Towards Interpretable Deep Neural Networks by Leveraging Adversarial Examples

Authors: Yinpeng Dong, Fan Bao, Hang Su, and Jun Zhu

Presenter: Masoud Hashemi
Facilitators: Ali Fathi and Zak Semenov

March 21, 2019

AISC

Motivation

- Despite of the impressive performance of the Deep Neural Networks (DNN), the results are not **interpretable**.
- Different approaches have been suggested to address this problem.
- Another active area of research in DNN is **adversarial examples and adversarial attacks**.

- Main Question of this paper: What is the reaction of the interpretation algorithms to the adversarial examples.

Key Contributions/Results

1. Representation of the adversarials are not consistent with the normal images
2. A new metric is proposed to measure the consistency of the neurons of a DNN
3. A new adversarial training is suggested to increase the consistency
4. The proposed adversarial training reduces the ambiguity of neurons

Main Results/claims

Example of the inconsistency in the normal vs. adversarial representation.

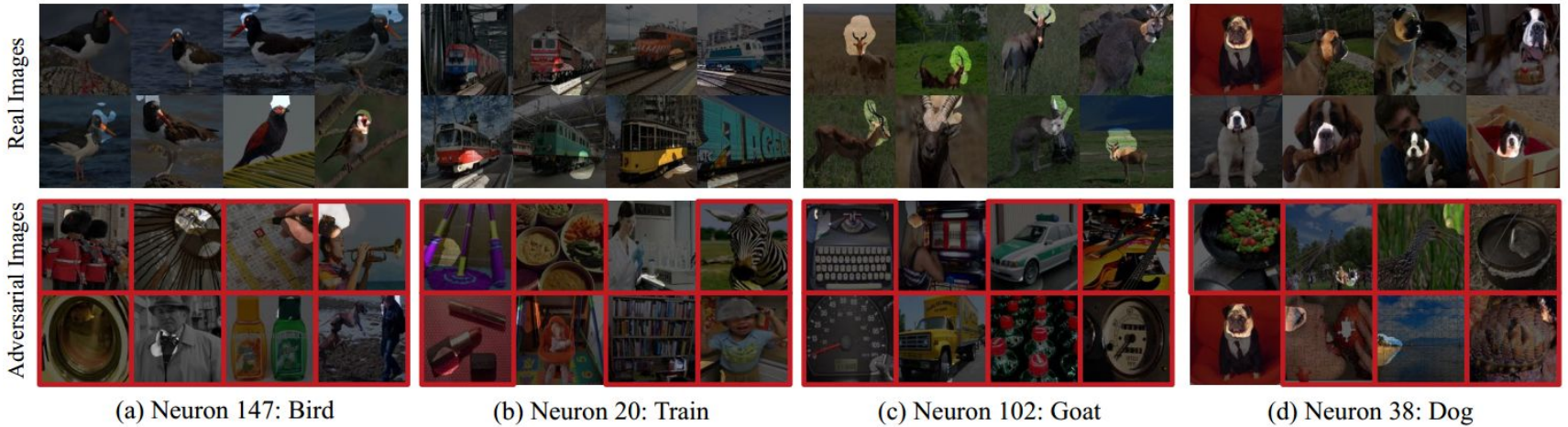


Figure 3: The real and adversarial images with highest activations for neurons in VGG-16 *conv5_3* layer.

Outline

1. Quick review of DNN interpretability.
2. What is adversarial attack?
3. Adversarial Training

4. Consistency metric
5. The proposed adversarial training method
6. Experiments and Results

DNN Interpretation

1. Methods based on Forward passing
2. Methods based on Gradient backpropagation
3. Perturbation/Black-box methods

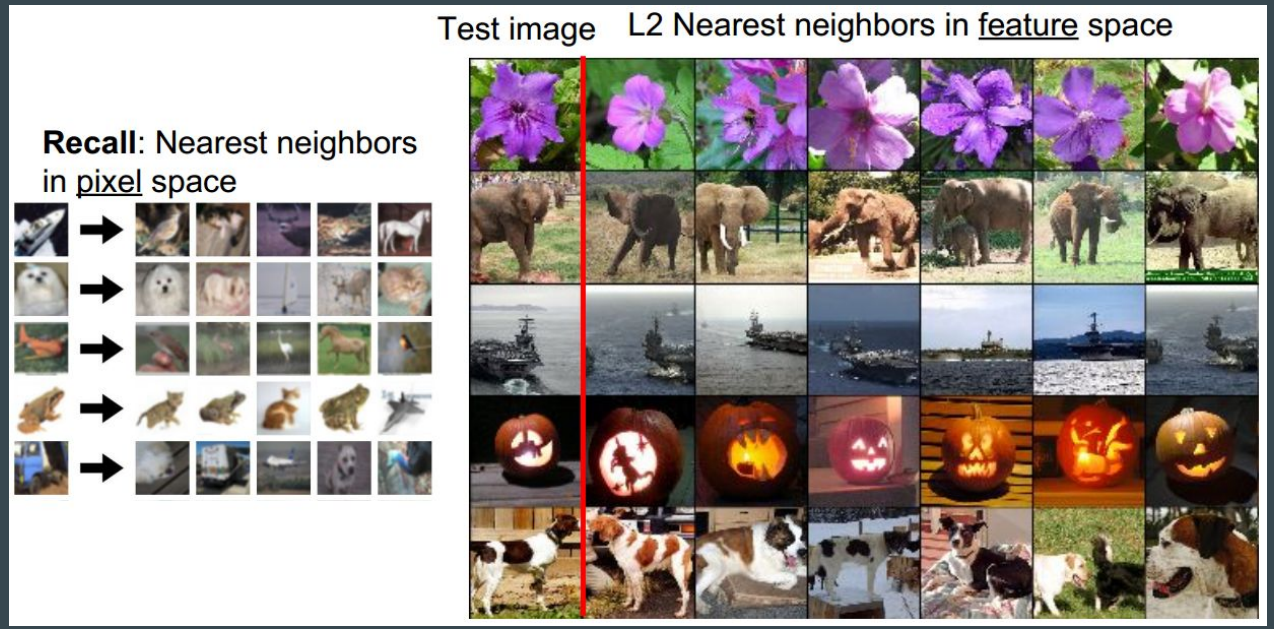
DNN Interpretation

First Layer: Visualize Filters

Krizhevsky, “One weird trick for parallelizing convolutional neural networks”, arXiv 2014
He et al, “Deep Residual Learning for Image Recognition”, CVPR 2016
Huang et al, “Densely Connected Convolutional Networks”, CVPR 2017

