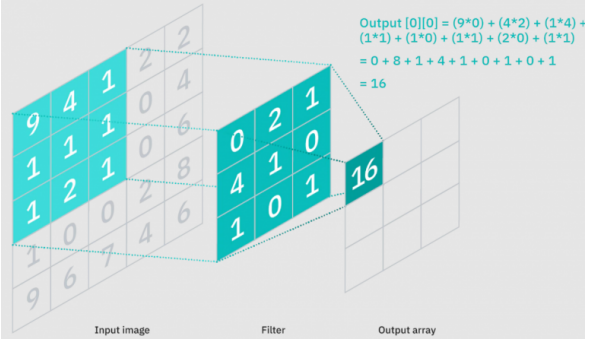
Convolutional Neural Networks



Lluís Talavera

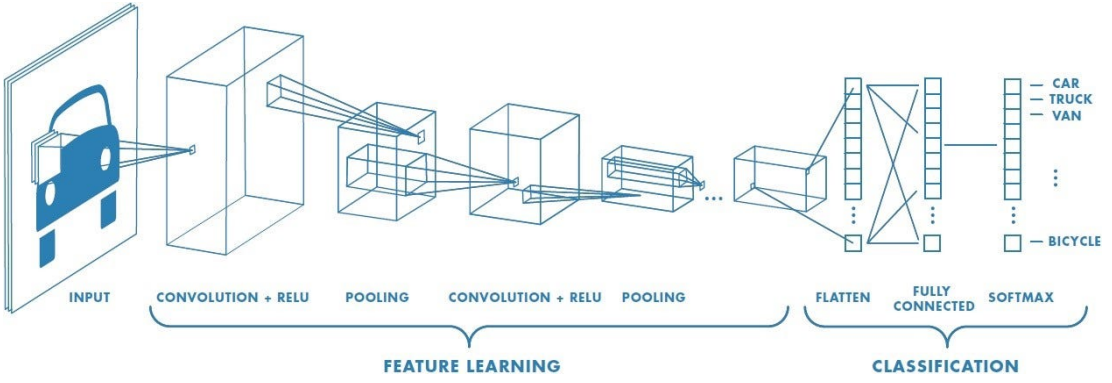
UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH
UPC
ETSEIB

Convolutional Neural Networks (CNN)

CNNs learn a large number of filters in parallel specific to a dataset and particular predictive modeling task, such as image classification. Each filter creates a feature map that summarizes the detected features in the input.

Multiple convolutional layers allows a hierarchical decomposition of the input. For example, if the input is an image, they can extract lines, shapes and even faces at deeper levels.

Dimensionality reduction is achieved by using pooling filters.

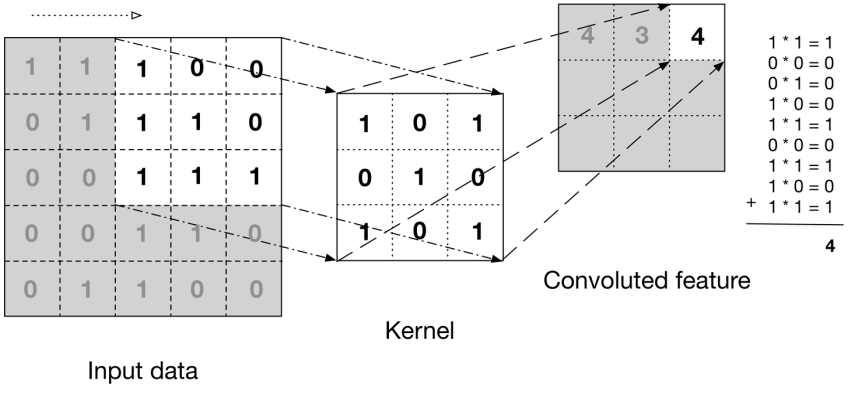


The convolution operation

Initialize randomly a filter (kernel), i.e., a small "window", usually 3x3 pixels. The values of the filter are changed during training.

Apply (overlap) the filter onto the image and compute the dot product (convolution).

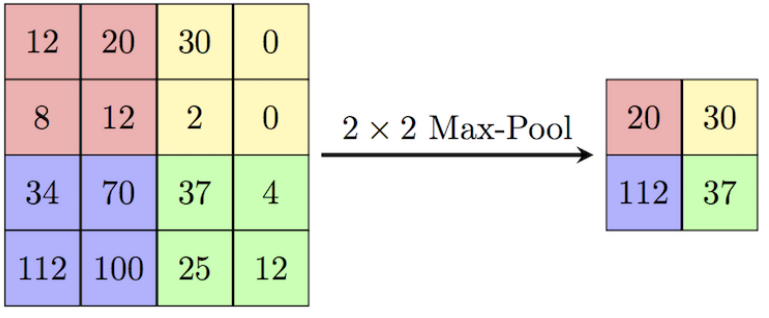
Add a bias and add the result to a feature map. Complete the feature map by sliding the filter and then apply the activation function.



