THE UNIVERSITY OF BRITISH COLUMBIA

Topics in AI (CPSC 532S):
Multimodal Learning with Vision, Language and Sound

Lecture 4: Convolutional Neural Networks (Part 1)

Course **Logistics**

- **Azure credits** have been distributed (to those who asked for them)
- Instructions for using Azure at **TA Office Hour 1-2pm today** and on Piazza
- **Assignment 1** was due yesterday
- **Assignment 2** will be out today (on CNNs) and is due Wednsday next week

(note, it will take computation time)

- Start thinking about **project** and forming groups

(groups will be due in 1-2 weeks, proposal is in ~month)

Research **Projects**

Business

"A good **entrepreneur** can take a mediocre **idea** and make it great, a bad entrepreneur will take the best idea and run it into a ground."

Robert Herjavec
(Canadian businessman and investor)
Shark Tank

Research

"A good **idea** can make a mediocre **project** great, a bad idea will take the best project and run it into a ground." — Me

Research **Projects** Ideas Levels of Abstraction

I want to **solve vision** / language / etc.

Marvin Minsky

Research **Projects** Ideas Levels of Abstraction

I want to **solve vision** / language / etc.

I want to do X (e.g., **image captioning**)

- This is excellent if **X** is something no one has done or thought about and is important (guaranteed success)
- Requires **forward thinking**, knowledge of the field
- A sure way to get tenure
- Difficult to do as the field matures

Rosland Picard, MIT — Affective Computing

Research **Projects** Ideas Levels of Abstraction

Geoffrey Hinton

I want to **solve vision** / language / etc.

I want to do X (e.g., **image captioning**)

I think the **right way to solve** (or improve) X is Y

- More incremental; a lot of science is incremental (“standing on the shoulders of giants”)
- **Retrospective:** compare existing approaches see why they work what is missing (guaranteed success)
- **Perspective:** come up with an idea or the insight that you truly believe and test it
- Requires through knowledge of the sub-field (lots of reading)
- Requires strong intuition and high level (intuitive) thinking
- Requires understanding of the mathematical tools and formulations to know what maybe possible
- Helps to bringing knowledge from other fields (field cross pollination)

Research **Projects** Ideas Levels of Abstraction

I want to **solve vision** / language / etc.

I want to do X (e.g., **image captioning**)

I think the **right way to solve** (or improve) X is Y

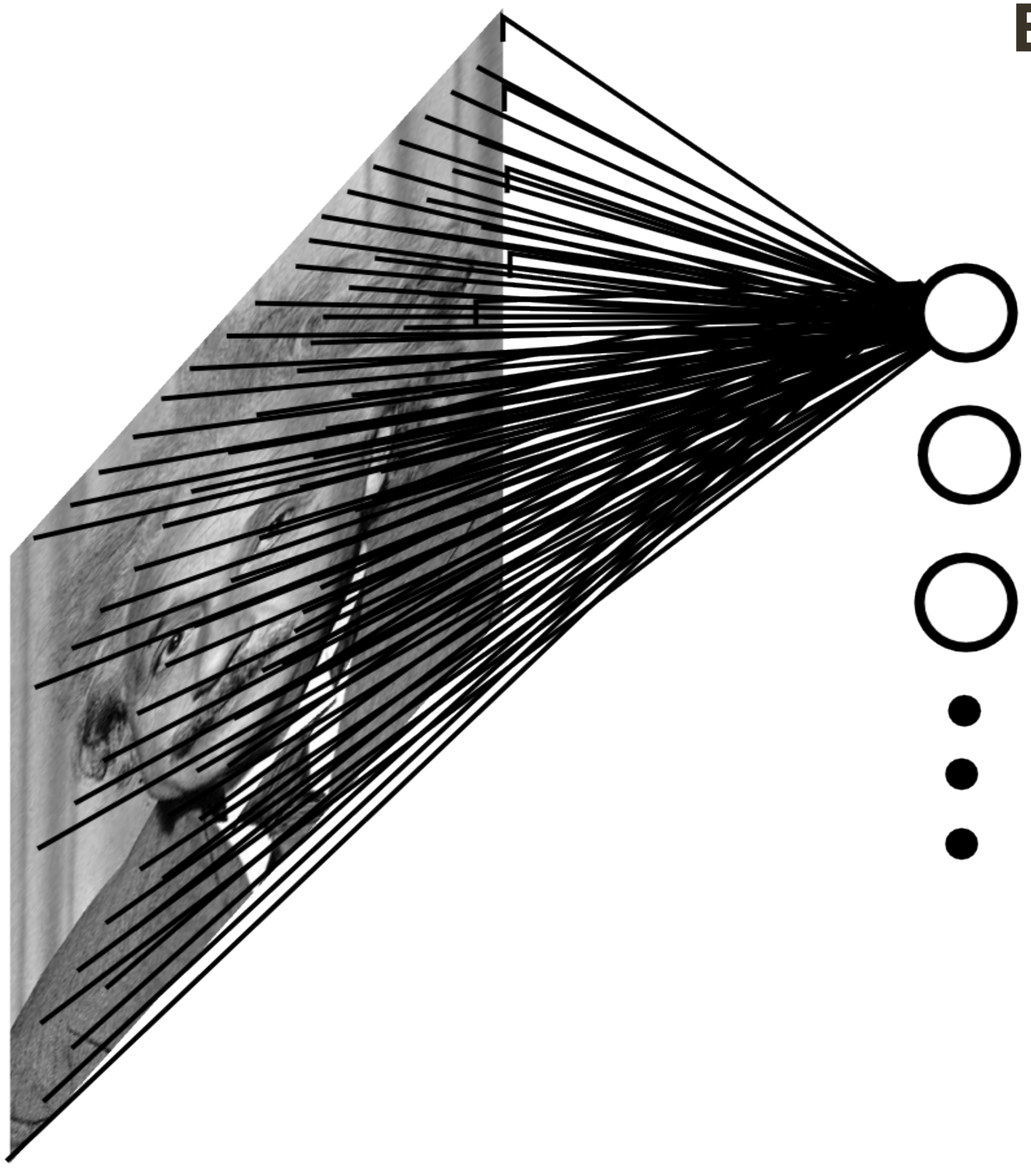
Mathematical **formulation**

Implementation / engineering

Experimental testing

On to **todays** lecture …

Fully Connected Layer



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

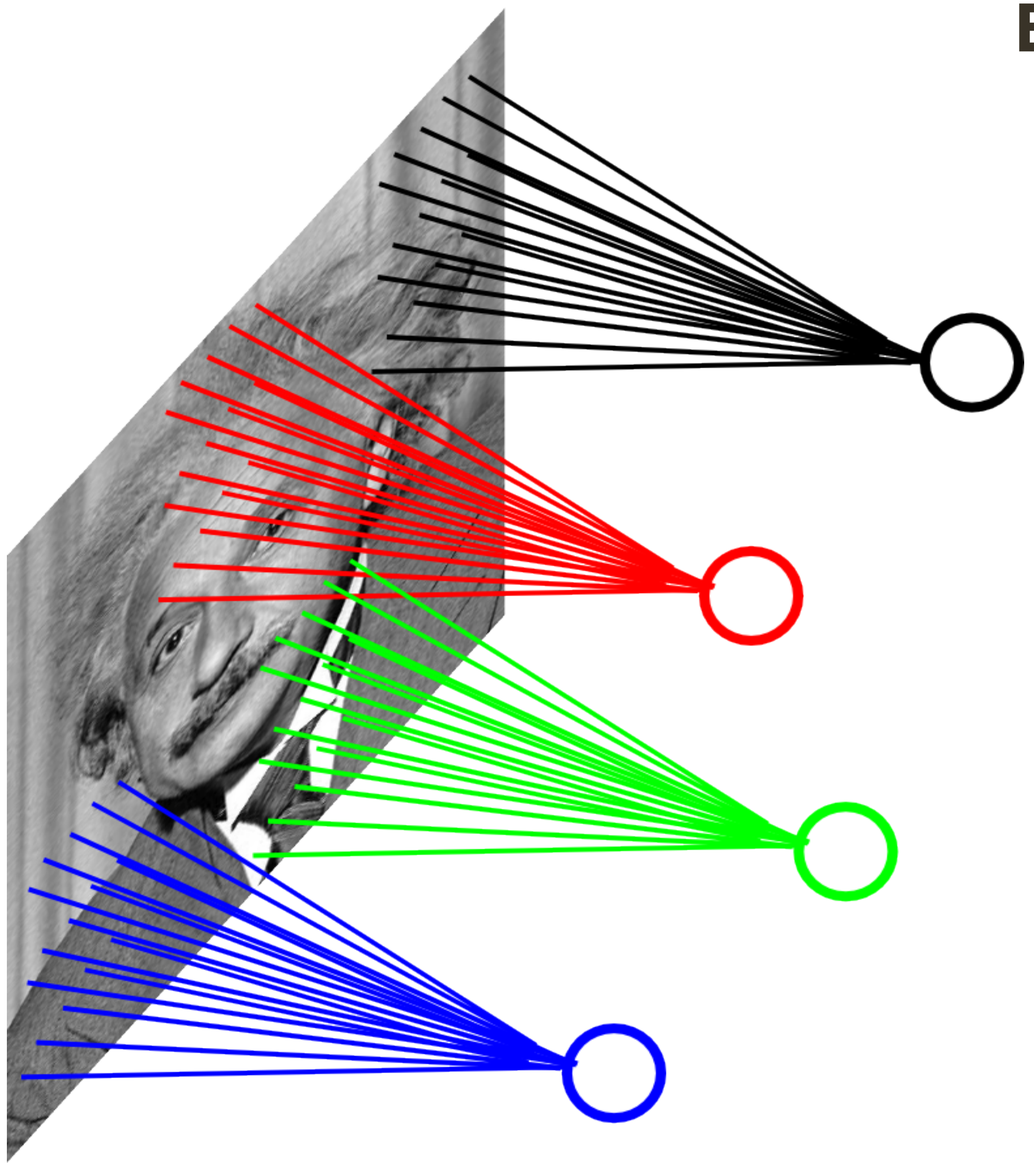
= ~ **2 Billion** parameters (for one layer!)

Spatial correlations are generally local

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

* slide from Marc'Aurelio Renzato

Locally Connected Layer



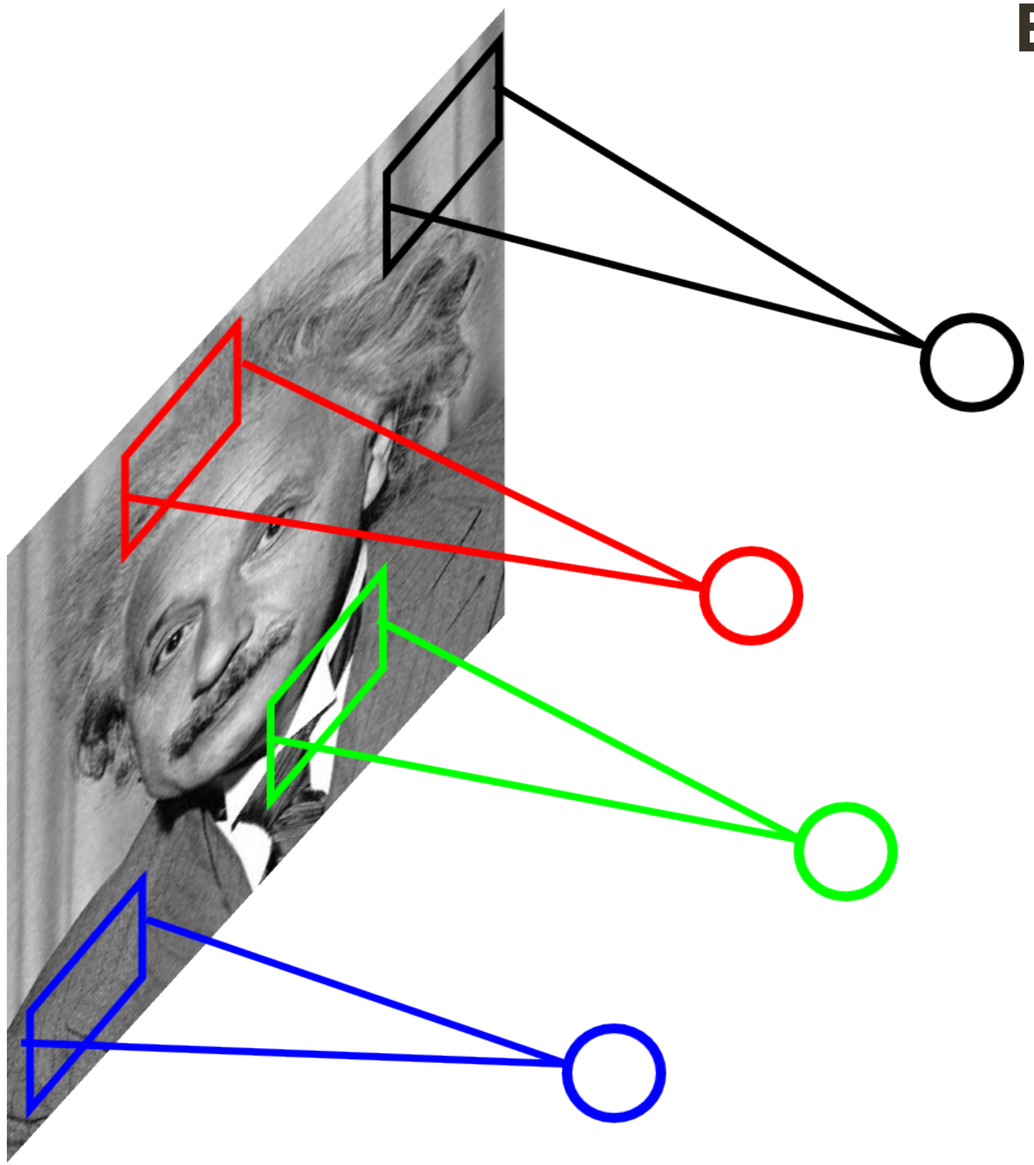
Example: 200 x 200 image (small)

Filter size: 10 x 10

= ~ **4 Million** parameters

* slide from Marc'Aurelio Renzato

Locally Connected Layer



Example: 200 x 200 image (small)

