Improving the Accuracy-Robustness Trade-off of Classifiers via Adaptive Smoothing

Yatong Bai[1], Brendon G. Anderson[1], Aerin Kim[2], Somayeh Sojoudi[1]
[1] *University of California, Berkeley* [2] *Scale AI*
{yatong_bai, bganderson, sojoudi}@berkeley.edu, aerinykim@gmail.com

– ***Abstract*** –

While it is shown in the literature that simultaneously accurate and robust classifiers exist for common datasets, previous methods that improve the adversarial robustness of classifiers often manifest an accuracy-robustness trade-off. We build upon recent advancements in data-driven "locally biased smoothing" to develop classifiers that treat benign and adversarial test data differently. Specifically, we tailor the smoothing operation to the usage of a robust neural network as the source of robustness. We then extend the smoothing procedure to the multi-class setting and adapt an adversarial input detector into a policy network. The policy adaptively adjusts the mixture of the robust base classifier and a standard network, where the standard network is optimized for clean accuracy and is not robust in general. We provide theoretical analyses to motivate the use of the adaptive smoothing procedure, certify the robustness of the smoothed classifier under realistic assumptions, and justify the introduction of the policy network. We use various attack methods, including AutoAttack and adaptive attack, to empirically verify that the smoothed model noticeably improves the accuracy-robustness trade-off. On the CIFAR-100 dataset, our method simultaneously achieves an 80.09% clean accuracy and a 32.94% AutoAttacked accuracy. The code that implements adaptive smoothing is available at https://github.com/Bai-YT/AdaptiveSmoothing.

1. Introduction

The vulnerability of neural networks to adversarial attacks has been observed in various applications, such as computer vision [25, 44] and control systems [31]. In response, "adversarial training" [12, 13, 25, 36, 62] has been studied to alleviate the susceptibility. Adversarial training builds robust neural networks by training on adversarial examples.

A parallel line of work focuses on certified robustness. There are a number of techniques that provide robustness certifications to existing neural networks [5, 6, 41]. Among these methods, "randomized smoothing" seeks to achieve certified robustness at test time [19, 38, 49]. The recent work [7] has shown that a locally biased smoothing method provides an improvement over the traditional data-blind randomized smoothing. However, [7] only focuses on binary classification problems, significantly limiting the applications. Moreover, the method has a fixed balance parameter between clean accuracy (accuracy on clean data without attack) and adversarial robustness, and an accuracy-robustness trade-off thus limits its performance.

While some works have shown that there exists a fundamental trade-off between accuracy and robustness [57, 61], recent research has argued that it should be possible to simultaneously achieve robustness and accuracy on benchmark datasets [59]. To this end, variants of adversarial training that improve the accuracy-robustness trade-off have been proposed, including TRADE [61], Interpolated Adversarial Training (IAT) [37], and many others [11, 14, 50, 56, 58, 60]. However, even with these improvements, clean accuracy is often an inevitable price of achieving robustness. Moreover, standard non-robust models often take advantage of pre-training on larger datasets, gaining enormous performance gains, whereas the effect of pre-training on robust classifiers is less understood and may be less prominent [18, 23].

This work makes a theoretically disciplined step towards performing robust classification without sacrificing clean accuracy, with the contributions summarized below.

- In Section 3, under the observation that the perfor-

mance of the K-nearest-neighbor (K-NN) classifier, a crucial component of locally biased smoothing, becomes a bottleneck of the overall performance, we replace the K-NN classifier with a robust neural network that can be obtained via various existing methods, and modify the smoothing formulation accordingly. The resulting formulation (4) is a convex combination of the outputs of a standard neural network and a robust neural network. When the robust neural network has a certified Lipschitz constant, the combined classifier also has a closed-form certified robust radius.

- In Section 4, we propose a data-aware adaptive smoothing procedure that adaptively adjusts the mixture of a standard model and a robust model with the help of a policy network. This procedure uses a type of adversary detector as a policy network that adjusts the convex combination of the two networks, improving the accuracy-robustness trade-off. We then empirically verify the robustness of the proposed method using gray-box and white-box projected gradient descent (PGD) attack, AutoAttack, and adaptive attack, demonstrating that the policy network is robust against the types of adversaries it is trained with. When the policy is trained with examples generated by a carefully-constructed adaptive AutoAttack, the composite model sacrifices little robustness but significantly enhances the clean accuracy, demonstrating a significantly improved accuracy-robustness trade-off.

Note that we do not make any assumptions about how the standard and robust base models are obtained, nor does the method make assumptions on the type and budget of the adversarial attack. Thus, adaptive smoothing can take advantage of pre-trained weights via the standard base classifier and benefit from ever-improving robust training methods via the robust base classifier.

2. Background and related works

2.1. Notations

The symbol $\|\cdot\|_p$ denotes the ℓ_p norm of a vector, while $\|\cdot\|_{p*}$ denotes its dual norm. The matrix I_d denotes the identity matrix in $\mathbb{R}^{d\times d}$. For a scalar a, $\text{sgn}(a) \in \{-1, 0, 1\}$ denotes its sign. For a natural number c, $[c] = \{1, 2, \ldots, c\}$. For an event A, the indicator function $\mathbb{I}(A)$ evaluates to 1 if A takes place and 0 otherwise. The notation $\mathbb{P}_{X\sim\mathcal{S}}[A(X)]$ denotes the probability for an event $A(X)$ to occur, where X is a random variable drawn from the distribution $\mathcal{S}$.

Consider a model $g : \mathbb{R}^d \to \mathbb{R}^c$, whose components are $g_i : \mathbb{R}^d \to \mathbb{R}$, $i \in [c]$, where d is the dimension of the input and c is the number of classes. A classifier $f : \mathbb{R}^d \to [c]$ can be obtained via $f(x) \in \arg\max_{i\in[c]} g_i(x)$. In this paper, we assume that $g(\cdot)$ does not have the desired level of robustness, and refer to it as a "standard classifier" (as opposed to a "robust classifier" which we denote as $h(\cdot)$). We use $\mathcal{D}$ to denote the set of all validation input-label pairs (x_i, y_i).

In this work, we consider ℓ_p-norm-bounded attacks on differentiable neural networks. A classifier $f(\cdot)$ is considered robust against adversarial perturbations at an input data $x \in \mathbb{R}^d$ if it assigns the same label to all perturbed inputs $x + \delta$ such that $\|\delta\|_p \le \epsilon$, where $\epsilon \ge 0$ is the attack radius.

2.2. Related adversarial attacks and defenses

While the fast gradient sign method (FGSM) and PGD attacks based on differentiating the cross-entropy loss have been considered the most classic and straightforward attacks [25, 42], they have been shown to be too weak, as defenses that are only designed against the FGSM and PGD attacks are often easily circumvented [9, 10, 16, 48]. To this end, various attack methods based on alternative loss functions, Expectation Over Transformation, and black-box perturbations have been proposed. Such efforts include MultiTargeted attack loss [28], AutoAttack [22], adaptive attack [55], minimal distortion attack [21], and many others, even considering attacking test-time defenses [20].

On the defense side, while adversarial training [42] and TRADE [61] have seen enormous success, such methods are often limited by a significantly larger amount of required training data [52]. Initiatives that construct more effective training data via data augmentation [26, 27, 51] and generative models [53] have successfully produced more robust models. Improved versions of adversarial training [32, 54] have also been proposed.

Previous research has developed models that improve robustness by dynamically changing at test time. Specifically, "Input-Adaptive Inference (IAI)" improves the accuracy-robustness trade-off by appending side branches to a single network, allowing for early-exit predictions [30]. Other initiatives that aim to enhance the accuracy-robustness trade-off include using the SCORE attack during training [46] and applying adversarial training for regularization [63]. Moreover, ensemble-based defenses, such as random ensemble [39] and diverse

ensemble [2, 47], have been proposed. In comparison, this work considers two separate classifiers and uses their synergy to improve the accuracy-robustness trade-off, achieving much higher performances.

