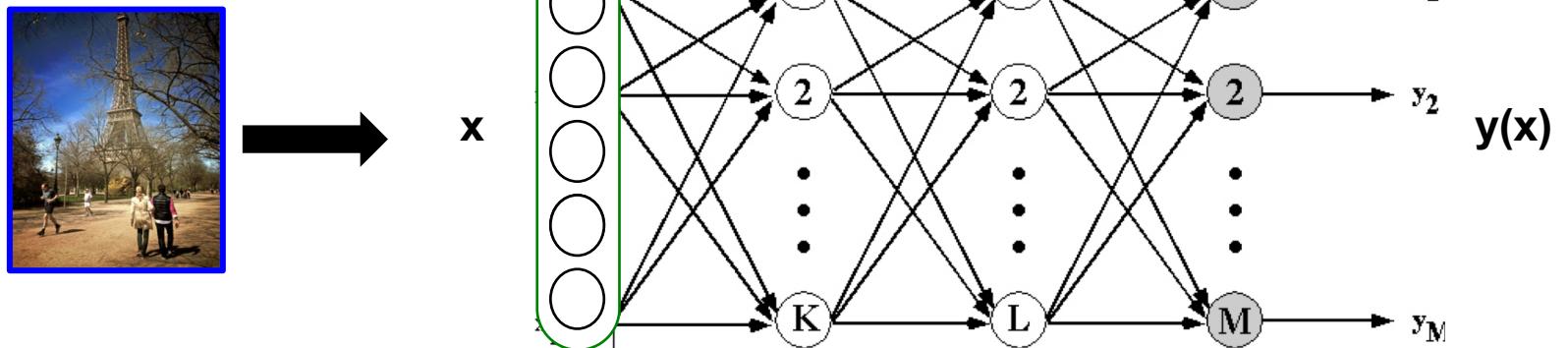


Neural Nets for computer vision

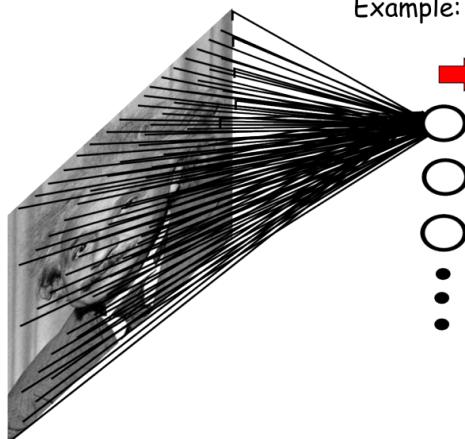
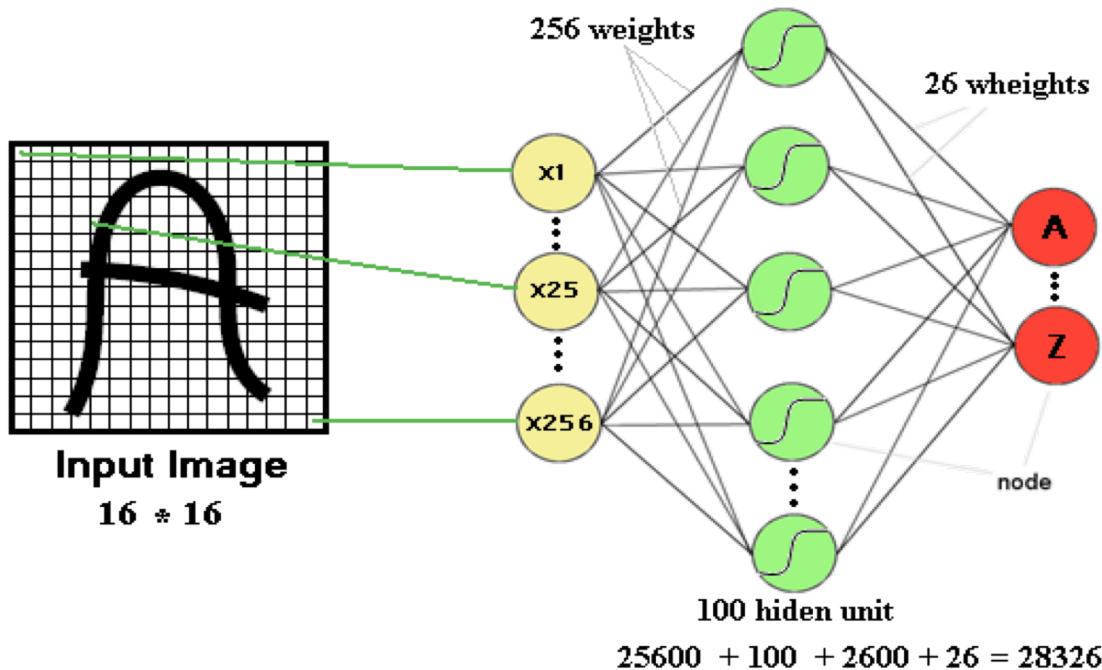
Matthieu Cord LIP6 / SU

Outline

1. Recap MLP
2. Convolutional Neural Networks
 - Basics for deep in images



MLP example: brute force connection



First Pb: Scalability

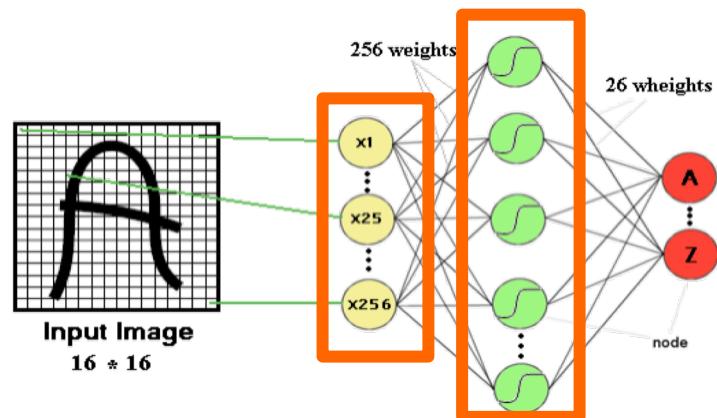
Large images => extremely large number of trainable parameters

MLP example: brute force connection

2d Pb: *Stability* of the representation

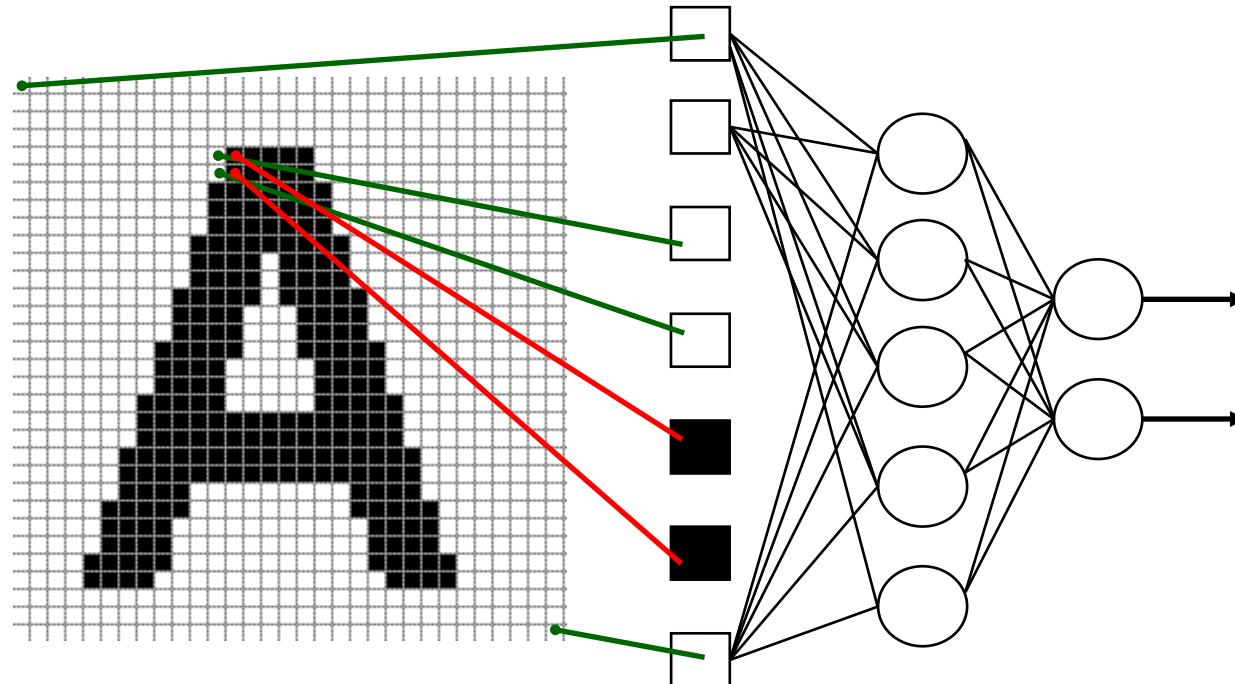
Expectation:

- *Small deformation in the input space*
=> *similar representations*
- *Large (or unexpected) transfo in the input space*
=> *very dissimilar representations*



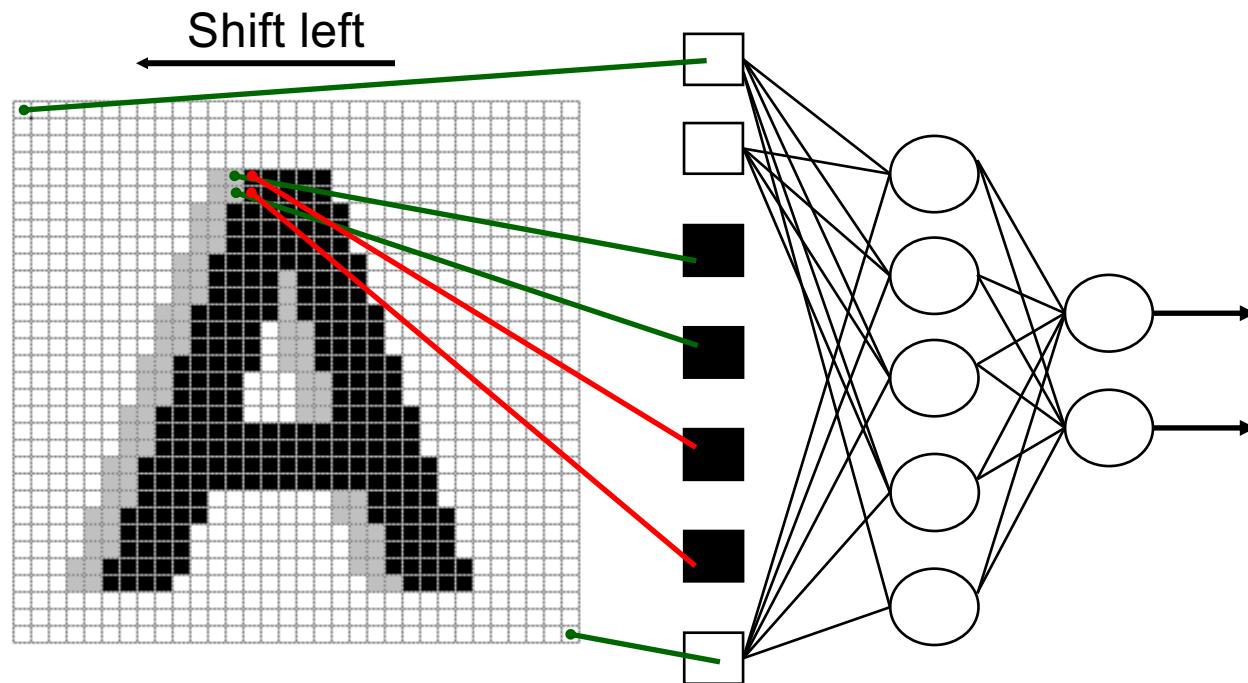
MLP example: brute force connection

Stability: Invariance/Robustness to (local) shifting, scaling, and other forms of (small) distortions?

