arXiv:2409.15190v1 [cs.CV] 23 Sep 2024

Interpretability-Guided Test-Time Adversarial Defense

Akshay Kulkarni[1] and Tsui-Wei Weng[2]

[1] CSE, UC San Diego
[2] HDSI, UC San Diego
{a2kulkarni,lweng}@ucsd.edu

Abstract. We propose a novel and low-cost test-time adversarial defense by devising interpretability-guided neuron importance ranking methods to identify neurons important to the output classes. Our method is a training-free approach that can significantly improve the robustness-accuracy tradeoff while incurring minimal computational overhead. While being among the most efficient test-time defenses (4× faster), our method is also robust to a wide range of black-box, white-box, and adaptive attacks that break previous test-time defenses. We demonstrate the efficacy of our method for CIFAR10, CIFAR100, and ImageNet-1k on the standard RobustBench benchmark (with average gains of 2.6%, 4.9%, and 2.8% respectively). We also show improvements (average 1.5%) over the state-of-the-art test-time defenses even under strong adaptive attacks.

1 Introduction

Despite the popularity and success of deep neural networks (DNNs), they have been shown to be brittle against carefully designed small perturbations that are imperceptible to humans [21]. These "adversarial" perturbations pose significant risks against deploying DNNs for safety-critical tasks like autonomous driving. To mitigate these risks, there has been great interest in developing defenses, *e.g.* adversarial training algorithms [42, 45, 47, 56] to obtain a model that is more robust to adversarial samples.

However, adversarial training is much more expensive than standard training (*e.g.* a strong defense like TRADES [56] takes 150-200× longer than standard training). This inspired training-free defenses known as test-time defenses [3, 26, 31, 41]. The underlying principles in existing test-time defenses are either input purification or model adaptation: input purification [3, 41] modifies the input with an additional perturbation to nullify any adversarial perturbation, whereas model adaptation [14, 26] updates model parameters or adds new parameters at test-time to make the overall model more robust. Despite eliminating the need for training, most existing methods are still quite expensive, 8-500× more than

[3] Code: https://github.com/Trustworthy-ML-Lab/Interpretability-Guided-Defense
[4] This work is accepted for publication at ECCV 2024.

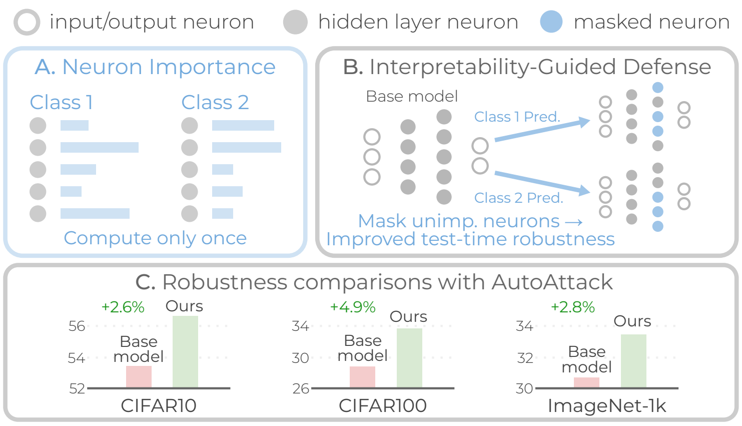

Fig. 1: Interpretability-guided masking based on neuron importance ranking for test-time adversarial defense.

a standard forward pass [16]. Moreover, existing test-time defenses have been broken under strong adaptive attacks [16]. Motivated by these limitations, in this paper, we intend to find an *efficient* and *effective* alternative to improve robustness at test-time, so that the defense does not break under adaptive attacks.

To design an effective defense, we leverage recent neuron-level interpretability methods [8, 34], which allow one to interpret the functionalities of individual neurons. In our analysis, we observed that for successful adversarial examples, the majority of activations at an intermediate layer shift from important neurons for the ground-truth class to the important neurons for other classes. Hence, we propose a novel test-time Interpretability-Guided Defense (**IG-Defense**) enabled by "neuron importance ranking" where class-wise importance of each neuron is computed (Fig. 1A). To design effective neuron importance ranking, we repurpose our analysis experiments and existing neuron-level interpretability works to propose two novel neuron importance ranking methods. The key idea of our **IG-Defense** is to restrict the observed activation shift by masking unimportant neurons (Fig. 1B). We demonstrate the usefulness of our insights by extensively evaluating our **IG-Defense** on the standard RobustBench benchmark (Fig. 1C). We observe consistent gains for CIFAR10 (+2.6%), CIFAR100 (+4.9%), and ImageNet-1k (+2.8%). We also devise an ensemble of strong adaptive attacks specifically targeting the weaknesses of **IG-Defense** and observe consistent gains (+1.5%), unlike existing test-time defenses that break under adaptive attacks. Due to our simple yet grounded approach, **IG-Defense** is among the most efficient test-time defenses ($4\times$ faster than the strongest existing defense).
Our primary contributions can be summarized as:

– We are the first to propose a neuron-interpretability-guided test-time defense (**IG-Defense**) utilizing neuron importance ranking (`CD-IR`, `LO-IR`) to improve adversarial robustness. **IG-Defense** is training-free, efficient, and effective.

– We uncover novel insights into improving adversarial robustness by analyzing adversarial attacks through the lens of neuron-level interpretability in Sec. 3.
– Our proposed **IG-Defense** consistently improves the robustness on standard CIFAR10, CIFAR100, and ImageNet-1k benchmarks. We also demonstrate improved robustness upto 3.4%, 3.8%, and 1.5% against a wide range of white-box, black-box, and adaptive attacks respectively with the lowest inference time (4$\times$ faster) among existing test-time defenses.

2 Related Work

Test-time adversarial robustness. Most of the test-time defenses can be categorized into two approaches: input purification or model adaptation. In input purification, the adversarial input is converted to a safer input by an additional perturbation. For instance, finding a perturbation that either increases top class confidence [3], minimizes contrastive loss [31], or minimizes a self-supervised objective [41]. [24] crafts the perturbation by fooling a real-adversarial discriminator [24] and [53] uses an input passed through a score-based generative model. For model adaptation [14,33,49], the model parameters are either updated at test-time or new parameters are introduced. However, existing test-time defenses are quite expensive (8-500$\times$ of single forward pass) and break under adaptive attacks. In contrast, our method is fast and effective (1.5% gain under adaptive attacks with only 2$\times$ inference time), and is the first test-time defense to leverage neuron-level interpretability information.
Neuron interpretability methods. To understand the functionalities (*i.e. concepts*) of individual neurons in DNNs, early works like NetDissect [8,9] compare the neuron activation patterns with patterns of predefined concept label masks. MILAN [23] produced natural language descriptions using a supervised image captioning model trained on a large-scale concept-labeled dataset. Recently, CLIP-Dissect [34] leveraged the CLIP paradigm [37] to identify concepts of neurons more efficiently than prior work and without the need of training or densely-concept-labeled datasets. There are other works to identify neuron concepts through clustering [20, 51], via language models [6], or using linear combinations [35]. In this work, we focus on repurposing CLIP-Dissect due to its efficiency and simplicity as well as NetDissect to propose neuron importance ranking methods.
Connections between interpretability and robustness. Prior works attempt to improve robustness via input-dependent interpretability methods like saliency or class activation maps [10, 30]. Wu et al. [48] observed that adversarial attacks lead to changes in attention maps, and proposed a training-time defense with losses to enforce similar attention maps for clean and adversarial samples. Several works [7, 19, 28, 40, 50, 57] use sparsity to improve adversarial robustness. However, they require retraining after or while sparsity constraints are applied. In contrast, we indirectly leverage sparsity without requiring any training. NetDissect [8] investigated the connection between adversarial attacks and interpretability information from their method, but they did not explore how to improve robustness. Hence, we aim to study the unexplored connection of neuron-level interpretability and improving adversarial robustness.

3 Analysis Experiments

To understand the connection of interpretability of individual neurons with robustness, we aim to design an analysis experiment. One way would be to investigate the changes in neuron activations before and after an adversarial attack. In neuron-labeling works [9,32], they assign an explainable concept to each neuron (or an activation channel in case of CNNs). Specifically, NetDissect [9] qualitatively show that a targeted adversarial attack causes a drop in peak activations for neurons whose concepts are related to the original class, while causing an increase in peak activations for neurons whose concepts are related to the target class of the attack. They also show that the peak activation change is lower for neurons with unrelated concepts. While they only perform the analysis to showcase their interpretability, we are the first to analyze adversarial attacks through neuron-level interpretability with the explicit goal of obtaining insights into improving adversarial robustness.

For a more thorough analysis, we observe the average activation change (instead of peak change) and associate the neurons with the category names directly instead of concepts. First, peak activation change may be misleading since it may be caused by specific images while average change takes all images into account for a more well-rounded analysis. Second, our automated evaluation eliminates the need for manual association of concepts with categories [9], which could be difficult, expensive, and noisy, especially with a large number of related categories (*e.g.* ImageNet classes). We analyze both successful and unsuccessful attacks while [9] only focused on successful attacks.

Concretely, we find top-k important neurons related to each task category using neuron importance ranking (discussed in Sec. 4.2). With this, in Fig. 2, we analyze the change in activations corresponding to ground-truth (GT) and non-GT categories before and after an attack. We discuss some preliminaries before delving into further details.

