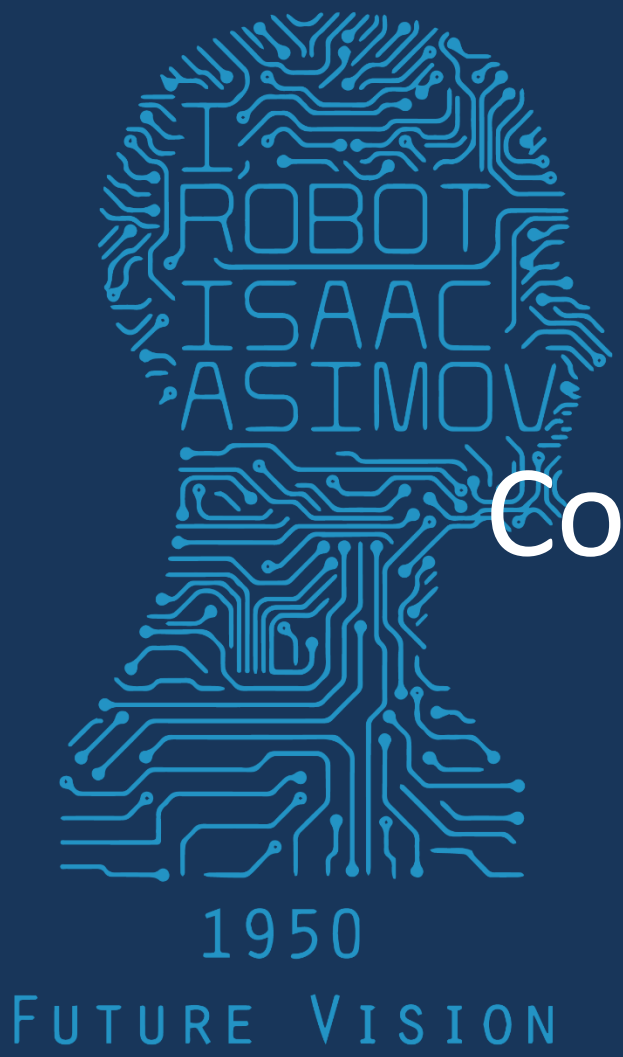
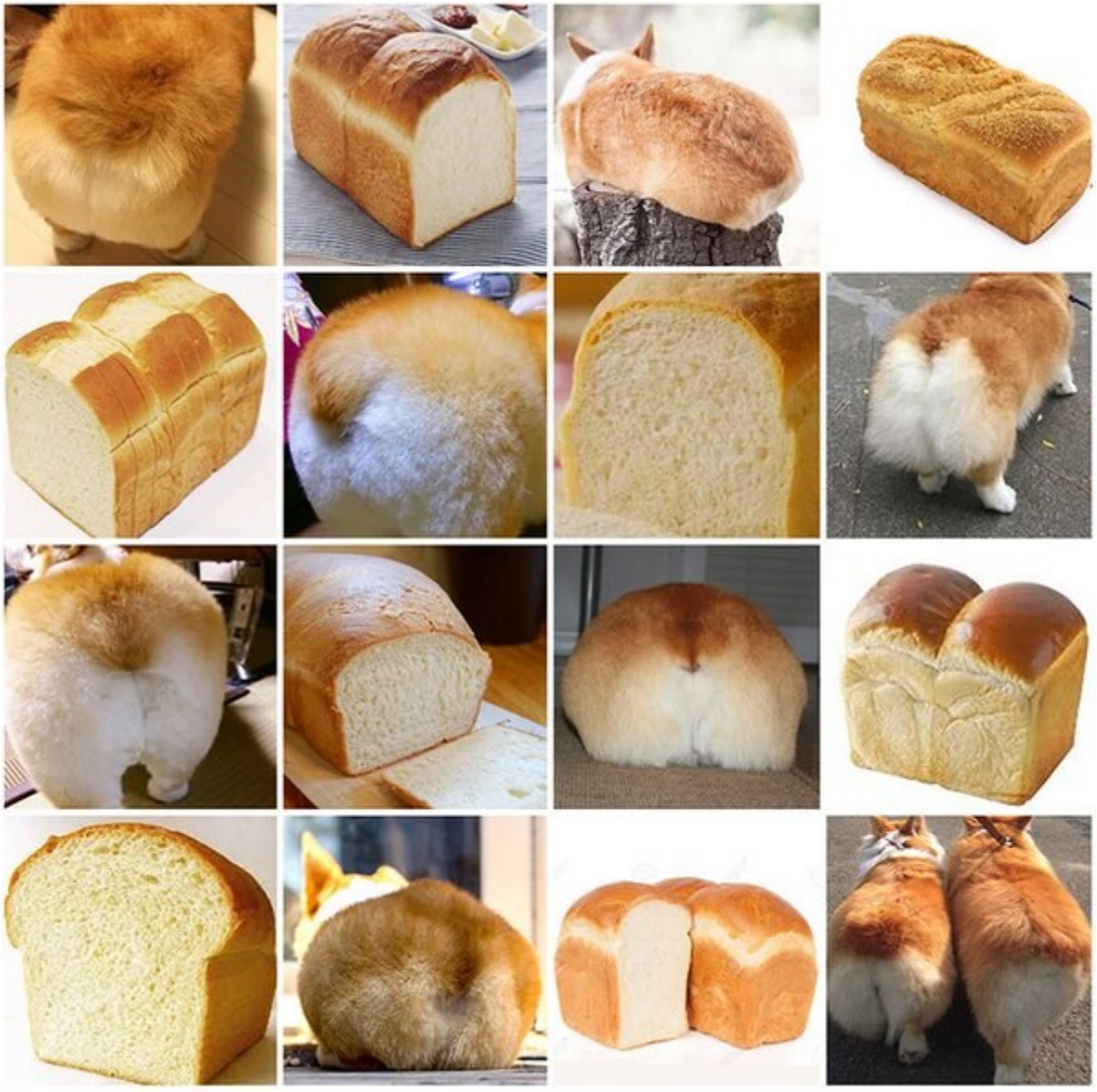
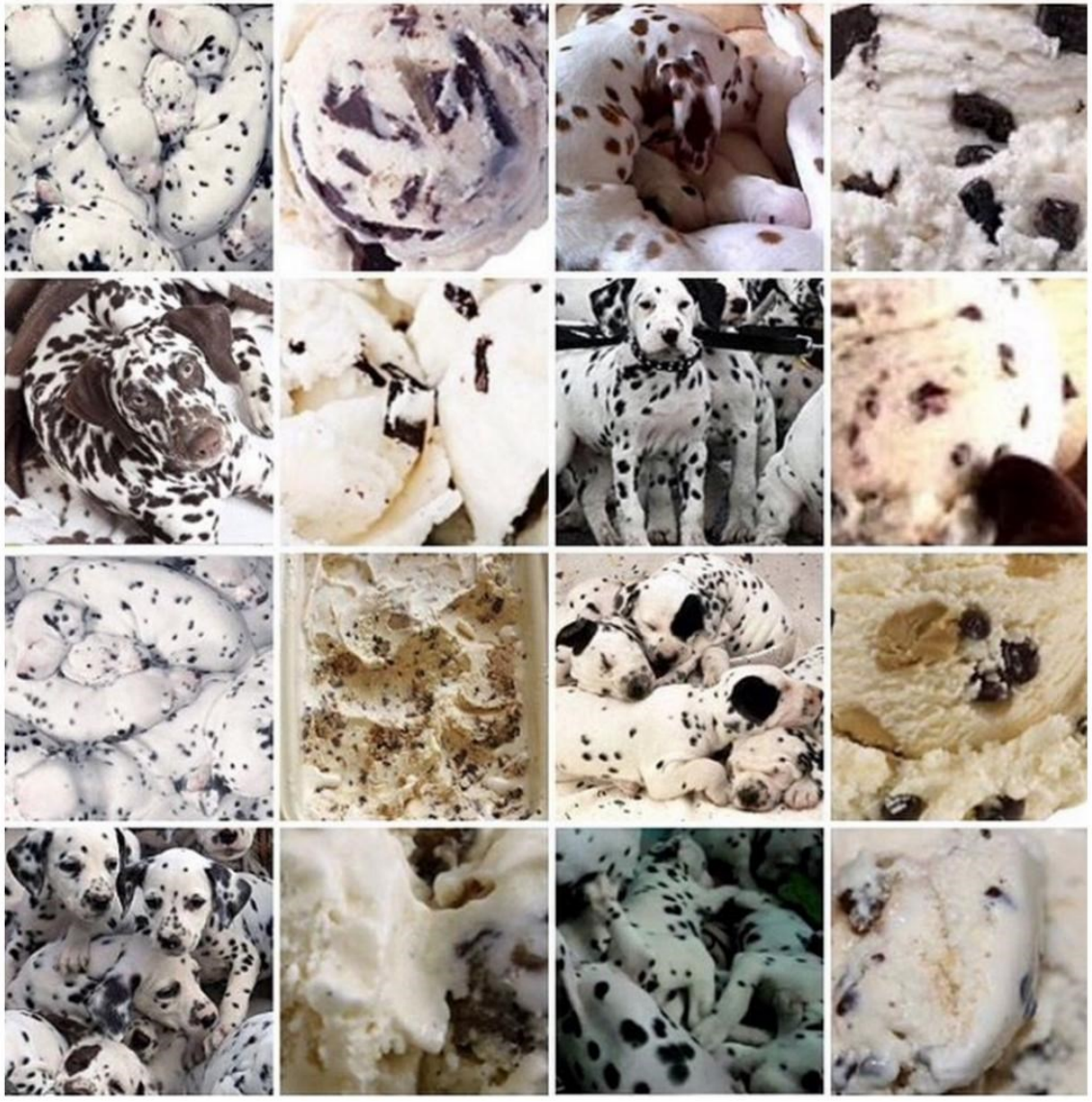
Advanced Computer Vision

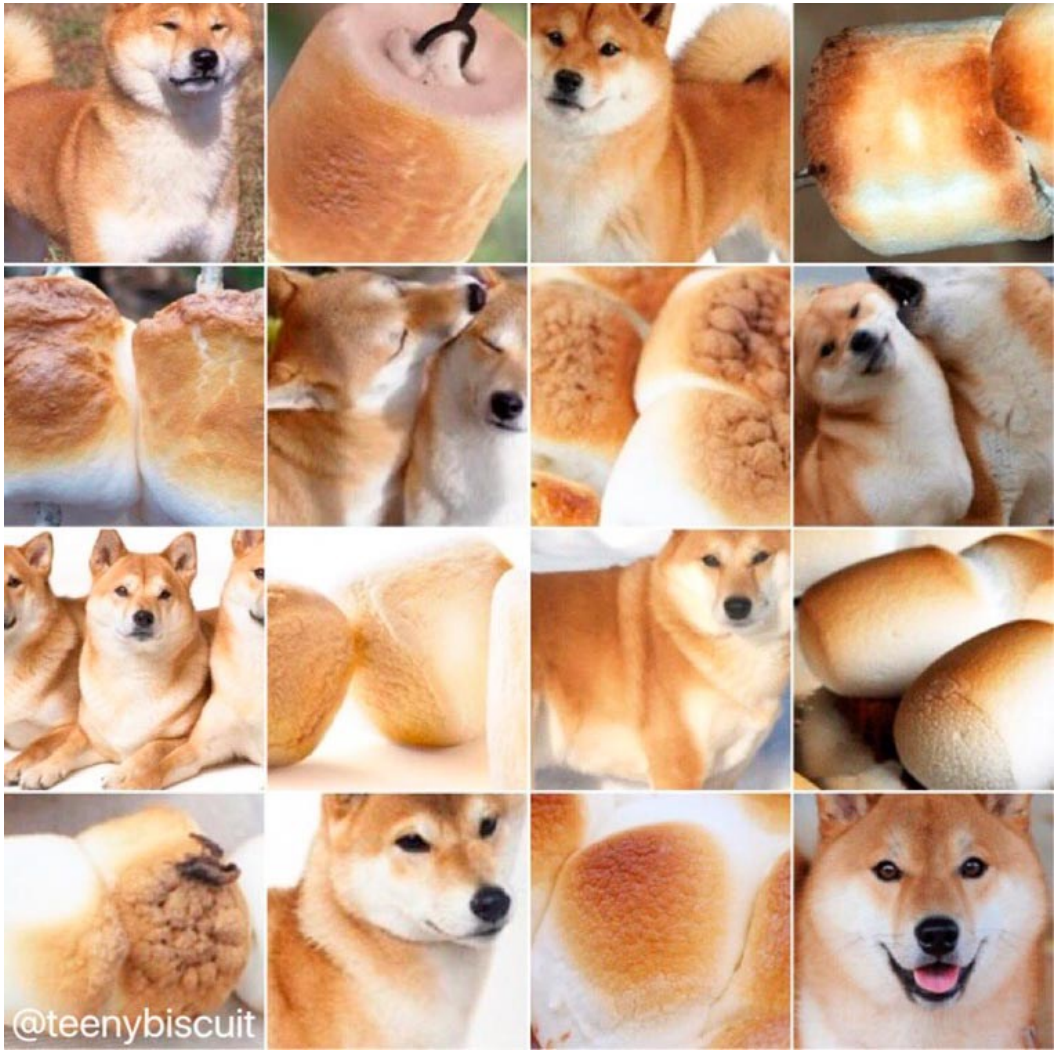


Convolutional Neural Networks





Wow



false positives

what class

no good filtr

so misclassified

cool kernel

Goals

Build a classifier which is more powerful at representing complex functions *and* more suited to the learning problem.

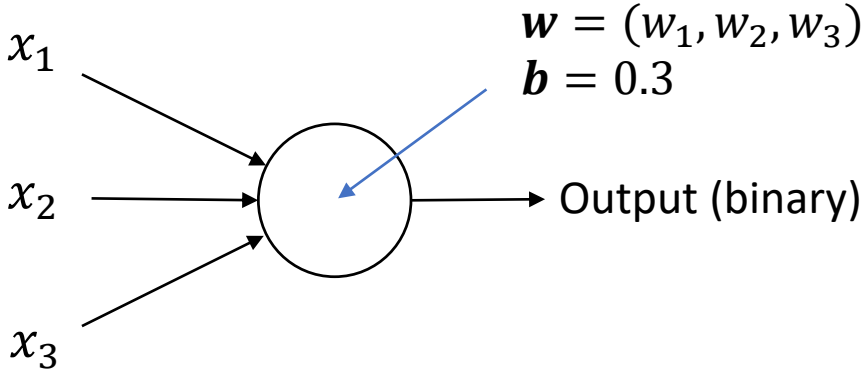
What does this mean?

1. Assume that the *underlying data generating function* relies on a composition of factors.

2. Learn a feature representation that is specific to the dataset.

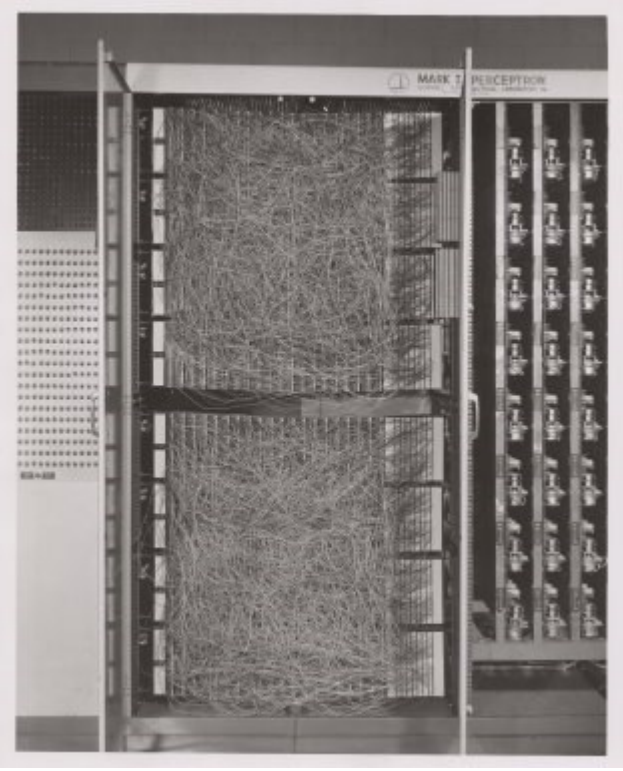
Neural Networks

- Basic building block for composition is a *perceptron* (Rosenblatt c.1960)
- Linear classifier – vector of weights *w* and a 'bias' *b*



$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases}$$

$$w \cdot x \equiv \sum_j w_j x_j$$



Mark 1 Perceptron
c.1960

20x20 pixel
camera feed

Wikipedia

Universality

A single-layer network can learn any function:

- So long as it is differentiable
- To some approximation;
  More perceptrons = a better approximation

Visual proof (Michael Nielson):

http://neuralnetworksanddeeplearning.com/chap4.html

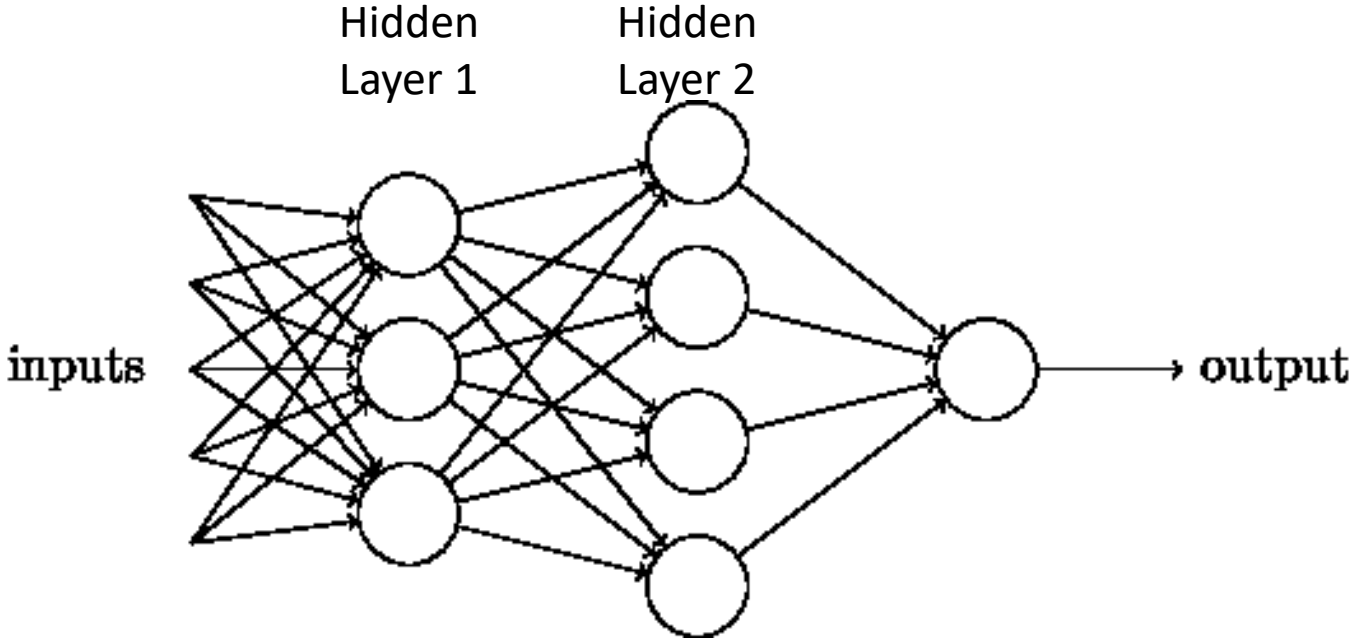
If a single-layer network can learn any function…
…given enough parameters…

…then why do we go deeper?

Intuitively, composition is efficient because it allows *reuse*.

Empirically, deep networks do a better job than shallow networks at learning such hierarchies of knowledge.

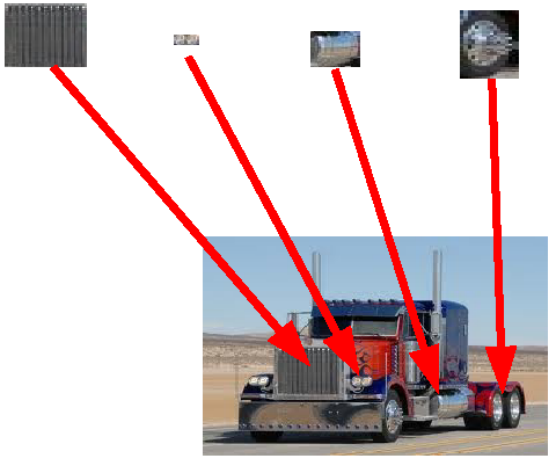
Composition



Layers that are in between the input and the output are called *hidden layers*, because we are going to *learn* their weights via an optimization process.

Interpretation of many layers

[0 0 **1** 0 0 0 0 **1** 0 0 **1** **1** 0 0 **1** 0 …] truck feature



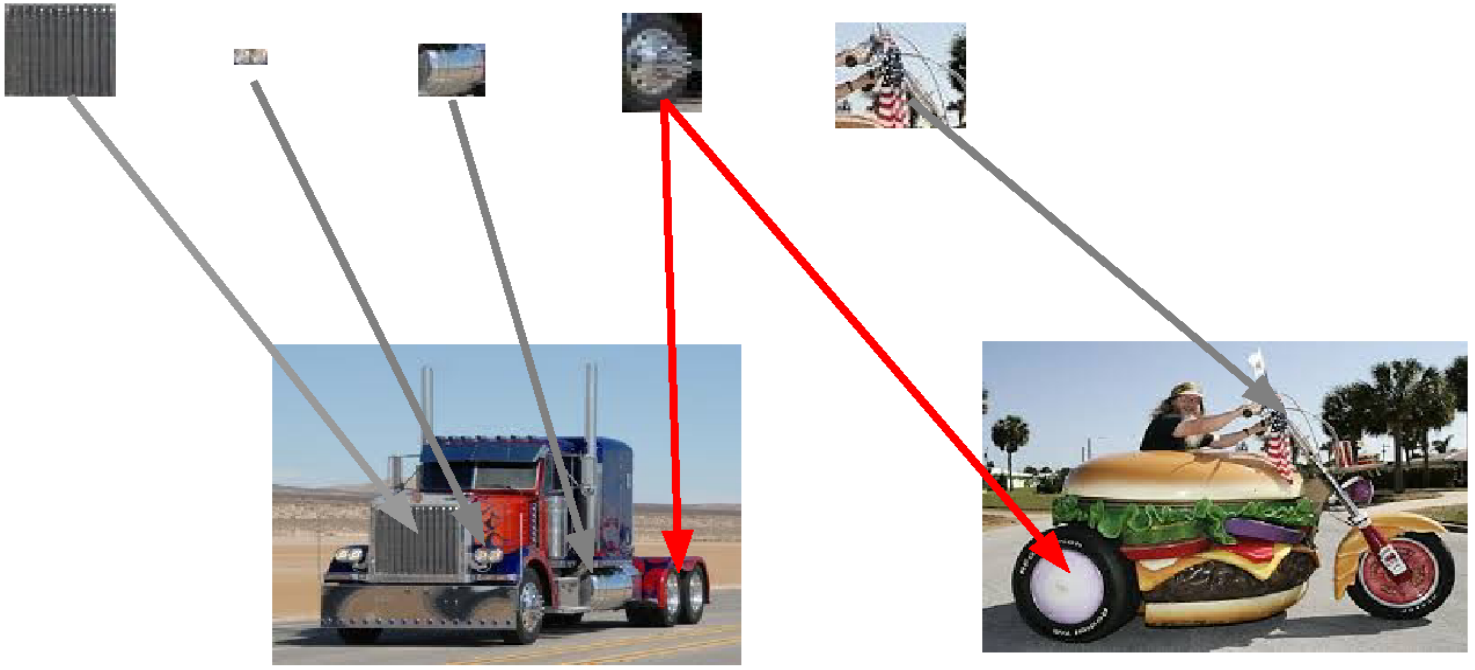
Exponentially more efficient than a 1-of-N representation (a la k-means)

Ranzato

Interpretation

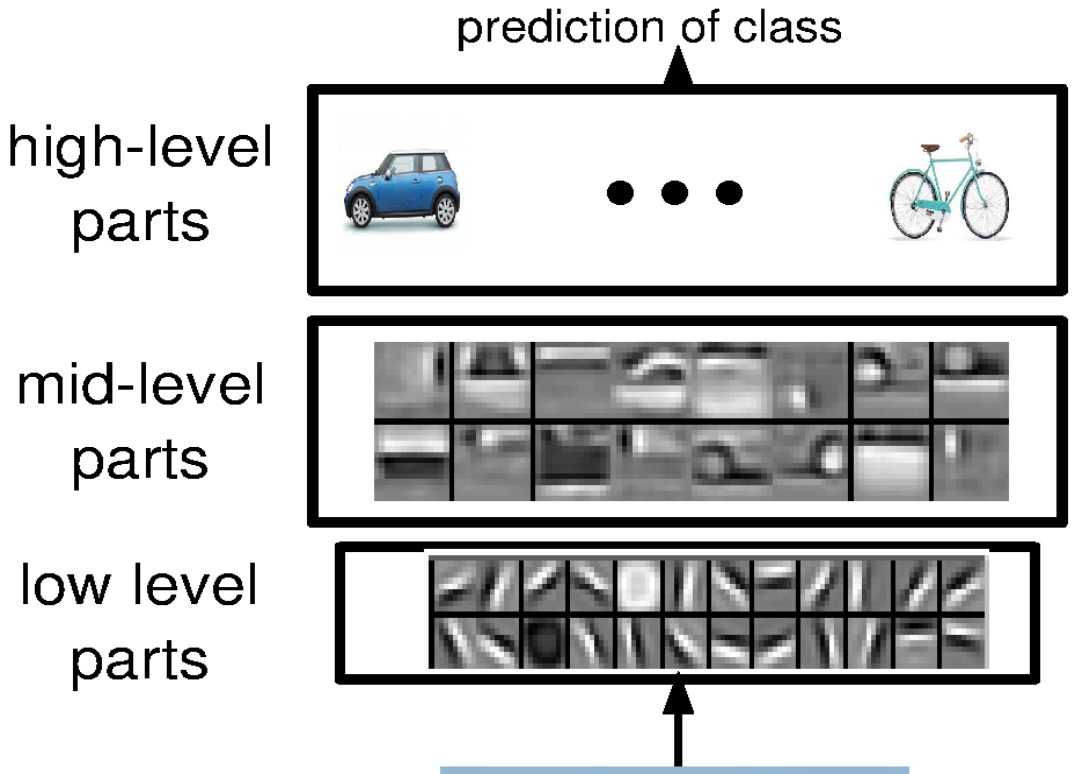
[1 1 0 0 0 1 0 **1** 0 0 0 0 1 1 0 1 . . .] motorbike

[0 0 1 0 0 0 0 **1** 0 0 1 1 0 0 1 0 . . .] truck



Ranzato

Interpretation



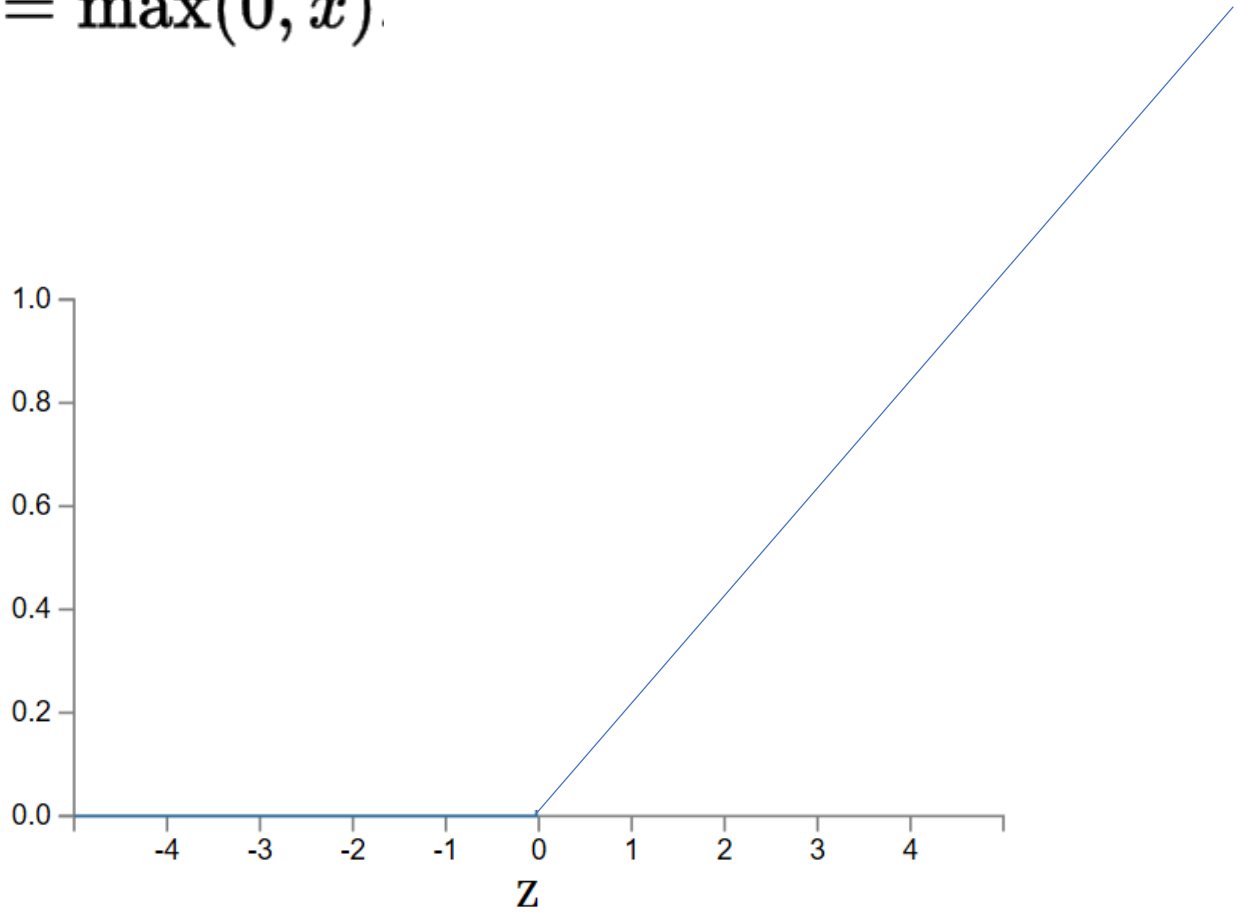
- distributed representations
- feature sharing
- compositionality

NOTE: Not actually the weights; a demonstrative visualization!

