

Deep Learning for Computer Vision

# Adversarial Robustness

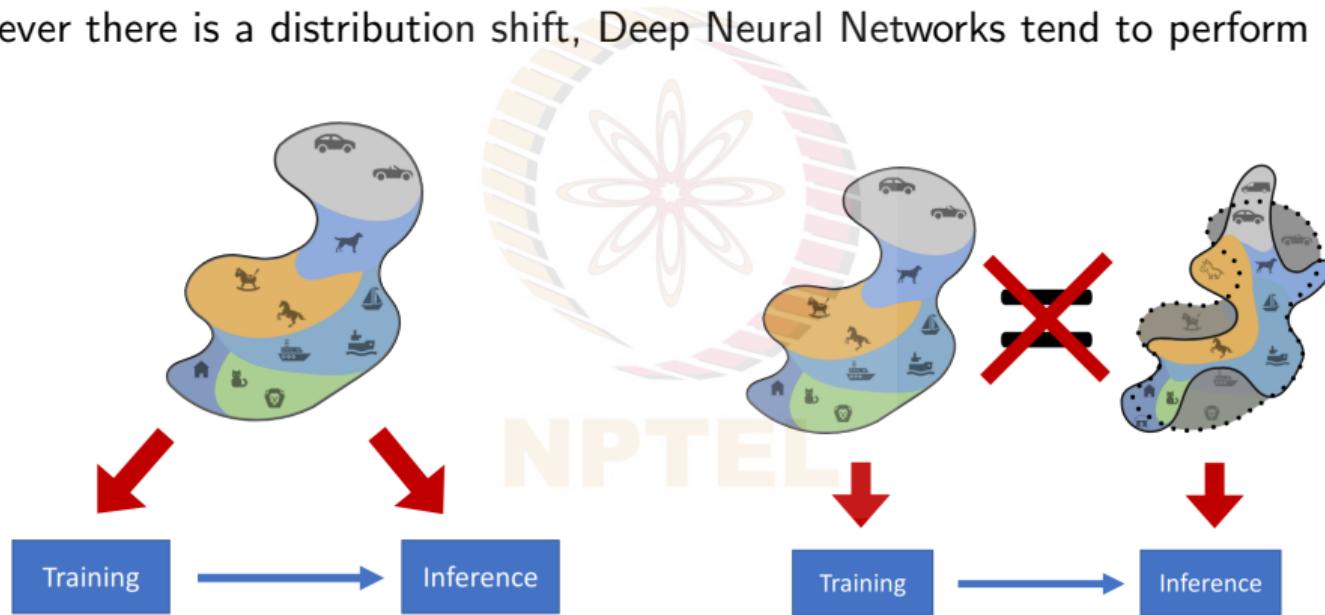
Vineeth N Balasubramanian

Department of Computer Science and Engineering  
Indian Institute of Technology, Hyderabad



# Limitation of Supervised Machine Learning

- In reality, the distributions we use ML on are often **NOT** the ones we train it on
- Whenever there is a distribution shift, Deep Neural Networks tend to perform poorly



Credit: <https://adversarial-ml-tutorial.org/>

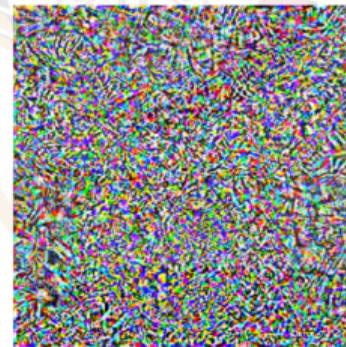
# Adversarial Examples

- Examples indistinguishable from original input which leads to wrong predictions
- $\delta^* = \arg \inf_{\delta} \|\delta\|_p$  such that  $f(x + \delta) \neq f(x)$



“pig” (91%)

+ 0.005 x



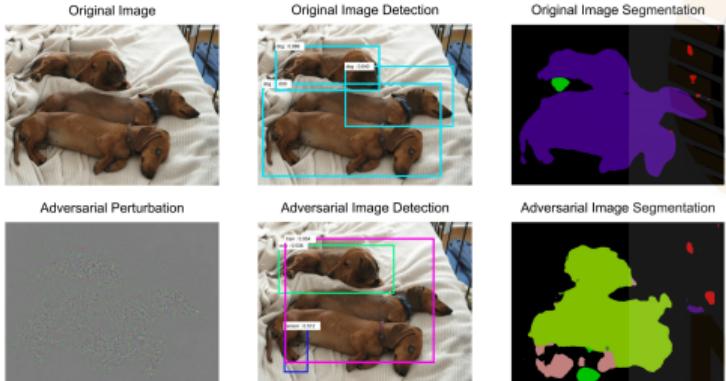
noise (NOT random)

=



“airliner” (99%)

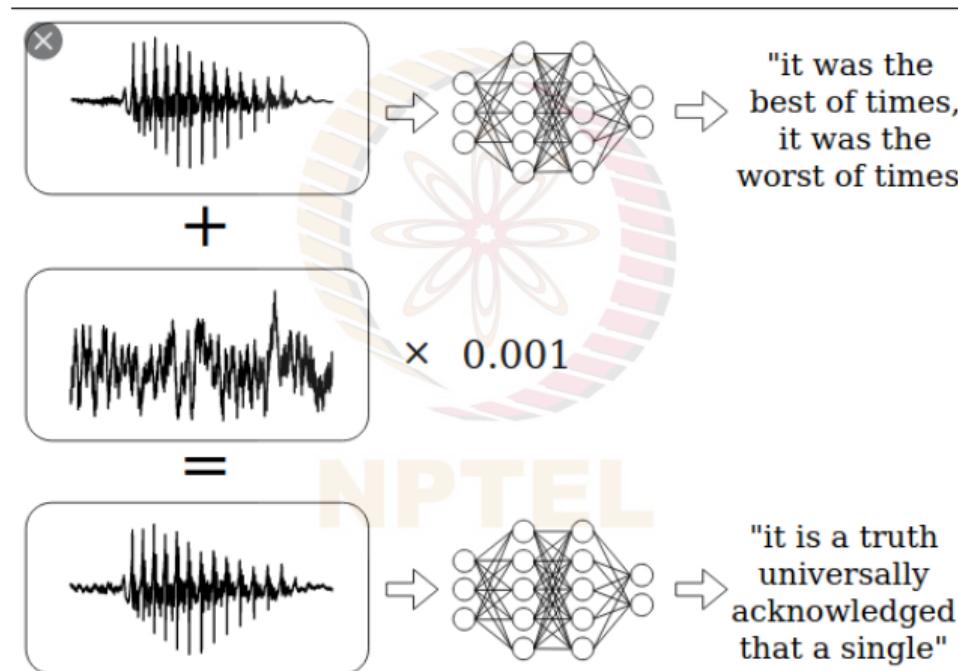
# Supervised Frameworks are Vulnerable to Adversarial Attacks<sup>1</sup>



Original Input	Connoisseurs of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: <b>Positive (77%)</b>
Adversarial example [Visually similar]	<b>Amonniseurs</b> of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: <b>Negative (52%)</b>
Adversarial example [Semantically similar]	Connoisseurs of Chinese <b>footage</b> will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: <b>Negative (54%)</b>

<sup>1</sup>Xie et al, Adversarial Examples for Semantic Segmentation and Object Detection, ICCV 2017

# Supervised Frameworks are Vulnerable to Adversarial Attacks<sup>2</sup>

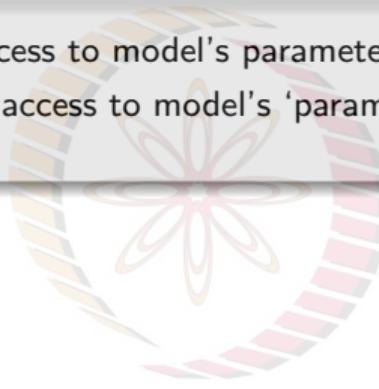


<sup>2</sup> Carlini et al, Audio Adversarial Examples: Targeted Attacks on Speech-to-Text, 2018

# Taxonomy of Adversarial Attacks

## On basis of Threat Model

- **White Box Attacks:** attacker has access to model's parameters.
- **Black Box Attacks:** attacker has no access to model's 'parameters, i.e., it a different model or no model at all to generate adversarial images



# Taxonomy of Adversarial Attacks

## On basis of Threat Model

- **White Box Attacks:** attacker has access to model's parameters.
- **Black Box Attacks:** attacker has no access to model's 'parameters, i.e., it a different model or no model at all to generate adversarial images