Preliminaries. We denote the dataset as $\mathcal{D} = \{(x, y)\}$ where $x \in \mathcal{X}$ is the input image, $y \in \mathcal{C}$ is the label, $\mathcal{C}$ is the label set, and $|\mathcal{C}|$ is the number of classes. Given a pretrained model $f : \mathcal{X} \to \mathcal{C}$ and its activations at the layer being studied $A : \mathcal{X} \to \mathbb{R}^{N \times H \times W}$, we can rank the importance of a neuron (out of N neurons/channels) by assessing its contribution to each class. We call this "neuron importance ranking" and propose two algorithms for this (Sec. 4.2). Then, we can compute the activations of the top-k important neurons for any class c_i as $A_{c_i}(x) = A(x)_{[n_{c_i}]} \in \mathbb{R}^{k \times H \times W}$, where n_{c_i} are the indices of the top-k activations for class c_i. With access to each n_{c_i}, the remaining indices absent from the top-k of any class are $n_u = [N] \setminus \{\cup_{c_i \in \mathcal{C}} n_{c_i}\}$, where $[N] = \{1, 2, \dots, N\}$ and $\setminus$ indicates set-minus operator. These indices n_u denote unimportant neurons, and corresponding unimportant activations can be computed as $A_u(x) = A(x)_{[n_u]} \in \mathbb{R}^{(N - k|\mathcal{C}|) \times H \times W}$. Next, we define activation change metrics to support our analysis.

Metrics. For an adversarial attack with ℓ_∞-norm bound of ϵ, the perturbation for input x is $\delta^* = \arg\max_{\|\delta\|_\infty \le \epsilon} \ell(f(x + \delta), y)$ where ℓ is the attack objective. Let $\hat{y}^* = f(x + \delta^*)$ be the post-attack prediction. The average activation change

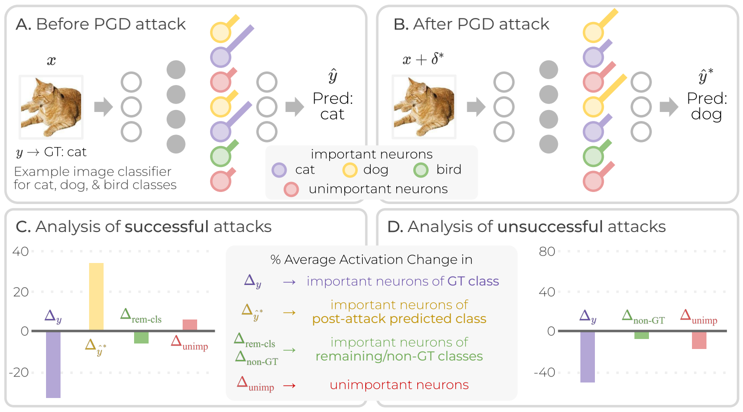
A. Before PGD attack
x
$y \to$ GT: cat
Example image classifier for cat, dog, & bird classes
$\hat{y}$
Pred: cat
B. After PGD attack
$x+\delta^*$
$\hat{y}^*$
Pred: dog
important neurons
cat
dog
bird
unimportant neurons
C. Analysis of successful attacks
40
20
0
-20
Δ_y
$\Delta_{\hat{y}^*}$
$\Delta_{\text{rem-cls}}$
Δ_{unimp}
% Average Activation Change in
Δ_y → important neurons of GT class
$\Delta_{\hat{y}^*}$ → important neurons of post-attack predicted class
$\Delta_{\text{rem-cls}}$ $\Delta_{\text{non-GT}}$ → important neurons of remaining/non-GT classes
Δ_{unimp} → unimportant neurons
D. Analysis of unsuccessful attacks
80
40
0
-40
Δ_y
$\Delta_{\text{non-GT}}$
Δ_{unimp}

Fig. 2: A. Cat-dog-bird image classifier before a PGD attack. **B.** After PGD attack, the prediction changes from the ground truth (GT) cat to dog since the activations of neurons important to dog class increase while those important to cat class decrease. **C.** Empirically, successful PGD attacks show a decrease in activations of important GT class neurons while those of post-attack predicted class' important neurons increase. **D.** For unsuccessful PGD attacks, the activations of all neurons reduce even though the prediction remains the same as before the attack.

in the important neurons w.r.t. any class c_i is,

$$\Delta_{c_i} = \mathop{\mathbb{E}}_{(x,y)\in\mathcal{D}} [A_{c_i}(x+\delta^*) - A_{c_i}(x)] \tag{1}$$

Replacing c_i with GT class y, the average activation change in the important neurons of GT class is Δ_y. Similarly, the average activation change in post-attack predicted class neurons is $\Delta_{\hat{y}^*}$, and that in other non-GT class neurons is $\Delta_{\text{non-GT}} = \mathbb{E}_{c_i \neq y, \hat{y}^*}[\Delta_{c_i}]$. For unsuccessful attacks, where post-attack prediction $\hat{y}^*$ is the same as y, we can compute average activation change in the remaining classes (other than GT) as $\Delta_{\text{rem-cls}} = \mathbb{E}_{c_i \neq y}[\Delta_{c_i}]$. Finally, we can also compute the average activation change in unimportant neurons,

$$\Delta_{\text{unimp}} = \mathop{\mathbb{E}}_{(x,y)\in\mathcal{D}} [A_u(x+\delta^*) - A_u(x)] \tag{2}$$

We empirically compute the above metrics on CIFAR10 with five base models (ResNet18: GAT [42], NuAT [43], OAAT [2], DAJAT [1], and WideResNet-34-10: TRADES-AWP [47]) and present the mean values in Fig. 2C and Fig. 2D. See Appendix B.1 for more experimental details and corresponding numerical results.
Observations. In Fig. 2C, we observe $\Delta_y < 0$ and $\Delta_{\hat{y}^*} > 0$, *i.e.* successful adversarial attacks boost the activations that are important to the post-attack predicted class $\hat{y}^*$ while causing a drop for those of the GT class y. Further, from Fig. 2C, the important activations for the remaining classes and unimportant activations are only marginally affected by successful attacks. In Fig. 2D, we observe that unsuccessful attacks cause a drop in the activations of all neurons.

Remarks. During both successful and unsuccessful attacks, the activation magnitude of GT class' important neurons drops. Specifically, successful attacks shift the activation magnitude from GT class' important neurons to non-GT class important neurons. Hence, we hypothesize that adversarial robustness can be improved if the activation shift to non-GT class important neurons is restricted.

Further, given the availability of a huge number of adversarially pretrained models [15], we aim to evaluate our hypothesis in a test-time adversarial defense setting, with a low inference overhead and without any training overhead.

4 Interpretability-Guided Defense

In this section, we propose a novel, training-free, test-time Interpretability-Guided Defense (**IG-Defense**). The core idea of our method is based on interpretability-guided masking, described in Sec. 4.1. In Sec. 4.2, we introduce neuron importance ranking, which is a key enabler in the success of our proposed defense.
Setup. Given a trained model $f : \mathcal{X} \to \mathcal{C}$, the goal of test-time defense is to improve the robustness of f while retaining its utility (*i.e.* clean accuracy). We refer to f as the "base model" in the rest of the paper.

4.1 Interpretability-guided masking

To validate our hypothesis, we design a simple approach for test-time defense where we mask all the neurons except the important neurons of the correct class. We propose a dual forward pass approach, since access to labels cannot be assumed at test-time. Our approach contains three key steps:

- **Step 1: Neuron Importance Ranking.** First, we compute class-wise neuron importance ranking (Step 1 of Fig. 3). This is computed only once for a given base model f. For each layer, we rank every neuron based on its importance to each class. More details are given in Sec. 4.2.
  - Given the number of important neurons k (hyperparameter), we compute a binary mask $m \in \{0,1\}^{N\times C}$ using the indices of important neurons $n_{c_i} \in \mathbb{N}^k$ (from the importance ranking in Sec. 4.2) for each class $c_i \in [C]$,

$$m[n_{c_i}, c_i] = 1 \;\forall\; c_i \in [C], \quad m[[N] \setminus n_{c_i}, c_i] = 0 \;\forall\; c_i \in [C] \tag{3}$$

  - In the mask m, the top-k important neurons of each class $c_i \in [C]$ are retained in $m[:, c_i]$. The remaining neurons unimportant to c_i are masked.
- **Step 2: Vanilla Forward Pass.** In the first forward pass (Step 2 of Fig. 3), we obtain a soft-pseudo-label $\hat{y} = \sigma\left(\frac{f(x)}{\tau}\right) \in (0,1)^C$ where $\sigma(\cdot)$ denotes the softmax function, and τ denotes the temperature term which controls the sharpness of the soft-pseudo-label.
- **Step 3: Masked Forward Pass.** In the second forward pass (Step 3 of Fig. 3), we apply a soft-pseudo-label weighted mask *i.e.* $m\hat{y} \in (0,1)^N$ to the activations of the layer being masked. Let $A : \mathcal{X} \to \mathbb{R}^{N\times H\times W}$ give

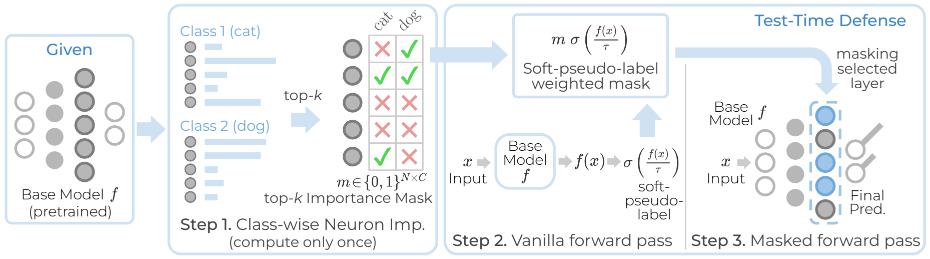
Given
Base Model f (pretrained)
Class 1 (cat)
Class 2 (dog)
top-k
cat
dog
$m \in \{0,1\}^{N\times C}$
top-k Importance Mask
Step 1. Class-wise Neuron Imp. (compute only once)
$m\,\sigma\left(\frac{f(x)}{\tau}\right)$
Soft-pseudo-label weighted mask
x Input
Base Model f
$f(x)$
$\sigma\left(\frac{f(x)}{\tau}\right)$
soft-pseudo-label
Step 2. Vanilla forward pass
Test-Time Defense
masking selected layer
Base Model f
x Input
Final Pred.
Step 3. Masked forward pass

Fig. 3: Step 1. Given a pretrained base model f (*e.g.* binary classifier here), class-wise neuron importance is computed for a selected layer (Sec. 4.2). A top-k mask $m \in \{0,1\}^{N\times C}$ is computed to identify top-k neurons important to each class (*e.g.* $k = 2, N = 5, C = 2$ here). **Step 2.** During evaluation, a soft-pseudo-label $\hat{y}$ is computed using the base model f. **Step 3.** The soft-pseudo-label weighted mask $m\hat{y} = m\ \sigma(\frac{f(x)}{\tau}) \in \mathbb{R}^N$ is applied to the selected layer to retain only the important neurons of the pseudo-label class.

the activations at the layer being masked, and $h : \mathbb{R}^{N\times H\times W} \to \mathcal{C}$ be the remaining layers after the layer being masked, *i.e.* $f = h \circ A$. Then, the final prediction is given by $\hat{y}' = h(m\hat{y} \cdot A(x))$.