DNN Interpretation

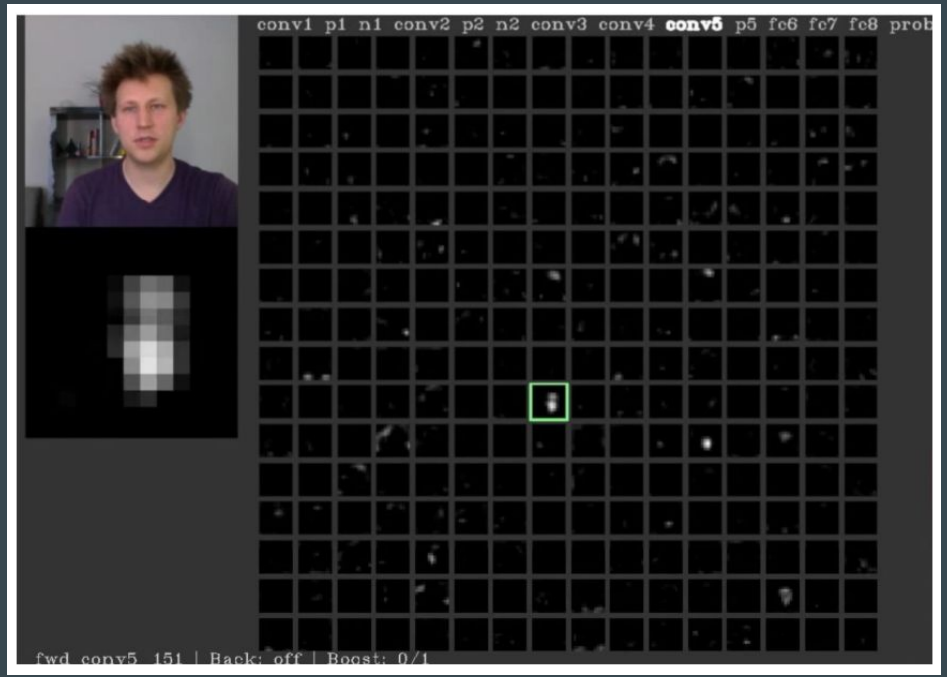
Last Layer: Nearest Neighbors



Krizhevsky et al, “ImageNet Classification with Deep Convolutional Neural Networks”, NIPS 2012

DNN Interpretation

conv5 feature map is 128x13x13; visualize as 128 13x13 grayscale images

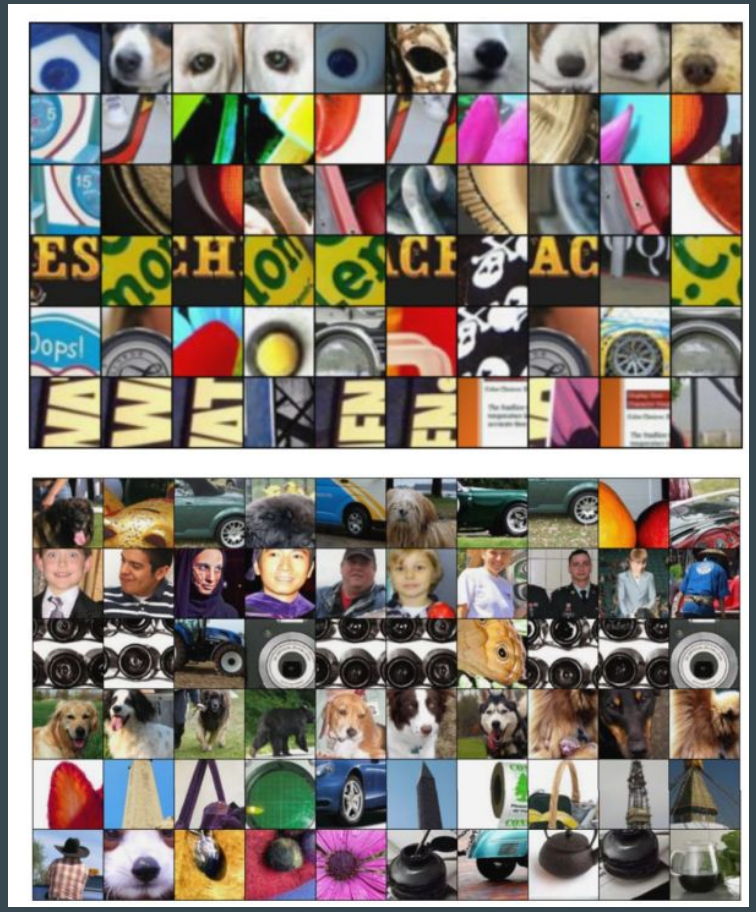


Yosinski et al, “Understanding Neural Networks Through Deep Visualization”, ICML DL Workshop 2014.

DNN Interpretation

Maximally Activating Patches

- Pick a layer and a channel
- Run many images through the network, record values of chosen channel
- Visualize image patches that correspond to maximal activations



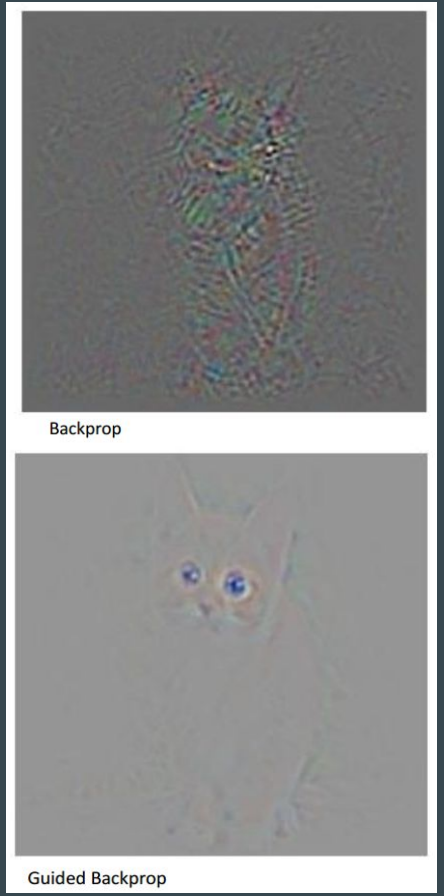
Springenberg et al, “Striving for Simplicity: The All Convolutional Net”, ICLR Workshop 2015

DNN Interpretation

Guided Backpropagation

Idea: neurons act like detectors of particular image features.

- We are only interested in what image features the neuron detects, not in what kind of stuff it doesn't detect.
- So when propagating the gradient, we set all the negative gradients to 0.