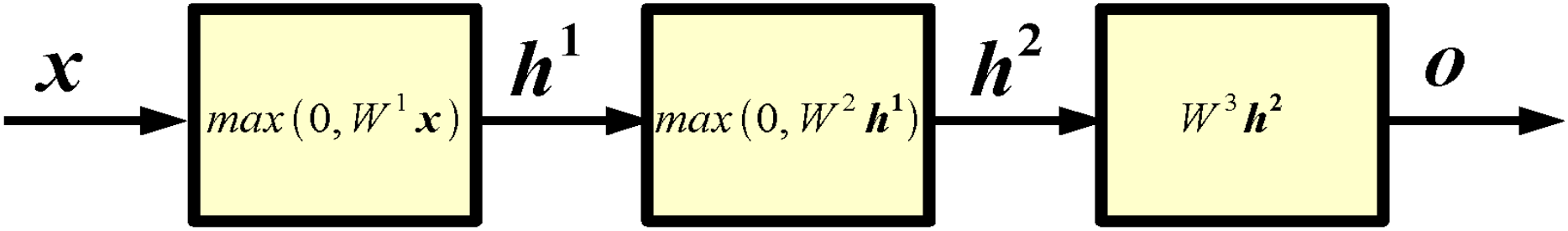
Lee et al. "Convolutional DBN's ..." ICML 2009

Activation functions: Rectified Linear Unit

- ReLU $f(x) = \max(0, x)$



Neural Networks: example



x input

h^1 1-st layer hidden units

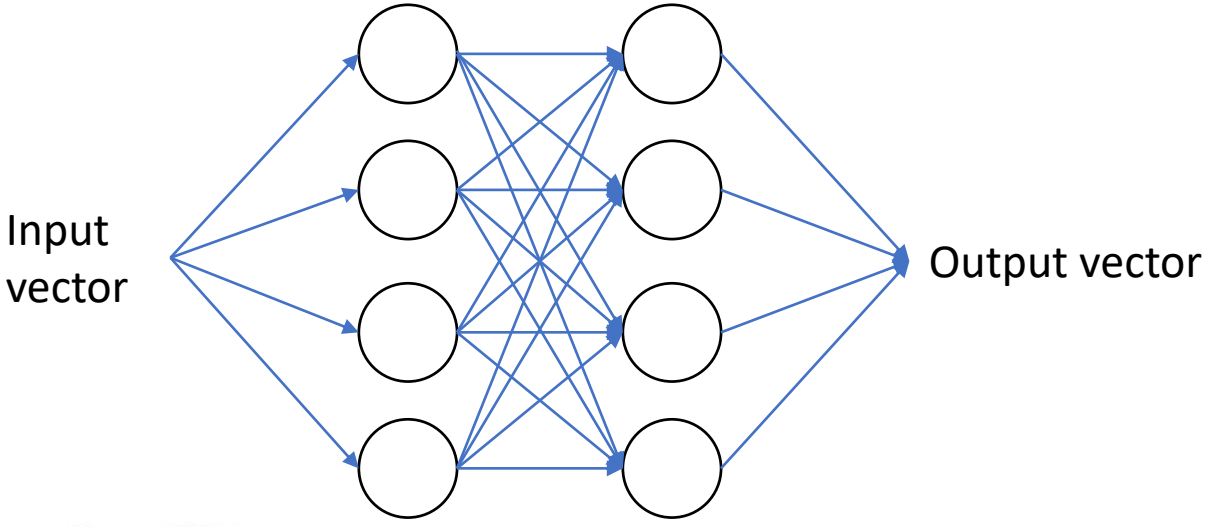
h^2 2-nd layer hidden units

o output

Example of a 2 hidden layer neural network (or 4 layer network, counting also input and output).

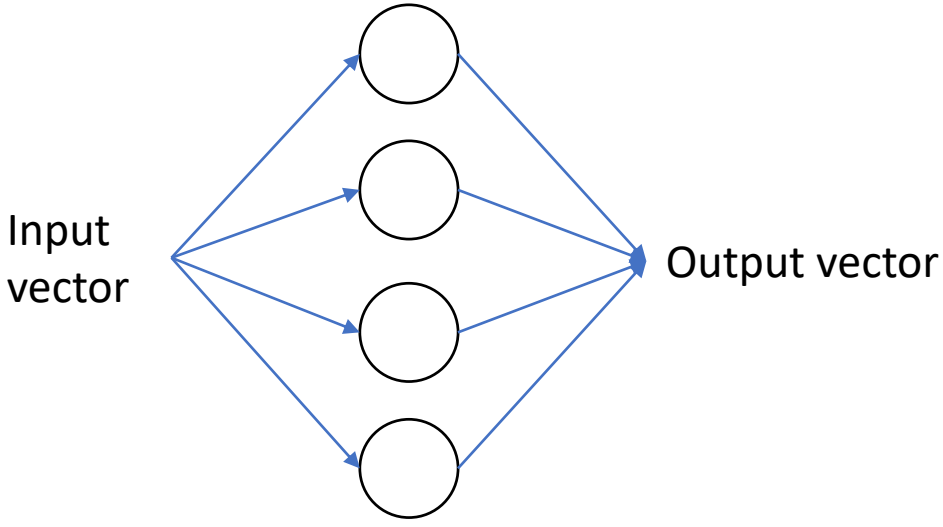
Does anyone pass along the weight without an activation function?

No – this is linear chaining.



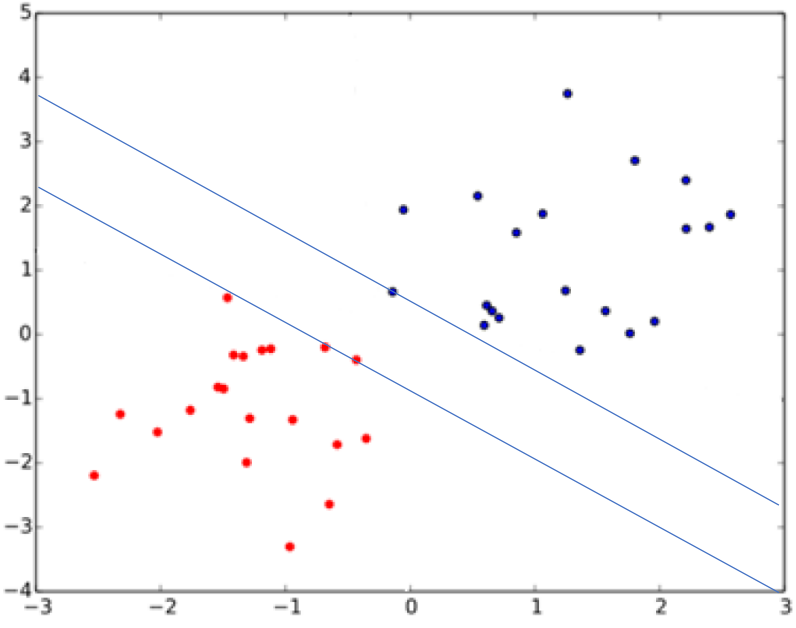
Does anyone pass along the weight without an activation function?

No – this is linear chaining.



What is the relationship between SVMs and perceptrons?

SVMs attempt to learn the support vectors which maximize the margin between classes.



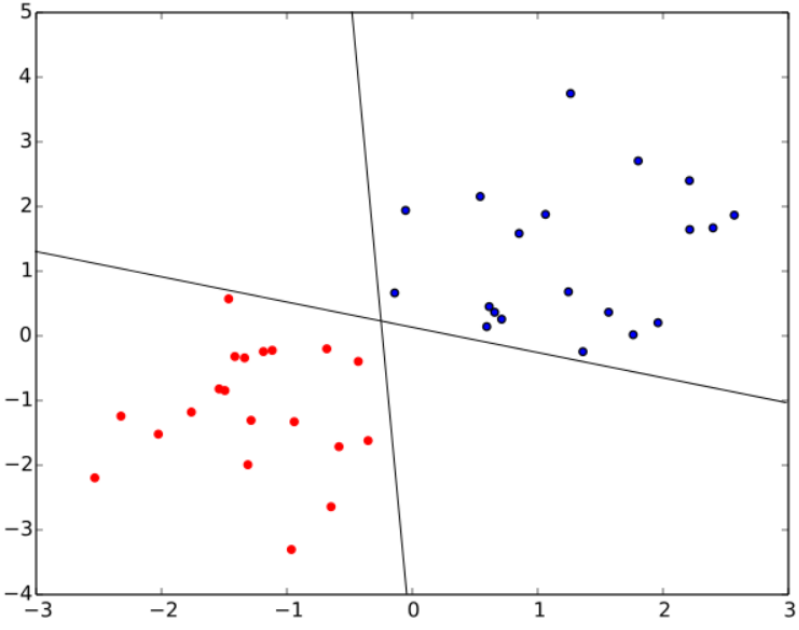
What is the relationship between SVMs and perceptrons?

SVMs attempt to learn the support vectors which maximize the margin between classes.

A perceptron does not.
Both of these perceptron classifiers are equivalent.

'Perceptron of optimal stability' is used in SVM:

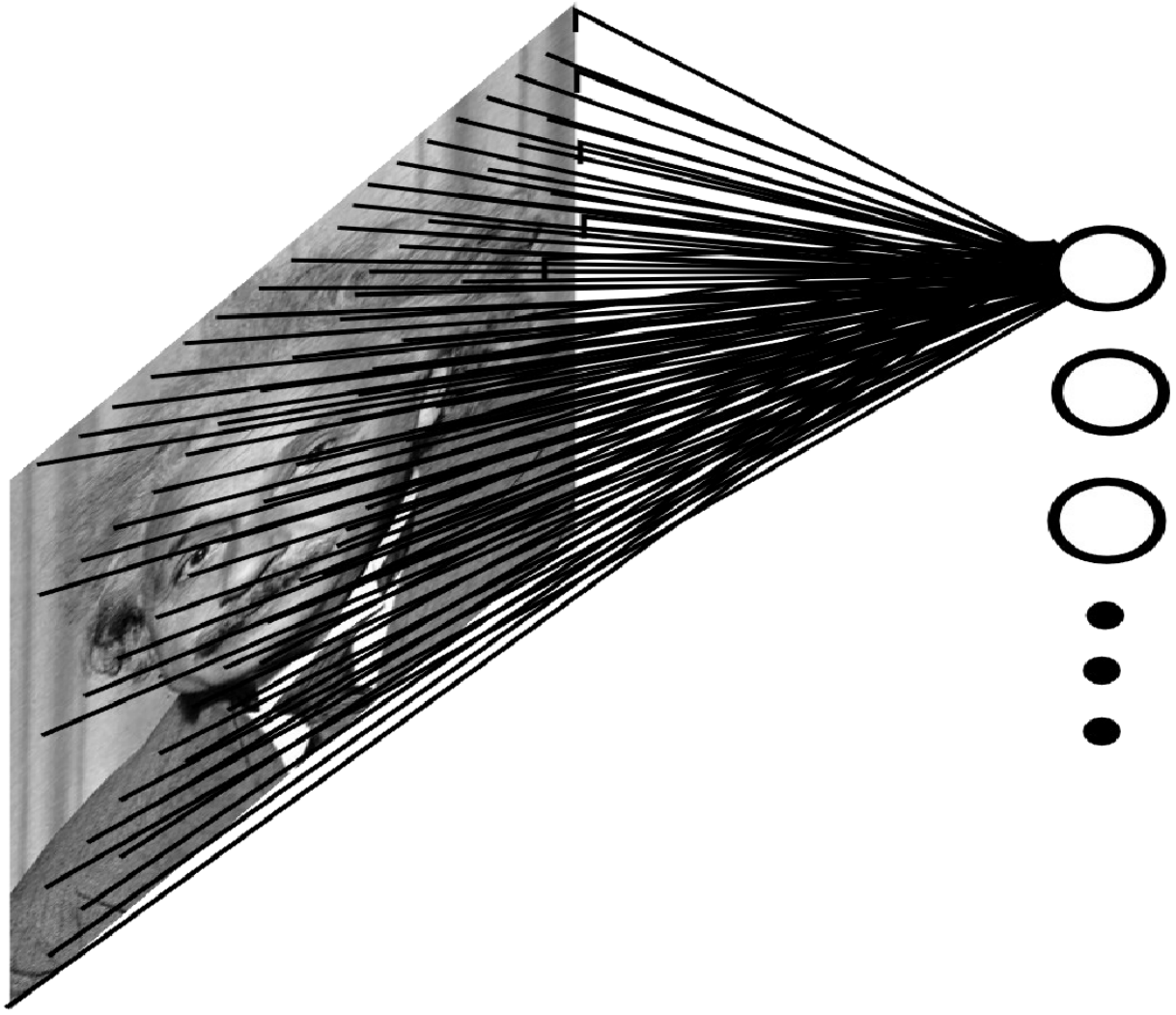
Perceptron
+ optimal stability
+ kernel trick
= *foundations of SVM*



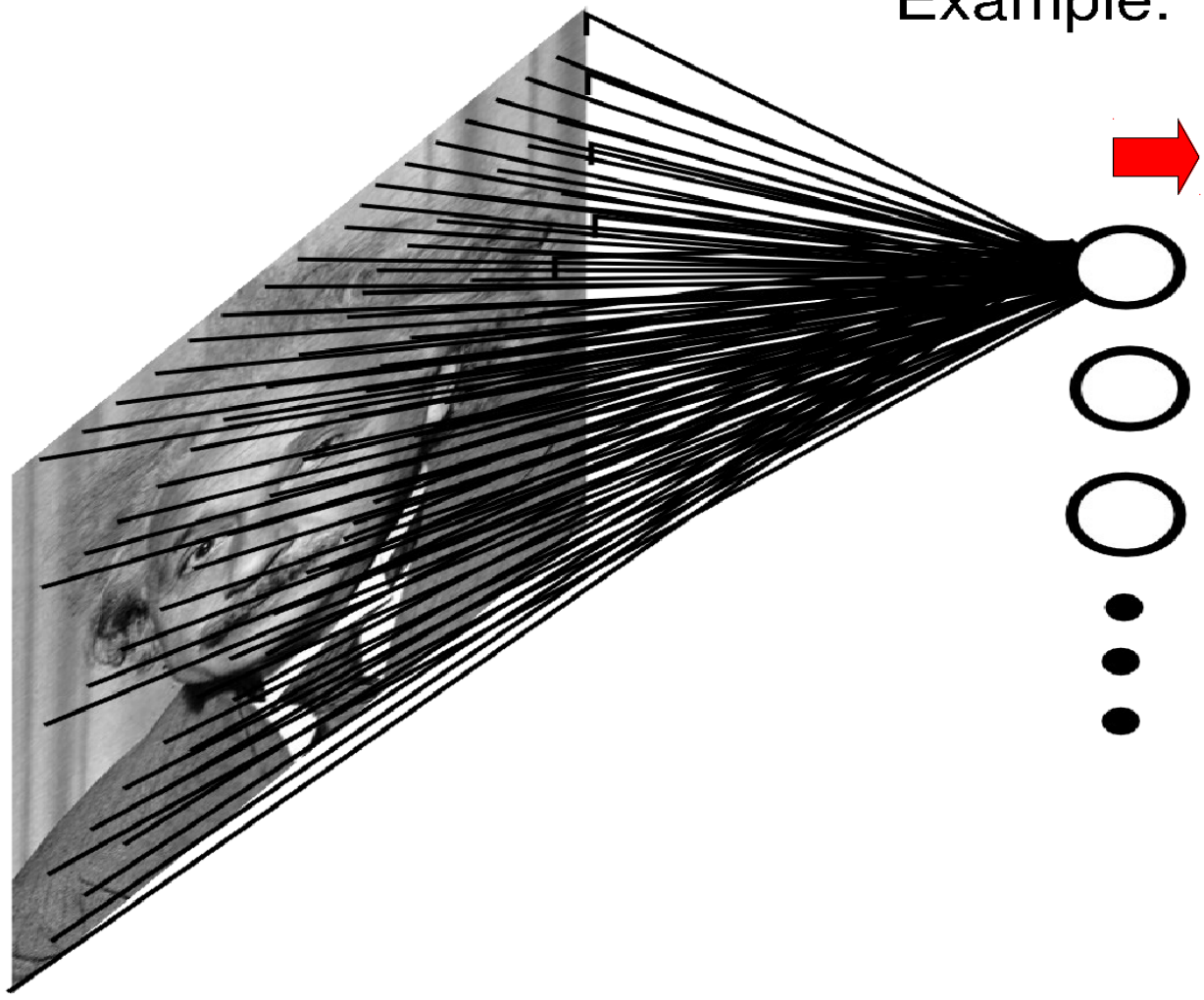
Outline

- Supervised Neural Networks
- Convolutional Neural Networks
- Examples
- Tips

Images as input to neural networks

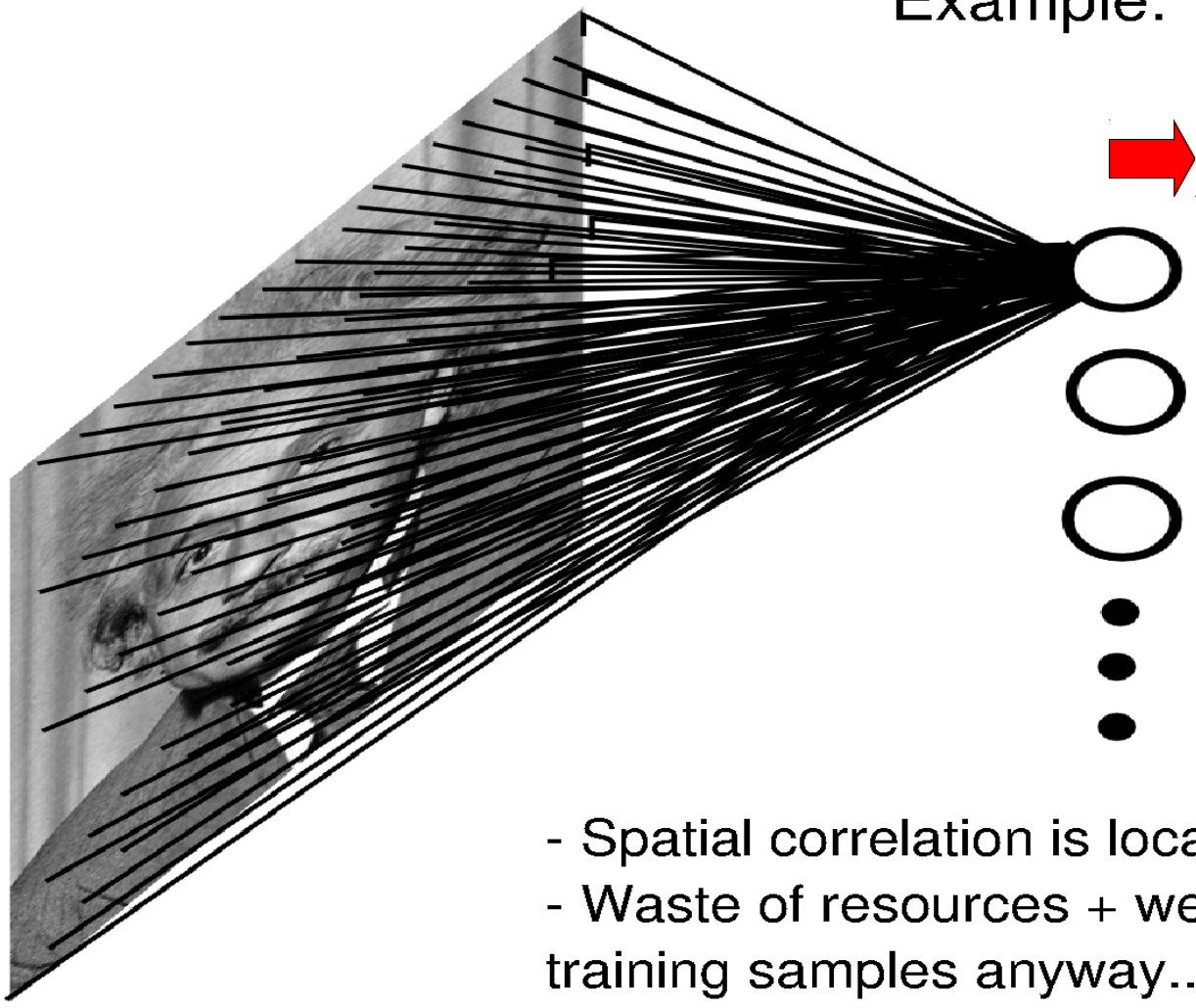


Images as input to neural networks



Example: 200x200 image
40K hidden units
~2B parameters!!!

Images as input to neural networks



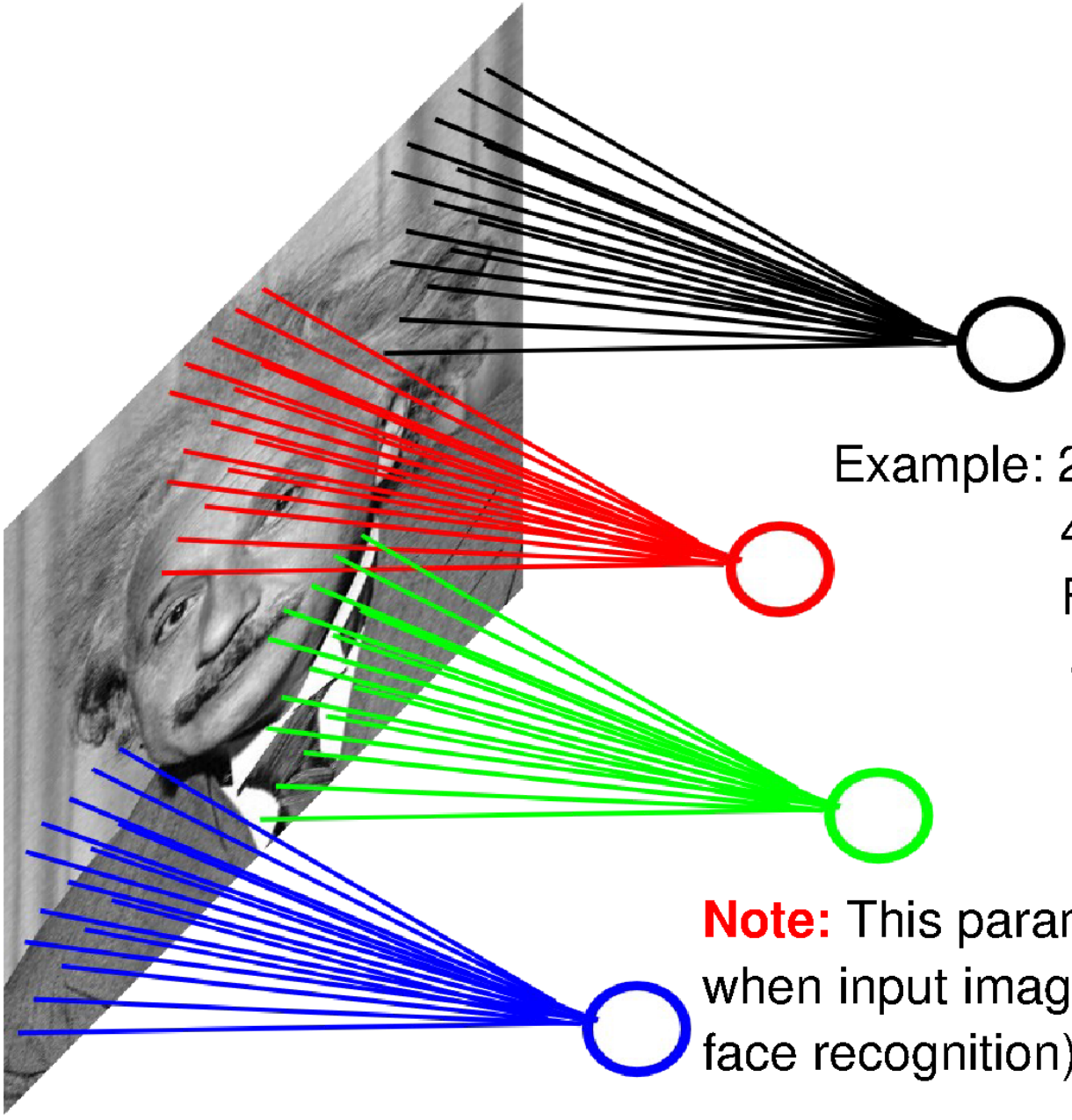
Example: 200x200 image
40K hidden units
~2B parameters!!!

- Spatial correlation is local
- Waste of resources + we have not enough training samples anyway..

Ranzato

Motivation

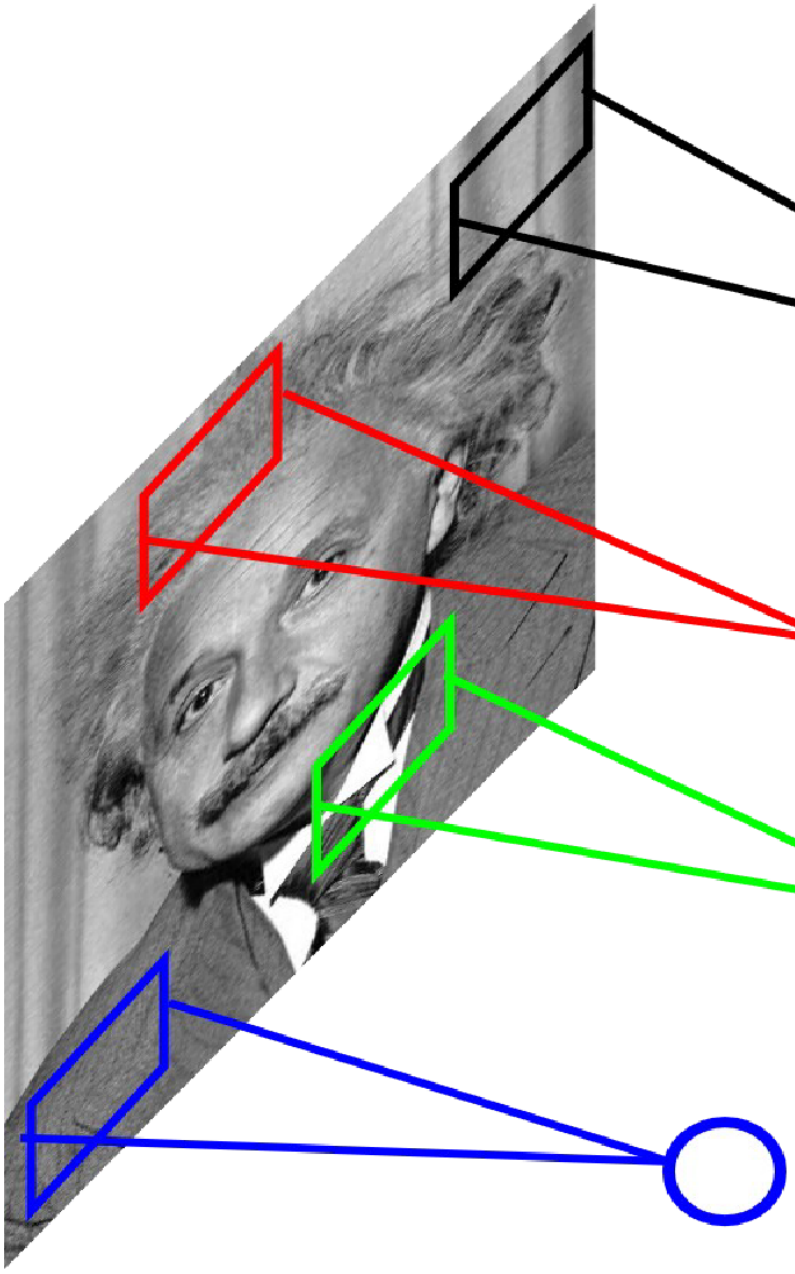
- Sparse interactions – *receptive fields*
  - Assume that in an image, we care about ‘local neighborhoods’ only for a given neural network layer.
  - Composition of layers will expand local -> global.



Example: 200x200 image
40K hidden units
Filter size: 10x10
4M parameters

Note: This parameterization is good when input image is registered (e.g., face recognition).

Ranzato



STATIONARITY? Statistics is similar at different locations

Example: 200x200 image
40K hidden units
Filter size: 10x10
4M parameters

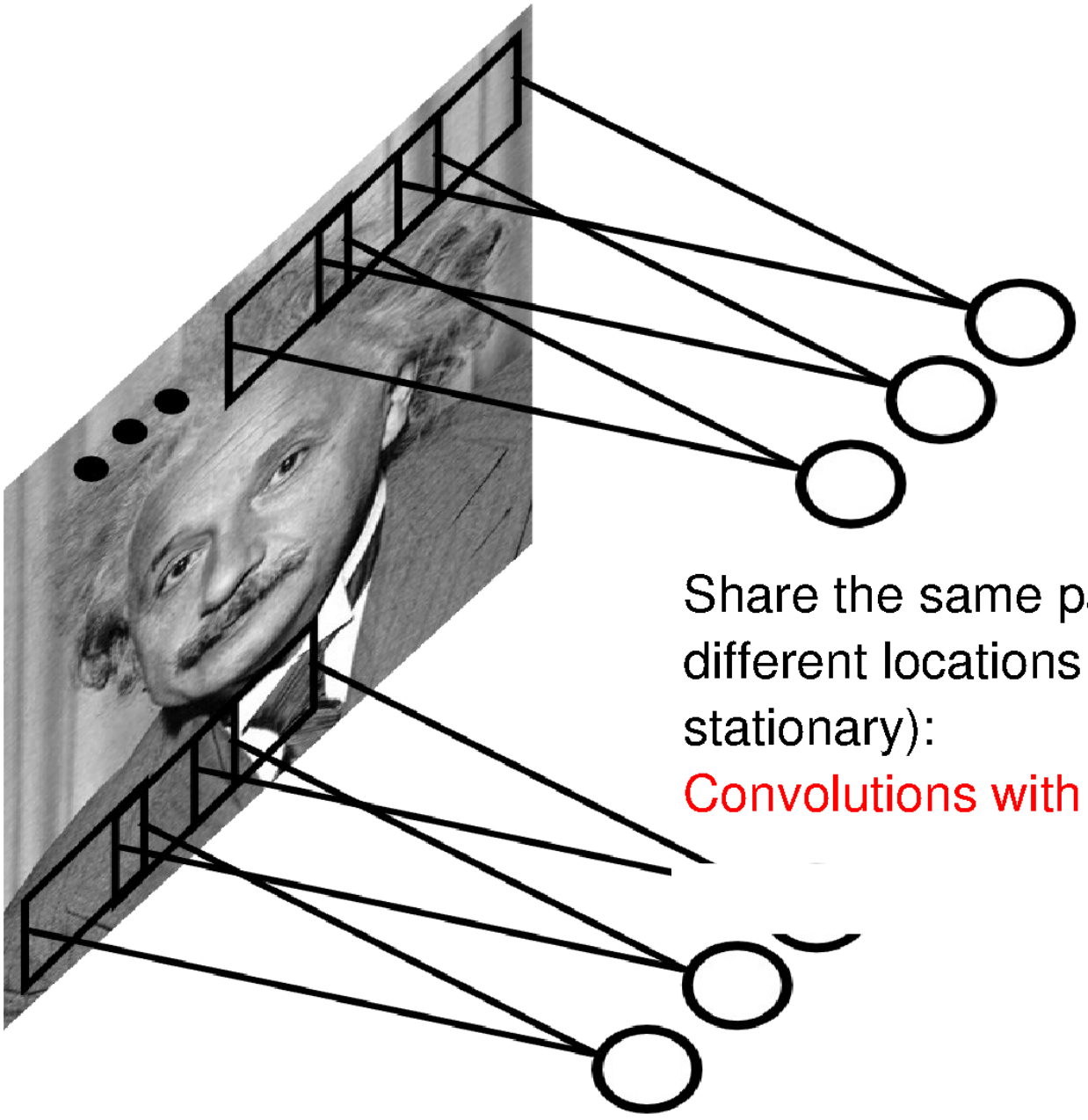
Note: This parameterization is good when input image is registered (e.g., face recognition).

