Pattern Recognition and Applications Lab

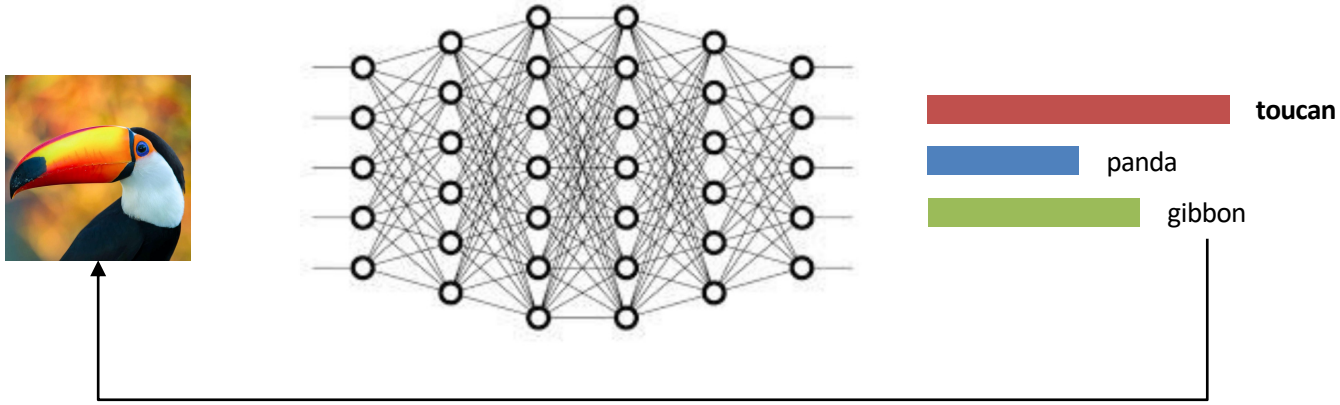
Università di Cagliari

Machine Learning Security Lab

Hands-on session on Evasion Attacks

Luca Demetrio
luca.demetrio93@unica.it
@zangobot

Recap on Adversarial Examples

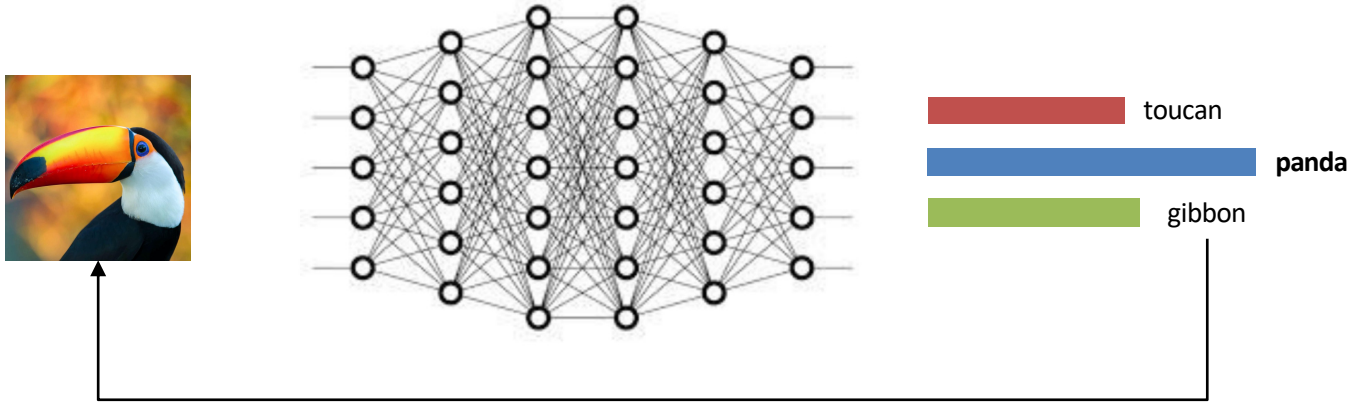


Gradient as a guide

Use gradient to compute perturbation towards the desired goal

[Biggio et al., Evasion Attacks Against Machine Learning at Test Time, ECML 2013]

Recap on Adversarial Examples



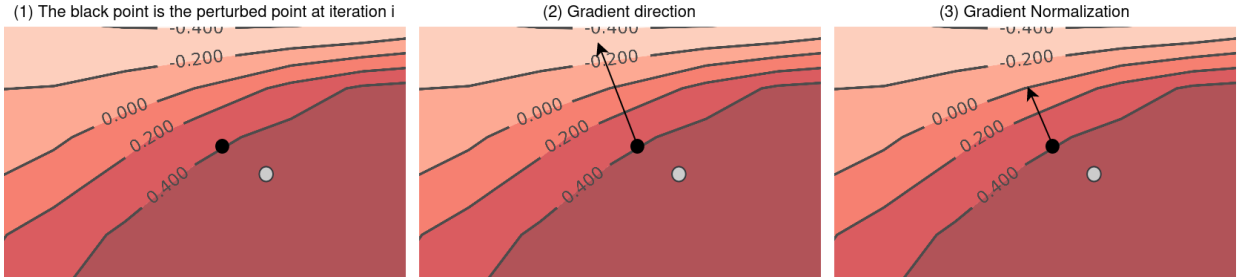
Gradient as a guide

Use gradient to compute perturbation towards the desired goal

[Biggio et al., Evasion Attacks Against Machine Learning at Test Time, ECML 2013]

