Evaluating the Adversarial Robustness of Adaptive Test-time Defenses

Francesco Croce [* 1] **Sven Gowal** [* 2] **Thomas Brunner** [* 3] **Evan Shelhamer** [* 2] **Matthias Hein** [1] **Taylan Cemgil** [2]

Abstract

Adaptive defenses, which optimize at test time, promise to improve adversarial robustness. We categorize such *adaptive test-time defenses*, explain their potential benefits and drawbacks, and evaluate a representative variety of the latest adaptive defenses for image classification. Unfortunately, none significantly improve upon static defenses when subjected to our careful case study evaluation. Some even weaken the underlying static model while simultaneously increasing inference computation. While these results are disappointing, we still believe that adaptive test-time defenses are a promising avenue of research and, as such, we provide recommendations for their thorough evaluation. We extend the checklist of Carlini et al. (2019) by providing concrete steps specific to adaptive defenses.

1. Introduction

Despite the successes of deep learning (Goodfellow et al., 2016), it is well-known that deep networks are not intrinsically robust. In particular, the addition of small but carefully chosen ℓ_p-norm bounded deviations to the input, called adversarial perturbations, can cause a deep network to make incorrect predictions with high confidence (Carlini & Wagner, 2017a;b; Goodfellow et al., 2015; Kurakin et al., 2016; Szegedy et al., 2014; Biggio et al., 2013). Starting with Biggio et al. (2013) and Szegedy et al. (2014), there has been a lot of work on understanding and generating adversarial perturbations (Carlini & Wagner, 2017b; Athalye & Sutskever, 2018; Uesato et al., 2018), and on building models that are robust to such perturbations (Goodfellow et al., 2015; Papernot et al., 2016; Madry

[*]Equal contribution [1]University of Tübingen, Germany [2]DeepMind, London, United Kingdom [3]Everyday Robots, Munich, Germany. Correspondence to: Francesco Croce <francesco.croce@uni-tuebingen.de>, Sven Gowal, Evan Shelhamer <{sgowal,shelhamer}@deepmind.com>, Thomas Brunner <tbrunner@everydayrobots.com>.

et al., 2018; Kannan et al., 2018; Zhang et al., 2019; Rice et al., 2020; Gowal et al., 2020; 2021).

A model f robust to ℓ_p-norm bounded perturbations aims to minimize the adversarial risk (Madry et al., 2018)

$$\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\max_{\boldsymbol{\delta}\in\{\boldsymbol{\delta}:\|\boldsymbol{\delta}\|_p<\epsilon\}} L(f(\boldsymbol{x}+\boldsymbol{\delta}),y)\right] \quad (1)$$

where $\mathcal{D}$ is a data distribution over pairs of examples $\boldsymbol{x}$ and corresponding labels y, $L(z, y)$ is a suitable loss function (such as the 0-1 loss for classification), and ϵ is a maximal norm which defines the set of allowed perturbations for a given example $\boldsymbol{x}$. As such, finding the worst-case perturbation (or optimal adversarial perturbation) $\boldsymbol{\delta}^\star$ is key for both training and testing models. Finding sub-optimal $\hat{\boldsymbol{\delta}}$ with $L(f(\boldsymbol{x}+\hat{\boldsymbol{\delta}}),y) \leq L(f(\boldsymbol{x}+\boldsymbol{\delta}^\star),y)$ estimates lower bounds on the true adversarial risk (or in the context of a classification task, upper bounds on the true robust accuracy), which may give a false sense of security. Hence, it is practically important to optimally solve the problem in Eq. 1.

The difficulty of solving this maximization problem close to optimality is highlighted by Uesato et al. (2018); Athalye et al. (2018); Tramèr et al. (2020), in which a number of published defenses are easily broken by adaptive adversarial attacks that are tuned to the defenses. This reversal highlights the importance of understanding the limitations of different robust training and testing methods (Pintor et al., 2021). Standardizing evaluation allows for the systematic tracking of real progress on adversarial robustness. However, to guarantee that standardized evaluations are accurate, defenses must adhere to some practical restrictions. RobustBench (Croce et al., 2021), for example, rules out *(i)* models which have zero gradients with respect to the input (Buckman et al., 2018; Guo et al., 2018), *(ii)* randomized models (Yang et al., 2019; Pang et al., 2020), and *(iii)* models that optimize during inference (Samangouei et al., 2018; Li et al., 2019; Schott et al., 2019). These restrictions may unnecessarily limit the development of robust models and, as a consequence, several researchers have offered general recommendations on how to evaluate adversarial defenses (Carlini et al., 2019). However, the evaluation of each new defense raises non-trivial choices.

In this work, we focus on *adaptive test-time defenses* that apply iterative optimization during inference, which

promise to circumvent limitations of *static* defenses. In the context of adversarial robustness, the optimization is designed to "purify" inputs before feeding them to a static model or to adapt the model itself (we elaborate on categorizations of adaptive test-time defenses in Sec. 2). We restrict our analysis to image classification because it is the most common test-bed for studying robustness against ℓ_p-norm bounded attacks.

We foresee adaptive defenses as an important step towards building robust defenses. However, given their novelty, the evaluation of such defenses has not been standardized. Indeed, test-time optimization can prevent the proper operation of standardized attacks established for static defenses, such as AUTOATTACK (Croce & Hein, 2020). This optimization also renders approximations such as the Backward Pass Differentiable Approximation (BPDA) (Athalye et al., 2018) more difficult to apply. As a result, recent works make strong robustness claims that turn out to be void or significantly weaker than claimed. In this paper:

❶ We categorize adaptive test-time defenses and explain their potential benefits and drawbacks;

❷ We evaluate 9 recent defenses (see Table 1), and show that they significantly overestimate their robustness. Relative to static defenses, most provide little improvement, and some are even detrimental. Furthermore all require more inference computation;

❸ Finally, we provide recommendations for evaluating such defenses and explain under which circumstances standard evaluations, like AUTOATTACK, are accurate.

2. Adaptive Test-Time Defenses

We provide a taxonomy of adaptive test-time defenses. We highlight their operating principles in Sec. 2.1, identify their building blocks in Sec. 2.2, and elaborate on potential pitfalls in Sec. 2.3. We also clearly delineate the threat model and adversarial capabilities that we assume during our case study of various techniques in Sec. 2.4.

2.1. Principles

Static defenses (e.g., standard deep networks) use a trained model for which the inputs and parameters are fixed at inference. In contrast, adaptive test-time defenses can alter the input or parameters for test inputs, either by optimizing an auxiliary loss or conditioning directly on the input. We identify two principles (highlighted in Fig. 1).

Input purification (IP): A model (possibly pre-trained for robustness) is augmented with test-time optimization to alter its inputs before the model is applied. The input optimization procedure may rely on hand-crafted (Alfarra et al., 2022; Wu et al., 2021) or learned objectives (Qian et al., 2021; Mao et al., 2021; Hwang et al., 2021) which
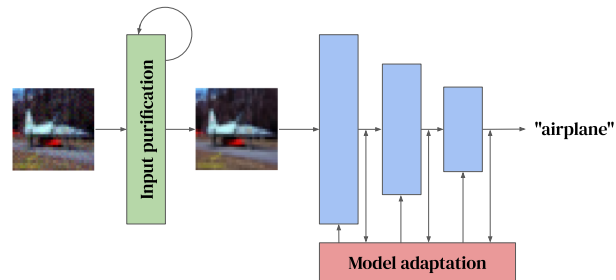

Figure 1. Adaptive test-time defenses operate by adapting their computation to the input. A first category of adaptive test-time defenses operate in the input space and aim to "purify" inputs before they are fed to the standard pre-trained model. A second category aims at adapting model parameters or intermediate activations.

may involve auxiliary networks such as generative models (Song et al., 2018; Samangouei et al., 2018; Yoon et al., 2021; Nie et al., 2022).

Model adaptation (MA): A model is made adaptive by applying an optimization procedure to alter its parameters or state (e.g., batch normalization statistics) (Wang et al., 2021a) or activations (Chen et al., 2021) during inference. Other iterative schemes, such as implicit layers (Kang et al., 2021), define their inference as an optimization process.