## On basis of Objective

- **Untargeted Attacks:** aim is to enforce the model to misclassify adversarial image
- **Targeted Attacks:** aim is to get the image classified as a specific target class, different from the true class

NPTEL

# Taxonomy of Adversarial Attacks

## On basis of Threat Model

- **White Box Attacks:** attacker has access to model's parameters.
- **Black Box Attacks:** attacker has no access to model's 'parameters, i.e., it a different model or no model at all to generate adversarial images

## On basis of Objective

- **Untargeted Attacks:** aim is to enforce the model to misclassify adversarial image
- **Targeted Attacks:** aim is to get the image classified as a specific target class, different from the true class

## On basis of Distance Metrics

- $L_0$ : total number of pixels that differ between clean and adversarial images
- $L_2$ : squared difference between pixel values of clean and adversarial images
- $L_\infty$ : maximum pixel difference between clean and adversarial images

# White-box Adversarial Attacks<sup>3</sup>

## Fast Gradient Sign Method (FGSM)

- Computes an adversarial image by adding a pixel-wise perturbation of magnitude in the direction of gradient



---

<sup>3</sup>Goodfellow et al, Explaining and Harnessing Adversarial Examples, ICLR 2015

# White-box Adversarial Attacks<sup>3</sup>

## Fast Gradient Sign Method (FGSM)

- Computes an adversarial image by adding a pixel-wise perturbation of magnitude in the direction of gradient
- **Single step method:** very efficient in terms of computation time:

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(x, y_{true}))$$

NPTEL

<sup>3</sup>Goodfellow et al, Explaining and Harnessing Adversarial Examples, ICLR 2015

# White-box Adversarial Attacks<sup>3</sup>

## Fast Gradient Sign Method (FGSM)

- Computes an adversarial image by adding a pixel-wise perturbation of magnitude in the direction of gradient
- **Single step method:** very efficient in terms of computation time:

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(x, y_{true}))$$

- In case of **targeted attack**, direction is negative gradient with respect to target class:

$$x_{adv} = x - \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(x, y_{target}))$$

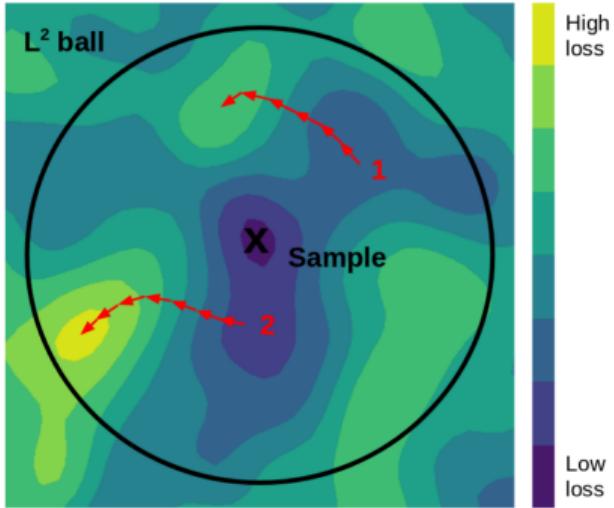
where  $x$  is clean input image,  $x_{adv}$  is corresponding adversarial image,  $\mathcal{L}$  is classification loss,  $y_{true}$  is actual label,  $y_{target}$  is target label and  $\epsilon$  is  $L_\infty$  budget

<sup>3</sup>Goodfellow et al, Explaining and Harnessing Adversarial Examples, ICLR 2015

# White-box Adversarial Attacks<sup>4</sup>

## Projected Gradient Descent (PGD)

- “Complete” method as it does not consider constraints on amount of time and effort



$$x_{adv}^0 = x$$

$$x_{adv}^{t+1} = \text{Proj}\{x_{adv}^t + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(x_{adv}^t, y_{true})\}$$

where  $\text{Proj}\{\cdot\}$  projects the updated adversarial sample into the  $\epsilon$  neighborhood and a valid range

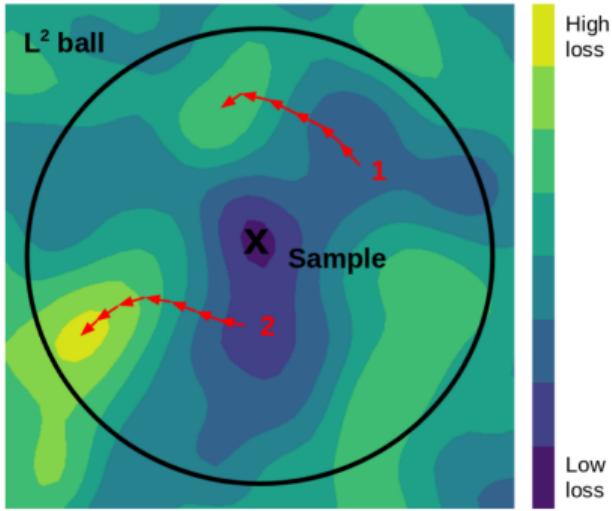
NPTEL

Credit: [Oscar Knagg, TowardsDataScience](#)

<sup>4</sup>Towards Deep Learning Models Resistant To Adversarial Attack — Madry et al 2017

# White-box Adversarial Attacks<sup>4</sup>

## Projected Gradient Descent (PGD)



- “Complete” method as it does not consider constraints on amount of time and effort

$$x_{adv}^0 = x$$

$$x_{adv}^{t+1} = \text{Proj}\{x_{adv}^t + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(x_{adv}^t, y_{true}))\}$$

where  $\text{Proj}\{\cdot\}$  projects the updated adversarial sample into the  $\epsilon$  neighborhood and a valid range

- In case of  $L_2$ , update is as follows:

$$x_{adv}^{t+1} = \text{Proj}\{x_{adv}^t + \epsilon \cdot \frac{\nabla_x \mathcal{L}(x_{adv}^t, y_{true})}{\|\nabla_x \mathcal{L}(x_{adv}^t, y_{true})\|_2}\}$$

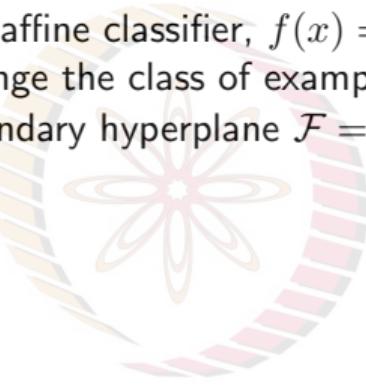
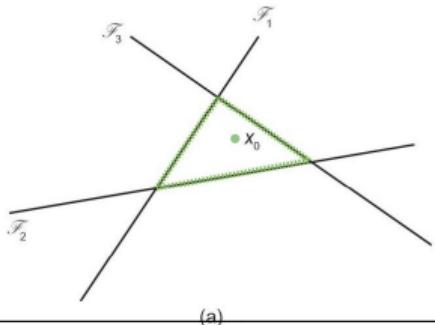
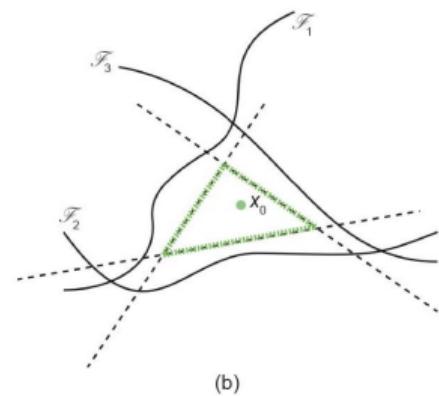
Credit: [Oscar Knagg, TowardsDataScience](#)

<sup>4</sup>Towards Deep Learning Models Resistant To Adversarial Attack — Madry et al 2017

# White-box Adversarial Attacks<sup>5</sup>

## DeepFool

- For affine classifier,  $f(x) = w^T x + b$ , minimum perturbation to change the class of example  $x_0$  is distance from the decision boundary hyperplane  $\mathcal{F} = \{x : w^T x + b = 0\}$ , i.e.,  $\frac{-f(x_0)}{\|w\|_2^2} \cdot w$



NPTEL

<sup>5</sup>Seyed-Mohsen et al, DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks, CVPR 2016

