Lecture 21: Adversarial Neural Networks

CS 109B, STAT 121B, AC 209B, CSE 109B

Mark Glickman and Pavlos Protopapas

Acknowledgments

Material taken from presentations given by:

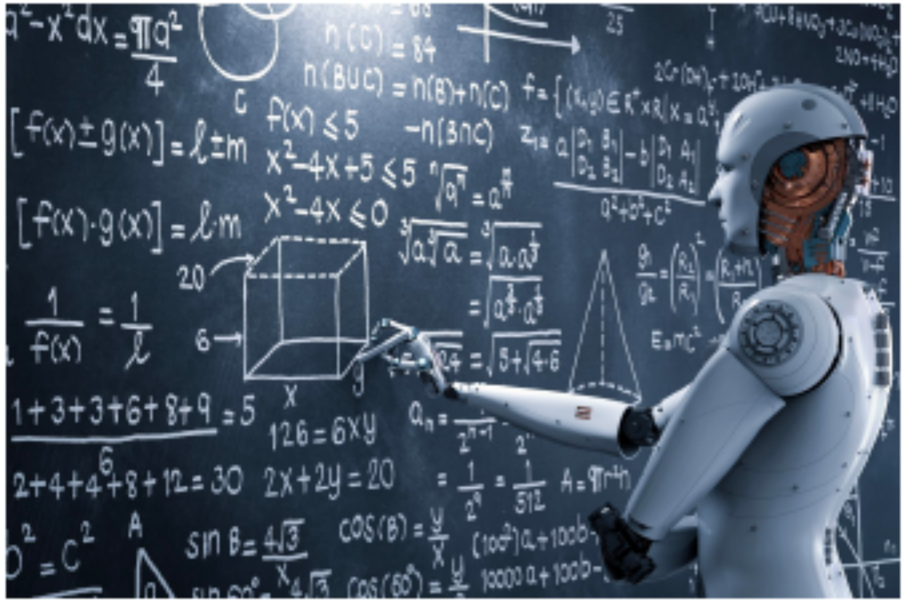
Amil Merchant, Alex Lin, Thomas Chang, ZiZi Zhang