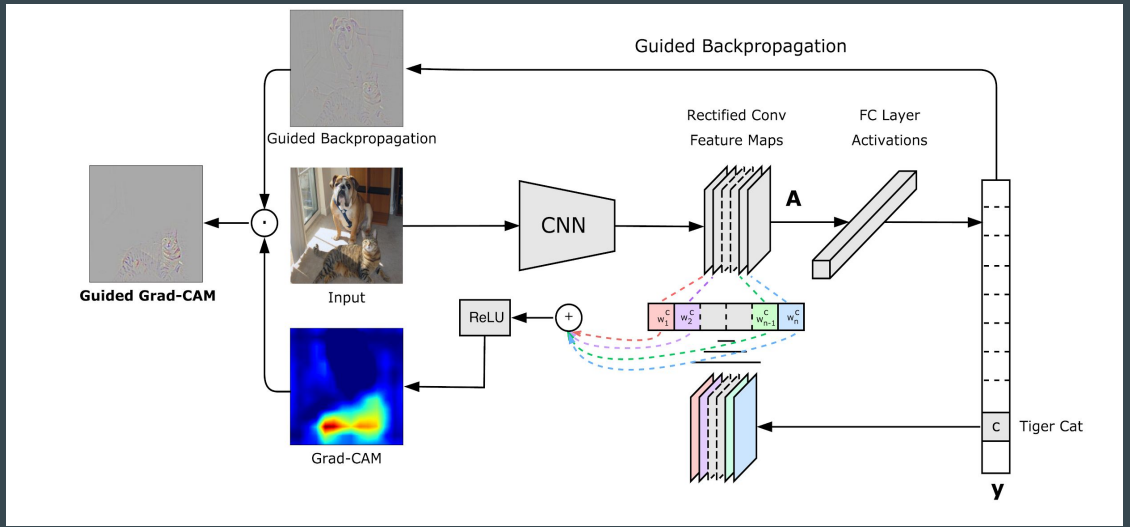
Springerberg et al, Striving for Simplicity: The All Convolutional Net (ICLR 2015 workshops)

DNN Interpretation

Grad-CAM
(Gradient weighted Class Activation Mapping)

Take the **final convolutional feature map** and then **weigh** every channel in that feature with the **gradient of the class** with respect to the **channel**.

It's just nothing but how intensely the input image **activates** different channels by how important each channel is with regard to the class.



Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." *IEEE Conference on Computer Vision*. 2017.

DNN Interpretation

Perturbation Methods:

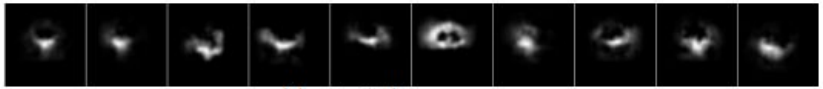
Perturbation-based methods directly compute the attribution of an input feature (or set of features) by removing, masking or altering them, and running a forward pass on the new input, measuring the difference with the original output.

DNN Interpretation

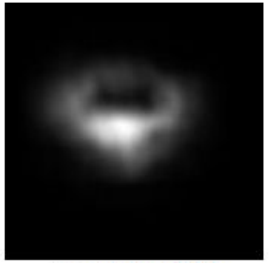
Occlusion Experiments and Saliency Maps

sliding-window stimuli

discrepancy maps for top 10 images



calibrated discrepancy maps



receptive field

Figure 3: The pipeline for estimating the RF of each unit. Each sliding-window stimuli contains a small randomized patch (example indicated by red arrow) at different spatial locations. By comparing the activation response of the sliding-window stimuli with the activation response of the original image, we obtain a discrepancy map for each image (middle top). By summing up the calibrated discrepancy maps (middle bottom) for the top ranked images, we obtain the actual RF of that unit (right).

Zhou, Bolei, et al. "Object detectors emerge in deep scene cnns." *arXiv preprint arXiv:1412.6856* (2014).

