Robustness of Vision Foundation Models to Common Perturbations

Hongbin Liu[1], Zhengyuan Jiang[1], Cheng Hong[2], Neil Zhenqiang Gong[1]
[1]Duke University, [2]Ant Group
{hongbin.liu, zhengyuan.jiang, neil.gong}@duke.edu
vince.hc@antgroup.com

Abstract

A vision foundation model outputs an embedding vector for an image, which can be affected by common editing operations (e.g., JPEG compression, brightness, contrast adjustments). These common perturbations *alter embedding vectors and may impact the performance of downstream tasks using these embeddings. In this work, we present the* first *systematic study on foundation models' robustness to such perturbations. We propose three robustness metrics and formulate five desired mathematical properties for these metrics, analyzing which properties they satisfy or violate. Using these metrics, we evaluate six industry-scale foundation models (OpenAI, Meta) across nine common perturbation categories, finding them generally non-robust. We also show that common perturbations degrade downstream application performance (e.g., classification accuracy) and that robustness values can predict performance impacts. Finally, we propose a fine-tuning approach to improve robustness without sacrificing utility.*

1. Introduction

A vision foundation model is a general-purpose feature extractor that outputs an embedding vector for an image. Typically, these models are pre-trained on vast collections of unlabeled images or image-text pairs in a self-supervised manner [18, 21] by major providers like OpenAI, Meta, and Google. For instance, OpenAI's CLIP [21] jointly trains vision and language foundation models on 400 million image-text pairs, while Meta's DINO v2 [18] trains a vision foundation model on a large set of unlabeled images. Foundation models empower various downstream applications like image classification and depth estimation.

Images in real-world settings often undergo common editing operations for various purposes. For instance, JPEG compression is widely used to reduce communication costs online, while brightness and contrast adjustments are also common (Table 8 in the Appendix lists 9 editing operations used in our experiments). These operations introduce *common perturbations* to an image, unlike *adversarial perturbations* [4, 23], which are worst-case modifications designed to mislead models. Common perturbations, by contrast, occur frequently in non-adversarial, real-world scenarios.

The robustness of foundation models and their downstream applications to adversarial perturbations has been widely studied [7, 11–14, 16, 20, 22]. However, robustness to common perturbations remains largely unexplored. Specifically, three key questions arise regarding robustness to common perturbations: 1) How robust are foundation models, i.e., how much does an embedding vector change when an image undergoes common perturbations? 2) How robust are downstream applications, i.e., to what extent does classifier accuracy degrade with perturbed images? 3) How can we improve robustness in foundation models and their downstream applications against common perturbations?

A key challenge in answering these questions is designing a metric to quantify a foundation model's robustness to common perturbations. Such a metric would enable systematic robustness assessments, facilitating comparisons across self-supervised learning algorithms, model architectures, and sizes. Additionally, a robustness metric could help predict the performance of downstream applications (e.g., accuracy) for perturbed images and provide guidance to enhance foundation model robustness.

Our work: In this work, we answer the three questions above via performing the *first* systematic study on the robustness of foundation models to common perturbations.

We begin by tackling the challenge of defining metrics to quantify a foundation model's robustness to common perturbations. Given an image, common perturbations produce various perturbed versions, each with its own embedding vector. We quantify robustness by measuring the variations among these embedding vectors, exploring three metrics: one based on *cosine similarity*, another on *Euclidean distance*, and a third, DivergenceRadius, which uses the radius of the smallest enclosing ball in the embedding space to capture robustness.

A suitable robustness metric should meet several intuitions, such as not increasing robustness as more perturba-

tions are applied. We formalize these intuitions with five *mathematical properties* and analyze which properties the metrics satisfy. We find that DivergenceRadius satisfies all five properties, whereas the other metrics fail to meet one; additionally, the Euclidean and cosine similarity metrics are equivalent.

Using our robustness metrics, we address the first question through a systematic study of six industry-scale foundation models from the CLIP (OpenAI) and DINO v2 (Meta) families, covering different self-supervised learning algorithms, architectures, and sizes, across nine categories of common perturbations. Our findings are consistent across the three robustness metrics: foundation models generally lack robustness to common perturbations, often producing divergent embeddings for perturbed images. Additionally, we observe that foundation models based on Vision Transformer architectures are more robust than those based on ResNet architectures.

To address the second question, we evaluate the robustness of downstream classifiers and depth estimation models built on industry-scale foundation models against common perturbations. We observe that these perturbations degrade both classifier accuracy and depth estimation performance; for example, glass-blurring reduces the accuracy of a zero-shot ImageNet classifier by 9.4%. This occurs due to variations in embedding vectors caused by perturbations. Additionally, we find that average classification accuracy and mean squared error of depth maps for perturbed images are roughly linear functions of the image's robustness value (e.g., cosine similarity or DivergenceRadius), enabling accurate performance predictions for downstream tasks using a simple linear regression model.

Finally, we propose a fine-tuning method to enhance a foundation model's robustness while preserving utility for downstream tasks. Our approach aims to balance two objectives: a *robustness goal* and a *utility goal*, each quantified by a corresponding loss term. We fine-tune the model by minimizing a weighted sum of these loss terms, with empirical results showing that our method successfully improves robustness without compromising utility.

2. Problem Formulation

Perturbation function: We represent common perturbations with a perturbation function $P(x, k)$, where x is an image and k a perturbation parameter, yielding a perturbed image $P(x, k)$. For instance, if P represents JPEG compression, k is the quality factor controlling compression level. For some functions, k is multi-dimensional, such as *fog blurring*, where k includes *density* and *frequency*.