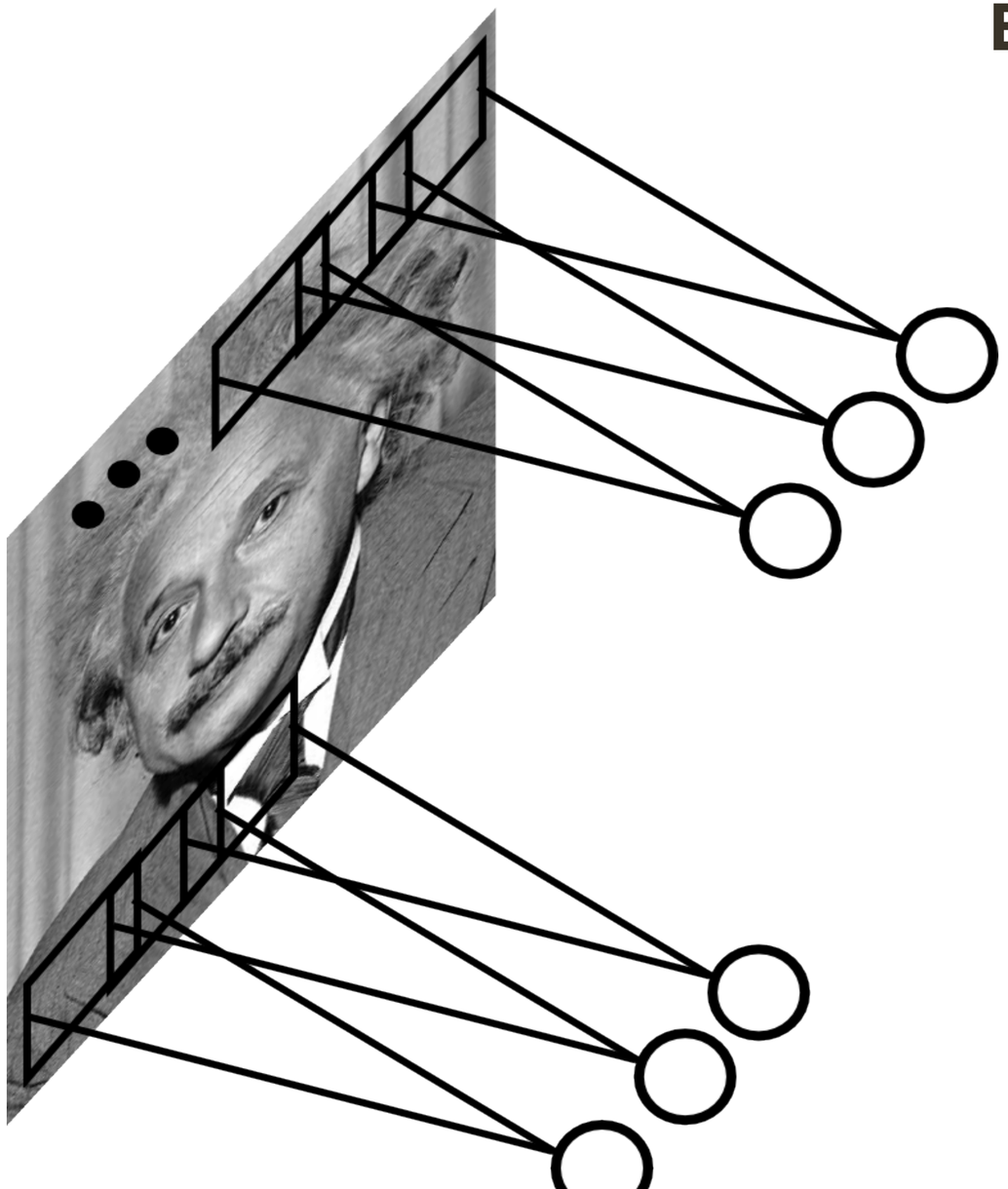
Filter size: 10 x 10

= ~ **4 Million** parameters

Stationarity — statistics is similar at different locations

* slide from Marc'Aurelio Renzato

Convolutional Layer



Example: 200 x 200 image (small)

Filter size: 10 x 10

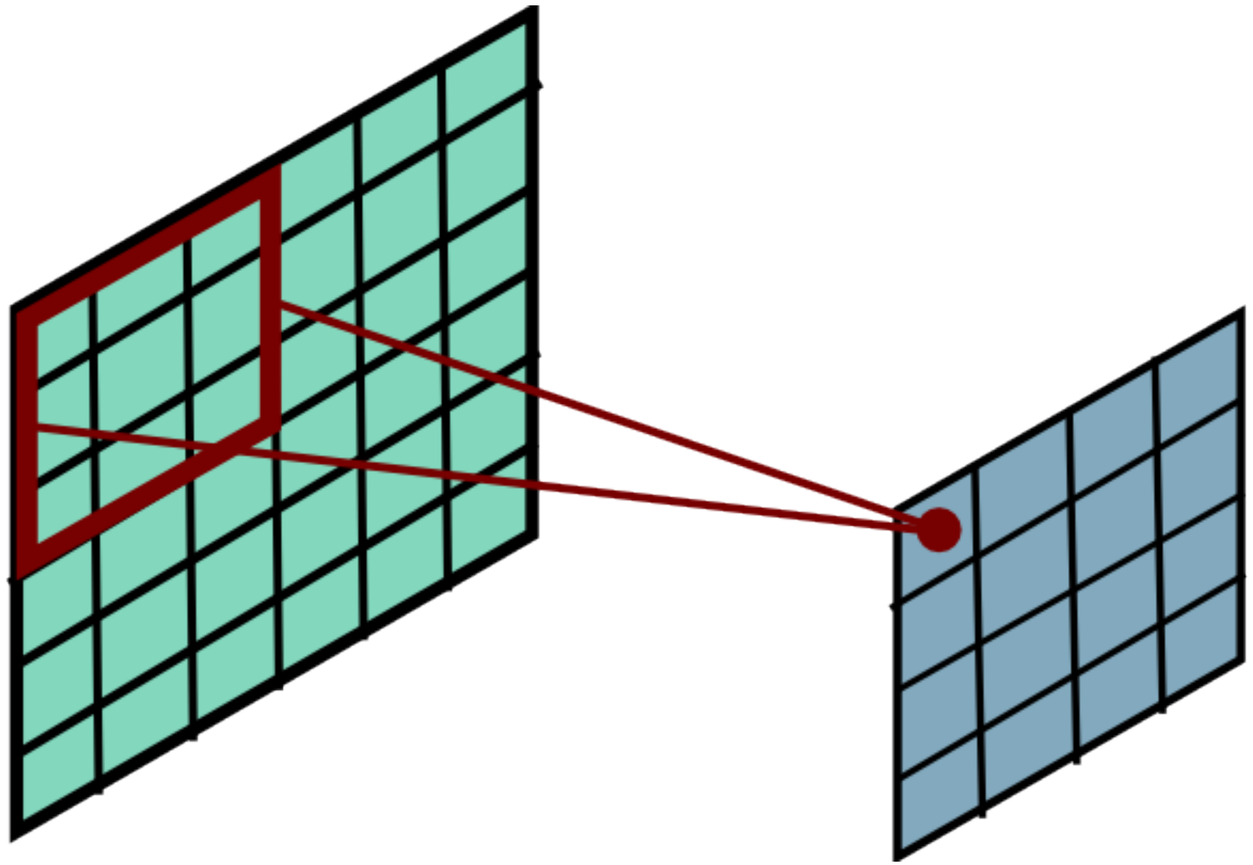
= ~ **4 Million** parameters

= 100 parameters

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

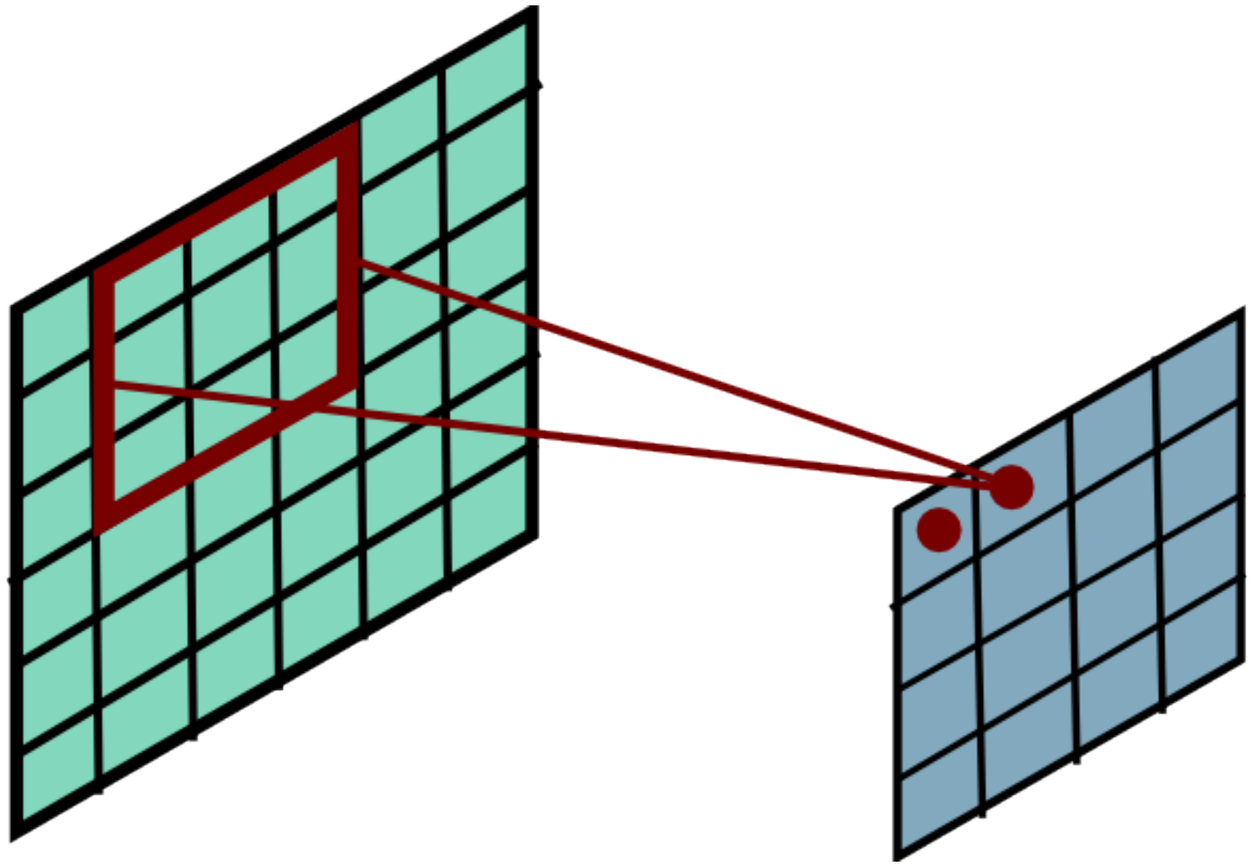
* slide adopted from Marc'Aurelio Renzato

