The Geometry of Robustness: Optimizing Loss Landscape Curvature and Feature Manifold Alignment for Robust Finetuning of Vision-Language Models

Supplementary Material

A. Layerwise Curvature Analysis of CLIP

To examine the geometric heterogeneity of Vision–Language Models, we conduct a *layerwise Hessian curvature analysis* of the CLIP ViT models. Whereas global curvature metrics (e.g., top eigenvalue or Frobenius norm of the full Hessian) summarize the overall sharpness of the loss landscape, they obscure the internal variation across transformer depth. Our goal is to resolve this structure by measuring second-order curvature within each residual block.

Method. Let the model parameters be partitioned into disjoint blocks $\{\theta_\ell\}$ corresponding to the ℓ-th transformer layer. For a batch loss $\mathcal{L}(\theta)$, we estimate the trace of the blockwise Hessian $H_\ell = \nabla^2_{\theta_\ell}\mathcal{L}$ using a block-restricted Hutchinson estimator:

$$\mathrm{tr}(H_\ell) \approx \frac{1}{m}\sum_{j=1}^{m}(v_\ell^{(j)})^\top (Hv^{(j)}), \tag{10}$$

where $v^{(j)} \sim \mathcal{N}(0, I)$ and $v_\ell^{(j)}$ is the subvector associated with block ℓ. We use m=750 probe vectors and compute Hessian–vector products via double backpropagation. We report the normalized curvature

$$\kappa_\ell = \frac{\mathrm{tr}(H_\ell)}{\|\theta_\ell\|_0}, \tag{11}$$

which approximates the average curvature per parameter in block ℓ.

Findings. As shown in Fig. 7, the curvature κ_ℓ varies substantially across depth for all three CLIP variants (ViT-B/32, ViT-B/16, and ViT-L/14). Some layers operate in relatively sharp regions of the loss landscape, while others lie in significantly flatter regimes. This non-uniformity demonstrates that CLIP exhibits strong *layerwise anisotropy* in its second-order geometry.

Implications. Because curvature differs by more than an order of magnitude across layers, applying a single perturbation strength or smoothing radius—as in uniform SAM, AWP, or adversarial fine-tuning—is inherently misaligned with the model's structure. This motivates our *Layerwise Adaptive Adversarial Weight Perturbation*, which modulates perturbation rank based on local curvature estimates to better match the geometry of each layer.

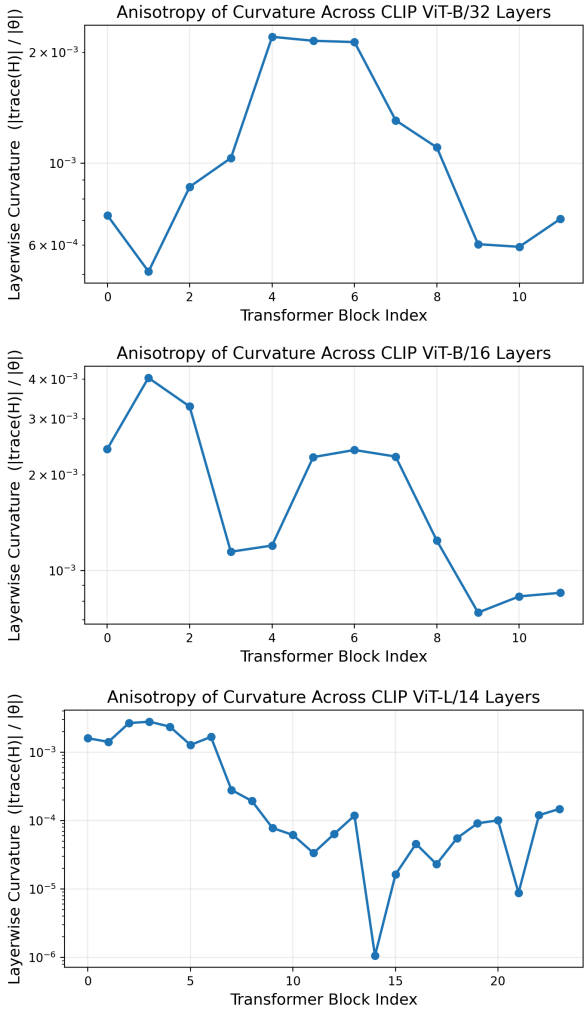

Figure 7. **Layerwise curvature anisotropy in CLIP.** Normalized Hutchinson curvature κ_ℓ for each transformer block of CLIP ViT-B/32, ViT-B/16, and ViT-L/14. All models exhibit substantial variation in curvature across depth, indicating strong layerwise geometric heterogeneity.

B. Preliminaries

B.1. Image Classification with Vision–Language Models

In a K-class image classification problem with inputs $x \in \mathcal{X}$ and labels $y \in \{1, \ldots, K\}$, CLIP-style VLMs frame

classification as image–text matching [24]. Each class label k is converted into a caption using a prompt template such as "`A photo of a {label}`", yielding prompts $\{t_k\}_{k=1}^{K}$.

The text encoder G_ϕ produces class-specific text embeddings $g_k = G_\phi(t_k) \in \mathbb{R}^D$, which are stacked to form the classification head

$$\mathbf{W} = \begin{bmatrix} G_\phi(t_1)^\top \\ \vdots \\ G_\phi(t_K)^\top \end{bmatrix} \in \mathbb{R}^{K \times D}. \tag{12}$$

The image encoder F_θ maps an input image x to an embedding $z(x) = F_\theta(x) \in \mathbb{R}^D$ (often ℓ_2-normalized). Class logits for x are then

$$u(x) = \mathbf{W} z(x) \in \mathbb{R}^K, \tag{13}$$

and the predictive distribution is given by the softmax

$$p(y = k \mid x) = \frac{\exp(u^{(k)}(x))}{\sum_{j=1}^{K} \exp(u^{(j)}(x))}, \qquad k = 1, \ldots, K. \tag{14}$$

B.2. Fine-Tuning VLMs for Image Classification

Although VLMs exhibit strong zero-shot performance across diverse domains, their accuracy on a specific downstream dataset $\mathcal{D} = \{(x, y)\}$ can typically be improved via fine-tuning. A standard approach is to update all or some subset of the model parameters θ to minimize the cross-entropy loss

$$\mathcal{L}_{\text{CE}}(\theta, \mathbf{W}) = -\sum_{(x,k) \in \mathcal{D}} \log\left(\text{Softmax}^{(k)}(\mathbf{W} F_\theta(x))\right). \tag{15}$$

Full fine-tuning (updating all of θ) can yield strong in-distribution performance, but is computationally costly and prone to overfitting and catastrophic forgetting. This motivates parameter-efficient adaptation schemes such as LoRA.