We denote the domain of k as $\mathbb{K}$, the set from which k is selected when applying P to x. The domain $\mathbb{K}$ may include discrete values (e.g., JPEG quality factors) or continuous values (e.g., Gaussian noise standard deviation). We assume a special parameter $\perp \in \mathbb{K}$, where $P(x, \perp) = x$, returning the original image, to simplify descriptions.

Embedding vector: A foundation model f outputs an embedding vector $f(x)$ for an image x. To prevent embedding magnitude from affecting downstream applications, foundation models often normalize embeddings to an ℓ_2-norm of 1, ensuring $||f(x)||_2 = 1$ for any image x. Thus, all embedding vectors lie on a unit-radius hyper-sphere in the embedding space.

Desired mathematical properties of a robustness metric: Given a foundation model f, an image x, and a perturbation function P with parameter domain $\mathbb{K}$, our goal is to define a robustness metric $\mathcal{R}(f, x, P, \mathbb{K})$ to quantify the robustness of f for x under P. This metric essentially measures variations among embedding vectors $\{f(P(x, k))\}_{k\in\mathbb{K}}$ for perturbed versions of x generated by P.

The robustness metric should allow quantitative comparisons of different foundation models' robustness to common perturbations. Since foundation models support downstream applications, the metric should also predict downstream performance for x under P. For instance, if the downstream task is classification, the robustness value $\mathcal{R}(f, x, P, \mathbb{K})$ should help predict the accuracy for perturbed versions of x within the domain $\mathbb{K}$.

We have the following mathematical properties:

1. **Bounded domain:** To ensure interpretability and comparability, we design a scalar robustness metric with a bounded interval output, normalized to $[0, 1]$ for simplicity. Thus, for any f, x, P, and $\mathbb{K}$, the robustness value $\mathcal{R}(f, x, P, \mathbb{K})$ should fall within $[0, 1]$, where a larger value indicates *less robustness*:

$$\mathcal{R}(f, x, P, \mathbb{K}) \in [0, 1], \forall f, x, P, \mathbb{K}. \tag{1}$$

2. **Monotonicity:** When the parameter domain $\mathbb{K}$ expands, the model should not become more robust under this larger domain. For instance, greater variation in JPEG quality factors should not decrease $\mathcal{R}(f, x, P, \mathbb{K})$. Formally:

$$\mathcal{R}(f, x, P, \mathbb{K}_1) \leq \mathcal{R}(f, x, P, \mathbb{K}_2), \forall \mathbb{K}_1 \subseteq \mathbb{K}_2. \tag{2}$$

3. **Best robustness:** The model is maximally robust for x under P if all perturbed versions of x have the same embedding as x, resulting in $\mathcal{R}(f, x, P, \mathbb{K}) = 0$ if:

$$f(P(x, k)) = f(x), \forall k \in \mathbb{K}. \tag{3}$$

4. **Worst robustness:** If the embedding vectors for perturbed images are uniformly distributed in the embedding space (sum to zero), then robustness is at its lowest with $\mathcal{R}(f, x, P, \mathbb{K}) = 1$, if $\exists \mathbb{K}' \subseteq \mathbb{K}$:

$$\sum_{k\in\mathbb{K}'} f(P(x, k)) = \mathbf{0}. \tag{4}$$

Table 1. Three robustness metrics explored in this work and the desired mathematical properties they violate.

Robustness Metric	Formulation	Violating Properties
Cosine similarity	$\mathcal{R}_{cs}(f,x,P,\mathbb{K}) = \frac{1-\min_{k_1,k_2\in\mathbb{K}} cos(f(P(x,k_1)),f(P(x,k_2)))}{2}$	Worst-robustness
Euclidean distance	$\mathcal{R}_{ed}(f,x,P,\mathbb{K}) = \frac{\max_{k_1,k_2\in\mathbb{K}} \|\|f(P(x,k_1))-f(P(x,k_2))\|\|_2}{2}$	Worst-robustness
DivergenceRadius	$\mathcal{R}_{dr}(f,x,P,\mathbb{K}) = \text{argmin}\, r$ s.t. $\exists c, \|\|f(P(x,k))-c\|\|_2 \leq r, \forall k \in \mathbb{K}$	None

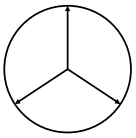


Figure 1. An example for the worst-robustness property.

where $\mathbf{0}$ is the zero vector. Figure 1 illustrates this in a two-dimensional space.

5. **Rotational invariance:** Since embeddings are rotation-free, the robustness metric should be invariant to rotations in the embedding space. Let M be a rotation matrix, then:

$$\mathcal{R}(M \cdot f, x, P, \mathbb{K}) = \mathcal{R}(f, x, P, \mathbb{K}), \tag{5}$$

where $M \cdot f$ denotes rotating $f(x)$ by M.

3. Robustness Metrics

We explore three robustness metrics and theoretically analyze what desired mathematical properties they satisfy/violate, which we summarize in Table 1.

3.1. Cosine Similarity

Since embedding vectors lie on a hypersphere, we can use the angles between them to quantify robustness. Specifically, the largest angle (smallest cosine similarity) between any two embedding vectors from perturbed images can define a robustness metric. Formally, for a model f, image x, perturbation function P, and parameter domain $\mathbb{K}$, the cosine similarity-based robustness metric $\mathcal{R}_{cs}$ is:

$$\begin{aligned}&\mathcal{R}_{cs}(f,x,P,\mathbb{K})\\ =&\frac{1-\min_{k_1,k_2\in\mathbb{K}} cos(f(P(x,k_1)),f(P(x,k_2)))}{2},\end{aligned} \tag{6}$$

where $cos(\cdot,\cdot)$ is the cosine similarity. The constants normalize the value to [0,1].