2.2. Building Blocks

To carry out input purification or model adaptation, adaptive test-time defenses can rely on distinct building blocks. We identify common blocks that span current defenses.

Iterative algorithm (IA): All defenses in our case study use an iterative algorithm. Five in nine solve their test-time optimization approximately by gradient descent, with some defenses using a single normalized gradient step (Qian et al., 2021) and others using as many as 40 gradient descent steps (Mao et al., 2021). Other defenses may also use layers defined via implicit functions whose output is computed with iterative algorithms (e.g., neural ODE; Kang et al., 2021) or may rely on iterative generative models (e.g., energy-based models; Yoon et al., 2021).

Auxiliary networks (AN): A significant proportion of adaptive test-time defenses (seven in nine defenses studied) rely on an external network besides the underlying static model. This network typically helps to compute the optimization objective (such as a self-supervised objective; Shi et al., 2020) or directly manipulates inputs (using a generative model for example; Yoon et al., 2021). It might be bound to (Mao et al., 2021; Hwang et al., 2021) and possibly trained jointly with (Qian et al., 2021; Shi et al., 2020; Chen et al., 2021) the static model, or used as a standalone component (Yoon et al., 2021).

Randomness (R): Three in nine defenses studied are randomized either explicitly (e.g., by adding noise to the input; Wu et al., 2021) or implicitly (e.g., by sampling different batches of inputs; Mao et al., 2021). As such, the same inputs may yield different outputs. We exclude pure randomization defenses (without test-time optimization).

External data (ED): One of the defenses studied exploits a collection of additional images, which are combined with the inputs. As such, inference does not depend only on the given input for classification (Mao et al., 2021).

2.3. Advantages, Drawbacks and Pitfalls

Test-time adaptation by optimization has shown promising results for robustness to natural shifts like common corruptions (Sun et al., 2020; Wang et al., 2021b), and have the potential to likewise improve adversarial robustness. Unlike static defenses, adaptive defenses have the freedom to alter an input (almost) arbitrarily or even update their own parameters on the input. This additional complexity seems to offer an advantage over static defenses that rely on adversarial training but remain fixed during testing. However, complex defenses are often difficult to evaluate (Tramèr et al., 2020), and adaptation might require techniques than those needed for static defenses. The accurate evaluation of adaptive test-time defenses presents several challenges:

Obfuscated gradients: First and foremost, iterative optimizers might be not differentiable (e.g., when using projections). BPDA, which is typically used in such circumstances and usually implemented as an identity function for the backward pass, can be weaker for procedures consisting of many iterations. Even in cases where the optimization is differentiable, gradients might vanish or explode when a large number of iterations is used.

Randomness: The use of randomized elements requires to resort to methods like Expectation over Transformation (EoT) (Athalye et al., 2018) and make the task of the attacker more expensive.

Runtime: The cost of inference is significantly higher than for static models, up to hundreds of times, which clearly impacts the amount of effort required for an accurate evaluation. Such high computational complexity calls into question the applicability of these defenses in the first place. Regardless of running attacks for evaluation, classification in deployment can be extremely slow. Hence, the trade-off in potential improvement versus additional computation should be the subject of further research.

2.4. Threat Model and Adversarial Capabilities

We focus exclusively on the robustness of image classifiers to ℓ_p-norm bounded input perturbations. Given a K-way classifier $f : \mathbb{R}^d \to \mathbb{R}^K$, a test input $\boldsymbol{x} \in [0, 1]^d$ with label y and a norm bound $\epsilon > 0$, an attack succeeds if it finds a perturbation $\boldsymbol{\delta}$ such that

$$\underset{k \in \{1,\dots,K\}}{\arg \max} \; [f(\boldsymbol{x} + \boldsymbol{\delta})]_k \neq y, \qquad (2)$$

$$\text{with } \|\boldsymbol{\delta}\|_p \leq \epsilon \text{ and } \boldsymbol{x} + \boldsymbol{\delta} \in [0, 1]^d.$$

where $[a]_i$ represents the i-th coordinate of a. We assume that the attacker has full white-box access to the defense. In other words, the attacker is aware of the test-time optimization in place, and has access to the parameters of the classifier and any auxiliary model(s). For randomized defenses, the attacker does not have access to the state of the random number generator. In some cases, we will fix the random seed to evaluate whether the impact of a proposed defense is mostly driven by optimization or randomization. Finally, when the defense uses external data, the attacker may have access to the set of images used (when fixed and hard-coded by the defense) or may try to infer the distribution from which these external images are drawn. For defenses that operate on a full batch of images, the attacker may be able to influence either one image from the batch or the whole batch. For all defenses, we respect the threat model defined by each defense when clearly defined in the corresponding paper or code.

3. Case Study of Adaptive Methods

In our case study, we evaluate nine adaptive test-time defenses which rely on the adaptation principles elaborated in Sec. 2. One evaluation (of Qian et al., 2021) is presented in the appendix due to space constraints. Table 1 summarizes the defenses considered in this case study, categorizes each defense (according to Sec. 2.1 and Sec. 2.2), and details the corresponding results against ℓ_∞-norm bounded attacks with budget $\epsilon = 8/255$ on CIFAR-10 (which is commonly evaluated by all defenses in their respective papers).[1] In particular, we report the robust accuracy (i.e., the classification accuracy on adversarially-perturbed inputs) originally reported in each work and the result of our evaluation. In parentheses, we report the robust accuracy of the underlying static model. We also measure the cost of inference computation in relation to performing the same inference with the underlying static model. Note that many factors can influence this value (e.g., architecture of the classifier, compute infrastructure, optimized implementation) and we use the runtime observed in our evaluation.

Overall, we find that the reported accuracy is consistently overestimated in all papers. Five defenses (Yoon et al., 2021; Hwang et al., 2021; Qian et al., 2021; Shi et al., 2020; Chen et al., 2021) have robust accuracies well below 50% and are not competitive with state-of-the-art static models (even of moderate size; Gowal et al., 2021; Rade & Moosavi-Dezfooli, 2021). Most importantly, four de-

---

[1]With the exception of $\epsilon = 2/255$ for Chen et al. (2021).

Table 1. Summary of the nine *adaptive test-time defenses* evaluated in our case study. We categorize each defense by its principles—input purification (IP) and model adaptation (MA)—and its building blocks: iterative algorithm (IA), auxiliary network (AN), randomization (R), and external data (ED). We measure the robust accuracy of each defense (and in parenthesis that of its underlying static model) against ℓ_∞-norm bounded perturbations of size $\epsilon = 8/255$ on CIFAR-10 ("Ours") along with the robust accuracy measured by the respective papers ("Reported"). * This evaluation uses $\epsilon = 2/255$. ** This evaluation uses batch size 50 instead of the original 512.

Defense	Venue	Principles		Building blocks				Inference time	Evaluation method	Robust Accuracy		
		IP	MA	IA	AN	R	ED			Reported	Ours	
Kang et al. (2021)	NeurIPS		●	●	●			2×	Transfer APGD	57.76%	52.2%	(53.9%)
Chen et al. (2021)*	ICLR		●	●	●			59×	APGD+BPDA	34.5%	5.6%	(0.0%)
Wu et al. (2021)	ArXiv	●		●			●	46×	Transfer APGD+BPDA+EoT	65.70%	61.0%	(63.0%)
Alfarra et al. (2022)	AAAI	●		●				8×	RayS (decision-based)	79.2%	66.6%	(66.6%)
Shi et al. (2020)	ICLR	●		●	●			518×	APGD+BPDA (traj.)	51.02%	3.7%	(0.0%)
Qian et al. (2021)	ArXiv	●		●	●			4×	APGD	65.07%	12.6%	(7.7%)
Hwang et al. (2021)	ICML(W)	●		●	●			40×	APGD+BPDA	52.65%	43.8%	(49.3%)
Mao et al. (2021)**	ICCV	●		●	●	●	●	407×	APGD+BPDA+EoT	63.83%	58.4%	(59.4%)
Yoon et al. (2021)	ICML	●		●	●	●		176×	APGD+EoT	69.71%	33.7%	(0.0%)

fenses (Wu et al., 2021; Kang et al., 2021; Hwang et al., 2021; Mao et al., 2021) weaken the underlying static defense (when it is already robust), while the others provide minor improvements at major computational cost.