- The masking re-weights the neurons based on $\hat{y}$. However, the temperature term τ is set to a small value so that $\hat{y}$ contains a sharp probability distribution. Thus, the weighted mask corresponds mainly to a single class and retains only the important neurons of that class.
- Also note that this enables gradient flow between the two forward passes, to avoid any gradient masking [5].

Intuitively, activations of a deeper layer (*e.g.* penultimate layer) are masked to minimize any negative impact of the masking on feature extraction, which depends more on the shallower layers [55]. To verify this, we analyze the sensitivity of our method to the layer being masked in Sec. 5.3b.

Dependence on soft-pseudo-label. This is an obvious limitation if the soft-pseudo-label is computed directly from the base model f. If it is incorrect, masking will yield no improvement in robustness. Hence, an adaptive attack can be designed to ensure that the soft-pseudo-label is incorrect. For instance, a transfer attack that transfers the perturbation from the base model f would yield an incorrect soft-pseudo-label, and the final robustness would be the same as the base model f. To circumvent this, we propose to use randomized smoothing in the first forward pass (*i.e.* in Step 2 of Fig. 3).

Randomized smoothing. To obtain the soft-pseudo-label $\hat{y}$ in a more robust manner, we perform randomized smoothing [13] in the first forward pass. We add a set of n_s Gaussian noises (with zero mean and standard deviation σ_d) to each input x. For the soft-pseudo-label, the logits of these noisy inputs are averaged and softmax σ is applied. Formally,

$$\hat{y} = \sigma\left(\frac{1}{\tau}\mathop{\mathbb{E}}_{v_i}[f(x+v_i)]\right);\quad v_i \sim \mathcal{N}(0_d, \sigma_d^2 I_d)\ \forall\, i \in [n_s] \tag{4}$$

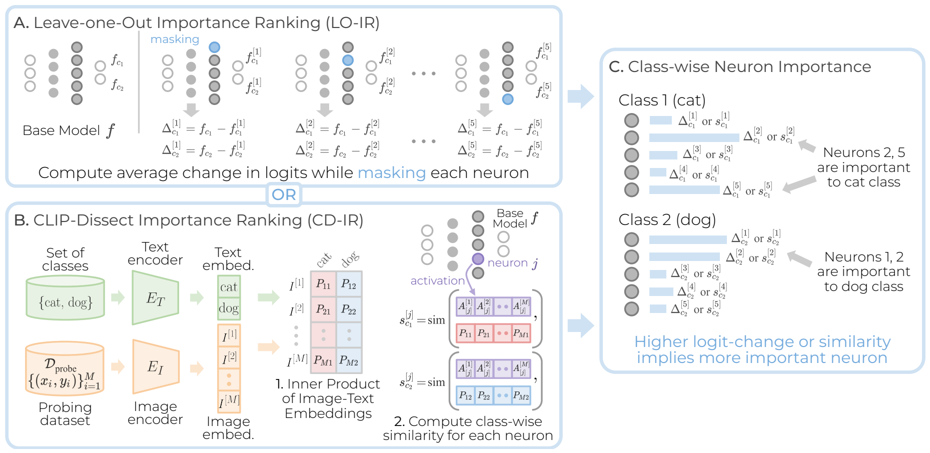
A. Leave-one-Out Importance Ranking (LO-IR)
masking
Base Model f
Compute average change in logits while masking each neuron
OR
B. CLIP-Dissect Importance Ranking (CD-IR)
Set of classes
Text encoder
Text embed.
Probing dataset
Image encoder
Image embed.
1. Inner Product of Image-Text Embeddings
2. Compute class-wise similarity for each neuron
activation
neuron j
C. Class-wise Neuron Importance
Class 1 (cat)
Neurons 2, 5 are important to cat class
Class 2 (dog)
Neurons 1, 2 are important to dog class
Higher logit-change or similarity implies more important neuron

Fig. 4: A. Leave-one-Out Importance Ranking (`LO-IR`) computes class-wise neuron importance as the average change in logits when masking each neuron. **B.** CLIP-Dissect Importance Ranking (`CD-IR`) first computes the inner product of class name text embeddings and probing set image embeddings. The importance ranking relies on the similarity between activations of each neuron to the precomputed inner product given the same probing set inputs. **C.** A higher logit-change $\Delta_{c_i}^{[j]}$ or similarity $s_{c_i}^{[j]}$ implies that neuron j is more important to class c_i.

where $0_d, I_d$ are tensors with the same shape as x but containing all zeros and ones respectively. The number of noises sampled n_s and standard deviation σ_d are hyperparameters.

4.2 Importance ranking methods

Here, we describe two novel importance ranking methods that can be used in Step 1 of our proposed **IG-Defense**.

a) Leave-one-Out Importance Ranking (`LO-IR`). Following the analysis in Sec. 3, given a neuron j, we can compute its importance to a class c_i as the average change in the class c_i logits, $\Delta_{c_i}^{[j]}$, before and after masking out that particular neuron j (Fig. 4A). Intuitively, a higher logit-change for a particular class implies higher dependence of the network on that neuron, *i.e.* higher importance (Fig. 4C). Formally,

$$\Delta_{c_i}^{[j]} = \mathop{\mathbb{E}}_{(x,y)\in\mathcal{D}_{\text{probe}}|y=c_i} [f_{c_i}(x) - f_{c_i}^{[j]}(x)]; \quad \forall\, c_i \in \mathcal{C} \tag{5}$$

where $f^{[j]}$ is the model where j^{th} neuron of the selected layer is masked in the base model f. And the subscript c_i indicates the logits of class c_i. However, this method could be computationally expensive as average change in logits has to be computed for each neuron. For our proposed method, it is only required once per base model. Thus, it is a feasible option that yields a very good estimate of the importance ranking, which we show in Sec. 5.

b) CLIP-Dissect Importance Ranking (`CD-IR`). CLIP-Dissect [34] uses the multimodal CLIP model [37] to assign concept labels to individual neurons. We extend this idea by replacing concepts with task labels to obtain an importance ranking of neurons for each task label (see Fig. 4B).

Concretely, probing images are passed through the image encoder E_I to get a set of image embeddings $\{A^{[i]} = E_I(x_i)\}_{i=1}^{M}$ (shown in orange in Fig. 4B) while the task label names $\mathcal{C}$ are passed through the text encoder E_T to get the corresponding set of text embeddings $\{E_T(c_k)\}_{c_k \in \mathcal{C}}$ (shown in green in Fig. 4B). These embeddings *i.e.* vectors are converted to scalars by taking the inner product of each pair of image-text embeddings, which results in a matrix P with entries $P_{ik} = A^{[i]\top} E_T(c_k)$. Now, for a neuron j, a vector of its activations q_j (shown in purple in Fig. 4B) can be obtained where each entry corresponds to each input from the probing dataset. We can compute the similarity score $s_{c_k}^{[j]}$ between each class c_k and neuron j using the k^{th} column of P and q_j. Intuitively, a higher similarity score $s_{c_k}^{[j]}$ implies higher importance of neuron j for class c_k (Fig. 4C). We use the same soft-WPMI similarity metric as the original paper [34]. For the probing dataset, we simply use the training dataset for our experiments.

In practice, `CD-IR` is computationally less expensive compared to `LO-IR` (*e.g.* 2 mins. vs. 28 mins. for 512 neurons). This is because CLIP embeddings can be precomputed and most of the computations can be vectorized easily. As we demonstrate in Sec. 5, although the importance ranking quality is slightly worse than `LO-IR` (*i.e.* `CD-IR` causes a marginal drop in clean accuracy), downstream robustness gains with `CD-IR` are still significant. Please refer to the Appendix C.3 for details on computational complexity comparisons.

5 Experiments

We extensively evaluate our **IG-Defense** on the standard RobustBench benchmark [15]. We also follow Croce et al. [16] for properly evaluating test-time defenses and thoroughly test against an ensemble of strong adaptive attacks.

Experimental setup. We evaluate our methods on standard datasets, CIFAR10, CIFAR100 [27] and ImageNet-1k [38], with ResNet [22] and WideResNet [54] base models. We use an ℓ_∞ threat model with an ϵ-bound of 8/255 for CIFAR10/100 and 4/255 for ImageNet-1k following prior works [15]. For evaluation, we use the full CIFAR10/100 test set (10000 samples) and the first 5000 ImageNet-1k validation samples, which is the standard setup as per [15,39]. The hyperparameter k for the top-k neurons in `CD-IR` or `LO-IR` is mentioned alongside the method name in the results. For randomized smoothing (RS), we use $n_s = 1, \sigma_d = \frac{\epsilon}{2}$, and $\tau = 0.01$, where ϵ is the ℓ_∞-bound for the attack. We show in Appendix C.2 that the choices of $n_s = 1, \sigma_d = \frac{\epsilon}{2}$ are sufficient for our purpose. Unless otherwise mentioned, RS is used and the penultimate layer output is masked. In the penultimate layer, there are 512 neurons (layer4) in ResNet18, 640 neurons (layer3 or block3) in WideResNet-34-10, and 2048 neurons (layer4) in ResNet50.

