$x + \delta$ $\delta \sim \mathcal{N}(0, \sigma^2 I)$

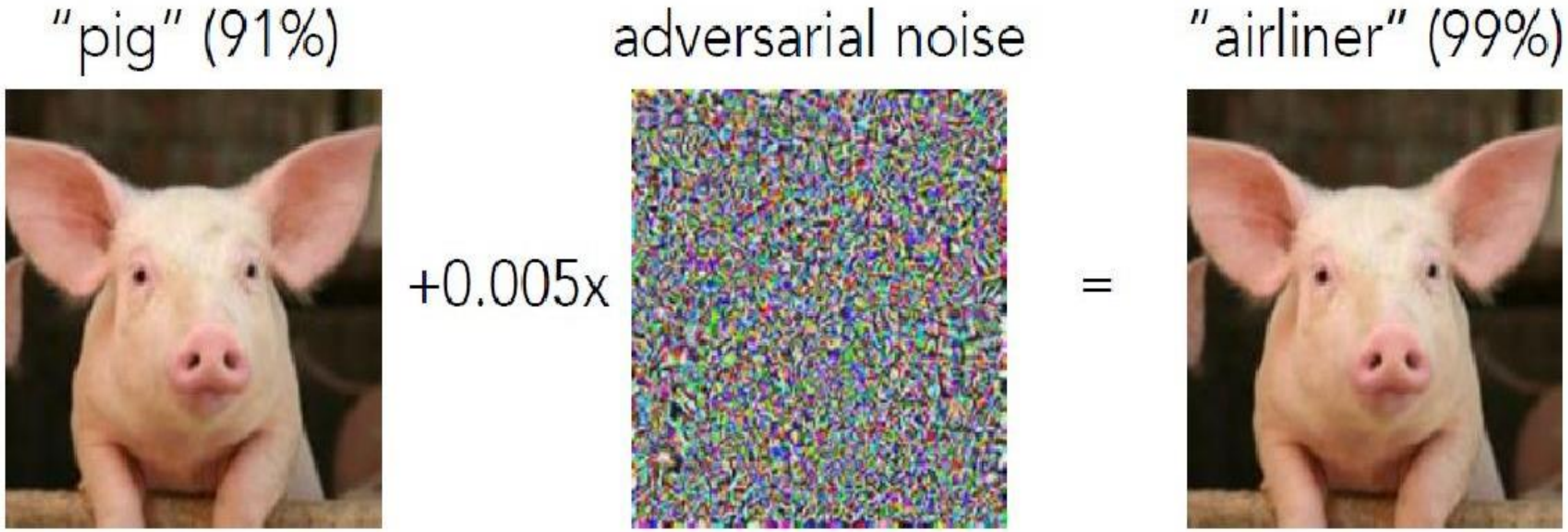
$x + \delta_{adv}$

Robustness: Attacks and Defenses

Mostafa Tavassolipour

Robustness

- Robust training
  - Train a model so that small changes in input do not change the classifier output



Neighborhood Metric

$x \longrightarrow x'$

$\boxed{f(x) \neq f(x')}$

- ℓ_p norms:

$$\|x\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{1/p}$$

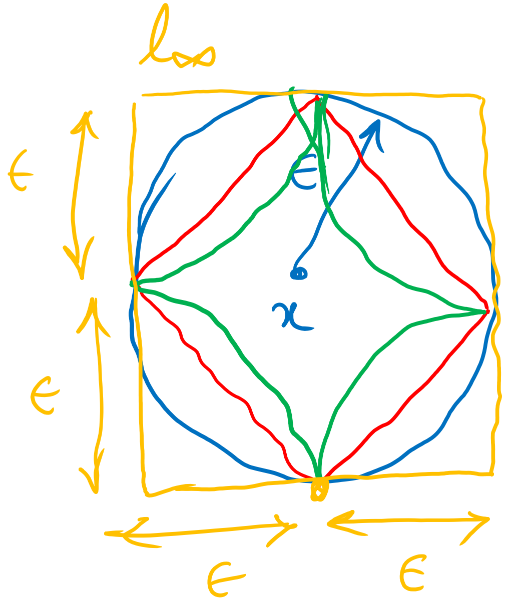
ℓ_0 $\|x\|_0$: تعداد درایه‌های غیر صفر

ℓ_1 $\|x\|_1$

ℓ_2 $\|x\|_2$

ℓ_∞ $\|x\|_\infty = \max(|x_1|, |x_2|, \cdots, |x_n|)$

$\|x - x'\|_p \leqslant \epsilon$



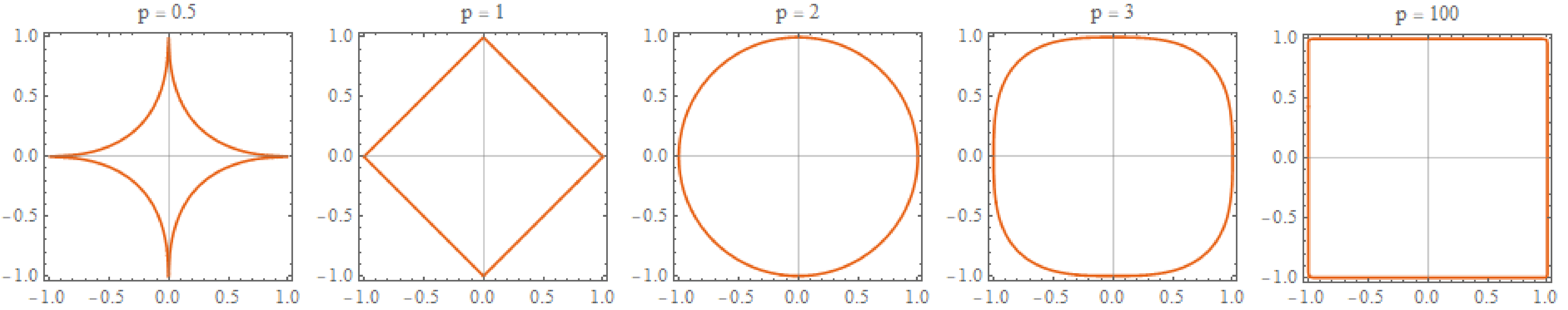
$\|x - x'\|_2 \leq \epsilon$

$\|x - x'\|_1 \leq \epsilon$

$\|x - x'\|_p \leq \epsilon$

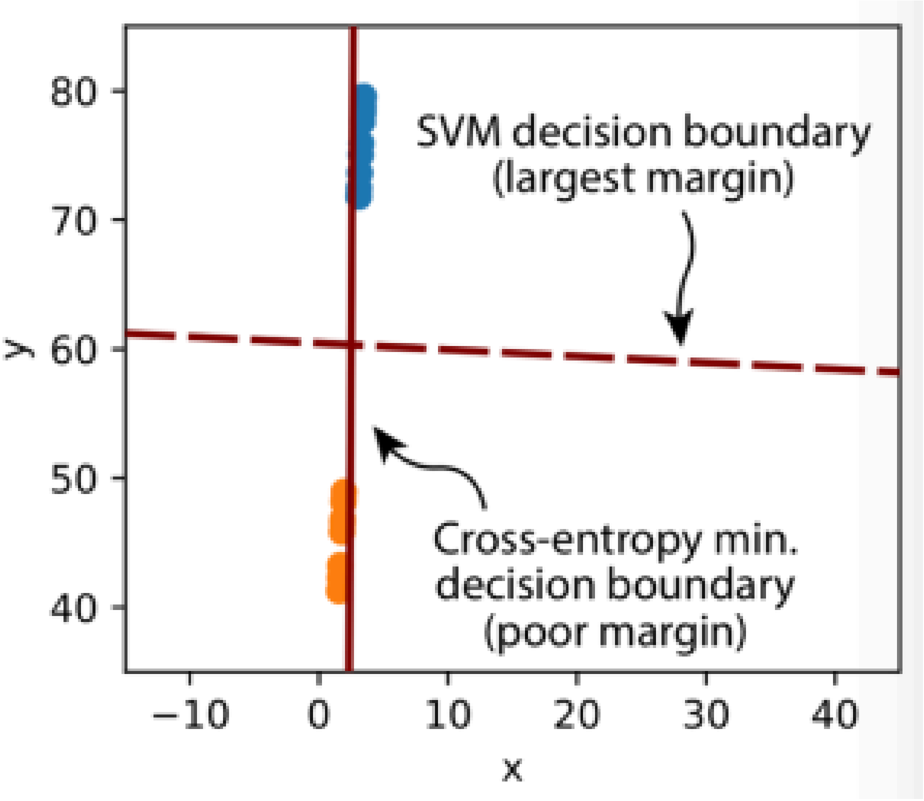
۰ <p< ۱

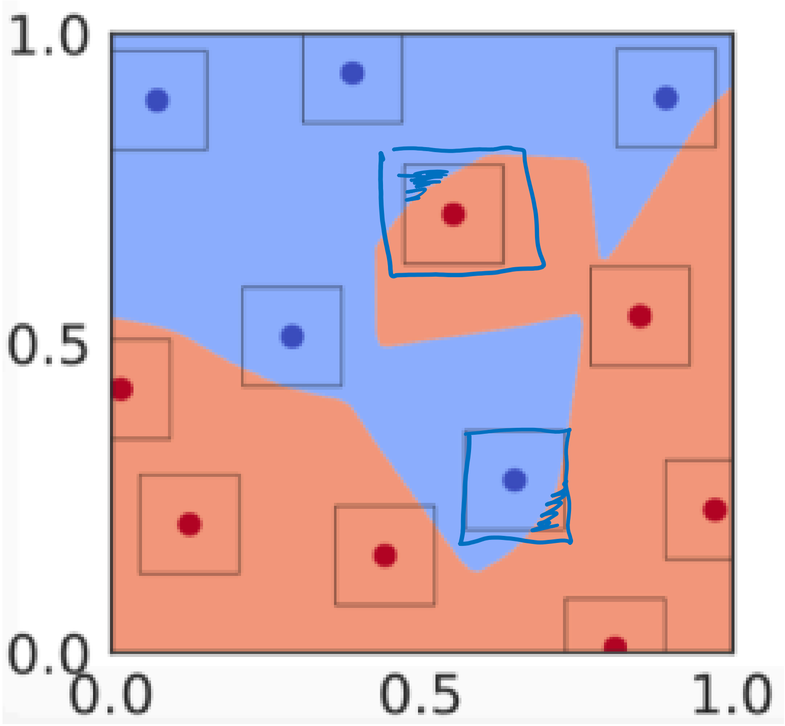
$\boxed{\|x - x'\|_\infty \leq \epsilon}$


p = 0.5
p = 1
p = 2
p = 3
p = 100
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

Why are there adversarial examples?

l_∞

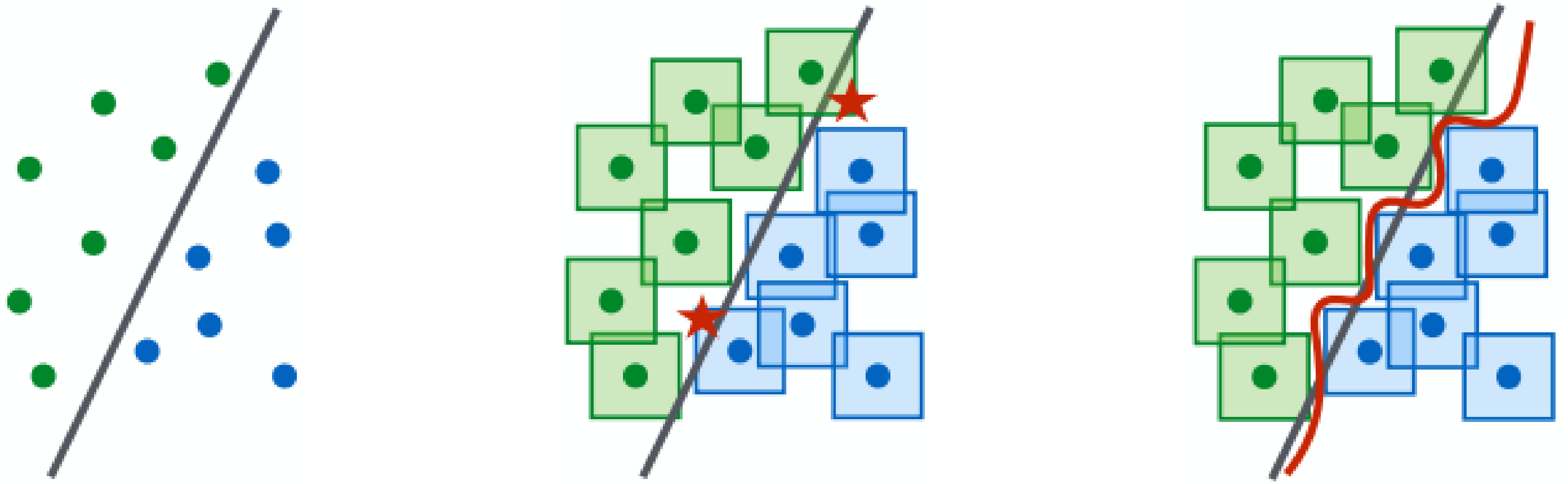




Linear Case

Non-Linear Case

Robustness and Accuracy tradeoff



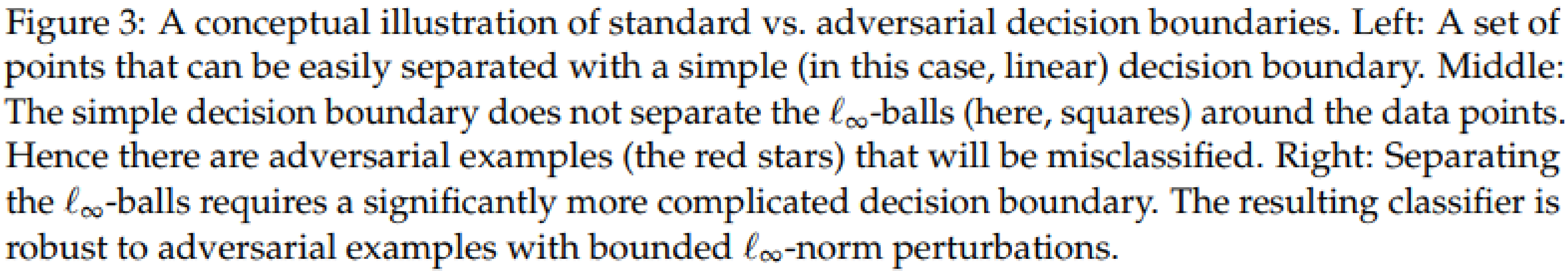


Figure 3: A conceptual illustration of standard vs. adversarial decision boundaries. Left: A set of points that can be easily separated with a simple (in this case, linear) decision boundary. Middle: The simple decision boundary does not separate the ℓ_∞-balls (here, squares) around the data points. Hence there are adversarial examples (the red stars) that will be misclassified. Right: Separating the ℓ_∞-balls requires a significantly more complicated decision boundary. The resulting classifier is robust to adversarial examples with bounded ℓ_∞-norm perturbations.