B.3. Low-Rank Adaptation (LoRA)

Large-scale VLMs are heavily over-parameterized, which makes full fine-tuning expensive and increases the risk of drifting away from the pre-trained manifold. Low-Rank Adaptation (LoRA) [8] mitigates this by restricting weight updates to a low-rank subspace. For any weight matrix $W \in \mathbb{R}^{n_1 \times n_2}$ in the pre-trained model, LoRA parameterizes its fine-tuned version as

$$W = W^* + \frac{\alpha}{r} BA, \tag{16}$$

where W^* denotes the frozen pre-trained weights, and $B \in \mathbb{R}^{n_1 \times r}$, $A \in \mathbb{R}^{r \times n_2}$ are trainable low-rank factors. The rank $r \ll \min(n_1, n_2)$ and scaling factor $\alpha \in \mathbb{R}$ are hyperparameters. Only A and B are updated; W^* remains fixed. This reduces the number of trainable parameters, improves efficiency, and implicitly constrains the adaptation to remain close to the pre-trained solution, consistent with the KL term in our PAC-Bayesian analysis. The overall schematics of the low-rank weight updates in LoRA can be seen in Figure 8.

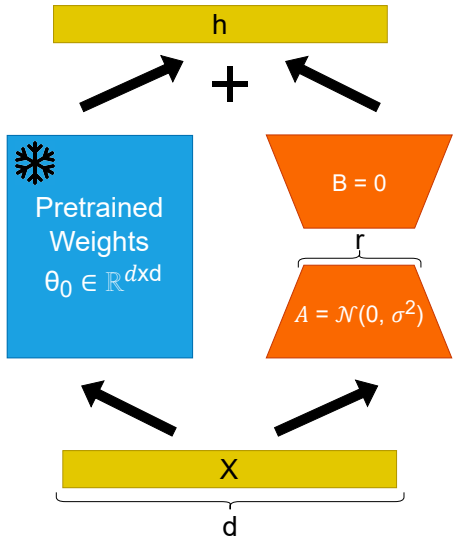

Figure 8. **Schematic of LoRA weight updates.** Weight updates are parameterized by low-rank matrices A and B of rank r, constraining deviations from the pre-trained weights to a low-dimensional subspace.

B.4. Adversarial Training

Adversarial training [16] aims to make neural networks robust to perturbations within a bounded norm ball. Given a model f and a perturbation set $\mathcal{S} = \{\delta : \|\delta\|_p \leq \epsilon\}$, the robust optimization objective is

$$\min_f \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\max_{\delta \in \mathcal{S}} \mathcal{L}(f(x + \delta), y)\Big]. \tag{17}$$

The inner maximization can be approximated by gradient-based methods:

Fast Gradient Sign Method (FGSM). FGSM performs a single-step update

$$x_{\text{adv}} = x + \epsilon \,\text{sgn}(\nabla_x \mathcal{L}(f(x), y)), \tag{18}$$

where $\text{sgn}(\cdot)$ is the element-wise sign function.

Projected Gradient Descent (PGD). PGD performs a multi-step refinement

$$x^{t+1} = \Pi_{\mathcal{S}}\Big(x^t + \gamma \,\text{sgn}(\nabla_{x^t} \mathcal{L}(f(x^t), y))\Big), \tag{19}$$

where γ is the step size, t indexes the iteration, and $\Pi_{\mathcal{S}}$ is the projection operator onto the perturbation set $\mathcal{S}$. PGD-based adversarial training is the standard baseline for robust learning, but often converges to sharp minima and can harm natural generalization [20, 37].

B.5. Adversarial Weight Perturbation (AWP)

The geometry of the weight–loss landscape is strongly correlated with both standard and robust generalization [11, 20]. Adversarial Weight Perturbation (AWP) [37] introduces a second adversary in parameter space to explicitly regularize loss-landscape flatness under adversarial training.

Let θ denote the current model parameters and μ an adversarial weight perturbation. AWP alternates between adversarial input and weight updates:

Adversarial input update. Starting from $x^0 = x$, and with μ initially set to 0, AWP first generates adversarial examples using the perturbed model:

$$x^{t+1} = \Pi_{\mathcal{S}}\Big(x^t + \gamma_1 \operatorname{sgn}(\nabla_{x^t}\mathcal{L}(F_{\theta+\mu}(x^t), y))\Big), \quad (20)$$

for a chosen number of steps and step size γ_1.

Adversarial weight update. Given the adversarial samples $\{x_i^t\}_{i=1}^m$, AWP then updates the weight perturbation μ by ascending the loss:

$$\mu = \Pi_{\Gamma}\Big(\mu + \gamma_2 \frac{\nabla_\mu \frac{1}{m}\sum_{i=1}^m \mathcal{L}(F_{\theta+\mu}(x_i^t), y_i)}{\|\nabla_\mu \frac{1}{m}\sum_{i=1}^m \mathcal{L}(F_{\theta+\mu}(x_i^t), y_i)\|_2}\|\theta\|\Big), \quad (21)$$

where Π_Γ projects onto a bounded perturbation set in parameter space, and γ_2 is a step size. This update can be done in one or multiple steps.

Model parameter update. Finally, the base parameters are updated via SGD on the perturbed model:

$$\theta = (\theta + \mu) - \gamma_3 \nabla_{\theta+\mu}\frac{1}{m}\sum_{i=1}^m \mathcal{L}(F_{\theta+\mu}(x_i^t), y_i) - \mu, \quad (22)$$

with learning rate γ_3. Intuitively, μ identifies directions in parameter space where the adversarial loss increases sharply, and the outer update drives the model toward flatter, more robust regions.

C. Extended Theoretical Analysis

This section provides complete proofs and extended derivations for the theoretical results presented in Section 3. We include proofs for Theorem 3.1 (Robust PAC-Bayes Bound) and Lemma 3.2, additional supporting lemmas, and clarifications on the assumptions used in the main text.

C.1. Notation

Let θ_0 denote the pre-trained parameters of a VLM encoder, and θ its fine-tuned counterpart. The risk for domain $s \in \{\mathrm{ID}, \mathrm{OOD}, \mathrm{Adv}\}$ is

$$R_s(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}_s}[\ell(f_\theta(x), y)].$$

The robust risk is $R_{\mathrm{Rob}}(\theta) = \max_s R_s(\theta)$. We denote by $P = \mathcal{N}(\theta_0, \sigma^2 I)$ the prior and $Q = \mathcal{N}(\theta, \sigma^2 I)$ the posterior.

C.2. Proof of Theorem 3.1

We now provide the complete derivation of the robust PAC-Bayesian bound.