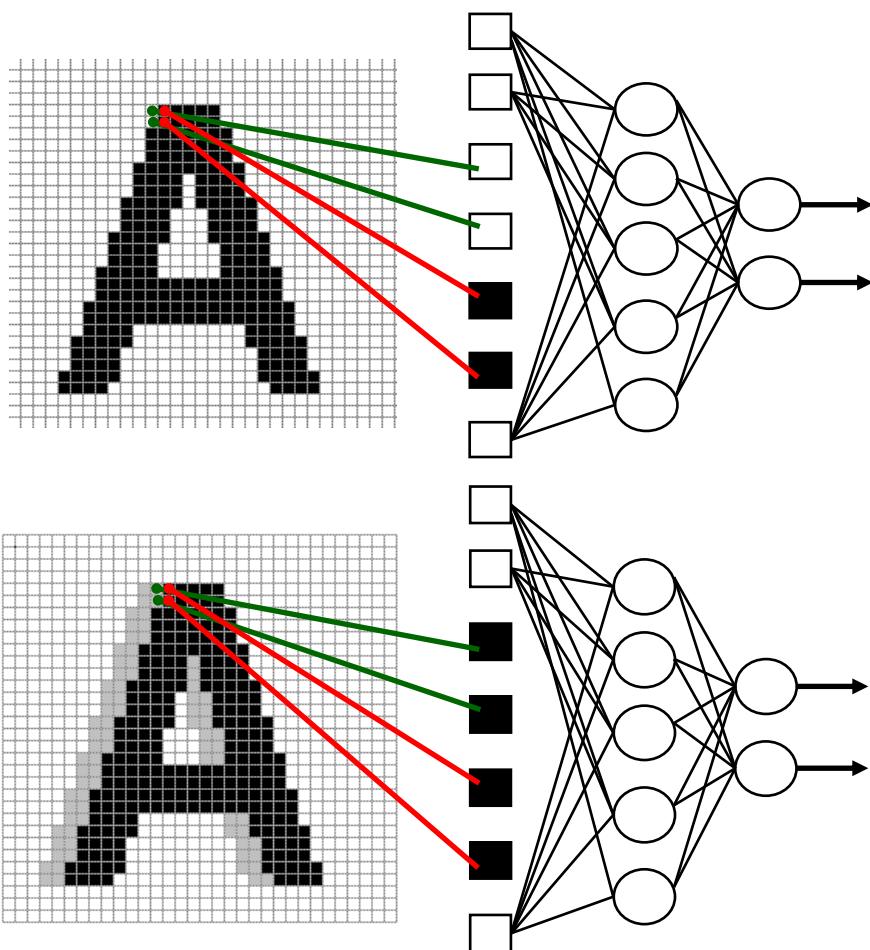


MLP example: brute force connection

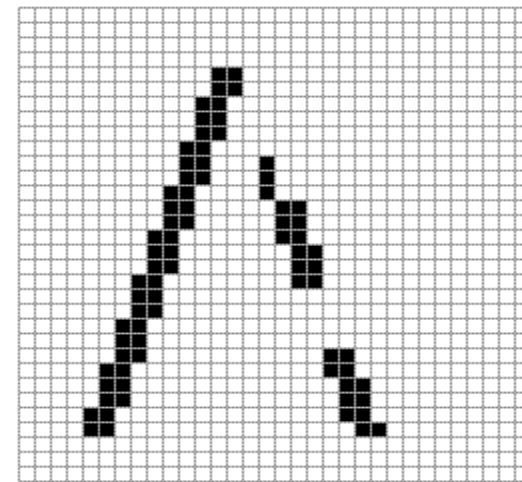
Little or no invariance to shifting, scaling, and other forms of distortion



MLP example: brute force connection



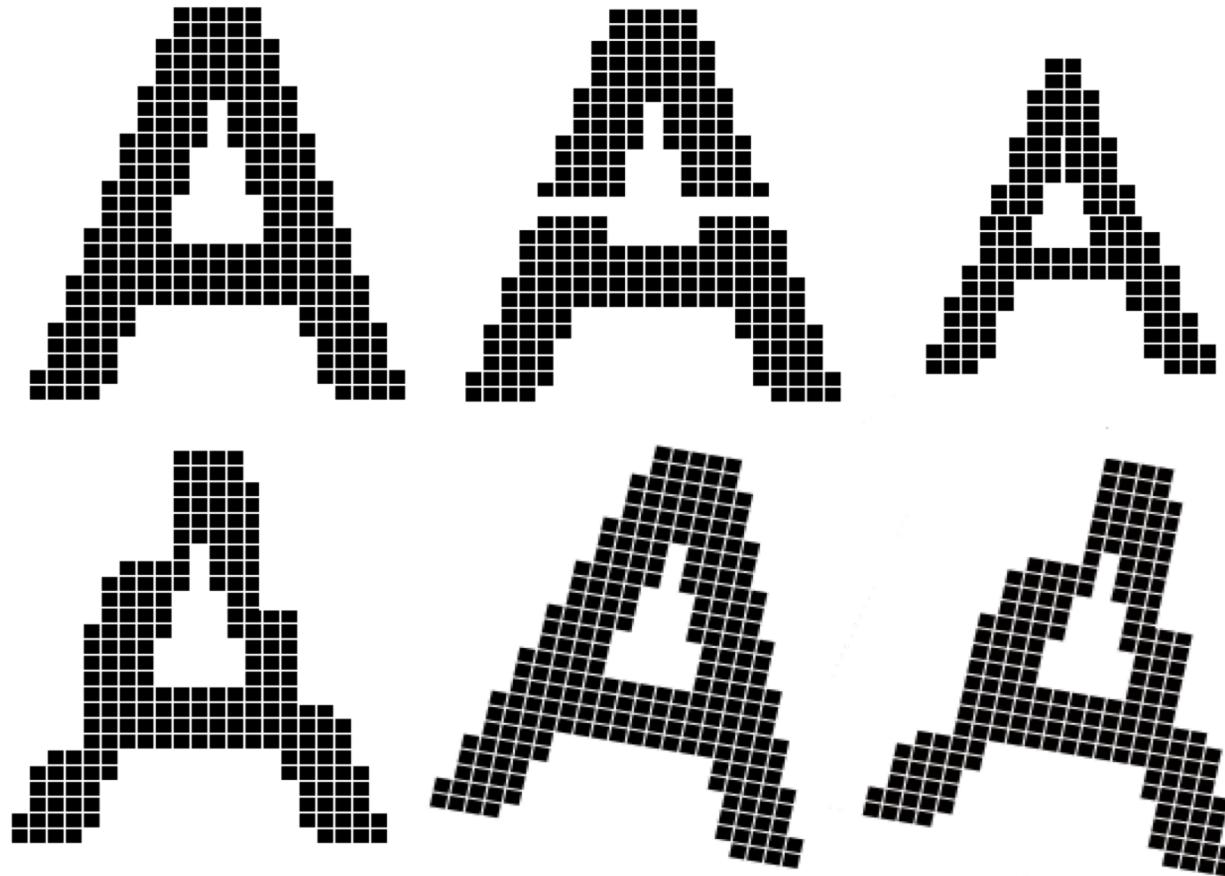
@LeCun



154 input change
from 2 shift left
77 : black to
white
77 : white to
black

MLP example: brute force connection

Scaling and other forms of distortions => same pb



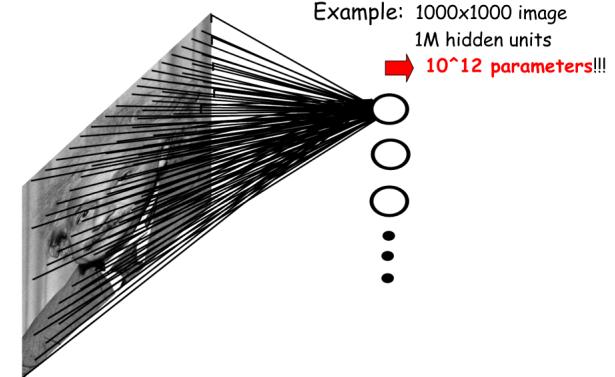
Conclusion of MLP on raw data

Brute force connection of images as input of MLP NOT a good idea

- No Invariance/Robustness of the representation because topology of the input data completely ignored
- Nb of weights grows largely with the size of the input image

How keep spatial topology?

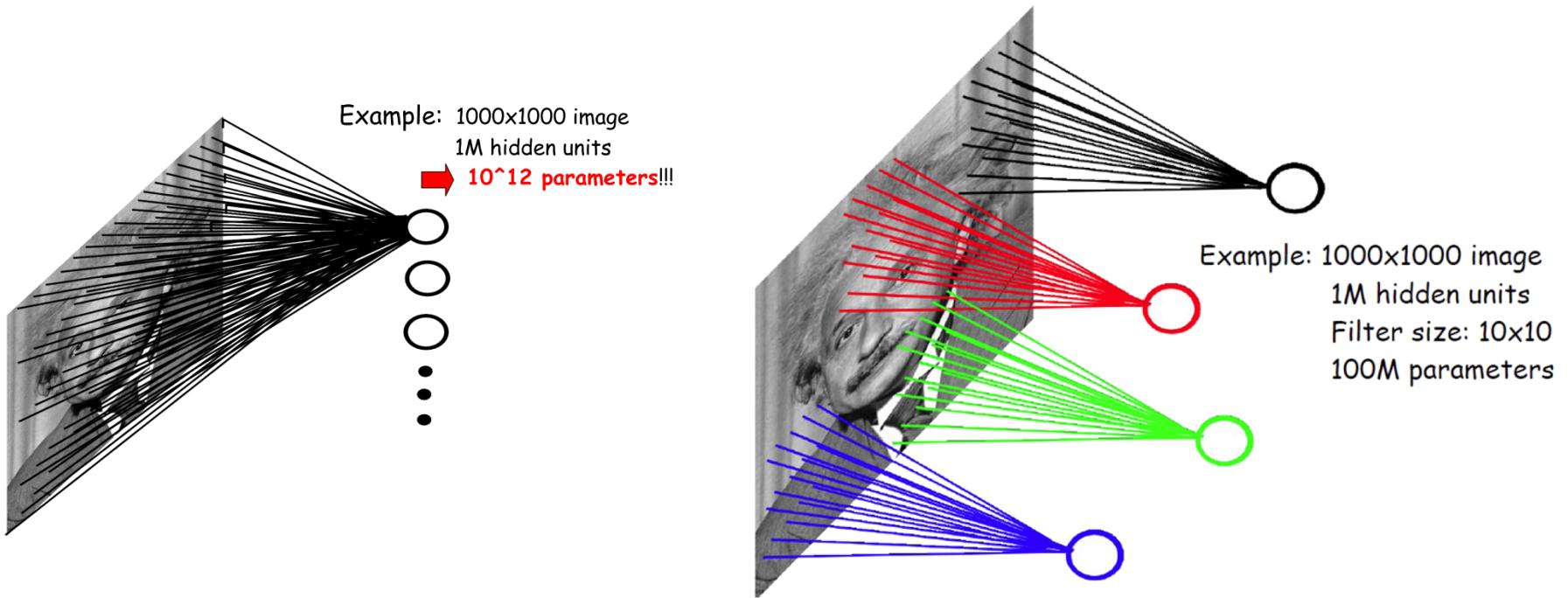
How to limit the weight number?



How to limit the weight numbers?