Figure from: Towards Deep Learning Models Resistant to Adversarial Attacks, 2019

Robustness and Accuracy tradeoff

certified

Robustness

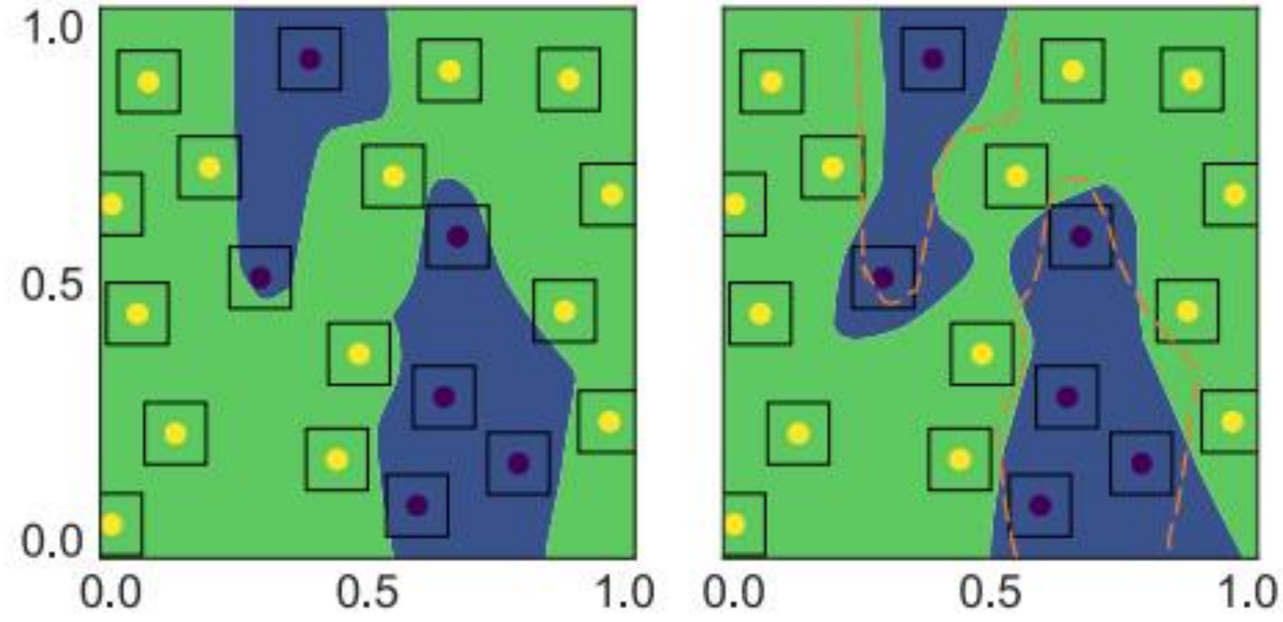


Figure 1: **Left figure:** decision boundary learned by natural training method. **Right figure:** decision boundary learned by our adversarial training method, where the orange dotted line represents the decision boundary in the left figure. It shows that both methods achieve zero natural training error, while our adversarial training method achieves better robust training error than the natural training method.

Figure form: Theoretically Principled Trade-off between Robustness and Accuracy, 2019

