
ZeroPur: Succinct Training-Free Adversarial Purification

**Xiuli Bi¹ Zonglin Yang¹ Bo Liu¹ Xiaodong Cun²
Chi-Man Pun³ Pietro Lio⁴ Bin Xiao^{1*}**

¹Chongqing University of Posts and Telecommunications

²Tencent AI Lab ³University of Macau ⁴University of Cambridge

Abstract

Adversarial purification is a kind of defense technique that can defend various unseen adversarial attacks without modifying the victim classifier. Existing methods often depend on external generative models or cooperation between auxiliary functions and victim classifiers. However, retraining generative models, auxiliary functions, or victim classifiers relies on the domain of the fine-tuned dataset and is computation-consuming. In this work, we suppose that adversarial images are outliers of the natural image manifold and the purification process can be considered as returning them to this manifold. Following this assumption, we present a simple adversarial purification method without further training to purify adversarial images, called ZeroPur. ZeroPur contains two steps: given an adversarial example, Guided Shift obtains the shifted embedding of the adversarial example by the guidance of its blurred counterparts; after that, Adaptive Projection constructs a directional vector by this shifted embedding to provide momentum, projecting adversarial images onto the manifold adaptively. ZeroPur is independent of external models and requires no retraining of victim classifiers or auxiliary functions, relying solely on victim classifiers themselves to achieve purification. Extensive experiments on three datasets (CIFAR-10, CIFAR-100, and ImageNet-1K) using various classifier architectures (ResNet, WideResNet) demonstrate that our method achieves state-of-the-art robust performance. The code will be publicly available.

1 Introduction

Recent studies show that adding carefully crafted, imperceptible perturbations to natural examples can easily fool deep neural networks (DNNs) to make wrong decisions [11, 42]. This potential vulnerability underlying their remarkable performance raises a significant challenge for security-critical applications. Thus, designing efficient adversarial defense techniques is necessary for real-world applications in DNNs.

One kind of adversarial defense technique is *adversarial training* [23, 27, 28, 54], which involves adversarial examples in the model training, enabling the model to adapt adversarial perturbations empirically. However, these approaches typically require huge computational resources [37, 48] and suffer from performance degradation [24] in the presence of unseen attacks that are not involved in training. This limitation hinders the application of adversarial training in realistic scenarios.

Different from adversarial training, *adversarial purification* [29, 30, 38, 51] aims to remove adversarial perturbations in adversarial examples. These methods do not require adversarial examples in the model training and effectively defend against unseen attacks, making them more applicable in real-world scenarios. However, the existing purification methods often depend on external

*Corresponding authors

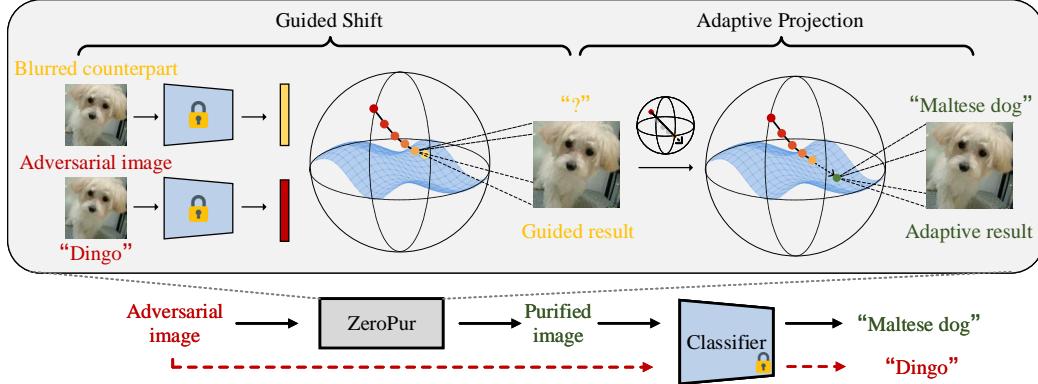


Figure 1: An illustration of ZeroPur.

generative models [10, 40] or cooperation between external auxiliary functions and the victim classifier [19, 29, 38]. Retraining generative models, parameterized auxiliary functions or the classifier relying on the domain of the fine-tuned dataset is computationally demanding and restricts their flexibility.

Inspired by the natural image manifold hypothesis, we suppose adversarial images are outliers of the natural image manifold and the purification process is to return them to this manifold. We present a simple adversarial purification method named ZeroPur, where "Zero" means our method does not need to train any external models, parameterized auxiliary functions, and victim classifiers. As illustrated in Fig. 1, given the adversarial image, ZeroPur comprises two stages, *i.e.*, Guided Shift (GS) and Adaptive Projection (AP), to purify the given image. In detail, GS guides adversarial examples towards the blurred counterparts to target natural images on the manifold to obtain the shifted embeddings. This is achieved by iteratively pulling the distance between adversarial examples and a set of their blurred counterparts in the embedding space. Since the inherent limitations of the low-quality embeddings from blurred counterparts, adversarial examples may not precisely return to the target locations, but these blurred counterparts do provide a reasonable direction. After that, we introduce AP to construct the directional vectors based on these shifted embeddings which provide a reference direction of the movement of adversarial images. The adversarial images are projected adaptively onto the manifold by projection maximization in this direction. Despite its simplicity, ZeroPur consistently outperforms most state-of-the-art adversarial training and purification methods.

The main contributions of the current work are as follows:

- We analyze the relationship between adversarial attack and adversarial purification based on the natural image manifold hypothesis, and show that a simple blurring operator can bring adversarial examples that are out of the natural image manifold closer to the manifold.
- We present a succinct adversarial purification approach named ZeroPur including two stages: Guided Shift and Adaptive Projection, which requires no retraining of external models, parameterized functions, and the victim classifier.
- Extensive experiments demonstrate that the proposed approach outperforms most state-of-the-art auxiliary-based adversarial purification methods and achieves competitive performance compared to other external model-based purification methods.

2 Review of Literature

Adversarial training (AT). AT [23, 27, 28, 54] improves the robustness of DNNs by integrating adversarial examples into the training data and reformulating the optimization objective. However, the computational cost of adversarial training is significantly huge due to the necessity of repeatedly performing backpropagation to craft adversarial examples. While recent works investigate reducing the time cost of adversarial training, they are still restricted by issues such as low robust performance [2, 37] and other unexpected results [2, 25] (*i.e.*, catastrophic overfitting). Moreover, even if a model is robust against a specific set of known attacks, it is still fragile against other unseen attacks that were not involved in training [8, 20, 24].

External model-based adversarial purification (EBP). Samangouei et al. [35] propose defense-GAN, a generator that models the distribution of natural images, which enables the transformation from adversarial examples to natural images. Song et al. [39] assume that adversarial examples primarily reside in the low probability density region of the training distribution, and design PixelDefend to approximate this distribution using the PixelCNN [44]. Recently, utilizing score-based models [51] and diffusion models [30, 45] as purification models is introduced and shown to achieve significantly improved robust performance. More recently, Lin et al. [26] propose a framework called AToP that fine-tunes the purification model adversarially to integrate the benefits of both adversarial training and adversarial purification. However, these works rely on external generative models, substantial datasets (e.g., crafting adversarial examples) and computational resources.

Auxiliary-based adversarial purification (ABP). In contrast to the above two approaches, ABP tends to introduce an auxiliary function to cooperate with the classifier to purify adversarial images. Shi et al. [38] propose a lightweight purification method SOAP, which uses self-supervised loss to realize online purification. SOAP no longer depends on generative models but requires classifiers to incorporate the corresponding auxiliary loss (self-supervised loss) in the training stage. To further reduce the time cost of purification, recent works introduce parameterized auxiliary functions, allowing purification to be accomplished through the exclusive training of these functions. Such auxiliary functions are designed to be lightweight compared to classifiers so that they can significantly reduce the time cost. For example, Mao et al. [29] introduce a two-layer network that estimates the contrastive features, which can purify the adversarial images. Hwang et al. [19] propose AID-Purifier, an auxiliary discriminator based on information maximization principles that can transform adversarial images into natural images.

3 ZeroPur

3.1 Adversarial Purification in The Natural Image Manifold

Following the natural image manifold hypothesis [17], natural images are assumed to reside on a specific manifold known as the natural image manifold. Given the classification loss function ℓ , the learning process of DNNs finding optimal parameters θ can be formulated by:

$$\min_{\theta} \ell(f \circ g(\mathbf{x} + \boldsymbol{\delta}^*), y; \theta), \quad (1)$$

where a natural image \mathbf{x} is embedded by an embedding function $f(\cdot) \in \mathbb{R}^d$ and then assigned its predicted label by a decision function $g(\cdot)$, and $y \in \mathbb{R}$ is its true label. Since the embedding space in which $f(\mathbf{x})$ is located can be considered as the common space of natural images associated with the decision made by g , this space can be viewed as an approximation of natural image manifold \mathcal{M} . Therefore, this learning process can be regarded as an attempt to model the natural image manifold.