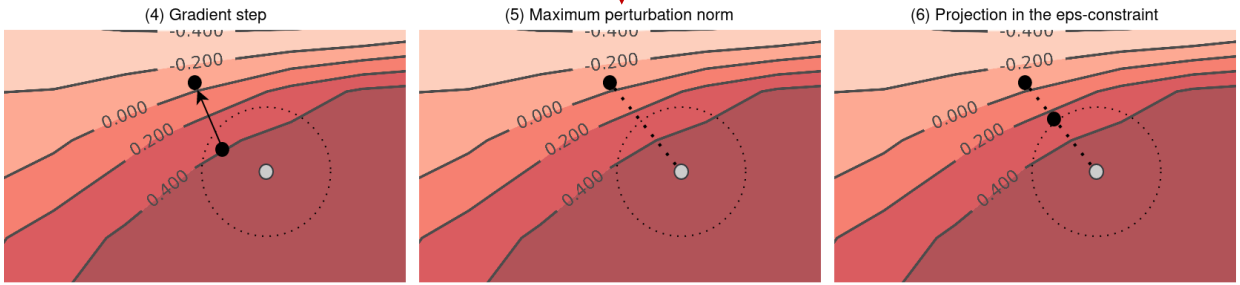
Step-by-step algorithm

```
for _ in range(self.steps):
    _, gradients = self.value_and_grad(loss_fn, x)
    gradients = self.normalize(gradients, x=x, bounds=model.bounds)
    x = x + gradient_step_sign * stepsize * gradients
    x = self.project(x, x0, epsilon)
    x = ep.clip(x, *model.bounds)
```



Step-by-step algorithm

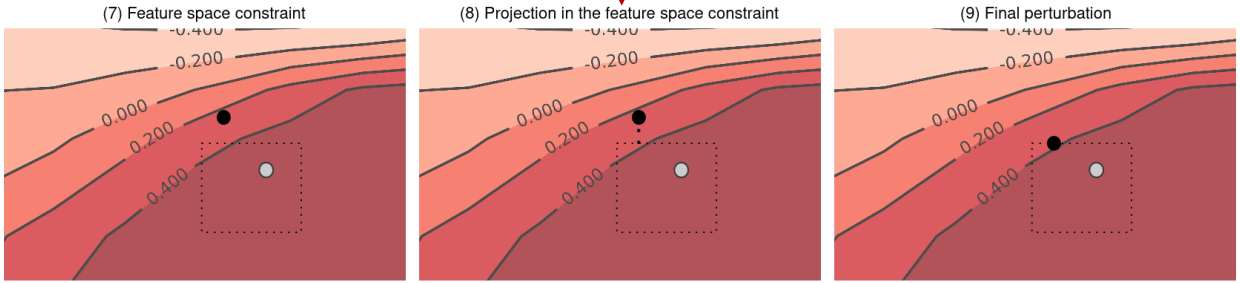
```
for _ in range(self.steps):
    _, gradients = self.value_and_grad(loss_fn, x)
    gradients = self.normalize(gradients, x=x, bounds=model.bounds)
    x = x + gradient_step_sign * stepsize * gradients
    x = self.project(x, x0, epsilon)
    x = ep.clip(x, *model.bounds)
```



