上海交通大学

计算机视觉

教师：赵旭

班级：AI4701

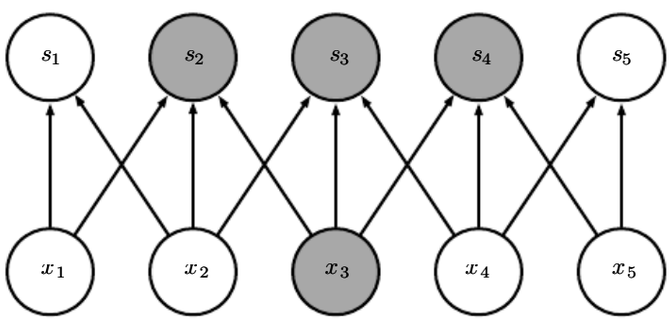
2024 春

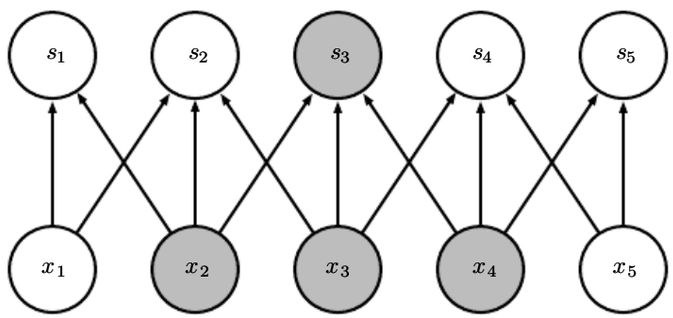
14. 表示学习： 卷积神经网络

Contents

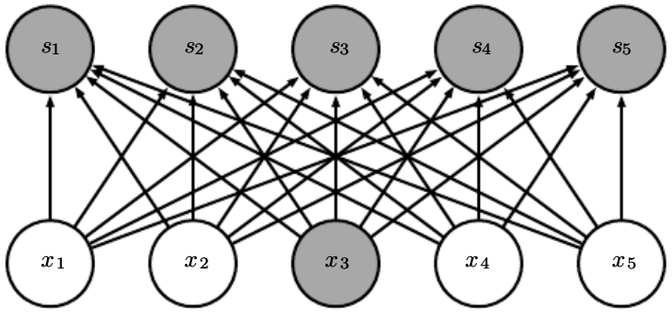
- **Motivations**
- **Structure and important components**
- **Network Architecture**

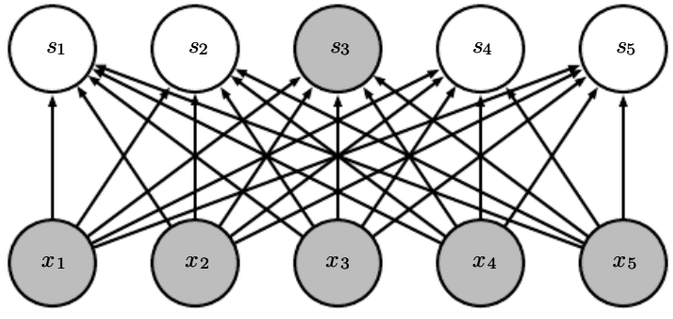
Motivations - Sparse interactions





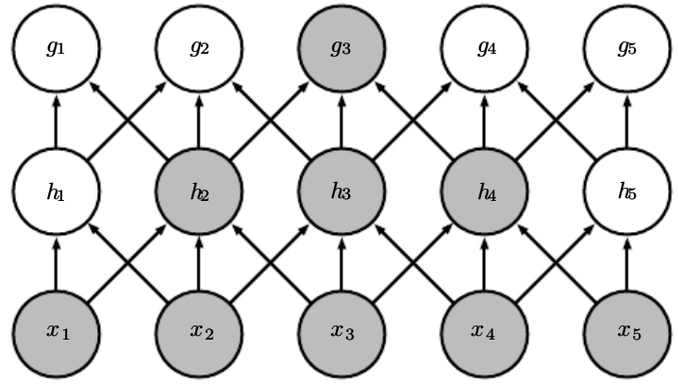
CNN





Full Connected Network

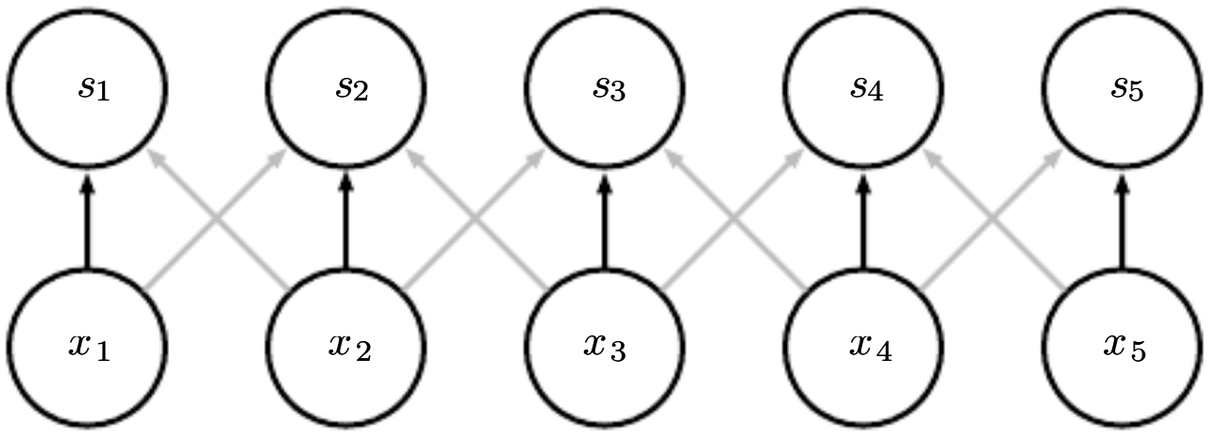
Receptive field



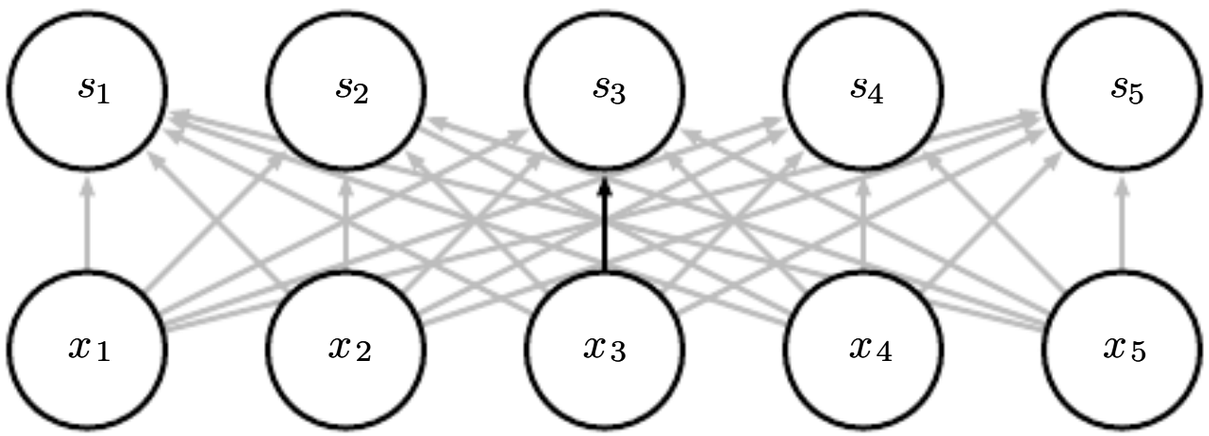
Motivations - Sparse interactions

- 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.

Motivations - Parameter sharing



In CNN, each member of the kernel is used at every position of the input.



In fully connected net, each element of the weight matrix is used exactly once when computing the output of a layer.

Motivations - Equivariance

- Convolution's property: equivariance to translation.
- To say a function is equivariant means that if the input changes, the output changes in the same way.
- For example, when processing images, it is useful to detect edges in the first layer of a convolutional network. The same edges appear more or less everywhere in the image, so it is practical to share parameters across the entire image.
- Convolution is not naturally equivariant to some other transformations, such as changes in the scale or rotation of an image.

CNN - Basic Structure

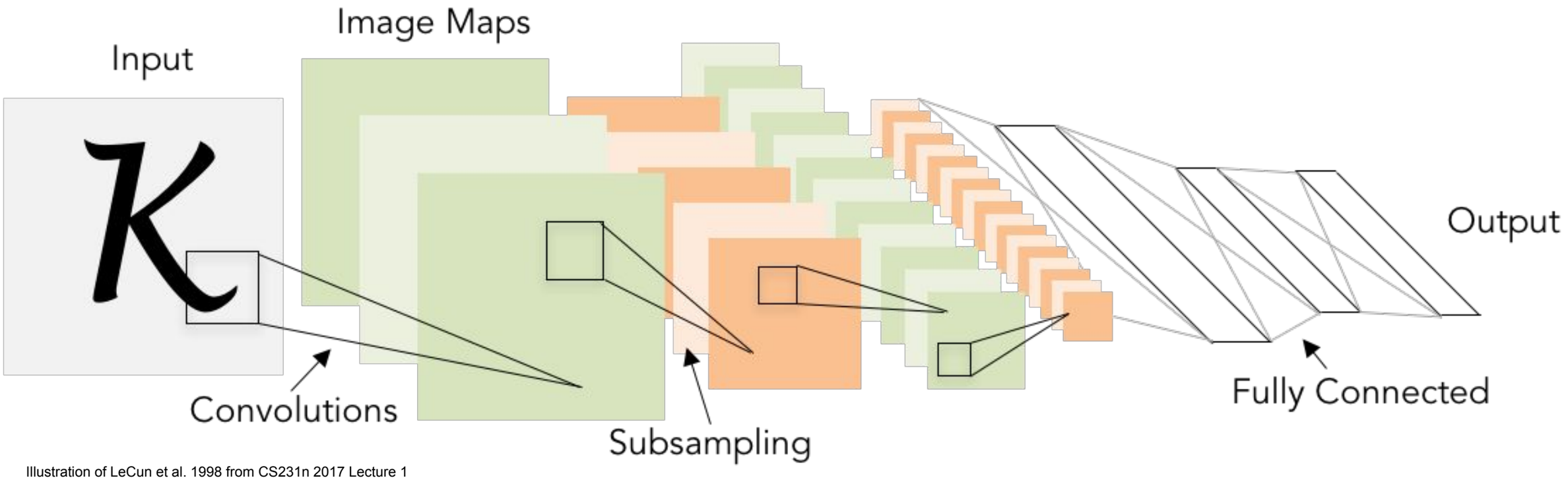
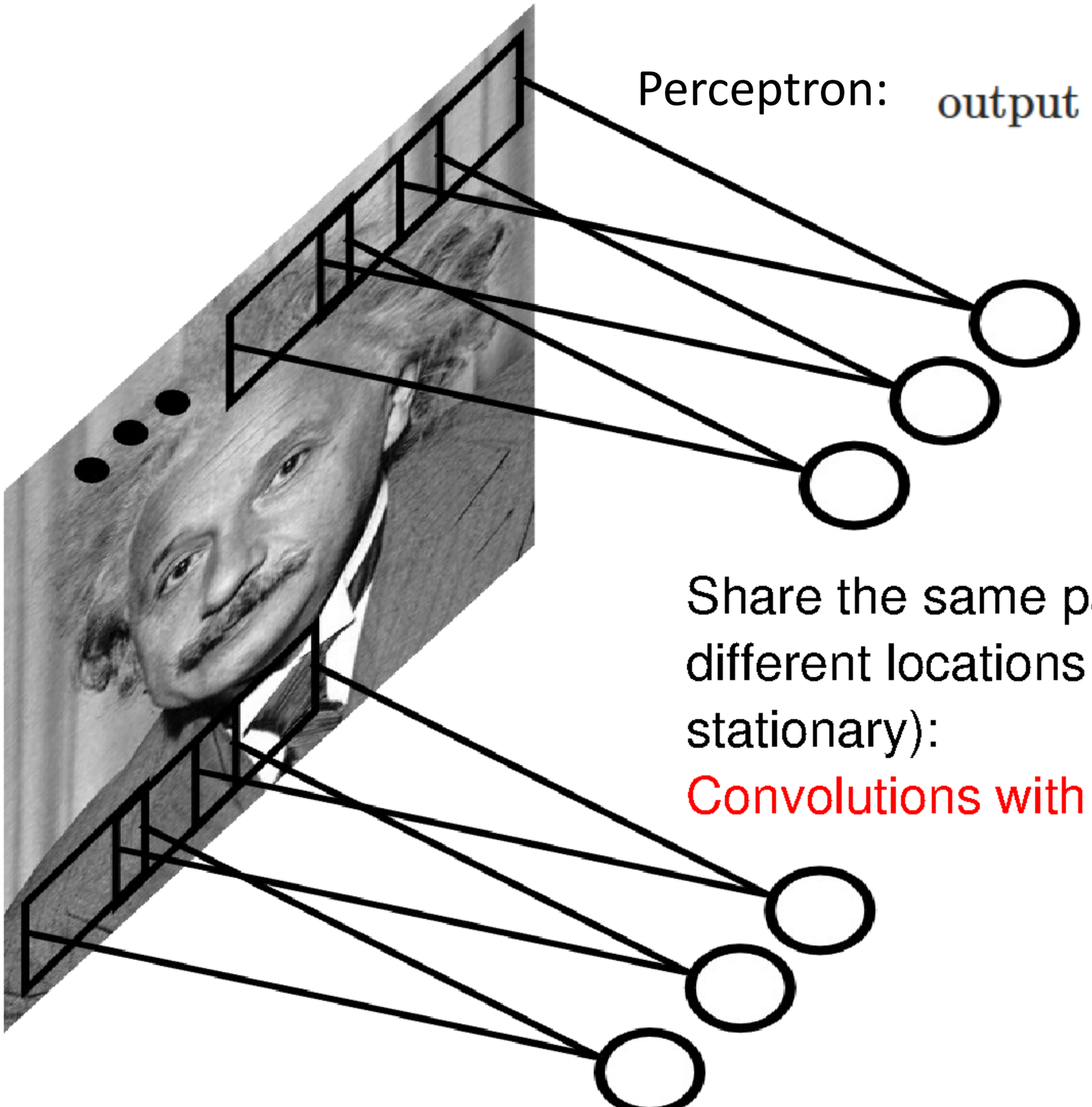


Illustration of LeCun et al. 1998 from CS231n 2017 Lecture 1

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

Filtering reminder:
Correlation (rotated convolution)

$f[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$I[.,.]$

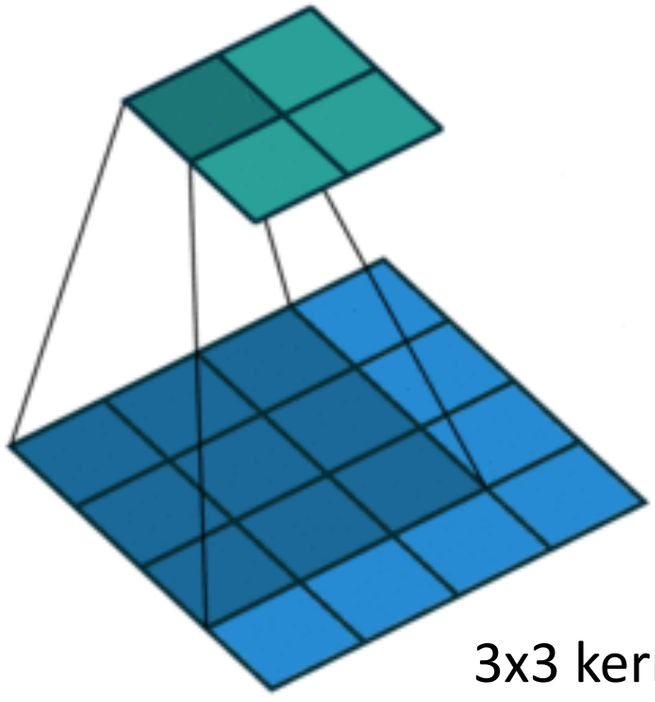
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	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

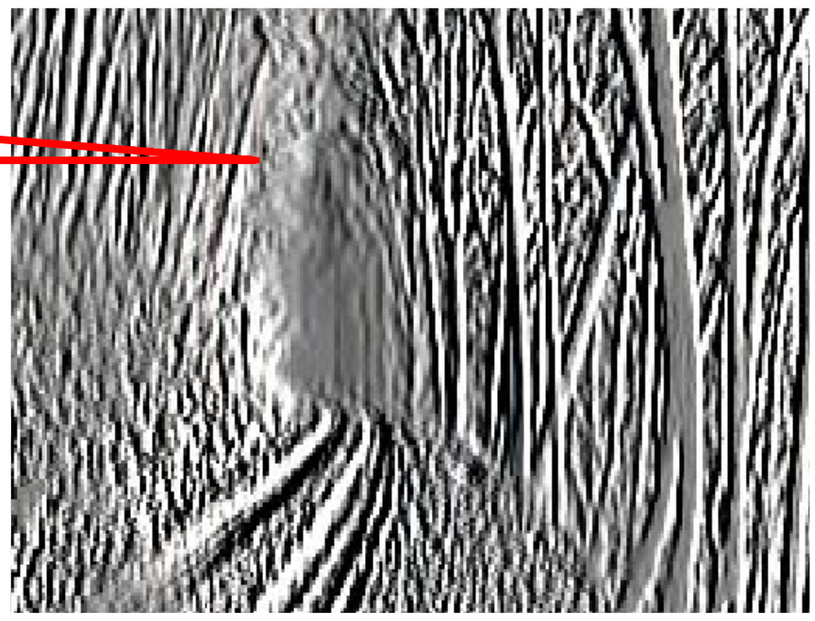


3x3 kernel

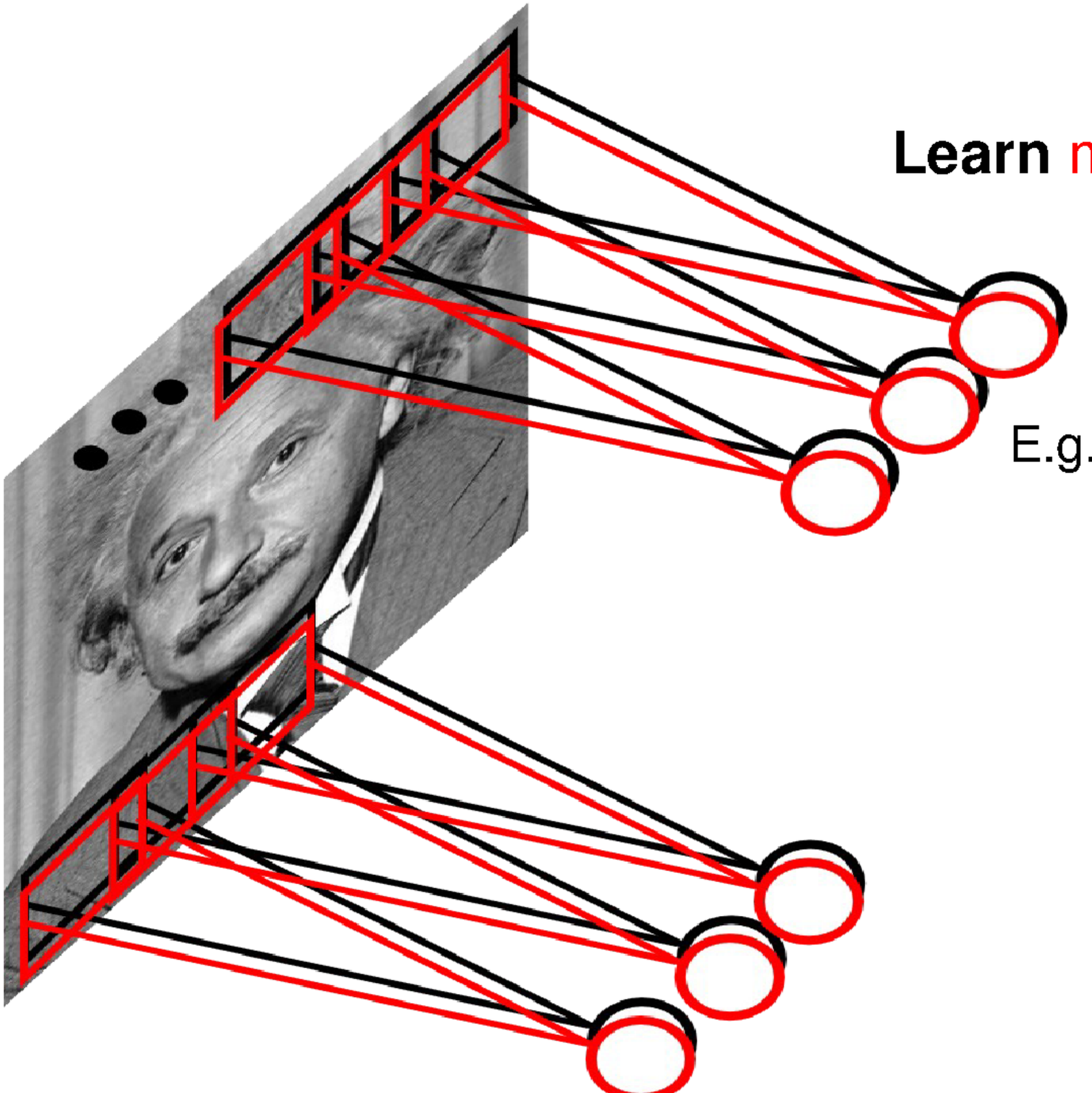
Convolution

$$* \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} =$$

Shared weights



Convolutional Layer



Learn multiple filters.

Filter = 'local' perceptron.

Also called *kernel.*

E.g.: 200x200 image

100 Filters

Filter size: 10x10

10K parameters

Convolutional Layer

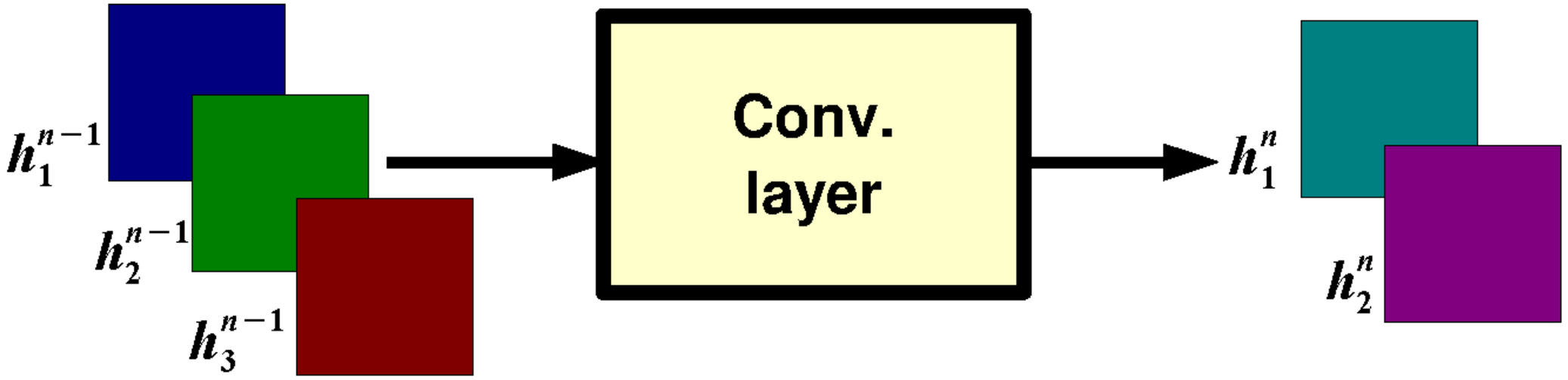
$$h_j^n = max\left(0, \sum_{k=1}^{K} h_k^{n-1} * w_{kj}^n\right)$$