We can verify that the robustness metric $\mathcal{R}_{cs}$ satisfies the five mathematical properties except the worst-robustness one. In particular, since $cos(\cdot,\cdot)$ is in [-1,1], $\mathcal{R}_{cs}(f,x,P,\mathbb{K})$ lies in [0, 1], meeting the bounded-domain property. As $\mathbb{K}$ expands, $\min_{k_1,k_2\in\mathbb{K}} cos(f(P(x,k_1)),f(P(x,k_2)))$ does not increase, so $\mathcal{R}_{cs}$ does not decrease, satisfying monotonicity. The best-robustness property holds because $\mathcal{R}_{cs}(f,x,P,\mathbb{K}) = 0$ when all perturbed versions of x have the same embedding. Finally, $\mathcal{R}_{cs}$ is rotation-invariant since cosine similarity is.

However, $\mathcal{R}_{cs}$ does not satisfy the worst-robustness property. For instance, in Figure 1, $\mathcal{R}_{cs}$ is 0.75, not 1, as $\mathcal{R}_{cs} = 1$ only when embedding vectors cover exactly half the hypersphere, with cosine similarity of -1. If embedding vectors are more evenly distributed, the smallest cosine similarity is greater than -1, violating worst-robustness.

Monotonically increasing function of cosine similarity: A robustness metric based on any monotonically increasing function of $\mathcal{R}_{cs}$ cannot satisfy all five desired properties. Formally, we define such a metric:

$$\mathcal{R}_g(f,x,P,\mathbb{K}) = g(\mathcal{R}_{cs}(f,x,P,\mathbb{K})), \tag{7}$$

where g is a monotonically increasing function satisfying $g(0) = 0$ and $g(1) = 1$. We can show $\mathcal{R}_g$ satisfies bounded-domain, best-robustness, and rotation-invariance properties but fails the worst-robustness property. Formally, we have:

Theorem 1 (Monotonically Increasing Function of $\mathcal{R}_{cs}$)**.** *Given any g that is a monotonically increasing function of $\mathcal{R}_{cs}$ and satisfies $g(0) = 0$ and $g(1) = 1$, $\mathcal{R}_g = g(\mathcal{R}_{cs})$ does not satisfy the worst-robustness property.*

Proof. To prove this, we construct a counter-example. Since $\mathcal{R}_g = g(\mathcal{R}_{cs})$ increases with $\mathcal{R}_{cs}$, and since $g(0) = 0$ and $g(1) = 1$, we have $\mathcal{R}_g < 1$ if $\mathcal{R}_{cs} < 1$. In Figure 1, $\mathcal{R}_{cs} = 0.75 < 1$, so $\mathcal{R}_g < 1$, countering the worst-robustness property. □

3.2. Euclidean Distance

Another intuitive robustness metric is to use the Euclidean distances between embedding vectors of perturbed images to quantify their variations. Formally, given a model f, image x, perturbation function P, and parameter domain $\mathbb{K}$, we define a Euclidean distance-based robustness metric $\mathcal{R}_{ed}$ as:

$$\begin{aligned}&\mathcal{R}_{ed}(f,x,P,\mathbb{K})\\ =&\frac{\max_{k_1,k_2\in\mathbb{K}} ||f(P(x,k_1)) - f(P(x,k_2))||_2}{2},\end{aligned} \tag{8}$$

where $||\cdot||_2$ denotes the Euclidean distance, with the constant 2 normalizing the metric to [0,1].

We can show that $\mathcal{R}_{ed}$ is a monotonically increasing function of $\mathcal{R}_{cs}$, specifically as follows:

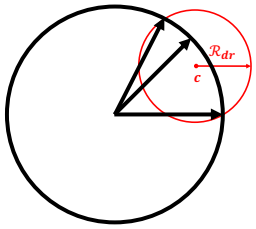


Figure 2. An example to illustrate the minimum enclosing ball. The red circle is the minimum enclosing ball of the three embedding vectors.

Theorem 2. *For any model f, image x, perturbation function P, and parameter domain $\mathbb{K}$, $\mathcal{R}_{ed}$ is the square root of $\mathcal{R}_{cs}$:*

$$\mathcal{R}_{ed}(f, x, P, \mathbb{K}) = \sqrt{\mathcal{R}_{cs}(f, x, P, \mathbb{K})}. \tag{9}$$

Proof. Please refer to the Appendix. □

Theorem 2 further shows that $\mathcal{R}_{cs}$ and $\mathcal{R}_{ed}$ are equivalent, as $\mathcal{R}_{cs}$ can be converted to $\mathcal{R}_{ed}$ by taking the square root. Therefore, we omit results for $\mathcal{R}_{ed}$ in our experiments for simplicity.

3.3. DivergenceRadius

3.3.1. Formulating an Optimization Problem

We first outline the intuition behind DivergenceRadius and then present its formulation.

Intuition: The cosine similarity and Euclidean distance metrics do not satisfy the worst-robustness property, which requires a maximum robustness value of 1 when the embedding vectors for perturbed images are equally distributed in the embedding space. Our intuition is that if the embedding vectors are uniformly spread on the unit hyper-sphere, the radius of the smallest ball enclosing these vectors will equal 1. Thus, constructing a minimum enclosing ball for the perturbed embedding vectors can satisfy the worst-robustness property and indicate robustness: a smaller radius implies greater robustness.