DNN Interpretation

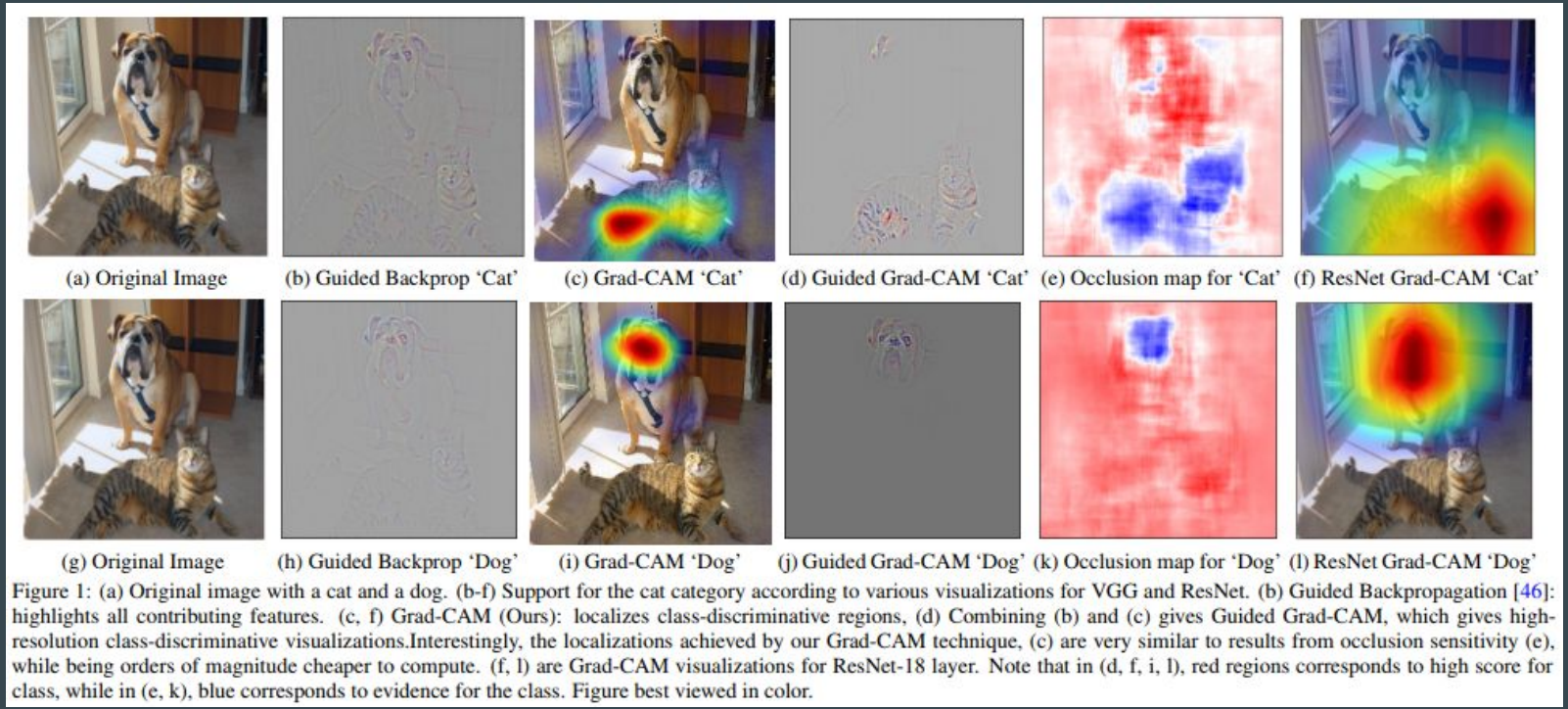


Figure 1: (a) Original image with a cat and a dog. (b-f) Support for the cat category according to various visualizations for VGG and ResNet. (b) Guided Backpropagation [46]: highlights all contributing features. (c, f) Grad-CAM (Ours): localizes class-discriminative regions, (d) Combining (b) and (c) gives Guided Grad-CAM, which gives high-resolution class-discriminative visualizations.Interestingly, the localizations achieved by our Grad-CAM technique, (c) are very similar to results from occlusion sensitivity (e), while being orders of magnitude cheaper to compute. (f, l) are Grad-CAM visualizations for ResNet-18 layer. Note that in (d, f, i, l), red regions corresponds to high score for class, while in (e, k), blue corresponds to evidence for the class. Figure best viewed in color.

From Grad-CAM Paper

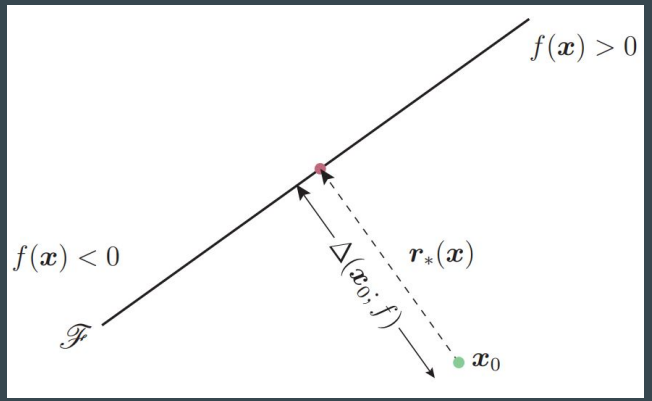
Adversarial Attack / Examples

1. Whitebox attacks: using gradient of the model
2. Black-box attacks: do not have access to the inside of the model

Adversarial Attack

For a given classifier, we define an adversarial perturbation as the minimal perturbation *r* that is sufficient to change the estimated label

$$\Delta(\boldsymbol{x}; \hat{k}) := \min_{\boldsymbol{r}} \|\boldsymbol{r}\|_2 \text{ subject to } \hat{k}(\boldsymbol{x} + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x})$$



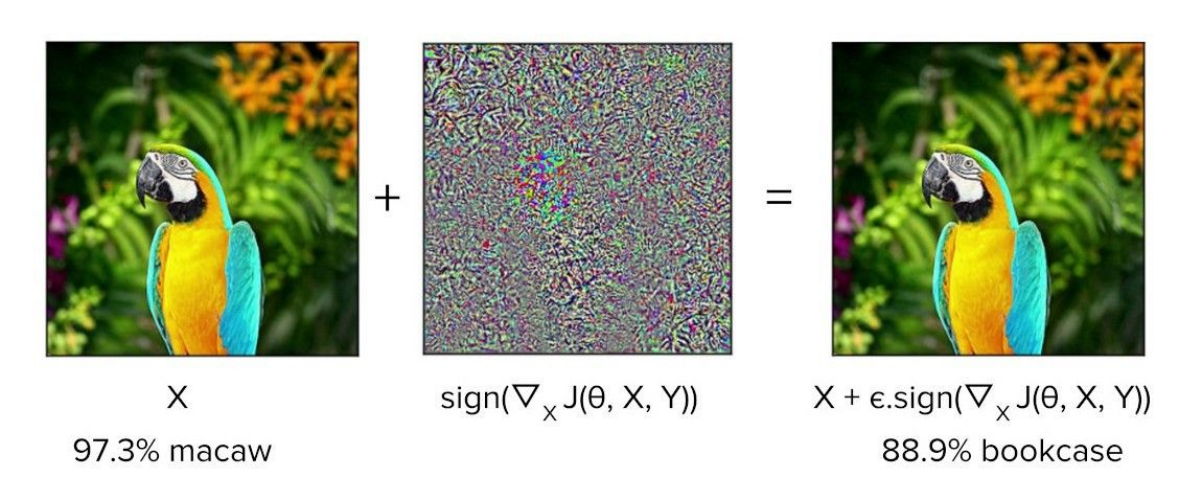
Algorithm 1 DeepFool for binary classifiers

input: Image $\boldsymbol{x}$, classifier f.
output: Perturbation $\hat{\boldsymbol{r}}$.
Initialize $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $i \leftarrow 0$.
while $\text{sign}(f(\boldsymbol{x}_i)) = \text{sign}(f(\boldsymbol{x}_0))$ **do**
    $\boldsymbol{r}_i \leftarrow -\frac{f(\boldsymbol{x}_i)}{\|\nabla f(\boldsymbol{x}_i)\|_2^2} \nabla f(\boldsymbol{x}_i)$,
    $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$,
    $i \leftarrow i + 1$.
end while
return $\hat{\boldsymbol{r}} = \sum_i \boldsymbol{r}_i$.