1/ Locally connected neural networks

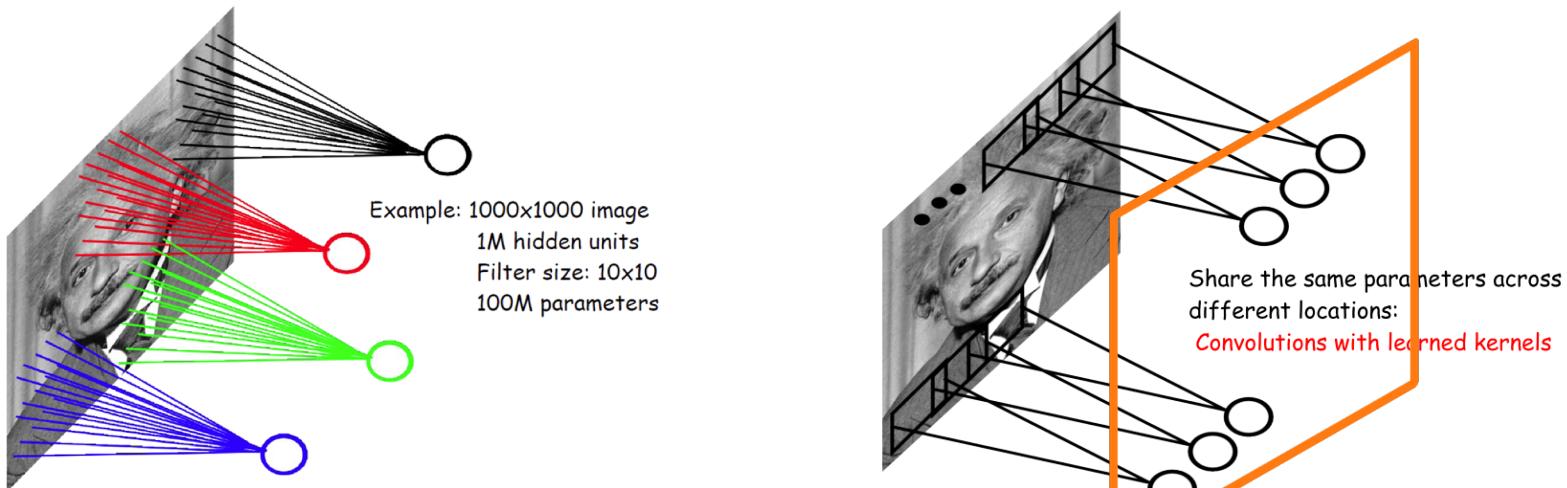
- **Sparse connectivity:** a hidden unit is only connected to a local patch (weights connected to the patch are called filter or kernel)
- Inspired by biological systems, where a cell is sensitive to a small sub-region of the input space, called a receptive field. Many cells are tiled to cover the entire visual field



How to limit the weight numbers?

2/ Shared Weights

- Hidden nodes at different locations share the same weights
 - greatly reduces the number of parameters to learn
- Keep spatial information in a **2D feature map** (hidden layer map)



- ⇒ Computing responses at hidden nodes equivalent to convoluting input image with a linear filter (learned)
- ⇒ A learned filter as a feature detector

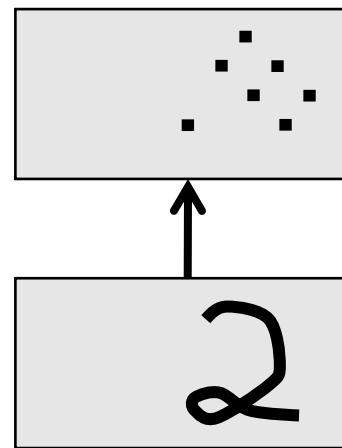
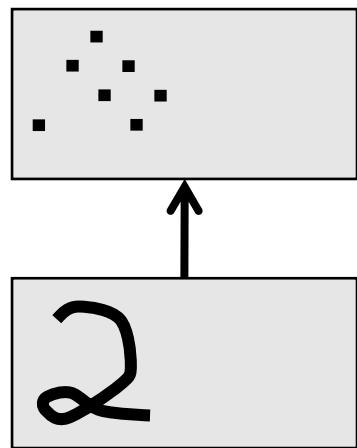
What does replicating the feature detectors achieve?

- Equivariant activities (Hinton Ex): Replicated features do not make the neural activities invariant to translation. The activities are equivariant.

Map representation
by one filter

AAA
AAA

image



translated
representation

translated
image

⇒ How to get invariance to 2D spatial transformation of the input?

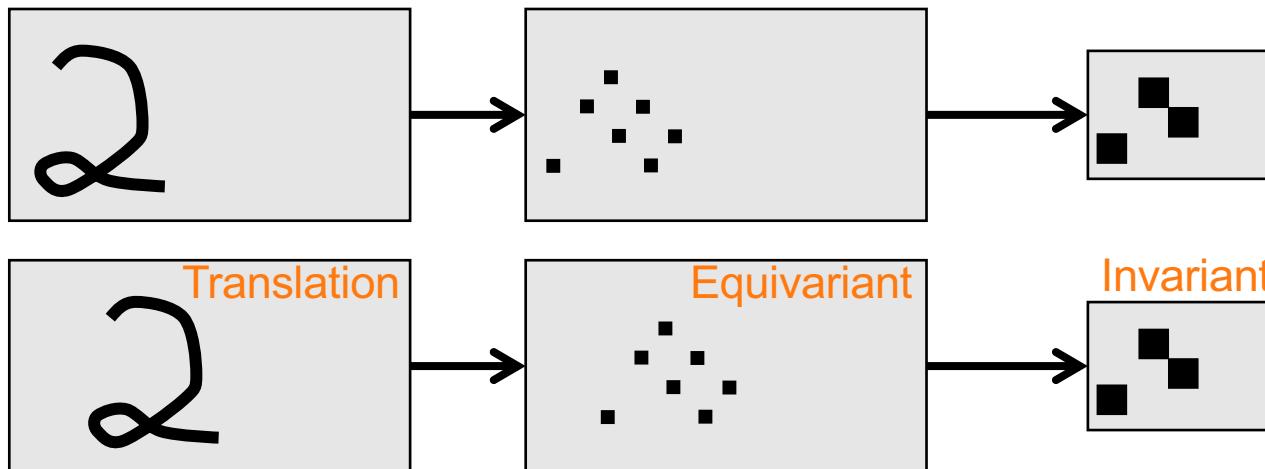
Getting local Invariance

3/ (local) spatial **POOLING** of the outputs of replicated feature detectors:

- Averaging neighboring replicated detectors to give a single output to the next level
- Max pooling: Taking the maximum in a neighboring

Get a small amount of translational invariance at each level

Reducing the number of inputs to the next layer of feature extraction



=> Stability OK (at least for local shift) for Convolutional Net!

Ex. of convolution operator and pooling operator

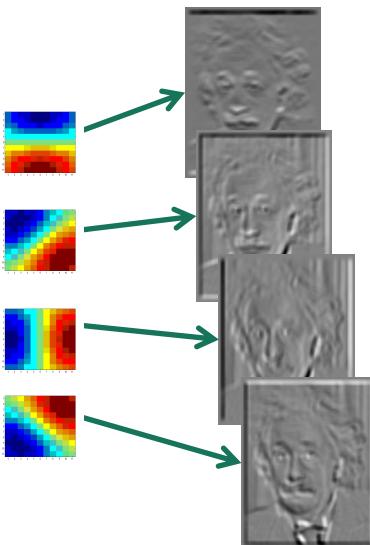
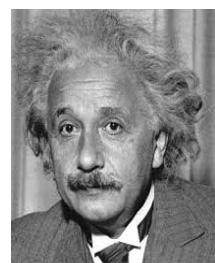
Convolution

2D

35	40	41	45	50
40	40	42	46	52
42	46	50	55	55
48	52	56	58	60
56	60	65	70	75

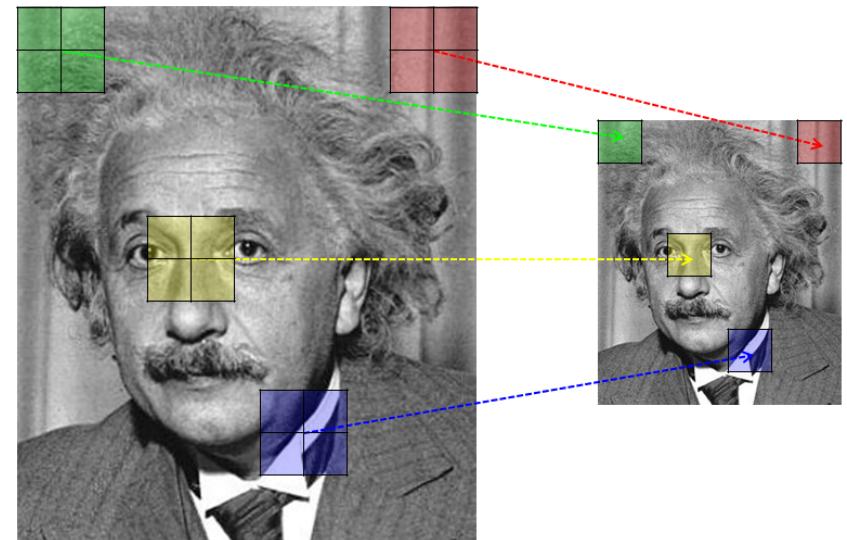
$$\begin{array}{c} \times \\ \quad \quad \quad = \end{array}$$

0	1	0
0	0	0
0	0	0



Pooling (Subsample)

local averaging operator



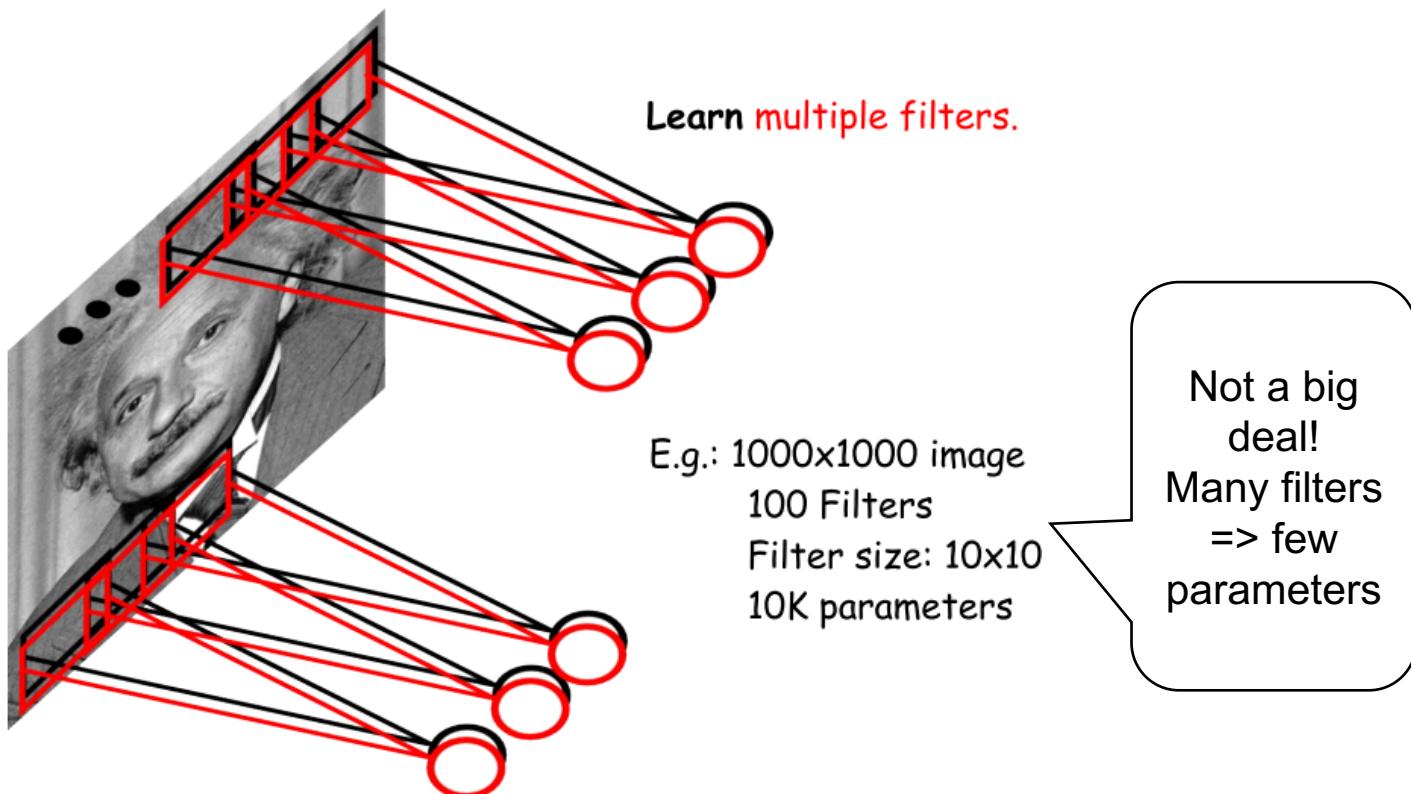
$$y_{ij} = \frac{1}{4} (x_{2i,2j} + x_{2i+1,2j} + x_{2i,2j+1} + x_{2i+1,2j+1})$$

From one to many filters

1 filter => 1 feature map (corresponding to 1 visual pattern)

