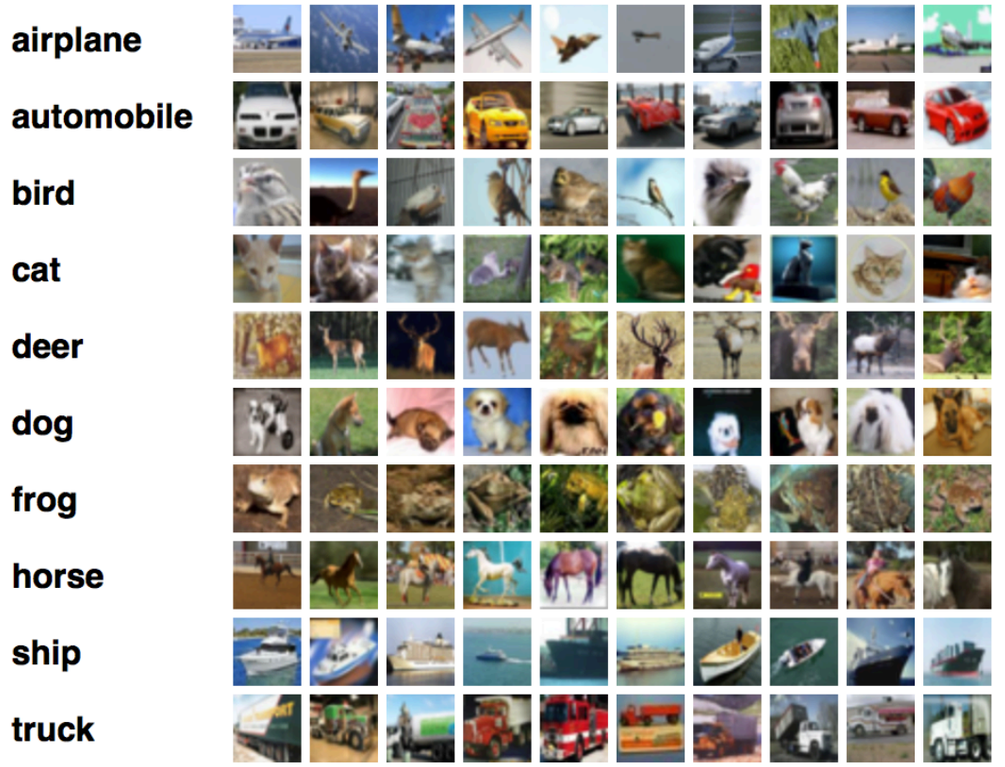
PRACTICAL MACHINE LEARNING: ROBUSTNESS & ADVERSARIAL EXAMPLES IN IMAGE CLASSIFICATION

Dr. Alexander Booth (he / him)
Universidad Católica del Norte, Chile
October 11th, 2024

Queen Mary University of London

Recap

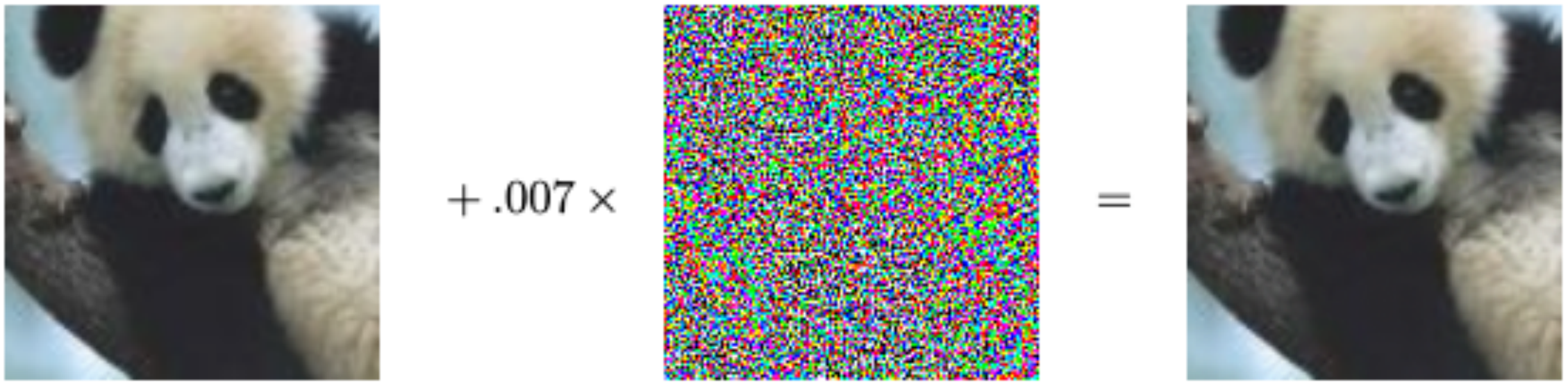
- Learned about all of the basic building blocks of a CNN.
- Discussed the process of convolution of images and important concepts such as zero-padding and pooling.
- Put all of this together to build a CNN to classify images in two datasets.
  - **MNIST**: library of 70000 labelled 28 x 28 pixel **greyscale** handwritten numbers [1].
  - **CFAR10**: library of 60000 labelled 32 x 32 pixel **colour** images in 10 classes [2].



