BlurNet: Defense by Filtering the Feature Maps

Ravi Raju

September 26, 2019

Outline

Introduction

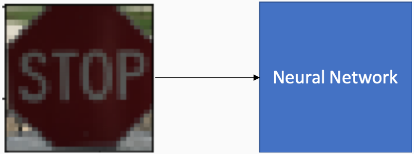
Background

Proposal

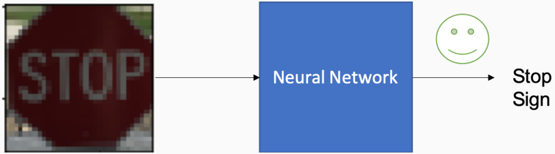
Learning the Filter Parameters

Introduction

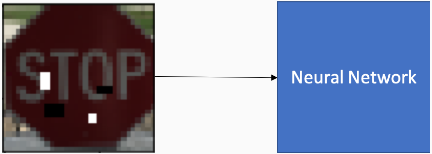
Vulnerabilities in NNs



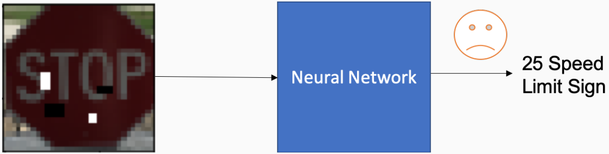
Vulnerabilities in NNs



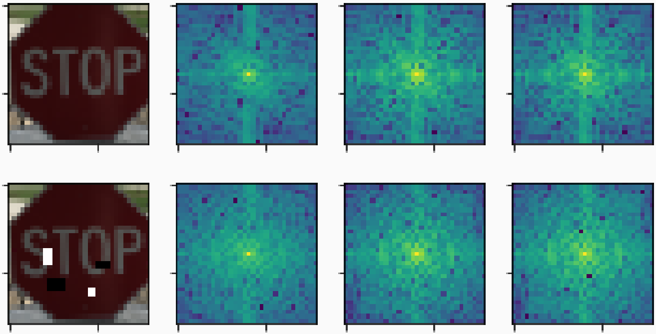
Vulnerabilities in NNs



Vulnerabilities in NNs

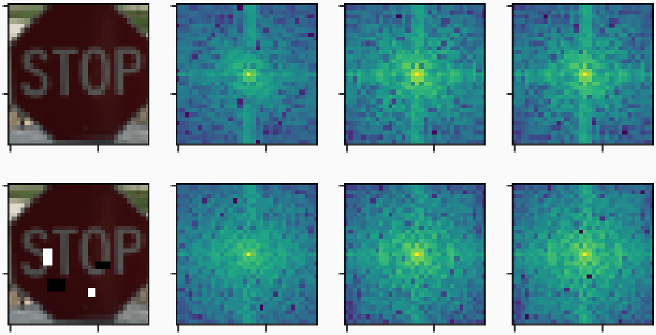


FFT Spectrum of Input Channels



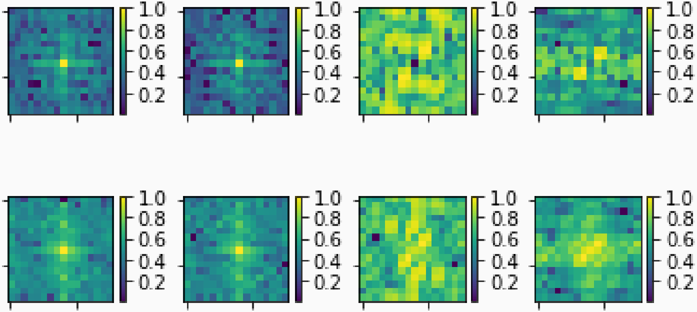
- Log-shifted and normalized frequency spectrum of RGB channels of a natural and perturbed stop sign image

FFT Spectrum of Input Channels



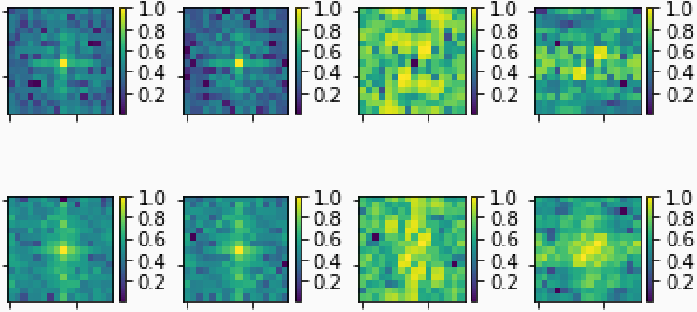
- Log-shifted and normalized frequency spectrum of RGB channels of a natural and perturbed stop sign image
- Lower freqencies correspond to the center and higher ones to the edge.

