regularization loss for f, encouraging invariance of f to the same input under varying multiplicative noise perturbations on the adapter weights, as follows:

$$D^{\text{pace}}(\boldsymbol{\theta})=\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\boldsymbol{z}_1,\boldsymbol{z}_2}\|f(\boldsymbol{x}_i;\boldsymbol{\theta}_0+\boldsymbol{z}_1\odot\Delta\boldsymbol{\theta})-f(\boldsymbol{x}_i;\boldsymbol{\theta}_0+\boldsymbol{z}_2\odot\Delta\boldsymbol{\theta})\|_2^2, \tag{7}$$

where $\boldsymbol{z}_1,\boldsymbol{z}_2\sim\mathcal{N}(\mathbf{1},\sigma^2\boldsymbol{I})$ is the multiplicative noise applied on adapter weight. To understand the generalization benefits in this consistency regularization, we simplify the analysis by focusing on one-dimensional output for a single sample, resulting in the following theorem.

Theorem 2 *Using notations from Prop. 1, let $f(\boldsymbol{\theta}_0+\boldsymbol{z}\odot\Delta\boldsymbol{\theta})\in\mathbb{R}$ be the one-dimensional output for $\boldsymbol{x}$. Define $\Delta\theta_j$ as j-th element in $\Delta\boldsymbol{\theta}$, ∇_j as the j-th element in $\boldsymbol{\nabla}$ and H_{jk} as the (j,k)-entry in $\boldsymbol{H}$. With $\boldsymbol{z}_1,\boldsymbol{z}_2\sim\mathcal{N}(\mathbf{1},\sigma^2\boldsymbol{I})$, the consistency loss over $\boldsymbol{x}$ can be approximated as:*

$$\mathbb{E}_{\boldsymbol{z}_1,\boldsymbol{z}_2}[f(\boldsymbol{\theta}_0+\boldsymbol{z}_1\odot\Delta\boldsymbol{\theta})-f(\boldsymbol{\theta}_0+\boldsymbol{z}_2\odot\Delta\boldsymbol{\theta})]^2$$
$$\approx 2\sigma^2\textstyle\sum_j\Delta\theta_j^2\nabla_j^2+\sigma^4\sum_{j,k}\Delta\theta_k^2\Delta\theta_j^2H_{jk}^2\ =2\sigma^2\|\Delta\boldsymbol{\theta}\odot\boldsymbol{\nabla}\|_2^2+\sigma^4\|(\Delta\boldsymbol{\theta}\Delta\boldsymbol{\theta}^T)\odot\boldsymbol{H}\|_F^2. \tag{8}$$

Theorem 2 (Proof in §B.2) shows that consistency regularization essentially penalizes the first- and second-order gradients of f at $\boldsymbol{\theta}$ (illustrated in Figure 1), with the regularization strength controlled by the noise variance σ^2 and adaptively influenced by the magnitude of elements in adapter weight $\Delta\boldsymbol{\theta}$. Thus, minimizing the consistency loss implicitly regularizes the gradients, improving generalization.

With the FP-distance in Prop. 1 and consistency loss in Theorem 2, we establish their relationship as:

Theorem 3 *With d as the dimension of $\boldsymbol{\theta}$, Eq. 6 can be upper bounded as:*

$$[\Delta\boldsymbol{\theta}^T\boldsymbol{\nabla}-\frac{1}{2}\Delta\boldsymbol{\theta}^T\boldsymbol{H}\Delta\boldsymbol{\theta}]^2\le 2d\|\Delta\boldsymbol{\theta}\odot\boldsymbol{\nabla}\|_2^2+d^2\|(\Delta\boldsymbol{\theta}\Delta\boldsymbol{\theta}^T)\odot\boldsymbol{H}\|_F^2. \tag{9}$$

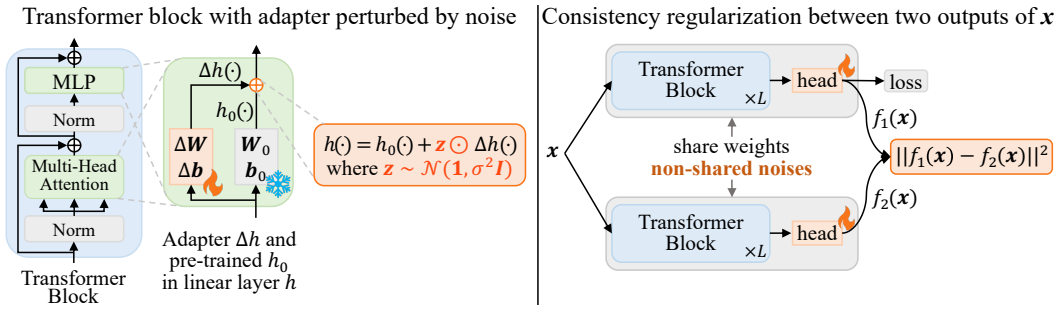


Figure 2: Our pipeline. Adapter Δh and h_0 from pre-trained model form the linear layer h of Multi-Head Attention and MLP in fine-tuned model. We perturb Δh with multiplicative noise and ensure the network remains consistent to same inputs under varying perturbations.

Theorem 3 (proof in B.3) establishes the relationship between Eq. 6 and Eq. 8, showing Eq. 6 is upper-bounded by terms involving $\|\Delta\boldsymbol{\theta} \odot \boldsymbol{\nabla}\|_2^2$ and $\|(\Delta\boldsymbol{\theta}\Delta\boldsymbol{\theta}^T) \odot \boldsymbol{H}\|_F^2$ which appear in Eq. 8. Reducing these terms results in a decrease in Eq. 6. Thus minimizing the consistency loss implicitly aligns the fine-tuned with pre-trained models (illustrated in Figure 1), preserving pre-trained knowledge.

3.5 Efficient implementation of PACE

Providing different weight perturbations for each input in a mini-batch increases memory and computational demands. To avoid this inefficiency, we perturb feature outputs from the adapter Δh, effectively simulating perturbation that shares noise across each row in the weight $\Delta\boldsymbol{W}$. Our simple pipeline is illustrated in Figure 2. Consider $\boldsymbol{X} \in \mathbb{R}^{B\times T\times d_{\text{in}}}$ as a batch of data where B, T be the batch and token sizes. The calculation for the linear layer of the fine-tuned model, which utilizes pre-trained weights $\boldsymbol{W}_0, \boldsymbol{b}_0$ and adapter weights $\Delta\boldsymbol{W}, \Delta\boldsymbol{b}$, processes an output size of d_{out} as:

$$h_0(\boldsymbol{X}) = \boldsymbol{W}_0\boldsymbol{X} + \boldsymbol{b}_0; \quad \Delta h(\boldsymbol{X}) = \Delta\boldsymbol{W}\boldsymbol{X} + \Delta\boldsymbol{b}, \tag{10}$$

$$h(\boldsymbol{X}) = h_0(\boldsymbol{X}) + \boldsymbol{Z} \odot \Delta h(\boldsymbol{X}). \tag{11}$$

Here $\odot$ is the element-wise multiplication after expanding the left matrix $\boldsymbol{Z} \in \mathbb{R}^{B\times d_{\text{out}}} \sim \mathcal{N}(\mathbf{1}, \sigma^2\boldsymbol{I})$ into $B \times T \times d_{\text{out}}$ where tokens within the same example share same noise. Motivated by [36], the σ decreases linearly as block depth increases. Let f_1 and f_2 be two networks share same weights but non-share noises. The loss function for PACE is:

$$\mathcal{L}^{\text{PACE}} = \frac{1}{n}\sum_{i=1}^{n} \ell(f_1(\boldsymbol{x}_i), \boldsymbol{y}_i) + \lambda\|f_1(\boldsymbol{x}_i) - f_2(\boldsymbol{x}_i)\|_2^2, \tag{12}$$

where ℓ is the classification loss and λ is a hyperparameter controlling regularization strength. During inference, noise and regularization are ommitted, $\Delta\boldsymbol{W}, \Delta\boldsymbol{b}$ are integrated with $\boldsymbol{W}_0, \boldsymbol{b}_0$ for efficiency:

$$\boldsymbol{W} = \boldsymbol{W}_0 + \Delta\boldsymbol{W}; \quad \boldsymbol{b} = \boldsymbol{b}_0 + \Delta\boldsymbol{b}; \quad h(\boldsymbol{X}) = \boldsymbol{W}\boldsymbol{X} + \boldsymbol{b}. \tag{13}$$

Efficient PACE variants. In §C, We present two variants that match the computational/memory costs of the baseline while achieving superior performance with substantially reduced resources.

4 Experiments

We combine LoRA_{mul} and VPT_{add} to form a strong baseline LoRA_{mul}+VPT_{add}, outperforming other combinations in most cases. We evaluate our method across four visual classification adaptation tasks: VTAB-1K [77], few-shot learning [29], FGVC [27] and domain adaptation [81]. We demonstrate PACE improves LoRA on GLUE [69] for text classification and GSM-8K [9] for text generation.

Datasets and evluations. VTAB-1K comprises 19 datasets clustered into (i) Natural images, (ii) Specialized datasets (remote sensing, medical) and (iii) Structured datasets (scene structure) domains. Each dataset has 1K training examples. Following [77, 27], we use the provided 800-200 train split for hyperparameter selection, evaluate using the full training set and report average accuracy across three trails. **Few-shot learning** involves 5 fine-grained datasets: FGVC-Aircraft [45], Food101 [4], OxfordFlowers102 [50], OxfordPets [52] and StanfordCars [34]. Following [29], we evaluate 1,

Table 1: Results on VTAB-1K with ViT-B/16. Mean Acc. is the average of group mean values.

