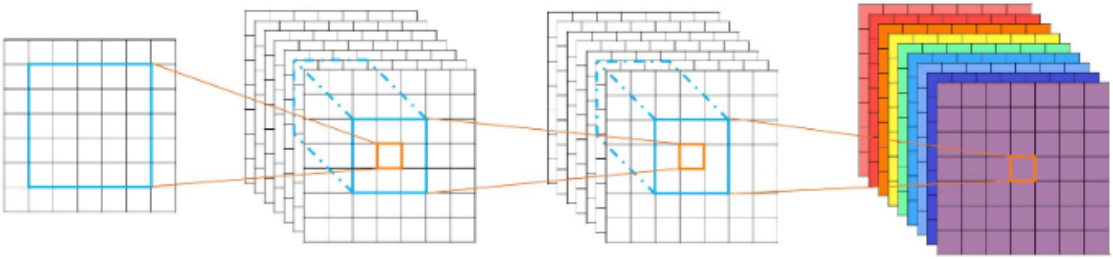
Neural Networks and Deep Learning: Convolution Hierarchies & Convolutional Neural Networks

Nicolas Thome

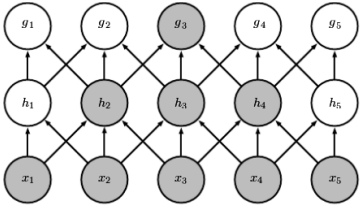
Conservatoire National des Arts et Métiers (Cnam)
Département Informatique

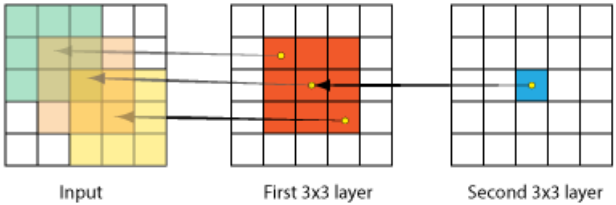
Convolution Hierarchies



- Convolution Layer: affine filtering + non-linear activation
- **Convolution Hierarchies:** stacking Convolution Layers
- **Motivation: depth increase modeling capacities**
  - Non-linearity crucial: hierarchical model $\neq$ flat model!
    $\sim$ fully connected network

Convolution Hierarchies: Receptive Field





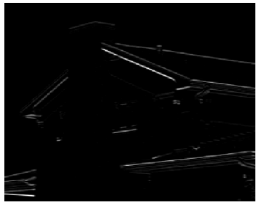
- Cascading two 3×3 convolutions: same receptive field as 5×5 convolution in input image
- **Convolution Hierarchies:**
  - **Feature combination**
  - **Gradual increase of spatial receptive field**
    $\Rightarrow$ **indirect global connectivity**

Convolution Hierarchies: Example

Edge detection with convolution hierarchy (pyramid) ⇒ two layers:

- Input: gray-scale image I ⇒ $W \times H \times 1$ tensor

$I_x^2 \sim$ filter 1



$I_y^2 \sim$ filter 2

1. 1^{st} layer: convolution with two filters

$$M_x = \frac{1}{4} \cdot \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad M_y = \frac{1}{4} \cdot \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

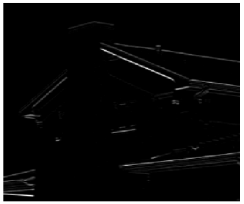
   - followed by non-linearity: $\sigma(z) = z^2$

⇒ Output: $W \times H \times 2$ tensor, $H_1 \sim (I_x)^2$, $H_2 \sim (I_y)^2$

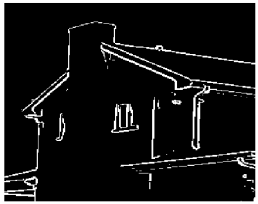
Convolution Hierarchies: Example

Edge detection with convolution hierarchy (pyramid): two-Layers

I_x^2



I_y^2



output

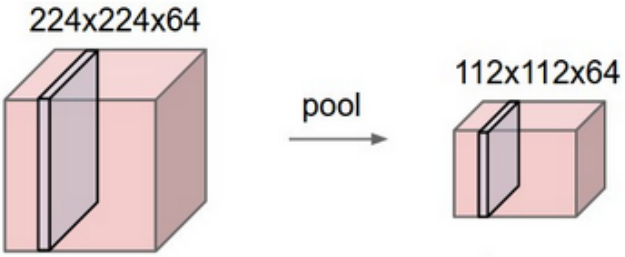
2. 2^{nd} layer: convolution with one 1×1 filter $[1\ 1]$
   - For each pixel: $(I_x)^2 + (I_y)^2 = \|\vec{G}(x,y)\|^2 = \|\vec{\nabla_I}\|^2$
   - $\sigma(z) = Step(z - T)$, T threshold on $\|\vec{G}(x,y)\|^2$