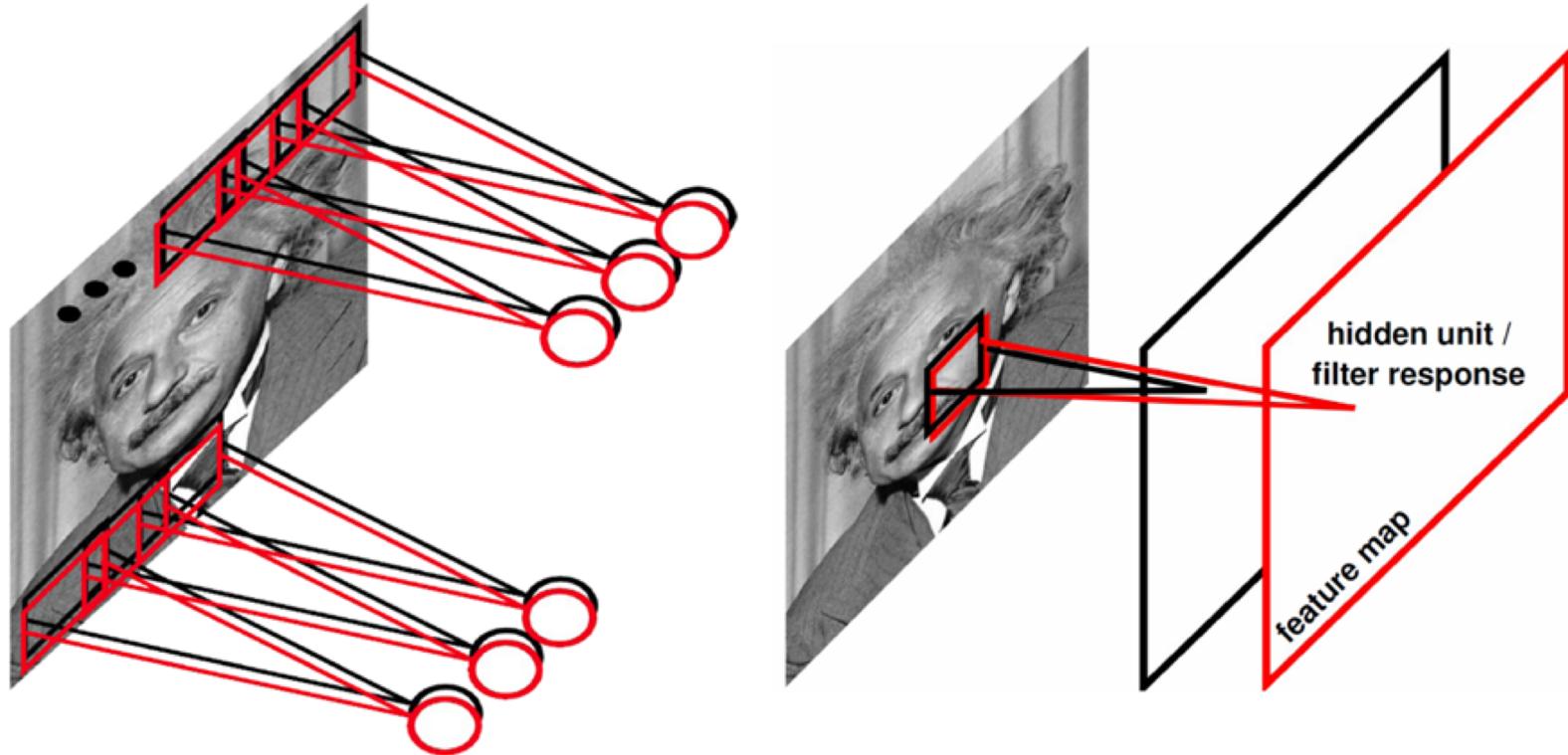
To detect spatial distributions of multiple visual patterns: Multiple filters

M filters => M feature maps! Get richer description

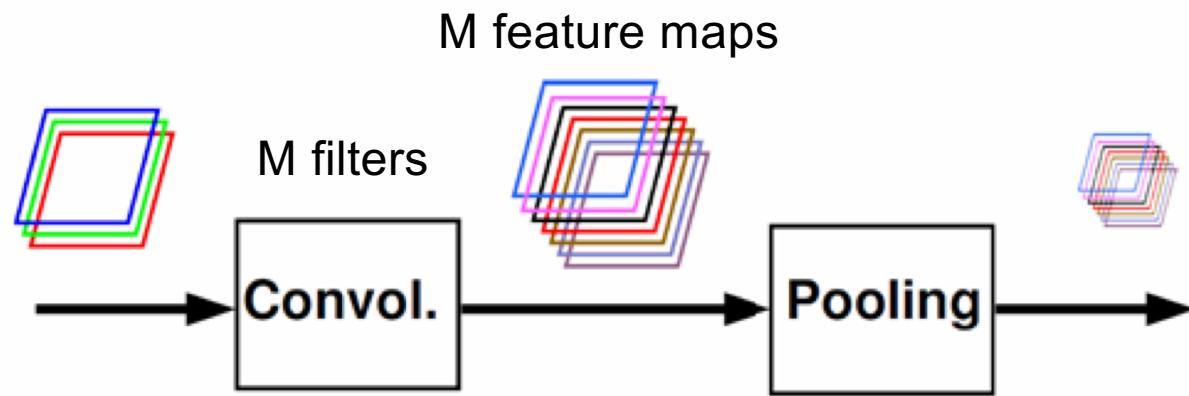


From one to many filters

M filters => M feature maps

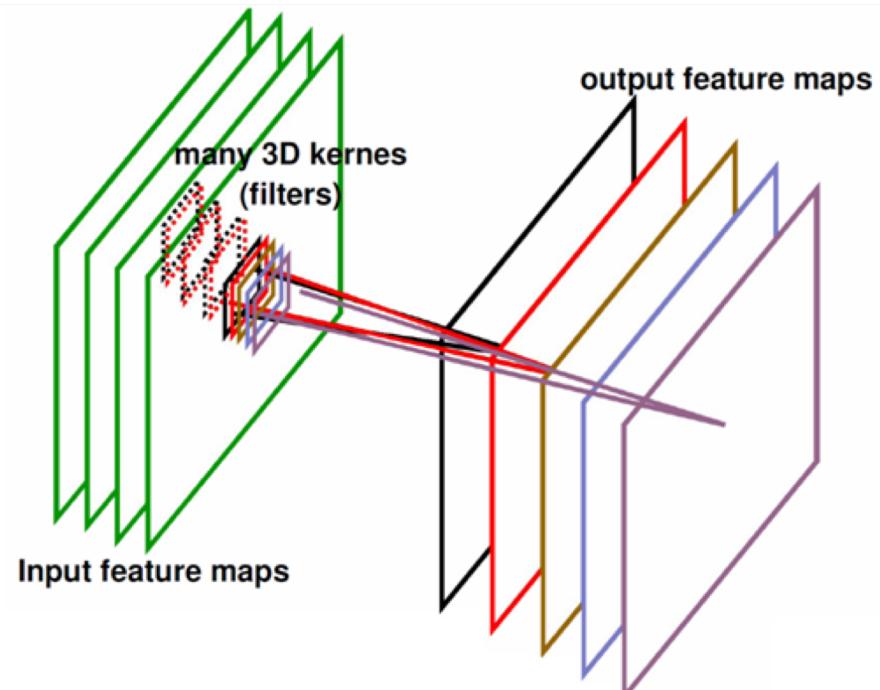
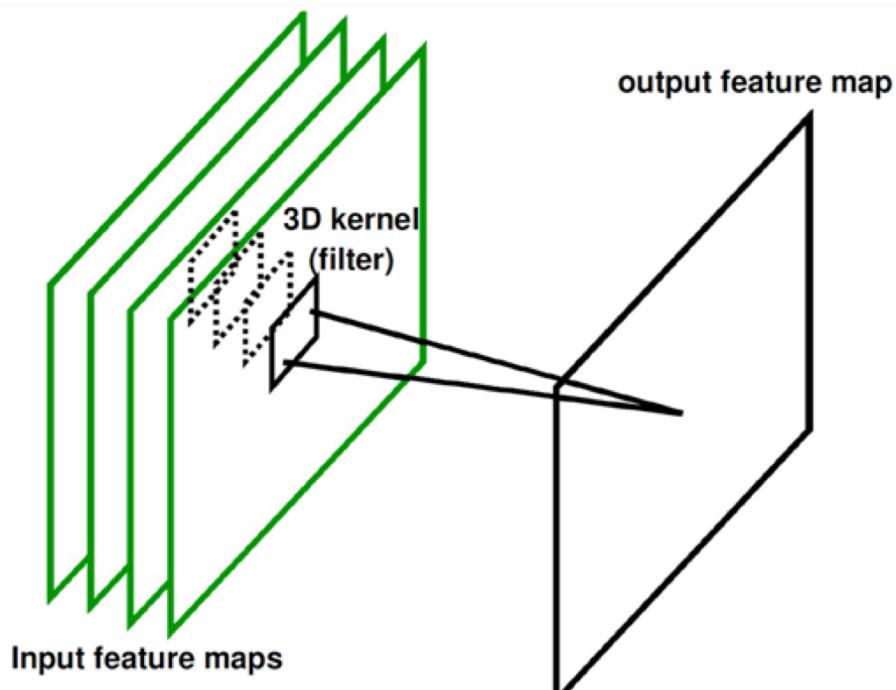


To sum up:



Color images: 3D kernels for filtering

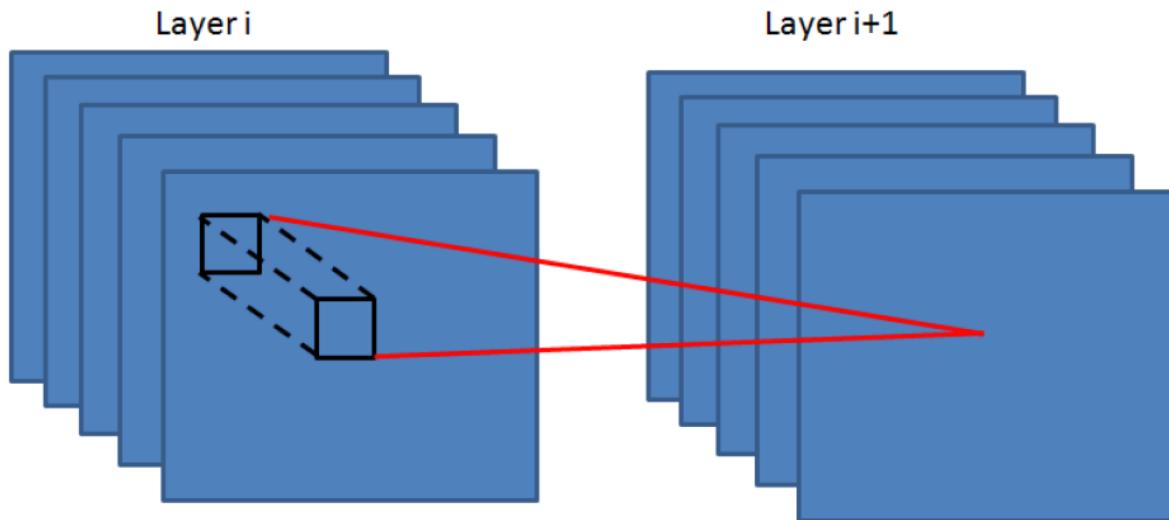
$m \times n \times d$ parameters per filter
Idem for any layer i to layer $i+1$