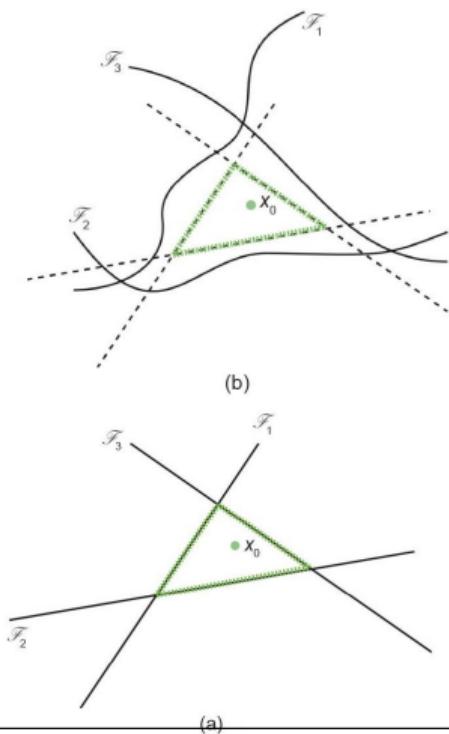
# White-box Adversarial Attacks<sup>5</sup>

## DeepFool

- For affine classifier,  $f(x) = w^T x + b$ , minimum perturbation to change the class of example  $x_0$  is distance from the decision boundary hyperplane  $\mathcal{F} = \{x : w^T x + b = 0\}$ , i.e.,  $\frac{-f(x_0)}{\|w\|_2^2} \cdot w$
- For a general differentiable classifier, assuming  $f$  is linear around  $x'_t$ , iteratively compute perturbation  $\delta_t$ :

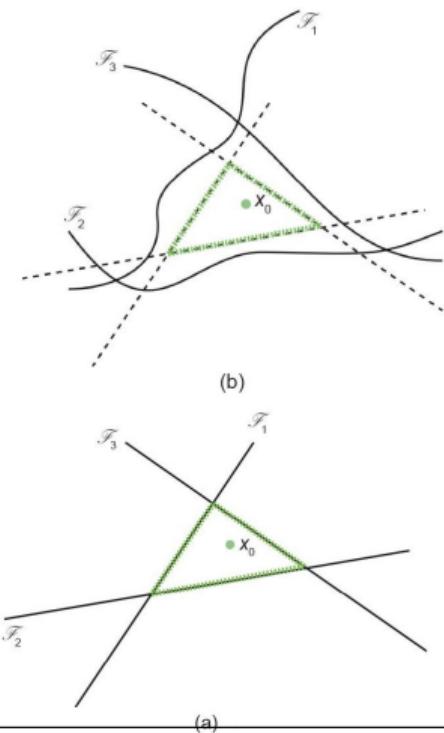
$$\arg \min_{\delta_t} \|\delta\|_2 \text{ subject to } f(x'_t) + \nabla_x f(x'_t)^T \delta_t = 0$$

Runs until  $f(x'_t) \neq f(x_0)$



<sup>5</sup>Seyed-Mohsen et al, DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks, CVPR 2016

# White-box Adversarial Attacks<sup>5</sup>



## DeepFool

- For affine classifier,  $f(x) = w^T x + b$ , minimum perturbation to change the class of example  $x_0$  is distance from the decision boundary hyperplane  $\mathcal{F} = \{x : w^T x + b = 0\}$ , i.e.,  $\frac{-f(x_0)}{\|w\|_2^2} \cdot w$
- For a general differentiable classifier, assuming  $f$  is linear around  $x'_t$ , iteratively compute perturbation  $\delta_t$ :

$$\arg \min_{\delta_t} \|\delta\|_2 \text{ subject to } f(x'_t) + \nabla_x f(x'_t)^T \delta_t = 0$$

Runs until  $f(x'_t) \neq f(x_0)$

- Multi-class classifiers:** Compute distance from  $x_0$  to surface of a convex polyhedron formed by decision boundaries between all classes

<sup>5</sup>Seyed-Mohsen et al, DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks, CVPR 2016

# White-box Adversarial Attacks<sup>6</sup>

## Carlini and Wagner (C&W) attack

- Optimization-based adversarial attack that can generate adversarial samples using:

$$\min_{\delta} D(x, x + \delta) \text{ such that } f(x + \delta) = t$$

subject to  $x + \delta \in [0, 1]$ ;  $D(., .)$  is  $L_0$ ,  $L_2$  and  $L_\infty$  distance measures



---

<sup>6</sup>Carlini and Wagner, Towards Evaluating the Robustness of Neural Networks, arXiv 2016

# White-box Adversarial Attacks<sup>6</sup>

## Carlini and Wagner (C&W) attack

- Optimization-based adversarial attack that can generate adversarial samples using:

$$\min_{\delta} D(x, x + \delta) \text{ such that } f(x + \delta) = t$$

subject to  $x + \delta \in [0, 1]$ ;  $D(., .)$  is  $L_0$ ,  $L_2$  and  $L_\infty$  distance measures

- To ensure  $x + \delta$  yields a valid image (i.e.,  $x + \delta \in [0, 1]$ ), it introduces a new variable,  $\kappa$ , to substitute as follows:

$$\delta = \frac{1}{2}[\tanh(\kappa) + 1] - x$$

such that  $x + \delta = \frac{1}{2}[\tanh(\kappa) + 1]$  which always resides in the range of the optimization process

<sup>6</sup>Carlini and Wagner, Towards Evaluating the Robustness of Neural Networks, arXiv 2016

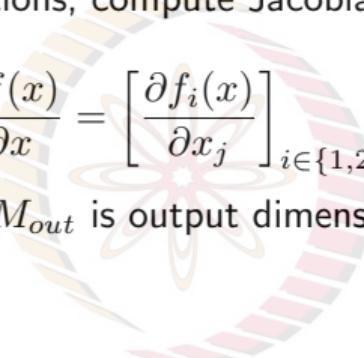
# White-box Adversarial Attacks<sup>7</sup>

## Jacobian-based Saliency Map Attack

- Fool DNNs with small  $L_0$  perturbations; compute Jacobian matrix of logits  $f(x)$  before softmax layer:

$$\nabla_x f(x) = \frac{\partial f(x)}{\partial x} = \left[ \frac{\partial f_i(x)}{\partial x_j} \right]_{i \in \{1, 2, \dots, M_{out}\}; j \in \{1, 2, \dots, M_{in}\}}$$

where  $M_{in}$  is input dimension and  $M_{out}$  is output dimension



<sup>7</sup>Papernot et al, The Limitations of Deep Learning in Adversarial Settings, EuroS&P 2016

# White-box Adversarial Attacks<sup>7</sup>

## Jacobian-based Saliency Map Attack

- Fool DNNs with small  $L_0$  perturbations; compute Jacobian matrix of logits  $f(x)$  before softmax layer:

$$\nabla_x f(x) = \frac{\partial f(x)}{\partial x} = \left[ \frac{\partial f_i(x)}{\partial x_j} \right]_{i \in \{1, 2, \dots, M_{out}\}; j \in \{1, 2, \dots, M_{in}\}}$$

where  $M_{in}$  is input dimension and  $M_{out}$  is output dimension

- Jacobian matrix - how input pixels affect logits; **adversarial saliency map**  $S(x, y_{true})$  - to select pixels that should be perturbed to obtain desired changes in logits

$$S(x_j, y_{true})[i] = \begin{cases} 0 & \text{if } \frac{\partial f_{y_{true}}(x)}{\partial x_j} \leq 0 \text{ or } \sum_{i \neq y_{true}} \frac{\partial f_i(x)}{\partial x_j} > 0 \\ \frac{\partial f_{y_{true}}(x)}{\partial x_j} \left| \sum_{i \neq y_{true}} \frac{\partial f_i(x)}{\partial x_j} \right| & \text{otherwise} \end{cases}$$

<sup>7</sup>Papernot et al, The Limitations of Deep Learning in Adversarial Settings, EuroS&P 2016