Method	Natural							Specialized				Structured								Mean Acc.
	Cifar100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	NsORB-Ele	
Full	68.9	87.7	64.3	97.3	86.9	87.4	38.8	79.7	95.7	84.2	73.9	56.3	58.6	41.7	65.5	57.5	46.7	25.7	29.1	68.9
Linear	64.4	85.0	63.2	97.0	86.3	36.6	51.0	78.5	87.5	68.5	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	57.6
VPT-Deep	78.8	90.8	65.8	98.0	88.3	78.1	49.6	81.8	96.1	83.4	68.4	68.5	60.0	46.5	72.8	73.6	47.9	32.9	37.8	72.0
Adapter	69.2	90.1	68.0	98.8	89.9	82.8	54.3	84.0	94.9	81.9	75.5	80.9	65.3	48.6	78.3	74.8	48.5	29.9	41.6	73.9
AdaptFormer	70.8	91.2	70.5	99.1	90.9	86.6	54.8	83.0	95.8	84.4	76.3	81.9	64.3	49.3	80.3	76.3	45.7	31.7	41.1	74.7
LoRA	67.1	91.4	69.4	98.8	90.4	85.3	54.0	84.9	95.3	84.4	73.6	82.9	69.2	49.8	78.5	75.7	47.1	31.0	44.0	74.5
NOAH	69.6	92.7	70.2	99.1	90.4	86.1	53.7	84.4	95.4	83.9	75.8	82.8	68.9	49.9	81.7	81.8	48.3	32.8	44.2	74.2
RepAdapter	69.0	92.6	**75.1**	99.4	91.8	90.2	52.9	87.4	95.9	87.4	75.5	75.9	62.3	53.3	80.6	77.3	54.9	29.5	37.9	76.1
RLRR	75.6	92.4	72.9	99.3	91.5	89.8	57.0	86.8	95.2	85.3	75.9	79.7	64.2	53.9	82.1	83.9	53.7	33.4	43.6	76.7
GLoRA	76.4	92.9	74.6	**99.6**	**92.5**	91.5	57.8	87.3	**96.8**	88.0	76.0	83.1	67.3	54.5	**86.2**	83.8	52.9	37.0	41.4	78.0
Baseline	74.9	93.3	72.0	99.4	91.0	91.5	54.8	83.2	95.7	86.9	74.2	83.0	70.5	51.9	81.4	77.9	51.7	33.6	44.4	76.4
+PACE	**79.0**	**94.2**	73.6	99.4	92.4	**93.7**	**58.0**	**87.4**	96.4	**89.3**	**77.1**	**84.9**	**70.9**	**54.9**	84.3	**84.7**	**57.3**	**39.3**	**44.8**	**79.0**

Table 2: Classification accuracy on Few-shot learning with ViT-B/16 pretrained on ImageNet-21K.

Method \ Shot	FGVCAircraft					Food101					Flowers102				
	1	2	4	8	16	1	2	4	8	16	1	2	4	8	16
LoRA_{add}	10.4	15.2	27.2	41.7	59.2	33.9	51.9	59.3	66.0	71.3	93.3	96.4	98.0	98.6	98.7
+PACE	10.7	16.3	28.2	42.1	61.0	40.6	55.9	63.8	70.3	75.2	95.0	98.0	98.9	99.5	99.6
VPT_{add}	11.2	15.1	23.7	36.3	51.5	34.3	56.6	64.8	71.7	75.4	94.3	97.6	98.2	99.3	99.6
+PACE	11.6	16.2	24.0	37.0	52.4	**39.9**	57.2	66.7	72.4	76.1	**95.3**	97.8	98.6	99.4	99.6
LoRA_{mul}+VPT_{add}	10.5	15.6	28.4	44.8	61.8	35.4	54.3	64.8	72.1	76.4	90.4	97.3	98.4	99.4	99.5
+PACE	**12.3**	**16.8**	**29.9**	**45.7**	**62.5**	39.3	**57.2**	**66.7**	**73.4**	**77.8**	93.4	**98.1**	**99.1**	**99.5**	**99.7**
	OxfordPets					StanfordCars					Average				
LoRA_{add}	73.2	83.1	87.5	89.2	91.1	8.7	15.3	30.2	55.3	74.5	43.9	52.3	60.4	70.1	78.9
+PACE	75.3	85.0	**90.7**	90.8	92.4	9.4	16.0	30.9	56.1	75.9	46.2	54.2	62.5	71.7	80.8
VPT_{add}	75.9	85.6	90.3	90.6	92.3	9.3	15.0	27.8	46.6	65.1	45.0	53.9	60.9	68.9	76.7
+PACE	**78.2**	87.4	90.3	91.1	92.3	**9.9**	15.4	27.9	47.0	65.9	**46.9**	54.8	61.5	69.3	77.2
LoRA_{mul}+VPT_{add}	69.9	84.1	89.1	91.3	91.9	9.0	16.3	32.7	59.0	76.4	43.0	53.5	62.6	73.2	81.2
+PACE	76.5	**88.0**	90.3	**91.4**	**92.4**	9.7	**16.4**	**33.7**	**59.8**	**77.3**	46.2	**55.3**	**63.9**	**73.9**	**81.9**

2, 4, 8 and 16 shots, train on the provided training set, tune hyperparameters using validation and report average test accuracy over three random seeds. **FGVC** includes 5 fine-grained datasets: CUB-200-2011 [68], NABirds [65], OxfordFlowers [50], StanfordDogs [10] and StanfordCars [34]. We follow [27] to use validation set for hyperparameter and report test results. For **domain adaptation**, following [81, 7], we train on ImageNet [11] with a 16-shot setting, use the validation split by [81] for hyperparameter selection and report the results on the official validation set and 4 out-of-domain datasets: ImageNet-Sketch [70], ImageNet-V2 [55], ImageNet-A [22] and ImageNet-R [21]. We evaluate on GLUE [69] for **text classification** and GSM-8K [9] for **mathematical reasoning**.

Pre-trained backbones. We experiment with two vision transformers, Vision Transforms (ViT-B/16) [15] and Swin Transformer (Swin-B) [43]. These two are pre-trained on ImageNet-21K [11]. We test a ViT-B-Laion-IN12K model, pre-trained on Laion-2B [59] and fine-tuned on ImageNet-12K [11]. We use $\text{RoBERTa}_{\text{base}}$ [42] and Phi-3-mini-4k-instruct [1] for text classification and generation.

Implementation details. We follow [27] for image processing. 224×224 resizing for VTAB-1K; random flips and crops to 224×224 for FGVC and few-shot learning; stronger augmentation for domain adaptation task, following [15, 81, 40]. We use the Adam optimizer [31] with cosine learning rate decay and linear warm-up (first 10 epochs). Models are fine-tuned for 300 epochs on VTAB-1K and 100 epochs on other vision adaptation tasks, with batch size 64. For text classification we follow [25]. See §G for mathematical reasoning details. All experiments used an NVIDIA H100 GPU.

Baseline. For each dataset, we identified the better method (LoRA_{mul}+VPT_{add} or LoRA_{add}) and tuned the rank, learning rate, and weight decay to form a strong baseline. The detailed baseline settings for each task and the number of trainable parameters, are provided in §F, where LoRA_{mul}+VPT_{add} generally outperformed other variants. Building on the strong LoRA_{mul}+VPT_{add}, we use grid search for our λ and σ, following strategies from previous studies [27, 40, 25]. Beyond LoRA_{mul}+VPT_{add}, PACE also enhances PEFT methods like AdaptFormer, GLoRA, COFT, and BOFT (§D.4).

Table 3: Results on FGVC with ViT-B/16.
* denotes using augmented ViT by AugReg [61].

Method	CUB -2011	NA-Birds	Oxford Flowers	Stan. Dogs	Stan. Cars	Mean Acc.
Full	87.3	82.7	98.8	89.4	84.5	85.9
Linear	85.3	75.9	97.9	86.2	51.3	79.3
VPT	88.5	84.2	99.0	90.2	83.6	89.1
LoRA	88.3	85.6	99.2	91.0	83.2	89.5
SSF*	89.5	85.7	99.6	89.6	89.2	90.7
ARC*	89.3	85.7	**99.7**	89.1	**89.5**	90.7
RLRR*	89.8	85.3	99.6	90.0	90.4	91.0
LoRA_{mul}+VPT_{add}	88.9	87.1	99.4	91.2	87.5	90.8
+PACE	**89.8**	**87.3**	99.5	**92.2**	88.8	**91.5**

Table 4: Results on domain adaptation with ViT-B/16 pretrained on ImageNet-21K.

Method	Source	Target				Mean
	ImageNet	-Sketch	-V2	-A	-R	Acc.
Full	63.9	18.5	52.5	3.2	21.2	31.8
Linear	67.9	14.4	60.8	9.4	25.6	35.6
Adapter	70.5	16.4	59.1	5.5	22.1	34.7
VPT	70.5	18.3	58.0	4.6	23.2	34.7
LoRA	70.8	20.0	59.3	6.9	23.3	36.0
NOAH	71.5	24.8	66.1	11.9	28.5	40.5
GLoRA	78.3	30.6	67.5	13.3	31.0	44.1
LoRA_{mul}+VPT_{add}	78.3	30.6	68.5	14.1	32.5	44.8
+PACE	**79.0**	**31.8**	**69.4**	**16.3**	**35.2**	**46.3**

Table 5: Results for GLUE w/ $\text{RoBERTa}_{\text{base}}$. Matthew's correlation for COLA, Pearson correlation for STSB, and accuracy for others.

Method	COLA	STSB	MRPC	RTE	QNLI	SST2	Avg.
Full	63.6	91.2	90.2	78.7	92.8	94.8	85.2
BitFit	62.0	90.8	**92.7**	81.5	91.8	93.7	85.4
Adapt	62.6	90.3	88.4	75.9	93.0	94.7	84.2
VeRA	65.6	90.7	89.5	78.7	91.8	94.6	85.2
LoRA	63.4	91.5	89.7	86.6	93.3	95.1	86.6
+PACE	**66.2**	**92.0**	91.4	**86.9**	**93.6**	**95.6**	**87.6**

Table 6: Results for GSM-8K using Phi-3-mini-4k-instruct.

Method	Accuracy
Pre-trained	62.01
Full	73.16
LoRA	75.66
+PACE	**78.77**

Table 7: Classification results on domain adaptation and CIFAR-100 in VTAB-1K based different pretrained models. Src. is short for 'source' in Table 4.