2.3. Locally biased smoothing

Randomized smoothing, popularized by [19], achieves robustness at test time by replacing $f(x)$ with a smoothed classifier, given by $\widetilde{f}(x) \in \arg\max_{i\in[c]} \mathbb{P}_{\delta\sim\mathcal{S}}\big[f(x+\delta)=i\big]$, where $\mathcal{S}$ is a smoothing distribution. A common choice for $\mathcal{S}$ is a Gaussian distribution.

The authors of [7] have recently argued that data-invariant randomized smoothing does not always achieve robustness. They have shown that in the binary classification setting, randomized smoothing with an unbiased distribution is suboptimal, and an optimal smoothing procedure shifts the input point in the direction of its true class. Since the true class is generally unavailable, a "direction oracle" is used as a surrogate. This "locally biased smoothing" method is no longer randomized and outperforms traditional data-blind randomized smoothing. The locally biased smoothed classifier $g^\gamma(\cdot)$ is obtained via the deterministic calculation

$$g^\gamma(x) = g(x) + \gamma h(x)\|\nabla g(x)\|_{p*},$$

where $h(x) \in \{-1, 1\}$ is the direction oracle and $\gamma \geq 0$ is a trade-off parameter. Since locally biased smoothing aims to improve robustness, the direction oracle should come from an inherently robust classifier (which is often less accurate). In [7], this direction oracle is chosen to be a one-nearest-neighbor classifier. Intuitively, when $\|\nabla g(x)\|_{p*}$ is large, $g(x)$ is more susceptible to adversarial attacks because perturbing the input by the same amount induces a larger output change. Thus, when $\|\nabla g(x)\|_{p*}$ is large, locally biased smoothing trusts the direction oracle more.

2.4. Adversarial input detectors

It has been shown that adversarial inputs can be detected via various methods. For example, [43] proposes to append an additional detection branch to an existing neural network, and uses adaptive adversarial data to train the detector in a supervised fashion. However, [15] has shown that it is possible to bypass this detection method. They constructed adversarial examples via the C&W attacks [16] and simultaneously targeted the classification branch and the detection branch by treating the two branches as an "augmented classifier". According to [15], the detector is effective against the types of attack that it is trained with, but not necessarily the attack types that are absent in the training data. It is thus reasonable to expect the detector to be able to detect a wide range of attacks if it is trained using sufficiently diverse types of attacks (including those targeting the detector itself). While exhaustively covering the entire adversarial input space is intractable, and it is unclear to what degree one needs to diversify the attack types in practice, our experiments show that our modified architecture based on [43] can recognize the state-of-the-art AutoAttack adversaries with a high success rate.

To mitigate the above challenges faced by detectors obtained via supervised training, unsupervised detectors have been proposed [3, 4]. Other detection methods include [1, 17]. Unfortunately, universally effective detectors have not been discovered yet, and therefore this paper focuses on transferring the properties of the existing detector towards better overall robustness. Future advancements in the field of adversary detection can further enhance the performance of our method.

3. Using a robust neural network as the smoothing oracle

Since locally-biased smoothing was designed for binary classification problems, we first extend it to the multi-classification setting. To achieve this, we treat the output of each class $g_i(x)$ independently, giving rise to:

$$g^\gamma_{\text{smoothed1},i}(x) = g_i(x) + \gamma h_i(x)\|\nabla g_i(x)\|_{p*}, \quad i \in [c]. \tag{1}$$

Note that if $\|\nabla g_i(x)\|_{p*}$ is large for some i, then $g^\gamma_{\text{smoothed1},i}(x)$ can be large even if both $g_i(x)$ and $h_i(x)$ are small, potentially leading to incorrect predictions. To remove the effect of the magnitude difference across the classes, we propose a normalized formulation as follows:

$$g^\gamma_{\text{smoothed2},i}(x) = \frac{g_i(x) + \gamma h_i(x)\|\nabla g_i(x)\|_{p*}}{1 + \gamma\|\nabla g_i(x)\|_{p*}}, \quad i \in [c]. \tag{2}$$

The parameter γ adjusts the trade-off between clean accuracy and robustness. When $\gamma = 0$, it holds that $g^\gamma_{\text{smoothed2},i}(x) \equiv g_i(x)$ for all i. When $\gamma \to \infty$, it holds that $g^\gamma_{\text{smoothed2},i}(x) \to h_i(x)$ for all x and all i.

With the smoothing procedure generalized to the multi-class setting, we are now ready to discuss the choice of the robust oracle $h_i(\cdot)$. While K-NN classifiers are relatively robust and can be used as the direction oracle, the representation power of K-NN classifiers is too

weak. On the CIFAR-10 image classification problem [35], K-NN only achieves around 35% accuracy on clean test data. In contrast, an adversarially trained ResNet can reach a 50.0% accuracy on adversarial test data [42]. Such a lackluster performance of K-NN becomes a significant bottleneck of the accuracy-robustness trade-off of the smoothed classifier. To this end, we replace the K-NN classifier with a robust neural network. The robustness of this network can be achieved via various methods, including adversarial training, TRADE, and traditional randomized smoothing.

Further scrutinizing (2) leads to the question of whether $\|\nabla g_i(x)\|_{p*}$ is the best choice for adjusting the mixture of $g(\cdot)$ and $h(\cdot)$. In fact, this gradient magnitude term is a result of the assumption of $h(x) \in \{-1, 1\}$, which is the setting considered in [7]. Here, we no longer have this assumption. Instead, we assume both $g(\cdot)$ and $h(\cdot)$ to be differentiable. Thus, we further generalize the formulation as

$$g^{\gamma}_{\text{smoothed3},i}(x) = \frac{g_i(x) + \gamma R_i(x) h_i(x)}{1 + \gamma R_i(x)}, \quad i \in [c], \quad (3)$$

where $R_i(x)$ is an extra scalar term that can potentially include $\nabla g_i(x)$ and $\nabla h_i(x)$ to determine the "trustworthiness" of the base classifiers. Here, we empirically compare four options for $R_i(x)$: 1, $\|\nabla g_i(x)\|_{p*}$, $\|\nabla \max_j g_j(x)\|_{p*}$, and $\frac{\|\nabla g_i(x)\|_{p*}}{\|\nabla h_i(x)\|_{p*}}$.

Another design question is whether $g(\cdot)$ and $h(\cdot)$ should be the pre-softmax logits or the post-softmax probabilities. Note that since most attack methods are designed based on the logits, incorporating the softmax function into the model may result in gradient masking, an undesired phenomenon that makes it hard to properly evaluate the proposed method. Therefore, we are left with the following two choices that will make the smoothed model compatible with existing gradient-based attacks:

- Use the logits for both $g(\cdot)$ and $h(\cdot)$;
- Use the probabilities for both $g(\cdot)$ and $h(\cdot)$, and then convert the smoothed probabilities back to logits. The required "inverse-softmax" operator is given simply by the natural logarithm, and does not change the overall prediction.

In Figure 1, we compare the different choices for $R_i(x)$ by visualizing the accuracy-robustness trade-off. Based on this "clean accuracy versus PGD_{10}-attacked accuracy" plot (PGD_T denotes T-step PGD), we conclude that $R_i(x) = 1$ gives the best accuracy-robustness trade-off, and $g(\cdot)$ and $h(\cdot)$ should be the probabilities. While

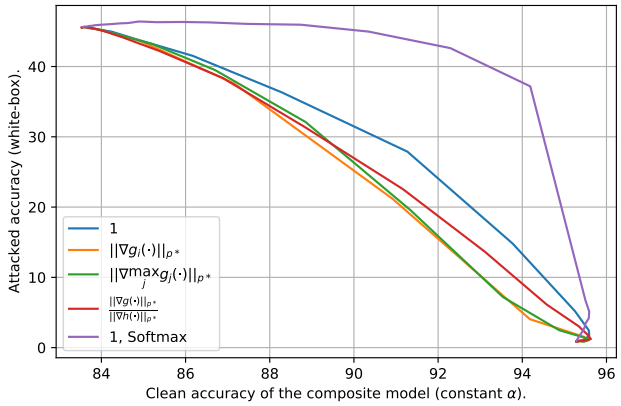


Figure 1. Comparing the options for $R_i(x)$. "Softmax" represents the formulation that use the probabilities for $g(\cdot)$ and $h(\cdot)$ followed by a natural log on $g^{\gamma}_{\text{smoothed3},i}(\cdot)$.