LCN: Local Contrast Normalization

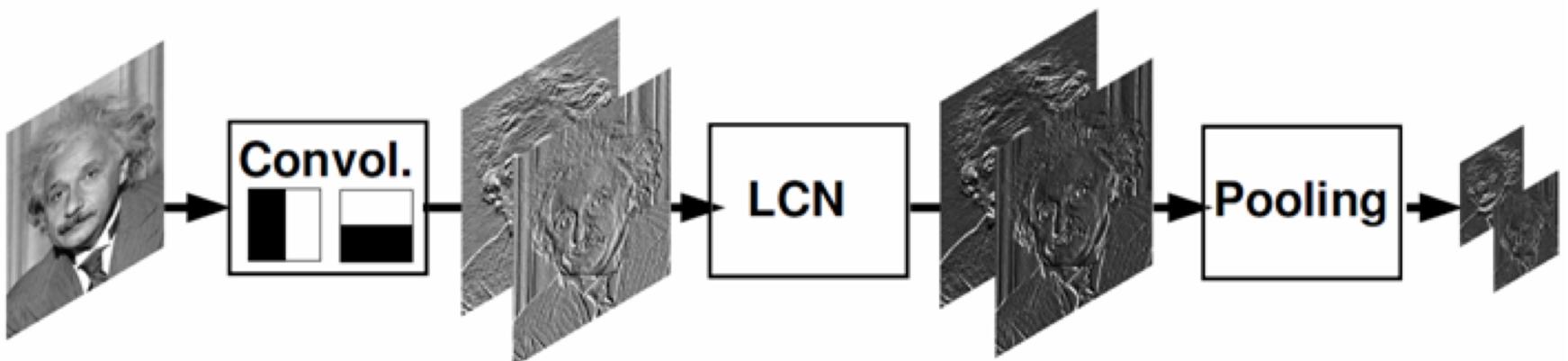
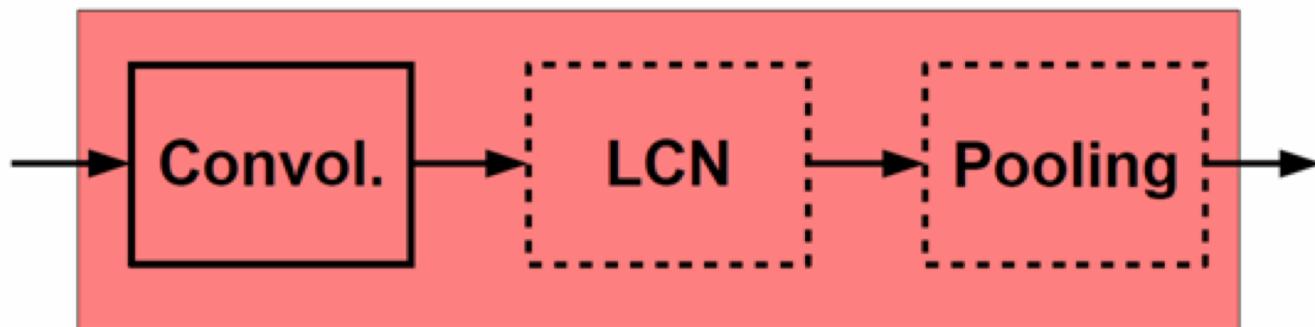
Normalization within a neighborhood along both spatial and feature dimensions

$$h_{i+1,x,y,k} = \frac{h_{i,x,y,k} - m_{i,N(x,y,k)}}{\sigma_{i,N(x,y,k)}}$$



=> Very important for training large nets to carefully consider normalization within mini-batches [S. Ioffe, C. Szegedy 2015]

1st stage of convolutional neural networks

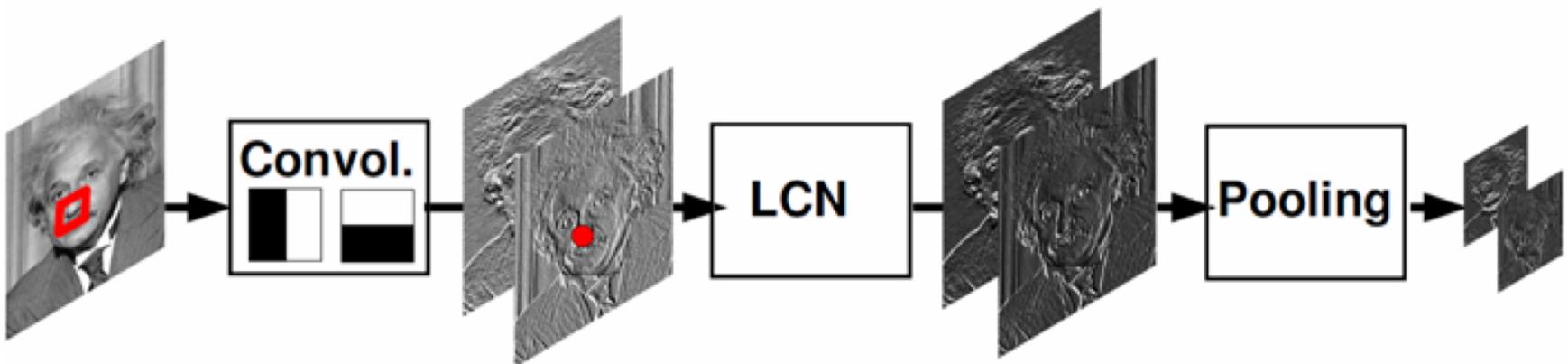
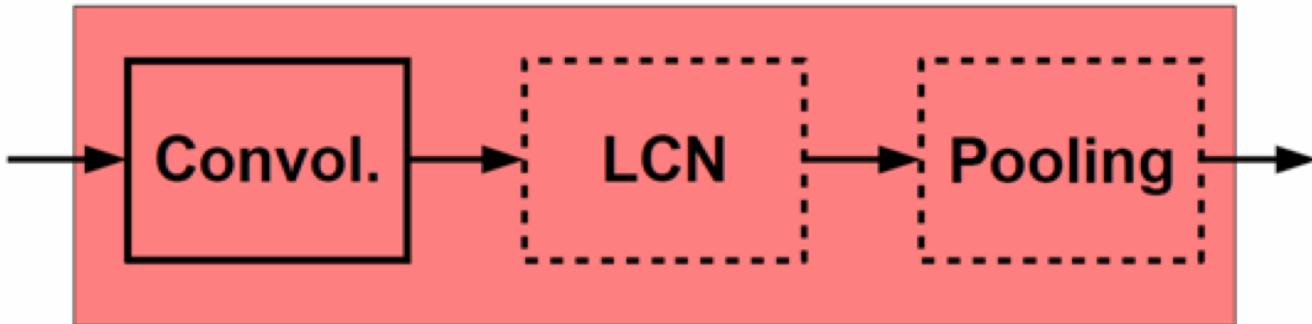


Example with only two filters.

Ranzato CVPR'13

1st stage of convolutional neural networks

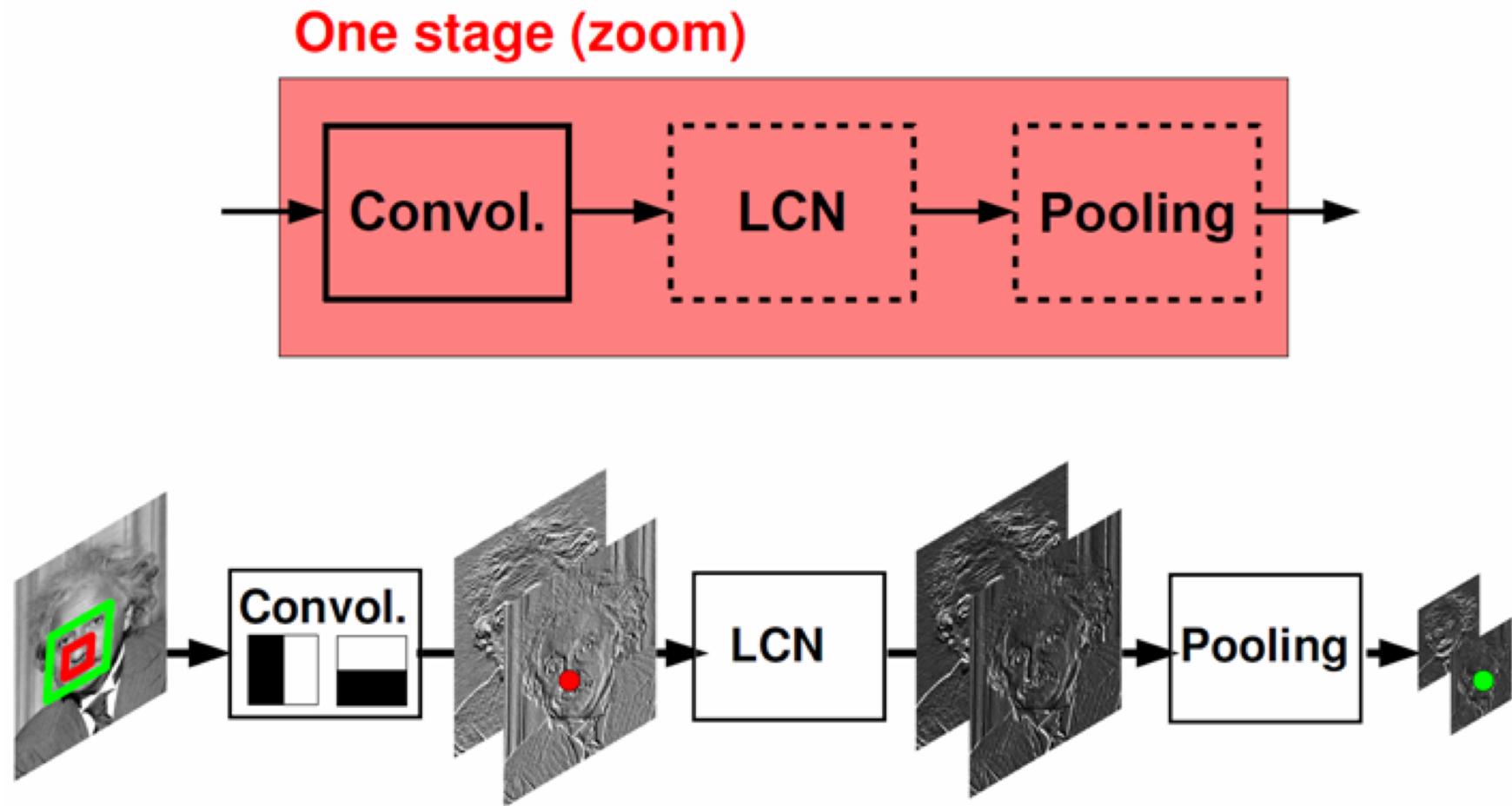
One stage (zoom)



A hidden unit in the first hidden layer is influenced by a small neighborhood (equal to size of filter).

Ranzato CVPR'13

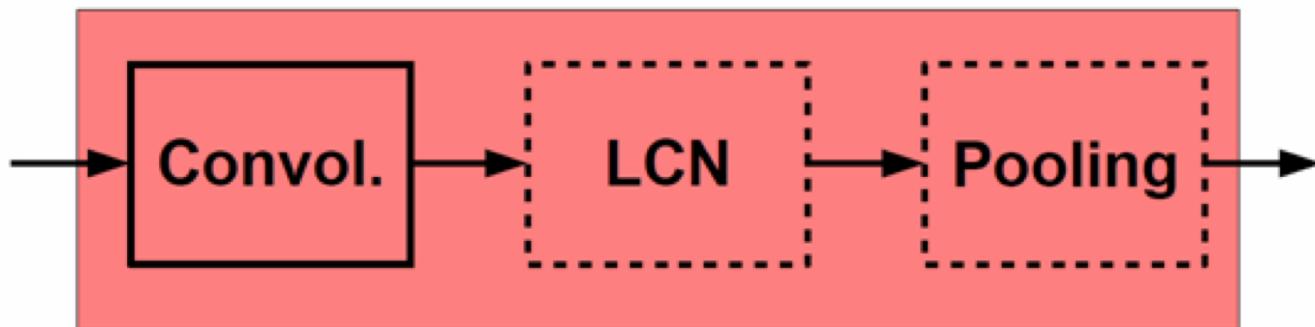
1st stage of convolutional neural networks