Method	ViT-B (ImageNet-21K)						ViT-B (Laion2B-ImageNet-12K)						Swin-B (ImageNet-21K)					
	CIFAR	ImageNet-1K					CIFAR	ImageNet-1K					CIFAR	ImageNet-1K				
	-100	Src.	-S	-V	-A	-R	-100	Src.	-S	-V	-A	-R	-100	Src.	-S	-V	-A	-R
Full	51.6	63.9	18.5	52.5	3.2	21.2	51.2	66.0	29.0	56.1	8.1	27.9	65.6	71.7	27.0	61.1	10.8	24.4
Linear	63.4	67.9	14.4	60.8	9.4	25.6	61.9	79.2	43.2	69.5	23.4	40.9	65.0	78.8	36.7	68.8	23.2	35.9
LoRA_{add}	71.2	73.8	27.1	64.8	13.6	25.0	71.3	77.5	39.8	67.8	20.4	35.6	74.3	76.3	30.7	65.7	16.8	28.9
VPT_{add}	73.6	74.3	27.1	65.9	11.5	26.7	71.8	78.4	40.4	68.7	22.4	38.4	72.7	76.2	30.6	66.2	17.6	29.1
LoRA_{mul}	73.4	78.1	31.2	68.3	13.4	32.7	73.2	78.6	41.9	68.8	22.6	37.8	73.9	76.1	30.8	65.7	18.1	28.9
LoRA_{add}+VPT_{add}	70.3	76.8	28.7	66.6	13.7	29.9	71.8	78.0	41.4	68.3	20.6	36.9	74.5	76.3	30.7	65.7	16.8	28.9
LoRA_{mul}+VPT_{add}	74.9	78.3	30.6	68.5	14.1	32.5	73.8	78.3	41.5	68.6	21.6	38.2	74.6	76.6	31.2	66.5	18.5	29.4
+PACE	**79.0**	**79.0**	**31.8**	**69.4**	**16.3**	**35.2**	**78.0**	**80.1**	**45.8**	**71.2**	**24.6**	**43.6**	**78.9**	**79.6**	**39.2**	**70.1**	**25.2**	**38.0**

4.1 Comparison with the State of the Arts

Results on VTAB-1K. Table 1 presents the results comparing PACE with recent state-of-the-art PEFT methods. PACE improves the strong baseline by 2.6% accuracy, surpassing the previous SOTA GLoRA [7] by 1%, which uses two stages for parameter search. In §D.1, we show that reducing training epochs to 50 or 100 has minimal impact on PACE performance.

Results on Few-shot Learning. Table 2 compares performance w/ and w/o our PACE. PACE improves LoRA_{add}, VPT_{add}, LoRA_{mul}+VPT_{add}, with LoRA_{mul}+VPT_{add} +PACE performing best in most cases. PACE yields notable improvement, especially when the number of shot is small.

Results on FGVC. Table 3 shows that PACE improves the strong LoRA_{mul}+VPT_{add} by 0.7%, outperforming SSF [40], ARC [13] and RLRR [14] that use strongly pre-trained ViT with augmentations. In §D.2, PACE achieves larger improvements on smaller datasets.

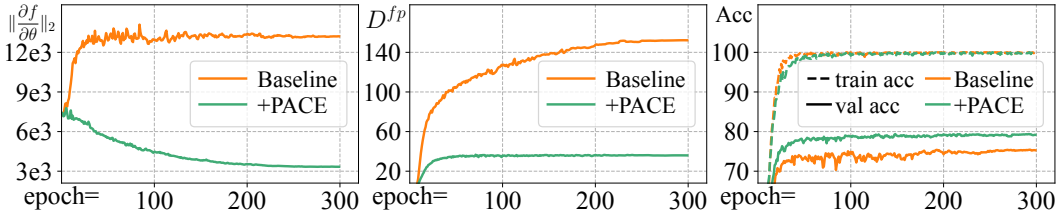
Results on domain adaptation. Table 4 compares PACE with others. LoRA_{mul}+VPT_{add} outperforms GLoRA [7] which relies on parameter search. Meanwhile, PACE improves LoRA_{mul}+VPT_{add} by 1.5%, outperforming other PEFT methods, demonstrating superior performance on domain adaptation.

Results on text classification and mathematical reasoning. Table 5 shows that PACE outperforms LoRA by 1% on GLUE text classification and by 3.11% on GSM-8K mathematical reasoning.

Generalize to other backbones. We evaluate PACE on CIFAR-100 (VTAB-1K) and domain adaptation using Swin-B [43] pretrained on ImageNet-21K and ViT-B (pretrained on Laion 2B, then fine-tuned on ImageNet-12K). Table 7 shows PACE outperforms baseline LoRA_{mul}+VPT_{add} and other PEFT methods across all backbones, demonstrating its effective generalizability. Further experiments in §D.3 show PACE works effectively with self-supervised models like MAE [18] and DINO [5].

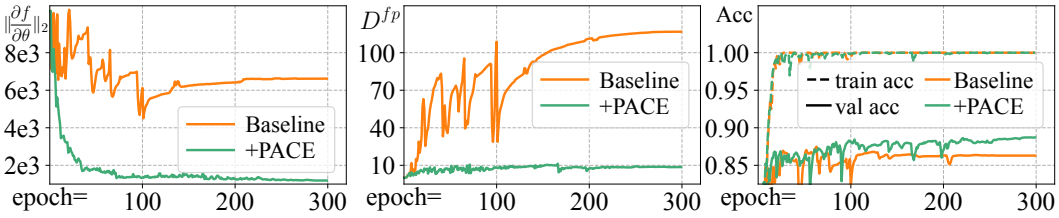
4.2 Analyses

To verify our theories, we conduct experiments on CIFAR-100 (VTAB-1K) using ViT-B/16 and Camelyon (VTAB-1K) on Swin-B. Figure 3 & 4 plot the gradient norm (summed across all layers) and FP-distance (Eq. 5) and the train & validation accuracy during training for baseline LoRA$_{\text{mul}}$+VPT$_{\text{add}}$ and PACE on validation set. Figures 3a & 4a show that PACE has a smaller gradient norm than baseline, verifying Theorem 2 that PACE can implicitly lower the weight gradient norm for better generalization. Figures 3b & 4b demonstrate that PACE maintains a lower FP-distance than the baseline, verifying Theorem 3 that PACE can implicitly align the fine-tuned model with pre-trained model, retaining knowledge developed from large-scale pre-training. Owing to the advantages of the gradient regularization and model alignment, PACE shortens the performance gap between seen and unseen data, yielding higher accuracy on the unseen validation set, as shown in Figures 3c & 4c.



(a) Gradient Norm. (b) FP-Distance (c) Train and validation accuracy.

Figure 3: Analysis for PACE. (a) gradient norm, (b) FP-Distance and (c) train & val accuracy, are evaluated on validation set of CIFAR-100 (VTAB-1K) with baseline LoRA$_{\text{mul}}$+VPT$_{\text{add}}$ on ViT-B/16.



(a) Gradient Norm. (b) FP-distance (c) Train and validation accuracy.

Figure 4: Analysis for PACE. (a) gradient norm, (b) FP-Distance and (c) train & val accuracy, are evaluated on validation set of Camelyon (VTAB-1K) with baseline LoRA$_{\text{mul}}$+VPT$_{\text{add}}$ on Swin-B.

To clarify why naive alignment is problematic, we vary the regularization strength λ over a wide range (1e-3 to 5e4) for both Fine-tuned Pre-trained model Alignment (FPA) by minimizing D^{fp} in Eq. 5) and PACE. Figure 5 plots the averaged gradient norm over training (see also Figures 8 & 9 for more visualizations). PACE robustly lowers gradient norms with larger λ, while FPA exhibits unpredictable behavior, even causing gradient explosion. This verifies Prop. 1 that minimizing D^{fp} is problematic for gradient regularization, complicating gradient management.

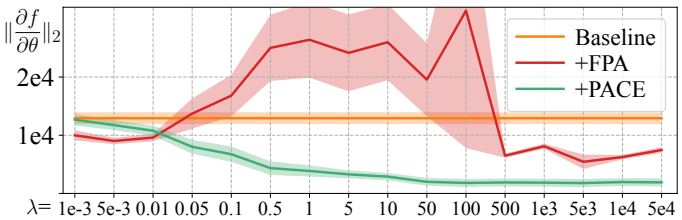


Figure 5: Gradient norms of models across wide range of regularization strengths λ on CIFAR-100 (VTAB-1K) w/ ViT-B/16. Line and shadow represent mean and std across training epochs.

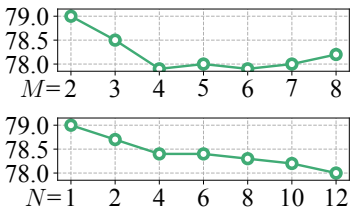


Figure 6: Ablation results for applying PACE among M nets and lazily at every N steps.

4.3 Ablation studies

We ablate PACE based on the baseline LoRA$_{\text{mul}}$+VPT$_{\text{add}}$ on CIFAR-100 (VTAB-1K) and ImageNet-1K in domain adaption as shown in Table 8. The ablations include Noise (baseline w/ noise perturbing

adapter), PACE$_{\text{add}}$ (replacing multiplicative noise with additive noise), PACE$_h$ (perturbing h instead of Δh in Eq. 11), PACE$_{\text{drop}}$ (replacing Gaussian noise with dropout noise), PACE$_{\sigma=}$ (all transformer blocks share the same σ), PACE$_{\sigma\uparrow}$ (σ increases linearly with depth), FPA (fine-tuned and pre-trined alignment by minimizing Eq. 5), SAM (sharpness-aware minimization [16]), GP (gradient penalization), ℓ_1 (sparsity regularization), and transfer learning methods L2SP [76], DELTA [39] and FTP [63]. We grid-search hyperparameters and report the best results.

Table 8 presents the results for all variants. PACE improves over Noise, which itself is better than baseline, justifying our adapter perturbation and consistency regularization. PACE$_{\text{add}}$ performs worse than PACE, showing the superiority of multiplicative noise. Although PACE$_h$ can implicitly regularize gradients, it underperforms PACE, verifying the advantages of perturbing adapter to implicitly align models. PACE$_{\text{drop}}$ is worse than PACE, indicating dropout noise is suboptimal. PACE$_{\sigma=}$ and PACE$_{\sigma\uparrow}$ performs worse, justifying our design of linearly decreasing σ. FPA, SAM and GP, which either only align models or only regularize gradients, are outperformed by PACE. Despite combining FPA+GP, it still underperforms ours, suggesting ineffective combination. ℓ_1, L2SP, DELTA, and FTP obtain worse results than PACE, showing their limitations in improving generalization. PACE regularizes gradients for better generalization and align models to retain knowledge, surpassing all other variants.