Ranzato

Motivation

- Sparse interactions – *receptive fields*
  - Assume that in an image, we care about 'local neighborhoods' only for a given neural network layer.
  - Composition of layers will expand local -> global.

- Parameter sharing
  - 'Tied weights' – use same weights for more than one perceptron in the neural network.
  - Leads to *equivariant representation*
    - If input changes (e.g., translates), then output changes similarly



Share the same parameters across different locations (assuming input is stationary):

Convolutions with learned kernels

Filtering reminder:
Correlation (rotated convolution)

$f[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$I[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

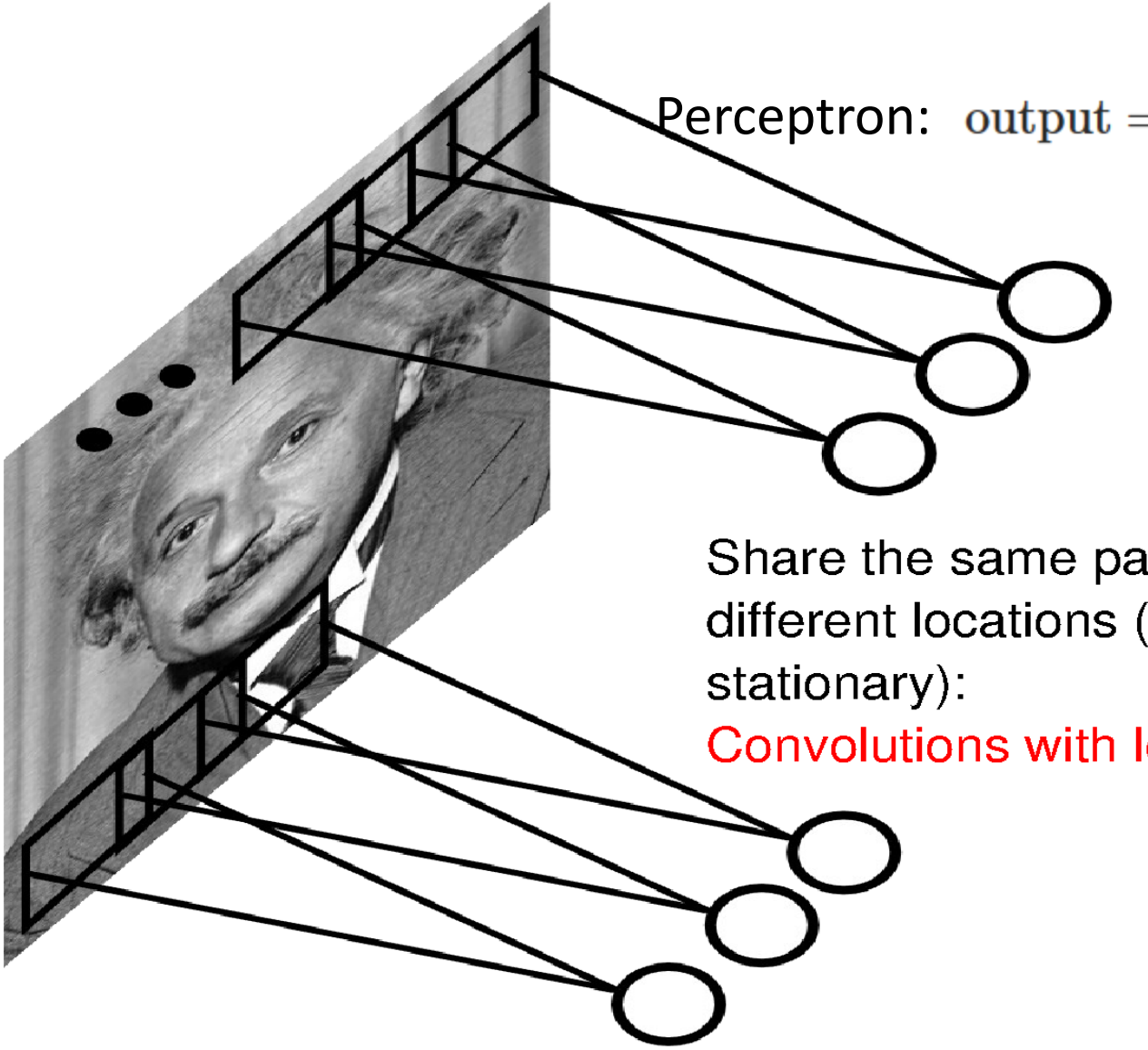
$h[.,.]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$

Credit: S. Seitz

Convolutional Layer



Perceptron: $\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases}$

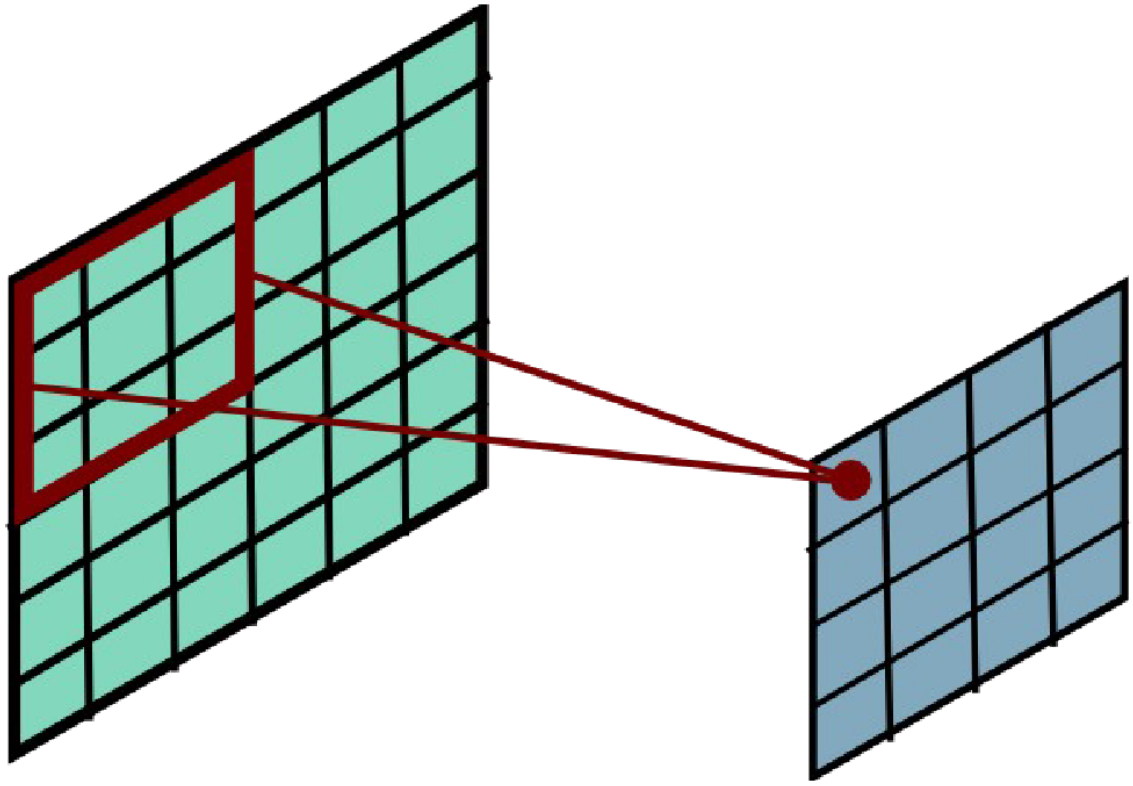
$w \cdot x \equiv \sum_j w_j x_j$

This is convolution!

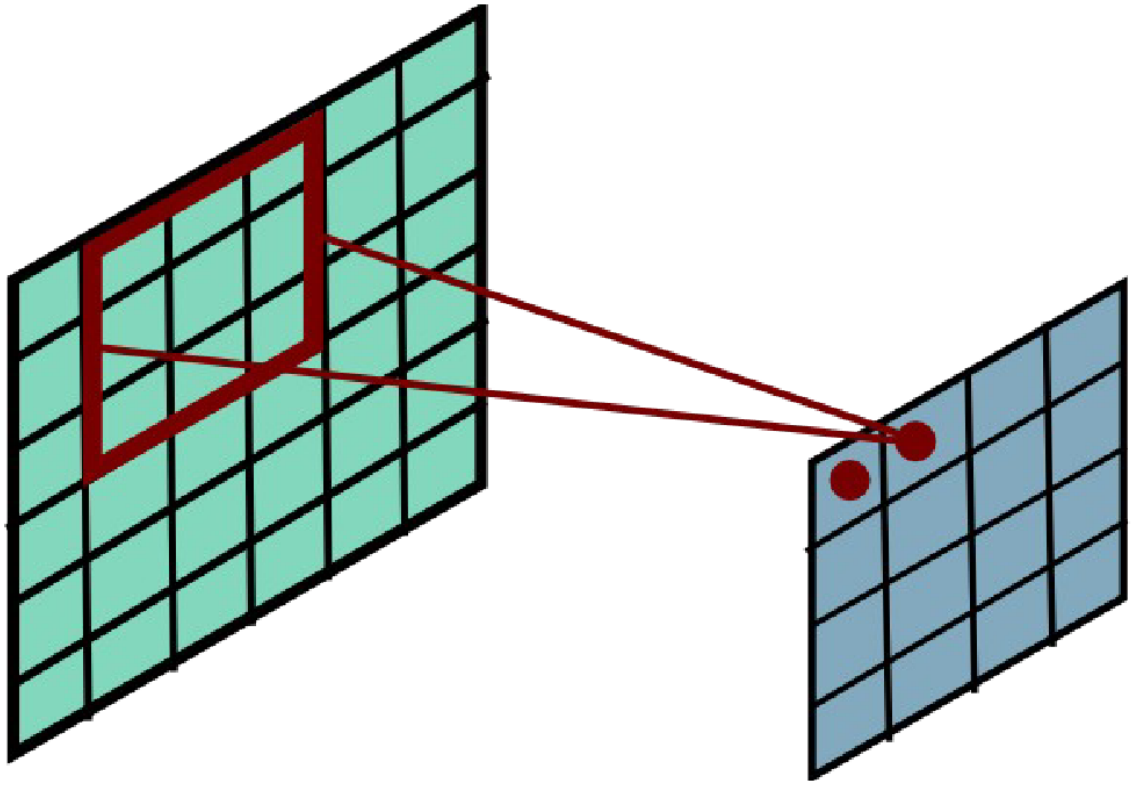
Share the same parameters across different locations (assuming input is stationary):