Alongside our case study, we index additional adaptive test-time defenses in the appendix. In contemporary work, Chen et al. (2022) carry out a complementary analysis of transductive test-time defenses (Wu et al., 2020b; Wang et al., 2021a), which depend on multiple test inputs and even joint optimization across training and testing data.

3.1. Evaluation Methods

For completeness, we give a short overview of the attacks and techniques used in our case study. More details are described in the evaluation of each defense. **AUTOATTACK** (Croce & Hein, 2020) is a common benchmark for evaluating static defenses, but it is not designed for adaptive test-time defenses. **AutoPGD (APGD)** (Croce & Hein, 2020) is a variant of Projected Gradient Descent (PGD)—one of the most popular technique for ℓ_p-norm bounded adversarial attacks. Its varied surrogate losses include cross-entropy, Carlini-Wagner (CW), margin (Carlini & Wagner, 2017a), or the (targeted) Difference of Logits Ratio (DLR) (Croce & Hein, 2020). **RayS** (Chen & Gu, 2020) is a decision-based attack designed for ℓ_∞-norm bounded perturbations. It only requires the label predicted by the model. **BPDA** (Athalye et al., 2018) permits the attack of non-differentiable defenses by approximating them with differentiable functions during gradient computation. The identity is a common approximation. **EoT** (Athalye et al., 2018) permits the attack of randomized defenses. The predictions and gradients are computed in expectation over the randomness of the model, approximated by averaging the results of multiple runs with the same input. **Transfer attacks** generate adversarial perturbations on a surrogate model and use them on the target model.

3.2. Stable Neural ODE with Lyapunov-Stable Equilibrium Points for Defending against Adversarial Attacks (Kang et al., 2021)

Summary of method. This defense applies a neural Ordinary Differential Equation (ODE) layer to improve robustness against ℓ_p-norm bounded attacks by the stability of equilibrium points of the underlying ODE. The proposed SODEF model consists of a feature extractor followed by a neural ODE layer and finally one or multiple fully connected layers. The feature extractor is a standard convolutional network, while the neural ODE can be solved with numerical methods (Runge-Kutta of order 5 in this case) up to some integration time T.

Evaluation. While Kang et al. consider many scenarios in their experimental evaluation, we test SODEF when the feature extractor consists of a robust classifier, trained with TRADES (Zhang et al., 2019) against ℓ_∞-norm perturbations of size $8/255$ on CIFAR-10, without the last linear layer (since checkpoints are available online). We use the original implementation with corresponding parameters, including integration time $T = 5$. A significant improvement in robustness against ℓ_∞- and ℓ_2-norm perturbations is reported (57.76% and 67.75%) when compared to the original TRADES model (53.69% and 59.42%), under the evaluation of AUTOATTACK. In particular, AUTOATTACK is used both directly on the SODEF model and as the basis for a transfer attack from the TRADES classifier. However, we note that by default AUTOATTACK returns the original images when they are originally misclassified or when no adversarial perturbation is found. This means that, in practice, most of the transferred points are clean images (since the TRADES model is highly robust). Moreover, AUTOATTACK does not aim at maximizing the confidence in the misclassification after this is achieved and, as such, it might be not the strongest method for transfer

Table 2. Robust accuracy on 1000 points against ℓ_p-norm bounded attacks: TRADES is tested with AUTOATTACK, and SODEF with transferring APGD from the static TRADES model.

Threat model	Static AA	with SODEF	
		AA (transf.)	APGD (transf.)
ℓ_∞	53.9%	60.9%	52.2%
ℓ_2	59.1%	66.8%	58.2%

attacks. As a consequence, we use APGD to maximize different target losses on all clean inputs using the TRADES classifier and then test them on SODEF. We run APGD for 100 iterations with cross-entropy, Carlini-Wagner (CW) and targeted DLR (with 3 target classes) loss, and consider the transfer attack successful if any of the resulting points fools the SODEF model. Table 2 shows the results of the described experiments for ℓ_∞- and ℓ_2-norm bounded attacks (on 1000 test images with $\epsilon = 8/255$ and $\epsilon_2 = 0.5$, respectively). When transferring inputs that maximize the surrogate losses, the robustness of SODEF is lower than that of the original TRADES classifiers.

3.3. Towards Robust Neural Networks via Close-loop Control (Chen et al., 2021)

Summary of method. Using the assumption that the representation of an adversarial input becomes progressively more corrupted as it moves through the layers of a deep network, this defense corrects its trajectory by adding offsets to intermediate activations of a pre-trained static model. At each layer i, *control parameters* $\boldsymbol{u}_i$ are added to its output. These control parameters are optimized to reduce the reconstruction error between the original activations and corresponding "reduced" activations. These reduced activations are encoded onto a lower-dimensional manifold and decoded back to the full state space using Principal Component Analysis (PCA) or shallow auto-encoders. At inference time, the parameters $\boldsymbol{u}_i$ are jointly optimized with multiple steps of gradient descent and then used to produce the final prediction.

Evaluation. Chen et al. report 11% robust accuracy against PGD attacks of size $\epsilon = 8/255$. This is not competitive with state-of-art robust classifiers and already hints that the defense may be ineffective. As such, we focus on their strongest result, at the smaller $\epsilon = 2/255$, where they report 50% robust accuracy. Even this reduced claim is voided by further evaluation. Crucially, their attack is not adaptive, as it does not exploit knowledge of the defense. Chen et al. argue that adaptive attacks are infeasible, as they would need to steer the control parameters $\boldsymbol{u}_i$ toward zero. They also argue that calculating the gradients of the optimization process is too difficult. Neither argument is sound, as the defense might still fail for nonzero values

of $\boldsymbol{u}_i$, and obfuscated gradients can be approximated. We demonstrate this by combining PGD and BPDA.

As pre-trained models are not publicly available, we train our own ResNet-20 model, and learn the linear embeddings with PCA using the code available. Unfortunately, some hyperparameters are not documented, and so we find a working configuration (using 5 iterations and a learning rate of $5 \cdot 10^{-3}$) by grid search that approximately reproduces the results of the paper. The resulting defense obtains 89.1% accuracy on clean images. We also reproduce their 20-step PGD evaluation by attacking the underlying classifier and testing the adversarial examples against their defense, and measure a robust accuracy of 34.5% (compared with 0% for the static model). The robustness gain is slightly less than reported (i.e., 50%) but still enough for our demonstration.[2] Each prediction takes 4 backward passes and 5 forward passes.

We now mount an adaptive attack. At test-time, the defense runs multiple iterations of gradient descent to optimize the control parameters $\boldsymbol{u}_i$. The optimized $\boldsymbol{u}_i^*$ are then used for the final prediction. Since $\boldsymbol{u}_i^*$ is merely added to the activations of the i-th layer, it is possible to derive an adversarial gradient for this final forward pass. Although this can only approximate the gradient of the full defense, we find that it suffices to drastically improve the attack success rate. We use this gradient with BPDA and repeat the same 20-step PGD attack. The dynamic model now achieves only 9.5% robust accuracy (instead of 34.5%). Finally, the gradient approximation also enables attack by APGD with BPDA, which drops the robust accuracy further to 5.6%.

3.4. Attacking Adversarial Attacks as a Defense (Wu et al., 2021)

Summary of method. The HD proposed by Wu et al. aims to defend against adversarial perturbations by *maximizing* the cross-entropy loss of the classifier's predictions summed over all classes. The intuition is that the images that are wrongly classified have stronger gradients, which dominate the optimization and easily reduce the confidence in the initial decision, while correctly classified points are minimally impacted by the defense. The proposed method does not require modifications of the training scheme of the underlying static model. HD solves, for a classifier f,

$$\arg \max_{\boldsymbol{\delta}} \sum_{k=1}^{K} L_{\text{ce}}(f(\boldsymbol{x} + \boldsymbol{\delta}), k), \qquad (3)$$

$$\text{with } \|\boldsymbol{\delta}\|_\infty \leq \epsilon_{\text{d}}, \ \boldsymbol{x} + \boldsymbol{\delta} \in [0, 1]^d$$

with 20 steps of PGD (and random initialization), $\epsilon_{\text{d}} = 8/255$ (the same used by the attacker) and step size $\eta =$

---

[2]This is the only configuration matching the reported accuracy on clean images while having non-trivial robustness against PGD.

