Feature Denoising for Improving Adversarial Robustness

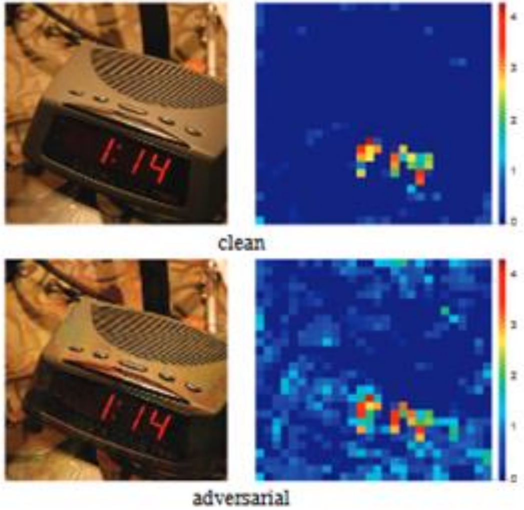
Saeed Rahaman & Quoc Le

Outline

- Problem Description
- Paper Solution
- Experiments & Results
- Conclusion
- Pros & Cons
- Q&A

What is the Problem?

- Perturbations add noise to an image which lead to noise in the feature map
- Changes the semantically important regions



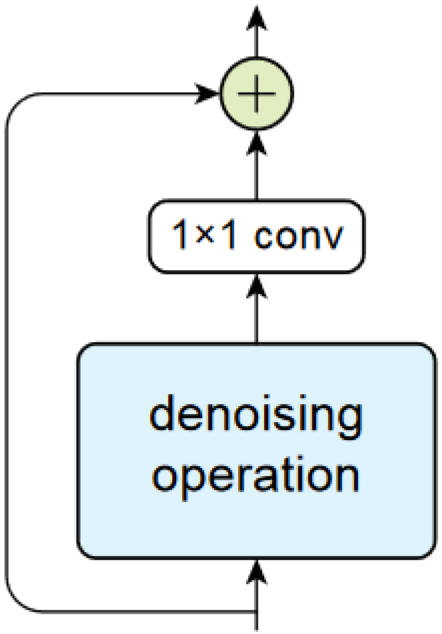
Proposed Solution

1. Denoise the feature map by adding denoising blocks in between layers

1. Train networks on adversarial examples to reduce feature-map noise

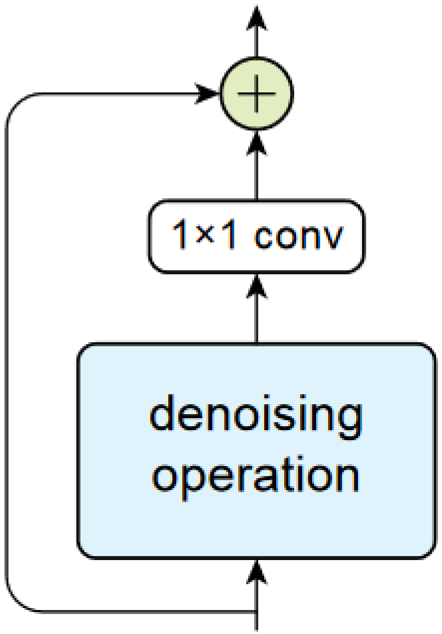
Denoising Block

- Input is feature maps from a previous convolutional layer
- Denoising operation is performed to remove noise
- 1x1 convolution to normalize the denoised feature map
- The denoised feature map is combined with the original feature map

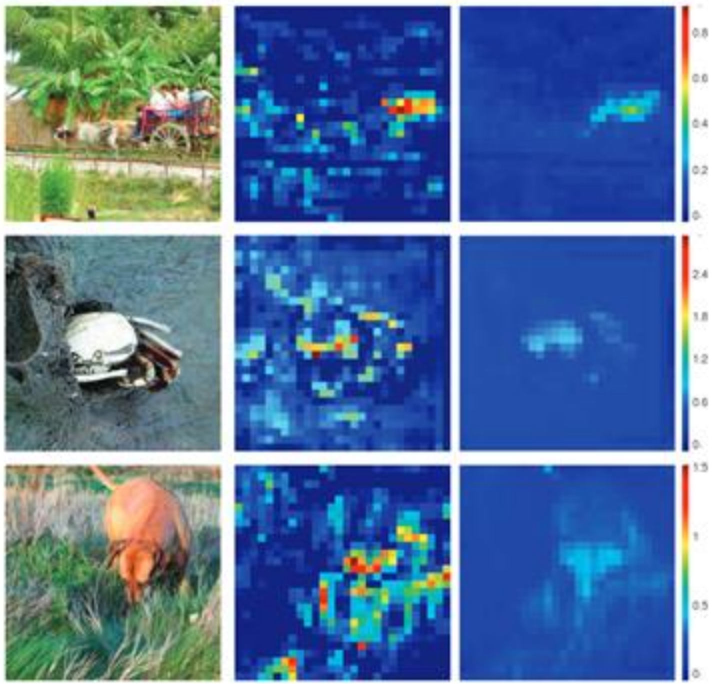


Denoising Block Design

- Removing 1x1 convolution reduces accuracy significantly
- Denoising by itself in not impactful
- Removing the residual connection makes network unstable
- Combining the input features with denoised features is imperative for increased accuracy