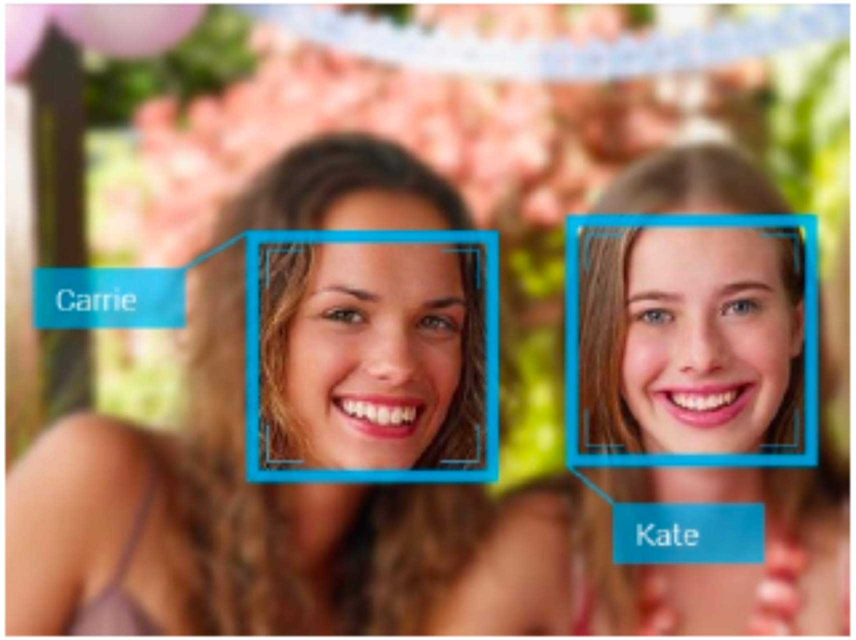
Harvard

Ekin Dogus Cubuk

Google Brains

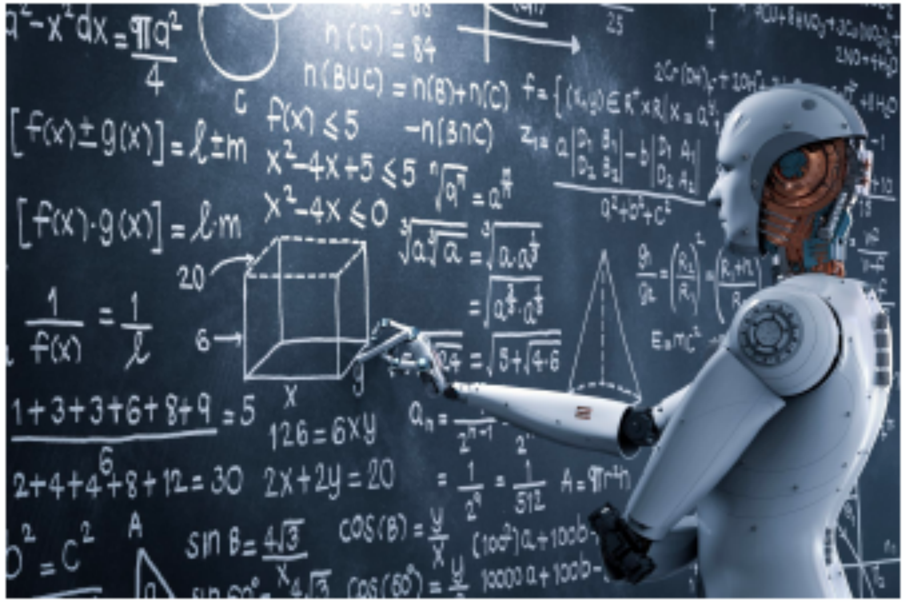
Deep Learning Impact



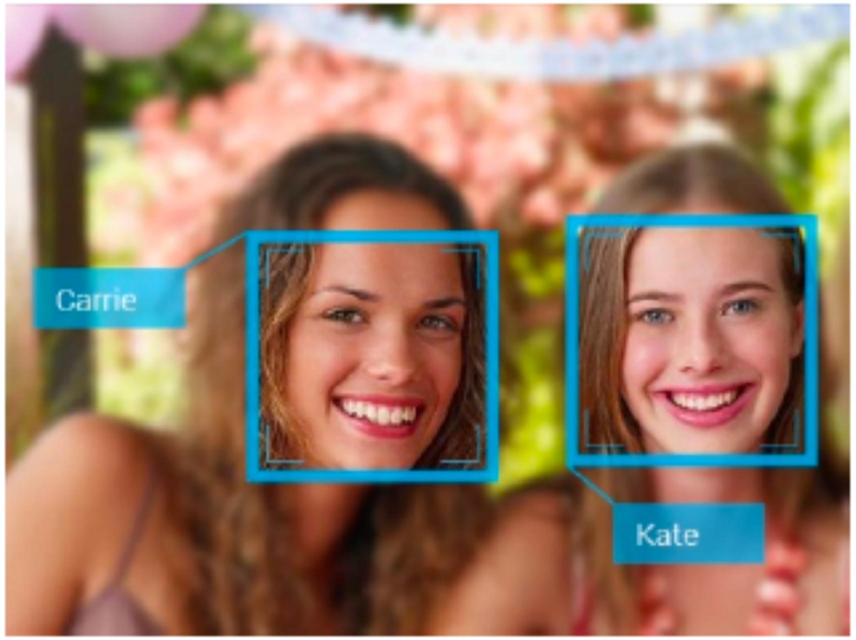


Deep Learning Impact

Overrated



Underrated



Cifar10 dataset

Fully connected) : 60-70%

Convolutional network: ~90-93%

Wide-Resnet: 96.1%

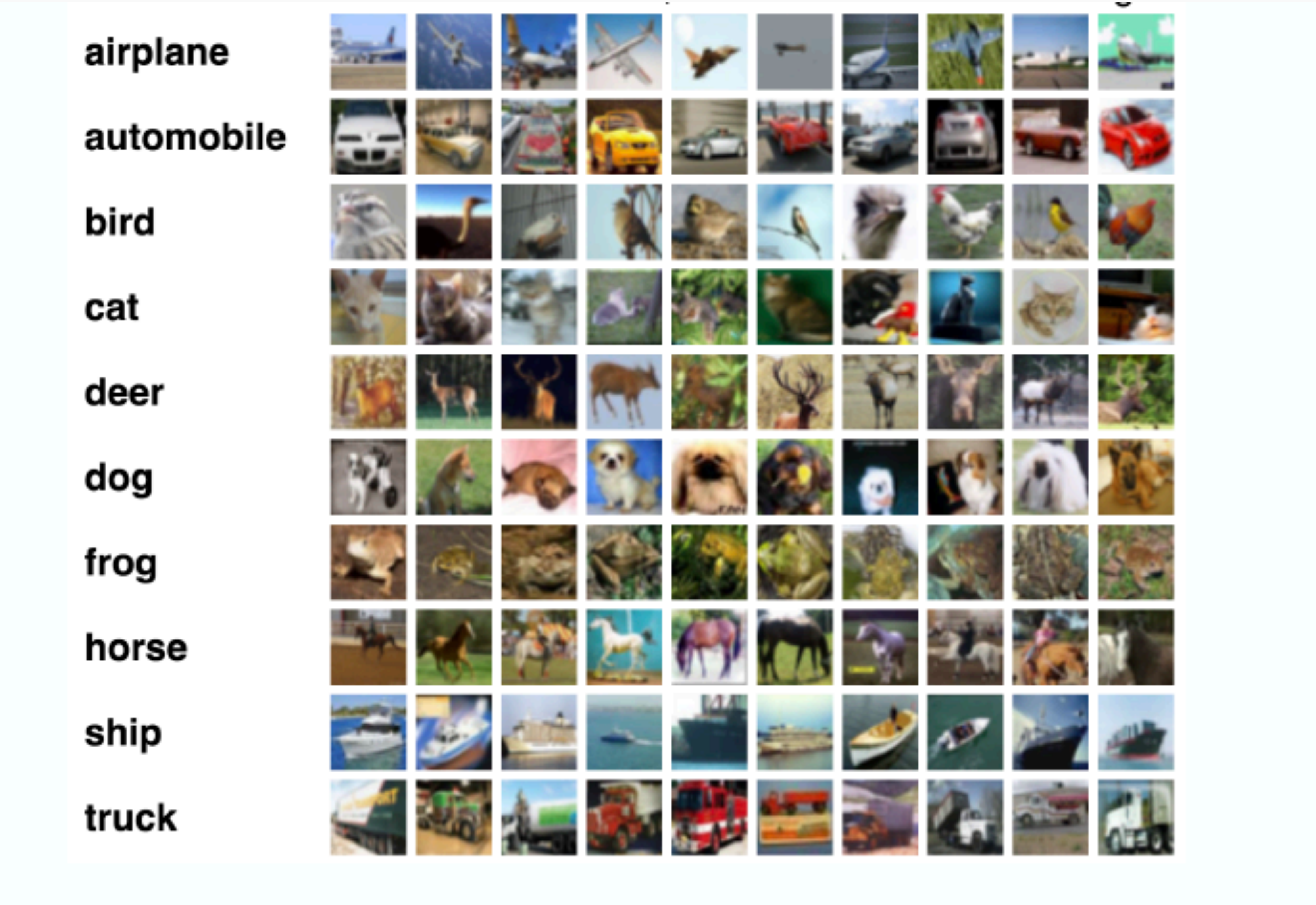
NASNet-A: 97.6%

These models are big!

50k training samples

32x32 pixels

>25M parameters



Deep learning success story: ImageNet competition

1.28M training samples ~300x300 pixels

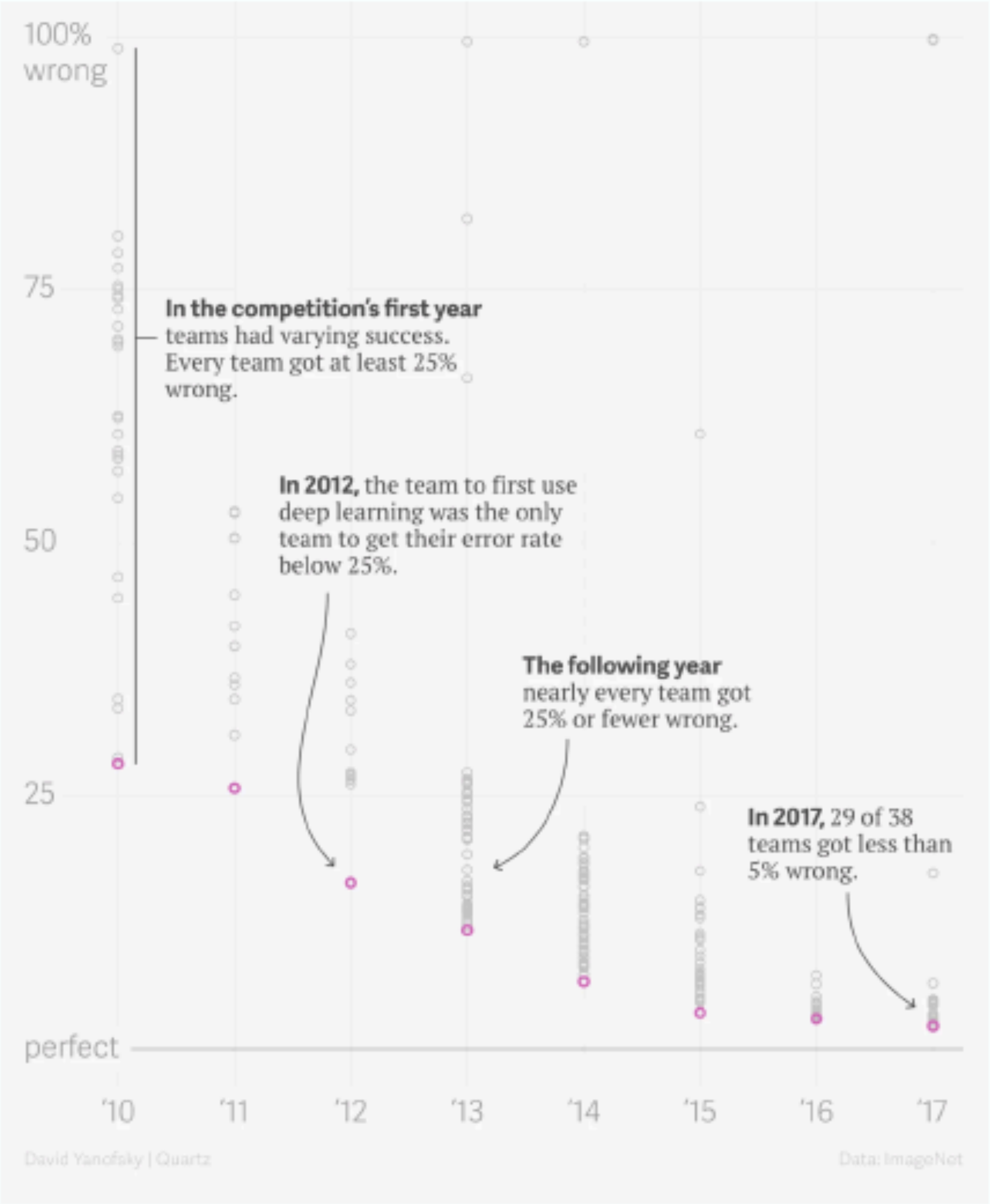
50k validation samples

Top-secret test set

Low prediction accuracy before deep learning. <50%

Still challenging. ~83%

Deep learning success story: ImageNet



Data augmentation is very helpful!

Random flip left-right:

Random shifts/crops:

Cutout / Random erasing:

Mixup / Pairing images:

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j$$

How vulnerable are Neural Networks?

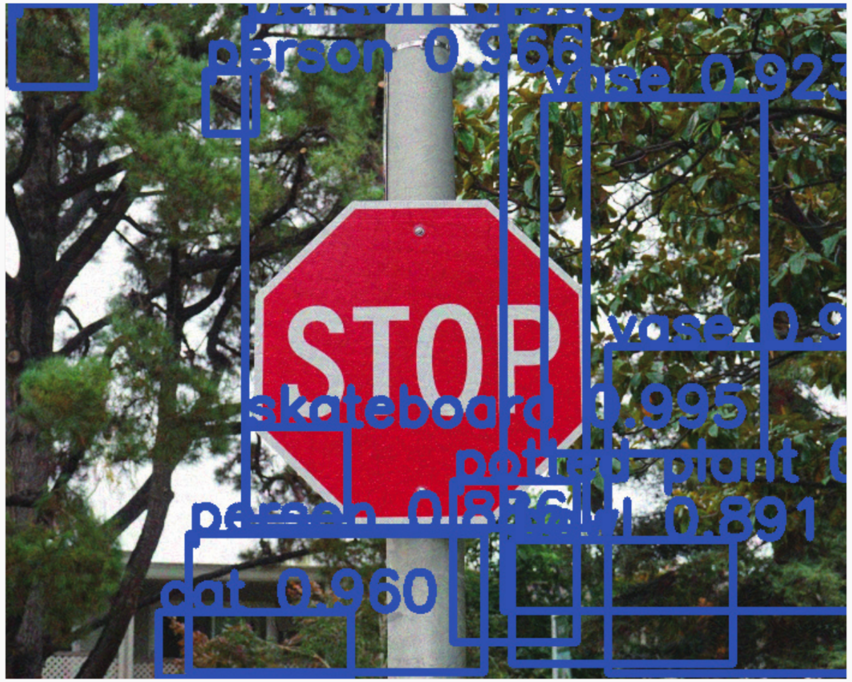
Uses of Neural Networks

How vulnerable are Neural Networks?

