Master's 2 Research Project

Report

SELF-SUPERVISED LEARNING FOR ADVERSARIAL ROBUSTNESS

by

NAZIM MEZHOUDI

DSML Training Student

ETIS UMR8051, CY Cergy Paris Université / ENSEA / CNRS
6 avenue du Ponceau, 95014 Cergy-Pontoise Cedex, France

Defended on March 30th 2022

Jury composed of :

Pr.	Son Vu	ENSEA	Jury President & Project Supervisor
Pr.	Xuan Son Nguyen	ENSEA	Reporter

Self-Supervised Learning for Adversarial Robustness

Nazim Mezhoudi, *ETIS, CY Cergy Paris Université,* Son Vu, *ENSEA, CNRS*

Abstract—In recent years, deep neural network approaches have been widely adopted for machine learning tasks. However, they were shown to be vulnerable to adversarial examples. In fact, carefully crafted small perturbations applied to input data can cause catastrophic mistakes in deep neural network based vision systems. Their vulnerability against such attacks can prove a major roadblock towards their real-world deployment.
In this work we propose a framework leveraging the expressive capability of generative models to defend deep neural networks against such attacks in a self-supervised fashion. Our approach counters the adversarial perturbations with respect to a self-supervised task. At inference time, it generates a close output to a given image which does not contain the adversarial changes. This output is then fed to the classifier. Our proposed method can be used with any classification model and does not modify the classifier's structure or training procedure. It can also be used as a defense against any attack as it does not assume knowledge of the process for generating the adversarial examples. Our code is publicly available at https://github.com/Dastamn/ssr-pix2pix.

Index Terms—Deep Learning, Adversarial Robustness, Self-Supervised Learning, Generative Adversarial Networks.

1 Introduction

Despite their performance on several machine learning tasks, deep networks are still susceptible to adversarial attacks [1]. These attacks come in the form of adversarial examples: legitimate input samples to which carefully crafted perturbations are added. In the context of classification, these perturbations cause the sample to be misclassified at inference time. Such perturbations do not affect human recognition but can drastically change the output of the classifier.

Various studies have been proposed to ensure the robustness of the trained networks against adversarial attacks. Perhaps the most popular approach to achieve adversarial robustness is adversarial training [2], [3], which trains the model with perturbed samples to maximize the loss on the target model. These adversaries can be generated in different ways: using Fast Gradient Sign Method [4] which applies a perturbation in the gradient direction, or Projected Gradient Descent [3] that maximizes the loss over iterations. However, conventional methods with adversarial learning all require class labels to generate adversarial attacks. Moreover, they cause a drastic drop in clean accuracy. In this work we investigate the use of conditional adversarial networks [5] as a general-purpose solution to purify perturbed images and project them back to their original space.

Many problems in image processing and computer vision can be posed as "translating" an input image into a corresponding output image. It is relevant to perceive the space of perturbed images as the input space and the space of clean original images as the output space. Our main idea is to train a Pix2Pix model [6], which leverages generative adversarial networks [7] in a conditional setting, in a self-supervised fashion and use the trained generator as a purifier network. Specifically, we generate perturbed samples using a self-supervised perceptual attack [8], [9] that disrupts the deep perceptual features of the input. We then provide our model with pairs of original clean data and their perturbed counterparts. The min-max formulation of the generative adversarial models coupled with the conditional distance metric between inputs and targets allows the model to learn an optimal input processing function that minimizes perturbations. In a sense, our optimization rule implicitly performs adversarial training. The main advantage of this purifying approach is the fact that it can be applied off the-shelf as a preprocessing step for classification models without impacting its parameters.

2 Related Work

2.1 Adversarial Robustness

Improving the robustness of Deep Neural Networks to adversarial attacks has been an active topic of research since [1] first showed their vulnerability to imperceptible distortions. [4] proposed the Fast Gradient Sign Method, which perturbs a target sample to its gradient direction to increase its loss, and suggested the use of the generated perturbed samples to train the model for improved robustness. Follow-up works proposed multi-step variants of the gradient attack. [10] proposed an Iterative Fast Gradient Sign Method (I-FGSM) that iteratively searches the loss surface of a network under a given metric norm. [3] proposed the Projected Gradient Descent method with better robustness capabilities in adversarial training. These gradient-based attacks have become standard in evaluating the robustness of Deep Neural Networks.

More recent works focus on the vulnerability of the latent representations of the input samples, hypothesizing them as the main cause of the adversarial vulnerability of deep neural networks. [11] demonstrates that adversarial examples can be directly attributed to the presence of non-robust features derived from patterns in the data distribution that are highly predictive, yet incomprehensible to humans. In fact, the distortions used in our approach

are based on the perceptual differences between clean and perturbed samples.

A common requirement of adversarial training techniques is the availability of class labels. For example gradient-based methods rely on cross-entropy loss to find the deceptive gradient direction. Recently, semi-supervised adversarial training approaches [12], [13] have been proposed to improve adversarial robustness. However, they still require a portion of labeled data, and do not change the dependence to class labels. In contrast, our work does not require any class labels since the robust training is done in a self-supervised fashion.