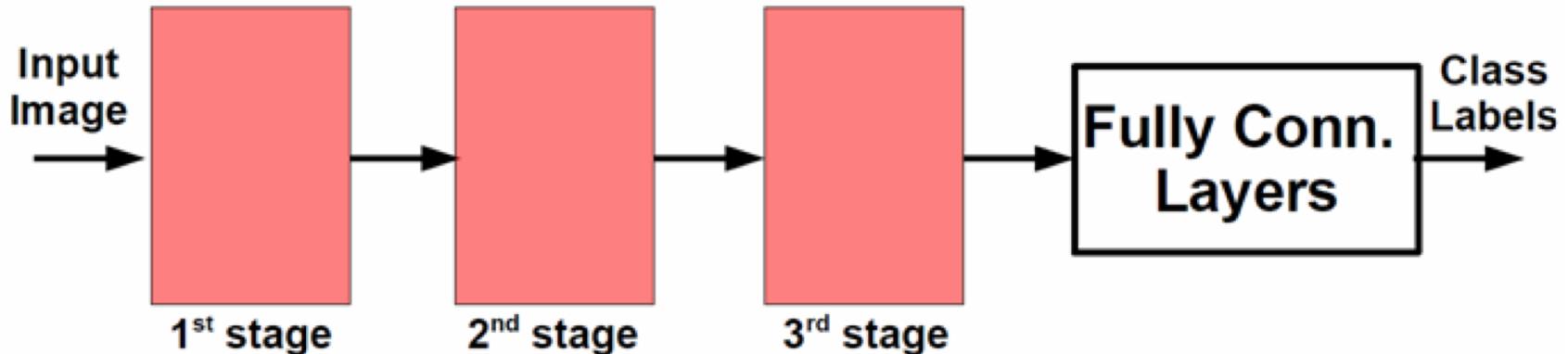
A hidden unit after the pooling layer is influenced by a larger neighborhood
(it depends on filter sizes and the sizes of pooling regions)

Full ConvNet architecture

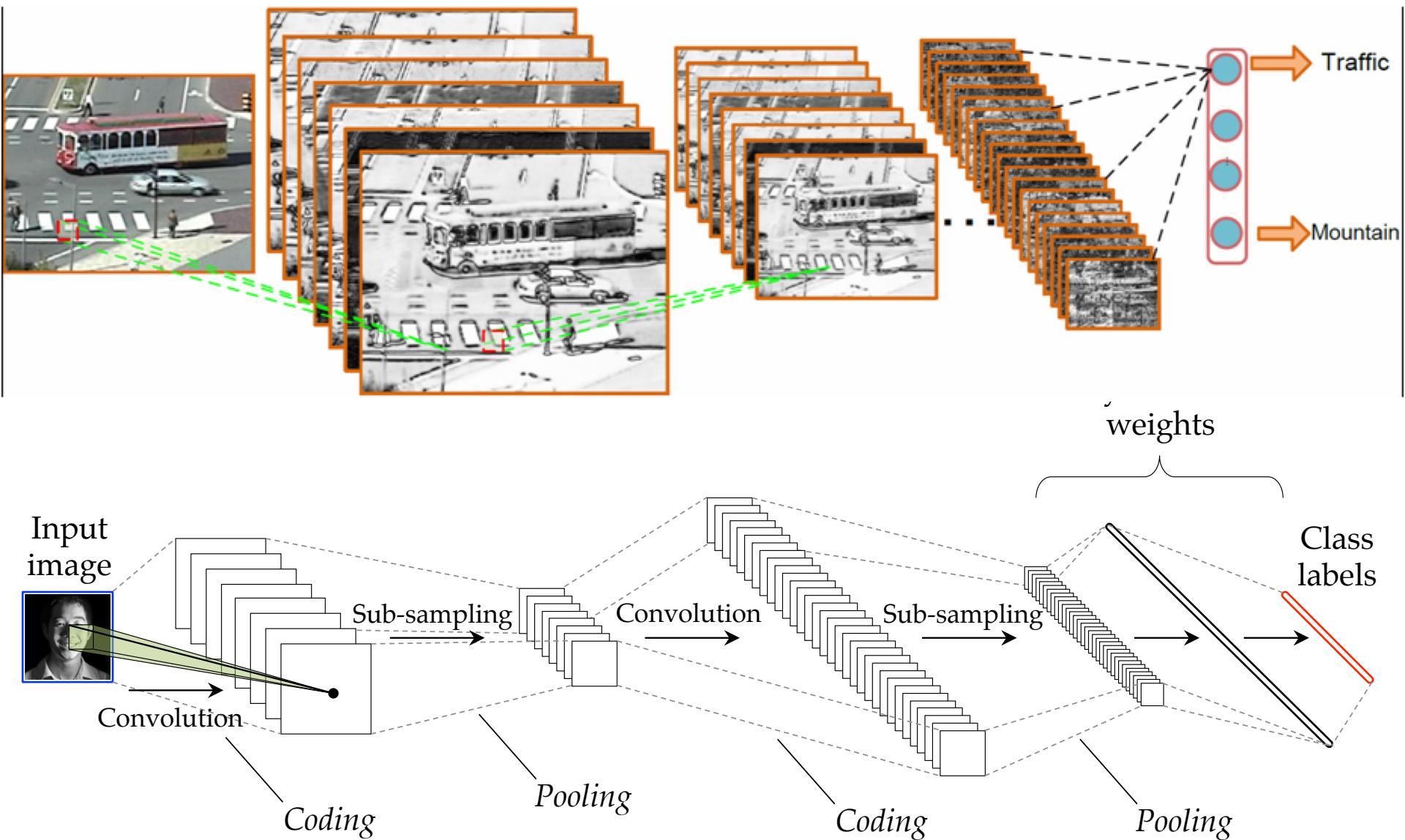
One stage (zoom)



Whole system



To sum up: Full ConvNet architecture

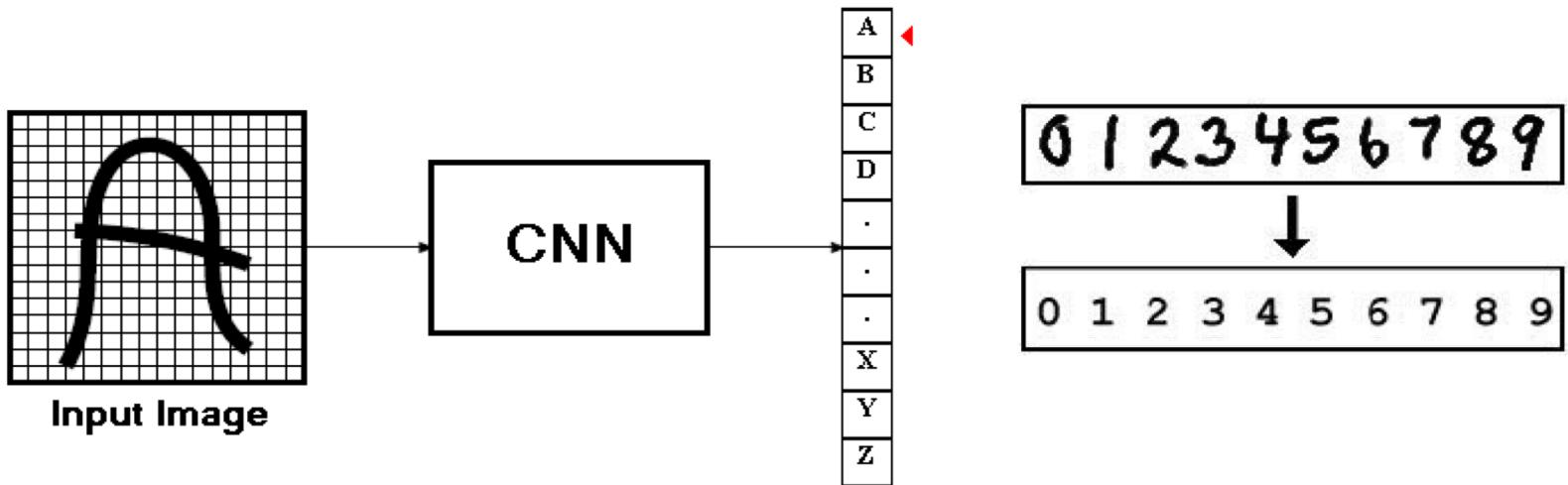


To sum up: Full ConvNet architecture

CNN: feed-forward network with
ability to extract topological properties from image
designed to recognize visual patterns

Working directly from pixel images with (no/minimal)
preprocessing

Trained with back-propagation



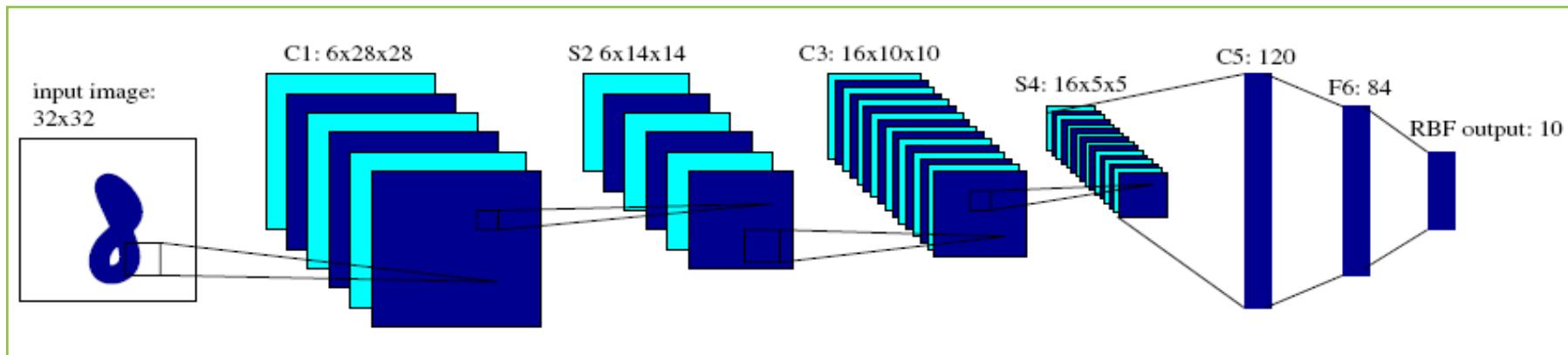
Outline

1. Recap MLP
2. Convolutional Neural Networks
 1. Basics for deep in images
 2. **One example: LeNet5**

Example: LeNet5

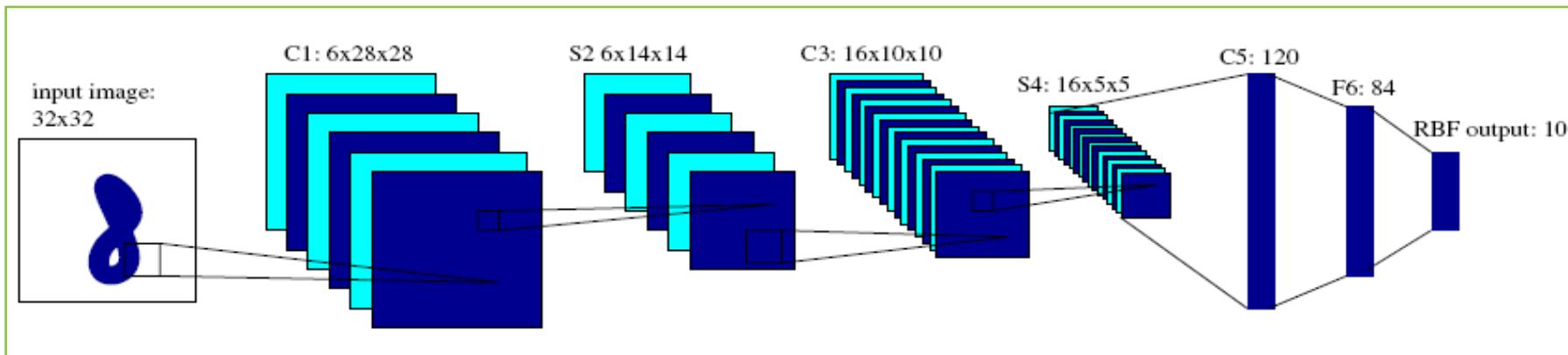
Introduced by Y. LeCun

Raw image of 32×32 pixels as input



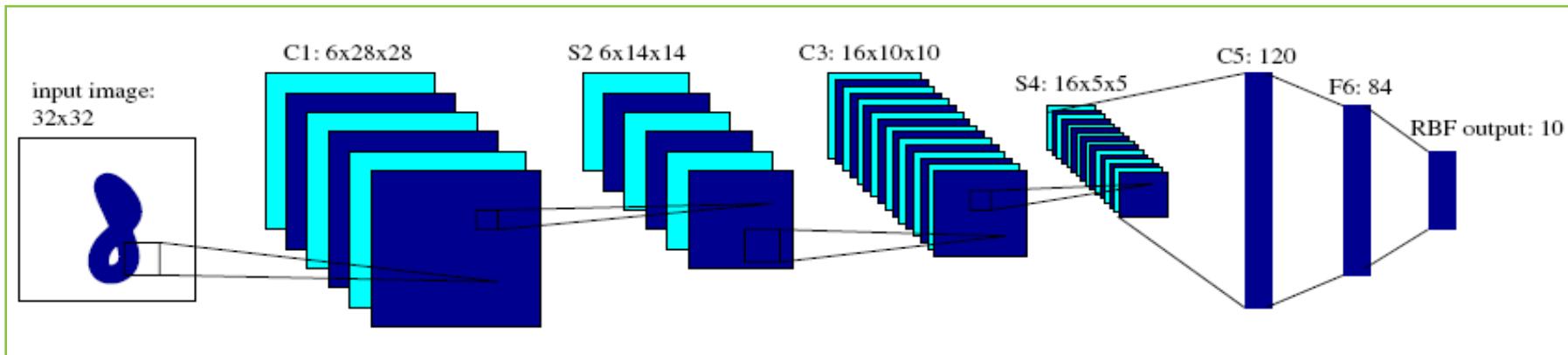
Example: LeNet5

- C1,C3,C5 : Convolutional layer
- 5×5 Convolution matrix
- S2 , S4 : Subsampling layer = Pooling+stride s=2
=> Subsampling by factor 2
- F6 : Fully connected layer