Optimization problem: We aim to find the smallest-radius high-dimensional ball with center c that encloses all embedding vectors of perturbed versions of an image x within the perturbation domain $\mathbb{K}$. This radius r, our DivergenceRadius, is denoted as $\mathcal{R}_{dr}(f, x, P, \mathbb{K})$. Formally:

$$\begin{aligned} &\mathcal{R}_{dr}(f, x, P, \mathbb{K}) = \operatorname{argmin} r, \\ &\text{s.t. } \exists c, ||f(P(x,k)) - c||_2 \leq r, \forall k \in \mathbb{K}. \end{aligned} \tag{10}$$

Figure 2 illustrates an example of a minimum ball enclosing embedding vectors for three perturbed versions of an image x. Note that $\mathbb{K}$ includes a special parameter $\perp$ where $P(x, \perp) = x$.

3.3.2. DivergenceRadius Satisfies the Mathematical Properties

We show that the robustness metric $\mathcal{R}_{dr}$ satisfies all the five mathematical properties.

Proof. Please refer to the Appendix. □

3.3.3. Solving DivergenceRadius

We consider both discrete and continuous $\mathbb{K}$.

Discrete $\mathbb{K}$: For a discrete domain $\mathbb{K}$, the embedding vectors $\{f(P(x,k))\}_{k\in\mathbb{K}}$ are discrete points on the unit hyper-sphere. In this case, we can use Welzl's algorithm [25] to efficiently find the minimum enclosing ball's center and radius, solving Equation 10 in $O(dn)$ time, where d is the embedding dimension and n the number of discrete values in $\mathbb{K}$.

Continuous $\mathbb{K}$: When $\mathbb{K}$ is continuous, finding an exact solution is infeasible due to infinite embedding vectors. To approximate, we sample a discrete subset $\bar{\mathbb{K}}$ from $\mathbb{K}$, transforming the problem into a discrete one. We use two sampling methods: *random sampling* (uniform random selection) and *equally-spaced sampling*, where equally spaced values better represent the domain when using fewer samples. For a domain $\mathbb{K} = [a, b]$ and m samples, equally-spaced sampling yields values $a, a+(b-a)/(m-1), \ldots, b$. Our experiments show that equally-spaced sampling outperforms random sampling for DivergenceRadius estimation accuracy.

After obtaining $\bar{\mathbb{K}}$, we apply Welzl's algorithm to find an approximate DivergenceRadius. For large discrete domains, a small sample set can similarly reduce computational cost. Random and equally-spaced sampling methods also apply to estimating the cosine similarity-based robustness $\mathcal{R}_{cs}$ and Euclidean distance-based robustness $\mathcal{R}_{ed}$.

4. Measuring Robustness of Real-world Foundation Models

In this section, we evaluate the robustness of industry-scale foundation models using cosine similarity and DivergenceRadius. Although cosine similarity does not theoretically satisfy the worst-robustness property, we include it in our experiments since the worst-robustness scenario does not occur in the evaluated perturbations. We omit Euclidean distance results due to its equivalence with cosine similarity.

4.1. Measurement Setup

Foundation models: We evaluate foundation models pre-trained using various algorithms, architectures, and sizes, allowing comparison of pre-training methods and model robustness. Specifically, we evaluate two popular families of vision foundation models: CLIP [21] and DINO v2 [18]. CLIP (by OpenAI) was trained with *multi-modal self-supervised learning* on 400 million image-text pairs, while DINO v2

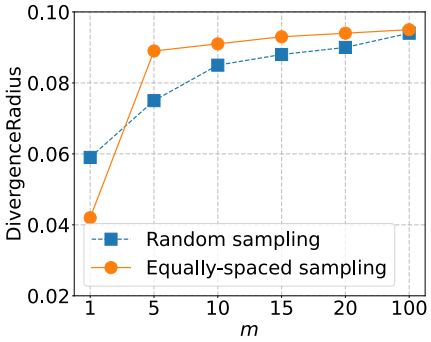


Figure 3. Comparing random sampling and equally-spaced sampling at computing DivergenceRadius, where CLIP ViT-L/14 foundation model, JPEG compression, and ImageNet images are used. m is the number of discrete values sampled from the domain $\mathbb{K}$.

(by Meta) used *self-supervised learning* on 142 million unlabeled images. In the CLIP family, we test ViT-B/16, ViT-L/14, RN50, and RN50×64 (Vision Transformer and ResNet architectures). In the DINO v2 family, we evaluate ViT-L/14 and ViT-g/14. Details on these models are in Table 5 in the Appendix.

Datasets: For robustness evaluation, we use two image classification datasets, ImageNet [6] and Food101 [3], and one depth estimation dataset, NYU-Depth V2 [17]. Details of these datasets are provided in Table 4 (Appendix). Although labels are not required for robustness testing, they are used later to evaluate downstream applications.

Common perturbations: Following [9], we use nine common perturbation functions representing typical image editing operations: *JPEG compression*, *Brightness adjustment*, *Contrast adjustment*, *Defocus blurring*, *Elastic blurring*, *Fog blurring*, *Frost blurring*, *Gaussian noise*, and *Glass blurring*. Each has one variable parameter, with fixed values for additional parameters if present.

Table 8 in the Appendix details each perturbation's parameter domain $\mathbb{K}$, additional fixed parameters, and a visualization of a maximally distorted example for each function. Following prior work [9], the selected $\mathbb{K}$ for each perturbation reflects realistic image editing in real-world scenarios.

