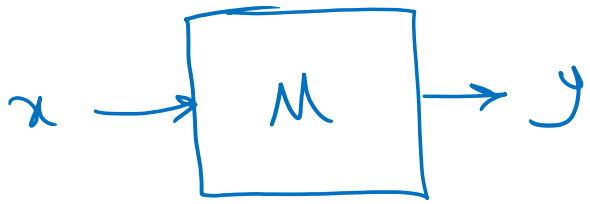
Robustness: Attacks and Defenses

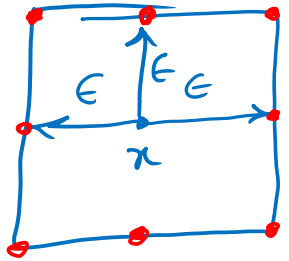
Mostafa Tavassolipour

Review



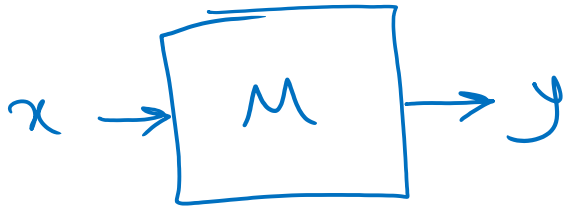
FGSM

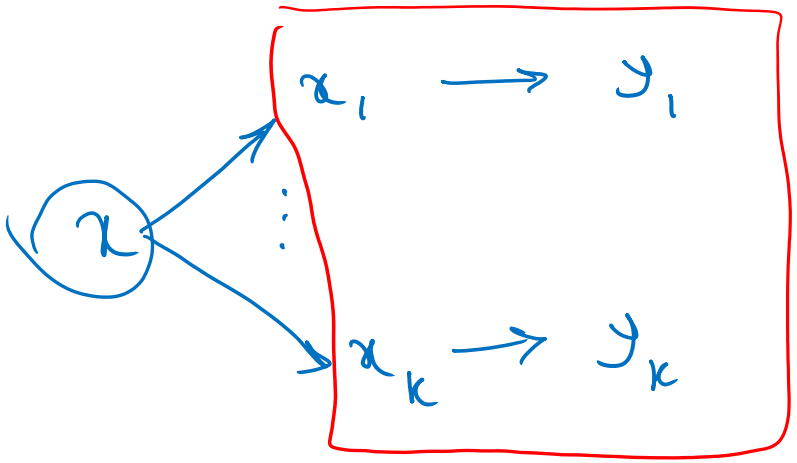
$$x' = x + \epsilon \, \mathrm{sign}\left[\nabla_x J(\theta, x, y)\right]$$

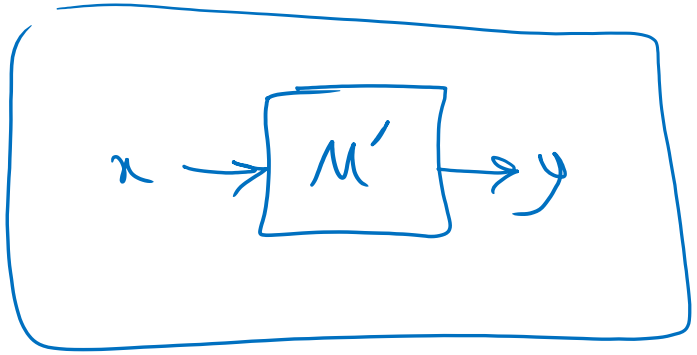


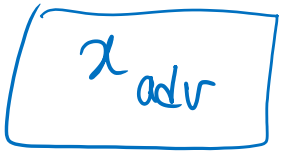
PGD

Black-Box Attack









Other Attacks

- Randomized FGSM
- MI-FGSM
- DeepFool
- Elastic-Net attack to DNNs
- …

Defenses

- Adversarial training
- FGSM adversarial training
- PGD adversarial training
- Defense-GAN
- Defense-VAE
- Ensemble adversarial training
- Adversarial logit pairing
- Generative adversarial training
- Randomization
- …

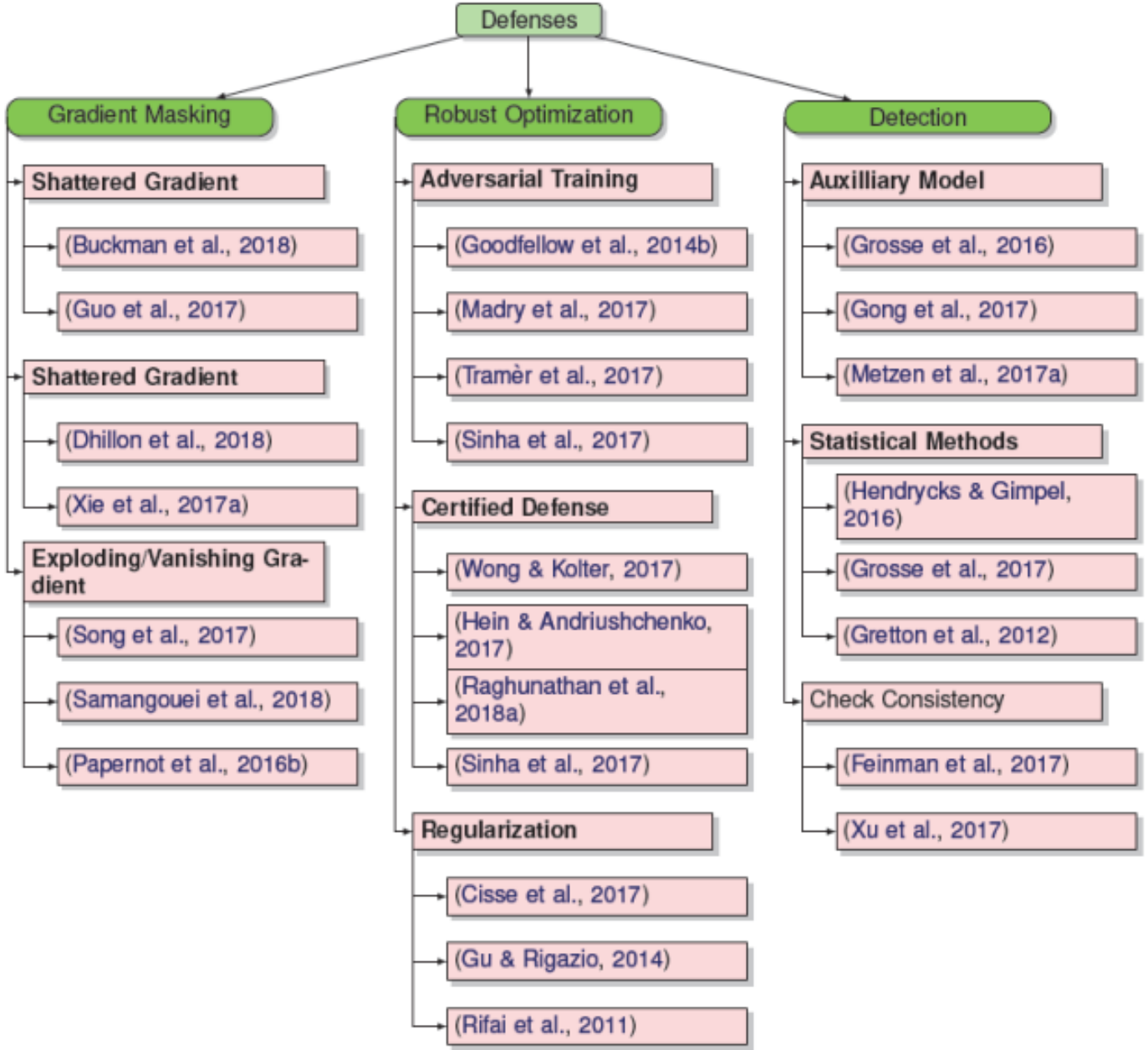


Figure from: Xu et al. (2019) - Adversarial Attacks and Defenses in Images, Graphs and Text: A Review