output feature map

input feature map

kernel

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



Ranzato

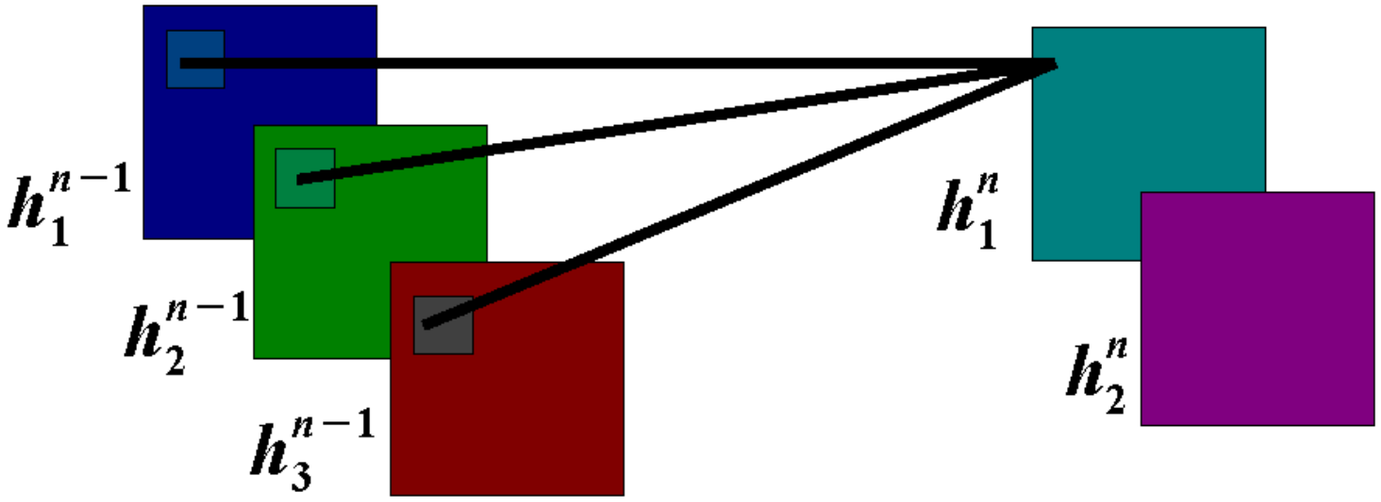
Convolutional Layer

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

output feature map

input feature map

kernel



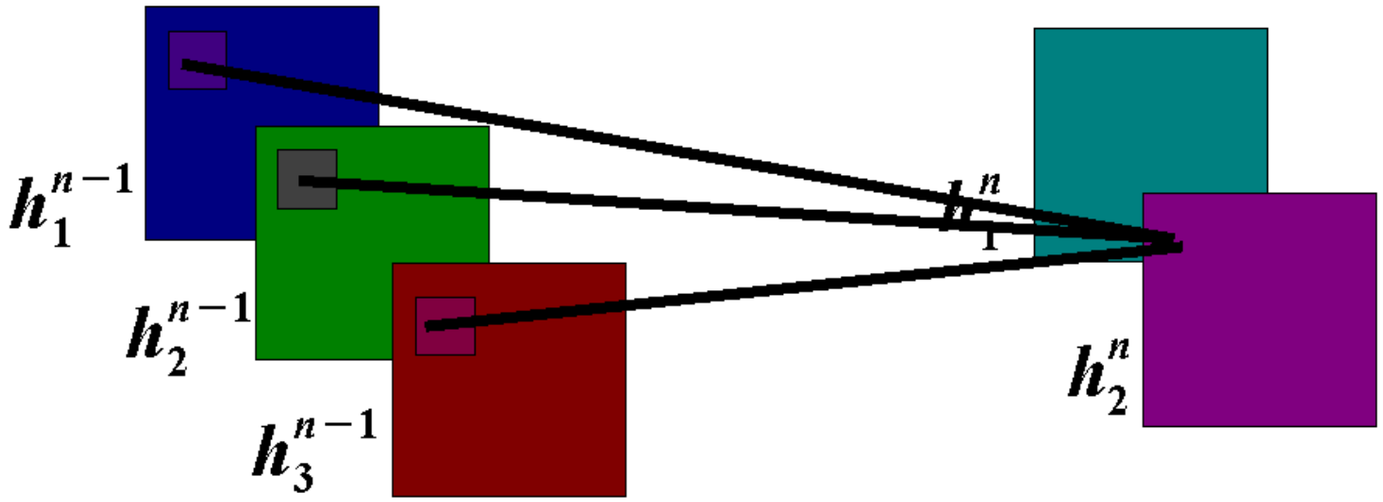
Convolutional Layer

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

output feature map

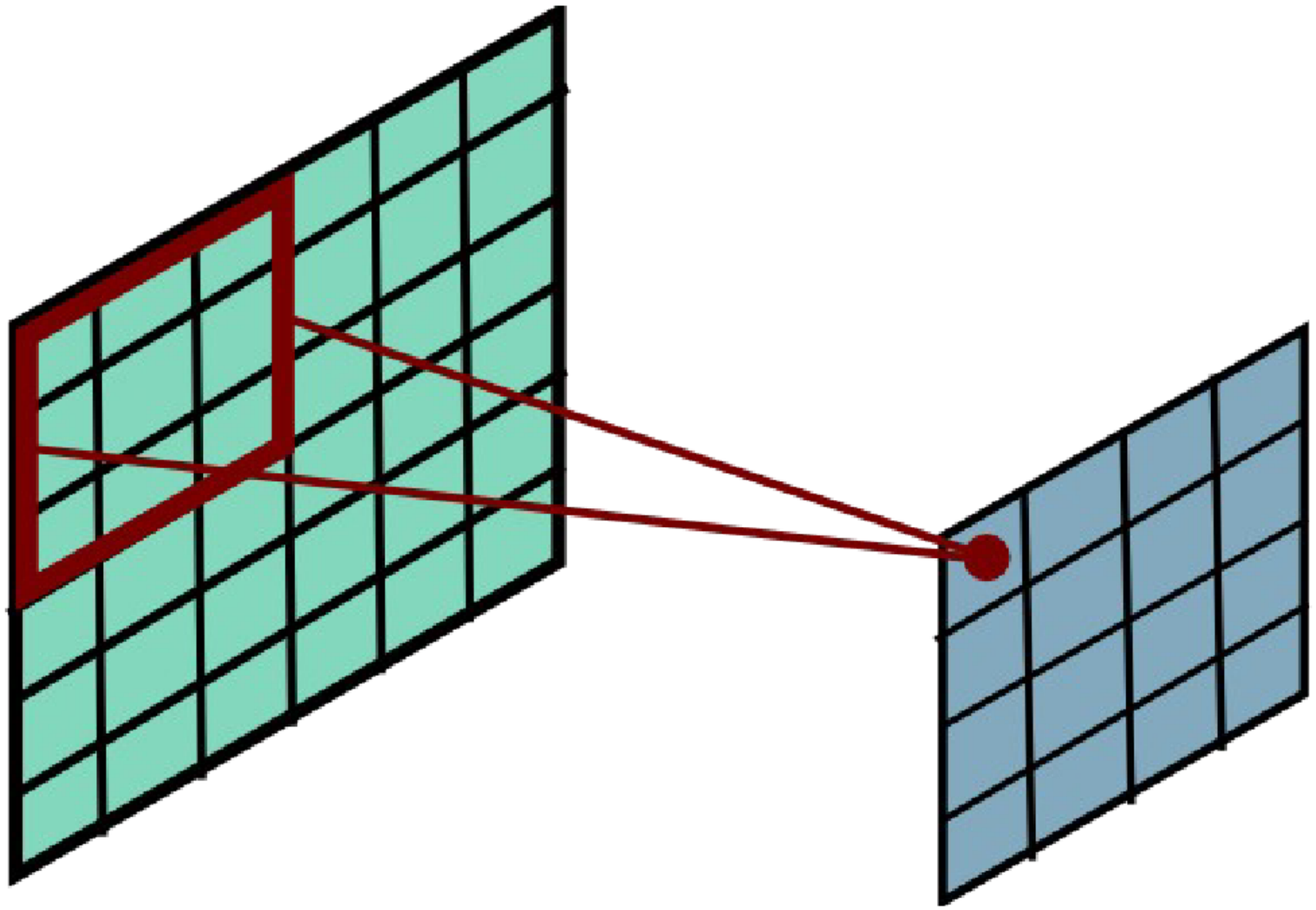
input feature map

kernel

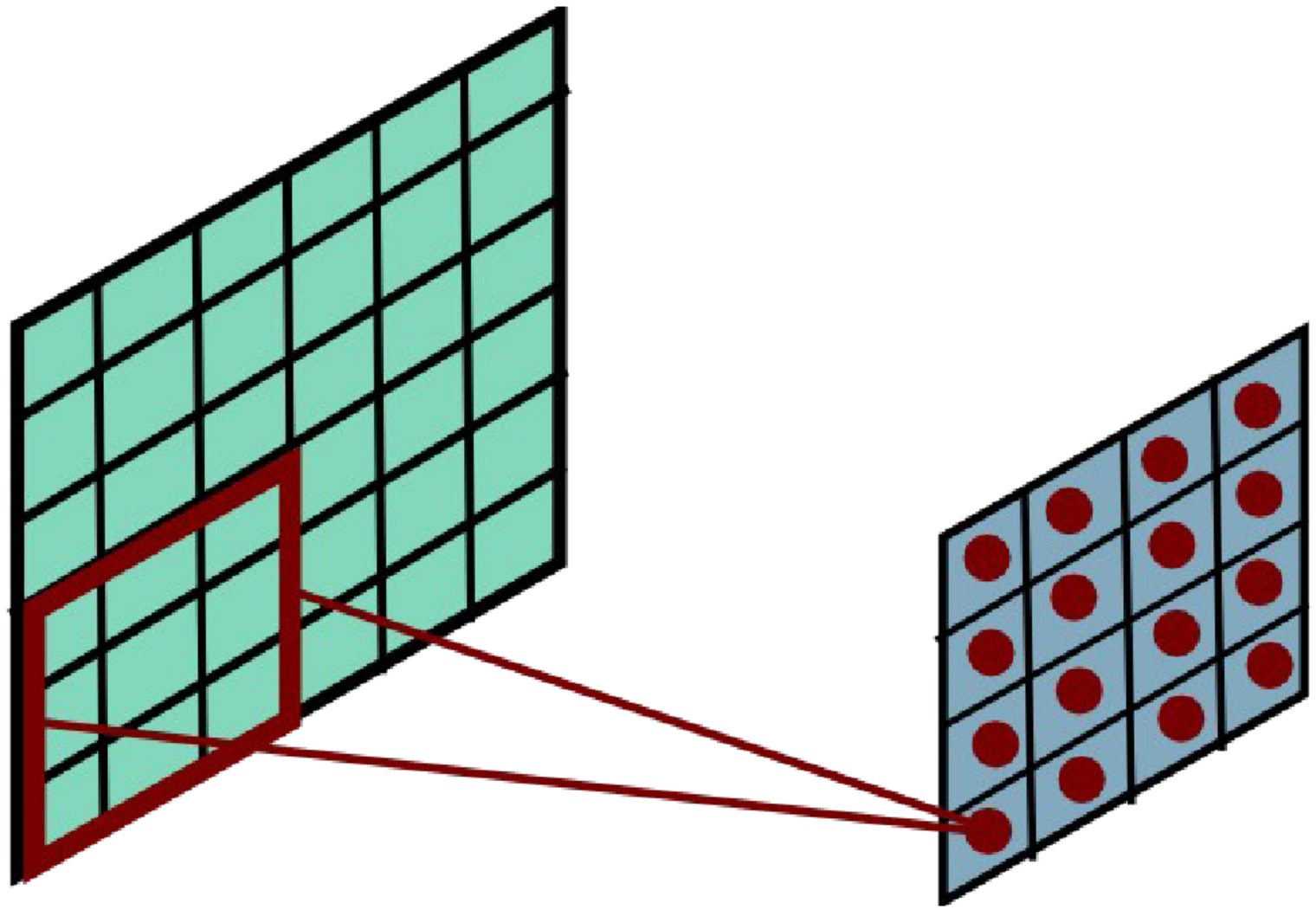


Ranzato

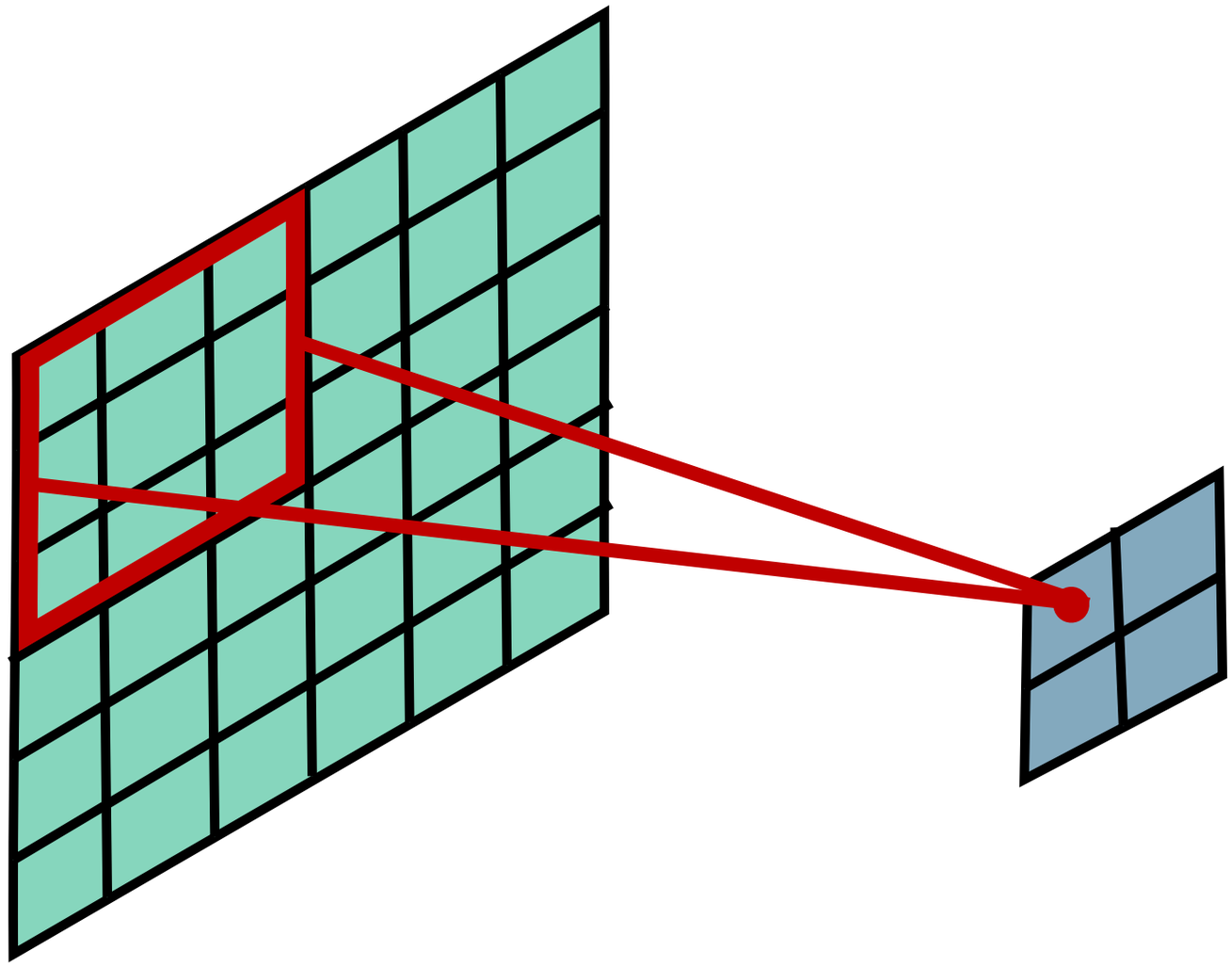
Stride = 1



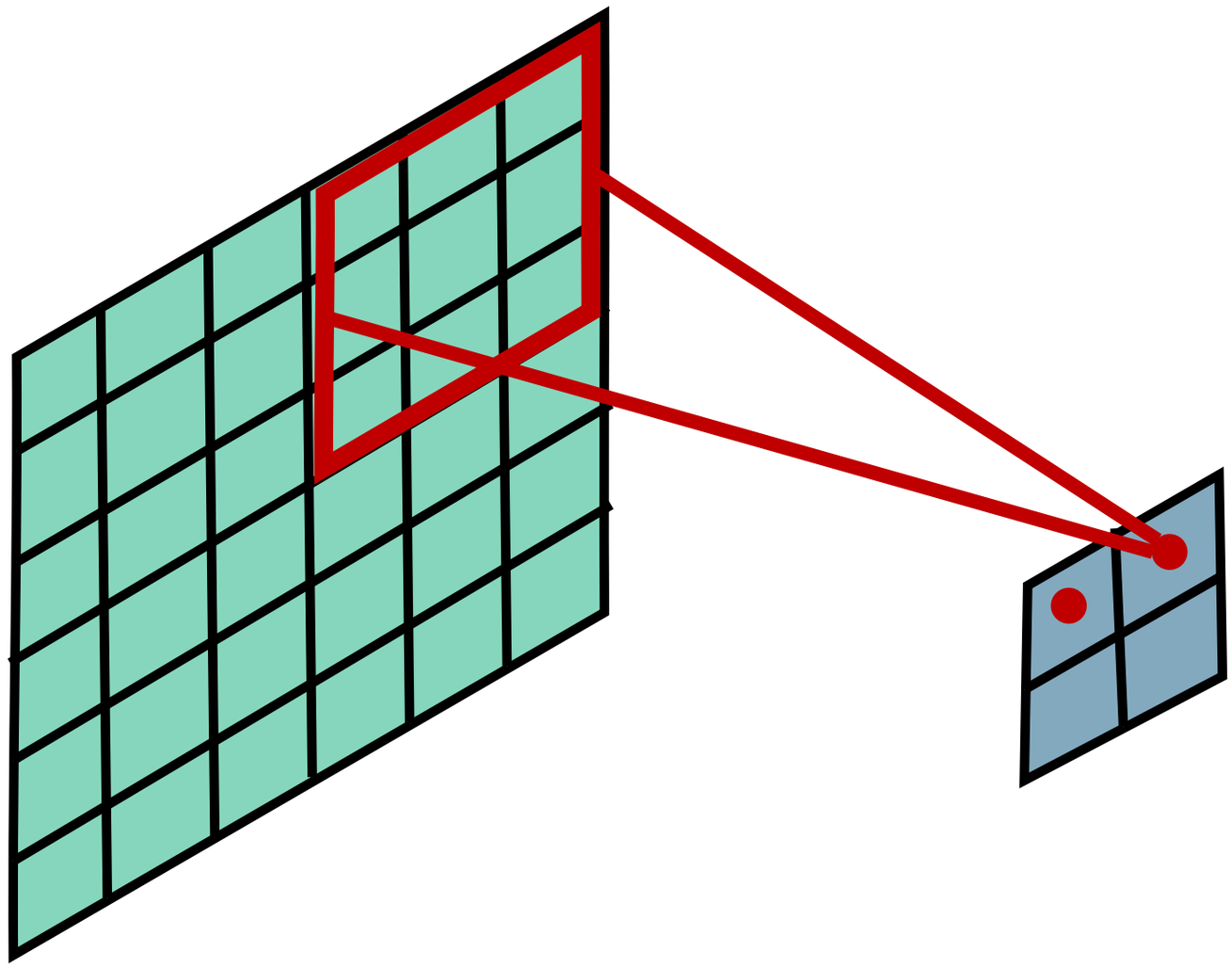
Stride = 1



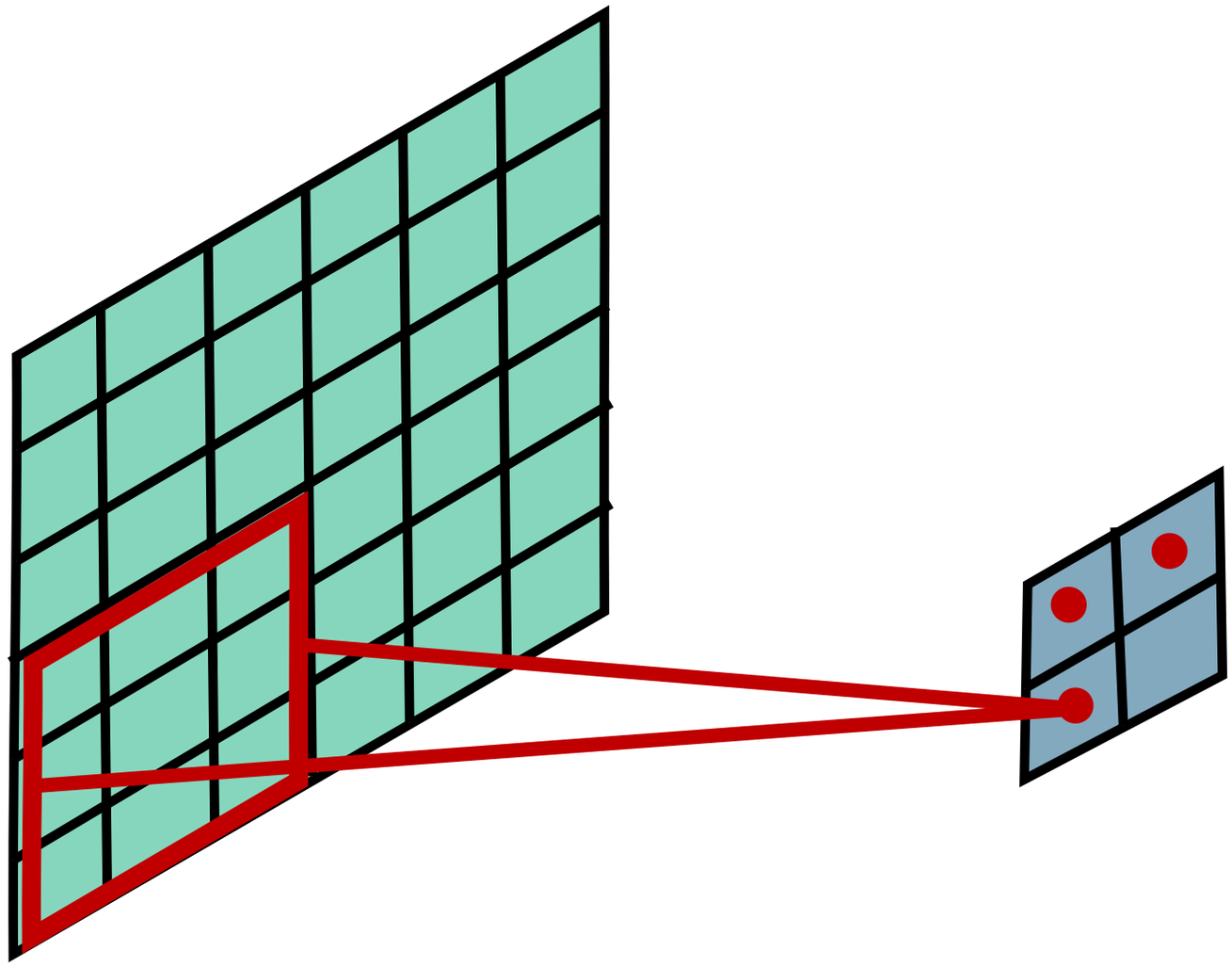
Stride = 3



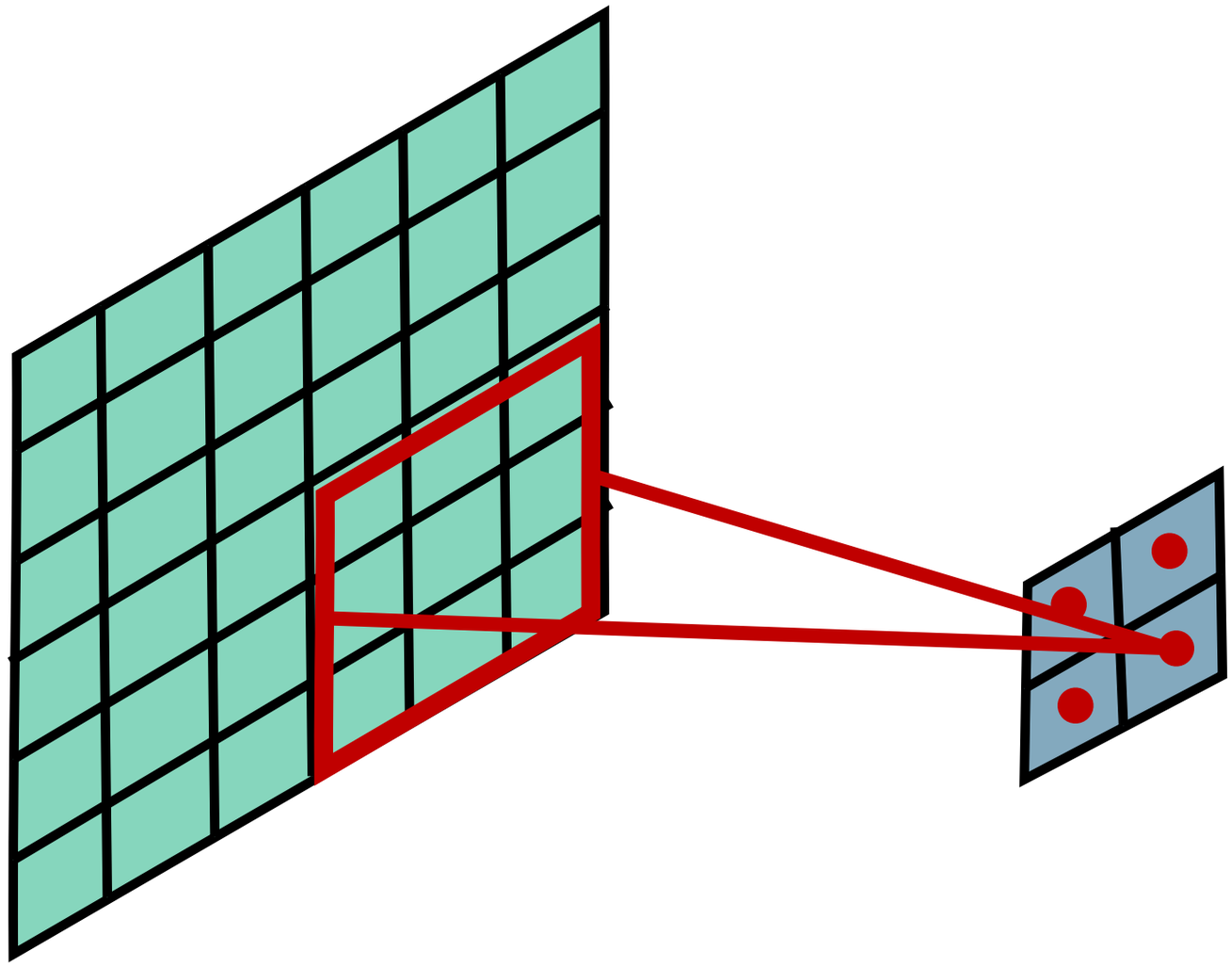
Stride = 3



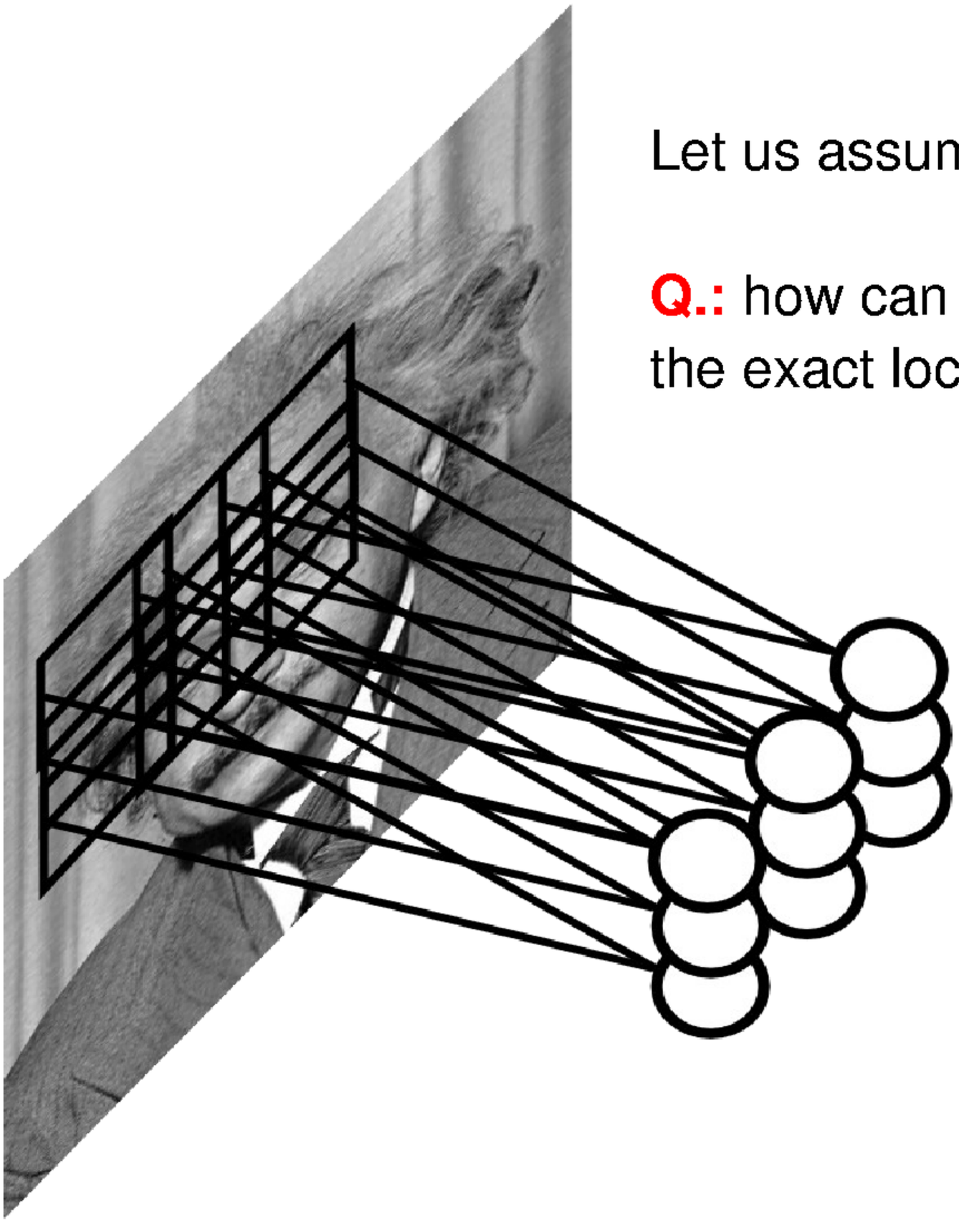
Stride = 3



Stride = 3



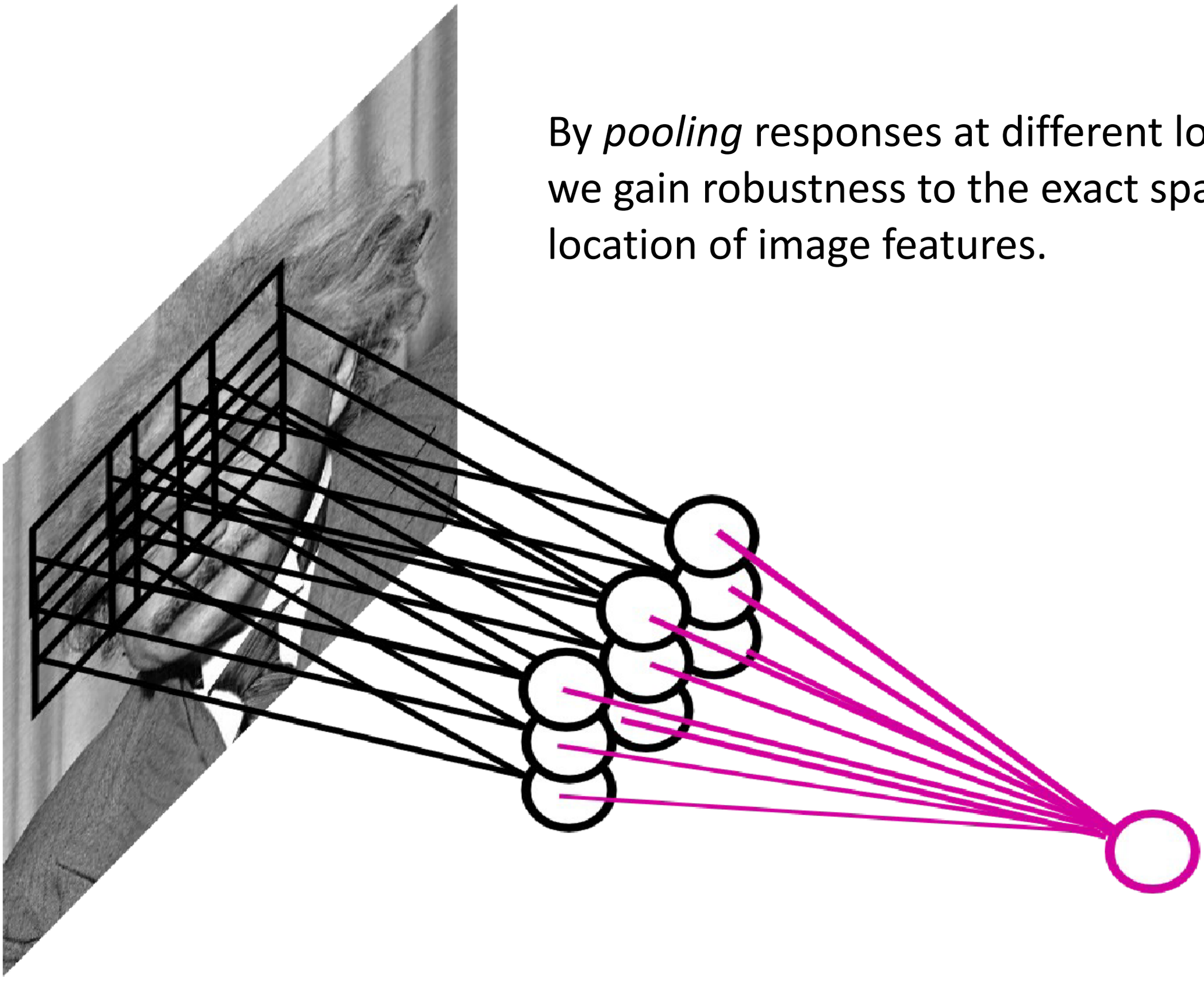
Pooling Layer



Let us assume filter is an “eye” detector.