Original Image Detected

original

Whole Image Attacked



Small perturbations. No detection

Sign Region Attacked

Small perturbations. Vase was detected

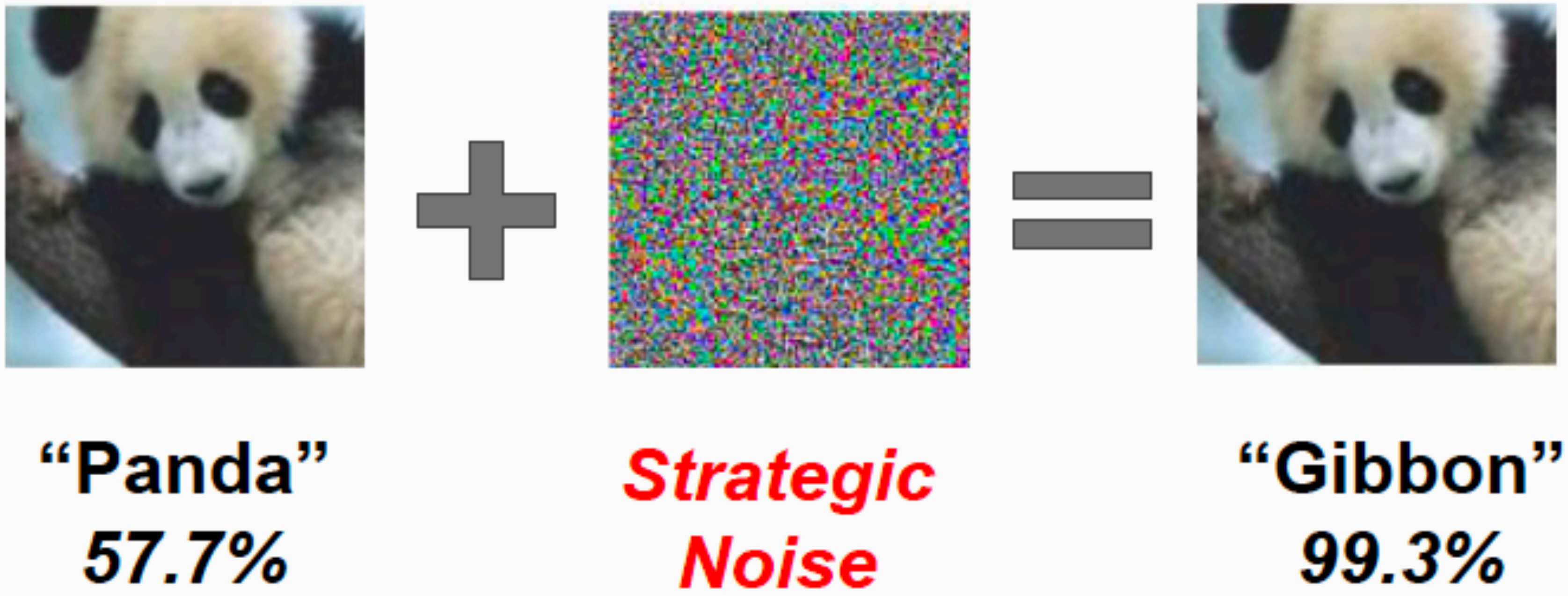
Jiajun Lu, Hussein Sibai, Evan Fabry University of Illinois at Urbana Champaign

How vulnerable are Neural Networks?

Explaining Adversarial Examples

[Goodfellow et. al ‘15]

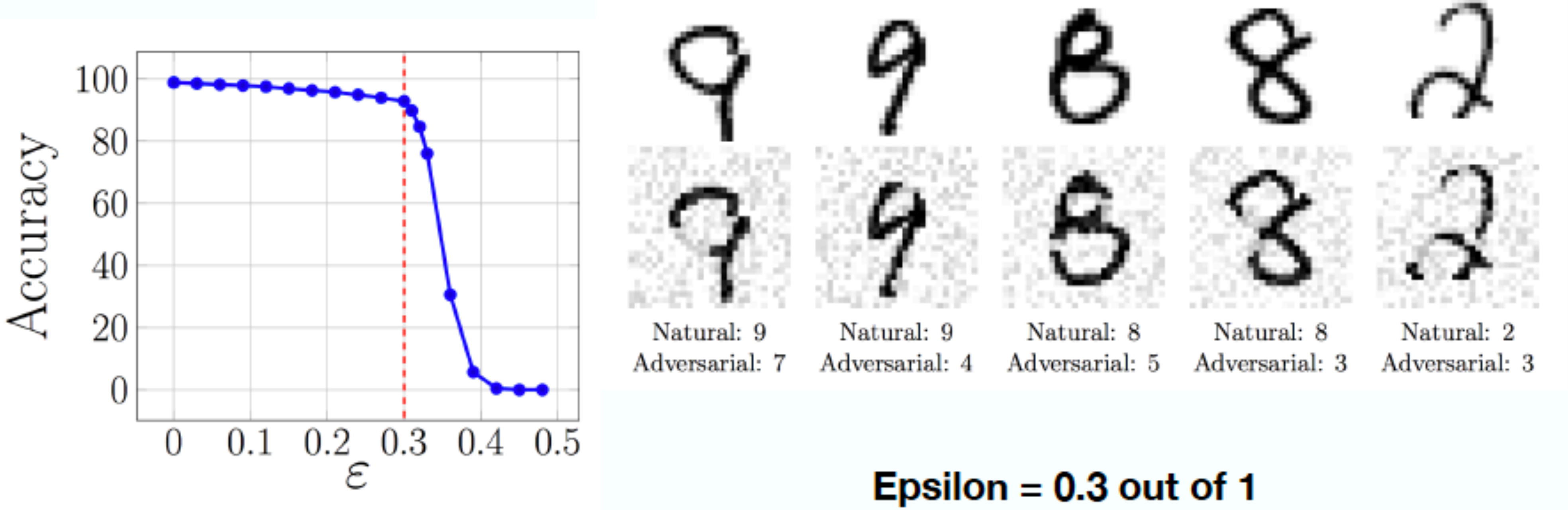
1. Robust attacks with FGSM
2. Robust defense with Adversarial Training



Explaining Adversarial Examples

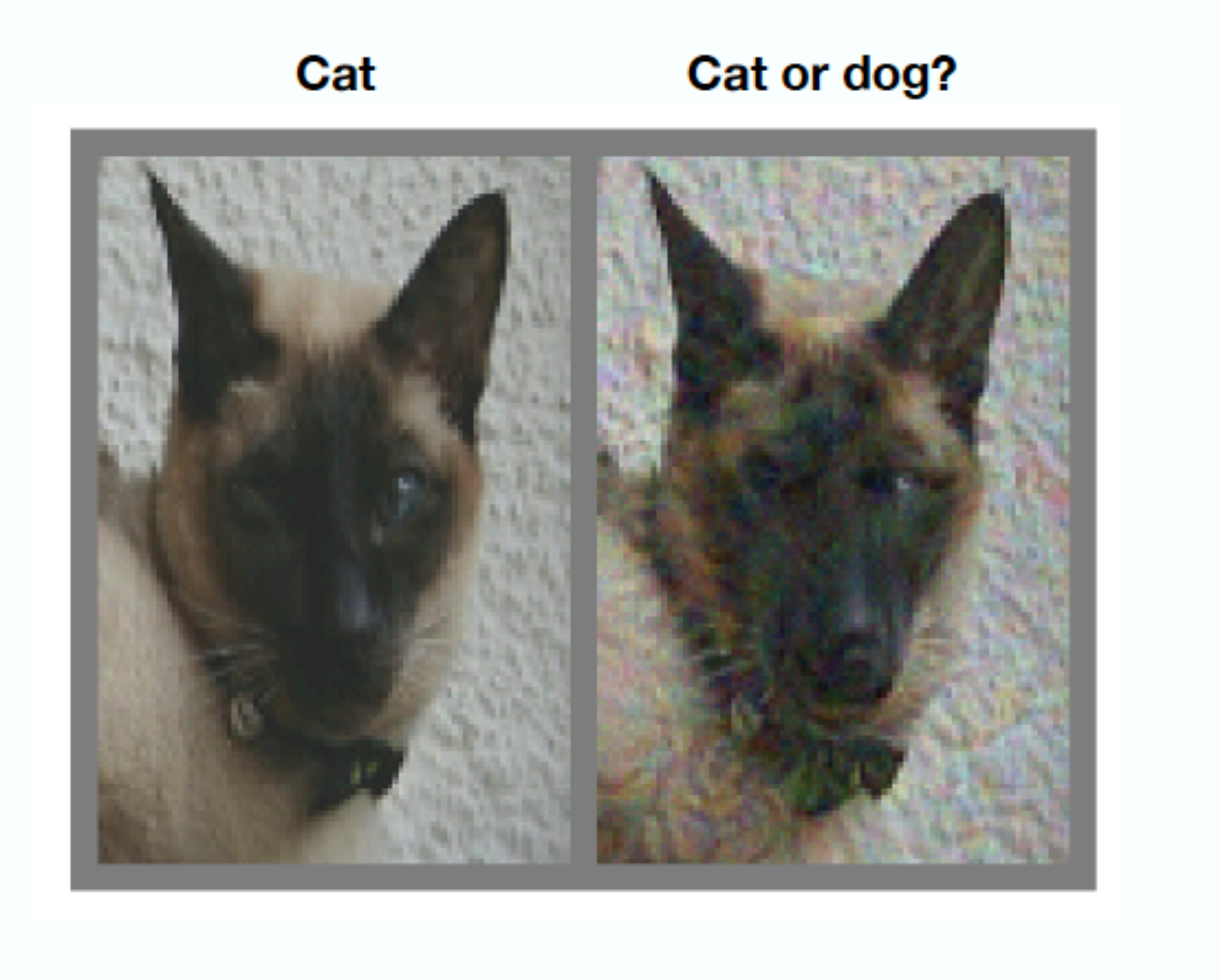
Adversarial examples: How bad is it?