Adversarial attack crafts adversarial examples by optimizing the following objective:

$$\max_{\|\boldsymbol{\delta}^*\| \leq \epsilon} \ell(f \circ g(\mathbf{x} + \boldsymbol{\delta}^*), y), \quad (2)$$

where ϵ is the maximal norm which defines the set of allowed perturbations for a given example \mathbf{x} . In most contexts, such as The Projected Gradient Descent (PGD) [28], $\boldsymbol{\delta}^*$ is approximated by the local worst-case $\boldsymbol{\delta}$.

We are starting with formulating the forward process of f . Suppose $f : \mathbb{R}^n \rightarrow \mathbb{R}^d$ is twice differentiable, then the forward of f can be defined as:

$$f_{1\dots l}(\mathbf{x}) = f_1 \circ f_2 \circ \dots \circ f_l(\mathbf{x}) = f_l(f_{l-1}(f_{l-2} \dots f_1(\mathbf{x}))), \quad (3)$$

where f_i typically refers to the output of the i^{th} layer in the function f . Based on the Taylor series, we have the following theorem:

Theorem 1 Suppose $f : \mathbb{R}^n \rightarrow \mathbb{R}^d$ is twice differentiable at point \mathbf{x} . Let $e_1 = \nabla f_1(\mathbf{x})^T \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^T \nabla^2 f_1(\bar{\mathbf{x}}) \boldsymbol{\delta}$, and there exists $w_1 \in [0, 1]$ such that $\bar{\mathbf{x}} = w_1 \mathbf{x} + (1 - w_1)(\mathbf{x} + \boldsymbol{\delta})$, then we have the forward of f :

$$f_{1\dots l}(\mathbf{x} + \boldsymbol{\delta}) = f_{1\dots l}(\mathbf{x}) + e_l(\boldsymbol{\delta}), \quad (4)$$

$$\text{where } e_l(\boldsymbol{\delta}) = \nabla f_l(f_{1\dots l-1}(\mathbf{x}))^T e_{l-1}(\boldsymbol{\delta}) + \frac{1}{2} e_{l-1}(\boldsymbol{\delta})^T \nabla^2 f_{l-1}(\bar{\mathbf{x}}) e_{l-1}(\boldsymbol{\delta}). \quad (5)$$

where there exists $w_l \in [0, 1]$ such that $\bar{\mathbf{x}} = w_l f_{1\dots l-1}(\mathbf{x}) + (1 - w_l)(f_{1\dots l-1}(\mathbf{x}) + e_{l-1}(\boldsymbol{\delta}))$.

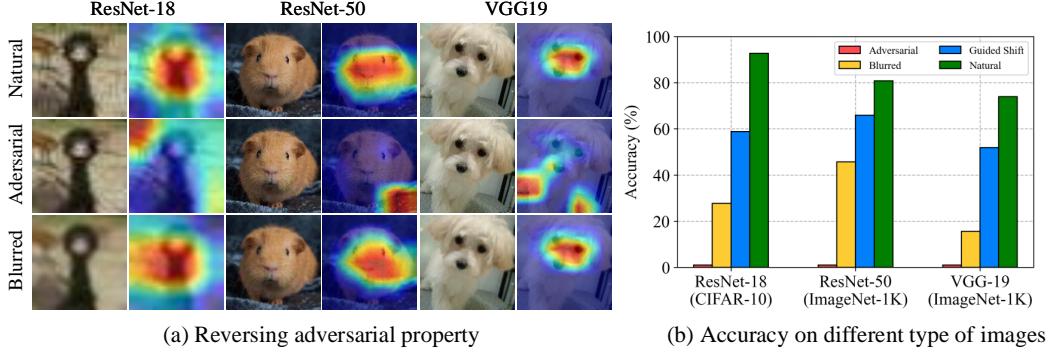


Figure 2: (a) An illustration of activation of crucial semantic features on CIFAR-10 and ImageNet-1K. (b) Accuracy of different properties of images on three models, where the blurring operator is the median filter. Left: AutoAttack results on CIFAR-10. Middle and Right: AutoAttack on ImageNet.

For notational simplicity, we denote $f_{1\dots l}(\mathbf{x}) = f(\mathbf{x})$ and $e_l(\delta) = e(\delta)$ when l is the last layer of f . Theorem 1 describes how the perturbation δ is amplified layer by layer during the forward of f so that the embedding of natural image \mathbf{x} deviates manifold \mathcal{M} and become the outlier, thereby demonstrates the reason for the incorrect decision made by g :

$$g(f(\mathbf{x} + \delta)) = g(f(\mathbf{x}) + e(\delta)) \neq g(f(\mathbf{x})), \quad (6)$$

where $e(\delta)$ can be viewed as the deviation distance from the natural image manifold.

By this formulation, adversarial purification can be naturally seen as the process of estimating $e(\delta)$ in the embedding space, the approximation of the natural image manifold. This means the adversarial embedding $f(\mathbf{x}_{\text{adv}}) = f(\mathbf{x} + \delta)$ can return the manifold along the same path $-e(\delta)$. While precise estimation of $e(\delta)$ is challenging, it is fortunate that only an approximation, denoted as $\tilde{e}(\delta)$ is sufficient to move the adversarial embedding back within the manifold.

3.2 Guided Shift

Considering adversarial purification as an inverse process to adversarial attacks, we can emulate Eq.(2) to formulate the optimization objective used to estimate $\tilde{e}(\delta)$:

$$\min_{\delta_{\text{pfy}}} \ell(f \circ g(\mathbf{x}_{\text{adv}} + \delta_{\text{pfy}}), y) \quad (7)$$

$$\text{s.t. } \|\delta_{\text{pfy}}\| \leq \epsilon_{\text{pfy}} \text{ and } e(\delta_{\text{pfy}}) = -\tilde{e}(\delta) \approx -e(\delta), \quad (8)$$

where δ_{pfy} and ϵ_{pfy} are defined to correspond to δ and ϵ in Eq.(2) to offset the perturbation. However, even if ϵ_{adv} can be considered as a hyperparameter, the ground-truth label y is not accessible. External model-based adversarial purification [17, 30, 35, 51] typically train a purification model to minimize the global ℓ_{pfy} . Auxiliary function-based adversarial method [19, 29, 38] tend to design a suitable ℓ_{pfy} to complete purification without relying on external generative models. They need to retrain the classifier or parameterized auxiliary function to effectively cooperate in removing adversarial perturbations, which means these methods all introduce parameters Θ in ℓ_{pfy} :

$$\min_{\delta_{\text{pfy}}} \ell_{\text{pfy}}(f(\mathbf{x}_{\text{adv}} + \delta_{\text{pfy}}); \Theta). \quad (9)$$

Based on our assumption in subsection 3.1 that adversarial images are outliers of the natural image manifold and the purification process can be considered as returning them to this manifold, we aim to design ℓ_{pfy} without Θ by allowing them to shift towards the manifold adaptively:

$$\min_{\delta_{\text{pfy}}} \ell_{\text{pfy}}^*(f(\mathbf{x}_{\text{adv}} + \delta_{\text{pfy}})). \quad (10)$$

We are starting to investigate whether a simple image transformation, such as *color jitter*, *grayscale*, *Gaussian blur*, *solarization*, and *equalization* can shift adversarial images towards the natural image manifold to improve classifier performance. We find that blurring adversarial examples can improve the robust accuracy of victim classifiers. Additionally, as shown in Fig 2(a), the blur operation can reactivate regions disrupted by adversarial images. This phenomenon suggests that a simple blurring operator can bring adversarial examples closer to the manifold and reverse adversarial properties.

Algorithm 1 Guided Shift

Input: Adversarial example \mathbf{x}_{adv} , iterations T_g , step size η_1 , a classifier f , purification bound ϵ_{pfy} , and the blurring operator $\text{blur}(\cdot)$.

