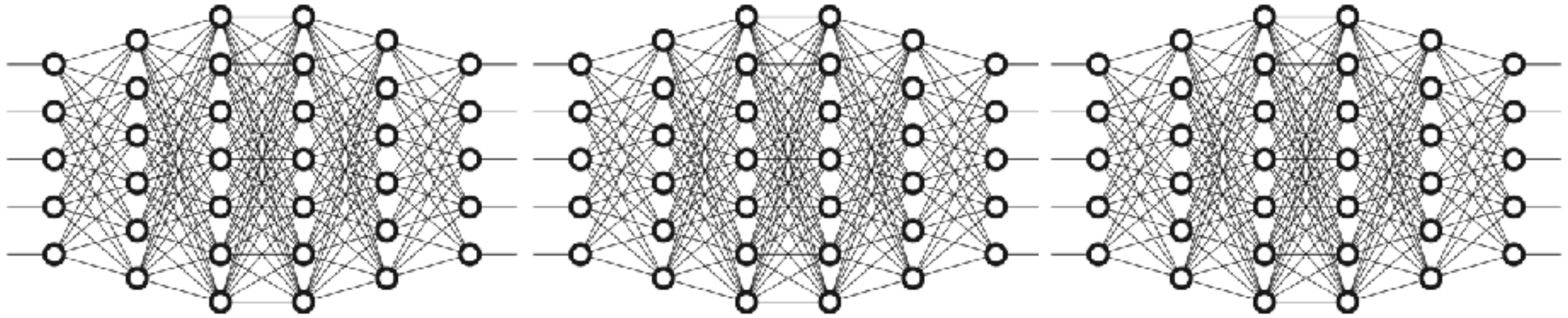
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 21: Neural Networks 2

Menu for Today

Topics:

— **Neural Networks** part 2
— **Linear + Convolutional** layers
— **Deep nets,** AlexNet, VGG

Readings:

— **Today's** Lecture: Szeliski 5.1.3, 5.3-5.4, Justin Johnson Michigan EECS 498/598

Reminders:

— **Quiz 6** April 7th
— **Assignment 6**: due Apr 10th <— watch out!

Many slides from this lecture are from

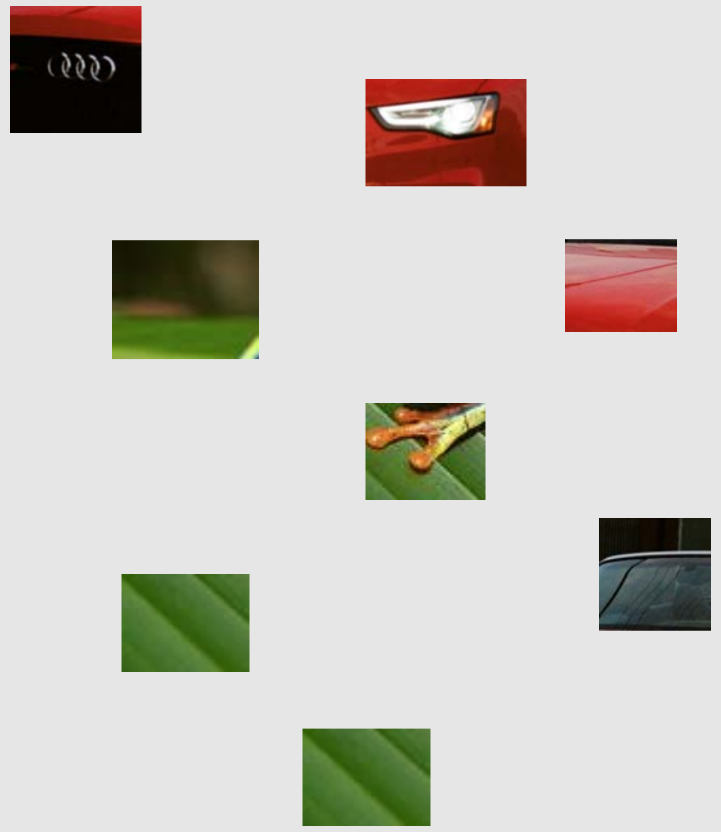
Justin Johnson, University of Michigan, EECS 498/598

https://web.eecs.umich.edu/~justincj/

Image Features: Bag of Words (Data-Driven!)

Step 1: Build codebook

Extract random patches



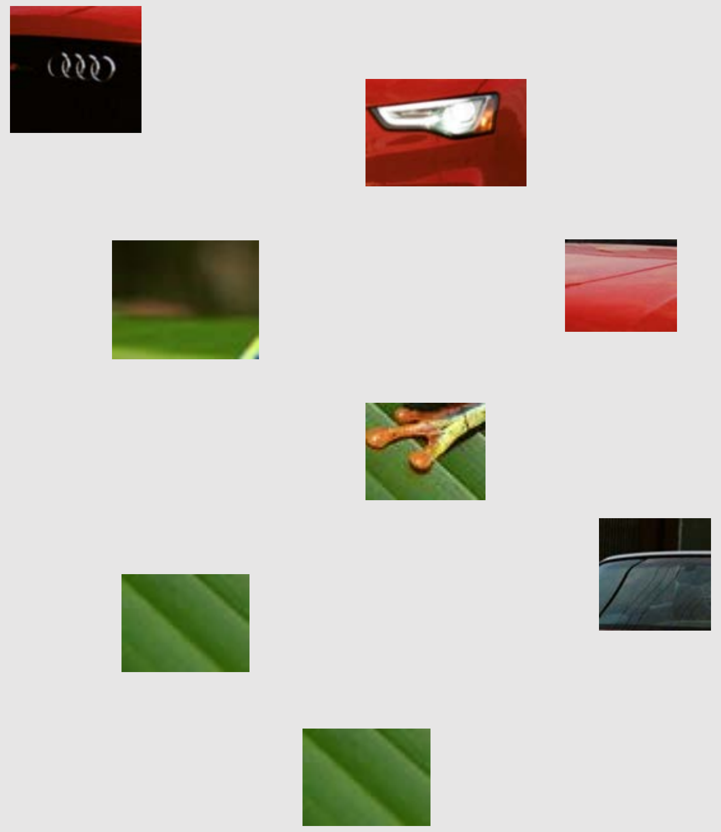
Cluster patches to form "codebook" of "visual words"

Fei-Fei and Perona, "A bayesian hierarchical model for learning natural scene categories", CVPR 2005

Image Features: Bag of Words (Data-Driven!)

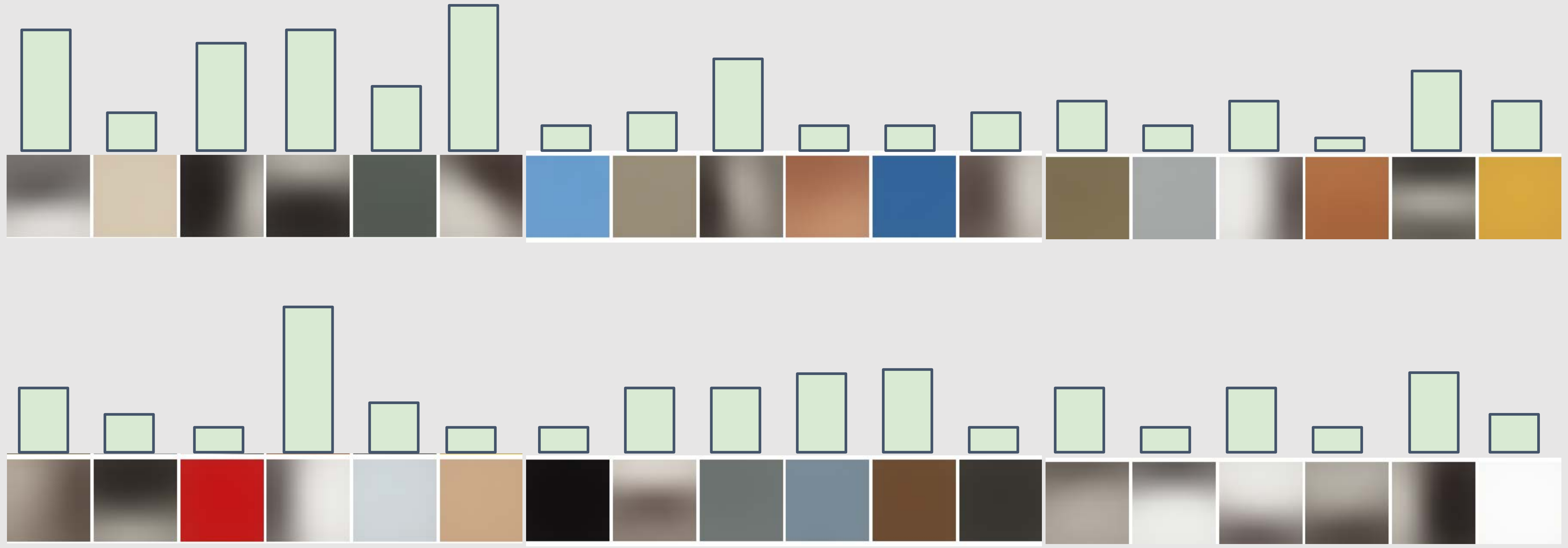
Step 1: Build codebook

Extract random patches



Cluster patches to form “codebook” of “visual words”

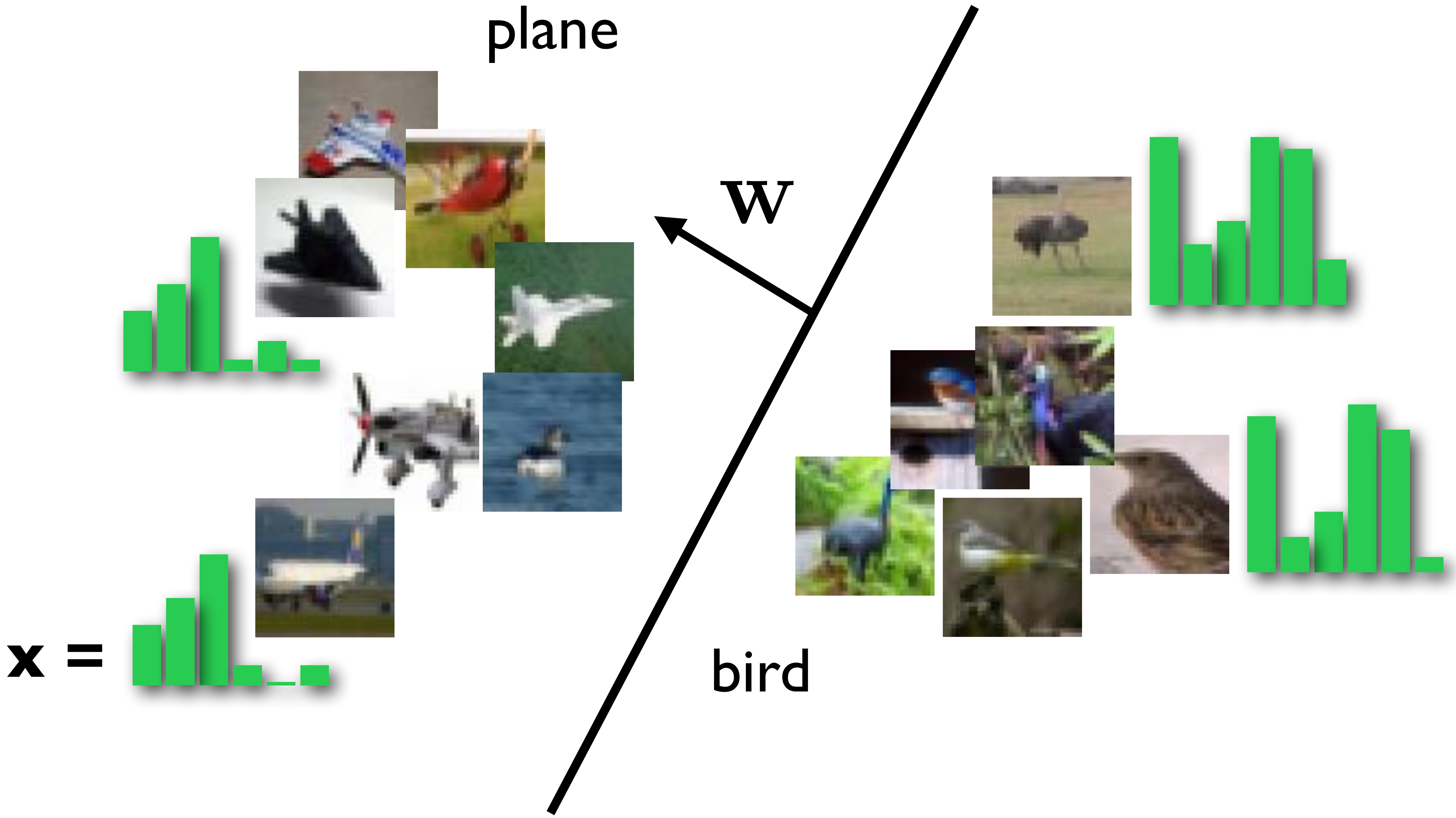
Step 2: Encode images



Fei-Fei and Perona, “A bayesian hierarchical model for learning natural scene categories”, CVPR 2005

Classify Visual Word Histograms

- e.g., bird vs plane classifier as linear classifier in space of histograms
- Histograms of visual word frequencies = vector **x**, linear classifier **w**



Example: Winner of 2011 ImageNet challenge

Low-level feature extraction ≈ 10k patches per image

- SIFT: 128-dim
- color: 96-dim

$\}$ reduced to 64-dim with PCA

FV extraction and compression:

- N=1,024 Gaussians, R=4 regions ⇨ 520K dim x 2
- compression: G=8, b=1 bit per dimension

One-vs-all SVM learning with SGD

Late fusion of SIFT and color systems

F. Perronnin, J. Sánchez, “Compressed Fisher vectors for LSVRC”, PASCAL VOC / ImageNet workshop, ICCV, 2011.

IMAGENET Large Scale Visual Recognition Challenge

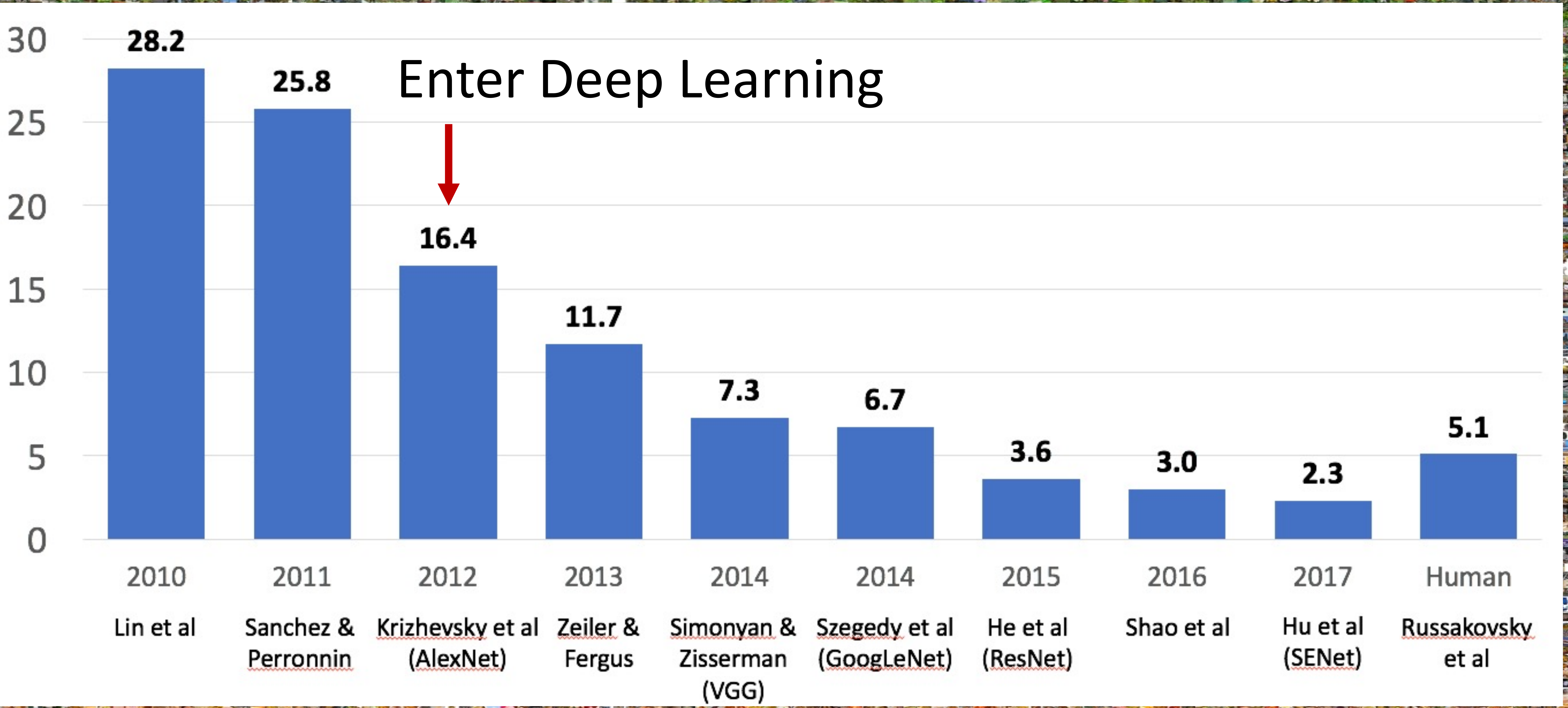
The Image Classification Challenge:
1,000 object classes
1,431,167 images

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

Deng et al, 2009
Russakovsky et al. IJCV 2015

1959 Hubel & Wiesel | 1963 Roberts | 1970s David Marr | 1979 Gen. Cylinders | 1986 Canny | AI Winter | 1997 Norm. Cuts | 1999 SIFT | 2001 V&J | 2007 PASCAL | 2009 ImageNet

IMAGENET Large Scale Visual Recognition Challenge



1959 Hubel & Wiesel

1963 Roberts

1970s David Marr

1979 Gen. Cylinders

1986 Canny

AI Winter

1997 Norm. Cuts

1999 SIFT

2001 V&J

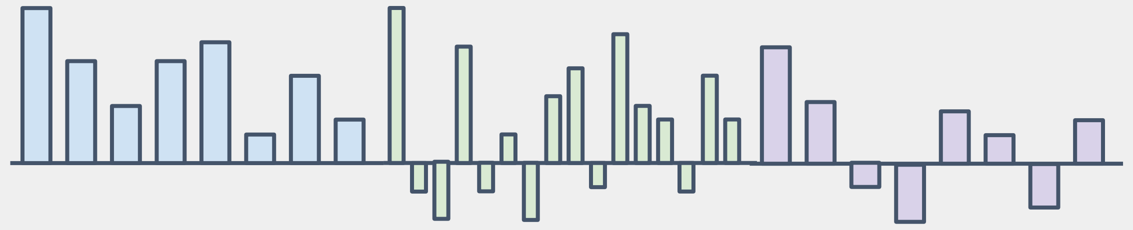
2007 PASCAL

2009 ImageNet

2012 AlexNet

Image Features vs Neural Networks

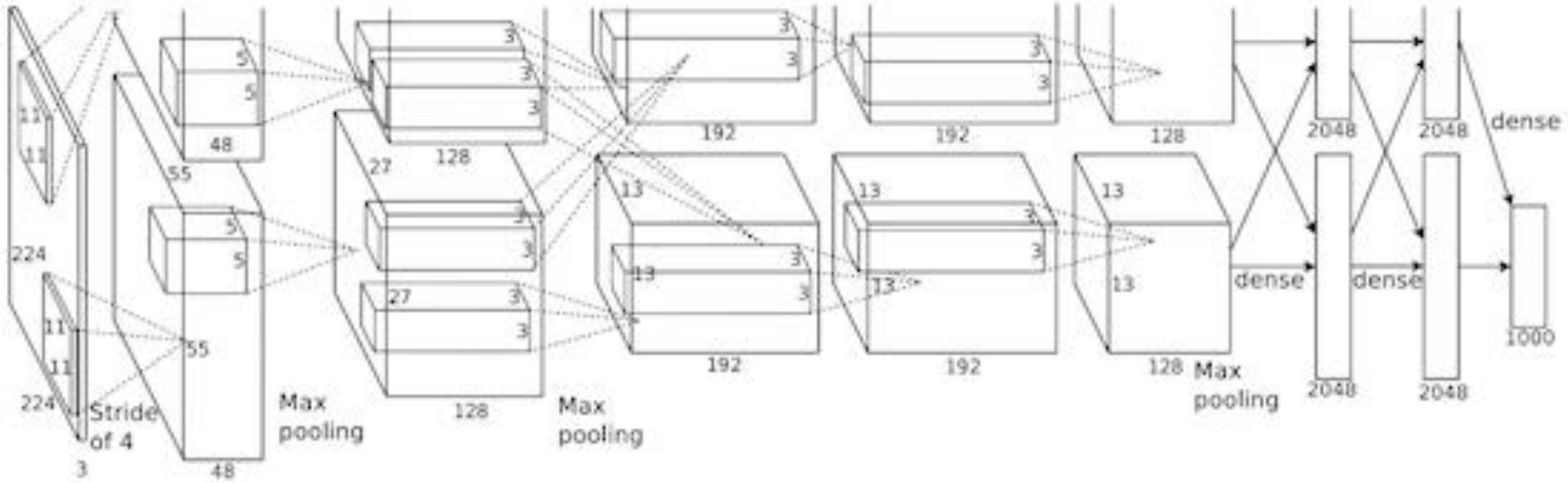
Feature Extraction



f

10 numbers giving scores for classes

training

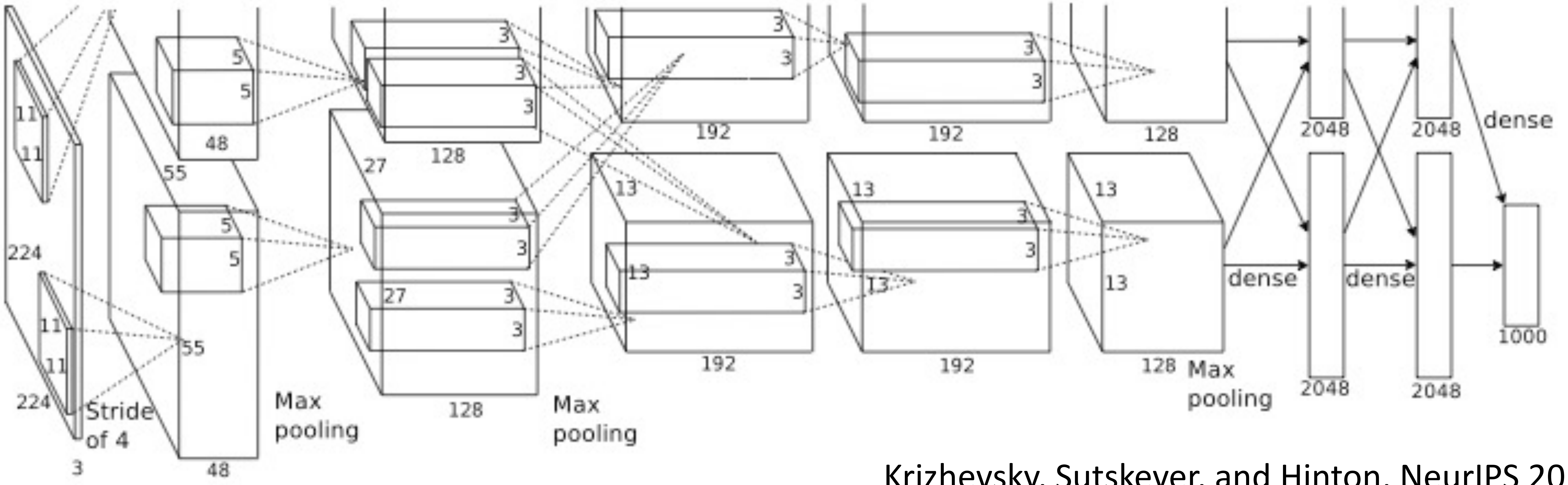


Krizhevsky, Sutskever, and Hinton, "Imagenet classification with deep convolutional neural networks", NIPS 2012. Figure copyright Krizhevsky, Sutskever, and Hinton, 2012. Reproduced with permission.

10 numbers giving scores for classes

training

AlexNet: Deep Learning Goes Mainstream



Krizhevsky, Sutskever, and Hinton, NeurIPS 2012

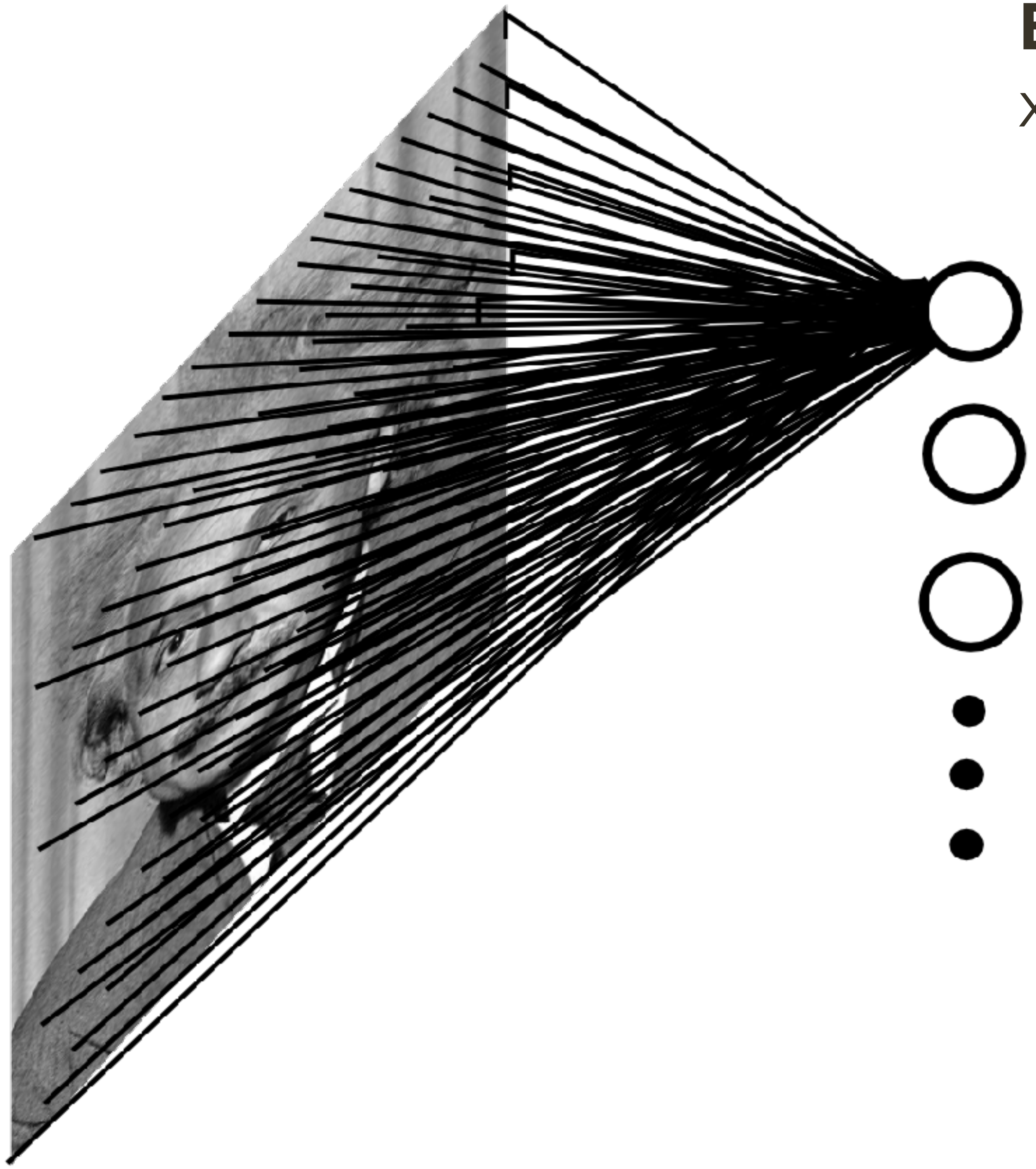
1959 Hubel & Wiesel | 1963 Roberts | 1970s David Marr | 1979 Gen. Cylinders | 1986 Canny | AI Winter | 1997 Norm. Cuts | 1999 SIFT | 2001 V&J | 2007 PASCAL | 2009 ImageNet | 2012 AlexNet

Backward Pass for Some **Common Layers**

Linear layers — fully connected

20.2

Fully Connected Layer



Example: 200 x 200 image (small)
x 40K hidden units (same size)

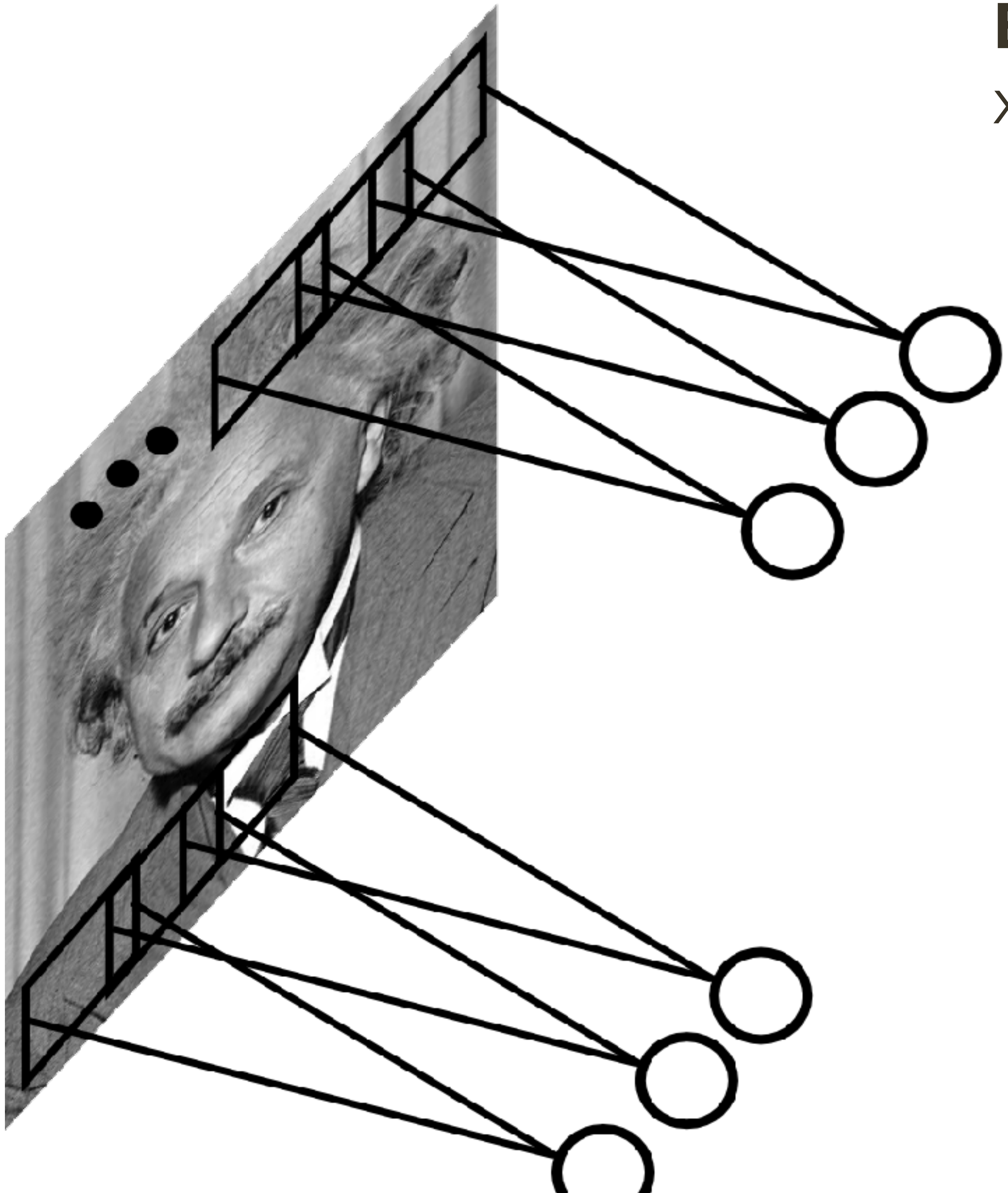
= **1.6 Billion** parameters (for one layer!)