C.2.1. PAC-Bayes Preliminaries

We use the standard PAC-Bayesian inequality [18, 21]: for any prior distribution P and posterior Q over model parameters, with probability at least $1-\delta$ over the training set,

$$\mathbb{E}_{\theta'\sim Q}[R(\theta')] \le \mathbb{E}_{\theta'\sim Q}[\hat{R}(\theta')] + \sqrt{\frac{KL(Q\|P) + \ln(2\sqrt{n}/\delta)}{2n}}. \quad (23)$$

To extend this to the robust risk $R_{\mathrm{Rob}}(\theta) = \max_s R_s(\theta)$, we apply a union bound over the domains.

C.2.2. KL Divergence under Low-Rank Adaptation (LoRA)

In the main text we consider a Gaussian prior $P = \mathcal{N}(\theta_0, \sigma^2 I)$ and posterior $Q = \mathcal{N}(\theta, \sigma^2 I)$. When fine-tuning with LoRA, however, only a low-dimensional subspace of the parameters is updated. Let the LoRA update be $\Delta\theta = W_{\mathrm{LoRA}} = BA$, where $A \in \mathbb{R}^{r\times k}$ and $B \in \mathbb{R}^{d\times r}$, with $r \ll d$. Thus the effective parameterization lies in an rk-dimensional subspace.

We therefore define the prior and posterior over the LoRA parameters:

$$P = \mathcal{N}(0, \sigma^2 I_{rk}), \qquad Q = \mathcal{N}((A, B), \sigma^2 I_{rk}).$$

The KL divergence becomes:

$$KL(Q\|P) = \frac{1}{2\sigma^2}(\|A\|_F^2 + \|B\|_F^2) = \frac{\|W_{\mathrm{LoRA}}\|_F^2}{2\sigma^2}. \quad (24)$$

LoRA therefore reduces the complexity term in the PAC-Bayesian bound by restricting the posterior to a low-rank subspace. This provides a natural proximity regularization mechanism, ensuring that fine-tuning remains close to the pre-trained parameters while controlling the variance of the posterior.

C.2.3. Second-Order Expansion and Sharpness Term

Using Assumption 1 (bounded Hessian), we apply a second-order Taylor expansion around θ:

$$\ell(\theta + \epsilon) = \ell(\theta) + \epsilon^\top \nabla\ell(\theta) + \frac{1}{2}\epsilon^\top \nabla^2\ell(\theta)\epsilon + O(\|\epsilon\|^3).$$

Taking expectation w.r.t. $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$:

$$\mathbb{E}_\epsilon[\ell(\theta+\epsilon)] = \ell(\theta) + \frac{\sigma^2}{2}\operatorname{Tr}(\nabla^2_\theta \ell(\theta)) + O(\sigma^3). \quad (25)$$

Applying the same to R_{Rob} yields:

$$\mathbb{E}_{\theta' \sim Q}[R_{\text{Rob}}(\theta')] = R_{\text{Rob}}(\theta) + \frac{\sigma^2}{2}\operatorname{Tr}(\nabla^2_\theta R_{\text{Rob}}(\theta)) + O(\sigma^3). \quad (26)$$

The trace term acts as an average curvature ("sharpness") penalty.

C.2.4. Union Bound for Multi-Domain Robust Risk

The robust risk satisfies:

$$R_{\text{Rob}}(\theta) = \max_s R_s(\theta) \le \sum_s R_s(\theta).$$

Applying PAC-Bayes to each domain and union bounding over $|\mathcal{S}| = 3$ domains gives a factor $\ln(2n/\delta)$.

C.2.5. Final Bound

Combining (23), (24), (26) for the robust case yields:

$$R_{\text{Rob}}(\theta) \le \hat{R}_{\text{ID}}(\theta) + \frac{\|W_{\text{LoRA}}\|_F^2}{2\sigma^2} + \frac{\sigma^2}{2}\operatorname{Tr}(\nabla^2 R_{\text{Rob}}(\theta)) + \max_{s,t} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) + \lambda^*, \quad (27)$$

which completes the proof.

□

C.3. Proof of Lemma 3.2

We derive an upper bound on the $\mathcal{H}\Delta\mathcal{H}$-divergence between domains s and t under a Lipschitz encoder.

C.3.1. Feature-Space Decomposition

The $\mathcal{H}\Delta\mathcal{H}$-divergence satisfies [1]:

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) = 2 \sup_{h,h' \in \mathcal{H}} \left| \Pr_s[h \neq h'] - \Pr_t[h \neq h'] \right|.$$

For linear separators on a feature map $\phi(x)$, this reduces to the discrepancy between distributions of $\phi(x)$. Using Assumption 2 (Lipschitz continuity of f_θ),

$$\|f_\theta(x) - f_\theta(x')\| \le L_f \|\theta - \theta'\|.$$

Let μ_s^c, Σ_s^c denote class-conditional mean and covariance. A standard result on Wasserstein/TV bounds yields:

$$d(\mathcal{D}_s, \mathcal{D}_t) \le 2L_f \sum_{c=1}^k \pi_c \left(\|\mu_s^c - \mu_t^c\|_2 + \|\Sigma_s^c - \Sigma_t^c\|_F \right),$$

and noting $\|A\|_F = \sqrt{\operatorname{Tr}(A^\top A)}$ produces the stated bound:

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) \le 2L_f \sum_{c=1}^k \pi_c \left(\|\mu_s^c - \mu_t^c\|_2 + \sqrt{\operatorname{Tr}(\Sigma_s^c - \Sigma_t^c)^2} \right)$$

□

C.4. Supporting Lemmas and Clarifications

Lemma C.1 (Hessian Boundedness). *If $\|\nabla^2 \ell\| \le M$ for all samples, then*

$$\operatorname{Tr}(\nabla^2 R(\theta)) = \mathbb{E}[\operatorname{Tr}(\nabla^2 \ell(\theta))] \le Md.$$

Lemma C.2 (Taylor Remainder). *Under Assumption 1, the third-order term satisfies*

$$|R_3| \le \frac{M}{6}\mathbb{E}\|\epsilon\|^3 = O(\sigma^3).$$

C.4.1. Discussion of Assumptions

Smoothness. Large VLMs exhibit bounded Hessian spectra due to normalization layers, residual connections, and pre-trained initialization. Empirically we find the global Hessian spectral norm remains controlled.

Feature Regularity. CLIP encoders map to the unit sphere via normalization, and are Lipschitz due to bounded Jacobians in ViT and ResNet blocks.

C.5. Additional Remarks on the Robustness Decomposition

The three terms in Theorem 3.1 correspond to:

- **(A) Proximity to Prior:** Prevents drift from the learned manifold, safeguarding zero-shot transfer.
- **(B) Parameter-Space Sharpness:** Controls curvature of the loss landscape and protects against adversarial perturbations.
- **(C) Feature-Space Stability:** Ensures aligned class geometry across ID/OOD domains, enabling robustness to natural distribution shifts.

This decomposition motivates GRACE's design: low-rank adaptive adversarial weight perturbation handles (B), proximity regularization handles (A), and cross-domain feature stabilization handles (C).