4.2. Measurement Results

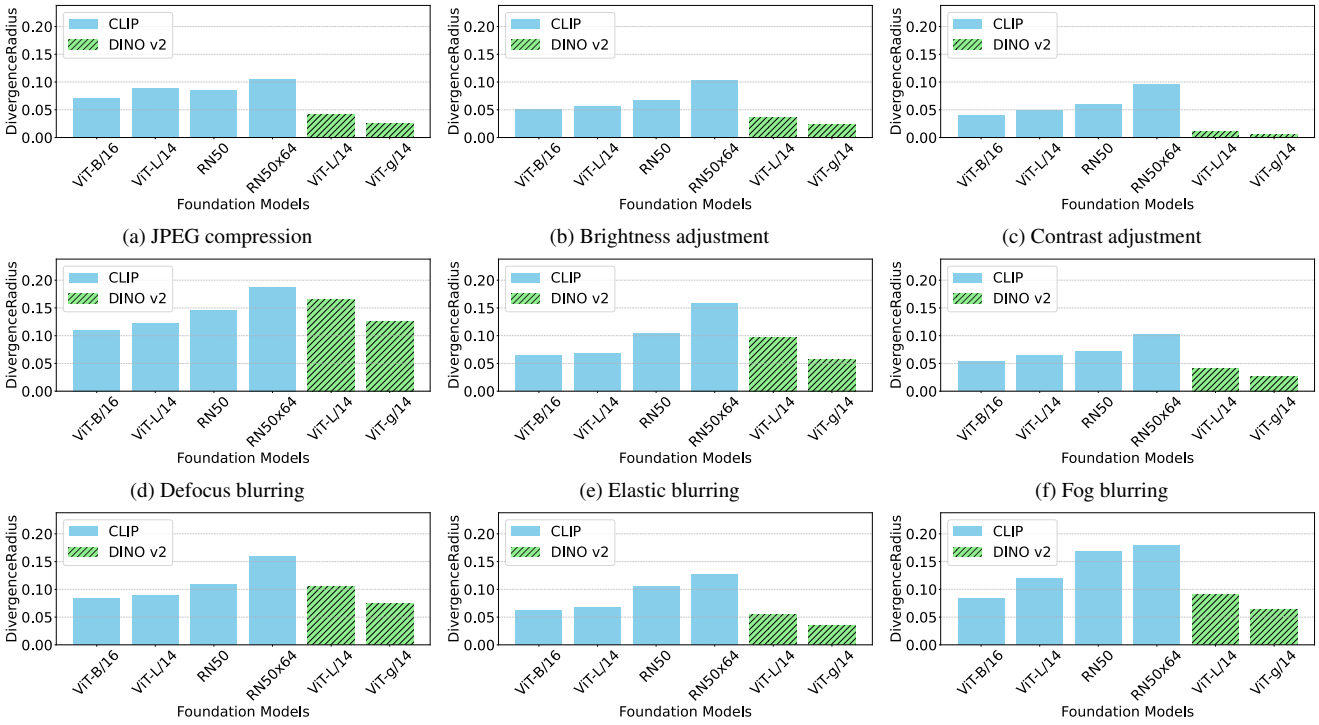
Random sampling v.s. equally-spaced sampling: When computing robustness values (i.e., $\mathcal{R}_{cs}$ or $\mathcal{R}_{dr}$), we can use *random sampling*, which selects m discrete values from $\mathbb{K}$ at random, or *equally-spaced sampling*, which selects m evenly spaced values. We compare these methods by computing $\mathcal{R}_{dr}$ for a foundation model, perturbation function, and image. Figure 3 shows the average $\mathcal{R}_{dr}$ on ImageNet images for CLIP ViT-L/14 under JPEG compression as m varies.

We observe that DivergenceRadius initially increases and then saturates with both sampling methods. As m grows, more diverse parameters from $\mathbb{K}$ yield a higher $\mathcal{R}_{dr}$, approaching the true robustness value. Equally-spaced sampling converges more quickly, reaching near-saturation at $m \geq 5$, whereas random sampling requires $m \geq 20$. This indicates that equally-spaced sampling provides a more efficient approximation of $\mathcal{R}_{dr}$. Thus, for computational efficiency, we use equally-spaced sampling with $m = 5$ for each perturbation function in other experiments.

Comparing model architectures: Figure 4, Figure 13 (Appendix), and Figure 15 (Appendix) show the average $\mathcal{R}_{dr}$ on ImageNet, Food101, and NYU-Depth V2 datasets across different foundation models and perturbations. Figures 12, 14, and 16 (Appendix) present the corresponding $\mathcal{R}_{cs}$ results. To compare architectures, we examine models with the same pre-training algorithm and similar sizes: specifically, CLIP ViT-B/16 vs. CLIP RN50 and CLIP ViT-L/14 vs. CLIP RN50×64. We observe that ViT-B/16 (or ViT-L/14) is consistently more robust than RN50 (or RN50×64) across all perturbations and datasets, as evidenced by lower DivergenceRadius values. This suggests that Vision Transformers are generally more robust to common image perturbations than ResNet architectures. Similar results are also observed when using cosine similarity.

Comparing pre-training algorithms: To compare pre-training algorithms, we evaluate the robustness of CLIP ViT-L/14 and DINO v2 ViT-L/14, which share the same architecture and model size. We find no consistent trend in robustness across perturbation types. For example, DINO v2 ViT-L/14 has lower DivergenceRadius (i.e., greater robustness) than CLIP ViT-L/14 on JPEG compression, Brightness adjustment, Contrast adjustment, Fog blurring, Gaussian noise, and Glass blurring across all datasets. However, DINO v2 ViT-L/14 shows higher DivergenceRadius (i.e., lower robustness) on other perturbations. For instance, under JPEG compression, the average DivergenceRadius for ImageNet images is 0.038 for DINO v2 ViT-L/14 and 0.084 for CLIP ViT-L/14, while under Defocus blurring, it is 0.146 for DINO v2 ViT-L/14 and 0.108 for CLIP ViT-L/14.