2.2 Self-Supervised Learning

As acquiring manual annotations on data could be costly, self-supervised learning, which generates supervised learning problems out of unlabeled data and solves for them, is gaining in popularity. This process aims to learn intermediate representations of the unlabeled data that are useful for unknown downstream tasks. This is done by solving a pretext task, where the supervision comes from the data itself. Recently, a variety of self-supervised tasks have been proposed on images. Predicting the relative location of image patches [14] has shown to be a successful pretext task. Other works focused on data reconstruction [15] or image colorization [16].

More recently, studies have shown how self-supervised learning can improve adversarial robustness. [9] proposed to use feature distortion as a self-supervised signal to generate attacks that generalize across different tasks. [17] proposed a self-supervised contrastive learning framework for adversarial training that aims to maximize the similarity between random augmentations of a data sample and its instance-wise adversarial perturbation.

2.3 Adversarial Purification

Other works focused on shifting the adversarial examples back to the clean data representation, namely input purification. [18] exploited the capabilities of a denoising auto-encoder DAE to remove adversarial noise. [19] used a U-Net [20] for the same purpose. [9] trained a Purifier network and a Critic network in an adversarial way in order to differentiate between clean and adversarial examples. The adversarial examples were generated in a self-supervised way by computing a perceptual loss between the inputs and the targets in random directions. Our proposed approach is mainly inspired by this work.

2.4 Generative Adversarial Networks

Generative adversarial networks (GANs) [7] aim to model the natural distribution of image input samples by forcing generated samples to be indistinguishable from natural images. They enable a wide variety of applications such as image generation [21], [22], representation learning [23] and others.

Many works have leveraged adversarial learning for image-to-image translation [6], whose goal is to translate an input image from one domain to another domain given input-output image pairs as training data using an adversarial loss. In fact, during the training, the discriminator network learns a trainable loss function and can automatically adapt to the differences between the generated and real images in the target domain. Other methods have also been proposed to learn an image-to-image translation in the absence of training pairs [24].

3 Proposed Approach

An intuitive way to think about solving adversarial robustness is to shift the representation of adversarial examples back to their true classes, i.e. purification. In this work, we consider image classification as our main task.

The basic intuition behind our approach is to effectively use the information contained in the feature space of pairs of clean and perturbed samples as a supervision signal. The objective is to recover the original clean image x given an input adversarial image x'. To this end, we use a Pix2Pix approach to design a Generator network that learns to remove the adversarial distortions based on a self-supervised signal coming from a Discriminator network. Towards this end, the Generator network G is trained in an adversarial manner by playing a min-max game with the Discriminator network D (equation 2) in order to restore clean images from perturbed ones. Our method is summarized in figure 1.

3.1 Pix2Pix Baseline

As mentioned before, the Pix2Pix approach [6] is a conditional GAN (cGAN) [5] framework for image-to-image translation. It consists of a Generator network G and a Discriminator network D. In our case, the Generator G takes as input perturbed images with the objective of translating them back to their original clean state, while the discriminator D takes as input pairs of images (x'_i, x_i) or $(x'_i, G(x'_i))$ where x'_i is a perturbed sample and x_i is its corresponding clean sample. Its goal is to distinguish the original clean "real" images from the purified "fake" ones. The adversarial training of both networks is formulated via the following min-max game:

$$\min_G \max_D \mathcal{L}_{cGAN}(G, D) \tag{1}$$

Where the objective function $\mathcal{L}_{cGAN}(G, D)$ is given by:

$$\mathbb{E}_{x',x}[\log(D(x', x))] + \mathbb{E}_{x'}[1 - \log(D(x', G(x')))] \tag{2}$$

The $\mathcal{L}_{cGAN}$ objective is also mixed with a more traditional loss, such as L1 or L2 distances. The L1 distance is used rather than L2 as it encourages less blurring in the output images.

The Discriminator's training remains unchanged, but the Generator is tasked to not only fool the Discriminator but also to be near the ground truth output, i.e. the original clean image, in an L1 sense as follows:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,x'}\left[||x - G(x')||_1\right] \tag{3}$$

The final Pix2Pix objective is formulated:

$$\arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \tag{4}$$

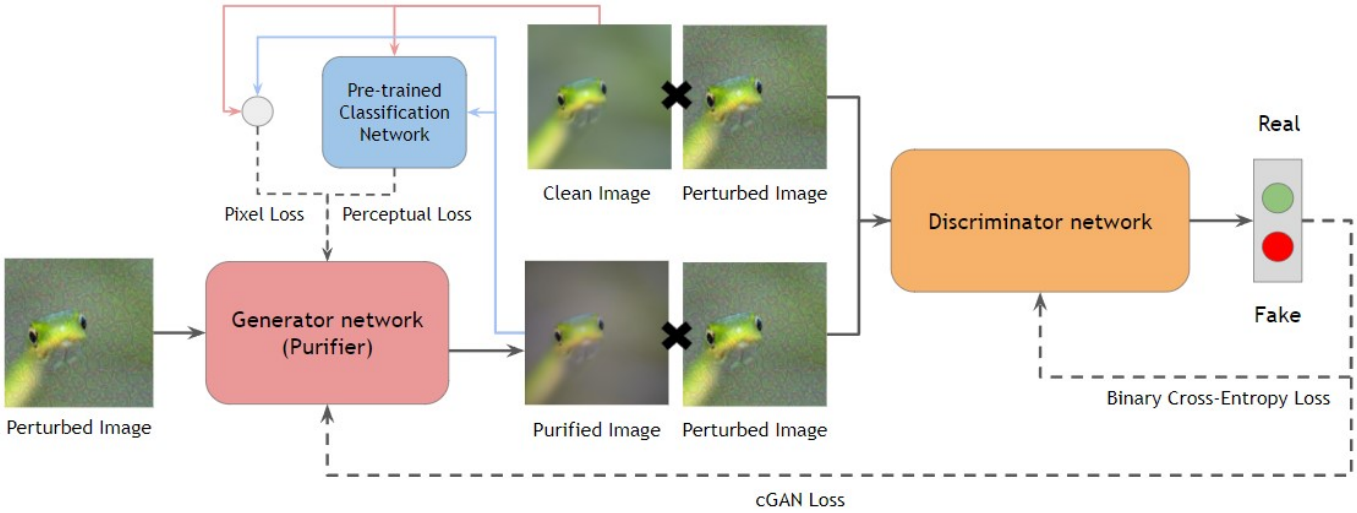


Fig. 1: Architecture of our Pix2Pix approach

Where λ is a weight parameter that characterizes the importance of the L1 loss in the objective. The Pix2Pix approach adopts U-Net [20] as the Generator and a patch-based fully convolutional network [25] as the Discriminator usually referred to as PatchGAN.

3.2 Adversarial Loss

We modify the Pix2Pix objective (equation 4) by minimizing a perceptual loss [8] between the ground truth outputs and the generated purified images at the level of the generator similarly to what have been done in [9], [26], [27]. To that end, we make use of a pre-trained network for image classification.

Convolutional layers first introduced in [28], rely on the fact that pixels in an image are not independent entities but depend in fact on their neighboring pixels. This idea allows the extraction of more information from the images by capturing these dependencies and harnessing them to perform better on the task at hand. Rather than encouraging the pixels of the generated purified image to exactly match the pixels of the original clean image, we instead encourage them to have similar feature representations as computed by the pre-trained network.

Let $P_j(x)$ be the activation of the j^{th} layer of the network P when processing an image sample x. The output of j, if j is a convolutional layer, is the feature map of x corresponding to that layer. The perceptual loss is the distance between these feature representations.

In our experiments, we use the concept of perceptual loss in order to better capture the relations between purified and clean samples and maximize their similarity. Moreover, we replace the L1 pixel-loss of the base Pix2Pix objective with an L2 loss in order to better capture high frequencies in image samples and encourage smoothing to help mitigate the adversarial perturbations.

The final objective function of our approach can be formulated as follows:

$$\arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \alpha\mathcal{L}_p(G) + \lambda\mathcal{L}_{L2}(G) \qquad (5)$$

Algorithm 1 Adversarial Perturbation Generation

Require: Pre-trained classification deep network P, random noise $\mathcal{R}$, batch of clean image samples x, perturbation budget ϵ, step size κ, number of iterations T.

Ensure: Perturbed sample x' satifies $||x - x'||_\infty \leq \epsilon$

$x'_0 = \mathcal{R}(x)$

for t=0 to T-1 **do**

Forward-pass x and x'_t through P and compute:

$$d(x, x'_t) = ||P_n(x) - P_n(x'_t)||_2$$

Compute gradient g_t:

$$g_t = \nabla_x d(x, x'_t)$$

Generate adversaries:

$$x'_{t+1} = x'_t + \kappa sign(g_t)$$

Ensure constraint:

$$x'_{t+1} = \text{clip}\,(x'_{t+1},\, x - \epsilon,\, x + \epsilon)$$

end for

return $x' = x'_T$

3.3 Adversarial Perturbations

The generated adversarial perturbations are obtained without access to model parameters. The approach is inspired by [9]. Strong attacks such as FGSM and PGD that are commonly used for adversarial training consider already-known network parameters to perturb the input samples. Since the perturbations are computed using gradient directions specific to these parameters, the resulting perturbed images would not generalize well to other networks. Using this kind of attacks would not only make the perturbation process dependent on class labels, but also dependent on specific networks which would not permit a transferable defense approach.

To this end, the perturbations used in our approach are based on the concept of feature distortion and take up

the idea of the perceptual loss. Given a clean image x and it's perturbed counterpart x', the feature distortion refers to the change that x' causes to the internal representation of the pre-trained network P relative to x. Basically this perturbation process does the opposite of the adversarial loss discussed previously since it maximizes the feature loss while keeping the pixel difference within a l_∞ norm.

$$\max_{x'} ||P_n(x) - P_n(x')||_2 \,,\ ||x - x'||_\infty \leq \epsilon \tag{6}$$

Where $P_n(x)$ denotes the internal representation of x obtained from the pre-trained deep network P at the n^{th} layer. The perturbation process is detailed in algorithm 1.