Defenses

- Gradient masking / Obfuscation
  - Since most attack algorithms are based on the gradient information of the classifier, masking or hiding the gradients will confound the adversaries.
- Robust optimization
  - Re-learning a DNN classifier's parameters can increase its robustness. The trained classifier will correctly classify the subsequently generated adversarial examples.
- Adversarial examples detection
  - It studies the distribution of natural/benign examples, detects adversarial examples, and disallows their input into the classifier.

Defenses

- Gradient masking / Obfuscation
  - Shattered gradients
  - Randomized gradients
  - Exploding/Vanishing gradients
- Robust optimization
  - Adversarial training
  - Certified defenses
  - Regularization
- Adversarial examples detection
  - Auxiliary models
  - Statistical methods

Defenses based on robust optimization

$$\min_{\theta} \mathbb{E}_{x,y\sim D} \left[\max_{\delta \in \Delta} loss(\theta, x + \delta, y) \right]$$

finding a robust model

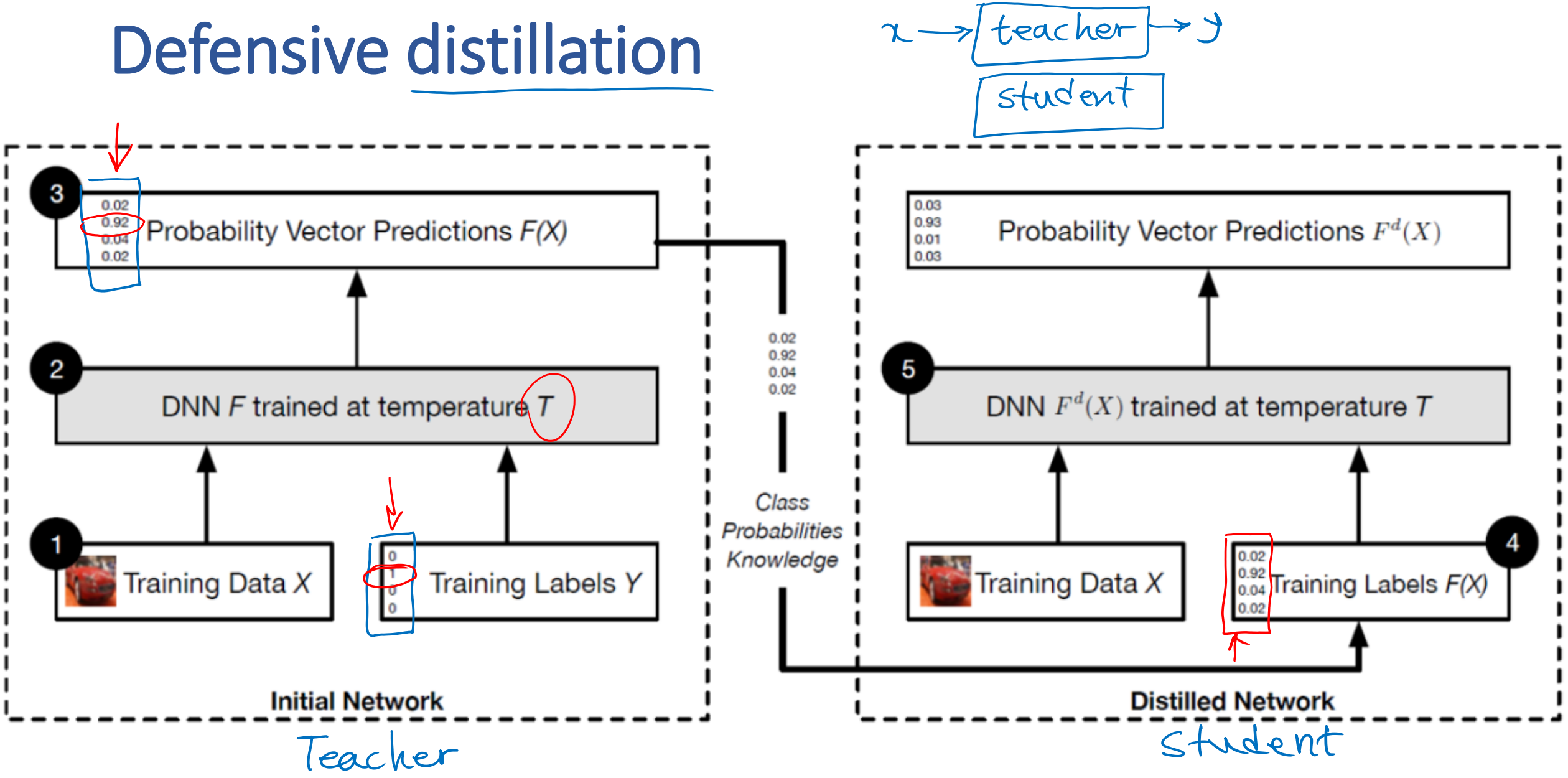
finding a worst-case perturbation

Improve robustness: Train on perturbed inputs

(aka "adversarial training" [Goodfellow et al. 2015])

Actually leads to **robust models** (with some care)

Defensive distillation



Papernot et al. "Distillation as a defense to adversarial perturbations against deep neural networks." In *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 582-597. IEEE, 2016

Softmax with large T

$$F(X) = \frac{e_i^{z/T}}{\sum_{j=1}^{k} e^{z_j/T}}$$

Results: The defensive distillation reduced the success rate of adversarial sample crafting:

MNIST: 95.89% → 0.45%

CIFAR10: 87.89% → 5.11%

Gradient Masking/Obfuscation

- Since most attack algorithms are based on the gradient information of the classifier, masking or hiding the gradients will confound the adversaries.

Shattered Gradients

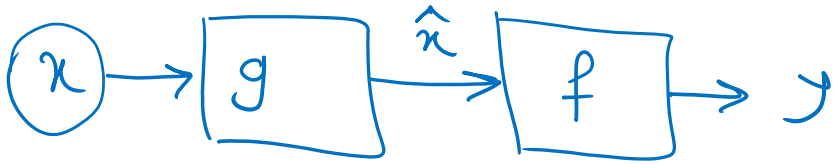
- **Concept**: The algorithm protects the model by preprocessing the input data.

