Barcelona Supercomputing Center
Centro Nacional de Supercomputación

HIGH PERFORMANCE ARTIFICIAL INTELLIGENCE

Deep Learning - MAI

Guided lab - CNNs

Dario Garcia Gasulla
dario.garcia@bsc.es

Outline

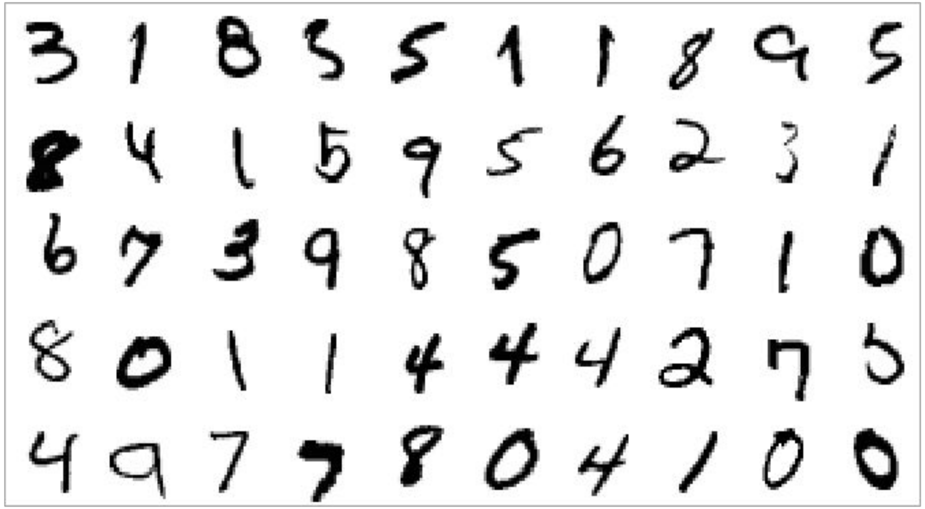
1. **Fully connected** networks applied to **MNIST**
2. **CNNs** applied to **MNIST**
3. **CNNs** applied to **CIFAR10**

MNIST

- MNIST is a *black and white* handwritten digit recognition dataset

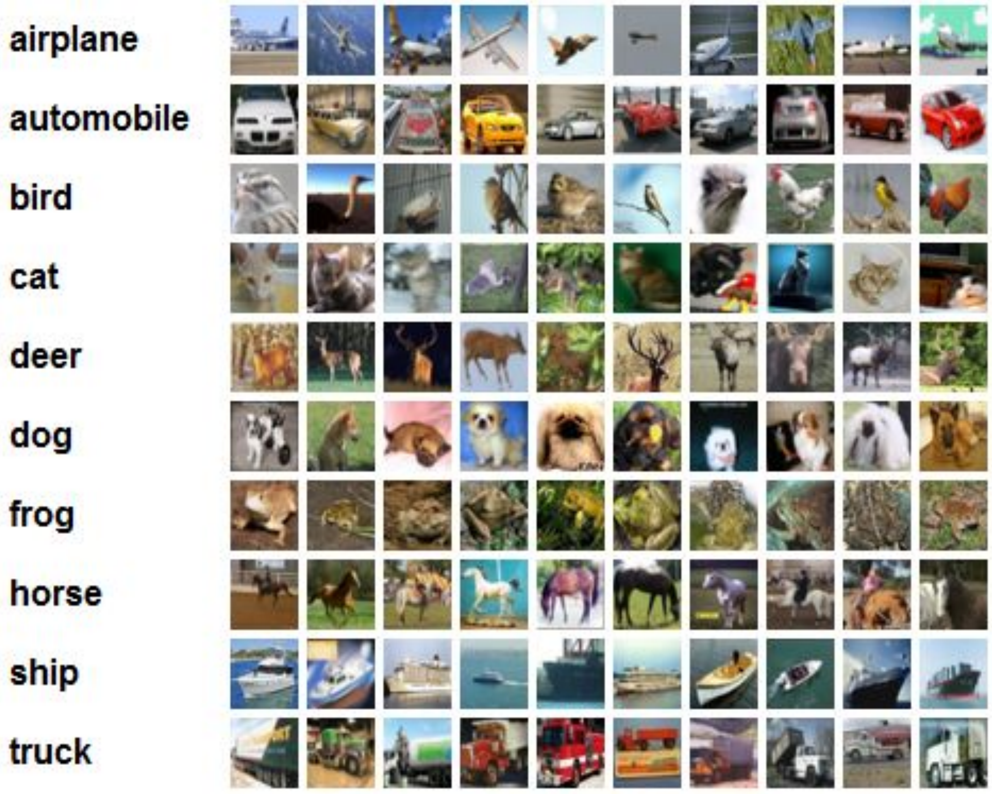
- First testing ground for new AI techniques

- See how far you can get using a fully connected network



CIFAR10

- CIFAR is a classification problem of low-resolution images (32x32)
- Version with 10 and 100 classes
- https://www.cs.toronto.edu/~kriz/cifar.html



Let’s look at the code

Get used to handling and loading data. It’s a big part of any DL experiment.