Output: Guided result \mathbf{x}_g

- 1: Random start $\mathbf{x}_g^0 \leftarrow \mathbf{x}_{\text{adv}} + \varepsilon$
- 2: **for** $t = 0, 1, 2, \dots, T_g - 1$ **do**
- 3: $\tilde{\mathbf{x}}_g^t \leftarrow \text{blur}(\mathbf{x}_g^t)$
- 4: $\mathbf{z} \leftarrow f(\mathbf{x}_g^t), \tilde{\mathbf{z}} \leftarrow f(\tilde{\mathbf{x}}_g^t)$
- 5: $\mathbf{x}_g^{t+1} \leftarrow \mathbf{x}_g^t + \eta_1 \cdot \text{sgn}(\nabla_{\mathbf{x}_g^t} d(\mathbf{z}, \tilde{\mathbf{z}}))$
- 6: $\mathbf{x}_g^{t+1} \leftarrow \text{Clip}(\mathbf{x}_g^{t+1}, -\epsilon_{\text{pfy}}, \epsilon_{\text{pfy}})$
- 7: $\mathbf{x}_g^{t+1} \leftarrow \text{Clip}(\mathbf{x}_g^{t+1}, 0, 1)$
- 8: **end for**
- 9: $\mathbf{x}_g \leftarrow \mathbf{x}_g^{T_g}$

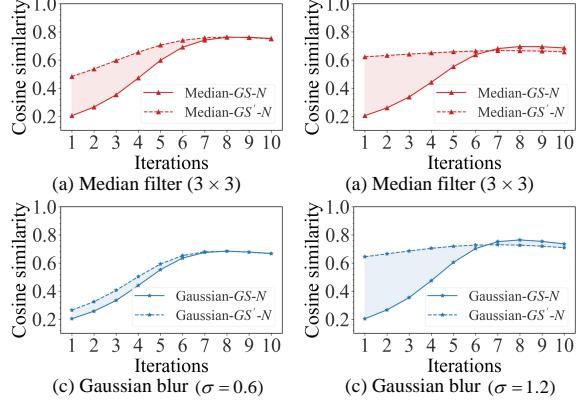


Figure 3: Details of GS on CIFAR-10.

Therefore, we can accumulate this reversion to guide the adversarial examples back towards the manifold. For instance, we can pull the distance between adversarial examples and their blurred counterparts in the embedding space, which allows adversarial examples to converge closer to the natural image manifold. The distance of feature embeddings is defined by the Cosine Similarity:

$$d(\mathbf{z}_{\text{adv}}, \mathbf{z}'_{\text{adv}}) = \frac{\mathbf{z}_{\text{adv}} \cdot \mathbf{z}'_{\text{adv}}}{\|\mathbf{z}_{\text{adv}}\| \|\mathbf{z}'_{\text{adv}}\|}, \quad (11)$$

where \mathbf{z}_{adv} and \mathbf{z}'_{adv} are the embeddings of adversarial examples and their blurred counterparts, respectively. we can shift the adversarial example \mathbf{x}_{adv} by the gradient of Eq.(11):

$$\mathbf{x}_{\text{adv}}^+ = \mathbf{x}_{\text{adv}} + \eta_1 \cdot \nabla_{\mathbf{x}_{\text{adv}}} d(f(\mathbf{x}_{\text{adv}}), f(\mathbf{x}'_{\text{adv}})). \quad (12)$$

Controlling the magnitude of the blurring applied carefully is essential to prevent excessive blurring that could render adversarial examples unrecognizable by the classifier. However, a small magnitude is not sufficient to shift images to reverse the distortion caused by adversarial perturbations. We therefore consider to move \mathbf{x}_{adv} by Eq.(12) with a small step size η_1 iteratively. This process is referred to as Guided Shift (GS). In each step of GS, we apply the same update rule:

$$\mathbf{x}_g^{t+1} = \mathbf{x}_g^t + \eta_1 \cdot \nabla_{\mathbf{x}_g^t} d(f(\mathbf{x}_g^t), f(\tilde{\mathbf{x}}_g^t)), \quad (13)$$

$$\tilde{\mathbf{x}}_g^{t+1} = \text{blur}(\mathbf{x}_g^{t+1}) \quad (14)$$

where $\text{blur}(\cdot)$ is the blurring operator and $\mathbf{x}_g^0 := \mathbf{x}_{\text{adv}}$, such as the median filter, Gaussian blur, and so on. The workflow of GS is shown in Algorithm 1, where we use $\text{sgn}(\cdot)$ to regulate the step size.

To demonstrate the process of GS, we employ two blurring operators: the median filter and Gaussian blur to compute the cosine similarity between single-step result \mathbf{x}_g^t of GS and the natural image \mathbf{x} (\star -GS-N), as well as between the blurred counterpart $\tilde{\mathbf{x}}_g^t$ of \mathbf{x}_g^t and \mathbf{x} (\star -GS'-N). As shown in Fig. 3, in each step, the cosine similarity (\star -GS'-N) is consistently smaller than the cosine similarity \star -GS-N. Furthermore, these cosine similarities consistently increase as GS progresses, and adversarial examples are gradually getting closer to the natural image manifold guided by their blurred counterparts.

3.3 Adaptive Projection

We also can observe an interesting phenomenon in Fig. 3. Regardless of the type of blurring operators, dash lines (\star -GS'-N) and solid lines (\star -GS-N) tend to converge. Therefore, we can have the following assumption:

Assumption 1 *There is a limitation for Guided Shift in that the adversarial images can be guided by their blurred counterparts only toward the manifold rather than fully back to the manifold. In other words, this indicates that Guided Shift approximates a convex function.*

Certainly, this limitation arises from the exceedingly blurred images whose embeddings cannot be recognized correctly by the classifier. To fully return adversarial images to the manifold, we must

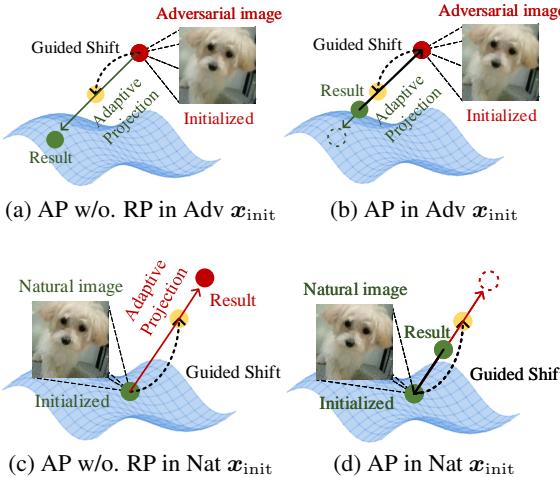


Figure 4: Intuitive explanation of AP in the context of adversarial or natural \mathbf{x}_{init} . PR: the perceptual regularization (Eq.(20)).

Algorithm 2 Adaptive Projection

Input: Input image \mathbf{x}_{init} , guided result \mathbf{x}_g , iterations T_p , a classifier f , purification bound ϵ_{pfy} , and candidate layers S .

Output: Adaptive result \mathbf{x}_p

```

1: Initialize  $\mathbf{x}_p^0 \leftarrow \mathbf{x}_{\text{init}}$ 
2: Set step size  $\eta_2 \leftarrow \epsilon_{\text{pfy}}/T_p$ 
3: for  $t = 0, 1, 2, \dots, T_p - 1$  do
4:   for  $l \in S$  do
5:      $\Delta \mathbf{u}_p^l \leftarrow f_l(\mathbf{x}_g) - f_l(\mathbf{x}_{\text{init}})$ 
6:      $\Delta \mathbf{u}_p^l \leftarrow f_l(\mathbf{x}_p^t) - f_l(\mathbf{x}_{\text{init}})$ 
7:      $\mathcal{L} \leftarrow \mathcal{L} + (-\Delta \mathbf{u}_g^l \cdot \Delta \mathbf{u}_p^l)$ 
8:   end for
9:    $\mathcal{L} \leftarrow \frac{\lambda_1}{\|S\|} \mathcal{L} + \lambda_2 \|\phi(\mathbf{x}_p^t) - \phi(\mathbf{x}_{\text{init}})\|_2$ 
10:   $\mathbf{x}_p^{t+1} \leftarrow \mathbf{x}_p^t - \eta_2 \cdot \text{sgn}(\nabla_{\mathbf{x}_p^t} \mathcal{L})$ 
11:   $\mathbf{x}_p^{t+1} \leftarrow \text{Clip}(\mathbf{x}_p^{t+1}, -\epsilon_{\text{pfy}}, \epsilon_{\text{pfy}})$ 
12:   $\mathbf{x}_p^{t+1} \leftarrow \text{Clip}(\mathbf{x}_p^{t+1}, 0, 1)$ 
13: end for
14:  $\mathbf{x}_p \leftarrow \mathbf{x}_p^{T_p}$ 