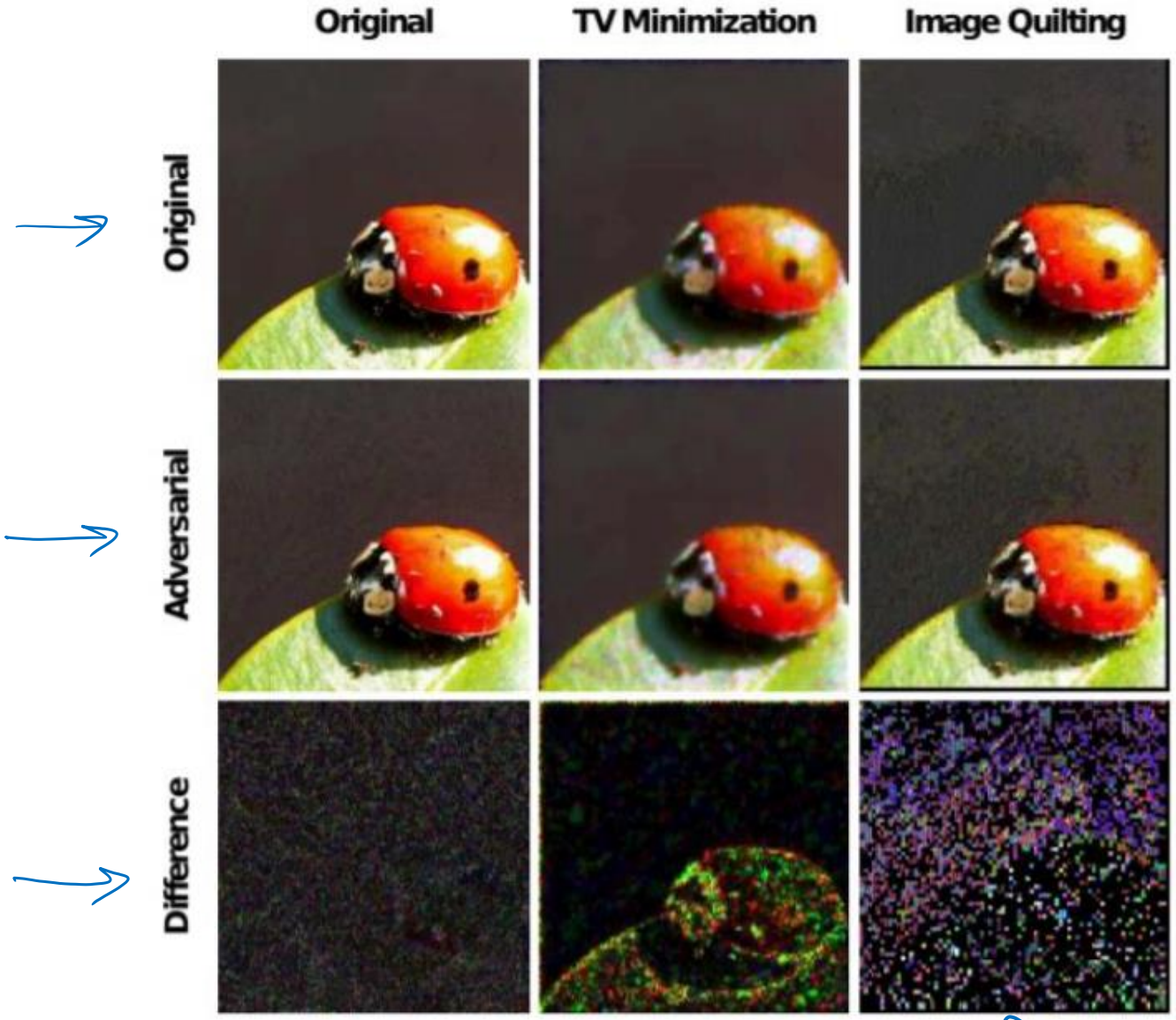
- For example:
  - Adding a non-smooth or non-differentiable preprocessor g(.) and then train a DNN model f on g(X). The trained classifier f(g(.)) is not differentiable in term of X, causing failure of adversarial attacks.

Shattered Gradients

- Guo studied a number of image processing tools, such as image cropping, compressing, total-variance minimization, bit-depth reduction, image quilting, to determine whether these techniques can help to protect the models against adversarial examples.

Guo et al. "Countering adversarial images using input transformations." *arXiv preprint arXiv:1711.00117* (2017)

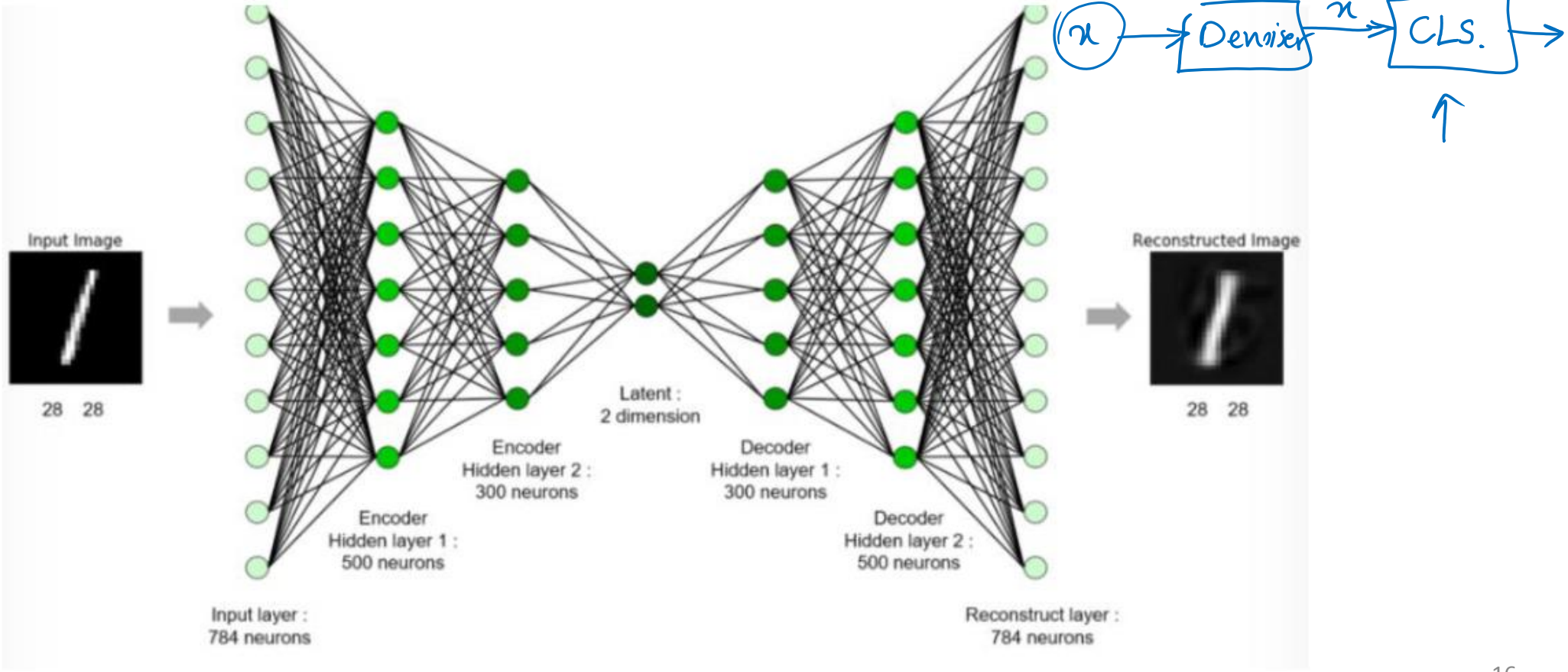




Results show that the total variance minimization and image quilting are efficient for defending against attacks.