Choice of base models. We experiment with base models from several single-step defenses like FAT [45], GAT [42], NuAT [43], OAAT [2], and multi-step

Table 1: Evaluating our **IG-Defense** (`CD-IR`, `LO-IR`) on **CIFAR10** with **ResNet18** and $\ell_\infty, \epsilon = 8/255$. IW-WC indicates image-wise worst-case, AA indicates AutoAttack.

Base Model:	NuAT [43] (RN18)			OAAT [2] (RN18)			DAJAT [1] (RN18)		
Test-Time Defense	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)
None	82.21	50.58	50.54	**80.23**	51.10	51.01	85.71	52.50	52.45
`CD-IR-50` (*Ours*)	**82.46**	52.08	51.58 (+1.04)	79.85	54.79	**52.35 (+1.34)**	85.11	**54.81**	**53.36 (+0.91)**
`LO-IR-50` (*Ours*)	82.17	**52.41**	**52.19 (+1.65)**	79.90	**54.90**	52.30 (+1.29)	85.18	54.53	53.34 (+0.89)

Table 2: Evaluating our **IG-Defense** (`CD-IR`, `LO-IR`) on **CIFAR10** with **WRN-34-10** and $\ell_\infty, \epsilon = 8/255$. IW-WC indicates image-wise worst-case, AA indicates AutoAttack.

Base Model:	NuAT [43] (WRN-34-10)			TR-AWP [56] (WRN-34-10)			DAJAT [1] (WRN-34-10)		
Test-Time Defense	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)
None	**86.32**	54.75	54.72	**85.36**	56.17	56.12	**88.71**	57.81	57.72
`CD-IR-50` (*Ours*)	86.28	**57.69**	56.38 (+1.66)	84.98	**60.26**	57.46 (+1.34)	88.51	**60.70**	59.22 (+1.50)
`LO-IR-50` (*Ours*)	86.21	57.04	**56.40 (+1.68)**	85.17	59.61	**57.88 (+1.76)**	88.61	60.08	**59.41 (+1.69)**

defenses like TRADES [56], AWP [47], DAJAT [1], RTB [39], and LAS [25]. We chose these models to ensure a diverse evaluation and showcase the generality of our **IG-Defense** across base model defense strategies. We also evaluate with the same base model on different architectures as well as on different datasets for a thorough analysis. Please see Appendix B.2 for more details on these defenses.
Evaluation. We evaluate our **IG-Defense** with the base models against a diverse range of attacks. **AutoAttack** [18] (AA) is an empirically strong attack with a sequence of white-box AutoPGD (APGD) attacks, fast adaptive boundary (FAB) attack [17], and black-box Square attack [4]. For a stronger adaptive attack, we devise an ensemble of attacks. For each image, the adaptive attack is considered successful if any one attack from the ensemble is successful. With this, we get the image-wise worst-case (**IW-WC**) robust accuracy. To be adaptive for our **IG-Defense**, the ensemble also includes transfer attacks that specifically target our weakness, *i.e.* our dependence on pseudo-labels (as discussed in Sec. 4.1).

Robust accuracy (IW-WC) is computed with 9 attacks. The first three are computed on the defended model: standard white-box AutoAttack, decision-boundary based black-box RayS attack [12], and white-box APGD with Expectation over Transformation (EoT) [5] attack. For the remaining 6 transfer attacks, we compute perturbations on the base model using 6 attacks: APGD with cross-entropy loss, APGD with Carlini-Wagner loss [11], targeted APGD attack with difference of logits ratio loss, standard AutoAttack, RayS attack, and APGD+EoT attack.
Design choices for IW-WC. While AutoAttack also uses APGD attacks, it returns unperturbed inputs when the attack is unsuccessful. Following [16], the four transfer APGD attacks in IW-WC each return a sample with the highest loss (*i.e.* closest to the decision boundary). This is a stronger attack than AutoAttack for test-time defenses since samples closer to the decision boundary tend to be misclassified under test-time defenses, as observed by [16]. **Expectation over Transformation** (EoT) [5] averages gradients over multiple iterations of

Table 3: Evaluating our **IG-Defense** (`CD-IR`, `LO-IR`) on **CIFAR100** with $\ell_\infty, \epsilon{=}8/255$. IW-WC indicates image-wise worst-case, AA indicates AutoAttack.

Base Model:	LAS [25] (WRN-34-10)			OAAT [2] (RN18)			DAJAT [1] (RN18)		
Test-Time Defense	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)
None	**64.89**	30.74	30.71	62.02	27.15	27.12	**65.45**	27.67	27.64
`CD-IR-50` (*Ours*)	62.13	33.57	31.41 (+0.70)	61.56	29.51	29.19 (+2.07)	64.97	30.22	28.27 (+0.63)
`LO-IR-50` (*Ours*)	64.85	**35.69**	**32.22 (+1.51)**	**62.04**	**32.06**	**30.12 (+3.00)**	65.37	**32.55**	**28.54 (+0.90)**

Table 4: Evaluating our **IG-Defense** (`CD-IR`, `LO-IR`) on **ImageNet-1k** with ResNet50 and $\ell_\infty, \epsilon{=}4/255$. IW-WC indicates image-wise worst-case, AA indicates AutoAttack.

Base Model:	FAT [45] (RN50)			RTB [39] (RN50)		
Test-Time Defense	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (AA)	Robust Acc. (IW-WC)
None	55.64	26.24	26.24	64.10	34.62	34.62
`CD-IR-750` (*Ours*)	54.30	27.28 (+1.04)	26.80 (+0.56)	62.50	36.42 (+1.80)	35.30 (+0.68)
`LO-IR-750` (*Ours*)	**55.88**	**29.14 (+2.90)**	**27.08 (+0.84)**	**64.10**	**37.36 (+2.74)**	**35.48 (+0.86)**

a particular input to counter the effect of randomness in a defense. Specifically, we use 20 iterations of EoT with APGD (more than $n_s = 1$). While [16] also recommend using Backward Pass Differentiable Approximation (BPDA) [5] for test-time defenses with non-differentiable components, our approach already includes BPDA since we use a soft-masking approach instead of hard-masking. Refer to Appendix B.3 for complete implementation details.

5.1 Evaluation on RobustBench

a) CIFAR10. In Table 1, we evaluate our methods on CIFAR10 with ResNet18 at $\epsilon = 8/255$. We observe consistent gains in AutoAttack (avg. +2.5% for `CD-IR` and +2.55% for `LO-IR`) and IW-WC robustness (avg. +1.1% for `CD-IR` and +1.28% for `LO-IR`), with a marginal drop in clean accuracy (avg. −0.24% for `CD-IR` and −0.3% for `LO-IR`). In Table 2, we evaluate on CIFAR10 with the more challenging, large WideResNet-34-10 model at $\epsilon = 8/255$. We observe consistent gains in IW-WC robustness (avg. +1.5% for `CD-IR` and +1.7% for `LO-IR`), with a marginal drop in clean accuracy (avg. −0.2% for `CD-IR` and −0.13% for `LO-IR`).
b) CIFAR100. In Table 3, we evaluate our methods on CIFAR100 with ResNet18 and WideResNet-34-10 at $\epsilon{=}8/255$. Despite the increased number of classes, we get consistently improved worst-case (IW-WC) robust accuracy (avg. +1.13% for `CD-IR` and +1.8% for `LO-IR`). However, the clean accuracy drops for `CD-IR` (avg. −0.43% drop), but only marginally for `LO-IR`.
c) ImageNet-1k. In Table 4, we evaluate our methods on ImageNet-1k with ResNet50 at $\epsilon = 4/255$. Despite the increased image size and the higher number of classes, we observe consistent gains in worst-case (IW-WC) robustness (avg. +0.6% for `CD-IR` and +0.85% for `LO-IR`) but with a drop in clean accuracy for `CD-IR` (avg. −1.47% similar to CIFAR100). Given the scale of ImageNet-1k, even

Table 5: Comparison against state-of-the-art test-time defenses with ResNet18 and WideResNet-34-10 on CIFAR10. The number in (green) shows the worst-case robustness gain over the base model. The first row is the base model without any test-time defense. Inference time is a multiple of the time for a single forward pass of the base model.

Base Model →	OAAT [2] (RN18)		DAJAT [1] (RN18)		TR-AWP [47] (WRN)		Inference Time
Test-Time Defense	Clean Acc.	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (IW-WC)	Clean Acc.	Robust Acc. (IW-WC)	
No test-time def.	80.23	51.01	85.71	52.45	85.36	56.12	1×
Existing Defenses							
HD [46]	79.89	50.68 (-0.33)	84.53	52.55 (+0.10)	84.78	55.72 (-0.40)	46×
SODEF [26]	80.23	50.67 (-0.34)	84.86	52.95 (+0.50)	85.25	56.05 (-0.07)	2×
CAAA [3]	80.23	51.00 (-0.01)	85.71	52.45 (+0.00)	85.35	56.11 (-0.01)	8×
Our IG-Defense							
`CD-IR-50`	79.85	**52.35 (+1.34)**	85.11	**53.36 (+0.91)**	84.98	57.46 (+1.34)	**2×**
`LO-IR-50`	79.90	52.30 (+1.29)	85.18	53.34 (+0.89)	85.17	**57.88 (+1.76)**	**2×**

robustness improvements of $\sim 0.5\%$ are quite significant, especially since there is no clean accuracy drop with `LO-IR`.