Attacks

$x \rightarrow y$

$x' \rightarrow y'$

- Targeted
- Non-targeted untargeted

$x \rightarrow y$

$x' \rightarrow \neq y$

$y = \overset{\text{classifier}}{f}(x)$

- Knowledge about model:
  - White-box
  - Gray-box
  - Black-box

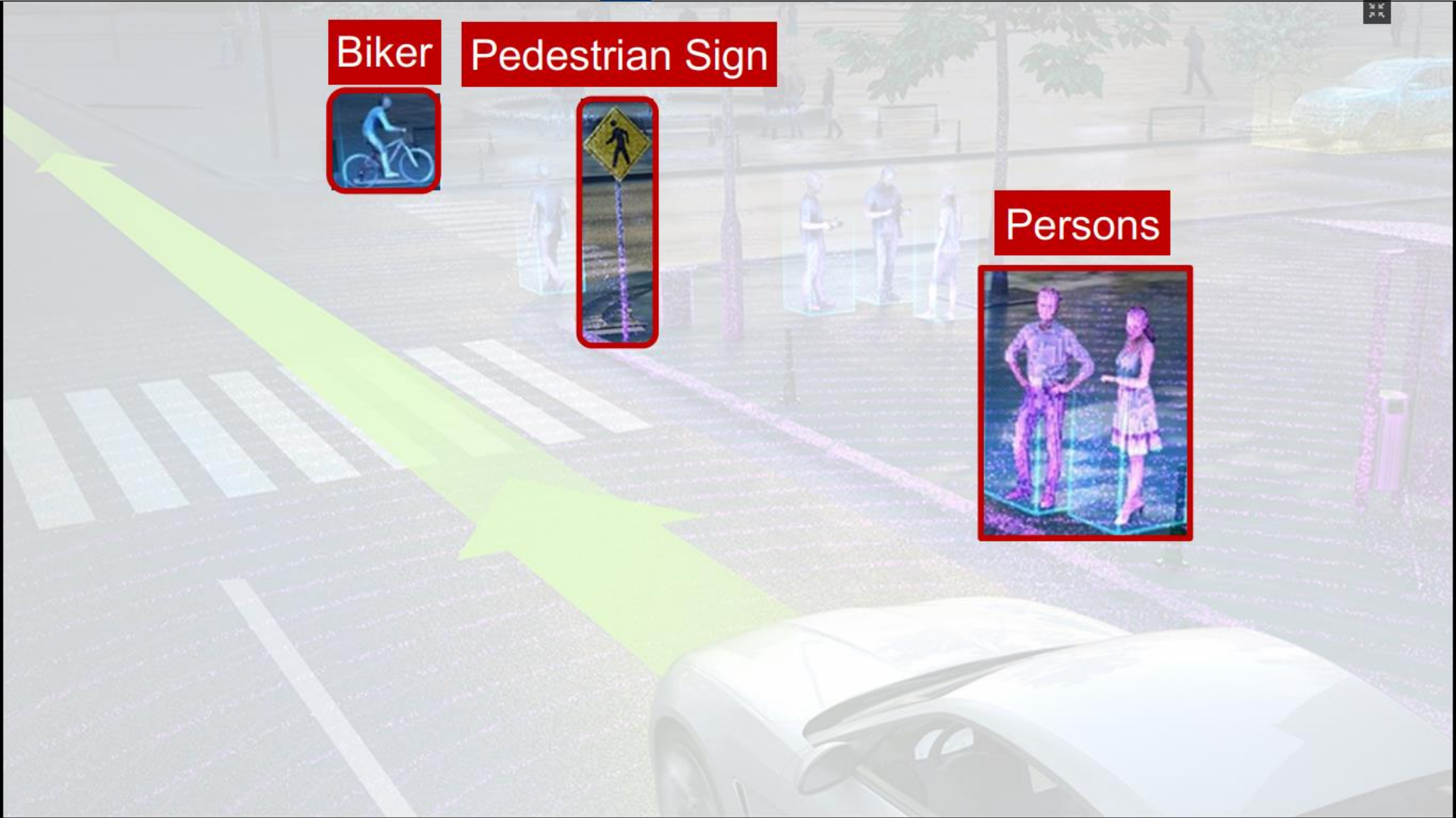
Targeted vs. Non-targeted

- Non-targeted attack
  - The goal is to fool the classifier for an adversarial input to output any label other than the ground-truth label
  - E.g., perturb an image of panda, so that the model predicts it is any other class than a panda

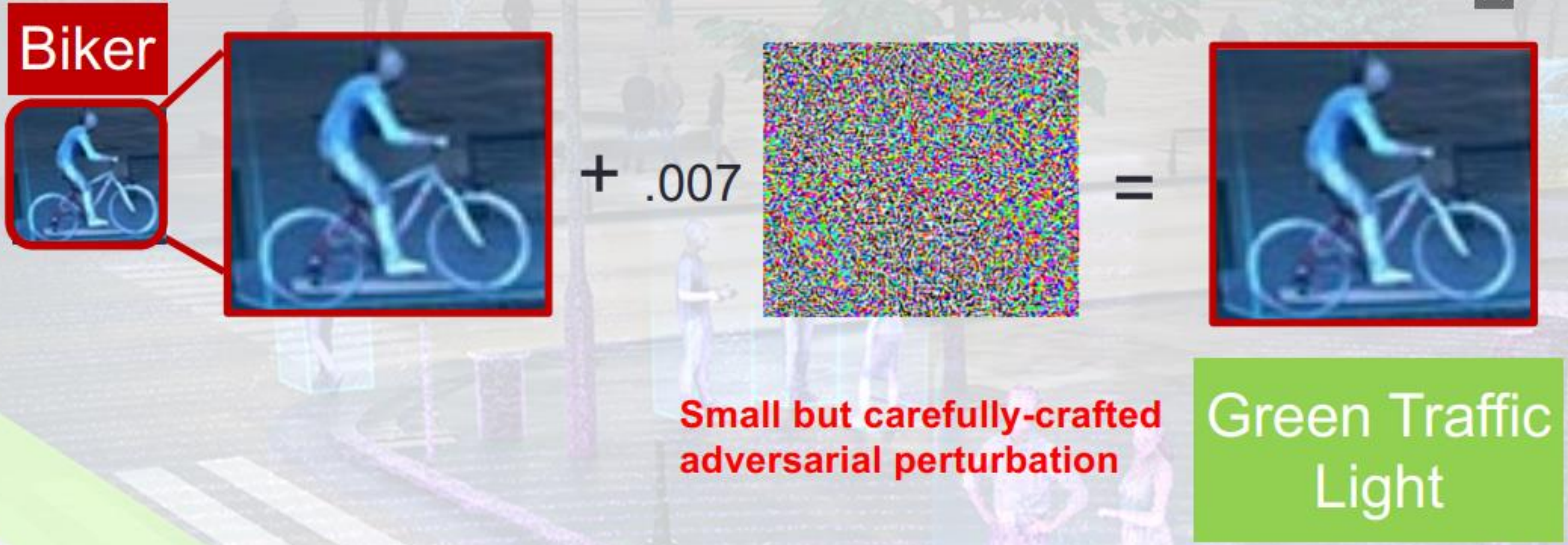
- Targeted attack
  - The goal is to fool the classifier to predict a target label for an adversarial input
  - More difficult, in comparison to non-targeted attack
  - E.g., perturb an image of a turtle, so that the model predicts it is a rifle
  - E.g., perturb an image of a STOP sign, so that the model predicts it is a Speed Limit 45 sign

White-box vs. Black-box Attacks

- ***White-box attack***
  - Attackers have full knowledge about the ML model
  - I.e., they have access to weights, hyper-parameters, gradients, architecture, etc.
- ***Black-box attack***
  - Attackers don’t have access to the ML model weights, gradients, architecture
  - Attackers may query the black-box model to obtain knowledge about the model. E.g., submit adversarial examples, and obtain the model’s output (class label)
  - A body of work has focused on query-efficient black-box attacks, where the goal is to generate adversarial examples using a limited number of queries
  - Black-box attacks are more realistic, because model designers usually do not open source the model parameters
- ***Gray-box attack***
  - Perhaps they have some knowledge about the used ML model
  - E.g., attackers may know that a ResNet50 model is used for classification, but they don’t have access to the model weights