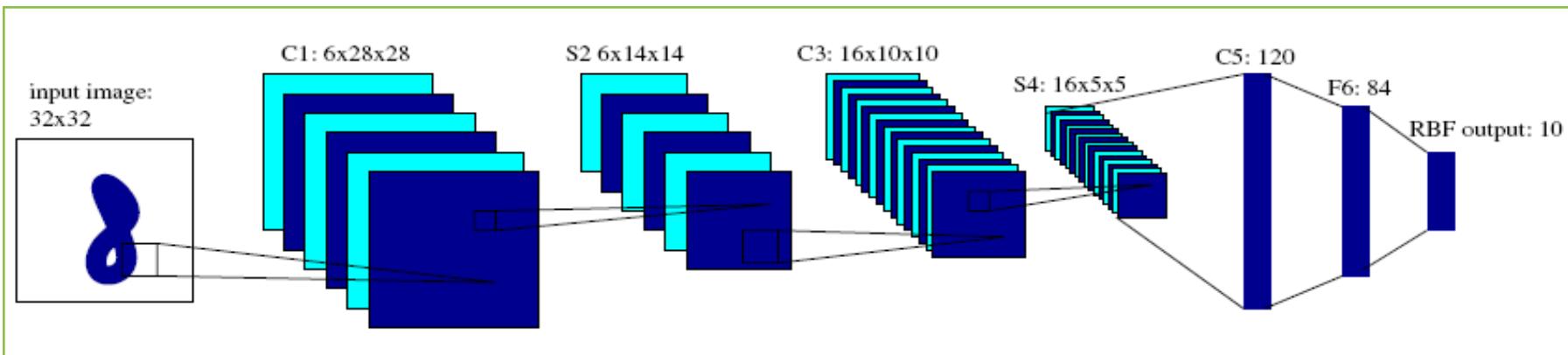


LeNet5

All the units of the layers up to F6 have a sigmoidal activation function

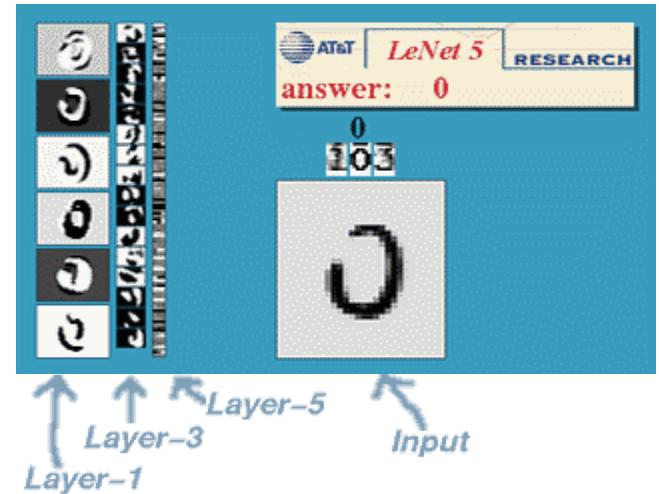


LeNet5

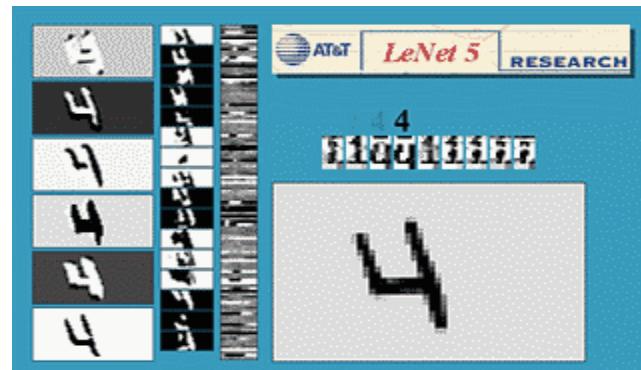
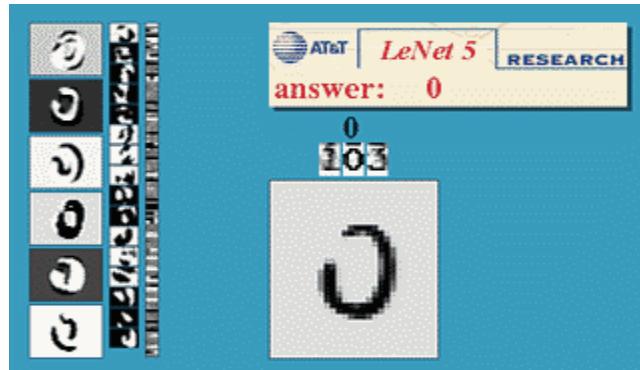


About 187,000 connections

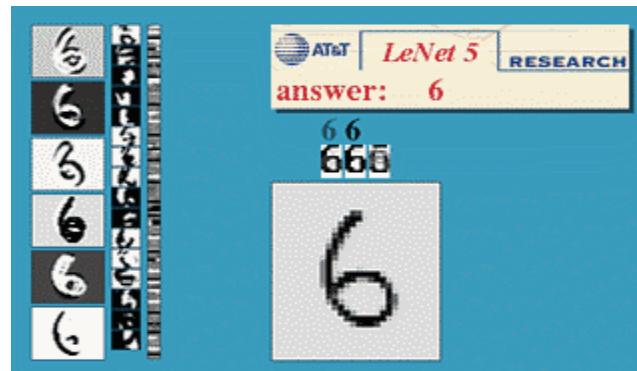
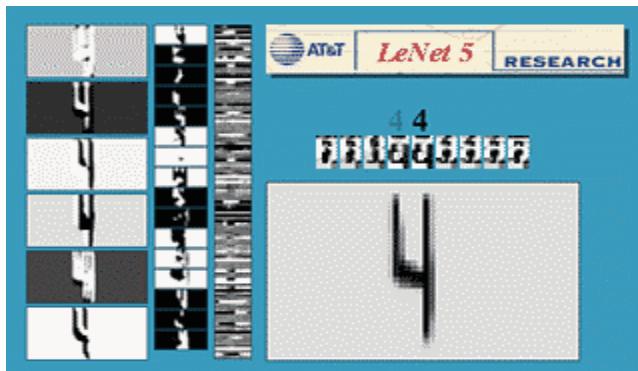
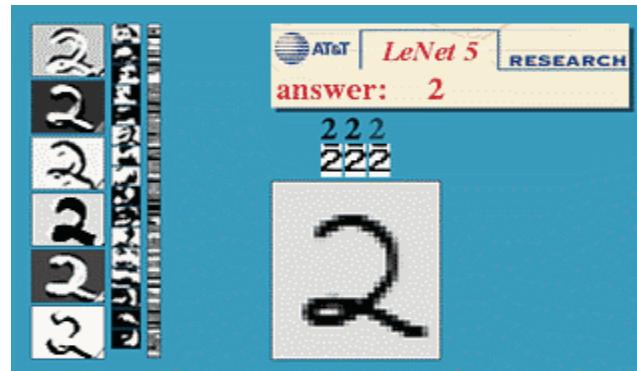
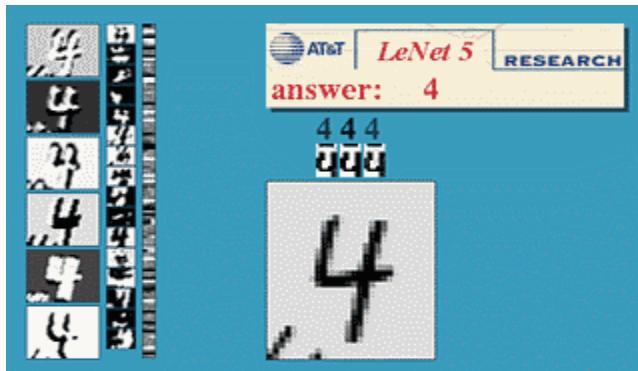
About 14,000 trainable weights



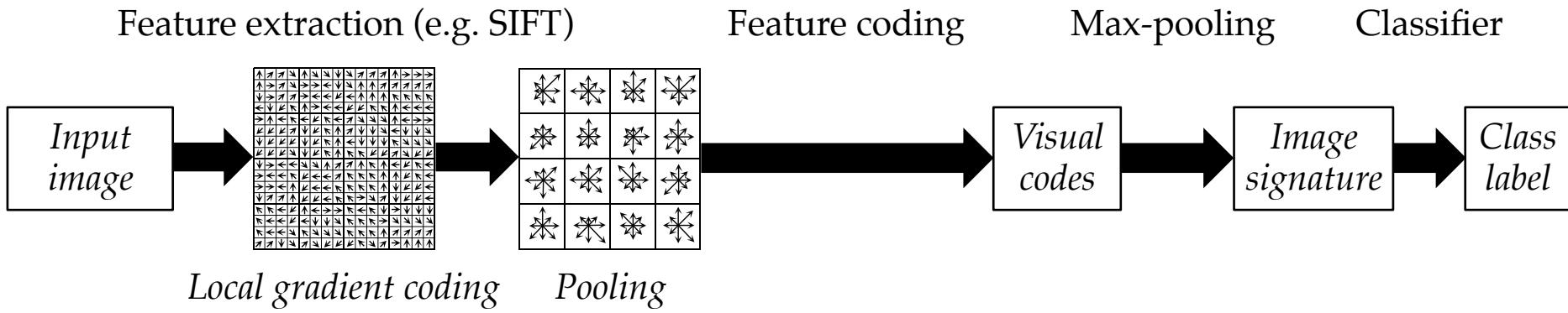
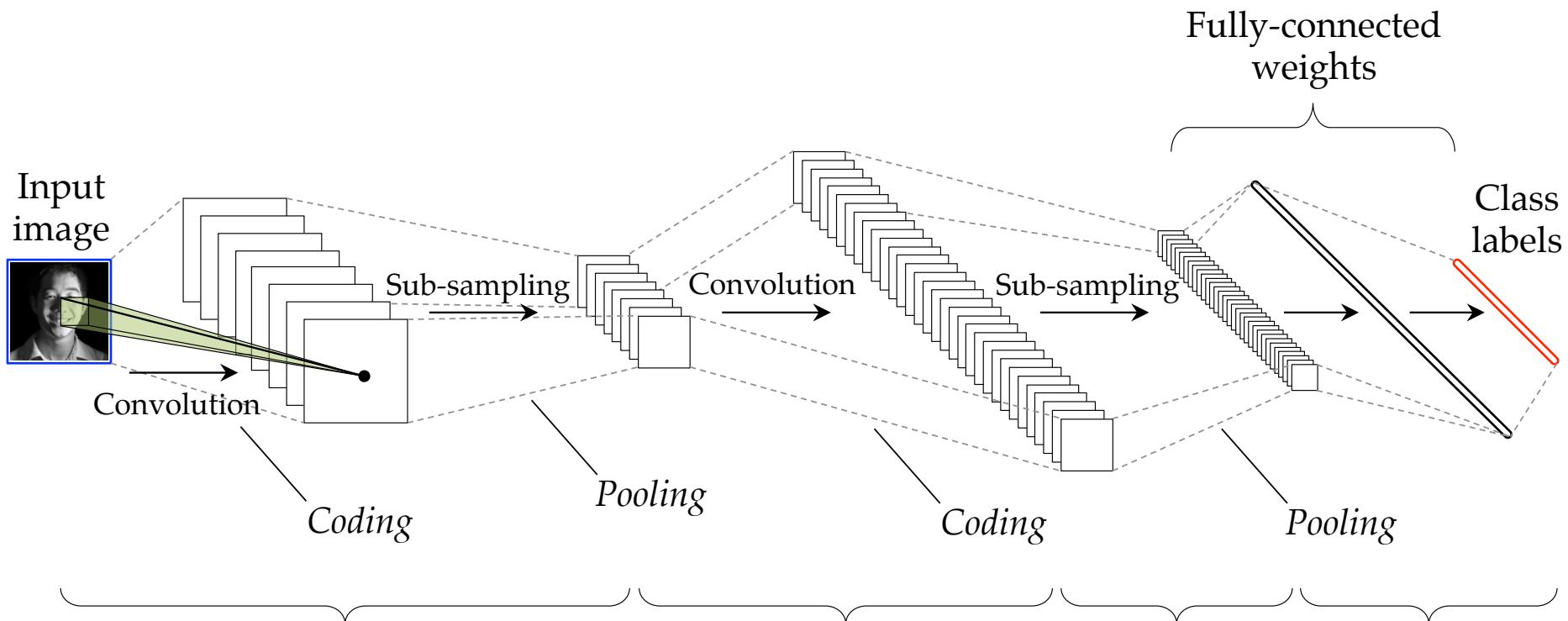
LeNet5 (@LeCun)