# White-box Adversarial Attacks<sup>7</sup>

## Jacobian-based Saliency Map Attack

- Fool DNNs with small  $L_0$  perturbations; compute Jacobian matrix of logits  $f(x)$  before softmax layer:

$$\nabla_x f(x) = \frac{\partial f(x)}{\partial x} = \left[ \frac{\partial f_i(x)}{\partial x_j} \right]_{i \in \{1, 2, \dots, M_{out}\}; j \in \{1, 2, \dots, M_{in}\}}$$

where  $M_{in}$  is input dimension and  $M_{out}$  is output dimension

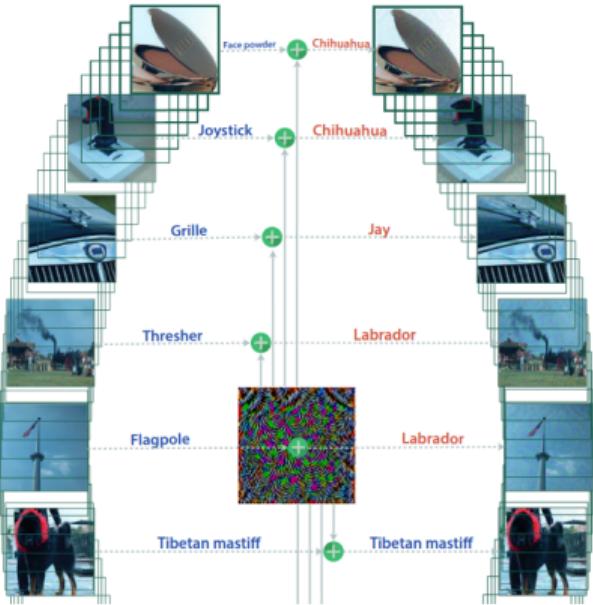
- Jacobian matrix - how input pixels affect logits; **adversarial saliency map**  $S(x, y_{true})$  - to select pixels that should be perturbed to obtain desired changes in logits

$$S(x_j, y_{true})[i] = \begin{cases} 0 & \text{if } \frac{\partial f_{y_{true}}(x)}{\partial x_j} \leq 0 \text{ or } \sum_{i \neq y_{true}} \frac{\partial f_i(x)}{\partial x_j} > 0 \\ \frac{\partial f_{y_{true}}(x)}{\partial x_j} \left| \sum_{i \neq y_{true}} \frac{\partial f_i(x)}{\partial x_j} \right| & \text{otherwise} \end{cases}$$

- Finally, perturb element with highest value of  $S(x, y_{true})[i]$  to increase/decrease logit outputs of target/other class significantly

<sup>7</sup>Papernot et al, The Limitations of Deep Learning in Adversarial Settings, EuroS&P 2016

# White-box Adversarial Attacks<sup>8</sup>



## Universal Adversarial Attack

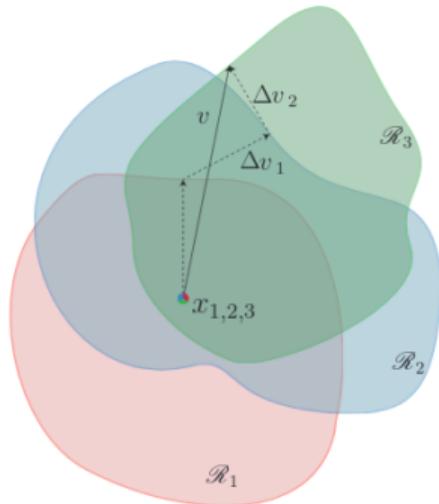
- One perturbation for all examples



NPTEL

<sup>8</sup>Moosavi-Dezfooli et al, Universal Adversarial Perturbations, CVPR 2017

# White-box Adversarial Attacks<sup>8</sup>



## Universal Adversarial Attack

- One perturbation for all examples
- For each  $x_i$ , compute minimal perturbation that sends  $x_i + v$  to decision boundary

Initialize  $v = 0$

$$\Delta v_i = \arg \min_r \|r\|_2 \text{ s.t. } f(x_i + v + r) \neq f(x)$$

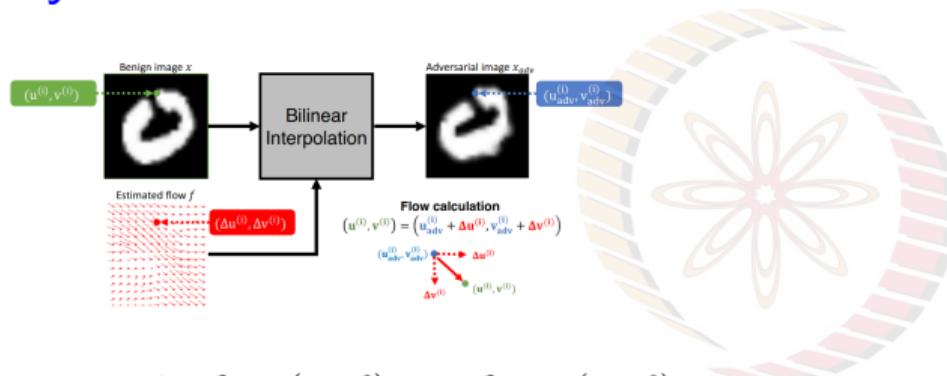
$$v = \mathcal{P}_{p,\epsilon}(v + \Delta v_i)$$

$$\mathcal{P}_{p,\epsilon}(v) = \arg \min_{v'} \|v - v'\|_2 \text{ subject to } \|v'\|_p \leq \epsilon$$

<sup>8</sup>Moosavi-Dezfooli et al, Universal Adversarial Perturbations, CVPR 2017

# Non- $L_p$ White-box Adversarial Attacks<sup>9 10</sup>

## Spatially Transformed Adversarial Attacks



$$f^* = \arg \min_f \mathcal{L}_{adv}(x, f) + \tau \mathcal{L}_{flow}(x, f)$$

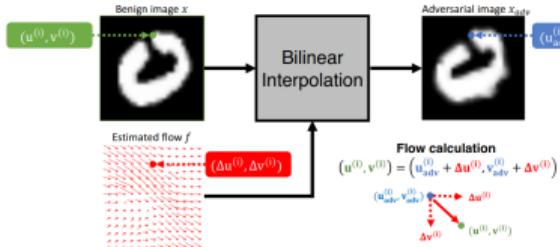
$\mathcal{L}_{adv}$  encourages generated examples to be misclassified;  $\mathcal{L}_{flow}$  ensures that spatial transformation distance is minimized

<sup>9</sup>Xiao et al, Spatially Transformed Adversarial Examples, ICLR 2018

<sup>10</sup>Laidlaw et al, Functional Adversarial Attacks, NeurIPS 2019

# Non- $L_p$ White-box Adversarial Attacks<sup>9 10</sup>

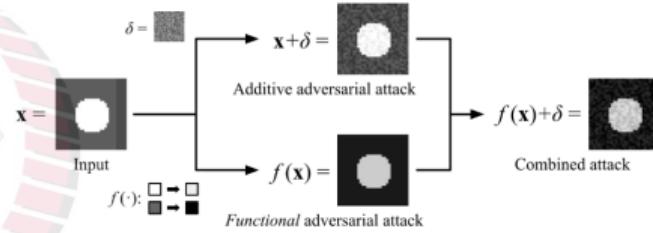
## Spatially Transformed Adversarial Attacks



$$f^* = \arg \min_f \mathcal{L}_{adv}(x, f) + \tau \mathcal{L}_{flow}(x, f)$$

$\mathcal{L}_{adv}$  encourages generated examples to be misclassified;  $\mathcal{L}_{flow}$  ensures that spatial transformation distance is minimized

## Functional Adversarial Attacks



- A single function to be used to perturb input features to produce an adversarial example
- E.g. attack applied on colors of an image can change all red pixels simultaneously to light red.

<sup>9</sup>Xiao et al, Spatially Transformed Adversarial Examples, ICLR 2018

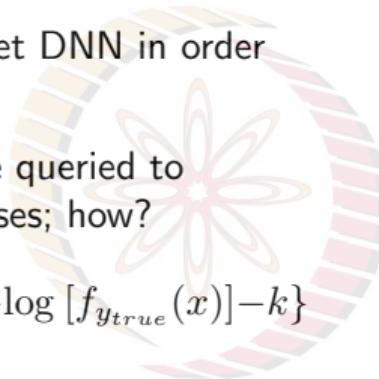
<sup>10</sup>Laidlaw et al, Functional Adversarial Attacks, NeurIPS 2019

# Black-box Adversarial Attacks: Gradient Estimation-based<sup>11</sup>

## Zeroth-Order Optimization (ZOO)

- Need to estimate gradients of target DNN in order to produce an adversarial image
- But now, target model can only be queried to obtain probability scores of all classes; how?

$$\mathcal{L}(x, y) = \max \left\{ \max_{i \neq y_{true}} \log [f_i(x)] - \log [f_{y_{true}}(x)] - k \right\}$$



NPTEL

<sup>11</sup>Chen et al, ZOO: Zeroth Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models, AISecW 2017, Cheng et al, Query-efficient Hard-label Black-box Attack: An Optimization-based Approach, 2018

# Black-box Adversarial Attacks: Gradient Estimation-based<sup>11</sup>

## Zeroth-Order Optimization (ZOO)

- Need to estimate gradients of target DNN in order to produce an adversarial image
- But now, target model can only be queried to obtain probability scores of all classes; how?

$$\mathcal{L}(x, y) = \max\left\{ \max_{i \neq y_{true}} \log [f_i(x)] - \log [f_{y_{true}}(x)] - k \right\}$$

- To estimate gradient:

$$\frac{\partial f(x)}{\partial x} \approx \frac{f(x + he_i) - f(x - he_i)}{2h}$$