Spatial correlations are generally local

Waste of resources + we don't have enough data to train networks this large

* slide from Marc'Aurelio Renzato

Convolutional Layer



Example: 200 x 200 image (small)
x 40K hidden units (same size)

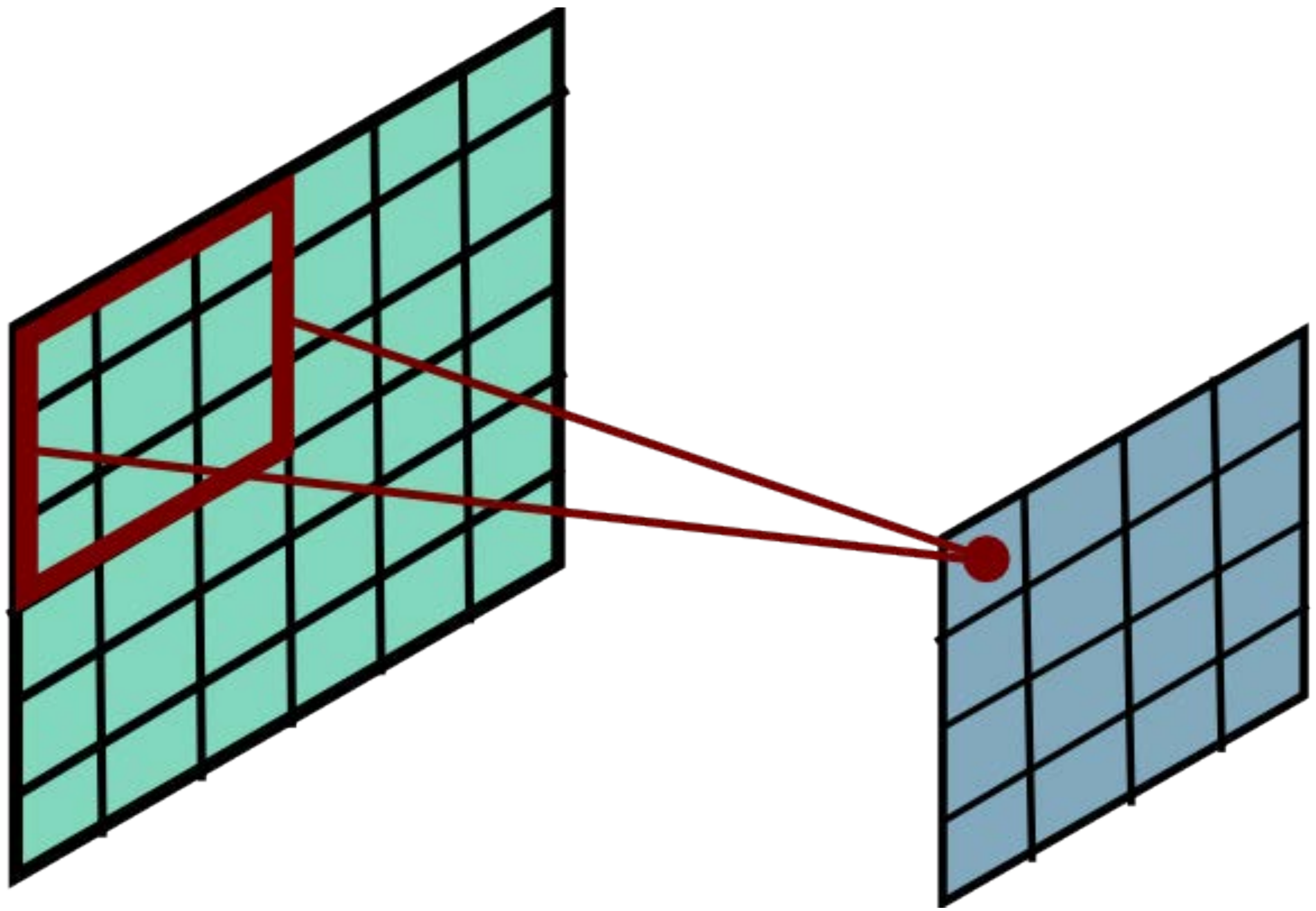
Filter size: 10 x 10

= 100 parameters

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

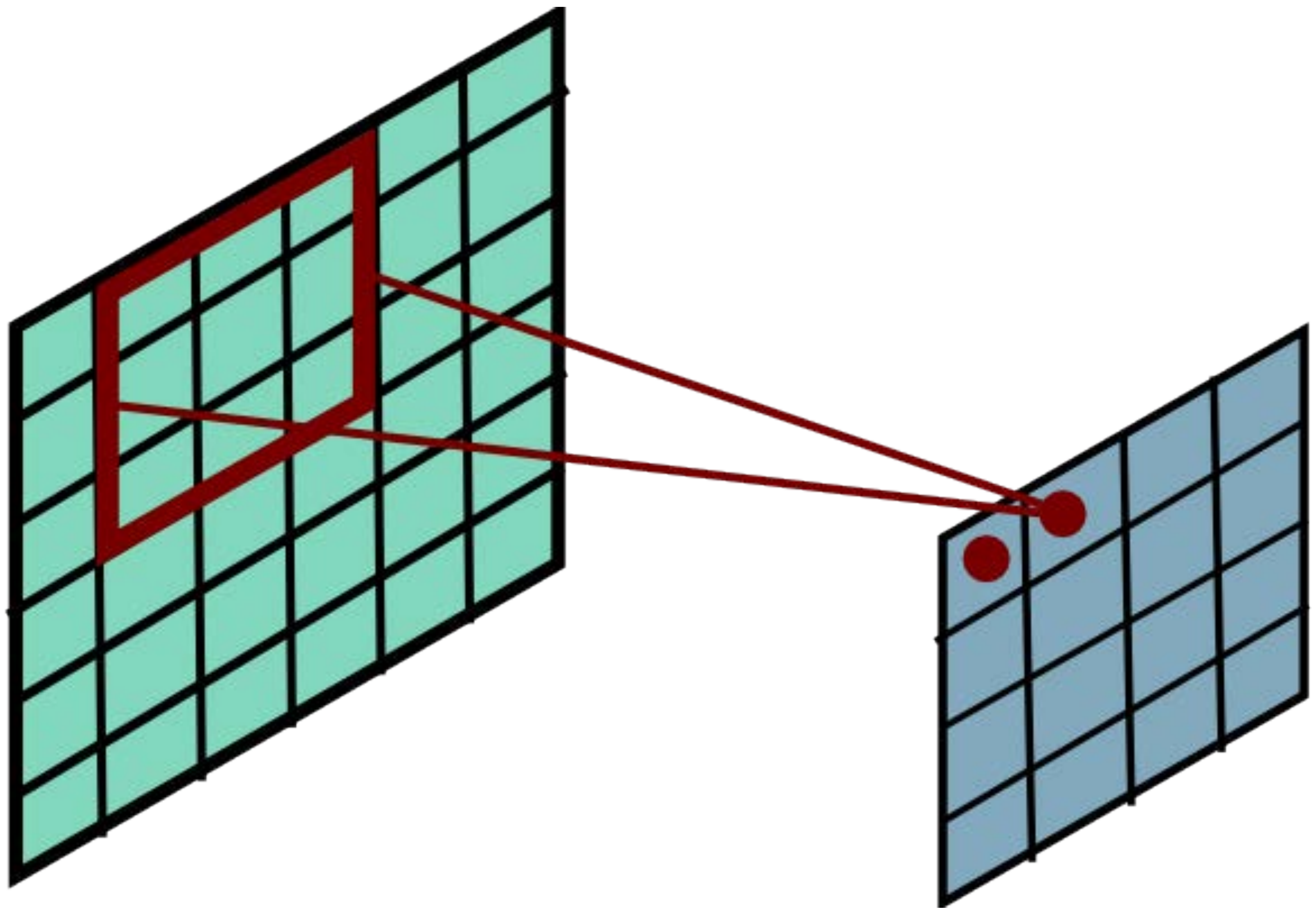
* slide adopted from Marc'Aurelio Renzato

