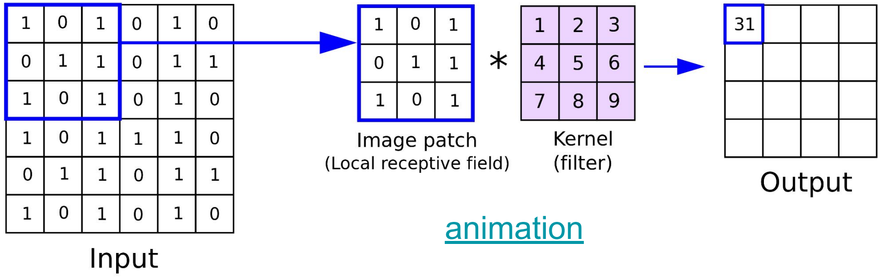
Image classification

Master InterUniversitario de Data Science
Santander, Spain
March 2022

Ignacio Heredia
iheredia@ifca.unican.es
Instituto de Física de Cantabria
(CSIC-UC)

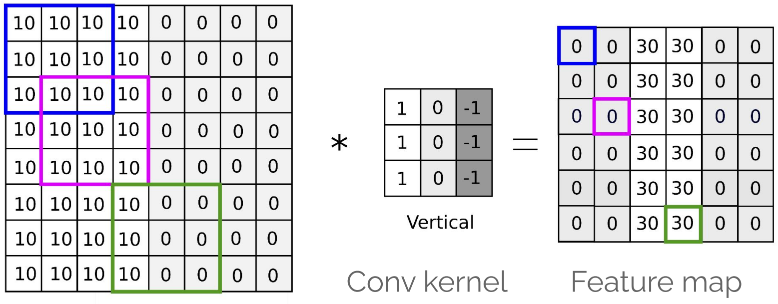
CNNs theory - Convolutional Layers

The convolution operation

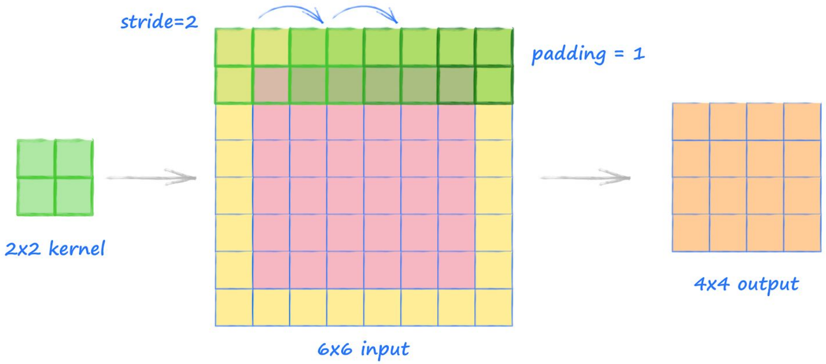


animation

Example. Edge detection



In addition to the kernel size you can set the **stride** (jump step) and the **padding** (fill border with zeros).
Animations: [1], [2], [3]

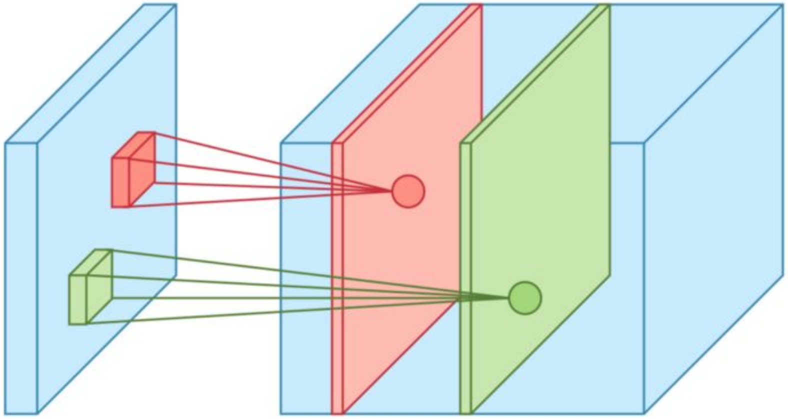


We have to repeat the operation across the depth of the feature map.

animation

CNNs theory - Convolutional Layers

Inside the same convolutional layer we can have *multiple kernels*. Each kernel is specialized in one task and is learnt through backpropagation. Each kernel will produce a different feature map.

