*ICLR 2018 Reproducibility Challenge

PIXELDEFEND: LEVERAGING GENERATIVE MODELS TO UNDERSTAND AND DEFEND AGAINST ADVERSARIAL EXAMPLES

Authors: Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, Nate Kushman

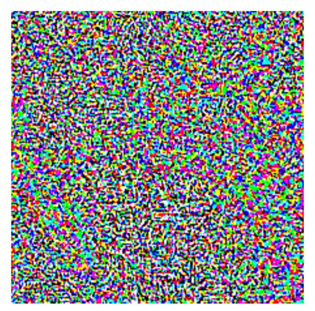
Reproducer: Ruifan Yu David R. Cheriton School of Computer Science, University of Waterloo

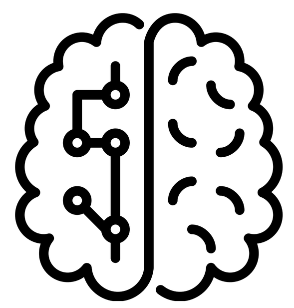
UNIVERSITY OF WATERLOO

Adversarial Attack

Input Image

Fooling a classifier by adding noise imperceptible to humans





Misclassified as **truck**

Perturbations

(generated from attack method)

Previously, these attacks are specific to a given model, where the architecture of the model and the gradient of the label are needed to calculate the perturbation.

Recently, there are results with black-box attack or universal adversarial examples.

Attack Methods

Table 1: Ten different attack methods in this reproduction project

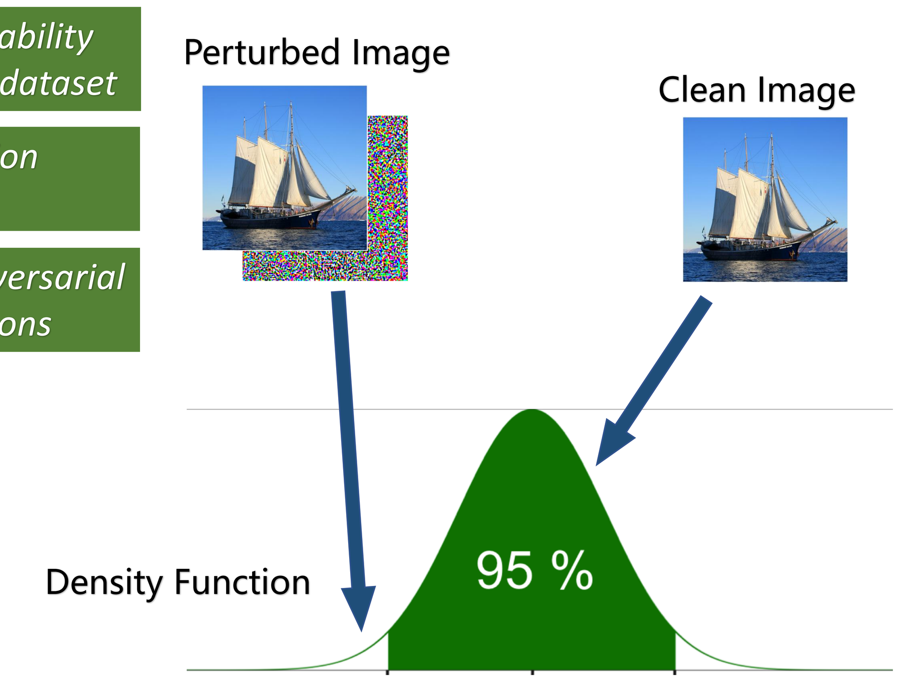
	Attack Method	Description
1	Gradient Sign Attack **(FGSM)**	Adds the sign of the gradient to the image, gradually increasing the magnitude until the image is misclassified.
2	Gradient Attack **(FGM)**	Perturbs the image with the gradient of with respect to the loss of label.
3	DeepFool Attack **(DeepFool)**	Attack the model by finding the nearest decision boundary and making linear iteration to perturb images
4	GaussianBlur Attack **(GaussianBlur)**	Blurs the image until it is misclassified.
5	Saliency Map Attack **(SaliencyMap)**	Attack the model by calculating the saliency map of target label.
6	Uniform Noise Attack **(Rand)**	Adds uniform noise to the image, gradually increasing the standard deviation until the image is misclassified.
7	Salt And Pepper Noise Attack **(SaltAndPepper)**	Increases the amount of salt and pepper noise until the image is misclassified.
8	LBFGS Attack **(LBFGS)**	Uses L-BFGS-B to minimize the distance between the image and the adversarial as well as the cross-entropy between the predictions for the adversarial and the one-hot encoded target class.
9	Contrast Reduction Attack **(ContrastReduction)**	Reduces the contrast of the image until it is misclassified.
10	Iterative FGSM **(BIM)**	Like GradientSignAttack but with several steps for each epsilon.