Convolutional Layer



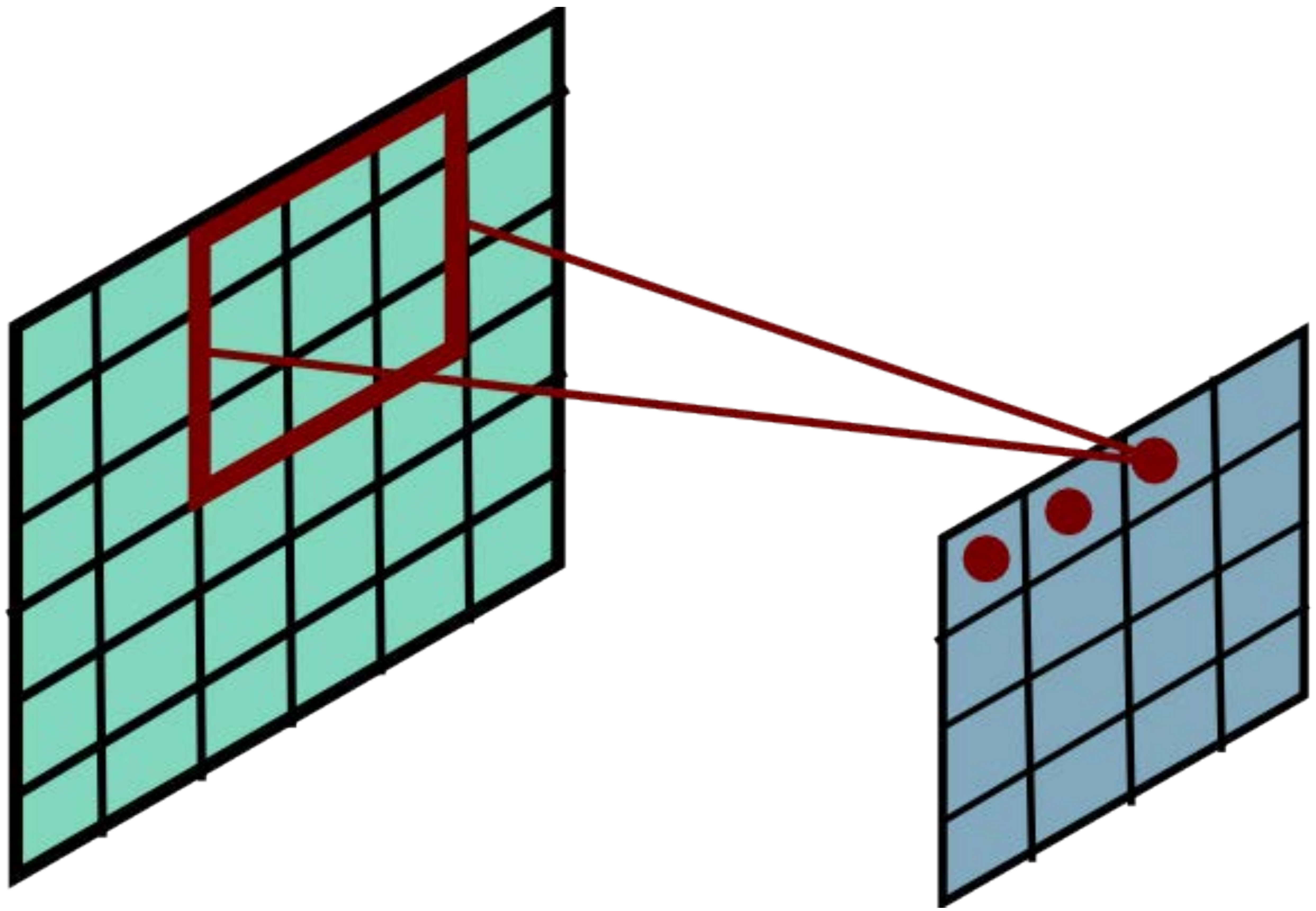
* slide from Marc'Aurelio Renzato

Convolutional Layer



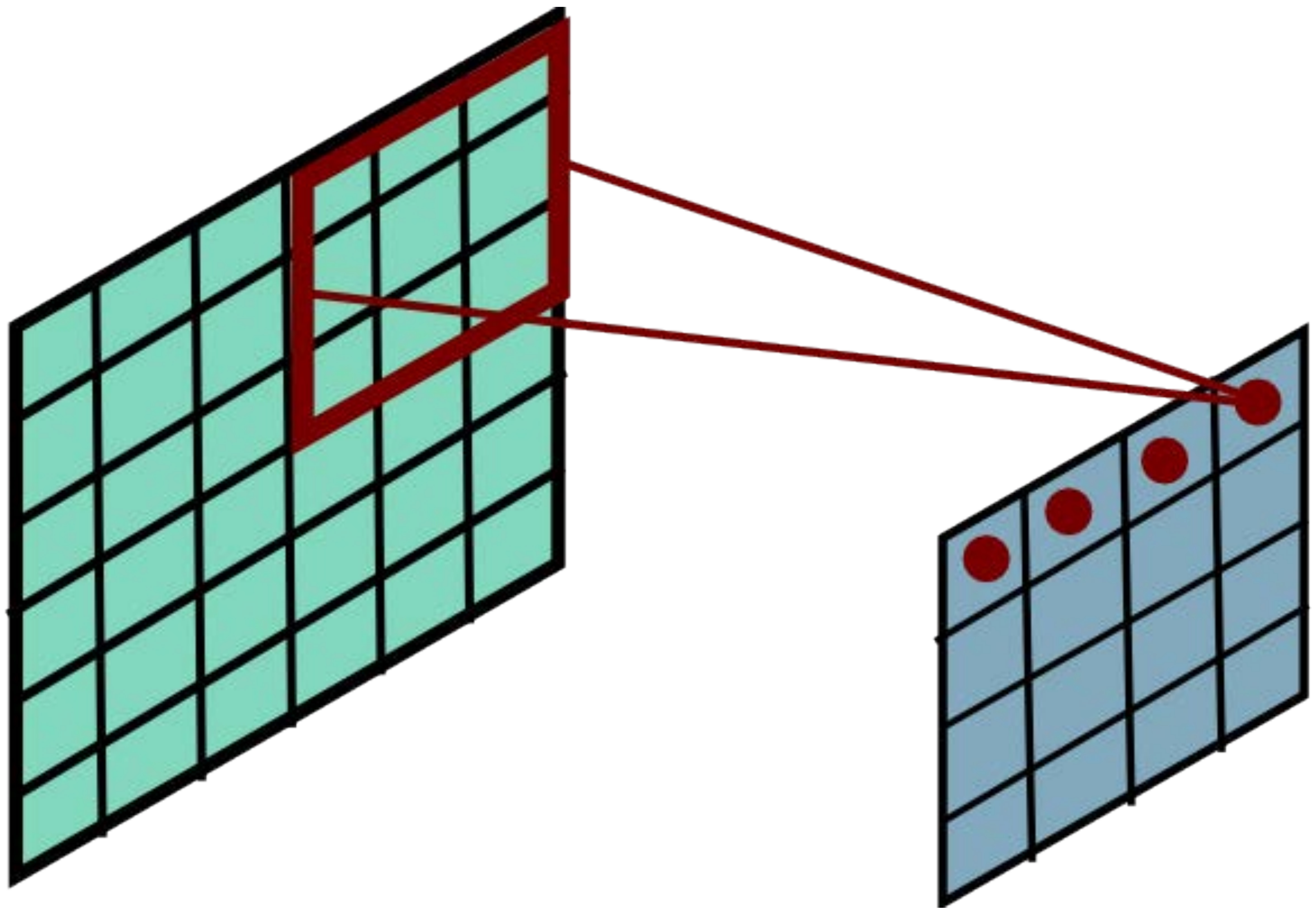
* slide from Marc’Aurelio Renzato

Convolutional Layer



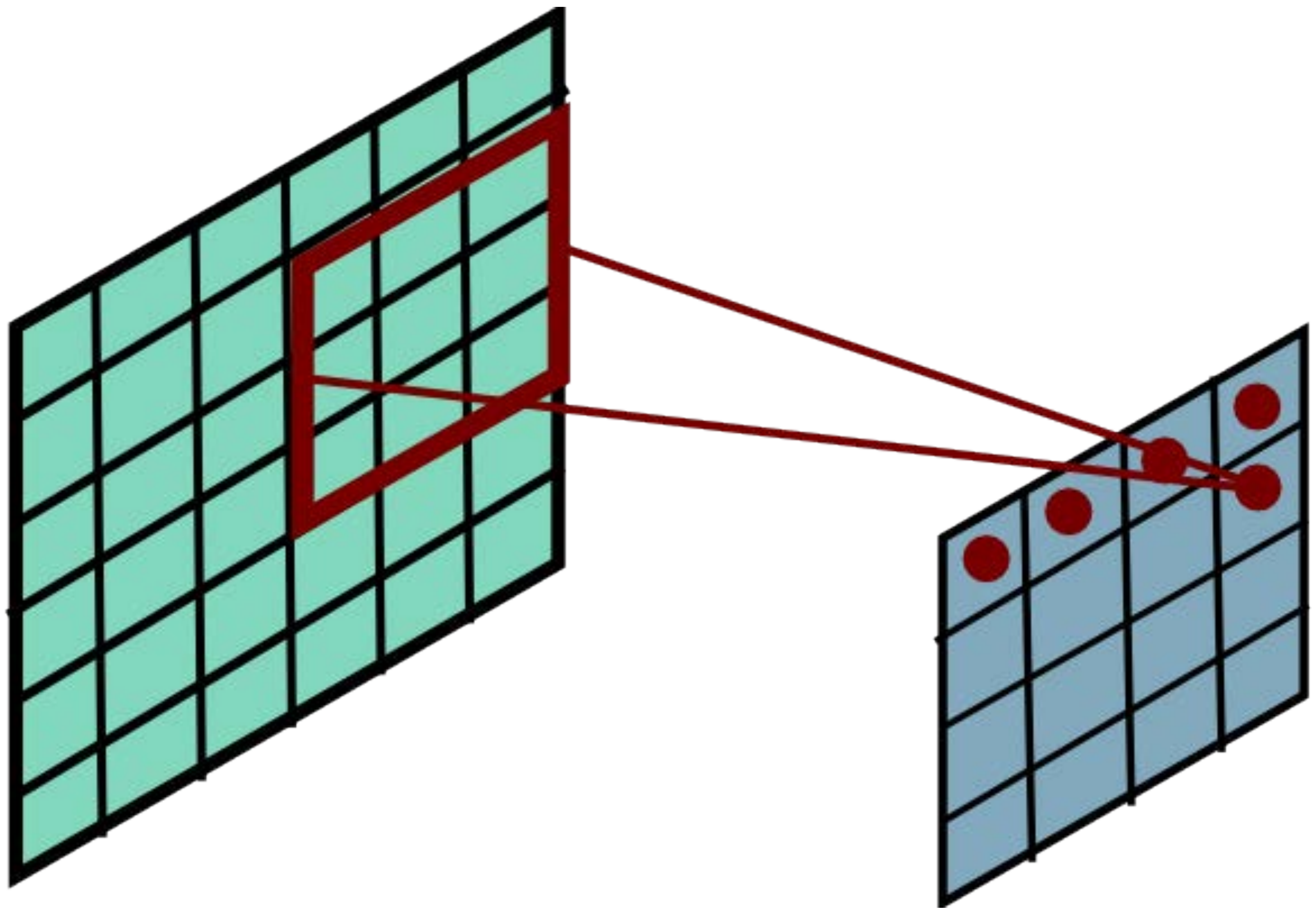
* slide from Marc'Aurelio Renzato

Convolutional Layer



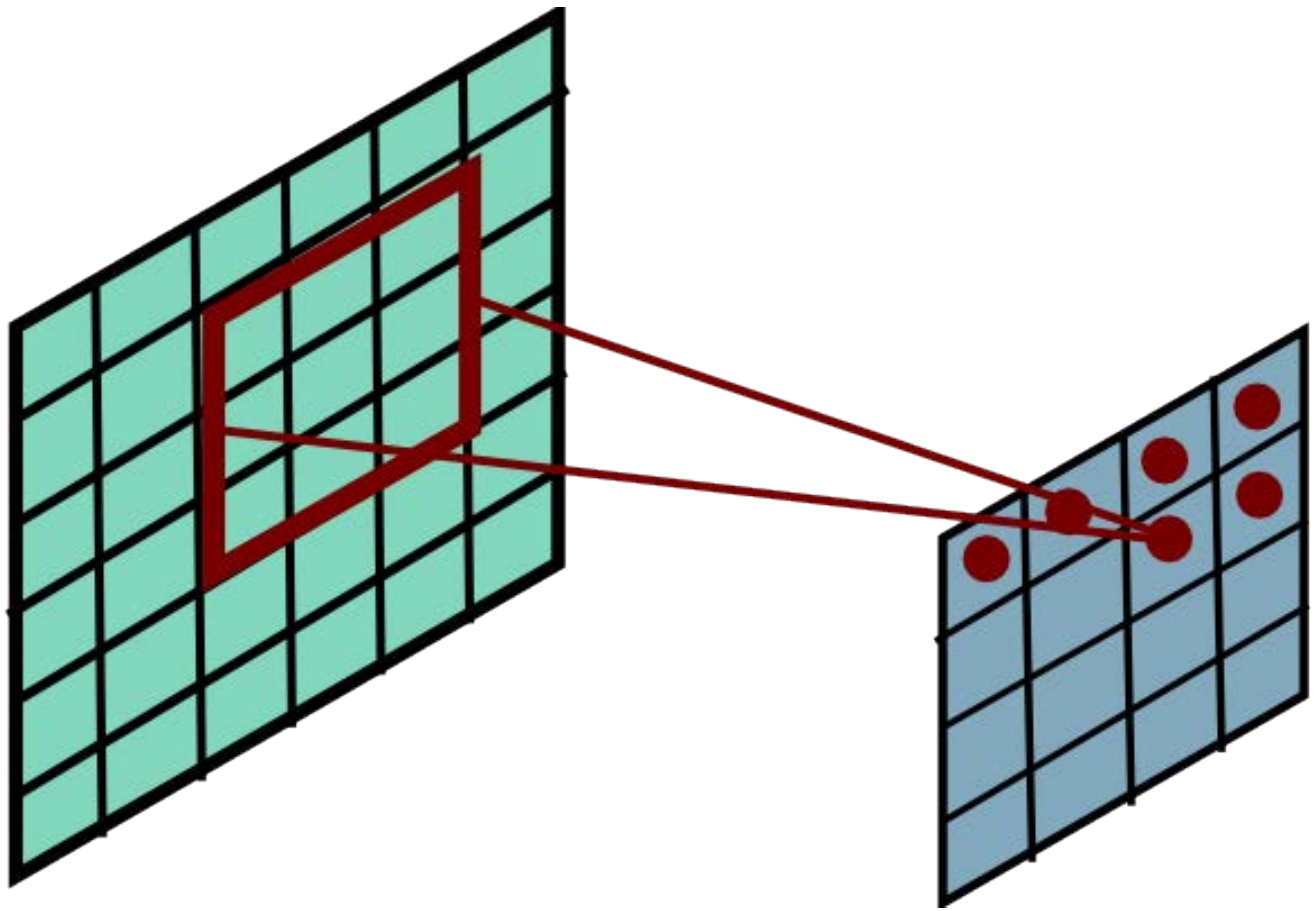
* slide from Marc'Aurelio Renzato

Convolutional Layer



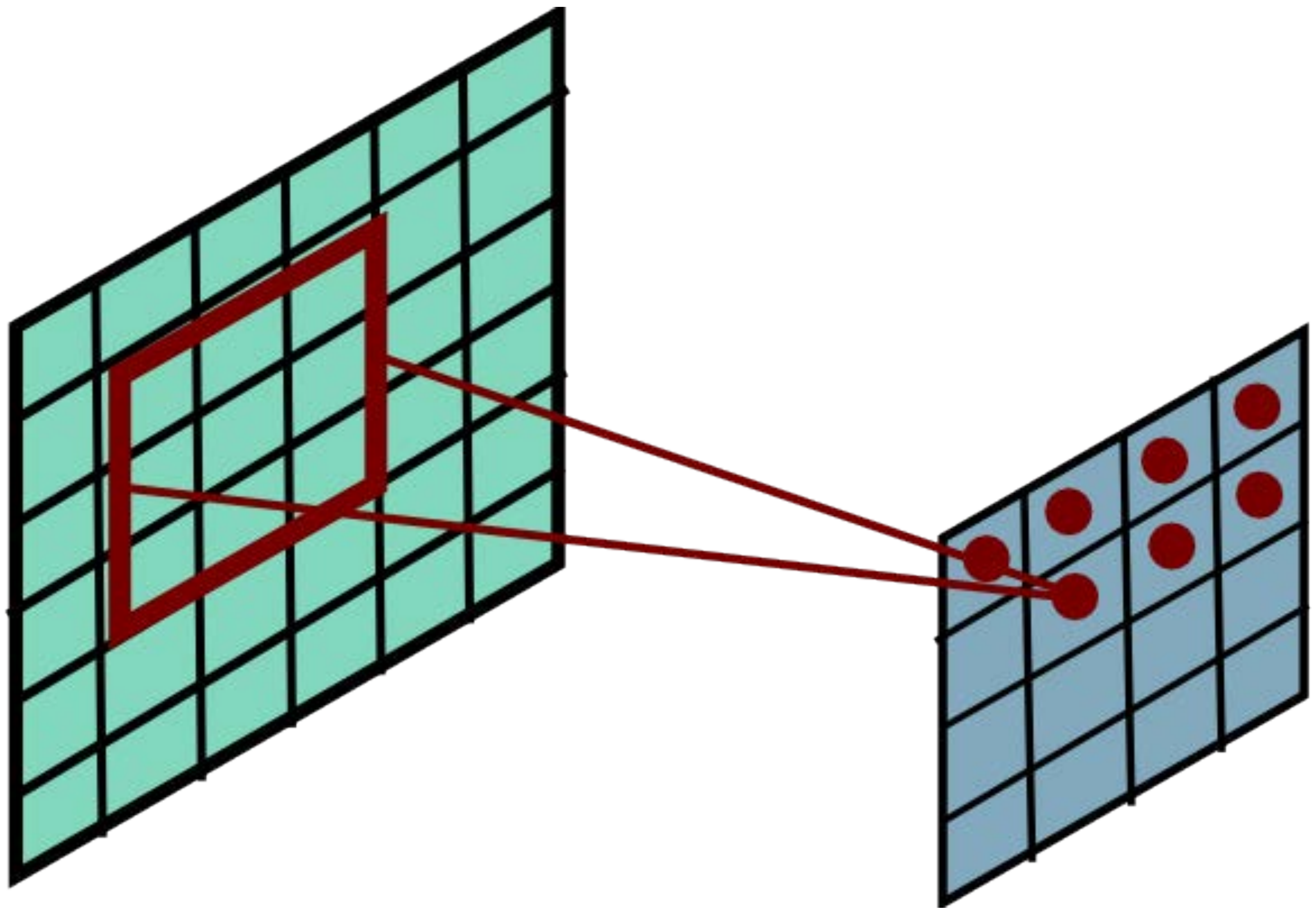
* slide from Marc'Aurelio Renzato

Convolutional Layer



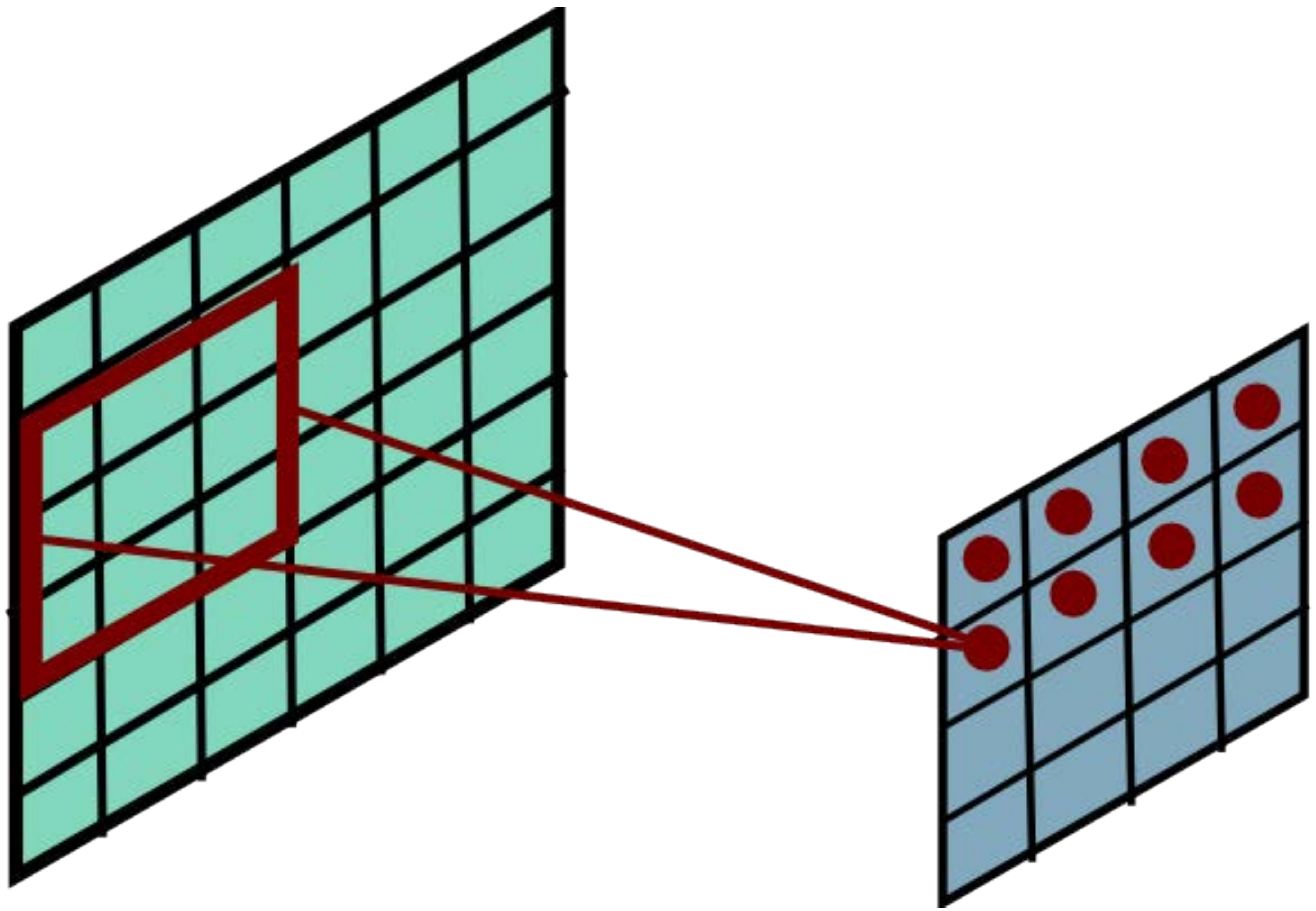
* slide from Marc'Aurelio Renzato

Convolutional Layer



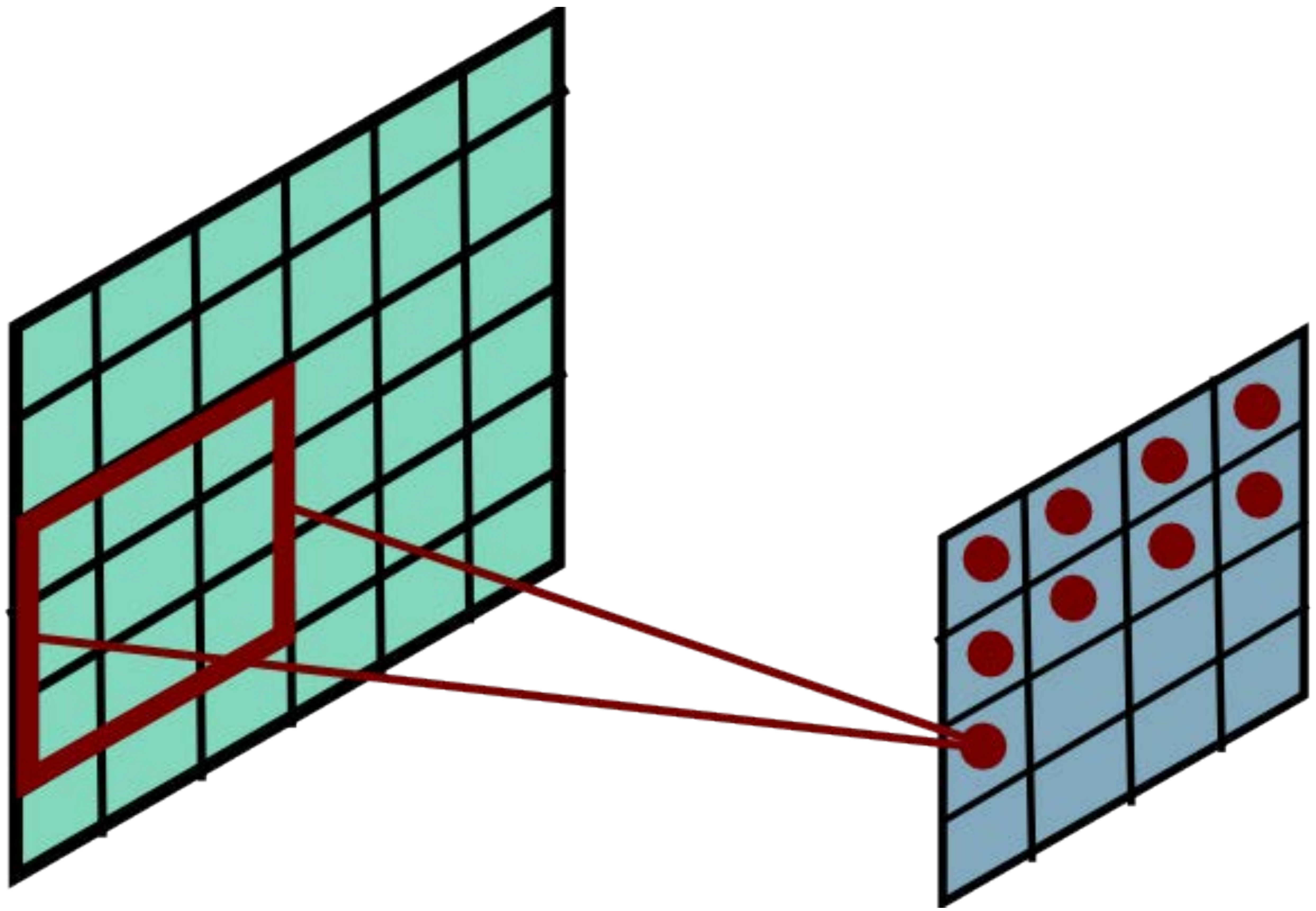
* slide from Marc'Aurelio Renzato

Convolutional Layer



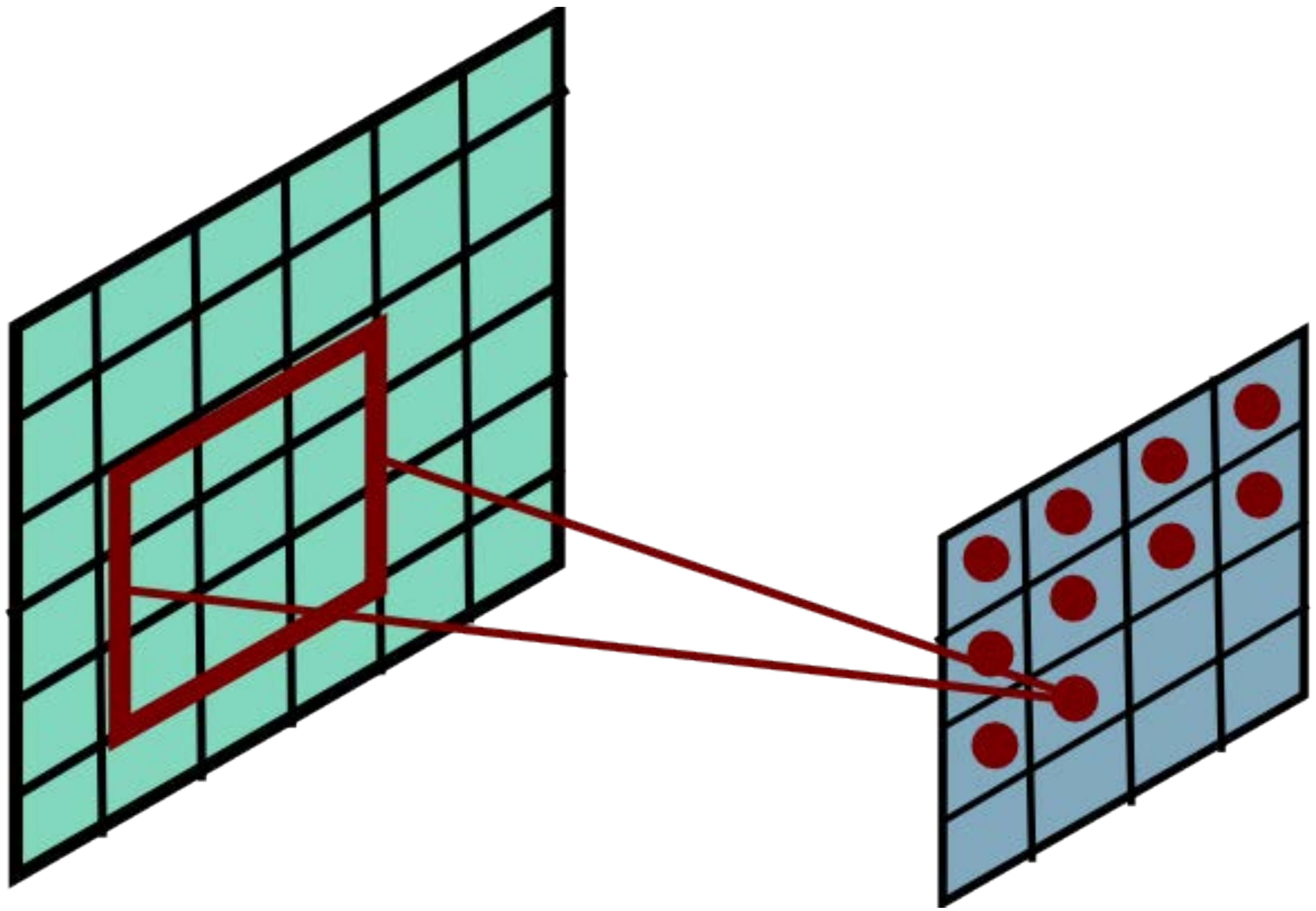
* slide from Marc'Aurelio Renzato

Convolutional Layer



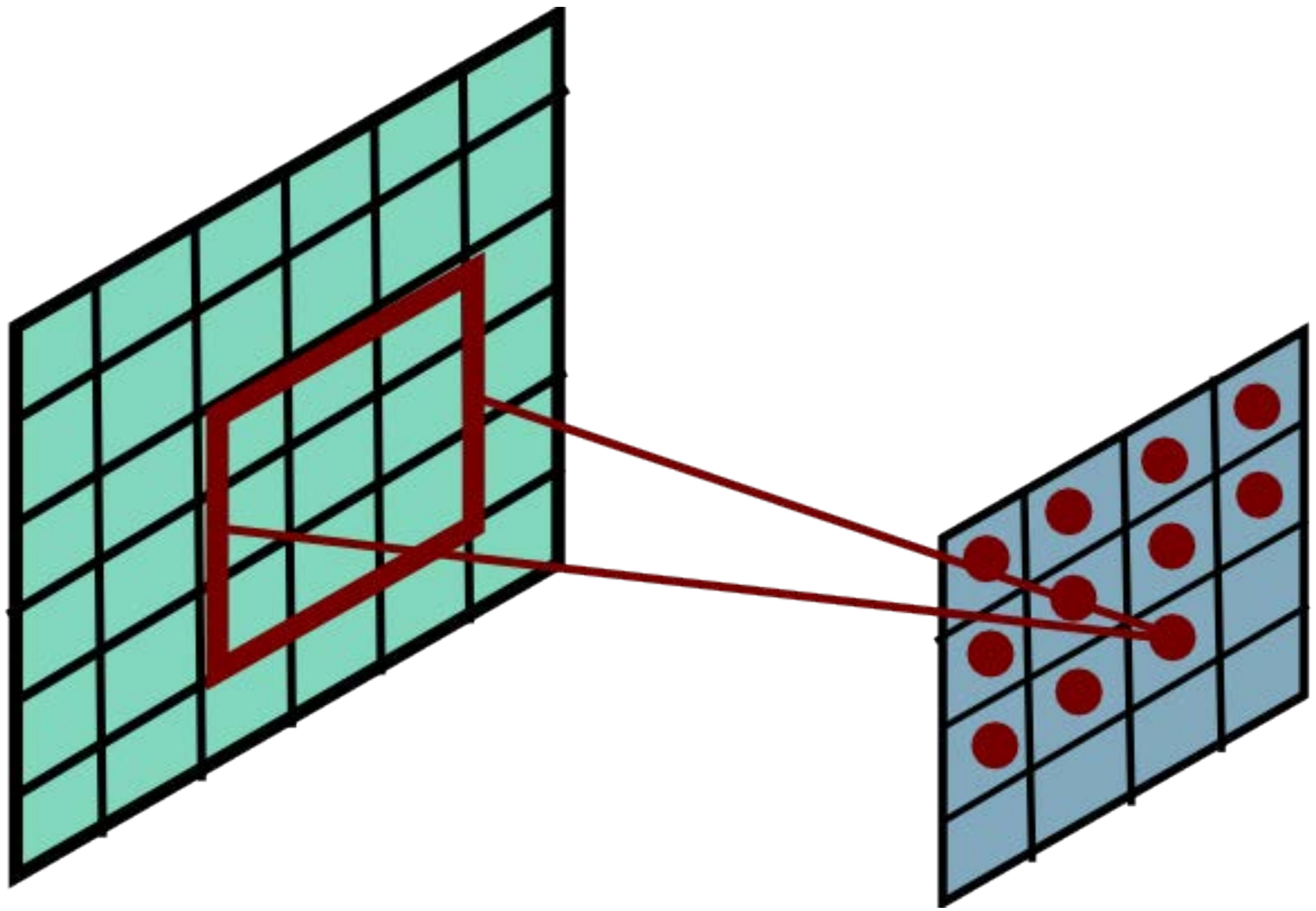
* slide from Marc'Aurelio Renzato

Convolutional Layer



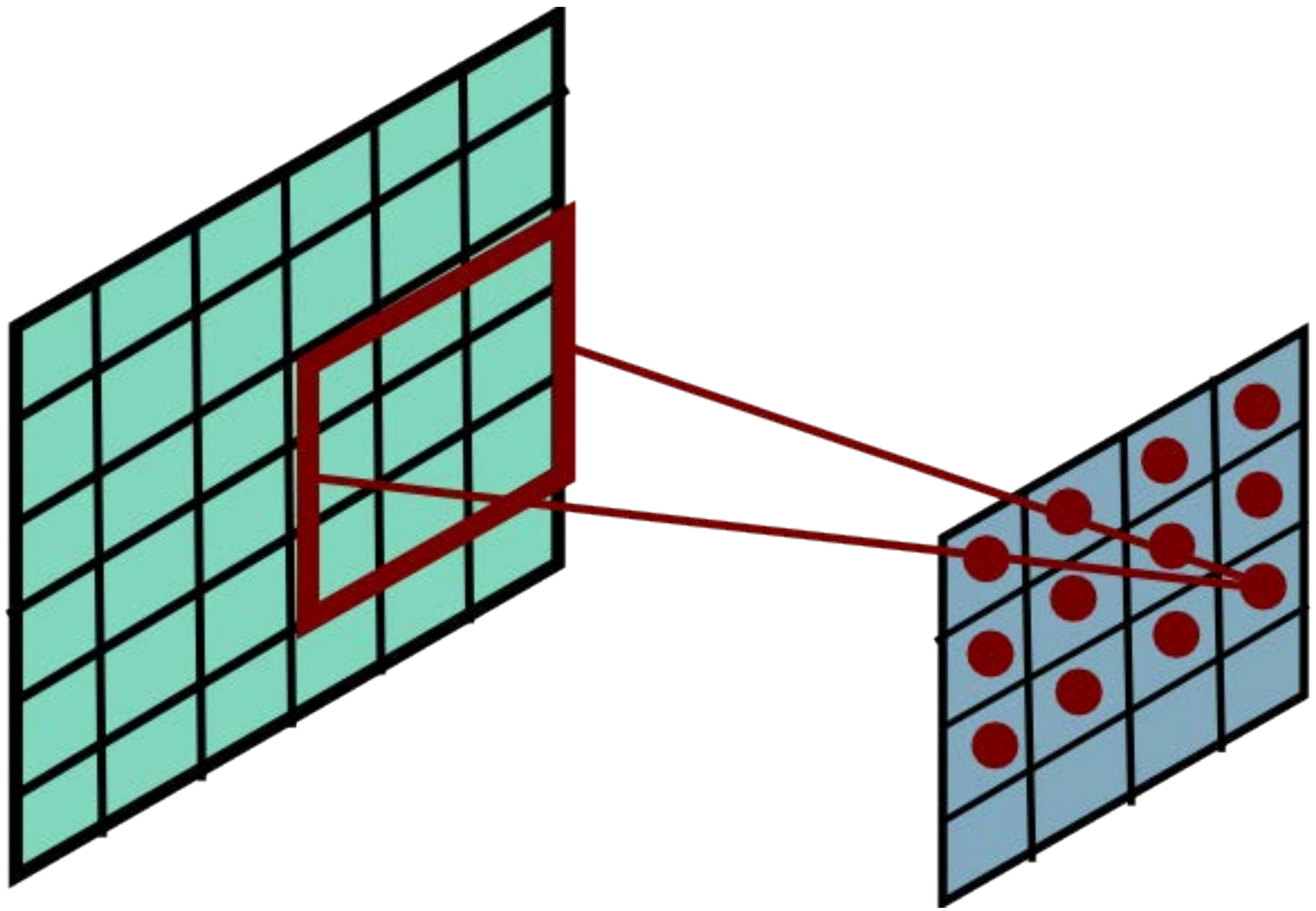
* slide from Marc'Aurelio Renzato

Convolutional Layer



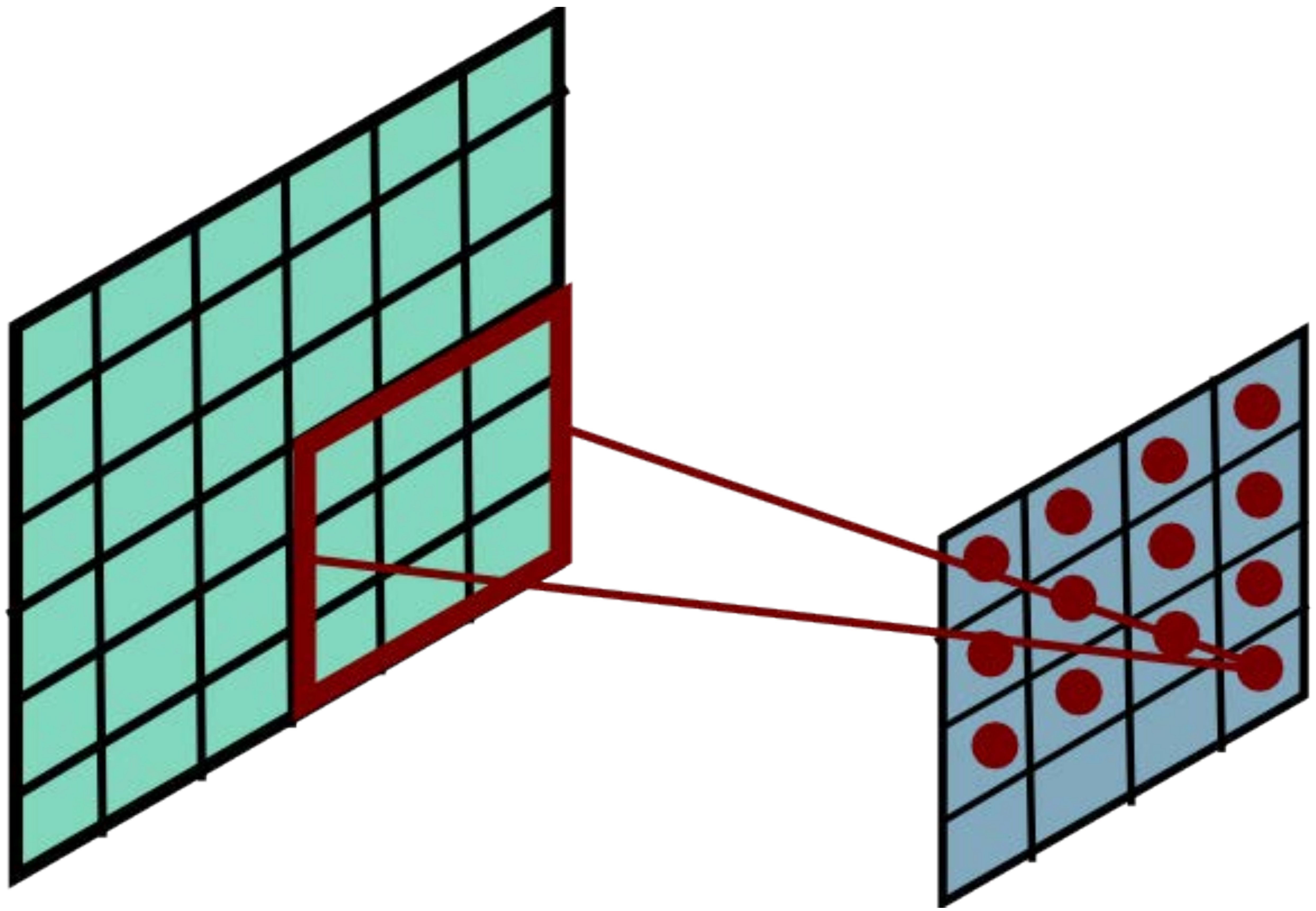
* slide from Marc'Aurelio Renzato

Convolutional Layer



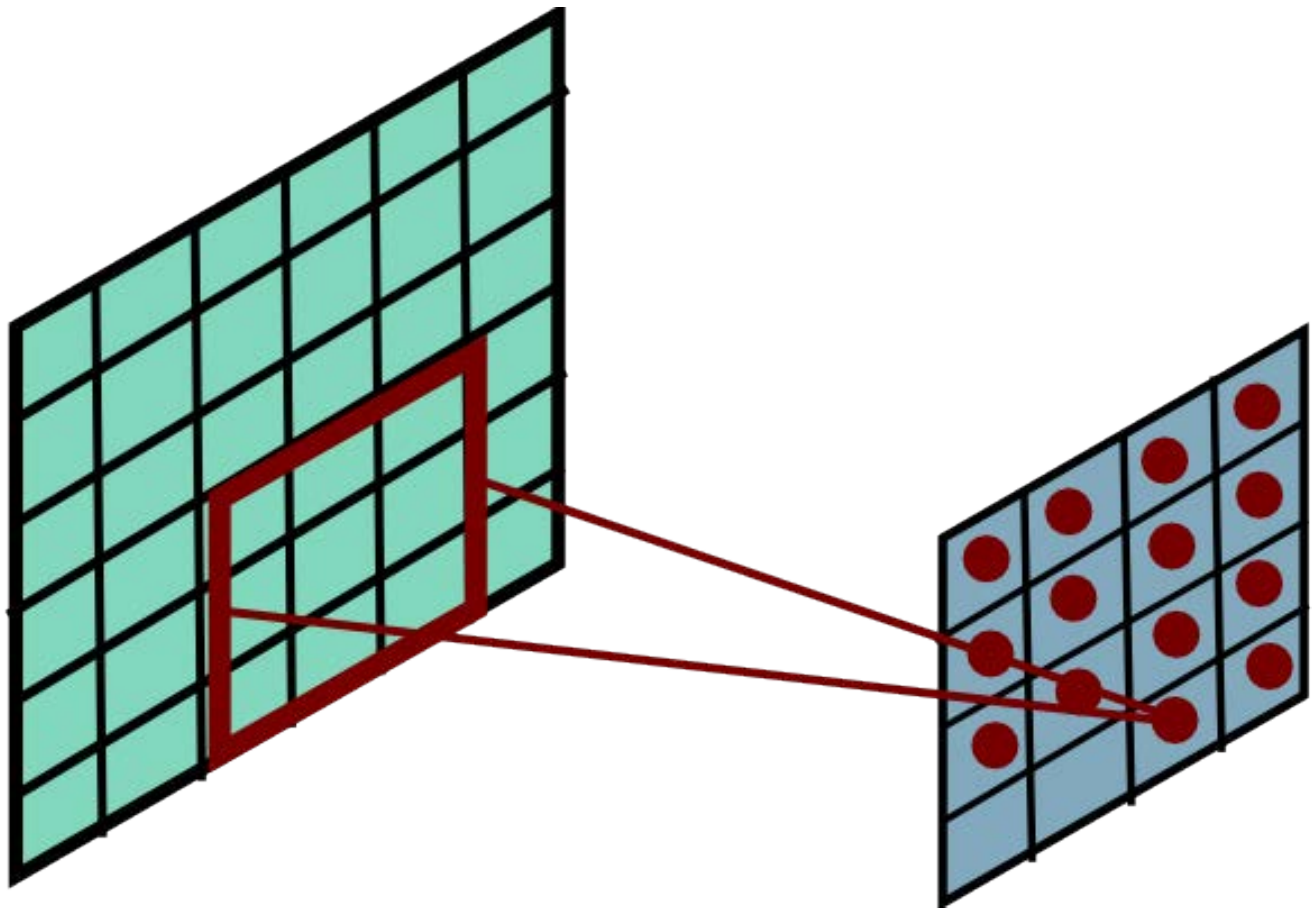
* slide from Marc'Aurelio Renzato

Convolutional Layer



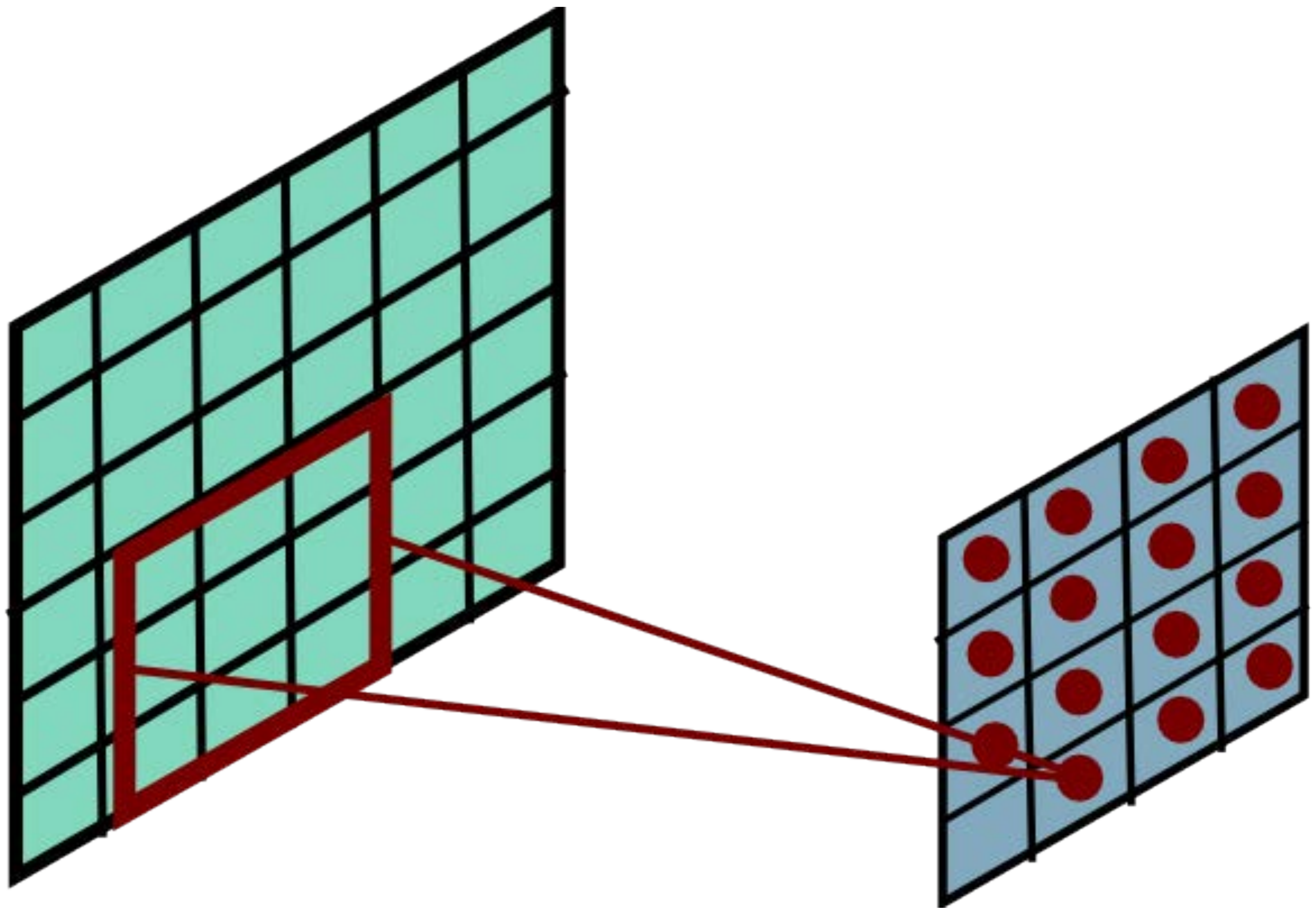
* slide from Marc'Aurelio Renzato

Convolutional Layer



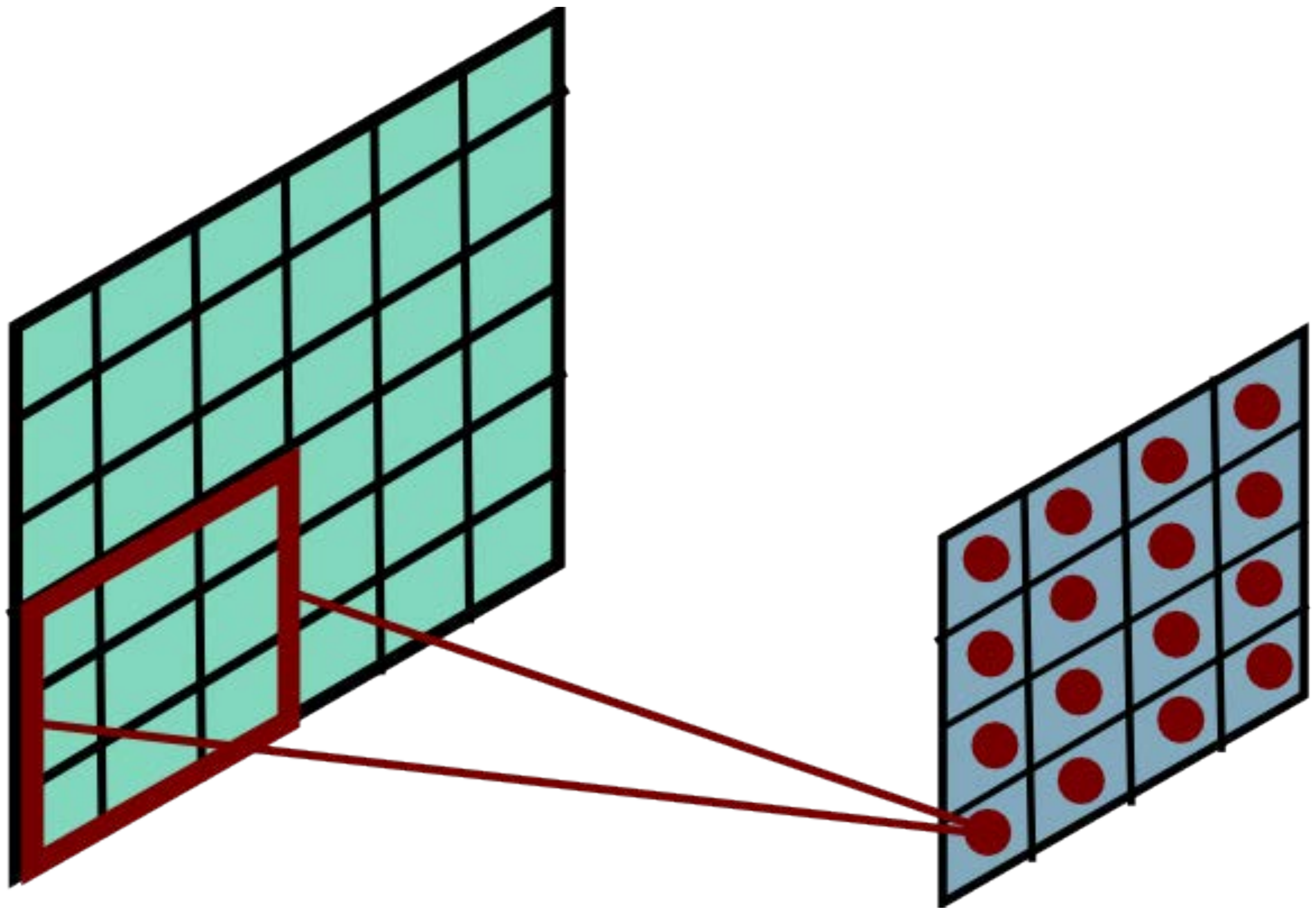
* slide from Marc'Aurelio Renzato

Convolutional Layer



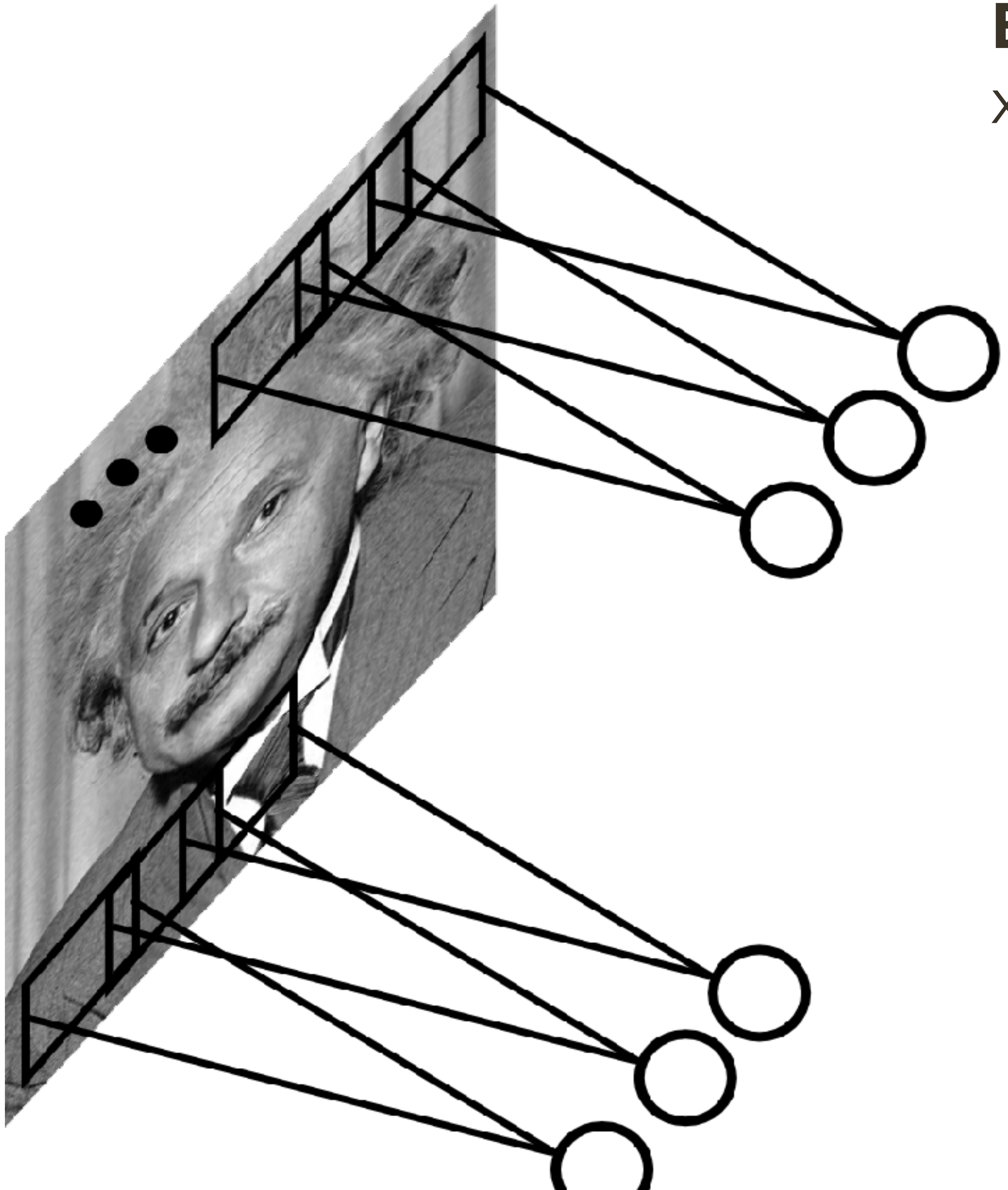
* slide from Marc'Aurelio Renzato

Convolutional Layer