Method	CIFAR -100	ImageNet-1K Source	-Sketch	-V2	-A	-R
LoRA$_{\text{mul}}$+VPT$_{\text{add}}$	74.9	78.3	30.6	68.5	14.1	32.5
+Noise	77.4	78.3	31.3	68.6	14.3	33.0
+PACE	**79.0**	**79.0**	**31.8**	**69.4**	16.3	**35.2**
+PACE$_{\text{add}}$	75.7	78.3	31.2	68.7	13.7	32.7
+PACE$_h$	75.9	78.4	31.2	68.1	13.8	32.6
+PACE$_{\text{drop}}$	78.3	78.9	31.2	68.9	16.0	34.6
+PACE$_{\sigma=}$	77.9	78.8	31.6	68.3	**16.6**	34.7
+PACE$_{\sigma\uparrow}$	77.3	78.7	31.3	68.9	14.0	33.6
+FPA	76.6	78.8	31.2	68.6	14.7	33.5
+SAM [16]	75.4	78.4	31.4	68.5	13.8	32.9
+GP	75.8	78.3	31.7	68.4	14.2	32.1
+FPA+GP	74.9	78.1	31.5	68.1	13.5	32.6
+ℓ_1	75.2	78.2	30.6	68.6	13.7	32.8
+L2SP [76]	75.9	78.5	30.4	68.7	14.9	33.5
+DELTA [39]	76.4	78.4	30.8	68.7	14.6	33.7
+FTP [63]	76.2	78.6	30.8	68.6	15.8	33.6

Table 8: Accuracy results on domain adaptation and VTAB-1K based different pretrained models.

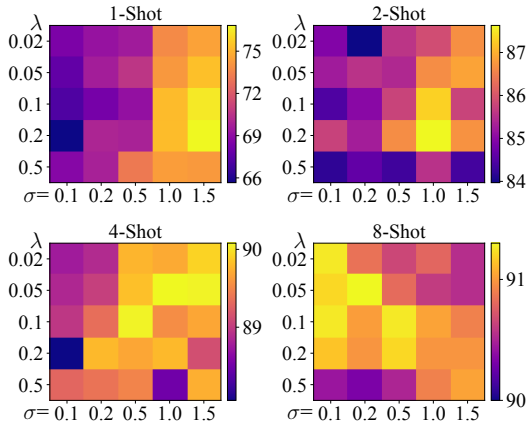


Figure 7: Results for varied λ and σ as well as shot on dataset OxfordPets in few-shot learning.

We further evaluate applying PACE across multiple M networks during training or applying it lazily with half batch size at every N steps (PACE$^{\text{half}}_{\text{lazy}}$ in §C). Figure 6 presents the results, showing that applying PACE among two networks at every training step performs best. However, lazy regularization applied every few steps can still provide reasonable results while saving computational/memory costs.

We test the sensitivity of hyperparameter λ and σ introduced in our PACE on OxfordPets for few-shot learning accross 1, 2, 4, 8 shots. The results presented in Figure 7 demonstrate that with less data, larger λ and σ are favoured, verifying that the effectiveness of PACE in improving generalization.

5 Conclusions

We have introduced PACE, a novel and effective method that combines generalization of PArameter-efficient fine-tuning with Consistency rEgularization. Through rigorous theoretical analyses, we have shown PACE reduces weight gradient for improved generalization and aligns the fine-tuned model with the pre-trained model for retaining pre-training knowledge. Our experimental results support the theoretical analyses, justifying the generalization advantages of PACE over other PEFT methods. With its dual advantages, PACE consistently outperforms other variants across different backbones, firmly establishing PACE as a powerful solution for enhancing generalization for PEFT methods. Limitations and border impacts are discussed in §A.

Acknowledgements. We thank Moyang Liu, Melody Ip, Chenyi Du, and Yinuo Xu for their valuable discussions and support. PK is funded by CSIRO's Science Digital.

References

[1] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024. 7, 22

[2] Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *JMLR*, 15(110):3743–3773, 2014. 5

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 1

[4] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 446–461. Springer, 2014. 6

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 8

[6] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems*, 34:22405–22418, 2021. 3

[7] Arnav Chavan, Zhuang Liu, Deepak Gupta, Eric Xing, and Zhiqiang Shen. One-for-all: Generalized lora for parameter-efficient fine-tuning. *arXiv preprint arXiv:2306.07967*, 2023. 2, 3, 7, 8, 21, 22

[8] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022. 1, 2, 3, 21

[9] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 6, 7, 22

[10] E Dataset. Novel datasets for fine-grained image categorization. In *First Workshop on Fine Grained Visual Categorization, CVPR. Citeseer. Citeseer*. Citeseer, 2011. 7

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1, 7, 20

[12] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[13] Wei Dong, Dawei Yan, Zhijun Lin, and Peng Wang. Efficient adaptation of large vision transformer via adapter re-composing. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 8

[14] Wei Dong, Xing Zhang, Bihui Chen, Dawei Yan, Zhijun Lin, Qingsen Yan, Peng Wang, and Yang Yang. Low-rank rescaled vision transformer fine-tuning: A residual design approach. *arXiv preprint arXiv:2403.19067*, 2024. 2, 8

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 3, 7

[16] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. 3, 4, 5, 10

[17] Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12799–12807, 2023. 2, 3, 4

[18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 8, 20

[19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 1

[20] Zhezhi He, Adnan Siraj Rakin, Jingtao Li, Chaitali Chakrabarti, and Deliang Fan. Defending and harnessing the bit-flip based adversarial weight attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14095–14103, 2020. 4

[21] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021. 7

[22] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15262–15271, 2021. 7

[23] 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. 1

[24] 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. 2, 3

[25] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 2, 3, 7

[26] Shengding Hu, Zhen Zhang, Ning Ding, Yadao Wang, Yasheng Wang, Zhiyuan Liu, and Maosong Sun. Sparse structure search for delta tuning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 2

[27] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022. 2, 4, 6, 7, 22

[28] Zeyinzi Jiang, Chaojie Mao, Ziyuan Huang, Ao Ma, Yiliang Lv, Yujun Shen, Deli Zhao, and Jingren Zhou. Res-tuning: A flexible and efficient tuning paradigm via unbinding tuner from backbone. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[29] Shibo Jie and Zhi-Hong Deng. Fact: Factor-tuning for lightweight adaptation on vision transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1060–1068, 2023. 2, 6

[30] Muhammad Uzair Khattak, Syed Talal Wasim, Muzammal Naseer, Salman Khan, Ming-Hsuan Yang, and Fahad Shahbaz Khan. Self-regulating prompts: Foundational model adaptation without forgetting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15190–15200, 2023. 3

[31] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[32] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 1

[33] Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. VeRA: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[34] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013. 6, 7

[35] Yoonho Lee, Annie S Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *The Eleventh International Conference on Learning Representations*, 2023. 2

[36] Bonan Li, Yinhan Hu, Xuecheng Nie, Congying Han, Xiangjian Jiang, Tiande Guo, and Luoqi Liu. Dropkey for vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22700–22709, 2023. 6

[37] Dongyue Li and Hongyang Zhang. Improved regularization and robustness for fine-tuning in neural networks. *Advances in Neural Information Processing Systems*, 34:27249–27262, 2021. 3

[38] Shengrui Li, Xueting Han, and Jing Bai. Adaptergnn: Parameter-efficient fine-tuning improves generalization in gnns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13600–13608, 2024. 2

[39] Xingjian Li, Haoyi Xiong, Hanchao Wang, Yuxuan Rao, Liping Liu, Zeyu Chen, and Jun Huan. Delta: Deep learning transfer using feature map with attention for convolutional networks. *arXiv preprint arXiv:1901.09229*, 2019. 3, 10

[40] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 3, 7, 8, 22

[41] Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, Yandong Wen, Michael J. Black, Adrian Weller, and Bernhard Schölkopf. Parameter-efficient orthogonal finetuning via butterfly factorization. In *ICLR*, 2024. 2, 21

[42] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 7

[43] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 7, 8

[44] Gen Luo, Minglang Huang, Yiyi Zhou, Xiaoshuai Sun, Guannan Jiang, Zhiyu Wang, and Rongrong Ji. Towards efficient visual adaption via structural re-parameterization. *arXiv preprint arXiv:2302.08106*, 2023. 2, 4, 22

[45] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 6

[46] Yao Ni and Piotr Koniusz. Chain: Enhancing generalization in data-efficient gans via lipschitz continuity constrained normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6763–6774, June 2024. 3

[47] Yao Ni and Piotr Koniusz. Nice: Noise-modulated consistency regularization for data-efficient gans. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 16

[48] Yao Ni, Piotr Koniusz, Richard Hartley, and Richard Nock. Manifold learning benefits gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11265–11274, 2022. 3

[49] Yao Ni, Dandan Song, Xi Zhang, Hao Wu, and Lejian Liao. Cagan: Consistent adversarial training enhanced gans. In *IJCAI*, pages 2588–2594, 2018. 2

[50] Maria-Elena Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, 2006. 6, 7

[51] Changdae Oh, Hyeji Hwang, Hee-young Lee, YongTaek Lim, Geunyoung Jung, Jiyoung Jung, Hosik Choi, and Kyungwoo Song. Blackvip: Black-box visual prompting for robust transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24224–24235, 2023. 2

[52] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 6

[53] Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. In *NeurIPS*, 2023. 2, 21

[54] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1

[55] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pages 5389–5400. PMLR, 2019. 7

[56] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1

[57] Shuvendu Roy and Ali Etemad. Consistency-guided prompt learning for vision-language models. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[58] Kuniaki Saito, Donghyun Kim, and Kate Saenko. Openmatch: Open-set semi-supervised learning with open-set consistency regularization. *Advances in Neural Information Processing Systems*, 34:25956–25967, 2021. 2

[59] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. 1, 7

[60] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020. 2

[61] Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research*, 2022. 8

[62] Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, et al. On transferability of prompt tuning for natural language processing. *arXiv preprint arXiv:2111.06719*, 2021. 1, 2

[63] Junjiao Tian, Yen-Cheng Liu, James S Smith, and Zsolt Kira. Fast trainable projection for robust fine-tuning. *Advances in Neural Information Processing Systems*, 36, 2024. 3, 10