Motivation

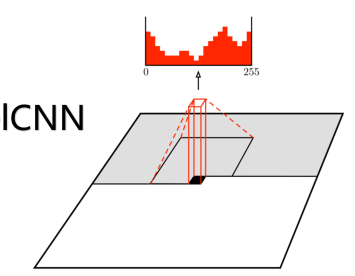
Adversarial examples lies in lower probability regions of density estimation of training dataset

PixelCNN is used as density estimation model

To lower the misclassification: Project adversarial examples back to higher density regions

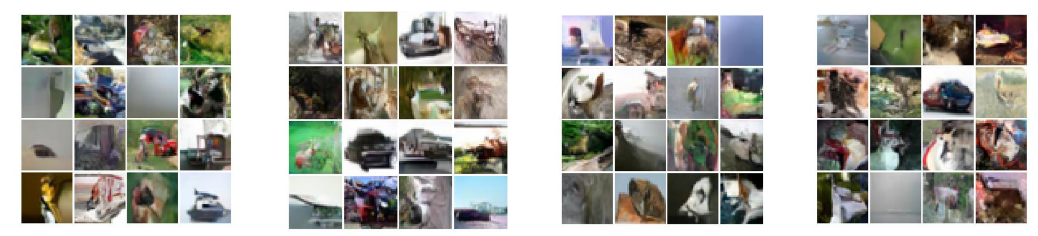


PixelCNN



$p(x) = \prod p(x_i | x_1, \ldots, x_{i-1})$

PixelCNN is an autoregressive generative model for generating image pixel by pixel with tractable probability



PixelDefend

Algorithm 1 PixelDefend

Input: Image $\mathbf{X}$, Defense parameter ϵ_{defend}, Pre-trained PixelCNN model p_{CNN}

Output: Purified Image $\mathbf{X}^*$

$\mathbf{X}^* \leftarrow \mathbf{X}$

for each row i **do**

  for each column j **do**

    for each channel k **do**

      $x \leftarrow \mathbf{X}[i, j, k]$

      Set feasible range $R \leftarrow [\max(x - \epsilon_{\text{defend}}, 0), \min(x + \epsilon_{\text{defend}}, 255)]$

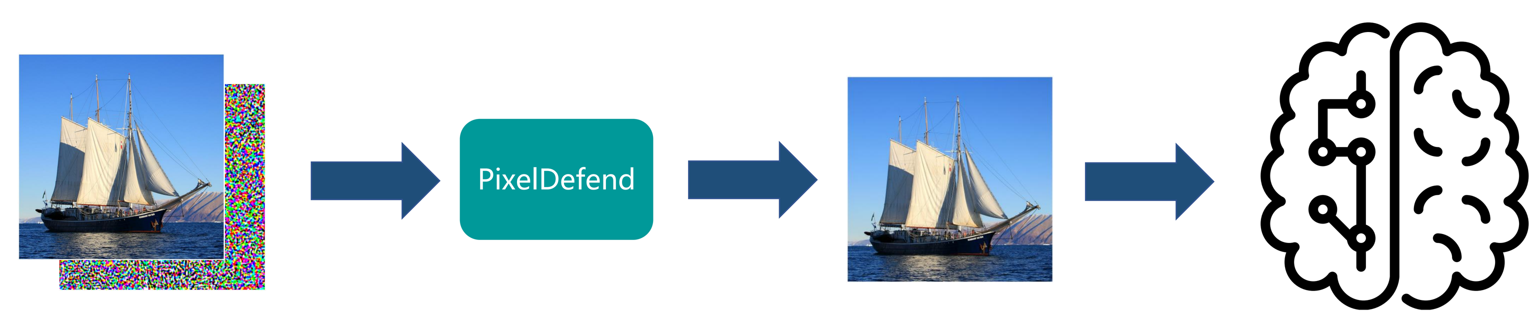
      Compute the 256-way softmax $p_{\text{CNN}}(\mathbf{X}^*)$.