Effect of Denoising Blocks



- After adding 4 denoising blocks to ResNet-152
- Noise in feature map significantly reduced

Denoising Operations - Non-local Means

$$y_i = \frac{1}{C(x)} \sum_{\forall j \epsilon L} f(x_i, x_j) \cdot x_j$$

- $C(x)$ is a normalization function
- $f(x_i, x_j)$ is a feature weighting function
- i and j refers to spatial location on the feature map
- L is all spatial locations in the image
- y_i is the feature map output for a specific location

Gaussian Implementation

- The weight function is $f(x_i, x_j) = e^{\frac{1}{\sqrt{d}}\theta(x_i)^T \phi(x_j)}$

- The normalization function is $\mathcal{C} = \sum_{\forall j \in \mathcal{L}} f(x_i, x_j)$

- $\Theta(x)$ and $\phi(x)$ are embedded versions of x obtained through 1x1 convolutions
    - Where $\Theta(x_i) = W_\Theta x_i$ and $\phi(x_j) = W_\phi x_j$

- d = number of channels in the feature map

- f/C is similar to the softmax functions

Dot Product Implementation

- The weight function is $f(x_i, x_j) = x_i^T x_j$
- Normalization function is C(x) = N
    - N is the number of pixels in x
- Requires less parameters than the Gaussian Version

Denoising Operations - Bilateral Filter

$$y_i = \frac{1}{C(x)} \sum_{\forall j \epsilon \Omega(i)} f(x_i, x_j) \cdot x_j$$

- Is same as Non-Local Mean except looks at the “Local Regions” **Ω**(i)
- Dot product and Gaussian implementation can be used here as well

Denoising Operations - Mean Filter

- Use average pooling with stride of 1
- Simplest form of denoising
- Reduces the noise image but will smoothes the structures of the image

Denoising Operations - Median Filter

$$y_i = \mathrm{median}\{\forall j \in \Omega(i) : x_j\},$$

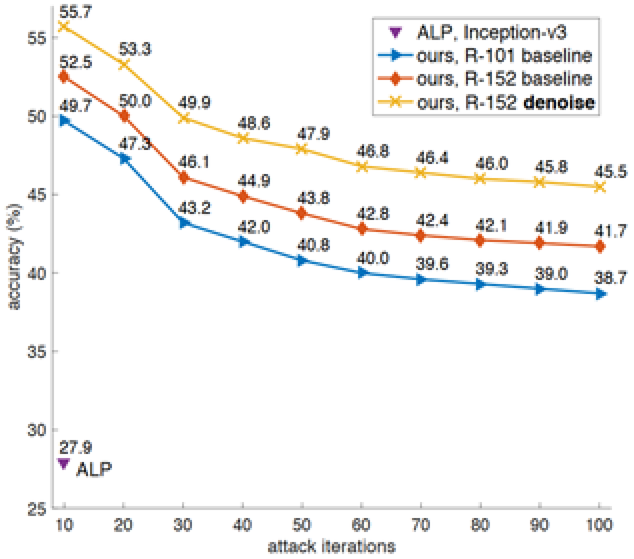
- Perform on the local region **Ω**(i)
- Perform on each channel separately
- Good at removing salt-and-pepper noise and other similar type of outliers

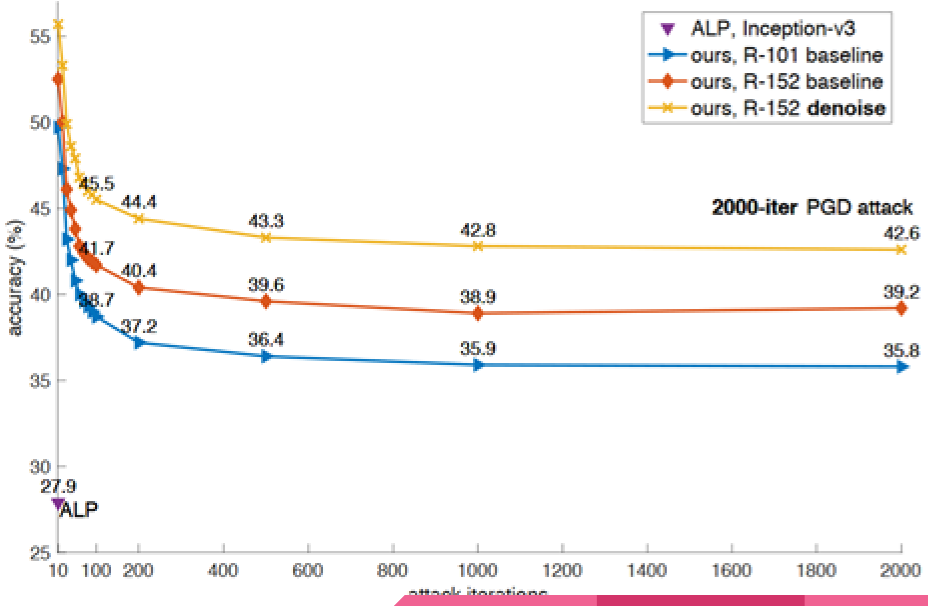
Adversarial Training

- Generated adversarial examples with PGD
- For each mini-batch of images
  - PGD attack is performed
  - SGD is performed on perturbated images and weights are updated
  - Drastically increases runtime (30 iterations per image)
- ResNet-101 and ResNet-152 trained as baseline models
- 4 denoising blocks are added to ResNet-152 and trained
  - Used non-local means with Gaussian