Table 3. Robust accuracy on 1000 points of various static models (measured by APGD) and their adaptive version with HD (using 20 steps), where we transfer APGD attacks obtained on the static model and APGD with BPDA and EoT attacks from the model using HD with 5 steps. We also include the worst-case among both transfer attacks.

Underlying model	Static defense	Transfer to HD with 20 steps using as surrogate		
		Static	HD (5 steps)	Worst-case
Gowal et al. (2020)	63.0%	$65.9\% \pm 0.09$	$61.1\% \pm 0.14$	$61.0\% \pm 0.14$
Carmon et al. (2019)	59.2%	$65.5\% \pm 0.34$	$59.6\% \pm 0.22$	$59.2\% \pm 0.32$
Andriushchenko & Flammarion (2020)	45.0%	$54.1\% \pm 0.15$	$45.8\% \pm 0.12$	$45.2\% \pm 0.28$

$4/255$. This preprocesses the input to the classifier f, and therefore applies to any model. Inference with HD costs 21 forward and 20 backward passes of the model.

Evaluation. The main experimental evaluation in Wu et al. (2021) consists in transferring perturbations generated by several attacks on the underlying classifier f to the model equipped with HD. This results in absolute improvements of 2% to 4% in robust accuracy. An adaptive attack, equivalent to BPDA, is developed on the full defense, but it does not consider the randomness of the defense. This attack is reported to slightly reduce the effectiveness of HD on one model. The official code is not provided. Hence, we implement the defense algorithm ourselves using the details available in the paper.

Wu et al. report that PGD is often more effective than stronger methods like AUTOATTACK. We hypothesize that this is caused by the original implementation of AUTOAT-TACK which returns copies of unperturbed inputs when the attack is unsuccessful or when the original input is already misclassified. However, HD might also turn an originally correctly classified input into a misclassified one, and this is more likely if it is close to the decision boundary (as we expect an unsuccessful adversarial example to be). Ultimately, we use APGD on the targeted DLR loss with 5 target classes (that is 5 restarts of 50 iterations). We use BPDA and reduce the number of steps of HD to 5 when crafting the perturbations. To counter the randomness of the initial step of HD, we use 4 steps of EoT. We evaluate the obtained perturbations on the full defense with 20 steps. In Table 3, we report the robust accuracy obtained with our evaluation on static models available in `RobustBench` (Croce et al., 2021). We also report the robust accuracy obtained by transferring adversarial perturbations from the underlying static models. We observe that HD does not provide clear improvements to the robustness of static models, and in one case it weakens it.

3.5. Online Adversarial Purification based on Self-Supervision (Shi et al., 2020)

Summary of method. This defense purifies adversarial perturbations by minimizing an auxiliary loss before performing inference. The auxiliary loss is connected to a self-supervised task, which may require an additional network which shares some feature representation with the classifier. Since the auxiliary loss is also optimized at training time, the system should learn to perform classification based on robust features that are shared across the supervised and auxiliary tasks. In the evaluation of Shi et al. (2020), Fast Gradient Sign Method (FGSM) is often a stronger attack than PGD, which contradicts the principles of Carlini et al. (2019).

We focus on the defense variant that uses label consistency as auxiliary task since it does not require an additional network at test-time and yields the best results on CIFAR-10. It uses as auxiliary loss

$$L_{\text{aux}}(f, \boldsymbol{x}) = \|f(a_1(\boldsymbol{x})) - f(a_2(\boldsymbol{x}))\|_2, \quad (4)$$

with f the classifier and a_1, a_2 augmentations of the input. At test-time, the problem

$$\arg\min_{\boldsymbol{\delta}} L_{\text{aux}}(f, \boldsymbol{x} + \boldsymbol{\delta}), \quad (5)$$
$$\text{with } \|\boldsymbol{\delta}\|_\infty \leq \epsilon_{\text{d}}, \ \boldsymbol{x} + \boldsymbol{\delta} \in [0, 1]^d$$

is optimized with 5 steps of PGD (without random initialization). The procedure is repeated for eleven values of ϵ_{d} and that attaining the lowest loss is chosen. Overall, inference requires $5 \cdot 11 \cdot 2$ forward and backward passes during purification plus 1 forward pass for the final classification.

Evaluation. We consider the small pre-trained model (referred to as ResNet-18 in Shi et al., 2020). It achieves a clean accuracy of 83.7% (after purification) on the first 1000 test images of CIFAR-10. The original evaluation reports for this model a robust accuracy of 51.02% under transfer from an FGSM attack on the static model. We run APGD with BPDA on the cross-entropy loss, but, instead of using as update direction (the sign of) the gradient of f with respect to the final purified image only (as one would get by approximating the whole purification process with the identity function), we average gradients over intermediate iterates produced by the purification process. Intuitively, this steers all intermediate images towards misclassification and make the attack more effective. Running this attack with 1000 iterations reduces the robust accuracy of the defense to 3.7%.

3.6. Adversarial Attacks are Reversible with Natural Supervision (Mao et al., 2021)

Summary of method. This defense states that images contain intrinsic structure that enables the reversal of adversarial attacks. In particular, they modify the inference step to purify the input using a trained contrastive representation g that leverages the intermediate activations of a pre-trained static model f. Given an input $\boldsymbol{x}$, the defense creates a modified input $\boldsymbol{x} + \boldsymbol{\delta}^{\star}$ where $\boldsymbol{\delta}^{\star}$ is the solution to

$$\arg\min_{\boldsymbol{\delta}} \; L_{\text{aux}}(g(t_1(\boldsymbol{x} + \boldsymbol{\delta})), g(t_2(\boldsymbol{x} + \boldsymbol{\delta}))), \quad (6)$$

$$\text{with } \|\boldsymbol{\delta}\|_{\infty} \leq 2\epsilon, \; \boldsymbol{x} + \boldsymbol{\delta} \in [0,1]^d,$$

and is found using N PGD steps. L_{aux} is the contrastive loss and t_1, t_2 are two transformations randomly sampled from a set of predefined image augmentations. The authors use either two or four augmentations for each image and leverage a separate set of images to build negative pairs (to be used in the contrastive loss). The paper remains unclear about the provenance of these additional images. The released code uses augmented views of other attacked images to build the negative pairs and thus operates at the level of a batch, which can both weaken the defense and attack. At inference, the modified input is fed to a static model. In practice, Mao et al. set $N = 40$.

Evaluation. According to Mao et al.'s evaluation their best model achieves a robust accuracy of 67.79% against AUTOATTACK, 64.64% against PGD with 50 steps and 63.83% against the Carlini-Wagner attack with 200 steps (Carlini & Wagner, 2017b) on CIFAR-10 against ℓ_{∞}-norm bounded perturbations of size $\epsilon = 8/255$. These attacks are performed on the underlying static model and transferred to the full defense. At first sight, it can seem surprising that AUTOATTACK is weaker since it consists of a suite of attacks with numerous restarts and many steps. However, by default AUTOATTACK returns the original images when these are originally misclassified or when no adversarial perturbation is found. This means that, in practice, most of the transferred points are clean images (since the underlying static model is already robust).

