PRACTICAL MACHINE LEARNING

ADVERSARIAL EXAMPLES

Queen Mary
University of London

THIS COURSE - 4 WORKSHOPS (LECTURE+EXERCISES)

- Lecture 1 (Monday) Introduction to machine learning with neural networks and linear regression
- Lecture 2 (Tuesday) Optimisation and non-linear regression with neural networks
- Lecture 3 (Wednesday) Classification and convolutional neural networks for image classification
- Lecture 4 (Thursday) Robustness and adversarial examples to image classification problems

WHO WE ARE

Dr Alex Booth [he/him]

Dr Linda Cremonesi [she/her]

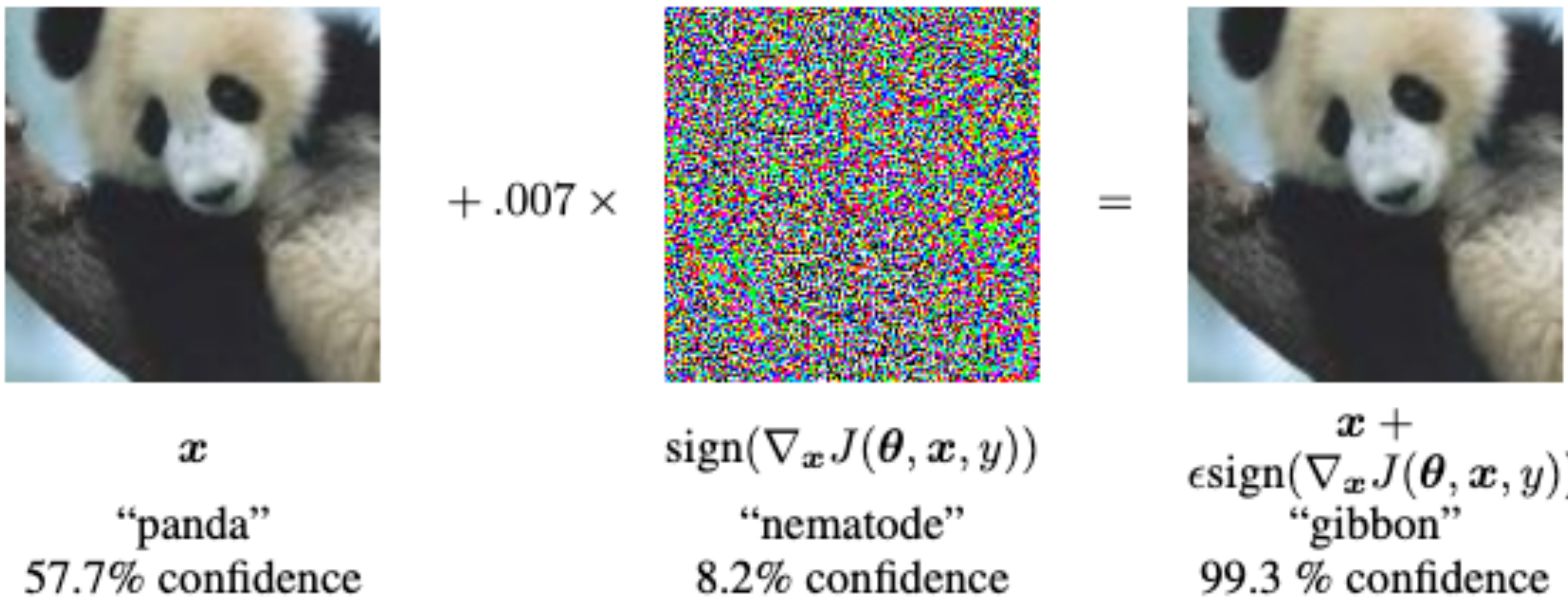
Dr Abbey Waldron [she/her]

ADVERSARIAL EXAMPLES

- Altering inputs (e.g. images) to neural networks so they get misclassified
- Try to do this in such a way as humans can't spot the difference