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

[65] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 595–604, 2015. 7

[66] Dániel Varga, Adrián Csiszárik, and Zsolt Zombori. Gradient regularization improves accuracy of discriminative models. *arXiv preprint arXiv:1712.09936*, 2017. 3

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 1, 2, 3

[68] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Thecaltech-ucsdbirds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 7

[69] Alex Wang. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018. 6, 7

[70] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. *Advances in Neural Information Processing Systems*, 32, 2019. 7

[71] Yihan Wang, Jatin Chauhan, Wei Wang, and Cho-Jui Hsieh. Universality and limitations of prompt tuning. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3

[72] Yaoming Wang, Yuchen Liu, Xiaopeng Zhang, Jin Li, Bowen Shi, Chenglin Li, Wenrui Dai, Hongkai Xiong, and Qi Tian. Violet: Vision-language efficient tuning with collaborative multi-modal gradients. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 4595–4605, 2023. 3

[73] Yeming Wen and Swarat Chaudhuri. Batched low-rank adaptation of foundation models. *arXiv preprint arXiv:2312.05677*, 2023. 2

[74] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in neural information processing systems*, 33:2958–2969, 2020. 4

[75] Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. R-drop: Regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34:10890–10905, 2021. 2

[76] LI Xuhong, Yves Grandvalet, and Franck Davoine. Explicit inductive bias for transfer learning with convolutional networks. In *International Conference on Machine Learning*, pages 2825–2834. PMLR, 2018. 3, 10

[77] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 6

[78] Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency regularization for generative adversarial networks. *arXiv preprint arXiv:1910.12027*, 2019. 2

[79] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? In *ICLR*, 2021. 5

[80] Shan Zhang, Yao Ni, Jinhao Du, Yanxia Liu, and Piotr Koniusz. Semantic transfer from head to tail: Enlarging tail margin for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1350–1360, 2024. 3

[81] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. *arXiv preprint arXiv:2206.04673*, 2022. 2, 6, 7

[82] Bingchen Zhao, Haoqin Tu, Chen Wei, Jieru Mei, and Cihang Xie. Tuning layernorm in attention: Towards efficient multi-modal LLM finetuning. In *The Twelfth International Conference on Learning Representations*, 2024. 2

[83] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Penalizing gradient norm for efficiently improving generalization in deep learning. In *International Conference on Machine Learning*, pages 26982–26992. PMLR, 2022. 3

PACE: marrying generalization of PArameter-efficient fine-tuning with Consistency rEgularization (Supplementary Material)

Yao Ni[†] **Shan Zhang**[‡,†] **Piotr Koniusz** [*,§,†]
[†]The Australian National University [§]Data61♥CSIRO
[‡]Australian Institute for Machine Learning, University of Adelaide
[†]yao.ni@anu.edu.au [‡]shan.zhang@adelaide.edu.au [§]piotr.koniusz@data61.csiro.au

A Broader impacts and limitations

A.1 Broader impacts

Our work provides a powerful solution for improving generalization in Parameter Efficient Fine-Tuning (PEFT), allowing for effective fine-tuning of pre-trained models while reduce the heavily reliance on pretraining from scratch using massive data. Our advancement in PEFT, supported by Theorems 1, 2 and 3, offer novel insights into gradient regularization and model alignment. These insights extend beyond PEFT and can be applied to other areas such as continual learning and transfer learning, potentially enhancing the performance and efficiency of models in various domains. By leveraging our findings, practitioners can develop more robust and adaptable models that generalize well to new tasks and environments, leading to more intelligent and versatile AI systems. In terms of negative impacts, the robustness of our fine-tuning method could potentially be misused to create more convincing deepfakes, raising concerns about the spread of misinformation, manipulation of public opinion, and malicious activities such as fraud, blackmail, or harassment.

A.2 Limitations

While our work effectively improves generalization ability, it introduces additional computational costs by requiring input samples to be passed through the network twice for regularization. However, this can be mitigated by using two efficient variants, $\text{PACE}_{\text{fast}}$ and $\text{PACE}^{\text{half}}_{\text{lazy}}$, proposed in §C, where we demonstrate the potential for resource-efficient finetuning. Additionally, our method introduces extra hyperparameters λ and σ, which require caution during hyperparameter search. Nonetheless, Figure 7 suggests that fewer training data requires larger λ and σ values, providing insight for hyperparameter tuning.

[*]The corresponding author.

B Proofs

B.1 Proof of Theorem 1

Settting $\boldsymbol{\epsilon} = \frac{\rho \nabla_{\boldsymbol{\theta}}}{\|\nabla_{\boldsymbol{\theta}}\|_2}$, we perform a second-order Taylor expansion of $\mathcal{L}_{\mathcal{D}^n}$ around $\boldsymbol{\theta}$. By incorporating the higher-order terms from the Taylor expansion into $R\Big(\frac{\|\boldsymbol{\theta}\|_2^2}{\rho^2}, \frac{1}{n}\Big)$, we derive:

$$\begin{aligned}\mathcal{L}_{\mathscr{D}}(\boldsymbol{\theta}) &\leq \mathcal{L}_{\mathcal{D}^n}\Big(\boldsymbol{\theta} + \frac{\rho \nabla_{\boldsymbol{\theta}}}{\|\nabla_{\boldsymbol{\theta}}\|_2}\Big) + R\Big(\frac{\|\boldsymbol{\theta}\|_2^2}{\rho^2}, \frac{1}{n}\Big) \\ &\approx \mathcal{L}_{\mathcal{D}^n}(\boldsymbol{\theta}) + \rho\|\nabla_{\boldsymbol{\theta}}\|_2 + \frac{\rho^2}{2\|\nabla_{\boldsymbol{\theta}}\|_2^2}\nabla_{\boldsymbol{\theta}}^T \boldsymbol{H}_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} + R\Big(\frac{\|\boldsymbol{\theta}\|_2^2}{\rho^2}, \frac{1}{n}\Big) \end{aligned} \tag{14}$$

Assuming that the approximation does not alter the inequality relationship, i.e., it preserves the $\leq$ relation on both sides and considering the largest eigenvalue of $\boldsymbol{H}_{\boldsymbol{\theta}}$ as $\lambda_{\max}^{\boldsymbol{H}}$, implying $\boldsymbol{v}^T \boldsymbol{H}_{\boldsymbol{\theta}} \boldsymbol{v} \leq \lambda_{\max}^{\boldsymbol{H}} \|\boldsymbol{v}\|_2^2$ for any $\boldsymbol{v}$, we further bound Eq. 14 as follows and arrive at:

$$\mathcal{L}_{\mathscr{D}}(\boldsymbol{\theta}) \leq \mathcal{L}_{\mathcal{D}^n}(\boldsymbol{\theta}) + \rho\|\nabla_{\boldsymbol{\theta}}\|_2 + \frac{\rho^2}{2}\lambda_{\max}^{\boldsymbol{H}} + R\Big(\frac{\|\boldsymbol{\theta}\|_2^2}{\rho^2}, \frac{1}{n}\Big)$$

B.2 Proof of Theorem 2

The proof is motivated from [47]. We include the proof process for completeness. Denote $\boldsymbol{m}_1 = \boldsymbol{z}_1 - \mathbf{1}, \boldsymbol{m}_2 = \boldsymbol{z}_2 - \mathbf{1}$ thus $\boldsymbol{m}_1, \boldsymbol{m}_2 \sim \mathcal{N}(\mathbf{0}, \sigma^2)$

$$\begin{aligned} d^{\text{pace}} =& \mathbb{E}_{\boldsymbol{z}_1, \boldsymbol{z}_2}[f(\boldsymbol{\theta}_0 + \boldsymbol{z}_1 \odot \Delta\boldsymbol{\theta}) - f(\boldsymbol{\theta}_0 + \boldsymbol{z}_2 \odot \Delta\boldsymbol{\theta})]^2 \\ =& \mathbb{E}_{\boldsymbol{z}_1, \boldsymbol{z}_2}[f(\boldsymbol{\theta}_0 + \Delta\boldsymbol{\theta} + (\boldsymbol{z}_1 - \mathbf{1}) \odot \Delta\boldsymbol{\theta}) - f(\boldsymbol{\theta}_0 + \Delta\boldsymbol{\theta} + (\boldsymbol{z}_2 - \mathbf{1}) \odot \Delta\boldsymbol{\theta})]^2 \\ =& \mathbb{E}_{\boldsymbol{m}_1, \boldsymbol{m}_2}[f(\boldsymbol{\theta} + \boldsymbol{m}_1 \odot \Delta\boldsymbol{\theta}) - f(\boldsymbol{\theta} + \boldsymbol{m}_2 \odot \Delta\boldsymbol{\theta})]^2 \end{aligned} \tag{15}$$

Defining $\boldsymbol{v} := \boldsymbol{m}_1 \odot \Delta\boldsymbol{\theta}$ and $\boldsymbol{u} := \boldsymbol{m}_2 \odot \Delta\boldsymbol{\theta}$, where $\boldsymbol{v}, \boldsymbol{u} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \text{diag}(\Delta\boldsymbol{\theta} \odot \Delta\boldsymbol{\theta}))$, we can rewrite Eq. 15 as follows:

$$\begin{aligned} & \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}[f(\boldsymbol{\theta} + \boldsymbol{v}) - f(\boldsymbol{\theta} + \boldsymbol{u})]^2 \\ \approx& \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[f(\boldsymbol{\theta}) + \boldsymbol{v}^T\nabla + \frac{1}{2}\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v} - f(\boldsymbol{\theta}) - \boldsymbol{u}^T\nabla - \frac{1}{2}\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}\big]^2 \\ =& \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[\boldsymbol{v}^T\nabla + \frac{1}{2}\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v} - \boldsymbol{u}^T\nabla - \frac{1}{2}\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}\big]^2 \\ =& \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[(\boldsymbol{v} - \boldsymbol{u})^T\nabla + \frac{1}{2}\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v} - \frac{1}{2}\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}\big]^2 \\ =& \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[(\boldsymbol{v} - \boldsymbol{u})^T\nabla\big]^2 && (16) \\ &+ \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[\big((\boldsymbol{v} - \boldsymbol{u})^T\nabla\big)\big(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v} - \boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}\big)\big] && (17) \\ &+ \frac{1}{4}\mathbb{E}_{\boldsymbol{v}}[\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v}]^2 + \frac{1}{4}\mathbb{E}_{\boldsymbol{u}}[\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}]^2 && (18) \\ &- \frac{1}{2}\mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v})(\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u})\big]. && (19) \end{aligned}$$