Biker
Pedestrian Sign
Persons



Biker

+ .007

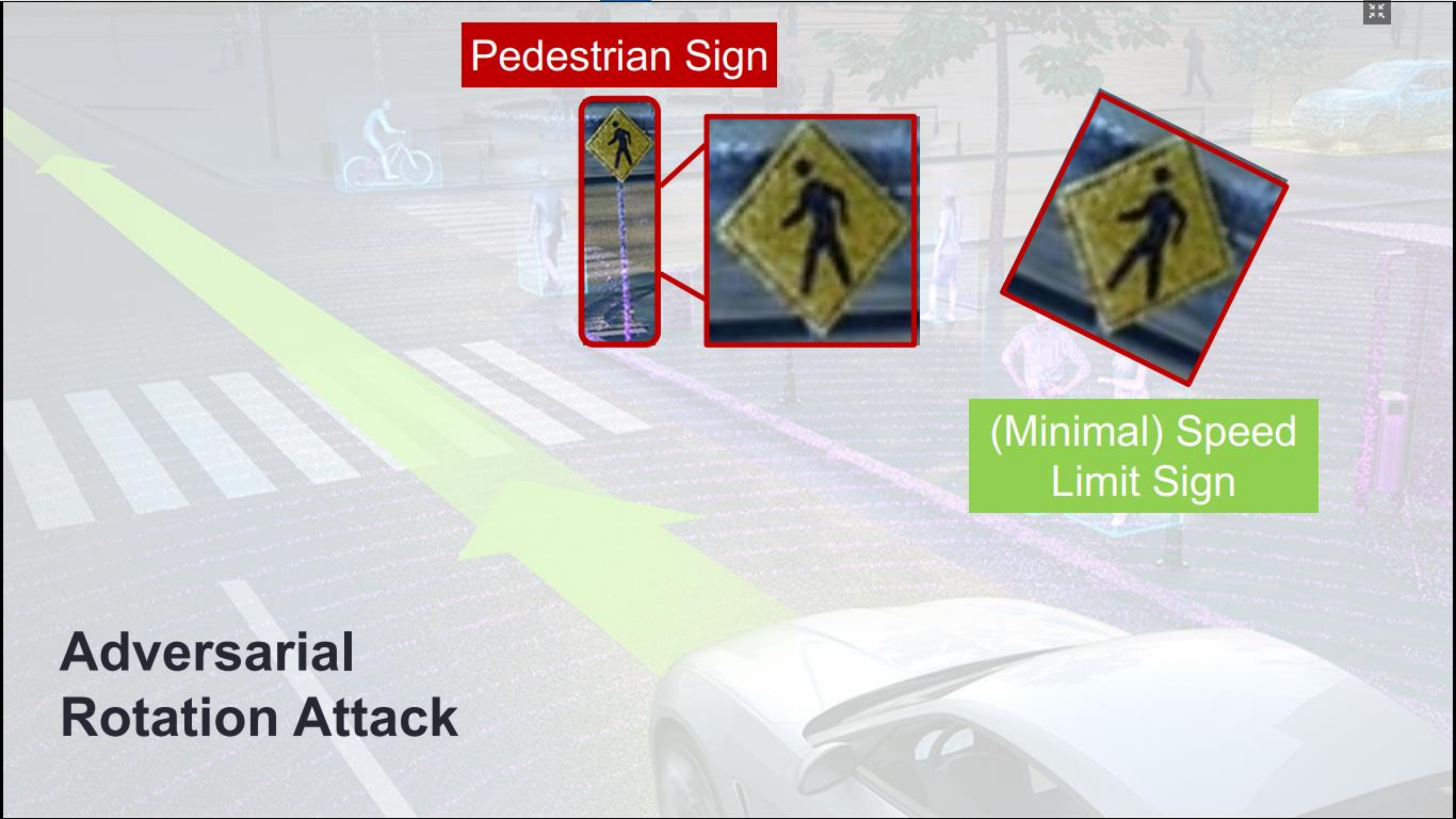
=

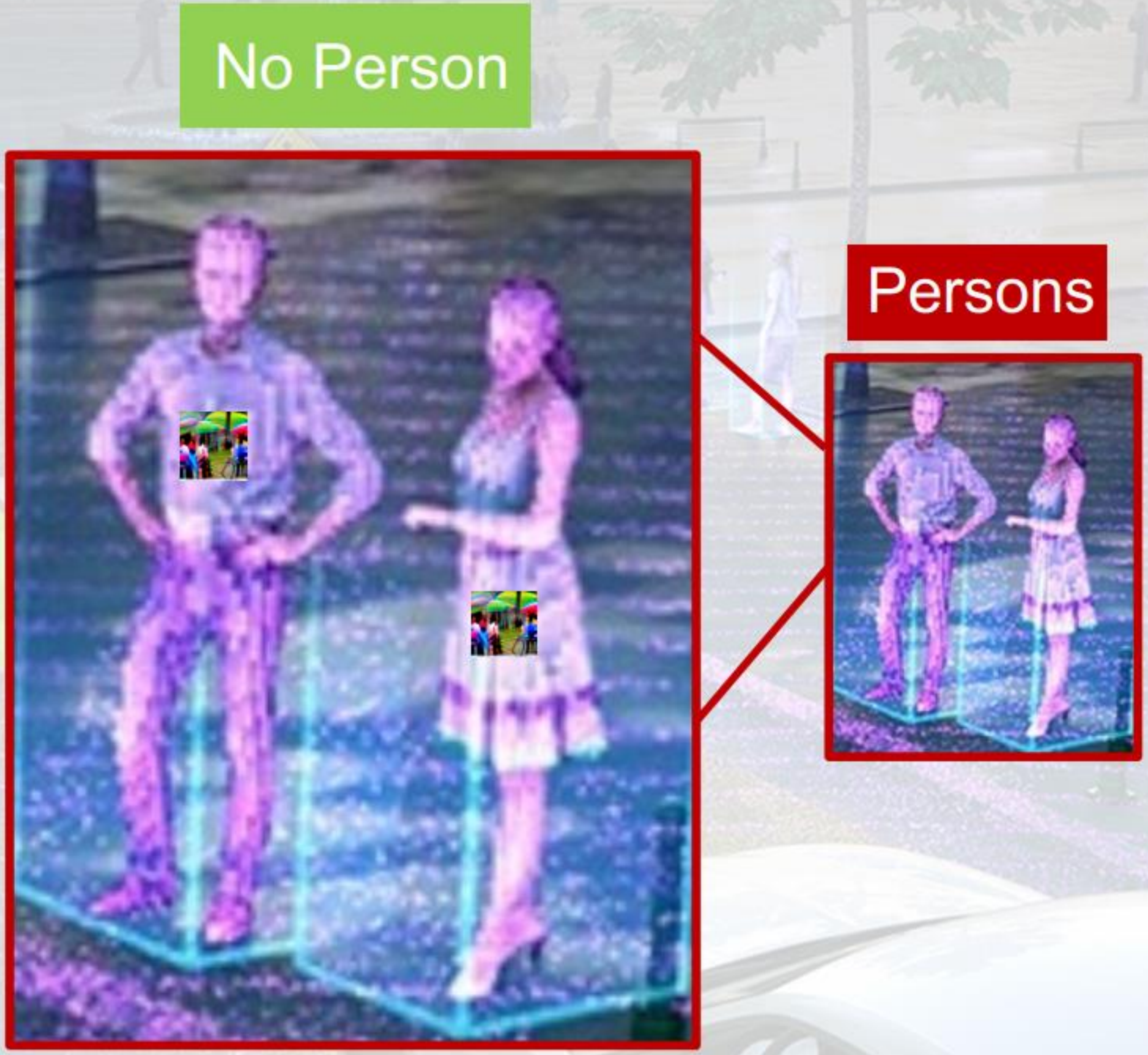
Small but carefully-crafted adversarial perturbation

Green Traffic Light

Adversarial Perturbation Attack

Adversarial Rotation Attack





Adversarial Patch Attack

Adversarial Attacks

- L-BFGS algorithm
- Fast gradient sign method (FGSM)
- BIM and PGD
- Carlini and Wagner (CW)
- Adversarial patch
- GAN-based attacks
- …

$\underbrace{\text{L-BFGS}}_{\text{optimization}}$ algorithm

$$\min_{x'} \|x - x'\|_p$$

$$\min_{x'} \|x - x'\|_p$$

$$s.t.\ \ f(x') = y'$$
$$y' \neq y$$

$$s.t.\ \ f(x') \neq y$$

non-targeted

- Relaxed version:

$$\min_{x'} c\|x - x'\|_p + \underbrace{\boxed{J(\theta, x', y')}}_{\text{Cross-entropy}}$$

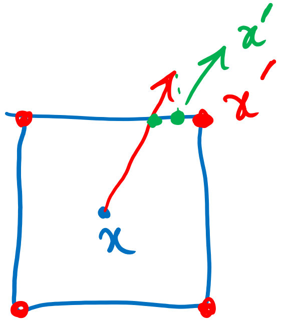
c is a hyperparameter.