Convolutions with learned kernels

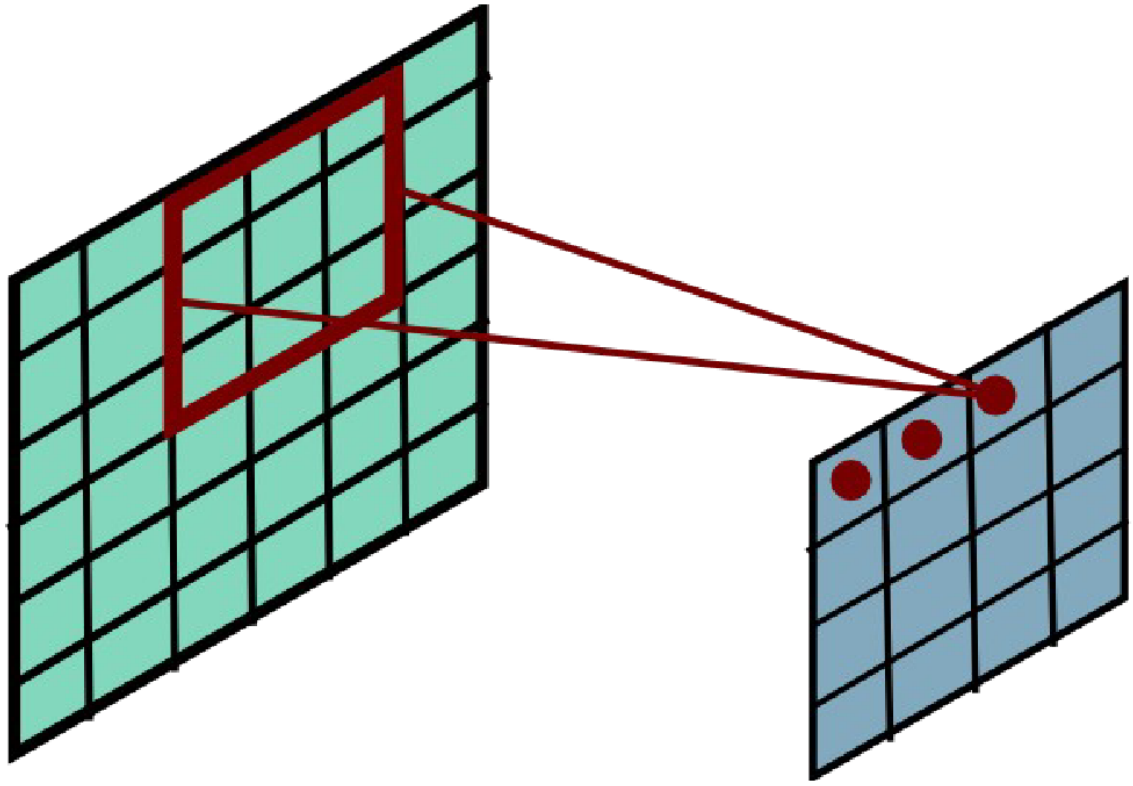
Convolutional Layer



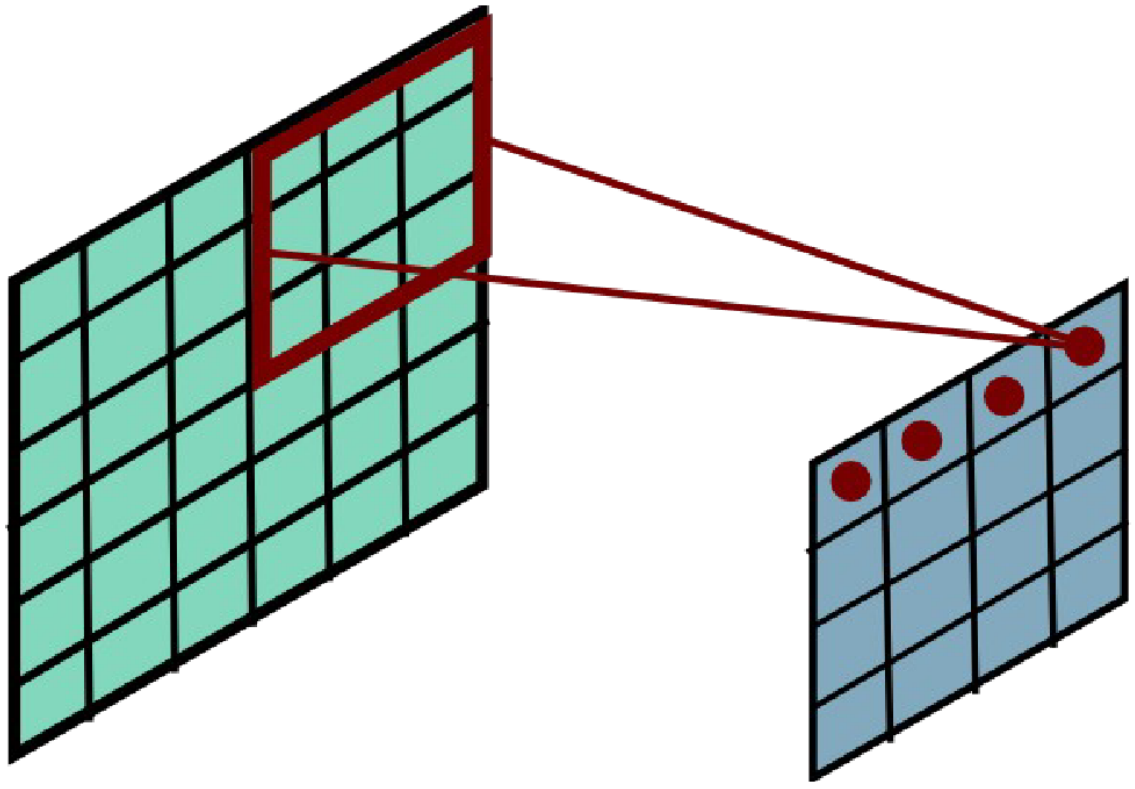
Convolutional Layer



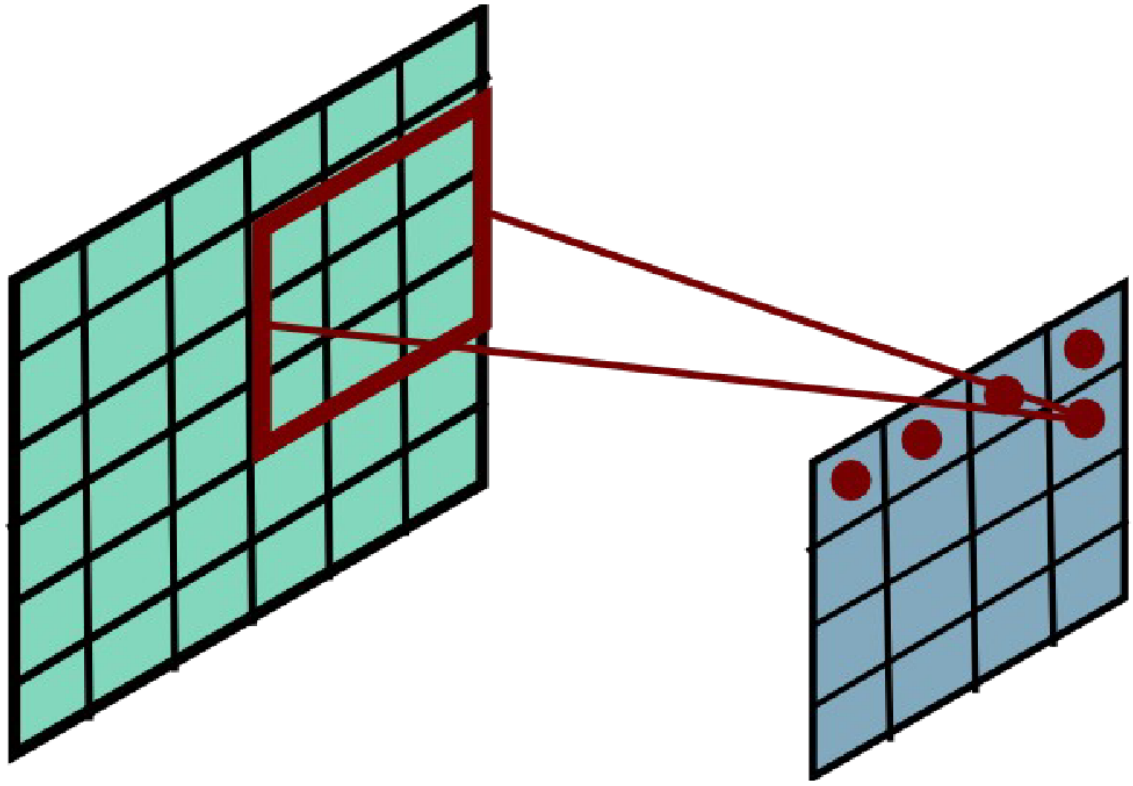
Convolutional Layer



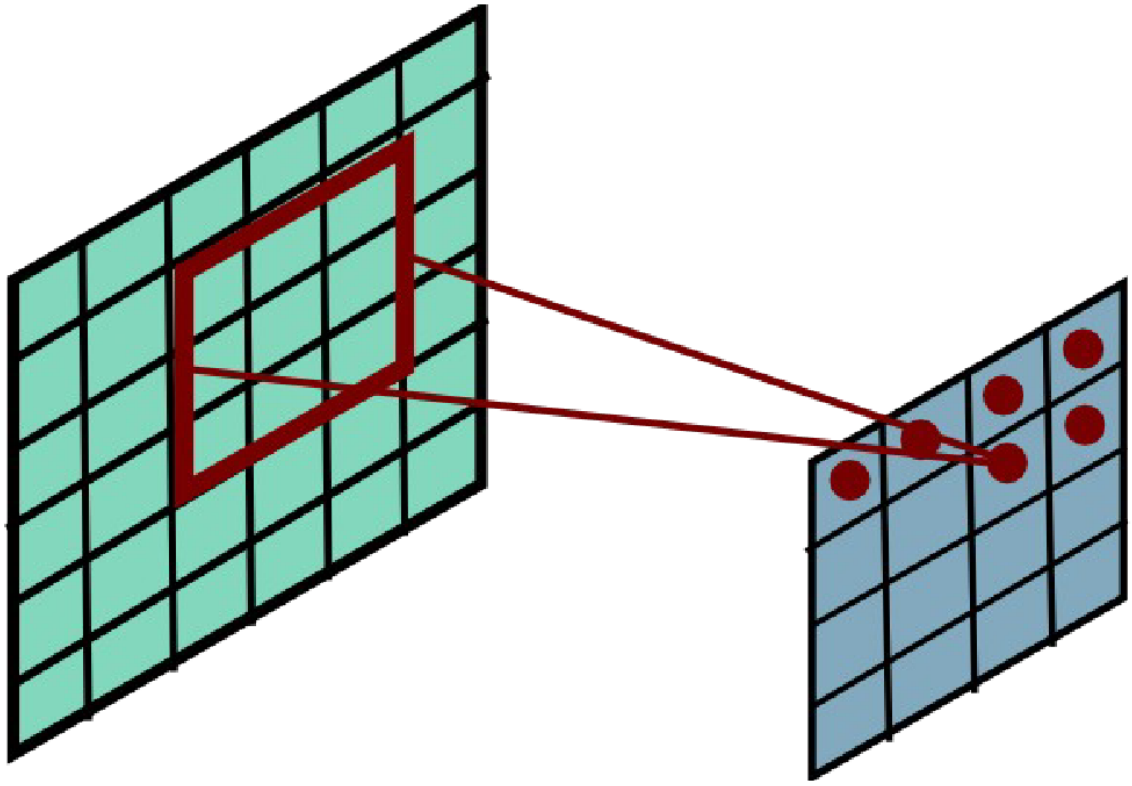
Convolutional Layer



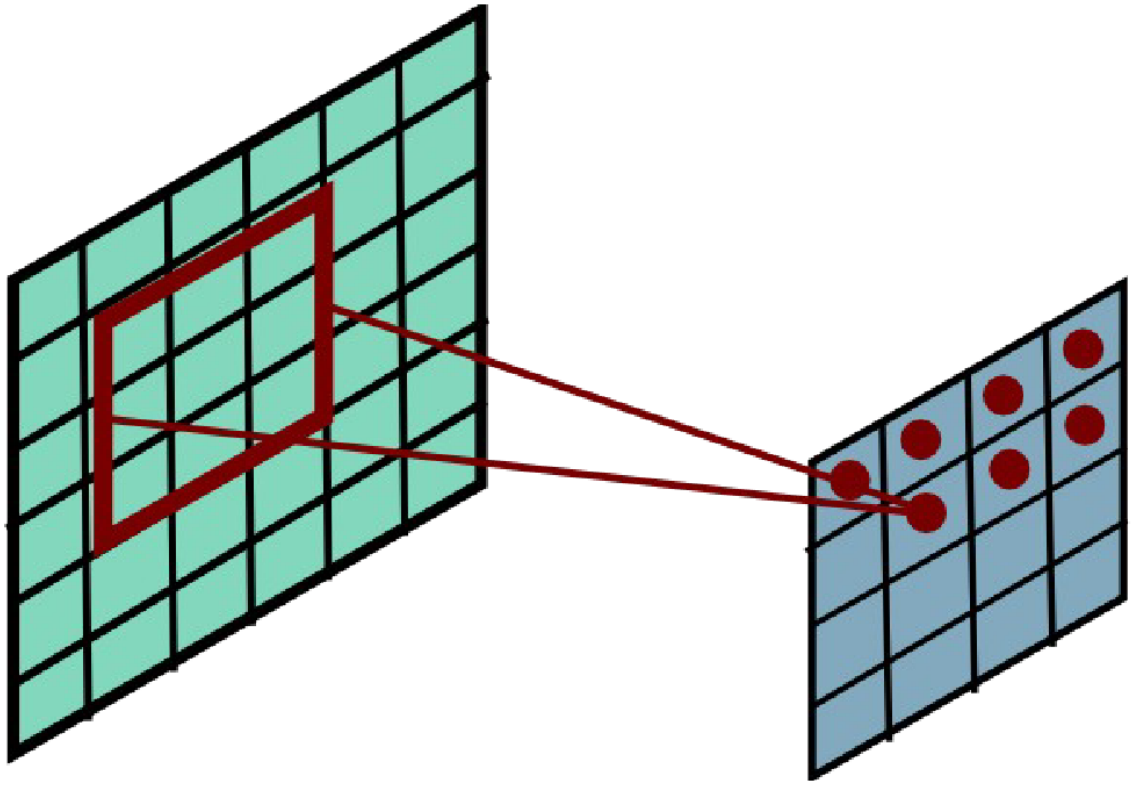
Convolutional Layer



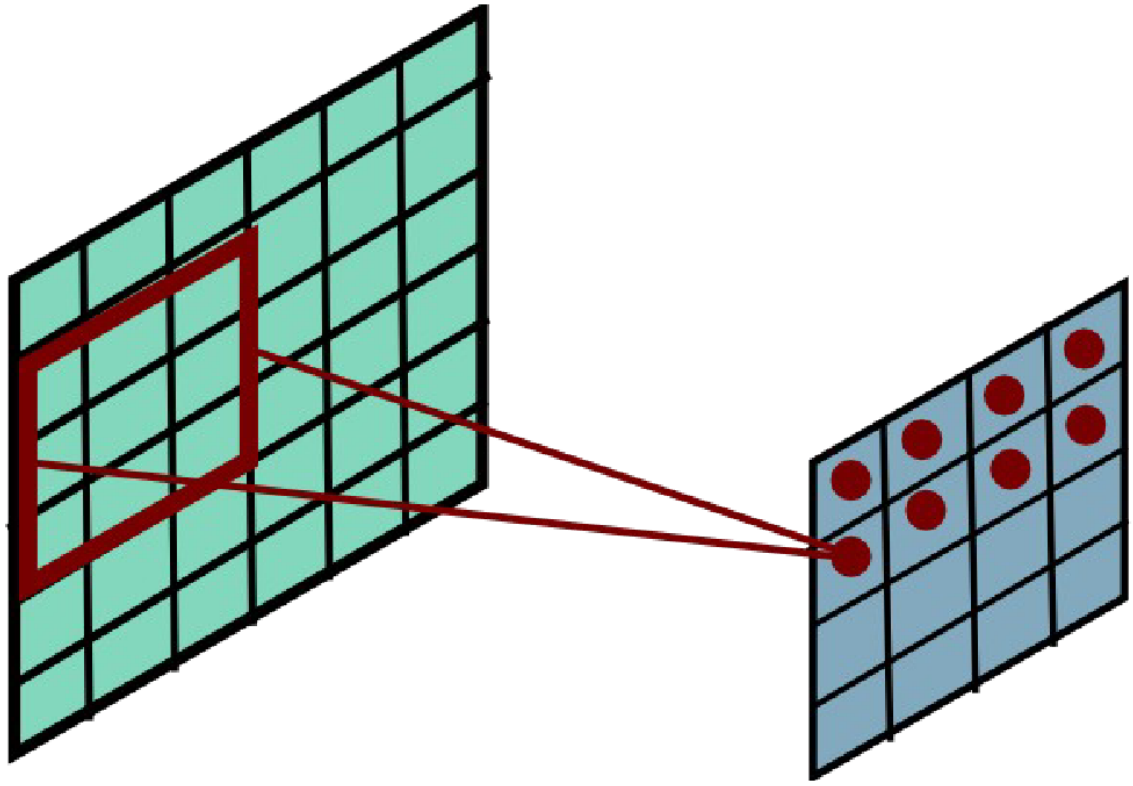
Convolutional Layer



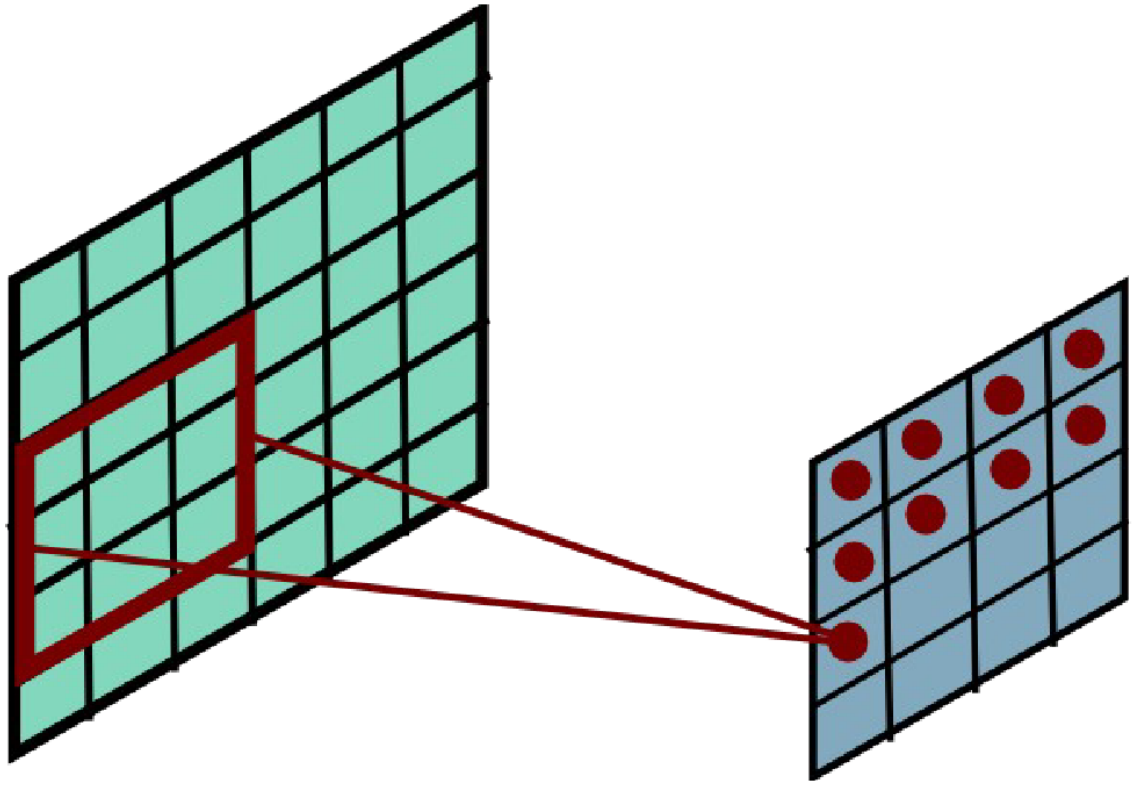
Convolutional Layer



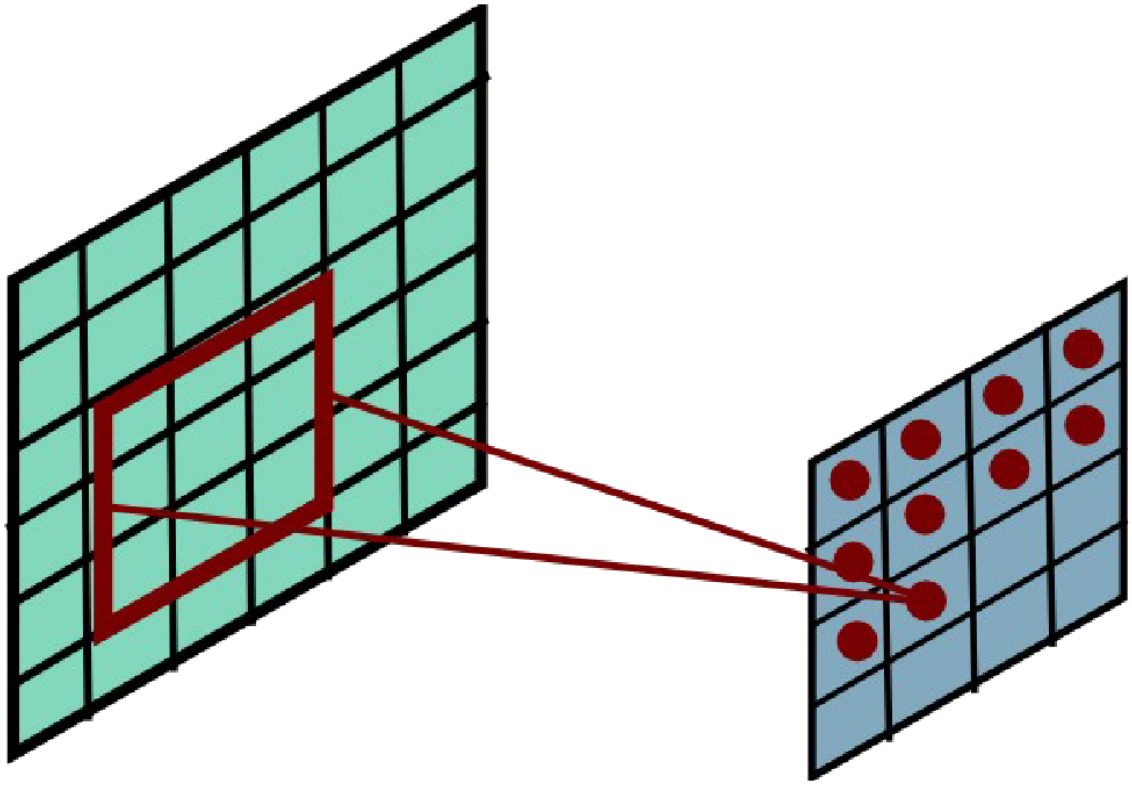
Convolutional Layer



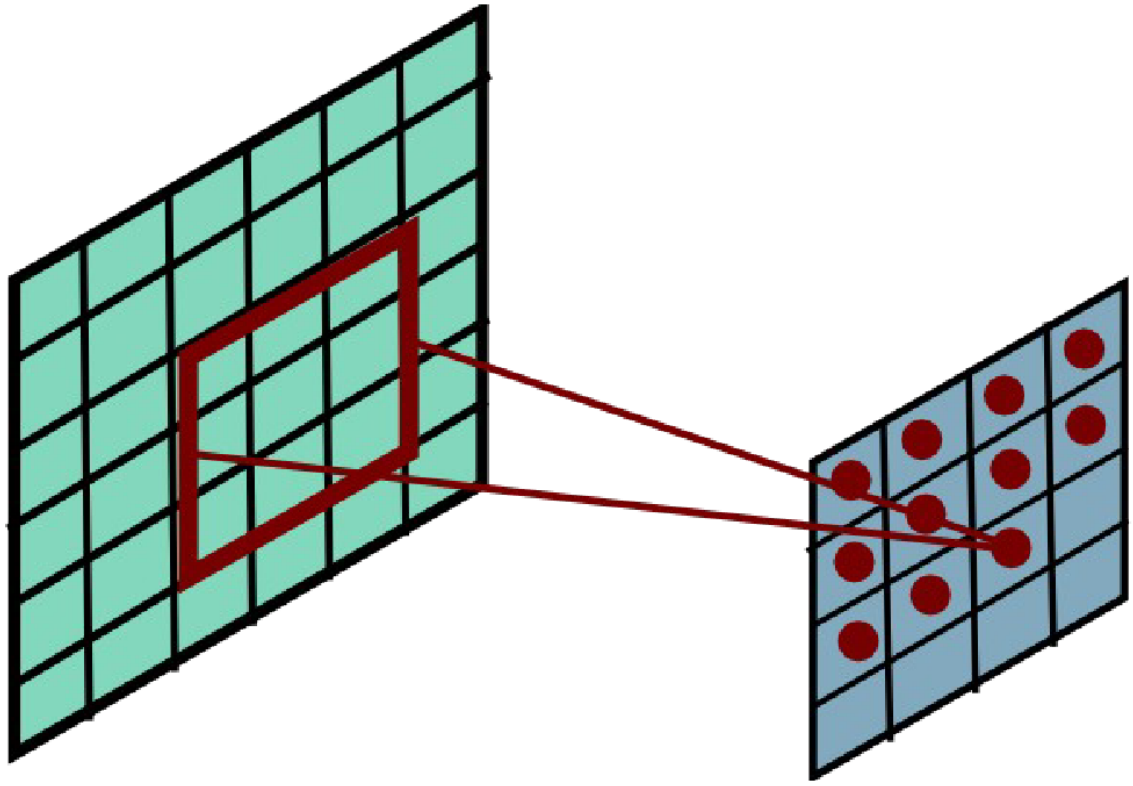
Convolutional Layer



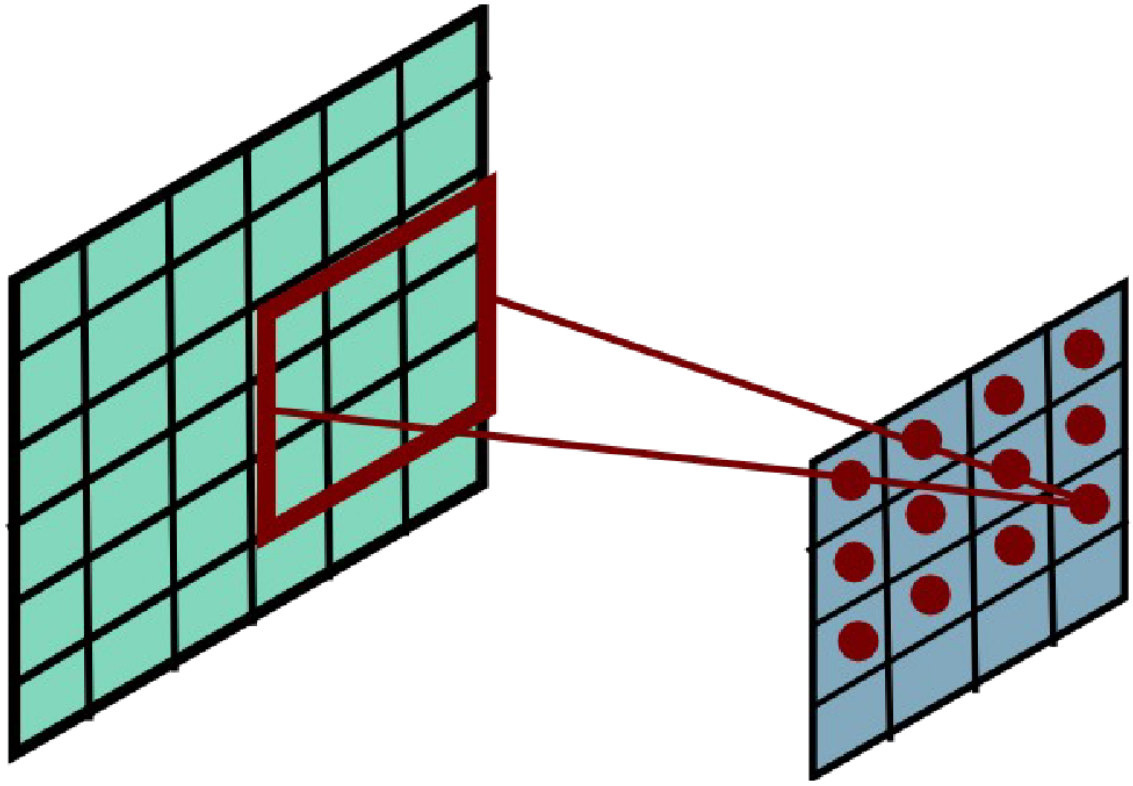
Convolutional Layer



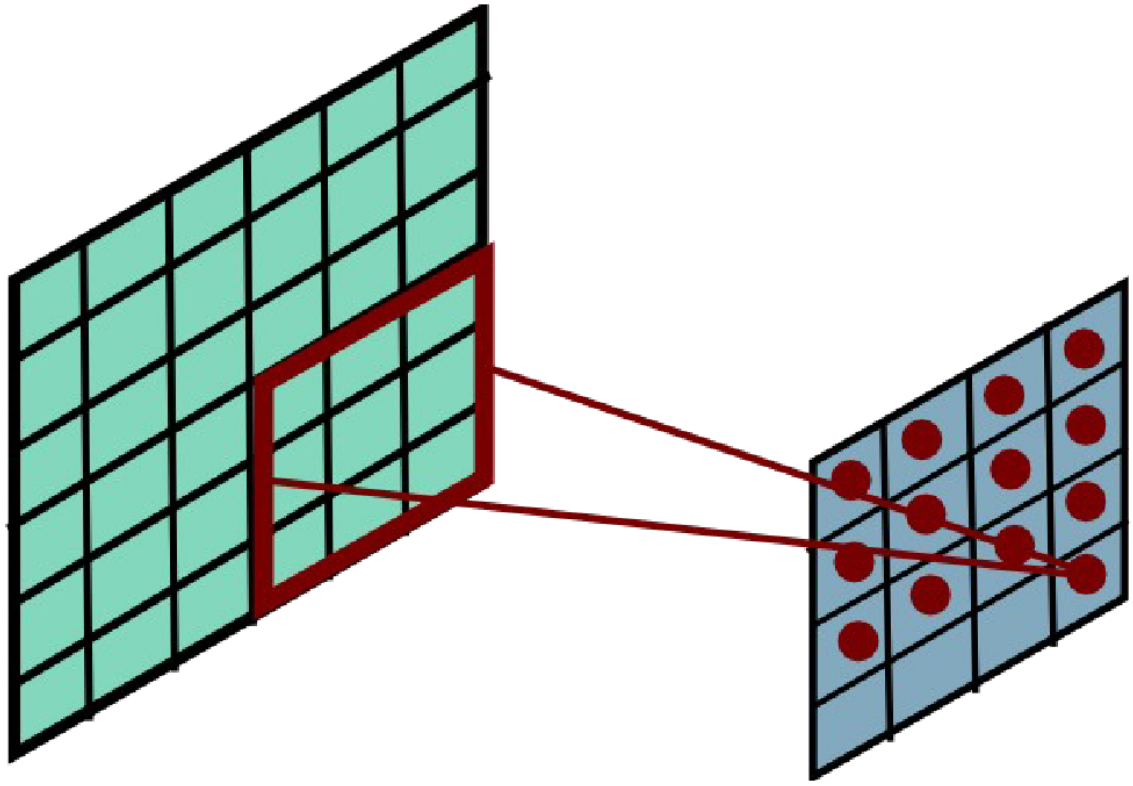
Convolutional Layer



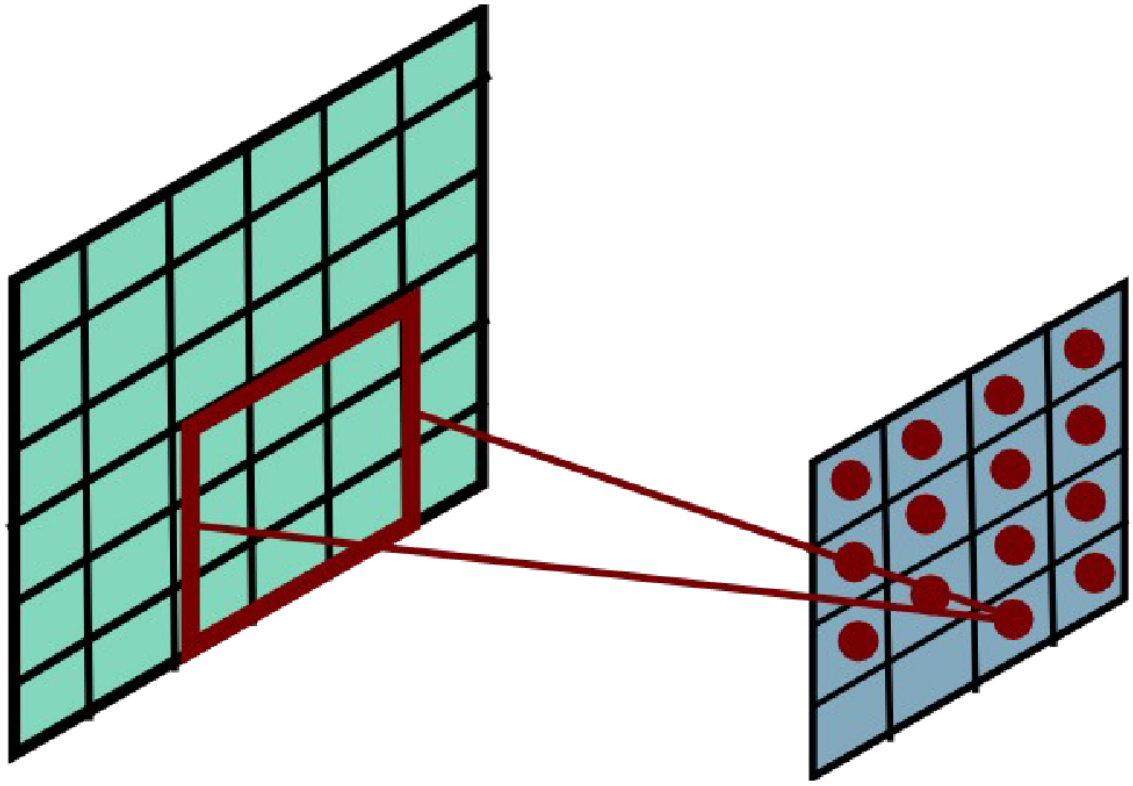
Convolutional Layer



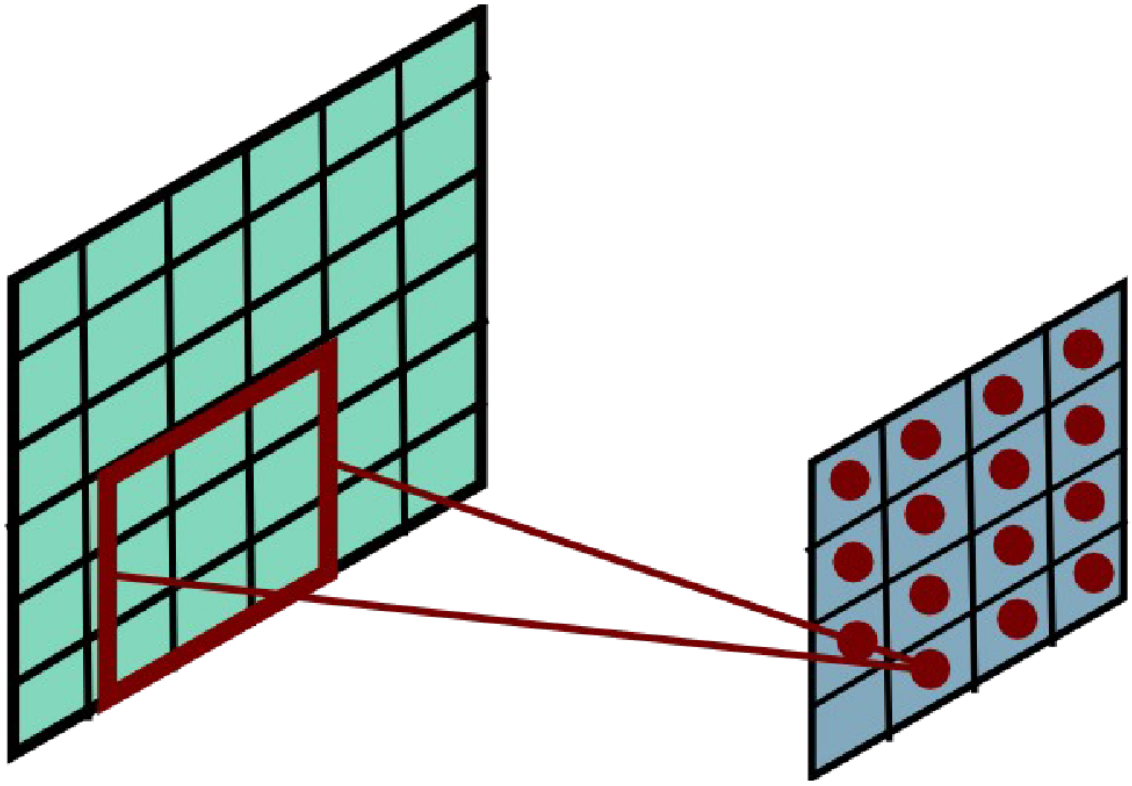
Convolutional Layer



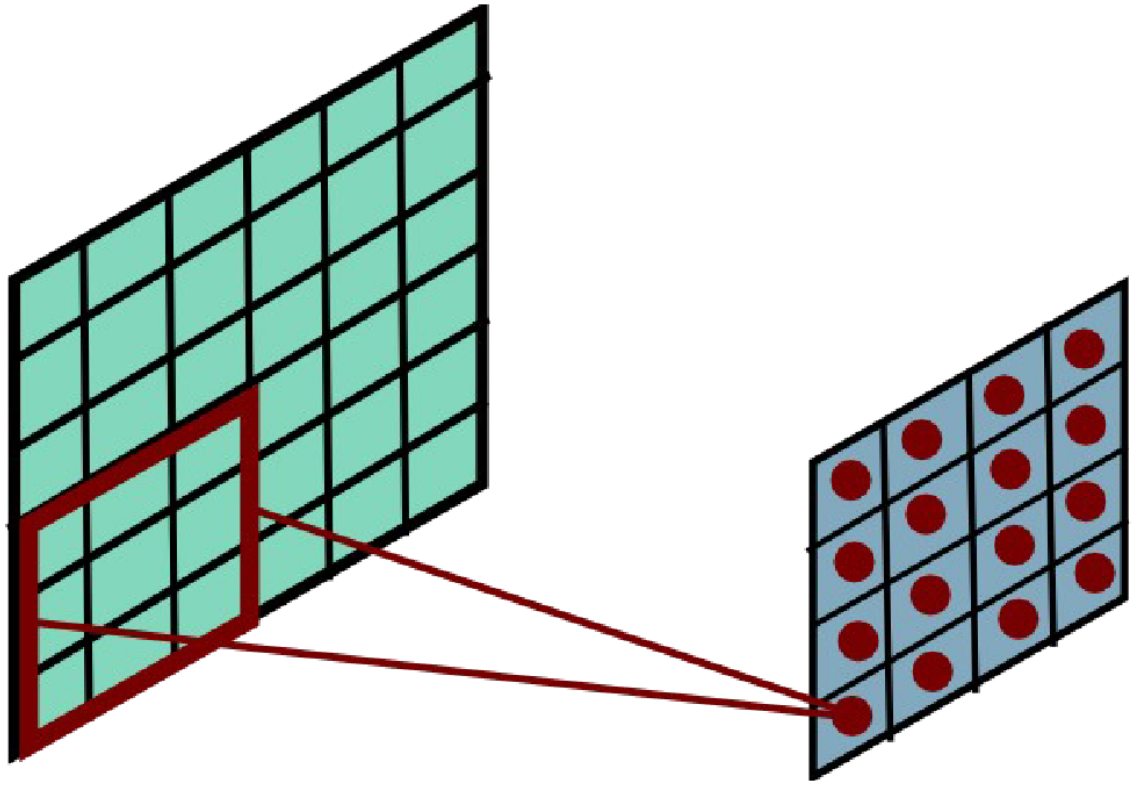
Convolutional Layer



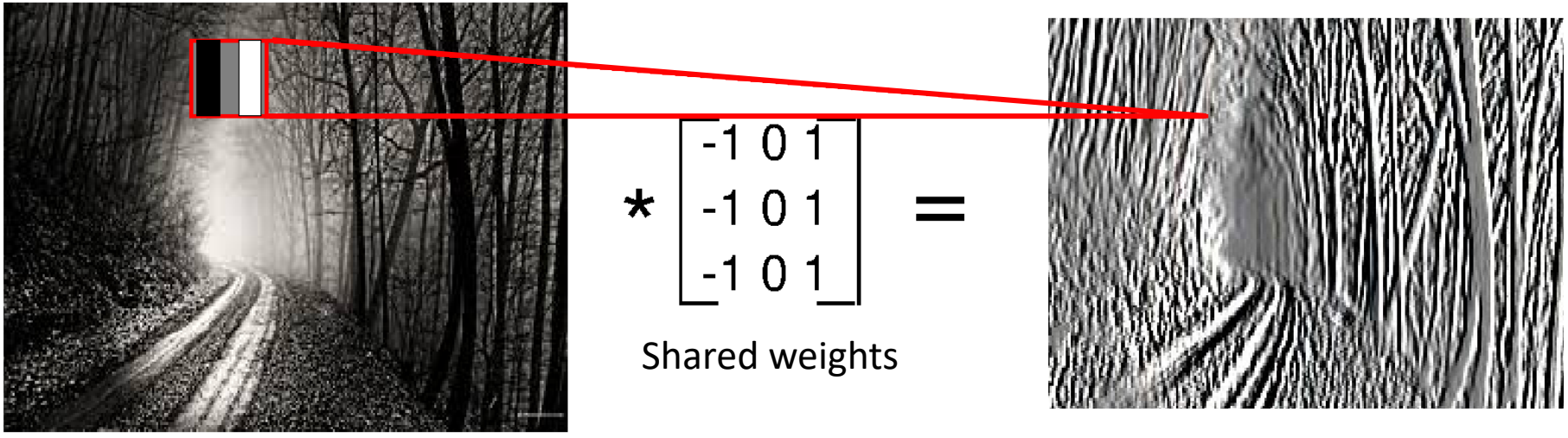
Convolutional Layer



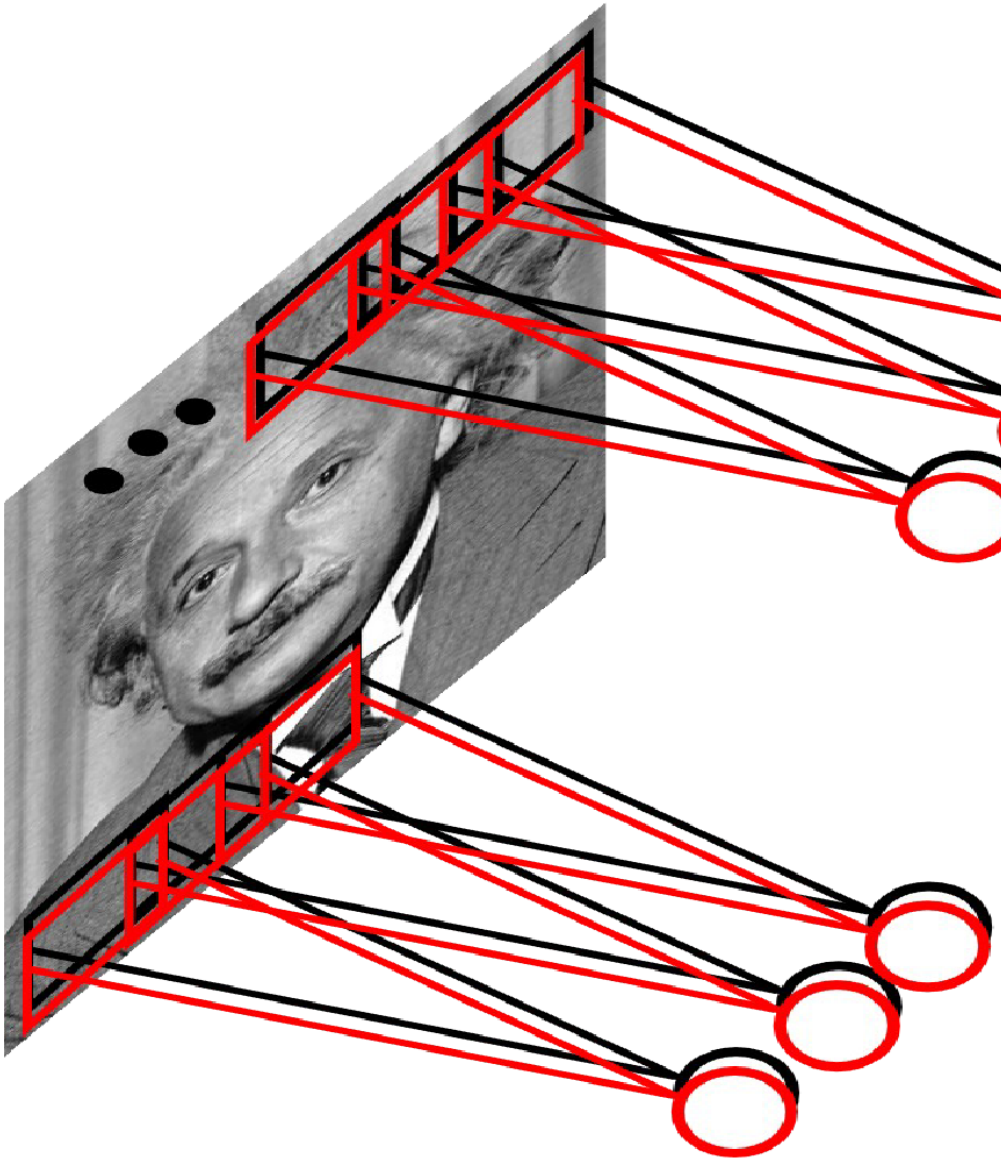
Convolutional Layer



Convolutional Layer



Convolutional Layer



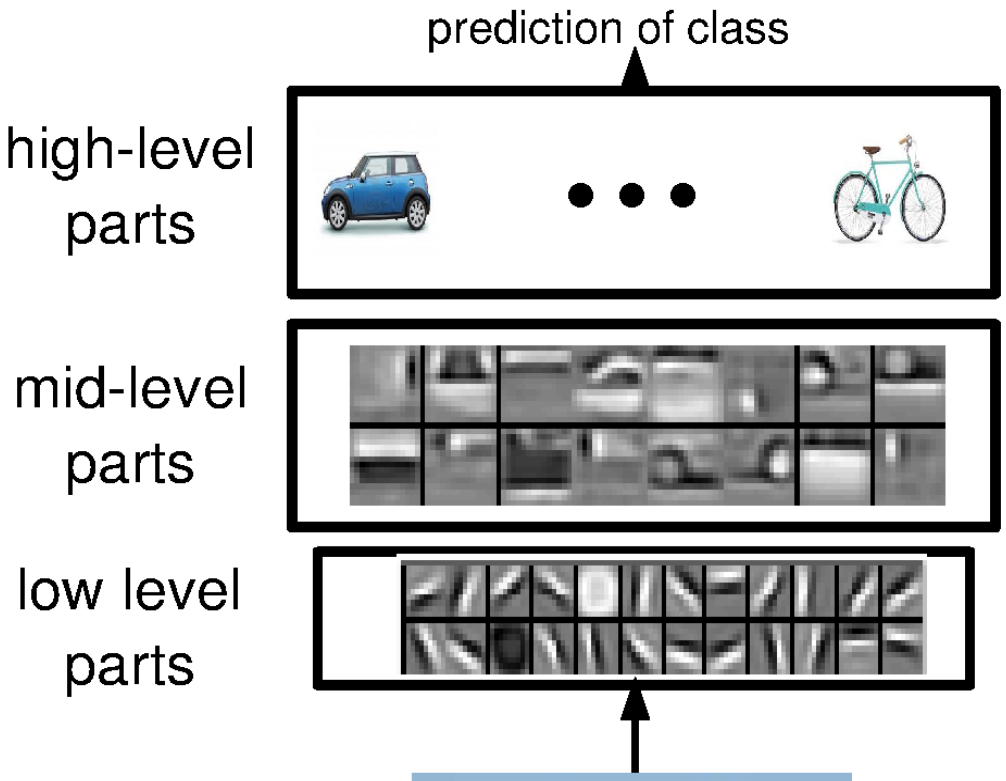
Learn multiple filters.