Convolutional Layer



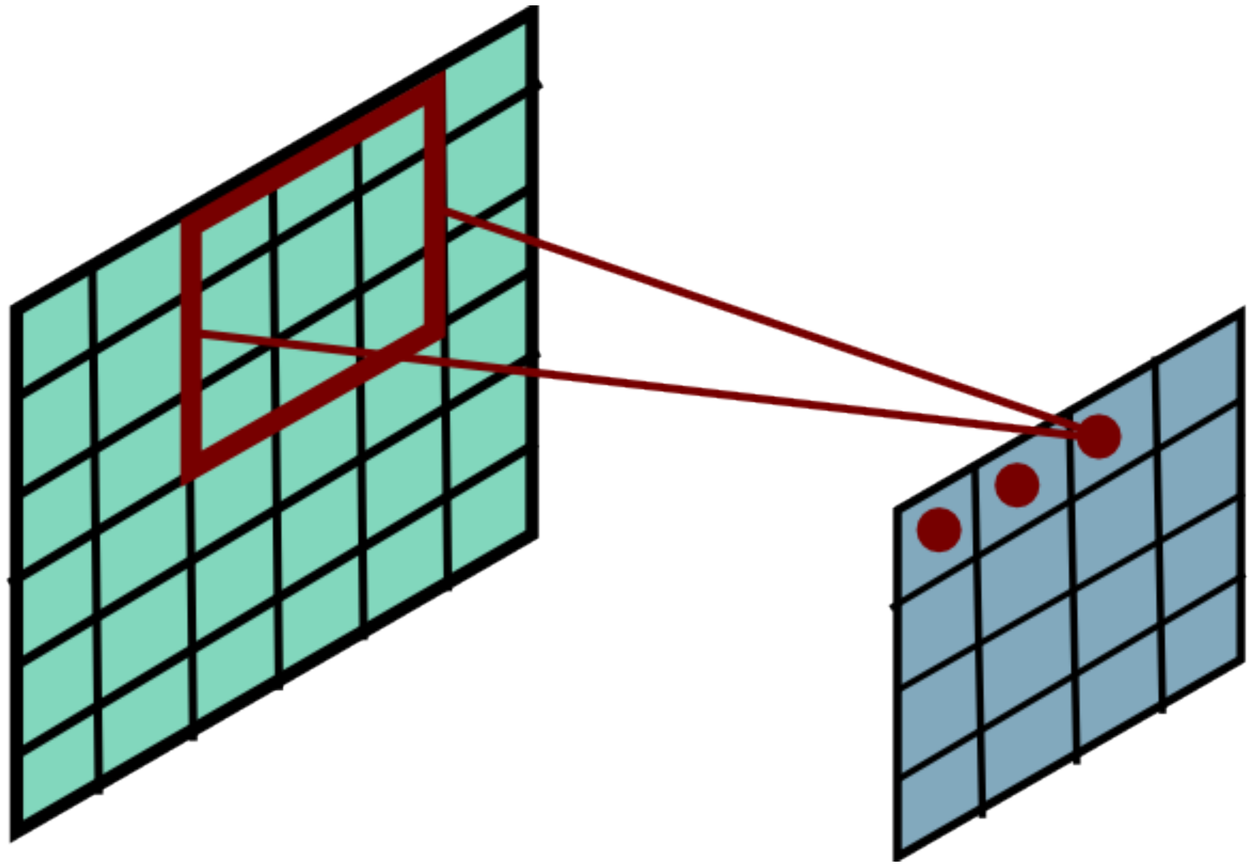
* slide from Marc'Aurelio Renzato

Convolutional Layer



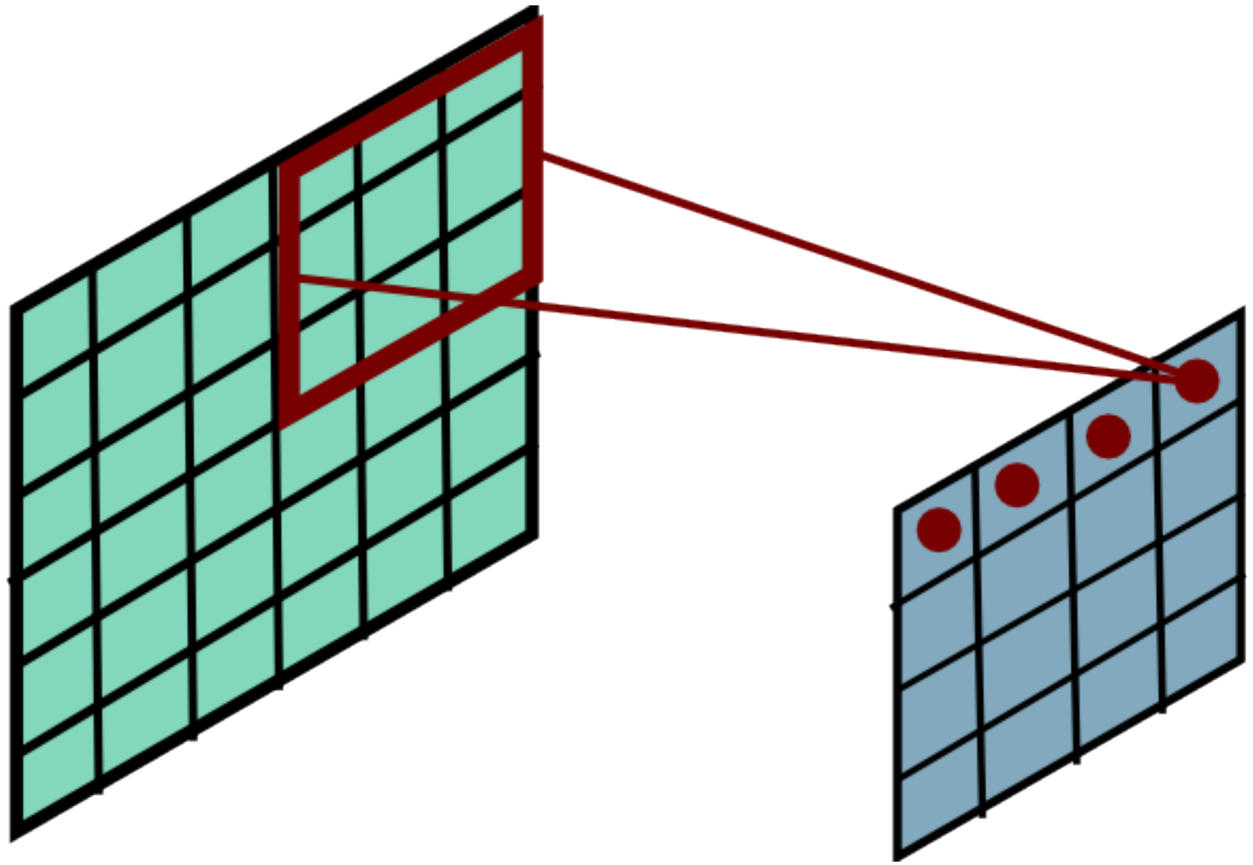
* slide from Marc'Aurelio Renzato

Convolutional Layer



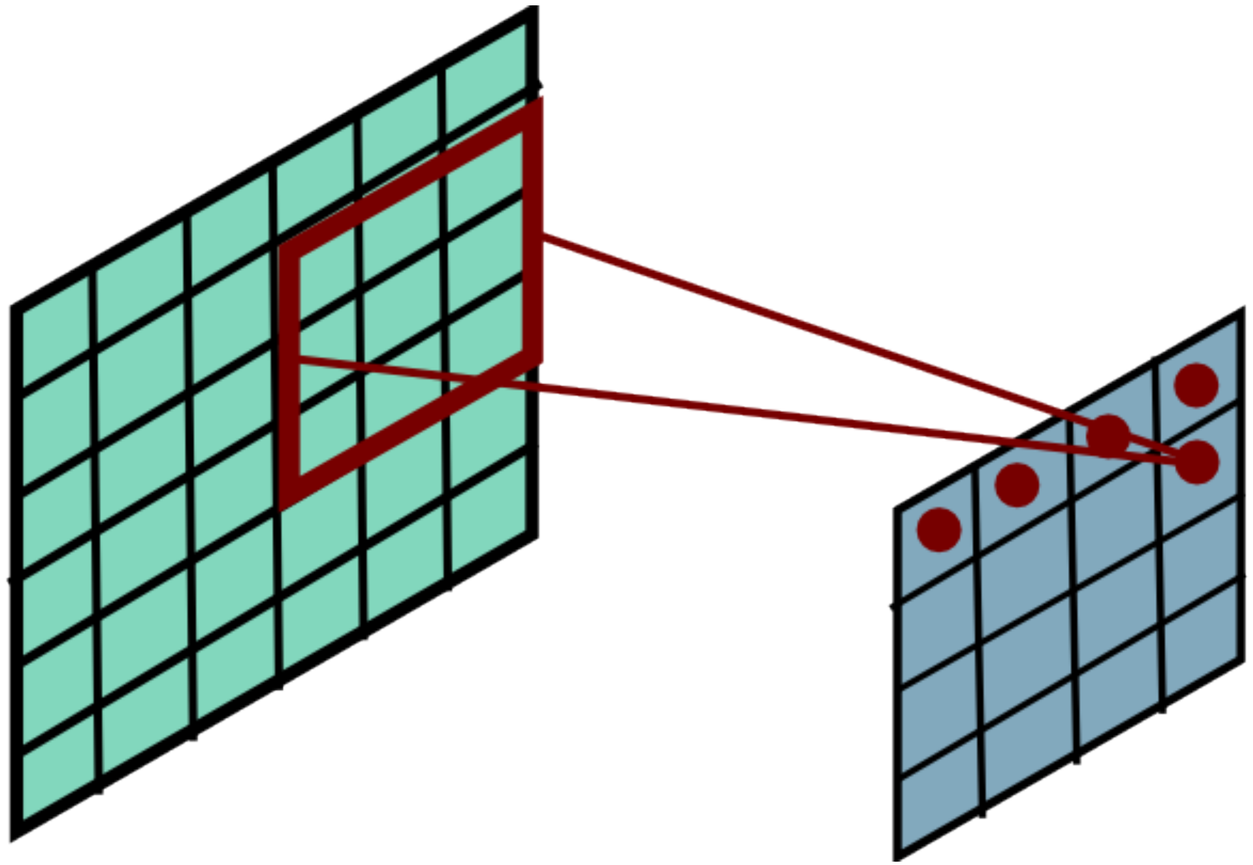
* slide from Marc'Aurelio Renzato

Convolutional Layer



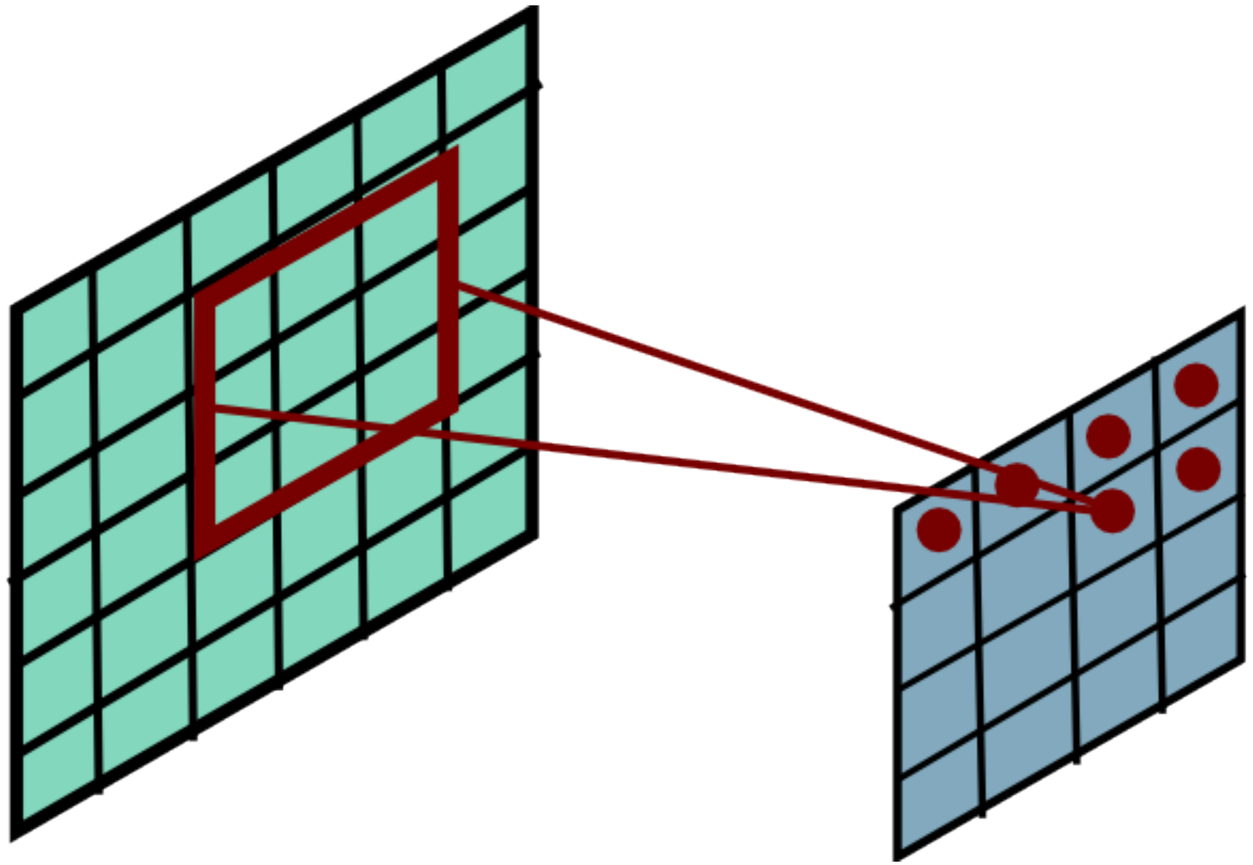
* slide from Marc'Aurelio Renzato

Convolutional Layer



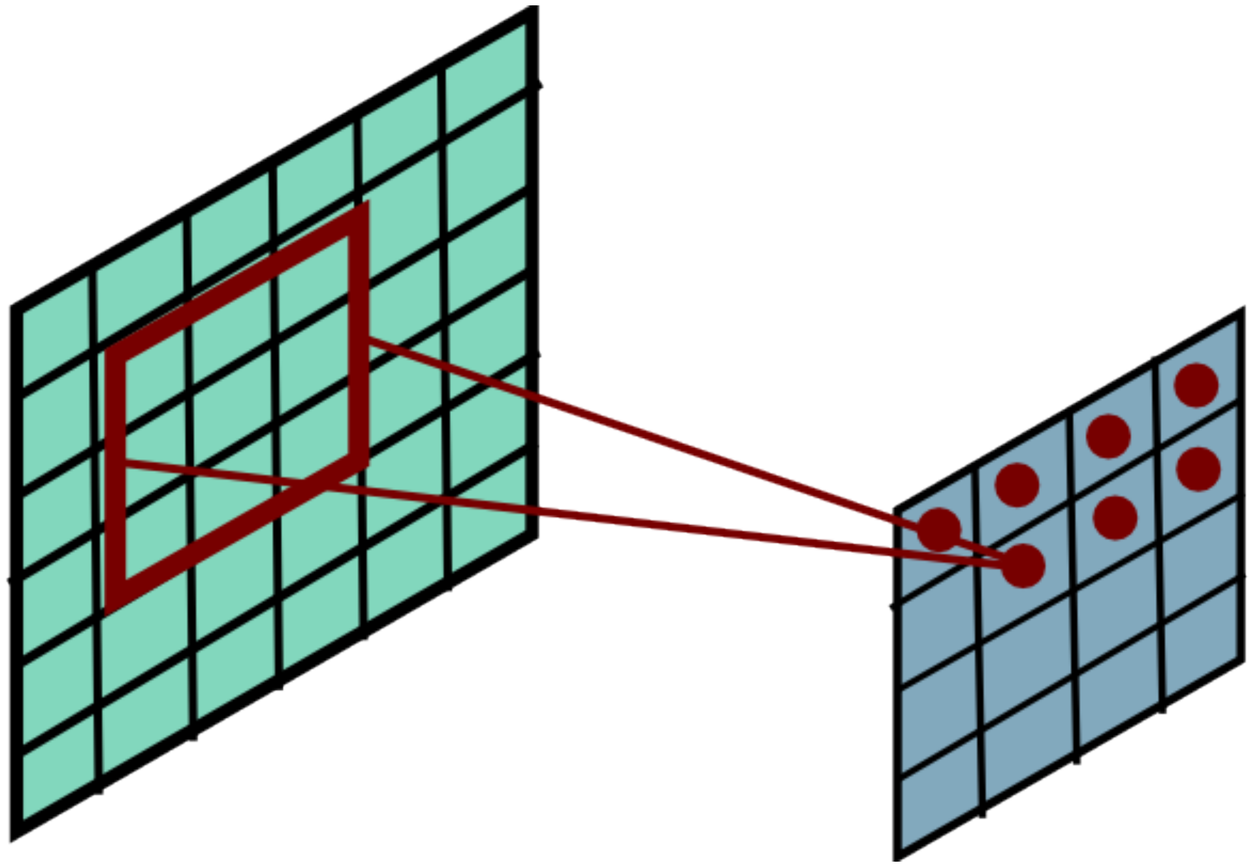
* slide from Marc'Aurelio Renzato

Convolutional Layer



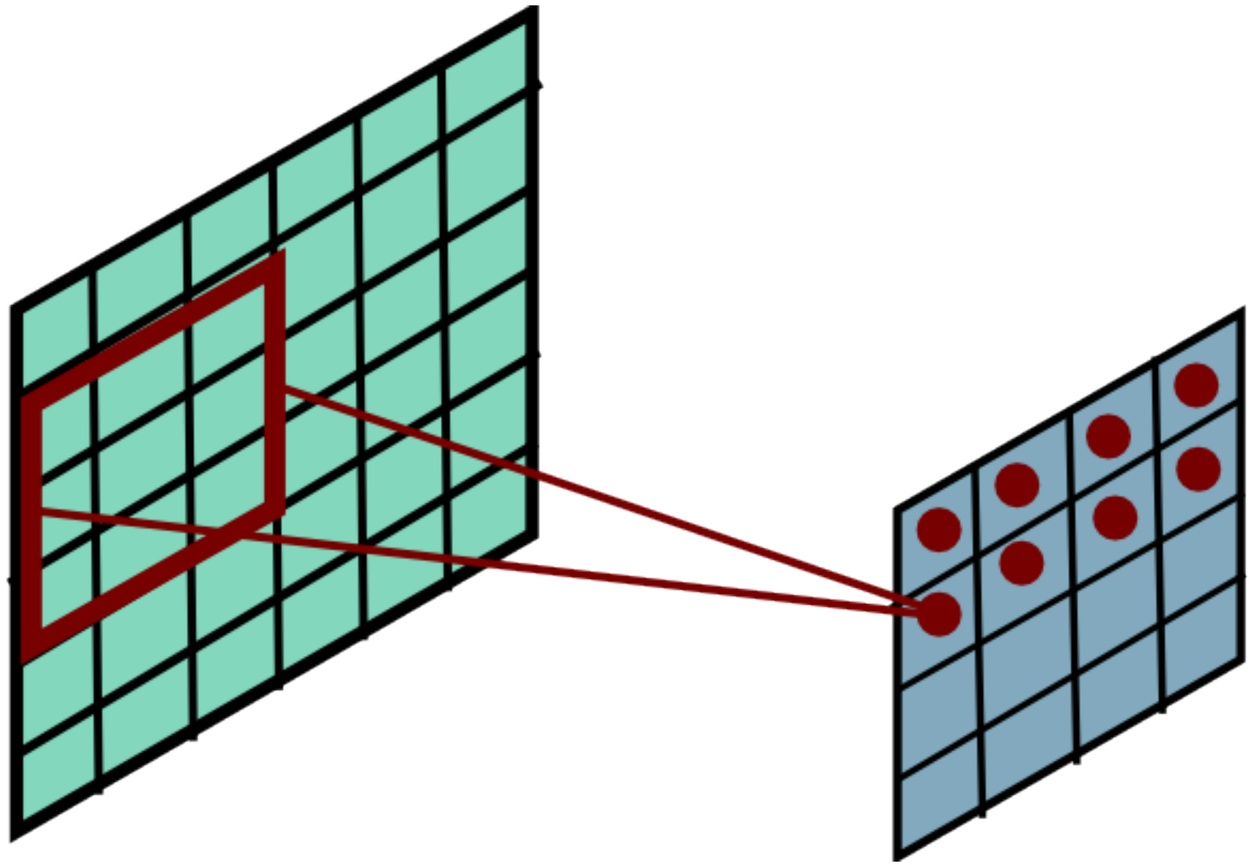