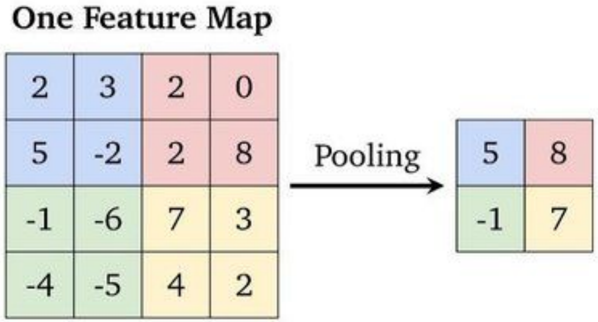


The convolution operation is *spatially invariant*, making it well suited for images (ie. we want to detect edges no matter where they are located).

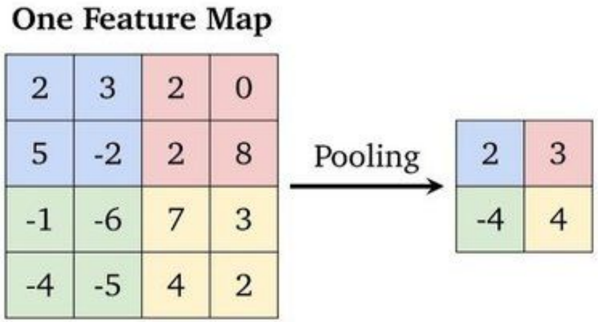
Additionally, reusing parameters (one simple 3X3 kernel instead of full dense layer) reduces the networks size and avoids overfitting.

Convolutional **N**eural **N**etwork**s** have been nevertheless successfully used with other types of data (eg. time series).

CNNs theory - Pooling Layers



(a) Max-Pooling (stride 2)



(b) Average-Pooling (stride 2)

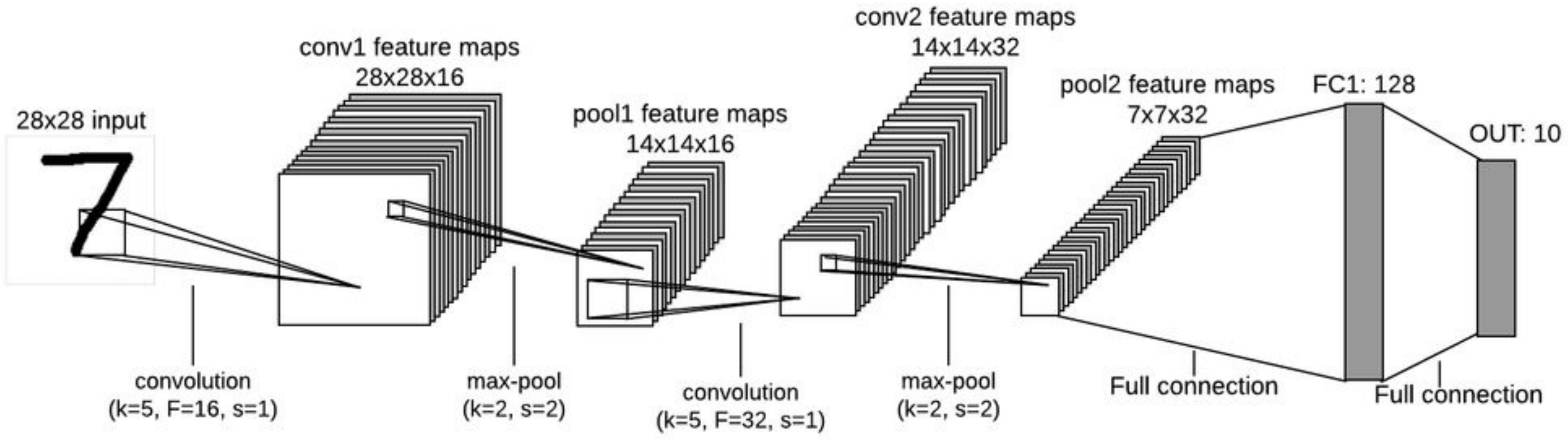
Pooling layers reduce the spatial dimensions of the feature maps (the outputs of the convolutional layers).

They can be omitted if one uses conv layers with large stride.