Adversarial Examples

- Altering dataset instances (e.g images) to neural networks so they get misclassified -> “tricking” the network.
- Trying to do this in such a way that **humans** cannot tell the difference.

[3]



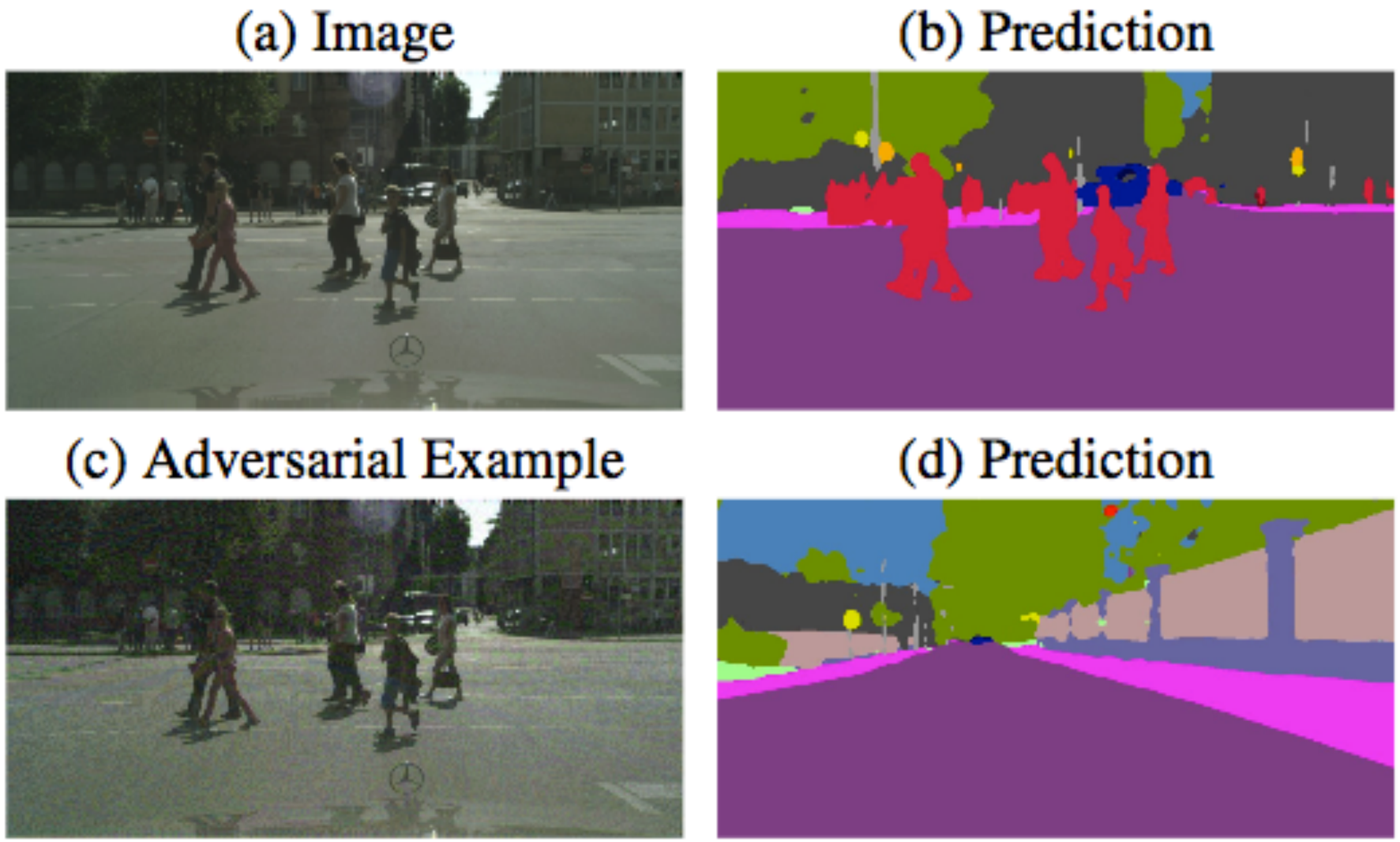
Adversarial Turtle

- 3D printed turtle that gets classified as a rifle from 8 out of 10 angles.
- Unperturbed model classifies correctly as a turtle almost 100% of the time from whatever angle.

[4]

Pedestrian Removal

- Add noise to remove specific target class in segmentation.



[5]

Attacks in the Physical Domain

- Physical changes made to objects (deliberate or accidental).

"Poster" attack (**100% misclassification**):

"Sticker" attack (**85% misclassification**):

[6]

How Are Non-physical Attacks Done?

- Assume that we have a fixed model (model weights already determined, $\tilde{W}$).

- In adversarial attacks the loss with respect to the input data is manipulated by **tweaking the input data**.
  - **Untargeted**: aiming to tweak the input data so that the **loss increases**.
  - **Targeted:** aiming to tweak the input data so that the **loss with respect to a chosen class decreases**.

[6]

How Are Non-physical Attacks Done?

- Assume that we have a fixed model (model weights already determined, $\tilde{W}$).

- In adversarial attacks the loss with respect to the input data is manipulated by **tweaking the input data**.
  - **Untargeted**: aiming to tweak the input data so that the **loss increases**.
  - **Targeted:** aiming to tweak the input data so that the **loss with respect to a chosen class decreases**.

$$X_{i+1} = X_i + \Delta X_i : \Delta L < 0 \text{ (targeted)}, \Delta L > 0 \text{ (untargeted)}$$

[6]

How Are Non-physical Attacks Done?

- Assume that we have a fixed model (model weights already determined, $\tilde{W}$).

- In adversarial attacks the loss with respect to the input data is manipulated by **tweaking the input data**.
  - **Untargeted**: aiming to tweak the input data so that the **loss increases**.
  - **Targeted:** aiming to tweak the input data so that the **loss with respect to a chosen class decreases**.

$$X_{i+1} = X_i + \Delta X_i : \Delta L < 0 \text{ (targeted)}, \Delta L > 0 \text{ (untargeted)}$$