      Update $\mathbf{X}^*[i, j, k] \leftarrow \arg\max_{z \in R} p_{\text{CNN}}[i, j, k, z]$

    end for

  end for

end for



The purpose of this paper is to adopt a PixelCNN generative model for detecting and defending against adversarial perturbed images as it is observed that most adversarial examples lie in low probability regions.

The output from PixelCNN is the exact probability estimation of the pixel value. By training the PixelCNN model on training dataset, the model is predicting the probability of given images that belongs to the same distribution of training data.

$$p = \frac{1}{N+1}\left(\sum_{i=1}^{N} \mathbb{I}[T_i \le T] + 1\right) = \frac{T+1}{N+1} = \frac{1}{N+1}\left(\sum_{i=1}^{N} \mathbb{I}[p_{\text{CNN}}(\mathbf{X}_i) \le p_{\text{CNN}}(\mathbf{X}')] + 1\right)$$

Intuitively, the p-value is telling how many images in training data are less likely to be sampled from training data distribution. For normal images, the rank of its p value should be random since every image is equally likely sampled from training data distribution.

$$\text{BPD}(\mathbf{X}) \triangleq -\log p_{\text{CNN}}(\mathbf{X}) / (I \times J \times K \times \log 2)$$

Here, Bits Per Dimension is another measurement of the probability, which is same as negative log-likelihood.

Reproducible? Yes

Claim #1 APPROVED

Adversarial examples mainly lie in the low probability regions of the training distribution, regardless of attack. The inputs are more outside of the training distribution if their p-value distribution has a larger deviation from uniform.

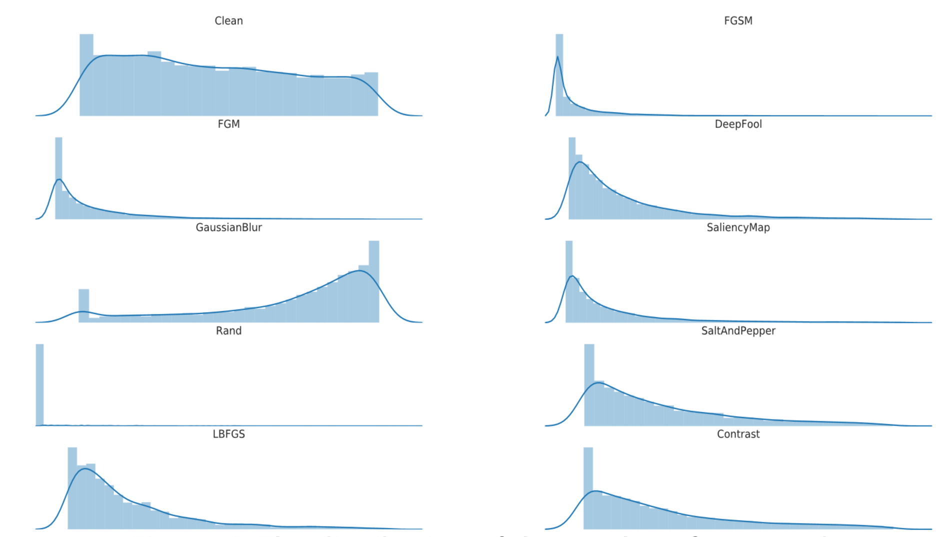


Figure 1: The distribution of the p-value of ten attacks on ResNet18 onCIFAR-10 dataset

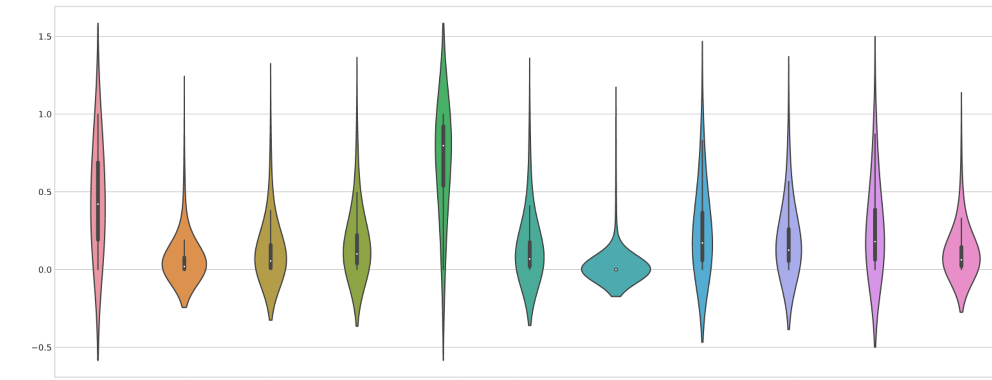


Figure 2: The violin plot of the Bits Per Dimension(BPD) Distributions over ten attack methods

The distribution of log-likelihoods show considerable difference between perturbed images and clean images. The log-likelihoods from PixelCNN also provide a quantitative measure for detecting adversarial examples.