Why does the clean accuracy drop? There are two possible reasons for the drop in clean accuracy. First, due to the randomized smoothing, some samples may get incorrect predictions if the random noise is significant enough to change the class prediction w.r.t. the base model. Second, importance ranking quality and number of neurons masked determine how much clean accuracy can be retained. That is, better quality or lower number of neurons masked would guarantee better clean accuracy retention. However, simply masking lower number of neurons may also reduce the improvements in robustness.

We see in Table 3, 4 that clean accuracy drop for `CD-IR` is more for CIFAR100 and ImageNet1k compared to CIFAR10 (Table 1, 2). This is possibly because the ratio of number of neurons to number of classes is decreasing, *i.e.* more number of neurons will be important to multiple classes. This cannot be captured well by `CD-IR`, leading to a larger drop in clean accuracy. Note that `LO-IR` is less susceptible to this problem as the importance ranking of each neuron is computed independently, but it is more expensive. This is why we need to retain more number of important neurons for ImageNet ($k=750$) than CIFAR ($k=50$).

Note that the clean accuracy drops are acceptable since they are significantly smaller than the worst-case robustness gains (*e.g.* upto -0.6% and $+1.7\%$ respectively for CIFAR10 in Table 1-2). Further, in some cases like ImageNet-1k with `LO-IR` (Table 4), there are no drops with pure gains in robustness.

5.2 Comparisons with existing test-time defenses

We compare with three recent state-of-the-art test-time defense methods on CIFAR10 in Table 5. CAAA [3] (input purification) optimizes an "anti-adversary" perturbation in a direction opposite to an adversarial attack, by minimizing the loss based on a pseudo-label, to counter the effect of any adversarial perturbation. HD or Hedge Defense [46] (input purification) finds a perturbation that maximizes

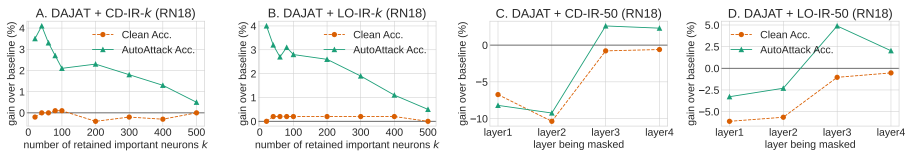

Fig. 5: Sensitivity analyses for number of retained important neurons k and layer being masked with CIFAR10. For **A**, **B**: layer4 with total 512 neurons is used. A lower k and a deeper layer yield better adversarial robustness with minimal loss of clean accuracy.

the cross-entropy loss summed over all classes. SODEF [26] (model adaptation) trains a neural ODE block with Lyapunov stability constraints between the pretrained feature extractor and classifier, which reduces the influence of the adversarial perturbation. In contrast, our method uses neither input purification nor model adaptation, since we only mask the activations.

In Table 5, we observe that existing test-time defenses like HD, CAAA, and SODEF do not significantly improve, but rather also reduce the worst-case robust accuracy (IW-WC) compared to the base model. Based on Croce et al. [16], CAAA is the strongest test-time defense among the nine defenses that they evaluated, since its worst-case robust accuracy did not drop below that of the base model. Hence, we only compare with the two strongest input purification defenses, CAAA and HD as well as an efficient model adaptation defense, SODEF from the nine defenses in [16]. Overall, we find that our proposed **IG-Defense** obtains consistent improvements in image-wise worst-case (IW-WC) robust accuracy, unlike existing test-time defenses. Note that we focus mainly on the worst-case (IW-WC) robust accuracy because gains under weaker attacks would not reflect fundamental gains in robustness [16].

5.3 Analysis

a) Ablation study. In Table 6, we study the significance of each proposed component of our approach. First, we start with the baseline (row #1) which uses only one forward pass without masking and without randomized smoothing (RS). Next, we add RS to this baseline (row #2) and observe no gains in robustness and a small drop in clean accuracy. This shows that RS alone cannot improve the robustness. Then, we perform two forward passes without masking and without RS (row #3). Here, the robust accuracy is the same as the baseline indicating that there is no gradient masking with two forward passes. Next, we perform random masking with two forward passes, without and with RS (row #4, #5). Each class is assigned a random set of neurons as the "important" neurons to perform the masking. We observe no gains in robustness, validating the significance of our neuron importance ranking. Finally, we show the impact of RS with our `CD-IR` (row #6, #7) and `LO-IR` based masking (row #8, #9). Here, we observe consistent gains in robustness, with a small drop in clean accuracy since RS likely changed the prediction w.r.t. the base model for a few samples.

b) Sensitivity analyses. We study the sensitivity of our proposed methods to the number of retained important neurons k in Fig. 5A, B on CIFAR10. Specifically, we test it on ResNet18 layer4 which has total 512 neurons and we vary k from 10 to 500. We observe that the robustness improvement is higher at lower k, although lower than 10 may reduce the clean accuracy due to a high loss of information. For `CD-IR`, the clean accuracy can be retained up to $k = 100$ while for `LO-IR`, it can be retained up to at least $k = 400$. As explained earlier, it is due to the relatively lower importance ranking quality of `CD-IR` w.r.t. `LO-IR`. However, both methods are fairly insensitive to the choice of k.

Table 6: Ablation study of proposed components with TRADES-AWP [47] (WideResNet-34-10) as the base model. RS indicates use of randomized smoothing in the first forward pass. The first row is the base model performance without any test-time defense.

Row #	No. of forw. passes	Masking type	RS	Clean Acc.	Rob. Acc. (AA)	Rob. Acc. (IW-WC)
1	1	None	✗	**85.36**	56.17	56.12
2	1	None	✓	84.93	56.19	56.14
3	2	None	✗	85.35	56.15	56.12
4	2	`Random-50`	✗	85.23	56.14	56.12
5	2	`Random-50`	✓	85.33	56.20	56.15
6	2	`CD-IR-50`	✗	85.09	59.00	56.12
7	2	`CD-IR-50`	✓	84.98	**60.26**	57.46
8	2	`LO-IR-50`	✗	85.33	58.59	56.12
9	2	`LO-IR-50`	✓	85.17	59.61	**57.88**

We also study the sensitivity to choice of layer being masked in Fig. 5C, D on CIFAR10 with ResNet18. We fix the number of neurons being retained k to 50 and vary the layer being masked from layer1 (shallower) to layer4 (deeper). We observe that masking deeper layers is better, since masking shallower layers affects the feature extraction [55], leading to a large drop in clean accuracy, and consequently a drop in robust accuracy. For all the other experiments, we simply mask the penultimate layer since the clean accuracy drop is the lowest. Please refer to Appendix C for more analysis experiments.

c) Inference time comparisons. Since we use $n_s = 1$ in randomized smoothing and perform two forward passes, the inference time of **IG-Defense** is $2\times$ of a single forward pass. The compared test-time defenses HD [46], SODEF [26], and CAAA [3] have computational complexities of $46\times$, $2\times$, and $8\times$ respectively. Also, the recent review paper [16] shows that the lowest inference time is $2\times$ for existing test-time defenses. Hence, as per Table 5, **we achieve the most gains in worst-case robustness with the lowest inference time**.

6 Conclusion

In this work, we proposed a novel, training-free, and effective test-time defense (**IG-Defense**) guided by interpretability through neuron importance ranking. Our **IG-Defense** outperformed existing test-time defenses on worst-case adaptive attacks while being among the most efficient test-time defenses to date, establishing the new state-of-the-art. We also demonstrated consistent improvements across standard CIFAR and ImageNet-1k benchmarks.

Acknowledgements

This work is supported in part by National Science Foundation (NSF) awards CNS-1730158, ACI-1540112, ACI1541349, OAC-1826967, OAC-2112167, CNS-2100237, CNS-2120019, the University of California Office of the President, and the University of California San Diego's California Institute for Telecommunications and Information Technology/Qualcomm Institute. Thanks to CENIC for the 100Gbps networks. The authors thank the anonymous reviewers for valuable feedback on the manuscript. The authors are partially supported by National Science Foundation awards CCF-2107189, IIS-2313105, IIS-2430539. T.-W. Weng also thanks the Hellman Fellowship for providing research support.

Appendix

In this document, we provide extensive implementation details and additional performance analysis. Towards reproducible research, we will release our complete codebase and saved importance rankings for the reported base models in this GitHub repository. The appendix is organized as follows:

A Related Work

Connections between interpretability and robustness.
Several works [7, 19, 28, 40, 50, 57] use sparsity to improve adversarial robustness. Eigen and Sadovnik [19] replace standard convolutions with top-k convolutions that only output the sum of top-k channels instead of all channels for each convolutional filter. This requires the model to be trained from scratch. Bai et al. [7] introduce learnable channelwise activation suppressing modules which select some channels to be masked. This adds new learnable parameters and also requires the model to be trained from scratch.

A line of work [28, 40, 57] focuses on pruning aiming to train a model with robust pruning objectives using both original and adversarial images to obtain a sparser, more robust model. Xiao et al. [50] propose a method similar to [19] except that they replace ReLU with a top-k activation function instead of modifying the convolutional layers. But their approach deliberately relies on gradient masking, which is apparent from their highly irregular or jagged loss surfaces. However, these methods require retraining or training from scratch, which is expensive. In contrast, our method indirectly leverages sparsity without requiring any training.

Table 7: Analysis experiment results in table form (the average results, *i.e.* last row, are illustrated in Fig. 2 of the main paper).