3.4 Self-Supervision

As mentioned before, self-supervised learning aims to learn intermediate representations of unlabeled data that are useful to other tasks. In other words, the supervision comes from the data itself.

In our case, the self-supervised signal comes from the Discriminator network. It takes as input two types of image samples: either "real" pairs of perturbed images using the feature distortion technique and their corresponding original clean images, or "fake" pairs of perturbed images and their purified counterpart.
The network is trained to predict if local patches of input images are from the original clean distribution, or purified by our generator. This self-supervised signal is then back-propagated to the Generator so that it can generate better purified images that fool the Discriminator into classifying them as from the original distribution. The training process is detailed in algorithm 2.

Algorithm 2 Pix2Pix Self-Supervised Training

Require: Training data $\mathcal{D}$, Generator G_θ with parameters θ, Discriminator D_ι with parameters ι, perturbation budget ϵ, number of iterations T.
for t=1 to T **do**
Sample a mini-batch of clean samples x from $\mathcal{D}$.
Find adversaries x' computed using Algorithm 1 with perturbation budget ϵ.
Forward-pass x' through G and generate purified samples $G(x') = z$ and compute perceptual loss.
Forward-pass pairs of samples (x', x) and (x', z) through D and compute loss.
Compute adversarial loss and update model parameters θ and ι.
end for

3.5 Pix2Pix architecture

Generator Our generator adopts U-Net architecture [20]. It consists of a convolution block with 2 convolution layers and leaky ReLU activations followed by a series of 4 downsampling and 4 upsampling blocks with symmetric skip connections. Leaky ReLU activations have been used in the encoder part and ReLU in the decoder part. Dropout has also been used at the end of the encoder. More details about the tested architectures are available in the appendix.

Discriminator Our discriminator consists of five convolutional layers with the first one being strided. We used Leaky ReLU as an activation function, dropout, as well as batch normalization.

4 Experiments

4.1 Experimental setup

Our training is done on CIFAR-10 [29] by randomly selecting a 30% portion of the dataset, so 15k training samples. Adversaries are created using the perceptual feature distortion method and are fed as inputs to Pix2Pix with their corresponding clean images used as target labels. During the training we randomly flip the images horizontally. The batch size is set to 16 and the training is done on a GTX 1650. We used an Adam optimizer for both our generator and discriminator with a learning rate of 10^{-4} and parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$.
We used VGG as our pre-trained classification network for the perceptual loss and MSE for the pixel-loss with weight parameters $\alpha = 100$ and $\lambda = 10$ as described in equation 5.

4.2 Feature distortion as an attack

We evaluate the strength of feature distortion based on the perceptual loss as an attack for the classification task. We use VGG as our pre-trained feature extractor network and test both VGG16 and VGG19 architectures by selecting either the first or the first two convolution blocks for low level feature extraction. We compute classification accuracies using VGG16, VGG19 and ResNet18. We also test two possible norm balls: either $l_\infty < 8$ with a perturbation budget of $\epsilon = 8/255$ or $l_\infty < 16$ with $\epsilon = 16/255$. The results are presented in table 1.

Table 1 shows that using two VGG convolution blocks for feature extraction globally provides the best results across various classification evaluations. In the following experiments, we use the feature distortion method with the VGG19 pre-trained model by selecting two convolution blocks and a perturbation budget of $\epsilon = 16/255$.

Table 2 shows comparaisons of the feature distortion attack with other standard supervised perturbation methods, namely FGSM and PGD. These perturbations have been generated with a perturbation budget of $\epsilon = 16/255$ using the VGG19 pre-trained model. They are tested in a white-box setting, i.e. against the same model used for their generation, and in a black-box setting against ResNet18.

Method	VGG19	ResNet18
No attack	92.43	87.99
Feature Distortion $l_\infty < 16$	9.87	11.17
FGSM $l_\infty < 16$	6.39*	17.87
PGD $l_\infty < 16$	**3.48***	**10.87**

TABLE 2: Comparison of feature distortion, FGSM and PGD attack perturbations against VGG19 and ResNet18. Perturbations were generated using VGG19. The (*) indicates a white-box setting for FGSM and PGD.

The results confirm the better generalization capability of feature distortion perturbations. When in the white-box

Pre-trained model	Convolution blocks	Norm	VGG16	VGG19	ResNet18
VGG16	1	$l_\infty < 8$	69.97	70.66	77.20
		$l_\infty < 16$	28.36	31/22	39.72
	2	$l_\infty < 8$	61.89	68.07	75.94
		$l_\infty < 16$	14.89	18.76	26.49
VGG19	1	$l_\infty < 8$	66.80	66.56	74.30
		$l_\infty < 16$	17.22	18.77	24.92
	2	$l_\infty < 8$	68.94	65.49	76.15
		$l_\infty < 16$	**11.17**	**9.87**	**13.80**

TABLE 1: Classification accuracies on VGG16, VGG19, ResNet18 with feature distortion attacks in different settings.