* slide from Marc'Aurelio Renzato

Convolutional Layer



Example: 200 x 200 image (small)
x 40K hidden units (same size)

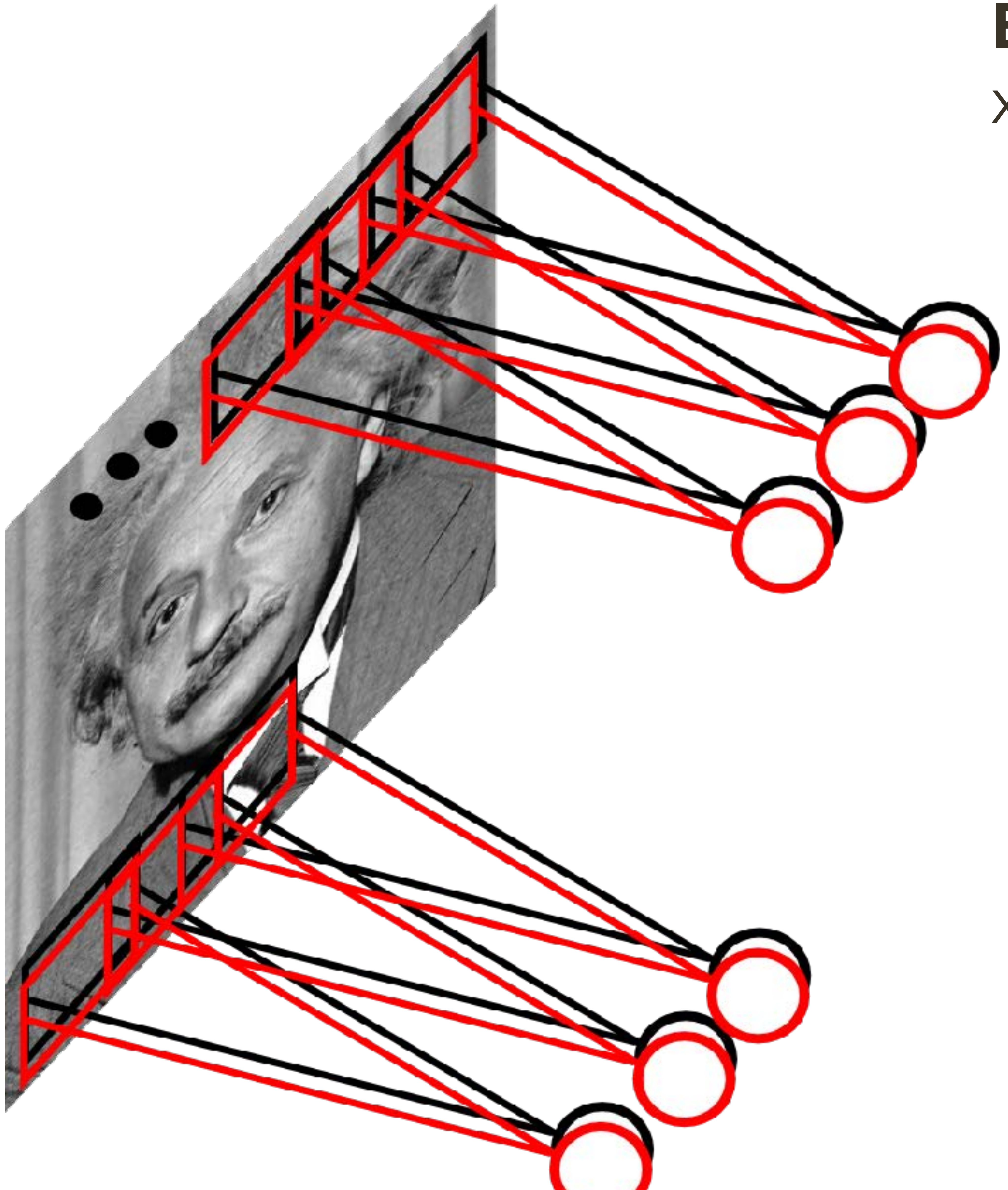
Filter size: 10 x 10

= 100 parameters

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

* slide adopted from Marc'Aurelio Renzato

Convolutional Layer



Example: 200 x 200 image (small)
x 40K hidden units (same size)

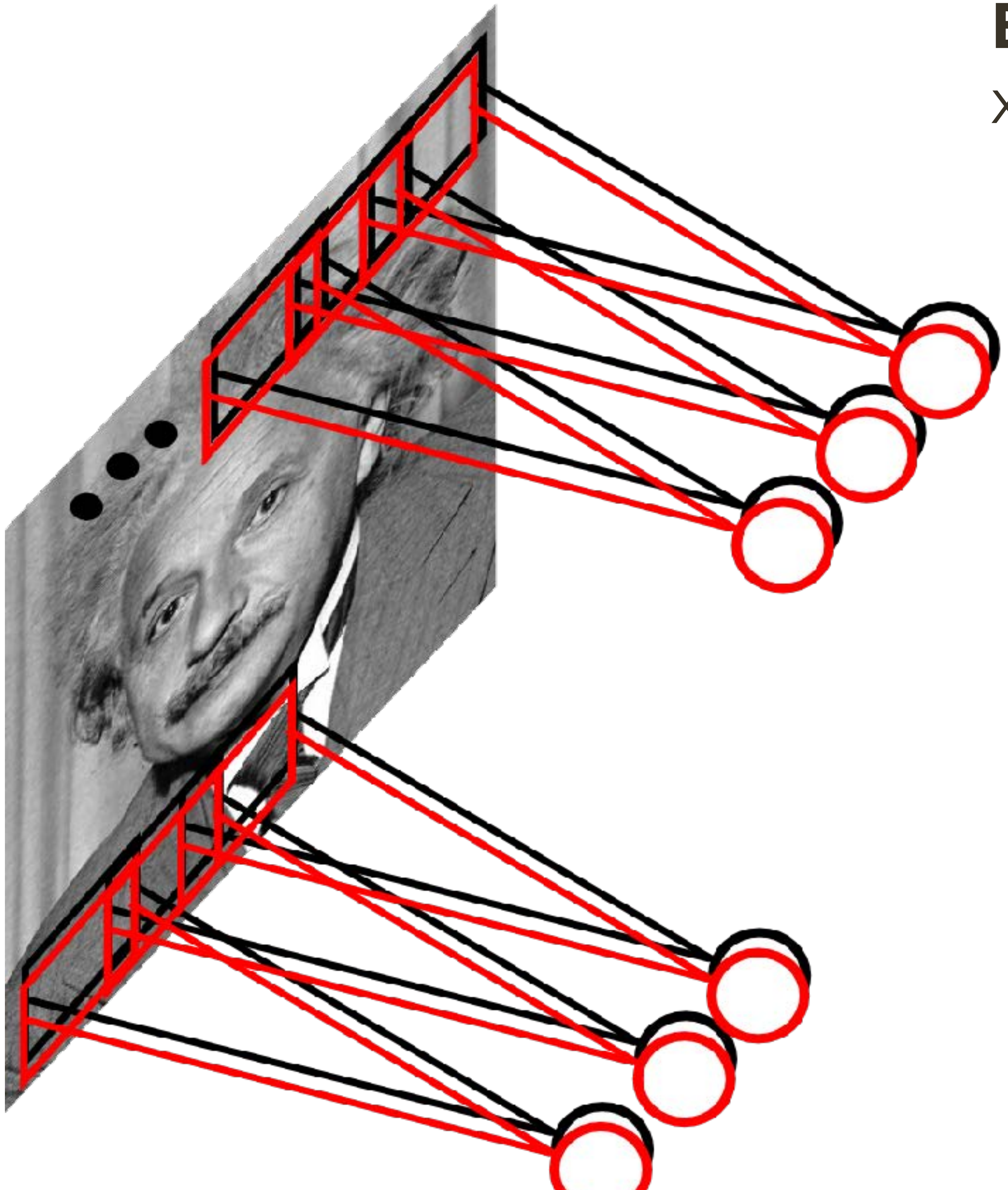
Filter size: 10 x 10

# of filters: 20

Learn **multiple filters**
→ multiple output channels

* slide from Marc'Aurelio Renzato

Convolutional Layer



Example: 200 x 200 image (small)
x 40K hidden units (same size)

Filter size: 10 x 10

# of filters: 20

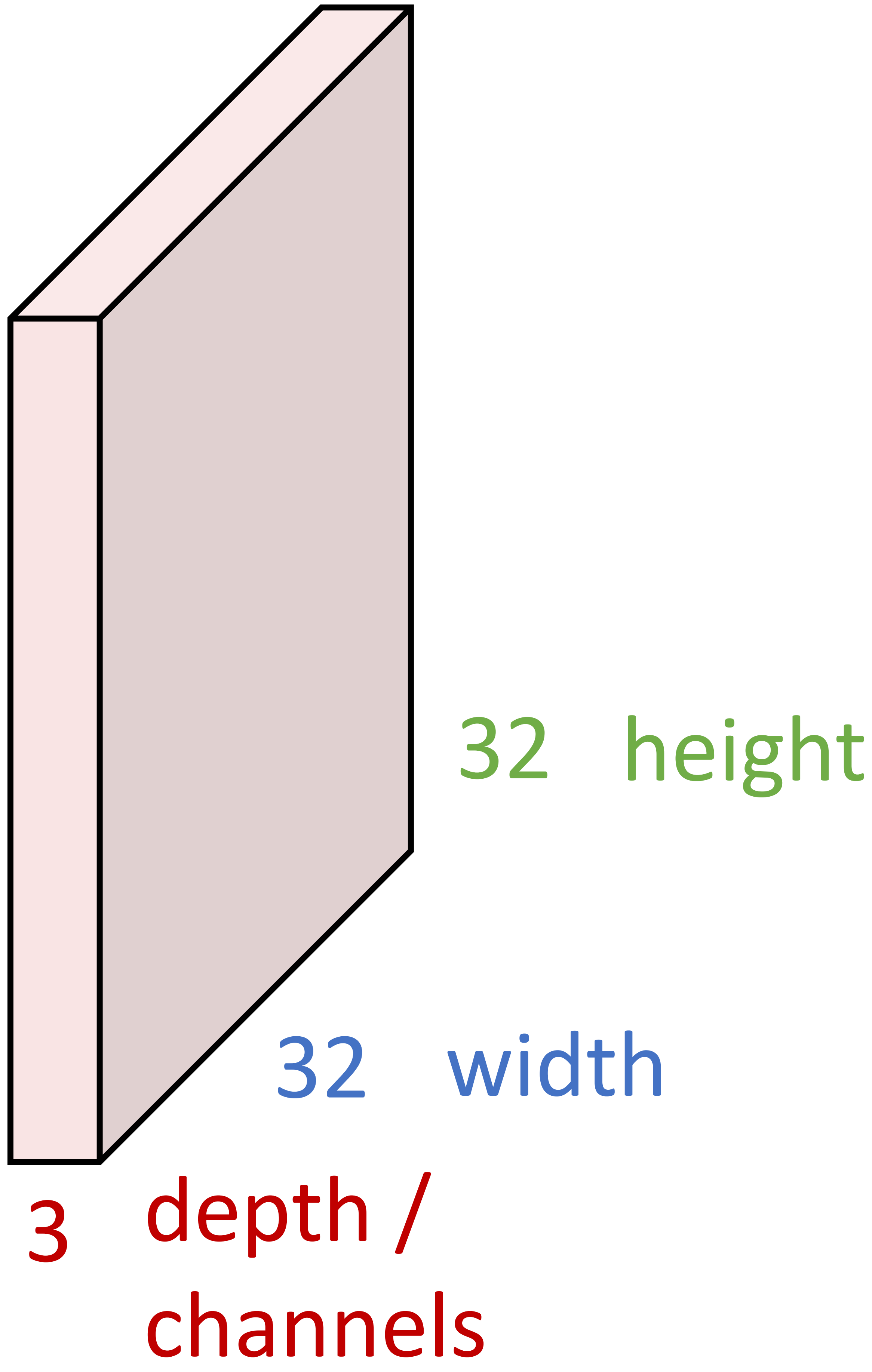
= 2000 parameters

Learn **multiple filters**
→ **multiple output channels**

* slide from Marc'Aurelio Renzato

Convolution Layer

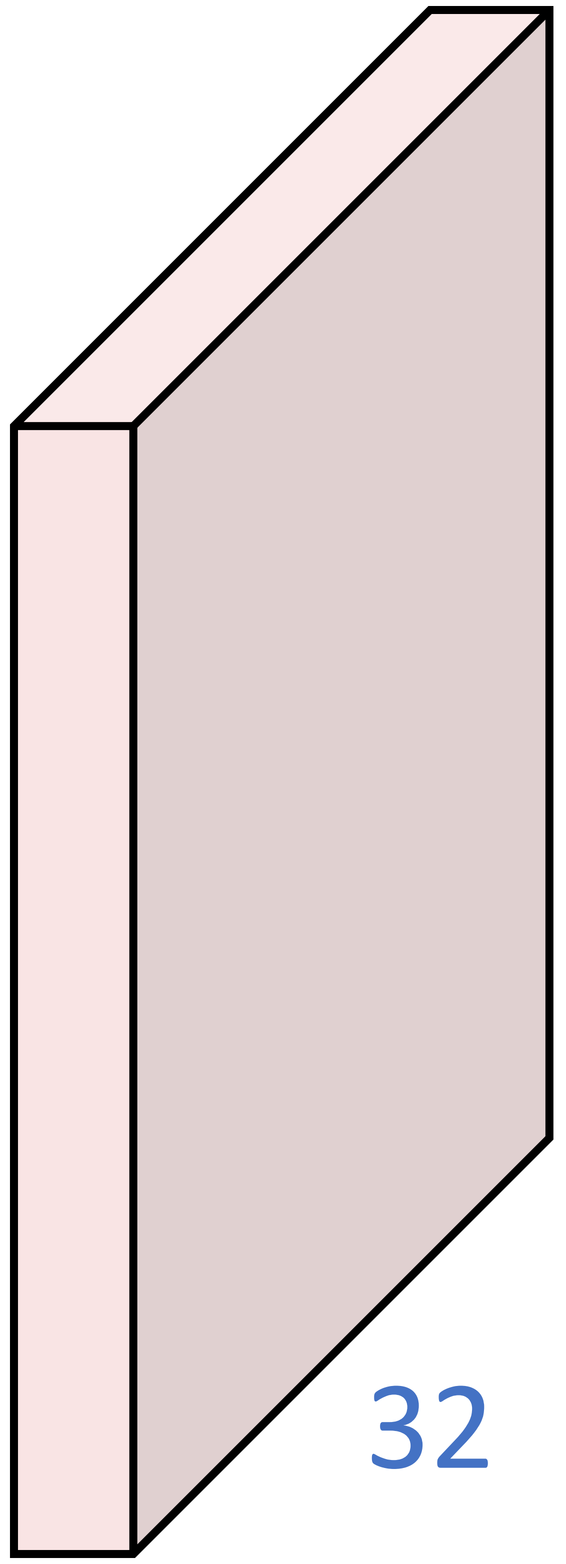
3x32x32 image: preserve spatial structure



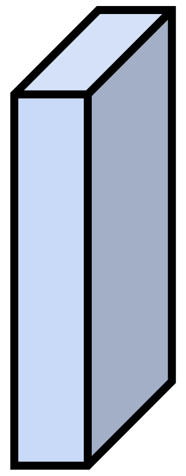
Convolution Layer

Filters always extend the full depth of the input volume

3x32x32 image

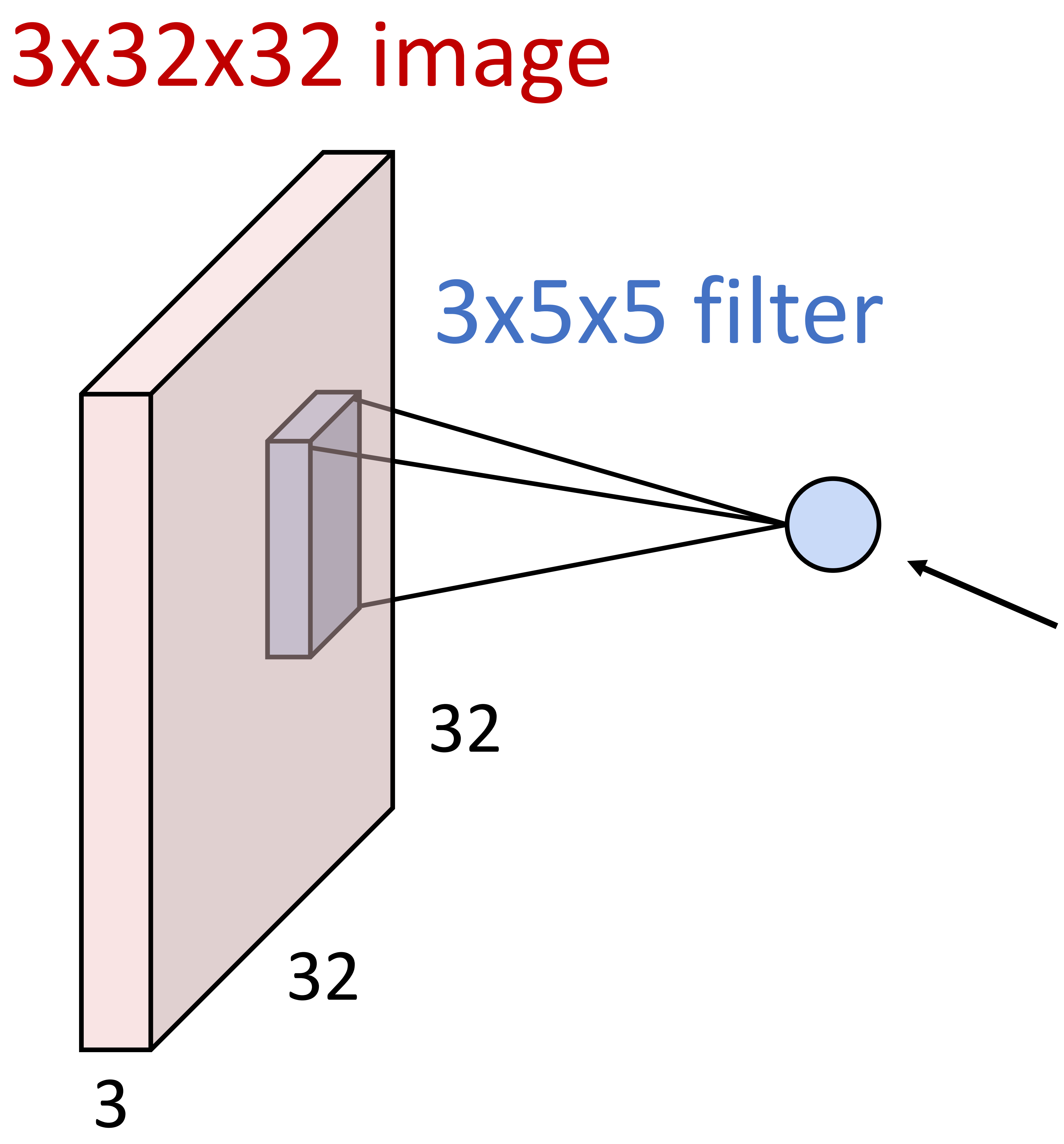


3x5x5 filter



Convolve the filter with the image i.e. "slide over the image spatially, computing dot products"

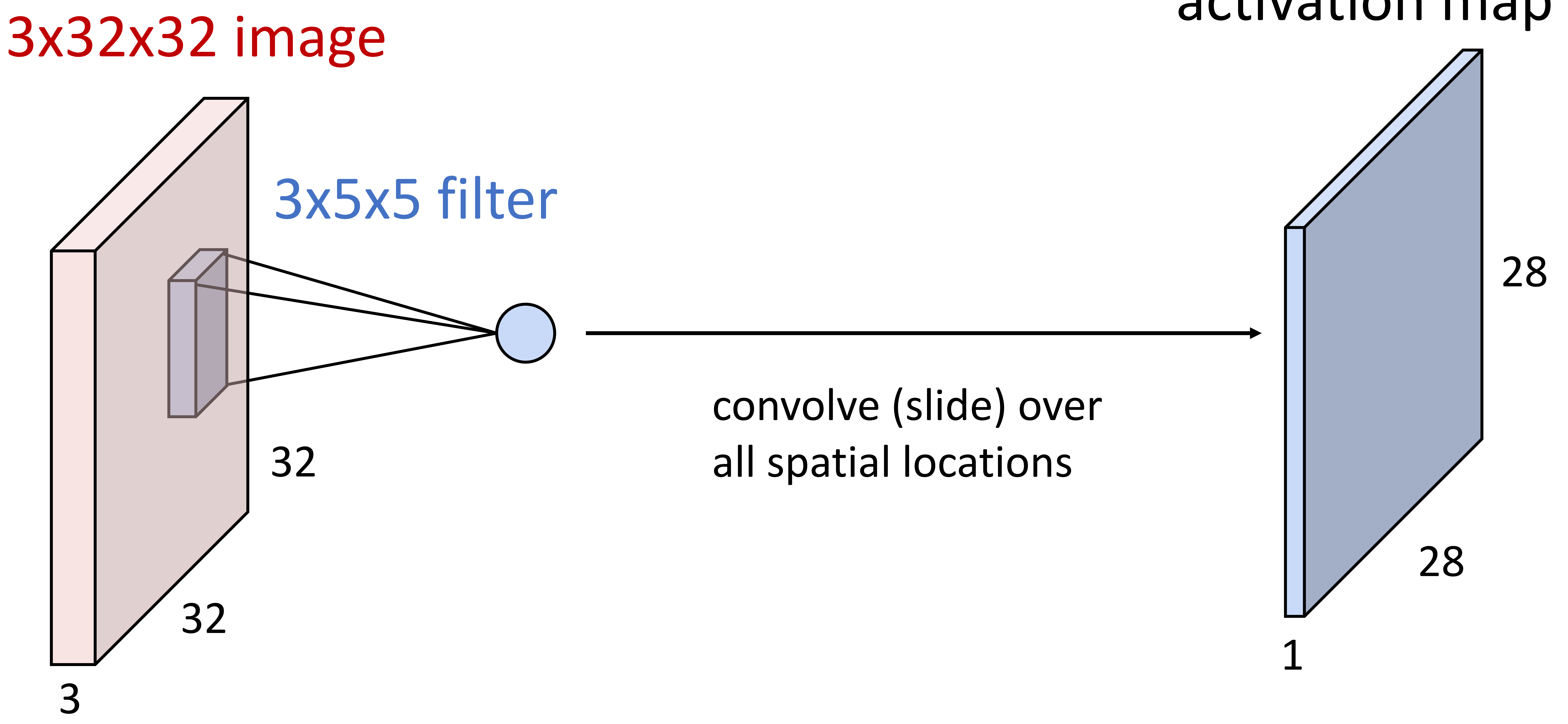
Convolution Layer



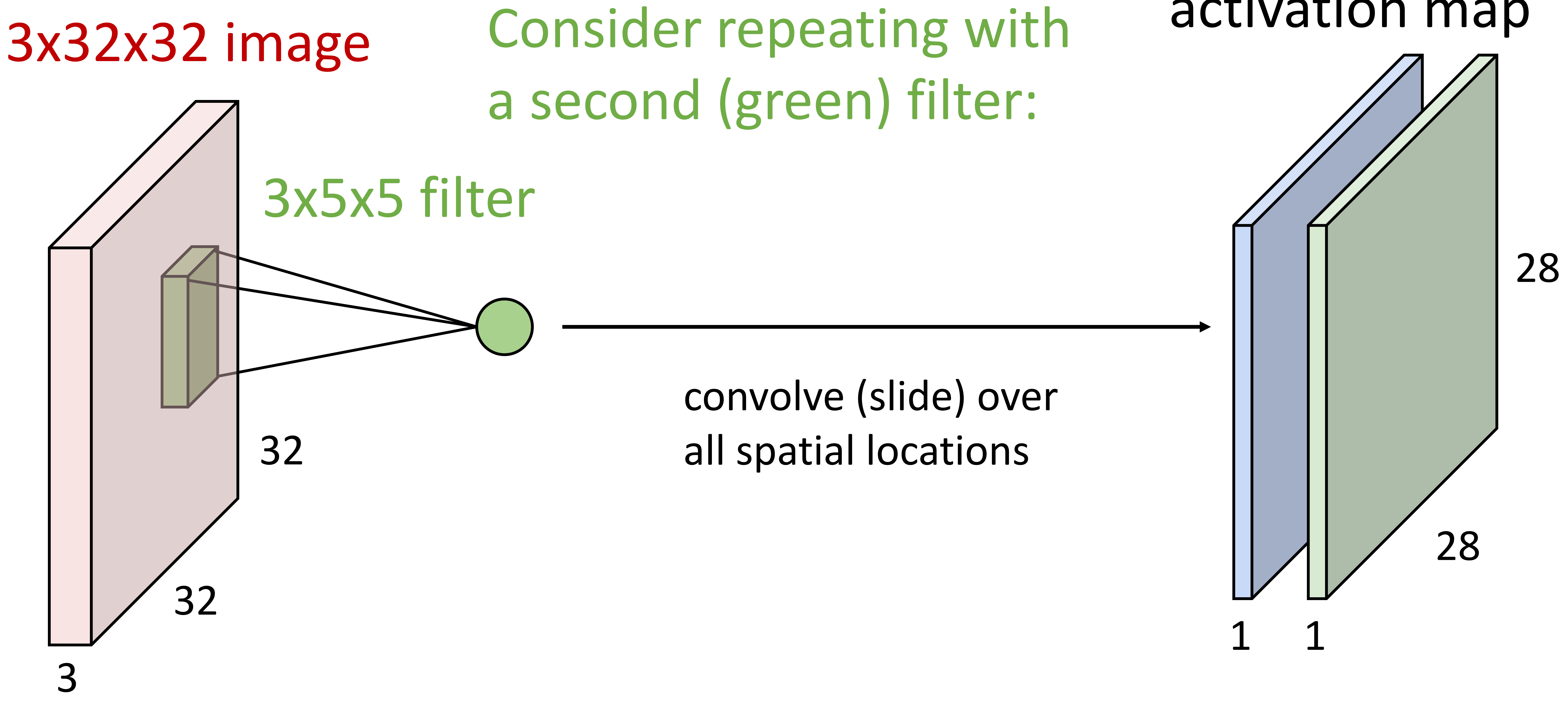
1 number:
the result of taking a dot product between the filter and a small 3x5x5 chunk of the image
(i.e. 3*5*5 = 75-dimensional dot product + bias)