Next, we derive the four terms, Eq. 16, 17, 18, and 19, respectively as follows:

Eq. 16. Using $\mathbb{E}_{z_1,z_2}[(z_1 - z_2)^2] = 2\sigma^2$ for $z_1, z_2 \sim \mathcal{N}(0, \sigma^2)$, we can simplify (Eq. 16) as follows, noting that terms related to different dimensions are canceled due to zero-mean independent Gaussian noise:

$$\mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[(\boldsymbol{v} - \boldsymbol{u})^T\nabla\big]^2 = \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[\sum_j (v_j - u_j)^2\nabla_j^2\big] = 2\sigma^2\sum_j \Delta\theta_j^2\nabla_k^2. \tag{20}$$

Eq. 17. Utilizing $E[z^3] = \mu^3 + 3\mu\sigma^2$ for $z \sim \mathcal{N}(\mu, \sigma^2)$, and noting that $E[z^3] = 0$ for $\mu = 0$, Eq. 17 is derived as:

$$\begin{aligned} & \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[\big((\boldsymbol{v} - \boldsymbol{u})^T\nabla\big)\big(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v} - \boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}\big)\big] \\ =& \mathbb{E}_{\boldsymbol{v}}\big[(\boldsymbol{v}^T\nabla)(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v})\big] + \mathbb{E}_{\boldsymbol{u}}\big[(\boldsymbol{u}^T\nabla)(\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u})\big] - \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[(\boldsymbol{v}^T\nabla)(\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u})\big] - \mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[(\boldsymbol{u}^T\nabla)(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v})\big] \\ =& 2\mathbb{E}_{\boldsymbol{v}}\big[(\boldsymbol{v}^T\nabla)(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v})\big] = 0. \end{aligned} \tag{21}$$

Eq. 18. We first decompose Eq. 18, then discuss each case and obtain the final result.

$$\frac{1}{4}\mathbb{E}_{\boldsymbol{v}}[\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v}]^2+\frac{1}{4}\mathbb{E}_{\boldsymbol{u}}[\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}]^2=\frac{1}{2}\mathbb{E}_{\boldsymbol{v}}[\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v}]^2=\frac{1}{2}\mathbb{E}_{\boldsymbol{v}}\big[\sum_{j,k,p,q}v_jH_{jk}v_kv_pH_{pq}v_q\big]. \tag{22}$$

Given the independence of elements in $\boldsymbol{v}$, only terms with an element repeated two or four times contribute non-zero results, leading to four distinct, non-overlapping cases. Using $\mathbb{E}[z^2]=\sigma^2+\mu^2$ and $\mathbb{E}[z^4]=\mu^4+6\mu^2\sigma^2+3\sigma^4$ for $z\sim\mathcal{N}(\mu,\sigma^2)$, and simplifying to $\mathbb{E}[z^2]=\sigma^2$ and $\mathbb{E}[z^4]=3\sigma^4$ when $\mu=0$, we have:

Case 1: $j=k\neq p=q$, given the independence of v_j and v_p, we have:

$$\mathbb{E}_{\boldsymbol{v}}\big[\sum_j\sum_{p\neq j}v_j^2H_{jj}v_p^2H_{pp}\big]=\sum_{j,p\neq j}H_{jj}H_{pp}\mathbb{E}[v_j^2]\mathbb{E}[v_p^2]=\sigma^4\sum_{j,k\neq j}H_{jj}H_{kk}\Delta\theta_j^2\Delta\theta_k^2. \tag{23}$$

Case 2: For $j=p\neq k=q$, the independence of v_j and v_k simplifies our calculation, leading to:

$$\mathbb{E}_{\boldsymbol{v}}\big[\sum_j\sum_{k\neq j}v_jH_{jk}v_kv_jH_{jk}v_k\big]=\sum_{j,k\neq j}H_{jk}^2\mathbb{E}[v_j^2]\mathbb{E}[v_k^2]=\sigma^4\sum_{j,k\neq j}H_{jk}^2\Delta\theta_j^2\Delta\theta_k^2. \tag{24}$$

Case 3: For $j=q\neq k=p$, utilizing the independence of v_j and v_k as well as the symmetry $H_{jk}=H_{kj}$, we obtain:

$$\mathbb{E}_{\boldsymbol{v}}\big[\sum_j\sum_{k\neq j}v_jH_{jk}v_kv_kH_{kj}v_j\big]=\sum_{j,k\neq j}H_{jk}^2\mathbb{E}[v_j^2]\mathbb{E}[v_k^2]=\sigma^4\sum_{j,k\neq j}H_{jk}^2\Delta\theta_j^2\Delta\theta_k^2. \tag{25}$$

Case 4: For $j=q=k=p$, using $\mathbb{E}[z^4]=3\sigma^4$ where $z\sim\mathcal{N}(0,\sigma^2)$, we have:

$$\mathbb{E}_{\boldsymbol{v}}\Big[\sum_j v_jH_{jj}v_jv_jH_{jj}v_j\Big]=\sum_j H_{jj}^2\mathbb{E}[v_j^4]=3\sigma^4\sum_j H_{jj}^2\Delta\theta_j^4. \tag{26}$$

Combining above four cases together, we have the result for Eq. 18:

$$\frac{\sigma^4}{2}\Big(\sum_j 3H_{jj}^2\Delta\theta_j^4+\sum_{j,k\neq j}(H_{jj}H_{kk}+2H_{jk}^2)\Delta\theta_j^2\Delta\theta_k^2\Big). \tag{27}$$

Eq. 19:

$$\begin{aligned}
&-\frac{1}{2}\mathbb{E}_{\boldsymbol{v},\boldsymbol{u}}\big[(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v})(\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u})\big]\\
=&-\frac{1}{2}\mathbb{E}_{\boldsymbol{v}}\big[(\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v})\big]\mathbb{E}_{\boldsymbol{u}}\big[(\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u})\big]\\
=&-\frac{1}{2}\mathbb{E}_{\boldsymbol{v}}\big[\sum_j H_{jj}v_j^2\big]\mathbb{E}_{\boldsymbol{u}}\big[\sum_k H_{kk}v_k^2\big]\\
=&-\frac{1}{2}\Big(\sum_j H_{jj}\mathbb{E}[v_j^2]\Big)\Big(\sum_k H_{kk}\mathbb{E}[v_k^2]\Big)\\
=&-\frac{\sigma^4}{2}\Big(\sum_j H_{jj}^2\Delta\theta_j^4+\sum_{j,k\neq j}H_{jj}H_{kk}\Delta\theta_j^2\Delta\theta_k^2\Big).
\end{aligned} \tag{28}$$

With results of Eq. 20, 21, 27, 28, we have the final results:

$$\begin{aligned}
d^{\text{pace}}\approx&2\sigma^2\sum_j\Delta\theta_j^2\nabla_j^2+0\\
&+\frac{\sigma^4}{2}\Big(\sum_j 3H_{jj}^2\Delta\theta_j^4+\sum_{j,k\neq j}(H_{jj}H_{kk}+2H_{jk}^2)\Delta\theta_j^2\Delta\theta_k^2-\sum_j H_{jj}^2\Delta\theta_j^4-\sum_{j,k\neq j}H_{jj}H_{kk}\Delta\theta_j^2\Delta\theta_k^2\Big)\\
=&2\sigma^2\sum_j\Delta\theta_j^2\nabla_j^2+\sigma^4\Big(\sum_j H_{jj}^2\Delta\theta_j^4+\sum_{j,k\neq j}H_{jk}^2\Delta\theta_j^2\Delta\theta_k^2\Big)\\
=&2\sigma^2\sum_j\Delta\theta_j^2\nabla_k^2+\sigma^4\sum_{j,k}H_{jk}^2\Delta\theta_j^2\Delta\theta_k^2=2\sigma^2\|\Delta\boldsymbol{\theta}\odot\boldsymbol{\nabla}\|_2^2+\sigma^4\|(\Delta\boldsymbol{\theta}\Delta\boldsymbol{\theta}^T)\odot\boldsymbol{H}\|_F^2
\end{aligned} \tag{29}$$

B.3 Proof of Theorem 3

The Cauchy-Schwarz inequality states that for $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d$, we have $(\sum_j u_j v_j)^2 \leq (\sum_j u_j^2)(\sum_j v_j^2)$. Let $\boldsymbol{u} = \mathbf{1}$, it follows that $(\sum_j v_j)^2 \leq d\|\boldsymbol{v}\|_2^2$. Using this inequality, we then prove the following:

$$[\Delta\boldsymbol{\theta}^T\boldsymbol{\nabla} - \frac{1}{2}\Delta\boldsymbol{\theta}^T\boldsymbol{H}\Delta\boldsymbol{\theta}]^2 \leq 2[\Delta\boldsymbol{\theta}^T\boldsymbol{\nabla}]^2 + [\Delta\boldsymbol{\theta}^T\boldsymbol{H}\Delta\boldsymbol{\theta}]^2$$

$$[\Delta\boldsymbol{\theta}^T\boldsymbol{\nabla}]^2 = \Big(\sum_j \Delta\theta_j\nabla_j\Big)^2 \leq d\|\Delta\boldsymbol{\theta}\odot\boldsymbol{\nabla}\|_2^2 \tag{30}$$

$$[\Delta\boldsymbol{\theta}^T\boldsymbol{H}\Delta\boldsymbol{\theta}]^2 = \Big(\sum_{j,k} \Delta\theta_j\Delta\theta_k H_{jk}\Big)^2 \leq d^2\big\|(\Delta\boldsymbol{\theta}\Delta\boldsymbol{\theta}^T)\odot\boldsymbol{H}\big\|_F^2 \tag{31}$$

Here, the inequality is obtained by treating $\Delta\theta_j\Delta\theta_k H_{jk}$ as an element of a vector with size of d^2. This leads to the final results.