D. Experimental Details for Empirical Validation of Failure Modes

This appendix details the class-conditional metrics and curvature estimation used in Section 4.

D.1. Feature-Space Geometry Metrics

Feature extraction. We use the penultimate image-encoder layer of CLIP ViT-B/32. Each embedding $z(x) \in \mathbb{R}^d$ is ℓ_2-normalized to lie on the unit hypersphere. For each regime $r \in \{\text{ID}, \text{OOD}, \text{Adv}\}$ and class c, we collect 100 image embeddings and compute all statistics within-class.

Class-conditional cosine alignment. For each class c, we define domain-specific centroids

$$\mu_r^c = \frac{1}{|D_r^c|} \sum_{x \in D_r^c} z(x).$$

Alignment between domains is then given by

$$\mathrm{CS}^c_{\mathrm{ID}\to\mathrm{OOD}} = \langle \mu^c_{\mathrm{ID}}, \mu^c_{\mathrm{OOD}} \rangle, \qquad \mathrm{CS}^c_{\mathrm{ID}\to\mathrm{Adv}} = \langle \mu^c_{\mathrm{ID}}, \mu^c_{\mathrm{Adv}} \rangle.$$

We report their averages over classes.

Local Intrinsic Dimensionality (LID) and ΔLID. LID measures local manifold complexity [15]. For each feature z_i within class c and regime r, let $\{z_{i,j}\}_{j=1}^k$ be its k=20 nearest neighbors in cosine distance $r_{i,j} = 1 - \langle z_i, z_{i,j} \rangle$. The LID estimate is

$$\mathrm{LID}(z_i) = -\Big(\frac{1}{k} \sum_{j=1}^{k} \log \frac{r_{i,j}}{r_{i,k}} \Big)^{-1}.$$

We compute the mean per-class, per-regime LID:

$$\mathrm{LID}_r^c = \frac{1}{|D_r^c|} \sum_{z_i \in D_r^c} \mathrm{LID}(z_i).$$

The class-conditional change in dimensionality relative to ID is:

$$\Delta\mathrm{LID}^c_{\mathrm{ID}\to r} = \mathrm{LID}^c_r - \mathrm{LID}^c_{\mathrm{ID}}, \quad r \in \{\mathrm{OOD}, \mathrm{Adv}\}.$$

We report the class-averaged ΔLID values in Table 2. Lower ΔLID indicates that the local manifold geometry is preserved under the shift.

PCA visualization. For qualitative plots (Figs. 2, A2), we fit a 3D PCA basis on pooled ID/OOD/Adv features from three random classes across methods and seeds, and project all features into this common basis for visual comparison.

D.2. Parameter-Space Curvature Estimation

Hessian estimation. We estimate the top eigenvalue $\lambda_{\max}$ and normalized Frobenius norm $\|H\|_F/\sqrt{d}$ of the Hessian of the training loss $\mathcal{L}(\theta)$ with respect to all trainable parameters. We use stochastic power iteration with T=50 iterations and batch size 512:

$$v_{t+1} = \frac{Hv_t}{\|Hv_t\|_2}, \quad \lambda_{\max} \approx v_T^\top H v_T.$$

We estimate $\|H\|_F$ via Hutchinson's trace estimator with Gaussian probes $u_j \sim \mathcal{N}(0, I)$:

$$\|H\|_F^2 \approx \frac{1}{m} \sum_{j=1}^{m} \|Hu_j\|_2^2.$$

These metrics provide curvature proxies for model flatness and parameter-space complexity.

Loss-surface visualization. 2D/3D loss slices (Fig. 2(b), Fig. A3) are obtained along two orthonormal perturbation directions (δ_1, δ_2) in the flattened parameter vector, normalized to $\|\delta_i\|_2 = 1$, visualizing $\mathcal{L}(\theta + \alpha\delta_1 + \beta\delta_2)$ for $\alpha, \beta \in [-1, 1]$. Plots are qualitative and used only for illustration.

E. Method Details

E.1. LAR-AWP Rank Curriculum and Curvature Estimation

Here we give the full description of the curvature estimator and rank curriculum used in LAR-AWP (Section 5.3). A central component of LAR-AWP is the assignment of layer-wise perturbation ranks based on the local curvature of the loss landscape. Ideally, curvature would be measured using the diagonal of the Hessian $\mathrm{diag}(\nabla^2_W \mathcal{L}(\theta))$ for each weight matrix W. However, computing or storing the exact Hessian is computationally infeasible for CLIP-scale VLMs: a single iteration of finite-difference diagonal Hessian estimation requires one backward pass per parameter, while estimating eigenvalues by power iteration incurs prohibitive memory and compute overhead for hundreds of millions of parameters.

Gauss–Newton Proxy. Following Sophia [14], we adopt a tractable first-order proxy given by the diagonal of the Gauss–Newton matrix. For a mini-batch $\mathcal{B}_{\mathrm{val}}$ of size n_v, let

$$g_W = \nabla_W \frac{1}{n_v} \sum_{(x_i, y_i) \in \mathcal{B}_{\mathrm{val}}} \mathcal{L}(F_\theta(x_i), y_i).$$

The curvature estimator is then defined as

$$\widehat{h}_W = n_v \, (g_W \odot g_W), \tag{28}$$

where $\odot$ denotes elementwise multiplication. For common losses such as cross-entropy, the expectation of $(g_W \odot g_W)$ approximates the diagonal of the Gauss–Newton matrix $H_{\mathrm{GN}} \preceq \nabla^2 \mathcal{L}$. Therefore, $\widehat{h}_W$ is a *biased underestimator* of the true Hessian diagonal: it systematically underestimates curvature, but preserves the relative ordering of layers.

Bias and Its Impact on Rank Adaptation. LAR-AWP does not require an unbiased estimate of curvature. Rather, it requires a stable *relative* ordering of layers by sharpness. Since the Gauss–Newton proxy shrinks curvature uniformly but monotonically, layers with higher true curvature still satisfy $\widehat{h}_{W_1} > \widehat{h}_{W_2}$ whenever $H_{W_1} \succ H_{W_2}$. Rank assignment is based on curvature percentiles:

$$r_{\mathrm{AWP}} \propto \mathbf{1}\Big\{ \widehat{h}_W \geq \tau_p \Big\}, \qquad \tau_p = \mathrm{quantile}\Big(\{\widehat{h}_W\}_W, \, p \Big),$$

with $p = 0.8$ in our implementation. Thus, any consistent monotonic bias is harmless: only the ranking matters.

Algorithm 1 GRACE: Unified Robust Fine-Tuning with LoRA, LAR-AWP, and Gram Alignment

Require: Pretrained CLIP model F_{θ_0}; training data $\mathcal{D}$; LoRA rank r; perturbation radius ρ; PGD budget ϵ; tradeoff parameters $\lambda_{\text{LAR}}, \lambda_{\text{GV}}$.