* slide from Marc'Aurelio Renzato

Convolutional Layer



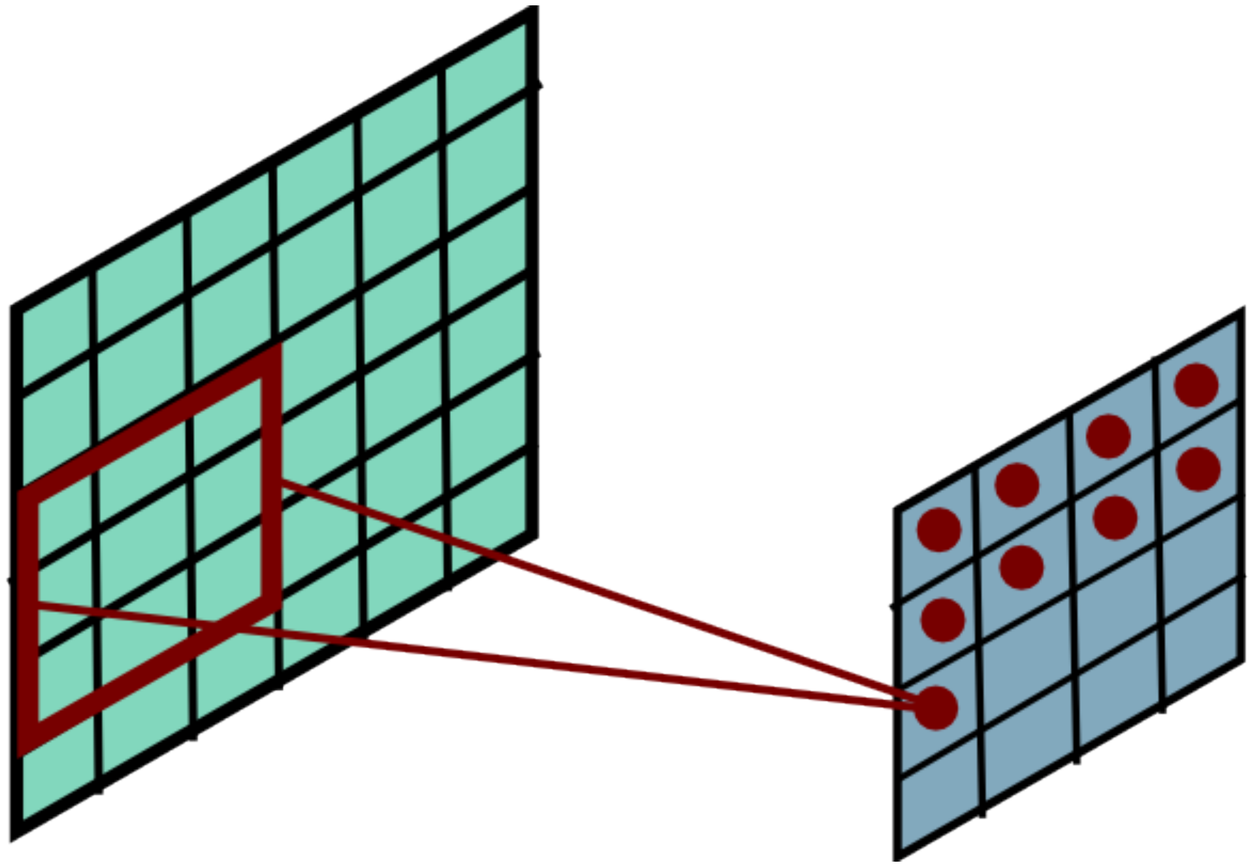
* slide from Marc'Aurelio Renzato

Convolutional Layer



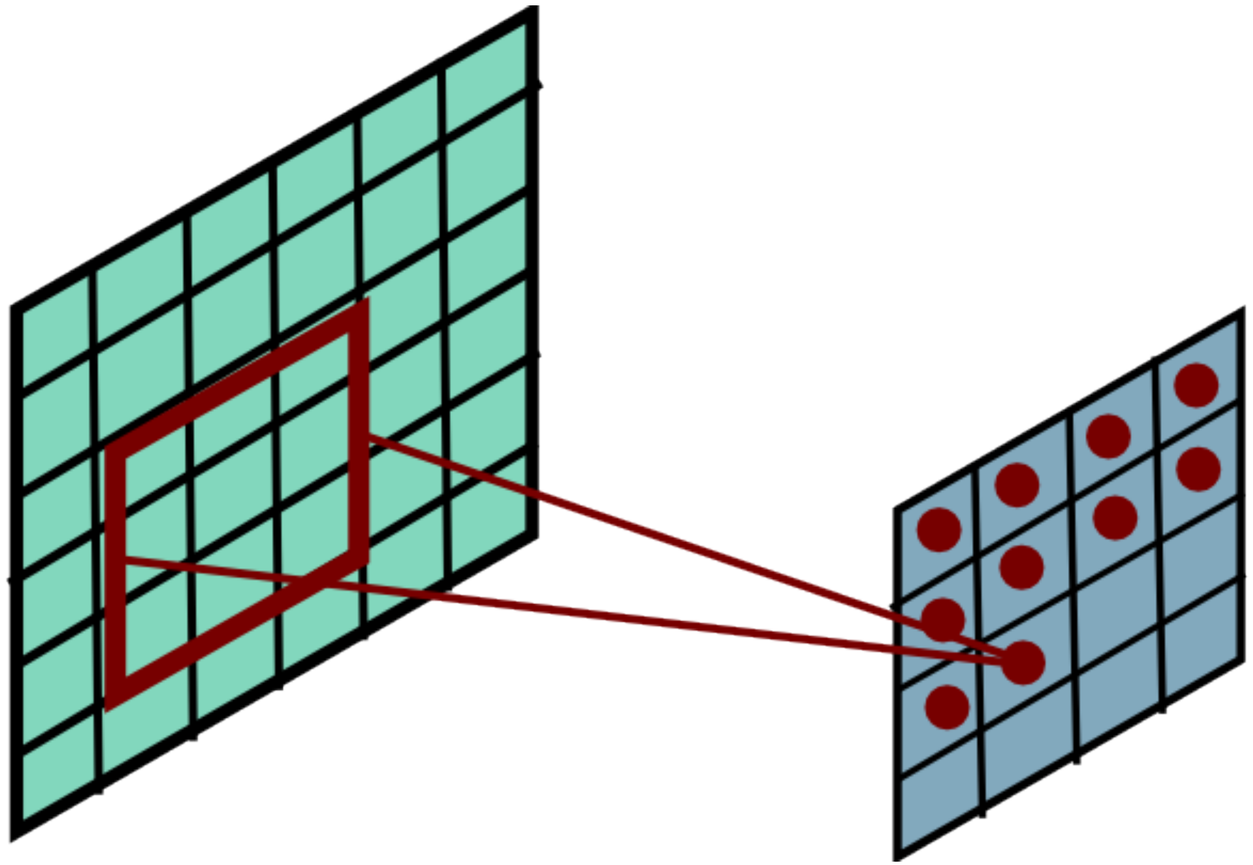
* slide from Marc'Aurelio Renzato

Convolutional Layer



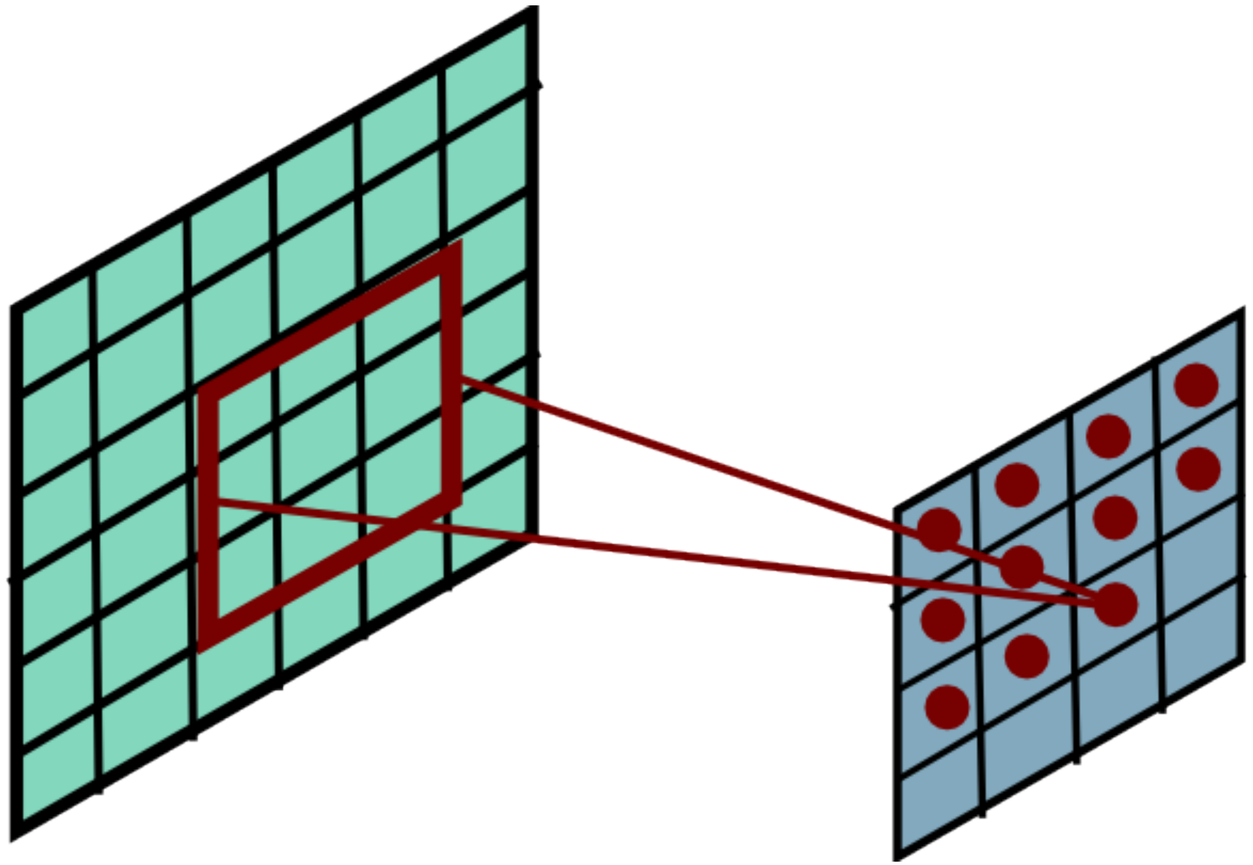
* slide from Marc'Aurelio Renzato

Convolutional Layer



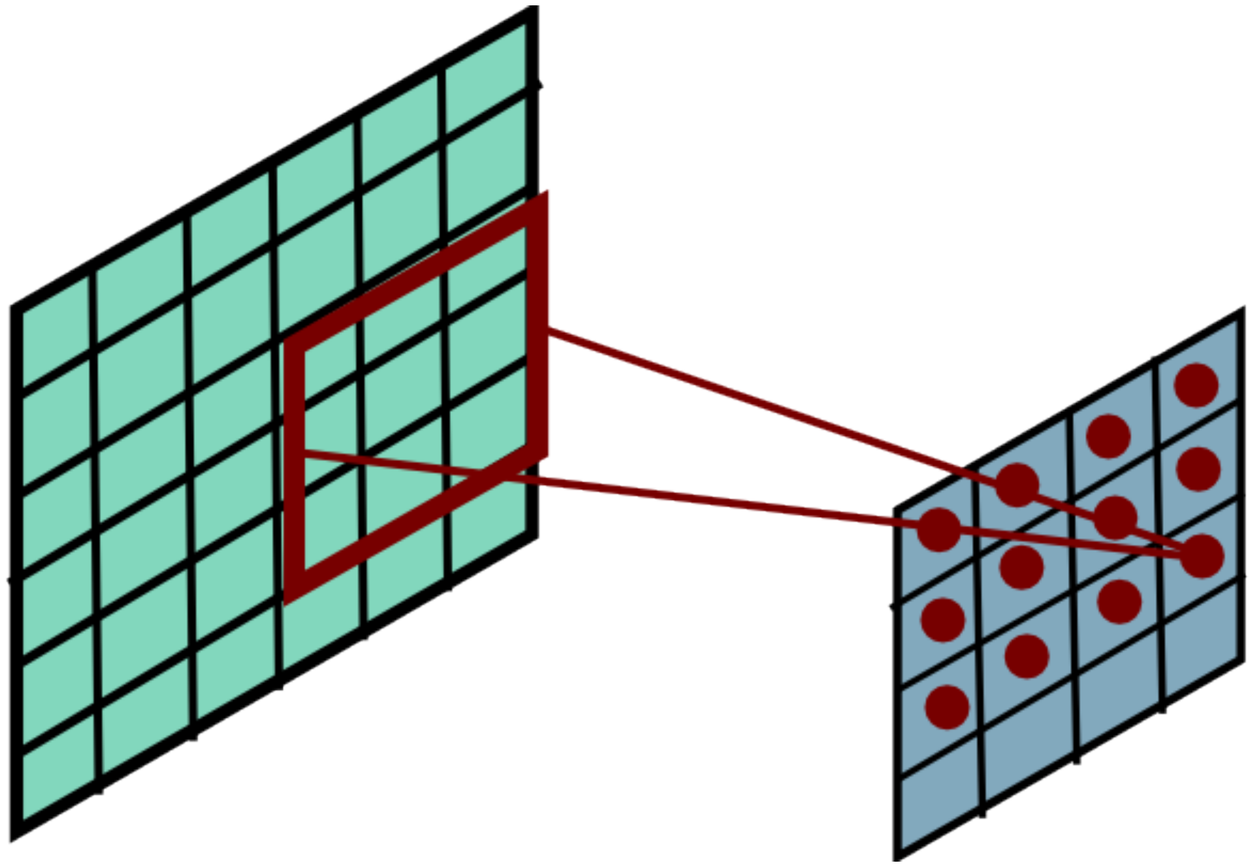
* slide from Marc'Aurelio Renzato

Convolutional Layer



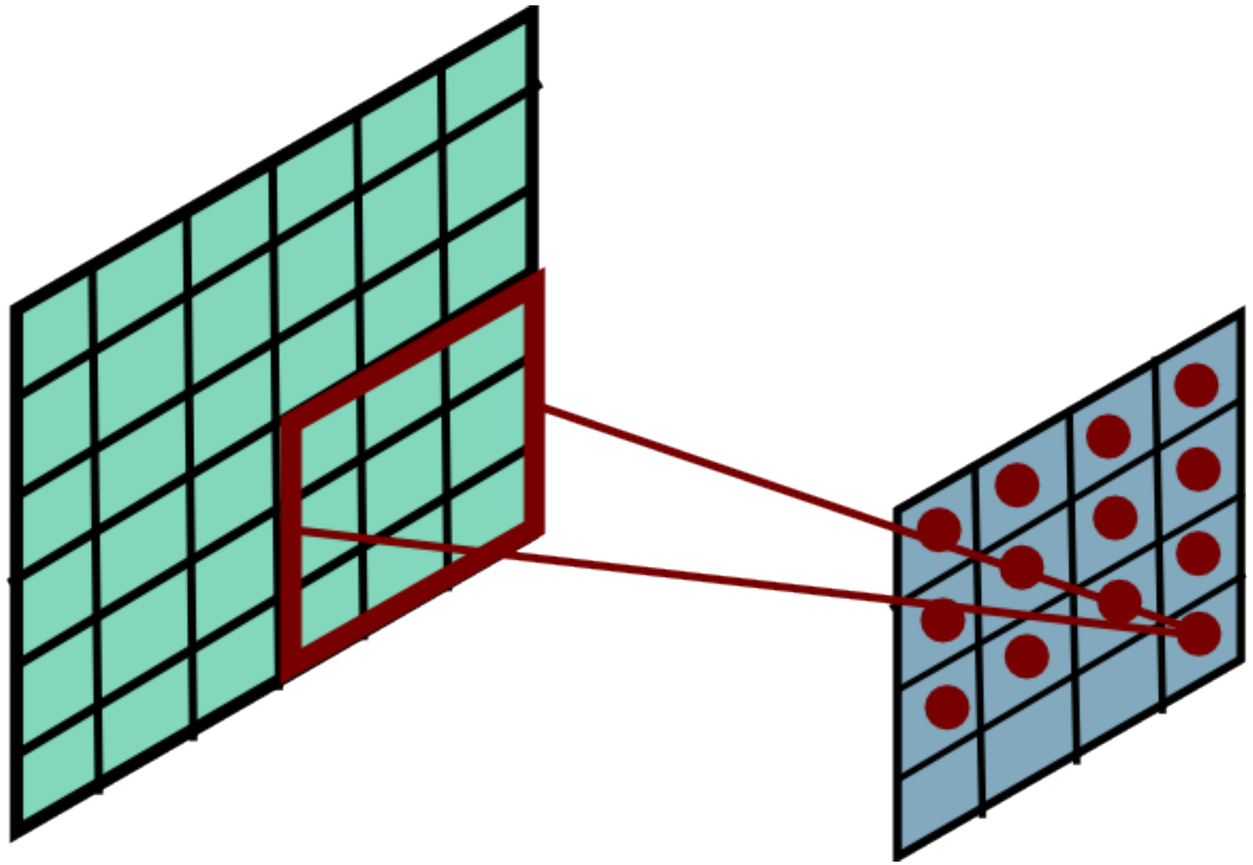
* slide from Marc'Aurelio Renzato

Convolutional Layer