We replicate the setup from the authors which operates at the batch level. We use four image transformations and set the number of iterations to 40. For the purpose of this demonstration, and to be able to run our evaluation on a single NVIDIA V100 GPU, we reduce the batch size to 50 (instead of 512). While this change may negatively impact the defense, we found that transfer attacks match the results from Mao et al. (2021): Using AUTOATTACK from the static model yields a robust accuracy of 67.0% (compared to 67.79%); Using a custom implementation of APGD (which returns worst-case adversarial examples) on the cross-entropy and DLR losses with 10 steps and 10 restarts, we obtain a robust accuracy of 63.9%, which in line with the worst-case robust accuracy obtained by Mao et al. (i.e., 63.83%). We note that under the same APGD attack, the static model obtains a robust accuracy of 59.4%. When attacking the full adaptive defense, we use BPDA and 16 EoT iterations. We obtain a robust accuracy of 58.4% and conclude that the proposed defense weakens the underlying static model. To determine the effect of randomness, we also perform our evaluation by fixing the random seed (and removing EoT). Without randomness, we obtain a robust accuracy of 56.4%.

3.7. AID-purifier: A Light Auxiliary Network for Boosting Adversarial Defense (Hwang et al., 2021)

Summary of method. This defense, proposed by Hwang et al. (2021), uses a discriminator to purify the input of a pre-trained classifier. The discriminator is trained to distinguish adversarially perturbed from clean inputs. It exploits the intermediate activations of the underlying static model. At inference time, every input is purified by minimizing the probability of perturbation according to the discriminator, with the goal of reducing any adversarial effect the input may have.

Given a discriminator g, an input $\boldsymbol{x}$, $\epsilon_{\text{d}} > 0$, AID-purifier approximately solves the problem

$$\boldsymbol{\delta}^{\star} \approx \arg\min_{\boldsymbol{\delta}} \; g(\boldsymbol{x} + \boldsymbol{\delta}) \quad (7)$$

$$\text{with } \|\boldsymbol{\delta}\|_{\infty} \leq \epsilon_{\text{d}}, \; \boldsymbol{x} + \boldsymbol{\delta} \in [0,1]^d$$

with N steps of PGD and a fixed step-size α (the values of N and α vary across datasets). Inference is performed using the pre-trained classifier on the point $\boldsymbol{x} + \boldsymbol{\delta}^{\star}$. The inference costs N additional forward and backward passes of the discriminator compared to the standard one (in practice $T = 10$ is used).

Evaluation. The original evaluation relies mostly on transferring perturbations adversarial to the pre-trained classifier to the classifier endowed with purification. The authors also propose an adaptive attack which optimizes a convex combination of the classification loss and the output of the discriminator. This adaptive attack is not effective in decreasing the robust accuracy compared to the non-adaptive counterpart. We focus our analysis on static models trained using adversarial training (Madry et al., 2018) as their respective pre-trained discriminators are available publicly. We note that the reported improvement of AID-purifier on these models is rather small (1 to 2%) for all datasets, with the exception of SVHN where clean and robust accuracy increase by 22% and 27%, respectively. In the following, we consider the ℓ_{∞}-norm bounded perturbations of size $\epsilon = 8/255$ for CIFAR-10 and CIFAR-100, and of size $\epsilon = 12/255$ for SVHN. We evaluate the ro-

Table 4. Robust accuracy on 1000 points against ℓ_∞-norm bounded attacks of static models alone and with AID-purifier.

Dataset	ϵ	Static model	AID-purified model	
			Transfer	Direct
CIFAR-10	8/255	49.3%	56.0%	43.8%
CIFAR-100	8/255	23.3%	32.3%	18.2%
SVHN	12/255	10.5%	62.0%	29.1%

bustness of the static models with AUTOATTACK, and that of the dynamic models with APGD combined with BPDA. Table 4 shows that for CIFAR-10 and CIFAR-100 the dynamic classifiers have lower robustness than the original static models. While AID-purifier seems to improve robustness on SVHN, we notice that, unlike on the other datasets, the static model has a low robust accuracy.[3]

3.8. Combating Adversaries with Anti-Adversaries (Alfarra et al., 2022)

Summary of method. This defense tries to counter gradient-based based attacks during inference by maximizing the classifier's confidence. It uses a static model f to predict pseudo-labels $\hat{y} = \arg\max_k [f(\boldsymbol{x})]_k$ for each input $\boldsymbol{x}$, and optimizes $\boldsymbol{x}' = \arg\min_{\boldsymbol{x}'} L_{\text{ce}}(f(\boldsymbol{x}', \hat{y}))$ with N gradient descent steps to increase the classifier's confidence in $\hat{y}$ (with L_{ce} being the cross-entropy loss). The goal is to cancel out attacks that aim to decrease this confidence score. In practice, N is set to 2, which leads to an increase in inference time of $8\times$ (the original implementation also makes an additional unnecessary forward pass).

This design implies that the defense inherits the decision boundary of the underlying classifier, as increasing a classifier's confidence in its own predictions will not change its decision.[4] Consequently, there is no fundamental gain in robustness and any measured increase can only be the result of obfuscation: forcing the classifier to report high confidence almost everywhere flattens the loss landscape of the cross-entropy loss and causes numerical problems not only for PGD, but also for any score-based attack such as the Square attack (Andriushchenko et al., 2020).

Evaluation. We focus on the strongest result reported by Alfarra et al. which consists of applying their defense for two iterations to a robust static CIFAR-10 model pretrained with Adversarial Weight Perturbation (AWP) (Wu et al., 2020a). As noted above and as noted by Alfarra et al., the defense can easily be circumvented by transferring adversarial examples from the underlying static model. Using

---

[3]For reference, we can obtain a classifier with more than 40% robust and 86% clean accuracy by standard adversarial training.

[4]We observe that the decision boundary can sometimes change due to numerical inaccuracies. This effect is rare and disappears when reducing the defense step-size.

APGD on the cross-entropy loss, we can reduce the robust accuracy of both the static and adaptive model to 63.7% against ℓ_∞-norm bounded perturbations of size $\epsilon = 8/255$ on CIFAR-10.

However, we find that the defense is ineffective even against black-box decision-based attacks that require no knowledge of the static model at all. Testing the defense against a range of attacks (which exclude transfer attacks), Alfarra et al. report a robust accuracy of 79.21%. They also conduct an evaluation against decision-based attacks with inconclusive results as they report a robust accuracy of 86.0% against both the adaptive and static model, which is close the clean baseline of 88.0%. This suggests an incorrect use of the attack, or perhaps not enough iterations. As a consequence, we apply the more efficient RayS attack with 10K queries. We obtain a robust accuracy of 66.6%, which is much lower than the result reported by Alfarra et al. (i.e., 79.2%). We also obtain 66.6% against the static model, which shows that the defense has no effect against an attack that does not use confidence scores.

3.9. Adversarial Purification with Score-based Generative Models (Yoon et al., 2021)

Summary of method. This defense preprocesses the input of a classifier with an Energy-Based Model (EBM) trained, with Denoising Score-Matching (DSM), to learn a score function to denoise perturbed images. A particularity is that they allow only a few deterministic updates in the purification process (after adding noise to the initial input).

The proposed defense, named Adaptive Denoising Purification (ADP), purifies an input $\boldsymbol{x}$ with the iterative scheme, for $i = 1, \ldots, T$.

$$\boldsymbol{x}_0 = \boldsymbol{x} + \boldsymbol{\xi}, \quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_{i-i} s(\boldsymbol{x}_{i-1}) \quad (8)$$

with $\boldsymbol{\xi} \sim \mathcal{N}(0, \sigma^2 I)$ an initial noise addition, α_i an adaptive step-size, and s the score function approximated by the EBM. The procedure is repeated S times, getting S points $\boldsymbol{x}_T^{(1)}, \ldots, \boldsymbol{x}_T^{(S)}$. Given a classifier f, the final classification is obtained as

$$\arg\max_{k=1,\ldots,K} \frac{1}{S} \sum_{s=1}^{S} [\text{softmax}(f(x_T^{(s)})]_k. \quad (9)$$

In practice, Yoon et al. set $T = 10$, $S = 10$ and $\sigma = 0.25$, and f is a pre-trained classifier. Hence, the inference costs S forward passes of f and $2 \cdot T \cdot S$ forward passes of s (since an additional call of the score networks is necessary for the computation of α_i).

Evaluation. In their experimental evaluation, Yoon et al. (2021) report that ADP achieves 69.71% against ℓ_∞-norm bounded perturbations of size $\epsilon = 8/255$ on CIFAR-10.