<sup>11</sup>Chen et al, ZOO: Zeroth Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models, AISecW 2017, Cheng et al, Query-efficient Hard-label Black-box Attack: An Optimization-based Approach, 2018

# Black-box Adversarial Attacks: Gradient Estimation-based<sup>11</sup>

## Zer0th-Order Optimization (ZOO)

- Need to estimate gradients of target DNN in order to produce an adversarial image
- But now, target model can only be queried to obtain probability scores of all classes; how?

$$\mathcal{L}(x, y) = \max \left\{ \max_{i \neq y_{true}} \log [f_i(x)] - \log [f_{y_{true}}(x)] - k \right\}$$

- To estimate gradient:

$$\frac{\partial f(x)}{\partial x} \approx \frac{f(x + he_i) - f(x - he_i)}{2h}$$

## Opt-Attack

- Target model can only be queried to obtain true label (hard-label setting); now, how?

$$g(\theta) = \min_{\lambda > 0} \lambda \text{ s.t } f(x + \lambda \cdot \frac{\theta}{\|\theta\|}) \neq y_{true}$$

NPTEL

<sup>11</sup>Chen et al, ZOO: Zer0th Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models, AISecW 2017, Cheng et al, Query-efficient Hard-label Black-box Attack: An Optimization-based Approach, 2018

# Black-box Adversarial Attacks: Gradient Estimation-based<sup>11</sup>

## Zer0th-Order Optimization (ZOO)

- Need to estimate gradients of target DNN in order to produce an adversarial image
- But now, target model can only be queried to obtain probability scores of all classes; how?

$$\mathcal{L}(x, y) = \max \left\{ \max_{i \neq y_{true}} \log [f_i(x)] - \log [f_{y_{true}}(x)] - k \right\}$$

- To estimate gradient:

$$\frac{\partial f(x)}{\partial x} \approx \frac{f(x + he_i) - f(x - he_i)}{2h}$$

## Opt-Attack

- Target model can only be queried to obtain true label (hard-label setting); now, how?

$$g(\theta) = \min_{\lambda > 0} \lambda \text{ s.t } f(x + \lambda \cdot \frac{\theta}{\|\theta\|}) \neq y_{true}$$

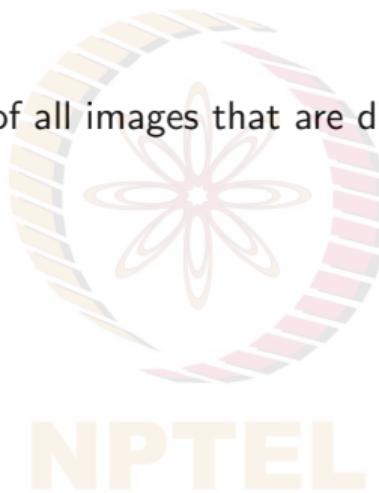
- A coarse-grained search to initially find a decision boundary and then finetune the solution using Binary Search

<sup>11</sup>Chen et al, ZOO: Zer0th Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models, AISecW 2017, Cheng et al, Query-efficient Hard-label Black-box Attack: An Optimization-based Approach, 2018

# Black-box Adversarial Attacks: Gradient-Free<sup>12</sup>

## Greedy Local-Search

- Create a neighborhood consisting of all images that are different from previous round's image by one pixel only



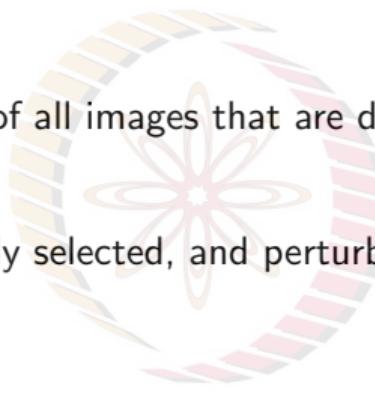
---

<sup>12</sup>Nina Narodytska et al, Simple Black-box Adversarial Perturbations for Deep Networks, 2016

# Black-box Adversarial Attacks: Gradient-Free<sup>12</sup>

## Greedy Local-Search

- Create a neighborhood consisting of all images that are different from previous round's image by one pixel only
- Initially, a pixel location is randomly selected, and perturbation is added to it



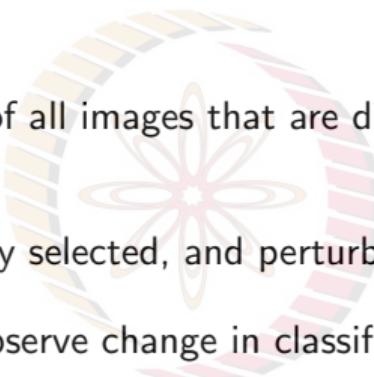
---

<sup>12</sup>Nina Narodytska et al, Simple Black-box Adversarial Perturbations for Deep Networks, 2016

# Black-box Adversarial Attacks: Gradient-Free<sup>12</sup>

## Greedy Local-Search

- Create a neighborhood consisting of all images that are different from previous round's image by one pixel only
- Initially, a pixel location is randomly selected, and perturbation is added to it
- Calculate importance of pixel by observe change in classification accuracy after adding noise



NPTEL

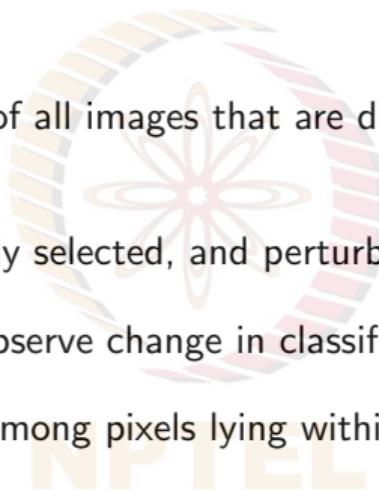
---

<sup>12</sup>Nina Narodytska et al, Simple Black-box Adversarial Perturbations for Deep Networks, 2016

# Black-box Adversarial Attacks: Gradient-Free<sup>12</sup>

## Greedy Local-Search

- Create a neighborhood consisting of all images that are different from previous round's image by one pixel only
- Initially, a pixel location is randomly selected, and perturbation is added to it
- Calculate importance of pixel by observe change in classification accuracy after adding noise
- The next pixel location is chosen among pixels lying within a square whose side length is  $2p$



<sup>12</sup>Nina Narodytska et al, Simple Black-box Adversarial Perturbations for Deep Networks, 2016

# Black-box Adversarial Attacks: Gradient-Free<sup>12</sup>

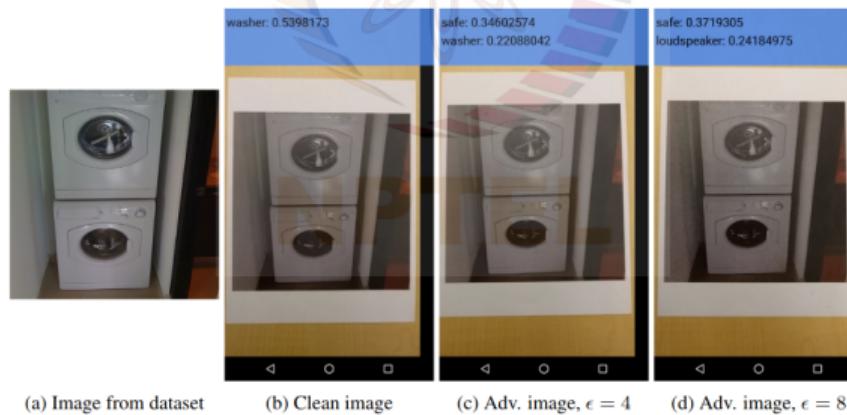
## Greedy Local-Search

- Create a neighborhood consisting of all images that are different from previous round's image by one pixel only
- Initially, a pixel location is randomly selected, and perturbation is added to it
- Calculate importance of pixel by observe change in classification accuracy after adding noise
- The next pixel location is chosen among pixels lying within a square whose side length is  $2p$
- Use these importance to approximate the loss function's gradient

<sup>12</sup>Nina Narodytska et al, Simple Black-box Adversarial Perturbations for Deep Networks, 2016

# Need for Adversarial Defenses: Adversarial Examples in Physical World <sup>13</sup>

- Poses a serious security threat to services using modern day Deep Learning models
- E.g. TensorFlow Camera Demo app to classify clean and adversarial generated images
- A clean image (b) recognized correctly as a “washer” when perceived through camera, while adversarial images (c) and (d) are misclassified