Filter = 'local' perceptron.
Also called *kernel.*

E.g.: 200x200 image
100 Filters
Filter size: 10x10
10K parameters

Ranzato

Interpretation



- distributed representations
- feature sharing
- compositionality

Lee et al. “Convolutional DBN's ...” ICML 2009

Convolutional Layer

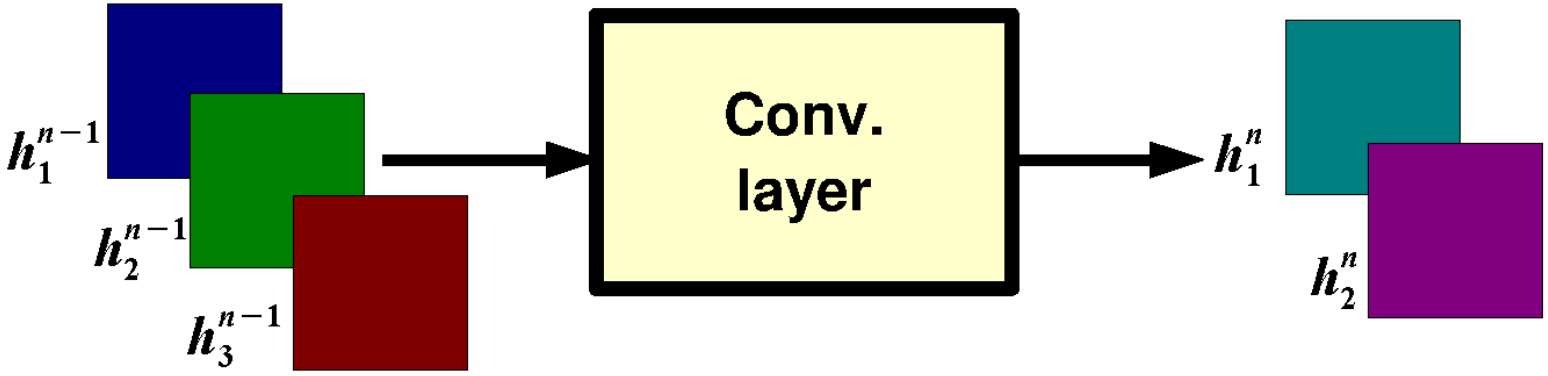
$$h_j^n = max\left(0, \sum_{k=1}^{K} h_k^{n-1} * w_{kj}^n\right)$$

output feature map

input feature map

kernel

n = layer number
K = kernel size
j = # channels (input) or # filters (depth)



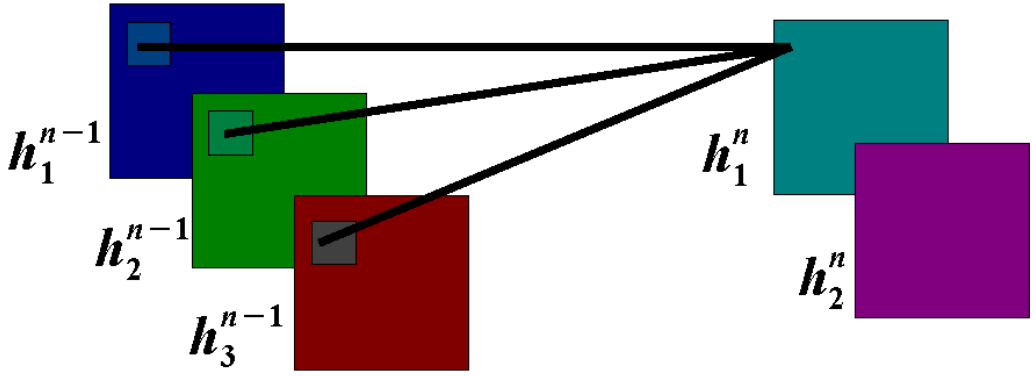
Convolutional Layer

$$h_j^n = max\left(0, \sum_{k=1}^{K} h_k^{n-1} * w_{kj}^n\right)$$

output feature map

input feature map

kernel



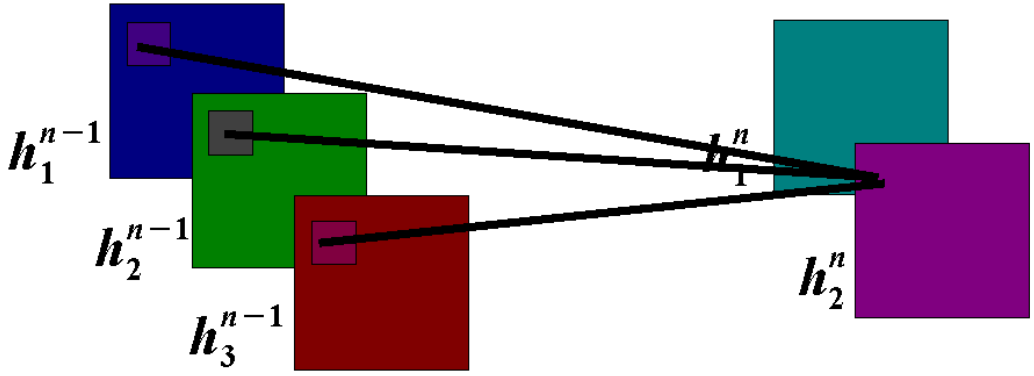
Convolutional Layer

$$h_j^n = max\left(0, \sum_{k=1}^{K} h_k^{n-1} * w_{kj}^n\right)$$

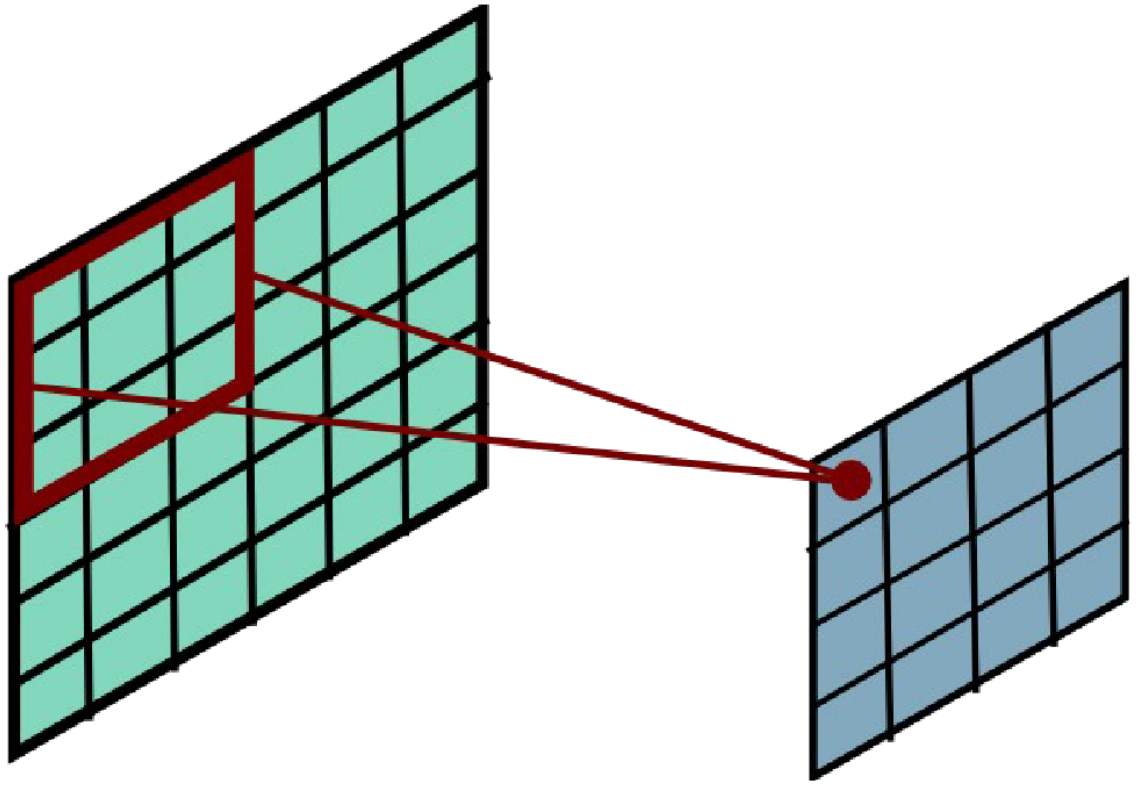
output feature map

input feature map

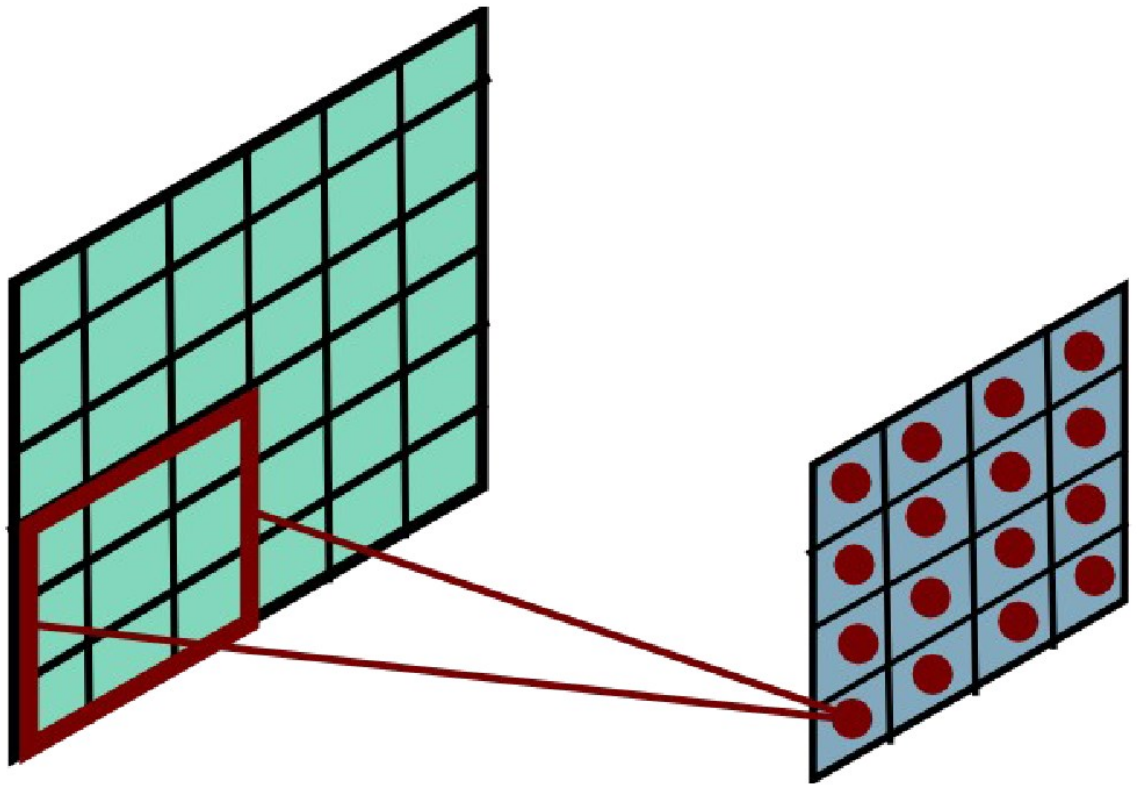
kernel



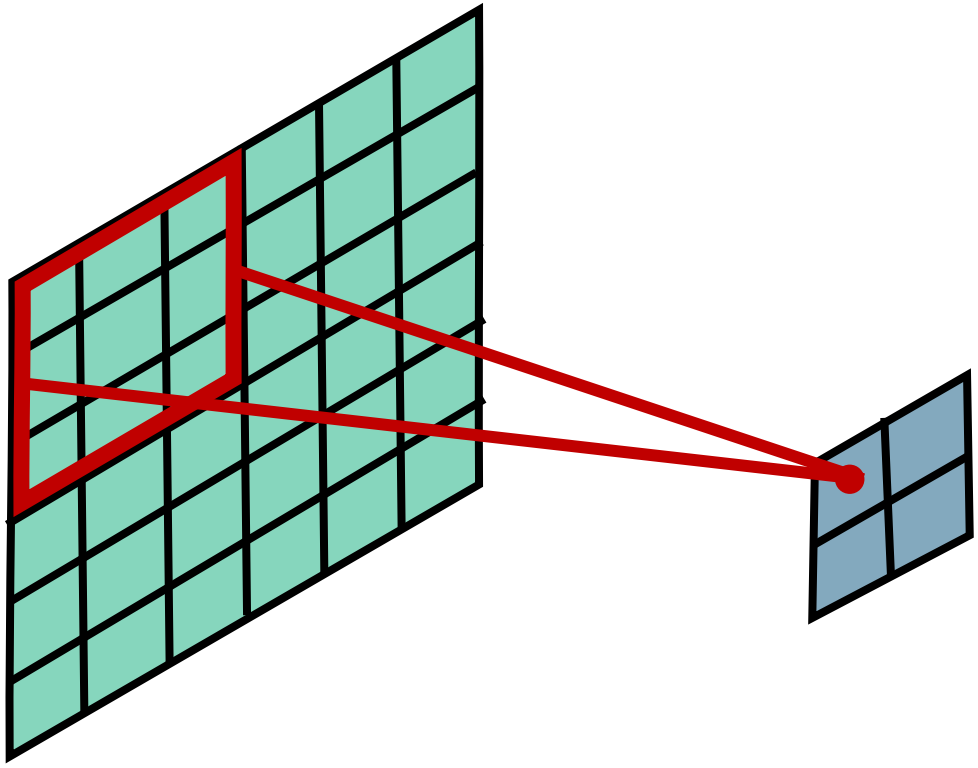
Stride = 1



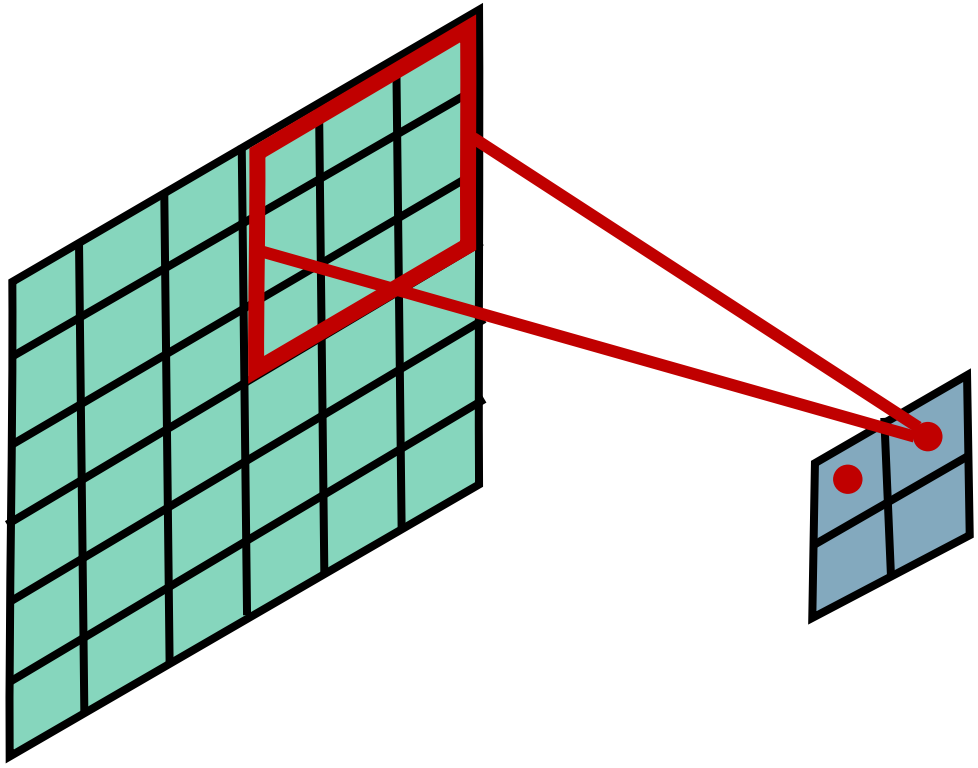
Stride = 1



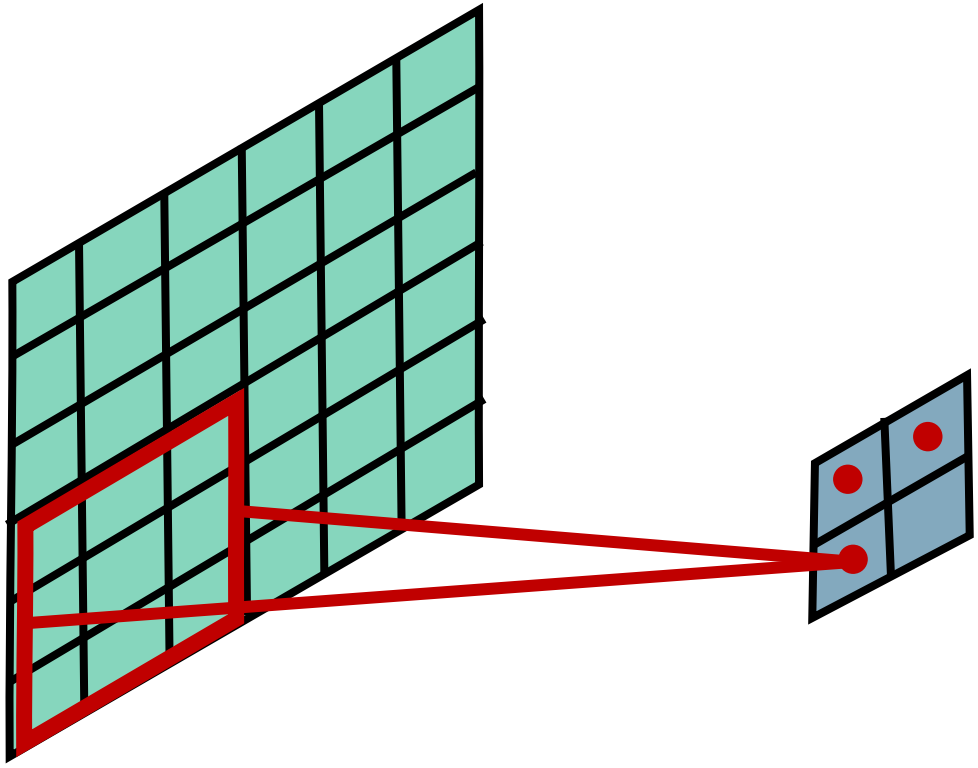
Stride = 3



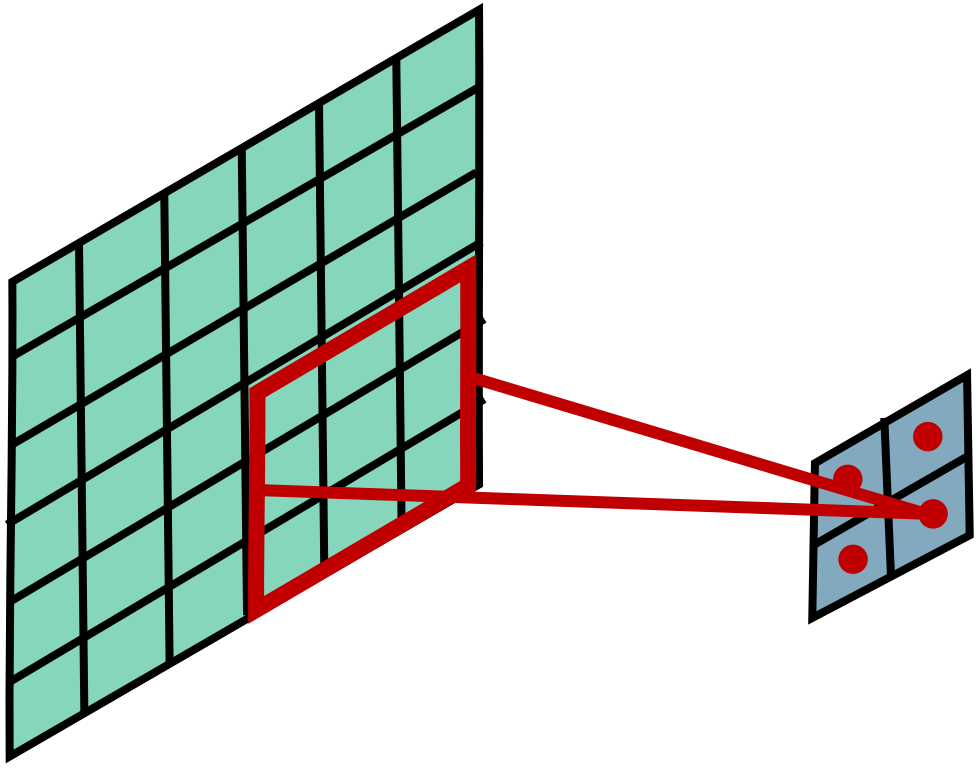
Stride = 3



Stride = 3



Stride = 3



Convolutional Layer

Question: What is the size of the output? What's the computational cost?

Answer: It is proportional to the number of filters and depends on the stride. If kernels have size KxK, input has size DxD, stride is 1, and there are M input feature maps and N output feature maps then:

- the input has size M@DxD
- the output has size N@(D-K+1)x(D-K+1)
- the kernels have MxNxKxK coefficients (which have to be learned)
- cost: M*K*K*N*(D-K+1)*(D-K+1)

Convolutional Layer

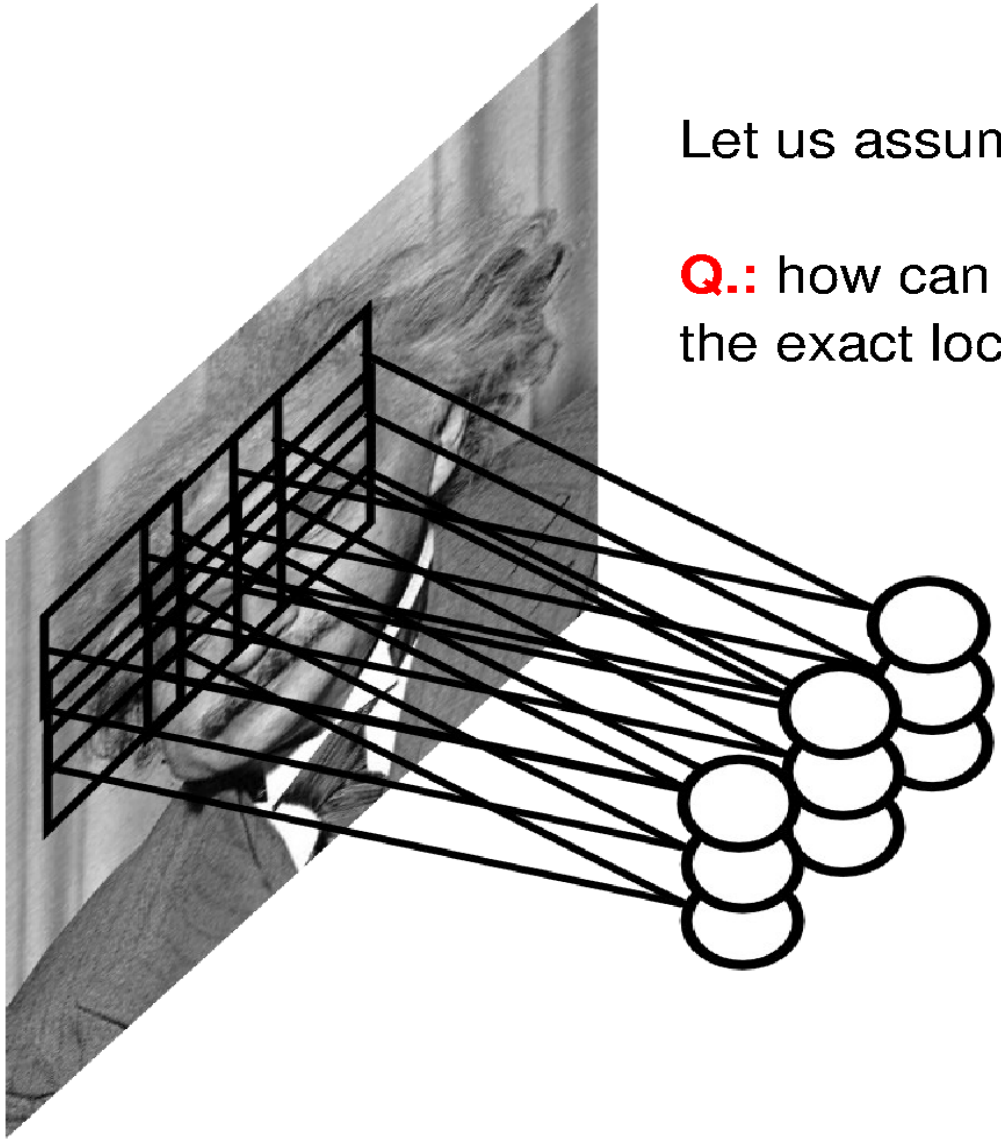
Question: What is the size of the output? What's the computational cost?

Answer: It is proportional to the number of filters and depends on the stride. If kernels have size KxK, input has size DxD, stride is 1, and there are M input feature maps and N output feature maps then:
- the input has size M@DxD
- the output has size N@(D-K+1)x(D-K+1)
- the kernels have MxNxKxK coefficients (which have to be learned)
- cost: M*K*K*N*(D-K+1)*(D-K+1)

Question: How many feature maps? What's the size of the filters?

Answer: Usually, there are more output feature maps than input feature maps. Convolutional layers can increase the number of hidden units by big factors (and are expensive to compute).
The size of the filters has to match the size/scale of the patterns we want to detect (task dependent).

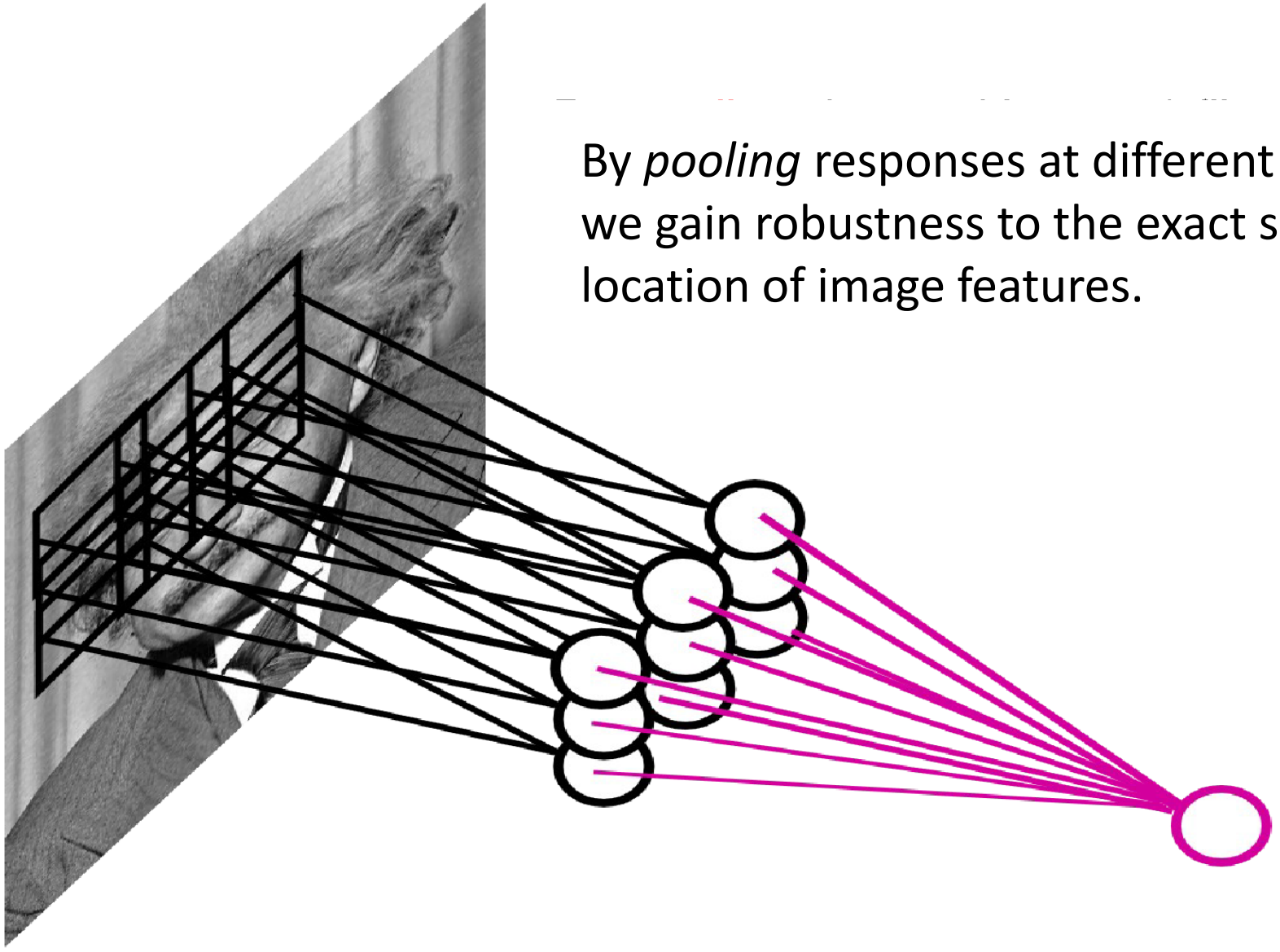
Pooling Layer



Let us assume filter is an “eye” detector.