Comparing model sizes: For model size comparisons, we examine foundation models within the same architecture and pre-training algorithm: CLIP ViT-B/16 vs. CLIP ViT-L/14, CLIP RN50 vs. CLIP RN50x64, and DINO v2 ViT-L/14 vs. DINO v2 ViT-g/14. In the CLIP family, larger models are generally less robust to common perturbations than smaller ones. For example, ViT-L/14 shows a higher average $\mathcal{R}_{dr}$ (or $\mathcal{R}_{cs}$) than ViT-B/16 across all perturbations and datasets, and RN50 $\times$ 64 has higher robustness metrics than RN50, except for a few cases on Food101 (e.g., JPEG compression, Defocus blurring). Conversely, in the DINO v2 family, larger models are more robust: ViT-g/14 consistently shows lower $\mathcal{R}_{dr}$ values than ViT-L/14 across



(a) JPEG compression (b) Brightness adjustment (c) Contrast adjustment

(d) Defocus blurring (e) Elastic blurring (f) Fog blurring

(g) Frost blurring (h) Gaussian noise (i) Glass blurring

Figure 4. Average DivergenceRadius of ImageNet testing images for different foundation models and perturbation functions.

perturbations and datasets. This contrast suggests that pre-training settings—multi-modal self-supervised learning for CLIP vs. image-only self-supervised learning for DINO v2—impact robustness differently as model size increases.

5. Measuring Performance of Downstream Applications

In this section, we measure the robustness of downstream applications to common perturbations and demonstrate that an image's robustness value (i.e., $\mathcal{R}_{dr}$ or $\mathcal{R}_{cs}$) can predict the performance of downstream applications on its perturbed versions. We omit Euclidean distance results due to its equivalence with cosine similarity.

5.1. Experimental Setup

Downstream applications: Given a pre-trained vision foundation model, we consider the following three popular downstream applications: zero-shot classification, linear-probe classification, and depth estimation. The Appendix shows more details of these applications.

Evaluation metrics (ACC, ACC_p, $RMSE$, and $RMSE_p$): For a downstream classifier $g \circ f$, where f is a foundation model and g a classifier head, *accuracy* (ACC) is the fraction of correctly predicted labels. *Accuracy under perturbation* (ACC_p) is calculated for each perturbed image. For depth estimation heads, we use $RMSE$ and $RMSE_p$ (definitions in the Appendix).

Table 2. ACC and average ACC_p of ImageNet's testing images for two downstream classifiers. Zero-shot classification is based on the CLIP ViT-L/14 foundation model and linear-probe classification is based on the DINO v2 ViT-g/14 foundation model.

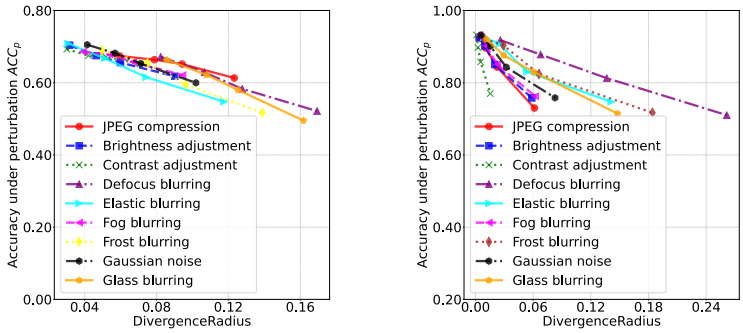
		Zero-shot Classification	Linear-probe Classification
ACC (%)		68.4	86.6
ACC_p (%)	JPEG compression	65.1 (↓ 3.3)	84.6 (↓ 2.0)
	Brightness adjustment	66.3 (↓ 2.1)	85.7 (↓ 0.9)
	Contrast adjustment	67.3 (↓ 1.1)	86.4 (↓ 0.2)
	Defocus blurring	60.1 (↓ 8.3)	82.2 (↓ 4.4)
	Elastic blurring	63.5 (↓ 4.9)	85.0 (↓ 1.6)
	Fog blurring	66.0 (↓ 2.4)	86.1 (↓ 0.5)
	Frost blurring	61.4 (↓ 7.0)	83.6 (↓ 3.0)
	Gaussian noise	66.0 (↓ 2.4)	85.7 (↓ 0.9)
	Glass blurring	59.0 (↓ 9.4)	82.9 (↓ 3.7)

Parameter settings: We evaluate the classifiers with the highest accuracy on zero-shot or linear-probe tasks. For zero-shot classification, we use CLIP ViT-L/14; for linear-probing, we use DINO v2 ViT-g/14, training a one-layer classifier for Food101.

5.2. Experimental Results

Due to space limits, we discuss results for downstream classifiers and defer depth estimation results to the Appendix.

ACC vs. ACC_p: Table 2 shows ACC and average



(a) Zero-shot classification (b) Linear-probe classification

Figure 5. Accuracy under perturbation ACC_p vs. DivergenceRadius of ImageNet testing images for (a) zero-shot classification and (b) linear-probe classification when different perturbation functions are used. Zero-shot classification is based on the CLIP ViT-L/14 foundation model and linear-probe classification is based on the DINO v2 ViT-g/14 foundation model.