Method	VGG16		VGG19		ResNet18	
	Clean	Feature Distortion $l_\infty < 16$	Clean	Feature Distortion $l_\infty < 16$	Clean	Feature Distortion $l_\infty < 16$
No defense	92.06	11.17	92.43	9.87	87.99	13.80
NRP	67.06	69.04	68.79	71.35	74.51	**74.52**
Ours	**84.45**	**69.43**	**84.78**	**75.16**	**85.22**	71.92

TABLE 3: Comparison between NRP purification and our approach against the feature distortion attack with VGG16, VGG19 and ResNet18

Method	VGG19					ResNet18				
	Clean	FGSM*		PGD*		Clean	FGSM		PGD	
		$l_\infty < 8$	$l_\infty < 16$	$l_\infty < 8$	$l_\infty < 16$		$l_\infty < 8$	$l_\infty < 16$	$l_\infty < 8$	$l_\infty < 16$
No Defense	92.43	21.17	6.39	7.80	3.48	87.99	48.25	17.87	29.87	10.87
AT FGSM $l_\infty < 16$	**72.70**	**59.02**	**38.89**	**51.91**	**37.73**	68.49	55.68	32.96	50.18	32.60
AT PGD $l_\infty < 16$	53.17	**52.30**	**50.94**	**50.88**	**49.64**	60.21	58.46	**55.72**	**56.30**	**53.89**
NRP	68.79	51.52	25.07	43.66	27.73	74.51	**61.01**	**42.71**	**54.60**	**40.69**
Ours	**84.78**	51.34	19.38	38.81	15.23	**85.22**	**66.00**	34.02	54.10	29.49

TABLE 4: Robust comparison between NRP, adversarial training on FGSM and PGD with $l_\infty < 16$, and our approach against FGSM and PGD attacks with various norm-balls. Best accuracy scores are reported with bold underlined **values** and second best results with bold **values**. The (*) indicates a white-box setting.

setting, FGSM and PGD have less impact on the classification accuracy compared to when the attacks have access to the model parameters. This is better reflected in figure 2 where we can clearly see the accuracy gaps.

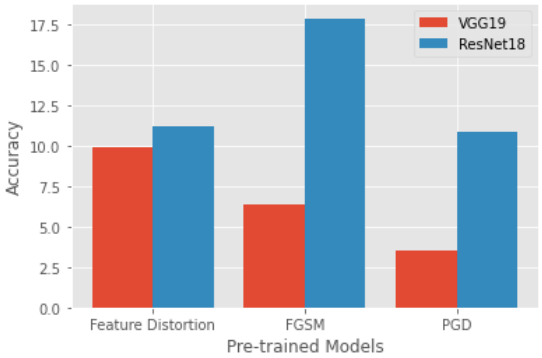


Fig. 2: Accuracy difference between feature distortion, FGSM and PGD with VGG19 and ResNet18

4.3 Defense mechanism

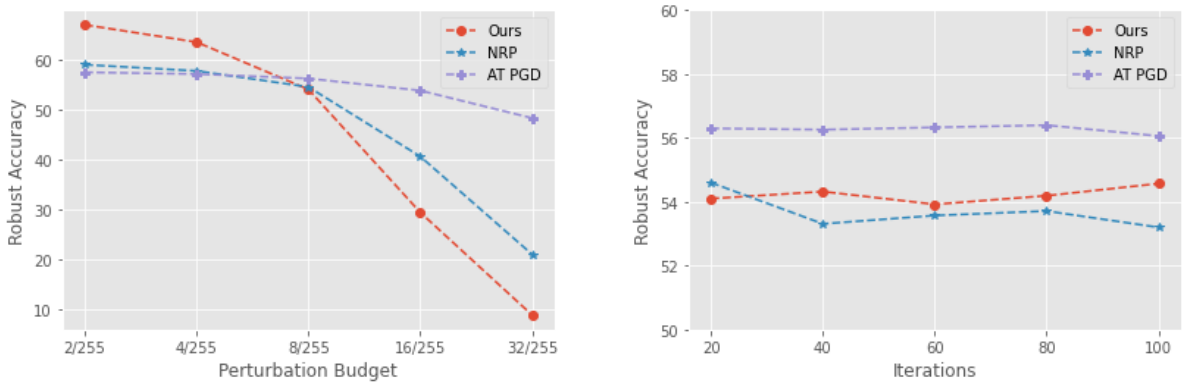
We evaluate the robustness of classification models when provided with perturbed input images.

We first compare the Neural Representation Purifier [9] denoted NRP to our approach when provided with perturbed inputs using the feature distortion attack. The results are reported in table 3. We then compare our defense mechanism against NRP and adversarial training against the gradient-based attacks FGSM [4] and PGD [3]. The results are reported in table 4.