LeNet5 (@LeCun)



Comparison BoW / deep CNN



Outline

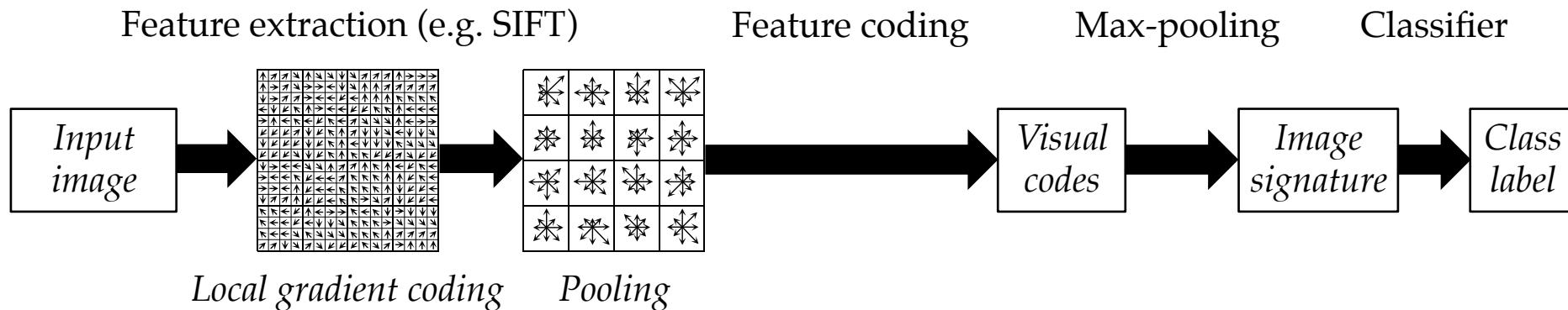
1. Recap MLP
2. Convolutional Neural Networks
 1. Basics for deep in images
 2. One example: LeNet5
 3. **Compa deep vs BoW**

Comparison BoW / CNN deep

1. Regular grid + gradient detection (SIFT) => bank of 8 linear filters (convolution) + Winner Take All (inter-maps) => 8 maps
2. Local histogram SIFT => spatial local sum pooling inside maps on a fixed grid 4x4 => $16 \times 8 = 128$ (smaller) maps
3. BoW Coding = projection on M vectors (visual dico elts) => a bank of M linear filters of size 4x4x8 (=1x1x128 convolution) => M maps
4. BoW Pooling => global pooling on each map => M scalar values = 1 vector representation BoW (extension: SPM)
5. Classification (SVM) => Fully connected layers

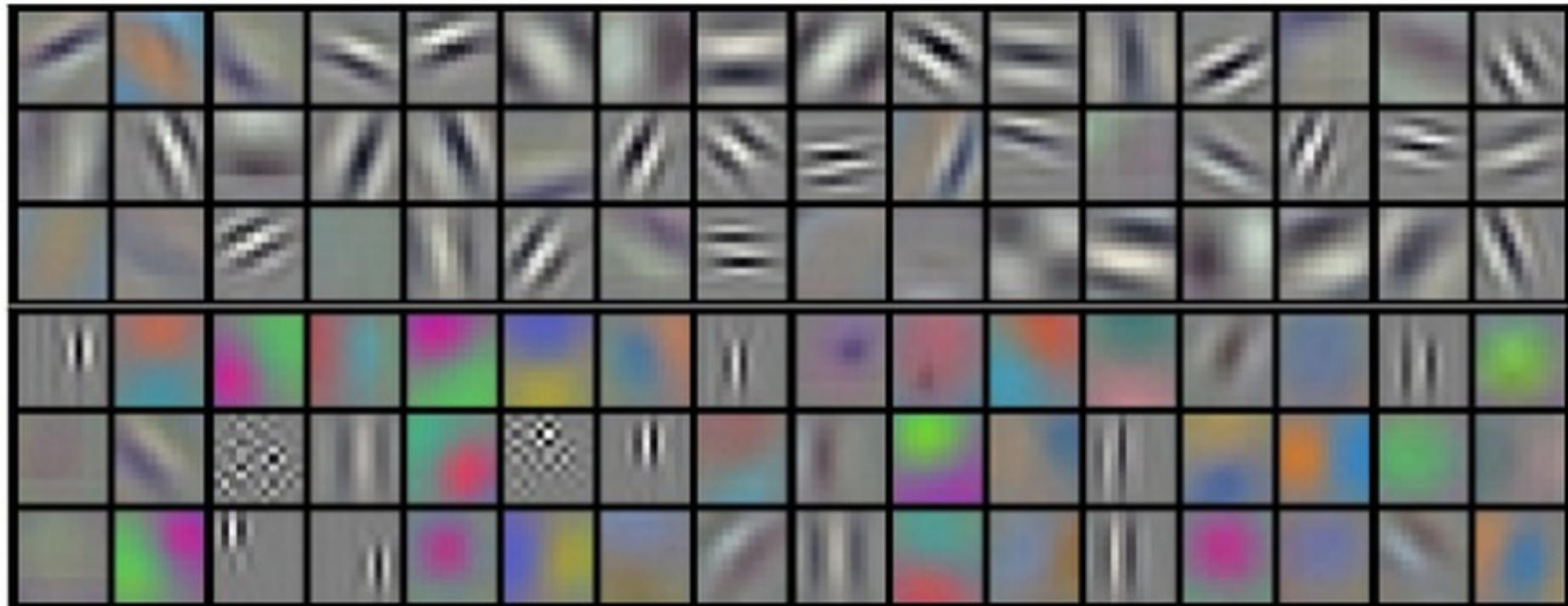
BoW = Conv1+pooling(loc)+Conv2+pooling(global)+Fconnected

- Handcrafted+unsupervised vs. end-to-end supervision
- Light deep vs. very deep

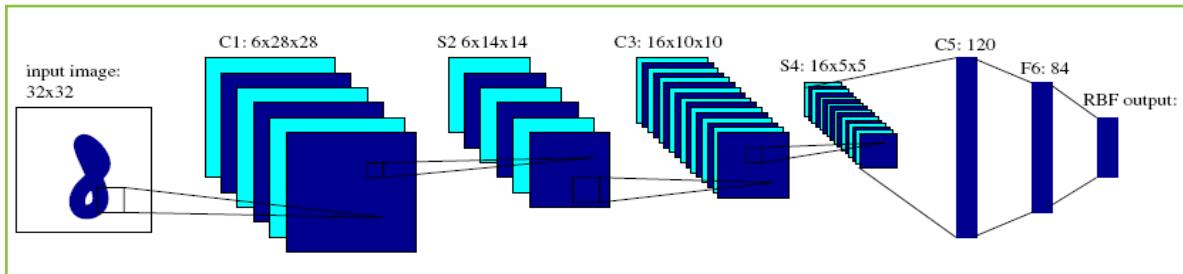


Deep vs shallow in Computer Vision

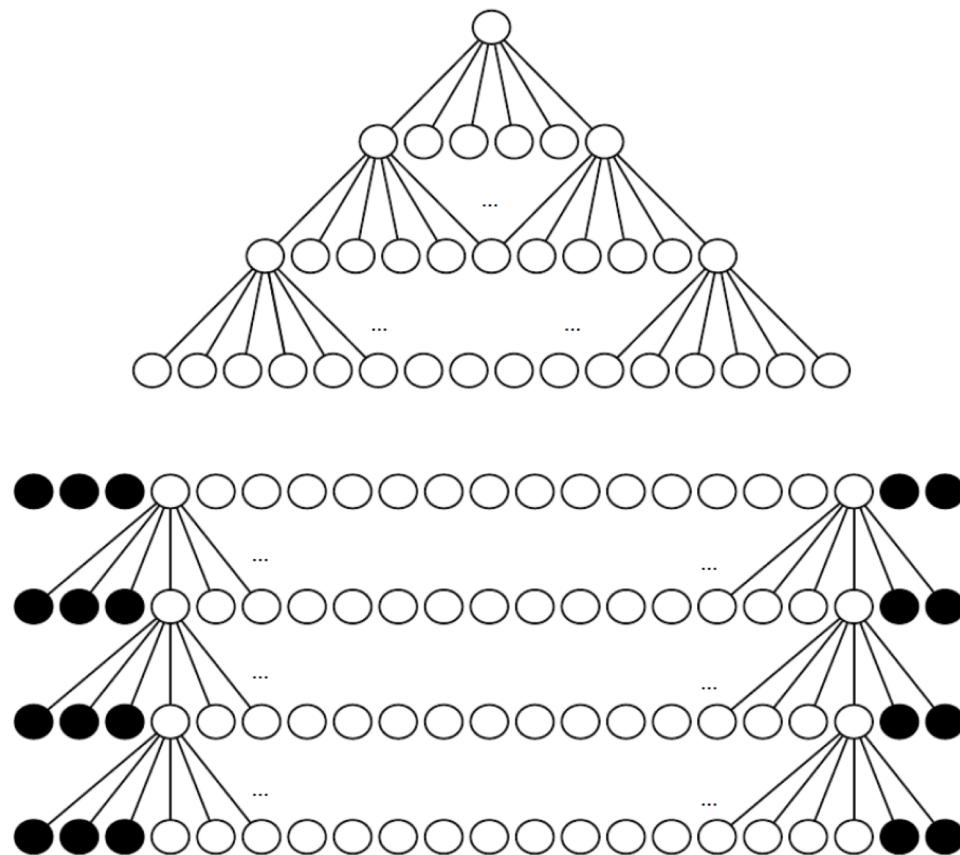
- CV work(ed) a lot on handcrafted local features
 - BoVW (Bag of Visual Words and extensions FisherVectors, BossaNova ...)
 - BoVW not so shallow but not end-to-end supervised learning
- CNN: end-to-end learning on a **handcrafted** architecture! [Chatfield BMVC 2014]
 - Why 8 layers? why 3x3 at the 5th layer without polling? ... => ad-hoc architecture



Zero-padding in convolutional neural network (optional)



To avoid shrinking the spatial extent of the network rapidly



More in getting local Invariance

Invariance to local translation (small shift) OK with pooling

Is convolution equivariant/invariant to changes in scale or rotation?

No such invariance with linear filters

Possible extension:

Pooling OVER outputs of separately parameterized convolutions

Become possible to LEARN invariance to rotation (or other)

Example (Bengio et al. Deep Learning 2014):

By learning to have each filter be a different rotation of the “5” template +
pooling over outputs => invariance to rotation of the “5”

5	5	5
5	5	5
5	5	5

“This is in contrast to translation invariance, which is usually achieved by hard-coding the net to pool over shifted versions of a single learned filter”