Base Model	% change: successful attacks				% change: unsuccessful attacks		
	Δ_y	$\Delta_{\hat{y}^*}$	$\Delta_{\text{rem-cls}}$	Δ_{unimp}	Δ_y	$\Delta_{\text{non-GT}}$	Δ_{unimp}
GAT [42]	-17.84	15.83	-0.7087	-0.3593	-51.00	-19.92	-16.1
NuAT [43]	-62.67	56.85	-1.44	-2.01	-84.77	-6.63	-25.06
OAAT [2]	-21.49	20.31	-0.32	-0.015	-52.37	-14.37	-21.35
DAJAT [1]	-44.84	46.58	0.86	0.63	-39.75	5.06	-0.34
TR-AWP [47]	-19.80	23.34	2.84	0.54	-13.36	13.43	4.48
Average	-33.33	32.58	0.246	-0.243	-48.25	-4.48	-11.67

B Implementation Details

B.1 Analysis experiments setup

For the analysis experiments in Sec. 3 of the main paper, we use 50-step PGD attack [29] on five base models, ResNet18: GAT [42], NuAT [43], OAAT [2], DAJAT [1], and WideResNet-34-10: TRADES-AWP [47] for the CIFAR10 dataset. We provide the results from Fig. 2 of the main paper in a numerical form in Table 7 for clarity. The % activation change is the relative change w.r.t. the original activation value.

B.2 Base models

We experiment with several single-step (first four) and multi-step defenses (remaining) to demonstrate the wide applicability of our **IG-Defense**. We briefly discuss the details of the base models' training strategies below.

- FAT [45] - This uses the randomized fast gradient signed method (R-FGSM) attack during adversarial training.
- GAT [42] - This uses the maximum margin loss instead of cross-entropy for the attack during adversarial training.
- NuAT [43] - This uses R-FGSM attack with a nuclear norm regularizer to locally smoothen the loss surface and reduce gradient masking.
- OAAT [2] - This uses mixup between an original and a perceptually-visible adversarial image (generated with $\epsilon = 24/255$ and LPIPS-based attack) to improve robustness.
- DAJAT [1] - This uses strong and light data augmentations to improve adversarial training by using separate batch-norm parameters for the two augmentation types.
- TRADES [56] - This uses a regularizer to push the decision boundary away from data points, which improves robustness.
- AWP [47] - This regularizes the loss landscape flatness by perturbing both input and weights.
- RTB [39] - Adversarial training via multi-step PGD attack.

- LAS [25] - This uses a learnable attack strategy to generate strong attacks for training.

We evaluate base models from these methods on the datasets for which they have released their pretrained models.

B.3 Evaluation Setup

We evaluate our approach added to each of the baselines with a diverse range of attacks including strong adaptive attacks, summarized below. The first three are applied directly to the defended model.

- **AutoAttack** [18] is an ensemble of attacks including two variants of the white-box AutoPGD (APGD) attack with cross-entropy (CE) loss and difference of logits ratio (DLR) loss, white-box fast adaptive boundary (FAB) attack [17], and a black-box Square attack [4]. Note that AutoAttack evaluates the model following a sequence of attacks, and only the samples unsuccessfully perturbed by an attack are given to the next attack. The hyperparameter settings for AutoAttack are followed from RobustBench [15].
- **Image-Wise Worst-Case** (IW-WC) robust accuracy is computed using an ensemble of 9 white-box, black-box, and adaptive attacks that we devised (extending the evaluation of [16]). The adaptive attacks specifically target our weakness of dependence on pseudo-labels. For each image, the attack is considered successful if any one of the 9 attacks is successful. We detail each attack included in the IW-WC robust accuracy computation below:
  1. Standard **AutoAttack** [18] applied to the defended model (*i.e.* including the test-time defense). The hyperparameters are the same as in RobustBench [15].
  2. **RayS** [12] is a black-box attack that searches for a perturbation that yields the lowest decision boundary radius. However, it is computationally very expensive and yet weaker than Square attack. We set the number of queries to 10000 following [12, 16]. This is applied to the defended model.
  3. **APGD with EoT** [5]. Expectation over Transformation (EoT) averages the gradients over multiple iterations of each input to counter the effect of randomness in a defense. We use 20 iterations of EoT with APGD, which is more than $n_s = 1$ used in our randomized smoothing. This is applied directly to the defended model.

- The following 6 attacks are computed on the base model and evaluated on the defended model (*i.e.* these are transfer attacks). Also, while AutoAttack returns the unperturbed samples when the base model produces a misclassification, the following APGD attacks always return perturbed samples that misclassify or produce the highest loss. This is important because test-time defenses tend to misclassify samples closer to the boundary [16].
  4. **Transfer-APGD-CE** attack uses the cross-entropy loss for 100 steps of APGD.

Table 8: Extended comparison with state-of-the-art test-time defenses on CIFAR10 (extending Table 5, main paper). The number in (green) shows the worst-case robustness gain over the base model. The first row of each block is the base model without test-time defense, AA indicates AutoAttack [18], Tr indicates Transfer attack. The **last column (IW-WC)** indicates image-wise worst-case robust accuracy over the 9 attacks.

Base Model	Test-Time Defense	Clean Acc.	Attacks on test-time def.			Attacks transferred from base model						**Robust Acc. (IW-WC)**
			AA	RayS	APGD +EoT	Tr-APGD -CE	Tr-APGD -CW	Tr-APGD -tgt-DLR	Tr-AA	Tr-RayS	Tr-APGD +EoT	
OAAT (RN18)	None	80.23	51.10	56.74	51.61	55.72	51.73	51.10	51.10	56.74	51.61	51.01
	HD	79.89	62.45	73.24	63.20	58.86	55.12	50.76	59.19	71.77	59.46	50.68 (-0.33)
	SODEF	80.23	59.05	74.55	62.55	55.09	52.94	57.80	63.09	74.33	63.45	50.67 (-0.34)
	CAAA	80.23	69.45	56.93	71.49	55.72	51.73	51.12	51.04	56.77	51.61	51.00 (-0.01)
	`CD-IR-50`	79.85	54.79	79.80	55.86	56.69	52.58	53.28	54.61	68.80	54.95	**52.35 (+1.34)**
	`LO-IR-50`	79.90	54.90	79.77	56.23	56.59	52.63	53.28	54.60	68.93	54.93	52.30 (+1.29)
DAJAT (RN18)	None	85.71	52.50	59.94	52.95	56.24	53.26	52.53	52.50	59.94	52.95	52.45
	HD	84.53	64.92	78.22	66.38	60.46	57.20	52.67	61.24	75.73	61.44	52.55 (+0.10)
	SODEF	84.86	64.11	82.06	65.09	56.27	54.60	57.42	61.38	76.41	61.64	52.95 (+0.50)
	CAAA	85.71	75.51	60.14	76.64	56.24	53.26	52.53	52.51	59.96	52.95	52.45 (+0.00)
	`CD-IR-50`	85.11	54.81	85.00	55.59	57.03	54.11	54.20	55.68	72.73	56.04	**53.36 (+0.91)**
	`LO-IR-50`	85.18	54.53	85.08	55.99	57.00	54.17	54.20	55.67	72.14	56.02	53.34 (+0.89)
TR-AWP (WRN)	None	85.36	56.17	61.68	56.42	58.83	56.76	56.25	56.17	61.68	56.42	56.12
	HD	84.78	68.61	78.84	69.51	61.70	60.15	55.79	63.71	76.56	63.87	55.72 (-0.40)
	SODEF	85.25	64.46	79.41	66.57	57.84	57.66	60.30	63.81	75.85	63.92	56.05 (-0.07)
	CAAA	85.35	75.74	61.75	77.47	58.83	56.75	56.25	56.20	61.76	56.43	56.11 (-0.01)
	`CD-IR-50`	84.98	60.26	83.84	61.41	59.73	58.02	57.53	59.43	72.74	59.60	57.46 (+1.34)
	`LO-IR-50`	85.17	59.61	85.03	60.47	59.07	57.32	57.58	59.02	72.98	59.17	**57.88 (+1.76)**

5. **Transfer-APGD-CW** attack uses Carlini-Wagner loss [11] for 100 steps of APGD.
6. **Transfer-Targeted-APGD-DLR** attack uses difference of logits ratio (DLR) loss [18] for 100 steps of targeted APGD with 3 random restarts.
7. **Transfer-AutoAttack** (same hyperparameters as 1).
8. **Transfer-RayS** attack (same hyperparameters as 2).
9. **Transfer-APGD+EoT** attack (same hyperparameters as 3).

B.4 Miscellaneous details

We implement our **IG-Defense** in PyTorch [36] and use the official code releases from RobustBench [15], RayS [12], and Croce et al. [16] for our evaluations. For base models, we use either RobustBench [15] implementations or the official base model authors' code releases. For all experiments, we use a single Nvidia RTX 3090 GPU with 24 GB VRAM, and with an 8-core CPU and 16 GB RAM.

C Experiments

C.1 Extended comparisons

In Table 8, we show the individual evaluations of the proposed image-wise worst-case (IW-WC) ensemble attack. While prior works show improvement over certain attacks, only our **IG-Defense** consistently improves over the base model. Further, we achieve significant gains in the IW-WC robust accuracy while maintaining the lowest inference time.

For existing test-time defenses as well as our **IG-Defense**, we find transfer attacks to be stronger than directly attacking the test-time defended model. Hence, we focus on evaluating against stronger attacks since improvements against weaker attacks are not considered fundamental gains in robustness [16]. Specifically, we find that transfer of targeted APGD-DLR attack and transfer of APGD+EoT are the strongest attacks that break existing test-time defenses (HD, CAAA, SODEF). This is in-line with [16], who show that HD breaks under the targeted APGD-DLR attack, SODEF breaks under an ensemble of transfer-APGD attacks, and CAAA breaks under any transfer attack. We showcase the robustness of **IG-Defense** against these strong adaptive attacks as well as against RayS, transfer-RayS, APGD+EoT, and transfer of APGD+EoT.