Not that bad on MNIST

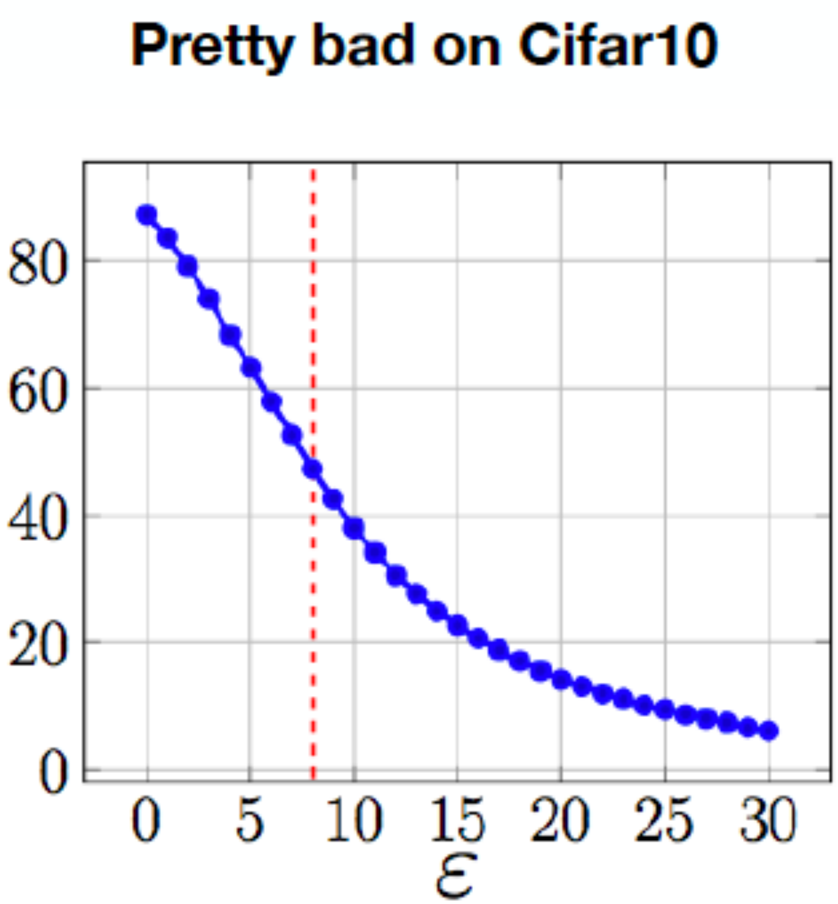


Epsilon = 0.3 out of 1

Some of these adversarial examples can even fool humans:

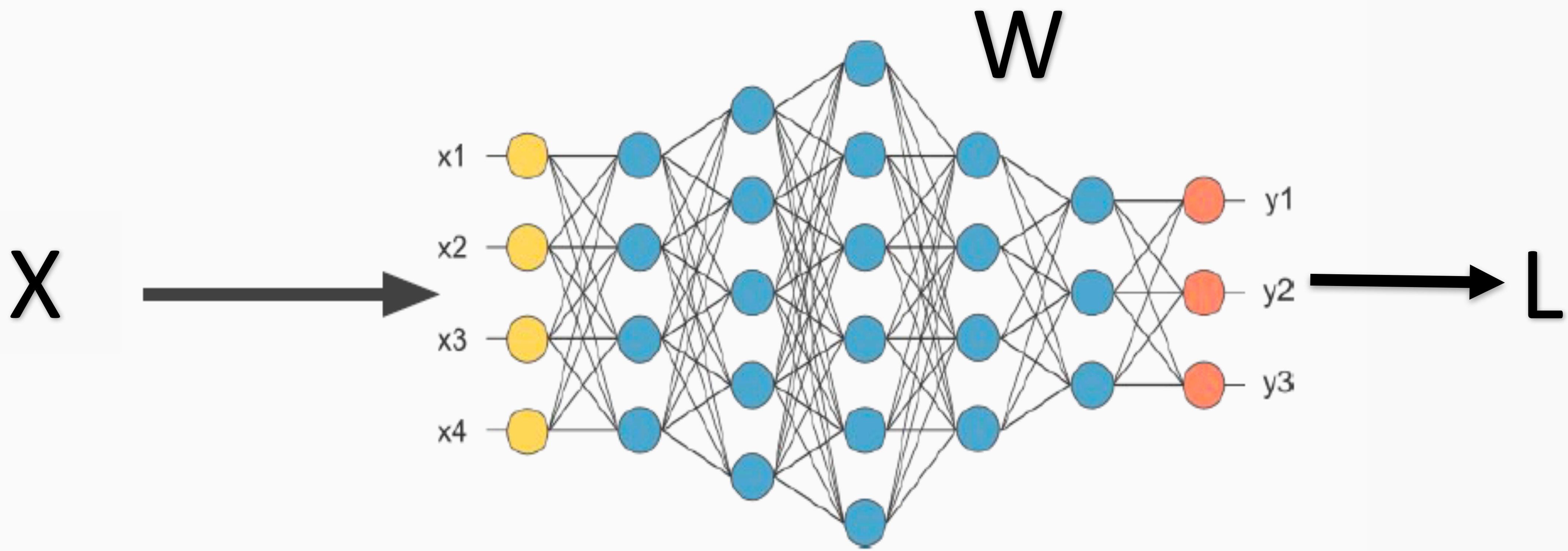


CIFAR10



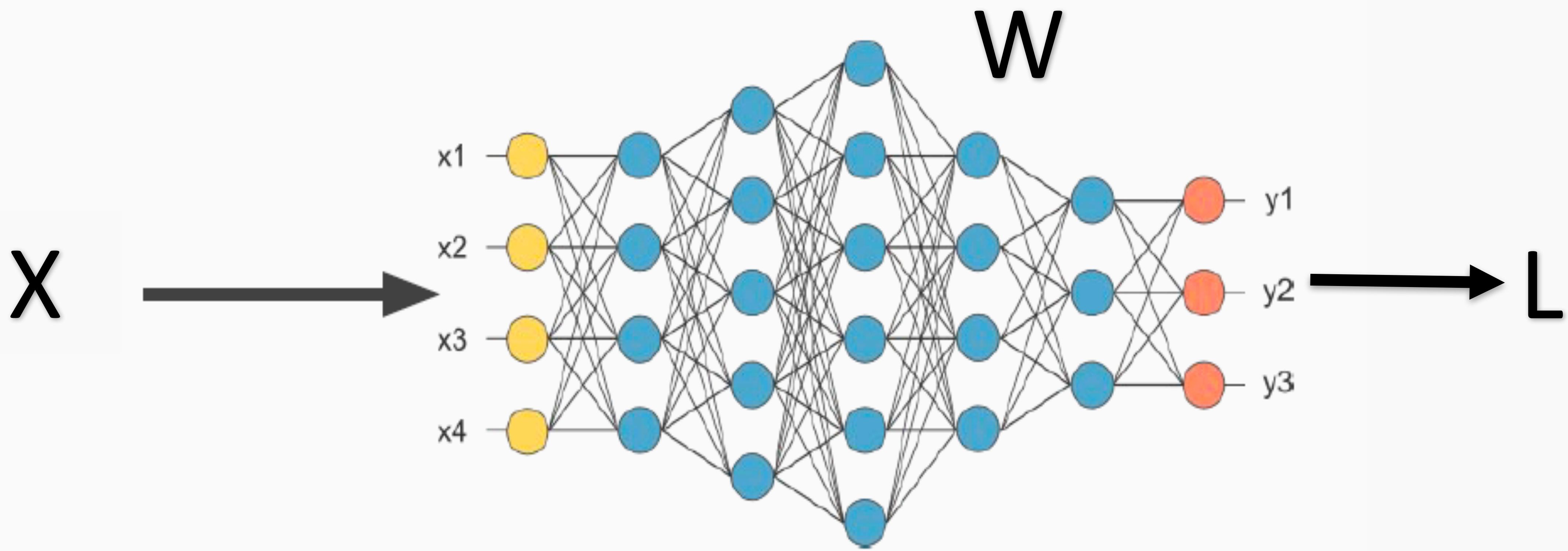
Epsilon = 8 out of 255

Attacking with **F**ast **G**radient **S**ign **M**ethod (FGSM)