Figure 1 only considers one set of base classifiers (a pair of standard and adversarially-trained ResNet18s), we provide additional examples in Appendix B using alternative model architectures, different methods to train robust base classifiers, and various attack budgets. Note that the resulting formulation can then be re-parameterized as

$$g^{\alpha}_{\text{CNN},i}(x) = \log\big((1-\alpha) g_i(x) + \alpha h_i(x)\big), \quad i \in [c], \quad (4)$$

where $\alpha = \frac{\gamma}{1+\gamma} \in [0, 1]$. Therefore, we select (4) as our formulation of Adaptive Smoothing, which builds a composite classifier $g^{\alpha}_{\text{CNN}}(\cdot)$ that outputs the natural log of a convex combination of the probabilities of $g(\cdot)$ and $h(\cdot)$.

3.1. Theoretical certified robust radius

Similar to local biased smoothing, adaptive smoothing provides a certified robust radius when the base classifier $h(\cdot)$ is certifiably robust, with the robust radius depending on the constant α. We present this theoretical result below:

Definition 1. A function $f\colon \mathbb{R}^d \to \mathbb{R}$ is called *ℓ_p-Lipschitz continuous* if there exists $L \in (0, \infty)$ such that $|f(x') - f(x)| \leq L\|x' - x\|_p$ for all $x', x \in \mathbb{R}^d$. The *Lipschitz constant* of such f is defined to be $\text{Lip}_p(f) := \inf\{L \in (0, \infty) : |f(x') - f(x)| \leq L\|x' - x\|_p \text{ for all } x', x \in \mathbb{R}^d\}$.

Let $\alpha \in (0, 1)$. Consider the convex combination classifier introduced in the adaptive smoothing formulation (4). Since we use probabilities for both $g(\cdot)$ and $h(\cdot)$, it holds that $0 \leq g_i(\cdot) \leq 1$ and $0 \leq h_i(\cdot) \leq 1$ for all i.

Assumption 1. The classifier $h(\cdot)$ is robust in the sense that, for all $i \in \{1, 2, \ldots, n\}$, $h_i(\cdot)$ is Lipschitz continuous with Lipschitz constant $\mathrm{Lip}_p(h_i) \in (0, \infty)$.

Assumption 1 is not restrictive in practice. For example, randomized smoothing with Gaussian smoothing variance $\sigma^2 I_d$ on the input yields robust classifiers with ℓ_2-Lipschitz constant $\sqrt{\frac{2}{\pi\sigma^2}}$.

Theorem 1. *Let $x \in \mathbb{R}^d$ and let $i, j \in \{1, 2, \ldots, n\}$. Then the relation $\mathrm{sgn}(g^{\alpha}_{CNN,i}(x+\delta) - g^{\alpha}_{CNN,j}(x+\delta)) = \mathrm{sgn}(h_i(x) - h_j(x))$ holds for all $\delta \in \mathbb{R}^d$ such that*

$$\|\delta\|_p \leq r^{\alpha}_p(x) \coloneqq \frac{\alpha\,|h_i(x) - h_j(x)| + \alpha - 1}{\alpha\left(\mathrm{Lip}_p(h_i) + \mathrm{Lip}_p(h_j)\right)}.$$

The proof of Theorem 1 is given in Appendix A.1. We remark that the ℓ_p-norm that we certify using Theorem 1 may be arbitrary (e.g., ℓ_1, ℓ_2, or ℓ_∞), so long as the Lipschitz constant of the robust network $h(\cdot)$ is computed with respect to the same norm.

Notice that, if $\alpha \to 1$, then $r^{\alpha}(x) \to \frac{|h_i(x) - h_j(x)|}{\mathrm{Lip}_p(h_i) + \mathrm{Lip}_p(h_j)}$, which is the standard (global) Lipschitz-based robust radius of $h(\cdot)$ around x (see, e.g., [24, 29] for further discussions on Lipschitz-based robustness). On the other hand, if α is too small in comparison to the relative confidence of $h(\cdot)$, namely,

$$\alpha \leq \frac{1}{1 + |h_i(x) - h_j(x)|},$$

then $r^{\alpha}(x) < 0$, and in this case we cannot provide nontrivial certified robustness for $g^{\alpha}_{\mathrm{CNN}}(\cdot)$. This is rooted in the fact that a tiny α amounts to an excess weight into the non-robust classifier $g(\cdot)$. If $h(\cdot)$ is 100% confident in its prediction, then $|h_i(x) - h_j(x)| = 1$, and therefore this threshold value of α becomes $\frac{1}{2}$, leading to a nontrivial certified radii for $\alpha > \frac{1}{2}$. However, once we put over $\frac{1}{2}$ of the weight into $g(\cdot)$, we no longer certify a nonzero radius around x. Again, this is intuitive since we have made no assumptions on the robustness of $g(\cdot)$ around x.

The above result clearly generalizes to the even less restrictive scenario of using local Lipschitz constants over a neighborhood $\mathcal{U}$ of x as a surrogate for the global Lipschitz constants, so long as the condition $\delta \in \mathcal{U}$ is also added to the hypotheses.

4. Adaptive smoothing strength with the policy network

So far, α has been treated as a fixed hyperparameter. A more intelligent approach is to allow α to be different for each x by replacing the constant α with a function $\alpha(x)$. Here, we take $\alpha(x)$ to be deterministic, as stochastic defenses can be much harder to evaluate.

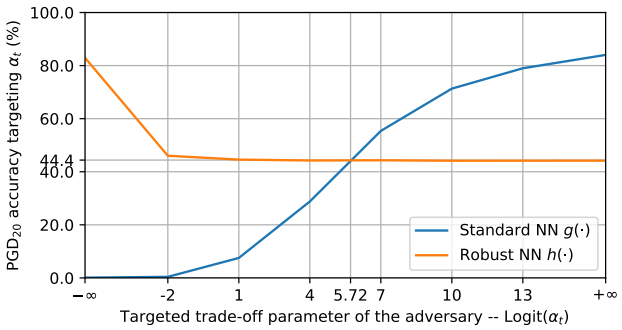


Figure 2. Attacked accuracy of the standard classifier $g(\cdot)$ and the robust classifier $h(\cdot)$ when the adversary targets different values of α_t. For better readability, we use $\mathrm{Logit}(\alpha_t)$ as the horizontal axis labels, where $\mathrm{Logit}(\cdot)$ denotes the inverse function of Sigmoid.

One motivation for adopting the adaptive trade-off parameter $\alpha(x)$ is that the optimal $\alpha^\star$ can vary when x changes. For example, when x is clean and unperturbed, the standard model $g(\cdot)$ outperforms the robust model $h(\cdot)$. If x is an attacked input targeting $g(\cdot)$, then the robust model $h(\cdot)$ should be used. However, as shown in Figure 2, if the target of the attack is $h(\cdot)$, then even though $h(\cdot)$ is robust, a better choice is to feed x into $g(\cdot)$. This is because the loss landscapes of $g(\cdot)$ and $h(\cdot)$ differ enough that an adversarial perturbation targeting $h(\cdot)$ is benign to $g(\cdot)$.

When the PGD adversary targets a smoothed classifier $g^{\alpha_t}_{\mathrm{CNN}}(\cdot)$, as α_t varies, the optimal strategy also changes. We provide a visualization in Figure 2 based on the CIFAR-10 dataset. Specifically, we put together a composite model $g^{\alpha_t}_{\mathrm{CNN}}(\cdot)$ using a ResNet18 standard classifier $g(\cdot)$ and a ResNet18 robust classifier $h(\cdot)$ via (4).[1] Then, we attack $g^{\alpha_t}_{\mathrm{CNN}}(\cdot)$ with different values of α_t via PGD_{20}, save the adversarial instances, and report the accuracy of $g(\cdot)$ and $h(\cdot)$ evaluated on these instances. When $\alpha_t \leq \mathrm{Sigmoid}(5.72) = 0.9967$, the robust model $h(\cdot)$ performs better. When $\alpha_t > 0.9967$, the standard model $g(\cdot)$ is more suitable.