$$\min_{x'} c\|x - x'\|_p - J(\theta, x', y)$$

Fast gradient sign method (FGSM)

ℓ_∞ ϵ

- Non-targeted Attack

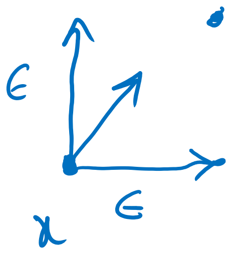


$$x' = x + \epsilon \, \underbrace{\mathrm{sign}[\nabla_x \underbrace{J(\theta, x, y)}]}$$

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}$$

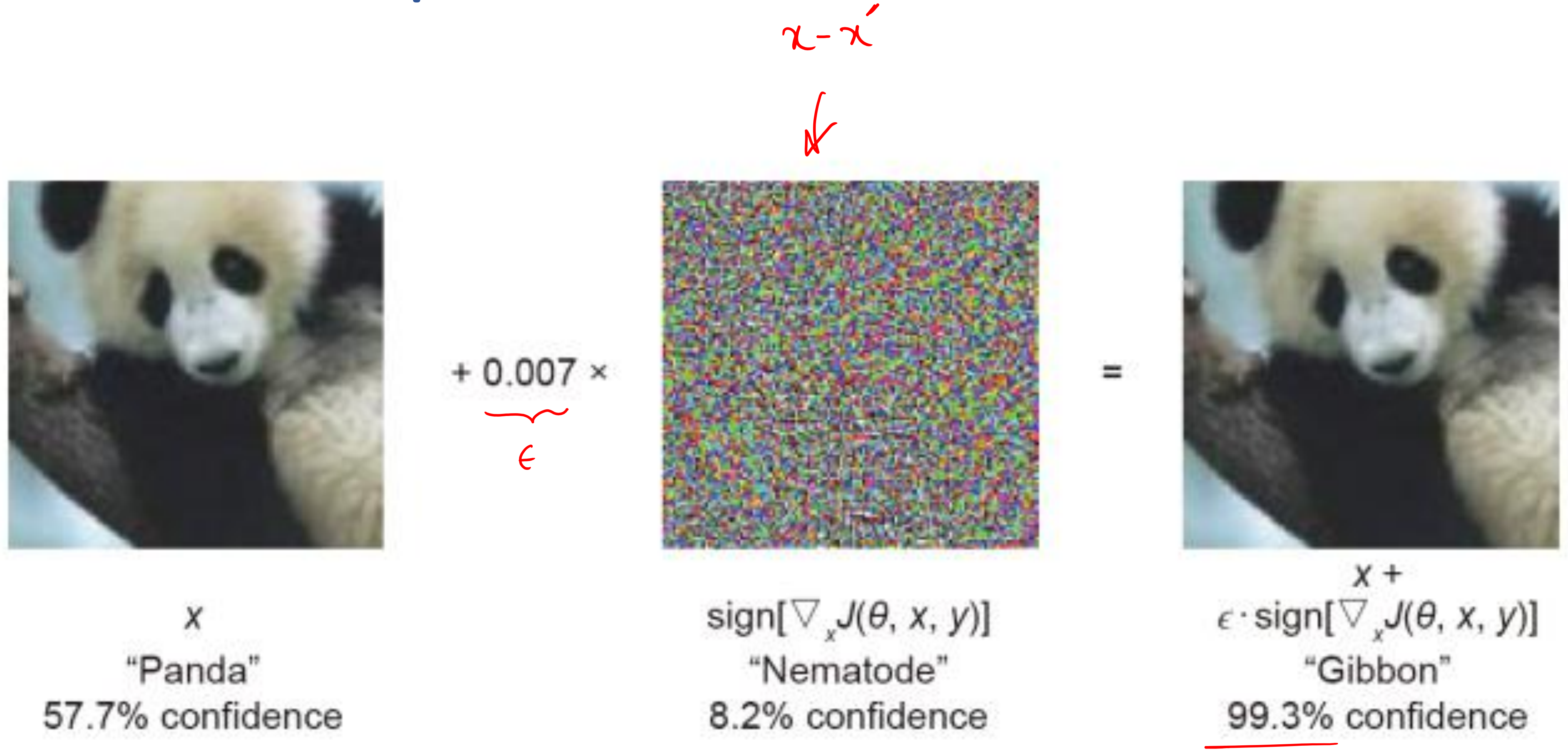
$$x' = \begin{bmatrix} x_1 \pm \epsilon \\ x_2 \pm \epsilon \\ \vdots \\ x_d \pm \epsilon \end{bmatrix}$$

- Targeted Attack:



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

FGSM example


$x - x'$
x
"Panda"
57.7% confidence
$+ \underbrace{0.007}_{\epsilon} \times$
$\text{sign}[\nabla_x J(\theta, x, y)]$
"Nematode"
8.2% confidence
=
$x + \epsilon \cdot \text{sign}[\nabla_x J(\theta, x, y)]$
"Gibbon"
99.3% confidence

Basic Iterative Method (BIM)

- BIM improves the performance of FGSM
- The BIM performs FGSM with a smaller step size and clips the updated adversarial sample into a valid range for T iterations

$$x'_{t+1} = \text{Clip}\,\{\underbrace{x'_t + \alpha\,\text{Sign}[\nabla_x J(\theta, x'_t, y)]}\}$$

where $\boxed{\alpha T = \epsilon}$

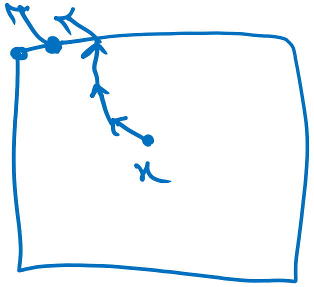
$$\boxed{\alpha = \frac{\epsilon}{T}}$$

$x'_0 = x$

x'_1

$\vdots$

$\boxed{x'_T}$



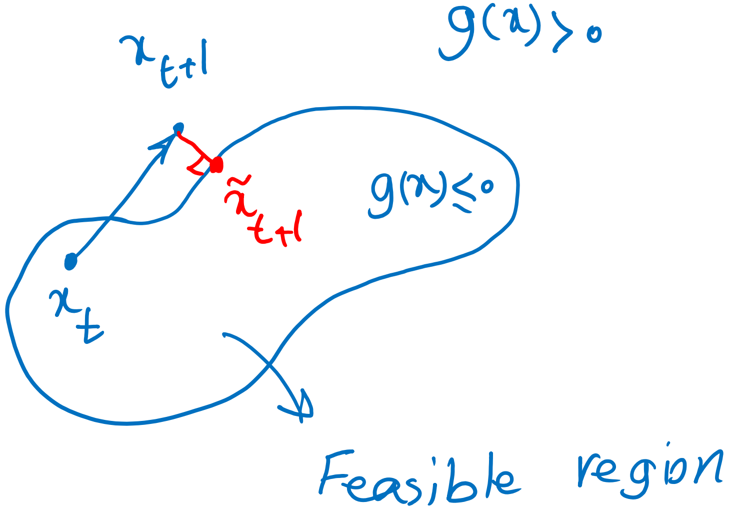
Projected Gradient Descent (PGD)