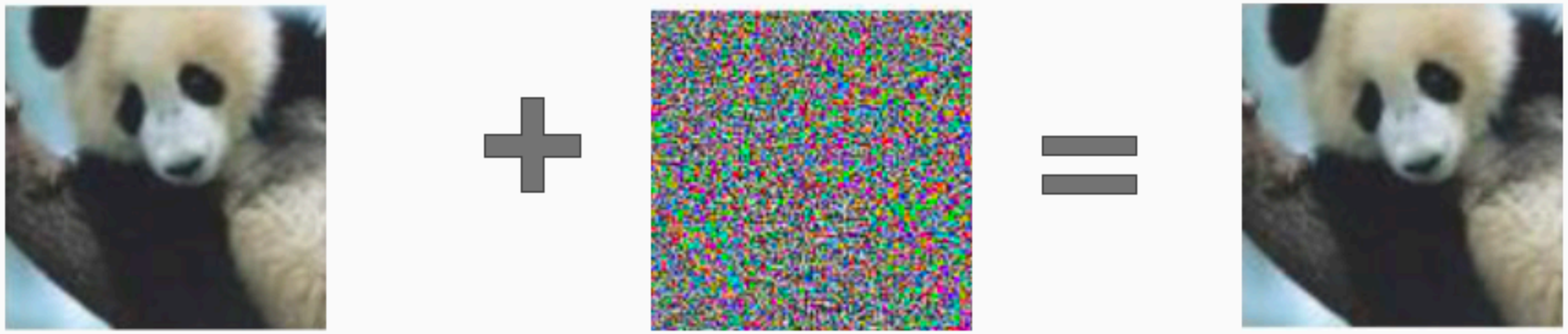
$$x + \lambda \cdot \mathrm{sign}(\nabla_{\mathrm{x}} \mathrm{L}) \Rightarrow \mathrm{x}^*$$

Attacking with **F**ast **G**radient **S**ign **M**ethod (FGSM)

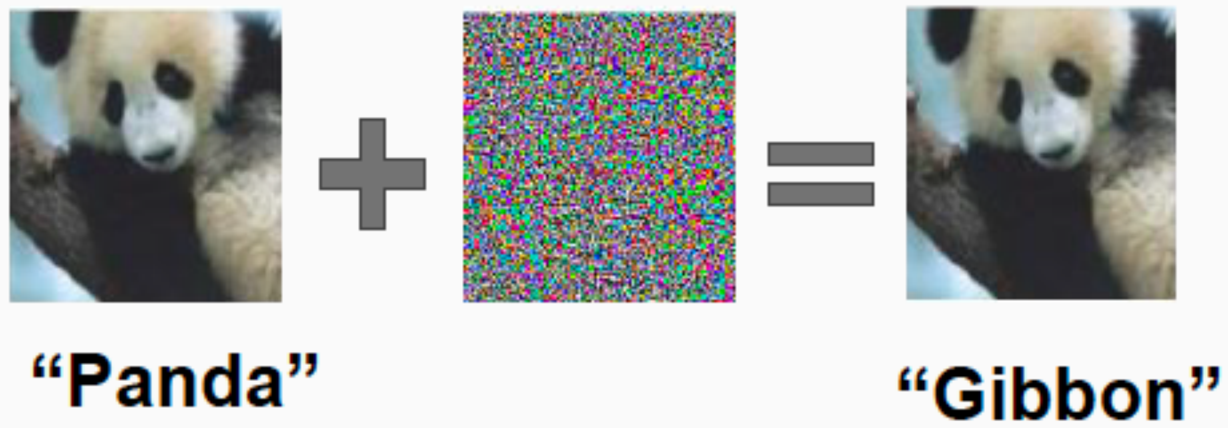


$$x + \lambda \cdot \mathrm{sign}(\nabla_{\mathrm{x}} \mathrm{L}) \Rightarrow \mathrm{x}^*$$

$$x + \lambda \cdot \mathrm{sign}(\nabla_{\mathrm{x}} \mathrm{L}) \Rightarrow \mathrm{x}^*$$



Defending with Adversarial Training



1. Generate adversarial examples
2. Adjust labels

Defending with Adversarial Training

1. Generate adversarial examples
2. Adjust labels

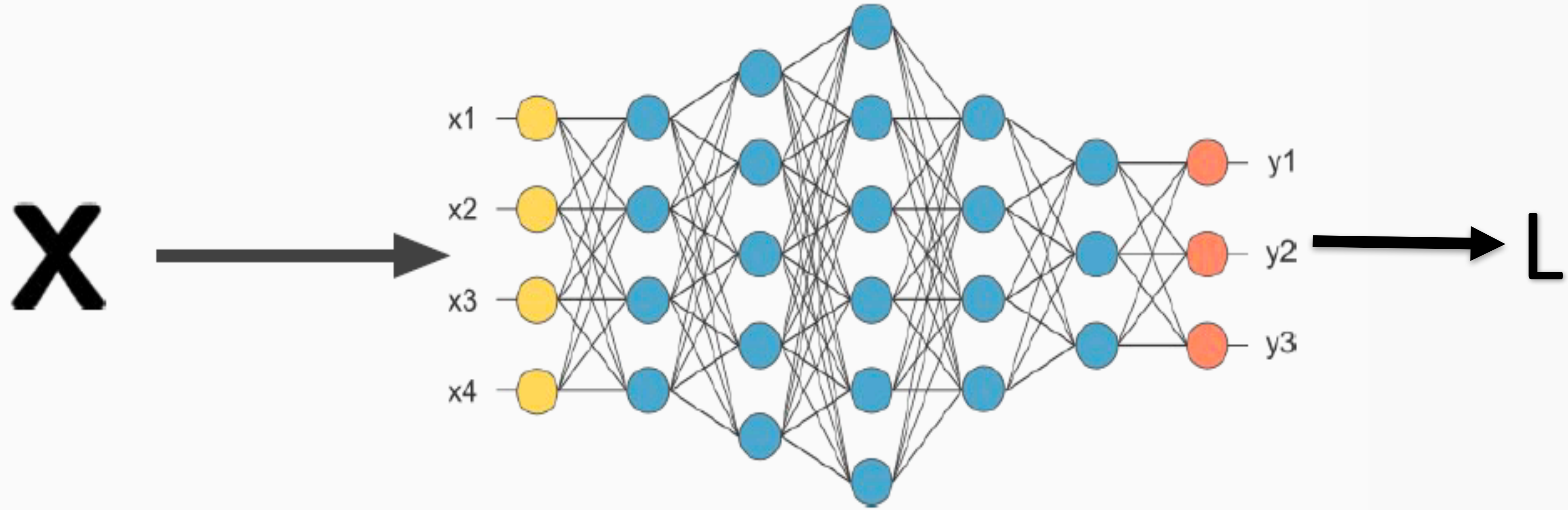
Defending with Adversarial Training

1. Generate adversarial examples
2. Adjust labels
3. Add them to the training set
4. Train new network

Attack methods post GoodFellow 2015

- FGSM [Goodfellow et. al ‘15]
- JSMA [Papernot et. al ‘16]
- C&W [Carlini + Wagner ‘16]
- Step-LL [Kurakin et. al ‘17]
- I-FGSM [Tramer et. al ‘18]

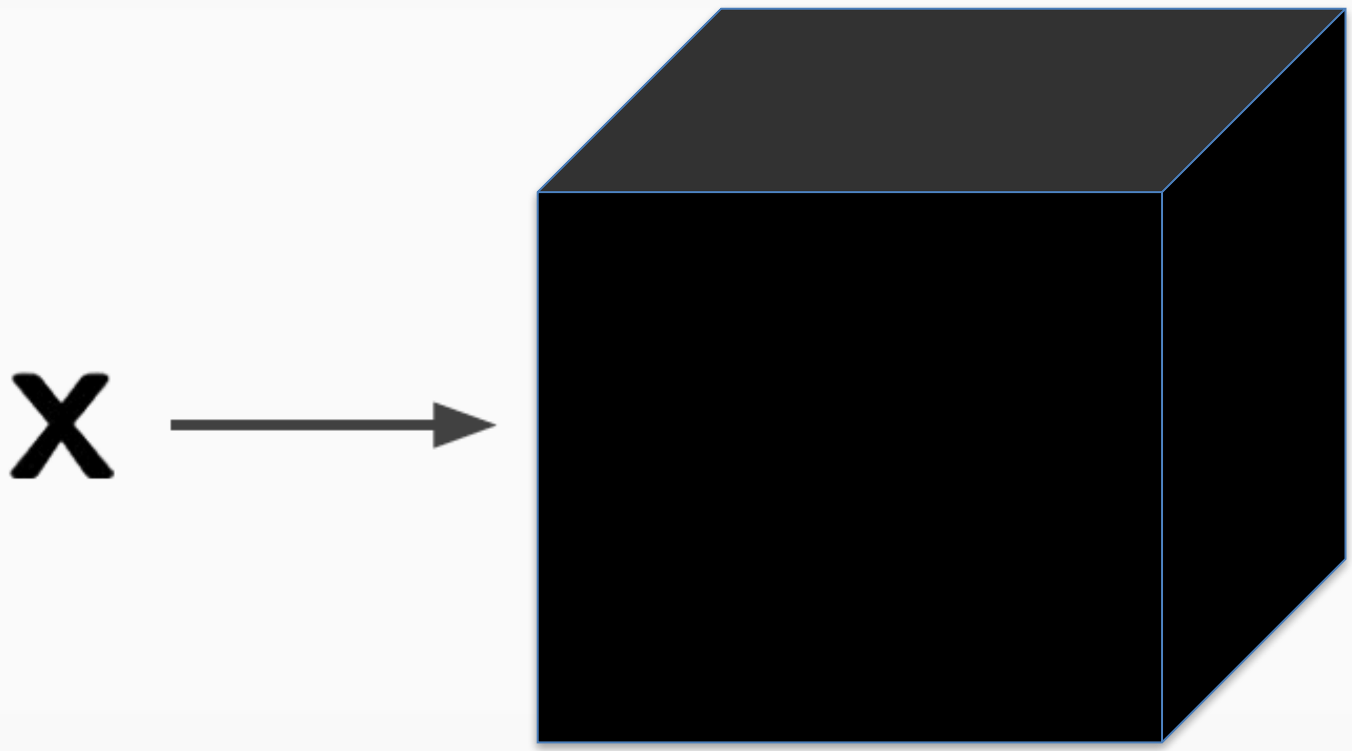
White box attacks



$$x + \lambda \cdot \mathrm{sign}(\nabla_{\mathrm{x}} \mathrm{L}) \Rightarrow \mathrm{x}^*$$