4.1. The existence of $\alpha(x)$ that achieves the trade-off

The following theorem shows that when α is a function of the input, there exists an $\alpha(\cdot)$ that makes the combined classifier correct whenever either $g(\cdot)$ and $h(\cdot)$

[1] The ResNet classifiers are obtained from [45].

makes the correct prediction, which further implies that the combined classifier matches the clean accuracy of $g(\cdot)$ and the attacked accuracy of $h(\cdot)$.

Theorem 2. *Let $\epsilon > 0$, $(x_1, y_1), (x_2, y_2) \sim \mathcal{D}$, and $y_1 \neq y_2$ (i.e., each input corresponds to a unique true label). Assume that $h_i(\cdot)$, $\|\nabla h_i(\cdot)\|_{p*}$, and $\|\nabla g_i(\cdot)\|_{p*}$ are all bounded and that there does not exist $z \in \mathbb{R}^d$ such that $\|z - x_1\|_p \leq \epsilon$ and $\|z - x_2\|_p \leq \epsilon$. Then, there exists a function $\alpha(\cdot)$ such that the assembled classifier $g^{\alpha}_{CNN}(\cdot)$ satisfies*

$$\mathbb{P}_{(x,y)\sim\mathcal{D},\delta\sim\mathcal{F}}\Big[\arg\max_{i\in[c]} g^{\alpha}_{CNN,i}(x+\delta) = y\Big] \geq \max\left\{\begin{matrix}\mathbb{P}_{(x,y)\sim\mathcal{D},\delta\sim\mathcal{F}}\big[\arg\max_{i\in[c]} g_i(x+\delta) = y\big],\\ \mathbb{P}_{(x,y)\sim\mathcal{D},\delta\sim\mathcal{F}}\big[\arg\max_{i\in[c]} h_i(x+\delta) = y\big]\end{matrix}\right\},$$

where $\mathcal{F}$ is any distribution such that $\mathbb{P}_{\delta\sim\mathcal{F}}\big[\|\delta\|_p > \epsilon\big] = 0$.

The proof of Theorem 2 is shown in Appendix A.2. Note that the distribution $\mathcal{F}$ is arbitrary, implying that the test data can be clean data, any type of adversarial data, or some combination of both. As a special case, when the probability density function (PDF) of $\mathcal{F}$ is a Dirac delta at zero, Theorem 2 implies that the clean accuracy of $g^{\alpha}_{\text{CNN}}(\cdot)$ is as good as the standard classifier $g(\cdot)$. Conversely, when the PDF of $\mathcal{F}$ is a Dirac delta at the worst-case perturbation, the adversarial accuracy of $g^{\alpha}_{\text{CNN}}(\cdot)$ is not worse than the robust model $h(\cdot)$, implying that if $h(\cdot)$ is inherently robust, then $g^{\alpha}_{\text{CNN}}(\cdot)$ inherits the robustness. One can then conclude that there exists a $g^{\alpha}_{\text{CNN}}(\cdot)$ that matches the clean accuracy of $g(\cdot)$ and the robustness of $h(\cdot)$.

While finding an $\alpha(\cdot)$ function that perfectly achieves this trade-off is hard, we will use experiments to show that an $\alpha(\cdot)$ represented by a neural network can retain most of the robustness $h(\cdot)$ while vastly boosting the clean accuracy, even on challenging datasets such as CIFAR-100.

4.2. Attacking the adaptive classifier

When the combined model $g^{\alpha}_{\text{CNN}}(\cdot)$ is under adversarial attack, the policy $\alpha(\cdot)$ provides an addition gradient flow path. Intuitively, the attack should be able to force α to be small through this additional gradient path, tricking the policy to favor the non-robust $g(\cdot)$. Following the guidelines for constructing adaptive attacks [55], in the experiments, we consider the following types of attacks:

A Gray-box PGD$_{20}$: In this setting, the adversary has access to the gradients of both $g(\cdot)$ and $h(\cdot)$, but is not given the gradient of the policy network. We use untargeted PGD attack with a fixed initialization to generate the attacks.

B White-box PGD$_{20}$: Since the smoothed classifier is end-to-end differentiable, following [55], we allow the adversary to access the end-to-end gradient, including the gradient of the policy network.

C White-box AutoAttack: [22] has proposed to use an ensemble of four automated attack algorithms to form a stronger attack – "AutoAttack". The method considers APGD attacks generated via the untargeted cross-entropy loss and the targeted DLR loss, in addition to the targeted FAB attack and the black-box Square attack [8]. Again, the end-to-end gradient of the smoothed classifier is available to the adversary. AutoAttack requires much more computation budget than PGD$_{20}$.

D Adaptive white-box AutoAttack: Since the policy network is a crucial component of the defense, we adapt AutoAttack to target the policy by adding an APGD loss component that aims to decrease α. We use this additional attack type for evaluation purposes.

We will show that the adaptively smoothed model is robust against the attack that it is trained against. When trained using APGD$_{75}$ attack with untargeted and targeted loss functions, our model becomes robust against AutoAttack. Furthermore, a significant improvement in the accuracy-robustness trade-off is achieved.

4.3. Training the policy network

In practice, we use a neural network $\alpha_\theta(\cdot) : \mathbb{R}^d \to [0, 1]$ to learn an effective policy that adjusts the outputs of $g(\cdot)$ and $h(\cdot)$. Here, θ represents the trainable parameters of the policy, and we refer to $\alpha_\theta(\cdot)$ as the "policy network". The output range constraint is enforced by applying a Sigmoid function to the policy network. Note that when training the policy network $\alpha_\theta(\cdot)$, the base classifiers $g(\cdot)$ and $h(\cdot)$ are frozen to avoid unnecessary feature distortions.

Since the policy network should treat clean and attacked inputs differently, its task is closely related to the adversary detection problem. To this end, we adapt the detection architecture introduced in [43] for our policy network. While [15] has argued that simultaneously attacking the base classifier and the adversary detector can bring the detection rate of the detection method proposed in [43] to near zero, we make a few key modifications:

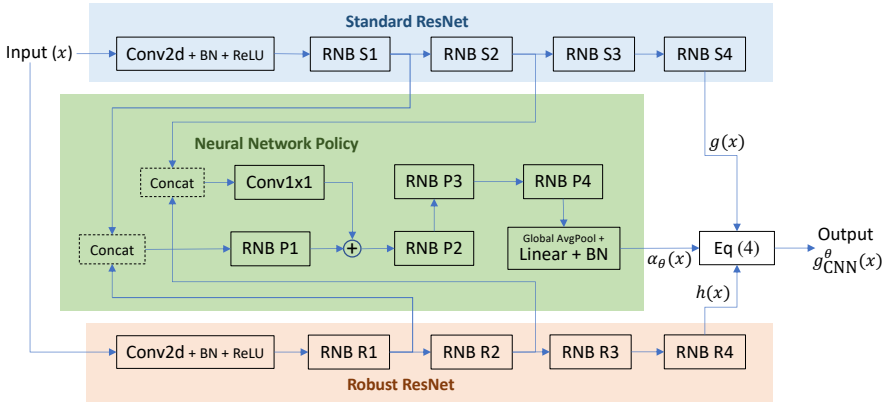


Figure 3. The overall architecture of the adaptively smoothed classifier introduced in Section 4 when applied to a pair of ResNet18 classifiers. "RNB" is an abbreviation of ResNetBlock and "BN" represents the 2D batch normalization layer. The Conv1x1 layer serves the role of reducing the number of features and improving efficiency.