$$w^T x + b$$

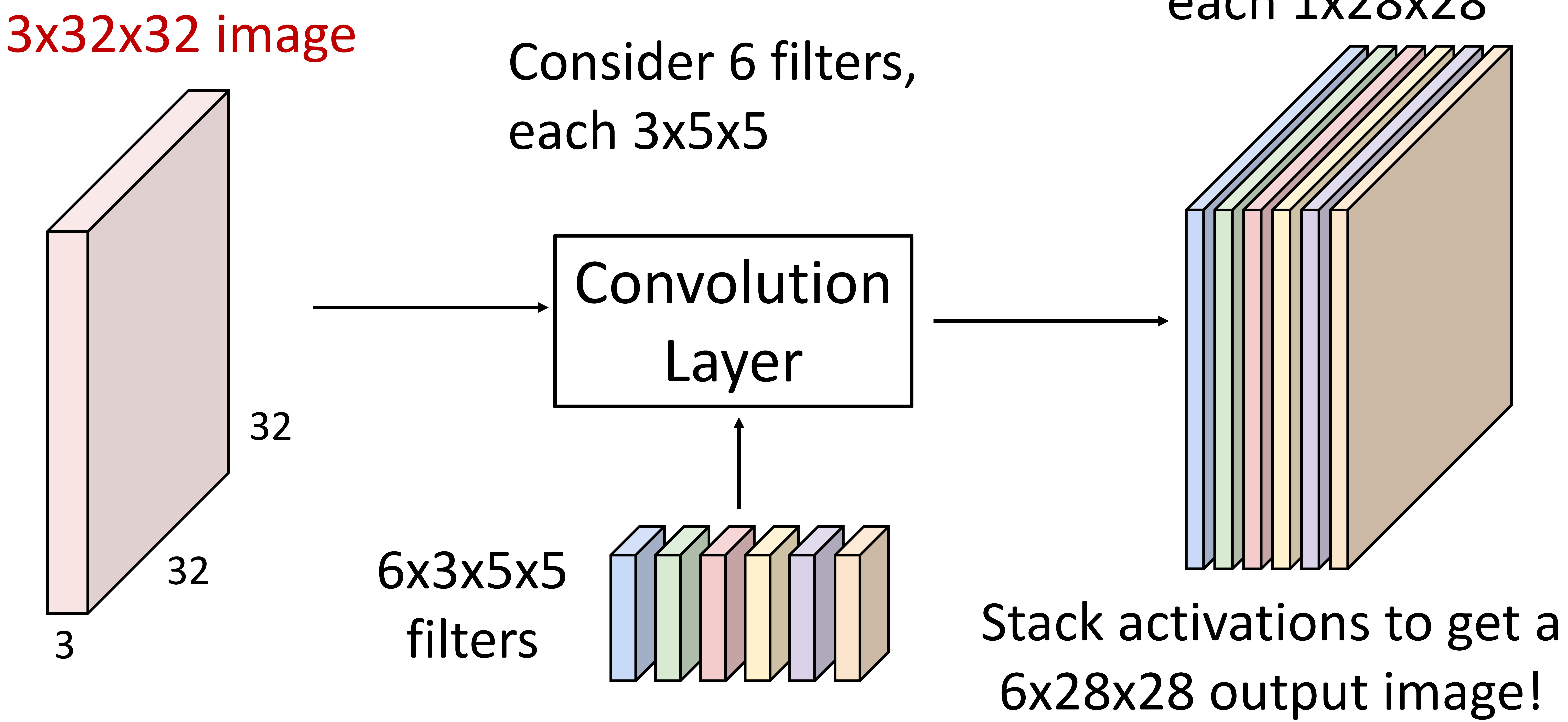
Convolution Layer



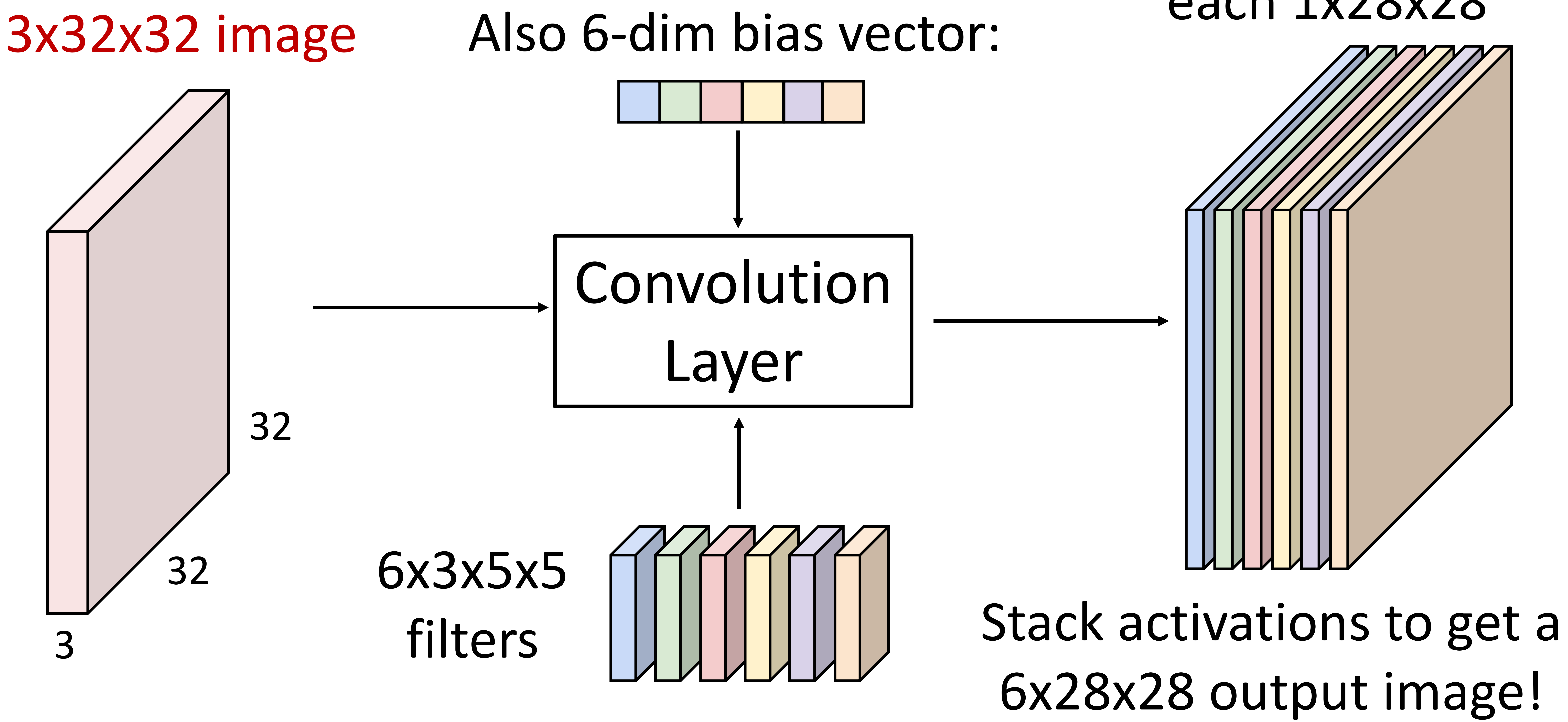
Convolution Layer



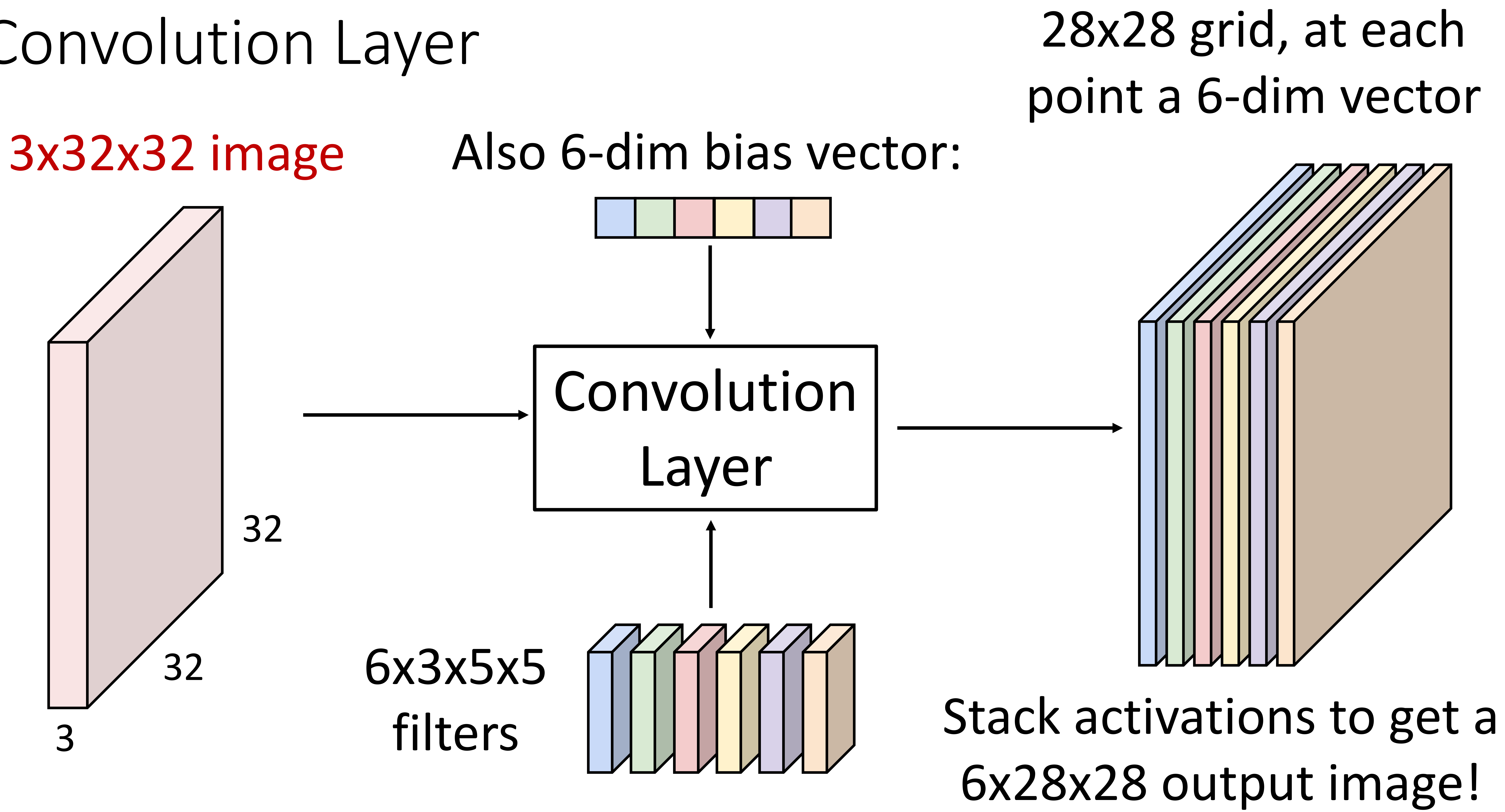
Convolution Layer



Convolution Layer



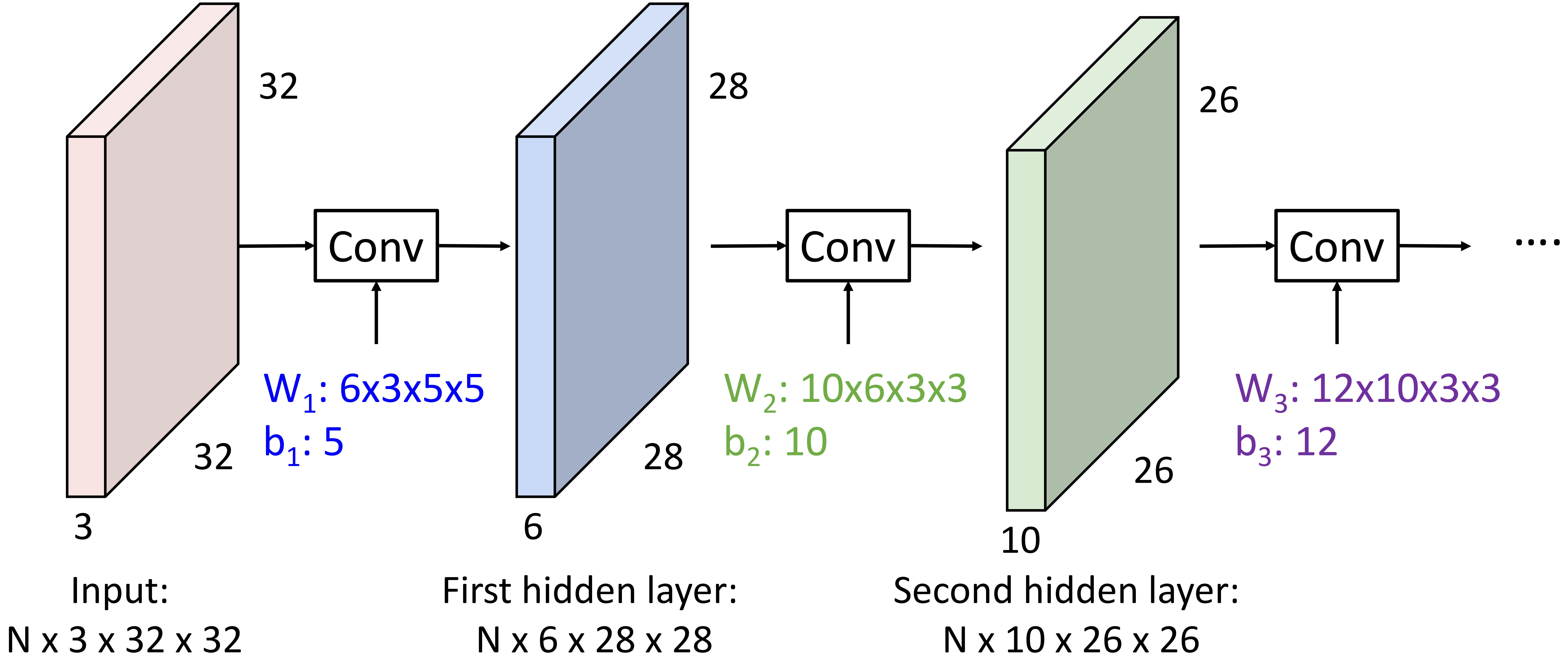
Convolution Layer



Convolution Layer

Convolution Layer

Stacking Convolutions

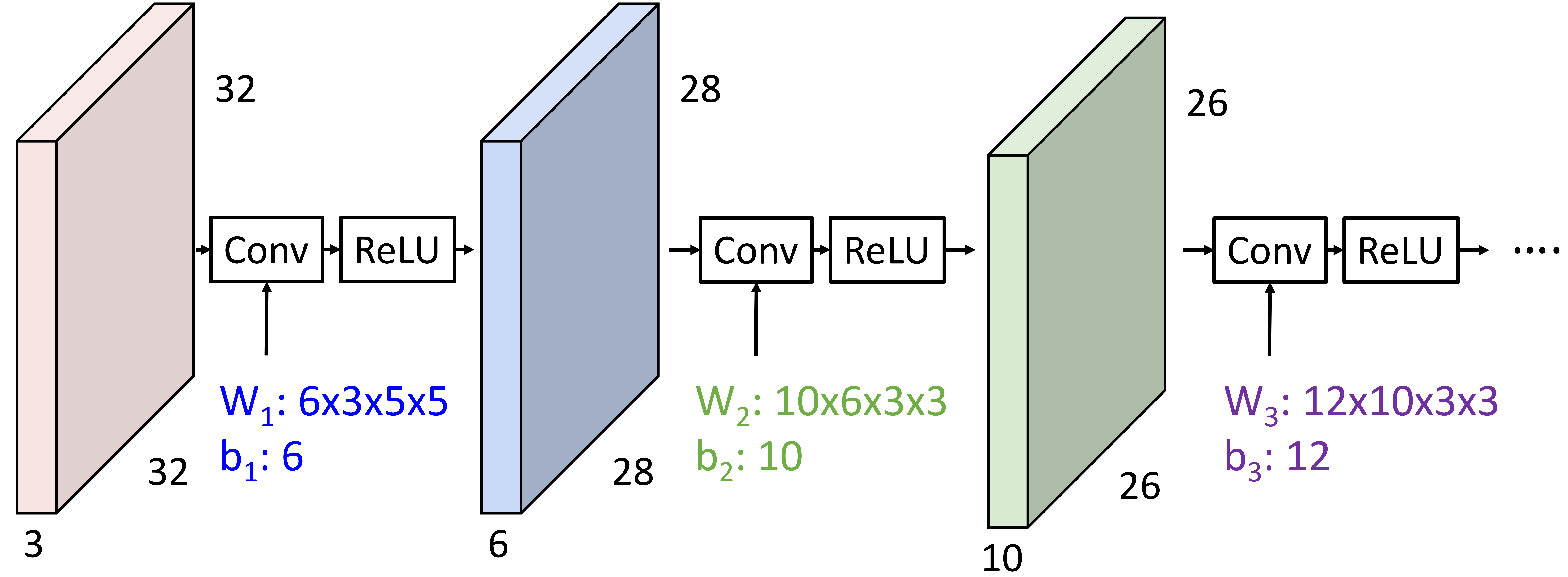


Stacking Convolutions

Q: What happens if we stack two convolution layers?

(Recall $y=W_2W_1x$ is a linear classifier)

A: We get another convolution!

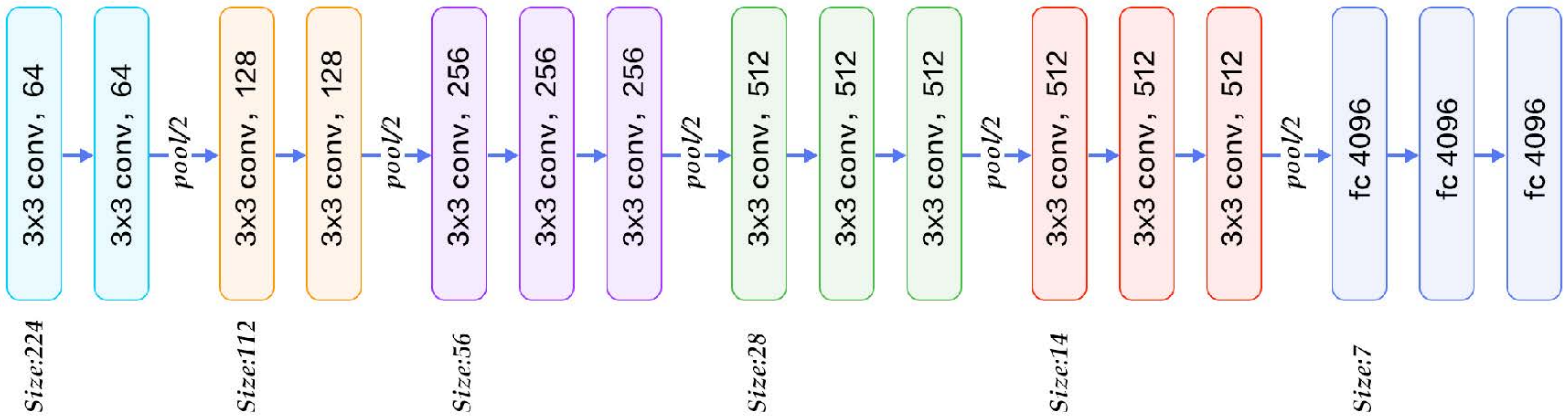


Input:
N x 3 x 32 x 32

First hidden layer:
N x 6 x 28 x 28

Second hidden layer:
N x 10 x 26 x 26

Convolutional Neural Networks



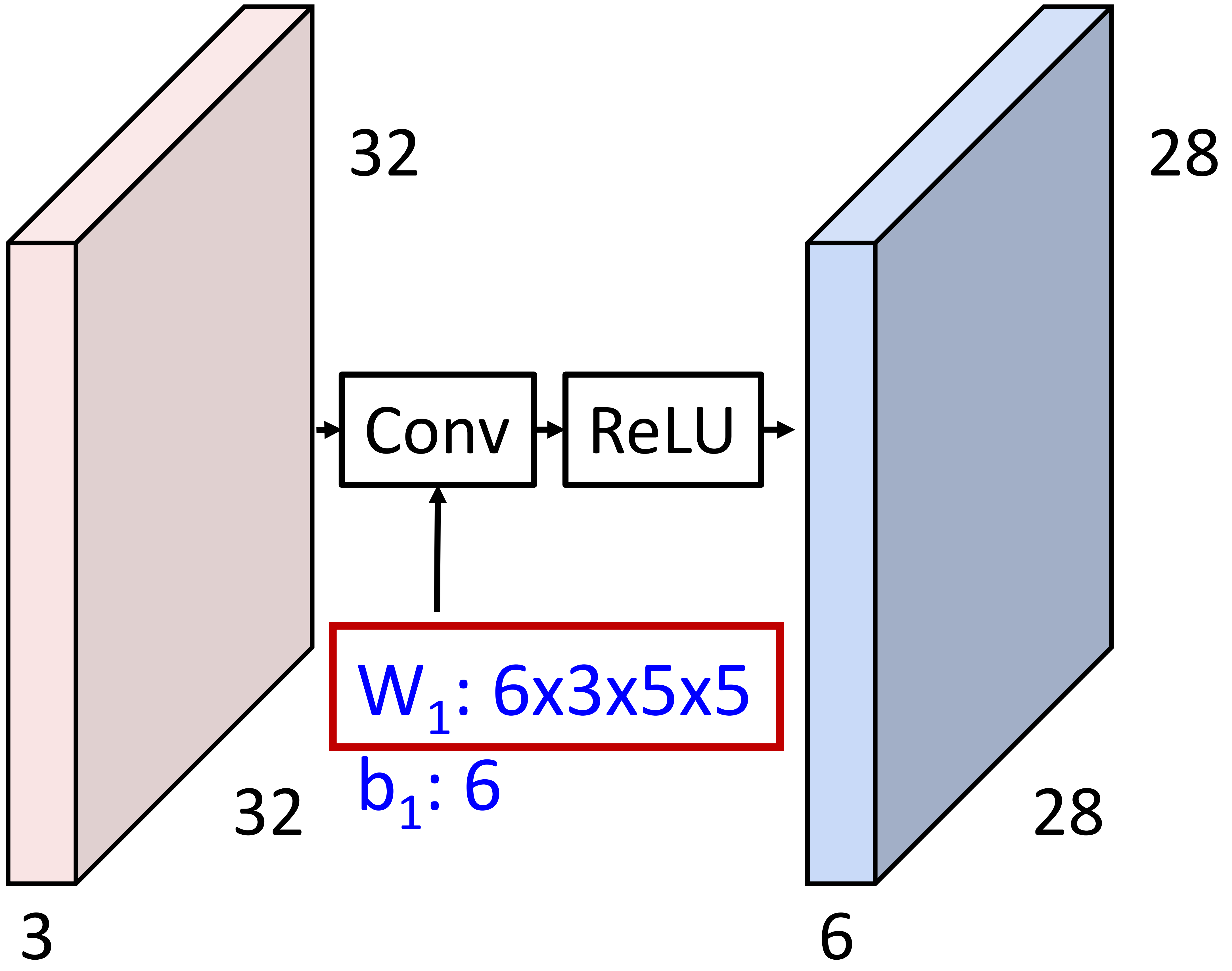
VGG-16 Network

Backward Pass for Some **Common Layers**

Convolutional layer

20.3

What do convolutional filters learn?

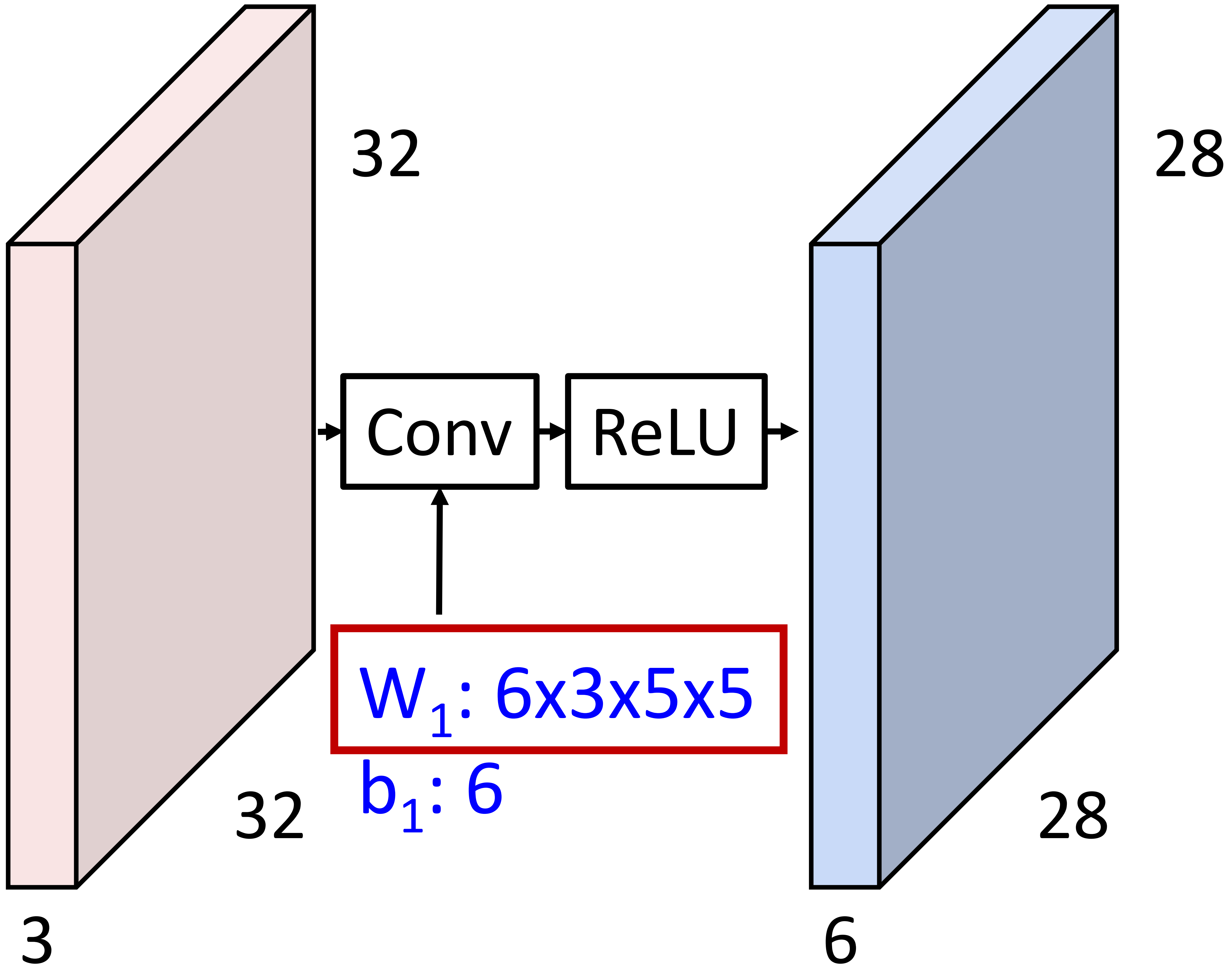


Input:
N x 3 x 32 x 32

First hidden layer:
N x 6 x 28 x 28

Linear classifier: One template per class

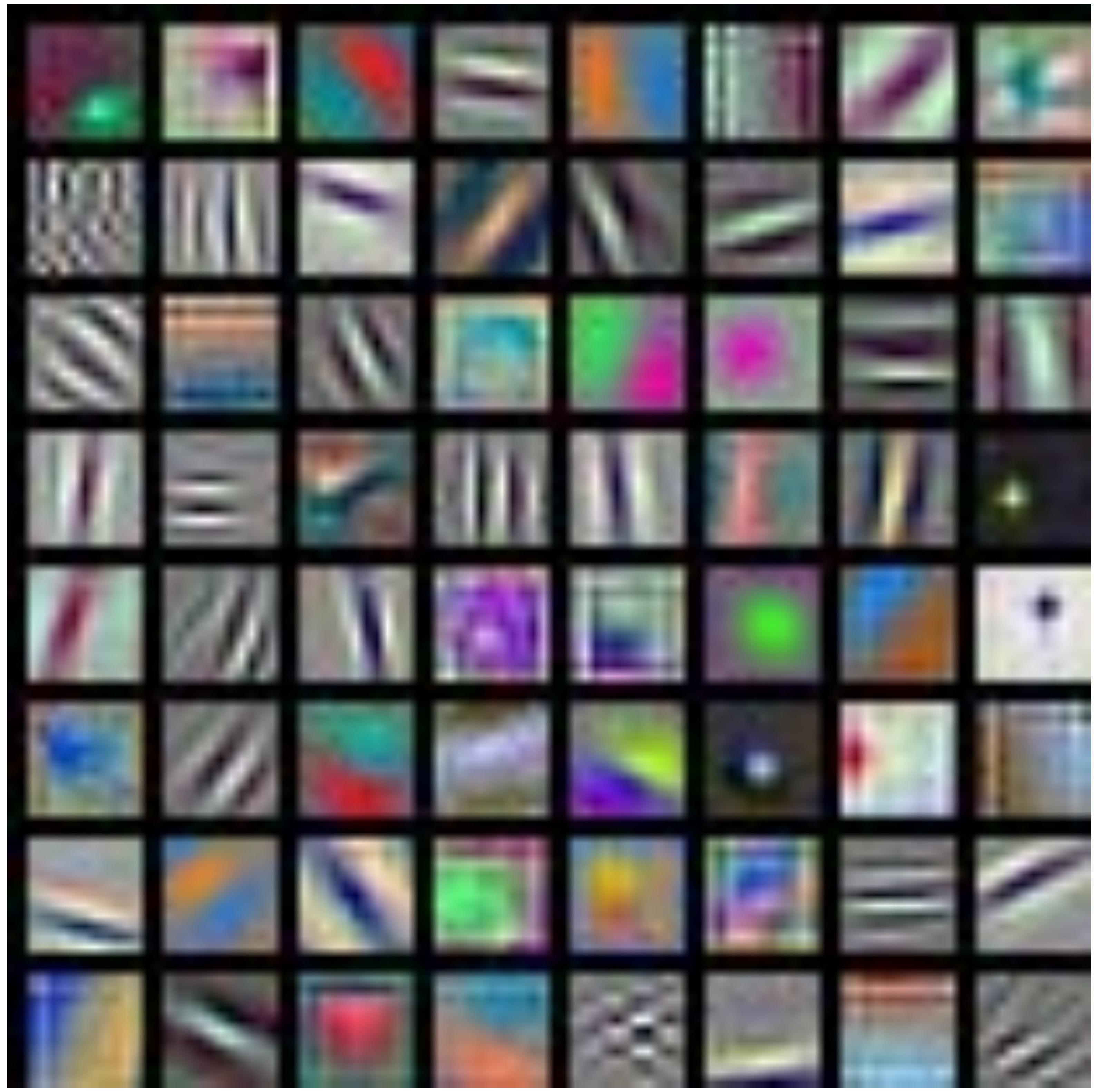
What do convolutional filters learn?