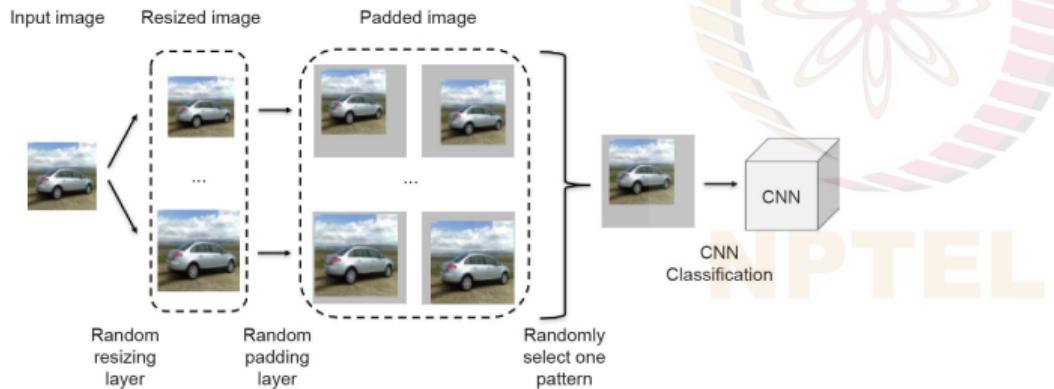


<sup>13</sup>Alexey Kurakin et al, Adversarial Examples in the Physical World, ICLRW 2017

# Adversarial Defenses: Randomization<sup>14 15</sup>

## Random Input Transformation

- Two random transformations—random resizing and padding—to mitigate adversarial effects at inference time



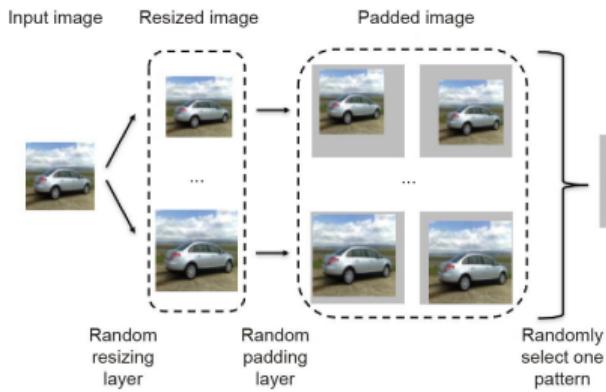
<sup>14</sup> Xie et al, Mitigating Adversarial Effects through Randomization, 2017

<sup>15</sup> Liu et al, Towards Robust Neural Networks via Random Self-Ensemble, ECCV 2018

# Adversarial Defenses: Randomization<sup>14 15</sup>

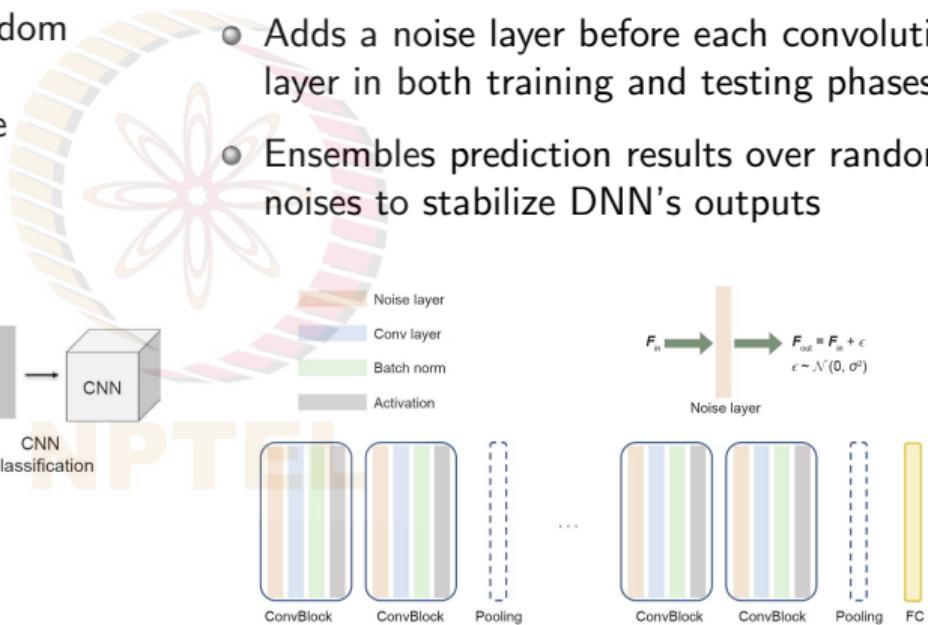
## Random Input Transformation

- Two random transformations—random resizing and padding—to mitigate adversarial effects at inference time



## Random Noising

- Adds a noise layer before each convolution layer in both training and testing phases
- Ensembles prediction results over random noises to stabilize DNN's outputs

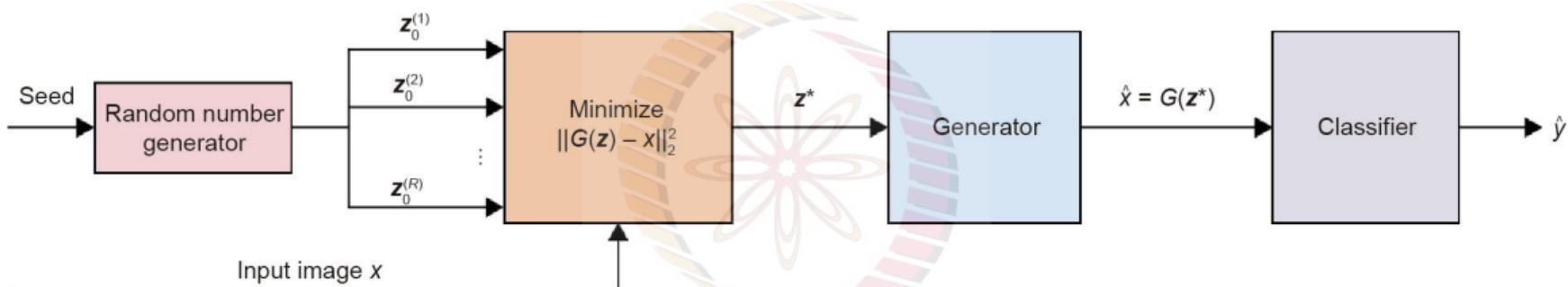


<sup>14</sup> Xie et al, Mitigating Adversarial Effects through Randomization, 2017

<sup>15</sup> Liu et al, Towards Robust Neural Networks via Random Self-Ensemble, ECCV 2018

# Adversarial Defenses: Input Cleansing/Reconstruction <sup>16</sup>

## Defense-GAN

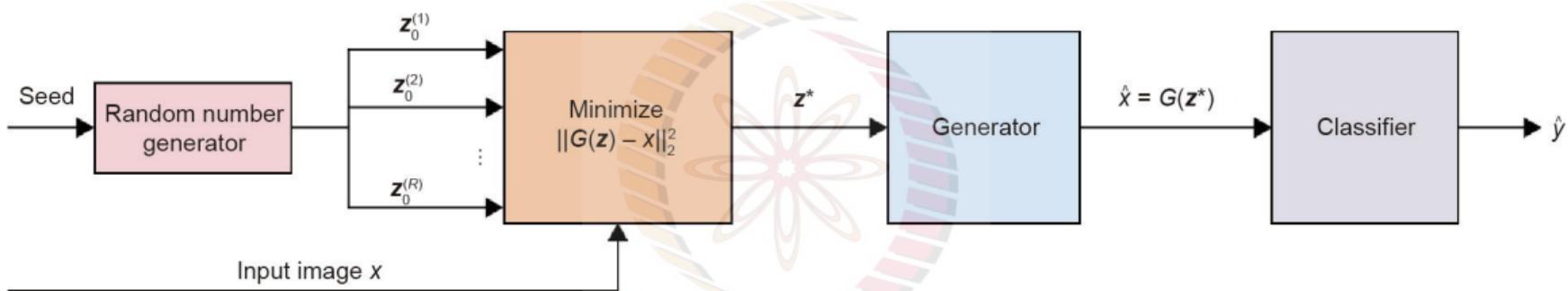


- Trains a generator to model distribution of benign images

<sup>16</sup>Samangouei et al, Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models, ICLR 2018