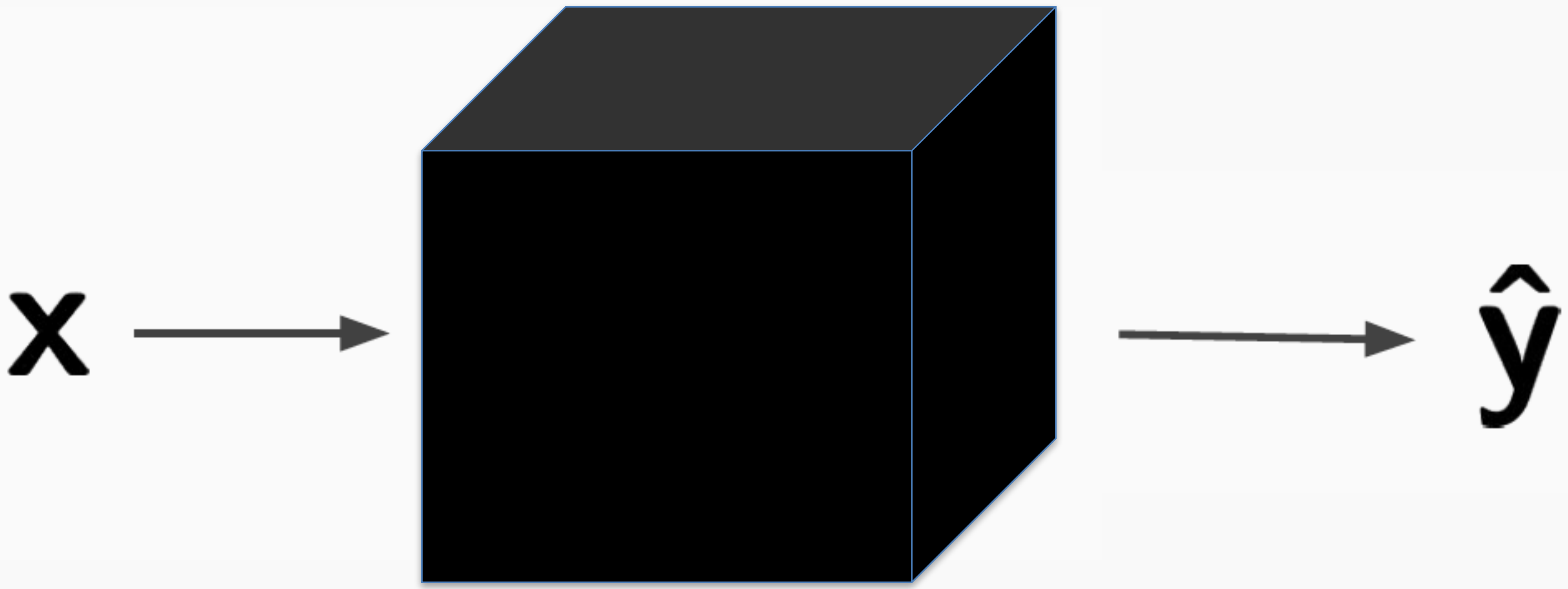
“Black Box” Attacks

“Black Box” Attacks [Papernot et. al ‘17]