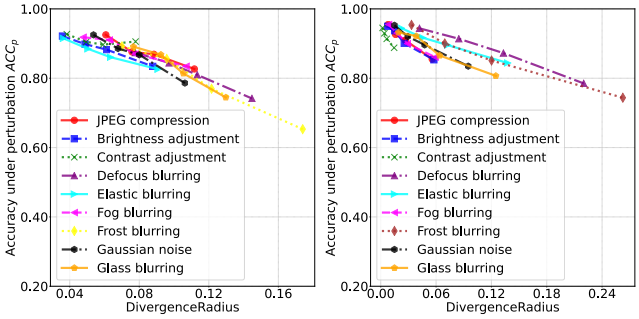
Table 3. ACC and average ACC_p of Food101's testing images for two downstream classifiers. Zero-shot classification is based on the CLIP ViT-L/14 foundation model and linear-probe classification is based on the DINO v2 ViT-g/14 foundation model.

		Zero-shot Classification	Linear-probe Classification
ACC (%)		92.0	94.1
ACC_p (%)	JPEG compression	87.4 (↓ 4.6)	91.2 (↓ 2.9)
	Brightness adjustment	88.4 (↓ 3.6)	90.8 (↓ 3.3)
	Contrast adjustment	90.9 (↓ 1.1)	91.8 (↓ 2.3)
	Defocus blurring	81.7 (↓ 10.3)	87.0 (↓ 7.1)
	Elastic blurring	87.2 (↓ 4.8)	89.6 (↓ 4.5)
	Fog blurring	88.3 (↓ 3.7)	90.9 (↓ 3.2)
	Frost blurring	79.3 (↓ 12.7)	85.0 (↓ 9.1)
	Gaussian noise	86.6 (↓ 5.4)	89.7 (↓ 4.4)
	Glass blurring	82.9 (↓ 9.1)	87.3 (↓ 6.8)

ACC_p under different perturbations for ImageNet; results for Food101 are in Table 3. Common perturbations reduce ACC_p compared to ACC, indicating degraded accuracy. For example, Glass blurring reduces ImageNet zero-shot accuracy by 9.4%, and Frost blurring reduces Food101 linear-probe accuracy by 3.3%. The accuracy drop correlates with the average robustness value across perturbations. For instance, in Figure 4, both CLIP ViT-L/14 and DINO v2 ViT-g/14 exhibit the highest DivergenceRadius under Defocus blurring, corresponding to the largest ACC_p drop in Table 2. This suggests that higher DivergenceRadius values indicate greater embedding diversity, increasing the chance of misclassification.

ACC_p vs. robustness value: Figure 5 shows the relationship between ACC_p and DivergenceRadius for ImageNet images under various perturbations; results for Food101 are in Figure 6. Across datasets and perturbations, ACC_p decreases approximately linearly as DivergenceRadius or cosine similarity increases, indicating that greater embedding diversity leads to lower accuracy.

Predicting ACC_p using robustness values: The linear trend between ACC_p and DivergenceRadius (or cosine similarity) enables accurate ACC_p prediction via linear regression. We divide the dataset, train a linear model on the first half (predicting ACC_p from DivergenceRadius or cosine similarity), and test it on the second half. Figures 10 and 11 in the Appendix show low mean squared errors, indicating that DivergenceRadius or cosine similarity can reliably predict downstream accuracy under perturbations.



(a) Zero-shot classification (b) Linear-probe classification

Figure 6. Accuracy under perturbation ACC_p vs. DivergenceRadius of Food101 testing images for (a) zero-shot classification and (b) linear-probe classification when different perturbation functions are used. Zero-shot classification is based on the CLIP ViT-L/14 foundation model and linear-probe classification is based on the DINO v2 ViT-g/14 foundation model.

6. Robustness Enhancement

6.1. Method

Robustness and utility goals: We propose a fine-tuning method to enhance a foundation model's robustness against common perturbations. Given a foundation model f, our goal is to produce a model f' that meets both a *robustness goal* (increased robustness to perturbations) and a *utility goal* (maintaining performance on unperturbed images).

Formulating an optimization problem: We define two loss terms for robustness and utility. Minimizing these terms involves optimizing a weighted sum. For a set of unlabeled images $\mathcal{D}$, we quantify robustness by the cosine

similarity between an image's embedding and those of its perturbed versions. Specifically, the robustness loss term $\mathcal{L}_1$ is: $\mathcal{L}_1 = -\frac{1}{\|\mathcal{D}\|} \sum_{x \in \mathcal{D}} \cos\left(f'(x), f'(P(x,k))\right)$, where P is a perturbation function and k is sampled from $\mathbb{K}$ each epoch. To ensure utility, we use cosine similarity between embeddings of unperturbed images from f and f'. The utility loss term $\mathcal{L}_2$ is: $\mathcal{L}_2 = -\frac{1}{\|\mathcal{D}\|} \sum_{x \in \mathcal{D}} \cos\left(f(x), f'(x)\right)$.

The optimization problem then minimizes the weighted sum of these terms: $\min_{f'} \mathcal{L}_1 + \lambda \mathcal{L}_2$, where λ balances robustness and utility.

Solving the optimization problem: We use a gradient-based method, initializing f' as f, and iteratively updating f' using the gradient computed over mini-batches from $\mathcal{D}$.

6.2. Experimental Setup

We use CLIP ViT-L/14 for zero-shot classification. Fine-tuning data $\mathcal{D}$ includes 20,000 randomly selected ImageNet training images, with JPEG compression as the default perturbation. Models are fine-tuned for 50 epochs with a learning rate of 1×10^{-5} and $\lambda = 1$ unless otherwise stated.