SHATTERED GRADIENTS - Denoiser

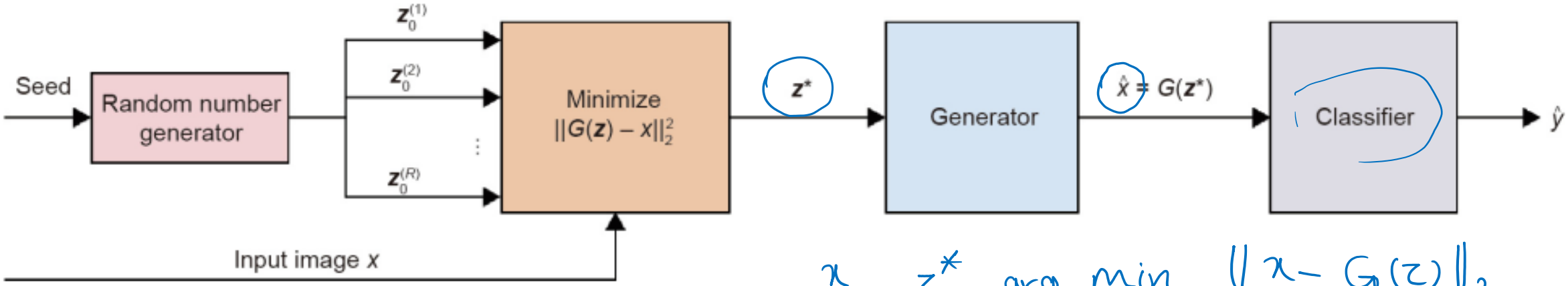
- One defense strategy is to restore the adversarial examples closer to the originals examples, and remove the added manipulation — a.k.a., denoising.



Denoising: GAN-based

- Defense-GAN

$z \sim N(0, I)$



$x \quad z^* = \arg\min_z \|x - G(z)\|_2$

Samangouei P, Kabkab M, Chellappa R. Defense-GAN: protecting classifiers against adversarial attacks using generative models. 2018

Defense-GAN: Method

Defense-GAN is a defense strategy to combat both white-box and black-box adversarial attacks against classification networks.

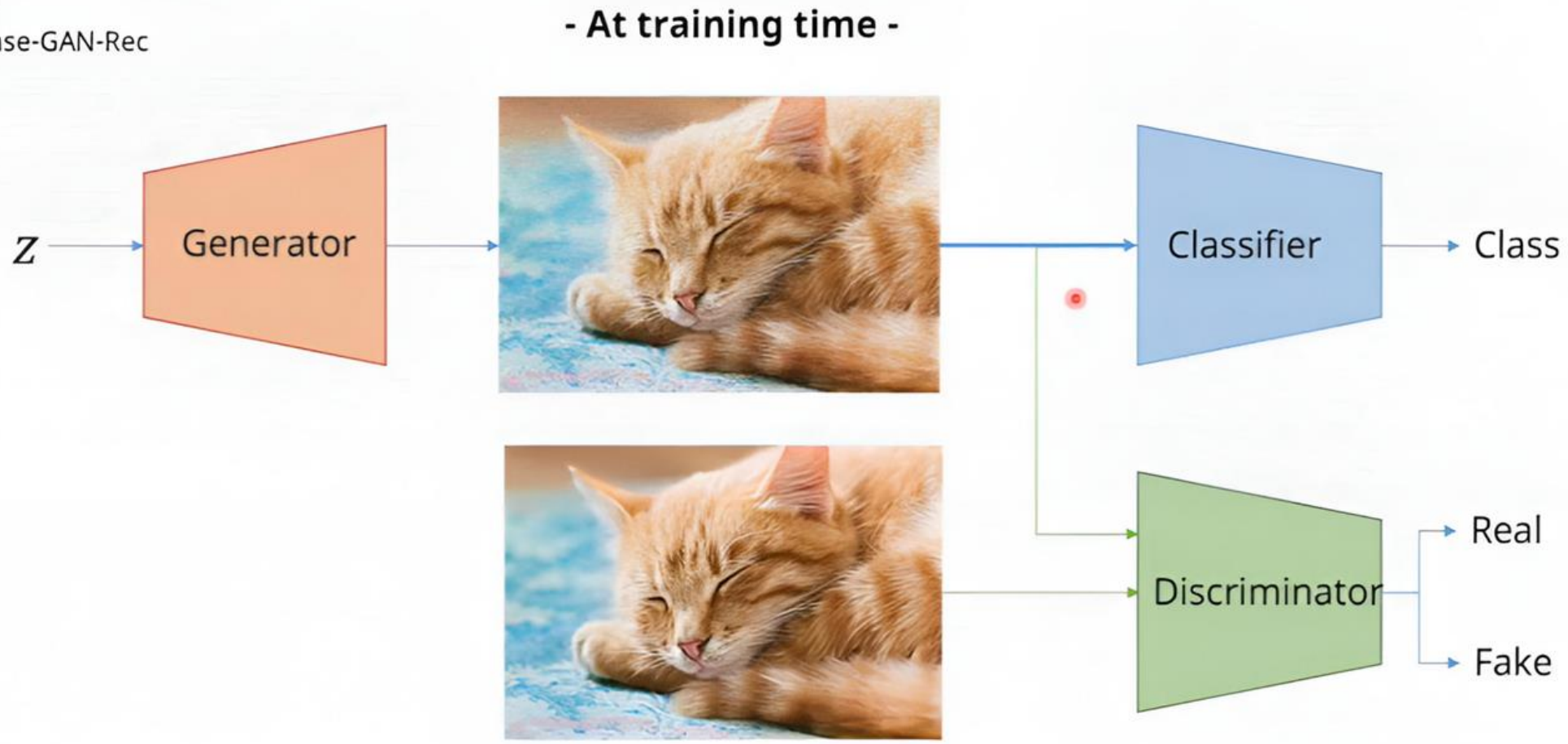
- Assume we have already trained a GAN generator G.
- Now at inference time, we are given a new image x that we want to classify.
- Before feeding x into the classifier, we first find a latent code z* that minimizes the squared error between G(z) and x:

$$\min_{z} \|G(\boldsymbol{z}) - \boldsymbol{x}\|_2^2$$

- Specifically, we want to find a z* such that the reconstructed image G(z*) is as close as possible to the original image x, as measured by mean squared error.
- So we minimize the reconstruction loss over all possible z, and the z that minimizes this is z*.
- Intuitively, we are finding a latent code z* that encodes the image x as best as possible based on what the GAN generator G has learned.
- Once we find this z*, we can reconstruct the image by computing G(z*).
- This reconstructed image G(z*) will be similar to x but potentially cleaned up and lacking adversarial perturbations.
- Finally, we classify the reconstructed image G(z*) rather than the original x.

Defense-GAN

Defense-GAN-Rec



The GAN is trained on the available classifier training dataset in an unsupervised manner.

The classifier can be trained on the original training images, their reconstructions using the generator G, or a combination of the two.