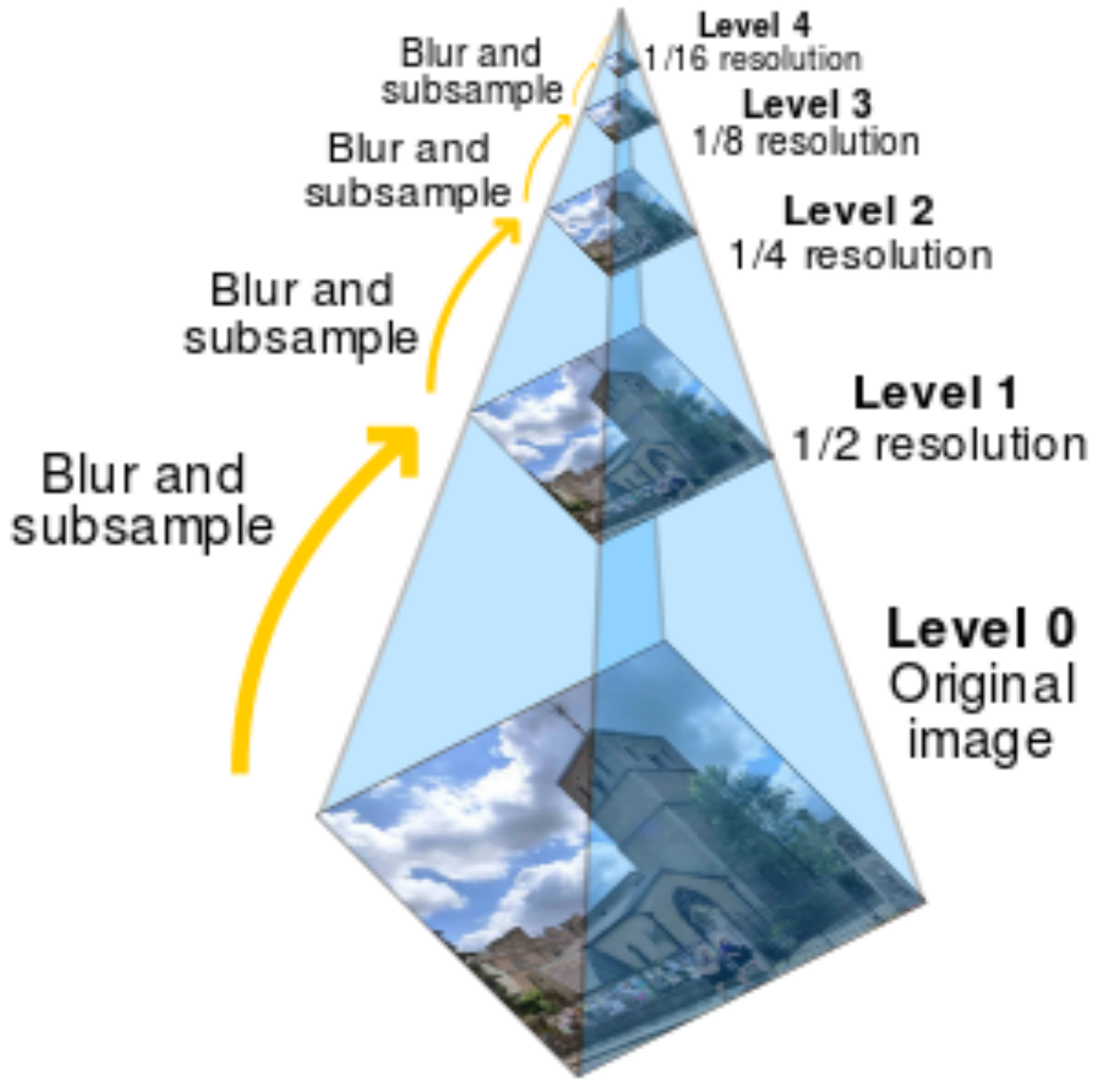
Q.: how can we make the detection robust to the exact location of the eye?

Pooling Layer



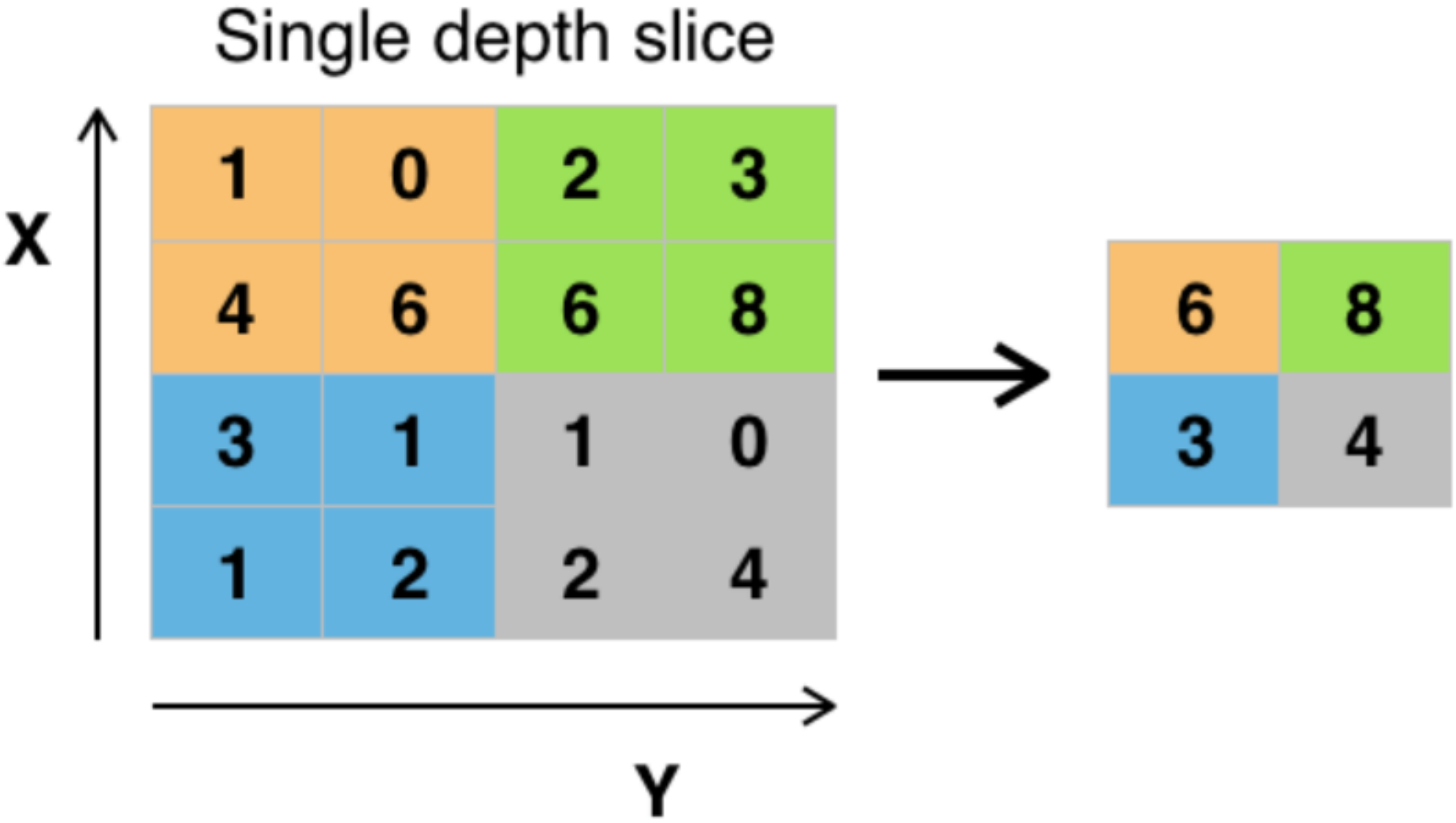
By *pooling* responses at different locations, we gain robustness to the exact spatial location of image features.

Pooling is similar to downsampling



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

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

- Pooling helps to make the representation become approximately invariant to small translations of the input.
- Invariance to local translation can be a very useful property if we care more about whether some feature is present than exactly where it is.

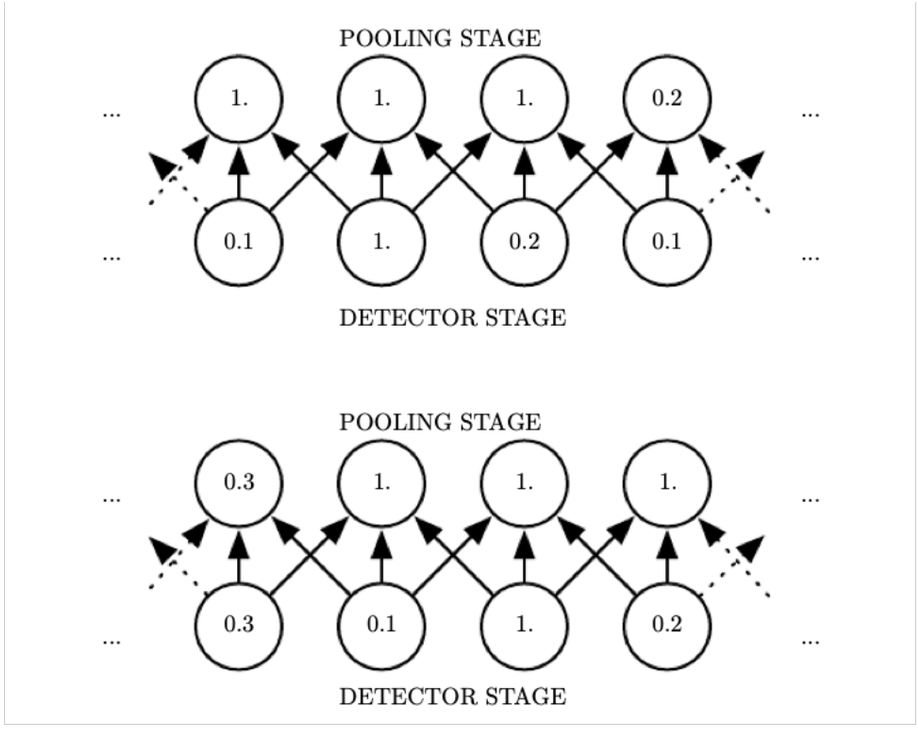


Figure 9.8: Max pooling introduces invariance. *(Top)* A view of the middle of the output of a convolutional layer. The bottom row shows outputs of the nonlinearity. The top row shows the outputs of max pooling, with a stride of one pixel between pooling regions and a pooling region width of three pixels. *(Bottom)* A view of the same network, after the input has been shifted to the right by one pixel. Every value in the bottom row has changed, but only half of the values in the top row have changed, because the max pooling units are only sensitive to the maximum value in the neighborhood, not its exact location.

Pooling Layer

- If we pool over the outputs of separately parameterized convolutions, the features can learn which transformations to become invariant to.

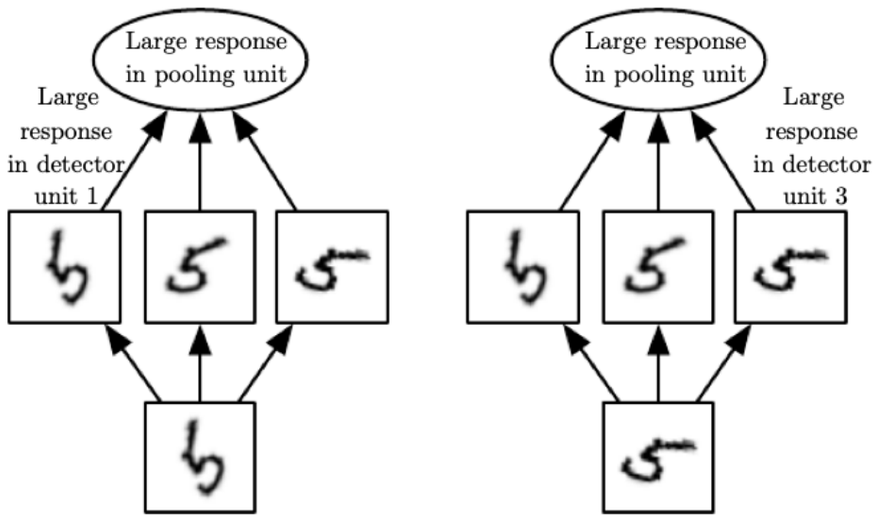
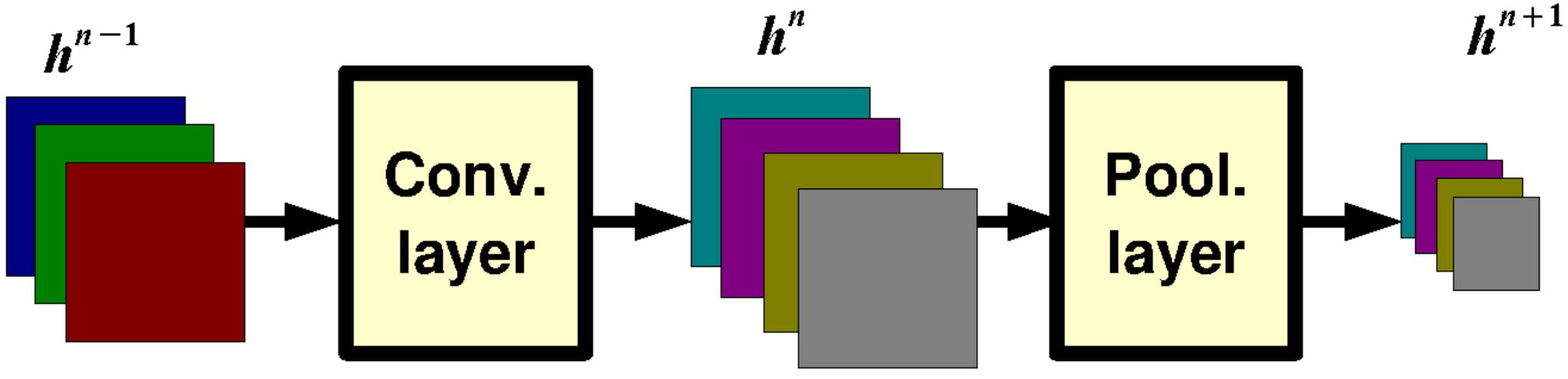
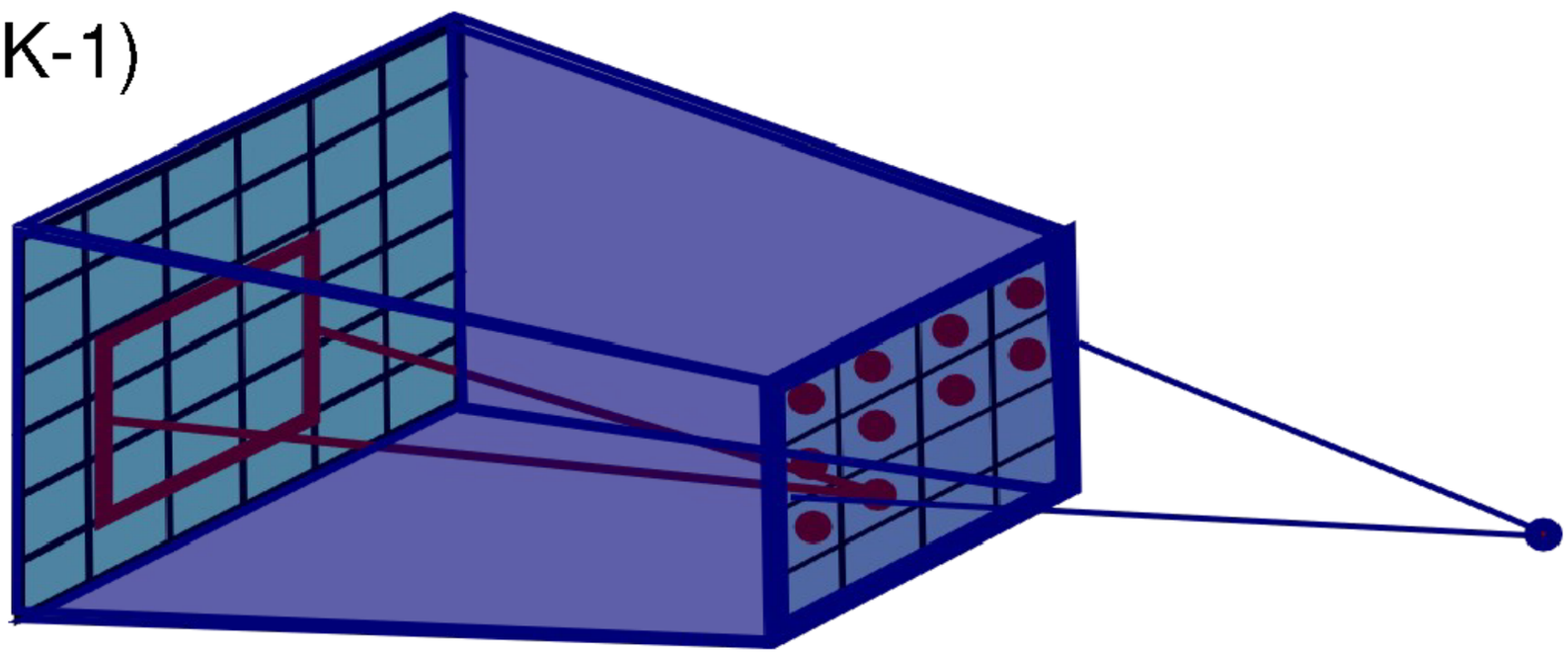


Figure 9.9: *Example of learned invariances:* A pooling unit that pools over multiple features that are learned with separate parameters can learn to be invariant to transformations of the input. Here we show how a set of three learned filters and a max pooling unit can learn to become invariant to rotation. All three filters are intended to detect a hand-written 5. Each filter attempts to match a slightly different orientation of the 5. When a 5 appears in the input, the corresponding filter will match it and cause a large activation in a detector unit. The max pooling unit then has a large activation regardless of which pooling unit was activated. We show here how the network processes two different inputs, resulting in two different detector units being activated. The effect on the pooling unit is roughly the same either way. This principle is leveraged by maxout networks (Goodfellow *et al.*, 2013a) and other convolutional networks. Max pooling over spatial positions is naturally invariant to translation; this multi-channel approach is only necessary for learning other transformations.

Pooling Layer: Receptive Field Size

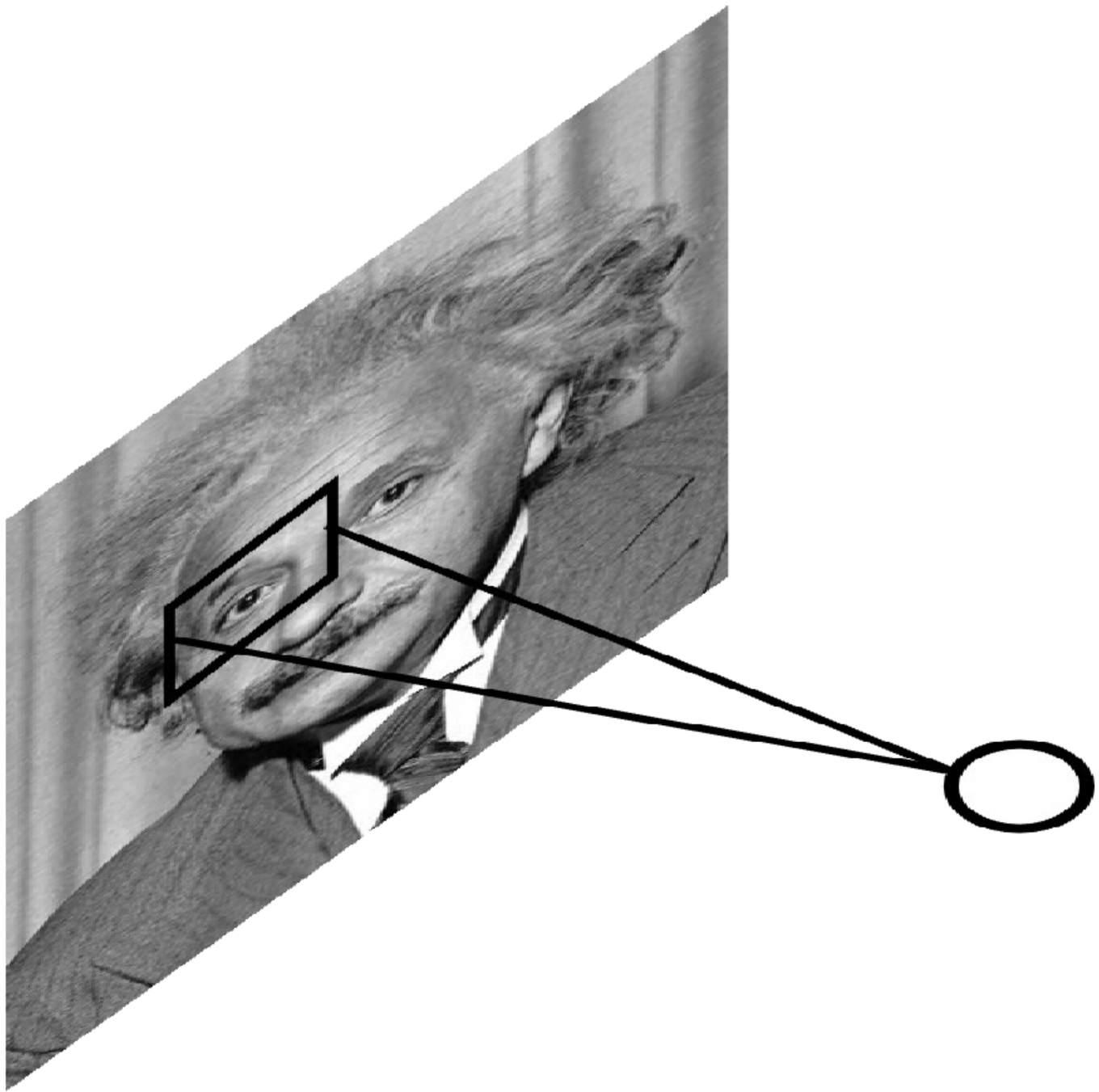


If convolutional filters have size KxK and stride 1, and pooling layer has pools of size PxP, then each unit in the pooling layer depends upon a patch (at the input of the preceding conv. layer) of size: (P+K-1)x(P+K-1)