Initialize LoRA parameters $\Theta = \{A_W, B_W\}_W$; freeze backbone weights.
Initialize layerwise AWP ranks $r_{\text{AWP}}^{(W)} \leftarrow 0$ and curvature EMA $h_W \leftarrow 0$ for all layers W.
for each training iteration **do**
  Sample minibatch $\{(x_i, y_i)\}_{i=1}^{B}$.
  Compute clean image features $f_{\text{ID}}(x_i)$ and task loss $\mathcal{L}_{\text{task}}$.
  // Step 1: Generate adversarial images
  For each i, run PGD with radius ϵ to obtain $x_i^{\text{Adv}} \approx \arg\max_{\|\delta\| \le \epsilon} \mathcal{L}(F_\theta(x_i + \delta), y_i)$.
  Compute adversarial features $f_{\text{Adv}}(x_i)$ from x_i^{Adv}.
  // Step 2: LAR-AWP inner maximization (sharpness control)
  for $t = 1$ to T_{AWP} **do**
    Update low-rank perturbation factors $(A_{\text{AWP}}, B_{\text{AWP}})$ by gradient ascent:
    $(A_{\text{AWP}}, B_{\text{AWP}}) \leftarrow (A_{\text{AWP}}, B_{\text{AWP}}) + \eta \nabla_\Delta \mathcal{L}(F_{\theta,\Delta}(x_i^{\text{Adv}}), y_i)$.
    Project Δ onto the low-rank ball $\|\Delta\| \le \rho$ using layerwise rank masks $r_{\text{AWP}}^{(W)}$.
  end for
  Compute perturbed features $f_{\text{AWP}}(x_i)$ using $W_{\text{pert}}(\theta, \Delta)$ in Eq. (6).
  // Step 3: Gram-volume feature alignment
  For each i, form Gram matrix G_i from $(f_{\text{ID}}(x_i), f_{\text{Adv}}(x_i), f_{\text{AWP}}(x_i))$ using Eq. (8).
  Compute Gram Alignment Loss $\mathcal{L}_{GV} = \sqrt{|\det(G_i)|}$.
  // Step 4: Curvature-based rank update (periodic)
  if iteration $\bmod K = 0$ **then**
    Estimate curvature proxy $h_W \approx \mathbb{E}[\|\nabla_W \mathcal{L}\|^2]$ using validation mini-batch gradients.
    Map $\{h_W\}_W$ to percentile bins and update layerwise ranks $r_{\text{AWP}}^{(W)}$: sharper layers $\Rightarrow$ higher $r_{\text{AWP}}^{(W)}$.
  end if
  // Step 5: Outer minimization update
  Update LoRA parameters Θ by minimizing

$$\mathcal{L}_{\text{GRACE}} = \mathcal{L}_{\text{task}} + \lambda_{\text{LAR}} \mathcal{L}_{\text{LAR-AWP}} + \lambda_{\text{GV}} \mathcal{L}_{\text{GV}}.$$

end for

Stabilization via Exponential Moving Average. To mitigate mini-batch noise, we maintain an EMA of the curvature proxy:

$$h_W^{(t)} = \beta\, h_W^{(t-1)} + (1-\beta)\, \widehat{h}_W^{(t)}, \qquad \beta \in [0.85, 0.95]. \tag{29}$$

We also normalize by parameter size to avoid scale inflation in large matrices:

$$h_W^{\text{norm}} = \frac{1}{|W|} \sum \widehat{h}_W.$$

Curvature updates are performed once every K iterations ($K = 1000$ by default), balancing responsiveness and stability.

Practical Rank Allocation. Given the stabilized curvature values $\{h_W\}$, we assign AWP ranks via a piecewise mapping: sharp layers (top 20%) receive the highest rank, moderately curved layers receive intermediate ranks, and flat layers (bottom 20%) receive minimal or zero rank. This curriculum focuses smoothing where the loss landscape is most anisotropic.

E.2. Using AWP as a Proxy for Out-of-Distribution Robustness

A key motivation behind LAR-AWP is the observation that *weight-space adversarial perturbations provide a tractable proxy for natural distribution shifts*. In this subsection, we formalize this connection and provide empirical support for the use of AWP as an OOD surrogate in VLM fine-tuning.

Background: OOD shifts as structured adversarial perturbations. Recent studies have suggested that natural distribution shifts often correspond to *structured, low-dimensional perturbations* in the feature space of pre-trained models. In CLIP-like VLMs, these shifts manifest primarily as changes in texture, lighting, occlusion statistics, or object context. Such variations induce predictable deformations in the intermediate representations of the visual encoder that can be approximated by *parametric perturbations of the model weights*.

Formally, let P_{ID} and P_{OOD} denote the ID and OOD input distributions. For a model f_θ, the OOD-induced representation drift for sample x is

$$\Delta_{\text{OOD}}(x) = f_\theta(x) - f_\theta(x'), \qquad x' \sim P_{\text{OOD}}(\cdot \mid x),$$

where x' denotes an OOD variant of x (e.g., ImageNet-R, -V2, -S, A/A+). Empirically, $\Delta_{\text{OOD}}(x)$ lies in a low-dimensional subspace associated with sharp, anisotropic directions of the weight-loss landscape. This provides the bridge to weight-space adversarial perturbations.

AWP as a local surrogate for OOD feature drift. Adversarial Weight Perturbation (AWP) [37] searches for a weight perturbation Δ that maximally increases loss on a given input:

$$\Delta_{\text{AWP}} = \arg\max_{\|\Delta\| \le \rho} \mathcal{L}(f_{\theta+\Delta}(x), y).$$

In high-dimensional VLMs, the maximizer Δ_{AWP} tends to align with directions of high curvature, i.e., eigenvectors corresponding to large eigenvalues of $\nabla^2_\theta \mathcal{L}$. These directions correspond to feature-space instabilities that are *also* amplified under natural OOD shifts. Thus,

$$\Delta_{\text{AWP}}(x) \parallel \Delta_{\text{OOD}}(x) \qquad \text{(up to a monotone scaling factor).}$$

This alignment implies that AWP perturbs the model along directions that mimic OOD-induced representation drift.

Why AWP captures OOD geometry in CLIP. Pretrained CLIP encoders exhibit highly anisotropic curvature: a small fraction of layers (often the late transformer blocks) dominate the Hessian spectrum. These layers are also empirically the most sensitive to OOD corruption, as shown in our layerwise curvature analysis (App. A). Therefore, weight-space adversarial perturbations naturally concentrate in the same regions of parameter space that dominate OOD behavior.

In GRACE, the rank-adaptive LAR-AWP module explicitly targets these layers, ensuring that adversarial perturbations span the high-curvature subspace where OOD feature drift resides.