Defense-GAN

Main contribution

adv

- At test time -

x

$\hat{x}$

z → Minimize $\| G(z) - x \|_2^2$ → z^* → Generator → Classifier → Class

Defense-GAN

$$z^* = \arg\min_z \|G(z) - x\|_2^2$$

$$z_t = z_{t-1} - \eta \nabla_z \|G(z)_{t-1} - x\|_2^2$$

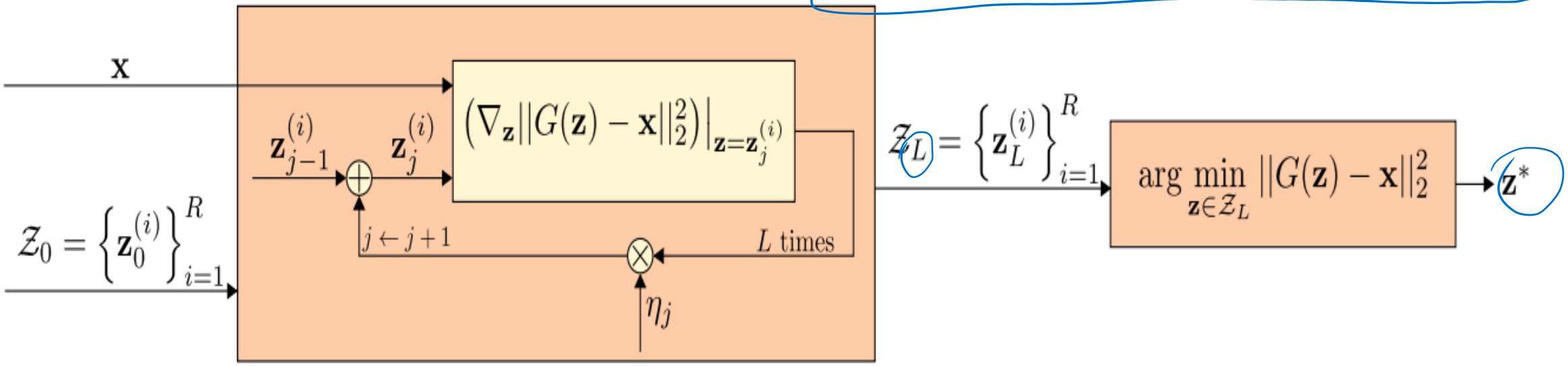
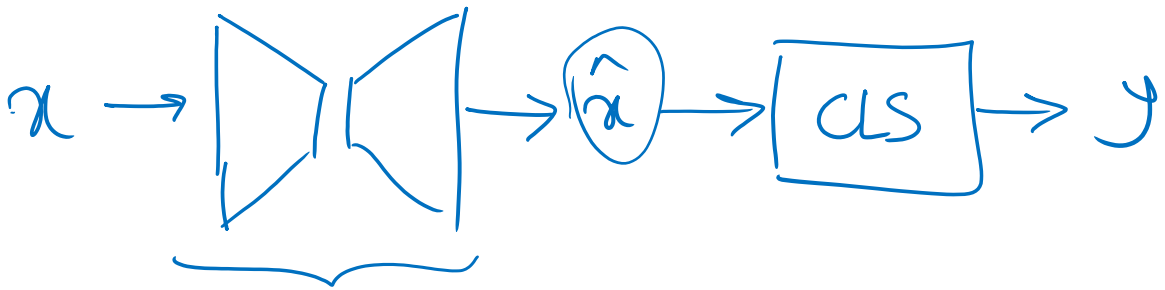
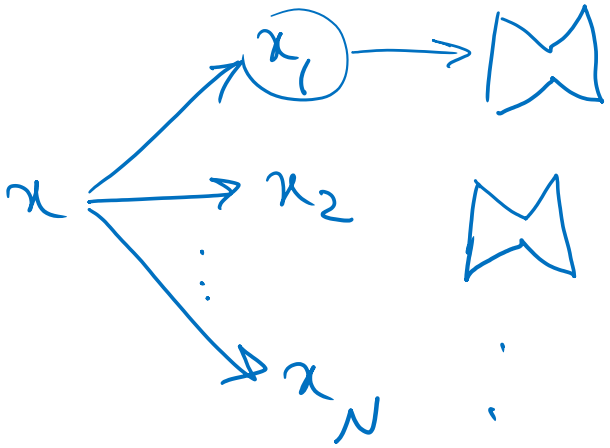


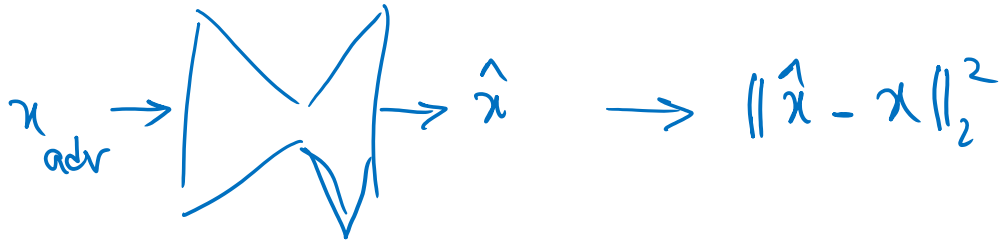
Figure 2: L steps of Gradient Descent are used to estimate the projection of the image onto the range of the generator.