6.3. Experimental Results

Achieving the robustness goal: Figures 7a and 7b show the average cosine similarity and DivergenceRadius for ImageNet/Food101 testing images before and after fine-tuning. We observe decreases in both metrics, indicating that embedding vectors of perturbed images become closer to those of unperturbed images, thus enhancing robustness.

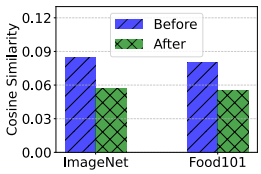
Achieving the utility goal: Figure 7c shows that zero-shot classification accuracy ACC remains nearly unchanged on unperturbed images after fine-tuning, indicating that the utility goal is met due to the inclusion of $\mathcal{L}_2$.

Impact of λ: Table 9 in the Appendix shows how λ affects the robustness-utility trade-off. When λ is small (e.g., 0), robustness is achieved but utility declines, whereas large λ (e.g., 5) favors utility over robustness.

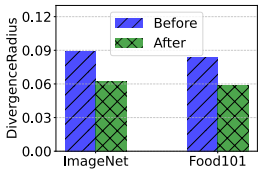
7. Related Work

Foundation models: Foundation models [5, 8, 15, 18, 21] are pre-trained neural networks used as general-purpose feature extractors, often for vision tasks. Vision foundation models are typically pre-trained on large datasets of unlabeled images [5, 18] or image-text pairs [21]. For example, Meta's DINO v2 [18] is trained on 142 million images, while OpenAI's CLIP [21] is trained on 400 million image-text pairs.

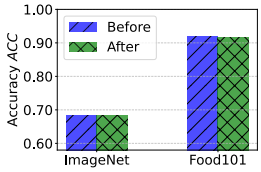
Common perturbations: Common perturbations frequently arise in real-world, non-adversarial settings. While adversarial robustness of foundation models has been widely studied, robustness to common perturbations is less explored.



(a) Cosine similarity



(b) DivergenceRadius



(c) ACC

Figure 7. (a) Average cosine similarity, (b) average DivergenceRadius, and (c) ACC of zero-shot classification for the two datasets before and after robustness enhancement, where the foundation model is CLIP ViT-L/14.

Zhu et al. [26] investigated language models' robustness to prompt perturbations but did not focus on vision models. Hendrycks and Dietterich [9] assessed classifier robustness to common perturbations, and studies [1, 2, 19, 24] compared robustness of vision transformers and CNNs, though they focused on classifiers, not foundation models. Hendrycks et al. [10] found that common perturbations could improve out-of-distribution robustness for classifiers, while our work centers on in-distribution robustness.

8. Conclusion and Future Work

In this work, we introduce three metrics—cosine similarity, Euclidean distance, and DivergenceRadius—to quantify foundation model robustness to common perturbations. Theoretically, we showed that DivergenceRadius meets all five desired mathematical properties, while cosine similarity and Euclidean distance do not satisfy the worst-robustness property. Using these metrics, we empirically evaluated the robustness of industry-scale foundation models and downstream applications, finding limited robustness to common perturbations, which impacts downstream performance. We also demonstrated that fine-tuning models to align perturbed embeddings with the original can enhance robustness without affecting utility. Future work includes extending these analyses to language models and investigating robustness against adversarial perturbations.

References

[1] Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? In *NeurIPS*, 2021. 8

[2] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. In *ICCV*, 2021. 8

[3] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *ECCV*, 2014. 5

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *S&P*, 2017. 1

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 8

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[7] Lijie Fan, Sijia Liu, Pin-Yu Chen, Gaoyuan Zhang, and Chuang Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? 2021. 1

[8] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 8

[9] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019. 5, 8

[10] 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 *ICCV*, 2021. 8

[11] Jinyuan Jia, Yupei Liu, and Neil Zhenqiang Gong. Badencoder: Backdoor attacks to pre-trained encoders in self-supervised learning. In *S&P*, 2022. 1

[12] Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. *NeurIPS*, 2020.

[13] Zhengyuan Jiang, Jinghuai Zhang, and Neil Zhenqiang Gong. Evading watermark based detection of ai-generated content. In *CCS*, 2023.

[14] Changjiang Li, Ren Pang, Zhaohan Xi, Tianyu Du, Shouling Ji, Yuan Yao, and Ting Wang. An embarrassingly simple backdoor attack on self-supervised learning. In *ICCV*, 2023. 1

[15] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022. 8

[16] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. PoisonedEncoder: Poisoning the unlabeled pre-training data in contrastive learning. In *USENIX Security Symposium*, 2022. 1

[17] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 5

[18] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv*, 2023. 1, 4, 8

[19] Sayak Paul and Pin-Yu Chen. Vision transformers are robust learners. In *AAAI*, 2022. 8

[20] Wenjie Qu, Jinyuan Jia, and Neil Zhenqiang Gong. Reaas: Enabling adversarially robust downstream classifiers via robust encoder as a service. In *NDSS*, 2023. 1

[21] 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 *ICML*, 2021. 1, 4, 8

[22] Aniruddha Saha, Ajinkya Tejankar, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Backdoor attacks on self-supervised learning. In *CVPR*, 2022. 1

[23] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 1

[24] Zeyu Wang, Yutong Bai, Yuyin Zhou, and Cihang Xie. Can cnns be more robust than transformers? In *ICLR*, 2023. 8

[25] Emo Welzl. Smallest enclosing disks (balls and ellipsoids). In *New Results and New Trends in Computer Science*. Springer, 2005. 4

[26] Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, et al. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv*, 2023. 8