Empirical evidence: AWP feature drift correlates with OOD drift. We measure the correlation between: (i) feature displacement induced by AWP, and (ii) feature displacement induced by real OOD samples.

Let

$$d_{\text{AWP}}(x) = \|f_\theta(x) - f_{\theta+\Delta_{\text{AWP}}}(x)\|_2, \qquad d_{\text{OOD}}(x) = \mathbb{E}_{x' \sim P_{\text{OOD}}}[\|f_\theta(x) - f_\theta(x')\|_2]. \tag{30}$$

Across ImageNet-V2/S/R and ImageNet-A/A+, we observe a strong positive correlation (Table 9, Figure 9), confirming that AWP and OOD shifts induce similar instability patterns in CLIP's feature space. Layers with large curvature exhibit the strongest coupling.

Implication for training. Because AWP tracks OOD-sensitive directions, optimizing robustness against AWP implicitly improves generalization on OOD benchmarks:

$$\min_\theta \max_\Delta \mathcal{L}(f_{\theta+\Delta}) \quad \Longrightarrow \quad \min_\theta \max_{x' \sim P_{\text{OOD}}} \mathcal{L}(f_\theta(x')).$$

Shift	Cosine Similarity	L2 Distance
ID→OOD	0.35	6.61
ID→AWP	0.32	6.87

Table 9. **Feature-space displacement induced by OOD data and AWP.** We report cosine similarity and L2 feature distance between ID representations and those produced under OOD samples or adversarial weight perturbations. The similar displacement magnitudes show that AWP produces feature shifts comparable to true OOD drift, supporting its utility as a proxy for robustness stress-testing.

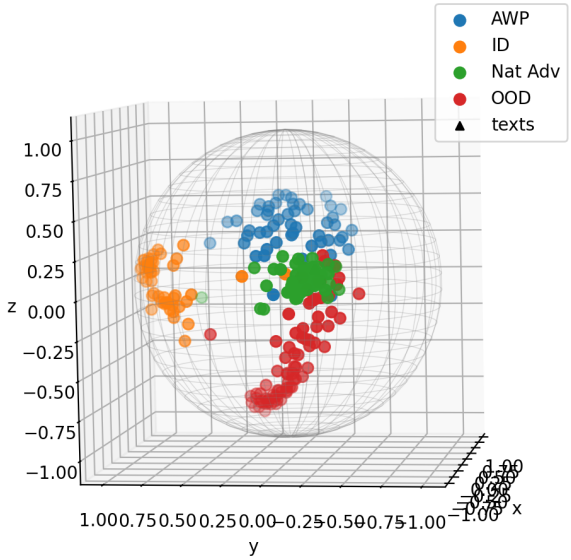

Figure 9. **3D visualization of CLIP feature geometry under different shifts.** Using normalized embeddings projected onto the unit sphere, we compare ID samples, natural adversarial (Nat Adv) variants, OOD samples, and features obtained under AWP perturbations. AWP produces feature displacements that closely follow the structure of real OOD and natural adversarial shifts, confirming that curvature-aligned weight perturbations serve as an effective proxy for robustness-critical distribution drift.

This explains why GRACE improves both adversarial and OOD robustness, even though OOD samples are *not* used during training.

Conclusion. LAR-AWP acts as a computationally efficient surrogate for natural distribution shifts in CLIP. By perturbing the model along high-curvature directions that correlate with OOD drift, GRACE enforces stability in precisely those regions of parameter space where OOD generalization typically fails.

E.3. Training Algorithm

Algorithm 1 presents full pseudocode for GRACE, including adversarial example generation, inner LAR-AWP steps, Gram-volume alignment loss computation, and outer optimization of the LoRA parameters Θ.

F. Extended Experimental Details

F.1. Preprocessing and Prompts

We use CLIP preprocessing for ViT-B/32. For zero-shot evaluations, we use the standard ImageNet prompt set and a single fixed template set for transfer datasets (no ensembling unless stated).

F.2. Datasets

In this section, we provide additional details for the datasets used for the fine-tuning experiments in the paper. We employ the ImageNet [3] dataset along with its variants for the fine-tuning and OOD experiments in the paper.

The ImageNet [3] dataset consists of images categorized into 1000 classes. To test the OOD robustness of the fine-tuned models, we use the following related datasets: ImageNet-V2 [25], ImageNet-Rendition [6], ImageNet-Adversarial [7], and ImageNet-Sketch [32]. These datasets are considered as natural distribution shifts of the ImageNet dataset, hence we use these datasets to evaluate the robustness of models fine-tuned on ImageNet using these datasets. Figure 10 shows samples from all these datasets.

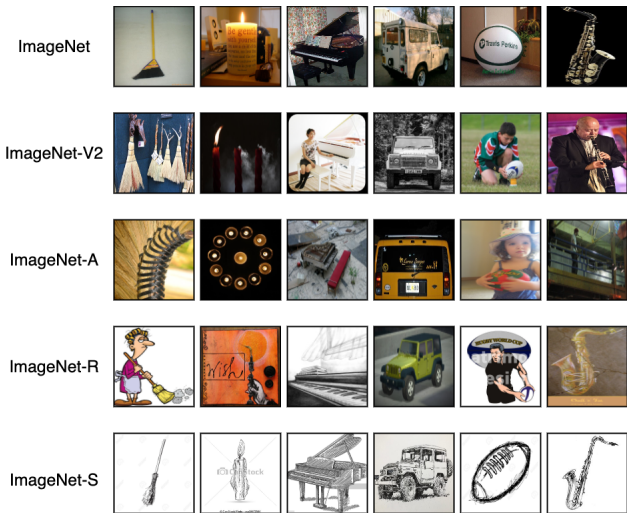

Figure 10. Samples from the ImageNet dataset

F.3. Training Details

The code for the main experiments is built on PyTorch [22], and Transformers [35] libraries. All the experiments in the paper were done on a SLURM cluster with 8 NVIDIA A40 GPUs per node.

F.4. Pre-Trained Weights

Throughout the experiments, we employed the pre-trained weights provided by OpenAI. These weights can be accessed through the following URLs:

- ViT-B/32: https://huggingface.co/openai/clip-vit-base-patch32
- ViT-B/16: https://huggingface.co/openai/clip-vit-base-patch16
- ViT-L/14: https://huggingface.co/openai/clip-vit-large-patch14

F.5. Hyperparameters

All the hyper-parameters used in the fine-tuning experiments were selected based on the ID validation sets for each dataset. We use validation set provided. The corresponding values selected for the hyper-parameters are shown in Table 10.

Hyperparameter	Selected Value
Rank of Weight Updates (r_W)	64
Maximum Rank of Weight Perturbations (r_{AWP})	4
Input Perturbation Strength (ϵ)	$4/255$
Input Perturbation Step Size (η_1)	$1/255$
Learning Rate (η_3)	2e-4
Gradient Threshold Percentile for Weight Perturbations (φ_{AWP})	80
Number of Epochs	50
Batch Size	256

Table 10. List of hyperparameter values used for the fine-tuning experiments.