- Our policy $\alpha_\theta(\cdot)$ takes advantage of the two available models $g(\cdot)$ and $h(\cdot)$ by using the intermediate features of both networks via concatenation.
- Instead of using the output of $\alpha_\theta(\cdot)$ directly for attack identification, we use it more delicately. Since Figure 1 shows that even a constant α can improve the accuracy-robustness trade-off, our method does not excessively rely on the performance of the policy network $\alpha_\theta(\cdot)$.
- We include stronger adaptive adversaries during training to generate more diverse training examples.

The modified architecture is shown in Figure 3. In Section 5.2, we provide empirical results demonstrating that the above modifications help the overall composite network defend against strong attacks. For the policy network, we choose a ResNet18-like structure, which is known to perform well for a wide range of computer vision applications and is often considered the go-to architecture.

Consider the following two loss functions for training the policy $\alpha_\theta(\cdot)$:

- **Multi-class cross-entropy:** We minimize the multi-class cross-entropy loss of the combined classifier, which is the ultimate goal of the policy network:

$$\min_\theta \mathbb{E}_{\substack{(x,y)\sim\mathcal{D} \\ \delta\sim\mathcal{F}}} \Big[\ell_{\text{CE}}\big(g^\theta_{\text{CNN}}(x+\delta), y\big) \Big], \qquad (5)$$

  where ℓ_{CE} is the cross-entropy loss for logits and $y \in [c]$ is the label corresponding to x. The base classifiers $g(\cdot)$ and $h(\cdot)$ are not updated. Again, δ denotes the perturbation and the distribution $\mathcal{F}$ is arbitrary. In our experiments, to avoid overfitting to a particular attack radius, $\mathcal{F}$ is selected to be formed by perturbations with randomized radii.

- **Binary cross-entropy:** The optimal $\alpha^\star$ that minimizes ℓ_{CE} in (5) can be estimated for each training point. Specifically, depending on whether the input is attacked and how it is attacked, either $g(\cdot)$ or $h(\cdot)$ should be prioritized. Thus, we treat the task as a binary classification problem and solve the optimization problem

$$\min_\theta \mathbb{E}_{\substack{(x,y)\sim\mathcal{D} \\ \delta\sim\mathcal{F}}} \Big[\ell_{\text{BCE}}\big(\alpha_\theta(x+\delta), \widetilde{\alpha}\big) \Big],$$

  where ℓ_{BCE} is the binary cross-entropy loss for probabilities and $\widetilde{\alpha} \in \{0, 1\}$ is the "pseudo label" for the output of the policy.

Using only the multi-class loss suffers from a distributional mismatch between the training set and the test set. The robust classifier $h(\cdot)$ may achieve a low loss on adversarial training data but a high loss on adversarial test data. For example, with the CIFAR-10 dataset and our ResNet18 robust classifier, the PGD_{10} adversarial training accuracy is 93.01% while the PGD_{10} test accuracy is 45.55%. As a result, approximating (5) with empirical risk minimization on the training set does not effectively optimize the true risk. When the adversary attacks a test input x targeting $h(\cdot)$, the standard prediction $g(x)$ yields a lower loss than $h(x)$. However, if x is an attacked example in the training set, then the losses of $g(x)$ and $h(x)$ are similar, and the policy network does not receive a strong incentive to choose $g(\cdot)$ when it detects an attack targeting $h(\cdot)$.

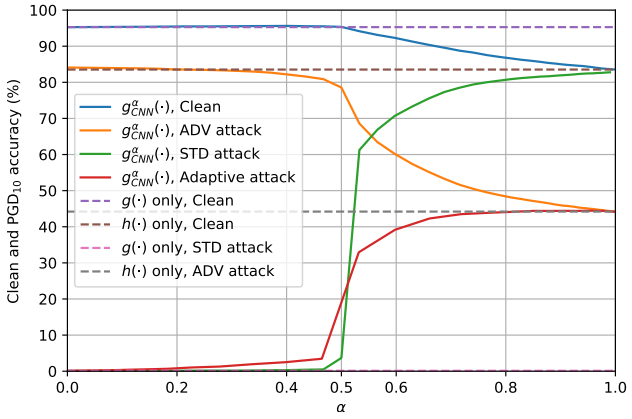


Figure 4. The performance of the smoothed model $g^{\alpha}_{\text{CNN}}(\cdot)$. "STD attack", "ADV attack", and "Adaptive attack" refer to the PGD_{10} attack generated using the gradient of $g(\cdot)$, $h(\cdot)$, and $g^{\alpha}_{\text{CNN}}(\cdot)$ respectively, with ϵ set to $\frac{8}{255}$.

The binary loss, however, does not capture the potentially different sensitivity of each input. Certain inputs can be more vulnerable against adversarial attacks, and ensuring the correctness of the policy on these inputs is more crucial.

To this end, we propose a composite loss function that combines the above two components, providing incentives for the policy to select the standard classifier $g(\cdot)$ when appropriate, while forcing the policy to remain conservative. The composite loss for a data-label pair (x, y) is given by

$$\begin{aligned}&\ell_{\text{composite}}\big(\theta, (x, y, \widetilde{\alpha})\big) \qquad (6)\\ &\quad = c_1 \cdot \ell_{\text{CE}}\big(g^{\theta}_{\text{CNN}}(x+\delta), y\big) + \\ &\qquad c_2 \cdot \ell_{\text{BCE}}\big(\alpha_\theta(x+\delta), \widetilde{\alpha}\big) + \\ &\qquad c_3 \cdot \ell_{\text{CE}}\big(g^{\theta}_{\text{CNN}}(x+\delta), y\big) \cdot \ell_{\text{BCE}}\big(\alpha_\theta(x+\delta), \widetilde{\alpha}\big),\end{aligned}$$

where the hyperparameters c_1, c_2, and c_3 control the weights of the loss components.

5. Numerical experiments

In this section, we use experiments on the CIFAR-10 and the CIFAR-100 datasets to validate the proposed method. Due to the lower difficulty of CIFAR-10, recent progress in learning robust models has made the accuracy-robustness trade-off less noticeable for this dataset [26, 27, 51]. On more challenging tasks, such as CIFAR-100, this trade-off is still highly noticeable, and the advantages of adaptive smoothing are more prominent for these tasks. Nonetheless, due to the popularity of CIFAR-10 in the field of adversarial robustness, we still use small models trained on this dataset to perform ablation analyses and proof-of-concept demonstrations. On the more suitable CIFAR-100 dataset, we use state-of-the-art classifiers as the base models $g(\cdot)$ and $h(\cdot)$, where $g(\cdot)$ takes advantage of accuracy-optimized pre-training and $h(\cdot)$ exploits recent robust training methods. We then apply adaptive smoothing to these high-performance models and demonstrate that our method trains simultaneously accurate and robust models, reconciling the accuracy-robustness trade-off to an unprecedented level.

5.1. Robust neural network smoothing with a fixed strength

We first use the CIFAR-10 dataset to evaluate the performance of the composite models $g^{\alpha}_{\text{CNN}}(\cdot)$ with different fixed values of α. Specifically, we use a ResNet18 model trained on clean data as the standard model $g(\cdot)$ and use another ResNet18 trained on PGD_{20} data as the robust model $h(\cdot)$. We consider PGD_{20} attacks that target $g(\cdot)$ and $h(\cdot)$, in addition to the adaptive PGD_{20} attacks generated using the end-to-end gradient of $g^{\alpha}_{\text{CNN}}(\cdot)$.

The test accuracy of each composite model is presented in Figure 4. As α increases, the clean accuracy of $g^{\alpha}_{\text{CNN}}(\cdot)$ converges from the clean accuracy of $g(\cdot)$ to the clean accuracy of $h(\cdot)$. In terms of the attacked performance, when the attack targets $g(\cdot)$, the attacked accuracy increases with α. When the attack targets $h(\cdot)$, the attacked accuracy decreases with α, showing that the attack becomes more benign to the composite model when it emphasizes $g(\cdot)$ more. When the adaptive attack targets $g^{\alpha}_{\text{CNN}}(\cdot)$, the attacked accuracy increases with α.