Look into *“flow_from_directory”* from keras to avoid memory issues, when loading large datasets.

Experiment 1 (FC & MNIST)

- Code
  - https://raw.githubusercontent.com/UPC-MAI-DL/UPC-MAI-DL.github.io/master/_codes/1.FNN-CNN/mnist_fnn_example.py
- Launcher
  - https://raw.githubusercontent.com/UPC-MAI-DL/UPC-MAI-DL.github.io/master/_codes/1.FNN-CNN/launcher.sh
- Resources
  - Copy necessary files from /gpfs/projects/nct00/nct00018
  - Within GPFS, store datasets in ~/.keras/datasets. Store models ~/.keras/models

“wget” to download from internet to your pc
“scp” to upload from your pc to P9

Experiment 2 (CNN & MNIST)

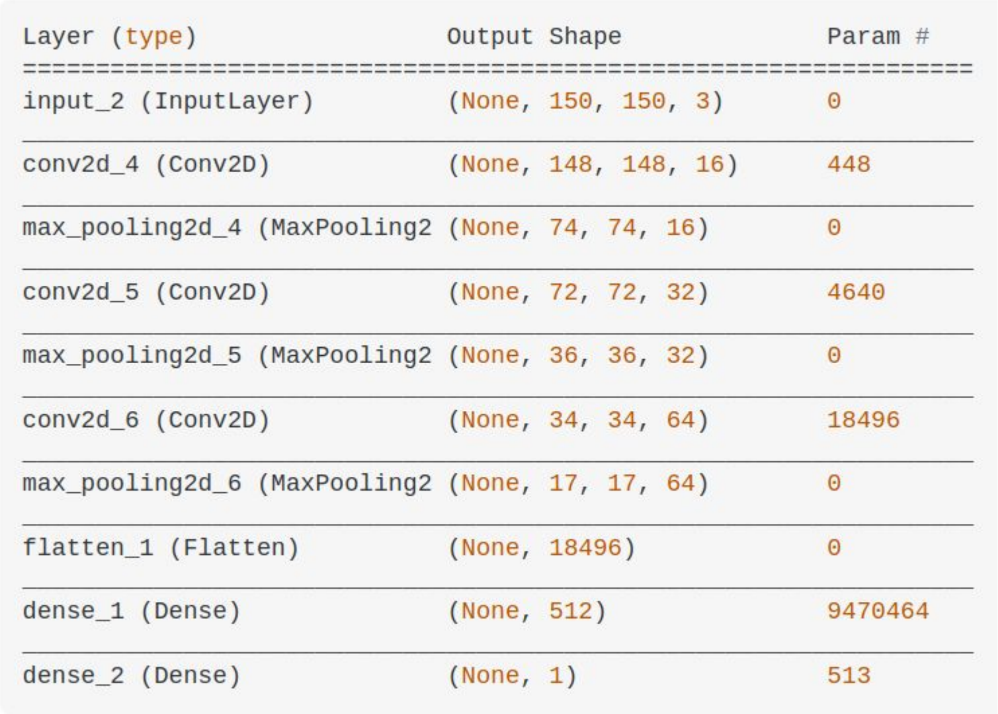
- Code
  - `https://raw.githubusercontent.com/UPC-MAI-DL/UPC-MAI-DL.github.io/master/_codes/1.FNN-CNN/mnist_cnn_example.py`
- Launcher
  - `Adapt the launcher for experiment 1`

Experiment 3 (CNN & CIFAR10)

- Code
  - Adapt the code from experiment 2
  - Notice data dimensions
- Launcher
  - Adapt the launcher for experiment 1
- Data
  - cifar-10-python.tar.gz Rename to cifar-10-batches-py.tar.gz and store in ~/.keras/datasets

Practical tips

- “tail -f file.out” to keep open for reading a live file
- model.summary()
  - track volumes
  - track complexity



```
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 150, 150, 3)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 148, 148, 16)      448
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 74, 74, 16)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 72, 72, 32)        4640
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 36, 36, 32)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 34, 34, 64)        18496
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 17, 17, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 18496)             0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               9470464
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 513
```

Dario Garcia-Gasulla (BSC)

dario.garcia@bsc.es

HIGH PERFORMANCE
ARTIFICIAL INTELLIGENCE