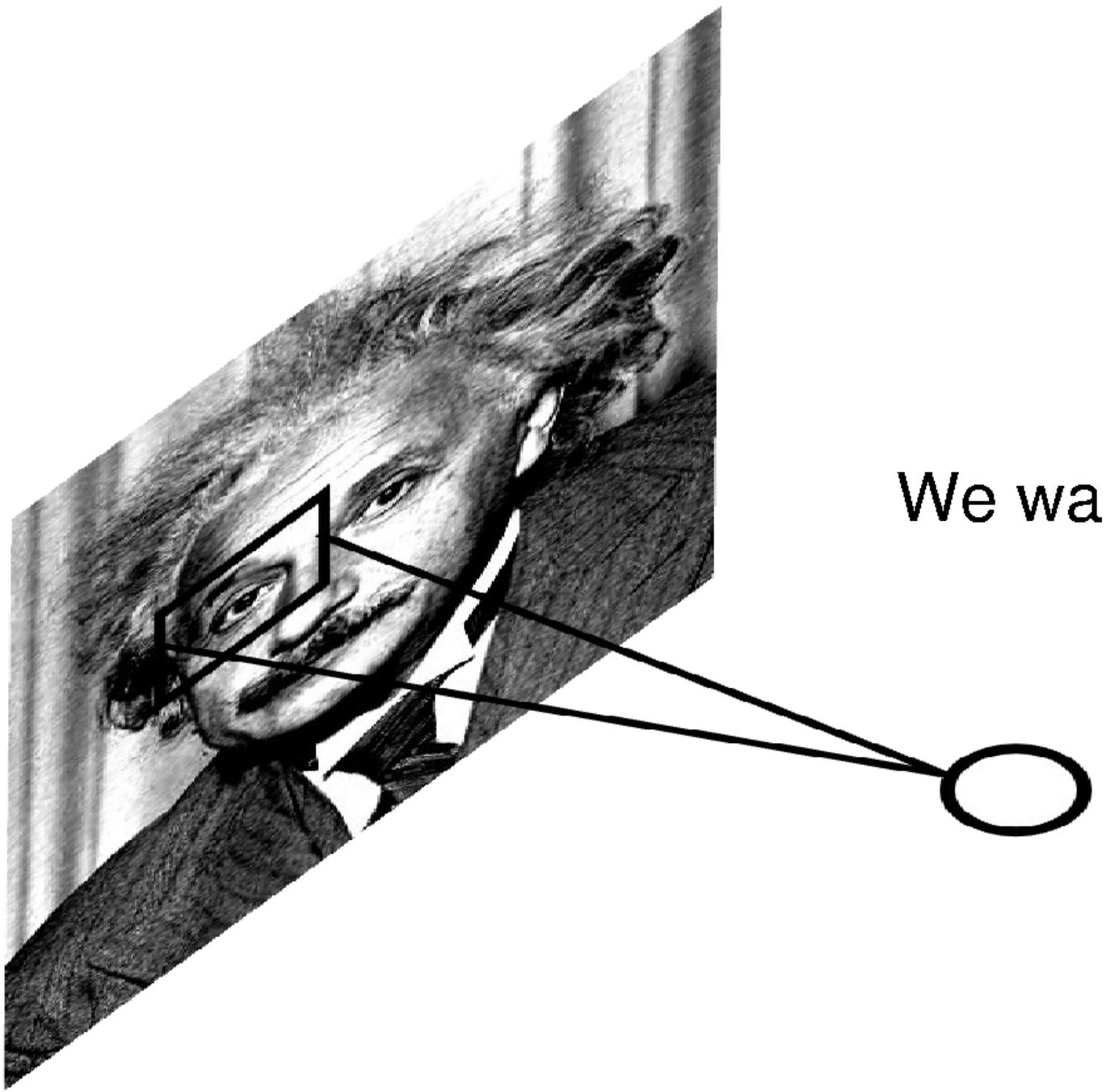


Ranzato

Local Contrast Normalization



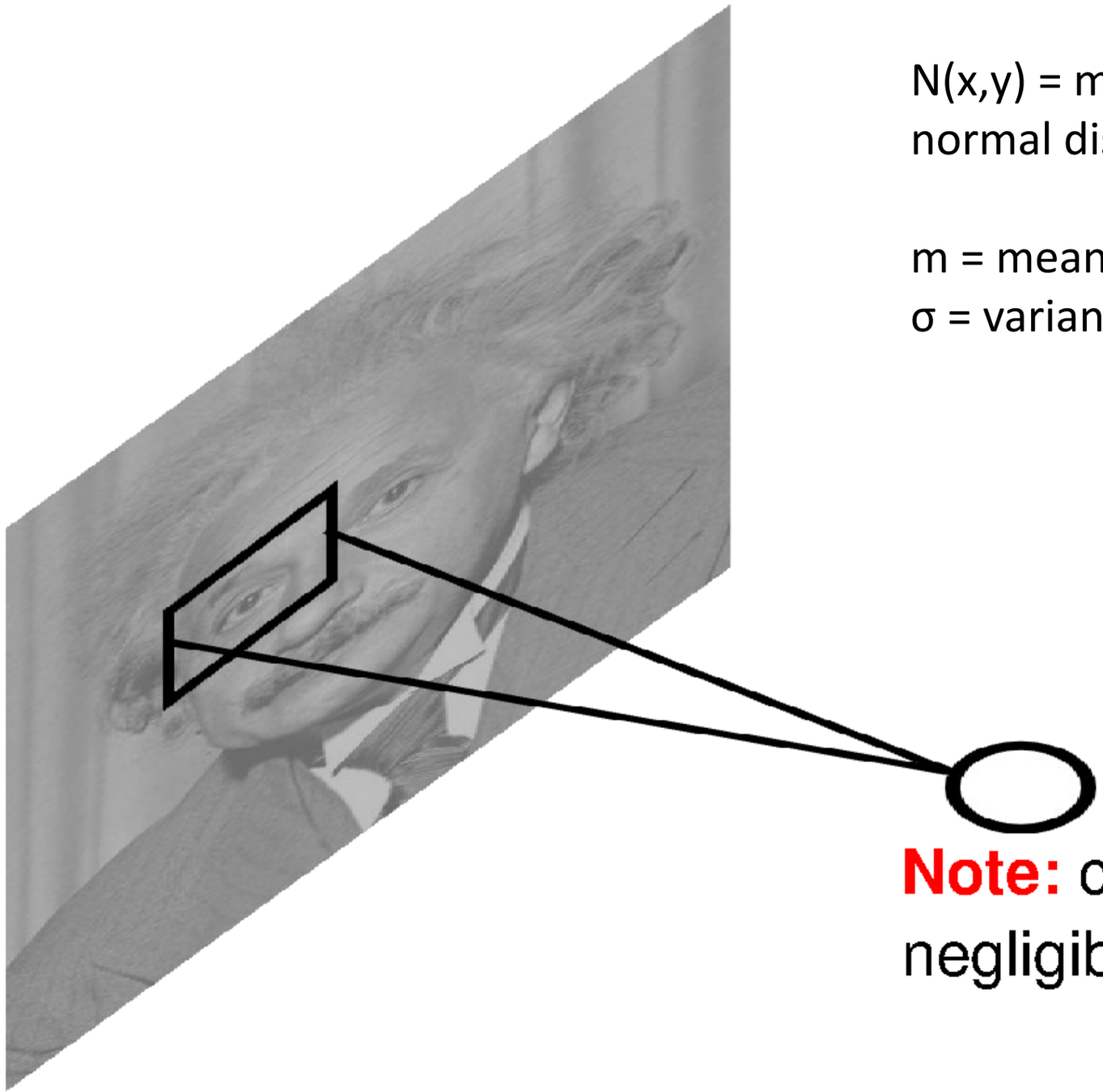
Local Contrast Normalization



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

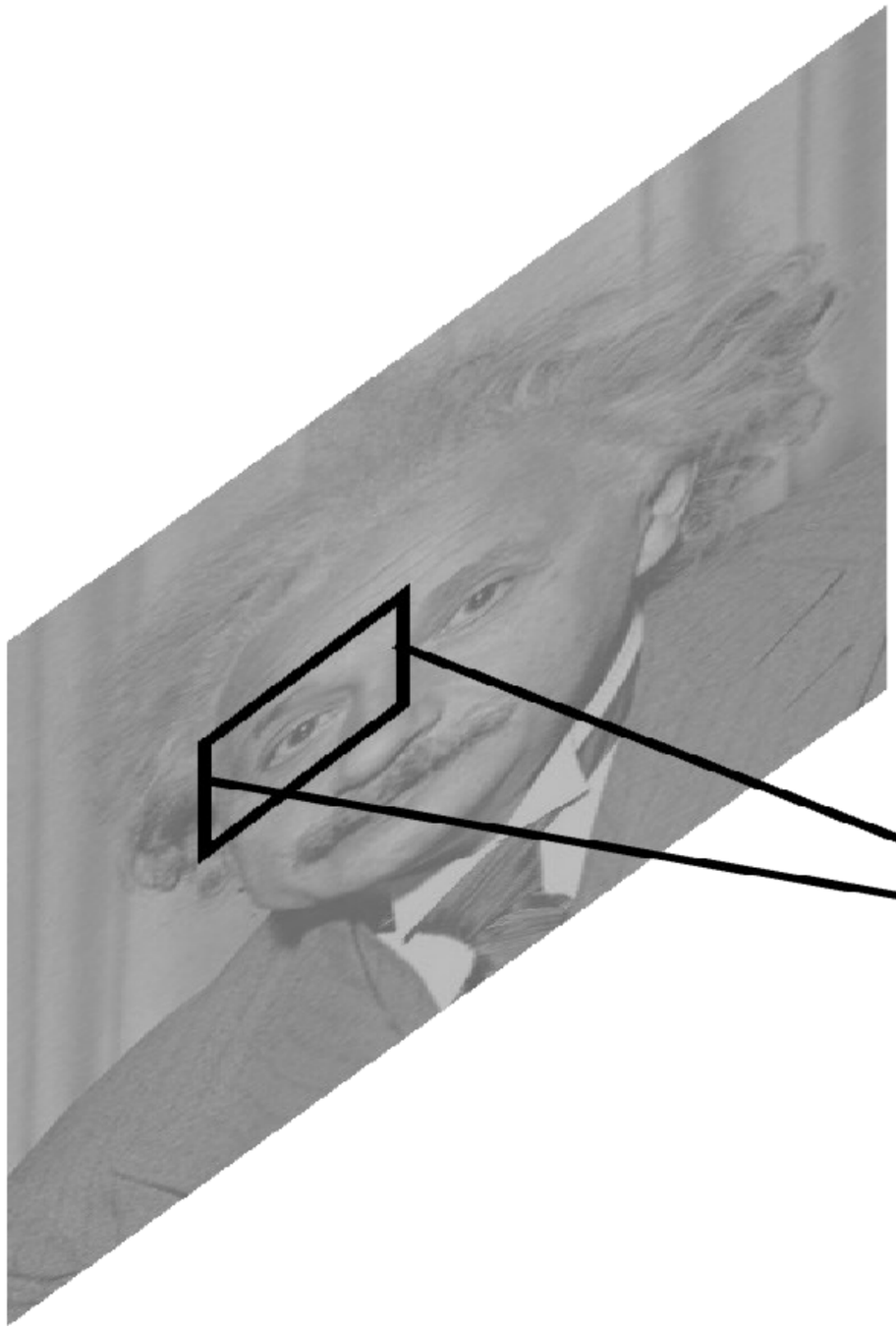
m = mean

σ = variance

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

Local Contrast Normalization

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

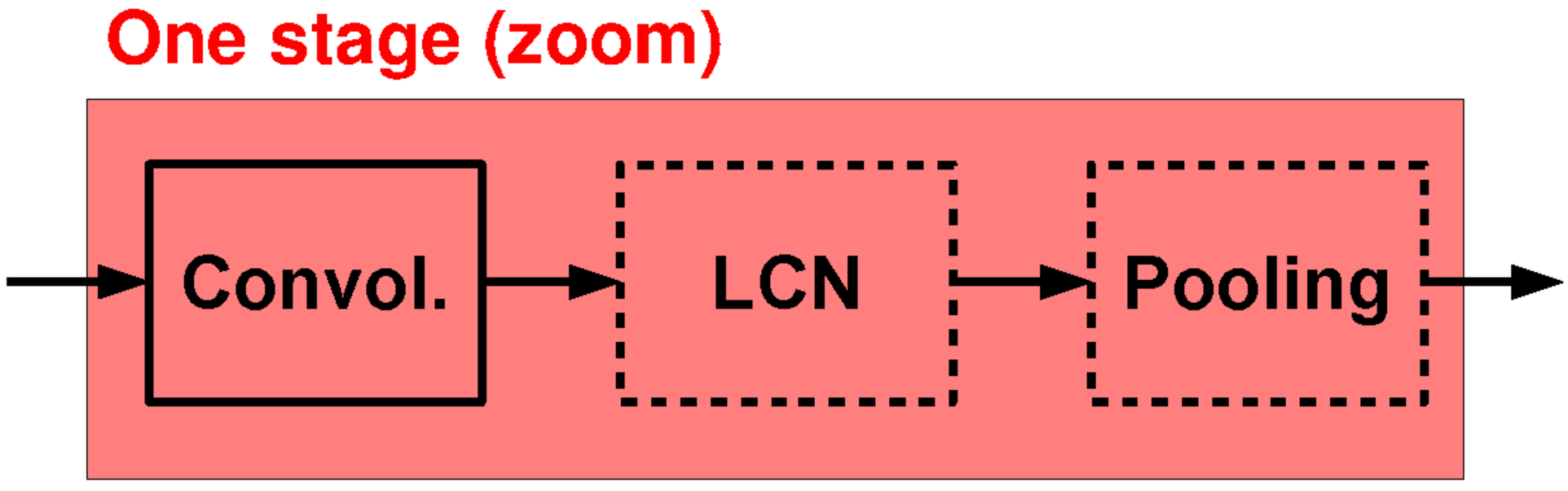


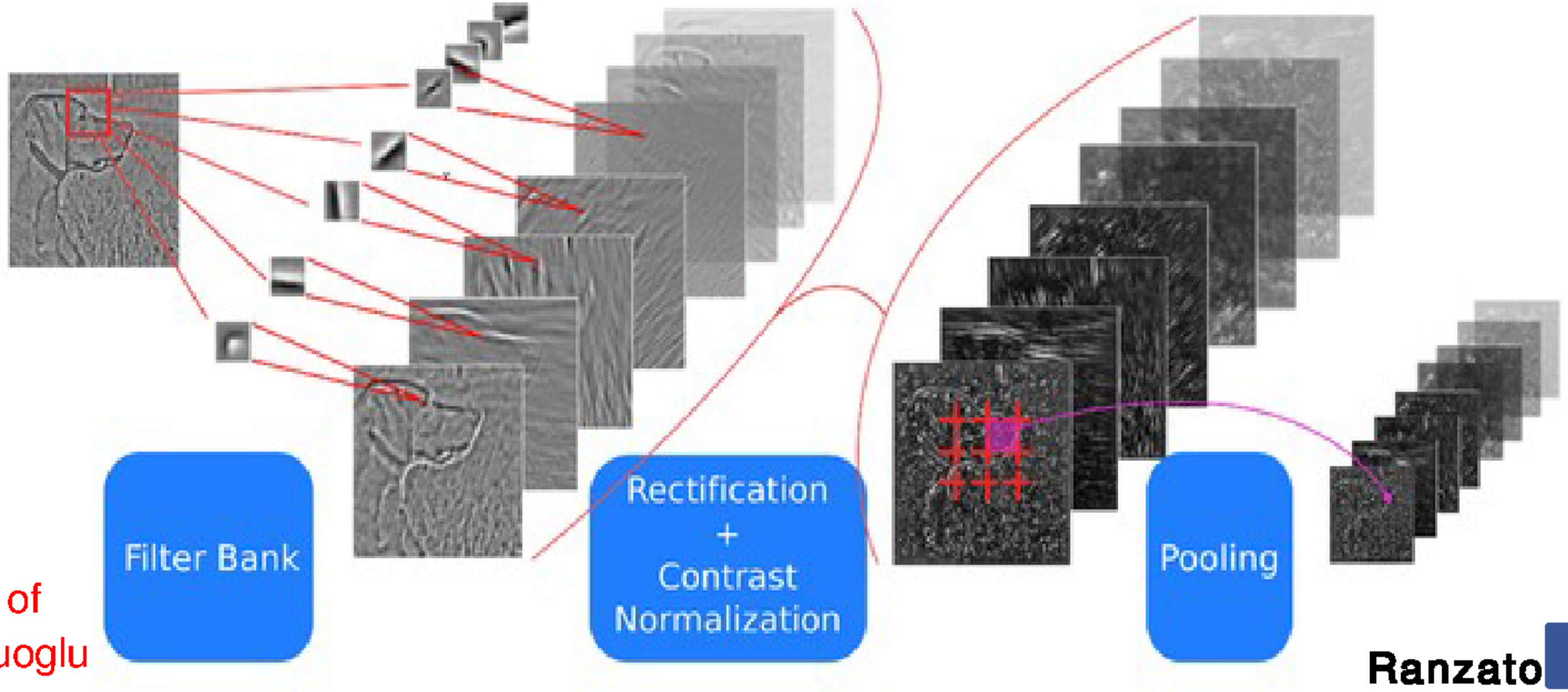
Performed also across features and in the higher layers..

Effects:
– improves invariance
– improves optimization
– increases sparsity

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

ConvNets: Typical Stage

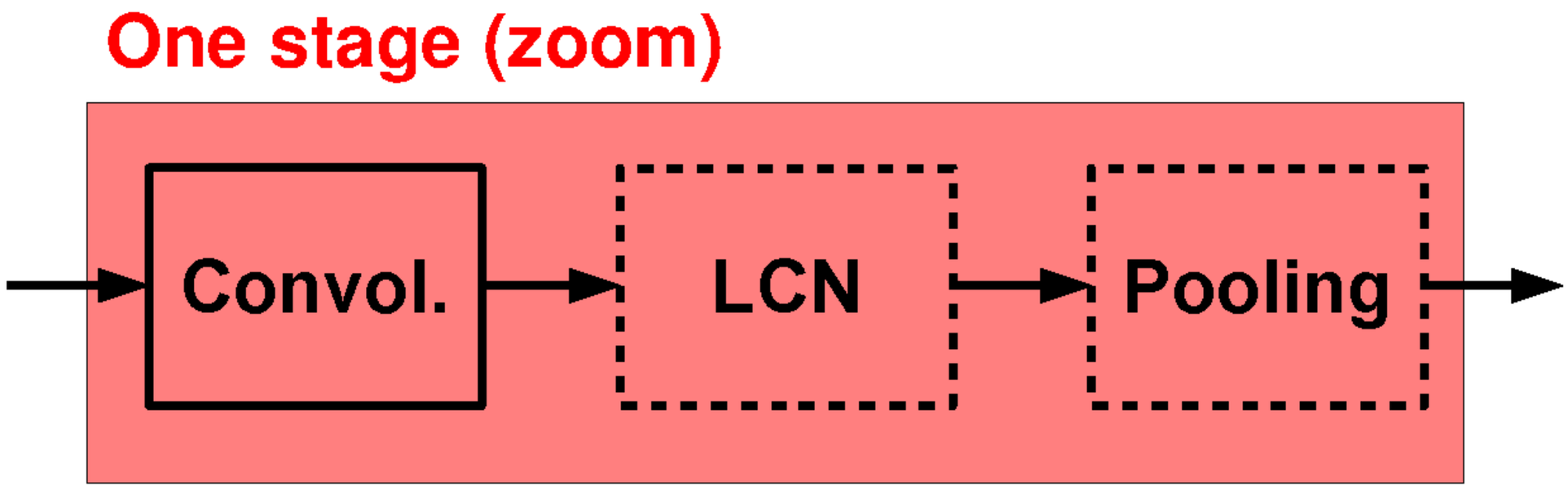




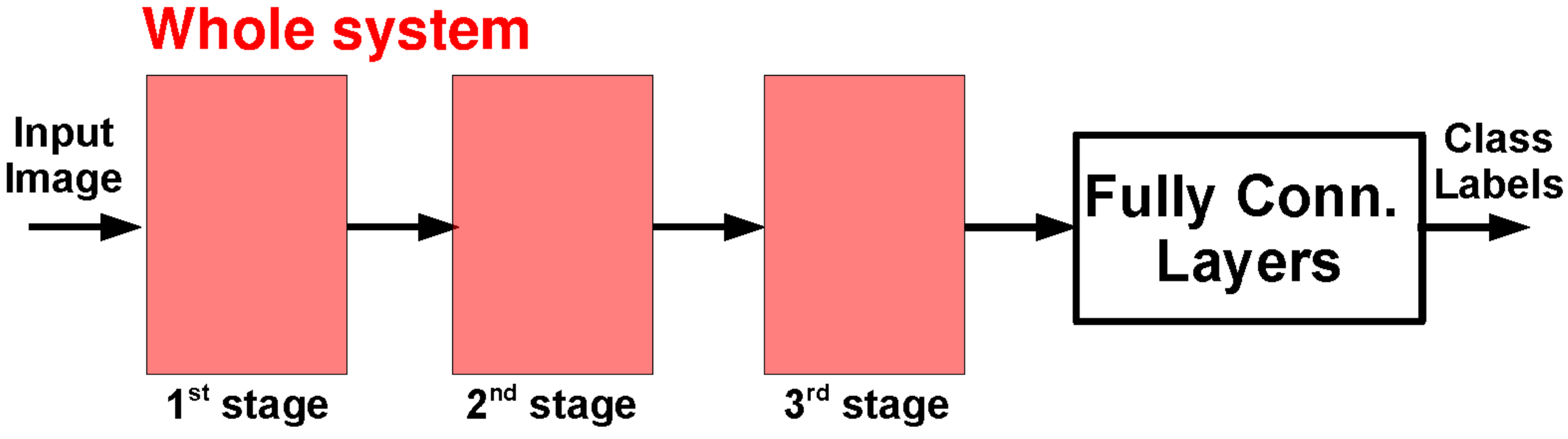
courtesy of
K. Kavukcuoglu

Ranzato

ConvNets: Typical Architecture



ConvNets: Typical Architecture



Conceptually similar to:

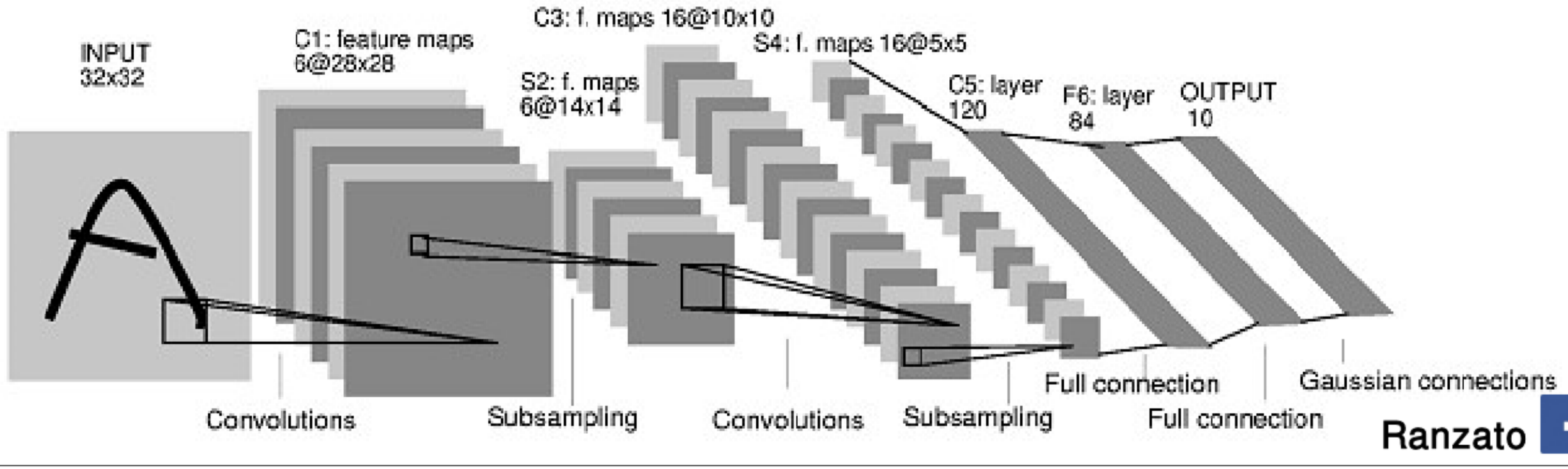
SIFT → K-Means → Pyramid Pooling → SVM

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

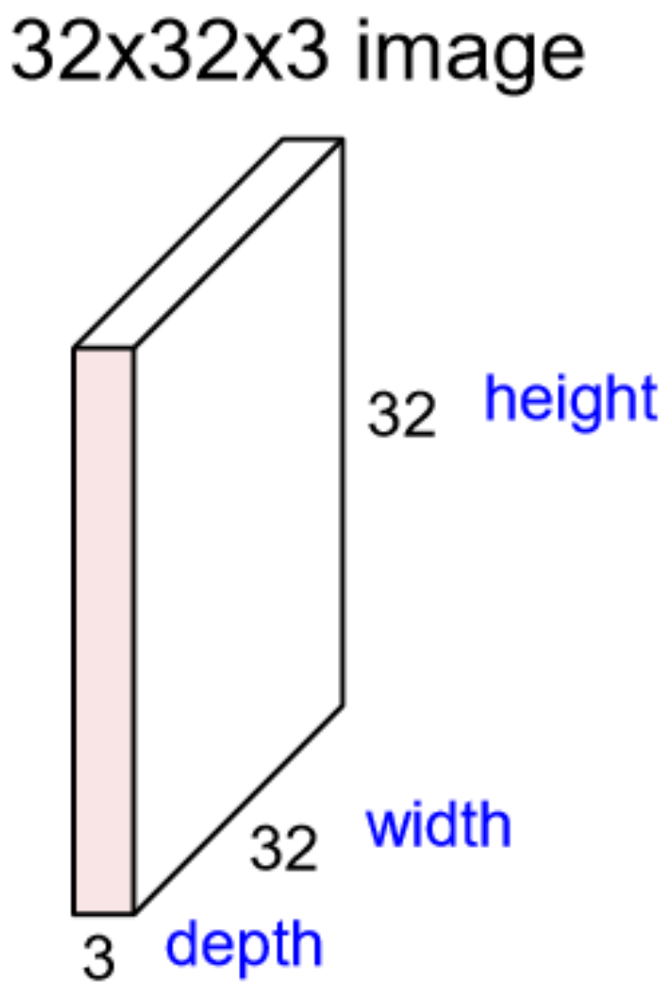
SIFT → Fisher Vect. → Pooling → SVM

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

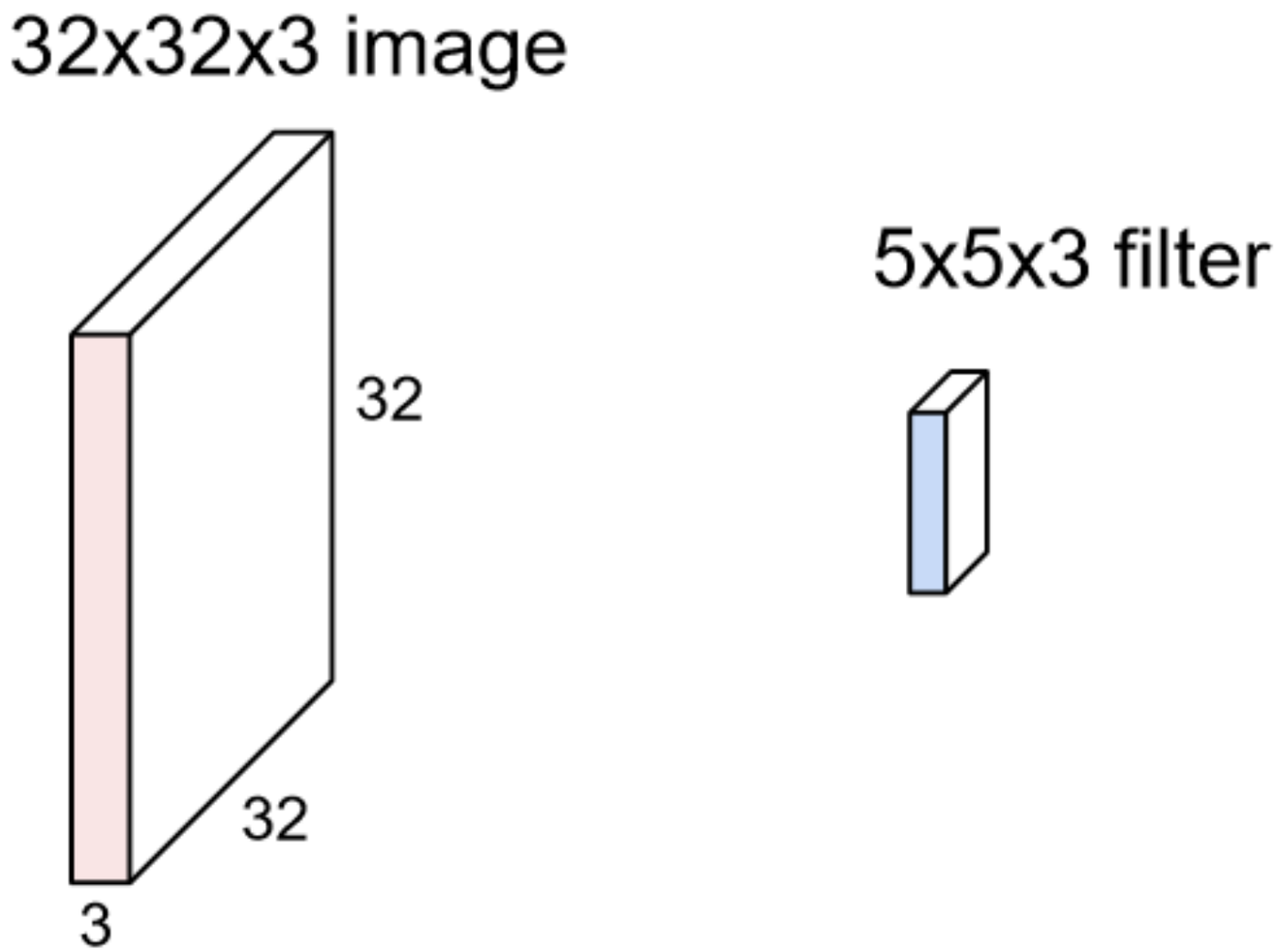
Yann LeCun's MNIST CNN architecture



Convolutions: More detail

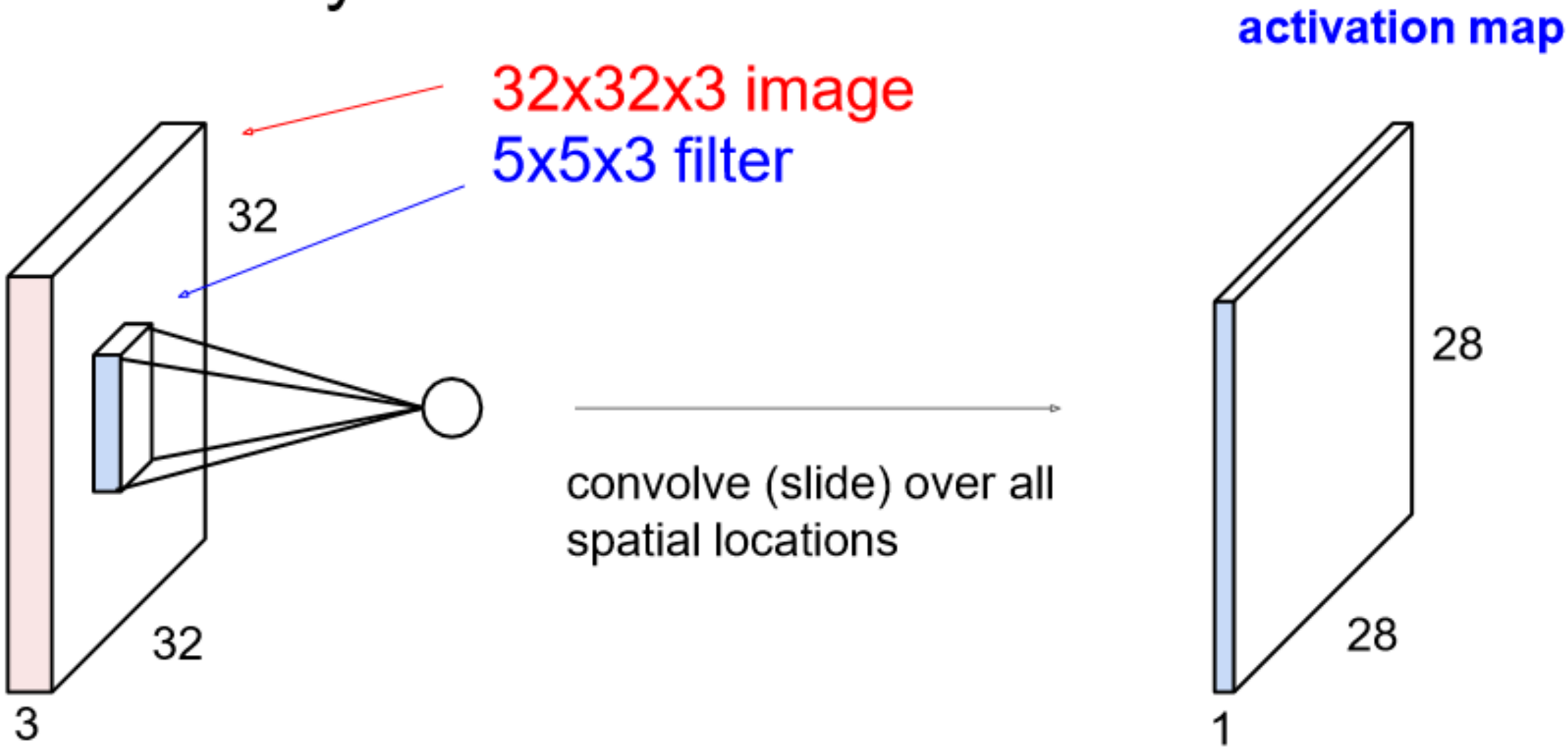


Convolutions: More detail



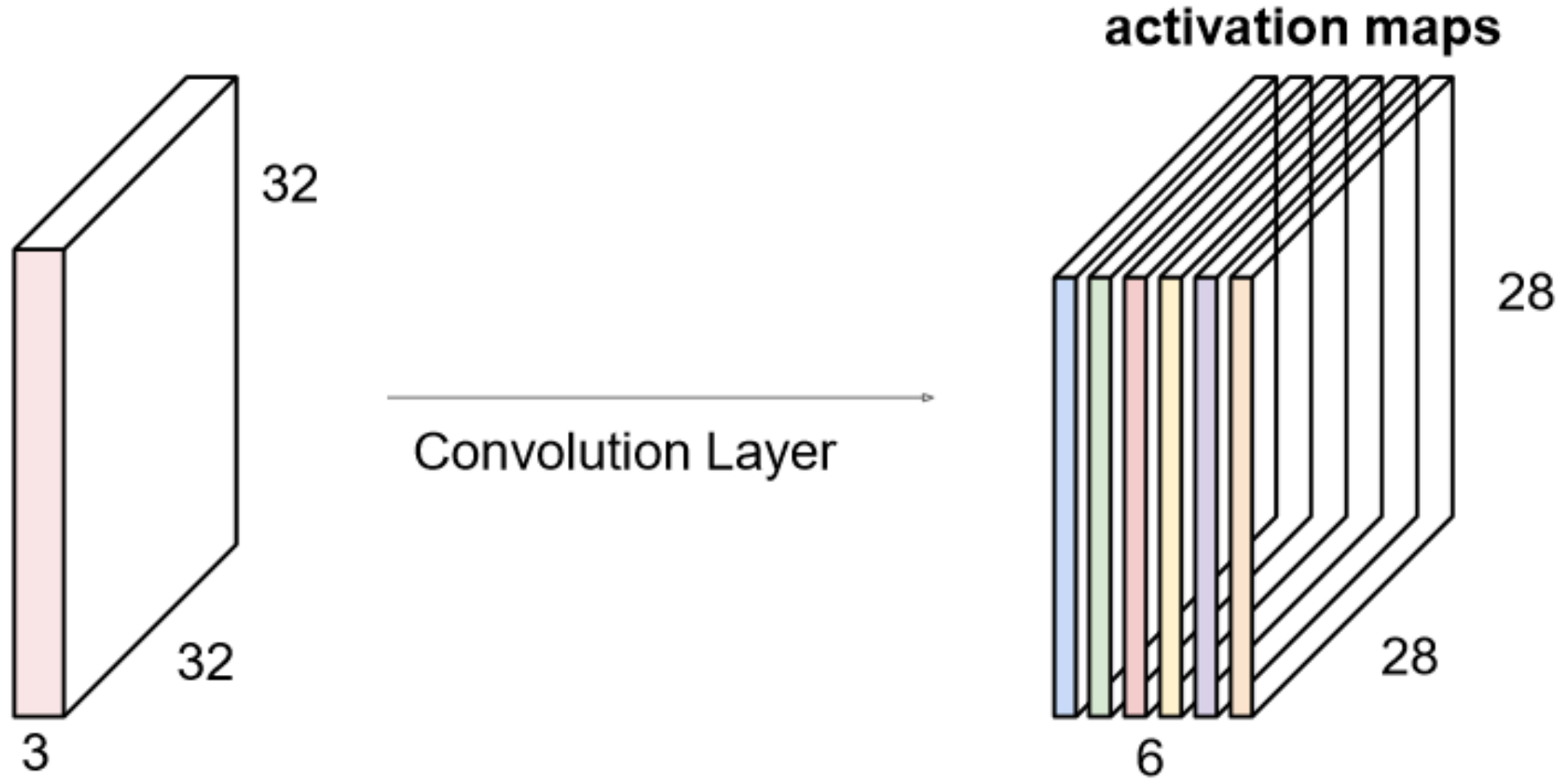
Convolutions: More detail

Convolution Layer



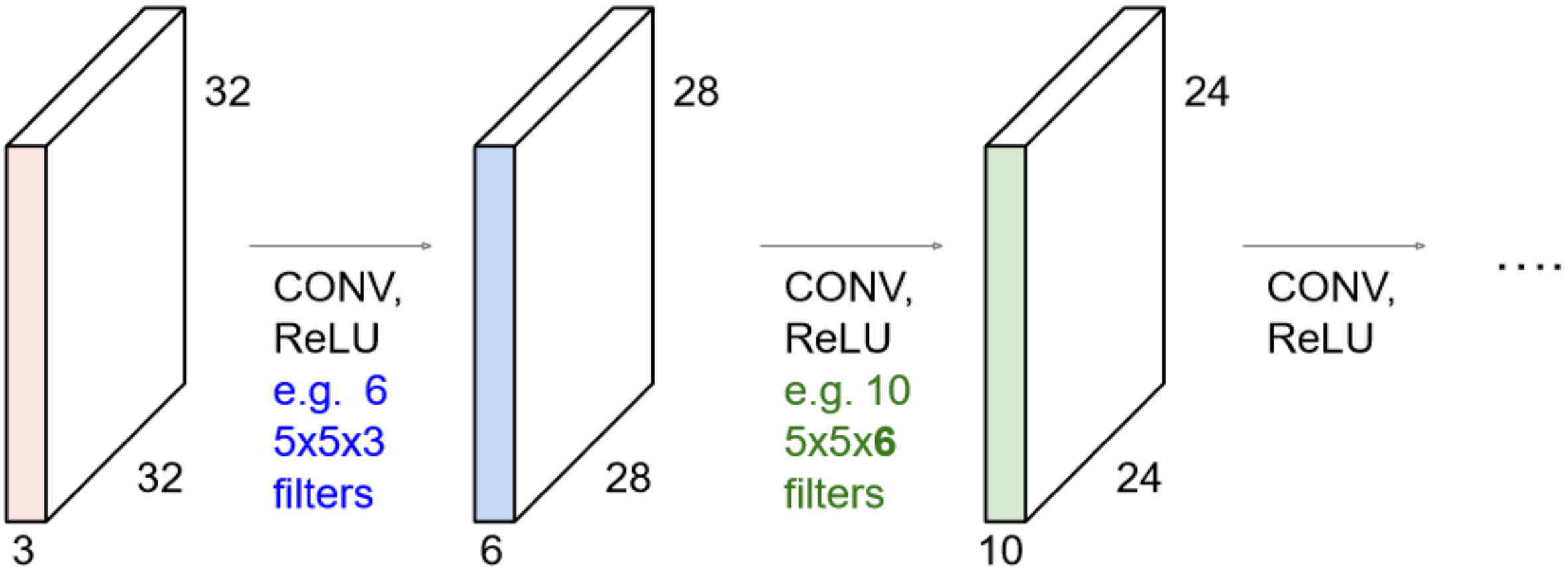
Convolutions: More detail

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

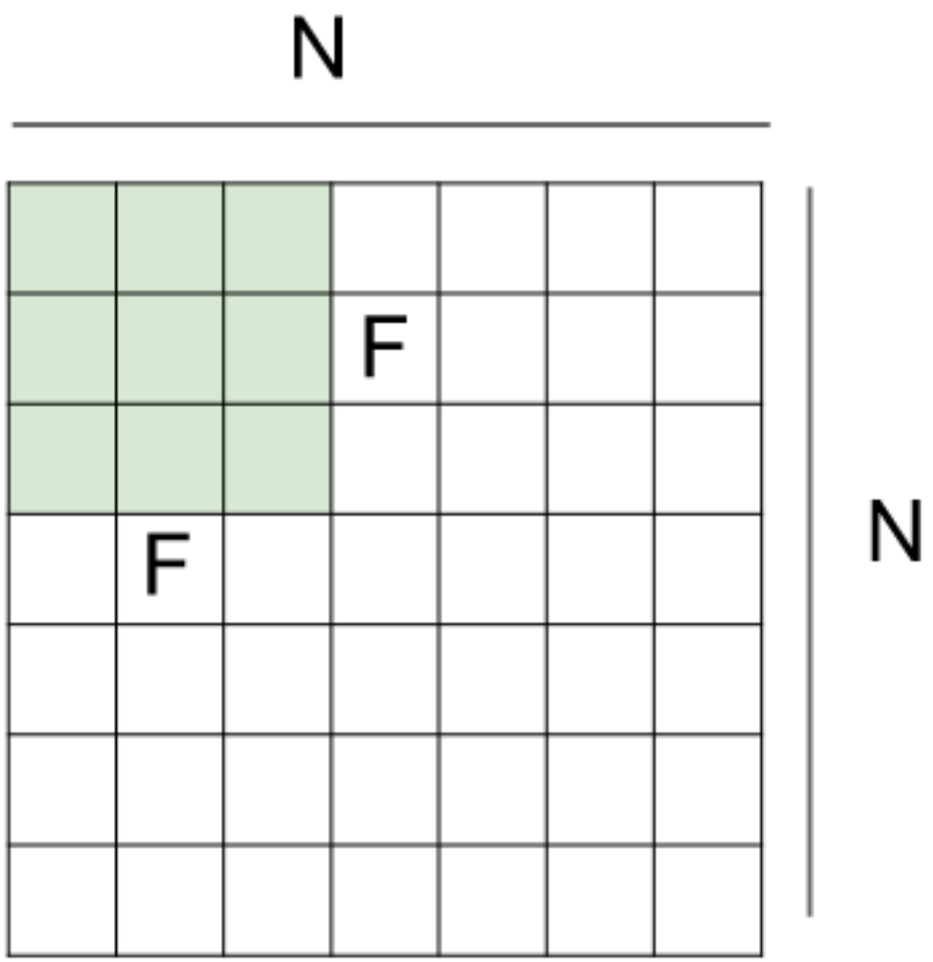


We stack these up to get a "new image" of size 28x28x6!

Convolutions: More detail



Convolutions: More detail

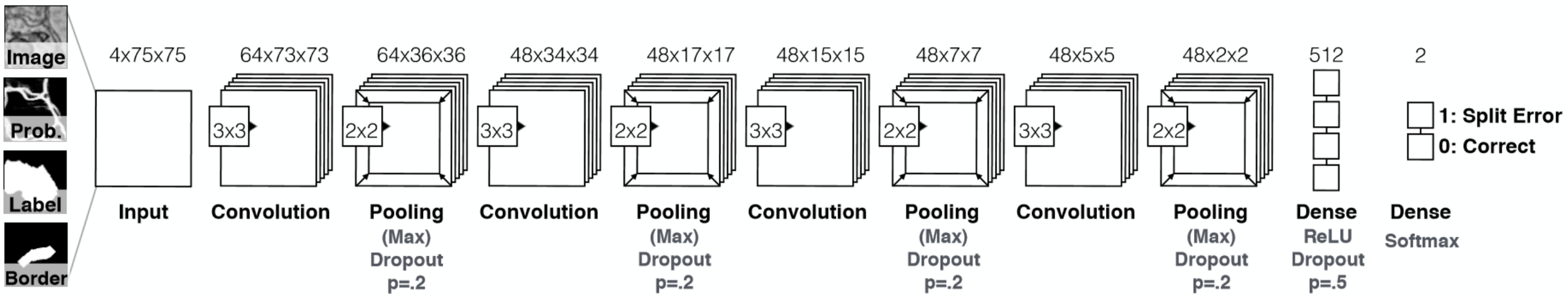


Output size:
(N - F) / stride + 1

Our connectomics diagram

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

Input
75x75x4



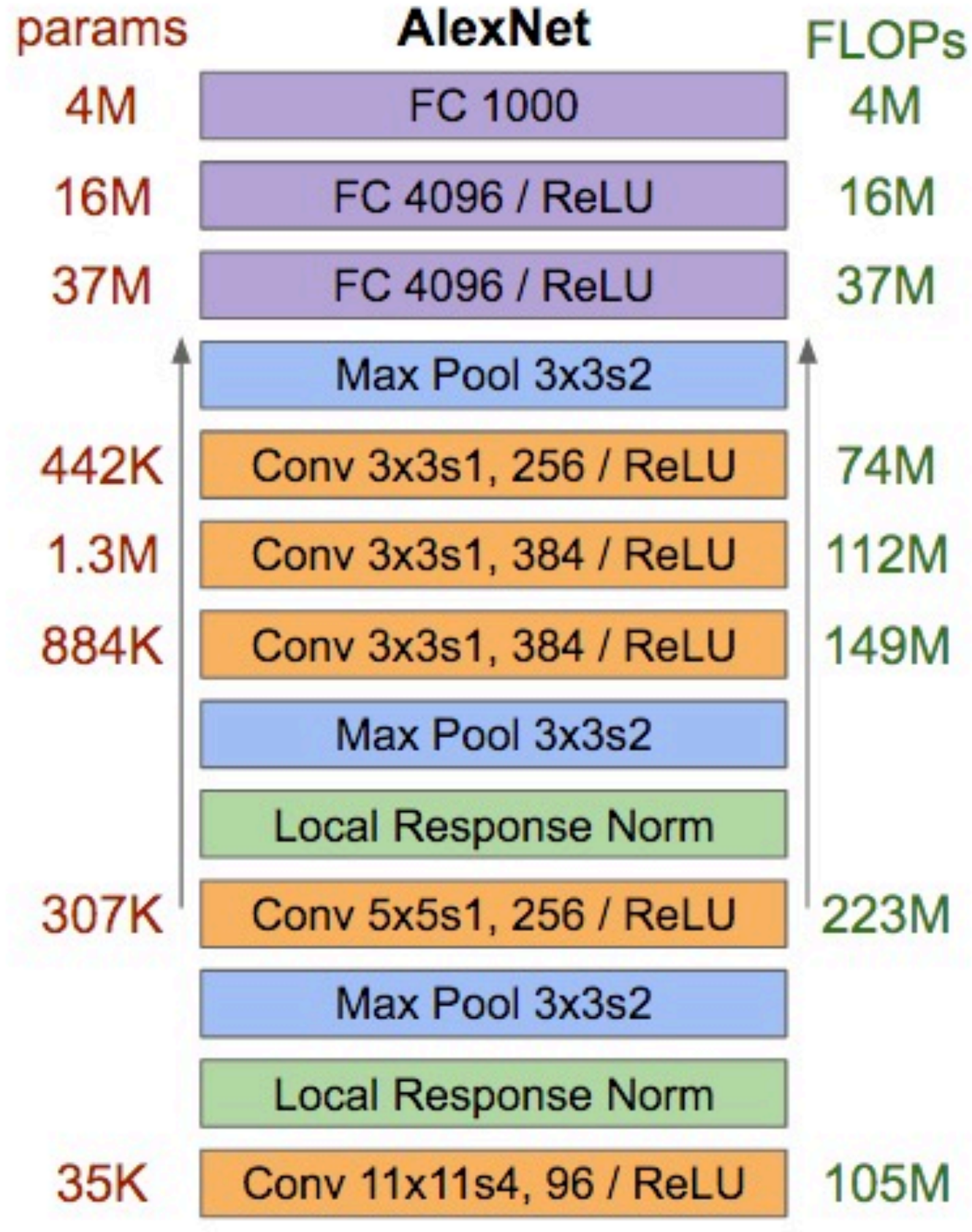
Conv 1	Conv 2	Conv 3	Conv 4
3x3x4	3x3x64	3x3x48	3x3x48
64 filters	48 filters	48 filters	48 filters

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

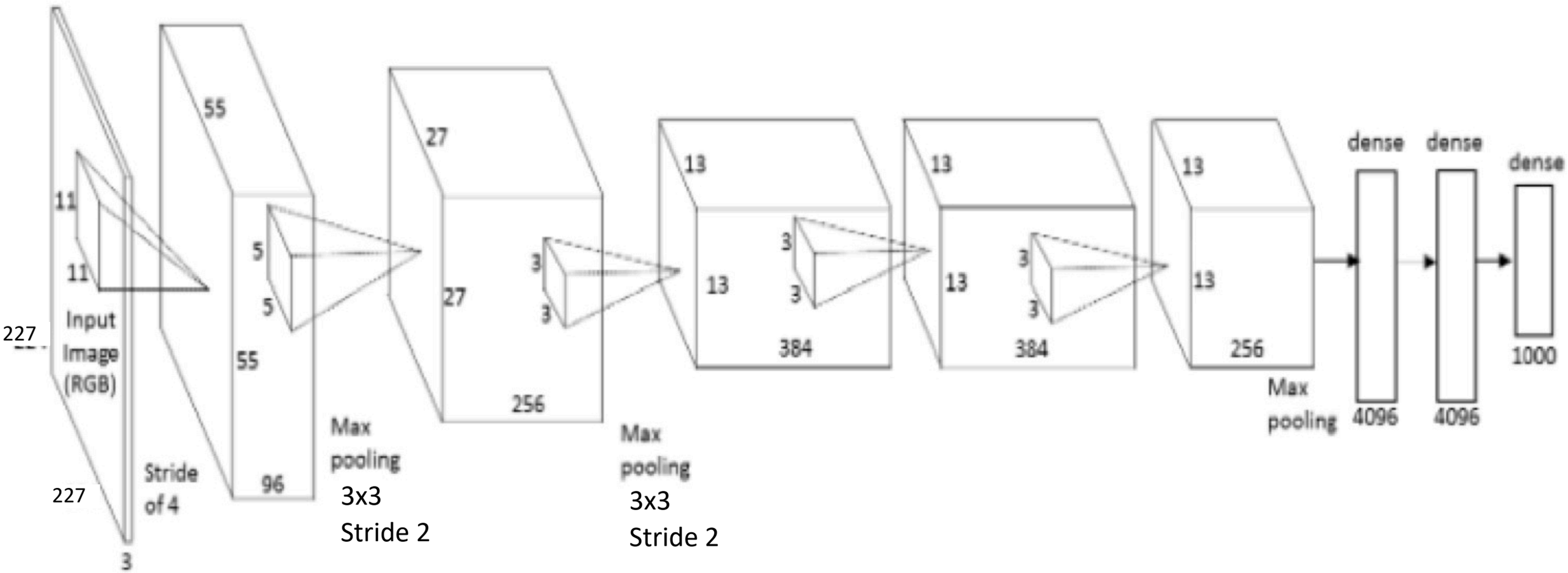
FLOP： Floating-point operations per second



[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 96
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

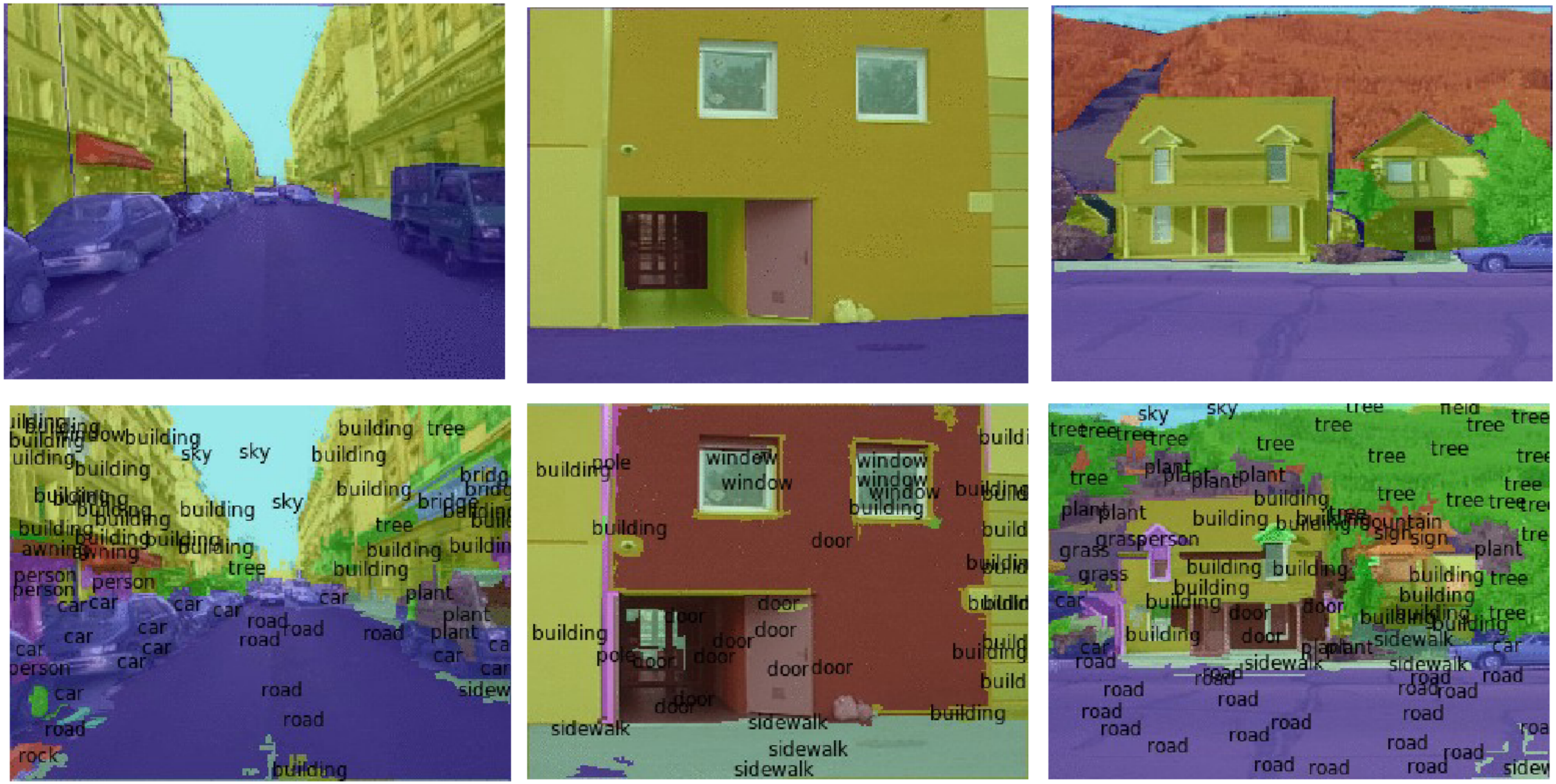
CONV NETS: EXAMPLES

- 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

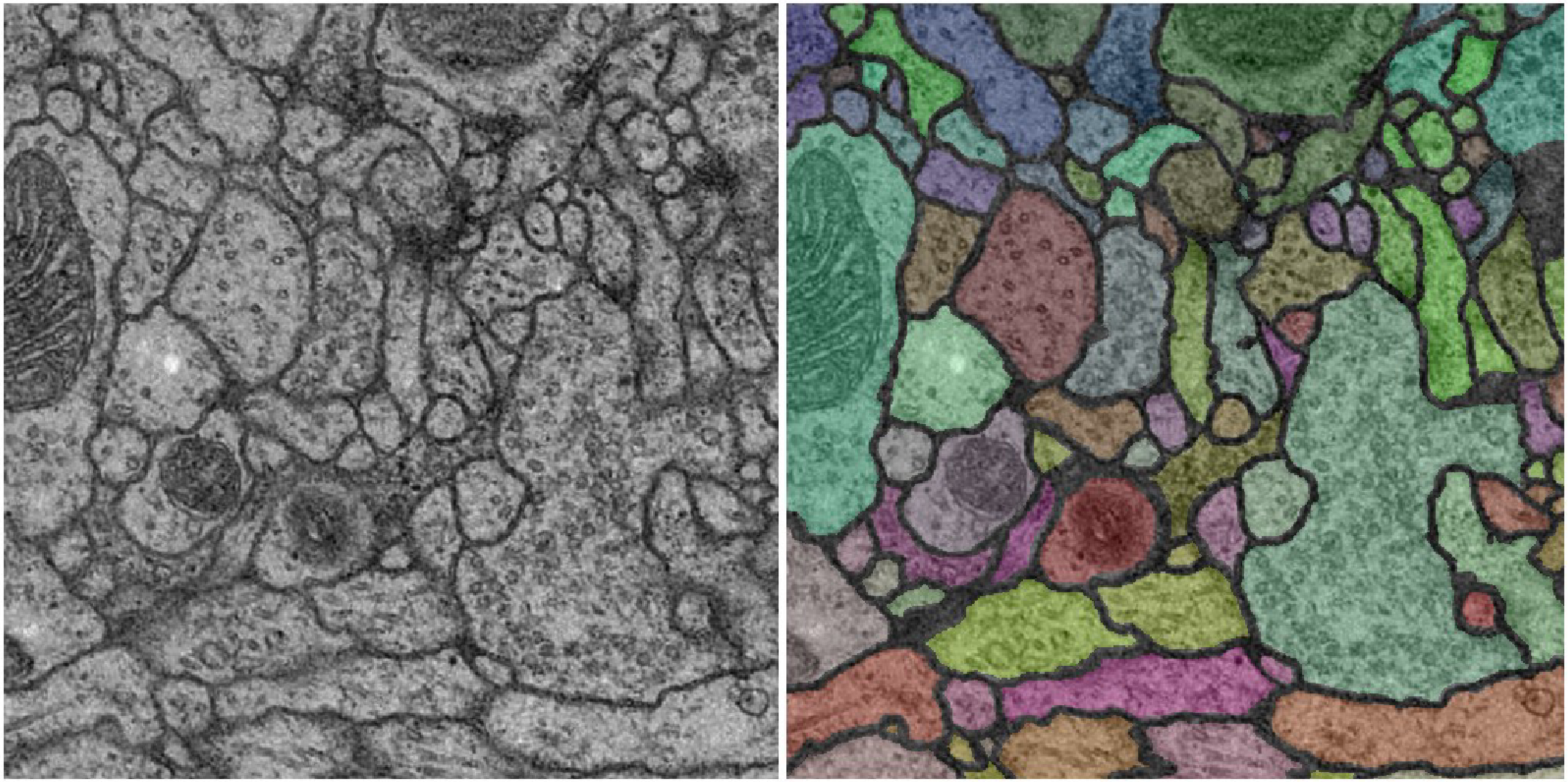
- Scene Parsing



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

- Action recognition from videos

Taylor et al. “Convolutional learning of spatio-temporal features” ECCV 2010
Karpathy et al. “Large-scale video classification with CNNs” CVPR 2014
Simonyan et al. “Two-stream CNNs for action recognition in videos” arXiv 2014