FFT of First Layer Feature Maps



- Each row corresponds to a unique feature map from L1 layer of network.

FFT of First Layer Feature Maps



- Each row corresponds to a unique feature map from L1 layer of network.
- Difference image suggests that the perturbations induce high frequency components not found in natural stop signs

Background

Terminology and Preliminaries: Adverserial Robustness

Let x be an image and a neural network, $F(x) = y$, such that $F : \mathbb{R}^{h*w*c} \rightarrow \mathbb{R}$ where y is the correct label.

- Main goal of an attacker - generating an image, x_{adv} such that $F(x_{adv}) \neq y$ by perturbing the input pixels.

Terminology and Preliminaries: Adverserial Robustness

Let x be an image and a neural network, $F(x) = y$, such that $F : \mathbb{R}^{h*w*c} \rightarrow \mathbb{R}$ where y is the correct label.

- Main goal of an attacker - generating an image, x_{adv} such that $F(x_{adv}) \neq y$ by perturbing the input pixels.
- Main goal of a defense algorithm - classify the perturbed image with the correct label

Terminology and Preliminaries: Adverserial Robustness

Let x be an image and a neural network, $F(x) = y$, such that $F : \mathbb{R}^{h*w*c} \rightarrow \mathbb{R}$ where y is the correct label.

- Main goal of an attacker - generating an image, x_{adv} such that $F(x_{adv}) \neq y$ by perturbing the input pixels.
- Main goal of a defense algorithm - classify the perturbed image with the correct label
- Threat Model - rules which dictate what kind of information an attacker
  1. White-box attacks - attacker has access to the model, architecture gradients, etc.

Terminology and Preliminaries: Adverserial Robustness

Let x be an image and a neural network, $F(x) = y$, such that $F : \mathbb{R}^{h*w*c} \rightarrow \mathbb{R}$ where y is the correct label.

- Main goal of an attacker - generating an image, x_{adv} such that $F(x_{adv}) \neq y$ by perturbing the input pixels.
- Main goal of a defense algorithm - classify the perturbed image with the correct label
- Threat Model - rules which dictate what kind of information an attacker
  1. White-box attacks - attacker has access to the model, architecture gradients, etc.
  2. Black-box attacks (Transfer Attacks) - attacker has knowledge of model architecture but not parameters, etc.

Metrics

1. Attack Success Rate - number of predictions altered by an attack, $\frac{1}{N}\sum_{n=1}^{N} \mathbb{1}[F(x_n) \neq F(x_{nadv})]$.

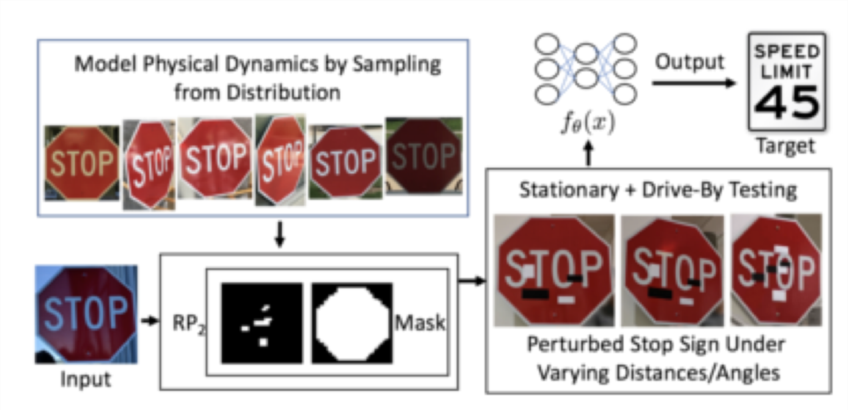
Metrics

1. Attack Success Rate - number of predictions altered by an attack, $\frac{1}{N}\sum_{n=1}^{N}\mathbb{1}[F(x_n) \neq F(x_{nadv})]$.
2. L_2 Disimilarity Distance - how different is the adverserial image from the original, $\frac{1}{N}\sum_{n=1}^{N}\frac{||x-x_{adv}||_p}{||x_{adv}||_p}$.