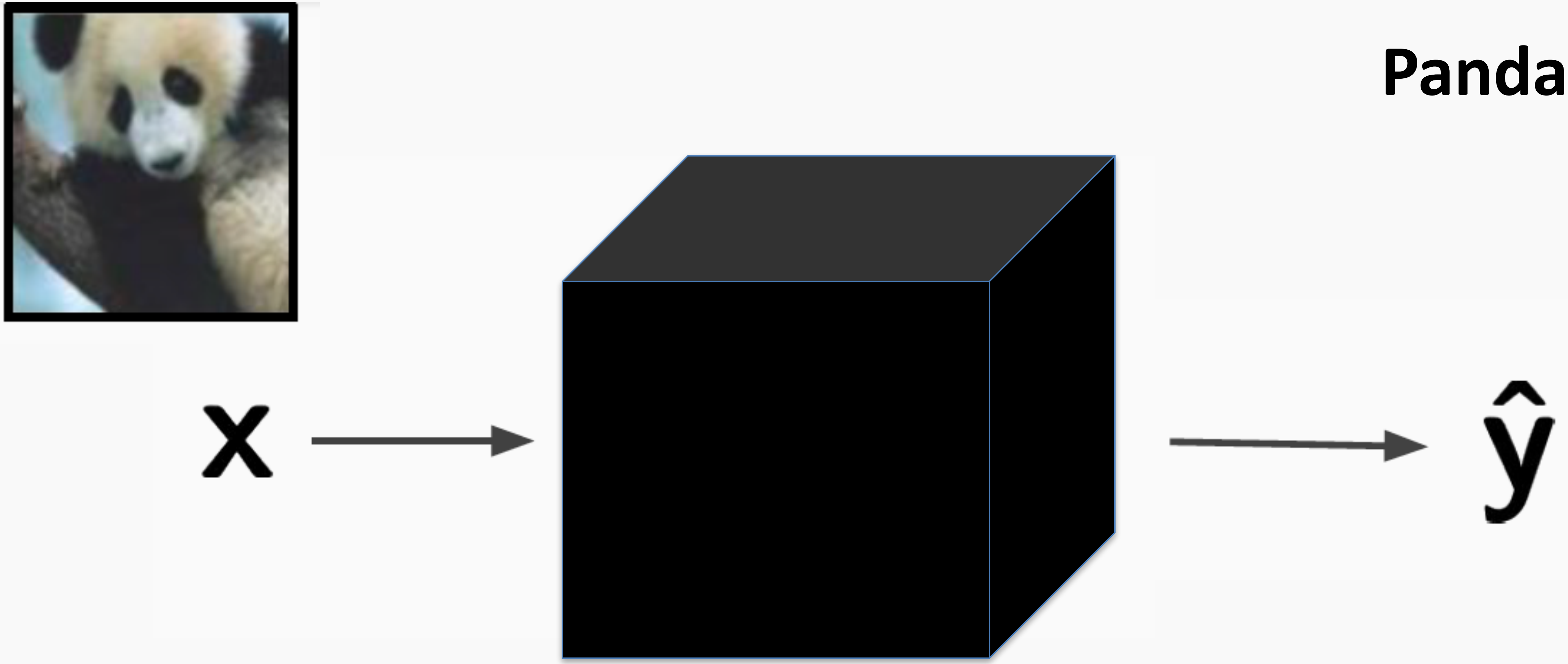


"Black Box" Attacks

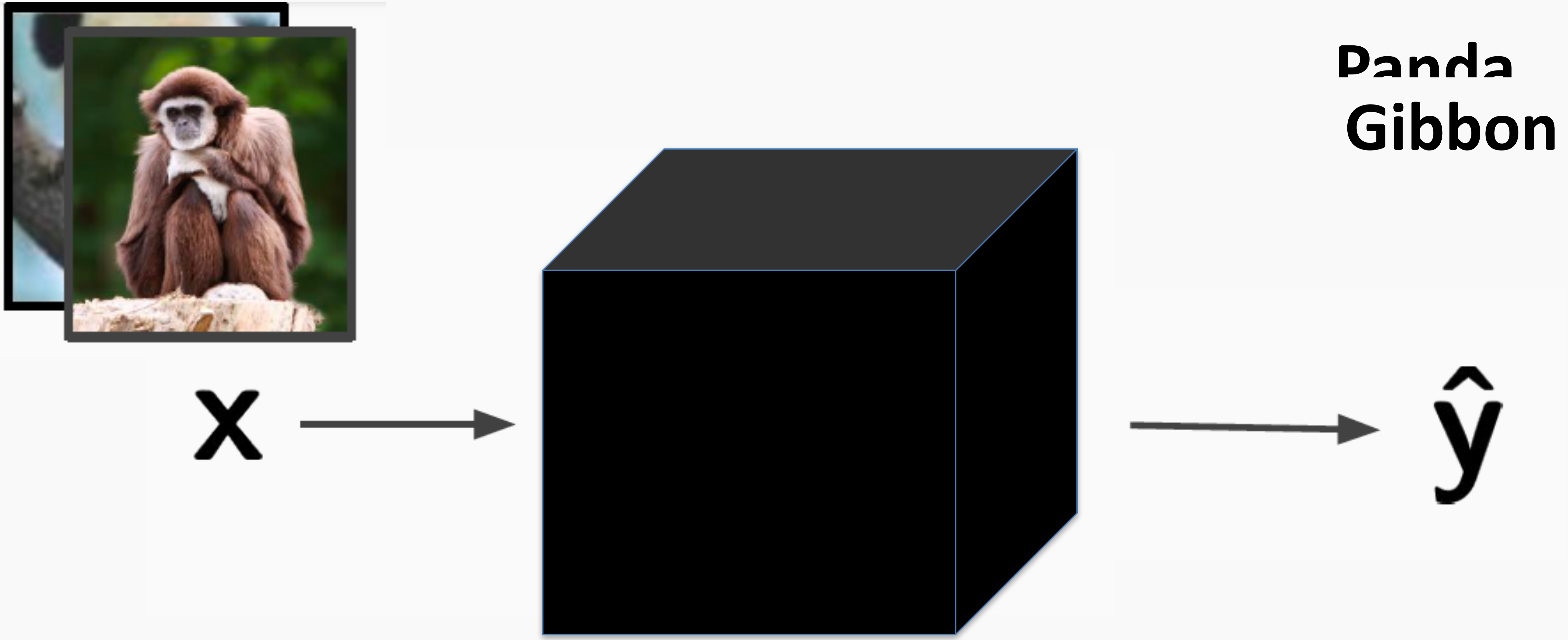
Examine inputs and outputs of the model



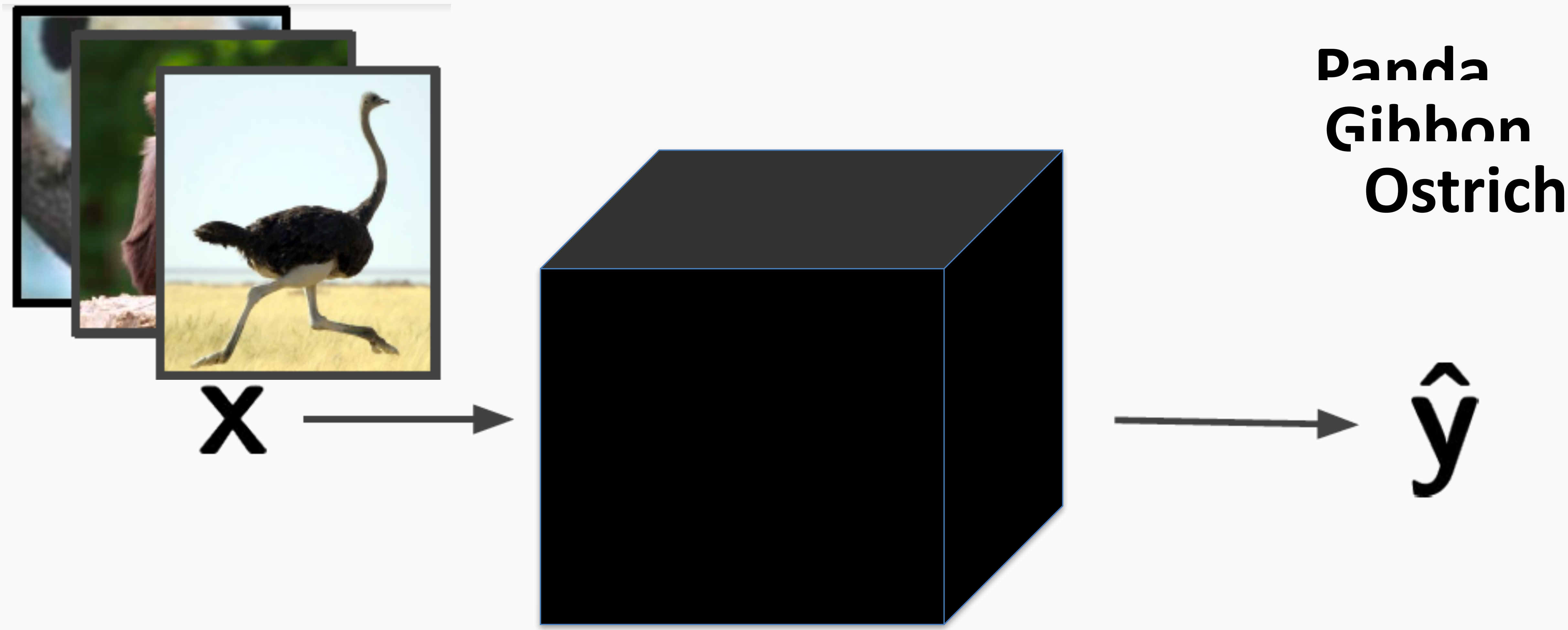
“Black Box” Attacks



“Black Box” Attacks



“Black Box” Attacks

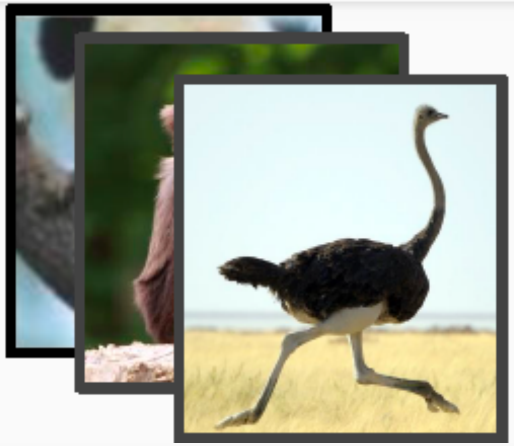


“Black Box” Attacks

Train a model that performs the same as the black box

“Black Box” Attacks

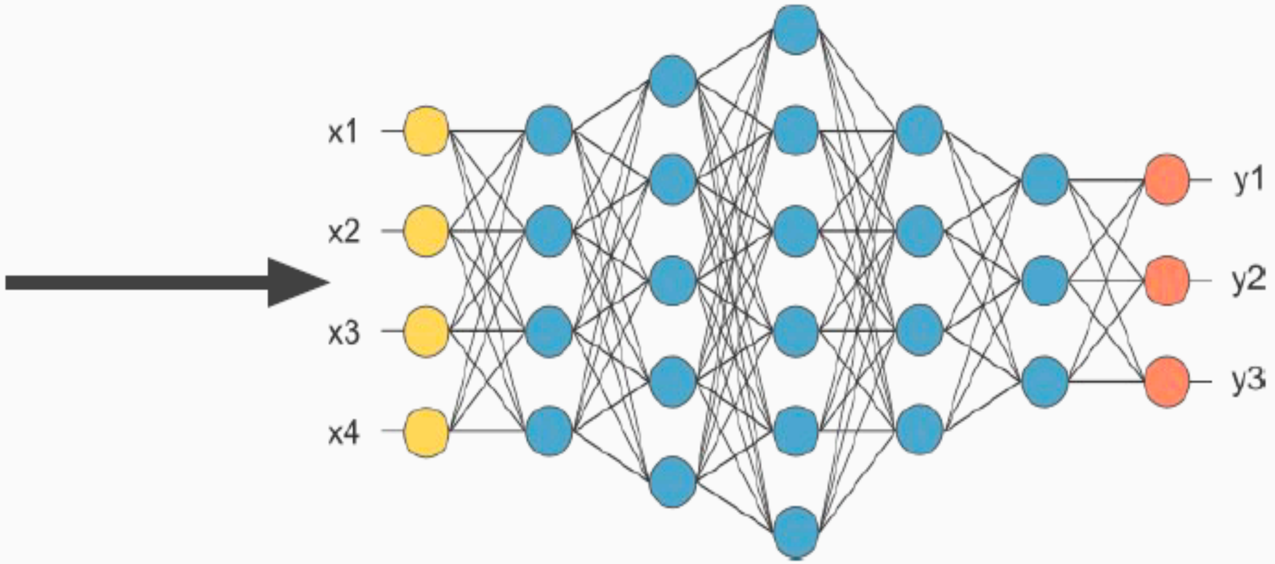
Train a model that performs the same as the black box