Input:
N x 3 x 32 x 32

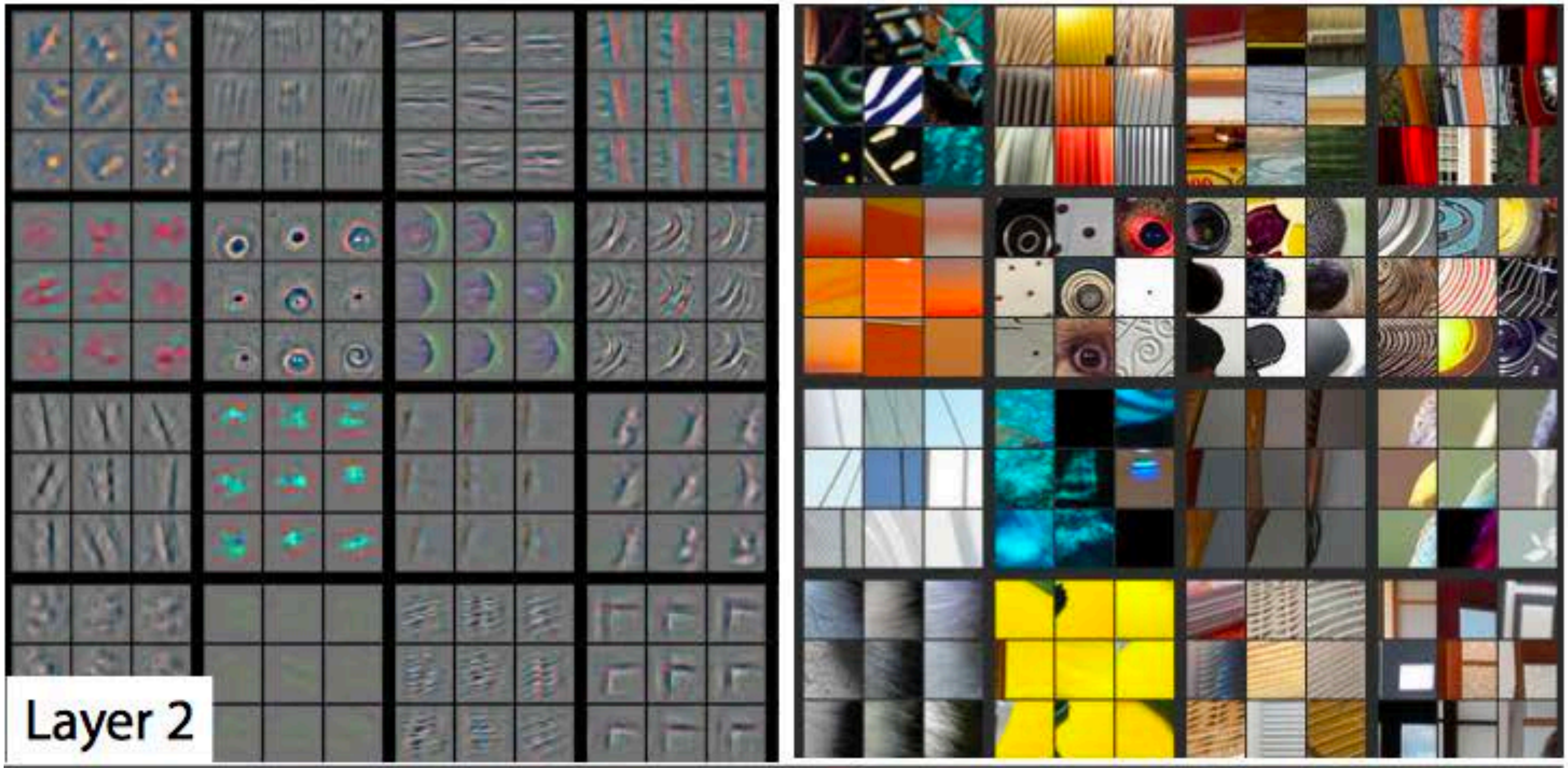
First hidden layer:
N x 6 x 28 x 28

First-layer conv filters: local image templates
(Often learns oriented edges, opposing colors)

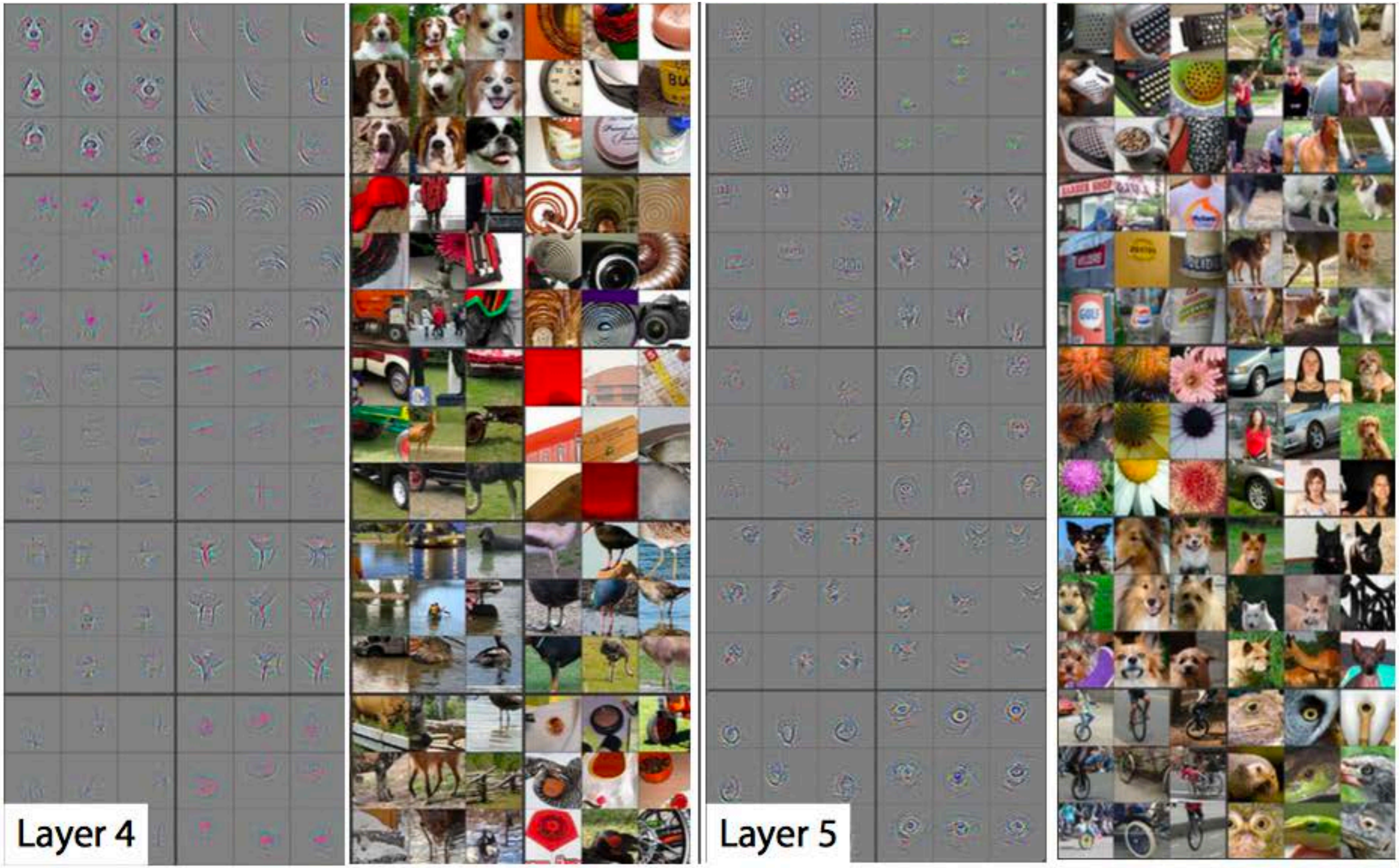


AlexNet: 64 filters, each 3x11x11

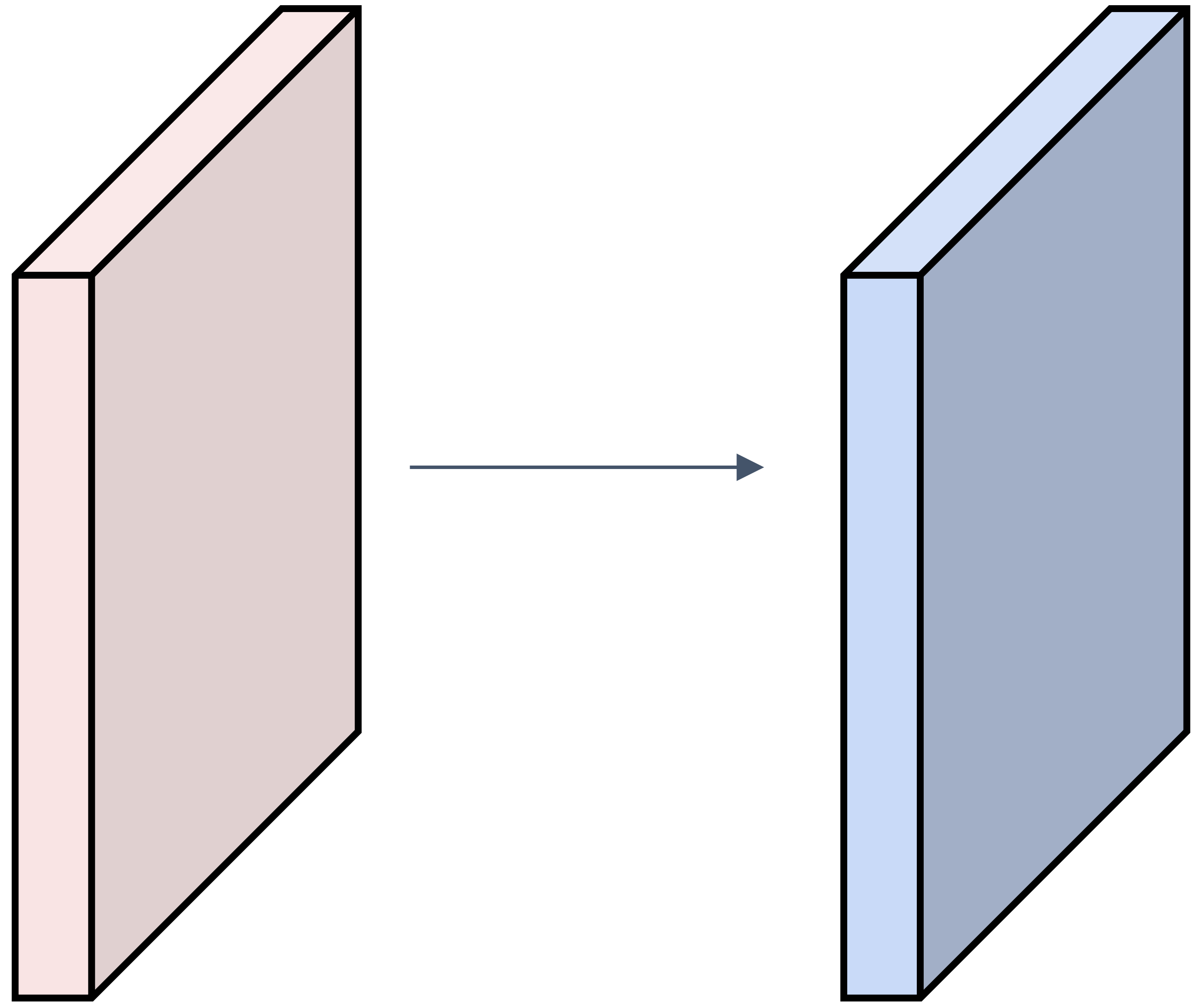
What **filters** do networks learn?



[Zeiler and Fergus, 2013]


Layer 4
Layer 5

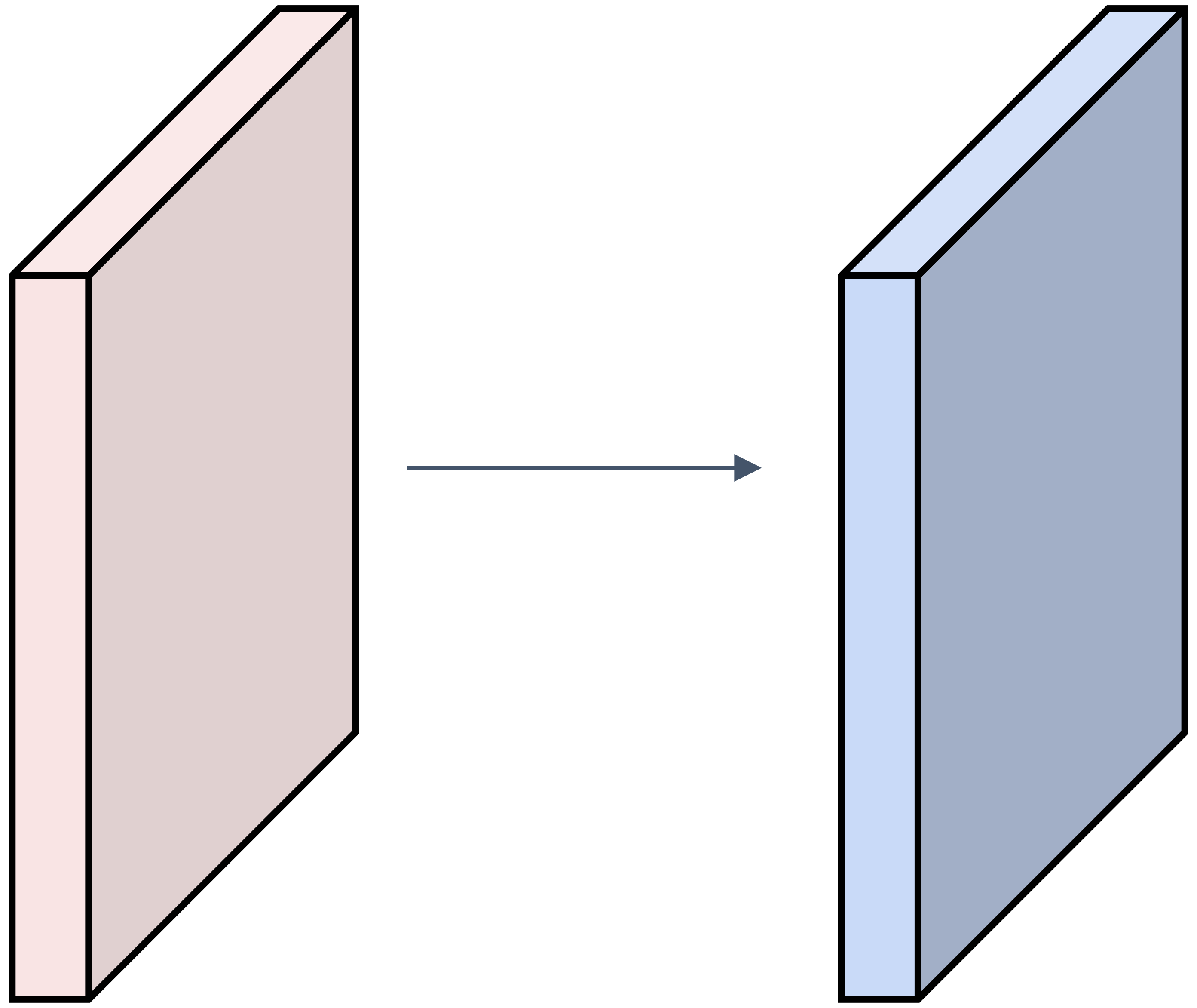
Convolution Example



Input volume: 3 x 32 x 32
10 5x5 filters with stride 1, pad 2

Output volume size: ?

Convolution Example



Input volume: 3 x **32** x **32**

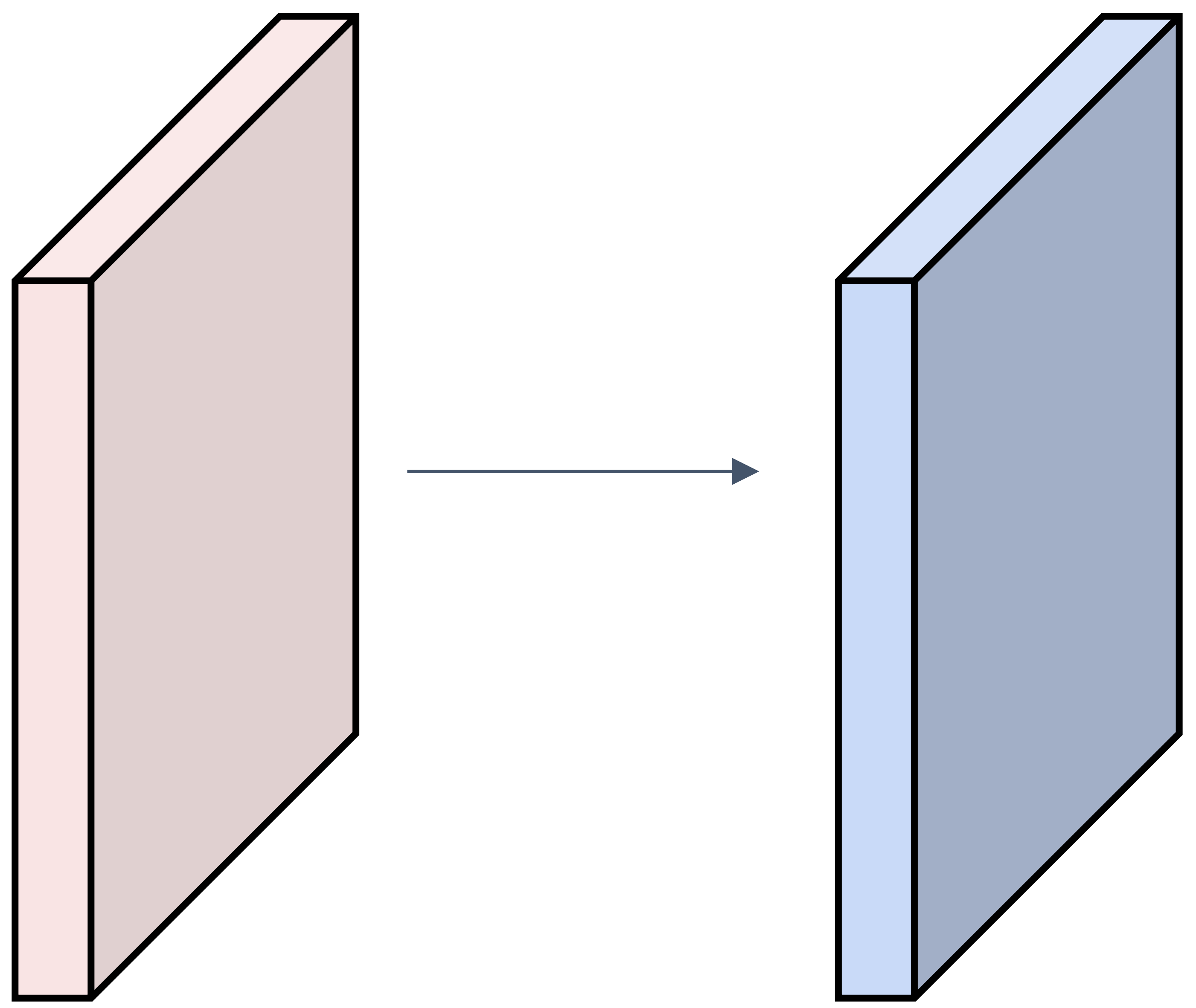
10 5x5 filters with stride **1**, pad **2**

Output volume size:

$(32+2*2-5)/1+1 = 32$ spatially, so

10 x 32 x 32

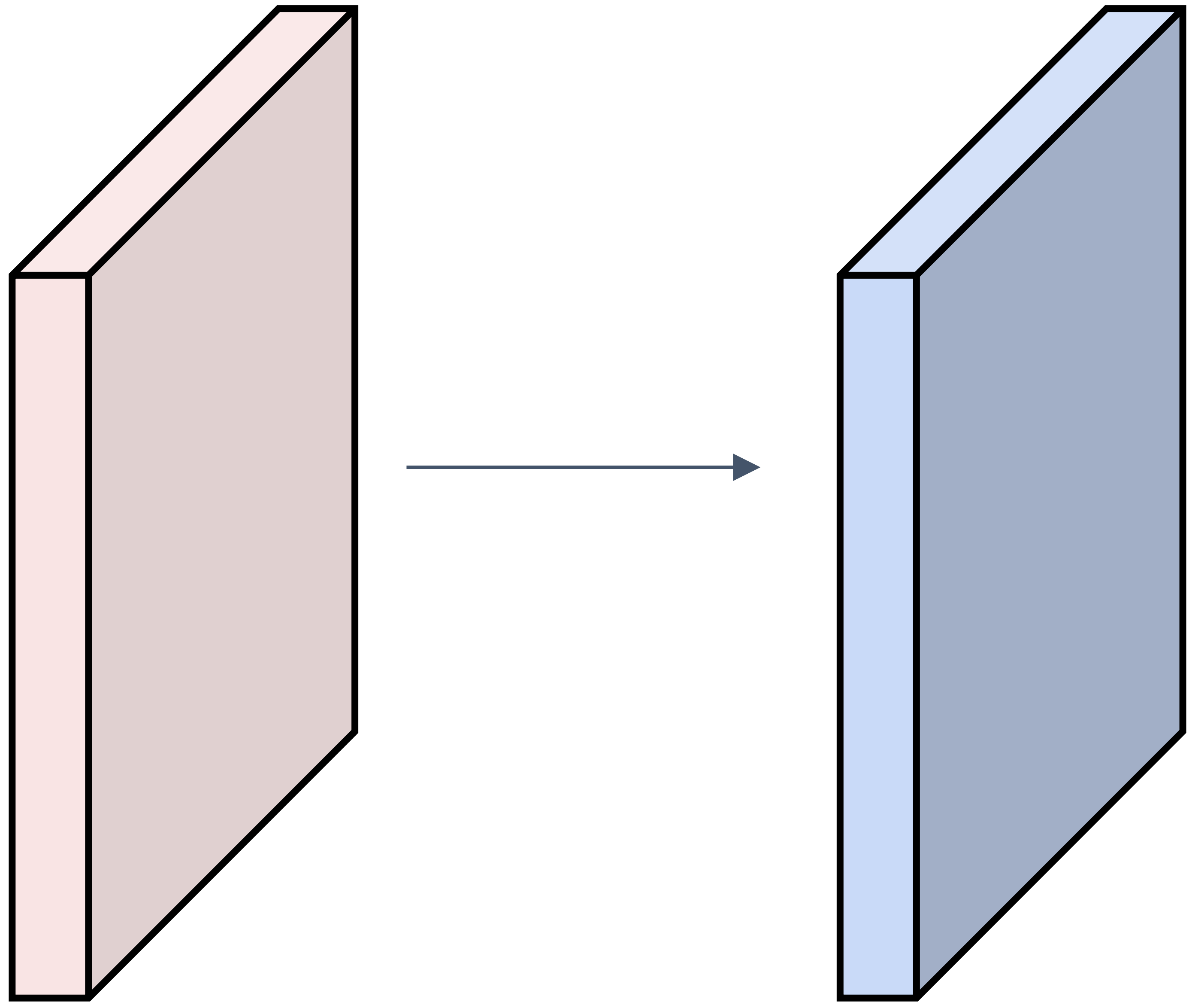
Convolution Example



Input volume: 3 x 32 x 32
10 5x5 filters with stride 1, pad 2

Output volume size: 10 x 32 x 32
Number of learnable parameters: ?

Convolution Example



Input volume: 3 x 32 x 32

10 5x5 filters with stride 1, pad 2

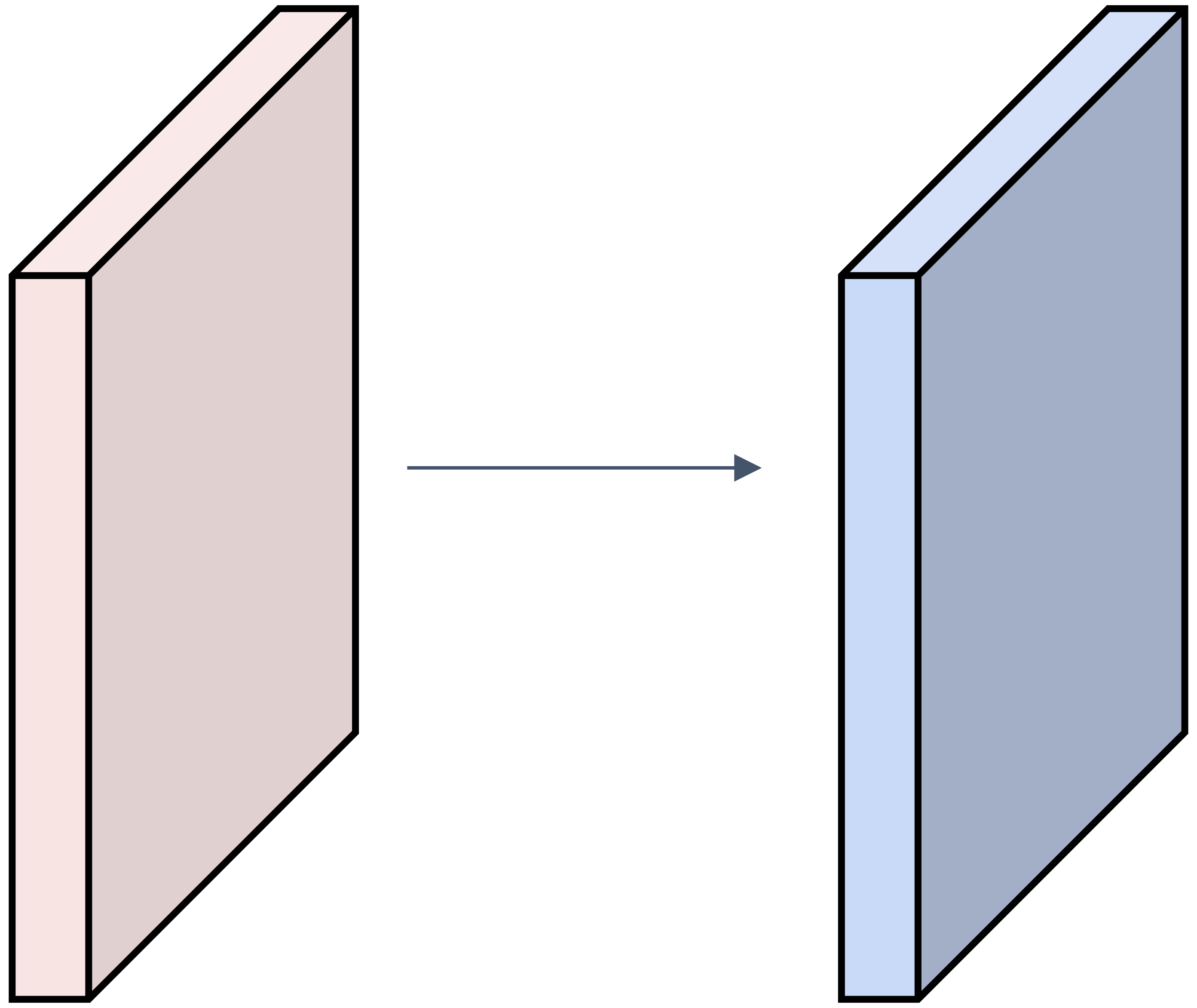
Output volume size: 10 x 32 x 32

Number of learnable parameters: **760**

Parameters per filter: 3*5*5 + 1 (for bias) = 76

10 filters, so total is 10 * 76 = **760**

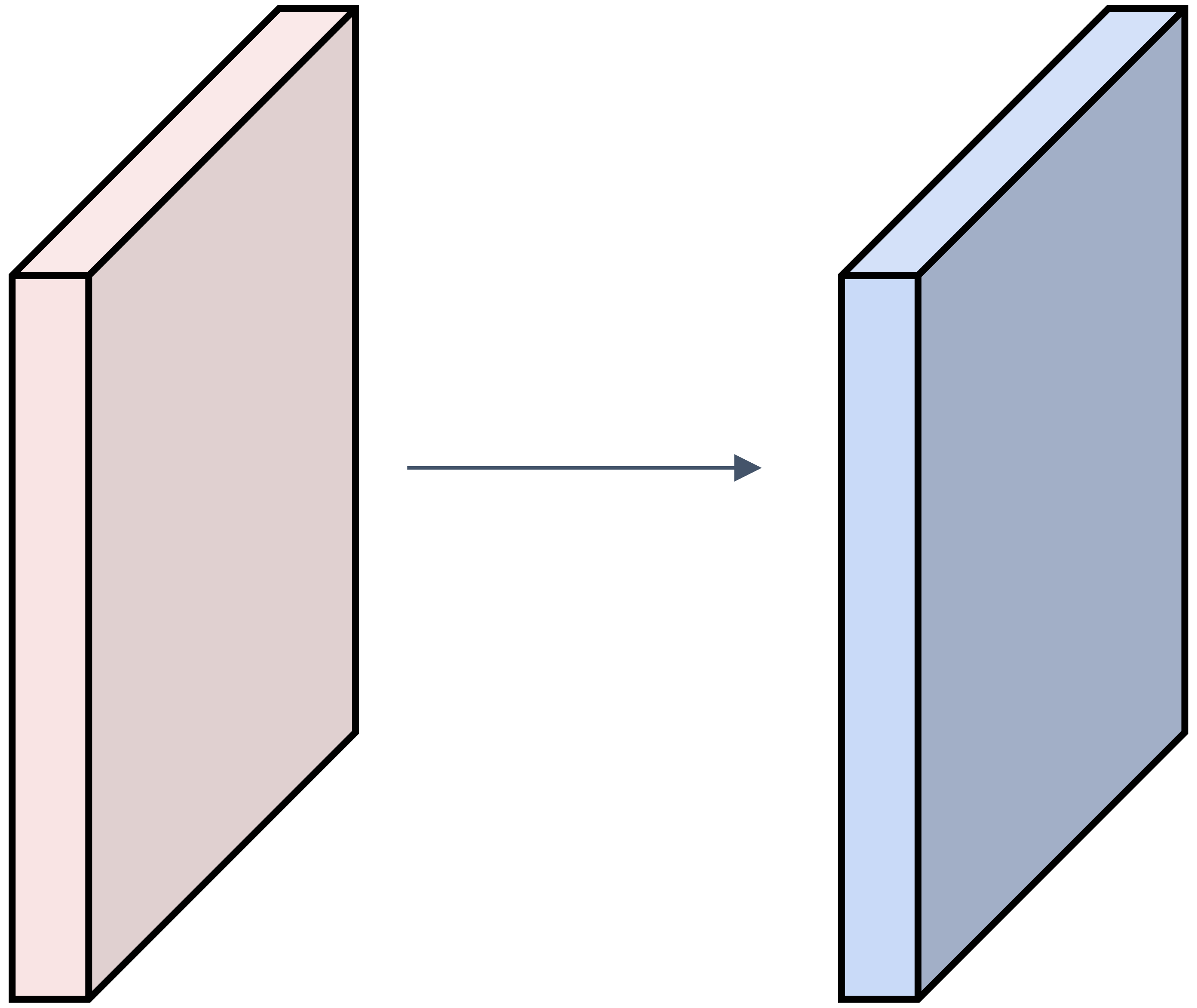
Convolution Example



Input volume: 3 x 32 x 32
10 5x5 filters with stride 1, pad 2

Output volume size: 10 x 32 x 32
Number of learnable parameters: 760
Number of multiply-add operations: ?