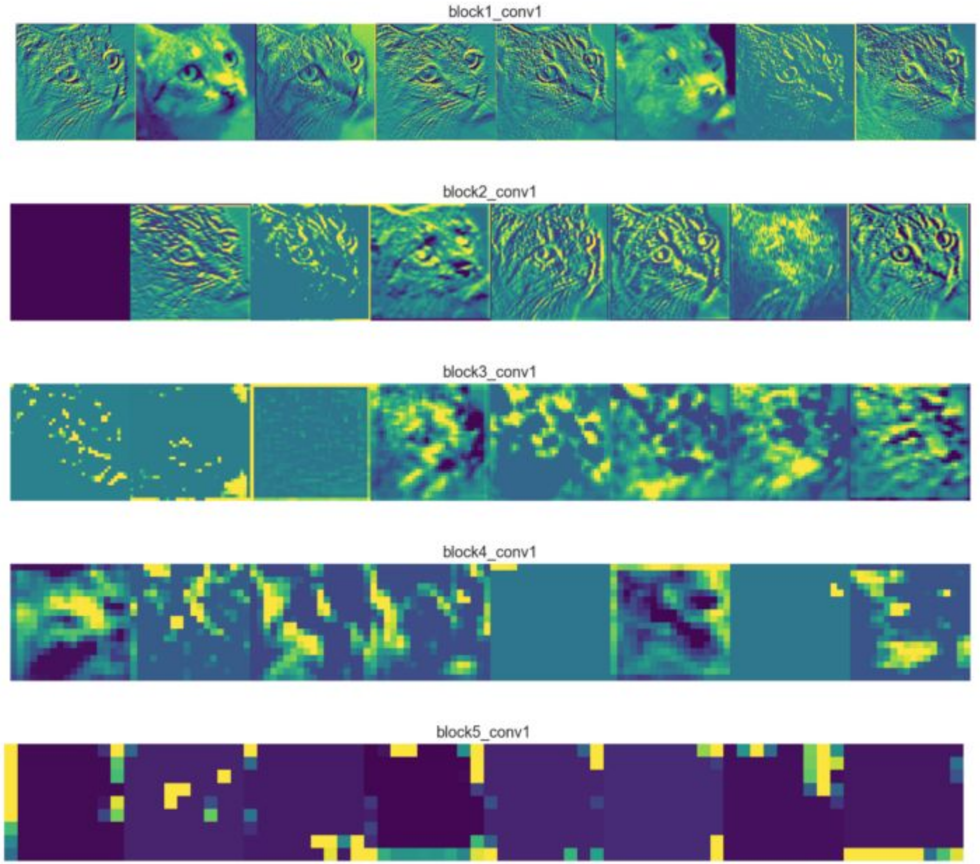
CNNs theory - Overall structure

The idea is to use alternatively **convolutional** layers and **pooling** layers (optional) until we reach the situation of having *many feature maps* with *low spatial size*. You can stack several conv layers (two 3✕3 is better than one 5✕5) before each pooling.

We flatten the last feature maps and apply some **dense** layers till we reach the output. Then apply softmax if we are doing classification.