Experiments

- PGD used on ImageNet to attack models
  - Iterations range from 10-2000





Denoising Blocks in Non-Adversarial Networks

- Denoising blocks do not affect accuracy against non-adversarial images
- Adversarial training effects accuracy on non-adversarial images

Model (testing on clean images)	**Accuracy**
ResNet-152 (baseline) (no A.T.)	78.91%
ResNet-152(denoised)(no A.T.)	79.08%
ResNet-152 (baseline)(A.T.)	62.32%
ResNet-152(denoised)(A.T.)	65.30%

Denoising Operations in Non-Adversarial Networks

- Different denoising operations have similar performance on clean images

model	accuracy (%)
R-152 baseline	78.91
R-152 baseline, run 2	+0.05
R-152 baseline, run 3	-0.04
+4 bottleneck (R-164)	+0.13
+4 denoise: null (1×1 only)	+0.15
+4 denoise: 3×3 mean filter	+0.01
+4 denoise: 3×3 median filter	-0.12
+4 denoise: bilateral, Gaussian	+0.15
+4 denoise: non-local, Gaussian	+0.17

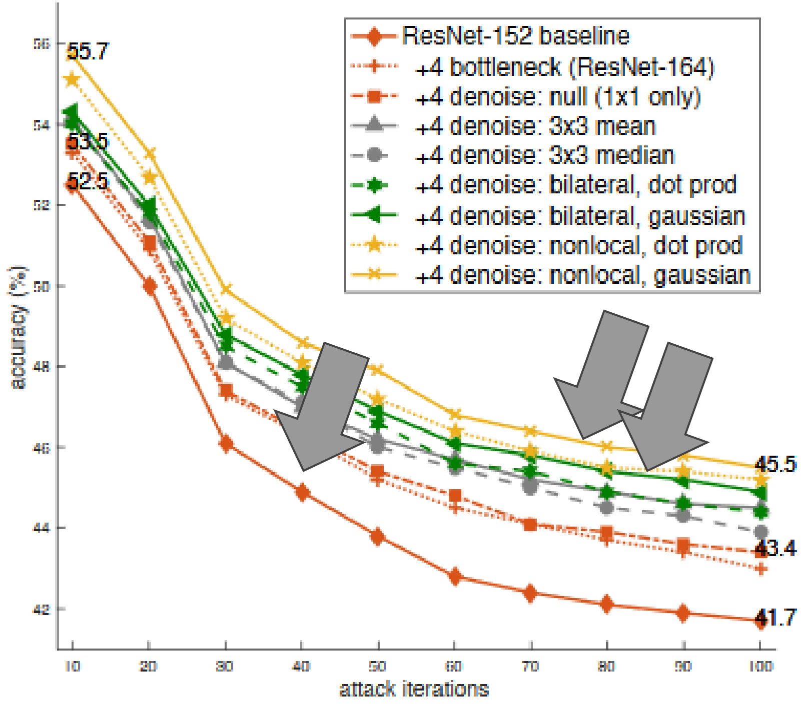
Comparison of Denoising Operators

- Compare Denoising block against the adversarial trained ResNet-152, Null Denoising Block, and Bottleneck Block
    - Null denoising block only had the 1 x 1 normalization block
    - Bottleneck block refers bottleneck block designed by He et al. [1]

- Used 4 blocks were added to ResNet-152

[1] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

Result - Comparison of Denoising Operators



Black-Box Attack Result

- Evaluation criteria was to poll all of black-box attack they used to determine if a misclassification occurred
- Used the 5 best attackers of the IPS 2017 CAAD competitions

model	accuracy (%)
CAAD 2017 winner	0.04
CAAD 2017 winner, under 3 attackers	13.4
ours, R-152 baseline	43.1
+4 denoise: null (1×1 only)	44.1
+4 denoise: non-local, dot product	46.2
+4 denoise: non-local, Gaussian	**46.4**
+all denoise: non-local, Gaussian	**49.5**

Conclusion

- Showed that denoising the feature map improves the robustness of the model
- Proposed a new architecture style that includes a denoising block to improve robustness of the model
- Their proposed model had a significant improvement on accuracy against adversarially attacks

Strengths

- Shows a significant improvement against adversarial images
- Implements a new architecture style to improve the robustness of networks
- Robust against a 2000-iteration PGD attack
- Denoising blocks do not reduce performance against clean images

Weaknesses

- Did not have metric on the loss of feature detail on the feature map
- Does not explore why denoising blocks are effective
- Does not test against other white-box attacks
- Only looked at one architecture

Questions?