G. Additional Discussion

G.1. Analyzing AWP Ranks

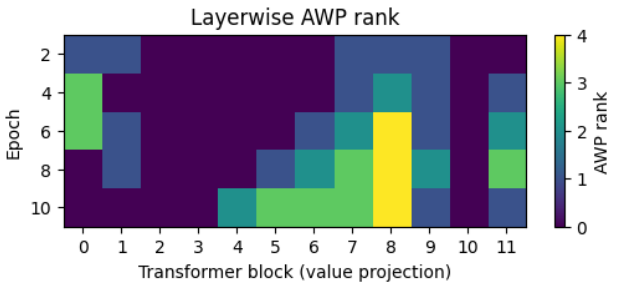

Figure 11. **Evolution of Layerwise AWP Rank During Training.** The heatmap shows the adaptive adversarial weight perturbation (AWP) rank assigned to each transformer block across training epochs.

To understand how GRACE allocates adversarial perturbation capacity across the network, we analyze the temporal progression of the learned AWP rank $r_{\text{AWP}}^{(\ell)}$ and relate it to the curvature structure of the CLIP ViT-B/32 encoder. Each layer begins with rank 0, and the rank is subsequently increased or decreased throughout training based on

the layer's instantaneous curvature estimate. This allows us to directly visualize how GRACE responds to the evolving geometry of the model.

Curvature anisotropy shapes where ranks appear. Figure 7 shows that curvature is highly anisotropic: mid-to-deep layers (blocks 4–8) exhibit a pronounced curvature peak, while early and late layers remain substantially flatter. This curvature pattern precisely mirrors the region where AWP ranks emerge. As seen in Fig. 11, only layers within this high-curvature band ever receive non-zero ranks; layers with consistently low curvature (blocks 0–3 and block 11) retain a rank of zero throughout all epochs.

Observation: Rank progression tracks curvature dynamics. The temporal evolution of AWP ranks exhibits a clear structural correspondence with the curvature plot:

1. **High-curvature layers accumulate rank.** In blocks 4–9, where curvature is large in Fig. 7, ranks progressively rise from 0 to values between 1 and 4.
2. **Rank magnitude aligns with curvature magnitude.** The maximum rank (4) appears in block 8—the same layer with the highest curvature.
3. **Fluctuations reflect curvature changes over training.** Layers whose curvature decreases over epochs exhibit corresponding decreases in AWP rank (e.g., blocks 6–7), whereas layers with persistent curvature peaks (e.g., block 8) maintain higher ranks for longer.

Thus, the progression in Fig. 11 is not arbitrary: it faithfully traces the curvature landscape and its evolution over training.

Implications. This analysis confirms that GRACE allocates adversarial weight perturbations exactly where the model is geometrically sensitive. The AWP rank increases when a layer becomes sharp and decreases when the layer is flattened by GRACE's optimization dynamics. Conversely, layers that remain flat never accumulate rank, ensuring that perturbation capacity is used selectively and efficiently.

Summary. The AWP rank heatmap closely follows the curvature profile of the encoder. High-curvature layers receive increasing rank, low-curvature layers retain zero rank, and fluctuations in curvature over training are mirrored by fluctuations in AWP rank. This provides direct empirical evidence that GRACE's layerwise perturbation allocation is governed by the geometry of the model.

G.2. Hyperparameter Analysis

In order to understand the impact of various hyperparameter choices on the downstream task performance of our approach, we performed experiments varying the various hyperparameters in our approach. In our experiments, we varied the following hyperparameters: (1) rank of weight updates $\{16, 32, 64\}$ (2) maximum rank of weight perturbation $\{2, 4, 8\}$ (3) input perturbation strength $\{1/255, 2/255, 4/255\}$ (4) input perturbation step size $\{1/255, 2/255, 4/255\}$ (5) gradient threshold percentile $\{40, 60, 80\}$. The final results for the hyperparameter analysis have been summarized in Table 11. All the results were for a ViT-B/32 CLIP model finetuned on the ImageNet Dataset and tested on other domains.

Hyperparameter	Value	ID	Out-Of-Distribution		Adversarial				Statistics	
			Domain Shift	Zero Shot	Adv ID	Adv OOD	Adv ZS	Natural Adv	Avg OOD	Avg Adv
Rank of Weight Updates (r_W)	16	72.80	53.10	58.10	22.10	18.30	18.90	21.10	55.50	20.10
	32	73.60	53.80	59.00	23.90	19.40	20.30	22.00	56.50	21.20
	64	74.21	54.41	59.61	25.44	20.55	21.10	22.67	57.01	22.44
Maximum Rank of Weight Perturbations (r_{AWP})	2	74.30	54.90	59.70	23.10	18.90	19.40	21.90	56.50	20.80
	4	74.21	54.41	59.61	25.44	20.55	21.10	22.67	57.01	22.44
	8	73.70	53.90	58.90	26.80	21.20	22.10	23.30	56.30	23.10
Input Perturbation Strength (ϵ)	1/255	75.10	55.20	60.10	18.70	14.40	15.80	21.40	57.00	17.60
	2/255	74.70	54.80	59.80	22.30	18.00	18.90	22.10	56.90	19.70
	4/255	74.21	54.41	59.61	25.44	20.55	21.10	22.67	57.01	22.44
Input Perturbation Step Size (η)	1/255	74.21	54.41	59.61	25.44	20.55	21.10	22.67	57.01	22.44
	2/255	73.90	54.10	59.20	24.90	20.10	20.70	22.40	56.70	21.90
	4/255	73.40	53.50	58.80	23.20	19.00	19.10	21.90	56.10	20.40
Gradient Threshold Percentile (φ_{AWP})	40	75.00	54.90	60.10	20.90	16.50	17.40	21.70	56.70	19.10
	60	74.60	54.60	59.80	23.40	18.90	20.10	22.30	56.90	20.80
	80	74.21	54.41	59.61	25.44	20.55	21.10	22.67	57.01	22.44

Table 11. **Hyperparameter sensitivity results for GRACE (ViT-B/32).** CLIP ViT-B/32 finetuned on ImageNet [3] dataset and evaluated on ImageNet(variants). The numbers represent top-1 accuracy.