Panda

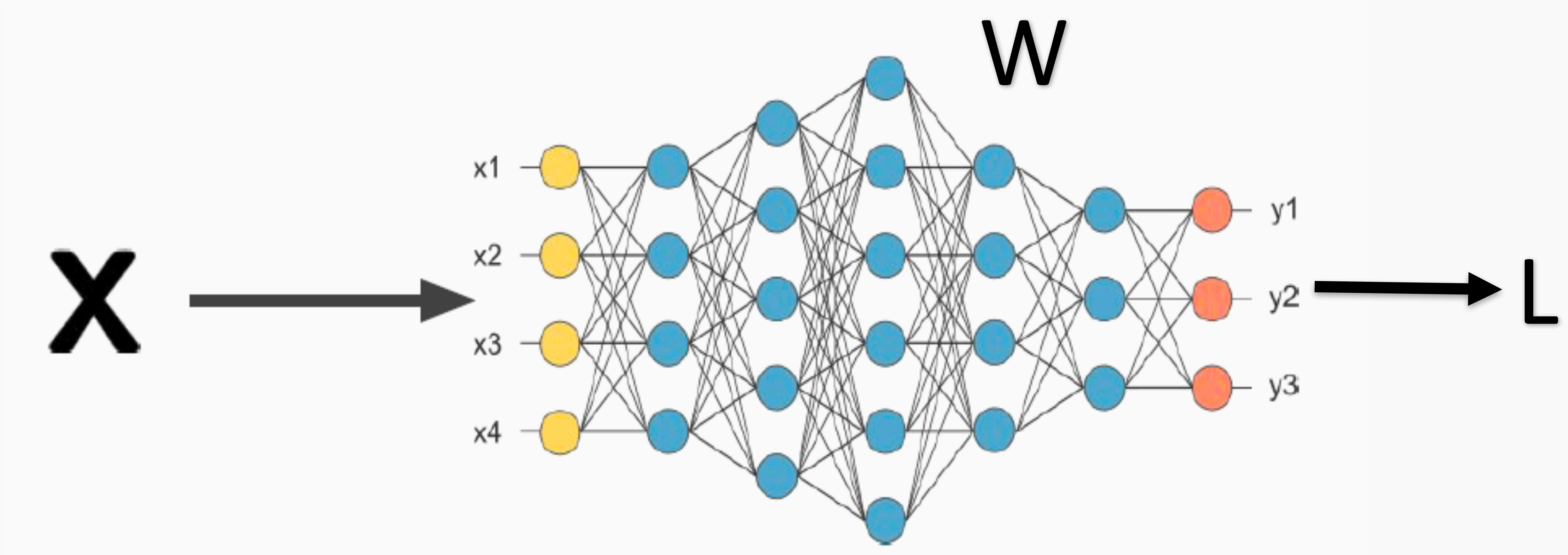
Gibbon

Ostrich



“Black Box” Attacks

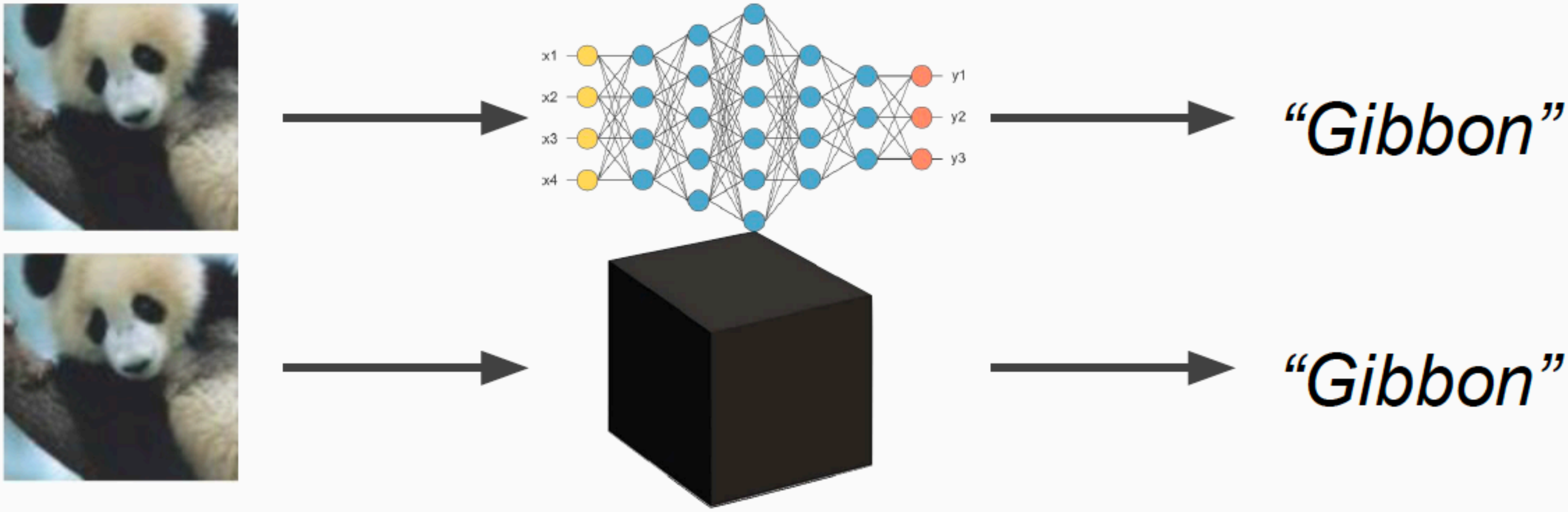
Now attack the model you just trained with “white” box attack



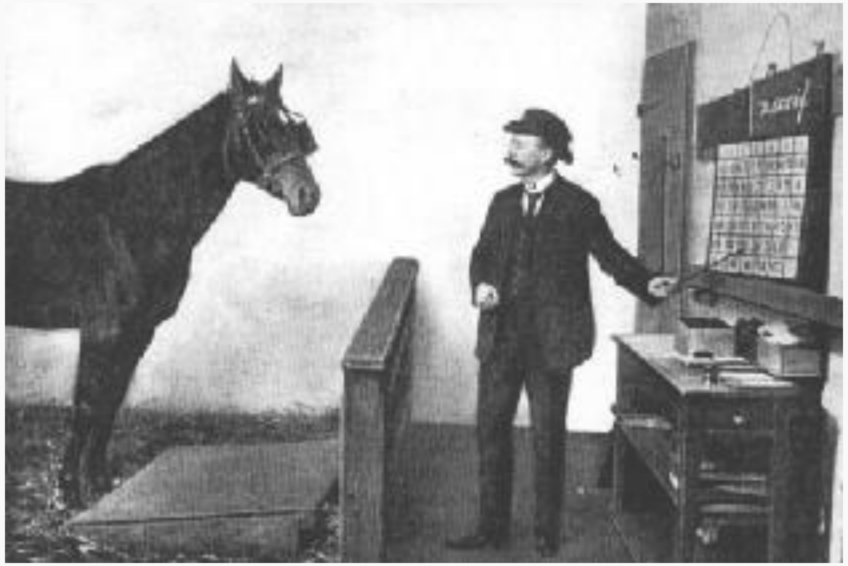
$$x + \lambda \cdot \mathrm{sign}(\nabla_{\mathrm{x}} \mathrm{L}) \Rightarrow \mathrm{x}^*$$

“Black Box” Attacks

Use those adversarial examples to the “black” box



CleverHans



A Python library to benchmark machine learning systems' vulnerability to adversarial examples.

https://github.com/tensorflow/cleverhans
http://www.cleverhans.io/

More Defenses

Mixup:

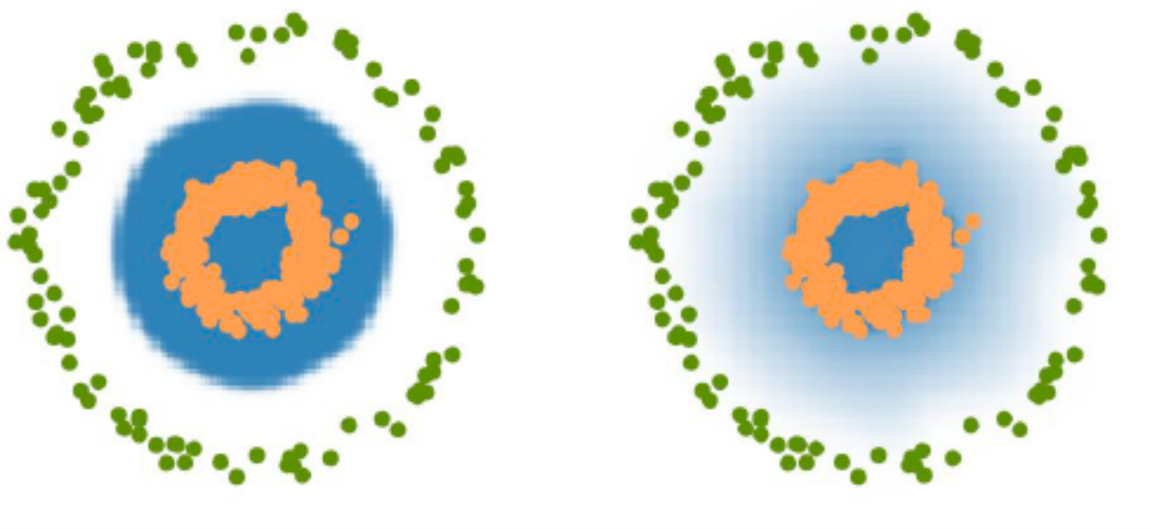
- Mix two training examples
- Augment training set

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j$$

Smooth decision boundaries:

- Regularize the derivatives wrt to x



Physical attacks

- Object Detection
- Adversarial Stickers