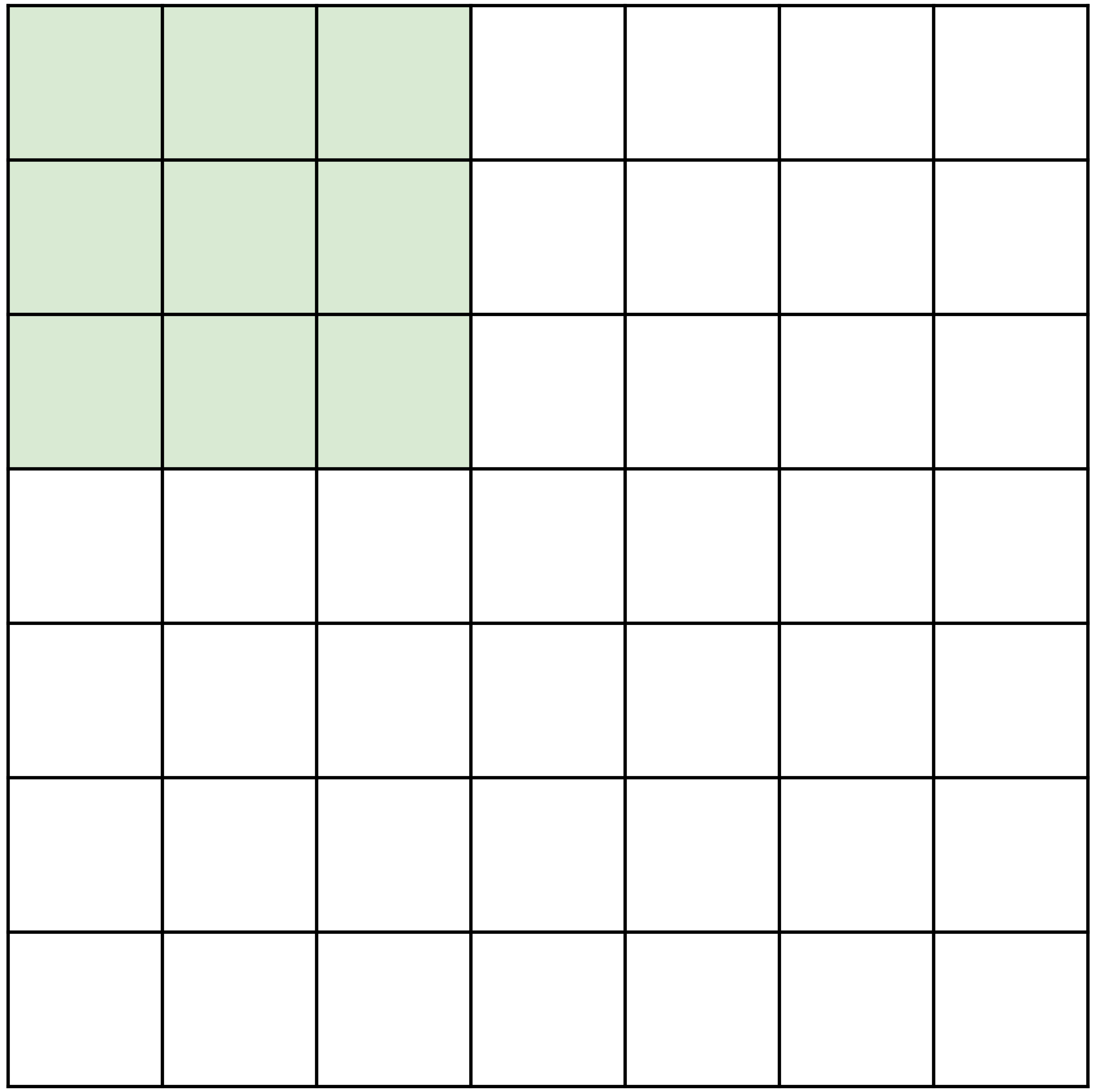
Convolution Example



Input volume: **3** x 32 x 32
10 **5x5** filters with stride 1, pad 2

Output volume size: **10 x 32 x 32**
Number of learnable parameters: 760
Number of multiply-add operations: **768,000**
10*32*32 = 10,240 outputs; each output is the inner product of two **3**x**5x5** tensors (75 elems); total = 75*10240 = **768K**

Strided Convolution

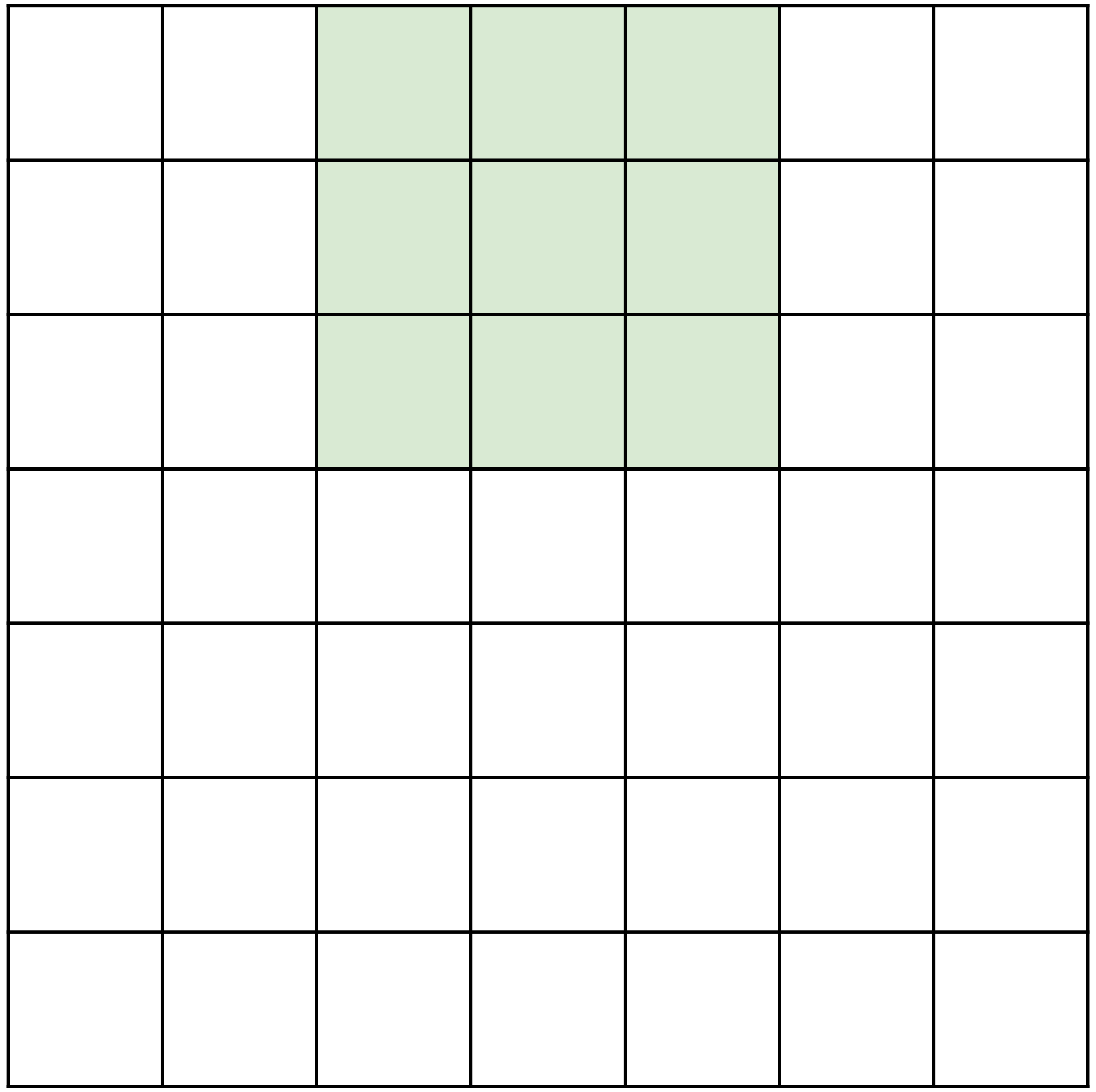


Input: 7x7

Filter: 3x3

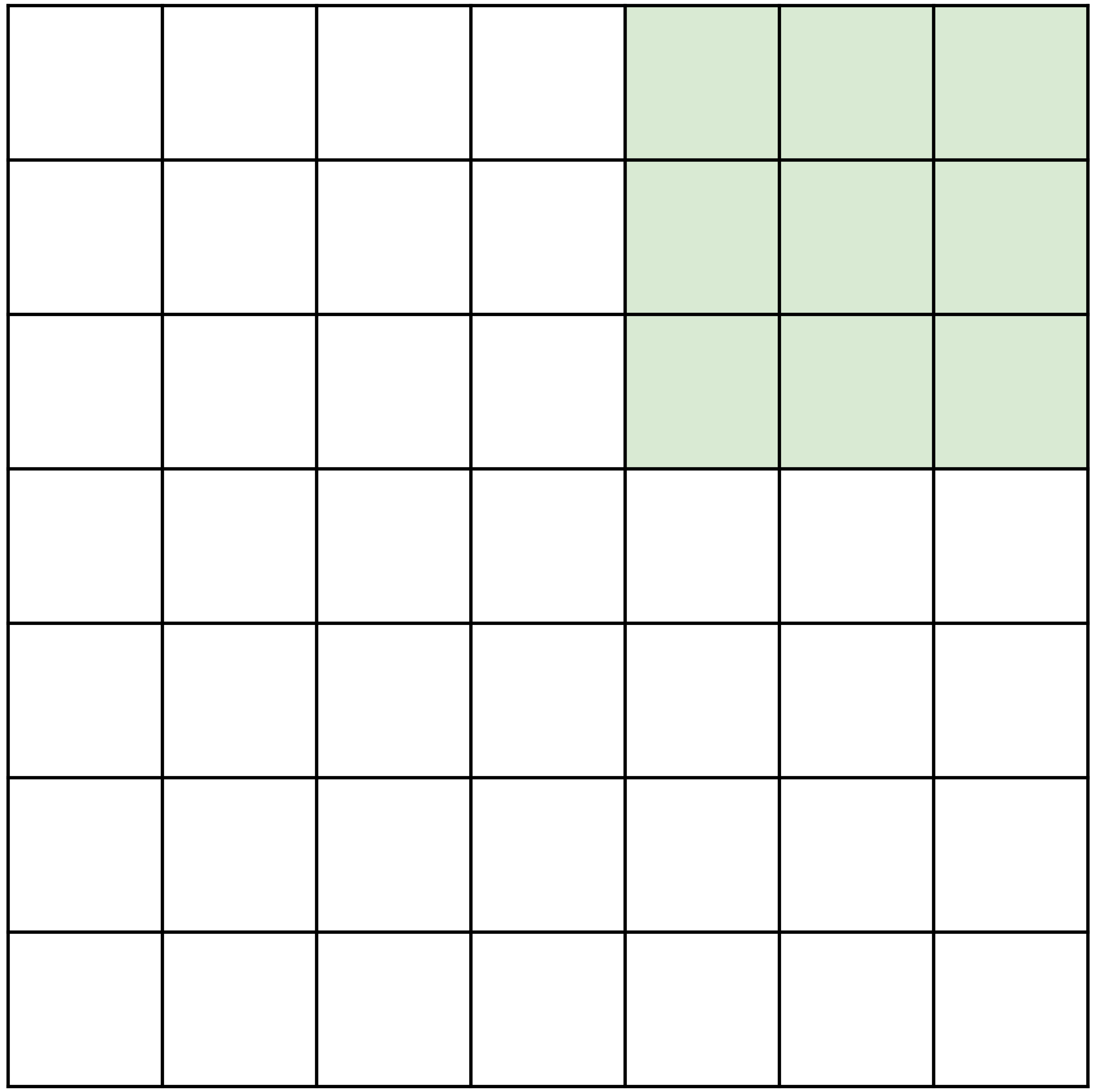
Stride: 2

Strided Convolution



Input: 7x7
Filter: 3x3
Stride: 2

Strided Convolution



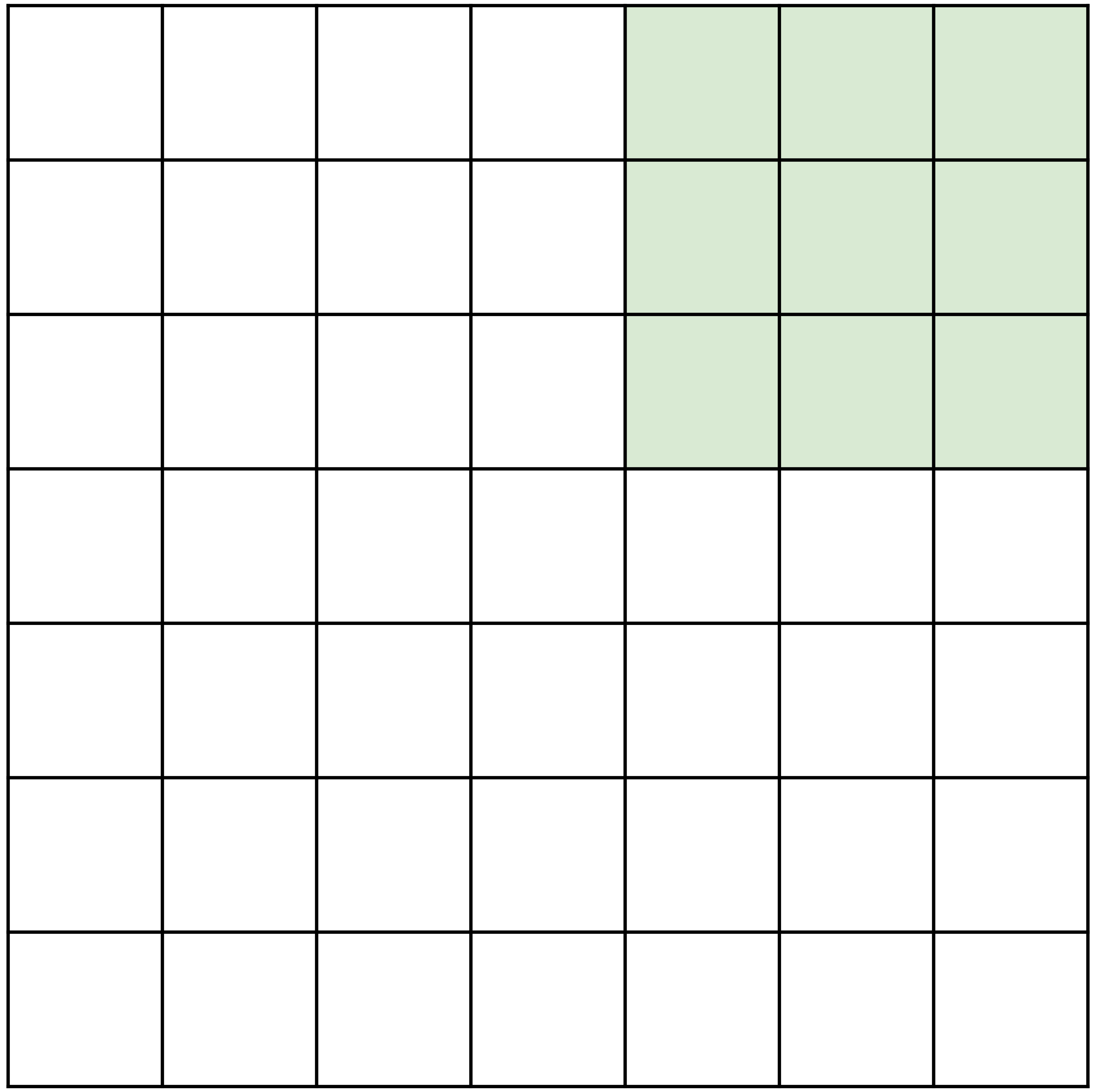
Input: 7x7

Filter: 3x3

Stride: 2

Output: 3x3

Strided Convolution



Input: 7x7

Filter: 3x3

Stride: 2

Output: 3x3

In general:

Input: W

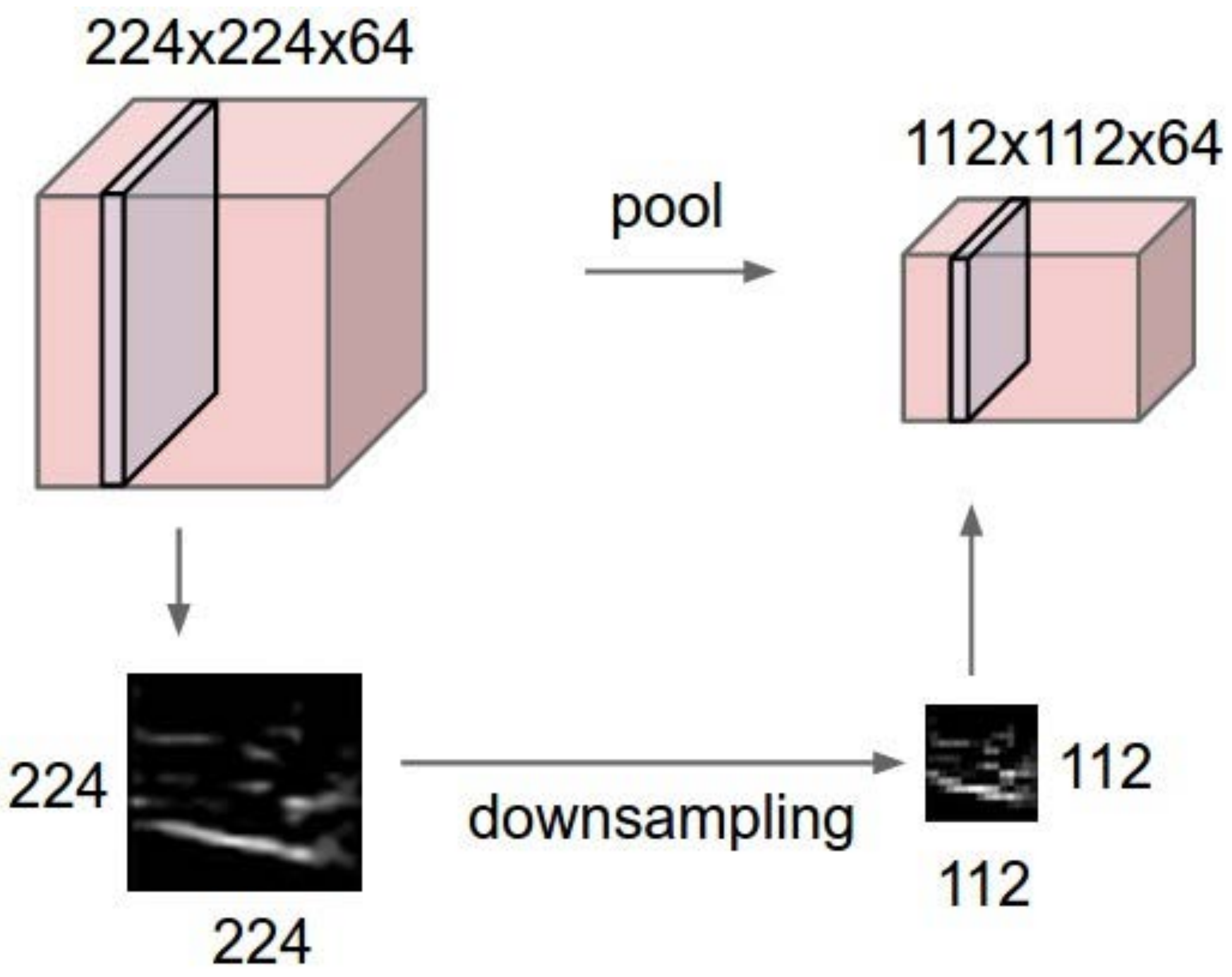
Filter: K

Padding: P

Stride: S

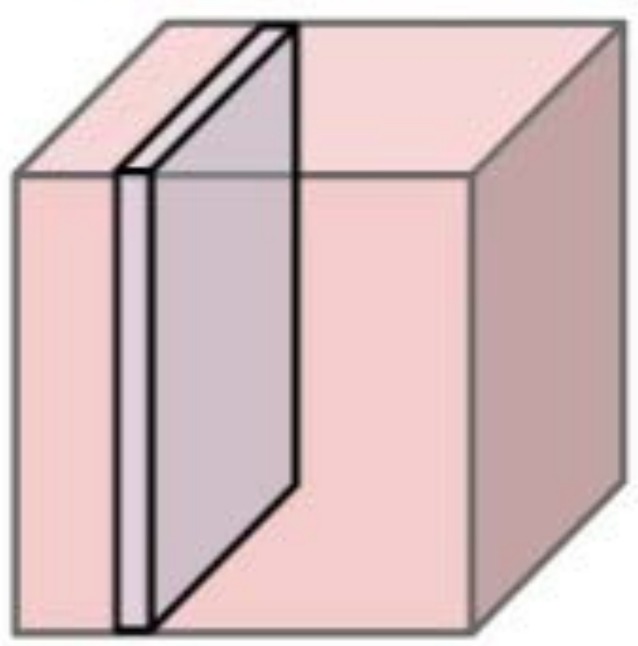
Output: $(W - K + 2P) / S + 1$

Pooling Layers: Another way to downsample



Hyperparameters:
Kernel Size
Stride
Pooling function

Max Pooling



Single depth slice

x

1	1	2	4
5	**6**	7	**8**
3	2	1	0
1	2	3	**4**

y

Max pooling with 2x2 kernel size and stride 2

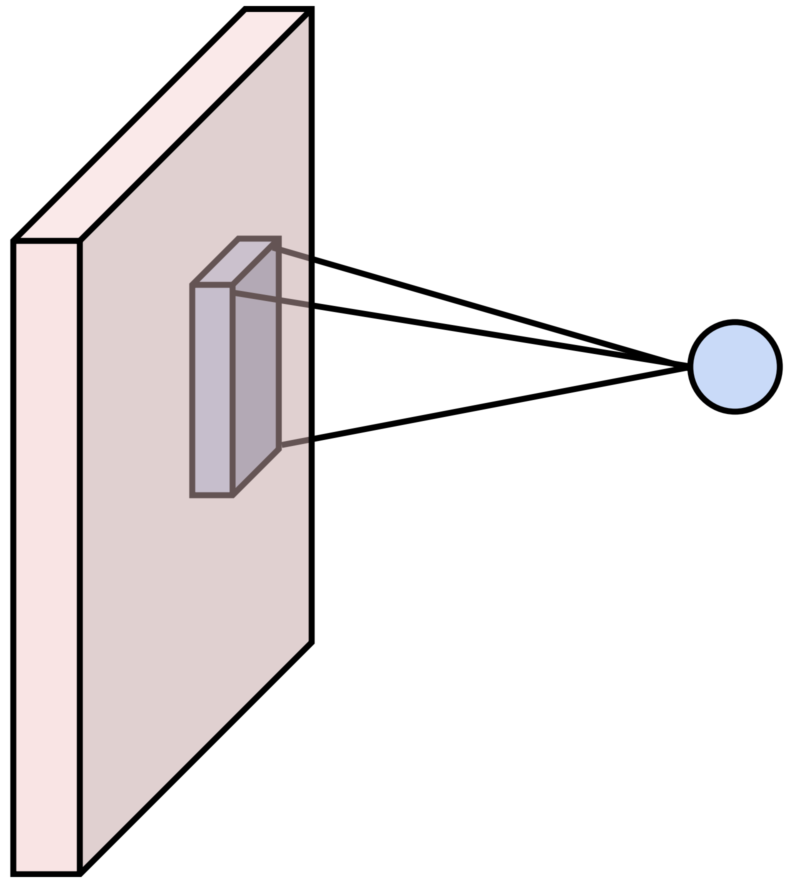
6	8
3	4

Introduces **invariance** to small spatial shifts

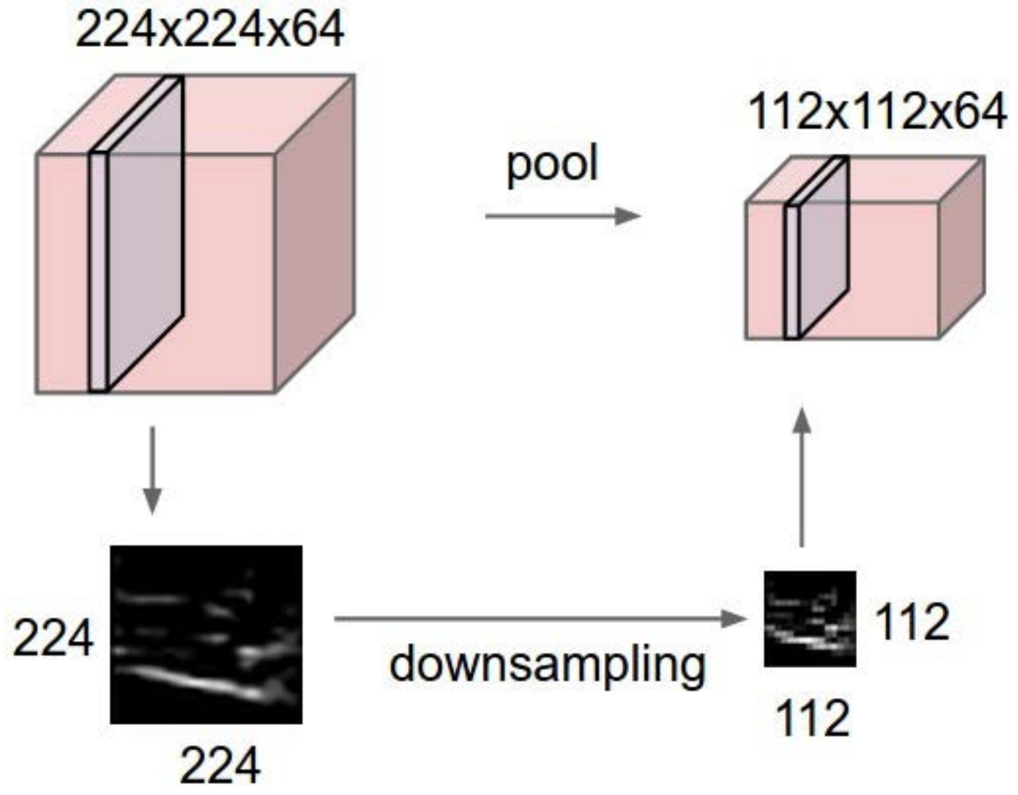
No learnable parameters!