5.2. Robust neural network smoothing with adaptive strength

Next, we evaluate the performance of the adaptive composite model $g^{\theta}_{\text{CNN}}(\cdot)$ using the CIFAR-10 and the CIFAR-100 datasets. We consider ℓ_∞ attacks and use different robust neural networks for $h(\cdot)$. In all experiments, the hyperparameters for the composite loss function (6) are $c_1 = 0.5$, $c_2 = 1$, and $c_3 = 0.1$. The AdamW optimizer [33] is used for optimization.

The training inputs for the policy $\alpha_\theta(\cdot)$ include the clean data and the corresponding types of attacked data. For each dataset, we train three policy networks using adversarial examples generated with the attack settings A, B, and C presented in Section 4.2, respectively. To alleviate overfitting, we randomize the attack radius and the number of steps. Moreover, we add a

CIFAR-10 base classifier performances

Model	Architecture	Clean	PGD_{20}	AutoAtt.
$g(\cdot)$ (accurate)	ResNet-18†	95.28 %	0.12 %	0.00 %
$h(\cdot)$ (robust)	WRN-34‡	84.92 %	57.16 %	53.09 %

CIFAR-10 $g^{\theta}_{\text{CNN}}(\cdot)$ performance

Training Setting / Eval Setting	**A**	**B**	**C**
Clean	92.05 %	92.07 %	91.51 %
A (gray-box PGD_{20})	57.22 %	57.25 %	56.30 %
B (white-box PGD_{20})	56.63 %	57.09 %	56.29 %
C (white-box AutoAtt.)	40.04 %	40.02 %	42.78 %
D (adaptive AutoAtt.)	39.85 %	39.70 %	42.66 %

†: [45] (Vanilla training). ‡: [61] (TRADE).

Table 1. CIFAR-10 results of the smoothed models trained with three different settings.

CIFAR-100 base classifier performances

Model	Architecture	Clean	PGD_{20}	AutoAtt.
$g(\cdot)$ (accurate)	ResNet-152‡‡	91.38 %	0.14 %	0.00 %
$h(\cdot)$ (robust)	WRN-70††	69.17 %	40.86 %	36.98 %

CIFAR-100 $g^{\theta}_{\text{CNN}}(\cdot)$ performance

Training Setting / Eval Setting	**A**	**B**	**C**
Clean	83.99 %	83.96 %	**80.90** %
A (gray-box PGD_{20})	40.04 %	39.80 %	39.26 %
B (white-box PGD_{20})	30.59 %	34.48 %	38.92 %
C (white-box AutoAtt.)	23.54 %	26.37 %	**32.94** %
D (adaptive AutoAtt.)	23.78 %	26.17 %	32.80 %

‡‡: Based on [34] (BiT). ††: [26].

Table 2. CIFAR-100 results of the smoothed models trained with the three settings. When the training setting C is used, an 80.90% clean accuracy and a 32.94% AutoAttacked accuracy is achieved.

randomly-weighted binary cross-entropy loss component that targets the policy (this loss tries to trick the policy to favor $g(\cdot)$). For the setting C (AutoAttack), the training data only include targeted and untargeted APGD attacks. The other two AutoAttack components, FAB and Square, are excluded during training in the interest of efficiency but are included for evaluation. Tables 1 and 2 present the test accuracy of $g^{\theta}_{\text{CNN}}(\cdot)$ for each setting, where each column represents the performance of one adaptively smoothed model.

The empirical results show that the combined classifier can defend against the attacks it is trained on. Specifically, for the attack setting A (gray-box PGD), $g^{\theta}_{\text{CNN}}(\cdot)$ is able to achieve the same level of PGD_{20}-attacked accuracy as $h(\cdot)$ while retaining a similar level of clean accuracy as $g(\cdot)$. For the setting B (white-box PGD), the attack is allowed to follow the gradient path provided by $\alpha(\cdot)$ and deliberately evade the part of the adversarial input space recognized by $\alpha_{\theta}(\cdot)$. While the training task becomes more challenging, the improvement in the accuracy-robustness trade-off is still substantial. Furthermore, the composite model can generalize to examples generated via the stronger AutoAttack.

For the setting C (AutoAttack), the difficulty of the training problem further escalates. While the performance of $g^{\theta}_{\text{CNN}}(\cdot)$ on clean data slightly decreases, the policy network can offer a more vigorous defense against AutoAttack data, still improving the accuracy-robustness trade-off. Note that the improvement is more significant on the CIFAR-100 dataset, where $g^{\theta}_{\text{CNN}}(\cdot)$ correctly classifies 1173 additional clean images compared with $h(\cdot)$ (cutting the error rate by a third) while making only 404 additional incorrect predictions on AutoAttacked inputs (increasing the error rate by merely 6.4 relative percent). Since the attacked accuracy of the non-robust base classifier $g(\cdot)$ on the CIFAR-100 dataset is zero, the observation that $g^{\theta}_{\text{CNN}}(\cdot)$ preserves $\frac{32.94}{36.98} \approx 89\%$ of the AutoAttacked accuracy of $h(\cdot)$ implies that among all AutoAttacked inputs that are correctly predicted by $h(\cdot)$, the policy helps $g^{\theta}_{\text{CNN}}(\cdot)$ identify 89% of them.

The above results show that $\alpha_{\theta}(\cdot)$ is capable of approximating a robust and high-performance policy when trained with sufficiently diverse attacked data. The fact that $g^{\theta}_{\text{CNN}}(\cdot)$ combines the clean accuracy of $g(\cdot)$ and the robustness of $h(\cdot)$ highlights that our method significantly improves the accuracy-robustness trade-off. If a different type of attack needs to be considered, the training set for $\alpha_{\theta}(\cdot)$ can be further augmented with the corresponding adversarial data.

6. Conclusions

This paper proposes "adaptive smoothing", a flexible framework that leverages the mixture of the outputs of an accurate classifier and a robust model to mitigate the accuracy-robustness trade-off of neural networks. We mathematically prove that the smoothed model can inherit the certified robustness of the robust base model under realistic assumptions. We then adapt an adversarial input detector into a deterministic policy network, further improving the accuracy-robustness trade-off. Solid empirical results show that our method can simultaneously benefit from the high accuracy of modern pre-trained standard (non-robust) models and

the robustness achieved via state-of-the-art robust classification methods. Because our theoretical study demonstrates the possibility of leveraging the policy network to avoid the accuracy-robustness trade-off entirely, future advancements in adversarial example identification can reconcile this trade-off even more effectively via our framework. Thus, this work paves the way for future research to focus on either accuracy or robustness without sacrificing the other.

References

[1] Morteza Ali Ahmadi, Rouhollah Dianat, and Hossein Amirkhani. An adversarial attack detection method in deep neural networks based on re-attacking approach. *Multimedia Tools and Applications*, 80(7):10985–11014, 2021. 3

[2] Manaar Alam, Shubhajit Datta, Debdeep Mukhopadhyay, Arijit Mondal, and Partha Pratim Chakrabarti. Resisting adversarial attacks in deep neural networks using diverse decision boundaries. *arXiv preprint arXiv:2208.08697*, 2022. 3

[3] Ahmed Aldahdooh, Wassim Hamidouche, and Olivier Déforges. Selective and features based adversarial example detection. *arXiv preprint arXiv:2103.05354*, 2021. 3

[4] Ahmed Aldahdooh, Wassim Hamidouche, Sid Ahmed Fezza, and Olivier Déforges. Adversarial example detection for dnn models: A review and experimental comparison. *arXiv preprint arXiv:2105.00203*, 2021. 3

[5] Brendon Anderson, Ziye Ma, Jingqi Li, and Somayeh Sojoudi. Tightened convex relaxations for neural network robustness certification. In *IEEE Conference on Decision and Control*, 2020. 1

[6] Brendon G. Anderson and Somayeh Sojoudi. Data-driven assessment of deep neural networks with random input uncertainty. *arXiv preprint arXiv:2010.01171*, 2020. 1