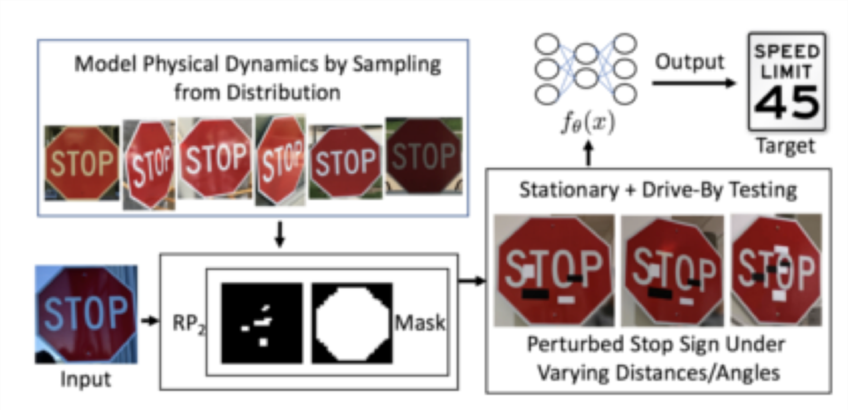
Metrics

1. Attack Success Rate - number of predictions altered by an attack, $\frac{1}{N}\sum_{n=1}^{N} \mathbb{1}[F(x_n) \neq F(x_{nadv})]$.
2. L_2 Disimilarity Distance - how different is the adverserial image from the original, $\frac{1}{N}\sum_{n=1}^{N} \frac{||x - x_{adv}||_p}{||x_{adv}||_p}$.
3. An attacker is considered strong if its attack success rate is high while having a low dissimilarity metric.

Attacker: Robust Physical Perturbation (RP_2) Attack



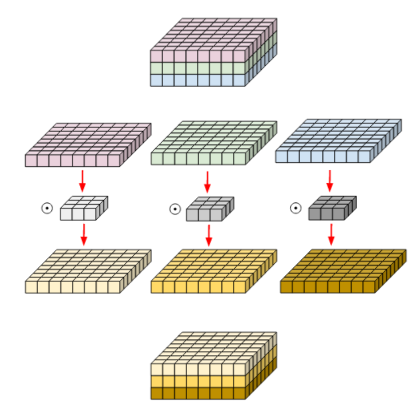
Attacker: Robust Physical Perturbation (RP_2) Attack



$$\arg\min_{\delta} \quad \lambda \| M_x \cdot \delta \|_p + \mathbb{E}_{x_i \sim X^V} J(f_\theta(x_i + T_i(M_x \cdot \delta)), y^*)$$

Proposal

Low-pass filtering



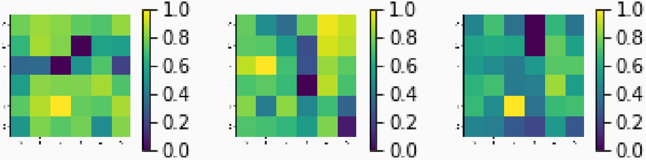
- Induce low-pass filtering with standard blur kernels by inserting a depthwise convolution after the first convolution layer.

Filtering input vs. Filtering Feature Maps

Table 1: Results from black box evaluation

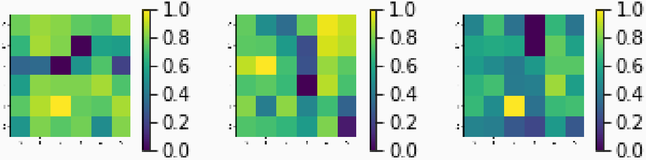
	Accuracy	**Attack Success Rate**
Baseline	100%	90%
Input filter 3x3	100%	87.5%
Input filter 5x5	100%	67.5%
3x3 filter on L1 feature maps	100%	65%
5x5 filter on L1 feature maps	87.5%	17.5%

Filtering in the higher layers



1. The FFT Spectrum of a subsampling of feature maps from the second layer of the network.

Filtering in the higher layers



1. The FFT Spectrum of a subsampling of feature maps from the second layer of the network.
2. High frequency information is relevant to maintain decent classification.