# Adversarial Defenses: Input Cleansing/Reconstruction <sup>16</sup>

## Defense-GAN

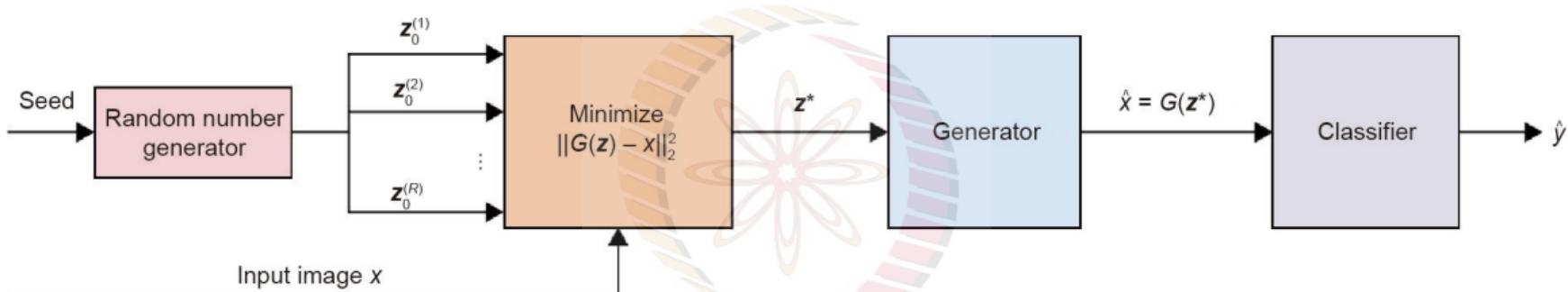


- Trains a generator to model distribution of benign images
- In the testing stage, cleanses an adversarial input by searching for a close image, and feeds this benign image into the classifier

<sup>16</sup>Samangouei et al, Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models, ICLR 2018

# Adversarial Defenses: Input Cleansing/Reconstruction <sup>16</sup>

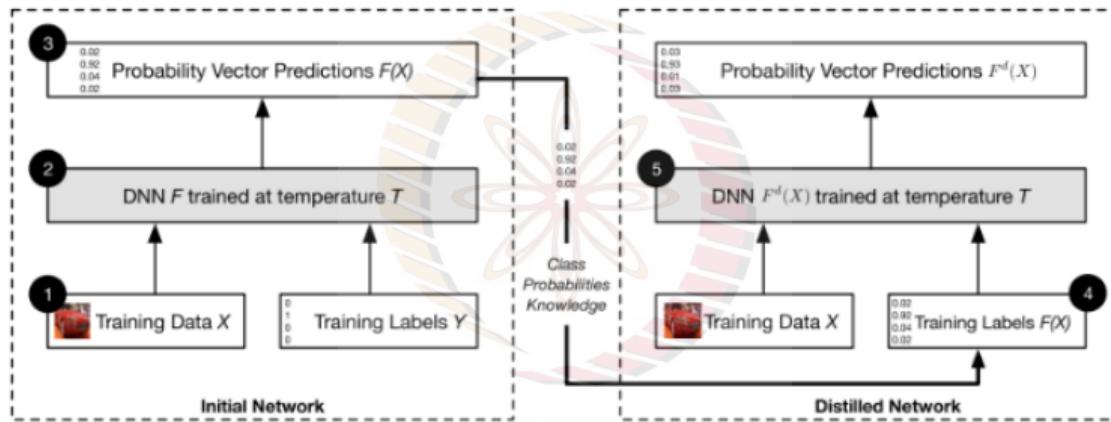
## Defense-GAN



- Trains a generator to model distribution of benign images
- In the testing stage, cleanses an adversarial input by searching for a close image, and feeds this benign image into the classifier
- Other similar methods: PixelDefend, MagNet, APE-GAN, etc

<sup>16</sup>Samangouei et al, Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models, ICLR 2018

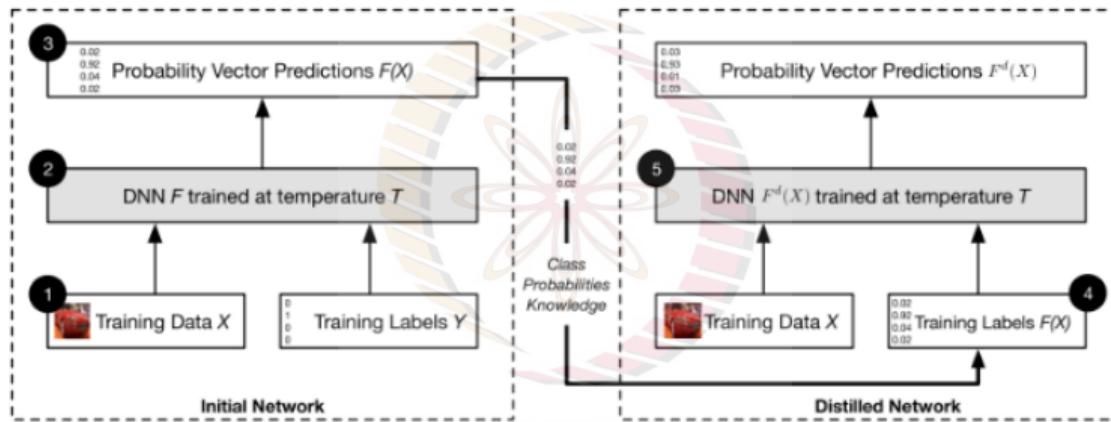
# Adversarial Defenses: Network Distillation <sup>17</sup>



- Probability of classes produced by first DNN used as inputs to train second DNN

<sup>17</sup>Papernot et al, Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks, arXiv 2016

# Adversarial Defenses: Network Distillation <sup>17</sup>



- Probability of classes produced by first DNN used as inputs to train second DNN
- Using high-temperature softmax reduces model sensitivity to small perturbations

<sup>17</sup>Papernot et al, Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks, arXiv 2016

# Adversarial Defenses: Adversarial Training<sup>18</sup>

## PGD/FGSM Adversarial Training

- Simply add PGD/FGSM attack inside your training loop
- “Ultimate data augmentation”
- Create specific perturbations that best fool our model and classify them correctly
- Most popular and widely opted; current SOTA



$$\min_{\theta} \mathcal{L}(x, y; \theta) \quad \text{vs} \quad \min_{\theta} \max_{\delta \in \Delta} \mathcal{L}(x + \delta, y; \theta)$$

<sup>18</sup>Goodfellow et al, Explaining and Harnessing Adversarial Examples, ICLR 2015, Madry et al, Towards Deep Learning Models Resistant To Adversarial Attack, 2017

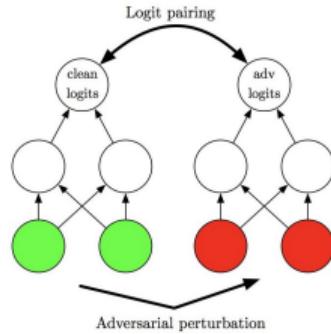
# Adversarial Defenses: Adversarial Training<sup>19, 20</sup>

## Adversarial Logit Pairing

- Encourages similarity between logits of clean  $x$  and adversarial  $x_{adv}$  examples

$$\mathcal{L}_{alp} = \mathcal{L}_{ce} + \frac{\lambda}{N} \cdot \sum_{i=1}^N D(f(x^i), f(x_{adv}^i))$$

$D$  encourages logits to be similar; e.g.  $L_2$  loss



<sup>19</sup>Kannan et al, Adversarial Logit Pairing, 2018

<sup>20</sup>Zhang et al, Theoretically Principled Trade-off between Robustness and Accuracy", ICML 2019

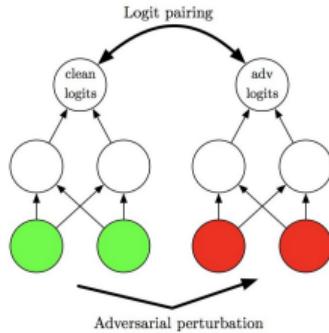
# Adversarial Defenses: Adversarial Training<sup>19, 20</sup>

## Adversarial Logit Pairing

- Encourages similarity between logits of clean  $x$  and adversarial  $x_{adv}$  examples

$$\mathcal{L}_{alp} = \mathcal{L}_{ce} + \frac{\lambda}{N} \cdot \sum_{i=1}^N D(f(x^i), f(x_{adv}^i))$$

$D$  encourages logits to be similar; e.g.  $L_2$  loss



## TRADES

- Decompose prediction error for adversarial examples as sum of natural error and boundary error