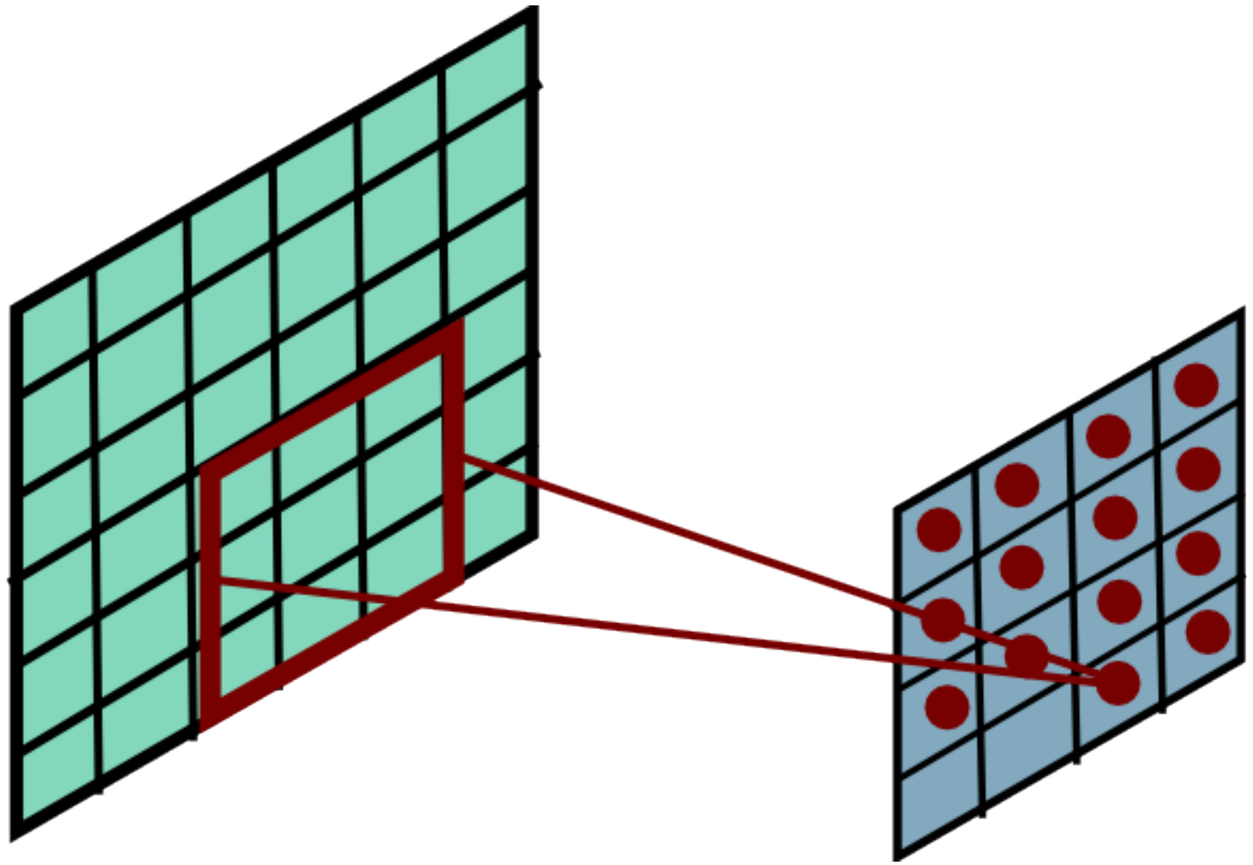
* slide from Marc'Aurelio Renzato

Convolutional Layer



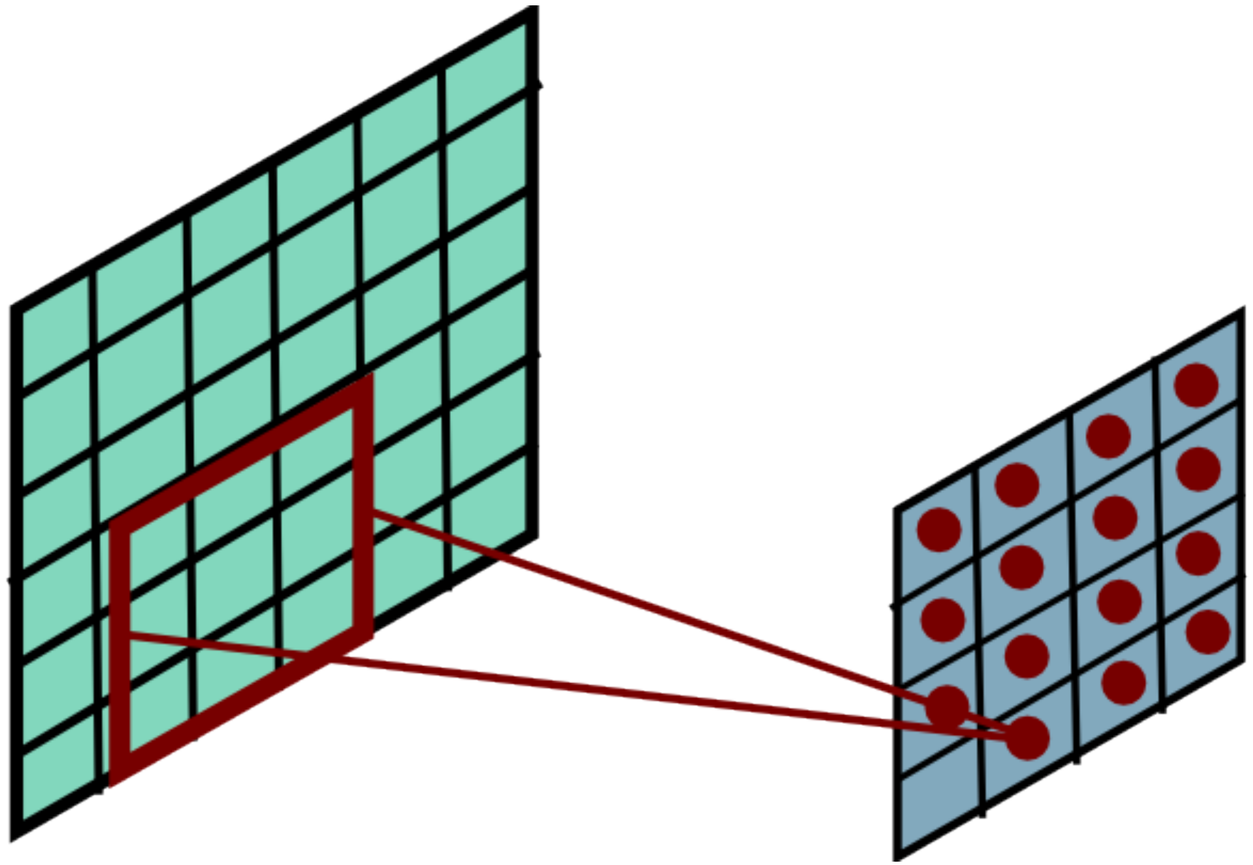
* slide from Marc'Aurelio Renzato

Convolutional Layer



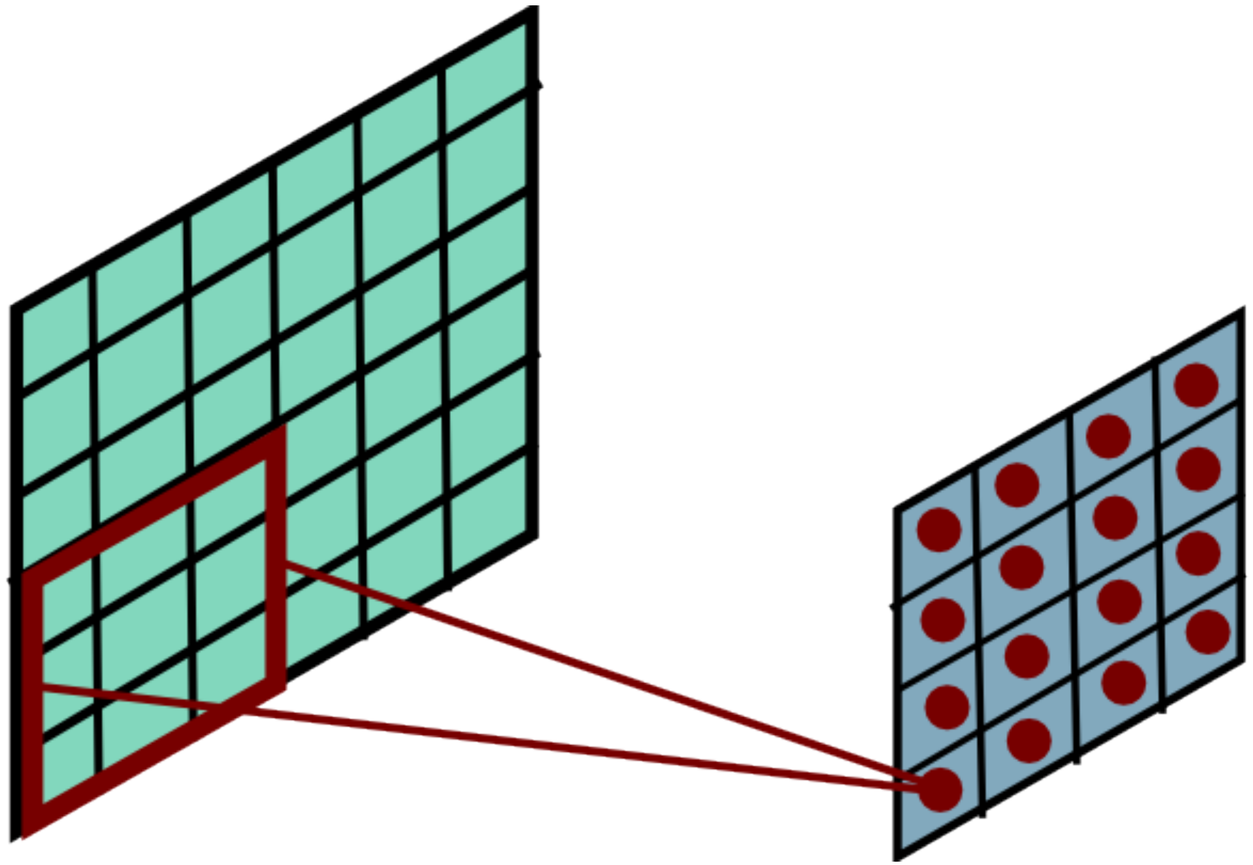
* slide from Marc'Aurelio Renzato

Convolutional Layer



* slide from Marc'Aurelio Renzato

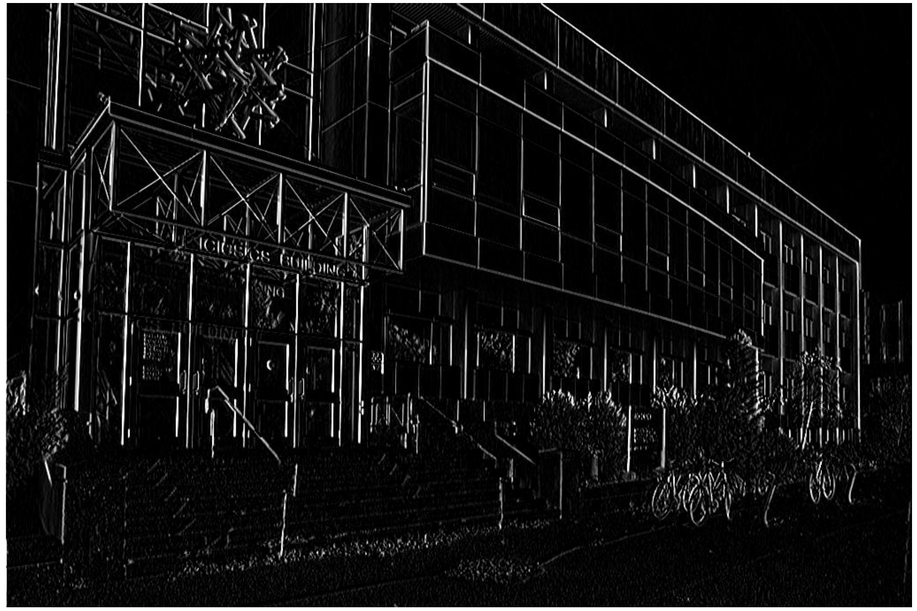
Convolutional Layer



* slide from Marc'Aurelio Renzato

Convolution Layer

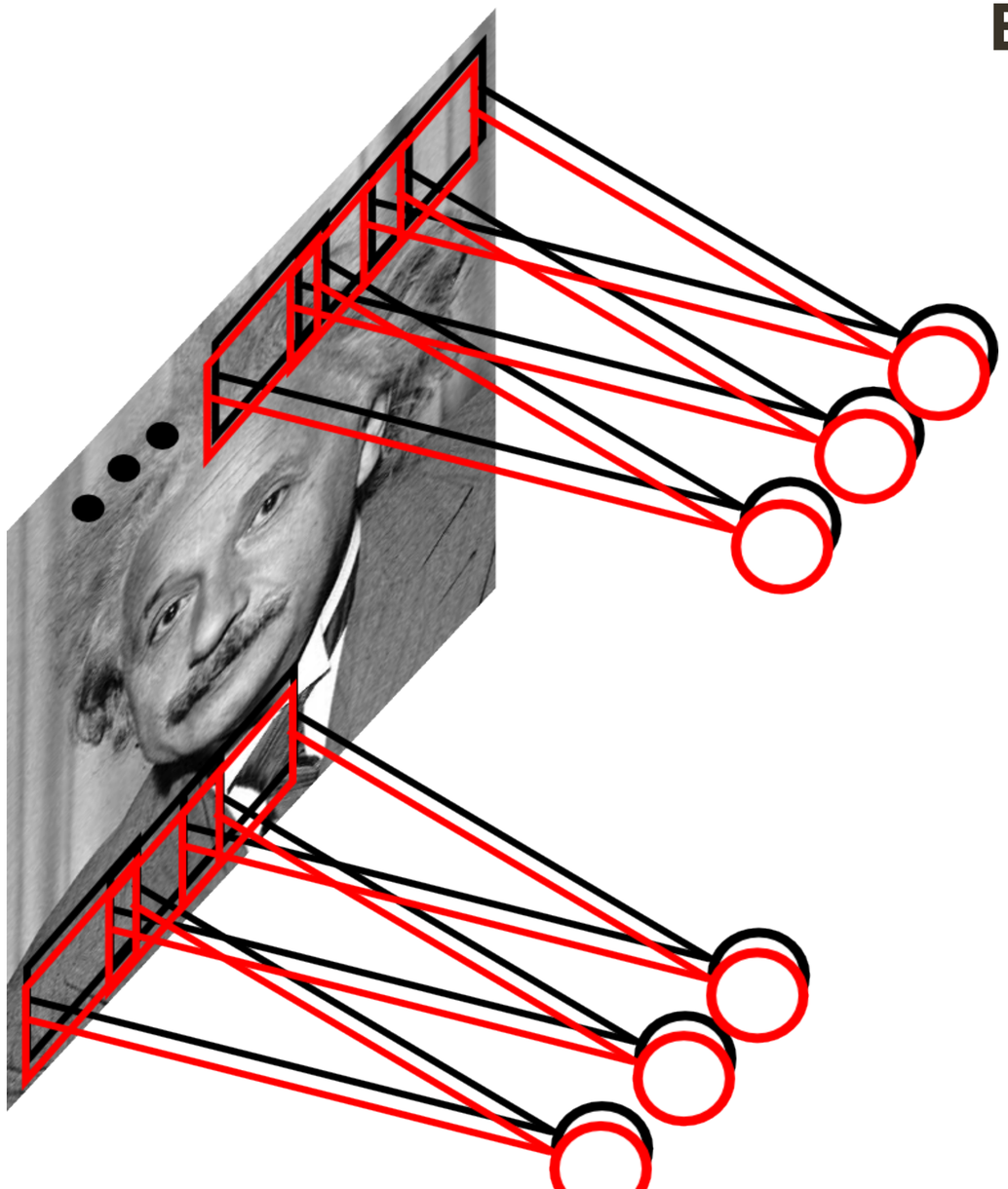
$$\star \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$