CONV NETS: EXAMPLES

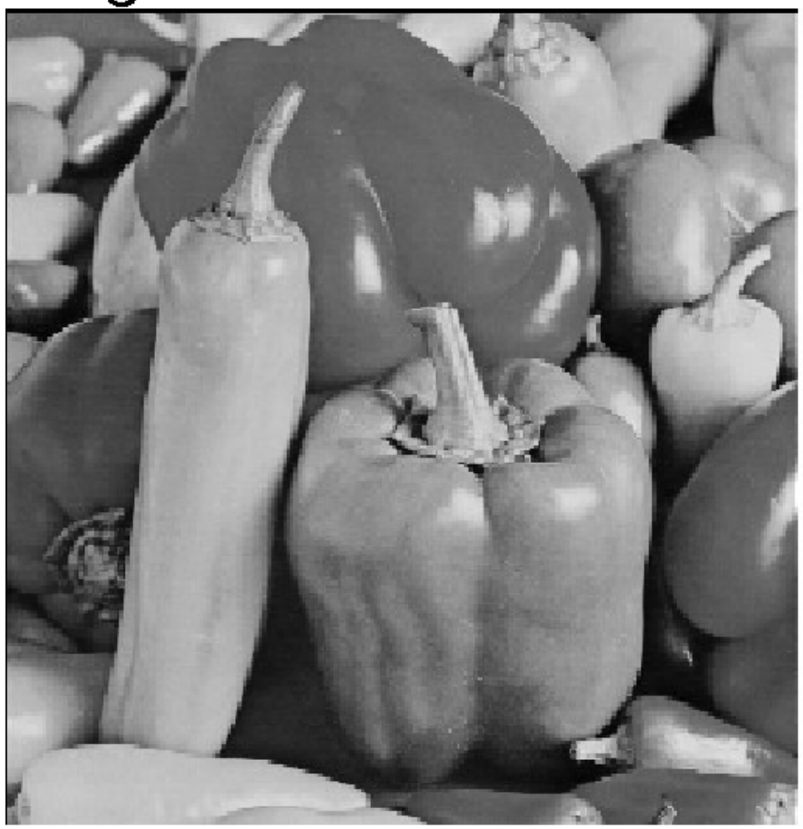
- **Robotics**

Sermanet et al. “Mapping and planning ...with long range perception” IROS 2008

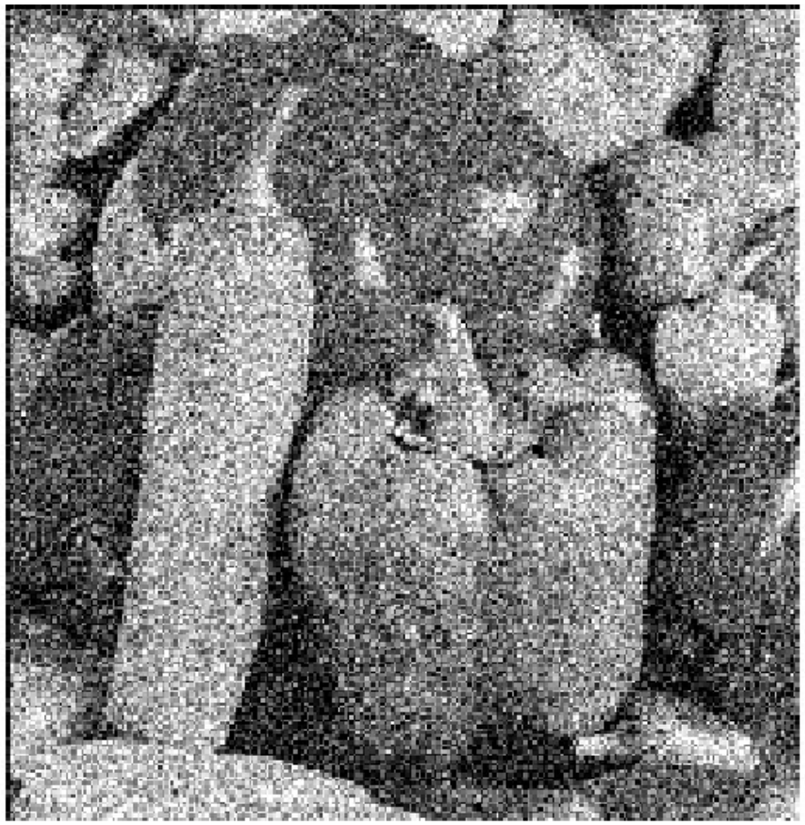
CONV NETS: EXAMPLES

- **Denoising**

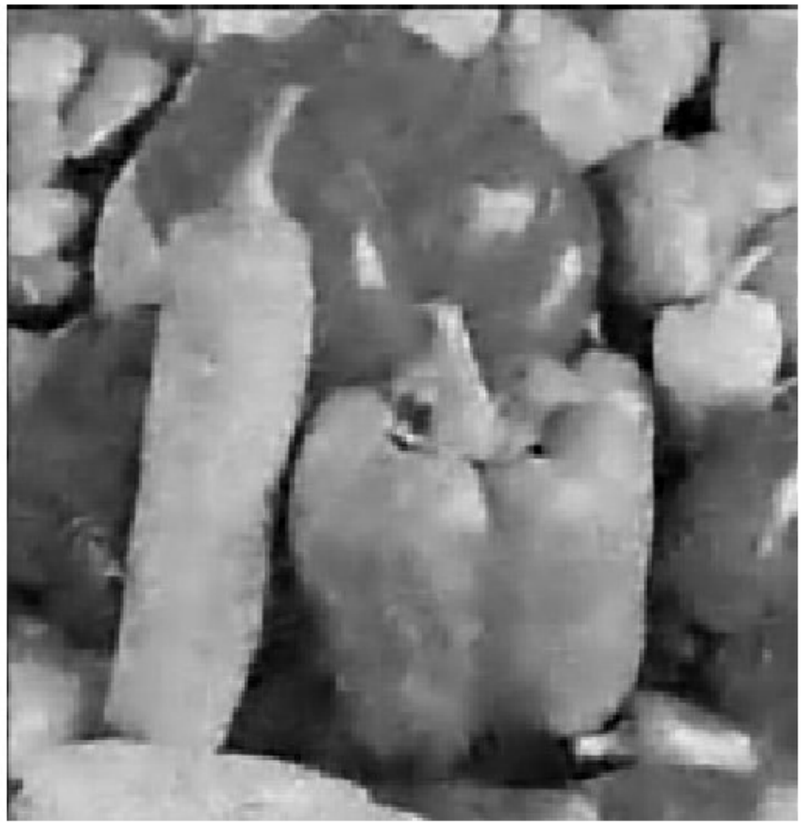
original



noised



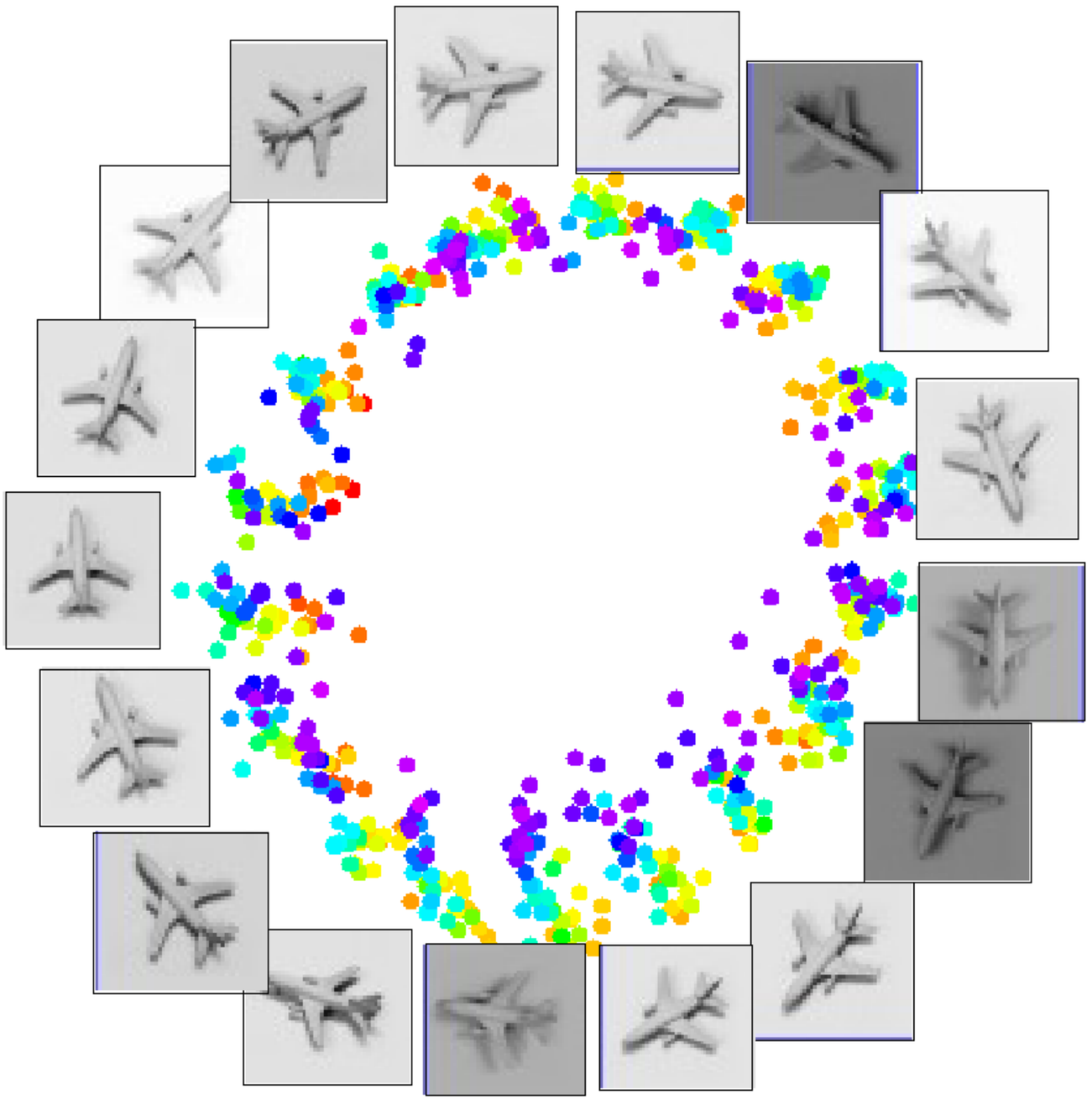
denoised



Burger et al. “Can plain NNs compete with BM3D?” CVPR 2012

CONV NETS: EXAMPLES

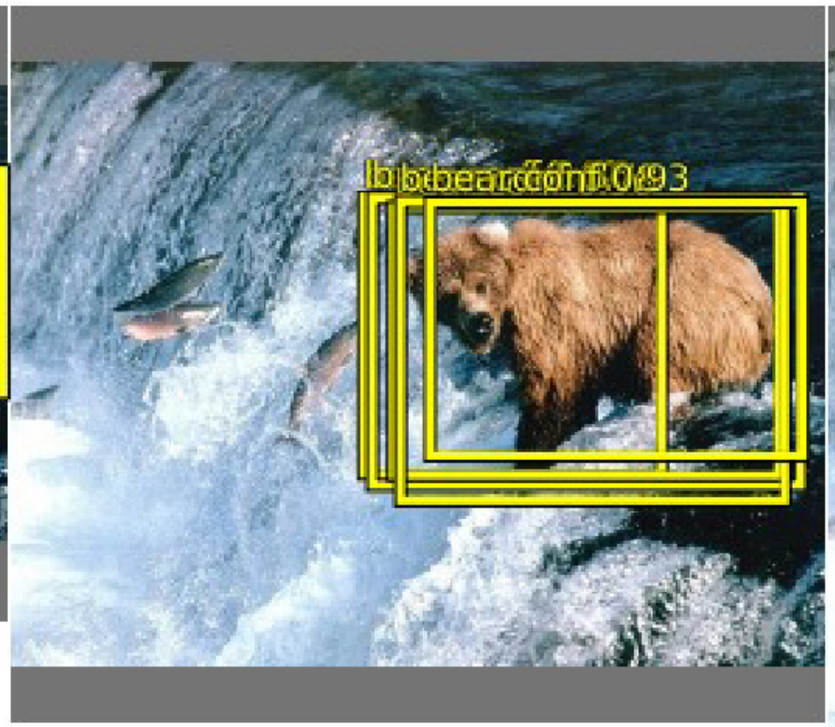
- Dimensionality reduction / learning embeddings

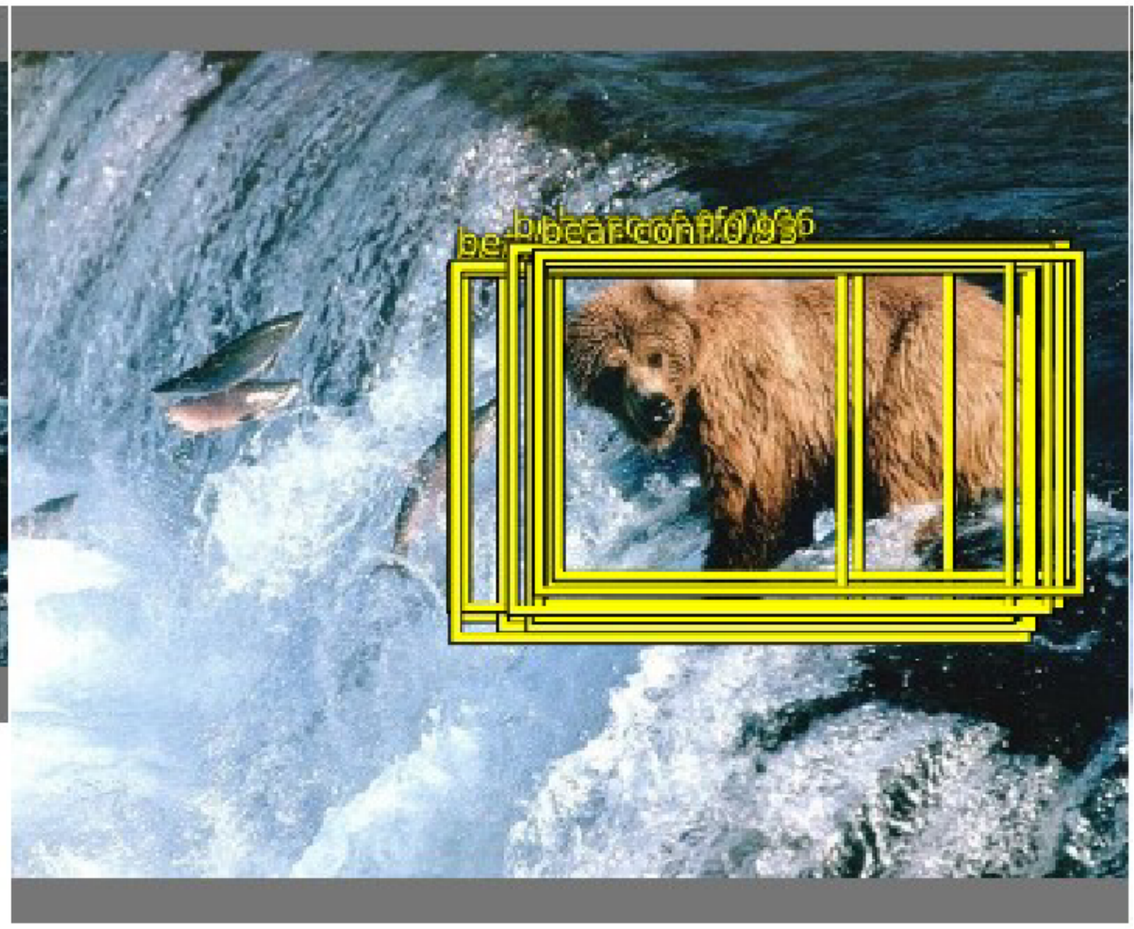


Hadsell et al. “Dimensionality reduction by learning an invariant mapping” CVPR 2006

CONV NETS: EXAMPLES

- **Object detection**

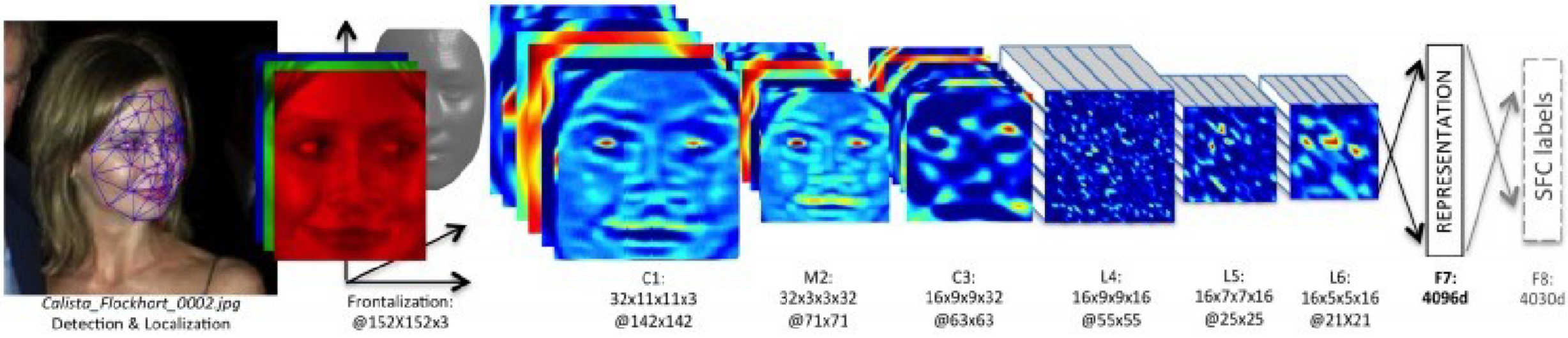




Sermanet et al. “OverFeat: Integrated recognition, localization, ...” arxiv 2013
Girshick et al. “Rich feature hierarchies for accurate object detection...” arxiv 2013
Szegedy et al. “DNN for object detection” NIPS 2013

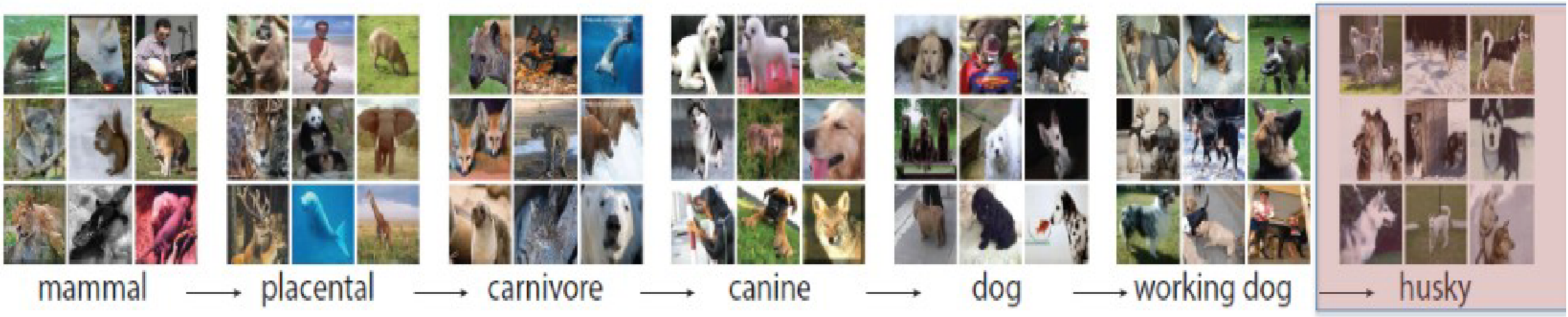
CONV NETS: EXAMPLES

- Face Verification & Identification



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

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

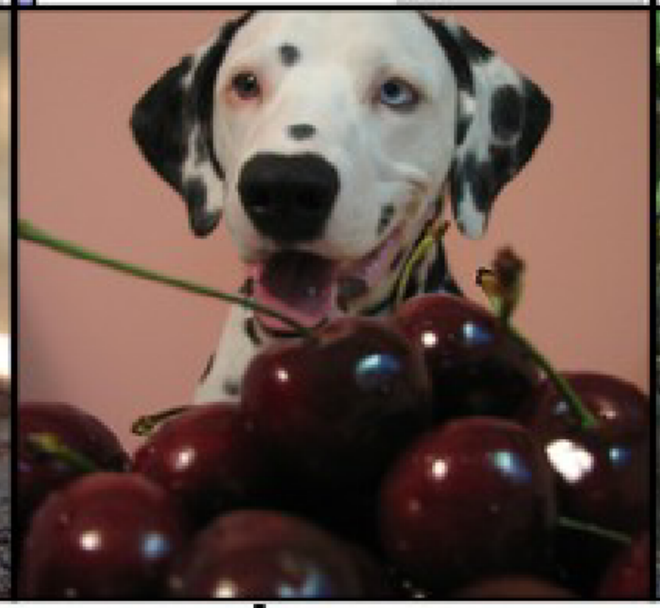
- leopard
- jaguar
- cheetah
- snow leopard
- Egyptian cat

grille

- convertible
- grille
- pickup
- beach wagon
- fire engine

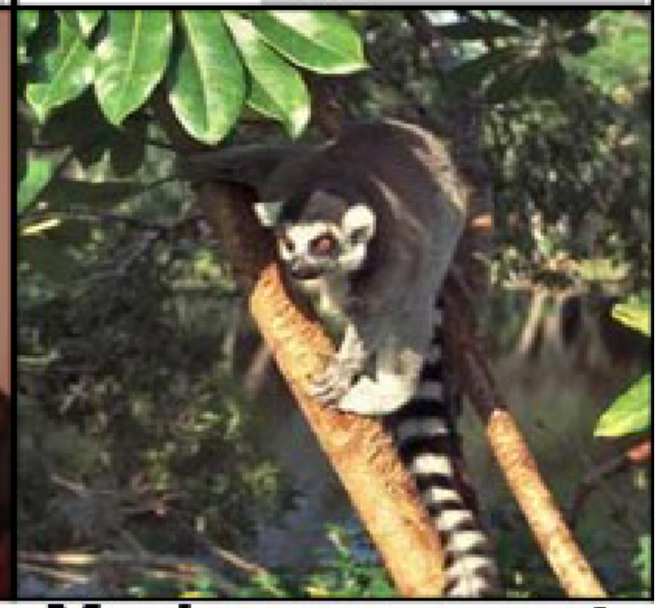
mushroom

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



cherry

- dalmatian
- grape
- elderberry
- ffordshire bullterrier
- currant



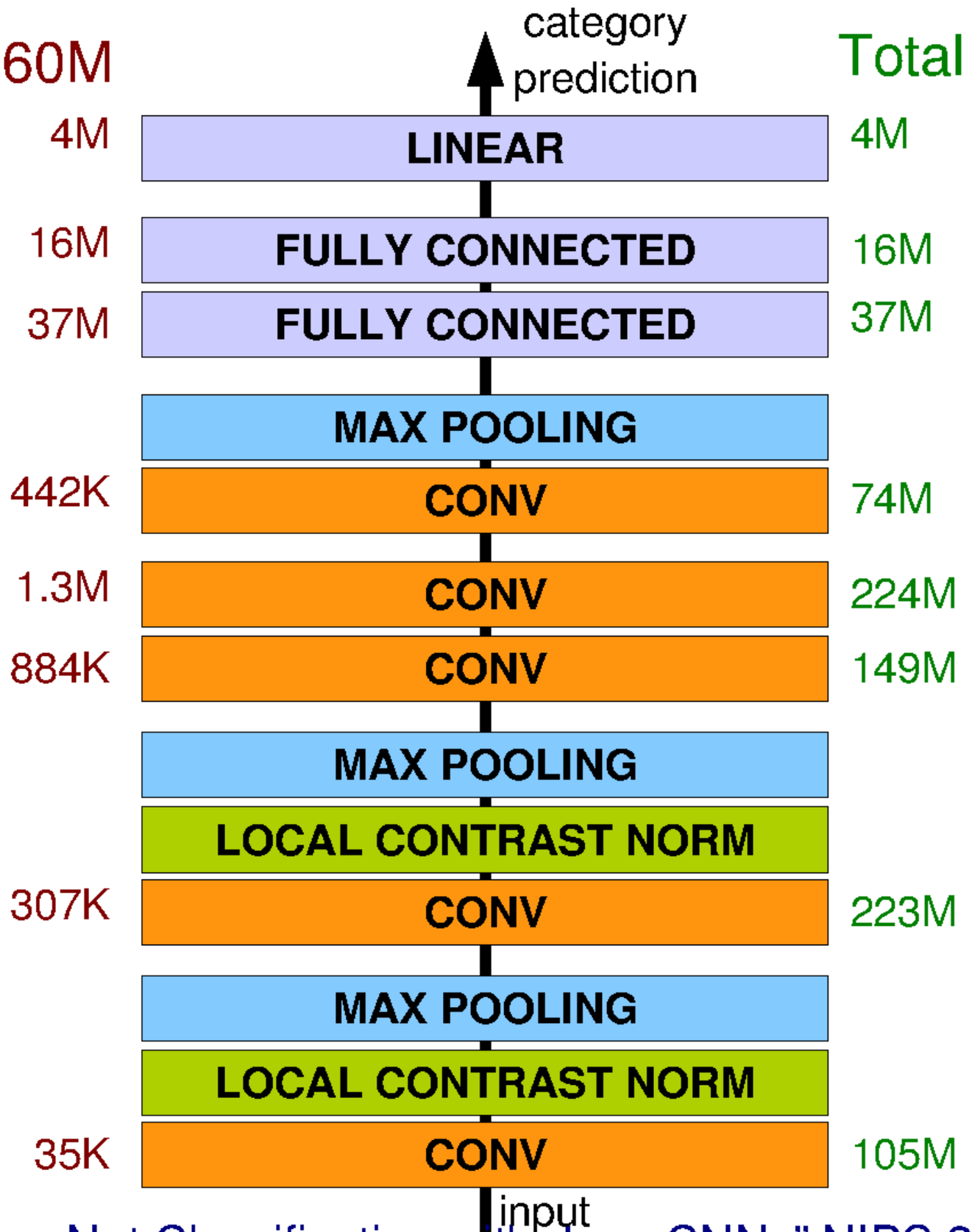
Madagascar cat

- squirrel monkey
- spider monkey
- titi
- indri
- howler monkey

Architecture for Classification

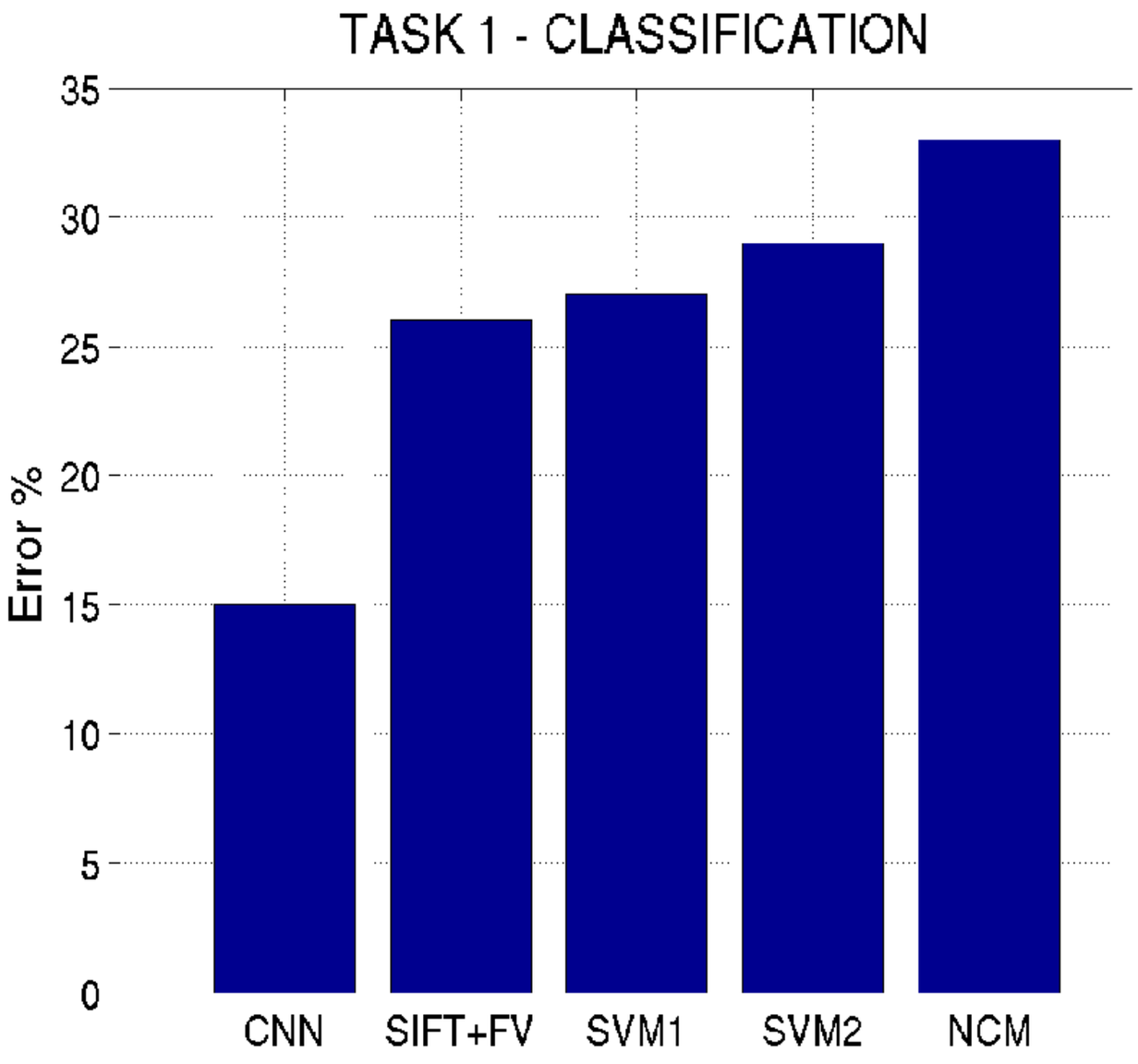
Total nr. params: 60M

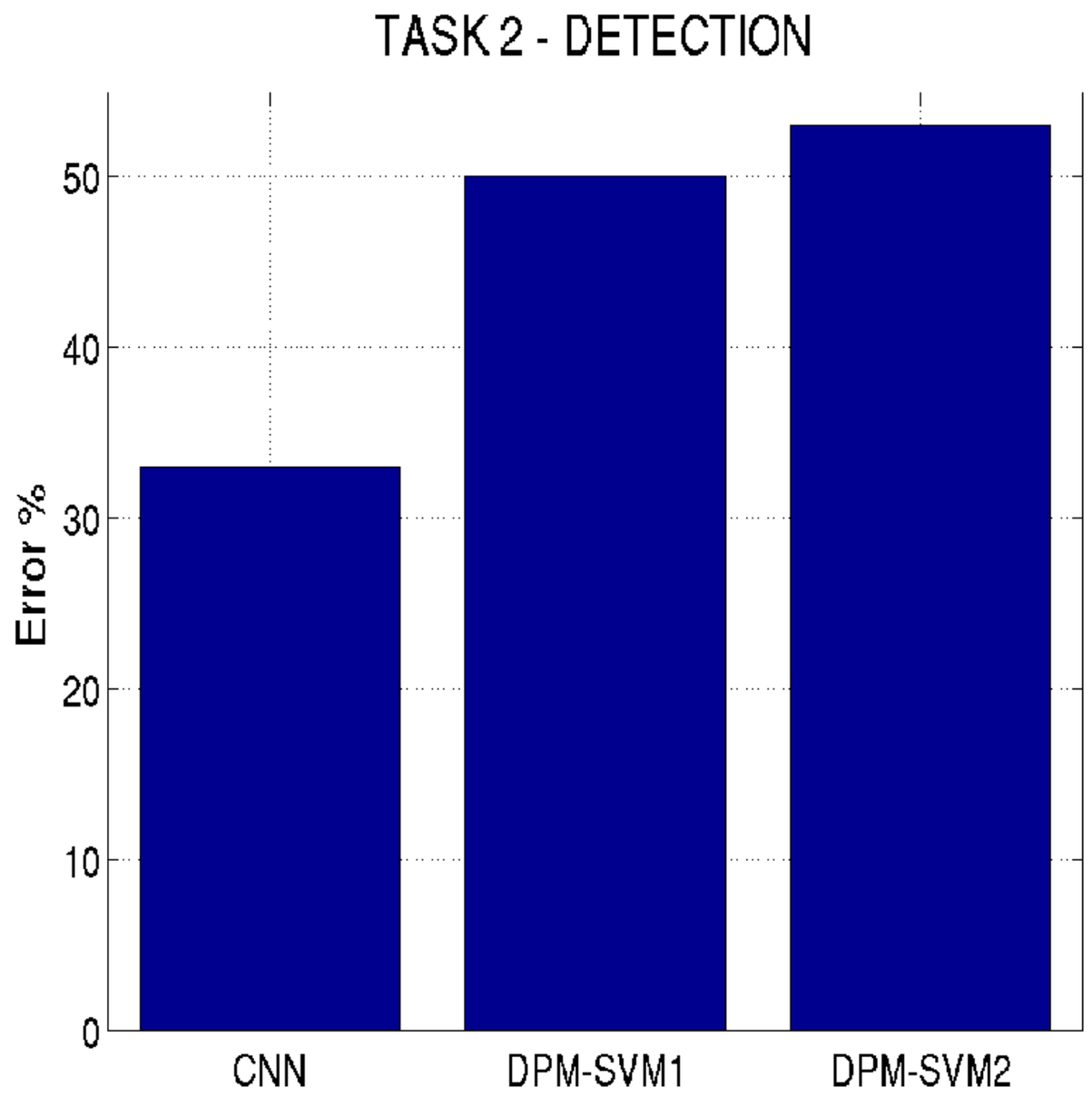
Total nr. flops: 832M



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

Results: ILSVRC 2012





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

Wait, why isn't it called a correlation neural network?

It could be.
Deep learning libraries actually implement correlation.

Correlation relates to convolution via a 180deg rotation of the kernel. When we *learn* kernels, we could easily learn them flipped.

Associative property of convolution ends up not being important to our application, so we just ignore it.

[p.323, Goodfellow]

More ConvNet explanations

- https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

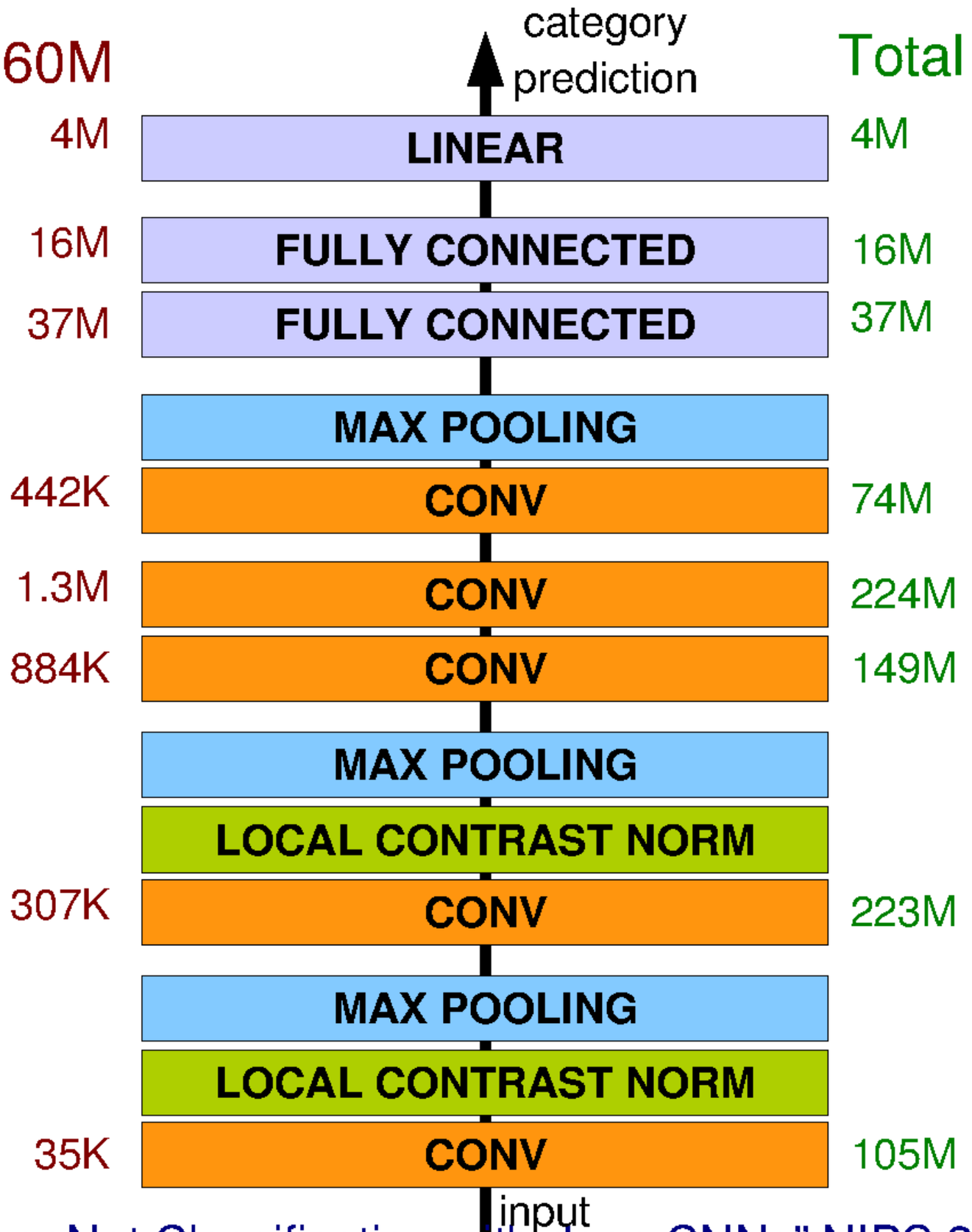
Network Architecture: big space of designs

- But we still don’t even know how many layers we need.