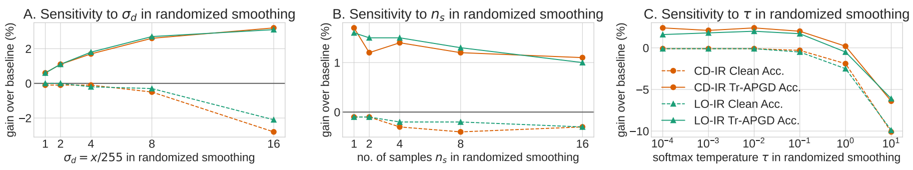

Fig. 6: Sensitivity to randomized smoothing hyperparameters. The base model is DAJAT (ResNet18) [1] for CIFAR10. The legend in **C.** applies to **A.** and **B.** as well.

C.2 Extended analysis

Sensitivity analysis. Due to space constraints in the main paper, we present the sensitivity analyses for the randomized smoothing (RS) hyperparameters in Fig. 6. Recall that RS adds n_s number of noises sampled from a Gaussian distribution with zero mean and standard deviation σ_d. Formally,

$$\hat{y} = \sigma\left(\frac{1}{\tau}\underset{v_i}{\mathbb{E}}[f(x+v_i)]\right); \quad v_i \sim \mathcal{N}(0_d, \sigma_d^2 I_d)\ \forall\, i \in [n_s] \tag{6}$$

where $0_d, I_d$ are tensors with shape same as x but containing all zeros and ones respectively. $\sigma(\cdot)$ is the softmax function, and τ is the softmax temperature term that controls the sharpness of the softmax distribution. Overall, we have three hyperparameters, n_s, σ_d, and τ.

In Fig. 6A, we analyze the sensitivity to σ_d by varying it from $1/255$ to $16/255$ while $n_s = 1$. While the robust accuracy gain increases with σ_d, the clean accuracy drop also increases for both `CD-IR` and `LO-IR`. Hence, we choose $\sigma_d = \frac{\epsilon}{2}$ (*i.e.* $4/255$ in Fig. 6A for CIFAR10) where the clean accuracy drop is minimal with significant robustness gains.

In Fig. 6B, we analyze the sensitivity to number of noises sampled n_s by varying it from 1 to 16 while σ_d is fixed to $4/255$. There is a consistent robustness gain across all n_s. But the clean accuracy drop as well as inference time is lower for a lower n_s. Hence, we choose $n_s = 1$.

In Fig. 6C, we analyze the sensitivity to the softmax temperature τ by varying it from 10^{-4} to 10. We find that a higher softmax temperature leads to a drop in both clean and robust accuracy since the initial pseudo-label may be wrong if the distribution is not sharp enough. Between 10^{-1} to 10^{-4}, we get almost the same performance gains and both `CD-IR` and `LO-IR` are fairly insensitive to τ.

A potential limitation of **IG-Defense** is that our importance ranking methods require access to probing data, which is a general requirement to use neuron interpretability tools. In our experiments, we use the training data as the probing data. Hence, in Fig. 7A and Fig. 7B, we analyze the sensitivity to the amount of data required by our importance ranking methods, `CD-IR` and `LO-IR`. We find that even 10% of the probing data is sufficient for both importance ranking methods to yield robustness gains.

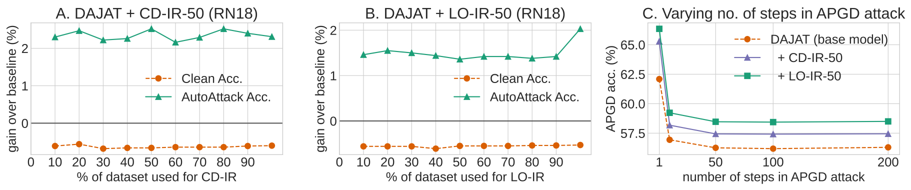

Fig. 7: A, B. Sensitivity to amount of dataset used for our importance ranking methods, `CD-IR` and `LO-IR`. **C.** Varying the number of steps in APGD attack [18] on our **IG-Defense** with DAJAT [1] (ResNet18) for CIFAR10.

Sanity checks for gradient masking.

In Table 9, we evaluate with black-box Square attack and stronger AutoAttack on CIFAR10 with the ResNet18 base model from DAJAT [1]. Specifically, we add a stronger AutoAttack evaluation where APGD uses 200 steps with 10 random restarts (RR) instead of 100 steps and 1 RR, targeted FAB attack also uses 10 RR instead of 1 RR, and Square attack uses 5 RR instead of 1 RR. Compared to the standard AutoAttack, the accuracy reduction for stronger AutoAttack is only marginal for all methods except SODEF (which has $\sim$9% drop). This shows that our methods are strong defenses irrespective of attack strength although SODEF is more vulnerable. Further, the black-box Square attack is less effective compared to the white-box AutoAttack indicating that gradients are useful, *i.e.* not masked. However, gradient masking exists in CAAA and SODEF since the black-box Square (Table 9) and RayS attacks (Table 8) respectively are stronger than the white-box AutoAttack.

In Fig. 8, we compare the loss surfaces or landscapes for the DAJAT [1] ResNet18 base model on CIFAR10 and with our proposed **IG-Defense**. Specifically, the loss is computed for perturbed examples $x' = x + \alpha g + \beta g^{\perp}$ where α, β are varied to cover the entire l_{∞}-norm ϵ-ball, g is the sign of gradient and $g^{\perp}$ is a direction orthogonal to g. We observe that loss surfaces are smooth for both `CD-IR` and `LO-IR`, indicating the absence of gradient masking since even

Table 9: Sanity check for gradient masking on CIFAR10. The number in (green) shows the worst-case robustness gain over the base model. The first row is the base model without test-time defense, AA indicates AutoAttack [18].

Base Model	Test-Time Defense	Clean Acc.	Robust Acc. (AA)	Robust Acc. (Stronger AA)	Robust Acc. (Square)	Robust Acc. (IW-WC)
DAJAT [1] RN18	None	85.71	52.50	52.45	59.23	52.45
	CAAA [3]	85.71	75.51	74.61	78.51	52.45 (+0.00)
	SODEF [26]	84.44	64.11	55.09	59.54	52.95 (+0.50)
	`CD-IR-50` (*Ours*)	85.11	54.81	53.87	61.76	**53.36 (+0.91)**
	`LO-IR-50` (*Ours*)	85.18	54.53	53.93	62.22	53.34 (+0.89)

a brute-force search in the ϵ-ball does not yield a higher loss than along the gradient direction.

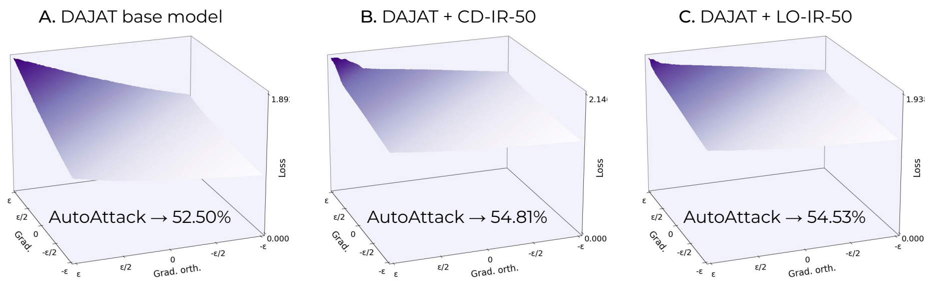

Fig. 8: Comparison of loss surfaces for DAJAT [1] and DAJAT with our **IG-Defense** at test-time.

In Fig. 7C, we experiment with single-step ($i.e.$ FGSM) and multi-step APGD attacks. We see consistent gains in robustness irrespective of number of attack steps. Multi-step attacks are stronger than single-step attack, indicating that gradients are reliable. Further, as number of steps are increased, robust accuracy saturates similar to the base model, indicating that gradients are as useful as in the base model.

Table 10: Varying ℓ_∞-bound ϵ of AutoAttack for ResNet18 with CIFAR10. We omit clean accuracy since it does not change with ϵ.

ϵ	DAJAT [1]	+`CD-IR-50`	+`LO-IR-50`
4/255	71.10	71.63 (+0.53)	71.44 (+0.34)
8/255	52.50	54.81 (+2.31)	54.53 (+2.03)
16/255	17.93	20.00 (+2.07)	19.75 (+1.82)
32/255	0.52	0.71 (+0.19)	0.72 (+0.20)

In Table 10, we vary the ℓ_∞-bound ϵ of AutoAttack from 4/255 to 32/255. As ϵ is increased, the base model and our defense robustness both can be reduced to almost zero. This again indicates that gradients are reliable.

Table 11: Analysis of class-wise accuracies for best and worst classes for the DAJAT [1] ResNet18 base model on CIFAR10

	Classwise Average		Best-Class		Worst-Class	
	Clean Acc.	Rob. Acc. (AA)	Clean Acc.	Rob. Acc. (AA)	Clean Acc.	Rob. Acc. (AA)
No test-time def.	**85.71**	52.50	**94.4**	74.7	**72.3**	25.7
`CD-IR`-50 *(Ours)*	85.11	**54.81**	94.3	76.5	71.8	**28.8**
`LO-IR`-50 *(Ours)*	85.18	54.53	94.2	**76.6**	72.1	28.5

Classwise disparity in adversarially-trained models.

Prior works [44, 52] have shown that adversarial training leads to significant disparity in class-wise robustness, which is also known as the robust fairness issue. We investigate this by evaluating the class-wise robust accuracies and report the best and worst class accuracies in Table 11. We observe similar (though relatively less severe) disparity results. It can also be seen that our test-time defenses yield uniform robustness gains across classes.

C.3 Efficiency analysis

We present the efficiency analysis for our proposed importance ranking methods `CD-IR` and `LO-IR` in Table 12. For `CD-IR`, most of the operations are vectorized and the training dataset is passed through the CLIP image encoder and the base model once while saving the CLIP embeddings and base model activations. Then, the similarity can be computed easily with the saved activations and we observe that `CD-IR` can be performed very quickly (~2 minutes).