ADVERSARIAL EXAMPLES

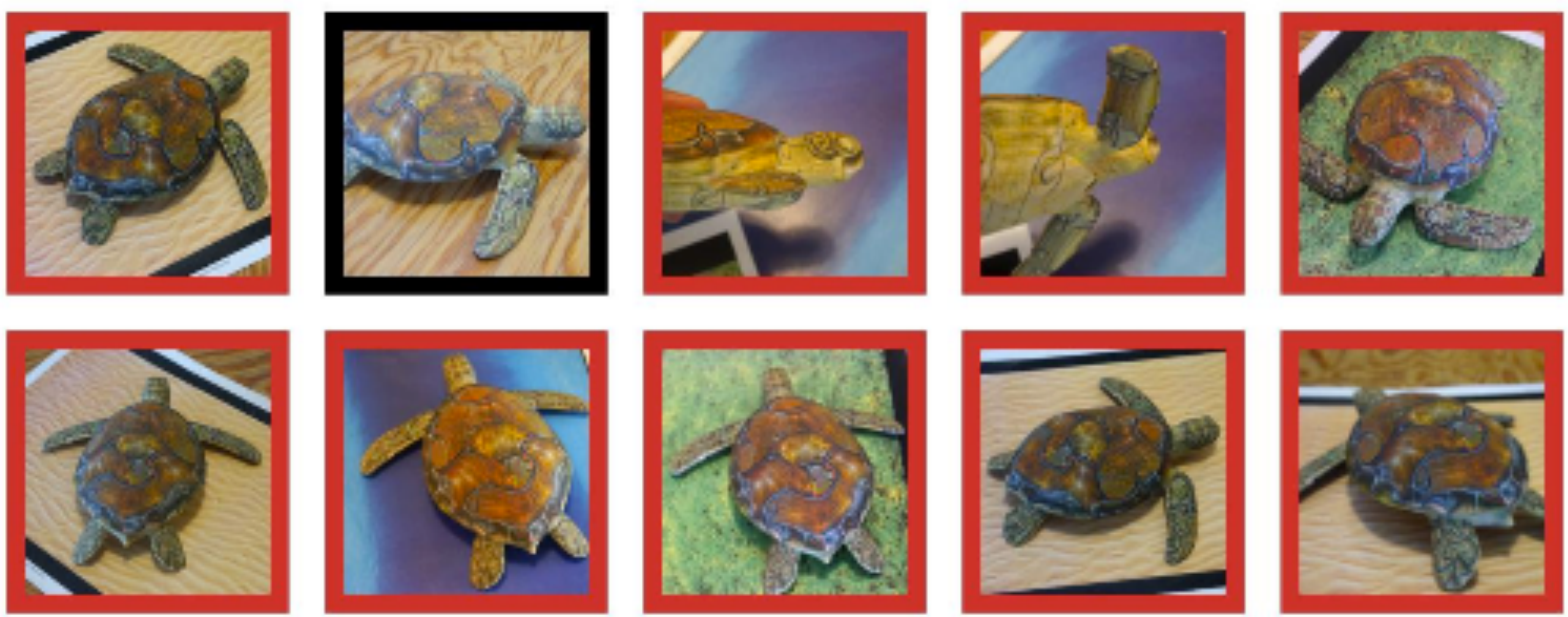
- Adding adversarial noise to images



Goodfellow et at, "Explaining and Harnessing Adversarial Examples", CoRR 1412.6572 (2014)