Defense-VAE

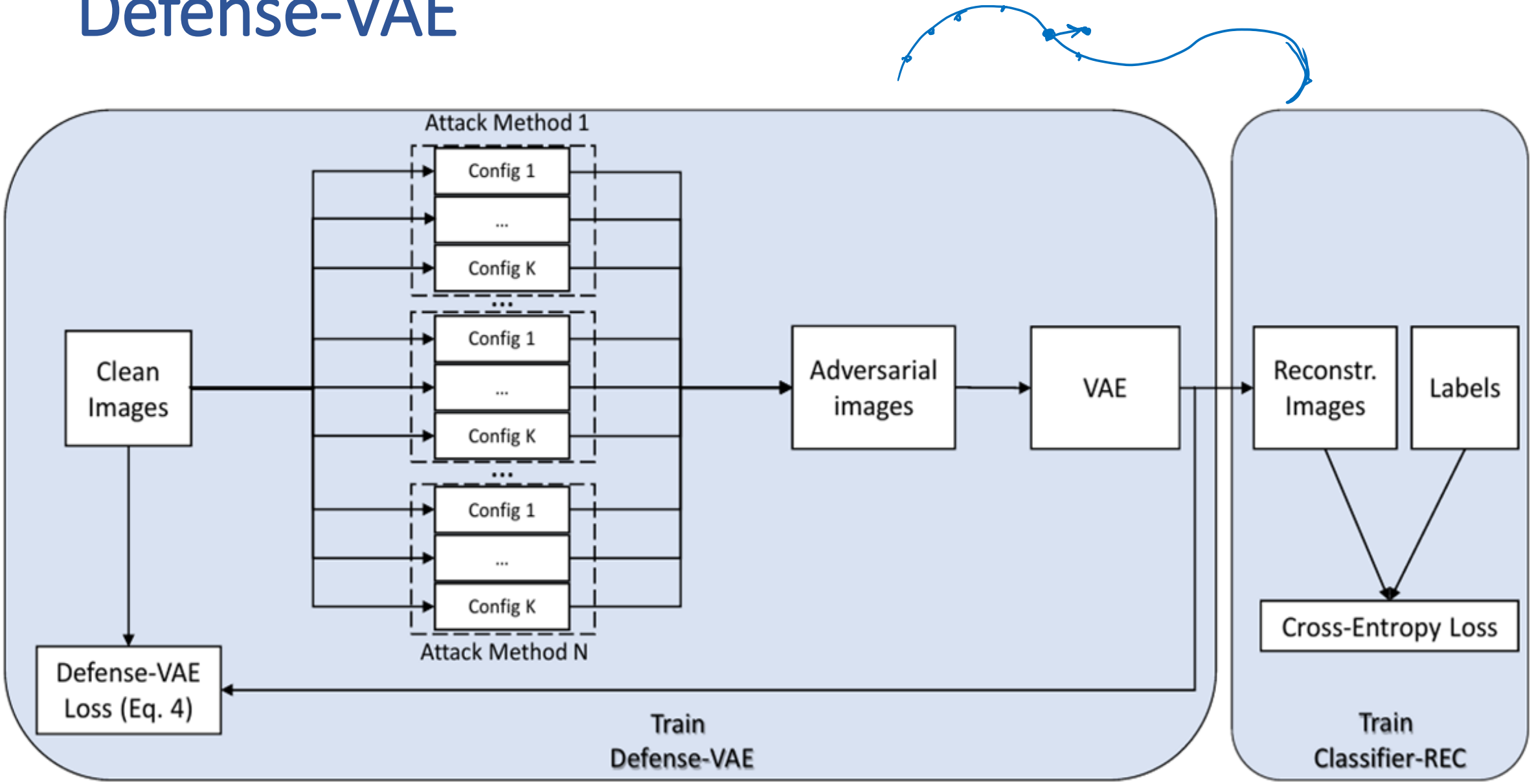
- Generic, defend white-box and black-box attacks without retraining the CNN-classifier
- Further strengthen the defense by retraining or end-to-end fine tuning
- Much higher accuracy that SOTA defense mechanism, outperforms Defense-GAN by 30% on black-box attacks
- Very efficient compared to optimization-based defense mechanism (no iterative optimization)
  - 50x faster than Defense-GAN


x
x̂
cls
y
VAE


x
x1
x2
xN

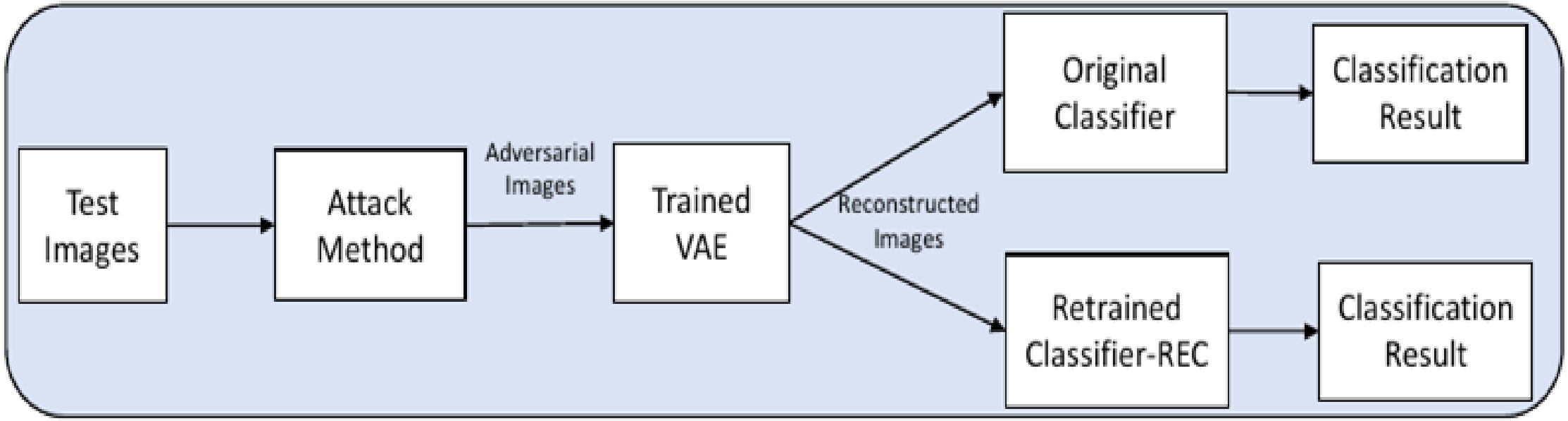

x adv
x̂
‖x̂ - x‖2 2

Defense-VAE


Attack Method 1
Config 1
...
Config K
...
Config 1
...
Config K
...
Config 1
...
Config K
Attack Method N
Clean Images
Adversarial images
VAE
Reconstr. Images
Labels
Cross-Entropy Loss
Defense-VAE Loss (Eq. 4)
Train Defense-VAE
Train Classifier-REC

Defense-VAE

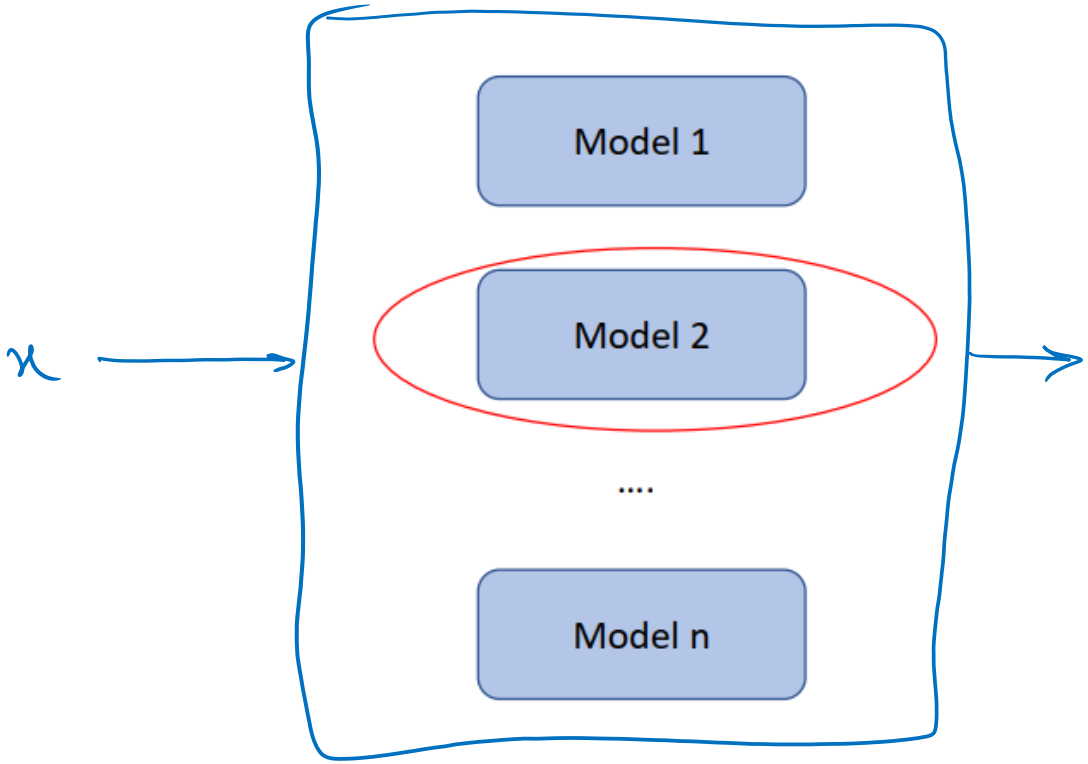
- We can use the trained Defense-VAE to purge the adversarial perturbations from any contaminated images, and the reconstructed images are then fed to the original CNN classifier or retrained CNN classifier for the final image classification.