[7] Brendon G. Anderson and Somayeh Sojoudi. Certified robustness via locally biased randomized smoothing. In *Annual Learning for Dynamics and Control Conference*, 2022. 1, 3, 4

[8] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: A query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, 2020. 6

[9] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, 2018. 2

[10] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International Conference on Machine Learning*, 2018. 2

[11] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. In *International Joint Conference on Artificial Intelligence*, 2021. 1

[12] Yatong Bai, Tanmay Gautam, Yu Gai, and Somayeh Sojoudi. Practical convex formulation of robust one-hidden-layer neural network training. *American Control Conference*, 2022. 1

[13] Yatong Bai, Tanmay Gautam, and Somayeh Sojoudi. Efficient global optimization of two-layer relu networks: Quadratic-time algorithms and adversarial training. *SIAM Journal on Mathematics of Data Science*, 2022. 1

[14] Yogesh Balaji, Tom Goldstein, and Judy Hoffman. Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. *arXiv preprint arXiv:1910.08051*, 2019. 1

[15] Nicholas Carlini and David Wagner. *Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods*. 2017. 3, 6

[16] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, 2017. 2, 3

[17] Fabio Carrara, Fabrizio Falchi, Roberto Caldelli, Giuseppe Amato, and Rudy Becarelli. Adversarial image detection in deep neural networks. *Multimedia Tools and Applications*, 78(3):2815–2835, 2019. 3

[18] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1

[19] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, 2019. 1, 3

[20] Francesco Croce, Sven Gowal, Thomas Brunner, Evan Shelhamer, Matthias Hein, and Taylan Cemgil. Evaluating the adversarial robustness of adaptive test-time defenses. *arXiv preprint arXiv:2202.13711*, 2022. 2

[21] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, 2020. 2

[22] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, 2020. 2, 6

[23] Lijie Fan, Sijia Liu, Pin-Yu Chen, Gaoyuan Zhang, and Chuang Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? In *Advances in Neural Information Processing Systems*, 2021. 1

[24] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems*, 2019. 5

[25] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. 1, 2

[26] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020. 2, 8, 9

[27] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan A. Calian, and Timothy Mann. Improving robustness using generated data. *arXiv preprint arXiv:2110.09468*, 2021. 2, 8

[28] Sven Gowal, Jonathan Uesato, Chongli Qin, Po-Sen Huang, Timothy Mann, and Pushmeet Kohli. An alternative surrogate loss for pgd-based adversarial testing. *arXiv preprint arXiv:1910.09338*, 2019. 2

[29] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, 2017. 5

[30] Ting-Kuei Hu, Tianlong Chen, Haotao Wang, and Zhangyang Wang. Triple wins: Boosting accuracy, robustness and efficiency together by enabling input-adaptive inference. In *International Conference on Learning Representations*, 2020. 2

[31] Sandy H. Huang, Nicolas Papernot, Ian J. Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. In *International Conference on Learning Representations*, 2017. 1

[32] Xiaojun Jia, Yong Zhang, Baoyuan Wu, Ke Ma, Jue Wang, and Xiaochun Cao. Las-at: Adversarial training with learnable attack strategy. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2

[33] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 8

[34] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *European Conference on Computer Vision*, 2020. 9

[35] Alex Krizhevsky. Learning multiple layers of features from tiny images, 2012. 4

[36] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations*, 2017. 1

[37] Alex Lamb, Vikas Verma, Juho Kannala, and Yoshua Bengio. Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy. In *ACM Workshop on Artificial Intelligence and Security*, 2019. 1

[38] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. In *Advances in Neural Information Processing Systems*, 2019. 1

[39] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *European Conference on Computer Vision*, 2018. 2

[40] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 14

[41] Ziye Ma and Somayeh Sojoudi. A sequential framework towards an exact SDP verification of neural networks. In *International Conference on Data Science and Advanced Analytics*, 2021. 1

[42] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 2, 4

[43] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. In *International Conference on Learning Representations*, 2017. 3, 6

[44] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1

[45] Dongbin Na. Pytorch adversarial training on cifar-10. https://github.com/ndb796/Pytorch-Adversarial-Training-CIFAR, 2020. 5, 9

[46] Tianyu Pang, Min Lin, Xiao Yang, Jun Zhu, and Shuicheng Yan. Robustness and accuracy could be reconcilable by (proper) definition. *arXiv preprint arXiv:2202.10103*, 2022. 2

[47] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, 2019. 3

[48] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM Asia conference on computer and communications security*, 2017. 2

[49] Samuel Pfrommer, Brendon G Anderson, and Somayeh Sojoudi. Projected randomized smoothing for certified adversarial robustness. *Preprint.* https://brendon-anderson.github.io/files/publications/pfrommer2022projected.pdf, 2022. 1

[50] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C. Duchi, and Percy Liang. Understanding and mitigating the tradeoff between robustness and accuracy. In *International Conference on Machine Learning*, 2020. 1

[51] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021. 2, 8

[52] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. *Advances in neural information processing systems*, 31, 2018. 2

[53] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? In *International Conference on Learning Representations*, 2022. 2

[54] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 2019. 2

[55] Florian Tramèr, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *Advances in Neural Information Processing Systems*, 2020. 2, 6

[56] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian J. Goodfellow, Dan Boneh, and Patrick D. McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018. 1

[57] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019. 1

[58] Jianyu Wang and Haichao Zhang. Bilateral adversarial training: Towards fast training of more robust models against adversarial attacks. In *International Conference on Computer Vision*, 2019. 1

[59] Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Russ R. Salakhutdinov, and Kamalika Chaudhuri. A closer look at accuracy vs. robustness. In *Annual Conference on Neural Information Processing Systems*, 2020. 1

[60] Haichao Zhang and Jianyu Wang. Defense against adversarial attacks using feature scattering-based adversarial training. In *Annual Conference on Neural Information Processing Systems*, 2019. 1

[61] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019. 1, 2, 9

[62] Haizhong Zheng, Ziqi Zhang, Juncheng Gu, Honglak Lee, and Atul Prakash. Efficient adversarial training with transferable adversarial examples. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1

[63] Yaowei Zheng, Richong Zhang, and Yongyi Mao. Regularizing neural networks via adversarial model perturbation. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 2

A. Appendix: Proofs

A.1. Proof of Theorem 1

Consider an arbitrary input x. Suppose that $h_i(x) \geq h_j(x)$. Since, under our framework (4), the network $g(\cdot)$ maps into the probability simplex, it holds that $g_k(x+\delta) \in [0,1]$ for all k, and therefore

$$\begin{aligned}
\exp\left(g^{\alpha}_{\mathrm{CNN},i}(x+\delta)\right) - \exp\left(g^{\alpha}_{\mathrm{CNN},j}(x+\delta)\right) &= (1-\alpha)(g_i(x+\delta) - g_j(x+\delta)) + \alpha(h_i(x+\delta) - h_j(x+\delta)) \\
&\geq (1-\alpha)(0-1) + \alpha(h_i(x+\delta) - h_j(x+\delta)) \\
&= \alpha - 1 + \alpha(h_i(x+\delta) - h_j(x+\delta)) \\
&= \alpha - 1 + \alpha(h_i(x) - h_j(x) + h_i(x+\delta) - h_i(x) + h_j(x) - h_j(x+\delta)) \\
&\geq \alpha - 1 + \alpha(h_i(x) - h_j(x)) - \alpha\,\mathrm{Lip}_p(h_i)\|\delta\|_p - \alpha\,\mathrm{Lip}_p(h_j)\|\delta\|_p \\
&= (\alpha|h_i(x) - h_j(x)| + \alpha - 1) - \alpha\left(\mathrm{Lip}_p(h_i) + \mathrm{Lip}_p(h_j)\right)\|\delta\|_p \\
&\geq (\alpha|h_i(x) - h_j(x)| + \alpha - 1) - \alpha\left(\mathrm{Lip}_p(h_i) + \mathrm{Lip}_p(h_j)\right) r^{\alpha}_p(x) \\
&= 0.
\end{aligned}$$

Hence, $\mathrm{sgn}\left(\exp\left(g^{\alpha}_{\mathrm{CNN},i}(x+\delta)\right) - \exp\left(g^{\alpha}_{\mathrm{CNN},j}(x+\delta)\right)\right) = \mathrm{sgn}(h_i(x) - h_j(x))$ in this case.