S. Moosavi-Dezfooli, A. Fawzi, P. Frossard, DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks, CVPR 2016.

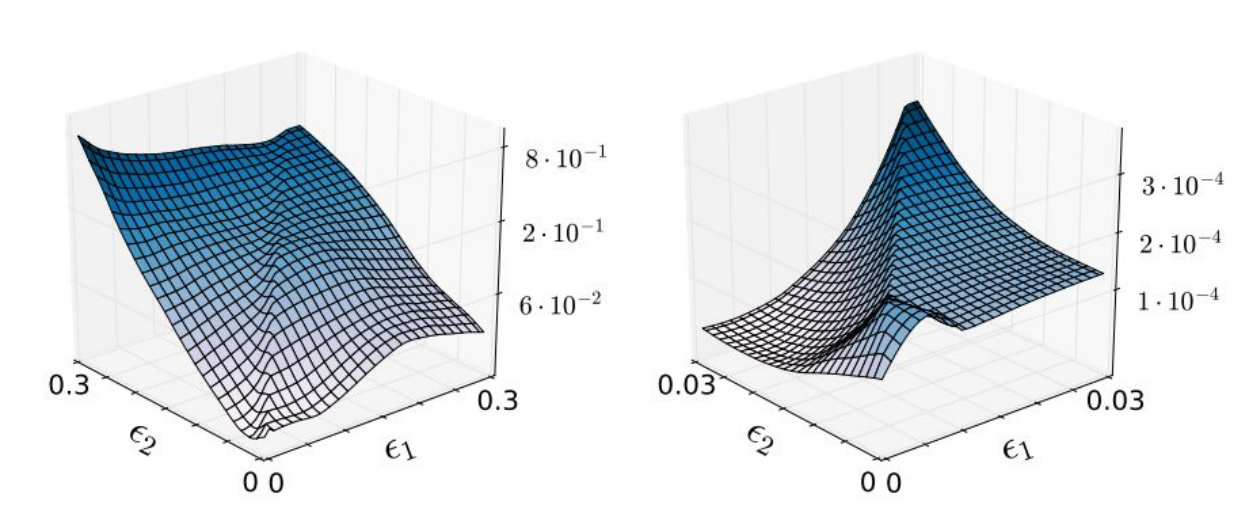
Adversarial Attack

Assuming linearity, Fast Gradient Sign Method (FGSM) can be used:
Let x be the original image, y the class of x, θ the weights of the network and J(θ, x, y) the loss function used to train the network.



Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." *arXiv:1412.6572* (2014).

Adversarial Attack

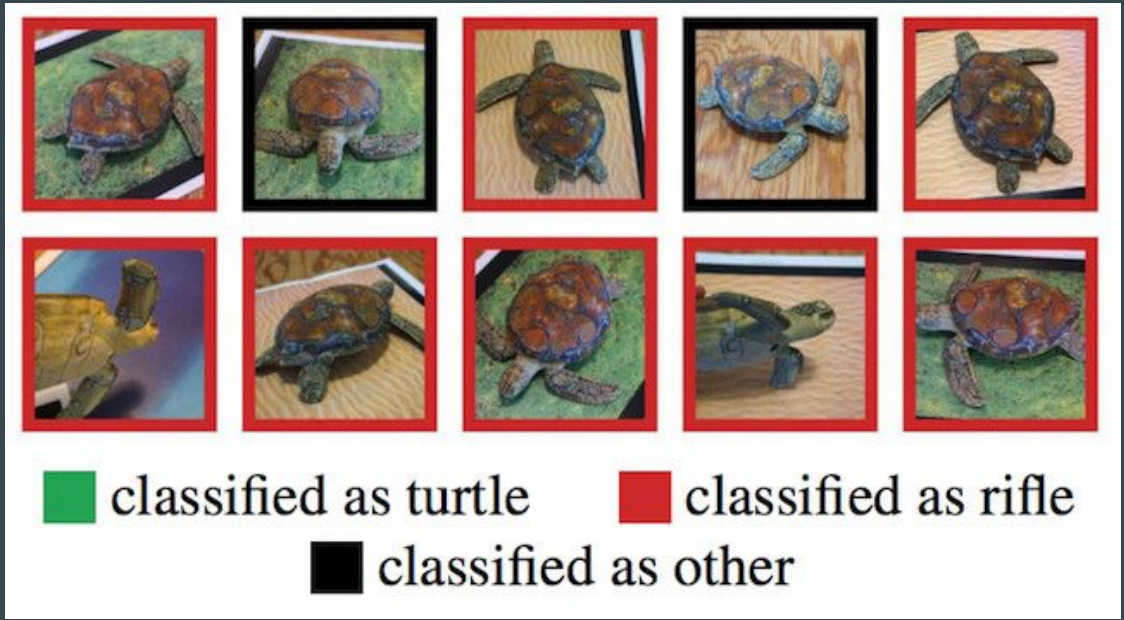


(a) Loss surface for model A_{adv} (log-scale). (b) Zoom in of the loss for small ϵ_1, ϵ_2.

Figure 5: **Illustrations of the local curvature artifacts introduced by adversarial training on MNIST.** We plot the loss of model A_{adv} on samples of the form $x^* = x + \epsilon_1 \cdot g + \epsilon_2 \cdot g^{\perp}$, where g is the signed gradient of model A_{adv} and $g^{\perp}$ is an orthogonal adversarial direction, obtained from model B. The right-side plots are zoomed in versions of the left-side plots.

Tramèr, Florian, et al. "Ensemble adversarial training: Attacks and defenses." *arXiv preprint arXiv:1705.07204* (2017).

Adversarial Attack



A 3D-printed turtle that is recognized as a rifle by Tensorflow's standard pre-trained InceptionV3 classifier.

Athalye et. al, Synthesizing Robust Adversarial Examples, 2017.

Adversarial Attack

Black Box attack using surrogate model:

1. Start with a few images that come from the same domain as the training data, e.g. if the classifier to be attacked is a digit classifier, use images of digits. The knowledge of the domain is required, but not the access to the training data.
2. Get predictions for the current set of images from the black box.
3. Train a surrogate model on the current set of images (for example a neural network).
4. Create adversarial examples for the surrogate model using the fast gradient method (or similar).
5. Attack the original model with adversarial examples.

The aim of the surrogate model is to approximate the decision boundaries of the black box model, but not necessarily to achieve the same accuracy.

Papernot, Nicolas, et al. “Practical black-box attacks against machine learning.” ACM 2017.

Adversarial Training

Adversarial training can be considered as a variant of standard Empirical Risk Minimization (ERM), where our aim is to minimize the risk over adversarial examples:

$$h^* = \arg\min_{h \in \mathcal{H}} \mathbb{E}_{(x, y_{\text{true}}) \sim \mathcal{D}} \left[\max_{\|x^{\text{adv}} - x\|_\infty \leq \epsilon} L(h(x^{\text{adv}}), y_{\text{true}}) \right]$$

for some target model $h \in H$ and inputs $(x; y_{true})$. This can be simplified to

$$h^* = \arg\min_{h \in \mathcal{H}} \mathbb{E}_{(x, y_{\text{true}}) \sim \mathcal{D}} \left[L(h(x^{\text{adv}}_{\text{FGSM}}), y_{\text{true}}) \right]$$

Ensemble Adversarial Training, augments a model's training data with adversarial examples crafted on other static pre-trained models.