$\Rightarrow$ Output: $W \times H \times 1$ tensor $\Rightarrow$ **edge detector**

Pooling in Convolution Layer

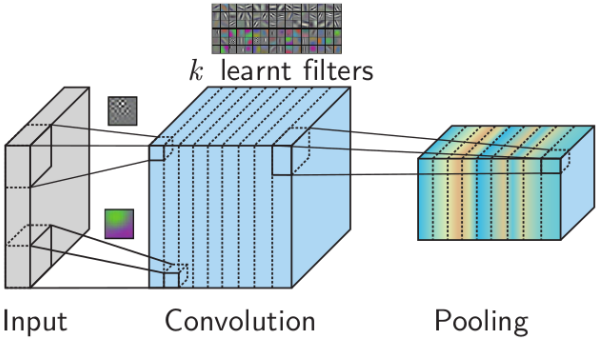
Where to pool in a convolution tensor?

- Most common choice: pool in each feature map independently
  $\Rightarrow$ **spatial pooling** on top of convolution Layer
  - Input / Output: a tensor of depth D
  - Output smaller spatial size (pooling stride)

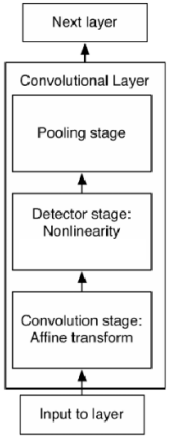


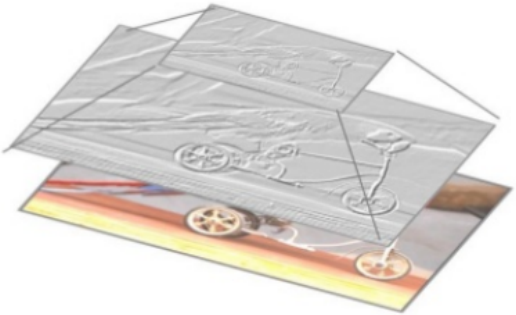
Convolution / Pooling Layer

- Pooling on top of convolution Layer
- **An elementary block:** Convolution Layer + Pooling [Conv-Pool]



Convolution / Pooling Layer

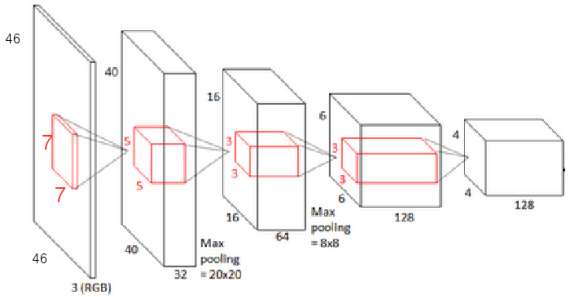




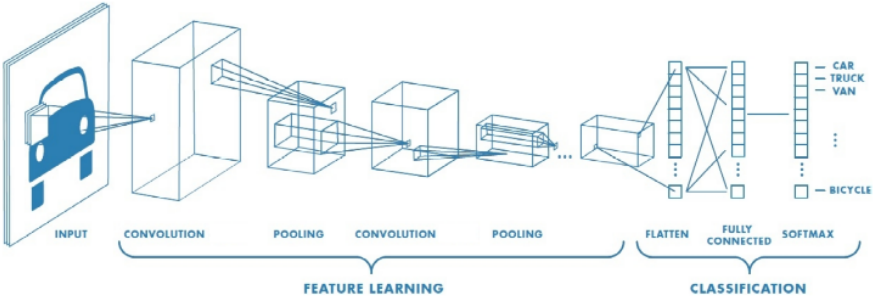
- **An elementary block:** $\overbrace{\text{Convolution Layer}}^{\text{Convolution + Non linearity}} + \text{Pooling}$