B.4 Rationale for one-dimensional output analysis

we use the squared L_2 distance for multi-dimensional outputs for D^{fp} and D^{pace}, which allows our one-dimensional analysis to naturally generalize to multiple dimensions. For example, for a vector-valued function in the naive alignment, $f(\boldsymbol{\theta}) = [f_1(\boldsymbol{\theta}), ..., f_m(\boldsymbol{\theta})]$, where m is the output dimension, we have:

$$\|f(\boldsymbol{\theta}_0) - f(\boldsymbol{\theta}_0 + \Delta\boldsymbol{\theta})\|_2^2 = \sum_{i=1}^{m}[f_i(\boldsymbol{\theta}_0) - f_i(\boldsymbol{\theta}_0 + \Delta\boldsymbol{\theta})]^2.$$

This equality shows that the squared L_2 distance in multiple dimensions is simply the sum of non-negative squared differences in each dimension. Consequently, this additive nature enables our one-dimensional analysis to extend seamlessly to multiple dimensions in practice, aligning with our empirical observations.

C Efficient PACE variants

Building upon strong theoretical foundation of PACE for generalization, we demonstrate that simple modifications can reduce its memory and training time requirements. In this section, we explore two efficient variants: $\text{PACE}_{\text{fast}}$ and $\text{PACE}^{\text{half}}_{\text{lazy}}$, both maintaining similar computational and memory requirements as the baseline while improving performance. We then provide empirical results show that $\text{PACE}_{\text{fast}}$ slightly outperforms $\text{PACE}^{\text{half}}_{\text{lazy}}$ while requiring no additional hyperparameters and fewer computational resources. Given its superior efficiency, we further explore the potential of $\text{PACE}_{\text{fast}}$ for resource-efficient fine-tuning. By simply reducing batch size and epochs, $\text{PACE}_{\text{fast}}$ outperforms the baseline while using significantly less GPU memory and training time.

$\text{PACE}_{\text{fast}}$: Building on the observation that only small datasets are typically available for fine-tuning, we assume model behavior changes gradually across epochs. Under this assumption, we store the model outputs from the previous epoch ($f_{e-1}(\boldsymbol{x})$), which contain inherent noise due to adapter perturbation, and compute the consistency regularization loss between these stored outputs and the current epoch's noised outputs:

$$d^{\text{pace}}_{\text{fast}}(\boldsymbol{x}) = \|f(\boldsymbol{x}) - \boldsymbol{o}_{e-1}\|_2^2; \quad \text{where} \quad \boldsymbol{o}_{e-1} = f_{e-1}(\boldsymbol{x}) \tag{32}$$

Here the output vector $\boldsymbol{o} \in \mathbb{R}^C$ where C is the number of classes. Since f applies noise perturbation to the adapter and changes gradually between epochs, $f_{e-1}(\boldsymbol{x})$ and $f(\boldsymbol{x})$ can be seen as applying different i.i.d. noises to similar model states. This approach preserves the theoretical foundation of PACE while incurring minimal storage and computation costs. With typically few classes C and limited samples in fine-tuning, storing $\boldsymbol{o}_{e-1}$ is manageable within GPU or CPU memory.

$\text{PACE}^{\text{half}}_{\text{lazy}}$: During training, the network always applies noise perturbations. Every N-th iteration uses a half batch size and consistency regularization, while all other iterations use the full batch size.

Memory and computational efficiency of two variants. Both variants maintain similar computational and memory requirements as the baseline. To demonstrate this, we conduct experiments

on CIFAR-100 (VTAB-1K) using ViT-16/B, Camelyon (VTAB-1K) with Swin-B, and ImageNet (domain adaptation) with ViT-16/B. Table 9 compares maximum GPU memory usage, total training time, and accuracy for each task, showing that $\text{PACE}_{\text{fast}}$ and $\text{PACE}^{\text{half}}_{\text{lazy}}$ significantly improve upon the baseline while maintaining similar computational demands.

We find that $\text{PACE}_{\text{fast}}$ slightly outperforms $\text{PACE}^{\text{half}}_{\text{lazy}}$ without requiring additional hyperparameters, yet it needs to store outputs from the previous epoch. We therefore analyze its memory requirements.

Table 9: GPU memory usage, training time, and accuracy for $\text{PACE}_{\text{fast}}$ and $\text{PACE}^{\text{half}}_{\text{lazy}}$. 'm' denotes minutes, Both variants outperform the baseline while maintaining similar computational demands.

Method	CIFAR-100 (ViT/16-B)			Camelyon (Swin-B)			ImageNet (ViT/16-B)		
	GPU Memory	Time	Accuracy	GPU Memory	Time	Accuracy	GPU Memory	Time	Mean Acc.
$\text{LoRA}_{\text{mul}}+\text{VPT}_{\text{add}}$	8.9GB	29m	74.6	15.7GB	33m	86.7	8.9GB	161m	44.8
+PACE	17.7GB	53m	79.0	29.4GB	60m	89.3	17.7GB	278m	46.3
$+\text{PACE}_{\text{fast}}$	**9.0GB**	**29m**	**78.3**	15.7GB	34m	88.8	**9.0GB**	**162m**	**46.1**
$+\text{PACE}^{\text{half}}_{\text{lazy}}$ $(N=2)$	9.3GB	29m	78.7	**15.7GB**	**36m**	**89.2**	9.0GB	165m	46.0
$+\text{PACE}^{\text{half}}_{\text{lazy}}$ $(N=4)$	9.3GB	29m	78.4	15.7GB	35m	88.9	9.0GB	163m	45.6
$+\text{PACE}^{\text{half}}_{\text{lazy}}$ $(N=6)$	9.3GB	29m	78.4	15.7GB	35m	89.0	9.0GB	163m	45.7
$+\text{PACE}^{\text{half}}_{\text{lazy}}$ $(N=10)$	9.3GB	29m	78.2	15.7GB	35m	88.9	9.0GB	162m	45.6

Memory efficiency of $\text{PACE}_{\text{fast}}$. We compare the additional memory requirement of $\text{PACE}_{\text{fast}}$ with baseline GPU memory consumption. Table 10 shows that the memory overhead of $\text{PACE}_{\text{fast}}$ is negligible compared to baseline GPU memory requirements and can be easily stored in GPU. Moreover, even in the rare scenario of fine-tuning on the full ImageNet 1K dataset (1.2 million samples), $\text{PACE}_{\text{fast}}$ requires only 4.8GB of additional memory for storing the output of the model's classification head. This is significantly smaller than the dataset itself (>100GB) and can be easily accommodated in CPU/GPU memory.

Table 10: Comparison of $\text{PACE}_{\text{fast}}$ memory overhead and baseline GPU memory requirements.

Dataset	Memory of $\text{PACE}_{\text{fast}}$	Baseline GPU Memory	Ratio
CIFAR-100 (VTAB-1K w/ ViT/16-B)	390KB	8.9GB	0.0042%
Camelyon (VTAB-1K w/ Swin-B)	7.81KB	15.7GB	0.000047%
ImageNet (Domain adaptation w/ ViT/16-B)	61MB	8.9GB	0.67%

Resource-Efficient training with $\text{PACE}_{\text{fast}}$. Given the superior performance, minimal memory overhead, and no need for additional hyperparameters of $\text{PACE}_{\text{fast}}$, we explore its potential for resource-efficient training by maintaining the same number of updates with reduced batch size and proportionally reduced epochs. Table 11 shows that even with 1/8 batch size and epochs, $\text{PACE}_{\text{fast}}$ still outperforms the baseline by 1.7% while only using $\sim$1/3 GPU memory and $\sim$1/4 training time. This demonstrates the robustness and generalization benefits that $\text{PACE}_{\text{fast}}$ brings to models, enabling them to excel under constrained training configurations. Such efficiency is particularly valuable for fine-tuning large foundation models, where resource constraints necessitate small batch sizes and typically lead to sharp loss landscapes, yet the theoretical guarantee of PACE for smooth loss landscapes provides a promising solution for these challenges.

Table 11: Results of $\text{PACE}_{\text{fast}}$ with reduced batch size and epochs on CIFAR-100 (VTAB-1K w/ ViT-16/B), Camelyon (VTAB-1K w/ Swin-B), ImageNet (Domain adaptaion w/ ViT-16/B). $\text{PACE}_{\text{fast}}$ outperforms baseline while using less GPU memory and training time.

Method	CIFAR-100			Camelyon			ImageNet			Average		
	Mem.	Time	Acc.	Mem.	Time	Acc.	Mem.	Time	MeanAcc.	Mem.	Time	Acc.
$\text{LoRA}_{\text{mul}}+\text{VPT}_{\text{add}}$	8.9GB	29m	74.6	15.7GB	33m	86.7	8.9GB	161m	44.8	11.1GB	74m	68.7
$+\text{PACE}_{\text{fast}}$ ($\frac{1}{2}$ batch size, $\frac{1}{2}$ epochs)	5.4GB	17m	78.1	8.6GB	21m	88.9	5.4GB	85m	45.8	6.5GB	41m	70.9
$+\text{PACE}_{\text{fast}}$ ($\frac{1}{4}$ batch size, $\frac{1}{4}$ epochs)	3.5GB	10m	77.8	6.0GB	14m	88.7	3.5GB	50m	45.6	4.3GB	25m	70.7
$+\text{PACE}_{\text{fast}}$ ($\frac{1}{8}$ batch size, $\frac{1}{8}$ epochs)	2.9GB	6m	77.2	5.2GB	10m	88.6	2.9GB	32m	45.5	3.7GB	16m	70.4

Table 12: Classification results for different methods on VTAB-1K with different training epochs.

#Epoch	Method	Natural	Specialized	Structured	Avg.
530	GLoRA	83.61	87.02	63.27	77.97
100	Baseline	81.94	85.40	61.40	76.24
100	+PACE	83.94	87.44	64.62	78.67
50	+PACE (half batch size)	83.77	87.32	63.92	78.34
200	Baseline	82.28	85.30	61.64	76.40
200	+PACE	84.13	87.57	64.85	78.85
300	Baseline	82.41	85.00	61.80	76.40
300	+PACE	84.32	87.55	65.13	79.00

D Additional Experiments

In this section, we provide additional experiments of PACE on VTAB-1K with different epochs, varying training data sizes on FGVC benchmarks, self-supervised pretrained backbones and combinations with other PEFT methods.