Motivation

There are two possible hypotheses to answer the following questions “what are the representations of adversarial examples, and why do the representations lead to inaccurate predictions?”

- The representations of adversarial examples align well with the semantic objects/parts, but are not discriminative enough, resulting in erroneous predictions.
- The representations of adversarial examples do not align with the semantic objects/parts, which means by adding small imperceptible noises, the neurons cannot detect corresponding objects/parts in adversarial images, leading to inaccurate predictions.

Overview of the Methodology

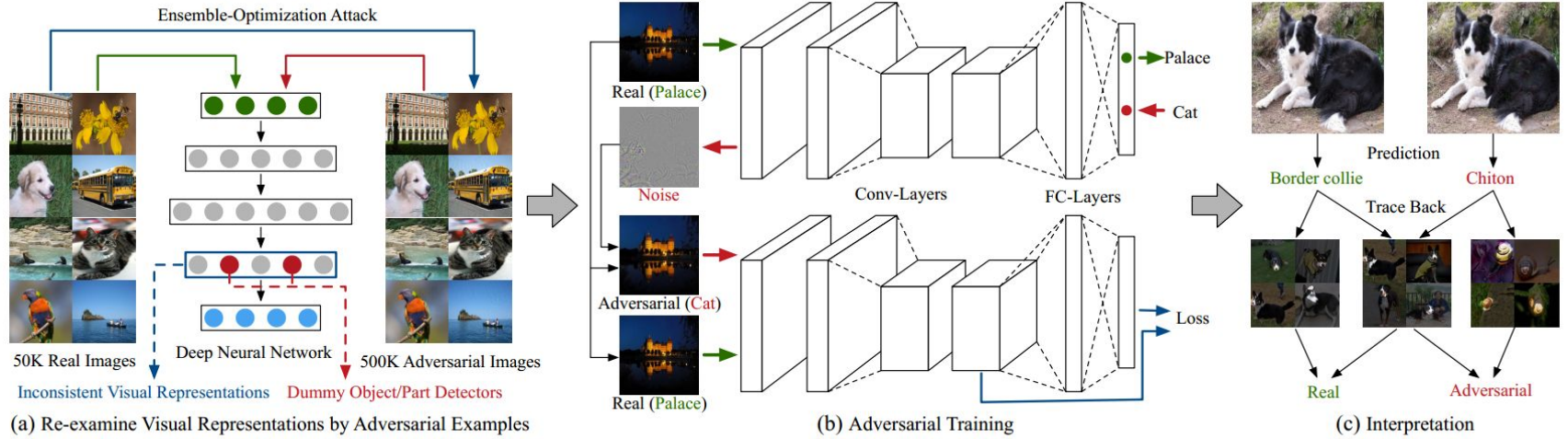


Figure 1. The overall framework in this paper. (a) We first generate a set of 500K adversarial images by the ensemble-optimization attack and re-examine the visual representations with the conclusions of dummy object/part detectors and inconsistent visual representations. (b) We show that adversarial training facilitates the improvement of the interpretability and consistency of representations in DNNs. (c) The interpretable representations enable us to trace the eventual predictions back to influential neurons, and explain how the predictions have been made as well as when and why an error occurs.

Inconsistency of representations

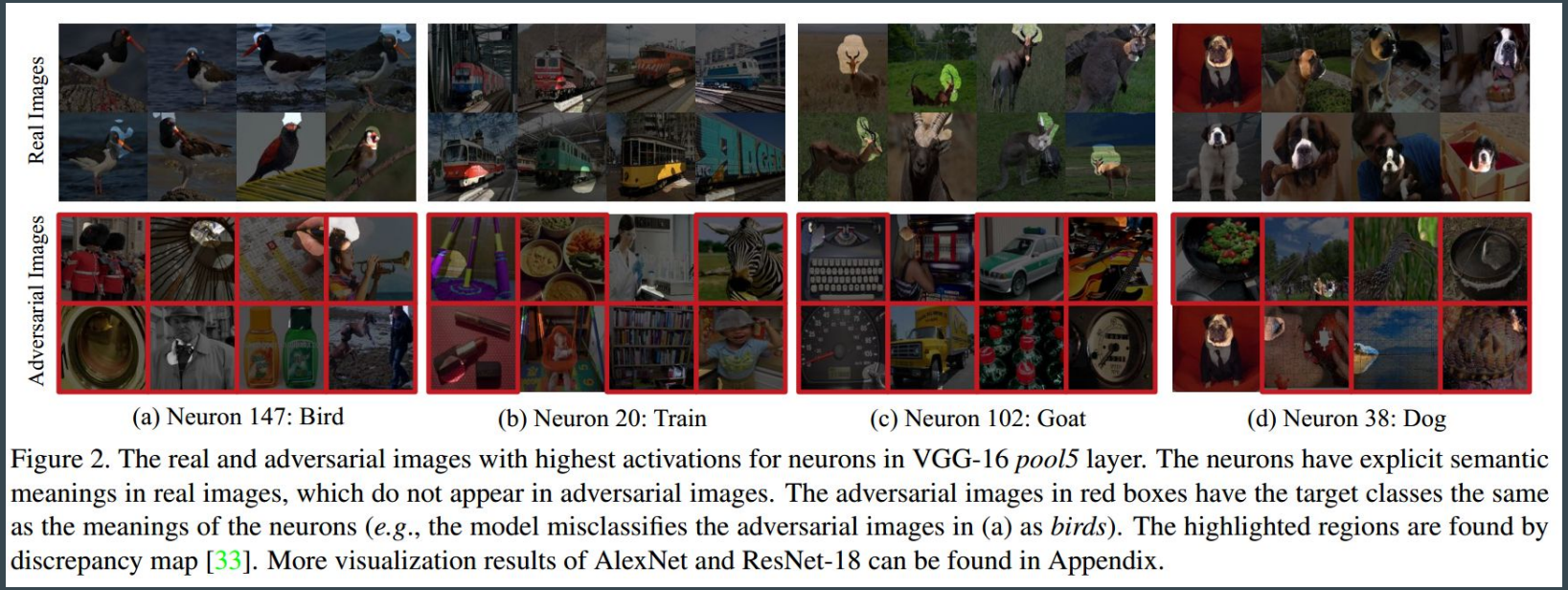


Figure 2. The real and adversarial images with highest activations for neurons in VGG-16 *pool5* layer. The neurons have explicit semantic meanings in real images, which do not appear in adversarial images. The adversarial images in red boxes have the target classes the same as the meanings of the neurons (*e.g.*, the model misclassifies the adversarial images in (a) as *birds*). The highlighted regions are found by discrepancy map [33]. More visualization results of AlexNet and ResNet-18 can be found in Appendix.