Test pipeline of Defense-VAE

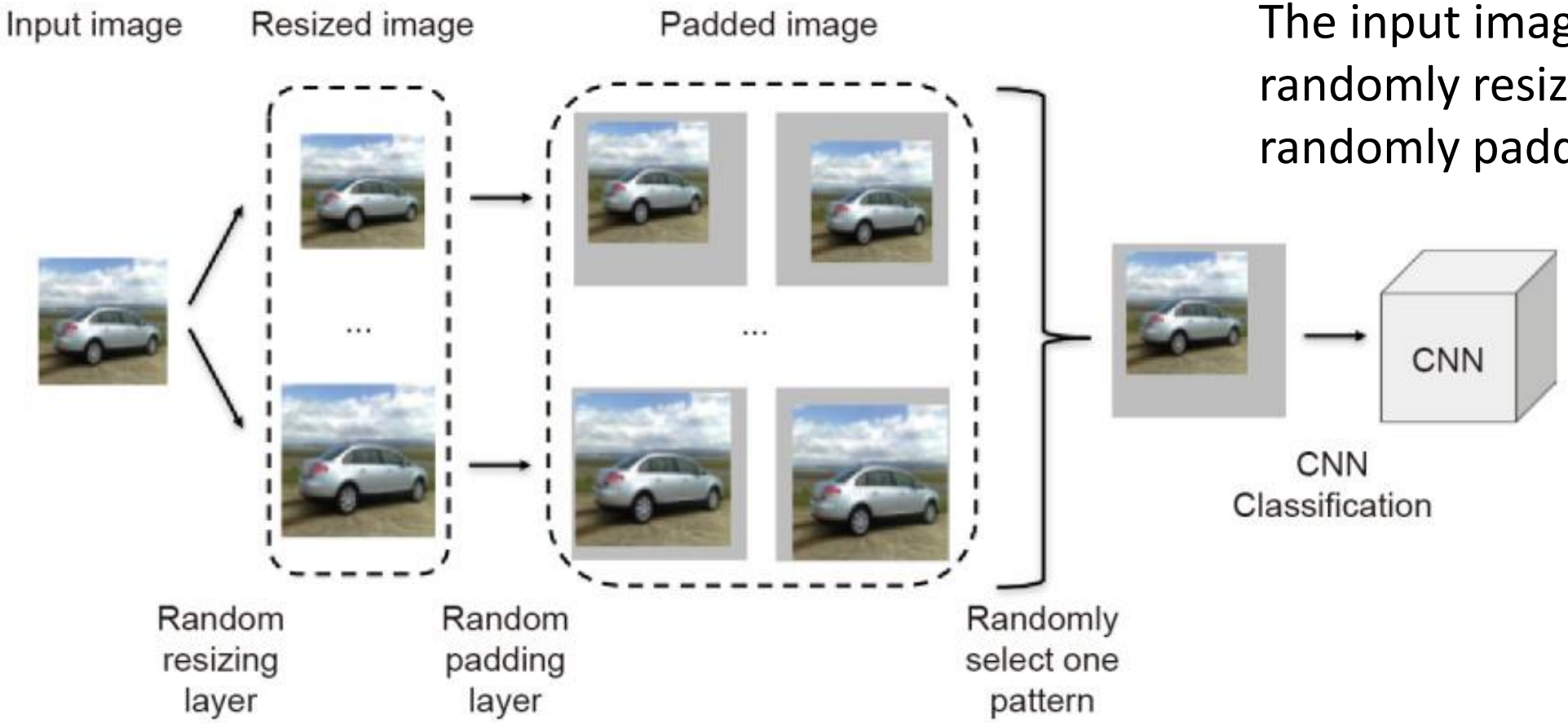
Randomized/Stochastic Gradients

- Randomized defenses, where either the network itself is randomized or the input is randomly transformed before being fed to the classifier, causing the gradients to become randomized.
- Try to randomize the DNN model in order to confound the adversary.
- For example: Prepare multiple classifiers, and randomly select one to predict.



Randomized Gradients: Example paper

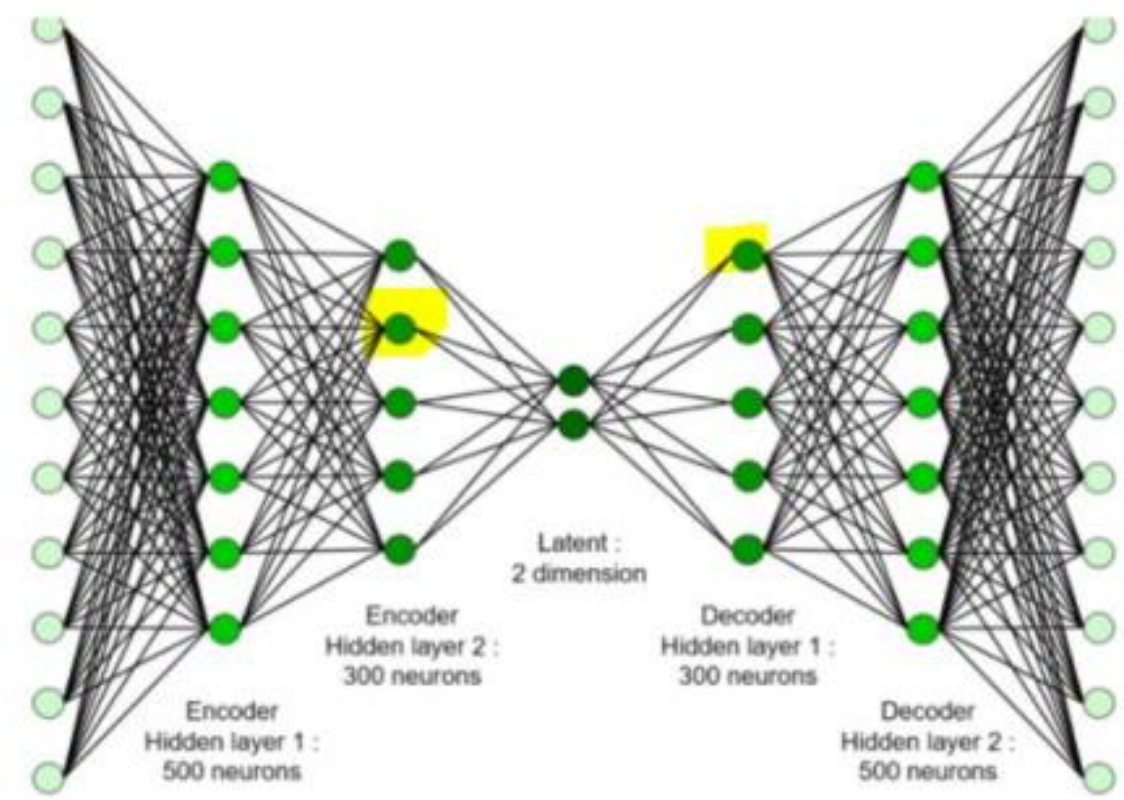
- In Xie's approach, the images were resized to a random size and padded with zeros around the input image. (Ranked 2nd in 2017 NIPS defense AML competition among 200 teams)



The input image is first randomly resized and then randomly padded.