Convolution Layer

$$\star \begin{bmatrix} 0.11 & 0.11 & 0.11 \\ 0.11 & 0.11 & 0.11 \\ 0.11 & 0.11 & 0.11 \end{bmatrix} \rightarrow$$

Convolutional Layer



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

Filter size: 10 x 10

# of filters: 20

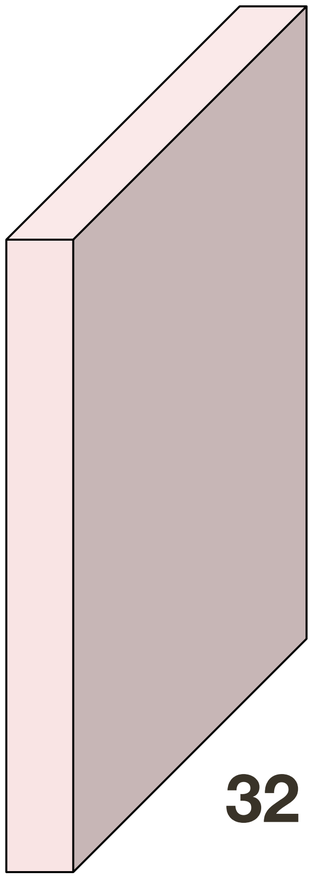
= 2000 parameters

Learn **multiple filters**

* slide from Marc'Aurelio Renzato

Convolutional Layer

32 x 32 x 3 **image** (note the image preserves spatial structure)



32 height

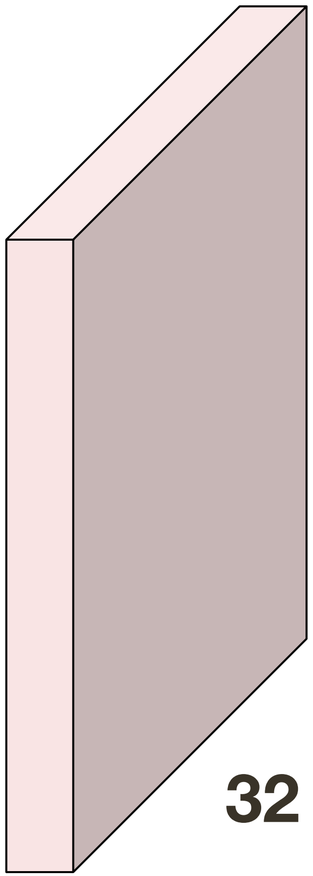
32 width

3 depth

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

Convolutional Layer

32 x 32 x 3 **image**

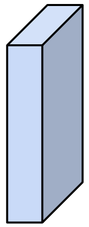


32 height

32 width

3 depth

5 x 5 x 3 **filter**

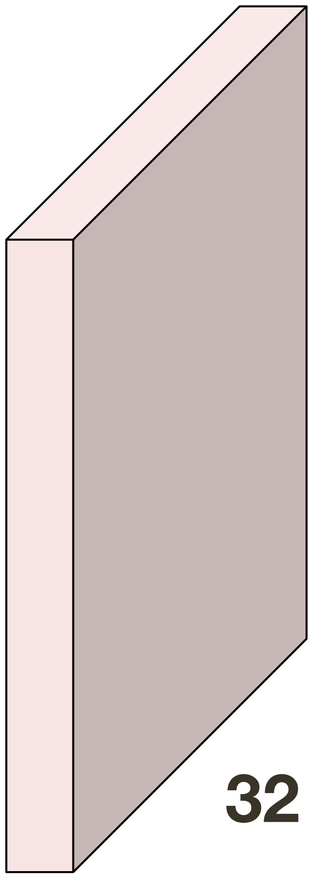


Convolve the filter with the image (i.e., “slide over the image spatially, computing dot products”)

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

Convolutional Layer

32 x 32 x 3 **image**



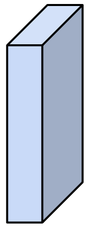
32 height

32 width

3 depth

Filters always extend the full depth of the input volume

5 x 5 x 3 **filter**

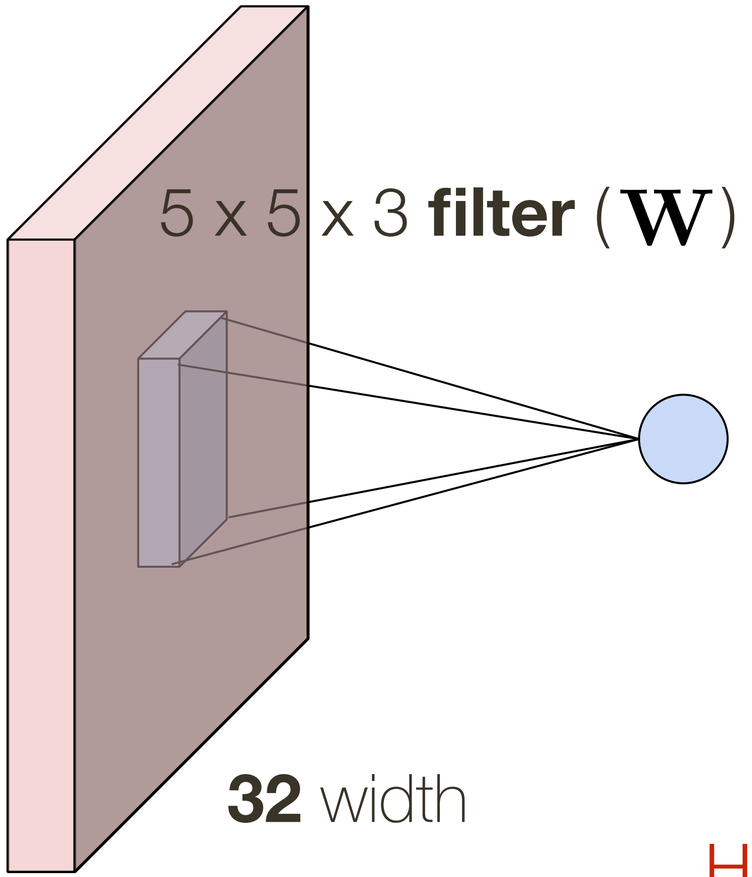


Convolve the filter with the image (i.e., “slide over the image spatially, computing dot products”

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

Convolutional Layer

32 x 32 x 3 **image**



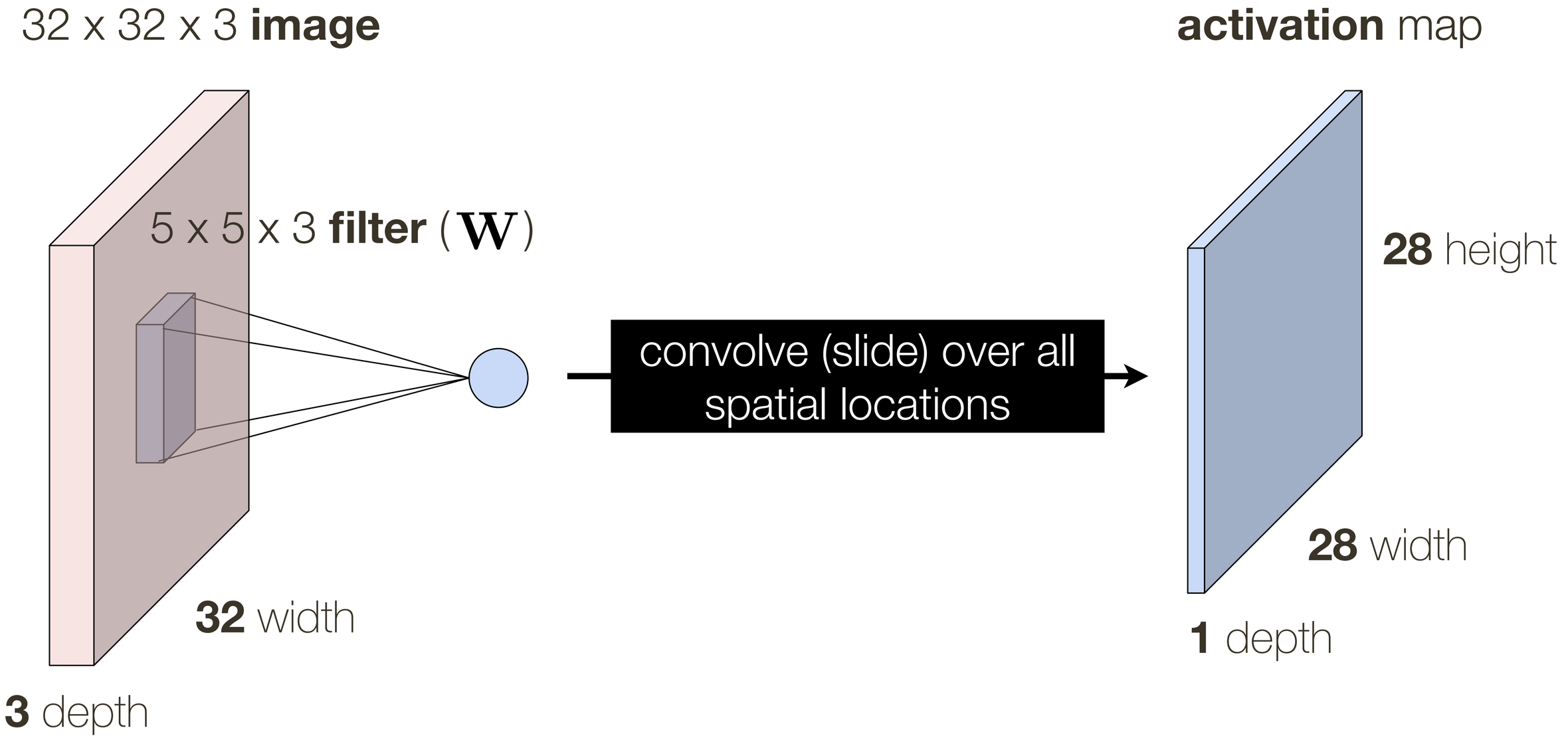
1 number: the result of taking a dot product between the filter and a small 5 x 5 x 3 part of the image

$$\mathbf{W}^T\mathbf{x} + b, \text{where } \mathbf{W}, \mathbf{x} \in \mathbb{R}^{75}$$

