# Convolutional Neural Networks

Tamás Grósz

# The problem with fully connected networks

- Each neuron see the whole input, each input has the same importance
- If we shuffle the input, then fully connected DNNs will achieve approximately the same accuracy

# The problem with fully connected networks

- Each neuron see the whole input, each input has the same importance
- If we shuffle the input, then fully connected DNNs will achieve approximately the same accuracy
- All these means that it disregards local information (the ordering of the inputs)

# The problem with fully connected networks

- Each neuron see the whole input, each input has the same importance
- If we shuffle the input, then fully connected DNNs will achieve approximately the same accuracy
- All these means that it disregards local information (the ordering of the inputs)
- Sometimes the local information are inportant, for example: images, speech etc.

# The problem with fully connected networks

- Each neuron see the whole input, each input has the same importance
- If we shuffle the input, then fully connected DNNs will achieve approximately the same accuracy
- All these means that it disregards local information (the ordering of the inputs)
- Sometimes the local information are inportant, for example: images, speech etc.
- A further problem is that simple DNNs are not translation invariant

# Convolution

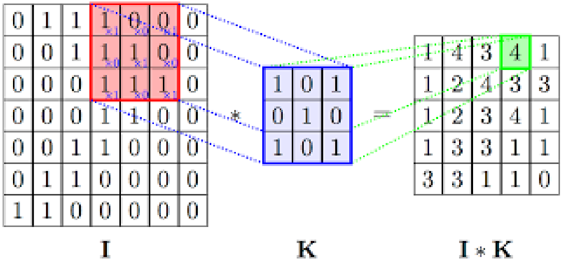
## Convolution

Convolution is a mathematical operation on two functions (I and K) to produce a third function that expresses how the shape of one is modified by the other. It is a popular tool in image processing.

# Convolution

## Convolution

Convolution is a mathematical operation on two functions (I and K) to produce a third function that expresses how the shape of one is modified by the other. It is a popular tool in image processing.



# Convolutional Neuron

- Each convolutional neuron (CN) processes data only for its small receptive field
- These CNs are convolved with the input, with the same weights (kernel)
- Each neuron has multiple output activation value
- CNs try to learn to recognize small local features and they are translation invariant as the are evaluated in every positions
- Two important parameters:
  **kernelsize** determines the size of its receptive field
  **stride** controls how the filter convolves around the input volume.

# Padding

What happens when you apply three 5 x 5 x 3 filters to a 32 x 32 x 3 input volume?

# Padding

What happens when you apply three 5 x 5 x 3 filters to a 32 x 32 x 3 input volume?

- The output volume would be 28 x 28 x 3. Notice that the spatial dimensions decrease.

# Padding

What happens when you apply three 5 x 5 x 3 filters to a 32 x 32 x 3 input volume?

- The output volume would be 28 x 28 x 3. Notice that the spatial dimensions decrease.
- If we keep applying conv layers, the size of the volume will continue to decrease.

# Padding

What happens when you apply three 5 x 5 x 3 filters to a 32 x 32 x 3 input volume?

- The output volume would be 28 x 28 x 3. Notice that the spatial dimensions decrease.
- If we keep applying conv layers, the size of the volume will continue to decrease.
- To avoid this padding "pads" the input volume with some valuse around the border. (zero padding: padd with zeros)

# Pooling

What happens when you apply ten 5 x 5 x 3 filters to a 32 x 32 x 3 input volume, with padding and stride 1 x 1?

# Pooling

What happens when you apply ten 5 x 5 x 3 filters to a 32 x 32 x 3 input volume, with padding and stride 1 x 1?

- The output volume would be 32 x 32 x 10. What if we use more than 10 neurons?

# Pooling

What happens when you apply ten 5 x 5 x 3 filters to a 32 x 32 x 3 input volume, with padding and stride 1 x 1?

- The output volume would be 32 x 32 x 10. What if we use more than 10 neurons?
- The pooling layer, is used to reduce the spatial dimensions.

# Pooling

What happens when you apply ten 5 x 5 x 3 filters to a 32 x 32 x 3 input volume, with padding and stride 1 x 1?

- The output volume would be 32 x 32 x 10. What if we use more than 10 neurons?
- The pooling layer, is used to reduce the spatial dimensions.

## Pooling layer

Pooling layer is also referred to as a downsampling layer. It basically takes a filter (normally of size 2 x 2) and a stride of some length. Pooling then applies it to the input volume and outputs some number (avg, max) in every subregion.

# Pooling

The advantages of pooling:

- By having less spatial information you gain computation performance
- Less spatial information also means less parameters, so less chance to overfit
- The neurons become translation invariant (to some degree)

# Pooling

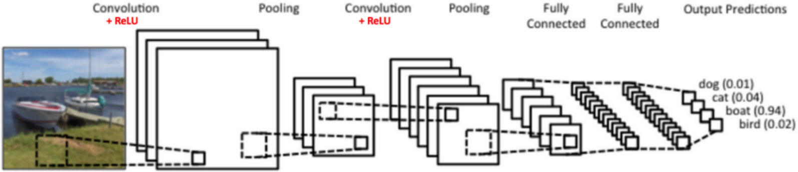
The advantages of pooling:

- By having less spatial information you gain computation performance
- Less spatial information also means less parameters, so less chance to overfit
- The neurons become translation invariant (to some degree)

During training the optimizer needs to propagate the gradient trough this layer! To aid this, if max pooling is aplied the position of the max needs to be stored, as the gradient will propagate in its direction.