The contents of the adversarial images do not align with the semantic meanings of the corresponding neurons, if we look at the second row. In general, the neurons tend to detect nothing in common for adversarial samples.

Consistency metric

- We first define the consistency level between a neuron n and a concept c:

$$consis(n, c) = Pr(x\, contains\, c | x\, activates\, n)$$

where Pr is the probability measure of image space. The consistency metric above is related to a certain concept.

- We define the correlation between the corresponding concepts as

$$a_{i,j} = \exp\left(-\frac{d^2(w_i, w_j)}{2\sigma^2}\right)$$

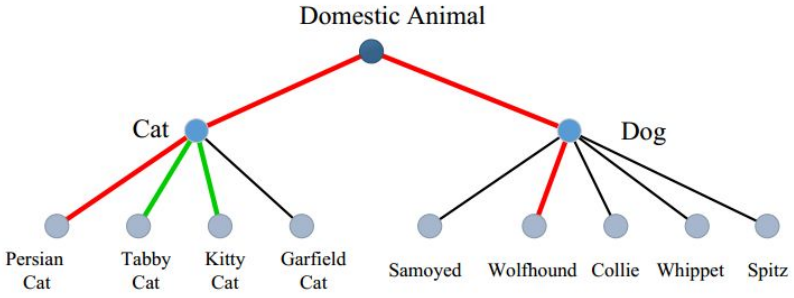


Figure 2: Illustration for quantifying the distance of different classes based on WordNet. The red path indicates the distance between *persian cat* and *wolfhound* $d = 4$, which is longer than the distance between *tabby cat* and *kitty cat* $(d = 2)$.

Consistency metric

We further collect $p_i = consis(n; c_i)$ for all concepts ci into a vector p and the consistency level of a neuron n is quantified as following

$$consis(n) = \|\boldsymbol{p}\|_{\mathbf{A}}^2 = \boldsymbol{p}^T \mathbf{A} \boldsymbol{p}$$

Similarly, we also measure the consistency level constrained on the adversarial samples, by replacing Pr with Pr_{adv}, where

$$Pr_{adv}(\cdot) = Pr(\cdot \mid x \text{ is an adversarial sample}).$$

Adversarial Training with a Consistent Loss

If the network is represented by θ, the following objective is used to train the network

$$\min_{\theta} L(\theta), \quad L(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\max_{x^* \in \mathcal{S}(x)} \ell(\theta, x^*, y) + \max_{x' \in \mathcal{S}(x)} d(\phi_\theta(x), \phi_\theta(x'))\Big]$$

Φ_θ is the feature representation (last convolutional layer of the model is used here), x' and x^* are two adversarial examples from the set of all possible samples $S(x)$, d is a distance metric (L_2 distance is used here), ℓ is the cross entropy loss.

Adversarial Training with a Consistent Loss

The first inner maximization is the definition of adversarial attack. So, it is replaced by FGSM and only one adversarial example is used:

$$\min_{\theta} L(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\alpha\ell(\theta, x, y) + (1-\alpha)\ell(\theta, x^*, y) + \beta d(\phi_\theta(x), \phi_\theta(x^*))\right]$$

$$\boldsymbol{x}_t^* = \text{clip}(\boldsymbol{x}_{t-1}^* - \epsilon \cdot \text{sign}\nabla_{\boldsymbol{x}}\ell(\mathbf{1}_{y^*}, f_\theta(\boldsymbol{x}_{t-1}^*)))$$

$\alpha = 0.5$ and $\beta = 0.1$, clipping keeps the values in [0, 255].

Results and Discussion

Break

Experiments

- For each neuron, top 1% images with highest activations in the real and adversarial validation sets are found (i.e., 500 real images and 5000 adversarial images) to represent the learned features of that neuron.
- The ensemble of AlexNet, VGG-16 and ResNet-18 models are used for adversarial generation.
- Adam optimizer with step size 5 for 10 ~ 20 iterations is used for generating adversarial examples.
- 10 adversarial images are generated for each image in the **ILSVRC 2012 and Broden** datasets, with the 10 least likely classes (least average probability), i.e., 10 different y^*, as targets.
- So a set of 500K adversarial images are generated.

Broadly and Densely Labeled Dataset (Broden) unifies several densely labeled image data sets: ADE, OpenSurfaces, Pascal-Context, Pascal-Part, and the Describable Textures Dataset, containing examples of a broad range of objects, scenes, object parts, textures, and materials in a variety of contexts.
Most examples are segmented down to the pixel level except textures and scenes which are given for full-images. In addition, every image pixel in the data set is annotated with one of the eleven common color names according to the human perceptions classified by van de Weijer.