For `LO-IR`, we need to compute the average logit change when masking each neuron separately. Given that each computation is independent and inference requires a very low amount of GPU memory (as it does not require backpropagation), we parallelize this into 8 threads. With this, we can perform `LO-IR` for 512 neurons in ~28 minutes.

A potential concern about compute cost could be due to hyperparameter search required for the number of retained neurons k. However, we observe that the compute cost to get k is actually not very high. This is because our importance ranking is independent of k, *i.e.* it only needs to be done once. Then, we can use binary search to find k, which requires evaluating clean accuracy (~2 mins per k). This is much cheaper than computing the robust accuracy (~6 hrs per k). Because, as a heuristic, we usually find a robustness gain when the clean accuracy is close to that of the base model.

Table 12: Efficiency analysis of our proposed importance ranking methods `CD-IR` and `LO-IR` for layer4 (with 512 neurons) of a ResNet18 base model and CIFAR10. Both experiments performed with a single RTX 3090 GPU with 24 GB VRAM, 8-core CPU, and 16 GB RAM.

Method	Time taken
`LO-IR`	~28 mins.
`CD-IR`	~2 mins.

References

1. Addepalli, S., Jain, S., Babu, R.V.: Efficient and effective augmentation strategy for adversarial training. In: NeurIPS (2022) 5, 10, 11, 12, 16, 19, 20, 21, 22
2. Addepalli, S., Jain, S., Sriramanan, G., Venkatesh Babu, R.: Scaling adversarial training to large perturbation bounds. In: ECCV (2022) 5, 9, 10, 11, 12, 16
3. Alfarra, M., Pérez, J.C., Thabet, A., Bibi, A., Torr, P.H.S., Ghanem, B.: Combating adversaries with anti-adversaries. In: AAAI (2022) 1, 3, 12, 14, 21
4. Andriushchenko, M., Croce, F., Flammarion, N., Hein, M.: Square attack: a query-efficient black-box adversarial attack via random search. In: ECCV (2020) 10, 17
5. Athalye, A., Carlini, N., Wagner, D.: Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In: ICML (2018) 7, 10, 11, 17
6. Bai, N., Iyer, R.A., Oikarinen, T., Weng, T.W.: Describe-and-dissect: Interpreting neurons in vision networks with language models. arXiv preprint arXiv:2403.13771 (2024) 3
7. Bai, Y., Zeng, Y., Jiang, Y., Xia, S.T., Ma, X., Wang, Y.: Improving adversarial robustness via channel-wise activation suppressing. In: ICLR (2021) 3, 15
8. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: CVPR (2017) 2, 3
9. Bau, D., Zhu, J.Y., Strobelt, H., Lapedriza, A., Zhou, B., Torralba, A.: Understanding the role of individual units in a deep neural network. Proceedings of the National Academy of Sciences (2020) 3, 4
10. Boopathy, A., Liu, S., Zhang, G., Liu, C., Chen, P.Y., Chang, S., Daniel, L.: Proper network interpretability helps adversarial robustness in classification. In: ICML (2020) 3
11. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: IEEE Symposium on Security and Privacy (2017) 10, 18
12. Chen, J., Gu, Q.: RayS: A ray searching method for hard-label adversarial attack. In: KDD (2020) 10, 17, 18
13. Chen, Y., Oliver, D.S.: Ensemble randomized maximum likelihood method as an iterative ensemble smoother. Mathematical Geosciences **44**, 1–26 (2012) 7
14. Chen, Z., Li, Q., Zhang, Z.: Towards robust neural networks via close-loop control. In: ICLR (2021) 1, 3
15. Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., Hein, M.: Robustbench: a standardized adversarial robustness benchmark. In: Datasets and Benchmarks Track, NeurIPS (2021) 6, 9, 17, 18

16. Croce, F., Gowal, S., Brunner, T., Shelhamer, E., Hein, M., Cemgil, T.: Evaluating the adversarial robustness of adaptive test-time defenses. In: ICML (2022) 2, 9, 10, 11, 13, 14, 17, 18, 19
17. Croce, F., Hein, M.: Minimally distorted adversarial examples with a fast adaptive boundary attack. In: ICML (2020) 10, 17
18. Croce, F., Hein, M.: Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In: ICML (2020) 10, 17, 18, 20, 21
19. Eigen, H., Sadovnik, A.: TopKConv: Increased adversarial robustness through deeper interpretability. In: ICMLA (2021) 3, 15
20. Gerasimou, S., Eniser, H.F., Sen, A., Cakan, A.: Importance-driven deep learning system testing. In: ACM/IEEE ICSE (2020) 3
21. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: ICLR (2015) 1
22. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016) 9
23. Hernandez, E., Schwettmann, S., Bau, D., Bagashvili, T., Torralba, A., Andreas, J.: Natural language descriptions of deep visual features. In: ICLR (2022) 3
24. Hwang, D., Lee, E., Rhee, W.: AID-Purifier: A light auxiliary network for boosting adversarial defense. In: ICPR (2022) 3
25. Jia, X., Zhang, Y., Wu, B., Ma, K., Wang, J., Cao, X.: LAS-AT: adversarial training with learnable attack strategy. In: CVPR (2022) 10, 11, 17
26. Kang, Q., Song, Y., Ding, Q., Tay, W.P.: Stable neural ODE with Lyapunov-stable equilibrium points for defending against adversarial attacks. In: NeurIPS (2021) 1, 12, 13, 14, 21
27. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., CIFAR (2009) 9
28. Kundu, S., Nazemi, M., Beerel, P.A., Pedram, M.: A tunable robust pruning framework through dynamic network rewiring of DNNs. In: ASP-DAC (2021) 3, 15
29. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: ICLR (2018) 16
30. Mangla, P., Singh, V., Balasubramanian, V.N.: On saliency maps and adversarial robustness. In: ECML-PKDD (2020) 3
31. Mao, C., Chiquier, M., Wang, H., Yang, J., Vondrick, C.: Adversarial attacks are reversible with natural supervision. In: ICCV (2021) 1, 3
32. Mu, J., Andreas, J.: Compositional explanations of neurons. In: NeurIPS (2020) 4
33. Nayak, G.K., Rawal, R., Chakraborty, A.: DAD: Data-free adversarial defense at test time. In: WACV (2022) 3
34. Oikarinen, T., Weng, T.W.: CLIP-Dissect: Automatic description of neuron representations in deep vision networks. In: ICLR (2023) 2, 3, 9
35. Oikarinen, T., Weng, T.W.: Linear explanations for individual neurons. In: ICML (2024) 3
36. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019) 18
37. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: ICML (2021) 3, 9

38. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015) 9
39. Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., Madry, A.: Do adversarially robust imagenet models transfer better? In: NeurIPS (2020) 9, 10, 11, 16
40. Sehwag, V., Wang, S., Mittal, P., Jana, S.: Hydra: Pruning adversarially robust neural networks. In: NeurIPS (2020) 3, 15
41. Shi, C., Holtz, C., Mishne, G.: Online adversarial purification based on self-supervised learning. In: ICLR (2021) 1, 3
42. Sriramanan, G., Addepalli, S., Baburaj, A., Babu, R.V.: Guided adversarial attack for evaluating and enhancing adversarial defenses. In: NeurIPS (2020) 1, 5, 9, 16
43. Sriramanan, G., Addepalli, S., Baburaj, A., Babu, R.V.: Towards efficient and effective adversarial training. In: NeurIPS (2021) 5, 9, 10, 16
44. Wei, Z., Wang, Y., Guo, Y., Wang, Y.: CFA: Class-wise calibrated fair adversarial training. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8193–8201 (2023) 22
45. Wong, E., Rice, L., Kolter, J.Z.: Fast is better than free: Revisiting adversarial training. In: ICLR (2020) 1, 9, 11, 16
46. Wu, B., Pan, H., Shen, L., Gu, J., Zhao, S., Li, Z., Cai, D., He, X., Liu, W.: Attacking adversarial attacks as a defense. arXiv preprint arXiv:2106.04938 (2021) 12, 14
47. Wu, D., Xia, S.T., Wang, Y.: Adversarial weight perturbation helps robust generalization. In: NeurIPS (2020) 1, 5, 10, 12, 14, 16
48. Wu, S., Sang, J., Xu, K., Zhang, J., Yu, J.: Attention, please! adversarial defense via activation rectification and preservation. ACM Transactions on Multimedia Computing, Communications and Applications **19**(4), 1–18 (2023) 3
49. Wu, Y.H., Yuan, C.H., Wu, S.H.: Adversarial robustness via runtime masking and cleansing. In: ICML (2020) 3
50. Xiao, C., Zhong, P., Zheng, C.: Enhancing adversarial defense by k-winners-take-all. In: ICLR (2020) 3, 15
51. Xie, X., Li, T., Wang, J., Ma, L., Guo, Q., Juefei-Xu, F., Liu, Y.: NPC: Neuron path coverage via characterizing decision logic of deep neural networks. ACM Transactions on Software Engineering and Methodology **31**(3) (2022) 3
52. Xu, H., Liu, X., Li, Y., Jain, A., Tang, J.: To be robust or to be fair: Towards fairness in adversarial training. In: ICML (2021) 22
53. Yoon, J., Hwang, S.J., Lee, J.: Adversarial purification with score-based generative models. In: ICML (2021) 3
54. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: BMVC (2016) 9
55. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: ECCV (2014) 7, 14
56. Zhang, H., Yu, Y., Jiao, J., Xing, E.P., Ghaoui, L.E., Jordan, M.I.: Theoretically principled trade-off between robustness and accuracy. In: ICML (2019) 1, 10, 16
57. Zhao, Q., Wressnegger, C.: Holistic adversarially robust pruning. In: ICLR (2023) 3, 15