Table 3 shows that our proposed purification approach achieves high robust accuracies with VGG16, VGG19 and ResNet18 when tested against the feature distortion attack. It also has a minimal impact on accuracy when provided with original clean inputs. Our approach is very competitive compared to NRP purification and has better performances in both clean and adversarial situations with VGG16 and VGG19.

Table 4 shows that our proposed approach has the lowest impact on clean accuracy with a score of 85.22% and obtained competitive results when faced with small

Fig. 3: A visual illustration of generated purified images when perturbed with feature distortions. Top: perturbed samples, middle: purified samples, bottom: target clean samples.



(a) Perturbation budget variation with 20 iterations

(b) Iteration variation with $\epsilon = 8/255$

Fig. 4: Robust accuracy of NRP, PGD adversarial training (black-box) and our approach, tested on ResNet18, when varying the perturbation budget and number of iterations of the PGD attack.

perturbations. However, it is not as efficient when faced with attacks with a perturbation budget of $\epsilon = 16/255$.
In the black-box setting, PGD adversarial training obtained the best robust accuracy results when faced with the strongest attacks. Yet, it performed poorly when presented with clean inputs. The same can be said about the FGSM robust model, the latter obtained the worst results in our evaluations. In the white-box setting, gradient-based attacks PGD and FGSM provided the best robust accuracies.
NRP obtained the best overall results with ResNet18 by consistently obtaining the second best robust accuracy scores.

We further investigate the performance of our proposed defense mechanism against the PGD gradient-based perturbations. We first set the perturbation budget to $\epsilon = 8/255$ and record the robust accuracy while increasing the number of iterations of the attack. We then test different perturbation budgets while having a fixed number of 20 iterations.

Figure 4a show the performances of NRP, PGD adversarial training and our approach when increasing the PGD attack norm-ball. Our approach provides the highest robust accuracies with low perturbation budgets. We can see that $\epsilon = 8/255$ is the budget value where accuracies start to diverge. Our method does not scale well with larger perturbations.

Figure 4b shows the impact of PGD iterations on the evaluated defense methods. Our approach remains robust with any number of PGD iterations (e.g. 54.57% under 100 iteration steps).

5 Discussion

Our proposed Pix2Pix approach comes in as a preprocessing defense mechanism for image purification. It provides a reusable solution to effectively purify perturbed image samples in the input space before feeding them to classification models.

For our training procedure, perturbations have been generated by maximizing the perceptual feature differences between perturbed images and their clean counterparts which proved to be an efficacious method to find adversaries that can generalize across different classification models. Our experimental results demonstrate the high robust accuracy of our defense when faced with the above mentioned feature distortion perturbations, it proved to be very competitive when compared to state-of-the-art methods. Our approach proved to be successful in defending against small gradient-based perturbations but is less effective when the perturbation budget is larger. Furthermore, it consistently maintains high accuracy scores when provided with clean original

samples which is hard to achieve when performing adversarial training. Examples of purified image samples can be visualized in figure 3.

6 Conclusion

In this paper, we proposed a new defense mechanism for removing input feature distortion perturbations using an adversarially trained purifier with no dependence to class labels. This purifier has been trained with a self-supervised signal coming from a discriminator network that learns how to distinguish perturbed input samples from original clean ones. It also performs a feature level and a pixel level comparison between its perturbed inputs and its groundtruth target outputs in order to maximize their similarity.

Our proposed approach successfully defends classification tasks with low impact on clean accuracy. It provides a reusable defense mechanism that can be seen as a preprocessing step for various classification tasks. However, it is less performant when faced with strong l_∞ gradient-based attacks. Improving this aspect with further investigations is a key point in developing a fully robust defense mechanism.

Appendix

CIFAR-10 Dataset

The CIFAR-10 dataset [29] is a collection of 32×32 coloured images of objects and animals. The images are classified in 10 classes with 6,000 images per class. There are 50,000 training samples and 10,000 test samples.

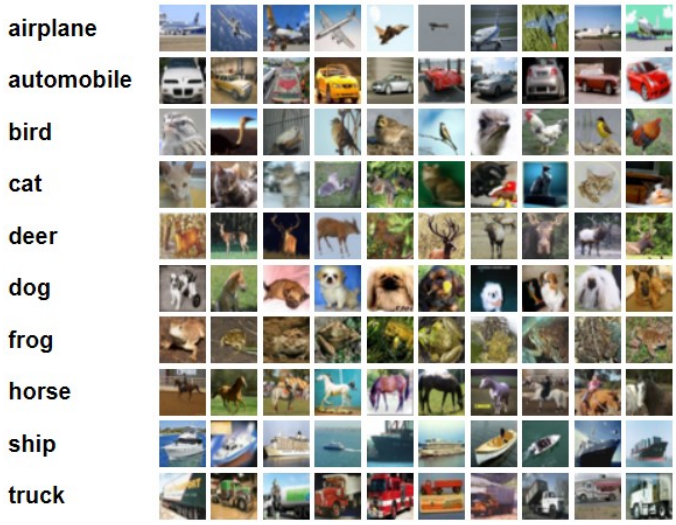


Fig. 5: Random image samples from CIFAR-10 for each class

Training Environment

All our coding was done in Python. We used PyTorch as our deep learning library and Torchvision for easy access to the CIFAR-10 benchmark dataset. We used Albumentations for fast image augmentations and matplotlib as a plotting library. Our training was initially done on an Nvidia GTX 1650 Max-Q, then, in the last few weeks of work, we were given access to ETIS servers and were able to use an Nvidia Quadro RTX 3000.