```

overcome this limitation. A natural intuition is to allow current images to move adaptively instead of being guided by blurred counterparts, which requires us to target a basic direction for current images. Fortunately, the result of Guided Shift already provides this basic direction. Therefore, we propose Adaptive Projection (AP) that can be defined as:

$$\max_{\mathbf{x}_p} \lambda_1 \frac{1}{\|S\|} \sum_{l \in S} \mathcal{L}_l(\mathbf{x}_{\text{init}}, \mathbf{x}_g, \mathbf{x}_p) - \lambda_2 \|\phi(\mathbf{x}_p) - \phi(\mathbf{x}_{\text{init}})\|_2 \quad \text{s.t.} \quad \|\mathbf{x}_p - \mathbf{x}_{\text{init}}\| \leq \delta_{\text{pfy}}, \quad (15)$$

where $S \subseteq L$ as a candidate in the set $L = \{l_1, l_2, \dots, l_m\}$ of a m layers model f , $\|S\|$ denotes the number of elements of the set S . \mathbf{x}_{init} represents the input image (i.e., $\mathbf{x}_{\text{init}} = \mathbf{x}_{\text{adv}}$ when the input image is adversarial). For the l^{th} layer of classifiers, the first term of Eq. (15) can be written as:

$$\mathcal{L}_l(\mathbf{x}_{\text{init}}, \mathbf{x}_g, \mathbf{x}_p) = -\Delta \mathbf{u}_p^l \cdot \Delta \mathbf{u}_g^l, \quad (16)$$

where \mathbf{x}_p is the final result, $\Delta \mathbf{u}_p^l$ and $\Delta \mathbf{u}_g^l$ are two vectors of flattened feature maps defined as follow:

$$\Delta \mathbf{u}_p^l = f_l(\mathbf{x}_p) - f_l(\mathbf{x}_{\text{init}}), \quad \Delta \mathbf{u}_g^l = f_l(\mathbf{x}_g) - f_l(\mathbf{x}_{\text{init}}), \quad (17)$$

where f_l denoted as feature maps at layer l of the classifier, \mathbf{x}_p is initialized by \mathbf{x}_{init} . Maximizing Eq.(16) is equivalent to maximizing the projection of \mathbf{u}_p^l onto \mathbf{u}_g^l since $\|\mathbf{u}_g^l\|$ is a constant. The increase in projection implies that \mathbf{x}_p is not restricted by the blurred images to continue moving along the direction of the result of GS, which allows \mathbf{x}_p move independently toward the natural image manifold.

However, We typically do not know whether an image is adversarial in real-world applications (i.e., $\mathbf{x}_{\text{init}} = \mathbf{x}$ when \mathbf{x} is natural), repeatedly guiding images by blurred counterparts may cause natural images to deviate from the manifold. As shown in Fig. 4, this deviation will be enhanced by AP. Therefore, we introduce a perceptual regularization term in Eq.(15). Let F_1 is the dynamic of Eq.(16):

$$F_1 = \nabla_{\mathbf{x}_p} \frac{1}{\|S\|} \sum_{l \in S} \mathcal{L}_l(\mathbf{x}_{\text{init}}, \mathbf{x}_g, \mathbf{x}_p), \quad (18)$$

If there exists a momentum F_2 , such that:

$$\begin{cases} F_1 > F_2, & \mathbf{x}_{\text{init}} \text{ is adversarial,} \\ F_1 < F_2 & \mathbf{x}_{\text{init}} \text{ is natural,} \end{cases} \quad (19)$$

then maximizing Eq. (16) allows the result \mathbf{x}_p to locate in the natural image manifold. In this work, the dynamic F_2 can be computed by LPIPS distance [55]. The LPIPS distance $d(\mathbf{x}_1, \mathbf{x}_2)$ between images \mathbf{x}_1 and \mathbf{x}_2 is then defined as $d(\mathbf{x}_1, \mathbf{x}_2) \triangleq \|\phi(\mathbf{x}_1) - \phi(\mathbf{x}_2)\|_2$. Let $\hat{f}(\mathbf{x})$ denote

channel-normalized activations at the l -th layer of the classifier. Then, $\phi(\mathbf{x}) \triangleq (\frac{\hat{f}(\mathbf{x})}{\sqrt{w_1 h_1}}, \dots, \frac{\hat{f}(\mathbf{x})}{\sqrt{w_m h_m}})$, where w_l and h_l are the width and height of activations of layer l , respectively.

Finally, the momentum F_2 can be written as:

$$F_2 = -\lambda_2 \cdot \nabla_{\mathbf{x}_p} \|\phi(\mathbf{x}_p) - \phi(\mathbf{x}_{\text{init}})\|_2. \quad (20)$$

In contrast to the original LPIPS implementation, we do not require this distance to approximate human perceptual judgments. Instead, we aim for the perceptual distance between \mathbf{x}_g and \mathbf{x}_{init} based on the classifier itself to be small. Hence, we can directly use $d(\mathbf{x}_g, \mathbf{x}_{\text{init}})$ to evaluate the distance from the perspective of the classifier. The workflow of AP is shown in Algorithm 2.

4 Experiments

Datasets and Metrics. Three benchmarks CIFAR-10 [22], CIFAR-100, and ImageNet-1K [9] are considered to evaluate our method. We compare our method with the state-of-the-art adversarial training methods reported in standard benchmark RobustBench [7] and other adversarial purification methods [16, 17, 19, 26, 29, 30, 38, 41, 43, 45, 51]. In all experiments, we consider two metrics to evaluate the performance of all methods: standard accuracy and robust accuracy. The standard accuracy measures the performance of the defense method on natural images, while the robust accuracy measures the performance on adversarial images.

Adversarial attacks and victim classifiers. We evaluate our methods with three attacks: AutoAttack [6], DI²-FGSM [49], and BPDA [3]. AutoAttack is a powerful adaptive attack commonly used in most defense studies. We also use DI²-FGSM to seek whether an attack robust to the blurring operator can affect our method. Additionally, we consider BPDA to attack our purification module in the worst-case scenario. These attacks will attack three victim classifiers including ResNet-18 [14], ResNet-50, and WideResNet-28-10 [53].

Table 1: Standard and robust accuracy (%) against AutoAttack $\ell_\infty(\epsilon = 8/255)$ in comparison with AT & ABP methods. The first section corresponds to AT methods and the second to ABP methods. † Since Hwang et al. [19] only reports the performance of WideResNet-34-10 in the original paper, and the pre-trained parameterized auxiliary checkpoint has been removed, we use this result for comparison.

| Require Training | | ResNet-18 | | | WideResNet-28-10 | | |
|------------------|-----------|--------------------------|--------------|--------------|------------------------|--------------|--------------|
| Classifier | Auxiliary | Method | Standard | Robust | Method | Standard | Robust |
| CIFAR-10 | | | | | | | |
| ✓ | ✗ | (Gowal et al., 2021) | 87.35 | 59.12 | (Gowal et al., 2021) | 87.50 | 63.99 |
| ✓ | ✗ | (Sehwag et al., 2021) | 84.59 | 56.19 | (Pang et al., 2022) | 88.61 | 61.40 |
| ✓ | ✗ | (Rade et al., 2021) | 89.02 | 58.17 | (Xu et al., 2023) | 93.69 | 65.62 |
| ✓ | ✗ | (Addepalli et al., 2022) | 85.71 | 52.90 | (Wang et al., 2023) | 92.44 | 67.31 |
| ✓ | ✗ | (Shi et al., 2021) | 84.07 | 66.62 | (Shi et al., 2021) | 91.89 | 68.56 |
| ✗ | ✓ | (Mao et al., 2021) | - | 58.20 | (Mao et al., 2021) | - | 67.15 |
| ✗ | ✓ | (Hwang et al., 2023)† | 87.02 | 56.63 | (Hwang et al., 2023) | 87.02 | 56.63 |
| ✗ | ✗ | GS | 49.56 | 58.82 | GS | 50.24 | 59.68 |
| ✗ | ✗ | GS+AP (ZeroPur) | 92.56 | 69.62 | GS+AP (ZeroPur) | 91.81 | 68.60 |
| CIFAR-100 | | | | | | | |
| ✓ | ✗ | (Rade et al., 2021) | 61.50 | 29.50 | (Pang et al., 2022) | 63.66 | 31.67 |
| ✓ | ✗ | (Addepalli et al., 2022) | 65.45 | 28.58 | (Rebuffi et al., 2021) | 62.41 | 33.03 |
| ✓ | ✗ | (Shi et al., 2021) | 52.91 | 32.38 | (Shi et al., 2021) | 61.01 | 34.37 |
| ✗ | ✓ | (Mao et al., 2021) | - | 25.45 | (Mao et al., 2021) | - | 33.16 |
| ✗ | ✓ | (Hwang et al., 2023) | 64.73 | 32.86 | (Hwang et al., 2023) | 64.73 | 32.86 |
| ✗ | ✗ | GS | 30.42 | 34.35 | GS | 26.56 | 32.06 |
| ✗ | ✗ | GS+AP (ZeroPur) | 52.45 | 41.42 | GS+AP (ZeroPur) | 64.88 | 34.58 |

4.1 Quantitative Evaluation and Comparison

For adversarial defense, adversarial training (AT) involves adversarial examples in the classifier training, and auxiliary-based purification (ABP) introduces an auxiliary function to cooperate with the victim classifier. These methods do not rely on external generative models but still necessitate training

auxiliary functions or retraining classifiers. External model-based purification (EBP) fine-tunes or retrains an external generative model to remove adversarial perturbations in adversarial images. Since the performance of EBP is ensured by the modeling capability of generative models, it typically outperforms ABP. In this section, for a fair comparison, we will first compare our method with both AT and ABP and then compare it with EBP. ‘Require Training’ in Table 1, 2 and 3 denotes that the methods require retraining external generative models or auxiliary functions or victim classifiers.