https://github.com/unica-mlsec/mlsec/tree/main/code

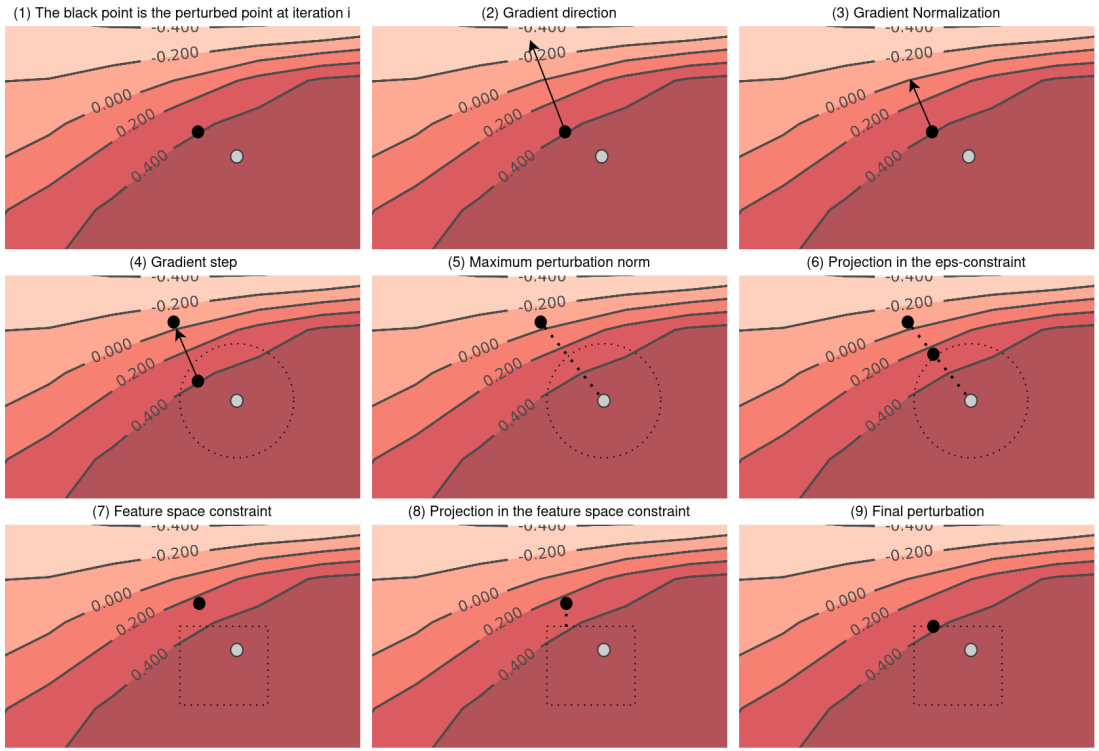
Step-by-step algorithm

```
for _ in range(self.steps):
    _, gradients = self.value_and_grad(loss_fn, x)
    gradients = self.normalize(gradients, x=x, bounds=model.bounds)
    x = x + gradient_step_sign * stepsize * gradients
    x = self.project(x, x0, epsilon)
    x = ep.clip(x, *model.bounds)
```



Final recap

```python
for _ in range(self.steps):
    _, gradients = self.value_and_grad(loss_fn, x)
    gradients = self.normalize(gradients, x=x, bounds=model.bounds)
    x = x + gradient_step_sign * stepsize * gradients
    x = self.project(x, x0, epsilon)
    x = ep.clip(x, *model.bounds)
```



https://github.com/unica-mlsec/mlsec/tree/main/code

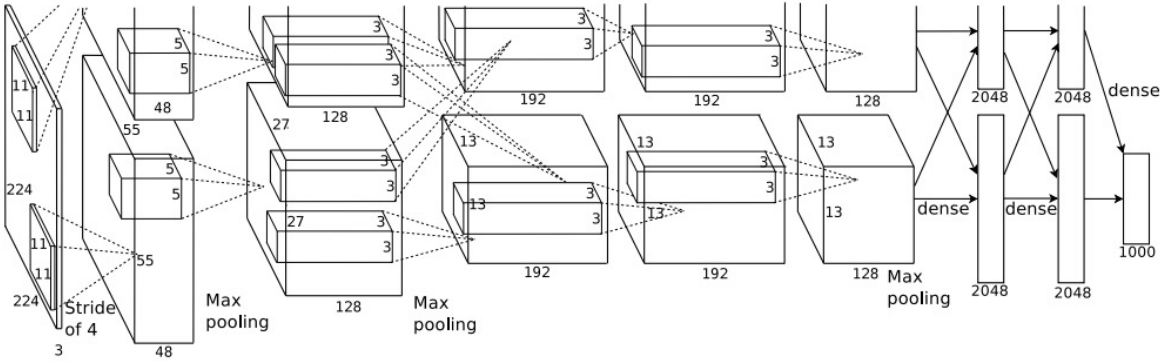
Exercise 1: Flawed implementation

- Open the challenge in Google Colab
  https://colab.research.google.com/github/zangobot/adversarial_challenge/blob/main/chall1.ipynb
- Execute the notebook (run all the cells) and collect the result
- **Evasion is achieved, but the code has a bug! Can you spot it?**

Exercise 2: Fool multiple models at once

- Patch the bug found during the previous exercise
- Instantiate a new network:
  ```
  net = SimpleNet().load_pretrained_mnist('mnist_net2.pth')
  ```
- Use the new network to classify the input image
- **Compute a perturbation that fool both networks (one with target 2 and the other with target 9)**

EXTRA Example: Model trained on ImageNet



- 1M Images, 1000 classes
- Pretrained models available from torchvision

http://www.image-net.org/challenges/LSVRC/ (2012 edition)
https://en.wikipedia.org/wiki/ImageNet (historical remarks)
https://arxiv.org/pdf/1409.0575.pdf (ImageNet paper)
https://en.wikipedia.org/wiki/AlexNet (ILSVRC 2012 challenge winners with AlexNet)

EXTRA Example: Model trained on ImageNet

- Open the example notebook in Google Colab
  https://colab.research.google.com/github/unica-ml/ml/blob/master/notebooks/lab06.ipynb
- Execute the notebook

Thanks!

Luca Demetrio
luca.demetrio93@unica.it
@zangobot

If you know the enemy and know yourself, you need not fear the result of a hundred battles
Sun Tzu, The art of war, 500 BC