Max-pooling

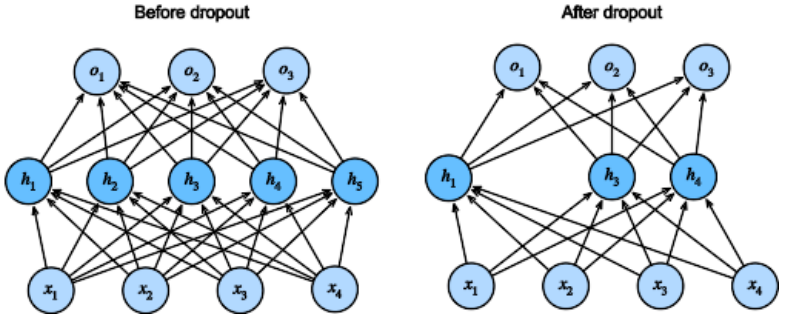
The goal is to reduce the dimensionality.

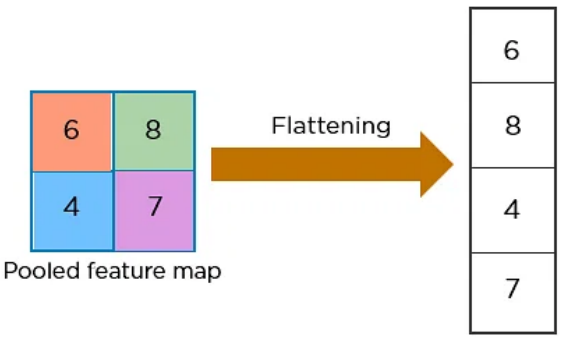
Apply a filter, usually 2x2, without overlapping and select the max value of each region.



Depending on the task, other pooling methods may be applied, e.g., average pooling.

Additional layers: Dropout and Flatten

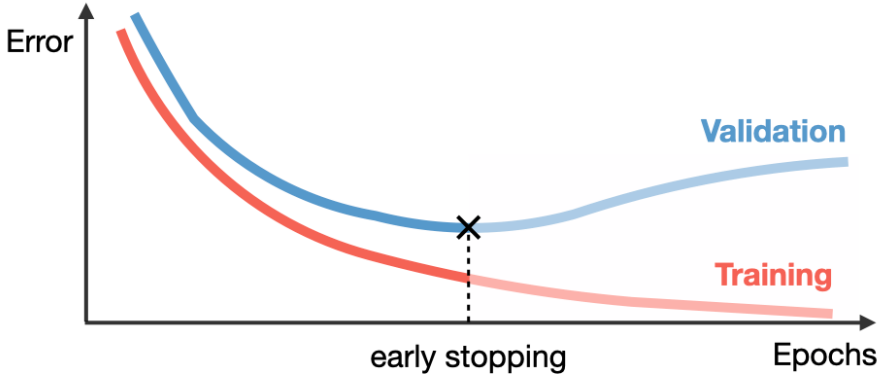




Early stopping

We split the dataset into the training and testing set, as usual. Then, we separate part of the training set to use it as a validation set during learning. If the error on the training and validation set diverges, it can suggest overfitting.

One way of avoiding overfitting is using early stopping.



Decisions...

How many CNN layers?

For each CNN layer:

- Number of filters
- Kernel size (usually 3x3)
- Activation function (usually ReLu)
- Padding (valid, same, full)

For each max-pooling layer:

- pool size (usally 2x2)

Does the model suffers from overfitting? We can add dropout and/or early stopping

Do we need flattenning?

THe keras library

Keras is a library run on top of TensorFlow framework for numerical computation and large scale machine learning. Its focus is on deep learning tecnhiques.

We will use:

- the sequential model, a linear stack of layers.
- Dense layers: each neuron receives input from all neurons of the previous layer.
- Convolution2D layers.
- Additional layers (Dropout, Flatten)
- We need to compile the model to specify some parameters (loss function, optimizer, evaluation metrics) and then we fit the model to the training (and validation) data.

A code example with keras

```
model = Sequential()

model.add(Convolution2D(32, (3,3), activation='relu', input_shape=(28,28,1)))
model.add(Convolution2D(32, (3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))

model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

model.fit(X_train, Y_train,
          batch_size=32, epochs=10, verbose=1)
```