Classification Model Training

We used VGG16, VGG19 and ResNet18 as our classification models. Since these models are designed to take as input images of size 224x224, we modified their architectures so that they can be trained on 32x32 CIFAR-10 images. The new implementations are based on the work of GitHub user *kuangliu*[1] and *Cheng-Yang Fu*[2]. We trained our models for 100 epochs.

We also trained VGG16 on upsized 224x224 images from CIFAR-10 in order to use it in our adaptation of the NRP purifier [9] since it was originally designed for ImageNet data. To that end, we used the VGG16 model pre-trained on ImageNet and performed Transfer Learning in order to train it on CIFAR-10 by reducing the number of neurons of the last dense layer from 1,000 to 10. We trained this model for 5 epochs. The obtained accuracies are shown in table 5.

Model	Accuracy
VGG16 (224x224)	91.62
VGG16	92.06
VGG19	92.43
ResNet18	87.99

TABLE 5: Classifier accuracies trained on CIFAR-10

Gradient-based Perturbations

As mentioned in section 2.1, adversarial examples can be described as inputs that are specifically crafted to cause machine learning models to output wrong predictions. More formally, we define x as our legitimate input classified by a learner f as a class y:

$$f(x) = y \tag{7}$$

We generate an adversarial example by adding a small perturbation δ_x to x resulting in a misclassification y':

$$f(x + \delta_x) = y' \tag{8}$$

We aim to find the smallest perturbation that produces misclassification. This leads us to the following optimization problem:

$$\arg\min_{\delta_x \in \Delta} ||\delta_x|| \quad \text{s.t.} \quad f(x + \delta_x) = y' \tag{9}$$

Here Δ represents an allowable set of perturbations. A common perturbation set to use, also used in our work, is the l_∞ ball defined by the set:

$$\Delta = \{\delta : \|\delta\|_\infty \leq \epsilon\} \tag{10}$$

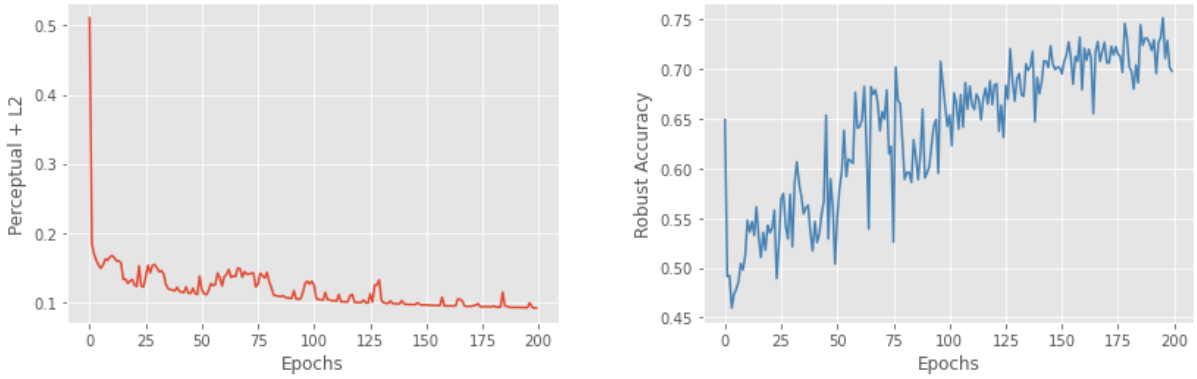
Where the l_∞ norm of a vector v is defined by:

$$\|v\|_\infty = \max_i |v_i| \tag{11}$$

1. https://github.com/kuangliu/pytorch-cifar
2. https://github.com/chengyangfu/pytorch-vgg-cifar10

Architecture	Loss	VGG19	ResNet18
A	L1 ($\lambda = 100$)	60.39	67.30
B1	Perceptual ($\alpha = 10$)	72.36	**76.52**
B2	Perceptual ($\alpha = 100$)	70.58	75.91
B3	Perceptual and L2 ($\alpha = 100$, $\lambda = 10$)	**75.16**	75.67

TABLE 6: Robust accuracy scores of tested Pix2Pix architectures in different settings using VGG19 and ResNet18.



(a) Distance loss (perceptual + L2)

(b) Robust accuracy

Fig. 6: Evolution of the distance loss between generated purified/clean images and the robust accuracy of architecture B3 of our proposed Pix2Pix model.

In other words, we allow the perturbations to have a magnitude between $[-\epsilon, \epsilon]$ in each of v's components, or each of the image pixels in our case.