Architecture for Classification

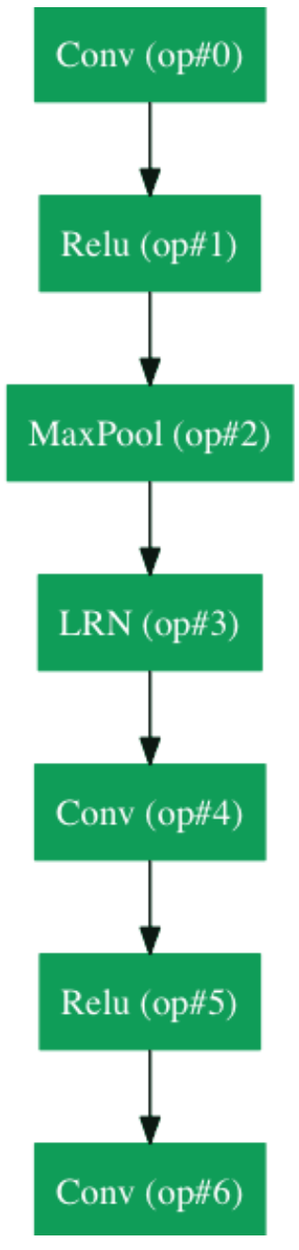
Total nr. params: 60M

Total nr. flops: 832M



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

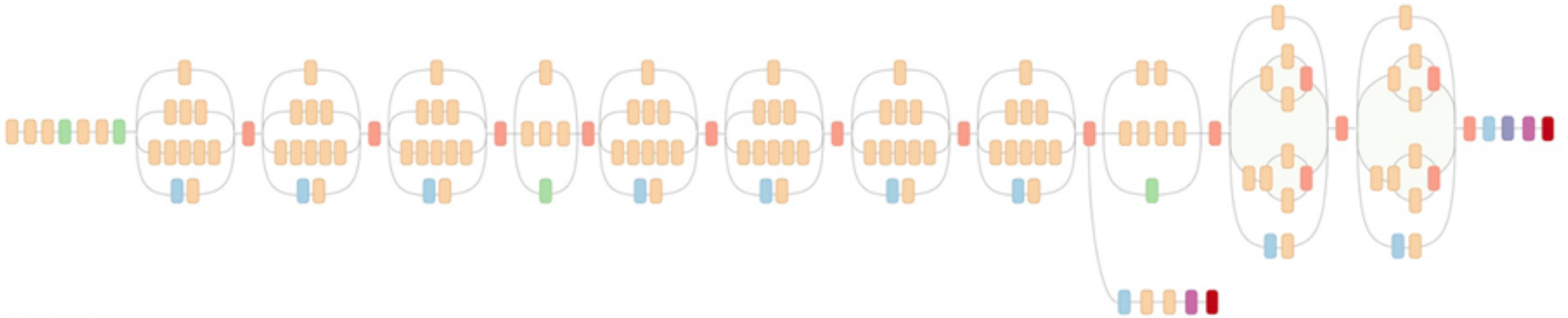
Google LeNet (2014)



22 layers

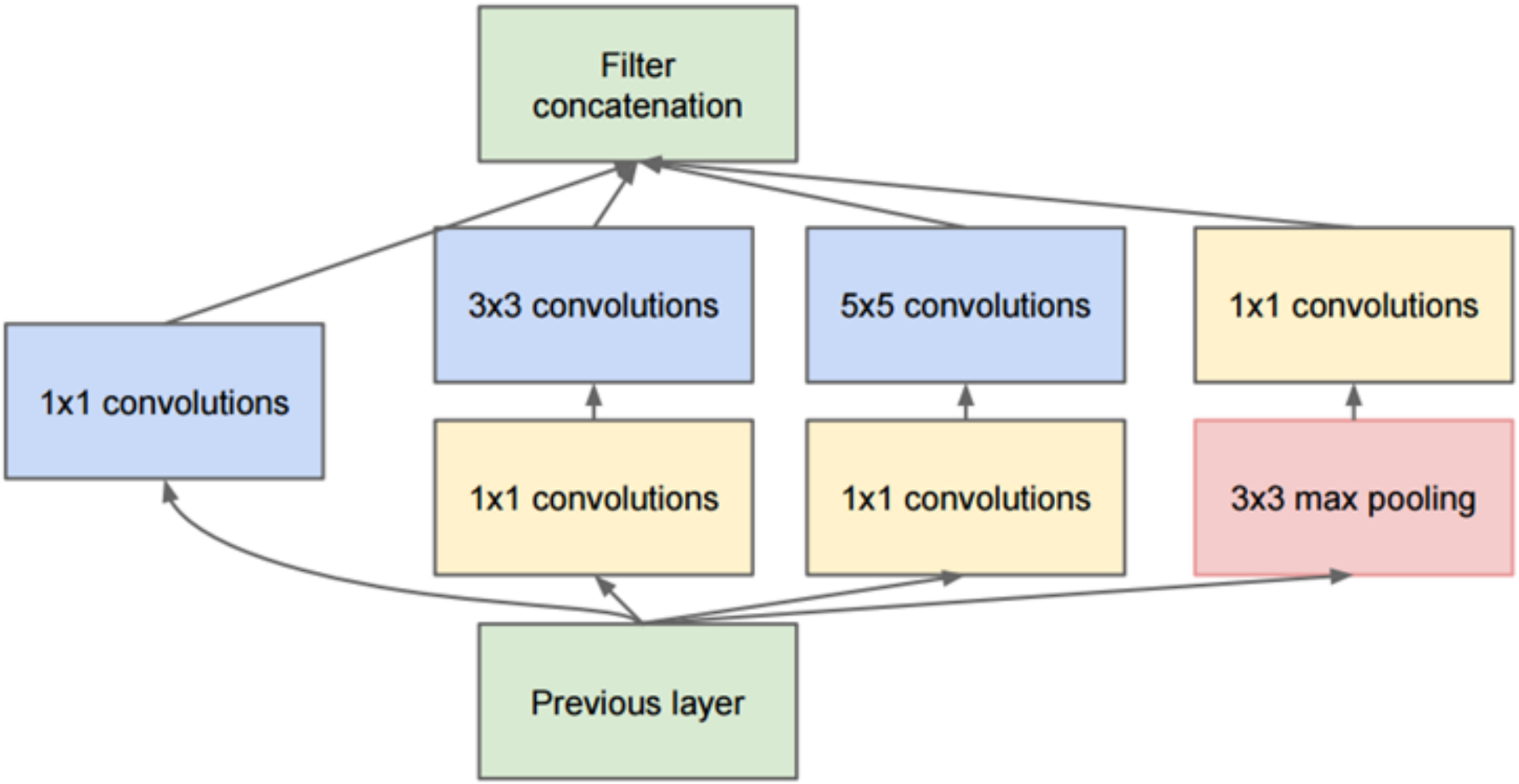
6.67% error
ImageNet top 5

Inception!



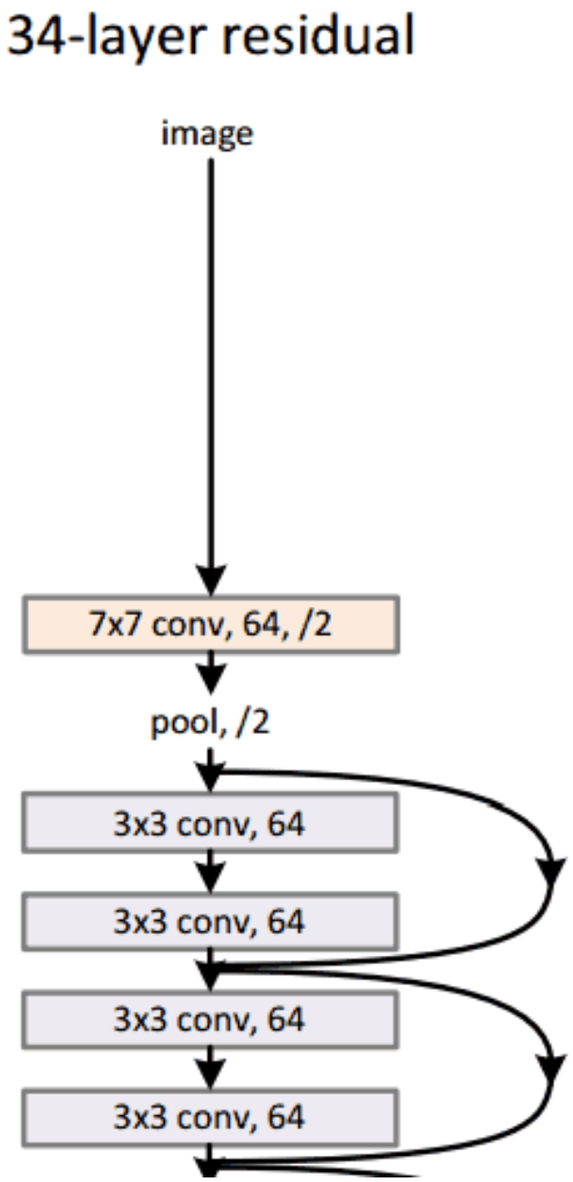
Another view of GoogLeNet's architecture.

Parallel layers



Full Inception module

ResNet (He et al., 2015)

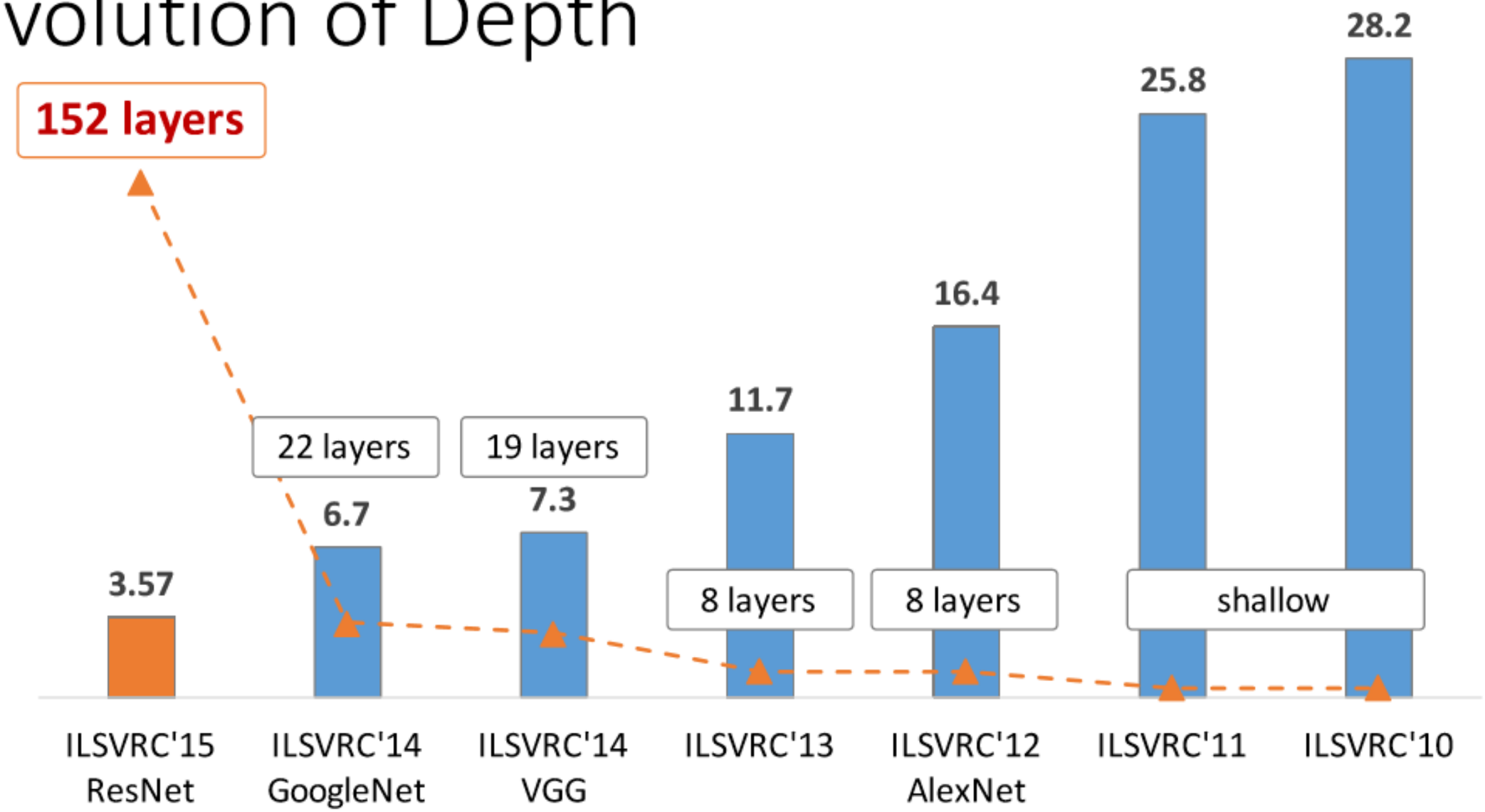


ResNet won ILSVRC 2015 with a top-5 error rate of 3.6%

Depending on their skill and expertise, humans generally hover around a 5-10% error.

~~Superhuman performance!~~
But the task is arguably not well defined.

Revolution of Depth



ImageNet Classification top-5 error (%)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)

VGG, 19 layers
(ILSVRC 2014)

ResNet, 152 layers
(ILSVRC 2015)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

Vanishing/exploding gradient problem

Backpropagation:

- Compute gradient update for every neuron which was involved in the output across layers

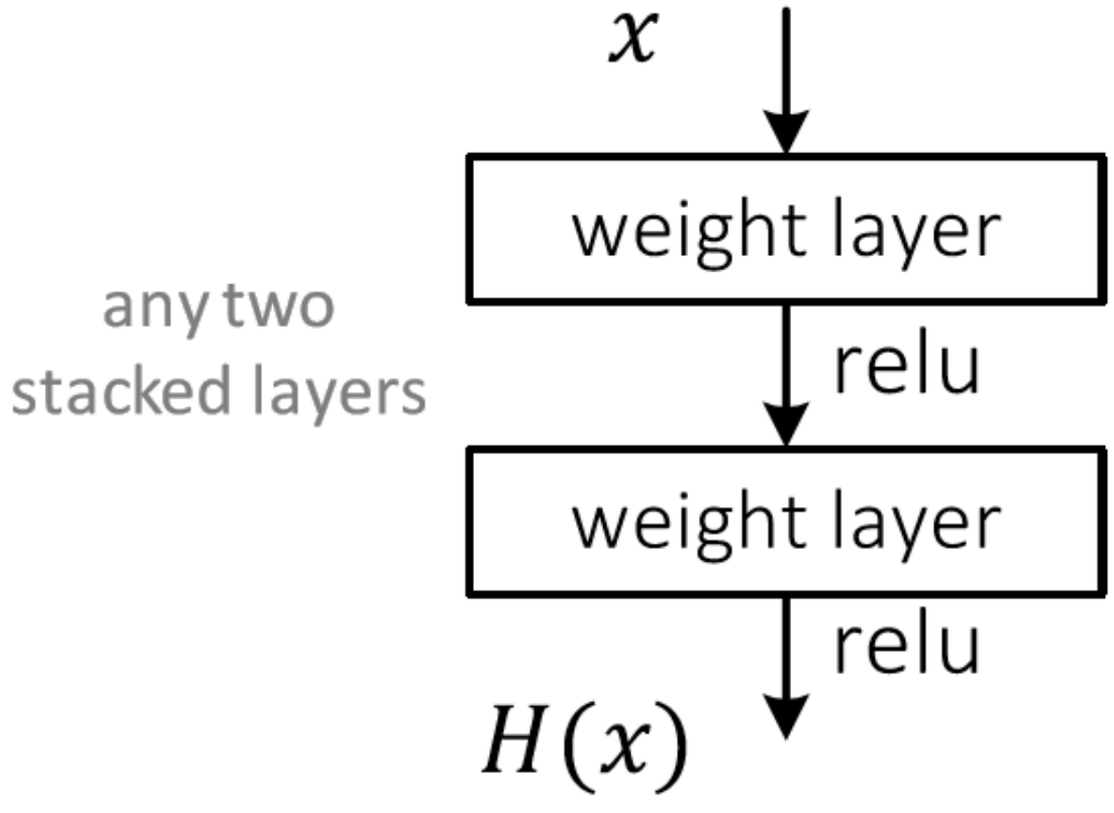
Involves chaining partial derivates over many layers!

- If derivative < 1, gradient gets smaller and smaller as we go deeper and deeper -> *vanishing gradients!*
- If derivative > 1, gradient gets larger and larger as we go deeper and deeper -> *exploding gradients!*

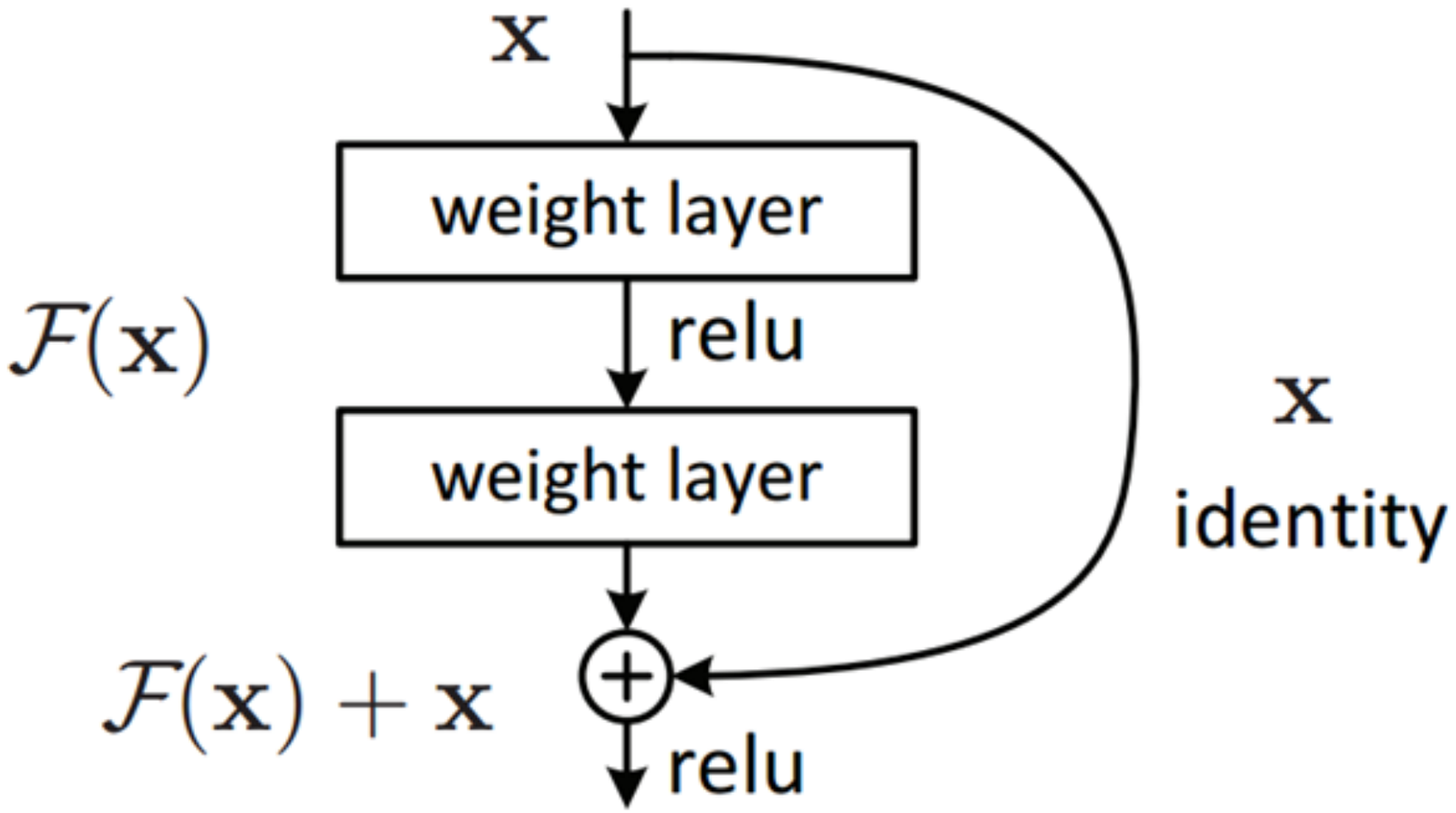
Regular net

$H(x)$ is any desired mapping,

hope the 2 weight layers fit $H(x)$



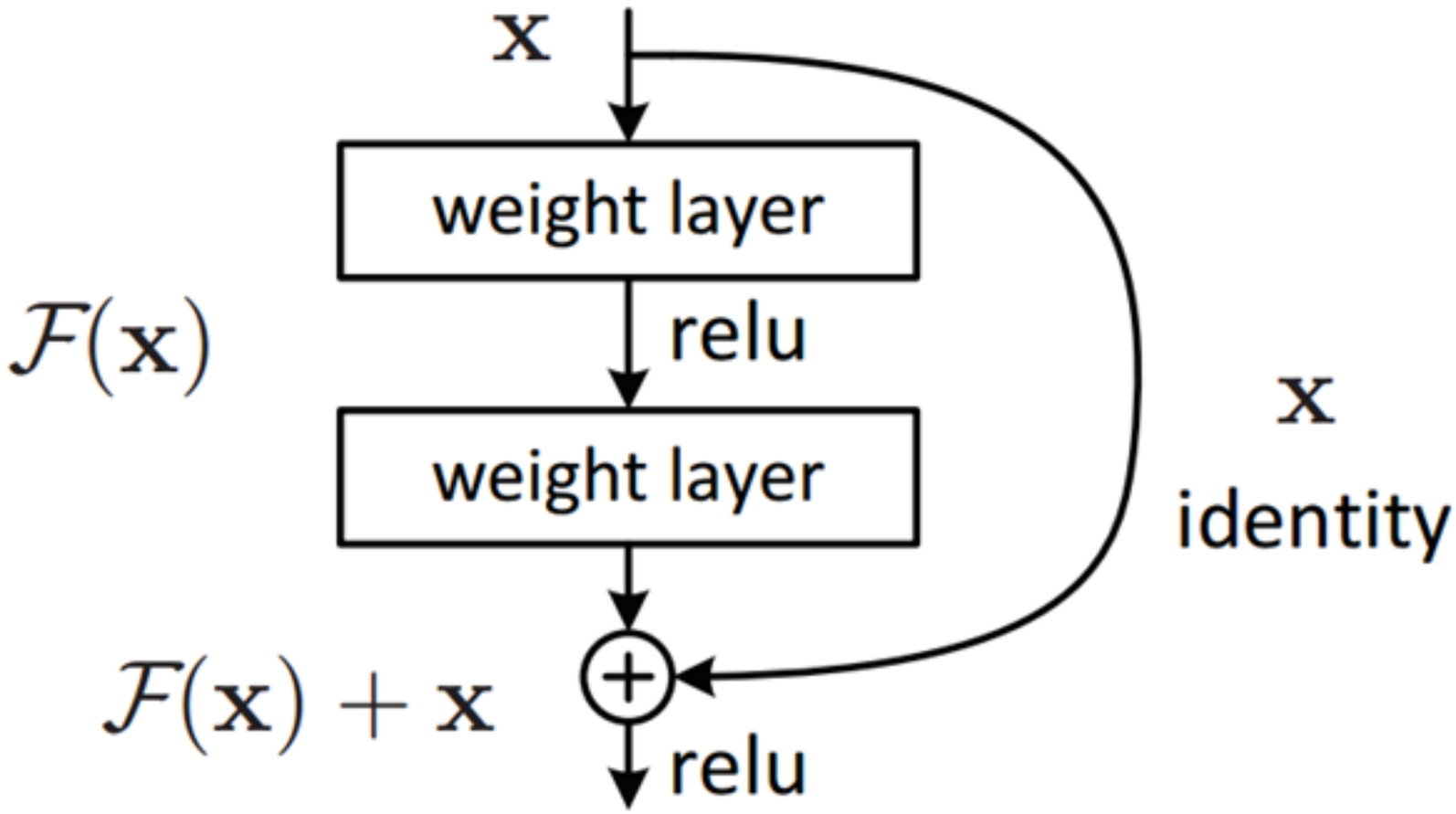
Residual Unit



A residual block

Residual Unit

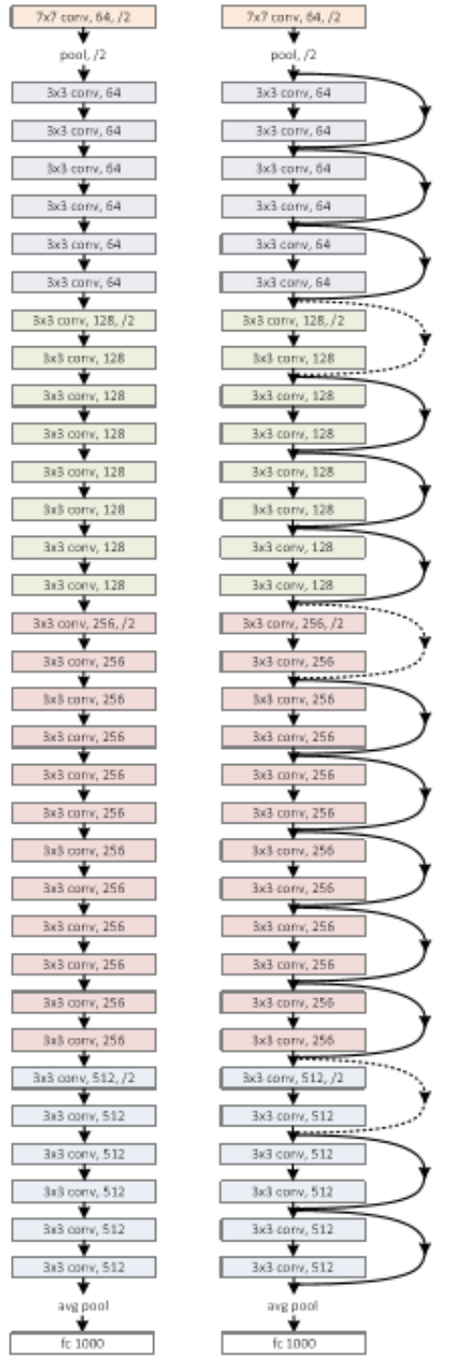
The inputs of a lower layer is made available to a node in a higher layer.