- PGD can be considered as a generalized version of BIM **without** the constraint $\boxed{\alpha T = \epsilon}$

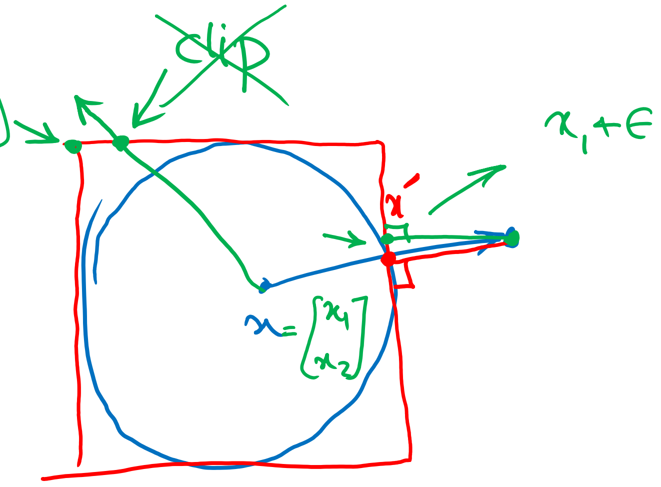
$$x'_{t+1} = \underline{\text{Proj}}\,\{x'_t + \alpha\ \text{Sign}[\nabla_x J(\theta, x'_t, y)]\}$$

$$\min_x \quad f(x)$$

s.t. $\boxed{g(x) \leq 0}$



$\boxed{\text{clip} = \text{proj}}$



Carlini and Wagner (CW)

$$\longrightarrow \min_{\delta} D(x, \overbrace{x+\delta}^{x'}) + c \overset{\downarrow}{f}(x+\delta)$$

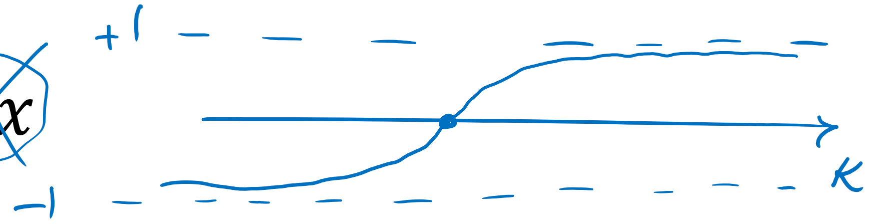
$$s.t. \ (x+\delta) \in [0,1]^n \longleftarrow$$

$$\boxed{\begin{array}{c} \boxed{F(x') = T} \\ \Updownarrow \\ f(x') \le 0 \end{array}}$$

$f(x+\delta) \le 0$ if and only if the DNN's prediction is the attack target.

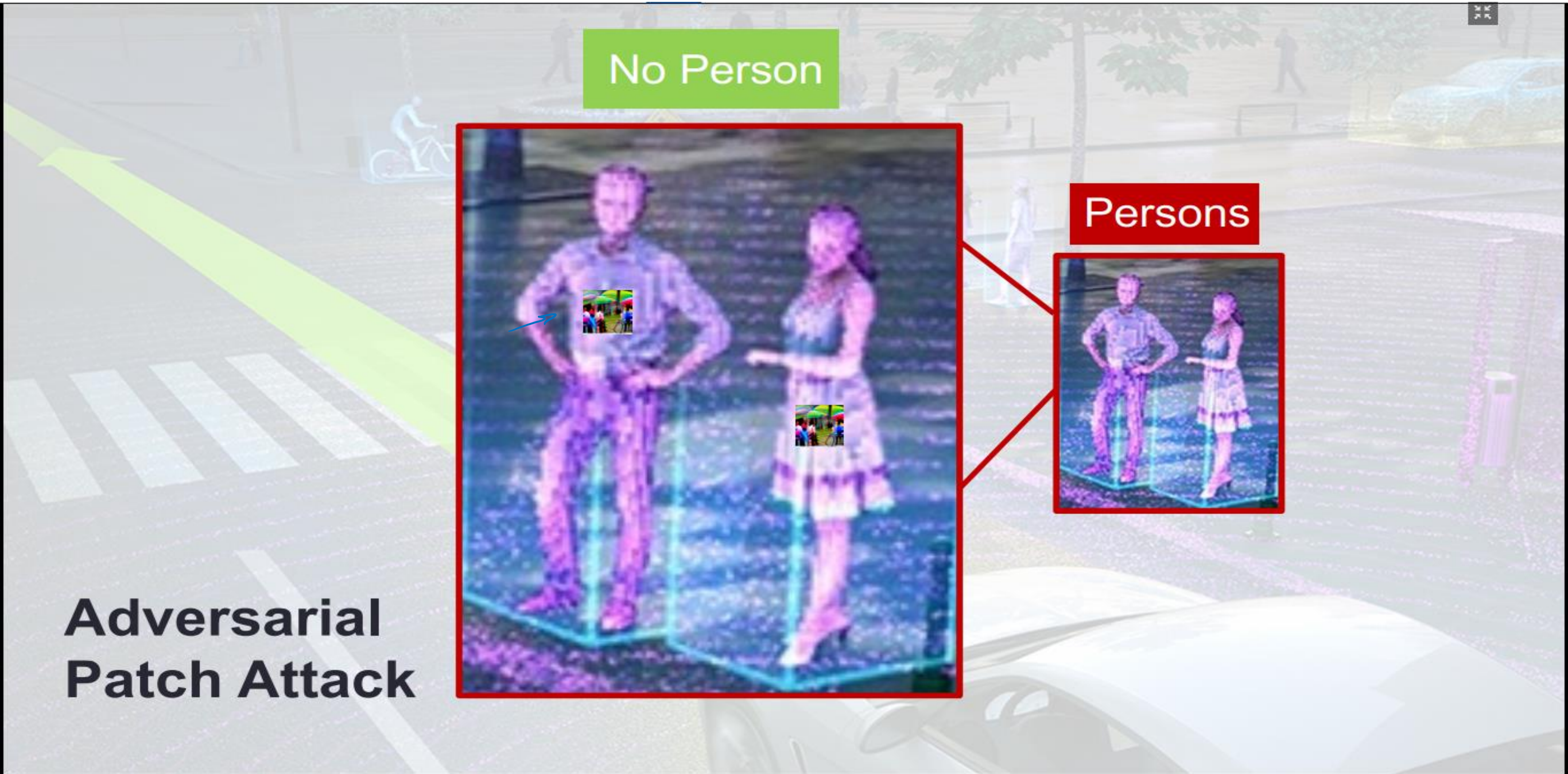
To ensure the constraint, they proposed:

$$x + \delta = \frac{1}{2}(\tanh(\underset{\uparrow}{\kappa}) + 1) \cancel{-x}$$



Towards Evaluating the Robustness of Neural Networks, 2017

Adversarial Patch



Adversarial Patch

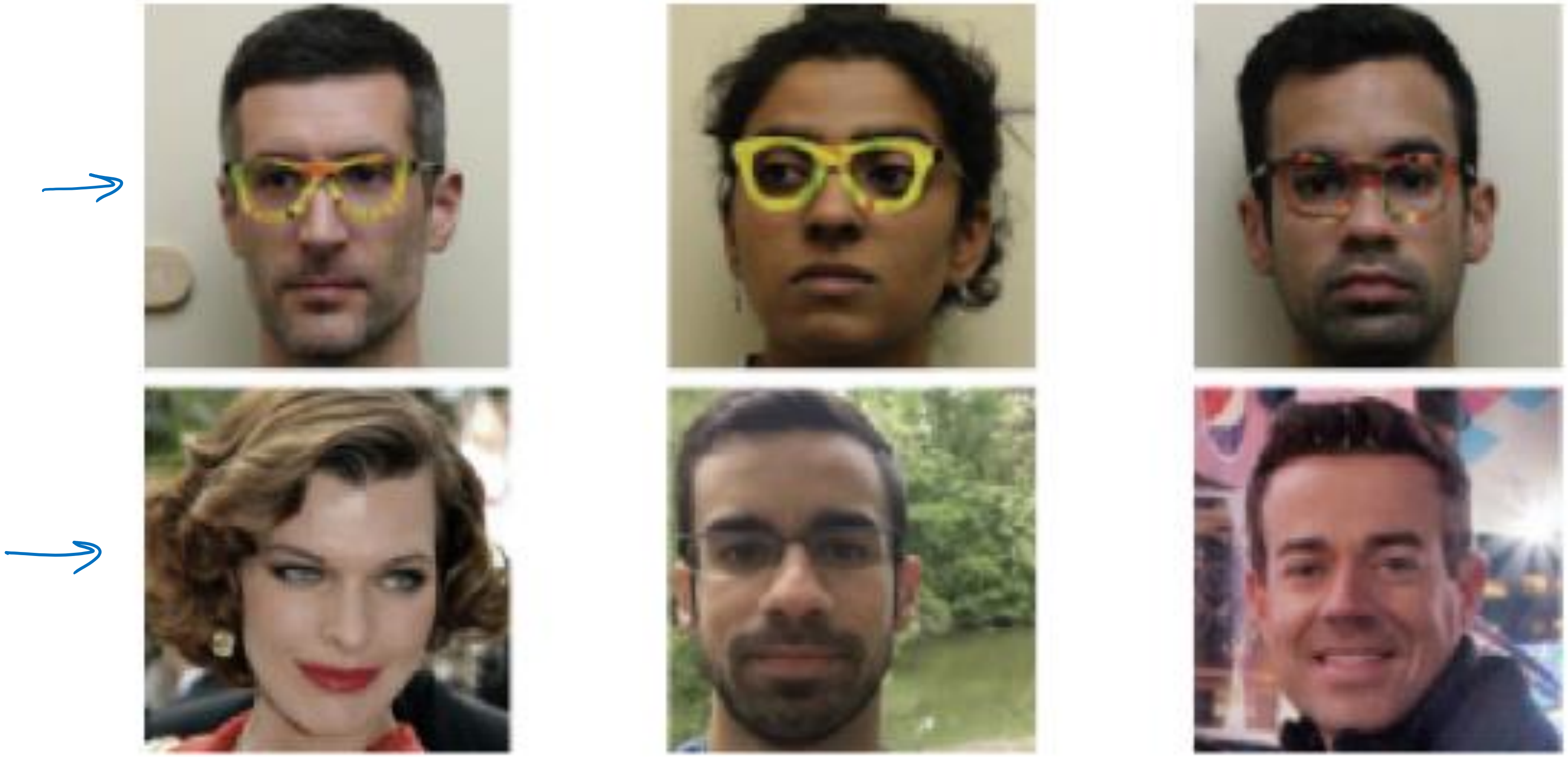
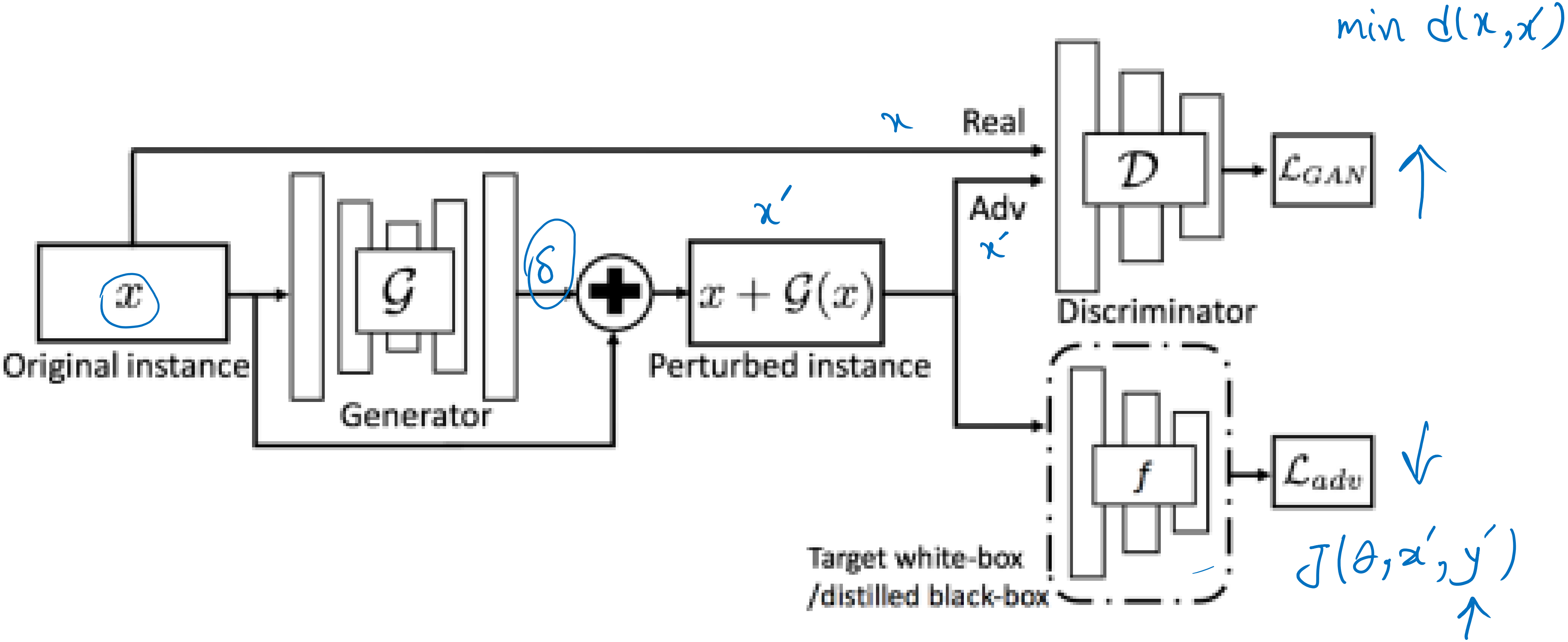


Fig. 4. Eyeglasses with adversarial perturbations deceive a facial recognition system to recognize the faces in the first row as those in the second row [25].

GAN-based attacks

- A generator is trained to learn the adversarial distribution by maximizing the target adversarial loss $J(\theta, x', y')$ and the GAN loss.



Paper: Generating Adversarial Examples with Adversarial Networks