How many **parameters** does the layer have? **76**

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

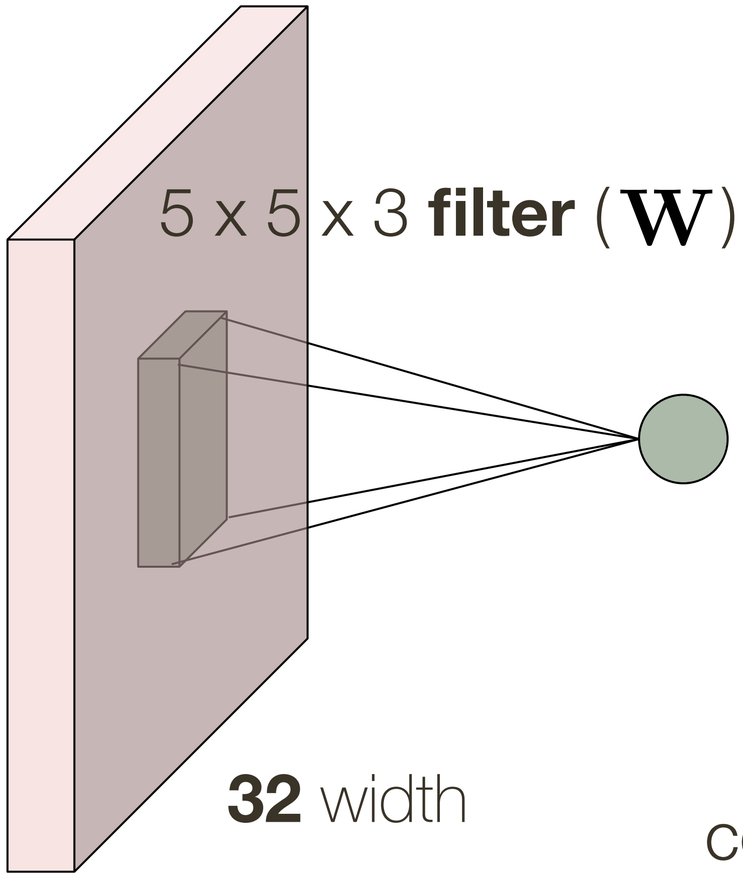
Convolutional Layer



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

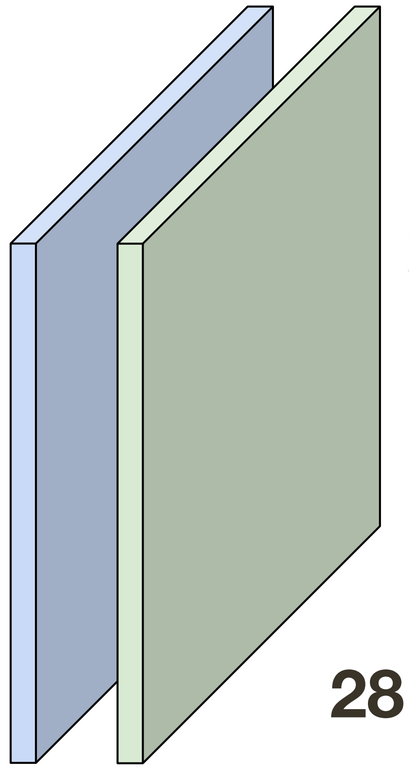
Convolutional Layer

32 x 32 x 3 **image**



convolve (slide) over all spatial locations

activation map

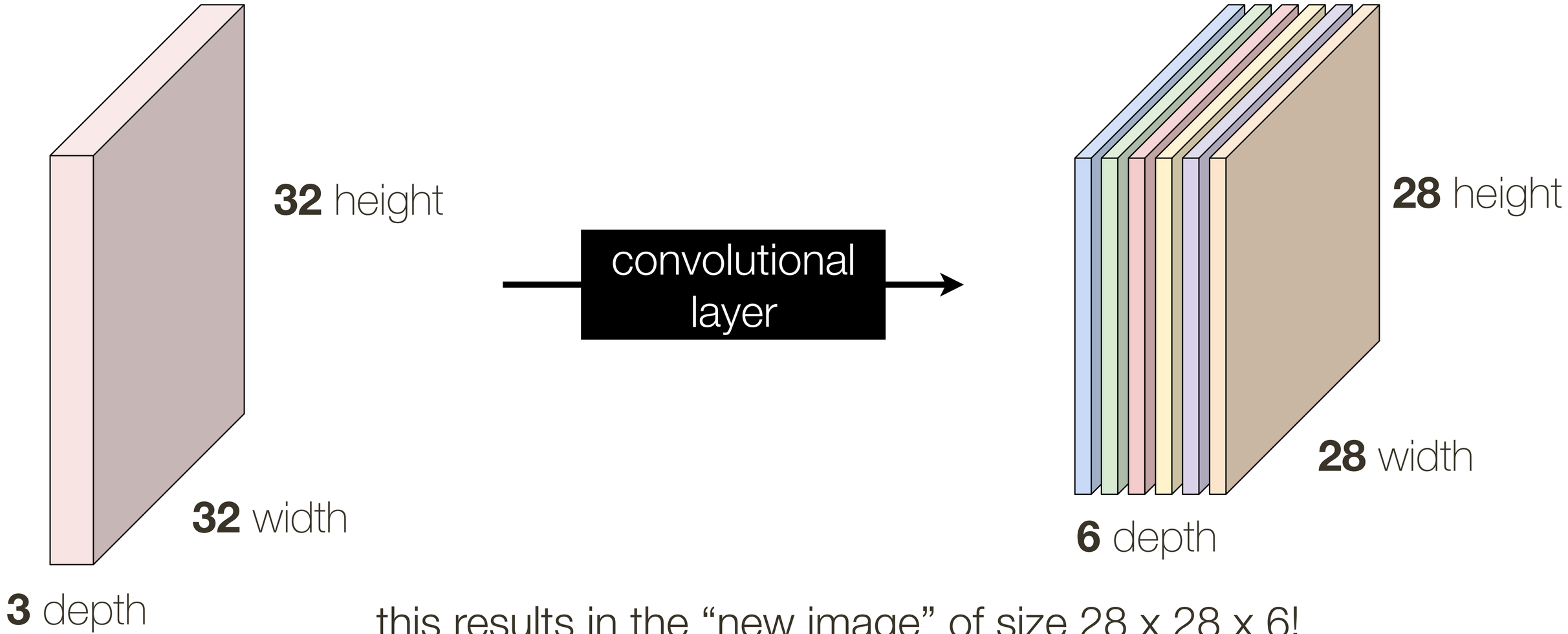


consider another **green** filter

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

Convolutional Layer

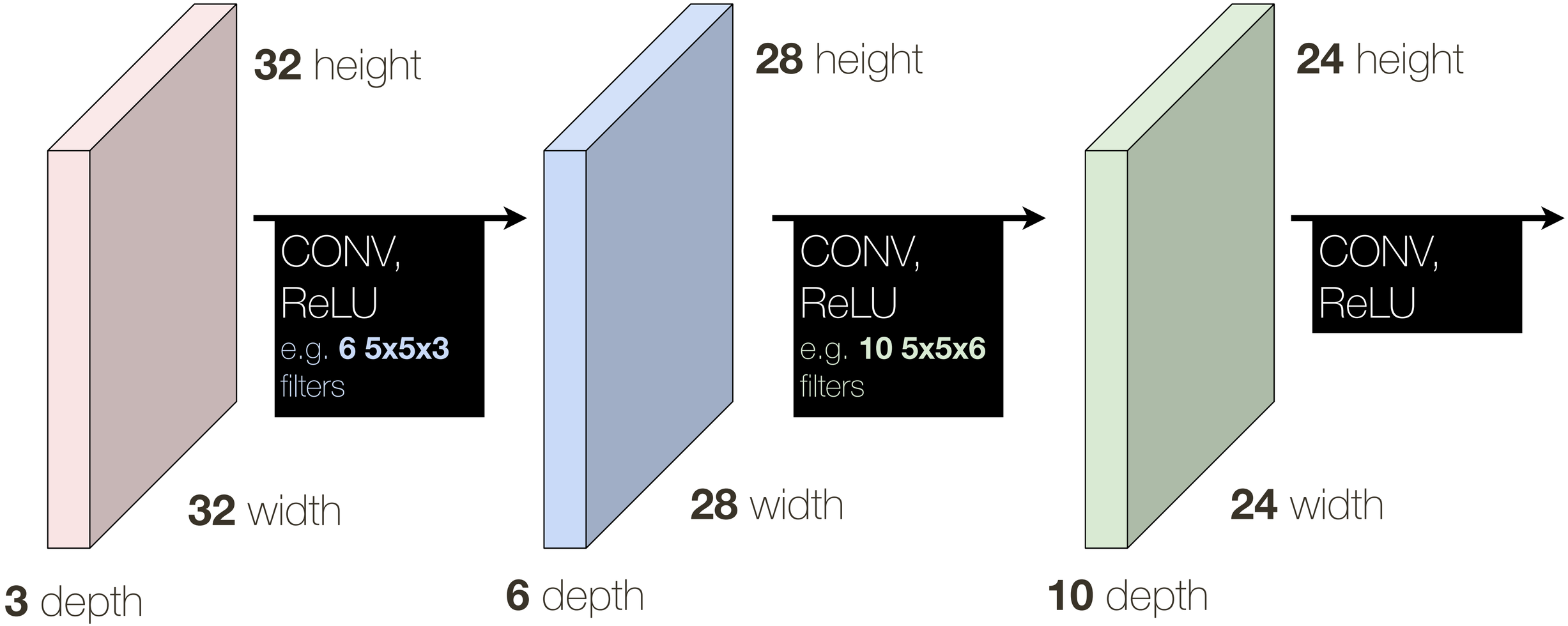
If we have 6 5x5 filter, we'll get 6 separate activation maps: **activation** map



this results in the "new image" of size 28 x 28 x 6!

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

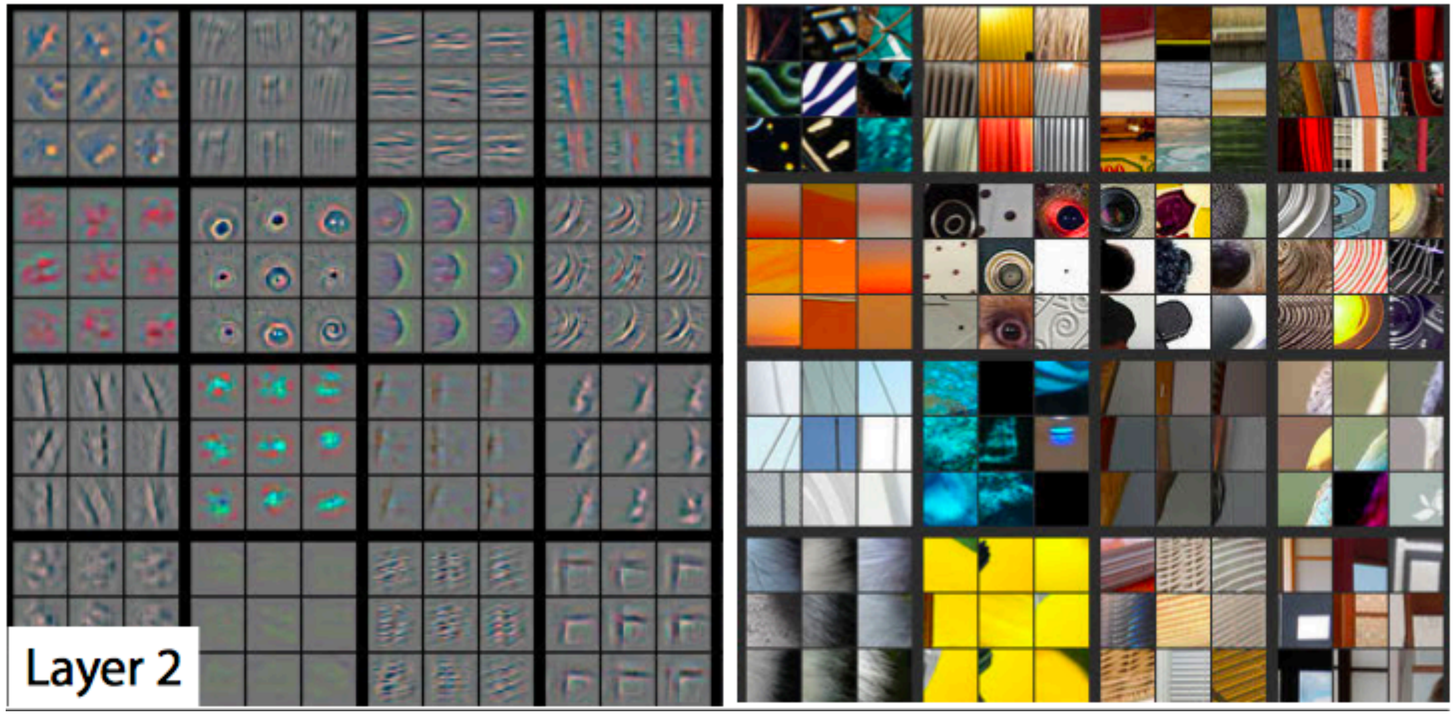
Convolutional Neural Network (ConvNet)



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

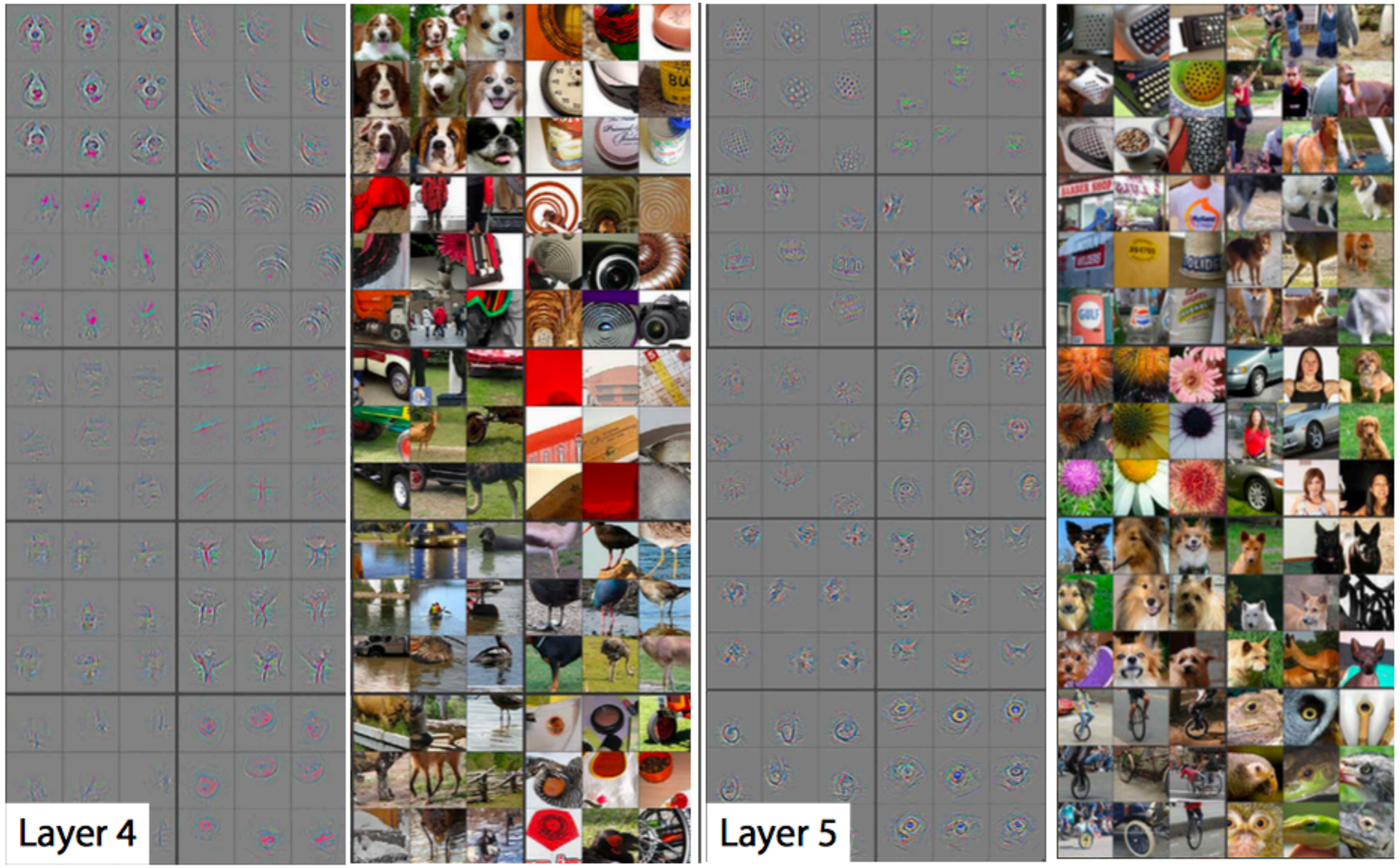
What **filters** do networks learn?

Layer 1



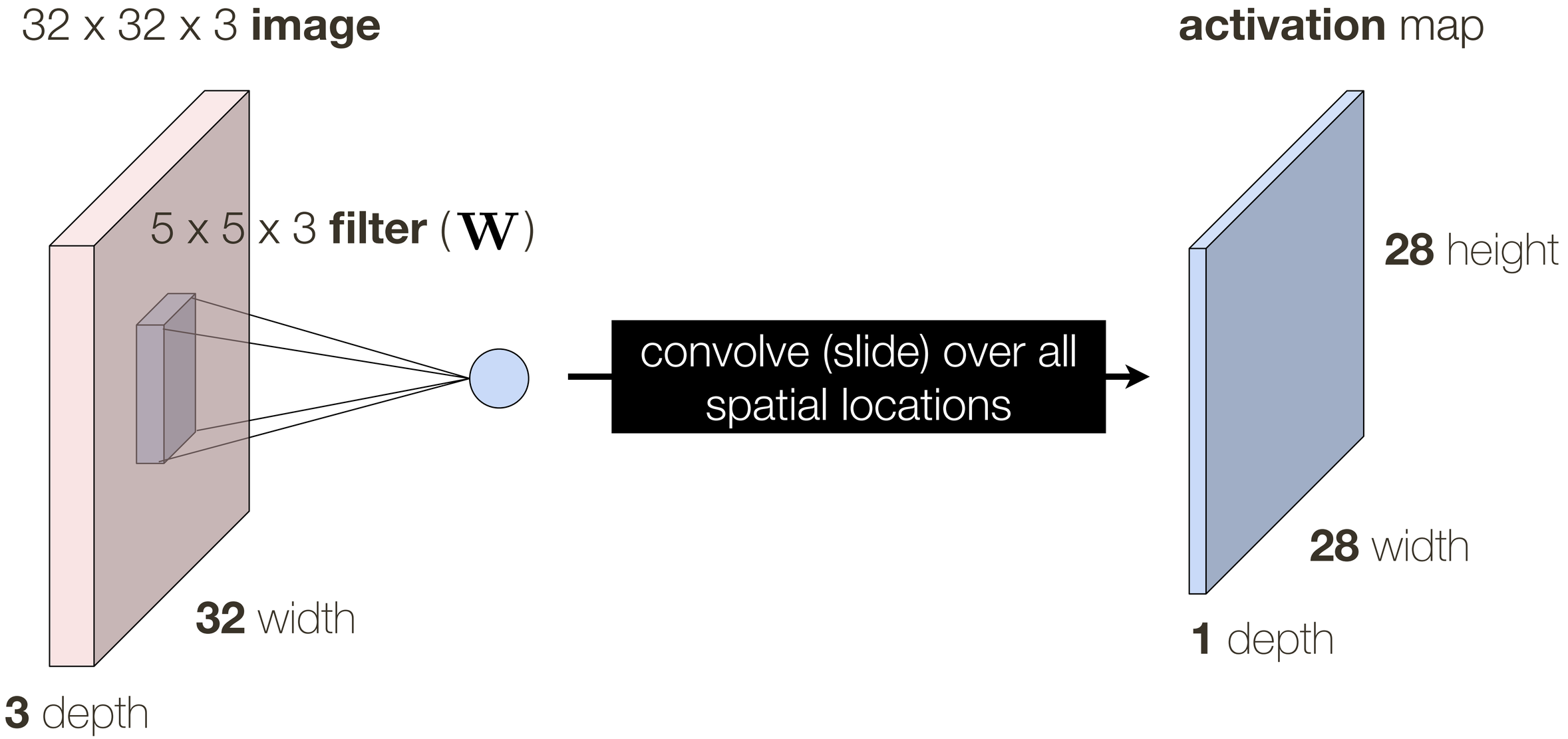
[Zeiler and Fergus, 2013]

What **filters** do networks learn?