The strongest attack is reported to be PGD with BPDA and EoT, where BPDA approximates the iterative purification process with an identity. However, we note BPDA is not necessary since the defense consists only in forward passes of the score network and of the classifier from which gradients can be computed. As such, we apply 10 steps of APGD combined with 200 EoT iterations to alleviate the randomness induced by ξ. This attack yields a robust accuracy of $33.7\% \pm 0.54$ averaged over 1000 images.[5]

4. Discussion and Recommendations

Our case study shows that evaluating adaptive test-time defenses is more challenging than evaluating static defenses because of several complicating factors. These factors include the possibly intricate optimization process, randomness, and the large computational requirements of running these defenses. Moreover, each defense has its own peculiarities, which frustrates attempts at standardizing evaluation. In fact, Table 1 shows that the most effective attack differs across cases. At the same time, the main elements constituting such attacks consist of techniques proposed in prior works, which can be adaptively combined depending on the elements of the defense. The main challenge is therefore to find the attack setup which is the most suitable for each case.

Carlini et al. (2019) proposed guidelines to evaluate adversarial robustness, and applied their principles to build a strong attack methodology. Moreover, they identify signs of overestimated robustness (e.g., single-step attacks stronger than multi-step ones), which also hold for adaptive test-time defenses. Our case study suggests taking the following steps to extend their foundational guidance:

❶ Transfer attacks that are effective against the underlying static model. Consider using other models as surrogates, such as those available in the RobustBench model zoo. When transferring attacks, make sure to transfer unsuccessful adversarial attacks (i.e., perturbations that attain high loss but do not lead to misclassification) rather than transferring the unperturbed input (as a library may do by default, e.g., AUTOATTACK).

❷ Verify with various black-box attacks that the adaptive test-time defense is more robust than the underlying static model. Consider Square (Andriushchenko et al., 2020) which is score-based and RayS (Chen & Gu, 2020) which is decision-based for this purpose.

❸ Apply strong white-box attacks to the full defense when possible. We found APGD (with multiple losses and restarts) to be a reliable gradient-based attack in most cases. Modern frameworks such as PyTorch (Paszke

et al., 2019) and JAX (Bradbury et al., 2018) enable easy gradient computation.

❹ When gradient attacks fail, due to non-differentiability or vanishing gradients caused by excessive iteration, consider combining APGD with BPDA and try various approximations for the backward pass. In particular, BPDA can simply replace the backward pass for the optimization process with the identity, or make use of intermediate iterates (see Sec. 3.5).

❺ When randomness is present, explictly or implictly, use EoT, or remove randomness altogether by fixing the seed at each attack step.

❻ Ultimately, although they can serve as baselines, attacks developed for static defenses are not guaranteed to be effective for adaptive test-time defenses. Consequently, always try to implement adaptive attacks that are specific to adaptive defenses. These attacks should be stronger than their non-adaptive counterparts, unlike the results reported in Mao et al. (2021) and Hwang et al. (2021).

5. Conclusion

Adaptive test-time defenses complicate robustness evaluation due to their complexity and computational cost. Despite these complications, our evaluation succeeds in reducing the apparent robustness of the studied defenses, with a relative reduction of more than 50% for four of the nine defenses. In all cases with an adversarially robust static model (five out of nine), the adaptive test-time defense does not improve upon it and might even weaken it. These results are disappointing, but we foresee that adaptive test-time defenses can still potentially lead to significant robustness gains, and clearly our results do not challenge the whole idea. While we could not provide a standardized evaluation protocol for adaptive test-time defenses, we hope that our case study and recommendations can guide future evaluations as new defenses are developed. Furthermore, we emphasize the need to measure gains in robustness against the computation required. This weakness may be turned into a potential strength, if the additional computation used by the defender can impose even more computation on the attacker. Finally, we note that the code developed for our case study is available at https://github.com/fra31/evaluating-adaptive-test-time-defenses.

Acknowledgements

---

[5]Average and standard deviation are computed over 5 repeated evaluations over ADP of the adversarially perturbed points.

References

Alet, F., Bauza, M., Kawaguchi, K., Kuru, N. G., Lozano-Perez, T., and Kaelbling, L. P. Tailoring: encoding inductive biases by optimizing unsupervised objectives at prediction time. In *Adv. Neural Inform. Process. Syst.*, 2021.

Alfarra, M., Perez, J. C., Thabet, A., Bibi, A., Torr, P., and Ghanem, B. Combating adversaries with anti-adversaries. In *AAAI*, 2022.

Andriushchenko, M. and Flammarion, N. Understanding and Improving Fast Adversarial Training. *Adv. Neural Inform. Process. Syst.*, 2020.

Andriushchenko, M., Croce, F., Flammarion, N., and Hein, M. Square Attack: a query-efficient black-box adversarial attack via random search. *Eur. Conf. Comput. Vis.*, 2020.

Athalye, A. and Sutskever, I. Synthesizing robust adversarial examples. *Int. Conf. Mach. Learn.*, 2018.

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *Int. Conf. Mach. Learn.*, 2018.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases*, 2013.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., and Wanderman-Milne, S. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Buckman, J., Roy, A., Raffel, C., and Goodfellow, I. Thermometer Encoding: One Hot Way To Resist Adversarial Examples. In *Int. Conf. Learn. Represent.*, 2018.

Carlini, N. and Wagner, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14. ACM, 2017a.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy*, 2017b.

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., and Madry, A. On Evaluating Adversarial Robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., and Liang, P. S. Unlabeled data improves adversarial robustness. In *Adv. Neural Inform. Process. Syst.*, 2019.

Chen, J. and Gu, Q. RayS: A ray searching method for hard-label adversarial attack. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1739–1747. ACM, 2020.

Chen, J., Wu, X., Guo, Y., Liang, Y., and Jha, S. Towards evaluating the robustness of neural networks learned by transduction. In *Int. Conf. Learn. Represent.*, 2022.

Chen, Z., Li, Q., and Zhang, Z. Towards robust neural networks via close-loop control. In *Int. Conf. Learn. Represent.*, 2021.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. *Int. Conf. Mach. Learn.*, 2020.

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. Robustbench: a standardized adversarial robustness benchmark. *Adv. Neural Inform. Process. Syst. Datasets and Benchmarks Track*, 2021.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. URL http://www.deeplearningbook.org.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *Int. Conf. Learn. Represent.*, 2015.

Gowal, S., Qin, C., Uesato, J., Mann, T., and Kohli, P. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

Gowal, S., Rebuffi, S.-A., Wiles, O., Stimberg, F., Calian, D. A., and Mann, T. Improving robustness using generated data. In *Adv. Neural Inform. Process. Syst.*, 2021.

Guo, C., Rana, M., Cisse, M., and van der Maaten, L. Countering Adversarial Images using Input Transformations. In *Int. Conf. Learn. Represent.*, 2018.

Gurumurthy, S., Bai, S., Manchester, Z., and Kolter, J. Z. Joint inference and input optimization in equilibrium networks. In *Adv. Neural Inform. Process. Syst.*, 2021.

Hill, M., Mitchell, J., and Zhu, S.-C. Stochastic security: Adversarial defense using long-run dynamics of energy-based models. In *Int. Conf. Learn. Represent.*, 2021.

Hwang, D., Lee, E., and Rhee, W. Aid-purifier: A light auxiliary network for boosting adversarial defense. In *ICML Workshop on Adversarial Machine Learning*, 2021.

Kang, Q., Song, Y., Ding, Q., and Tay, W. P. Stable neural ode with lyapunov-stable equilibrium points for defending against adversarial attacks. *Adv. Neural Inform. Process. Syst.*, 34, 2021.

Kannan, H., Kurakin, A., and Goodfellow, I. Adversarial Logit Pairing. *arXiv preprint arXiv:1803.06373*, 2018.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *ICLR Workshop*, 2016.

Li, Y., Bradshaw, J., and Sharma, Y. Are generative classifiers more robust to adversarial attacks? In *Int. Conf. Mach. Learn.*, 2019.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *Int. Conf. Learn. Represent.*, 2018.