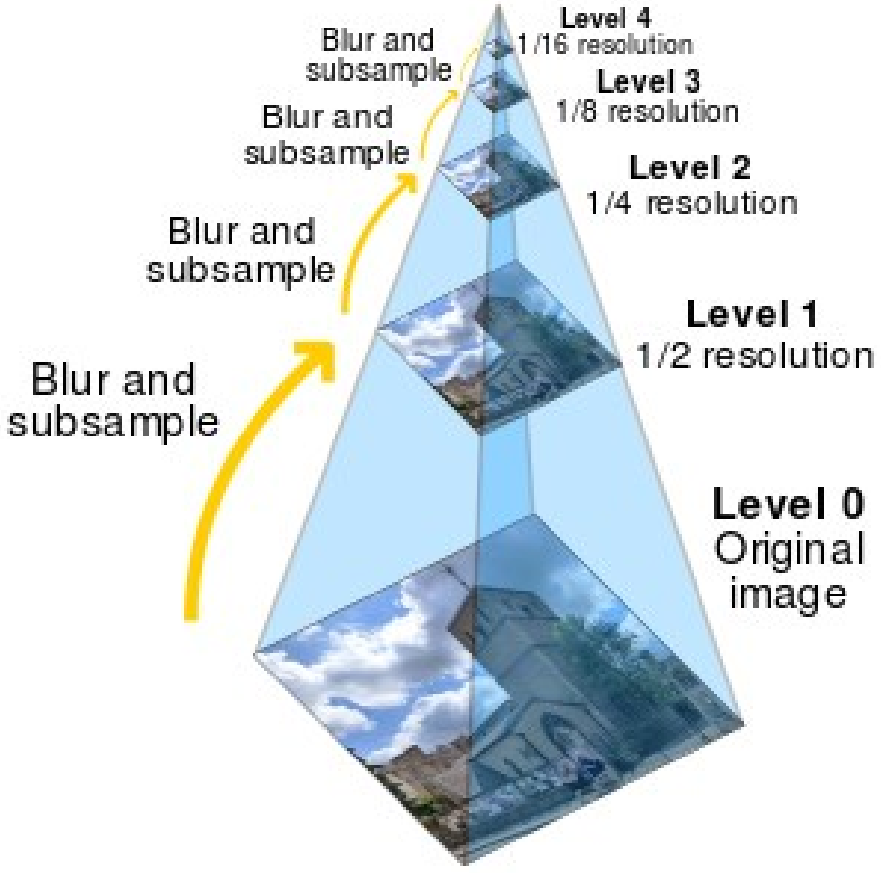
Q.: how can we make the detection robust to the exact location of the eye?

Pooling Layer



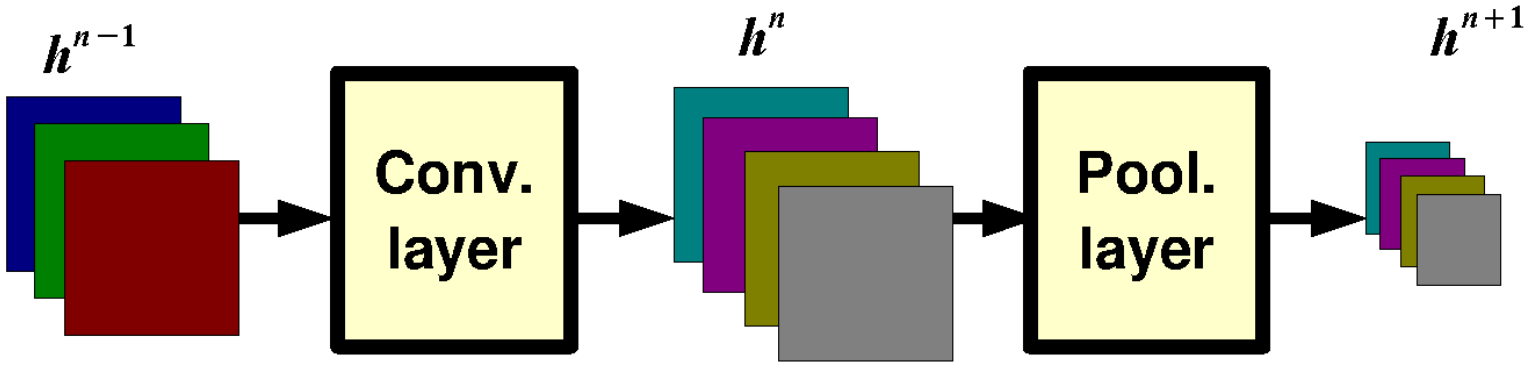
By *pooling* responses at different locations, we gain robustness to the exact spatial location of image features.

Pooling is similar to downsampling



…except sometimes we don't want to blur,
as other functions might be better for classification.

Pooling Layer: Receptive Field Size



Pooling Layer: Examples

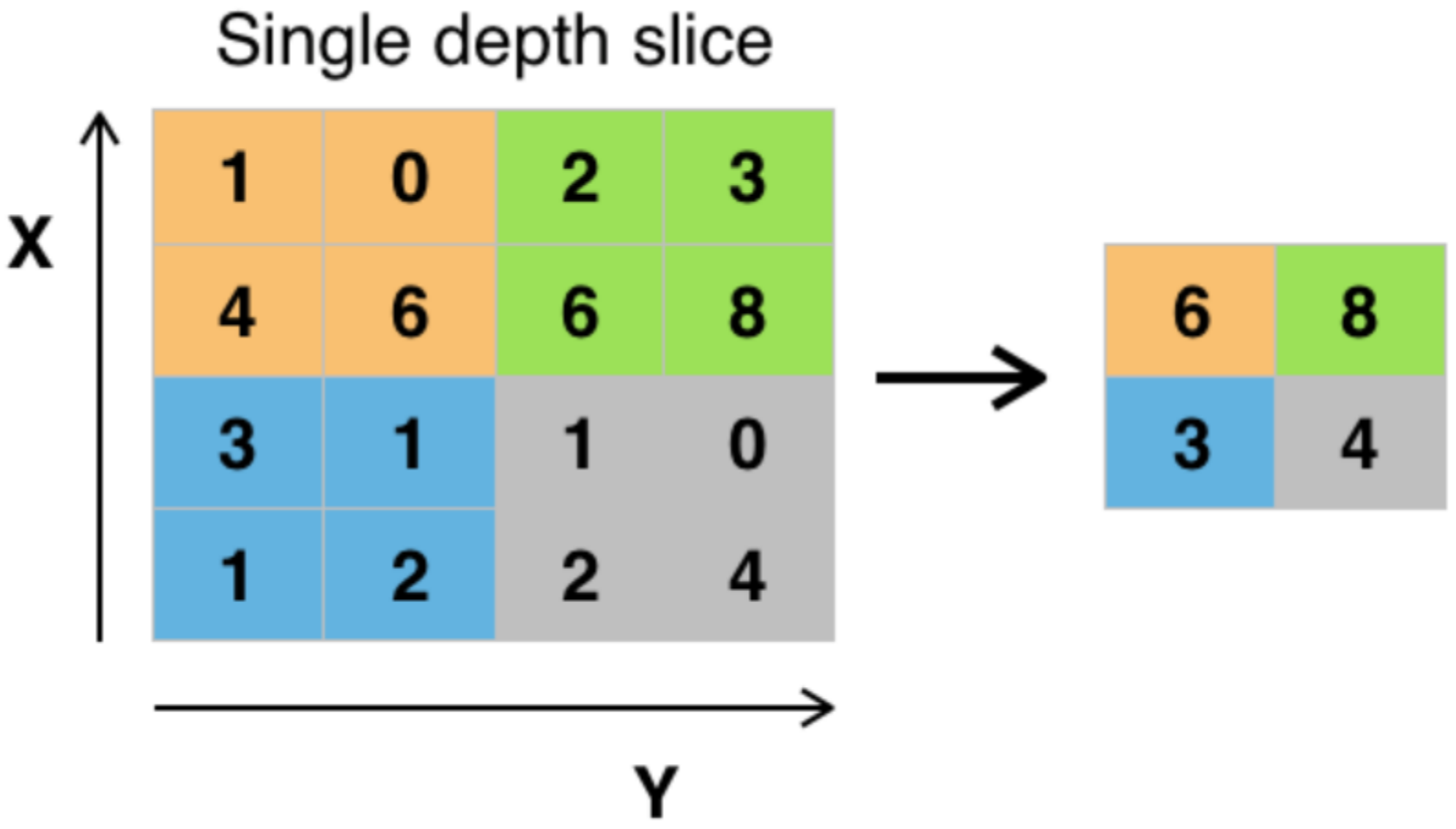
Max-pooling:

$$h_j^n(x, y) = max_{\bar{x} \in N(x), \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

Average-pooling:

$$h_j^n(x, y) = 1/K \sum_{\bar{x} \in N(x), \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

Max pooling



Wikipedia

Pooling Layer: Examples

Max-pooling:

$$h_j^n(x,y) = max_{\bar{x}\in N(x),\, \bar{y}\in N(y)} h_j^{n-1}(\bar{x},\bar{y})$$

Average-pooling:

$$h_j^n(x,y) = 1/K \sum_{\bar{x}\in N(x),\, \bar{y}\in N(y)} h_j^{n-1}(\bar{x},\bar{y})$$

L2-pooling:

$$h_j^n(x,y) = \sqrt{\sum_{\bar{x}\in N(x),\, \bar{y}\in N(y)} h_j^{n-1}(\bar{x},\bar{y})^2}$$

L2-pooling over features:

$$h_j^n(x,y) = \sqrt{\sum_{k\in N(j)} h_k^{n-1}(x,y)^2}$$

Pooling Layer

Question: What is the size of the output? What's the computational cost?

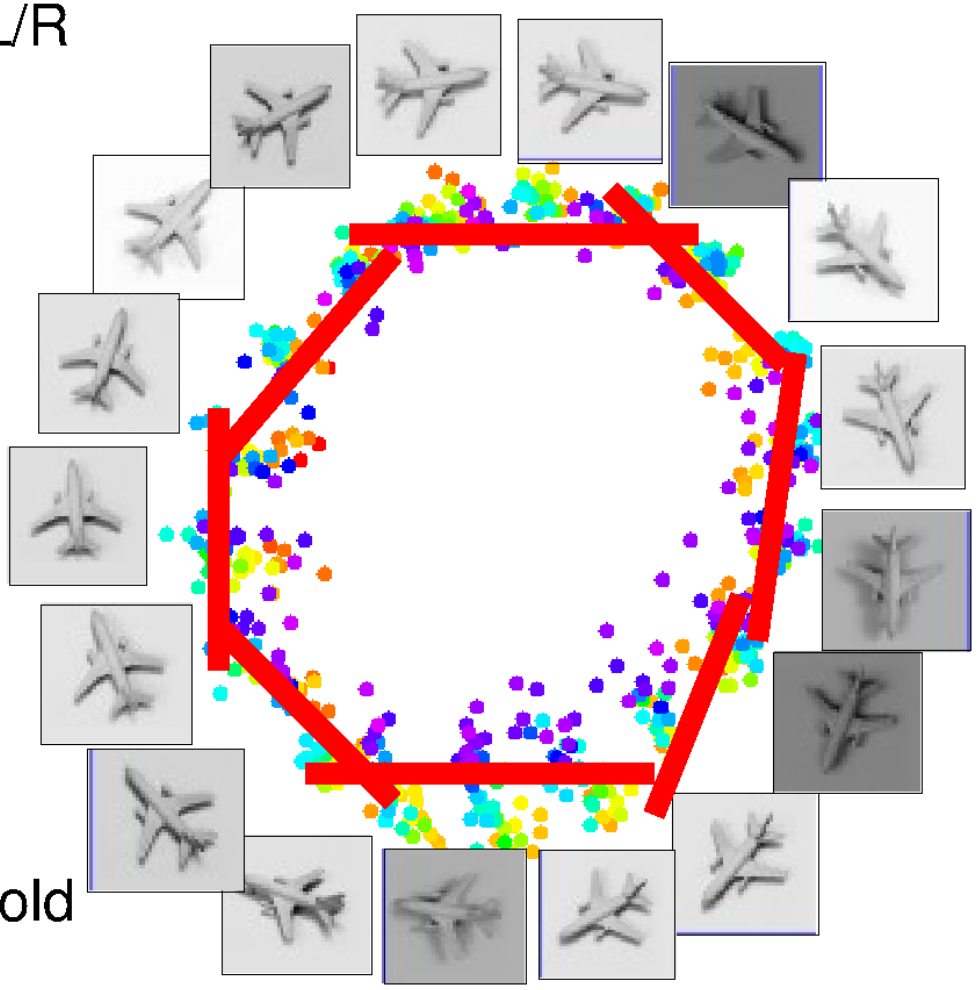
Answer: The size of the output depends on the stride between the pools. For instance, if pools do not overlap and have size KxK, and the input has size DxD with M input feature maps, then:
- output is M@(D/K)x(D/K)
- the computational cost is proportional to the size of the input (negligible compared to a convolutional layer)

Question: How should I set the size of the pools?

Answer: It depends on how much "invariant" or robust to distortions we want the representation to be. It is best to pool slowly (via a few stacks of conv-pooling layers).

Pooling Layer: Interpretation

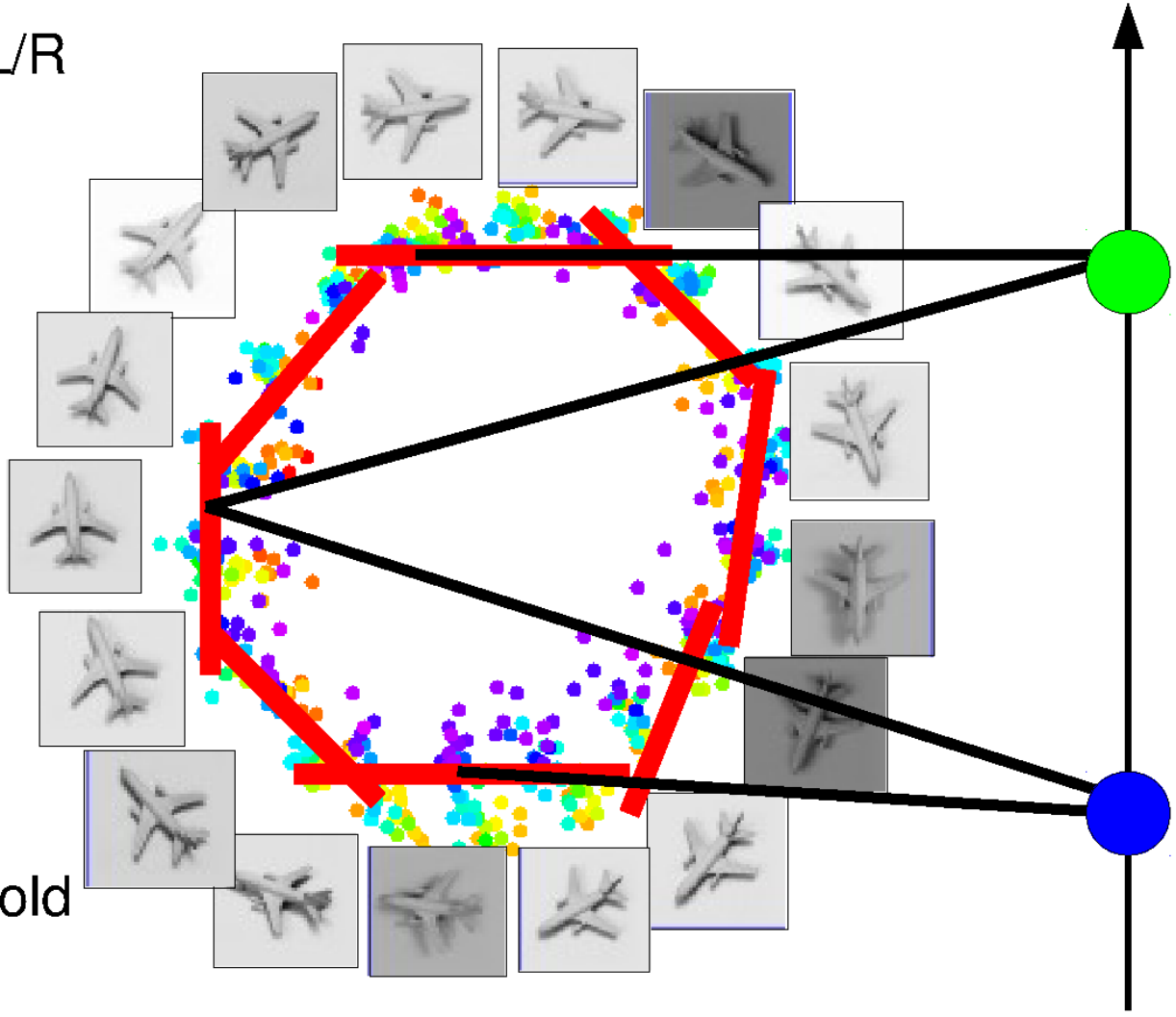
Task: detect orientation L/R



Conv layer:
linearizes manifold

Pooling Layer: Interpretation

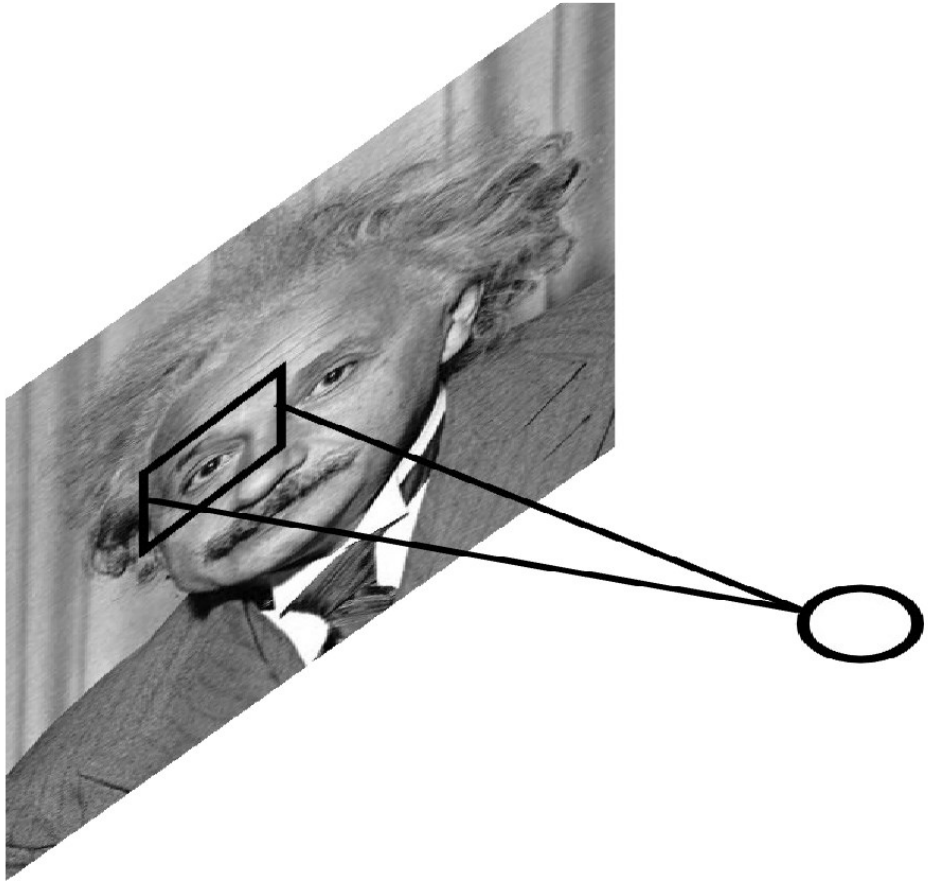
Task: detect orientation L/R



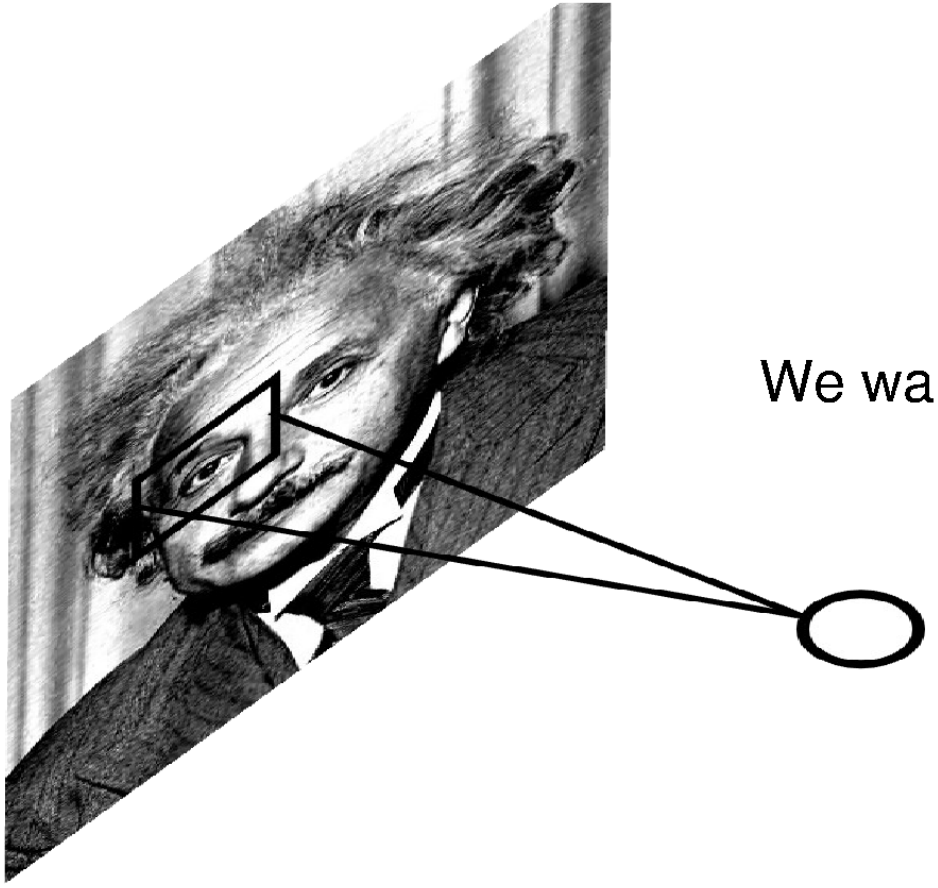
Conv layer:
linearizes manifold

Pooling layer:
collapses manifold

Local Contrast Normalization



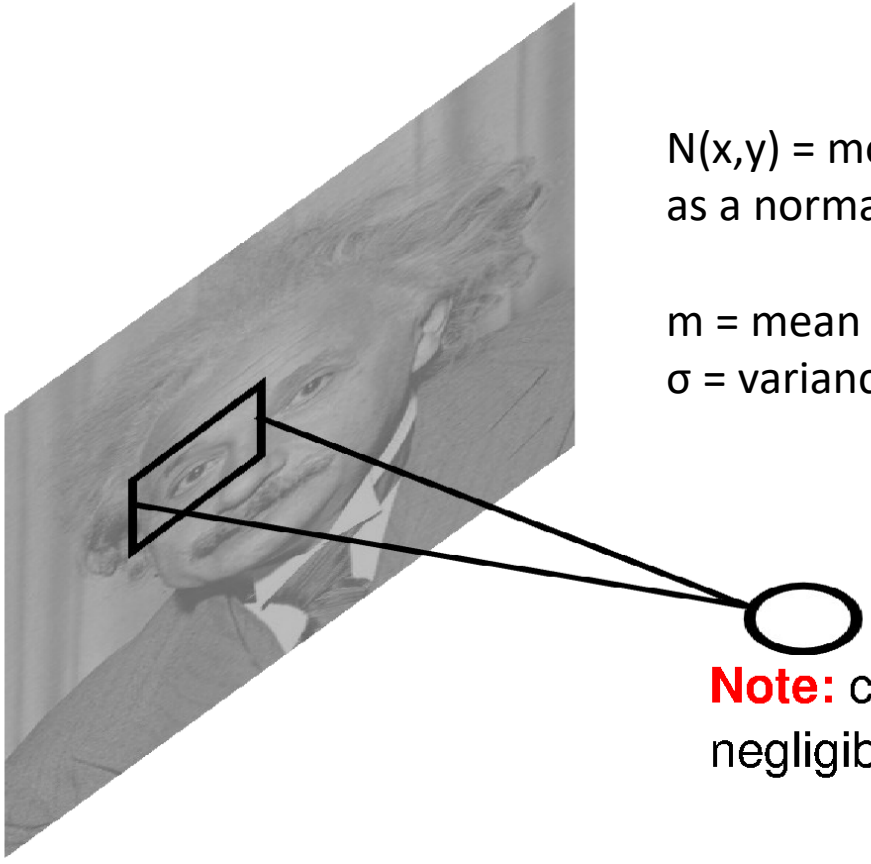
Local Contrast Normalization



We want the same response.

Local Contrast Normalization

$$h^{i+1}(x,y)=\frac{h^{i}(x,y)-m^{i}(N(x,y))}{\sigma^{i}(N(x,y))}$$



N(x,y) = model pixel values in window as a normal distribution

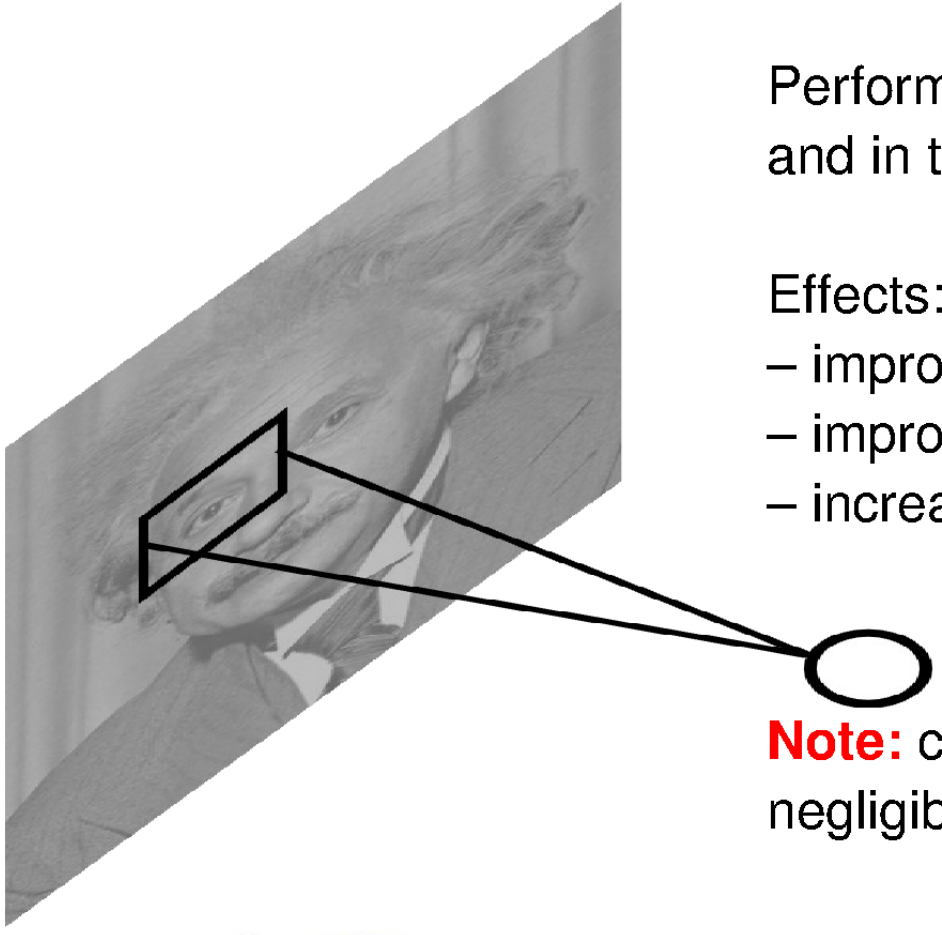
m = mean
σ = variance

Note: computational cost is negligible w.r.t. conv. layer.

Ranzato

Local Contrast Normalization

$$h^{i+1}(x,y) = \frac{h^i(x,y) - m^i(N(x,y))}{\sigma^i(N(x,y))}$$



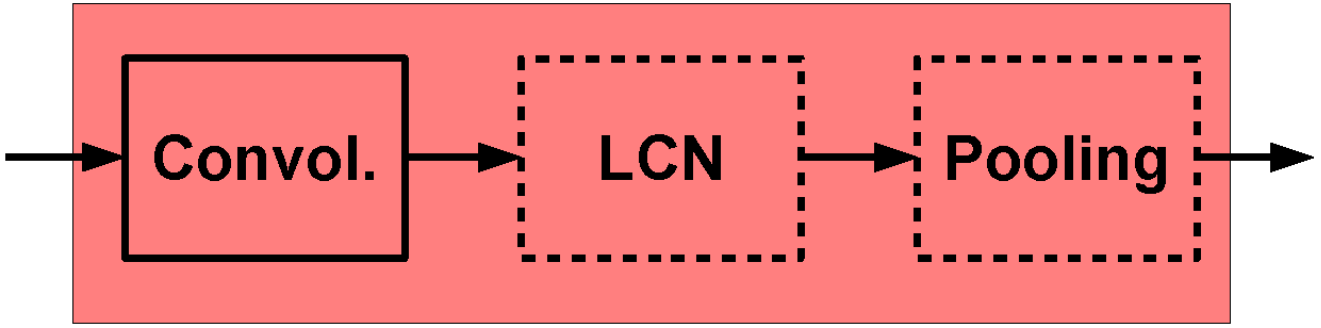
Performed also across features and in the higher layers..

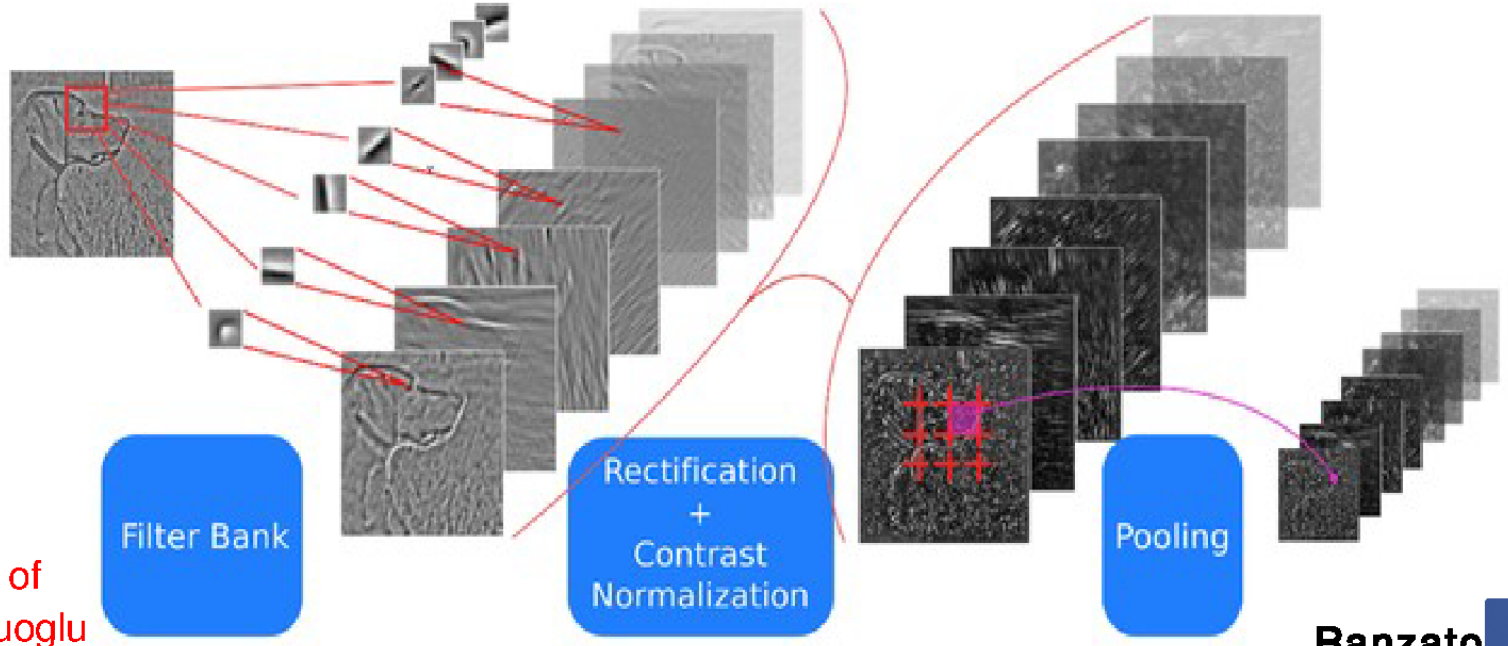
Effects:
– improves invariance
– improves optimization
– increases sparsity

Note: computational cost is negligible w.r.t. conv. layer.