Now suppose that $h_i(x) \leq h_j(x)$. Then, following the same line of reasoning as above, we find that

$$\begin{aligned}
\exp\left(g^{\alpha}_{\mathrm{CNN},i}(x+\delta)\right) - \exp\left(g^{\alpha}_{\mathrm{CNN},j}(x+\delta)\right) &= (1-\alpha)(g_i(x+\delta) - g_j(x+\delta)) + \alpha(h_i(x+\delta) - h_j(x+\delta)) \\
&\leq (1-\alpha)(1-0) + \alpha(h_i(x+\delta) - h_j(x+\delta)) \\
&= 1 - \alpha + \alpha(h_i(x+\delta) - h_j(x+\delta)) \\
&= 1 - \alpha + \alpha(h_i(x) - h_j(x) + h_i(x+\delta) - h_i(x) + h_j(x) - h_j(x+\delta)) \\
&\leq 1 - \alpha + \alpha(h_i(x) - h_j(x)) + \alpha\,\mathrm{Lip}_p(h_i)\|\delta\|_p + \alpha\,\mathrm{Lip}_p(h_j)\|\delta\|_p \\
&= -(\alpha|h_i(x) - h_j(x)| + \alpha - 1) + \alpha\left(\mathrm{Lip}_p(h_i) + \mathrm{Lip}_p(h_j)\right)\|\delta\|_p \\
&\leq -(\alpha|h_i(x) - h_j(x)| + \alpha - 1) + \alpha\left(\mathrm{Lip}_p(h_i) + \mathrm{Lip}_p(h_j)\right) r^{\alpha}_p(x) \\
&= 0.
\end{aligned}$$

Therefore, we find again that $\mathrm{sgn}\left(\exp\left(g^{\alpha}_{\mathrm{CNN},i}(x+\delta)\right) - \exp\left(g^{\alpha}_{\mathrm{CNN},j}(x+\delta)\right)\right) = \mathrm{sgn}(h_i(x) - h_j(x))$ in this case.

Combining the above two cases with the observation that the exponential function is monotonically increasing leads to the conclusion that $\mathrm{sgn}\left(g^{\alpha}_{\mathrm{CNN},i}(x+\delta) - g^{\alpha}_{\mathrm{CNN},j}(x+\delta)\right) = \mathrm{sgn}(h_i(x) - h_j(x))$. □

A.2. Proof of Theorem 2

Since it is assumed that the perturbation balls of the data are non-overlapping, the true label y corresponding to each perturbed data $x+\delta$ with the property $\|\delta\|_p \leq \epsilon$ is unique. Therefore, the indicator function

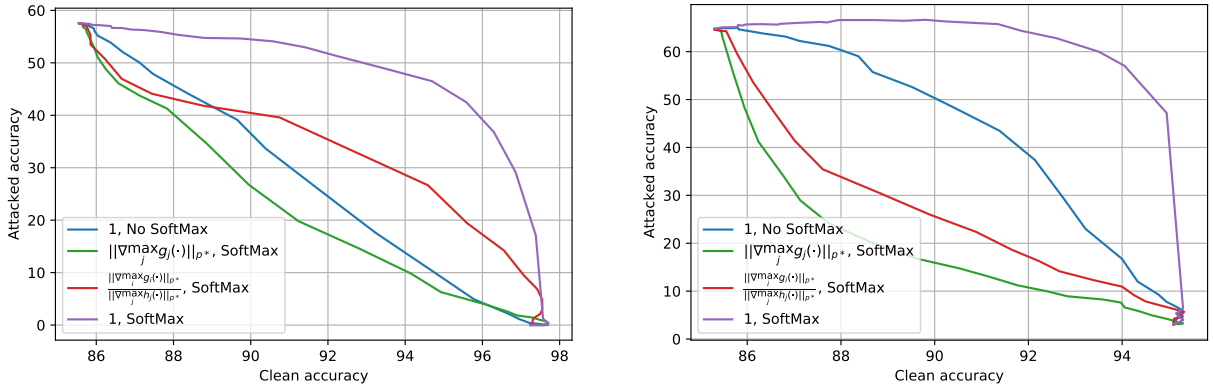
$$\alpha(x+\delta) = \begin{cases} 0 & \text{if } \arg\max_{i\in[c]} g_i(x+\delta) = y, \\ 1 & \text{otherwise,} \end{cases}$$

satisfies that

$$\begin{aligned}
\alpha(x+\delta) = 0 \quad &\text{if} \quad \arg\max_{i\in[c]} g_i(x+\delta) = y, \\
\alpha(x+\delta) = 1 \quad &\text{if} \quad \arg\max_{i\in[c]} g_i(x+\delta) \neq y \text{ and } \arg\max_{i\in[c]} h_i(x+\delta) = y.
\end{aligned}$$

	Attack Budget and PGD Steps	$g(\cdot)$ Architecture	$h(\cdot)$ Architecture
Figure 1	ℓ_∞, $\epsilon = \frac{8}{255}$, 10 Steps	Standard ResNet18	ℓ_∞-adversarially-trained ResNet18
Figure 5a	ℓ_∞, $\epsilon = \frac{8}{255}$, 20 Steps	Standard ConvNeXT-T	TRADE WideResNet-34
Figure 5b	ℓ_2, $\epsilon = 0.5$, 20 Steps	Standard ResNet18	ℓ_2-adversarially-trained ResNet18

Table 3. Experiment settings for comparing the choices of $R_i(x)$.



(a) ConvNeXT-T and TRADE WRN-34 under ℓ_∞ PGD attack.

(b) Standard and AT ResNet18s under ℓ_2 PGD attack.

Figure 5. Comparing the options for $R_i(x)$ with alternative choices of base classifiers.

Therefore, it holds that

$$g^{\alpha}_{\mathrm{CNN},i}(x+\delta) = g_i(x+\delta) \quad \text{if} \quad \arg\max_{i\in[c]} g_i(x+\delta) = y,$$
$$g^{\alpha}_{\mathrm{CNN},i}(x+\delta) = h_i(x+\delta) \quad \text{if} \quad \arg\max_{i\in[c]} g_i(x+\delta) \neq y \text{ and } \arg\max_{i\in[c]} h_i(x+\delta) = y,$$

implying that

$$\arg\max_{i\in[c]} g^{\alpha}_{\mathrm{CNN},i}(x+\delta) = y \quad \text{if} \quad \Big(\arg\max_{i\in[c]} g_i(x+\delta) = y \text{ or } \arg\max_{i\in[c]} h_i(x+\delta) = y\Big),$$

which leads to the desired statement. □

B. Appendix: Additional empirical supports for selecting $R_i(x) = 1$

In this section, we use additional empirical evidence (Figures 5a and 5b) to show that $R_i(x) = 1$ is the best choice for the adaptive smoothing formulation, and the post-SoftMax probabilities should be used for smoothing. While most of the experiments in this paper are based on ResNets, the architecture is chosen solely because of its popularity, and our method does not depend on any properties of ResNets. Therefore, for the experiment in Figure 5a, we select an alternative architecture by using a more modern ConvNeXT-T model [40] pre-trained on ImageNet-1k as $g(\cdot)$. We also use a robust model trained via TRADE in place of an adversarially-trained network for $h(\cdot)$. Moreover, while most of our experiments are based on ℓ_∞ attacks, our method applies to all ℓ_p attack budgets. In Figure 5b, we provide an example that considers the ℓ_2 attack. The experiment settings are summarized in Tab. 3.

Figures 5a and 5b demonstrate that setting $R_i(x)$ to the constant 1 achieves the best trade-off curve between clean and attacked accuracy. Moreover, smoothing using the post-SoftMax probabilities outperforms the pre-SoftMax logits. This result aligns with the conclusions of Figure 1 and our theoretical analyses.