ADVERSARIAL TURTLE

- 3D printed turtle that gets classified as a rifle from all angles



classified as turtle   classified as rifle   classified as other

arXiv: 1707.07397

STOP SIGN ATTACKS

- Poster attack, 100% mis-classification

arXiv: 1707.08945

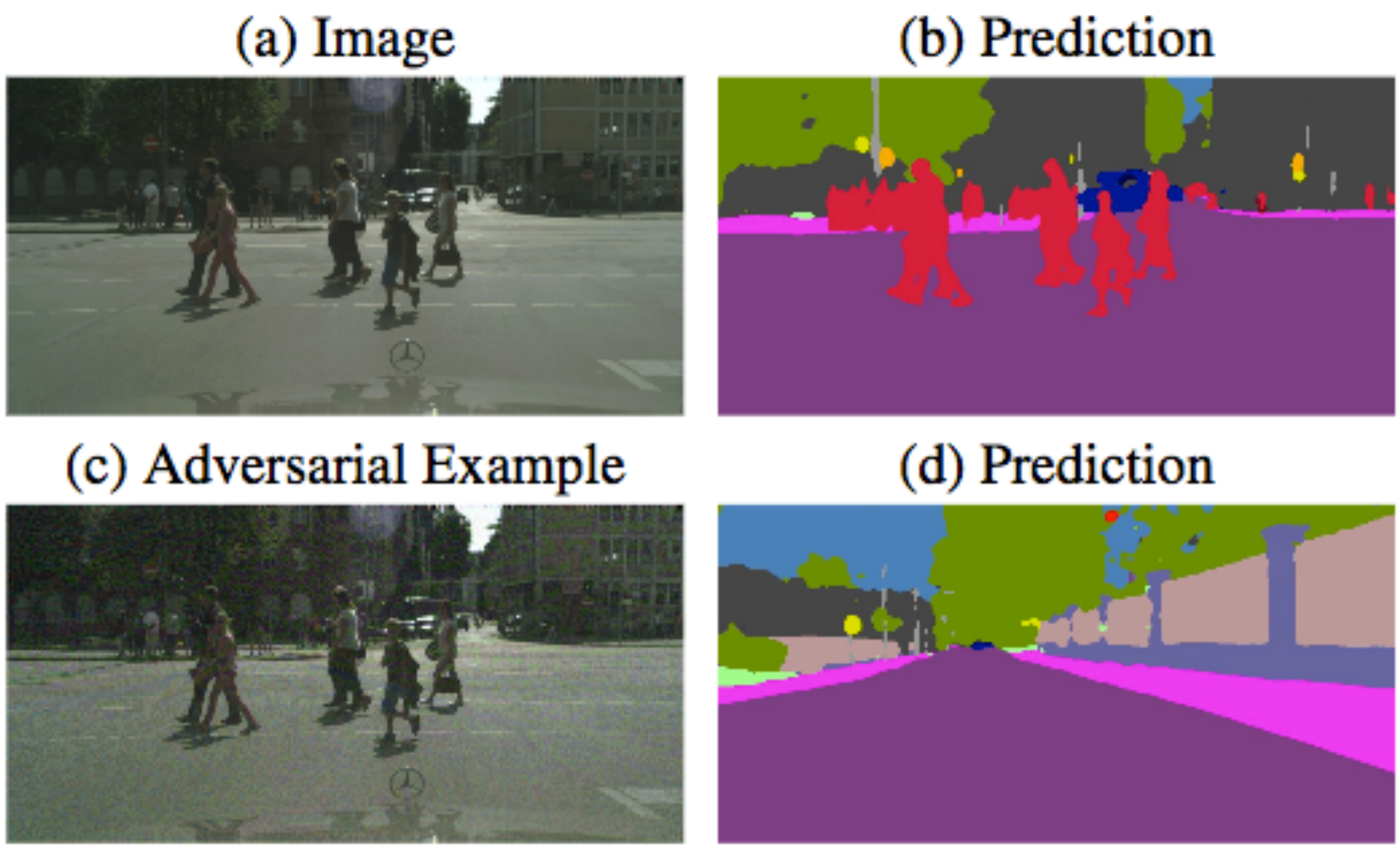
STOP SIGN ATTACKS

- Sticker attack, 85% mis-classification

arXiv: 1707.08945

PEDESTRIAN REMOVAL

- Noise added to remove pedestrians



arXiv: 1704.05712

HOW TO DO IT

- Usually during training we are updating the model weights to minimise the loss
- In adversarial attacks we increase the loss by altering the input data

FAST GRADIENT SIGN METHOD

- Calculate the gradient of the loss function for the input image
- Either go uphill (untargeted)
- Or downhill towards target class (targeted)
- Add (-1)**(is_targeted)*epsilon*sign(gradient) to each input pixel
- Implementation here: https://github.com/cleverhans-lab/cleverhans/blob/master/cleverhans/jax/attacks/fast_gradient_method.py

https://arxiv.org/abs/1412.6572

YOUR TURN!

- There is a new notebook for today on GitHub
- We'll be using the fast_gradient_method in the cleverhans library

PRE-TRAINED NETWORKS

- We're also going to be using a pre-trained neural network as the target of our attacks
- Many are available in keras, see here: https://keras.io/api/applications/
- Look at the time per inference step on CPU when deciding which to use!

NOTEBOOK FOR TODAY

https://github.com/abbeywaldron/cinvestav_ML_2024