ConvNets: Typical Stage

One stage (zoom)

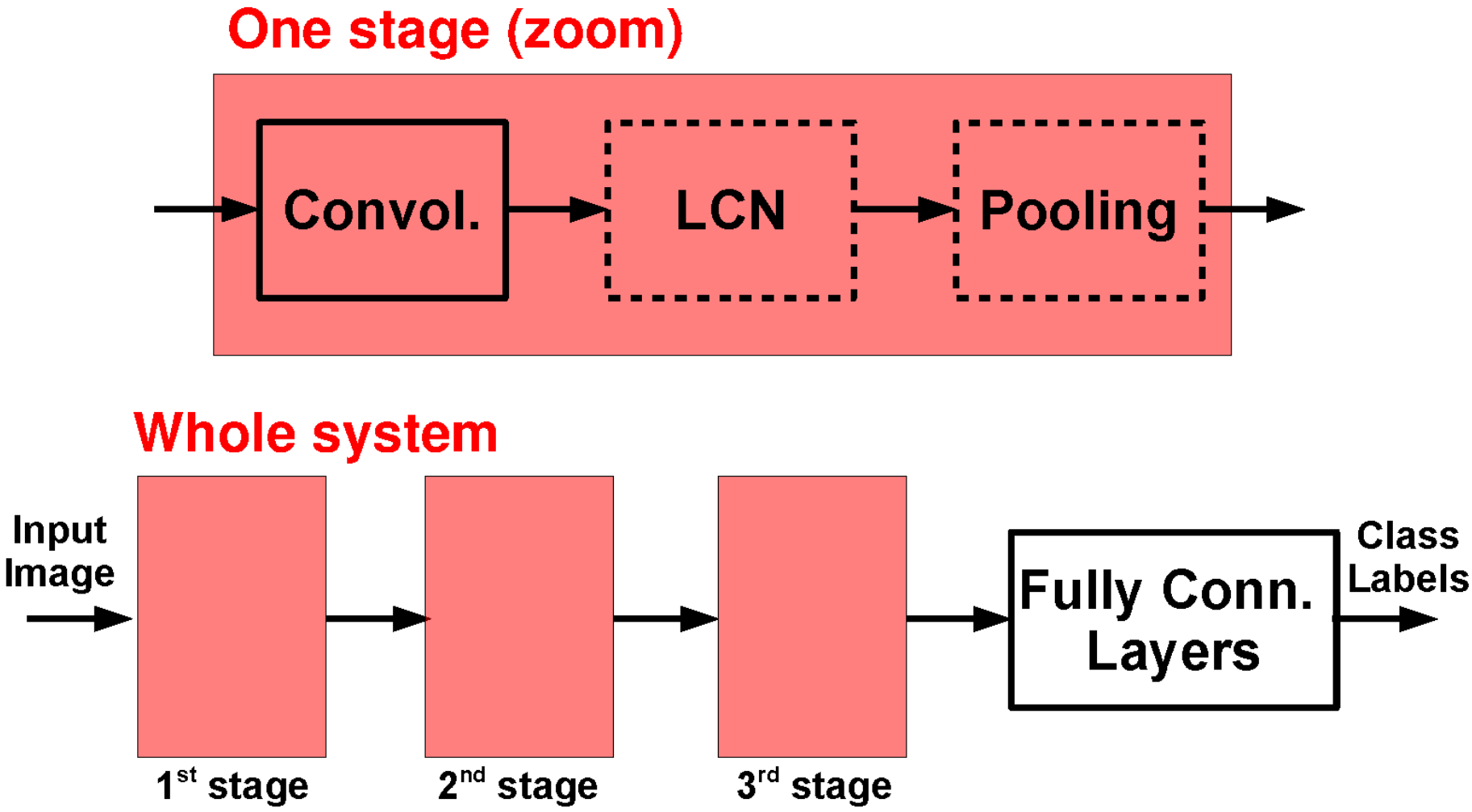




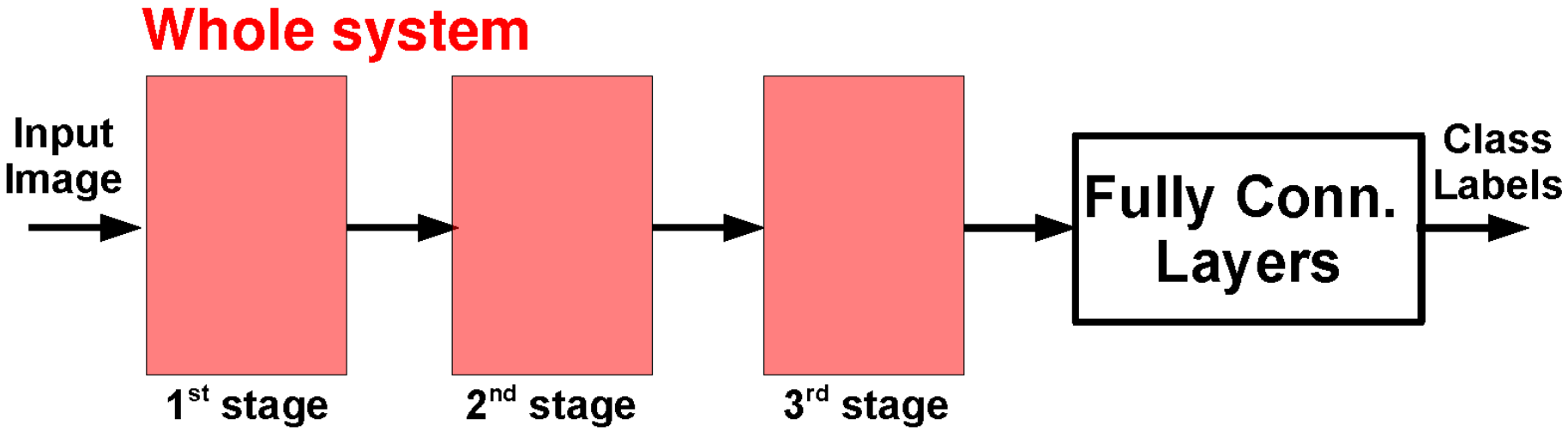
courtesy of
K. Kavukcuoglu

Ranzato

ConvNets: Typical Architecture



ConvNets: Typical Architecture



Conceptually similar to:

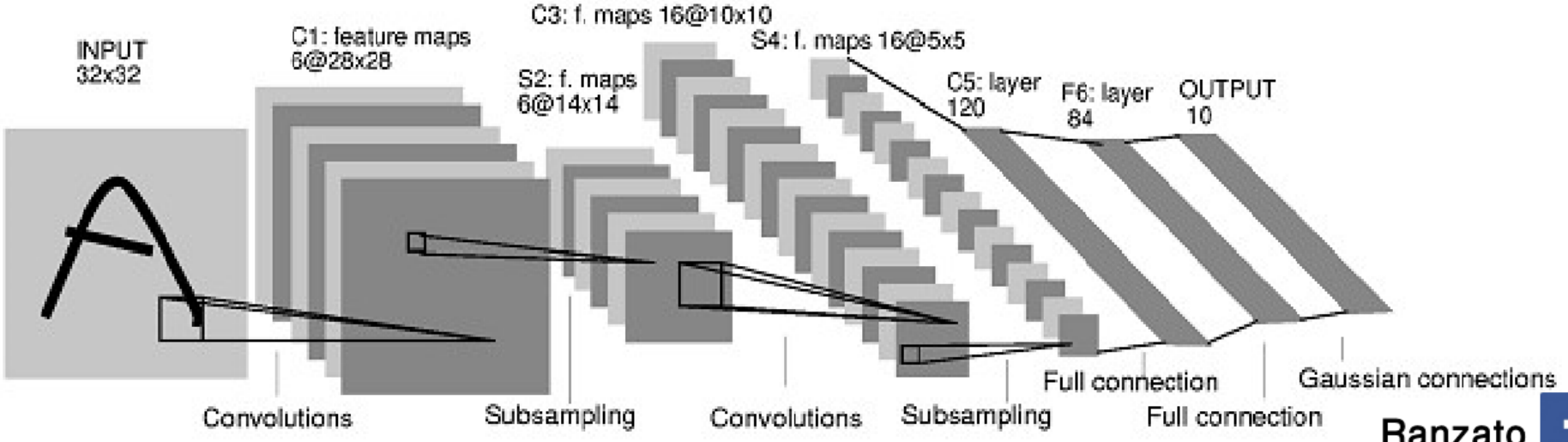
SIFT → K-Means → Pyramid Pooling → SVM

Lazebnik et al. "...Spatial Pyramid Matching..." CVPR 2006

SIFT → Fisher Vect. → Pooling → SVM

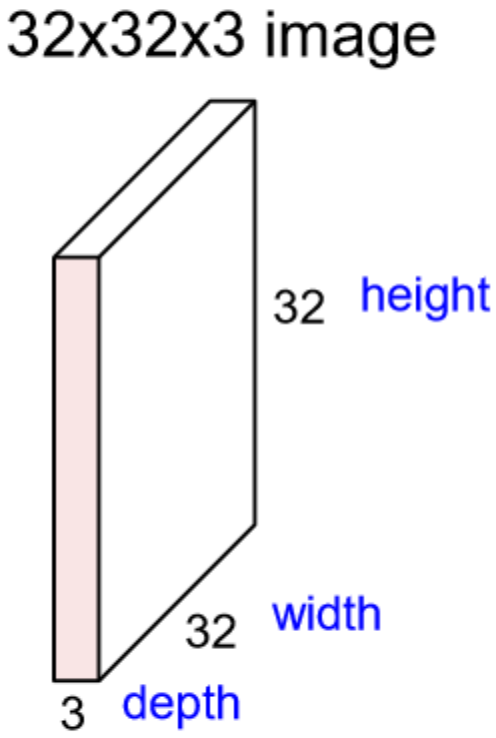
Sanchez et al. "Image classifcation with F.V.: Theory and practice" IJCV 2012

Yann LeCun's MNIST CNN architecture

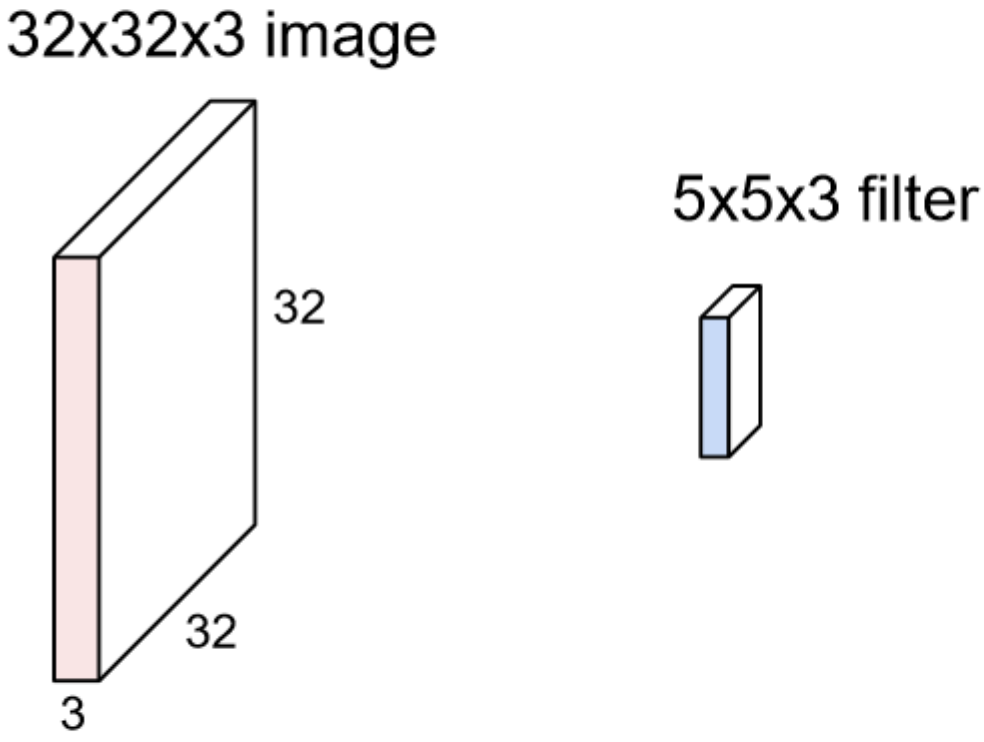


Ranzato

Convolutions: More detail

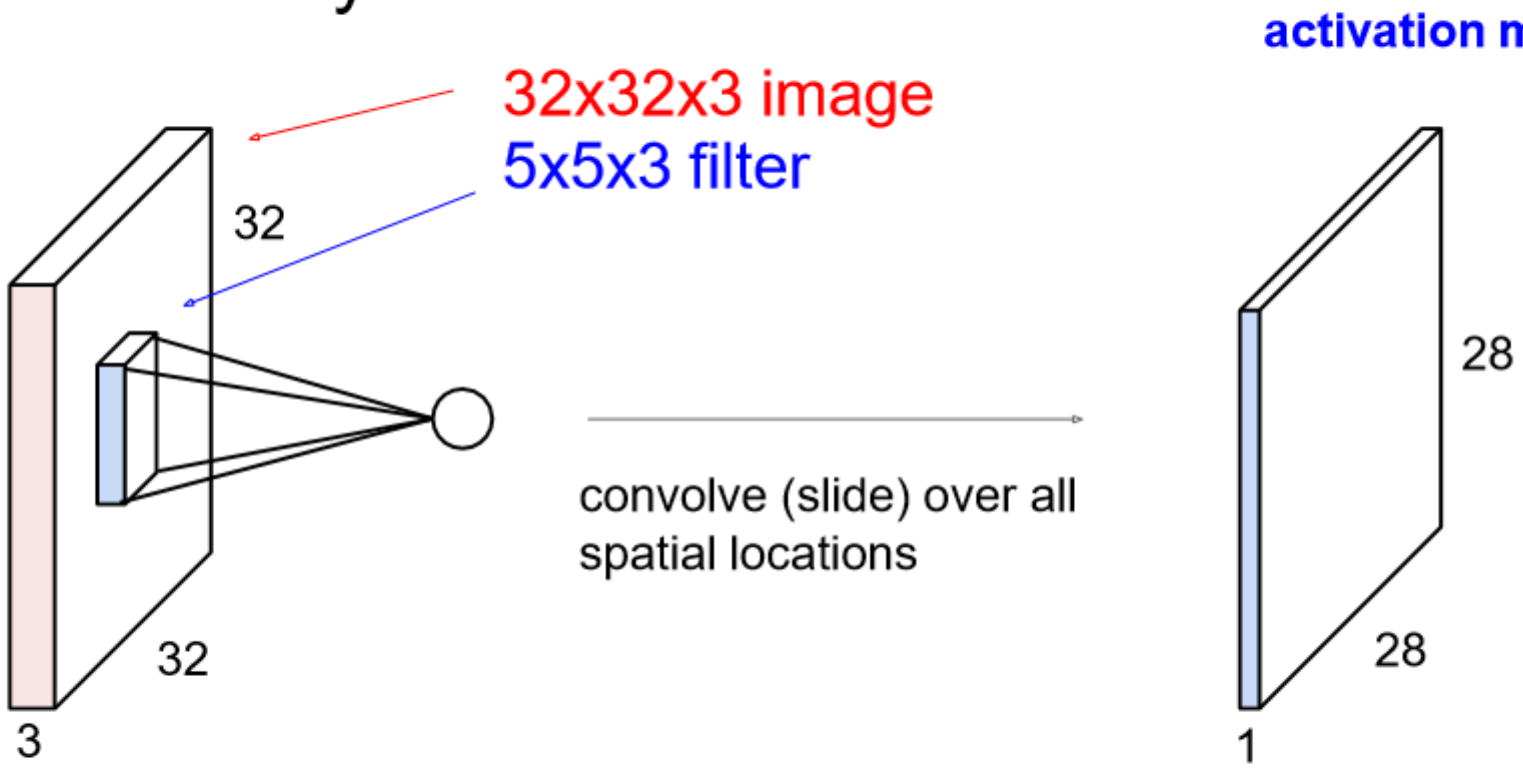


Convolutions: More detail



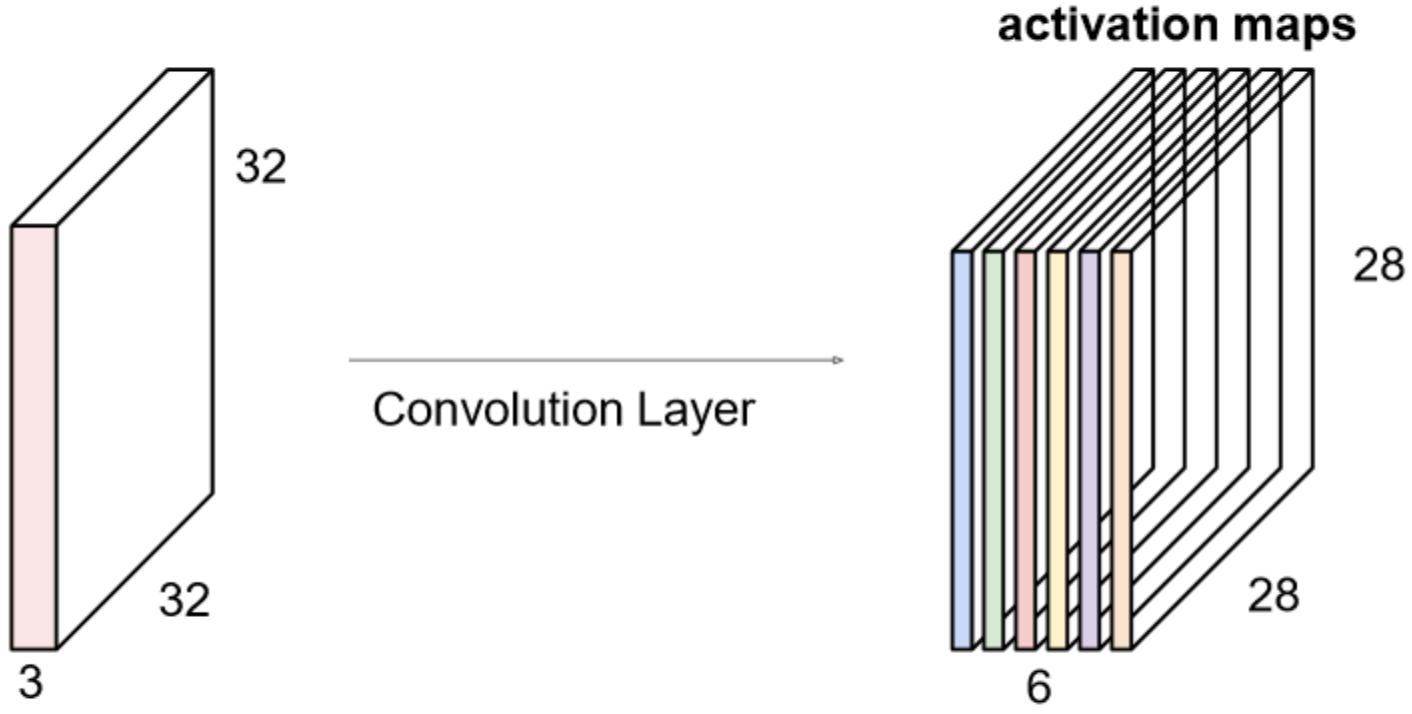
Convolutions: More detail

Convolution Layer



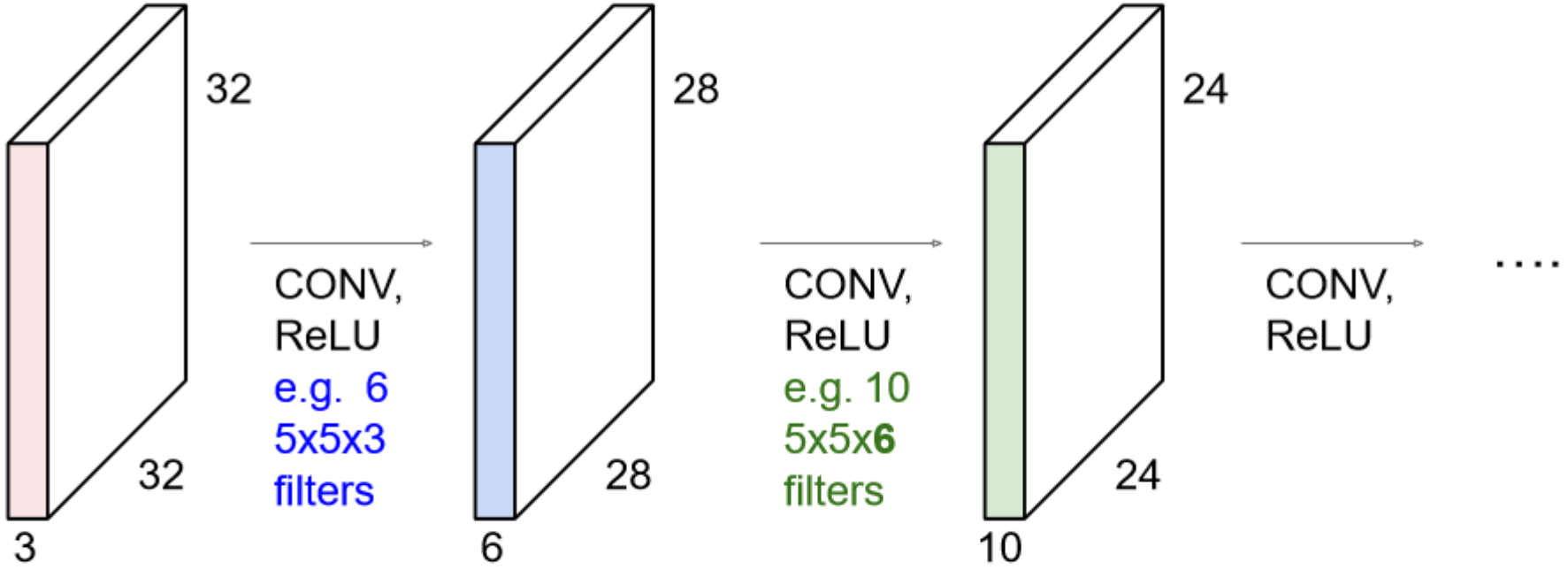
Convolutions: More detail

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:

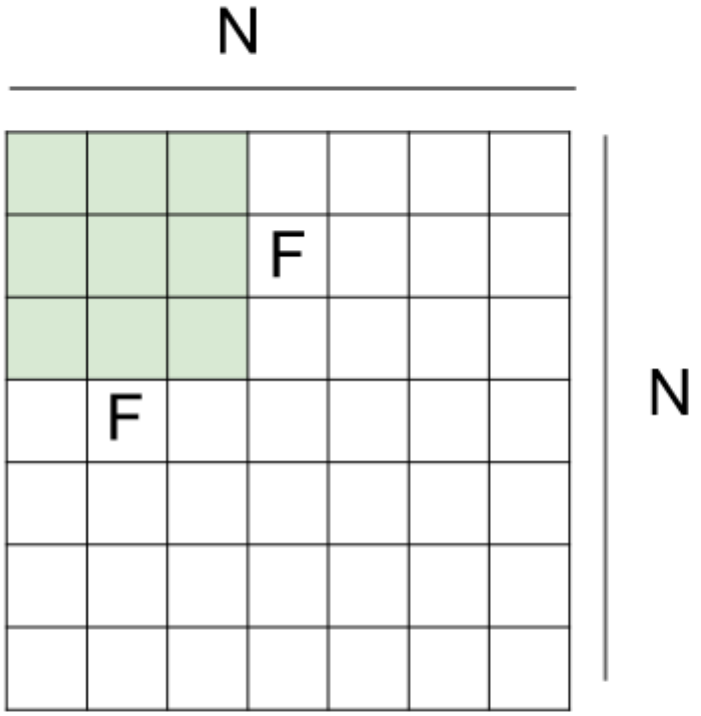


We stack these up to get a "new image" of size 28x28x6!

Convolutions: More detail



Convolutions: More detail

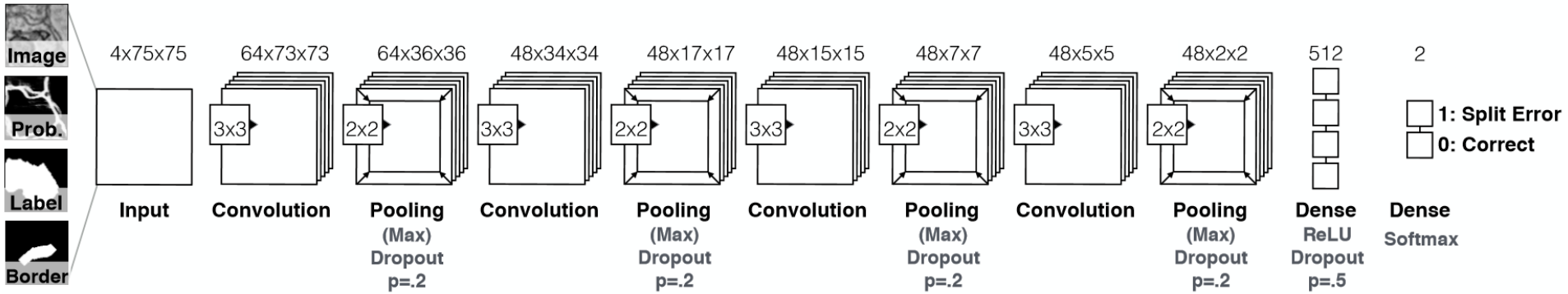


Output size:
(N - F) / stride + 1

Our connectomics diagram

Auto-generated from network declaration by *nolearn* (for Lasagne / Theano)

Input
75x75x4

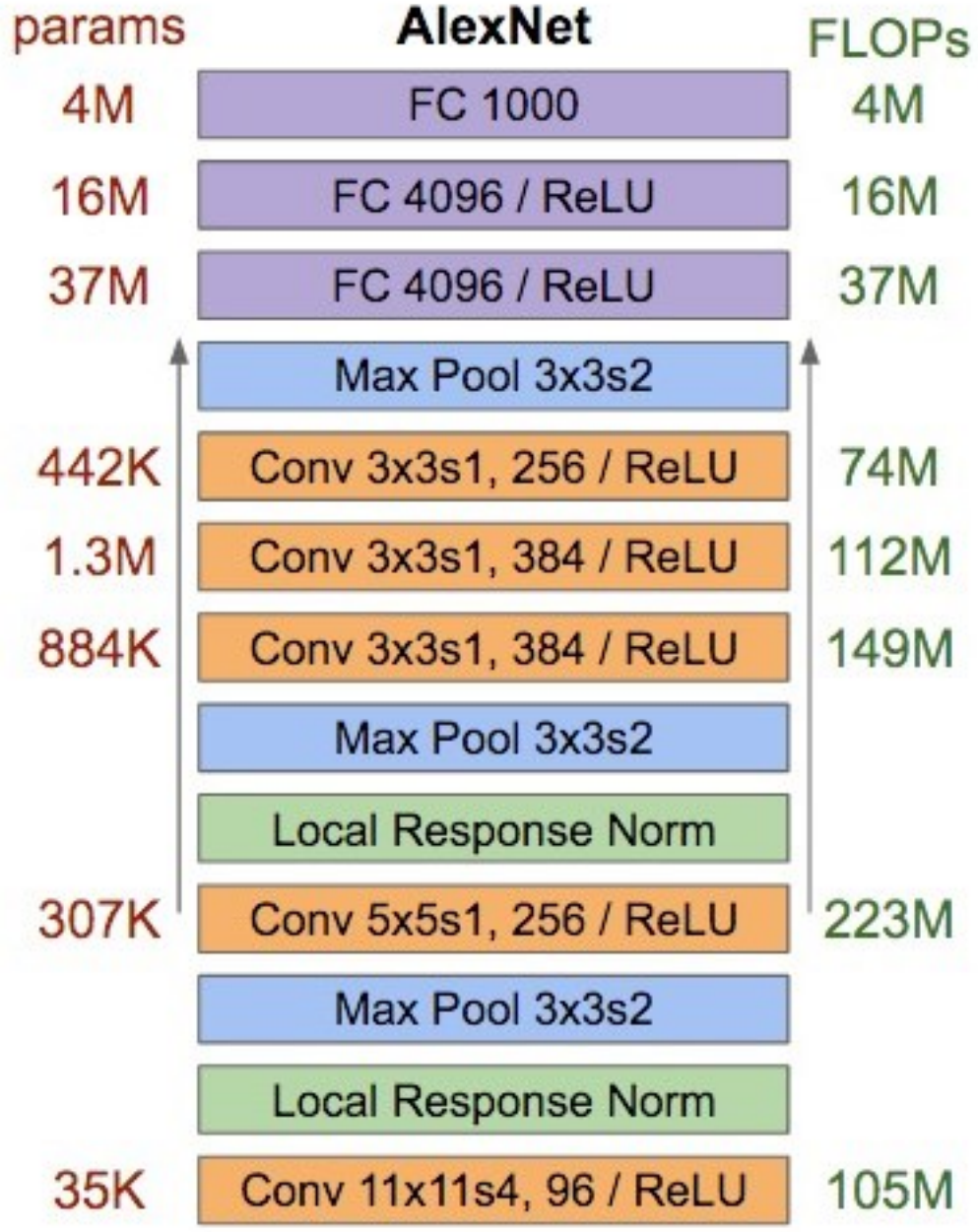


Conv 1	Conv 2	Conv 3	Conv 4
3x3x4	3x3x64	3x3x48	3x3x48
64 filters	48 filters	48 filters	48 filters