Mao, C., Chiquier, M., Wang, H., Yang, J., and Vondrick, C. Adversarial attacks are reversible with natural supervision. In *Int. Conf. Comput. Vis.*, 2021.

Nandy, J., Saha, S., Hsu, W., Lee, M. L., and Zhu, X. X. Adversarially trained models with test-time covariate shift adaptation. *arXiv preprint arXiv:2102.05096*, 2021.

Nie, W., Guo, B., Huang, Y., Xiao, C., Vahdat, A., and Anandkumar, A. Diffusion models for adversarial purification. In *Int. Conf. Mach. Learn.*, 2022.

Pang, T., Xu, K., and Zhu, J. Mixup inference: Better exploiting mixup to defend adversarial attacks. In *Int. Conf. Learn. Represent.*, 2020.

Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. *IEEE Symposium on Security and Privacy*, 2016.

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., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Adv. Neural Inform. Process. Syst.*, 2019.

Perez Tobia, J., Braun, P., and Narayan, A. AGS: Attribution guided sharpening as a defense against adversarial attacks. In *International Symposium on Intelligent Data Analysis*, pp. 225–236. Springer, 2022.

Pintor, M., Demetrio, L., Sotgiu, A., Manca, G., Demontis, A., Carlini, N., Biggio, B., and Roli, F. Indicators of attack failure: Debugging and improving optimization of adversarial examples. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021.

Qian, Z., Zhang, S., Huang, K., Wang, Q., Zhang, R., and Yi, X. Improving model robustness with latent distribution locally and globally. *arXiv preprint arXiv:2107.04401*, 2021.

Rade, R. and Moosavi-Dezfooli, S.-M. Helper-based adversarial training: Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021.

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., and Mann, T. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021.

Rice, L., Wong, E., and Kolter, J. Z. Overfitting in adversarially robust deep learning. *Int. Conf. Mach. Learn.*, 2020.

Rusu, A. A., Calian, D. A., Gowal, S., and Hadsell, R. Hindering adversarial attacks with implicit neural representations. In *Int. Conf. Mach. Learn.*, 2022.

Samangouei, P., Kabkab, M., and Chellappa, R. Defensegan: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018.

Schott, L., Rauber, J., Bethge, M., and Brendel, W. Towards the first adversarially robust neural network model on mnist. In *Int. Conf. Learn. Represent.*, 2019.

Shi, C., Holtz, C., and Mishne, G. Online adversarial purification based on self-supervised learning. In *Int. Conf. Learn. Represent.*, 2020.

Song, Y. and Ermon, S. Improved techniques for training score-based generative models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Adv. Neural Inform. Process. Syst.*, 2020.

Song, Y., Kim, T., Nowozin, S., Ermon, S., and Kushman, N. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *Int. Conf. Learn. Represent.*, 2018.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-time training with self-supervision for generalization under distribution shifts. In *Int. Conf. Mach. Learn.*, 2020.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *Int. Conf. Learn. Represent.*, 2014.

Tramèr, F., Carlini, N., Brendel, W., and Madry, A. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

Uesato, J., O'Donoghue, B., Oord, A. v. d., and Kohli, P. Adversarial Risk and the Dangers of Evaluating Against Weak Attacks. *Int. Conf. Mach. Learn.*, 2018.

Wang, D., Ju, A., Shelhamer, E., Wagner, D., and Darrell, T. Fighting gradients with gradients: Dynamic defenses against adversarial attacks. *arXiv preprint arXiv:2105.08714*, 2021a.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. In *Int. Conf. Learn. Represent.*, 2021b.

Wu, B., Pan, H., Shen, L., Gu, J., Zhao, S., Li, Z., Cai, D., He, X., and Liu, W. Attacking adversarial attacks as a defense. *arXiv preprint arXiv:2106.04938*, 2021.

Wu, D., Xia, S., and Wang, Y. Adversarial weight perturbation helps robust generalization. In *Adv. Neural Inform. Process. Syst.*, 2020a.

Wu, Y.-H., Yuan, C.-H., and Wu, S.-H. Adversarial robustness via runtime masking and cleansing. In *Int. Conf. Mach. Learn.*, 2020b.

Yang, Y., Zhang, G., Katabi, D., and Xu, Z. Me-net: Towards effective adversarial robustness with matrix estimation. In *Int. Conf. Mach. Learn.*, 2019.

Yoon, J., Hwang, S. J., and Lee, J. Adversarial purification with score-based generative models. In *Int. Conf. Mach. Learn.*, 2021.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically Principled Trade-off between Robustness and Accuracy. *Int. Conf. Mach. Learn.*, 2019.

A. Case Study of Adaptive Test-time Defenses

We evaluate the remaining defense included in our case study, which is only reported here due to space constraints.

A.1. Improving Model Robustness with Latent Distribution Locally and Globally (Qian et al., 2021)

Summary of method. This defense introduces a new technique called Adversarial Training with Latent Distribution (ATLD). ATLD builds adversarial examples by fooling a discriminator network g. The discriminator has $C + 1$ outputs, where the zero-th output distinguishes between clean and adversarial images, while the remaining outputs predict the class. In a process similar to Madry et al. (2018), adversarial examples are used within a cross-entropy loss to train a static model. The claimed benefit of this approach is that adversarial examples are not biased towards the decision boundary of the classifier. The experiments demonstrate that the resulting model (denoted ATLD-) is more robust to weak adversarial attacks such as PGD with 10 steps. In addition to ATLD, the authors also propose a modification to the inference process called Inference with Manifold Transformation (IMF). They modify inference by repairing inputs x to $x + \delta^\star$ such that the trained discriminator is more likely to classify them as clean examples. The optimal purification offset $\delta^\star$ is found using PGD with N steps to solve the problem

$$\arg\min_{\delta} L_{\text{softmax}}[g(x + \delta)]_0, \qquad (10)$$

$$\text{with } \|\delta\|_\infty \leq \epsilon_{\text{d}}, \ x + \delta \in [0, 1]^d.$$

The resulting models, dubbed ATLD and ATLD+, are seemingly more robust to evaluation pipelines like AUTOATTACK (reaching 65.07% robust accuracy against ℓ_∞-norm bounded perturbations of size $\epsilon = 8/255$). In practice, Qian et al. set $N = 1$ and $\epsilon_{\text{d}} = 2\epsilon$.

Evaluation. As noted by Qian et al., the ATLD- model is only robust to weak attacks and completely breaks under AUTOATTACK. In our evaluation, using APGD on cross-entropy and targeted DLR losses (with 9 targets), the robust accuracy of the ATLD- model reduces to 7.7% (compared to 65.40% against CW with 100 steps). The authors evaluate their full defense, which includes IMF, by transferring attacks from various static models. Surprisingly, transferring from the underlying ATLD- model results in weaker attacks than transferring from another model trained through adversarial training (Madry et al., 2018). This suggests that the underlying static model obfuscates gradients. Consequently, we increase the number of restarts to 100. Since the defense consists of a single differentiable step, we attack the full defense with APGD on the cross-entropy and targeted DLR losses. Overall, we

obtain a robust accuracy of 12.6% for the ATLD+ model, which is significantly lower than reported (i.e., 65.07%).

B. Additional Evaluation Details

We further detail the evaluation of the defenses described in our case study. Please note that the code for each evaluation is available for clarity and reproducibility.

B.1. Stable Neural ODE with Lyapunov-Stable Equilibrium Points for Defending against Adversarial Attacks (Kang et al., 2021)

We use the original implementation of the defense and the corresponding checkpoints which are publicly available.[6] In the main manuscript (Sec. 3.2), we use a WideResNet-34-10 as the underlying architecture for the static model. As a point of comparison, we also evaluate SODEF when combined with a larger WideResNet-70-16. The base model from Rebuffi et al. (2021) achieves a robust accuracy of 66.56%. According to Kang et al. (2021), this model combined with SODEF achieves 71.28% when evaluated with AUTOATTACK. Similarly to what is described in Sec. 3.2, we use APGD to maximize different loss functions (cross-entropy, margin and targeted DLR with 9 restarts) on the base model, and transfer the obtained perturbations to the SODEF model. This yields a robust accuracy of 65.02%, which can be further decreased to 64.20% by taking the worst-case between the transfer attack and AUTOATTACK.