Convolutional Neural Networks (ConvNets)

- Stack several Convolution / Pooling blocks
  $\Rightarrow$ **Convolutional Neural Network (ConvNet)**
- **Ex:** 7×7 convolution
  2×2 pooling area, stride $s = 2$
- Input image 46×46, 1^{st} [Conv-Pool] Layer:
  - Conv output $= 40$
  - Pooling output $= 20$
  - **Receptive field size for each pooled unit?**
    $\Rightarrow$ **Pooling $\uparrow$ receptive field**

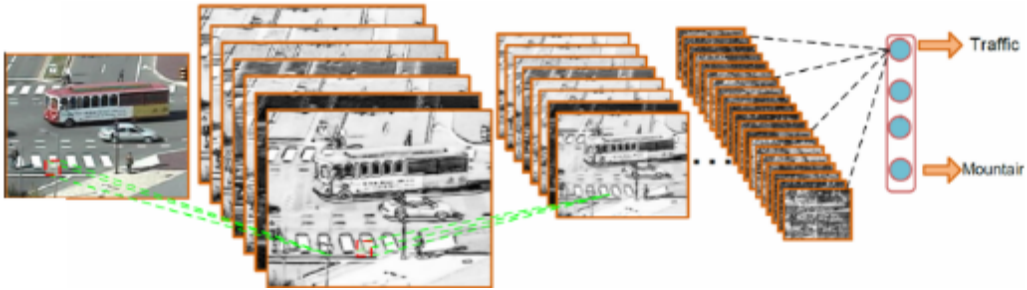


Convolutional Neural Networks (ConvNets)



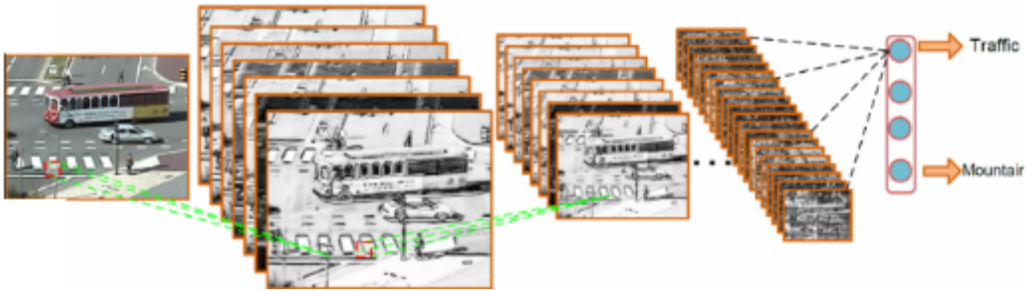
- ConvNets: hierarchical tensor modification
- At some (depth) point, often flattening input tensor
  $\Rightarrow$ vector

Convolutional Neural Networks (ConvNets)



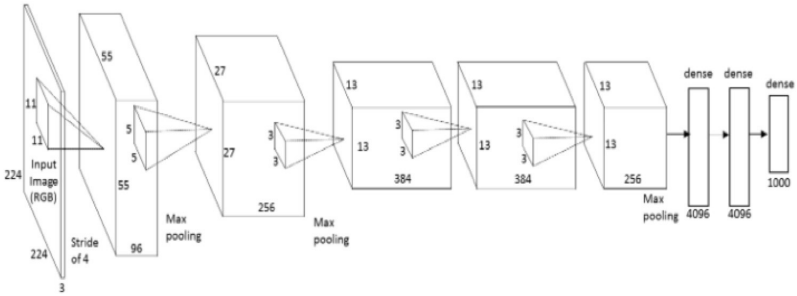
- ConvNet prediction: 2-stage process:
  1. Representation learning with [Conv-Pool] hierarchy:
     - Conv detects relevant features
     - Pool gains spatial invariance for classification

Convolutional Neural Networks (ConvNets)



- ConvNet prediction: 2-stage process:
  2. Classification: Tensor flattened ⇒ vector
     - Flattening: neuron position in initial tensor
       ⇒ **breaking translation invariance**
     - Followed by a hierarchy of fully connected layers

Conclusion



- ConvNet: hierarchical [Conv-Pool] + fully connected
  - Architecture for famous historical Nets, *e.g.* LeNet, or more recent, *e.g.* AlexNet (2012)
- **Deep Learning History?⇒ following!**