Comparison with AT & ABP. Table 1 reports the standard and robust accuracy against AutoAttack $\ell_\infty (\epsilon = 8/255)$. ZeroPur achieves an approximate 10% increase in robust accuracy for ResNet-18, and this improvement surpasses the performance of most other methods using WideResNet-28-10. We also report the comparison on ImageNet1-K in Table 3, where ZeroPur improves robust accuracy by 30% compared to AT and ABP methods. These results also demonstrate that AP can significantly move adversarial images toward the natural image manifold. It is noteworthy to mention that our method consistently presents better performance in small architectures such as ResNet-18 compared to large architectures. This stems from the inclination of our method to ‘fine-tune’ adversarial images, enabling them to approach natural images, whereas larger models often introduce more hindrances to this process.

Comparison with EBP. Training an external generative model to model a transformation from adversarial images to natural images can contribute to an outstanding purification method, but this process is exceedingly computation-consuming. Table 2 and Table 3 report the comparison result between ZeroPur and EBP. The robust accuracy achieved by ZeroPur[†] is only slightly lower than DISCO (Ho et al., 2022) which involves training an external model on ImageNet. Therefore, adjusting ZeroPur a little bit (see the first part of Subsection 4.2) can achieve comparable performance with these EBP methods. Other detailed results are reported in Appendix A.4.

Table 2: Standard and robust accuracy (%) against AutoAttack $\ell_\infty (\epsilon = 8/255)$ on CIFAR-10 in comparison with EBP methods.

| Require Training | | WideResNet-28-10 | | |
|------------------|-----|-----------------------|--------------|--------------|
| Clz | Ext | Method | Standard | Robust |
| ✗ | ✓ | (Sun et al., 2019) | 82.22 | 67.92 |
| ✗ | ✓ | (Hill et al., 2020) | 84.12 | 78.91 |
| ✗ | ✓ | (Yoon et al., 2021) | 86.14 | 80.24 |
| ✗ | ✓ | (Ughini et al., 2022) | - | 59.57 |
| ✗ | ✓ | (Nie et al., 2022) | 89.02 | 70.64 |
| ✗ | ✓ | (Ho et al., 2022) | 89.26 | 85.56 |
| ✗ | ✓ | (Lin et al., 2024) | 90.62 | 72.85 |
| ✗ | ✗ | ZeroPur | 91.81 | 68.60 |
| ✗ | ✗ | ZeroPur [†] | 85.02 | <u>82.76</u> |

Table 3: Standard and robust accuracy (%) against AutoAttack $\ell_\infty (\epsilon = 4/255)$ on ImageNet-1K.

| Training | | | | ResNet-50 | |
|--------------------------|-----|-----|-----------------------|-----------|--------------------|
| Clz | Aux | Ext | Method | Standard | Robust |
| Comparison with AT & ABP | | | | | |
| ✓ | ✗ | ✗ | (Salman et al., 2020) | 64.02 | 34.96 |
| ✓ | ✗ | ✗ | (Wong et al., 2020) | 55.62 | 26.24 |
| ✓ | ✗ | ✗ | (Bai et al., 2021) | 67.38 | 35.51 |
| ✗ | ✓ | ✗ | (Mao et al., 2021) | - | 31.32 |
| ✗ | ✗ | ✗ | ZeroPur | 61.05 | 63.09 |
| Comparison with EBP | | | | | |
| ✗ | ✗ | ✓ | (Wang et al., 2022) | 70.17 | 68.78 ² |
| ✗ | ✗ | ✓ | (Nie et al., 2022) | 67.79 | 40.93 |
| ✗ | ✗ | ✓ | (Ho et al., 2022) | 71.22 | 69.52 ¹ |
| ✗ | ✗ | ✗ | ZeroPur | 61.05 | 63.09 ³ |

4.2 Discussion on ZeroPur

Evaluating ZeroPur on protecting the victim classifier trained with different data augmentation. To further discuss the performance of ZeroPur, we consider three data augmentation strategies (‘Vanilla’, ‘Base’, ‘Strong’) to train victim classifiers (See Appendix A.2 for details). ‘Base’ is the most common data augmentation in training classifiers. In the experiment, ZeroPur using a median filter (3×3 window size) is applied in ‘Vanilla’ classifiers and Gaussian blur ($\sigma = 1.2$) in ‘Base’ and ‘Strong’ classifiers. Table 4 reports the standard and robust accuracy of ZeroPur. We can see that ‘Strong’ data augmentation benefits our method. The reason is that such augmentations enable the blurred counterparts to guide adversarial images toward the natural image manifold more effectively, which contributes to precise adaptive projection. The performance of ZeroPur[†] in Table 2 also demonstrates this phenomenon.

Impact of perceptual regularization. The important of incorporating the perceptual regularization in (Eq.(15)) is revealed by Fig. 5. We can see the robust accuracy of AP experiences a slight decrease in (a) and (b), while the standard accuracy shows significant improvement in (c) and (d). This

Table 4: Standard and robust accuracy (%) with different classifiers trained with three levels of data augmentation. We mark the best performance among the three levels by **bold value**.

| Accuracy | ResNet-18 | | | WRN-28-10 | | |
|-----------|-----------|--------------|--------------|-----------|--------------|--------------|
| | Vanilla | Base | Strong | Vanilla | Base | Strong |
| CIFAR-10 | | | | | | |
| Standard | 82.04 | 92.56 | 90.05 | 91.22 | 91.81 | 85.02 |
| Robust | 59.38 | 69.62 | 83.84 | 59.16 | 68.60 | 82.76 |
| CIFAR-100 | | | | | | |
| Standard | 55.45 | 52.45 | 67.11 | 64.89 | 64.88 | 66.39 |
| Robust | 30.42 | 41.42 | 50.66 | 30.81 | 34.58 | 50.25 |

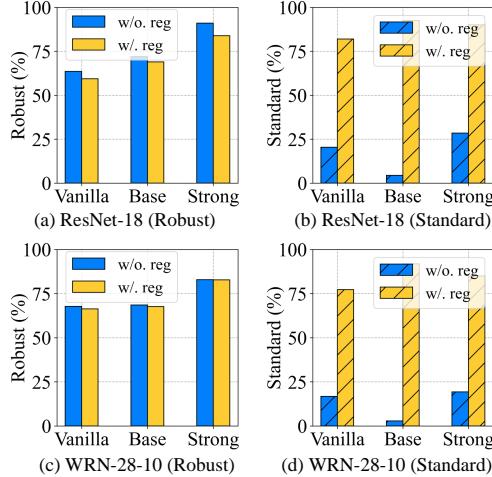


Table 5: The ablation study on CIFAR-10 using perceptual regularization.

phenomenon is observed in both ResNet-18 and WideResNet-28-10, indicating the effectiveness of perceptual regularization in AP.

ZeroPur defense against adversarial attacks robust to blurring. ZeroPur utilizes the blurring operator to return adversarial images to the natural image manifold. However, certain attacks, such as DI²-FGSM [49], inherently withstand blurring operations. Indeed, they cannot be entirely immune to ZeroPur. As shown in Table 6, replacing the blurring operator with TVM [13] leads to improved purification. The robust accuracy against AutoAttack increases to 75.96% without training, which outperforms the optimal performance of GS(Blur)+AP by 6.34%. This result also demonstrates that our method can enhance any operator capable of destroying adversarial properties, not just blurring.

ZeroPur defense against attacks that bypass the purification module. The efficiency of purification methods can be challenged by the BPDA attack that approximates gradients, bypassing the purification module. Table 7 reports the performance of ZeroPur defense against BPDA attack, where the victim classifiers are WideResNet-28-10 (WRN) and ResNet-18 (R18). We can see that the robust accuracy of ZeroPur decreases, but it remains stable, which suggests that ZeroPur still offers acceptable defense. Since the natural accuracy of 'Vanilla' WRN on natural images is 71.50%, while that of 'Vanilla' R18 is 57.01%, which may result in an approximately 20% gap in evaluation on BPDA (See Table 8 in Appendix A.2).