B.2. Towards Robust Neural Networks via Close-loop Control (Chen et al., 2021)

We use the official implementation of the defense, which is publicly available.[7] At the time of our evaluation, some hyperparameter choices in the code deviated from those in the paper while others were included in neither. Hoping to reproduce the results in the paper, we evaluate the following configuration:

- We use 5000 images from the CIFAR-10 training set to learn the linear projection.
- We set the regularization strength (decaying the magnitude of control parameters) to zero, disabling regularization. This matches the published code. We do not observe any notable effect from using regularization.
- We set the number of defense iterations to $N = 5$ and learning rate to $\alpha = 10^{-3}$, which we find to be the only configuration that is close to the robustness results claimed in the paper, while at the same time retaining

---

[6] https://github.com/kangqiyu/sodef
[7] https://github.com/zhuotongchen/Towards-Robust-Neural-Networks-via-Close-loop-Control

good accuracy on clean images.

- We apply the defense to a ResNet-20 trained with the provided code. As no pre-trained checkpoints are available, we refrain from performing costly adversarial training and evaluate only on this normally-trained model.

For the attack, we first run PGD with 20 steps of size $\frac{\epsilon}{4}$, mirroring the evaluation by Chen et al. (2021) but adding BPDA. We then run APGD with BPDA using the cross-entropy loss for 100 iterations (no restarts).

B.3. Attacking Adversarial Attacks as A Defense (Wu et al., 2021)

The official code for HD is not available. As such, we implement it by following the details provided by Wu et al. (2021). The models used in our evaluation are the WideResNet-28-10 trained with extra data (Gowal et al., 2020; Carmon et al., 2019) and the Pre-Activation ResNet-18 (Andriushchenko & Flammarion, 2020). As stated in Sec. 3.4, the default implementation of APGD (within the AUTOATTACK library) returns adversarial points only when they are successful (i.e., when misclassification occurs). When we use this default implementation, the HD model which uses the WideResNet-70-16 from Gowal et al. (2020) obtains 69.2% $\pm$ 0.10 (compared to 65.9% when using the points maximizing the target loss, see Table 3), which is in line with what reported by Wu et al. (2021) for the same classifier. This gives us confidence that our implementation matches the one from Wu et al..

B.4. Combating Adversaries with Anti-Adversaries (Alfarra et al., 2022)

We use the official implementation of the defense, which is publicly available.[8] For the underlying classifier, we use a WideResNet-28-10 trained with AWP (Wu et al., 2020a), with pre-trained weights obtained from the official AWP code repository.[9] Specifically, we use the RST-AWP checkpoint, which is also referenced by Alfarra et al. (2022) in their evaluation code. Further following the implementation by Alfarra et al. (2022), we set the number of defense iterations to $N = 2$ and the step size to $\alpha = 0.15$.

B.5. Online Adversarial Purification based on Self-Supervision (Shi et al., 2020)

We use the official implementation of the defense and the pre-trained model both publicly available.[10] Moreover, when transferring FGSM attacks from the static model, the

defended adaptive classifier obtains a robust accuracy of 54.5%, which is in line with what is reported by Shi et al. (2020).

B.6. Improving Model Robustness with Latent Distribution Locally and Globally (Qian et al., 2021)

We use the official implementation of the defense and the pre-trained models publicly available.[11] All hyperparameters are set as specified by Qian et al. (2021). Our only modification consists in allowing gradients to flow through the optimization steps. We use the standard implementation of AUTOATTACK.

B.7. AID-purifier: A Light Auxiliary Network for Boosting Adversarial Defense (Hwang et al., 2021)

In our evaluation, we use the pre-trained classifiers, with WideResNet-34-10 as architecture, and discriminators, as well as the publicly available[12] original implementation of the defense. For the parameters of AID-purifier we use the values provided by Hwang et al. (2021). Since on SVHN the defense seems to be beneficial, we further test it with APGD with 1000 iterations and 10 restarts (divided between cross-entropy and targeted DLR loss). This reduces the robust accuracy to 25.0%, suggesting that the robustness can be reduced by increasing the budget available to the attacker.

B.8. Adversarial Attacks are Reversible with Natural Supervision (Mao et al., 2021)

We use the official implementation and pre-trained classifier publicly available.[13] We focus our evaluation on the Semi-SL model (from Carmon et al., 2019). The original implementation of the defense retrieves negative pairs of images (to construct the contrastive loss) from the batch of images produced by the attacker. As such, we follow the same protocol and allow the attacker to modify all the images of each batch. Unfortunately, as the setup from Mao et al. (2021) uses a batch size of 512 and requires several GPUs to work in parallel, we are forced to change the default batch size to 50, which may negatively impact the defense. Other than that we keep all hyperparameters identical.

To evaluate whether the attacker obtains an unfair advantage by allowing it to modify the whole batch, we also re-implement our own version of the input purification pro-

---

[8] https://github.com/MotasemAlfarra/Combating-Adversaries-with-Anti-Adversaries
[9] https://github.com/csdongxian/AWP
[10] https://github.com/Mishne-Lab/SOAP

[11] https://github.com/LitterQ/ATLD-pytorch
[12] https://openreview.net/forum?id=3Uk9_JRVwiF
[13] https://github.com/cvlab-columbia/SelfSupDefense

cedure described by Mao et al. (2021) with the negative images kept separate and hidden from the attacker. In that setup, we were unable to improve upon the robust accuracy of the underlying static model when using transfer attacks (using APGD on the static model) to the contrary of the full-batch setup.

B.9. Adversarial Purification with Score-based Generative Models (Yoon et al., 2021)

As base classifier we use the WideResNet-28-10 model (as evaluated by Yoon et al., 2021) from RobustBench. The score network is provided by Song & Ermon (2020) online[14] (we use the checkpoint named best_checkpoint_with_denoising). The ADP classifier attains a clean accuracy of $86.9\% \pm 0.44$ on the 1000 points used for testing robustness. In our evaluation, we exclude the computation of the adaptive step size when getting the gradient of the target loss via backpropagation for simplicity (including it did not yield any improvements).

C. Index of Adaptive Test-time Defenses

Although our case study is necessarily finite, we provide further perspective by summarizing additional adaptive test-time defenses that are not included in our evaluation. Table 5 indexes additional defenses known to us at publication time with the same categorization as Table 1.

In general, these defenses were chronologically excluded from our evaluation: the older defenses have already been analyzed while the newer defenses have appeared concurrently or following our own work. While each defense adapts in its own way, our case study spans their principles and building blocks, and so we hope it can guide the further examination of future adaptive test-time defenses.

Table 5. Summary of additional adaptive test-time defenses. We categorize each defense by its principles—input purification (IP) and model adaptation (MA)—and its building blocks: iterative algorithm (IA), auxiliary network (AN), randomization (R), and external data (ED). Our case study covers defenses of each principle and building block but we encourage examining the full set for more perspective.

Defense	Venue	Principles		Building blocks			
		IP	MA	IA	AN	R	ED
Song et al. (2018)	ICLR	●		●	●		
Samangouei et al. (2018)	ICLR	●		●	●	●	
Guo et al. (2018)	ICLR	●		●		●	
Schott et al. (2019)	ICLR		●	●		●	
Wu et al. (2020b)	ICML	●		●		●	●
Hill et al. (2021)	ICLR	●		●	●	●	
Wang et al. (2021a)	arXiv		●	●		●	
Nandy et al. (2021)	arXiv	●		●		●	
Gurumurthy et al. (2021)	NeurIPS		●	●			
Alet et al. (2021)	NeurIPS		●	●		●	
Chen et al. (2022)	ICLR		●	●		●	●
Rusu et al. (2022)	ICML	●		●	●	●	
Nie et al. (2022)	ICML	●		●	●	●	
Perez Tobia et al. (2022)	ISIDA	●		●			

---

[14]https://github.com/ermongroup/ncsnv2