# CNNs

A traditional CNN structure consists of convolutional layers folowed by pooling layers, and at the end some fully connected layers:

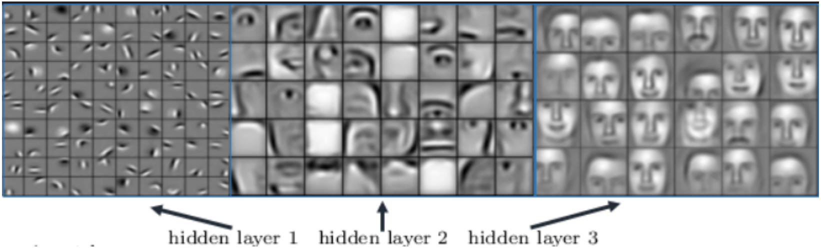


# CNNs

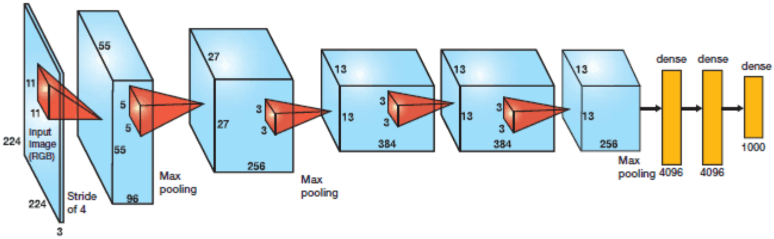
Higher conv layers are expected to extract higher features.

# CNNs

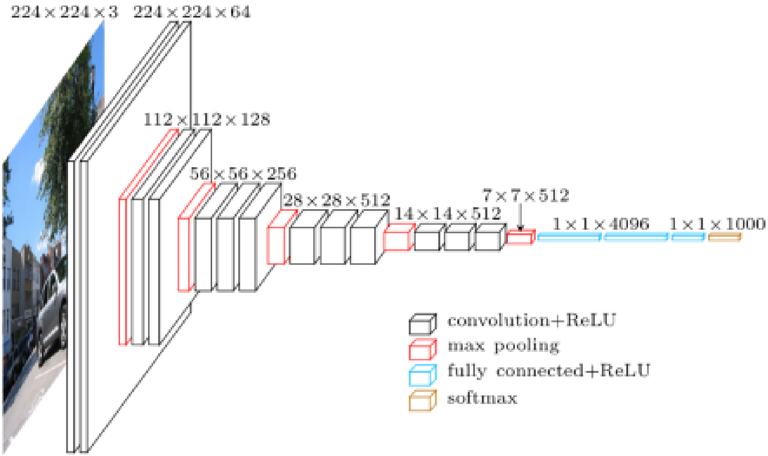
Higher conv layers are expected to extract higher features.



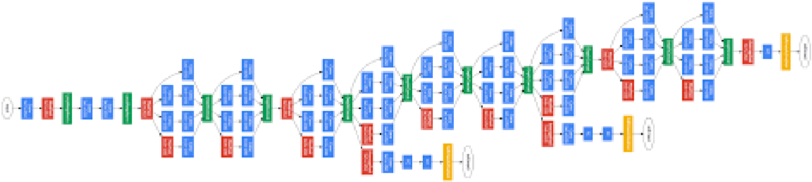
# Famous CNNs: AlexNet



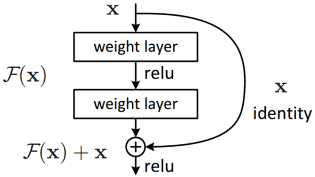
# Famous CNNs: VGG



# Famous CNNs: GoogLeNet

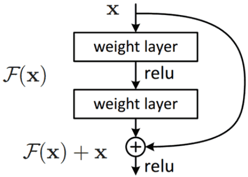


# Famous CNNs: ResNet

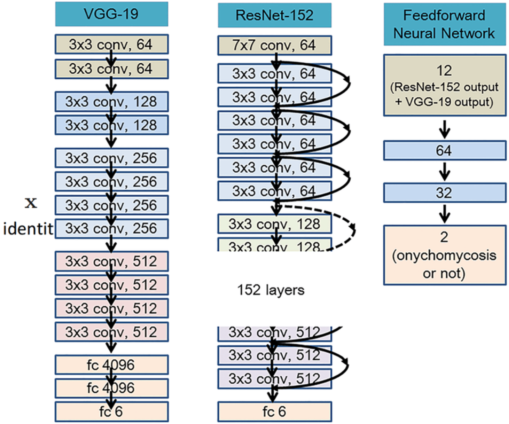



A residual block

# Famous CNNs: ResNet




A residual block



# Practice

Python tutorial: project_demo_en_colab.ipynb