Max pooling	Max pooling	Max pooling	Max pooling
2x2 per filter	2x2 per filter	2x2 per filter	2x2 per filter

Reading architecture diagrams

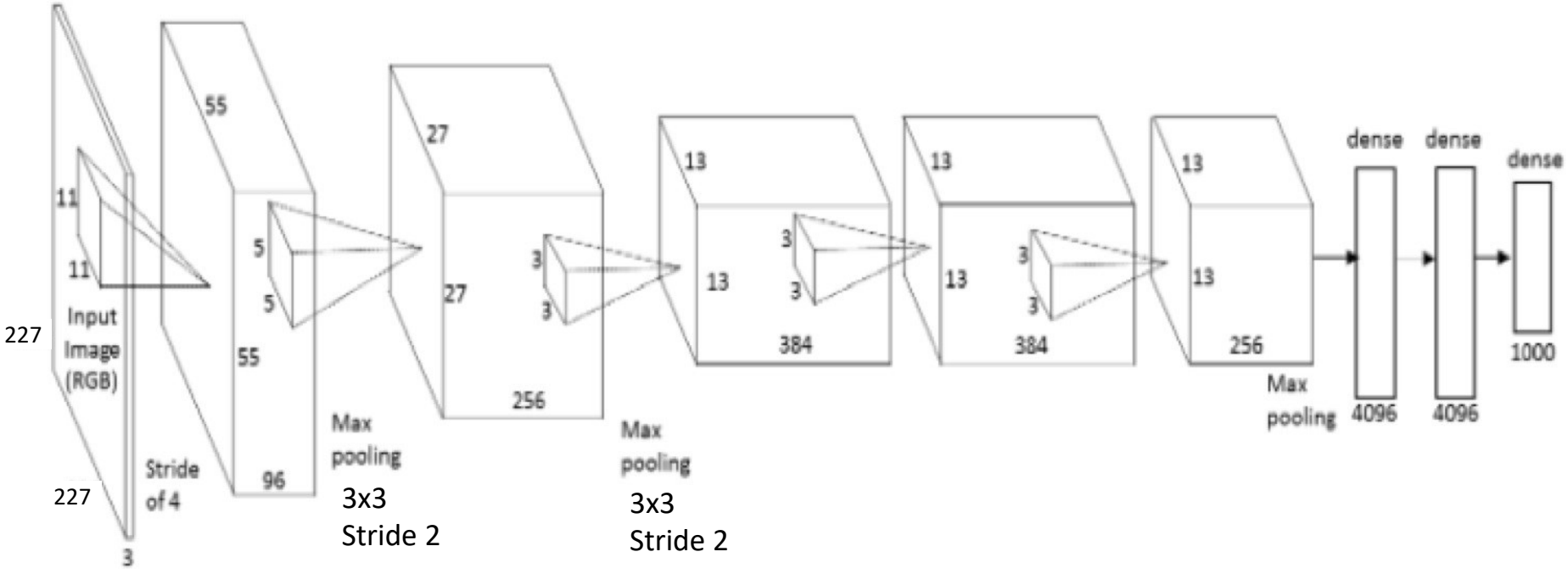
Layers
- Kernel sizes
- Strides
- # channels
- # kernels
- Max pooling



[Krizhevsky et al. 2012]

AlexNet diagram (simplified)

Input size
227 x 227 x 3



Conv 1
11 x 11 x 3
Stride 4
96 filters

Conv 2
5 x 5 x 48
Stride 1
256 filters

Conv 3
3 x 3 x 256
Stride 1
384 filters

Conv 4
3 x 3 x 192
Stride 1
384 filters

Conv 4
3 x 3 x 192
Stride 1
256 filters

Outline

- Supervised Neural Networks
- Convolutional Neural Networks
- Examples
- Tips

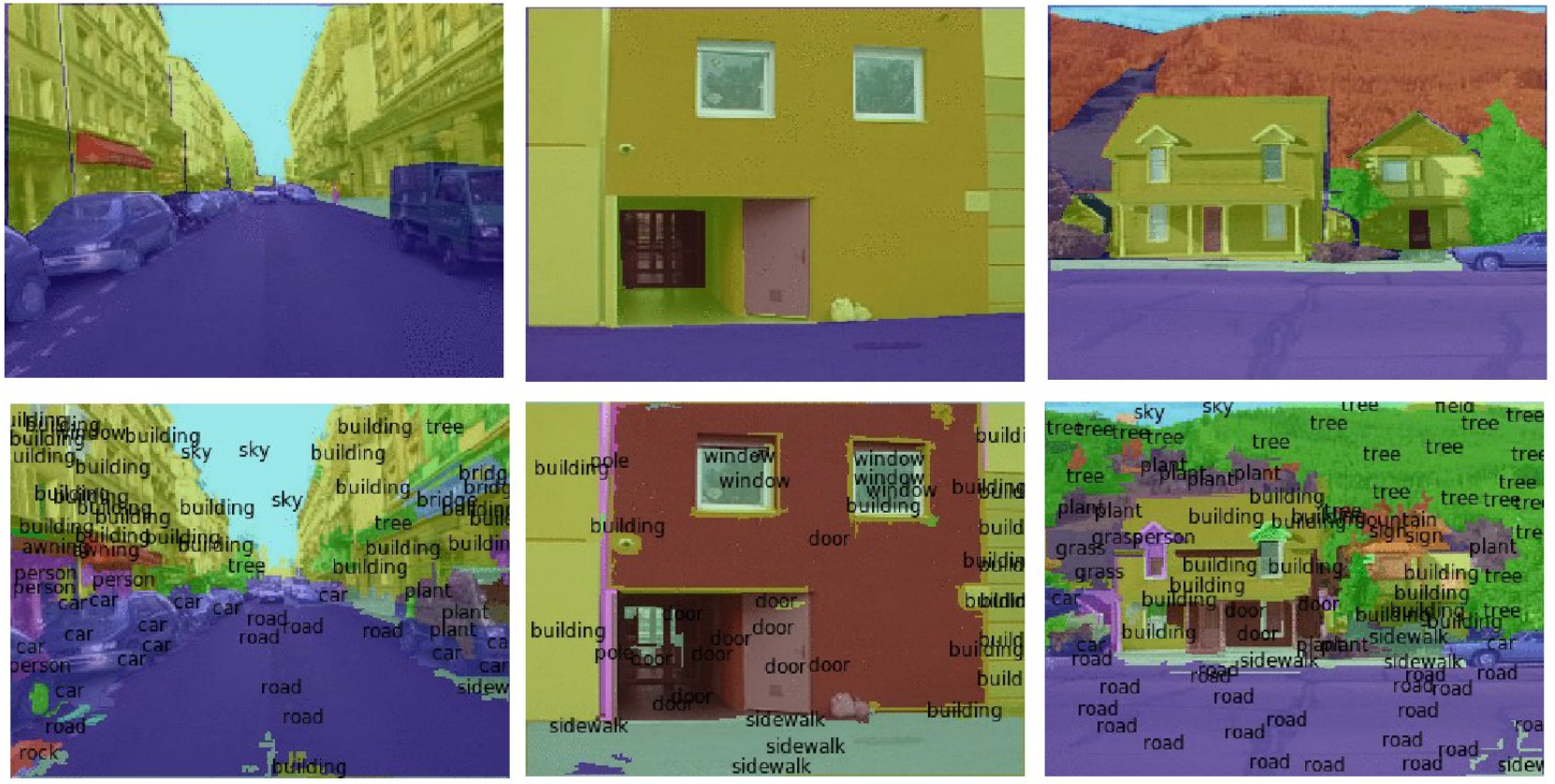
CONV NETS: EXAMPLES

- OCR / House number & Traffic sign classification

Ciresan et al. “MCDNN for image classification” CVPR 2012
Wan et al. “Regularization of neural networks using dropconnect” ICML 2013
Jaderberg et al. “Synthetic data and ANN for natural scene text recognition” arXiv 2014

CONV NETS: EXAMPLES

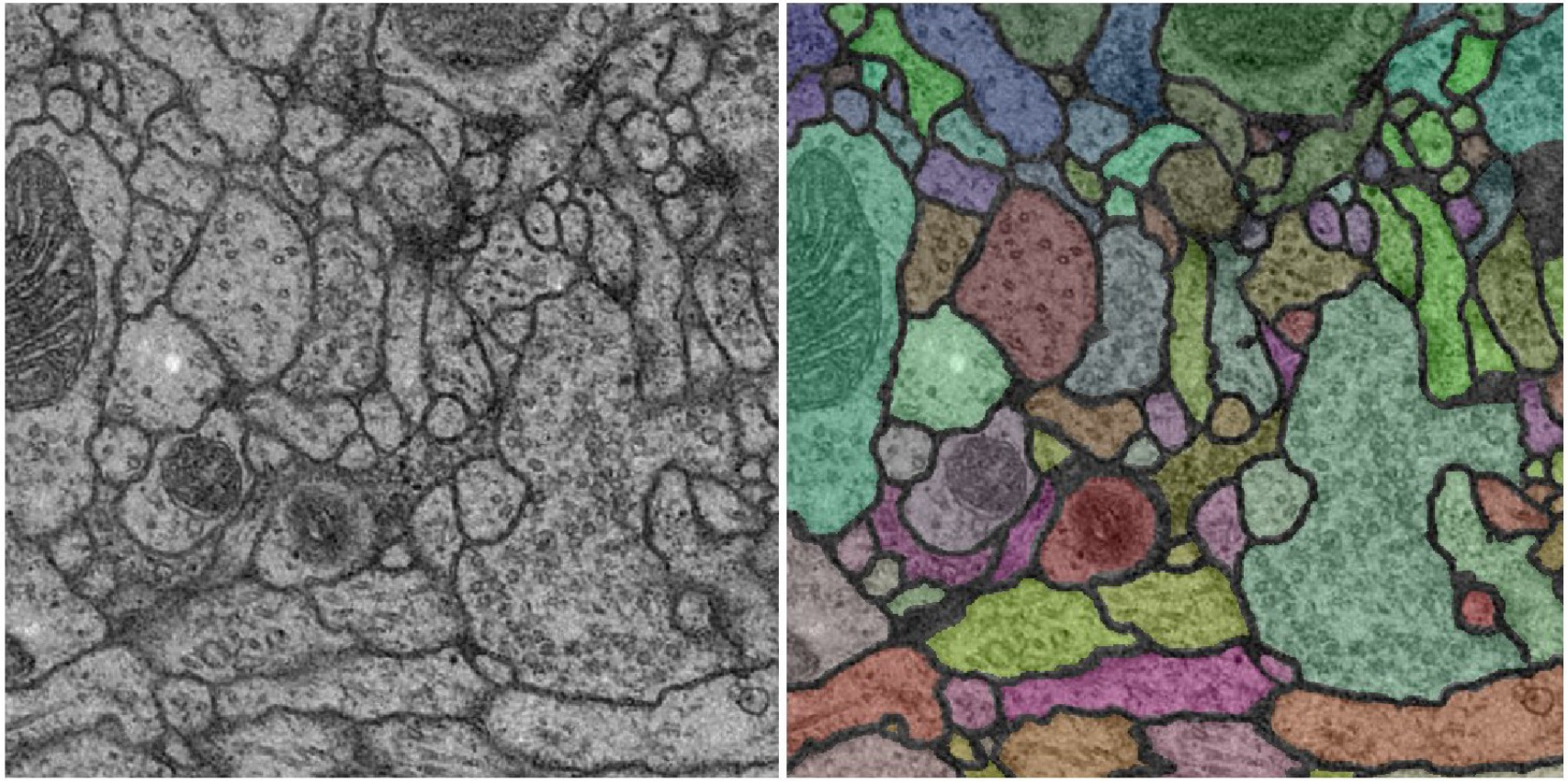
- **Scene Parsing**



Farabet et al. “Learning hierarchical features for scene labeling” PAMI 2013
Pinheiro et al. “Recurrent CNN for scene parsing” arxiv 2013

CONV NETS: EXAMPLES

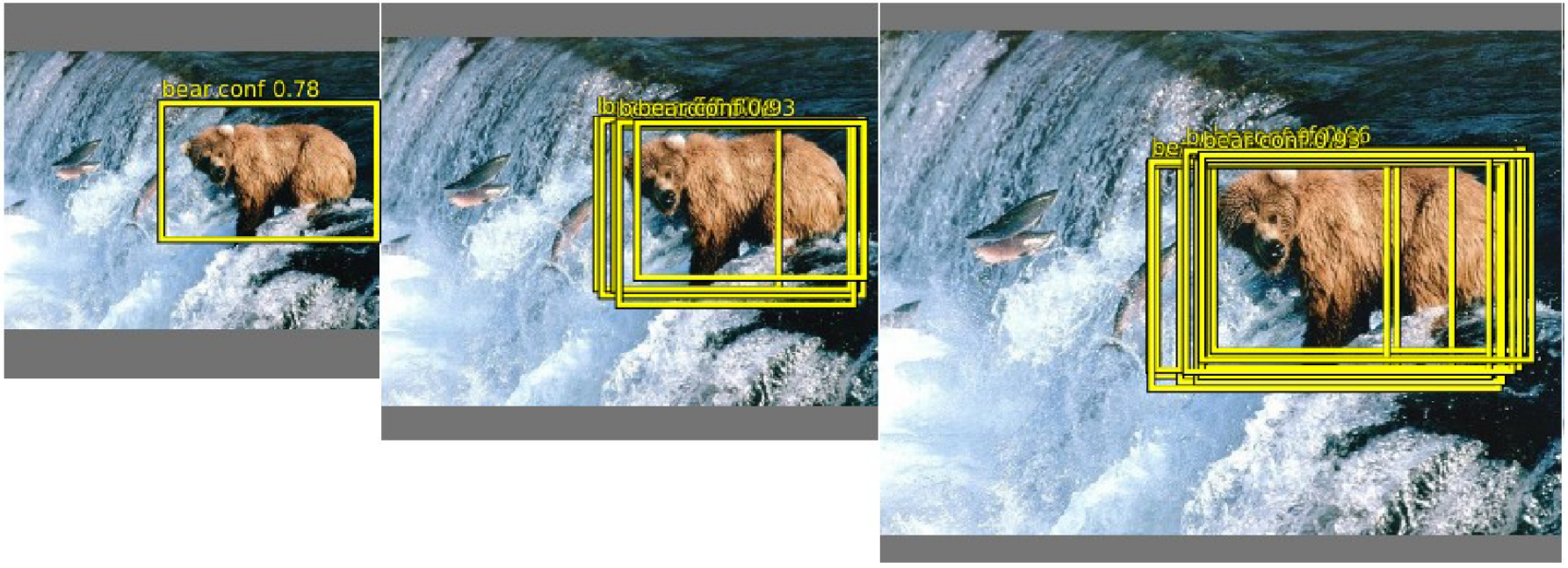
- **Segmentation 3D volumetric images**



Ciresan et al. “DNN segment neuronal membranes...” NIPS 2012
Turaga et al. “Maximin learning of image segmentation” NIPS 2009

CONV NETS: EXAMPLES

- **Object detection**

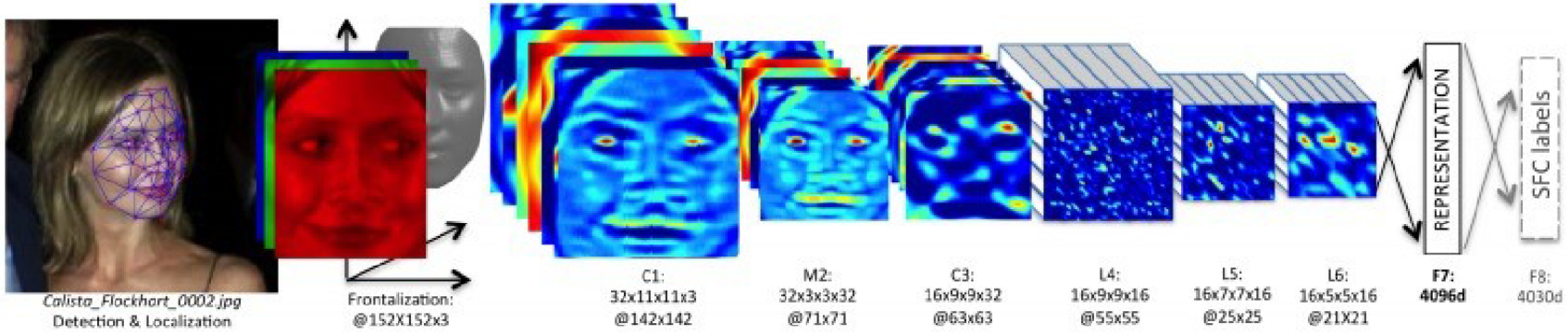


Sermanet et al. “OverFeat: Integrated recognition, localization, ...” arxiv 2013
Girshick et al. “Rich feature hierarchies for accurate object detection...” arxiv 2013
Szegedy et al. “DNN for object detection” NIPS 2013

Ranzato

CONV NETS: EXAMPLES

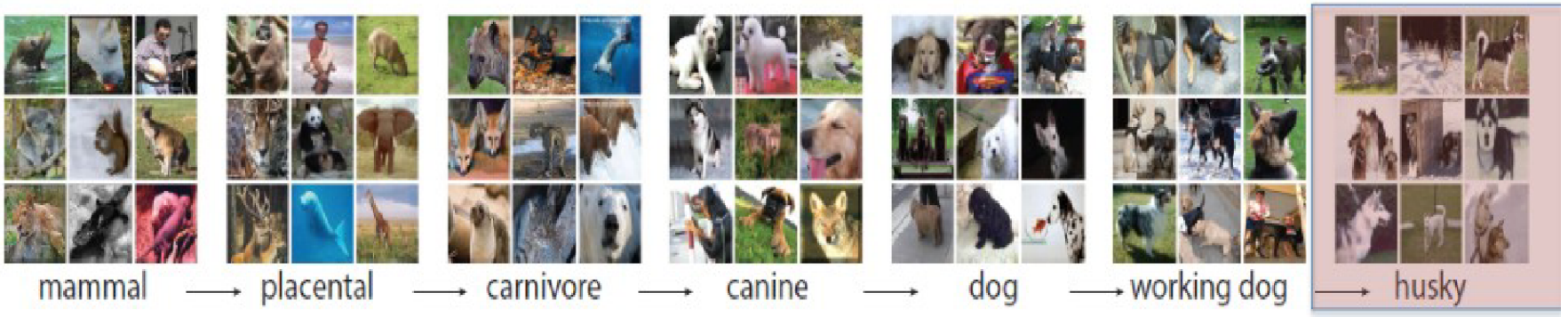
- **Face Verification & Identification**



Taigman et al. "DeepFace..." CVPR 2014

Ranzato

Dataset: ImageNet 2012



- S: (n) Eskimo dog, husky (breed of heavy-coated Arctic sled dog)
  - *direct hypernym / **inherited hypernym** / sister term*
    - S: (n) working dog (any of several breeds of usually large powerful dogs bred to work as draft animals and guard and guide dogs)
      - S: (n) dog, domestic dog, Canis familiaris (a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds) *"the dog barked all night"*
        - S: (n) canine, canid (any of various fissiped mammals with nonretractile claws and typically long muzzles)
          - S: (n) carnivore (a terrestrial or aquatic flesh-eating mammal) *"terrestrial carnivores have four or five clawed digits on each limb"*
            - S: (n) placental, placental mammal, eutherian, eutherian mammal (mammals having a placenta; all mammals except monotremes and marsupials)
              - S: (n) mammal, mammalian (any warm-blooded vertebrate having the skin more or less covered with hair; young are born alive except for the small subclass of monotremes and nourished with milk)
                - S: (n) vertebrate, craniate (animals having a bony or cartilaginous skeleton with a segmented spinal column and a large brain enclosed in a skull or cranium)
                  - S: (n) chordate (any animal of the phylum Chordata having a notochord or spinal column)
                    - S: (n) animal, animate being, beast, brute, creature, fauna (a living organism characterized by voluntary movement)
                      - S: (n) organism, being (a living thing that has (or can develop) the ability to act or function independently)
                        - S: (n) living thing, animate thing (a living (or once living) entity)
                          - S: (n) whole, unit (an assemblage of parts that is regarded as a single entity) *"how big is that part compared to the whole?"; "the team is a unit"*
                            - S: (n) object, physical object (a tangible and visible entity; an entity that can cast a shadow) *"it was full of rackets, balls and other objects"*
                              - S: (n) physical entity (an entity that has physical existence)
                                - S: (n) entity (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

Deng et al. “Imagenet: a large scale hierarchical image database” CVPR 2009

mite

mite
black widow
cockroach
tick
starfish

container ship

container ship
lifeboat
amphibian
fireboat
drilling platform

motor scooter

motor scooter
go-kart
moped
bumper car
golfcart

leopard

leopard
jaguar
cheetah
snow leopard
Egyptian cat

grille

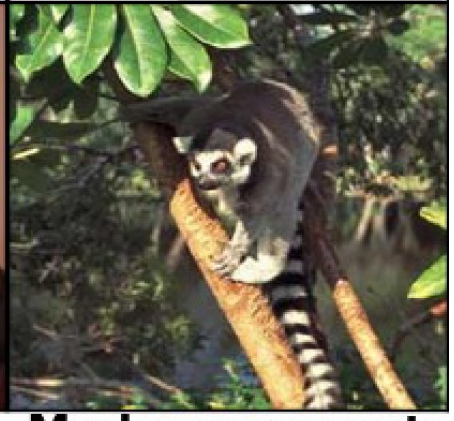
convertible
grille
pickup
beach wagon
fire engine

mushroom

agaric
mushroom
jelly fungus
gill fungus
dead-man's-fingers

cherry

dalmatian
grape
elderberry
ffordshire bullterrier
currant



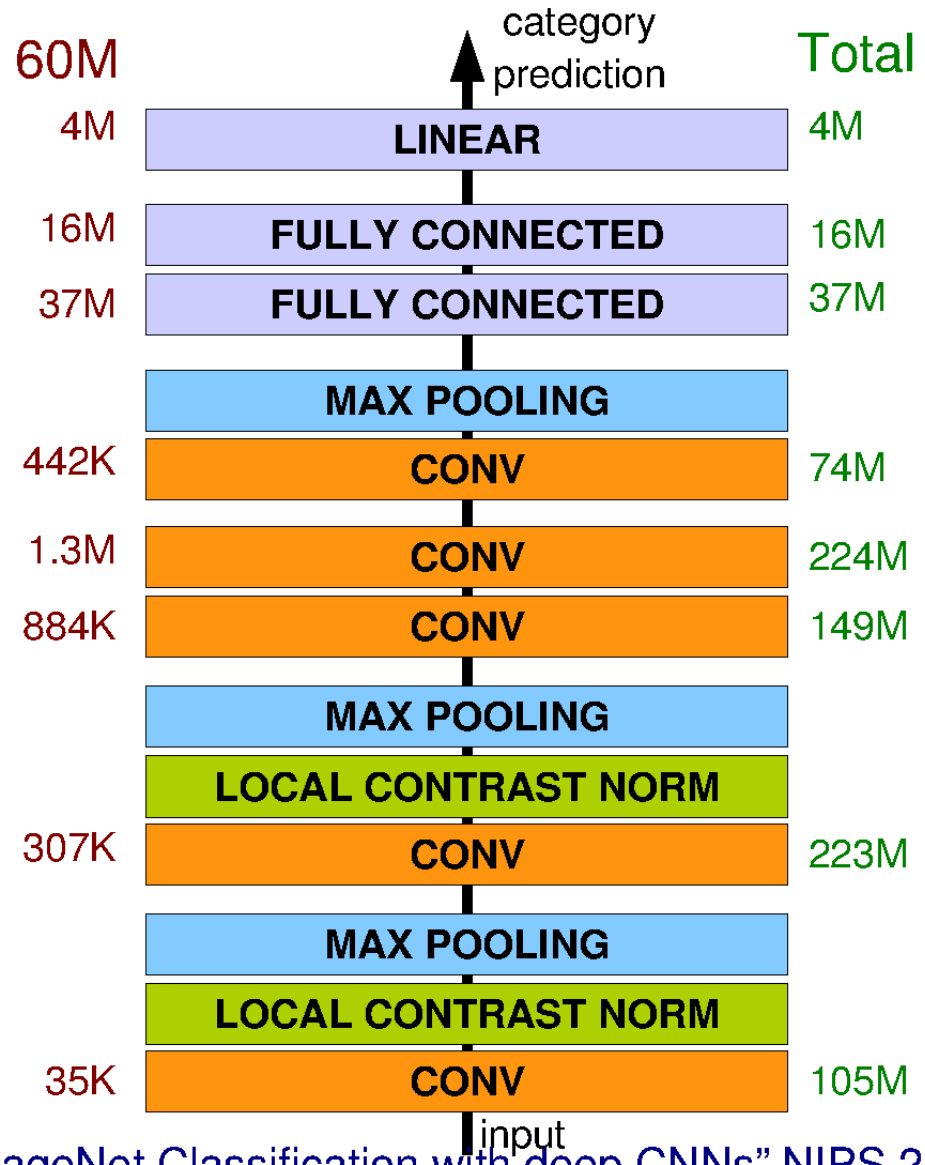
Madagascar cat

squirrel monkey
spider monkey
titi
indri
howler monkey

Architecture for Classification

Total nr. params: 60M

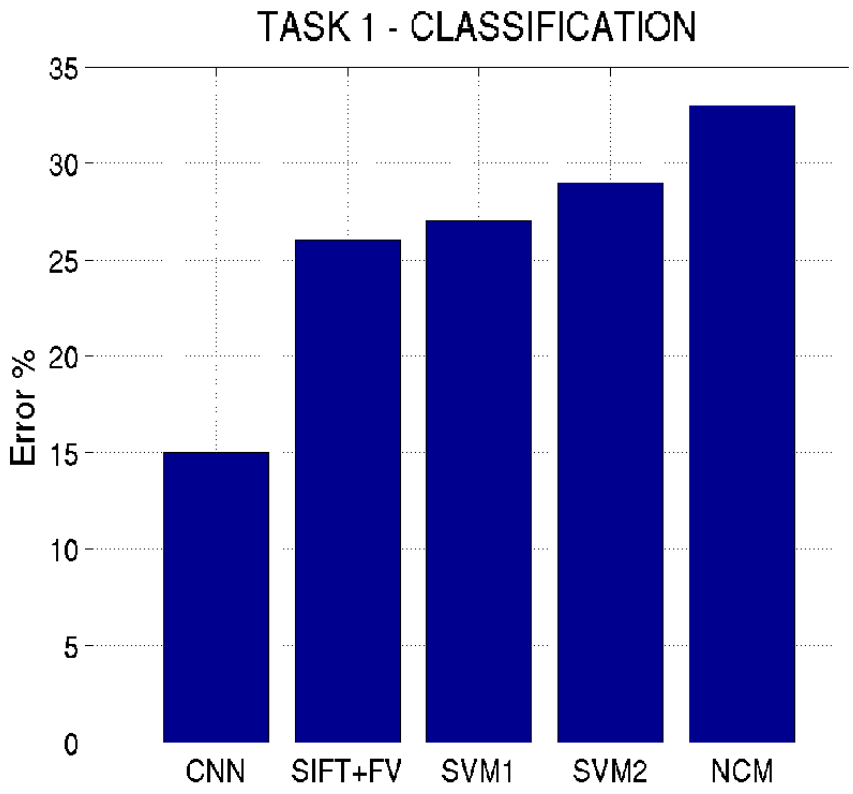
Total nr. flops: 832M

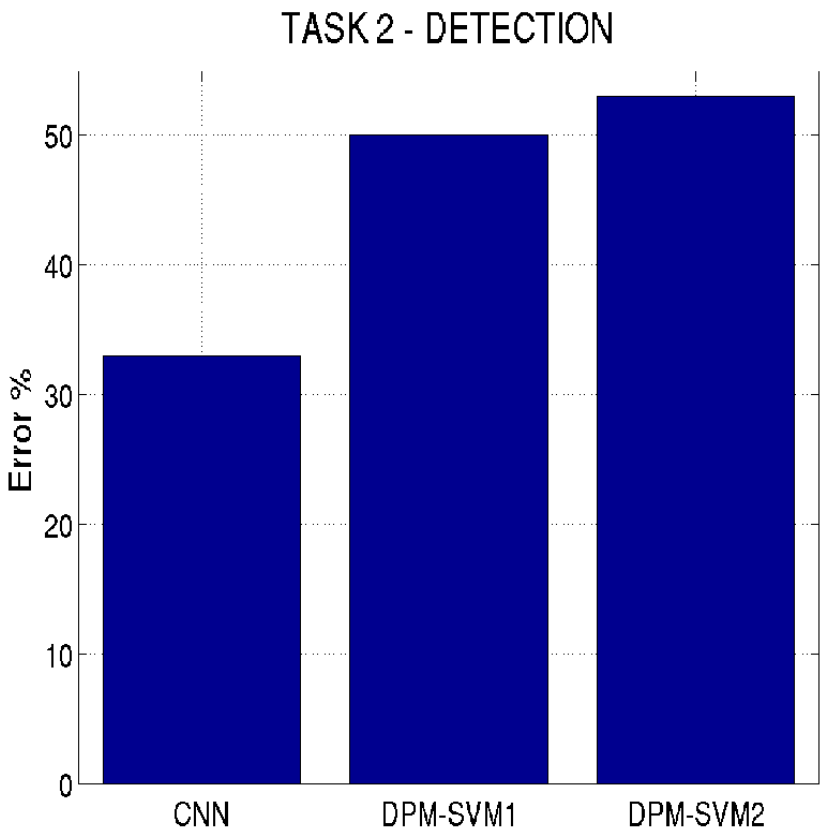


Krizhevsky et al. "ImageNet Classification with deep CNNs" NIPS 2012

Ranzato

Results: ILSVRC 2012





Krizhevsky et al. “ImageNet Classification with deep CNNs” NIPS 2012