A residual block

Network “Design”

plain net

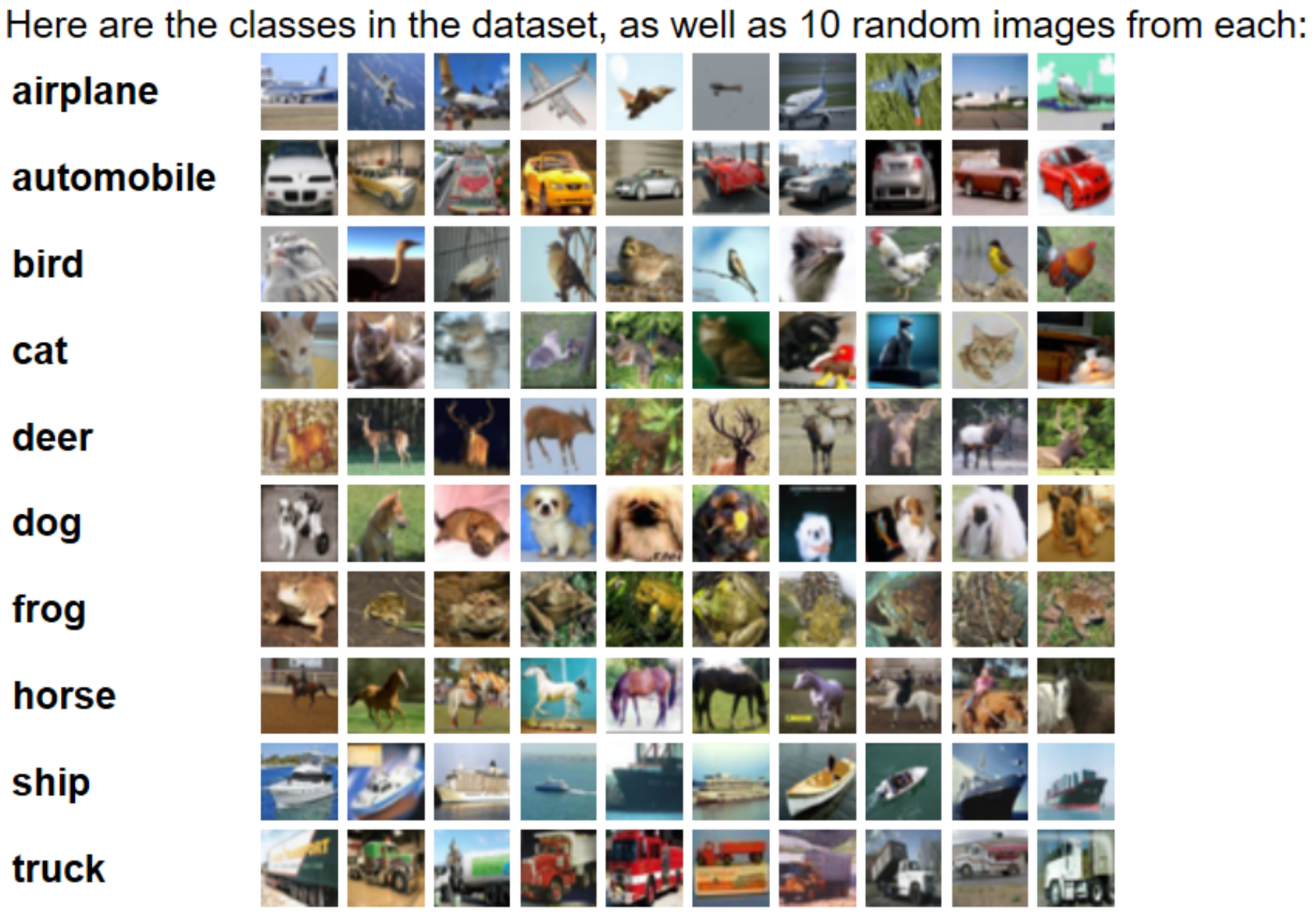


ResNet

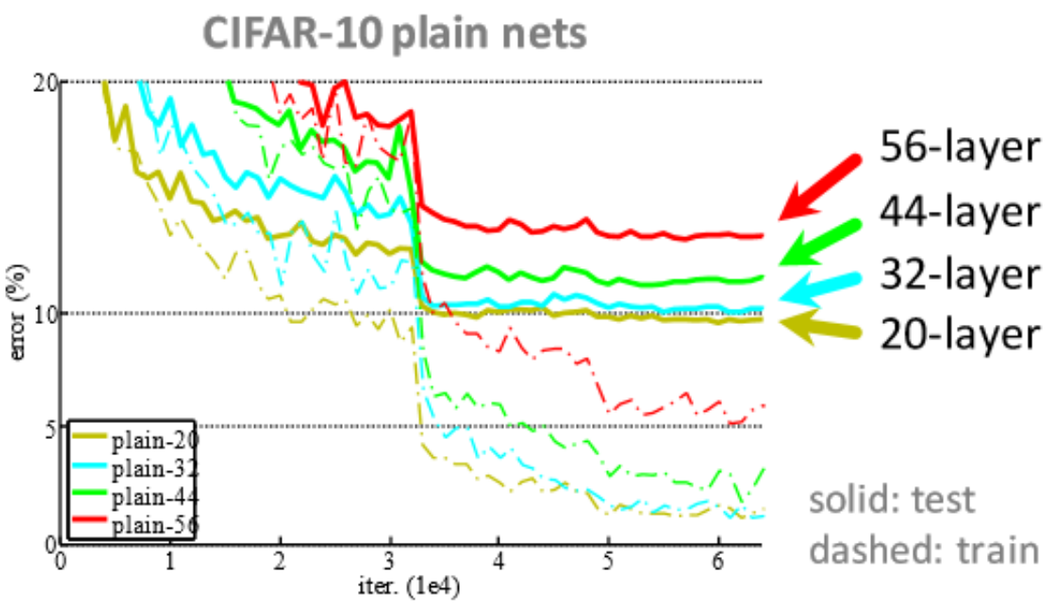
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. “Deep Residual Learning for Image Recognition”. CVPR 2016.

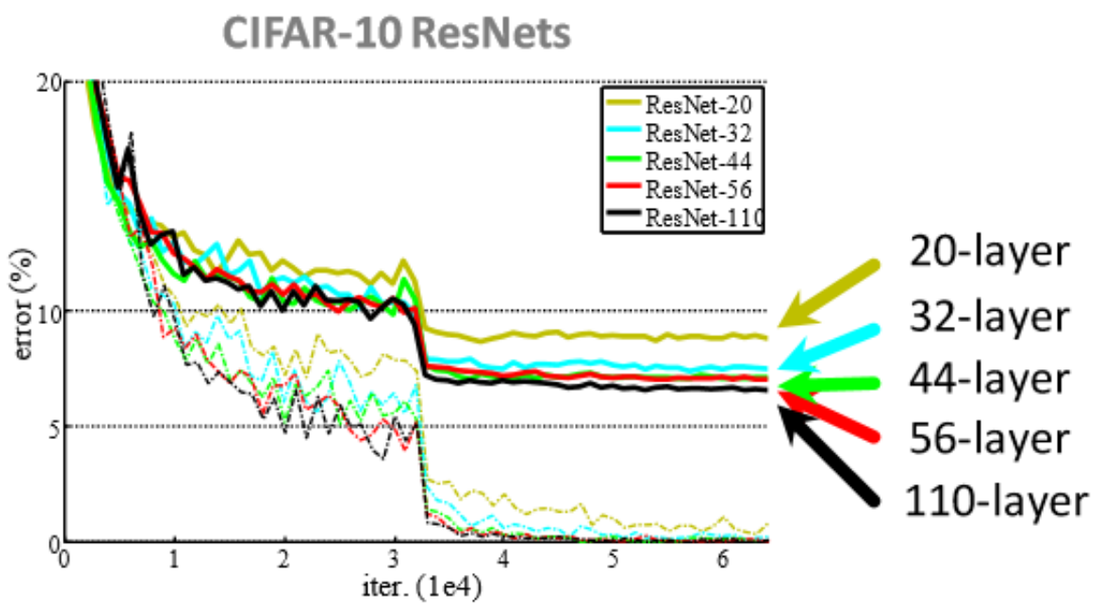
CIFAR-10

- 60,000 32x32 color images, 10 classes



CIFAR-10 experiments





- Deep ResNets can be trained without difficulties
- Deeper ResNets have **lower training error**, and also lower test error

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

Visualization

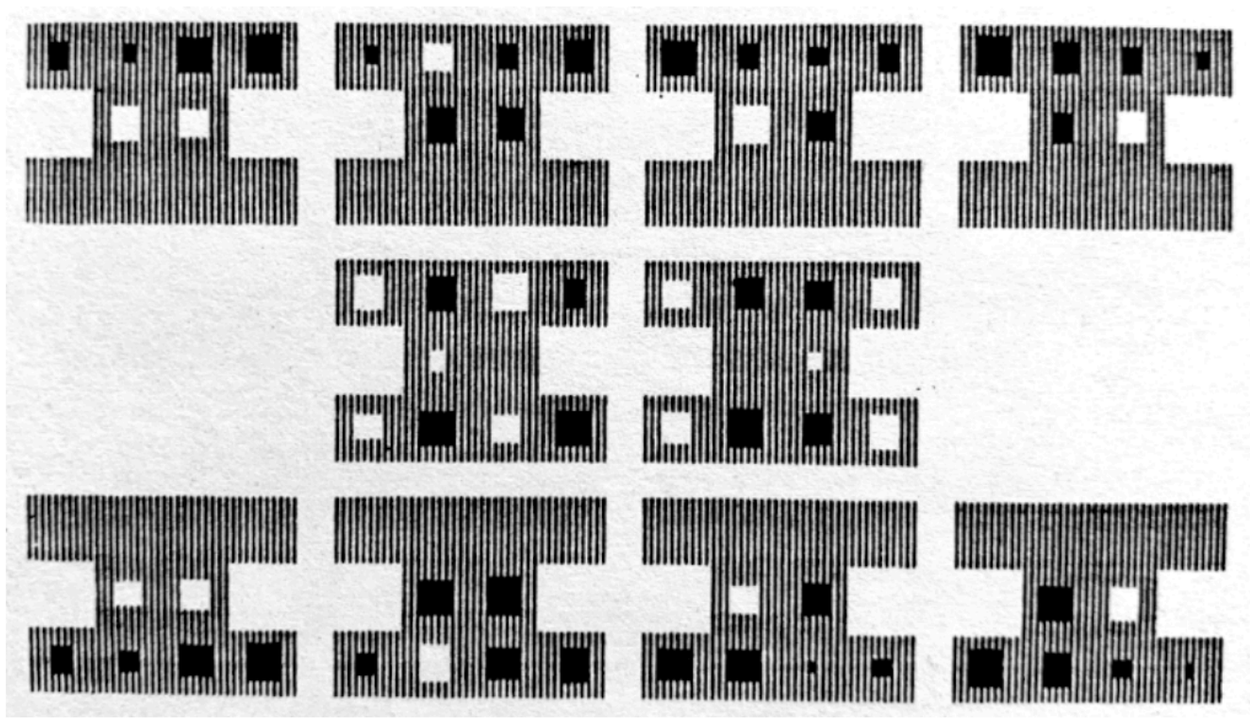


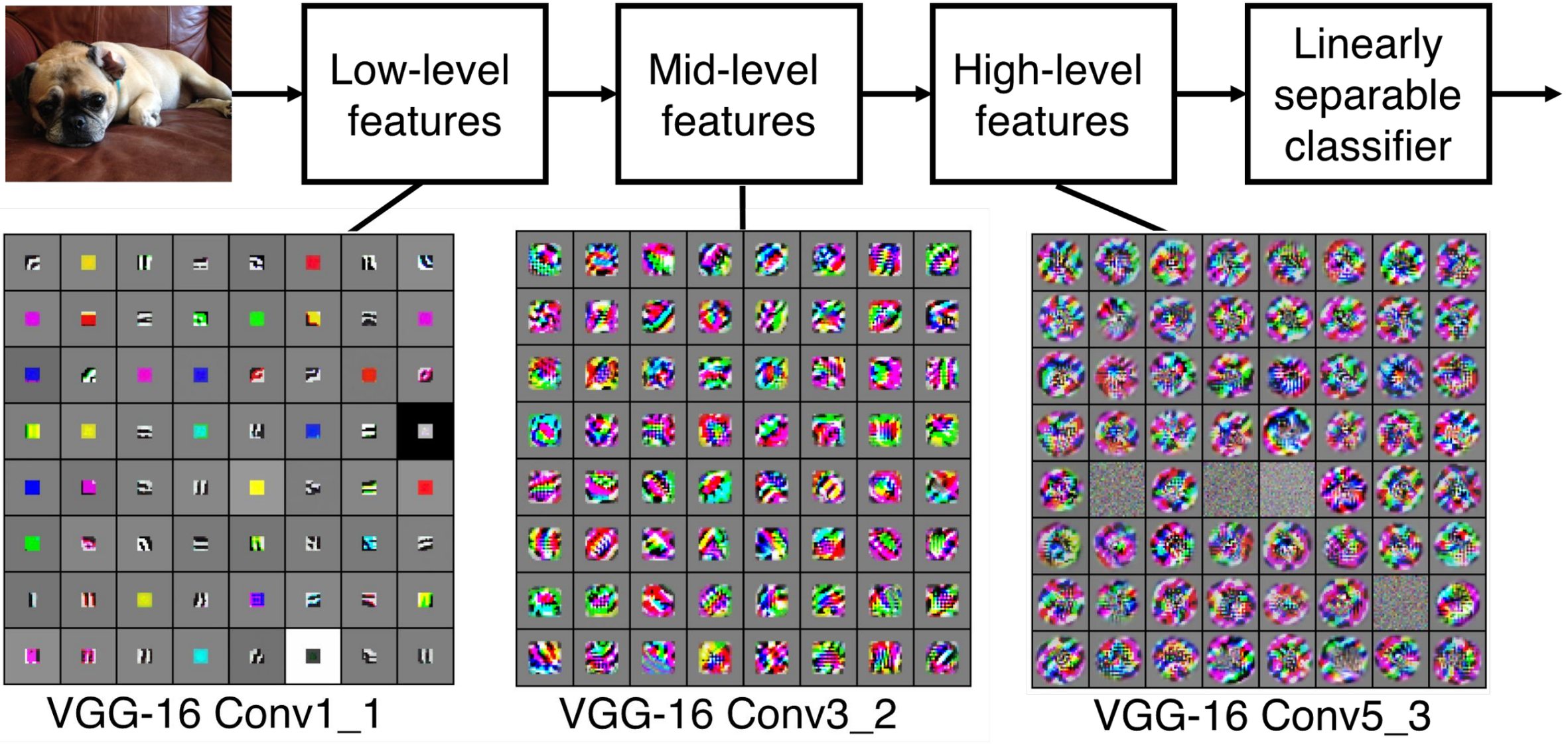
Figure 5.46 A Hinton diagram showing the weights connecting the units in a a three layer neural network, courtesy of Geoffrey Hinton. The size of each small box indicates the magnitude of each weight and its color (black or white) indicates the sign.

Visualization

Preview

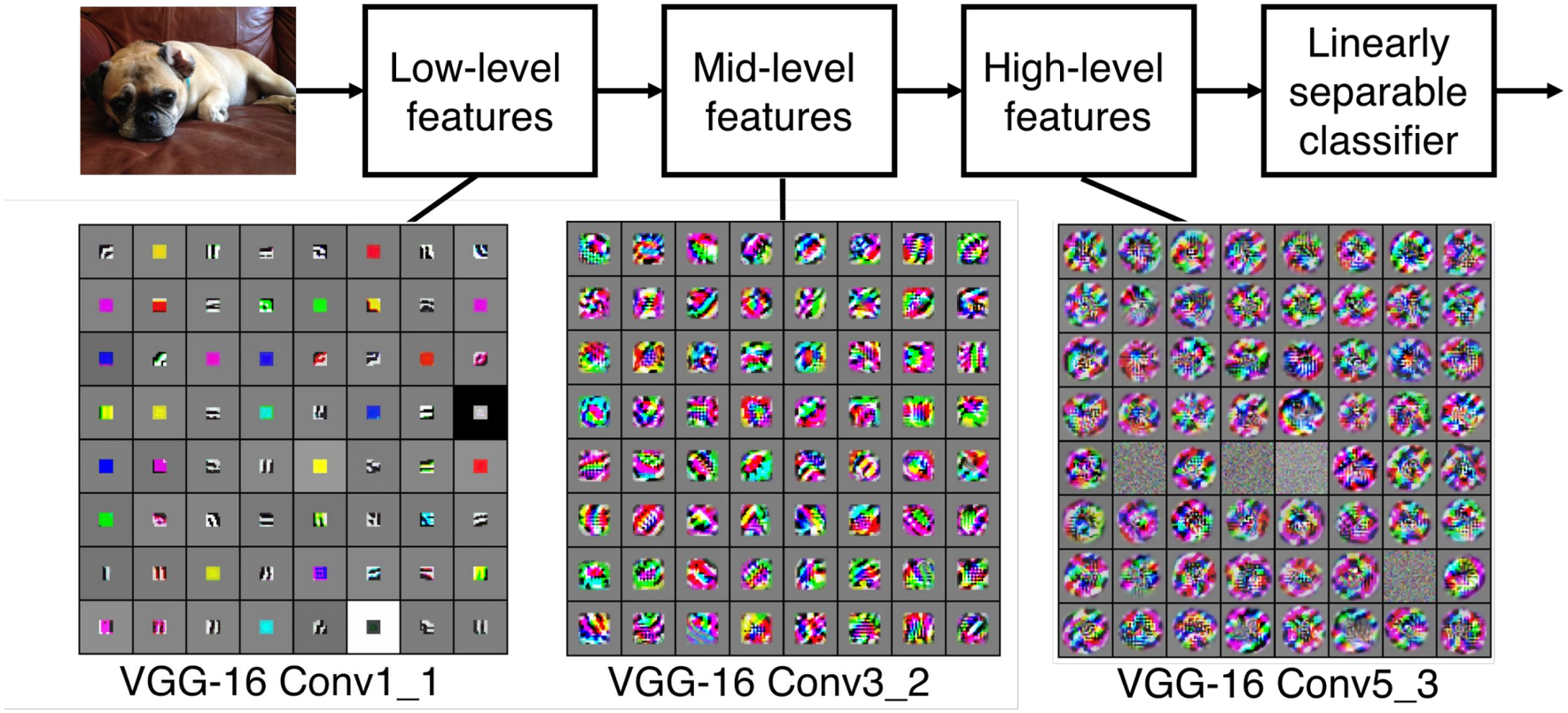
[Zeiler and Fergus 2013]

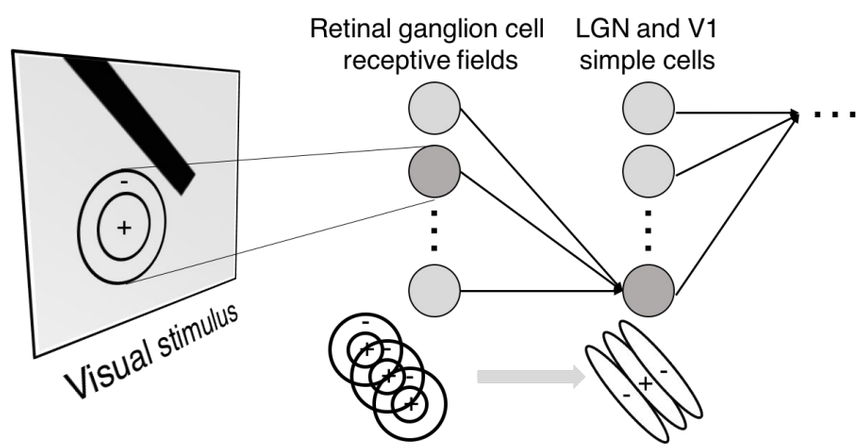
Visualization of VGG-16 by Lane McIntosh. VGG-16 architecture from [Simonyan and Zisserman 2014].



Visualization

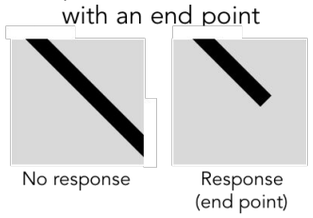
Preview



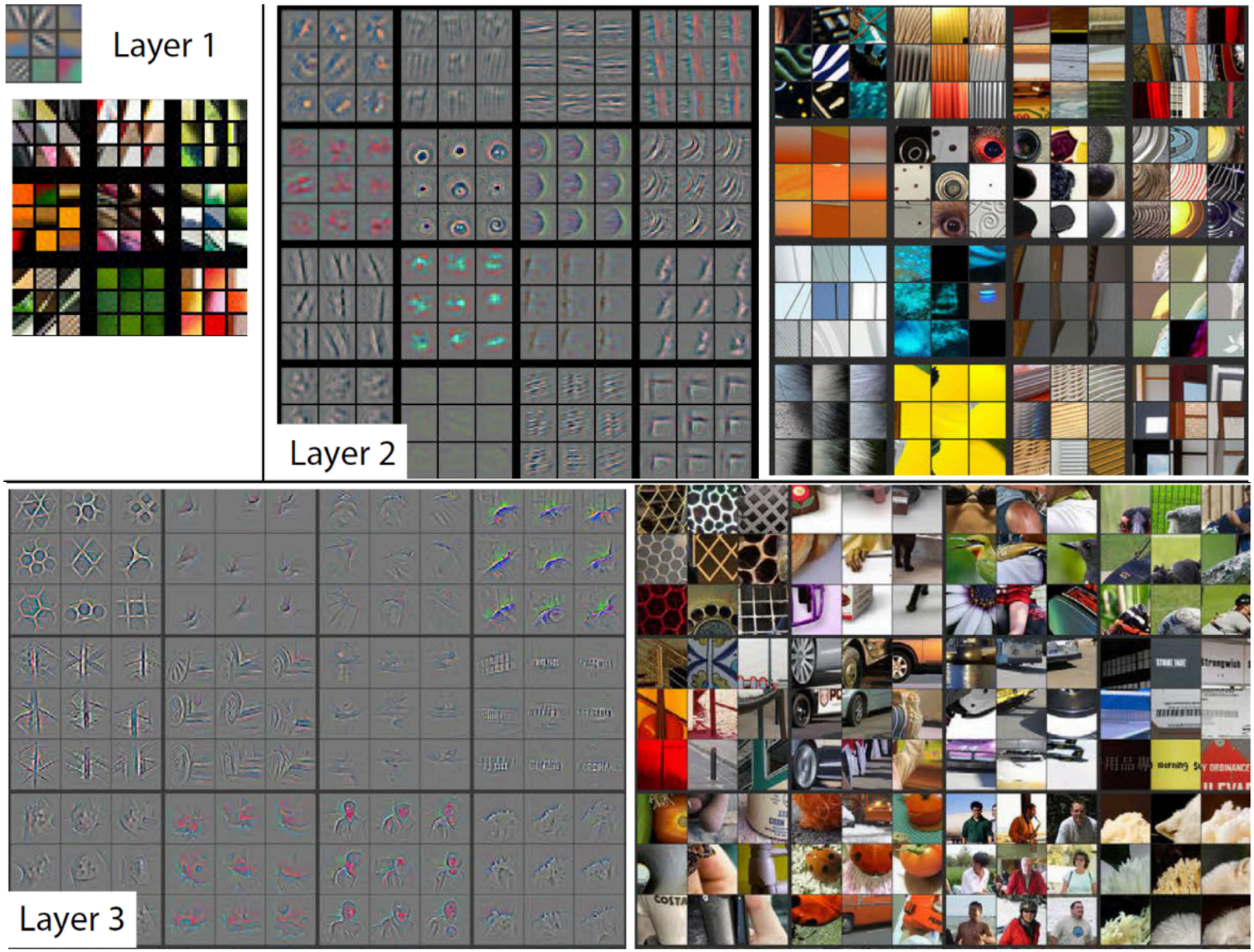


Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point



Visualization



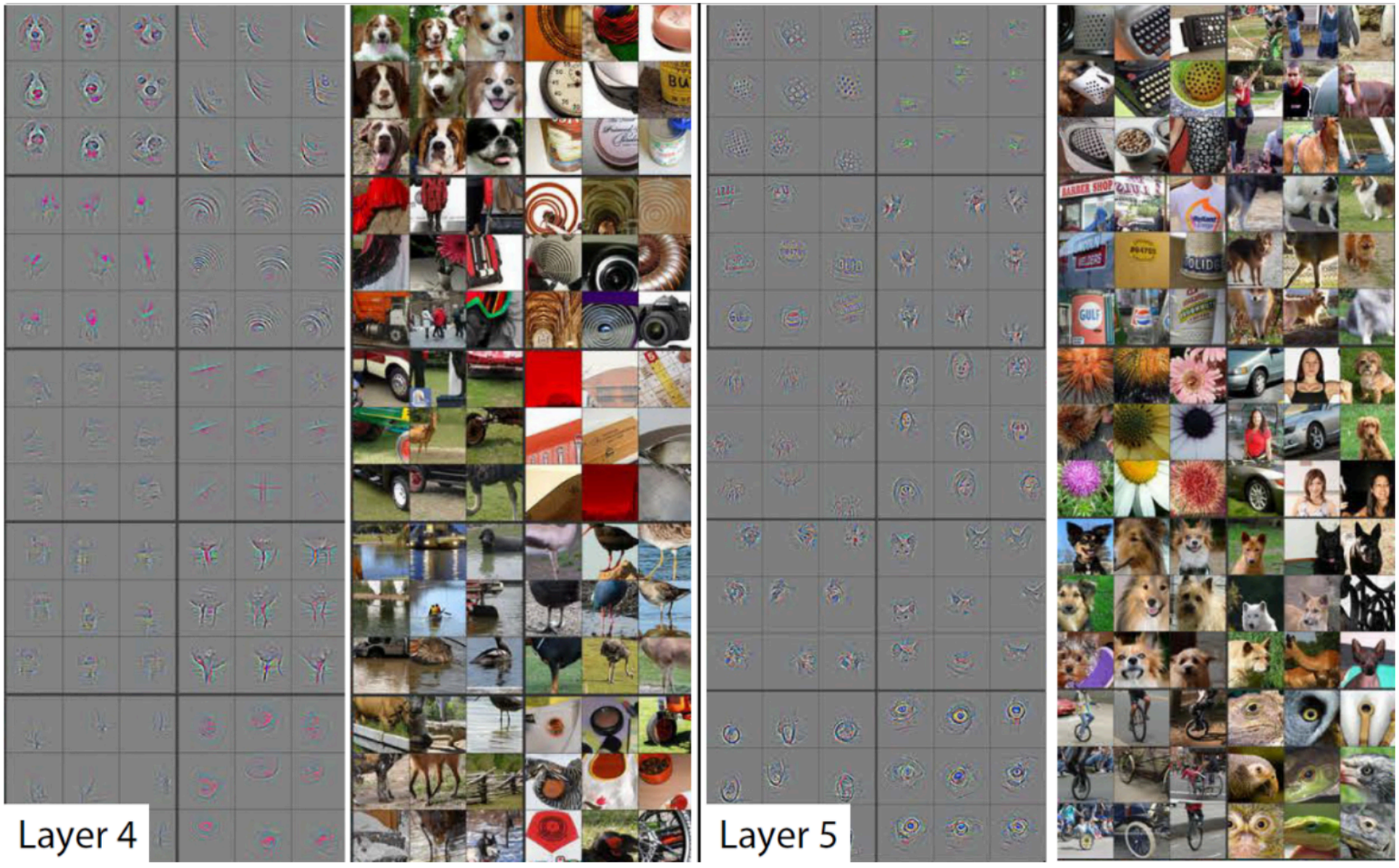


Figure 5.47 Visualizing network weights and features (Zeiler and Fergus 2014) © 2014 Springer. Each visualized convolutional layer is taken from a network adapted from the SuperVision net of Krizhevsky, Sutskever, and Hinton (2012). The 3×3 subimages denote the top nine responses in one feature map (channel in a given layer) projected back into pixel space (higher layers project to larger pixel patches), with the color images on the right showing the most responsive image patches from the validation set, and the grayish signed images on the left showing the corresponding maximum stimulus pre-images.

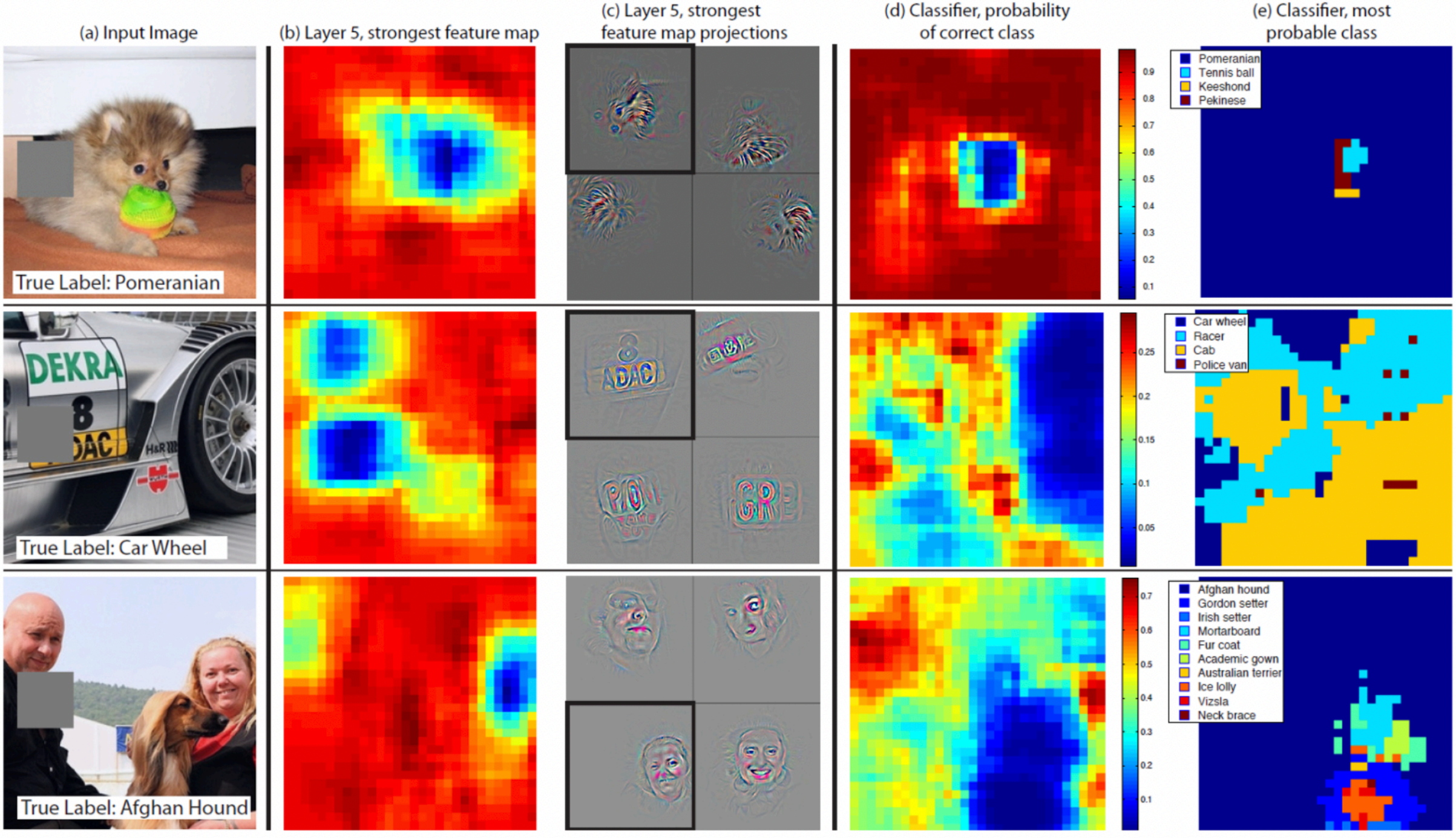


Figure 5.48 Heat map visualization from Zeiler and Fergus (2014) © 2014 Springer. By covering up portions of the input image with a small gray square, the response of a highly active channel in layer 5 can be visualized (second column), as can the feature map projections (third column), the likelihood of the correct class, and the most likely class per pixel.