Since 9 is a non-trivial problem to solve because of the non-linearity and non-convexity of deep neural networks, we employ techniques that approximate a reduced perturbation to misclassify the input.

Fast Gradient Sign Method

It was found in [4] that taking the sign of the gradient of the loss function of a deep neural network with respect to its input and applying perturbations based on these input images reliably produces adversarial examples. This perturbation process can be formulated as follows:

$$x' = x + \epsilon sign(\nabla_x \mathcal{L}(x, y)) \tag{12}$$

Too high values of ϵ result in perturbed samples that do not resemble the original image.

Projected Gradient Descent

Projected Gradient Descent [3] is basically an extension of FGSM where gradient perturbations are applied iteratively with a step size α. We used $\alpha = 10^{-1}$ and 20 iterations.

$$x'_0 = x\,,\; x'_{n+1} = \text{clip}_{x,\epsilon}(x'_n + \alpha sign(\nabla_x \mathcal{L}(x'_n, y))) \tag{13}$$

Adversarial Training

For our adversarial training on FGSM and PGD, we used VGG19 to generate perturbations with a budget of 16/255 and trained a robust classifier using ResNet18 for 50 epochs. For each epoch we generate adversarial examples using the specified attack and we inject them back into the training batch. This can be seen as an alternative data augmentation technique.

Pix2Pix Architecture

We tested two Pix2Pix network architectures. The first one, noted A, is based on the original Pix2Pix implementation [6]. For the generator, we removed the last two downsampling layers and their corresponding upsampling layers. For the discriminator, we removed the last convolution block. These alterations were made because the original model is designed for 256x256 input images. It is overparameterized for our task and computationally expensive to train with its original input size.
The second architecture tested, noted B, has less parameters. The generator has the same number of upsampling and downsampling layers as architecture A, but less parameters. The discriminator has the same number of layers as the original Pix2Pix but a larger patch size (9x9 against 2x2).

We trained our models for 200 epochs (approximately 5 minutes per epoch on a GTX 1650 Max-Q) and tested our architectures in different settings. First with the same loss as the original Pix2Pix using the L1 distance, then using the perceptual loss only, and finally using the perceptual loss mixed with the L2 pixel-loss. The results are shown in table 6. The parameters α and λ are used as in equation 5 and their values are inspired by the works in [6], [27] and [26]. We ended up using architecture B3, figure 6 shows the evolution of the distance loss and the robust accuracy during our training.

Acknowledgments

I would like to thank Pr. Son Vu, my project supervisor, for his advice and guidance. I would like to thank Pr. Xuan Son Nguyen for taking the time to read and evaluate this paper. I would also like to thank the ETIS Laboratory for providing the computing resources needed for this work.

REFERENCES

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2014.

[2] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. P. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada* (H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, eds.), pp. 3353–3364, 2019.

[3] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, OpenReview.net, 2018.

[4] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[5] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR*, vol. abs/1411.1784, 2014.

[6] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5967–5976, IEEE Computer Society, 2017.

[7] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial networks," *CoRR*, vol. abs/1406.2661, 2014.

[8] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*, pp. 694–711, Springer, 2016.

[9] M. Naseer, S. H. Khan, M. Hayat, F. S. Khan, and F. Porikli, "A self-supervised approach for adversarial robustness," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 259–268, Computer Vision Foundation / IEEE, 2020.

[10] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, OpenReview.net, 2017.

[11] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada* (H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, eds.), pp. 125–136, 2019.

[12] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. Liang, "Unlabeled data improves adversarial robustness," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada* (H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, eds.), pp. 11190–11201, 2019.

[13] J. Alayrac, J. Uesato, P. Huang, A. Fawzi, R. Stanforth, and P. Kohli, "Are labels required for improving adversarial robustness?," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada* (H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, eds.), pp. 12192–12202, 2019.

[14] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), vol. 9910 of *Lecture Notes in Computer Science*, pp. 69–84, Springer, 2016.

[15] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008* (W. W. Cohen, A. McCallum, and S. T. Roweis, eds.), vol. 307 of *ACM International Conference Proceeding Series*, pp. 1096–1103, ACM, 2008.

[16] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), vol. 9907 of *Lecture Notes in Computer Science*, pp. 649–666, Springer, 2016.

[17] M. Kim, J. Tack, and S. J. Hwang, "Adversarial self-supervised contrastive learning," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), 2020.

[18] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[19] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 1778–1787, Computer Vision Foundation / IEEE Computer Society, 2018.

[20] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III* (N. Navab, J. Hornegger, W. M. W. III, and A. F. Frangi, eds.), vol. 9351 of *Lecture Notes in Computer Science*, pp. 234–241, Springer, 2015.

[21] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2016.

[22] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *CoRR*, vol. abs/1701.07875, 2017.

[23] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain* (D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, eds.), pp. 2226–2234, 2016.

[24] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 2242–2251, IEEE Computer Society, 2017.

[25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pp. 3431–3440, IEEE Computer Society, 2015.

[26] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 105–114, IEEE Computer Society, 2017.

[27] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[29] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research),"