Controls size of perturbation

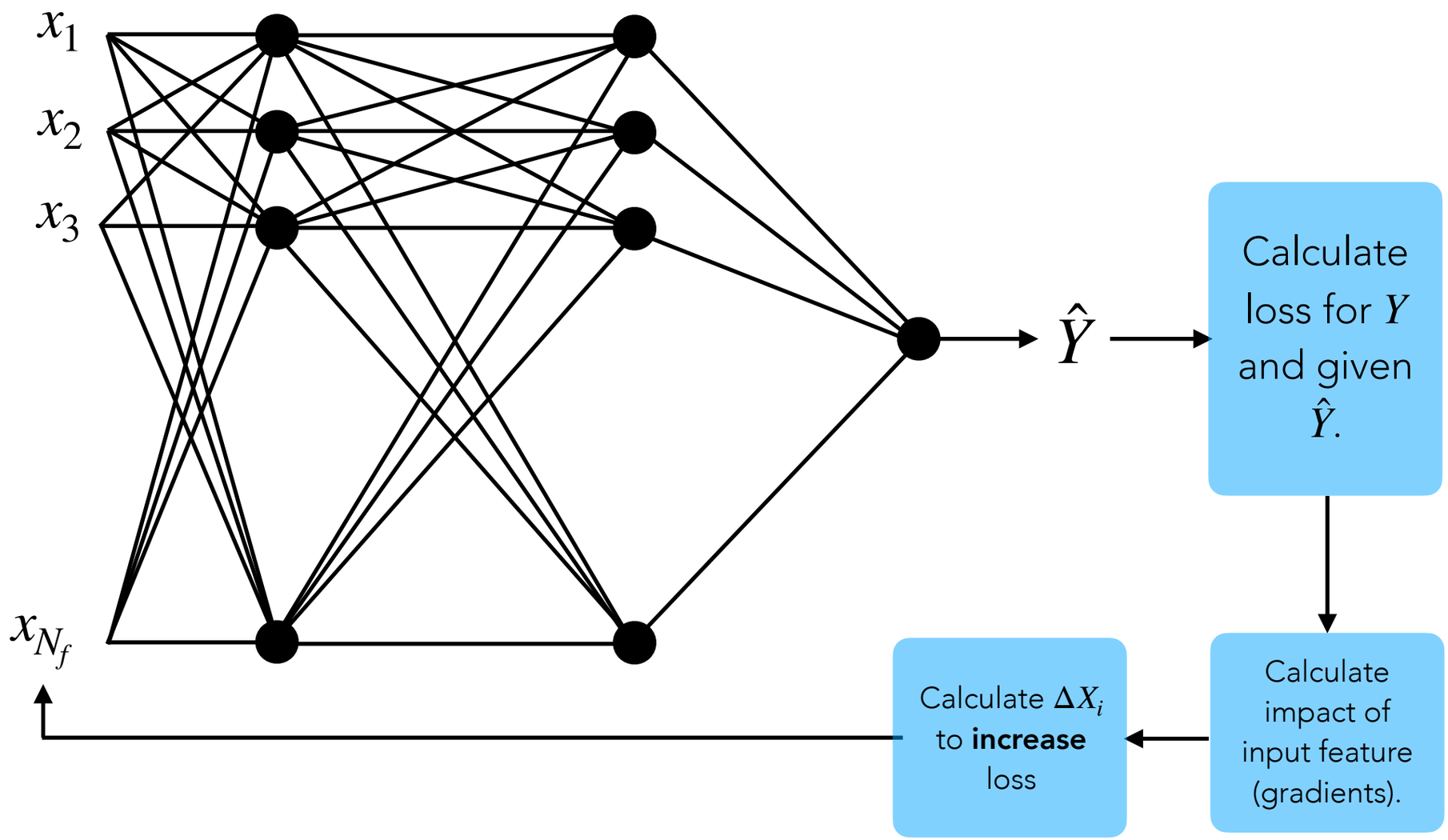
$$X_{i+1} = X_i + \eta\epsilon \; sign(\nabla_{X_i} L(\tilde{W}, X_i, Y_i))$$

{-1, 1} for {targeted, untargeted}

Direction of steepest ascent

Untargeted

$X = [x_1, \ldots, x_{N_f}]$ $Y = [y]$ Where y is the true label.