D.1 Experiments of VTAB-1K with different epochs

In Table 1, We use 300 epochs for VTAB-1K tasks as we observed slight improvements over 100 epochs. However, this does not mean PACE requires longer training to converge. Since the optimizer uses cosine learning rate decay, reducing the number of training epochs to 100 has minimal impact on performance, as shown in Table 12.

To ensure fair memory and computational budgets, we also tested PACE with half the batch size and 50 epochs. Table 12 shows that under these conditions, PACE still improves baseline accuracy by 2.10%, and outperforms the previous SOTA GLoRA, which uses 500 epochs for training and 30 for parameter search. These results demonstrate PACE's efficiency and effectiveness across various training configurations.

D.2 Experiments on FGVC with limited training data

To validate generalization benefits of PACE on limited data settings, we conduct experiments on FGVC using 50%, 20%, and 10% of the original training samples. Table 13 shows that PACE achieves larger improvements with smaller data sizes, aligning with our theoretical analyses.

Table 13: Classification results on FGVC using varying percentages of data based on ViT-16/B.

Method	CUB			NAB			Flowers			Stanford Dogs			Stanford Cars		
	50%	20%	10%	50%	20%	10%	50%	20%	10%	50%	20%	10%	50%	20%	10%
baseline	87.1	83.9	79.1	80.7	75.0	70.2	98.5	96.5	93.1	90.6	88.7	86.9	78.7	54.9	30.1
+PACE	88.4	85.5	81.4	82.9	77.5	73.8	99.2	97.9	96.1	91.8	90.9	89.8	80.5	57.3	33.2

D.3 Experiments on self-supervised pre-trained backbones

To further verify the effectiveness of PACE on a self-supervised pre-trained backbone, we conduct VTAB-1K experiments on SVHN, Camelyon, and Clevr-Count using MAE [18] and DINO [18], with ViT-16/B pretrained on ImageNet-1K [11]. Table 14 shows that PACE improves the baseline on these self-supervised backbones, confirming its applicability to fine-tuning self-supervised models.

Table 14: Classification results on VTAB-1K using self-supervised DINO and MAE, with ViT-16/B pretrained on the ImageNet-1K dataset.

Method	MAE			DINO		
	SVHN	Camelyon	Clevr-Count	SVHN	Camelyon	Clevr-Count
Full	90.1	74.6	52.5	89.7	73.1	34.5
Linear	44.5	79.9	57.1	50.7	82.5	44.2
LoRA_{mul}+VPT_{add}	89.3	82.7	82.1	90.0	85.4	55.7
+PACE	**93.5**	**85.8**	**86.4**	**91.7**	**88.1**	**61.0**

D.4 Experiments of Combining PACE with Other PEFT

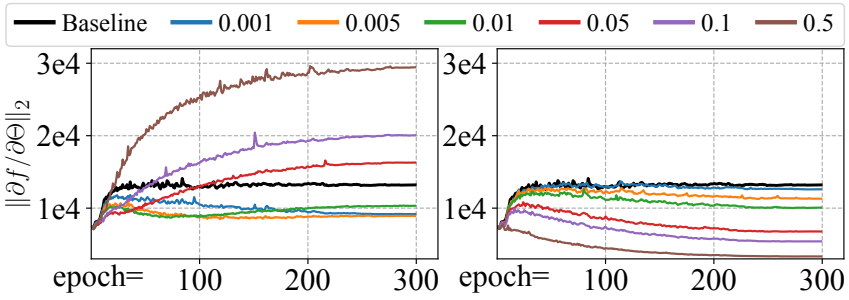
We conducted experiments combining PACE with several PEFT methods, including AdaptFormer [8], GLoRA [7], COFT [53], and BOFT [41], on CIFAR-100 (VTAB-1K) and ImageNet (domain adaptation) using ViT-16/B. Table 15 shows that integrating PACE improves the baseline performance.

Table 15: Classification results of different PEFT methods based on ViT-16/B.

Method	CIFAR-100 (VTAB-1K)	ImageNet (Domain Adaptation)					
		Source	-Sketch	-V2	-A	-R	Avg.
AdaptFormer	70.6	77.4	26.5	67.4	12.4	28.7	42.4
+PACE	**74.8**	**78.2**	**27.4**	**67.9**	**13.9**	**31.7**	**43.8**
GLoRA	75.9	78.2	30.3	68.1	13.5	31.6	44.3
+PACE	**78.6**	**78.8**	**31.7**	**69.0**	**15.9**	**34.4**	**45.9**
COFT	71.8	76.9	26.4	66.7	13.1	30.7	42.7
+PACE	**75.3**	**77.8**	**27.9**	**68.2**	**14.9**	**32.9**	**44.3**
BOFT	72.3	77.1	27.0	66.8	12.8	31.1	42.9
+PACE	**75.7**	**77.9**	**28.3**	**68.2**	**14.7**	**33.4**	**44.5**

E Additional Plots

Figures 8 and 9 show the gradient issues in FPA and the gradient regularization effects of PACE.



(a) FPA (b) PACE

Figure 8: Gradient norms of (a) FPA and (b) PACE with different regularization strengths λ during training on CIFAR-100 (VTAB-1K) w/ ViT-B/16. Figure 5 illustrates the average gradient norm over training epochs.

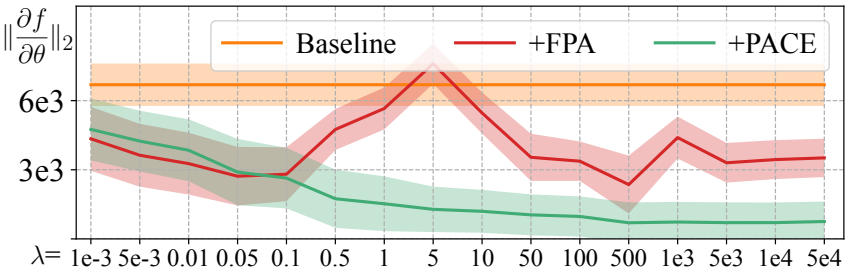


Figure 9: Gradient norms of models across wide range of regularization strengths λ on Camelyon (VTAB-1K) w/ Swin-B. Line and shadow represent mean and std over training epochs. While gradient explosion is less frequent for FPA in this setting, it exhibits unpredictable gradient norm with varied regularization strengths. In contrast, PACE reliably lowers gradient norms as regularization strength λ increases, demonstrating its robustness for effective gradient control.

F Hyperparameter settings

For each dataset, we follow strategies from previous works [40, 27, 7, 44] to apply grid search on the rank, learning rate and weight decay to establish strong baselines. Table 16, 17, 18 and 19 present the hyperparameters and number of trainable parameters used in our strong baseline for VTAB-1K, few-shot learning, FGVC and domain adaptation tasks.

With these strong baselines, we apply grid search on $\lambda \in \{0.02, 0.05, 0.1, 0.2, 0.5, 1\}$ and $\sigma \in \{0.1, 0.5, 1, 1.5, 2\}$ for PACE to optimize its performance.

Table 16: Hyperparameters for baseline on VTAB-1K with ViT-B/16. A: LoRA_{mul}+VPT_{add}, B: LoRA_{add}. lr: learning rate. WD: weight decay.

Hyperparameter	Natural							Specialized				Structured								Average parameter (M)
	Cifar100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	NsORB-Ele	
Method	A	A	A	A	A	A	A	A	A	A	B	B	B	A	A	A	A	A	B	1.81
Rank	10	14	12	18	18	14	10	8	8	10	2	2	8	18	4	10	10	22	4	
lr	1e-3	1e-3	1e-3	1e-3	1e-3	1e-2	1e-3	5e-3	5e-3	5e-3	5e-4	5e-4	1e-4	5e-3	5e-3	5e-3	5e-3	1e-2	2e-4	
WD	1e-4	1e-4	1e-3	1e-2	1e-3	1e-3	1e-2	1e-2	1e-2	1e-2	1e-4	1e-3	1e-4	1e-3	1e-3	1e-4	1e-2	1e-2	1e-2	

Table 17: Ranks for baselines in Few-shot learning. Weight decay is fixed at 1e-4.

learning rate / Baseline	FGVCAircraft	Food101	Flowers102	OxfordPets	StanfordCars	Mean Parameter (M)
	5e-3	5e-3	5e-3	2e-3	2e-3	
LoRA_{add}	4	4	4	4	10	0.93
VPT_{add}	1	1	1	1	1	0.14
LoRA_{mul}+VPT_{add}	14	10	18	18	24	2.70

Table 18: Hyperparameters for the baseline LoRA_{mul}+VPT_{add} in FGVC.

Hyperparameter	CUB-200-2011	NABirds	OxfordFlowers	StanfordDogs	StanfordCars	Mean Parameter (M)
learning rate	5e-3	5e-4	5e-3	5e-3	2e-4	2.80
weight decay	1e-2	1e-3	1e-3	1e-2	1e-3	
rank	14	18	18	24	14	

Table 19: Hyperparameters for baseline LoRA_{mul}+VPT_{add} in domain adaptation.

Baseline	rank	learning rate	weight decay	Parameter (M)
LoRA_{mul}+VPT_{add}	10	5e-4	1e-2	2.39

G Experiment details for GSM-8K

We conduct experiments on text generation tasks by fine-tuning Phi-3-mini-4k-instruct [1] on the GSM-8K [9] dataset using causal language modeling. We use learning rate of 2e-6, batch size of 4, LoRA rank of 16, prompt "Answer below question. First think step-by-step and then answer the final number:\n\n<Question>" as instruction and fine-tune models on the training set and evaluated the performance on the test set.

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: We theoretically and empirically verify the claims and contributions made in the abstract and introduction.

   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: The limitations of our work are discussed in §A

   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: Complete proofs for each theorem are provided in §B.

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: Training details and hyperparameter selection are presented in Sec. 4 and §F, respectively.

   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 will release our code.

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: Experimental settings and details are presented in Sec. 4.

   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: All reported results are averaged over three random seeds.

   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: All experiments were conducted on a single NVIDIA H100 GPU with 96 GB memory, with each experiment completing within 8 hours.

   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: We have carefully reviewed and adhered to the code of ethics throughout our research and writing process.

   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: [Yes]

    Justification: Potential impacts are discussed in §A.

    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: All publicly available assets (models, code, and data) used in this work have been properly credited, and their respective licenses and terms of use have been explicitly mentioned and adhered to.

    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: We do not release new assets in the submission.

    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: 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.
    - 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.