Stochastic Gradients

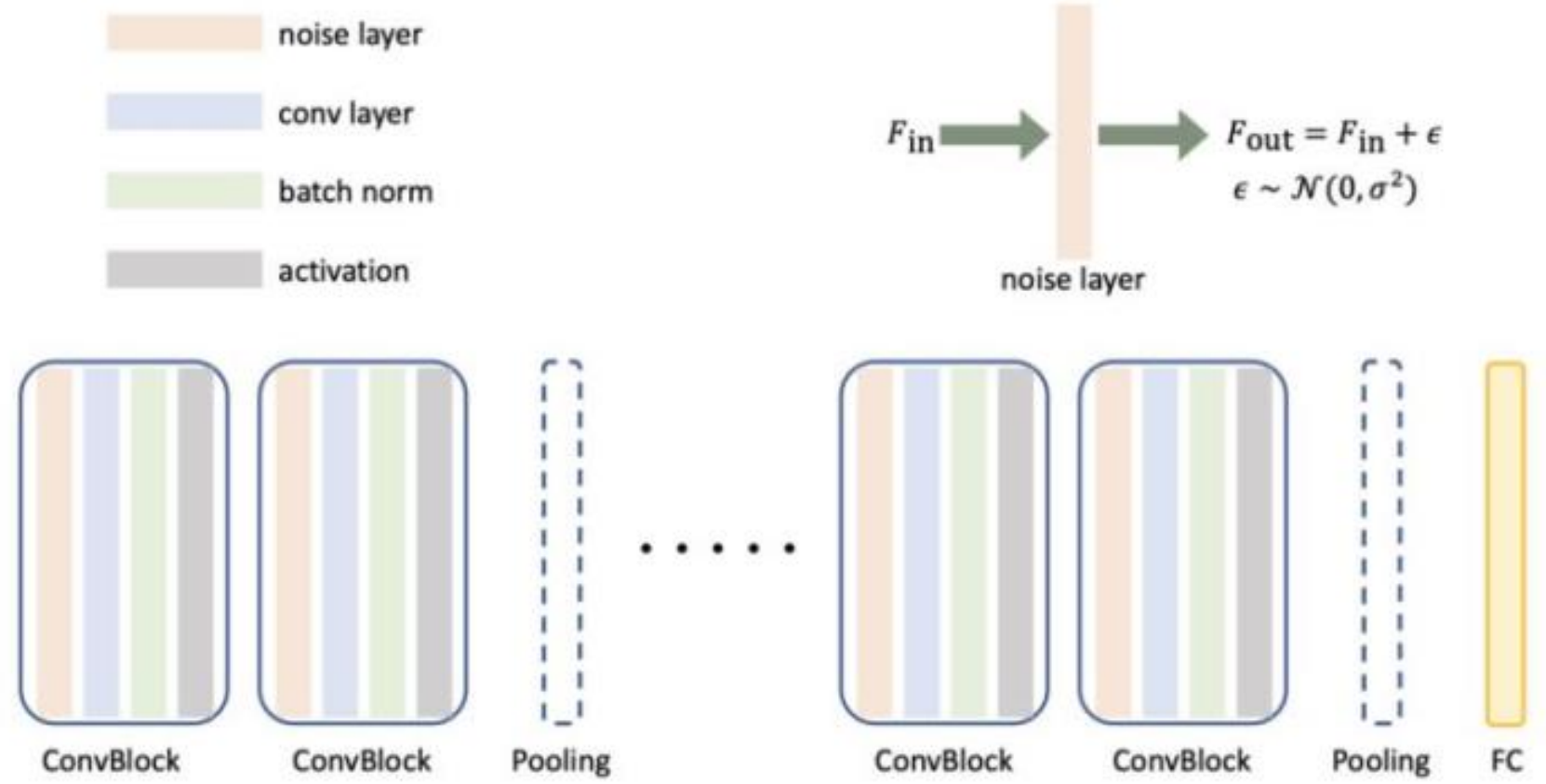
- In Dhillon's approach, during inference, random nodes in the DNN models are dropped.



Dhillon et al. "Stochastic activation pruning for robust adversarial defense." ICLR 2018

Stochastic Gradients: another example

- Liu's approach adds noise layers before the convolution layers to perturb the inputs to the CNN layers.



Liu, Xuanqing, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. "Towards robust neural networks via random self-ensemble." In ECCV, pp. 369-385. 2018

Exploding and Vanishing Gradients

- Exploding & Vanishing Gradients are often caused by defenses that consist of multiple iterations of neural network evaluation, feeding the output of one computation as the input of the next. This type of computation, when unrolled, can be viewed as an extremely deep neural network evaluation, which can cause vanishing/exploding gradients.

Defenses based on robust optimization

- Adversarial training:

$$\min_{\theta} \mathbb{E}_{x,y\sim D}\left[\max_{\delta\in\Delta} loss(\theta, x+\boldsymbol{\delta}, y)\right]$$

finding a robust model

finding a worst-case perturbation

FGSM adversarial training

$$\tilde{J}(\theta, x, y) = c\, J(\theta, x, y) + (1 - c)\, J(\theta, \underbrace{x + \epsilon\, \mathrm{Sign}[\nabla_x J(\theta, x, y)]}_{x'}, y)$$

$$\tilde{J}(\theta, x, y) = c\, J(\theta, x, y) + (1 - c)\, J(\theta, x', y) \qquad 0 \le c \le 1$$

Best paper of ICML 2018

Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples

Anish Athalye [*1] **Nicholas Carlini** [*2] **David Wagner** [2]

CW

Gradient Masking / Obfuscation

- A defense is said to cause gradient masking if it “does not have useful gradients” for generating adversarial examples (Papernot et al., 2017)

- Shattered Gradients
- Stochastic / Randomized gradients
- Exploding and vanishing gradients

Failure of defense methods in ICLR2018

Defense	Dataset	Distance	Accuracy
Buckman et al. (2018)	CIFAR	0.031 (ℓ_∞)	0%*
Ma et al. (2018)	CIFAR	0.031 (ℓ_∞)	5%
Guo et al. (2018)	ImageNet	0.005 (ℓ_2)	0%*
Dhillon et al. (2018)	CIFAR	0.031 (ℓ_∞)	0%
Xie et al. (2018)	ImageNet	0.031 (ℓ_∞)	0%*
Song et al. (2018)	CIFAR	0.031 (ℓ_∞)	9%*
Samangouei et al. (2018)	MNIST	0.005 (ℓ_2)	55%**
Madry et al. (2018)	CIFAR	0.031 (ℓ_∞)	47%
Na et al. (2018)	CIFAR	0.015 (ℓ_∞)	15%

Table from: Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples, 2018 (Best paper award ICML 2018)