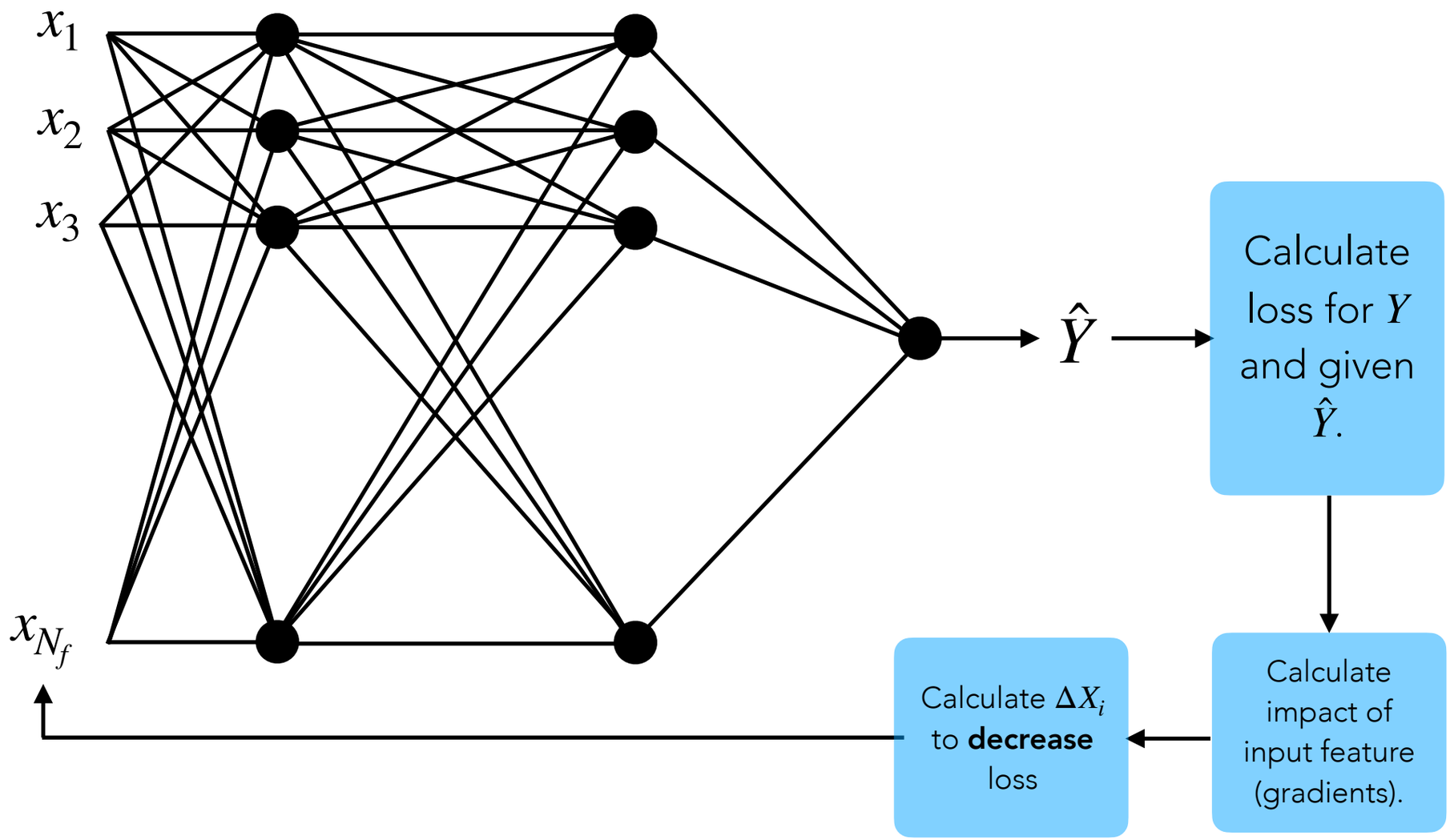


$$X_{i+1} = X_i + \epsilon \; sign(\nabla_{X_i} L(\tilde{W}, X_i, Y_i))$$

Targeted

$X = [x_1, \ldots, x_{N_f}]$ $Y = [y']$

Where y' is the **purposely incorrect** label you want to trick the network into thinking it is.



$$X_{i+1} = X_i - \epsilon \; sign(\nabla_{X_i} L(\tilde{W}, X_i, Y_i))$$

Fast Gradient Sign Method

- Known as the **fast gradient sign method** [3].
- Implementation via cleverhans [7].