`http://netdissect.csail.mit.edu/data/broden1_224.zip`

Experiments

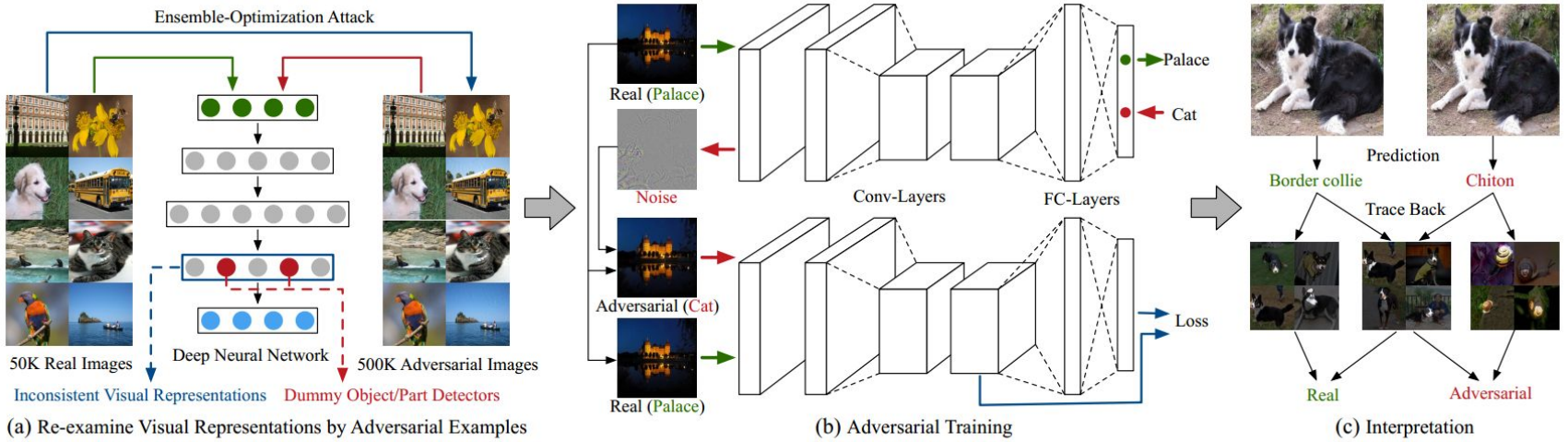
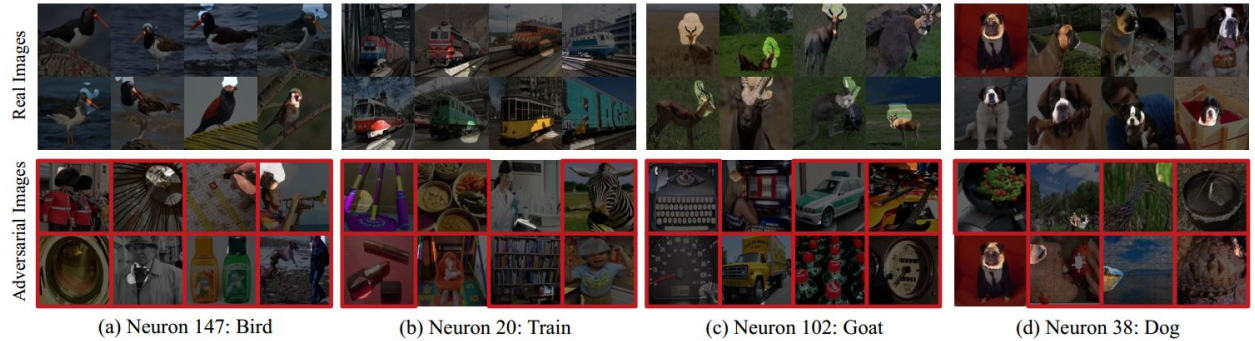


Figure 1. The overall framework in this paper. (a) We first generate a set of 500K adversarial images by the ensemble-optimization attack and re-examine the visual representations with the conclusions of dummy object/part detectors and inconsistent visual representations. (b) We show that adversarial training facilitates the improvement of the interpretability and consistency of representations in DNNs. (c) The interpretable representations enable us to trace the eventual predictions back to influential neurons, and explain how the predictions have been made as well as when and why an error occurs.

Results



(a) Neuron 147: Bird (b) Neuron 20: Train (c) Neuron 102: Goat (d) Neuron 38: Dog

Figure 3: The real and adversarial images with highest activations for neurons in VGG-16 *conv5_3* layer.

(a) Neuron 38: Dog (b) Neuron 20: Rail (c) Neuron 12: Tower (d) Neuron 135: Bird

Figure 4: The real and adversarial images with highest activations for neurons in VGG-16-Adv *conv5_3* layer.

Results

Table 1: The average of $consis(n)$ and $consis_{adv}(n)$ of neurons in the last convolutional layer in each network. The $consis(n)$ is calculated on the whole image space and the $consis_{adv}(n)$ is calculated on the adversarial image subset.

	$consis(n)$	$consis_{adv}(n)$
AlexNet	0.0162	0.0116
AlexNet-Adv	0.0161	0.0121
VGG-16	0.0296	0.0187
VGG-16-Adv	0.0287	0.0220
ResNet-18	0.0261	0.0133
ResNet-18-Adv	0.0251	0.0206

Table 3: Accuracy (%) on the ImageNet validation set and adversarial examples generated by FGSM with $\epsilon = 1$.

		AlexNet	VGG16	ResNet18
Real	Top-1	54.58	68.15	66.30
	Top-5	78.17	88.30	87.09
Adv.	Top-1	4.44	8.60	4.41
	Top-5	22.94	36.94	31.8
		AlexNet-Adv	VGG16-Adv	ResNet18-Adv
Real	Top-1	43.92	62.55	54.01
	Top-5	68.80	84.66	77.84
Adv.	Top-1	17.45	25.62	27.56
	Top-5	38.12	56.17	55.61

Results

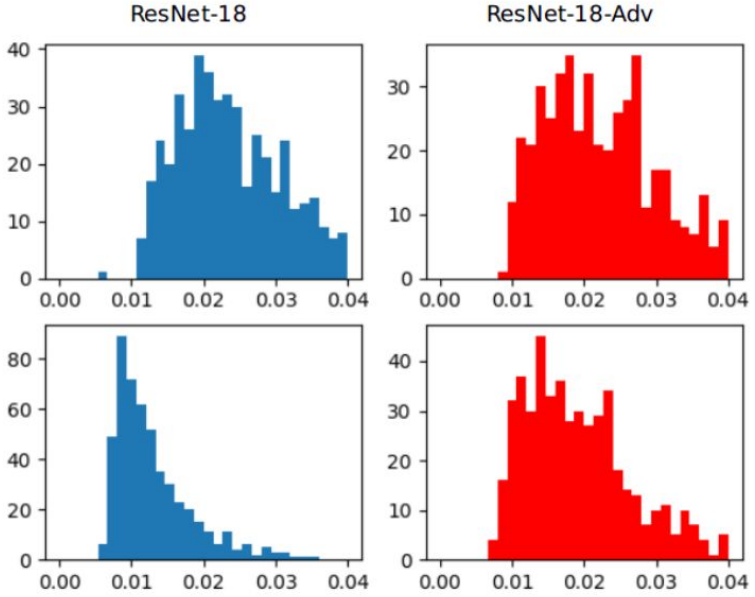


Figure 5: The distribution of $consis(n)$ and $consis_{adv}(n)$ of neurons in the last convolutional layer of ResNet-18 and ResNet-18-Adv. The left two histograms corresponds to ResNet-18 and the right two histograms corresponds to ResNet-18-Adv. The top two histograms corresponds to $consis(n)$ and the bottom two histograms corresponds to $consis_{adv}(n)$.

Discussion Points

1. Does it generalize to other interpretability methods? e.g. Grad-CAM claims to be robust to adversarial examples.
2. Application of FGSM in the adversarial training: Motivation of Ensemble Adversarial Learning.
3. The simplified adversarial training equation is not a surrogate of the original proposed equation: FGSM is the gradient of the loss function. The feature map distance won't be maximized: Jacobian-based Saliency Map Attack does this.
4. The results are not compared to any other adversarial training.

5. Phase vs. Amplitude in the adversarial examples.