Table 6: Robust accuracy (%) against DI²-FGSM and AutoAttack on CIFAR-10 by blurring and TVM.

| Method | DI ² -FGSM | | | AutoAttack | | |
|----------------|-----------------------|--------------|--------------|--------------|--------------|--------------|
| | Vanilla | Base | Strong | Vanilla | Base | Strong |
| GS (Blur) | 35.77 | 40.96 | 66.66 | 54.37 | 58.82 | 79.36 |
| GS (TVM) | 38.08 | 46.79 | 45.93 | 56.25 | 67.81 | 68.48 |
| GS (Blur) + AP | 35.34 | 59.81 | 84.41 | 59.38 | 69.62 | 83.84 |
| GS (TVM) + AP | 50.51 | 65.98 | 66.74 | 61.75 | 75.96 | 78.98 |

Table 7: Robust accuracy (%) against BPDA ℓ_∞ ($\epsilon = 8/255$) on CIFAR-10.

| Model | PGD-10 PGD-20 PGD-40 | | | |
|---------|----------------------|--------------|--------------|--------------|
| | Vanilla | WRN | 50.61 | 50.71 |
| Vanilla | R18 | 34.44 | 34.88 | 32.18 |
| Base | WRN | 34.16 | 33.38 | 34.42 |
| Base | R18 | 39.88 | 39.90 | 39.48 |
| Strong | WRN | 68.84 | 69.45 | 68.50 |
| Strong | R18 | 70.65 | 70.45 | 70.43 |

5 Conclusion

We propose a succinct training-free method for adversarial purification, named ZeroPur. Our method significantly outperforms previous state-of-the-art adversarial training and auxiliary-based purification methods, while demonstrating comparability with external model-based purification methods. Despite the improvements, ZeroPur has a major limitation: the absence of external models restricts our ability to enhance defense against strong adaptive attacks. To overcome this challenge, it is necessary to design a module that emulates recent works on purification by diffusion models capable of effectively defending BPDA.

References

- [1] Sravanti Addepalli, Samyak Jain, et al. Efficient and effective augmentation strategy for adversarial training. *Advances in Neural Information Processing Systems*, 35:1488–1501, 2022.
- [2] Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. *Advances in Neural Information Processing Systems*, 33:16048–16059, 2020.
- [3] 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*, pages 274–283. PMLR, 2018.
- [4] Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? *Advances in Neural Information Processing Systems*, 34:26831–26843, 2021.
- [5] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5939–5948, 2019.
- [6] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR, 2020.
- [7] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL <https://openreview.net/forum?id=SSKZPJct7B>.
- [8] Sihui Dai, Saeed Mahloujifar, and Prateek Mittal. Formulating robustness against unforeseen attacks. *Advances in Neural Information Processing Systems*, 35:8647–8661, 2022.
- [9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.
- [11] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- [12] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Advances in Neural Information Processing Systems*, 34:4218–4233, 2021.
- [13] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.
- [16] Mitch Hill, Jonathan Mitchell, and Song-Chun Zhu. Stochastic security: Adversarial defense using long-run dynamics of energy-based models. *arXiv preprint arXiv:2005.13525*, 2020.
- [17] Chih-Hui Ho, Nuno Vasconcelos, et al. Disco: Adversarial defense with local implicit functions. *Advances in Neural Information Processing Systems*, 35:23818–23837, 2022.
- [18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- [19] Duhun Hwang, Eunjung Lee, and Wonjong Rhee. Aid-purifier: A light auxiliary network for boosting adversarial defense. *Neurocomputing*, 541:126251, 2023.

[20] Daniel Kang, Yi Sun, Dan Hendrycks, Tom Brown, and Jacob Steinhardt. Testing robustness against unforeseen adversaries. 2019.

[21] Hoki Kim. Torchattacks: A pytorch repository for adversarial attacks. *arXiv preprint arXiv:2010.01950*, 2020.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[23] Soichiro Kumano, Hiroshi Kera, and Toshihiko Yamasaki. Adversarial training from mean field perspective. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Cassidy Laidlaw, Sahil Singla, and Soheil Feizi. Perceptual adversarial robustness: Defense against unseen threat models. *arXiv preprint arXiv:2006.12655*, 2020.

[25] Tao Li, Yingwen Wu, Sizhe Chen, Kun Fang, and Xiaolin Huang. Subspace adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13409–13418, 2022.

[26] Guang Lin, Chao Li, Jianhai Zhang, Toshihisa Tanaka, and Qibin Zhao. Adversarial training on purification (atop): Advancing both robustness and generalization. *arXiv preprint arXiv:2401.16352*, 2024.

[27] Jiang Liu, Chun Pong Lau, Hossein Souri, Soheil Feizi, and Rama Chellappa. Mutual adversarial training: Learning together is better than going alone. *IEEE Transactions on Information Forensics and Security*, 17:2364–2377, 2022.

[28] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[29] Chengzhi Mao, Mia Chiquier, Hao Wang, Junfeng Yang, and Carl Vondrick. Adversarial attacks are reversible with natural supervision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 661–671, 2021.

[30] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022.

[31] Tianyu Pang, Min Lin, Xiao Yang, Jun Zhu, and Shuicheng Yan. Robustness and accuracy could be reconcilable by (proper) definition. In *International Conference on Machine Learning*, pages 17258–17277. PMLR, 2022.

[32] Rahul Rade, Moosavi-Dezfooli, and Seyed-Mohsen. 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.

[33] 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.

[34] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? *Advances in Neural Information Processing Systems*, 33:3533–3545, 2020.

[35] Pouya Samangouei, Maya Kabkab, and R Defense-GAN Chellappa. Protecting classifiers against adversarial attacks using generative models. arxiv 2018. *arXiv preprint arXiv:1805.06605*, 1, 2018.

[36] 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? *arXiv preprint arXiv:2104.09425*, 2021.

[37] 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*, 32, 2019.

[38] Changhao Shi, Chester Holtz, and Gal Mishne. Online adversarial purification based on self-supervision. *arXiv preprint arXiv:2101.09387*, 2021.

[39] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

[40] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[41] Bo Sun, Nian-hsuan Tsai, Fangchen Liu, Ronald Yu, and Hao Su. Adversarial defense by stratified convolutional sparse coding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11447–11456, 2019.

[42] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[43] Giorgio Ughini, Stefano Samele, and Matteo Matteucci. Trust-no-pixel: A remarkably simple defense against adversarial attacks based on massive inpainting. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2022.

[44] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International conference on machine learning*, pages 1747–1756. PMLR, 2016.

[45] Jinyi Wang, Zhaoyang Lyu, Dahua Lin, Bo Dai, and Hongfei Fu. Guided diffusion model for adversarial purification. *arXiv preprint arXiv:2205.14969*, 2022.

[46] Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training. In *International Conference on Machine Learning*, pages 36246–36263. PMLR, 2023.

[47] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

[48] Boxi Wu, Jindong Gu, Zhifeng Li, Deng Cai, Xiaofei He, and Wei Liu. Towards efficient adversarial training on vision transformers. In *European Conference on Computer Vision*, pages 307–325. Springer, 2022.

[49] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2730–2739, 2019.

[50] Yuancheng Xu, Yanchao Sun, Micah Goldblum, Tom Goldstein, and Furong Huang. Exploring and exploiting decision boundary dynamics for adversarial robustness. *arXiv preprint arXiv:2302.03015*, 2023.

[51] Jongmin Yoon, Sung Ju Hwang, and Juho Lee. Adversarial purification with score-based generative models. In *International Conference on Machine Learning*, pages 12062–12072. PMLR, 2021.

[52] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[53] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[54] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR, 2019.

[55] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

A Appendix

A.1 Proofs

Theorem 1 Suppose $f : \mathbb{R}^n \rightarrow \mathbb{R}^d$ is twice differentiable at point \mathbf{x} . Let $e_1 = \nabla f_1(\mathbf{x})^T \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^T \nabla^2 f_1(\bar{\mathbf{x}}) \boldsymbol{\delta}$, and there exists $w_1 \in [0, 1]$ such that $\bar{\mathbf{x}} = w_1 \mathbf{x} + (1 - w_1)(\mathbf{x} + \boldsymbol{\delta})$, then we have

the forward of f :

$$f_{1\dots l}(\mathbf{x} + \boldsymbol{\delta}) = f_{1\dots l}(\mathbf{x}) + e_l(\boldsymbol{\delta}), \quad (1)$$

$$\text{where } e_l(\boldsymbol{\delta}) = \nabla f_l(f_{1\dots l-1}(\mathbf{x}))^T e_{l-1}(\boldsymbol{\delta}) + \frac{1}{2} e_{l-1}(\boldsymbol{\delta})^T \nabla^2 \bar{f}_{1\dots l-1}(\mathbf{x}) e_{l-1}(\boldsymbol{\delta}). \quad (2)$$

where there exists $w_l \in [0, 1]$ such that $\bar{f}_{1\dots l-1}(\mathbf{x}) = w_l f_{1\dots l-1}(\mathbf{x}) + (1-w_l)(f_{1\dots l-1}(\mathbf{x}) + e_{l-1}(\boldsymbol{\delta}))$.