[6]

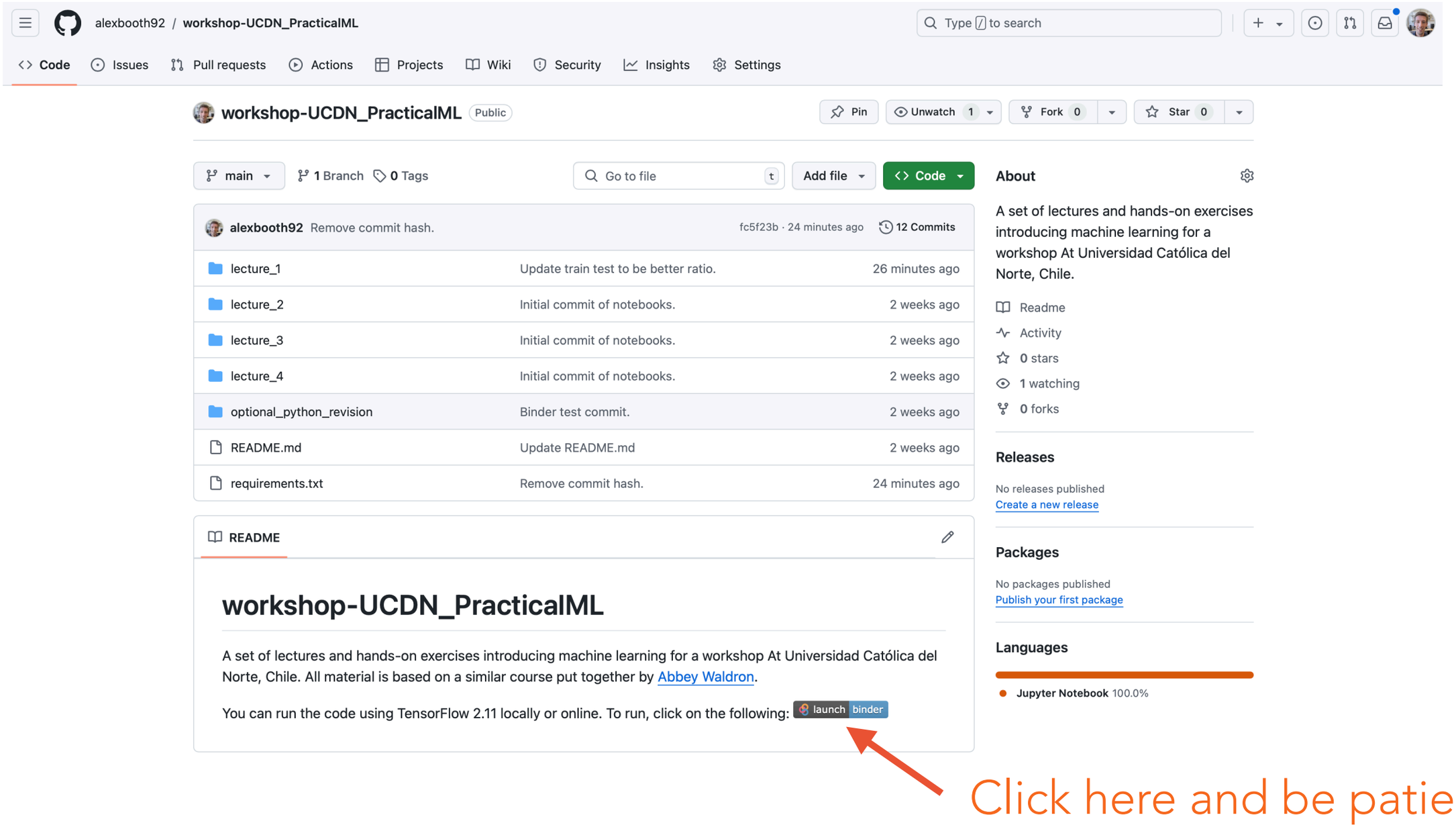
Today's Notebook

- Use a pretrained neural network as the target of our attack.
- Use fast gradient sign method to mount a targeted attack on a single image.
- Use the cleverhans implementation.