CNNs theory - Overall structure

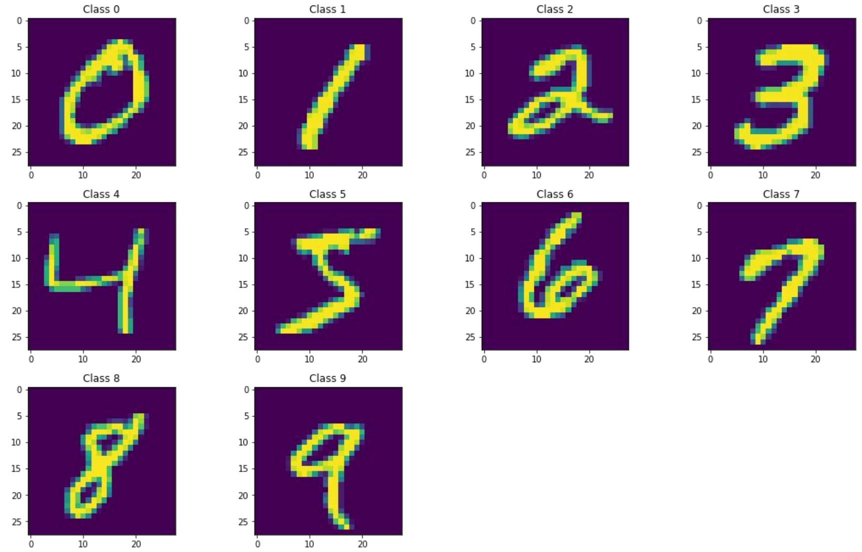


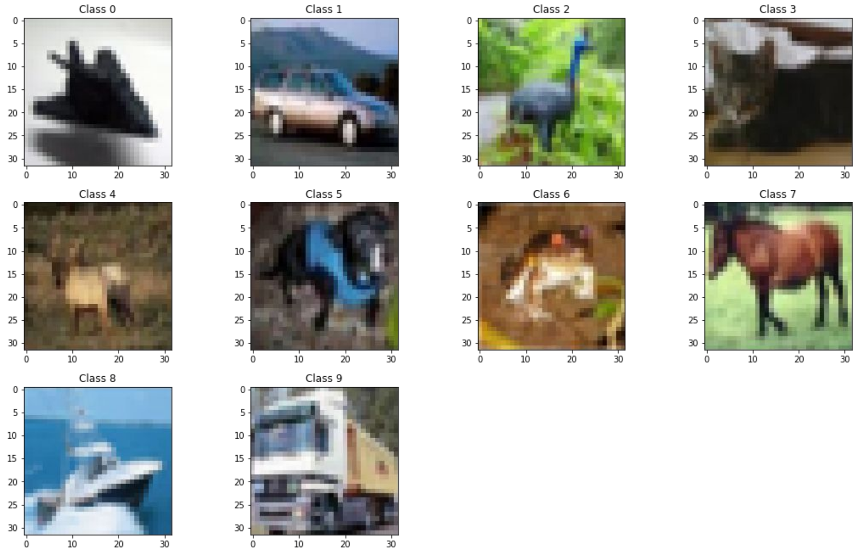
eg. detect simple edges

Progressively, as one goes deeper into the CNN, feature maps should encode *more complex/abstract* information.

eg. detect cat tail

Exercise 1 - Visualization





Exercise 2 - Create a CNN

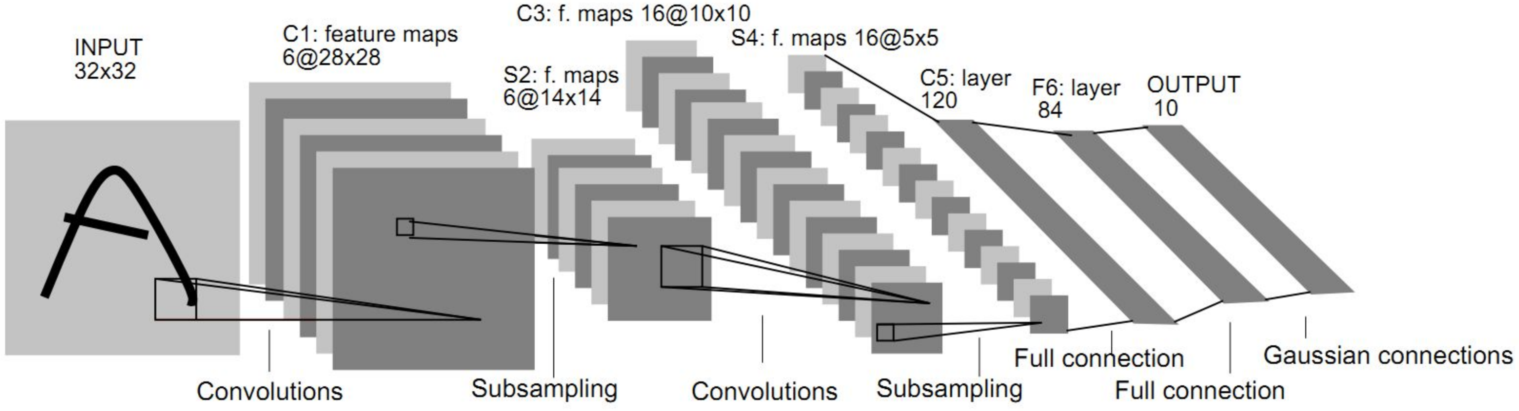


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Exercise 3.1 - Test the functions

Regularizers

```
0) None
    Loss: 2.2, Acc: 0.511
1) l1_l2
    Loss: 1.5e+02, Acc: 0.126
2) l2
    Loss: 6.6, Acc: 0.126
3) l1
    Loss: 1.3e+02, Acc: 0.099
```

Initializers

```
0) he_uniform
    Loss: 2.2, Acc: 0.364
1) RandomNormal
    Loss: 2.2, Acc: 0.275
2) he_normal
    Loss: 2.2, Acc: 0.231
3) TruncatedNormal
    Loss: 2.3, Acc: 0.213
4) glorot_uniform
    Loss: 2.3, Acc: 0.211
5) lecun_uniform
    Loss: 2.3, Acc: 0.198
6) RandomUniform
    Loss: 2.3, Acc: 0.18
7) VarianceScaling
    Loss: 2.3, Acc: 0.174
8) glorot_normal
    Loss: 2.4, Acc: 0.116
9) lecun_normal
    Loss: 2.3, Acc: 0.11
10) Ones
    Loss: 1.4e+01, Acc: 0.11
11) Zeros
    Loss: 2.3, Acc: 0.099
```