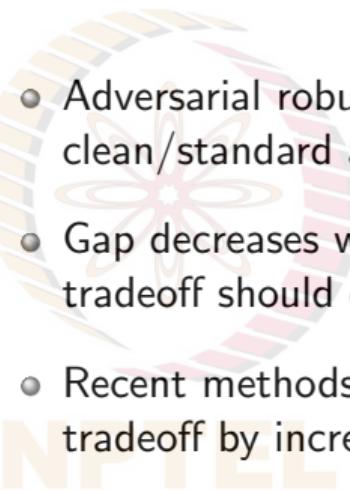
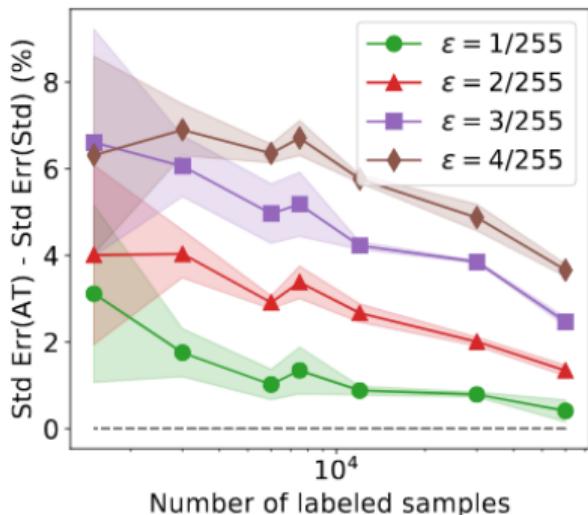
$$\min_f \mathbf{E} \left\{ \mathcal{L}(f(x), y_{true}) + \max_{\delta \in \Delta} \frac{\mathcal{L}(f(x), f(x + \delta))}{\lambda} \right\}$$

- While ALP uses PGD adversarial examples, TRADES computes  $x_{adv}$  as  $\max_{\delta \in \Delta} \mathcal{L}(f(x), f(x + \delta))$
- ALP enforces  $L_2$  loss while TRADES uses classification-calibrated loss

<sup>19</sup>Kannan et al, Adversarial Logit Pairing, 2018

<sup>20</sup>Zhang et al, "Theoretically Principled Trade-off between Robustness and Accuracy", ICML 2019

# Tradeoff between Robust and Clean Accuracy<sup>21</sup>

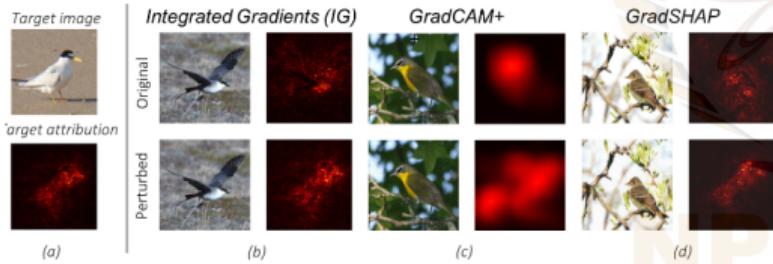


- Adversarial robustness comes at a cost of decreased clean/standard accuracy
- Gap decreases with increase in training set size; tradeoff should disappear with infinite data
- Recent methods like IAT, AVMixup reduce this tradeoff by increasing train set size

<sup>21</sup>Tsipras et al, Robustness May Be at Odds with Accuracy, ICLR 2019, Lamb et al, Interpolated Adversarial Training: Achieving Robust Neural Networks Without Sacrificing Too Much Accuracy, AISec 2019, Lee et al, Adversarial Vertex Mixup: Toward Better Adversarially Robust Generalization, CVPR 2020

# Other Notions of Robustness<sup>22</sup>

- **Attributional Robustness:** Robustness of attributions (explanations/ saliency maps)



- **Unseen Natural Corruptions:** Robustness to common occurring distribution shifts like fog, blur, snow, etc.



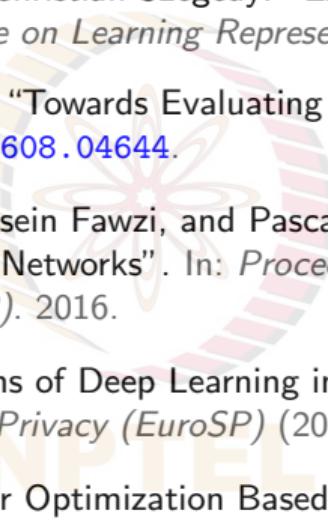
<sup>22</sup>Hendrycks et al, Benchmarking Neural Network Robustness to Common Corruptions and Perturbations, 2019, Singh et al, Attributional Robustness Training using Input-Gradient Spatial Alignment, ECCV 2020

# Homework

## Readings

- Adversarial Machine Learning Tutorials: [Tutorial 1](#), [Tutorial 2](#) and [Tutorial 3](#)
- Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey
- Adversarial Machine Learning Reading List from beginner to advanced.
- PyTorch and TensorFlow Libraries/Implementations: [MadryLab](#), [Cleverhans](#) and [Advertorch](#)

# References I

- 
-  Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *International Conference on Learning Representations*. 2015.
  -  Nicholas Carlini and David A. Wagner. "Towards Evaluating the Robustness of Neural Networks". In: *CoRR abs/1608.04644* (2016). arXiv: [1608.04644](https://arxiv.org/abs/1608.04644).
  -  Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.
  -  Nicolas Papernot et al. "The Limitations of Deep Learning in Adversarial Settings". In: *2016 IEEE European Symposium on Security and Privacy (EuroSP)* (2016), pp. 372–387.
  -  Pin-Yu Chen et al. "ZOO: Zeroth Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models". In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. AISeC '17. Dallas, Texas, USA: Association for Computing Machinery, 2017, 15–26.

## References II

- 
-  Alexey Kurakin, Ian Goodfellow, and Samy Bengio. "Adversarial examples in the physical world". In: *ICLR Workshop* (2017).
  -  Seyed-Mohsen Moosavi-Dezfooli et al. "Universal Adversarial Perturbations". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.
  -  Harini Kannan, Alexey Kurakin, and Ian J. Goodfellow. "Adversarial Logit Pairing". In: *CoRR* abs/1803.06373 (2018). arXiv: [1803.06373](https://arxiv.org/abs/1803.06373).
  -  Pouya Samangouei, Maya Kabkab, and Rama Chellappa. "Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models". In: *International Conference on Learning Representations*. 2018.
  -  Chaowei Xiao et al. "Spatially Transformed Adversarial Examples". In: *International Conference on Learning Representations*. 2018.
  -  Dan Hendrycks and Thomas G. Dietterich. "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations". In: *CoRR* abs/1903.12261 (2019). arXiv: [1903.12261](https://arxiv.org/abs/1903.12261).

## References III

-  Andrew Ilyas et al. "Adversarial Examples Are Not Bugs, They Are Features". In: *Advances in Neural Information Processing Systems* 32. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 125–136.
-  Cassidy Laidlaw and Soheil Feizi. "Functional Adversarial Attacks". In: *Advances in Neural Information Processing Systems* 32. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 10408–10418.
-  Alex Lamb et al. "Interpolated Adversarial Training: Achieving Robust Neural Networks Without Sacrificing Too Much Accuracy". In: *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*. AISeC'19. London, United Kingdom: Association for Computing Machinery, 2019, 95–103.
-  Seyed-Mohsen Moosavi-Dezfooli et al. "Robustness via Curvature Regularization, and Vice Versa". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.
-  Chongli Qin et al. "Adversarial Robustness through Local Linearization". In: *Advances in Neural Information Processing Systems* 32. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 13847–13856.

## References IV

-  Shibani Santurkar et al. "Image Synthesis with a Single (Robust) Classifier". In: *Advances in Neural Information Processing Systems* 32. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 1262–1273.
-  Lukas Schott et al. "Towards the first adversarially robust neural network model on MNIST". In: *International Conference on Learning Representations*. 2019.
-  Dimitris Tsipras et al. "Robustness May Be at Odds with Accuracy". In: *International Conference on Learning Representations*. 2019.
-  Hongyang Zhang et al. "Theoretically Principled Trade-off between Robustness and Accuracy". In: ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, 2019, pp. 7472–7482.
-  Tianyuan Zhang and Zhanxing Zhu. "Interpreting Adversarially Trained Convolutional Neural Networks". In: *CoRR* abs/1905.09797 (2019). arXiv: [1905.09797](https://arxiv.org/abs/1905.09797).

# References V



Saehyung Lee, Hyungyu Lee, and Sungroh Yoon. "Adversarial Vertex Mixup: Toward Better Adversarially Robust Generalization". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.



Hadi Salman et al. *Do Adversarially Robust ImageNet Models Transfer Better?* 2020. arXiv: [2007.08489 \[cs.CV\]](https://arxiv.org/abs/2007.08489).



Mayank Singh et al. *Attributional Robustness Training using Input-Gradient Spatial Alignment*. 2020. arXiv: [1911.13073 \[cs.CV\]](https://arxiv.org/abs/1911.13073).

