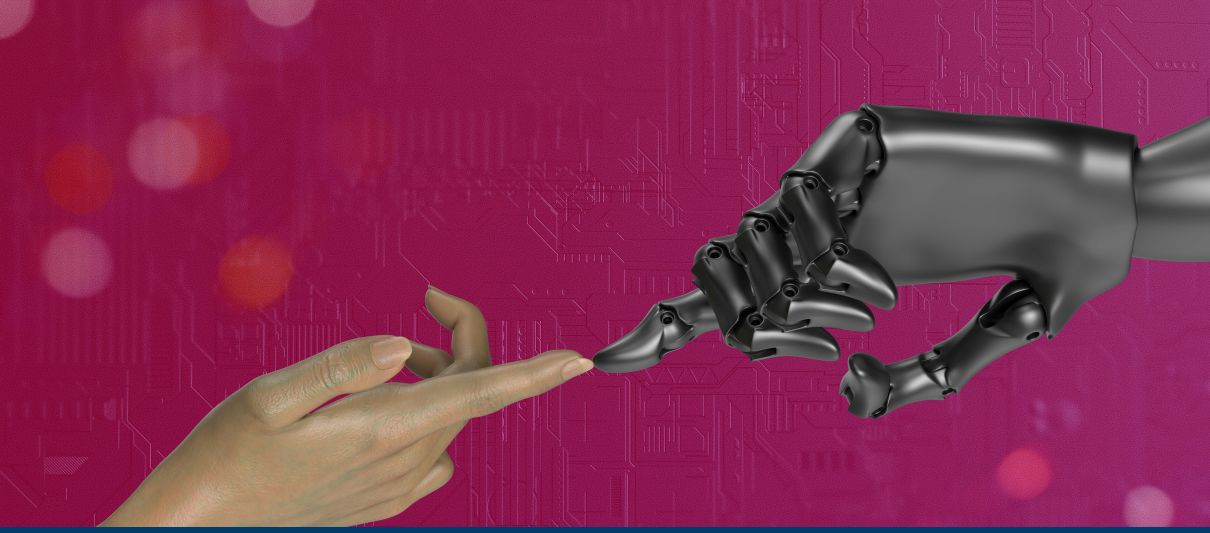

Hands-On Machine Learning

Dr. Derek Bridge | 2026/2027

UCC Coláiste na hOllscoile Corcaigh
University College Cork, Ireland

School of Computer Science and Information Technology
Scoil na Ríomheolaíochta agus na Teicneolaíochta Faisnéise

16. Keras

In this lecture, we again work with three datasets that we used previously — housing, CS1109 and Iris. But we also work with a dataset of images — the MNIST dataset, which contains handwritten digits.

The Keras library

- scikit-learn has very limited support for neural networks.
- There are many software frameworks that do support tensor computation, neural neworks and deep learning.
- Python examples: TensorFlow, PyTorch and JAX.
- We will use **Keras**, which is a high-level API, first released in 2015 by François Chollet then working for Google (https://keras.io), which has done a lot to make Deep Learning accessible to people.

Advantages

- It is very high-level, making it easy to construct networks, fit models and make predictions.
- It is multi-backend, meaning it works atop of whichever framework you prefer (TensorFlow, PyTorch or JAX).

Disadvantage

- The downside is it gives less fine-grained control than the backend framework itself.
- But, when fine-grained control is needed, you can mix in functions, methods and classes from the backend.

Specifying a Neural Network Architecture

- Keras neural networks are built from **layers**.
- Consecutive layers must be compatible: the shape of the input to one layer is the shape of the output of the preceding layer.
- Question: how many neurons will we use in the output layer of a network for (a) regression, (b) binary classification and (c) multiclass classification?
- We can choose the activation function of each layer.
- Question: which activation function will we typically use in hidden layers?
- Question: which activation function will we use in the output layer of a network for (a) regression, (b) binary classification and (c) multiclass classification?

Compiling the Neural Network

- Once the network is specified, we compile it.
- We provide a loss function.
- We provide an optimizer.
- We provide a list of metrics to monitor during training and testing.
- Question: which loss function will we use for (a) regression, (b) binary classification and (c) multiclass classification?

Training and Inference

- After compiling, we can train (fit).
- Then we can do inference (make predictions), including error estimation (evaluation on the test set).

Gradient Descent for Neural Networks

- We train a neural network using **Gradient Descent**.
- The loss function for a neural network is not **convex** — there may be, e.g., **local minima**.
- We tend to us **Mini-Batch Gradient Descent** — its 'bounciness' might escape a local minimum.

Optimizer

- In Keras, the optimizer selects the rule for updating the weights and biases.

Simulated Annealing

- Instead of setting the learning rate to a numeric value, in Keras we can specify a **learning schedule**, which modifies the learning rate during training.

Plateau Detection

- Separately, we can create a class that, during training, decreases the learning rate if the best validation loss does not improve for several consecutive epochs.

MNIST Dataset

- A dataset created by the National Institute of Standards and Technology (NIST) in the USA.
- In the modified version of the dataset (MNIST):
  - 60,000 training examples, 10,000 test examples
  - Grayscale images of handwritten digits from employees of the American Census Bureau and from American high school students.
  - Each image is 28×28 pixels.

Reshaping the Images