Learning the Filter Parameters

Minimizing accuracy loss

- Can we learn the filter parameters so that we gain low-pass filtering without a significant degradation in accuracy?

Minimizing accuracy loss

- Can we learn the filter parameters so that we gain low-pass filtering without a significant degradation in accuracy?
- Regularization
    1. L_∞ regularization of the depthwise convolution layer

Minimizing accuracy loss

- Can we learn the filter parameters so that we gain low-pass filtering without a significant degradation in accuracy?
- Regularization
    1. L_∞ regularization of the depthwise convolution layer
    2. Total Variation (TV) regularization of the Layer 1 convolution weights

Ł$_\infty$ Regularization

$$\min \quad \frac{\alpha}{K}\sum_{j=1}^{K} \|W_{depthwise}[:,:,j]\|_\infty + J(f_\theta(x,y))$$

Ł$_\infty$ Regularization

$$\min \quad \frac{\alpha}{K}\sum_{j=1}^{K}\|W_{depthwise}[:,:,j]\|_\infty + J(f_\theta(x,y))$$

L_∞ norm is an ideal choice for the depthwise weights. This will ensure that the weights in the kernel take similar values to each much like a low pass filter.

Total Variation Regularization

$$TV(x) = \sum_{i,j} |x_{i+1,j} - x_{i,j}| + |x_{i,j+1} - x_{i,j}|.$$

Total Variation Regularization

$$TV(x) = \sum_{i,j} |x_{i+1,j} - x_{i,j}| + |x_{i,j+1} - x_{i,j}|.$$

$$\min \quad \alpha_{TV} \frac{1}{N \cdot K} \sum_{i=1}^{N} \sum_{i=1}^{K} TV(\mathcal{F}[i, :, :, k]) + J(f_\theta(x, y)),$$

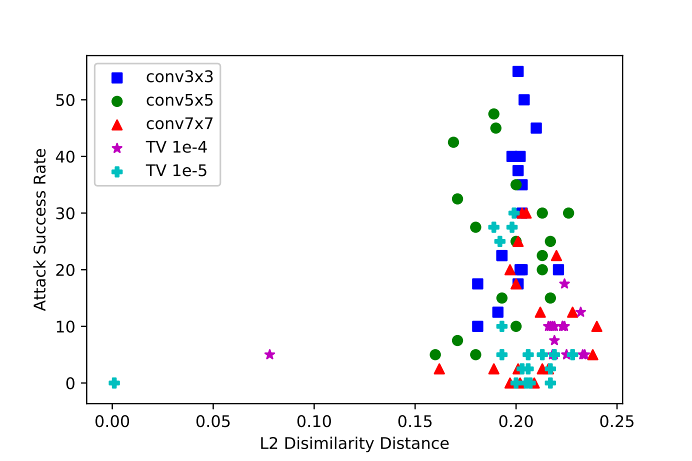
Total Variation Regularization

$$TV(x) = \sum_{i,j} |x_{i+1,j} - x_{i,j}| + |x_{i,j+1} - x_{i,j}|.$$

$$\min \quad \alpha_{TV} \frac{1}{N \cdot K} \sum_{i=1}^{N} \sum_{i=1}^{K} TV(\mathcal{F}[i,:,:,k]) + J(f_\theta(x,y)),$$

TV encourages the neighboring values in the feature maps to be similar so the high spike introduced by RP_2 would be diminished.

Whitebox Evaluation



Whitebox Evaluation

Table 2: Results from white box evaluation

	α	**Legitimate Acc.**	**Average Success Rate**	**Worst Success Rate**	L_2 **Distortion**
Baseline	0	91%	49.18%	90%	0.207
3x3 conv	10^{-5}	86.3%	30%	55%	0.201
5x5 conv	0.1	86.3%	24.11%	47.5%	0.189
7x7 conv	0.1	87%	11.61%	30%	0.203
TV	10^{-4}	85.6%	7.92%	17.5%	0.224
TV	10^{-5}	82.3%	8.47%	30%	0.199