Components of a Convolutional Network

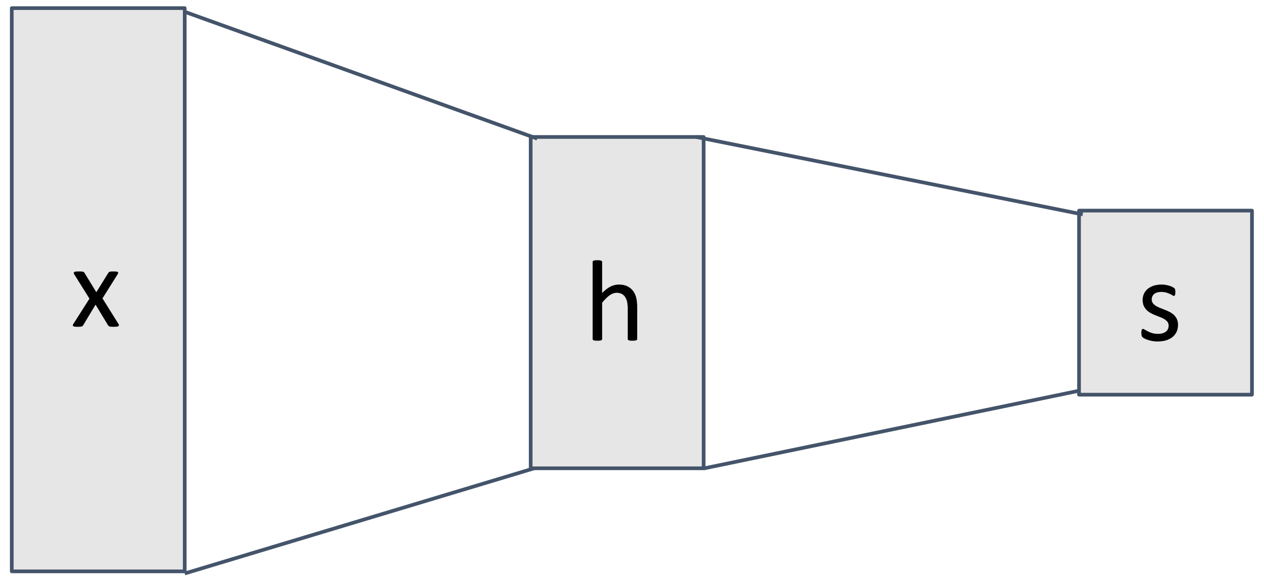
Convolution Layers



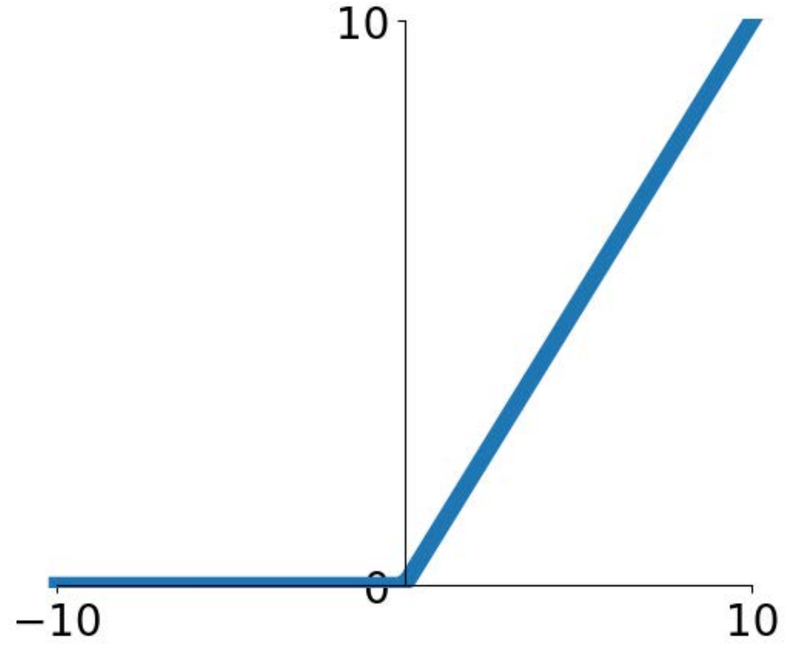
Pooling Layers



Fully-Connected Layers



Activation Function



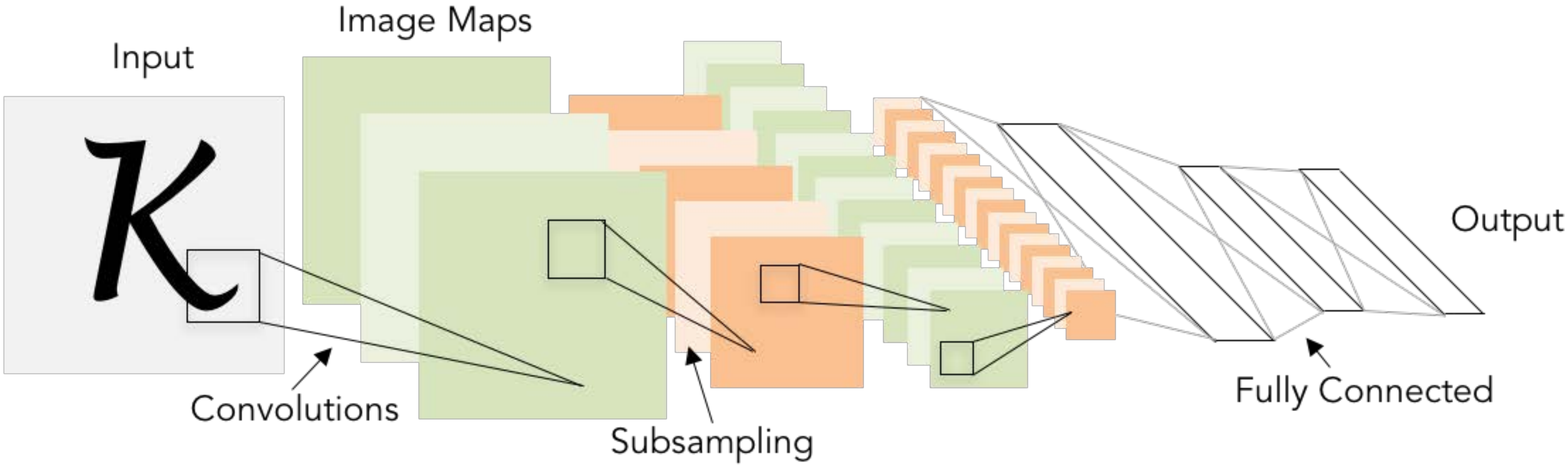
Normalization

$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$

Convolutional Networks

Classic architecture: [Conv, ReLU, Pool] x N, flatten, [FC, ReLU] x N, FC

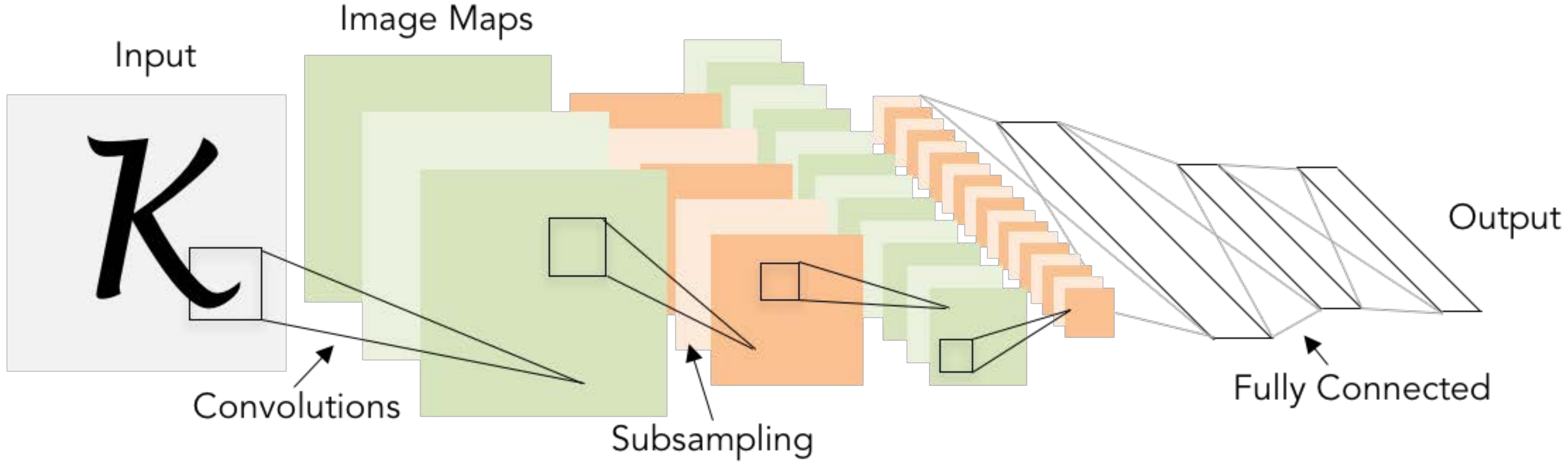
Example: LeNet-5



Lecun et al, "Gradient-based learning applied to document recognition", 1998

Example: LeNet-5

Layer	Output Size	Weight Size
Input	1 x 28 x 28	
Conv (C_{out}=20, K=5, P=2, S=1)	20 x 28 x 28	20 x 1 x 5 x 5
ReLU	20 x 28 x 28	
MaxPool(K=2, S=2)	20 x 14 x 14	
Conv (C_{out}=50, K=5, P=2, S=1)	50 x 14 x 14	50 x 20 x 5 x 5
ReLU	50 x 14 x 14	
MaxPool(K=2, S=2)	50 x 7 x 7	
Flatten	2450	
Linear (2450 -> 500)	500	2450 x 500
ReLU	500	
Linear (500 -> 10)	10	500 x 10



As we go through the network:

Spatial size **decreases**
(using pooling or strided conv)

Number of channels **increases**
(total "volume" is preserved!)

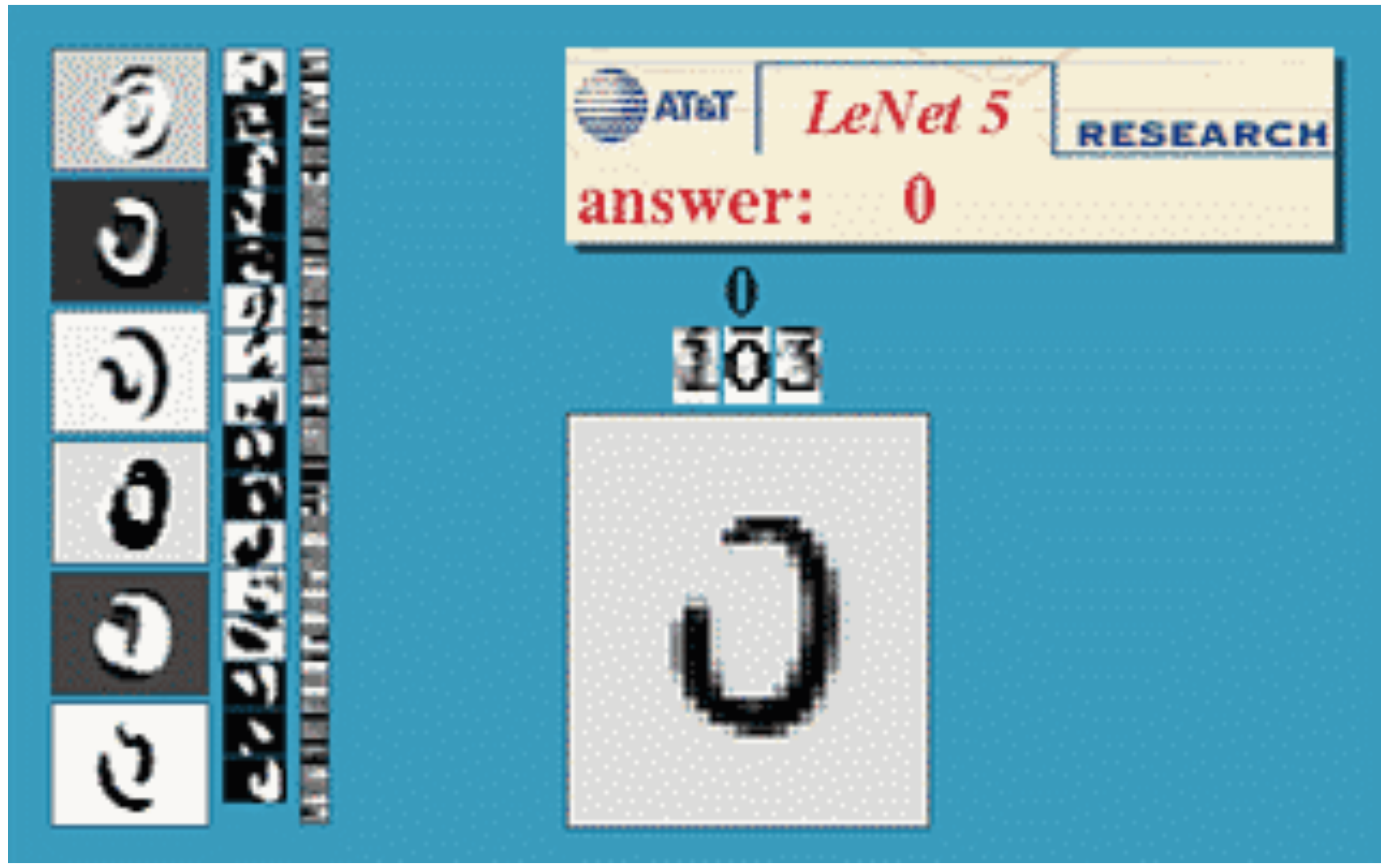
Lecun et al, "Gradient-based learning applied to document recognition", 1998

Optical Character Recognition (**OCR**)

Technology to convert **scanned documents to text**

(comes with any scanner now days)

Yann LeCun



Digit recognition, AT&T labs
http://www.research.att.com/~yann/

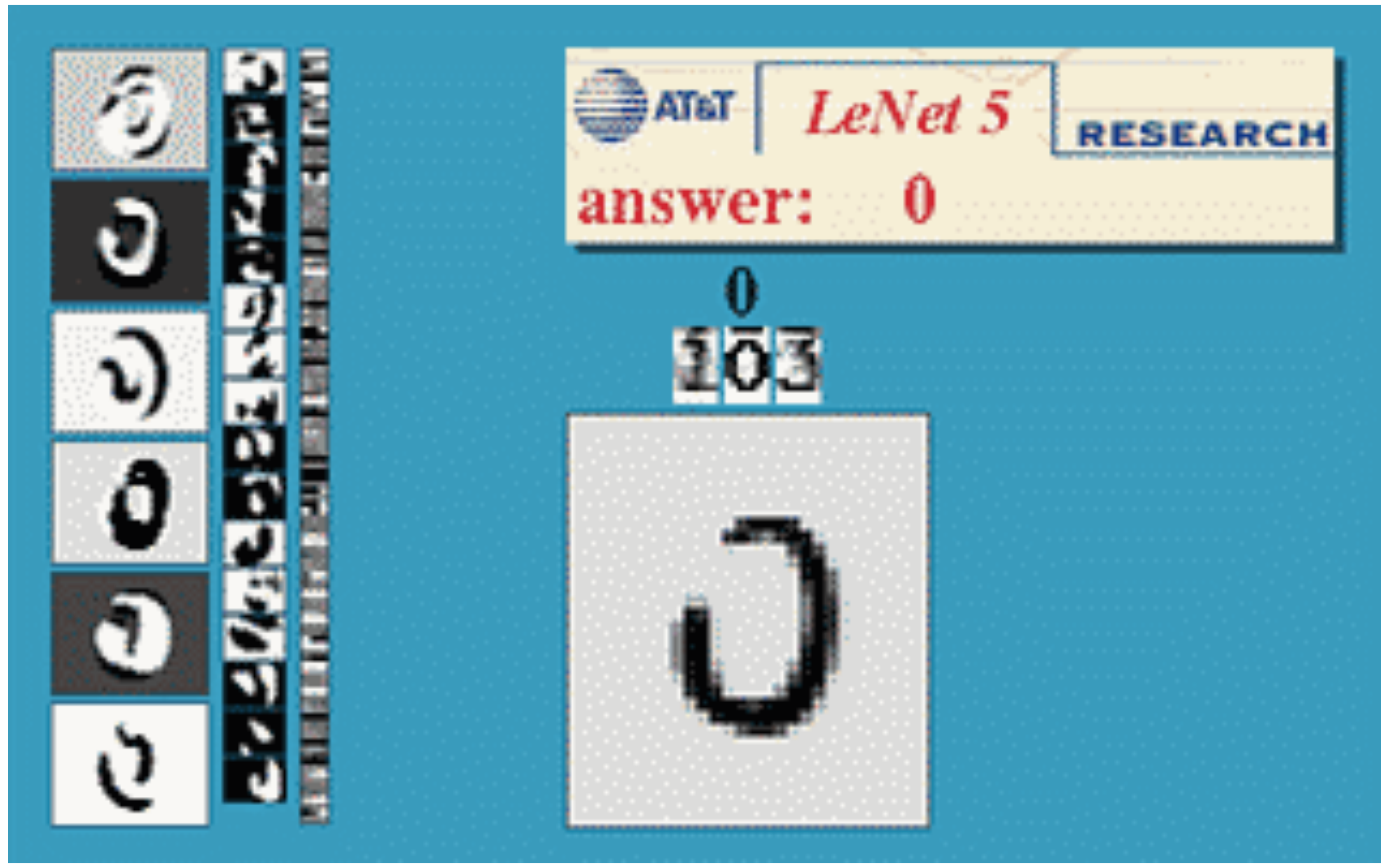
License plate readers
http://en.wikipedia.org/wiki/Automatic_number_plate_recognition

Optical Character Recognition (**OCR**)

Technology to convert **scanned documents to text**

(comes with any scanner now days)

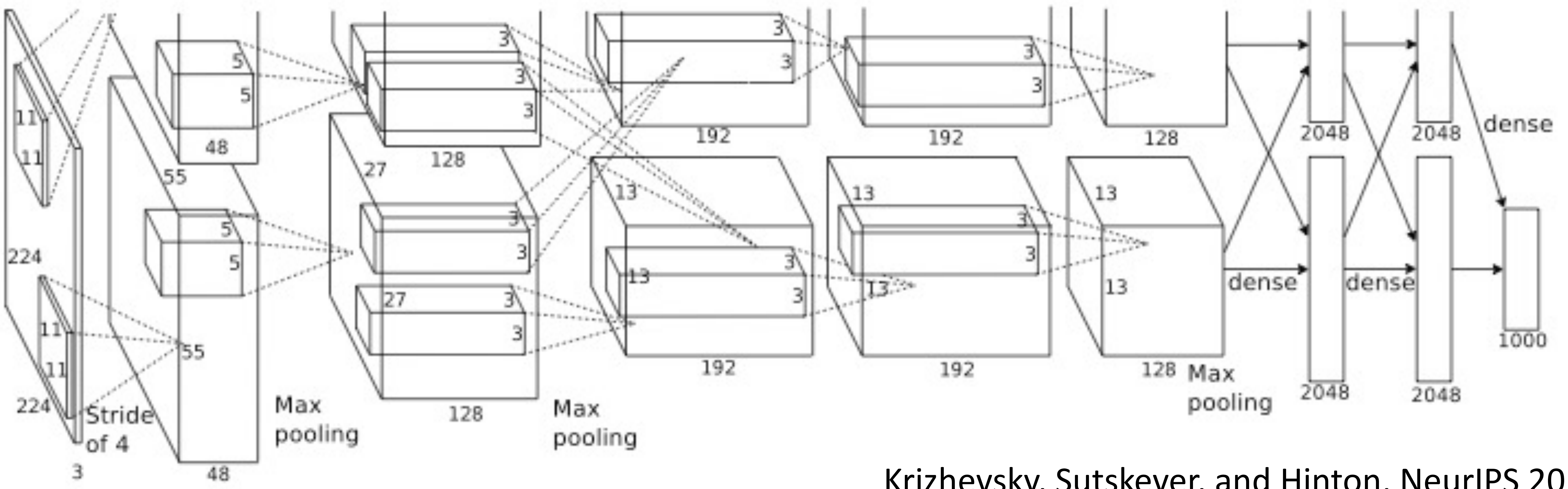
Yann LeCun



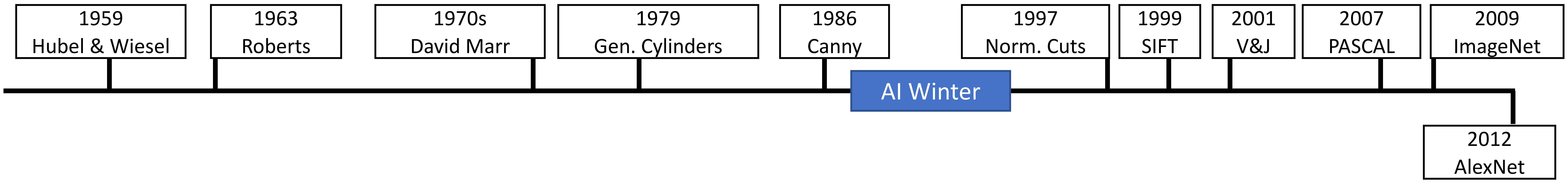
Digit recognition, AT&T labs
http://www.research.att.com/~yann/

License plate readers
http://en.wikipedia.org/wiki/Automatic_number_plate_recognition

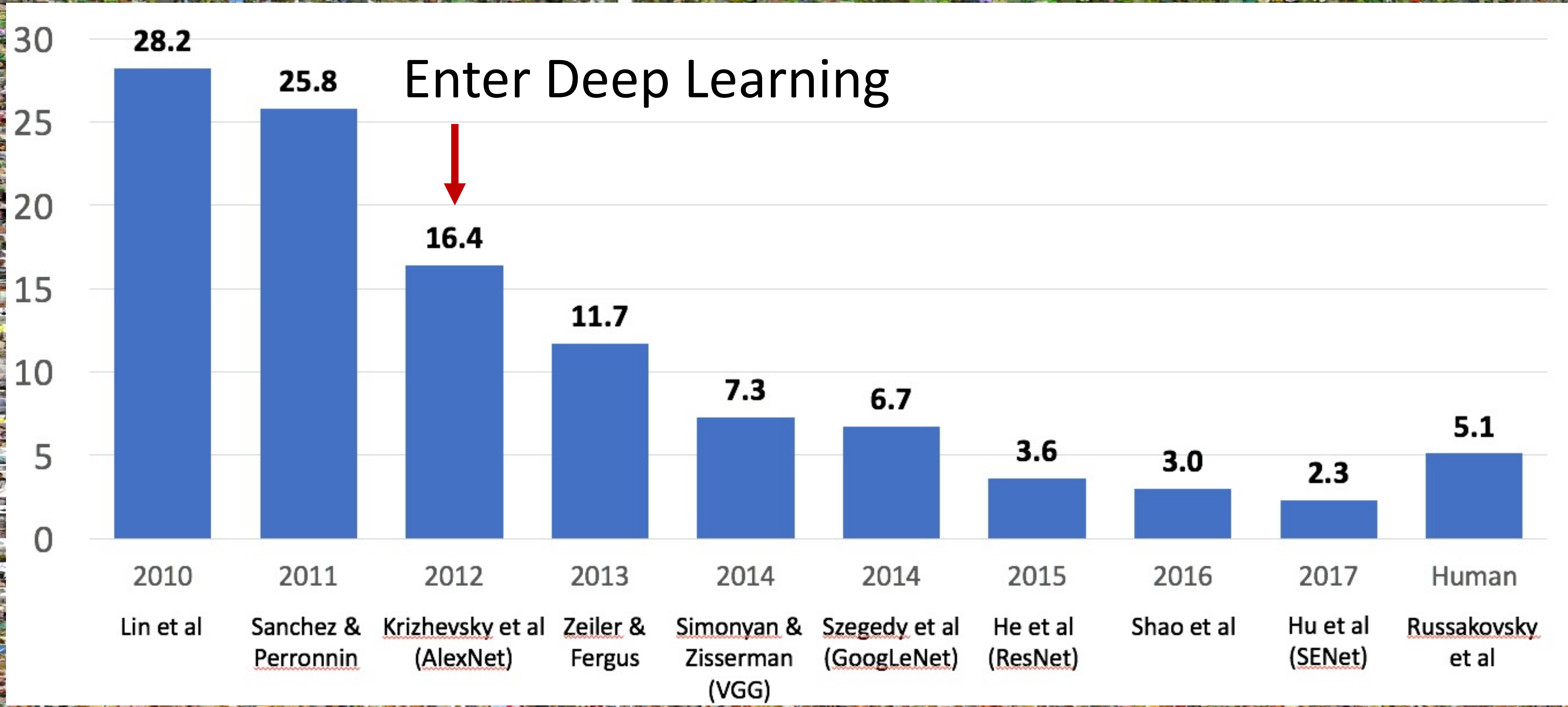
AlexNet: Deep Learning Goes Mainstream



Krizhevsky, Sutskever, and Hinton, NeurIPS 2012



IMAGENET Large Scale Visual Recognition Challenge



1959 Hubel & Wiesel

1963 Roberts

1970s David Marr

1979 Gen. Cylinders

1986 Canny

AI Winter

1997 Norm. Cuts

1999 SIFT

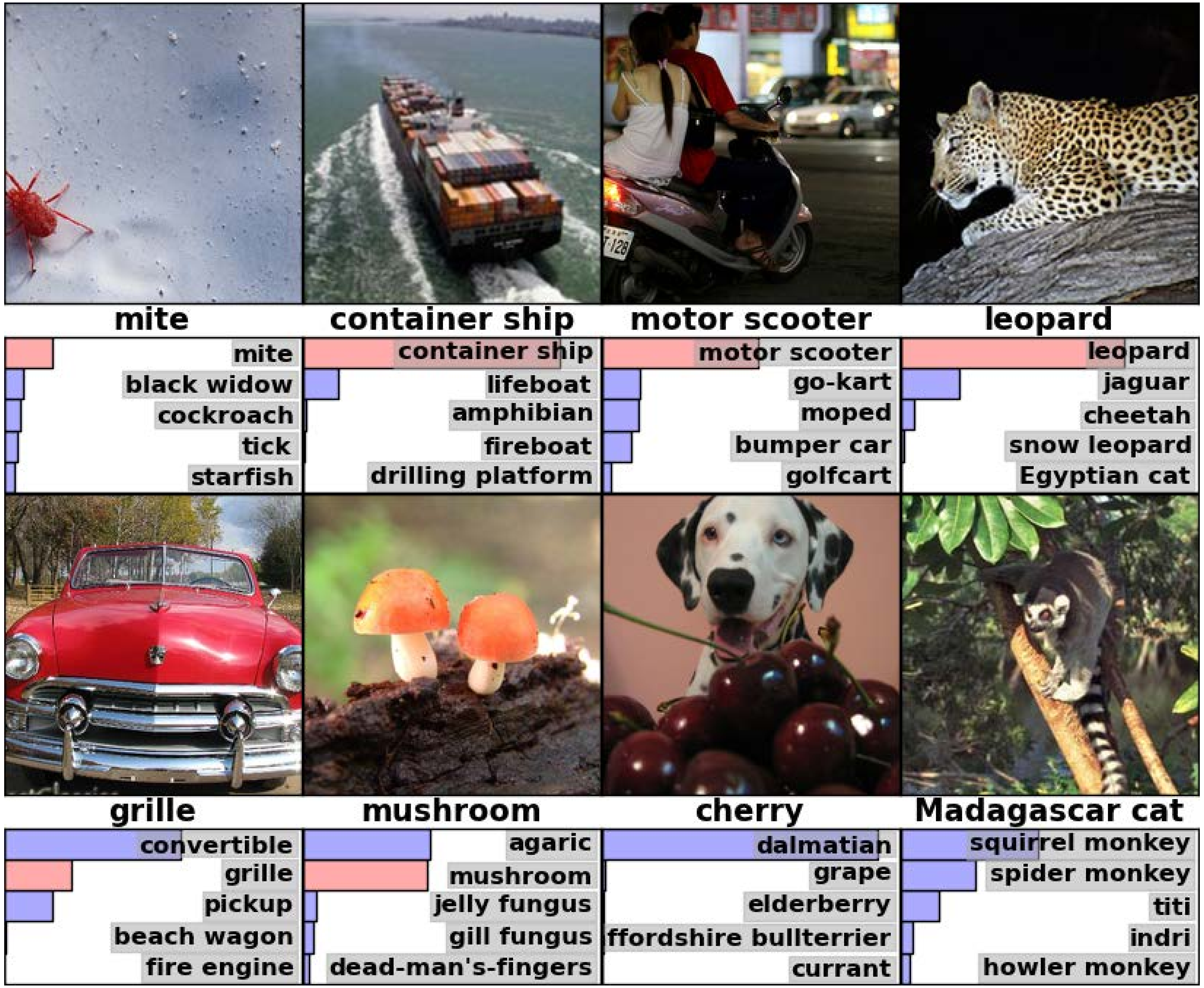
2001 V&J

2007 PASCAL

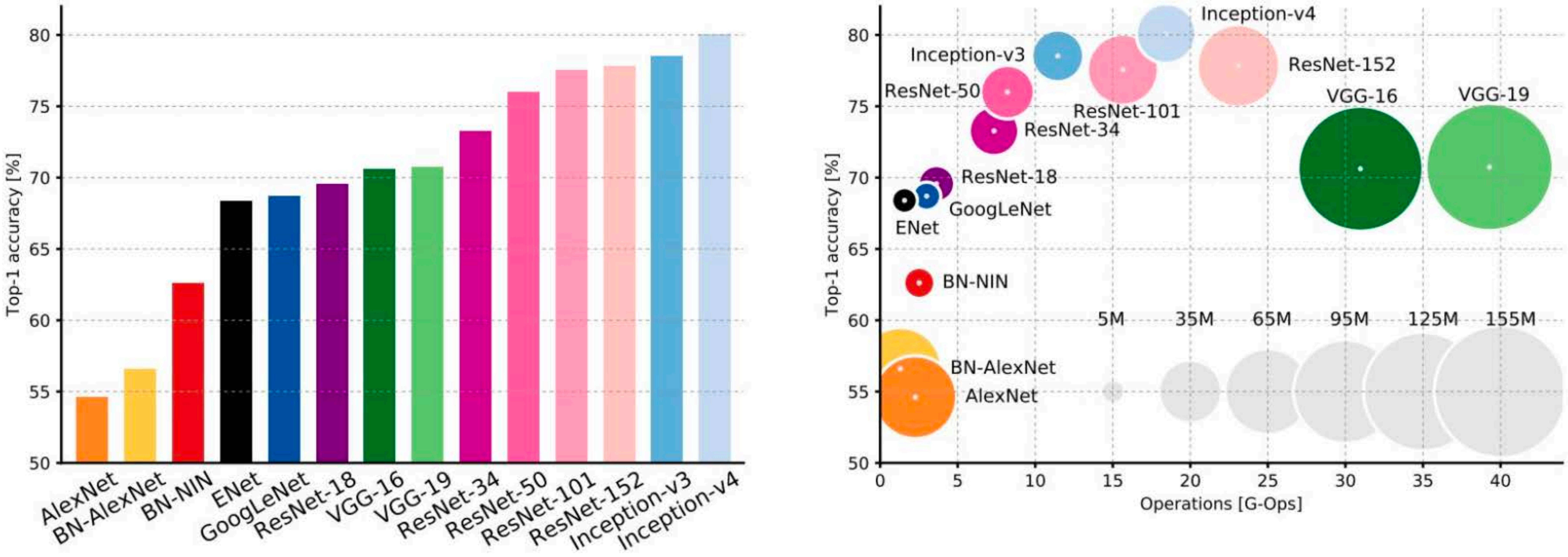
2009 ImageNet

2012 AlexNet

AlexNet on ImageNet



Comparing **Complexity**



An Analysis of Deep Neural Network Models for Practical Applications, 2017.

* adopted from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Summary

The parameters of a neural network are learned using **backpropagation**, which computes gradients via recursive application of the chain rule

A **convolutional neural network** assumes inputs are images, and constrains the network architecture to reduce the number of parameters

A **convolutional layer** applies a set of learnable filters

A **pooling layer** performs spatial downsampling

A **fully-connected** layer is the same as in a regular neural network

Convolutional neural networks can be seen as learning a hierarchy of filters