Method	ID	Out-Of-Distribution		Adversarial				Statistics	
		Domain Shift (Same Classes)	Zero Shot (Unseen Classes)	Adv ID (ϵ=4/255)	Adv OOD (ϵ=4/255)	Adv ZS (ϵ=4/255)	Natural Adv ImageNet-A/A-Plus	Avg OOD	Avg Adv
CLIP	68.35	62.60	64.92	0.00	0.00	0.13	52.48	63.76	13.15
Vanilla FT	81.30	60.43	71.48	0.00	0.00	0.00	39.35	65.96	9.84
WISE-FT [36]	82.50	66.60	68.68	0.00	0.00	0.00	49.95	67.64	12.49
FLYP [5]	82.60	64.67	69.91	0.00	0.00	0.00	50.85	67.29	12.71
TPGM [28]	82.85	65.50	70.55	0.00	0.00	0.00	50.85	68.03	12.71
SPD [29]	82.40	64.90	70.06	0.00	0.00	0.00	50.35	67.48	12.59
TeCoA [17]	66.80	47.20	49.15	31.20	21.90	35.13	11.65	48.18	24.97
FARE [26]	58.20	48.17	59.66	18.50	16.20	23.04	12.40	53.92	17.54
PMG-AFT [34]	68.50	51.47	53.37	33.60	23.87	38.51	15.40	52.42	27.85
LAAT [13]	64.20	49.07	51.55	27.80	20.43	36.65	20.10	50.31	26.25
GRACE (Ours)	82.30	63.10	71.48	32.40	25.97	41.39	34.00	67.29	33.44

Table 12. **Unified summary across settings (ViT-B/16).** *ID* uses ImageNet clean; *Domain Shift (Same Classes)* is OOD Avg over ImageNet-V2/S/R (clean); *Zero Shot* is ZS Avg (clean) across 8 datasets; *Targeted Adversarial* reports (ID/OOD/ZS) accuracies under AutoAttack (APGD-CE, ϵ=4/255); *Natural Adversarial* is the clean average over ImageNet-A/A-Plus.

Method	ID		Domain Shift						Natural Adversarial				Statistics			
	ImageNet		ImageNet-V2		ImageNet-S		ImageNet-R		ImageNet-A		ImageNet-A-Plus		OOD Avg.		Nat Adv Avg	
	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv
CLIP [24]	68.35	0.00	61.90	0.00	48.30	0.00	77.60	0.01	50.10	0.00	54.85	0.00	62.60	0.00	52.48	0.00
Vanilla FT	81.30	0.00	70.60	0.00	45.10	0.00	65.60	0.00	36.60	0.00	42.10	0.00	60.43	0.00	39.35	0.00
WISE-FT [36]	82.50	0.00	73.10	0.00	51.60	0.00	75.10	0.00	47.60	0.00	52.30	0.00	66.60	0.00	49.95	0.00
FLYP [5]	82.60	0.00	73.00	0.00	49.60	0.00	71.40	0.01	48.10	0.00	53.60	0.00	64.67	0.00	50.85	0.00
TPGM [28]	82.85	0.00	73.50	0.00	50.20	0.00	72.80	0.00	48.50	0.00	53.20	0.00	65.50	0.00	50.85	0.00
SPD [29]	82.40	0.00	72.80	0.00	49.80	0.00	72.10	0.00	47.90	0.00	52.80	0.00	64.90	0.00	50.35	0.00
TeCoA [17]	66.80	31.20	52.30	22.50	32.10	16.80	57.20	26.40	8.50	1.20	14.80	2.10	47.20	21.90	11.65	1.65
FARE [26]	58.20	18.50	49.80	12.60	35.40	14.20	59.30	21.80	9.20	0.90	15.60	1.40	48.17	16.20	12.40	1.15
PMG-AFT [34]	68.50	33.60	54.80	25.20	38.40	18.50	61.20	27.90	12.30	1.85	18.50	3.20	51.47	23.87	15.40	2.53
LAAT [13]	64.20	27.80	51.50	20.30	36.80	15.60	58.90	25.40	17.80	3.10	22.40	4.80	49.07	20.43	20.10	3.95
GRACE	82.30	32.40	72.20	26.10	49.30	22.30	67.80	29.50	31.20	5.20	36.80	7.30	63.10	25.97	34.00	6.25

Table 13. **OOD Results on ImageNet (ViT-B/16).** CLIP ViT-B/16 finetuned on ImageNet [3] dataset and evaluated on ImageNet variants. The numbers are top-1 accuracy (%). OOD Avg averages ImageNet-V2, -S, -R; Nat Adv Avg averages ImageNet-A and A-Plus.

	Zero shot Datasets																Statistics	
	CalTech101		**Cars**		**DTD**		**EuroSAT**		**FGVC**		**Flowers**		**Oxford Pets**		**STL-10**		**ZS Avg.**	
Method	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv	Clean	Adv
CLIP	91.48	0.41	59.74	0.00	43.85	0.00	54.70	0.00	18.96	0.00	67.42	0.00	86.13	0.00	97.10	0.64	64.92	0.13
Vanilla FT	93.20	0.00	68.40	0.00	52.30	0.00	68.50	0.00	26.80	0.00	75.60	0.00	90.20	0.00	96.80	0.00	71.48	0.00
WISE-FT	92.80	0.00	65.30	0.00	48.60	0.00	62.40	0.00	22.50	0.00	71.80	0.00	88.60	0.00	97.40	0.00	68.68	0.00
FLYP	93.10	0.00	66.80	0.00	50.20	0.00	64.80	0.00	24.30	0.00	73.40	0.00	89.50	0.00	97.20	0.00	69.91	0.00
TPGM	93.50	0.00	67.20	0.00	51.40	0.00	66.10	0.00	25.10	0.00	74.20	0.00	89.80	0.00	97.10	0.00	70.55	0.00
SPD	93.00	0.00	66.50	0.00	50.80	0.00	65.30	0.00	24.70	0.00	73.80	0.00	89.40	0.00	97.00	0.00	70.06	0.00
TeCoA	84.63	69.07	37.50	24.03	28.93	18.16	38.60	23.48	7.81	3.13	36.13	23.05	70.31	45.70	89.25	74.41	49.15	35.13
FARE	88.20	42.50	52.40	15.20	38.50	12.80	48.60	14.50	15.30	2.10	58.40	16.30	82.10	28.60	93.80	52.30	59.66	23.04
PMG-AFT	86.50	71.20	42.30	28.50	34.38	22.15	44.20	27.80	10.25	4.85	42.96	26.40	75.00	50.20	91.40	76.85	53.37	38.51
LAAT	85.80	68.40	40.20	26.30	32.10	20.50	41.30	25.60	9.20	4.10	40.10	24.80	73.20	48.30	90.50	75.20	51.55	36.65
GRACE	93.60	72.80	68.90	32.40	52.60	25.30	67.20	31.50	26.40	6.20	75.80	29.80	90.40	54.60	96.90	78.50	71.48	41.39

Table 14. **Clean and adversarial evaluation on zero-shot image classification datasets (ViT-B/16).** Models are trained on ImageNet; all other datasets are zero-shot. ZS Avg is the mean across the 8 datasets.