Accessing the Notebooks

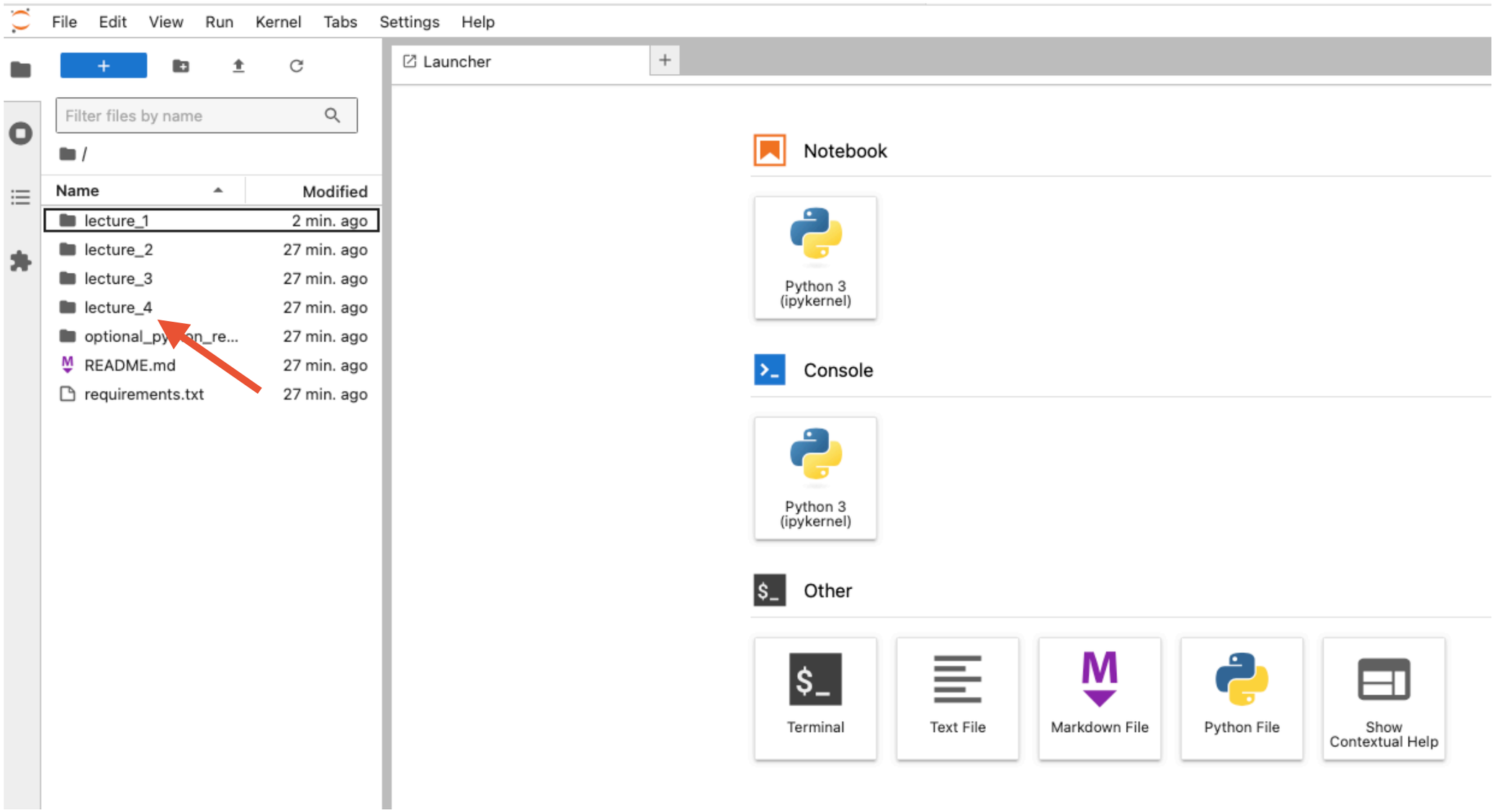
Follow this link:

https://github.com/alexbooth92/workshop-UCDN_PracticalML.git



Click here and be patient!

Accessing the Notebooks



Have Fun!

Queen Mary
University of London

References

[1] http://yann.lecun.com/exdb/mnist/
[2] https://www.cs.toronto.edu/~kriz/cifar.html
[3] Goodfellow et al. “Explaining and Harnessing Adversarial Examples”, CoRR 1412.6572 (2014)
[4] arXiv: 1707.07397
[5] arXiv: 1704.05712
[6] arXiv: 1707.08945
[7] https://github.com/cleverhans-lab/cleverhans/blob/master/cleverhans/jax/attacks/fast_gradient_method.py