Proof By a first-order Taylor series, we have:

$$f(\mathbf{x}) = f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T (\mathbf{x} - \mathbf{x}_0) + \frac{1}{2} (\mathbf{x} - \mathbf{x}_0)^T \nabla^2 f(\bar{\mathbf{x}}) (\mathbf{x} - \mathbf{x}_0). \quad (3)$$

There exists $w \in [0, 1]$ such that $\bar{\mathbf{x}} = w\mathbf{x} + (1-w)\mathbf{x}_0$. for the forward of an adversarial example $\mathbf{x} + \boldsymbol{\delta}$ in the first layer f_1 , we have:

$$f_1(\mathbf{x} + \boldsymbol{\delta}) = f_1(\mathbf{x}) + \underbrace{\nabla f_1(\mathbf{x})^T \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^T \nabla^2 f_1(\bar{\mathbf{x}}) \boldsymbol{\delta}}_{e_1(\boldsymbol{\delta})}. \quad (4)$$

Plugging f_2 into Eq.(4), we also have:

$$f_{1,2}(\mathbf{x} + \boldsymbol{\delta}) = f_2(f_1(\mathbf{x} + \boldsymbol{\delta})) = f_2(f_1(\mathbf{x}) + e_1(\boldsymbol{\delta})) \quad (5)$$

$$= \underbrace{f_2(f_1(\mathbf{x}))}_{f_{1,2}(\mathbf{x})} + \underbrace{\nabla f_2(f_1(\mathbf{x}))^T e_1(\boldsymbol{\delta}) + \frac{1}{2} e_1(\boldsymbol{\delta})^T \nabla^2 \bar{f}_1(\mathbf{x}) e_1(\boldsymbol{\delta})}_{e_2(\boldsymbol{\delta})}. \quad (6)$$

when $\mathbf{x} + \boldsymbol{\delta}$ is forward through the l -th layer, we have Theorem 1. \square

A.2 Implementation Details of Classifiers

Our experiments on CIFAR-10 and CIFAR-100 included ResNet-18 and WideResNet-28-10 trained on three different data augmentation strategies ‘Vanilla’, ‘Base’, and ‘Strong’. Table 8 shows the training settings and natural accuracy. ‘Base’ is the most common data augmentation used by adversarial training methods and adversarial purification methods for better natural accuracy.

| Victim Classifier | Case | Trained On | | | | | | Natural Accuracy (%) | | |
|-------------------|---------|------------|---------|--------|----------|--------|--------|----------------------|--------------|--------------|
| | | ReCrop. | ColorJ. | GrayS. | GauBlur. | Solar. | Equal. | HorFlip. | CIFAR-10 | CIFAR-100 |
| ResNet-18 | Vanilla | | | | | | | | 83.80 | 57.01 |
| | Base | ✓ | | | | | | ✓ | 93.10 | 71.58 |
| | Strong | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 91.08 | 68.52 |
| WRN-28-10 | Vanilla | | | | | | | | 91.34 | 71.50 |
| | Base | ✓ | | | | | | ✓ | 93.83 | 74.95 |
| | Strong | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 91.09 | 67.52 |

Table 8: The adopted data augmentation and natural accuracy (%) of victim classifiers. The enumerated data augmentation are, in order, ResizeCrop, ColorJitter, Grayscale, Solarization, Equalization, and HorizontalFlip.

All classifiers were trained with the SGD optimizer with a cosine decay learning rate schedule and a linear warm-up period of 10 epochs. The weight decay is 5.0×10^{-4} and the momentum is 0.9. The initial learning rate is set to 0.1. Classifiers were trained for 120 epochs on 4 Tesla V100 GPUs, where the batch size is 512 per GPU for ResNet-18 and 128 per GPU for WideResNet-28-10.

A.3 Implementation Details of Adversarial Attacks

AutoAttack [6]. We use AutoAttack to compare with the start-of-the-art methods. The robust classifier for adversarial training methods provided by RobustBench [7] benchmark available at

<https://robustbench.github.io>. The code for adversarial purification methods is provided by their respective papers.

There are two versions of AutoAttack: (i) the STANDARD including AGPD-CE, AGPD-T, FAB-T, and Square, and (ii) the RAND version including APGD-CE and APGD-DLR. Considering that most of the adversarial purifications choose the RAND version, all the performance in this work we report is also in the RAND version. Code is available at <https://github.com/fra31/auto-attack>.

DI²-FGSM [49]. DI²-FGSM crafts adversarial examples by applying various transformations, enhancing their robustness against blurring operations. We use this attack to evaluate ZeroPur in Section 4.2, implemented by torchattacks [21]. Code is available at <https://github.com/Harry24k/adversarial-attacks-pytorch>.

BPDA [3]. We use BPDA, approximating the gradient of the purifier module (ZeroPur) as 1 during the backward pass. The 10, 20, and 40 iterations are applied in our experiment. Other settings are the same as those used in PGD Attack.

A.4 More Experimental Results

In Table 9 and Table 10, we compare our method with AT, ABP, and EBP methods on CIFAR-10 against $\ell_\infty(\epsilon = 8/255)$ and $\ell_2(\epsilon = 0.5)$ threat model, detailing training requirements for each method. We also report the performance of ZeroPur when victim classifiers use strong data augmentation (ZeroPur[†]). Comparing all methods, ZeroPur always achieves the second-best performance. However, when comparing ABP methods, ZeroPur achieves optimal performance.

Table 9: Standard and robust accuracy (%) against AutoAttack $\ell_\infty(\epsilon = 8/255)$ on CIFAR-10 in comparison with three types methods, obtained by different classifier architectures.

| Defense Type | Method | Training | Architecture | Standard Acc | Robust Acc |
|--------------|--------------------------|------------------|--------------|--------------|--------------|
| AT | (Gowal et al., 2021) | Classifier | ResNet-18 | 87.35 | 59.12 |
| | (Gowal et al., 2021) | Classifier | WRN-28-10 | 87.50 | 63.99 |
| | (Sehwag et al., 2021) | Classifier | ResNet-18 | 84.59 | 56.19 |
| | (Rade et al., 2021) | Classifier | ResNet-18 | 89.02 | 58.17 |
| | (Addepalli et al., 2022) | Classifier | ResNet-18 | 85.71 | 82.90 |
| | (Pang et al., 2022) | Classifier | WRN-28-10 | 88.61 | 61.40 |
| | (Xu et al., 2023) | Classifier | WRN-28-10 | 93.69 | 65.62 |
| | (Wang et al., 2023) | Classifier | WRN-28-10 | 92.44 | 67.31 |
| EBP | (Sun et al., 2019) | STL | WRN-28-10 | 82.22 | 67.92 |
| | (Hill et al., 2020) | EBM+LD | WRN-28-10 | 84.12 | 78.91 |
| | (Yoon et al., 2021) | DSM+LD | WRN-28-10 | 86.14 | 80.24 |
| | (Ughini et al., 2022) | DeepFill [52] | WRN-28-10 | - | 59.57 |
| | (Nie et al., 2022) | DDPM [18] | WRN-28-10 | 89.02 | 70.64 |
| | (Nie et al., 2022) | DDPM [18] | WRN-70-16 | 90.07 | 71.29 |
| | (Ho et al., 2022) | LIIF [5] | WRN-28-10 | 89.26 | 85.56 |
| | (Lin et al., 2024) | MAE [15] | WRN-28-10 | 90.62 | 72.85 |
| ABP | (Lin et al., 2024) | MAE [15] | WRN-70-16 | 91.99 | 76.37 |
| | (Shi et al., 2021) | Classifier | ResNet-18 | 84.07 | 66.62 |
| | (Shi et al., 2021) | Classifier | WRN-28-10 | 91.89 | 68.56 |
| | (Mao et al., 2021) | Auxiliary branch | ResNet-18 | - | 58.20 |
| | (Mao et al., 2021) | Auxiliary branch | WRN-28-10 | - | 67.15 |
| | (Hwang et al., 2023) | Auxiliary branch | WRN-34-10 | 87.02 | 56.63 |
| | ZeroPur | N/A | ResNet-18 | <u>92.56</u> | 69.62 |
| | ZeroPur [†] | N/A | ResNet-18 | 90.05 | <u>83.84</u> |
| | ZeroPur | N/A | WRN-28-10 | 91.81 | 68.60 |
| | ZeroPur [†] | N/A | WRN-28-10 | 85.02 | 82.76 |

Table 10: Standard and robust accuracy (%) against AutoAttack $\ell_2(\epsilon = 0.5)$ on CIFAR-10 in comparison with three types methods, obtained by different classifier architectures.