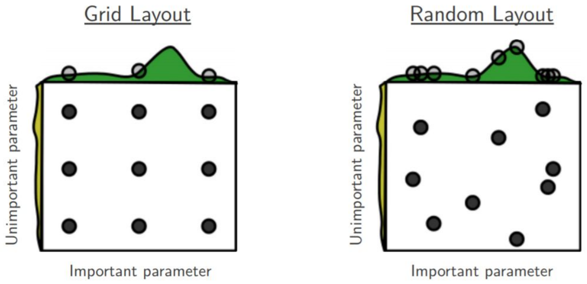
Optimizers

```
0) Adagrad
    Loss: 0.25, Acc: 0.926
1) Nadam
    Loss: 0.26, Acc: 0.921
2) Adamax
    Loss: 0.36, Acc: 0.889
3) RMSprop
    Loss: 0.41, Acc: 0.88
4) Adam
    Loss: 0.41, Acc: 0.871
5) Adadelta
    Loss: 1.2, Acc: 0.601
6) SGD
    Loss: 2.2, Acc: 0.371
```

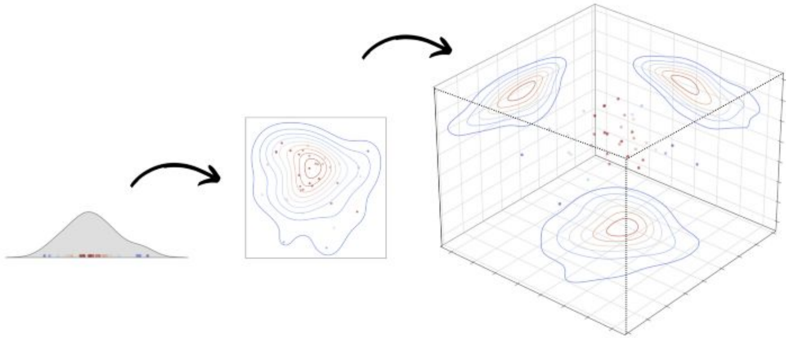
Exercise 3.2 - Hyperparameter search

Possible strategies

- Grid search ★
- Random search ★★
- Bayesian guided search ★★★



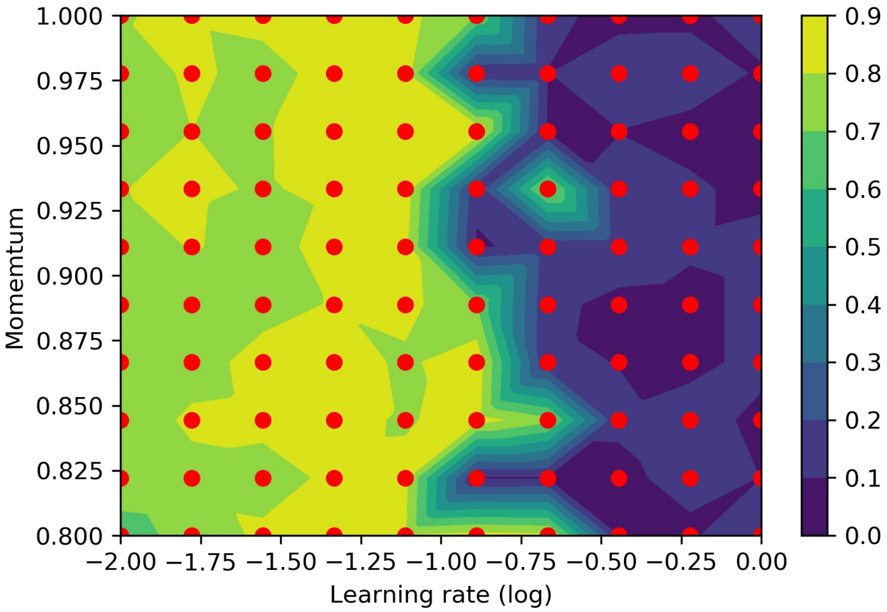
Bergstra, J., & Bengio, Y. (2012). [Random search for hyper-parameter optimization.](#) Journal of machine learning research, 13(2).



Search space grows with the increased dimensionality of permissible hyperparameters ([ref](#))

Exercise 3.2 - Hyperparameter search

Grid search



Random search