* PixelCNN model has no information about the attacking methods for producing those adversarial examples, and no information about the ResNet model either

APPROVED Claim #2

PixelDefend can be used to protect already deployed models

Table 2: All attacks are designed to fool the ResNet on every test image. These numbers show the accracy after a PixelDefend is deployed on a model

Attack Method	PixelDefend Accuracy
Clean Image	82%
Gradient Sign Attack (FGSM)	64%
Gradient Attack (FGM)	60%
DeepFool Attack (DeepFool)	75%
GaussianBlur Attack (GaussianBlur)	24%
SaliencyMap Attack (SaliencyMap)	45%
Uniform Noise Attack (Rand)	25%
Salt And Pepper Noise Attack (SaltAndPepper)	31%
LBFGS Attack (LBFGS)	55%
Contrast Reduction Attack (ContrastReduction)	87%
Iterative FGSM (BIM)	Unkown

*In this project, we found that PixelCNN is extremely slow in training and generating predictions. Therefore, in the purification part, we were only able to test a small dataset with 500 images. (Even with K80 GPU, the generation of purification takes around 8 minutes for each image).

* All experiments ran on ResNet18 on PyTorch

APPROVED Claim #3

PixelDefend is a model-attack-agnostic defense strategy that can make model predict the correct label.

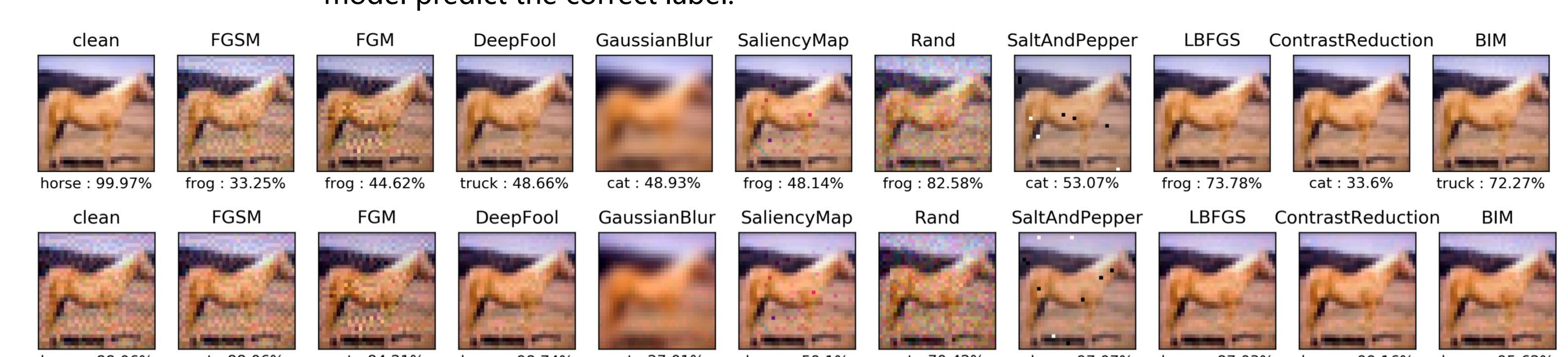


Figure 3: the images after the purification by PixelDefend
The texts below each image are the predicted label and confidence given by ResNet

Paper Reviews

“

- *This paper introduces a novel family of methods for defending against adversarial attacks based on the idea of purification.*
- *PixelDefend can be used to protect already deployed models and be combined with other model-specific defenses.*
- *Adversary to PixelCNN. It is not clear why a PixelCNN may not be adversarially attacked, nor if such a model would be able to guard against an adversarial attack.. Are there adversarial examples for PixelCNN model?*

In our opinion, it would be interesting to be what is the performance after the model is fine-tuned on the PixelDefend outputs. However, due to the slow generation speed, we were not able to generate enough purified images for further experiments.
Since this project is a reproduction challenge, we focused on the PixelCNN model. As future works, we will try other generative models and make a comparison.

”