| Defense Type | Method | Training | Architecture | Standard Acc | Robust Acc |
|--------------|------------------------|---------------|--------------|--------------|--------------|
| AT | (Rebuffi et al., 2021) | Classifier | ResNet-18 | 90.33 | 75.86 |
| | (Sehwag et al., 2021) | Classifier | ResNet-18 | 89.76 | 74.41 |
| | (Wang et al., 2023) | Classifier | WRN-28-10 | 95.16 | 83.68 |
| | (Rebuffi et al., 2021) | Classifier | WRN-28-10 | 91.79 | 78.80 |
| | (Sehwag et al., 2021) | Classifier | WRN-34-10 | 90.93 | 77.24 |
| EBP | (Sun et al., 2019) | STL | WRN-28-10 | 82.22 | 74.33 |
| | (Ughini et al., 2022) | DeepFill [52] | WRN-28-10 | - | 45.12 |
| | (Nie et al., 2022) | DDPM [18] | WRN-28-10 | 91.03 | 78.58 |
| | (Nie et al., 2022) | DDPM [18] | WRN-70-16 | 92.68 | 80.60 |
| | (Ho et al., 2022) | LIIF [5] | WRN-28-10 | 89.26 | 88.47 |
| ABP | (Lin et al., 2024) | MAE [15] | WRN-28-10 | 90.62 | 80.47 |
| | (Lin et al., 2024) | MAE [15] | WRN-70-16 | 91.99 | 81.35 |
| | ZeroPur | N/A | ResNet-18 | 80.14 | 78.10 |
| | ZeroPur [†] | N/A | ResNet-18 | 90.77 | 86.56 |
| | ZeroPur | N/A | WRN-28-10 | 72.38 | 72.12 |
| | ZeroPur [†] | N/A | WRN-28-10 | 81.52 | <u>86.61</u> |

Table 11: Robust accuracy (%) against AutoAttack $\ell_\infty(\epsilon = 8/255)$ on CIFAR-10, obtained by different blurring operators in Guided Shift.

| Operator | Level | ResNet-18 | | | WRN-28-10 | | |
|----------|----------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | | Vanilla | Base | Strong | Vanilla | Base | Strong |
| Median | 3×3 | 59.38 | 72.27 | 60.74 | 59.16 | 67.56 | 70.01 |
| | 5×5 | 34.17 | 64.81 | 44.63 | 39.81 | 58.41 | 50.49 |
| | 7×7 | 27.25 | 54.41 | 34.68 | 28.94 | 48.26 | 40.58 |
| Gaussian | $\sigma = 0.6$ | 38.82 | 65.43 | 64.92 | 53.21 | 58.31 | 73.79 |
| | $\sigma = 1.2$ | 26.20 | 69.62 | 83.84 | 27.97 | 68.60 | 82.76 |
| | $\sigma = 1.8$ | 20.45 | 60.67 | 67.89 | 26.82 | 59.91 | 78.82 |

Table 11 and Table 12 report the robust accuracy and standard accuracy of ZeroPur using different blur operators in GS. We use 5 different blurring operators including median filters with windows size 3×3 , 5×5 , and 7×7 and Gaussian blurring kernel with $\sigma = 0.6, 1.2$, and 1.8 . The results show that if victim classifiers are not trained on strong data augmentation, low-quality embeddings of images from excessive blurring will hinder ZeroPur’s ability to move adversarial images. On the contrary, the victim classifiers trained on strong data augmentation can recognize these embeddings, enabling ZeroPur to provide precise movement direction.

We provide visual examples in Fig. 5 to illustrate several results and the visualization of this process, where $\text{diff}_I_a_I_b = \text{abs}(I_a - I_b)$ represents the difference between the two images. The resemblance between $\text{diff}_{\text{gs_nat}}$ and $\text{diff}_{\text{adv_nat}}$ outlines suggests that Guided Shift can effectively reverse the adversarial perturbation at a coarse level. Furthermore, the close correspondence in detail between $\text{diff}_{\text{adv_nat}}$ and $\text{diff}_{\text{ap_nat}}$ demonstrates the precise approximation of this perturbation by Adaptive Projection. $\text{diff}_{\text{ap_adv}}$ quantifies the distance of the adversarial image moved by ZeroPur.

Table 12: Standard accuracy (%) on CIFAR-10, obtained by different blurring operators in Guided Shift.

| Operator | Level | ResNet-18 | | | WRN-28-10 | | |
|----------|----------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | | Vanilla | Base | Strong | Vanilla | Base | Strong |
| Median | 3×3 | 82.04 | 92.34 | 90.90 | 91.22 | 91.81 | 90.65 |
| | 5×5 | 83.73 | 91.87 | 90.92 | 91.06 | 91.34 | 88.97 |
| | 7×7 | 83.66 | 91.56 | 90.87 | 90.97 | 90.34 | 87.70 |
| Gaussian | $\sigma = 0.6$ | 83.57 | 91.64 | 90.72 | 91.09 | 90.02 | 90.68 |
| | $\sigma = 1.2$ | 83.63 | 92.56 | 90.05 | 91.14 | 91.81 | 85.02 |
| | $\sigma = 1.8$ | 83.69 | 92.36 | 90.79 | 51.26 | 35.95 | 80.27 |

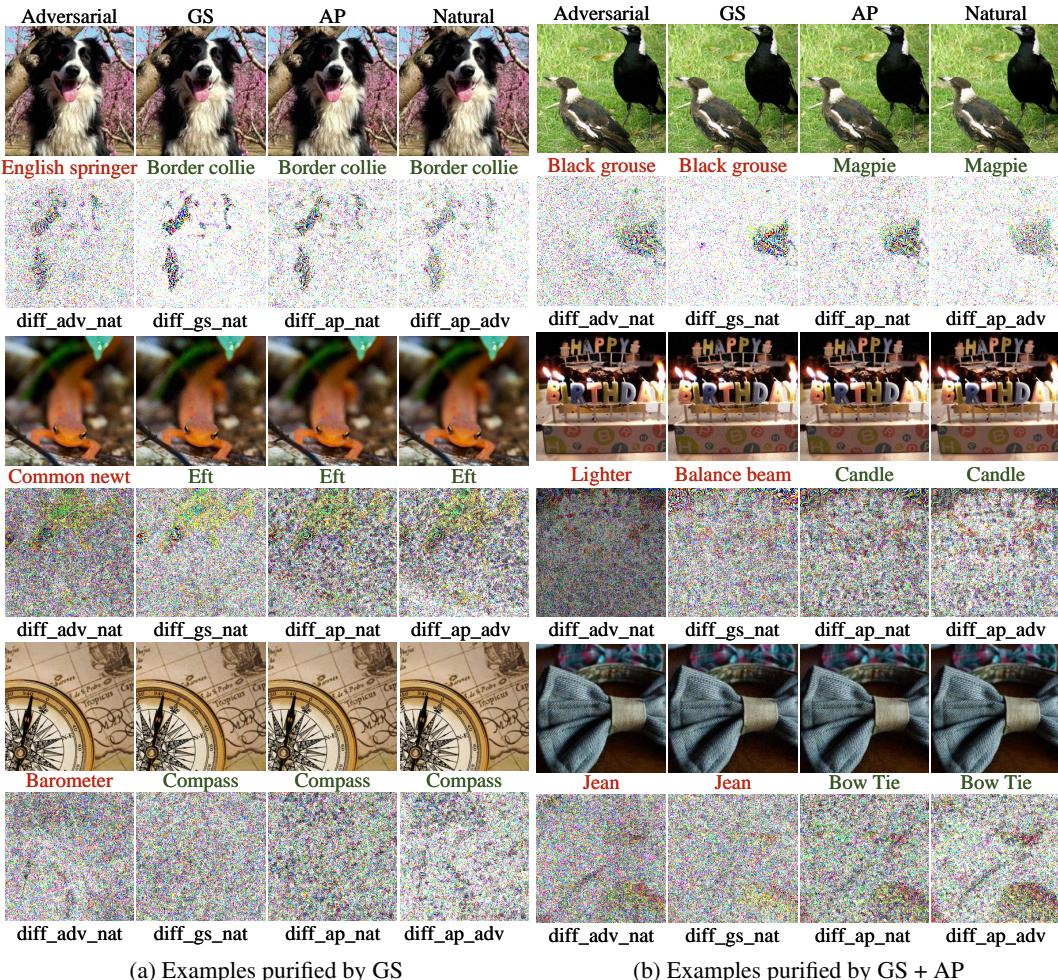


Figure 5: Visual examples of ZeroPur against ℓ_∞ threat model ($\epsilon = 4/255$) on ImageNet-1K. The central two columns depict the results of GS and AP, while the red label denotes error prediction and the green label denotes correct prediction. $\text{diff}_I_a_I_b = \text{abs}(I_a - I_b)$ represents the difference between the two images.