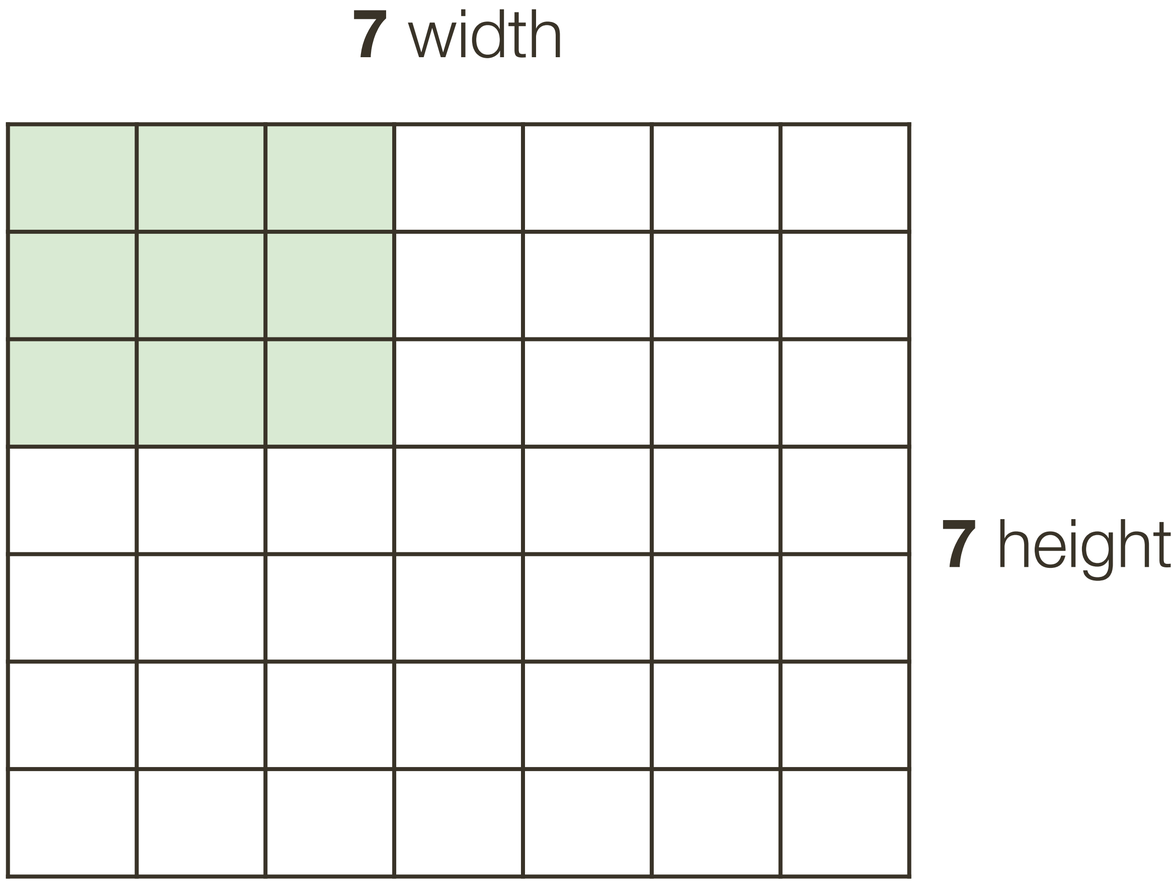
[Zeiler and Fergus, 2013]

Convolutional Layer: Closer Look at **Spatial Dimensions**



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

Convolutional Layer: Closer Look at **Spatial Dimensions**

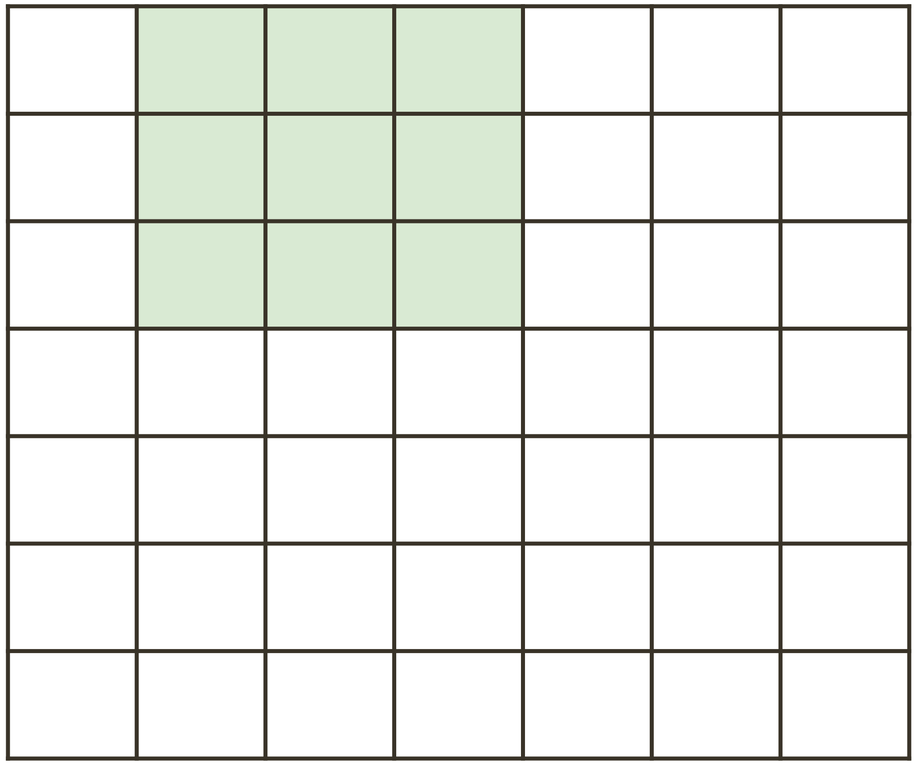


7 x 7 input image (spatially)
3 x 3 filter

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

Convolutional Layer: Closer Look at **Spatial Dimensions**

7 width

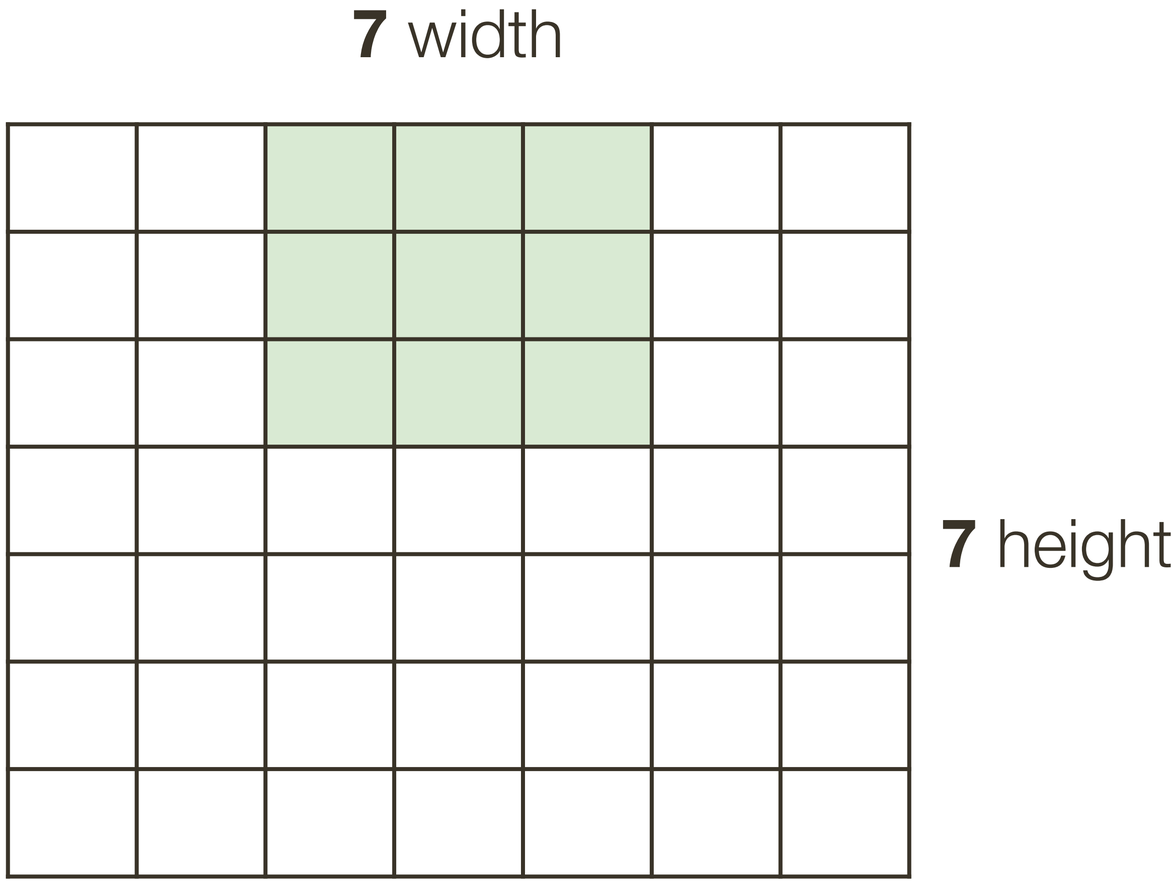


7 height

7 x 7 input image (spatially)
3 x 3 filter

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

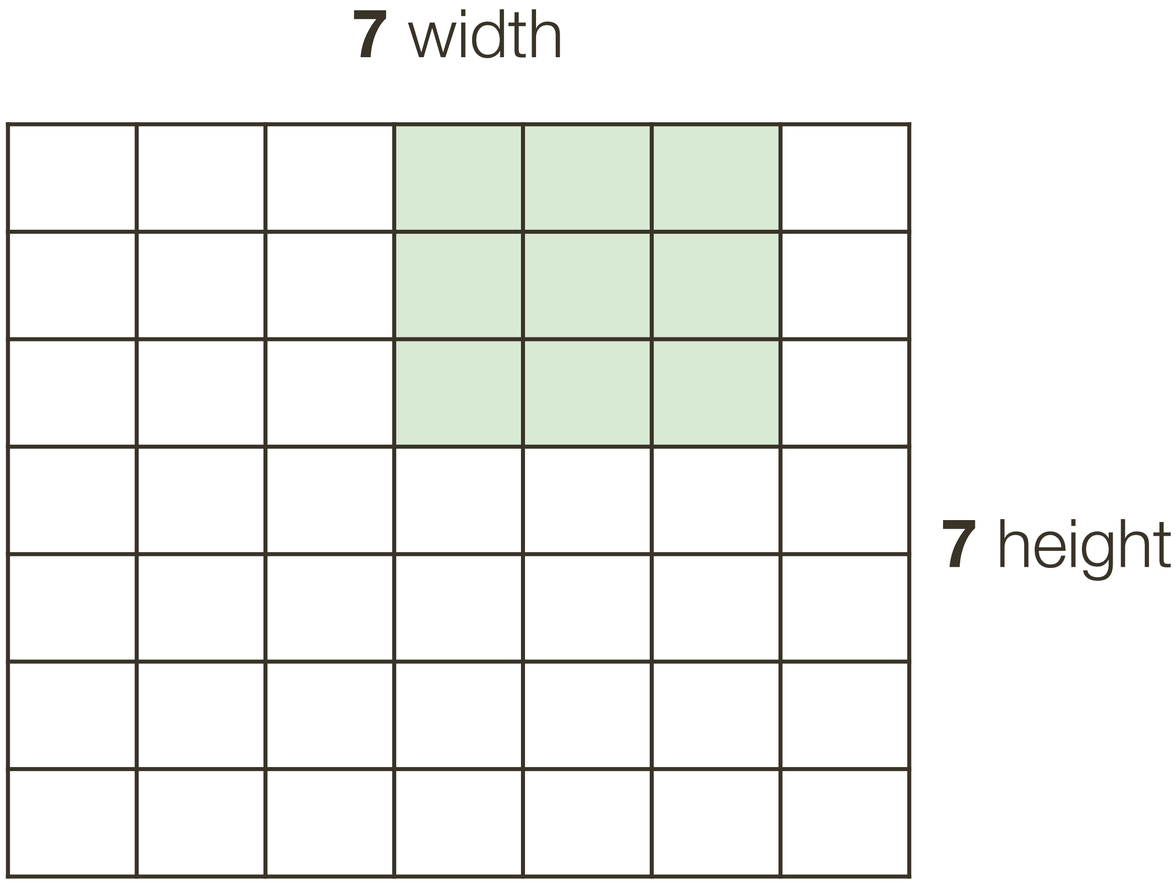
Convolutional Layer: Closer Look at **Spatial Dimensions**



7 x 7 input image (spatially)
3 x 3 filter

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

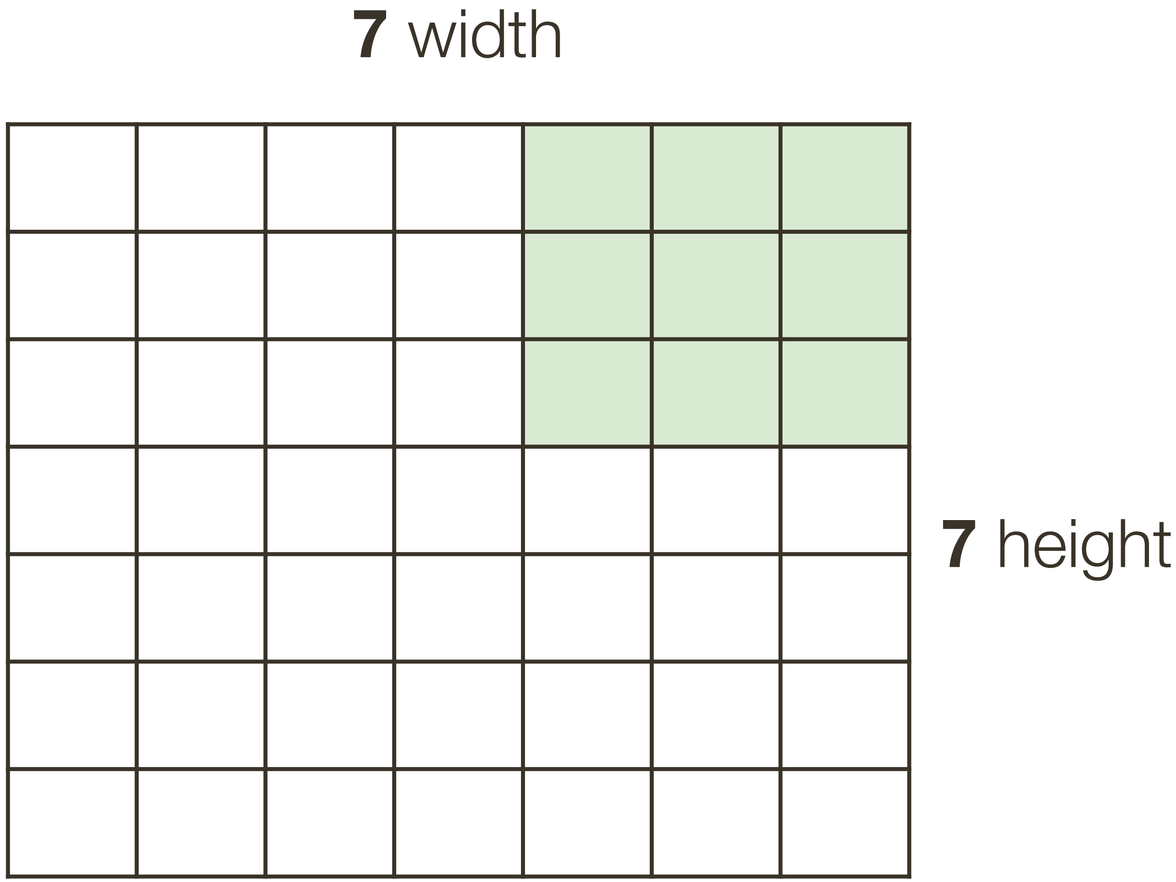
Convolutional Layer: Closer Look at **Spatial Dimensions**



7 x 7 input image (spatially)
3 x 3 filter

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

Convolutional Layer: Closer Look at **Spatial Dimensions**



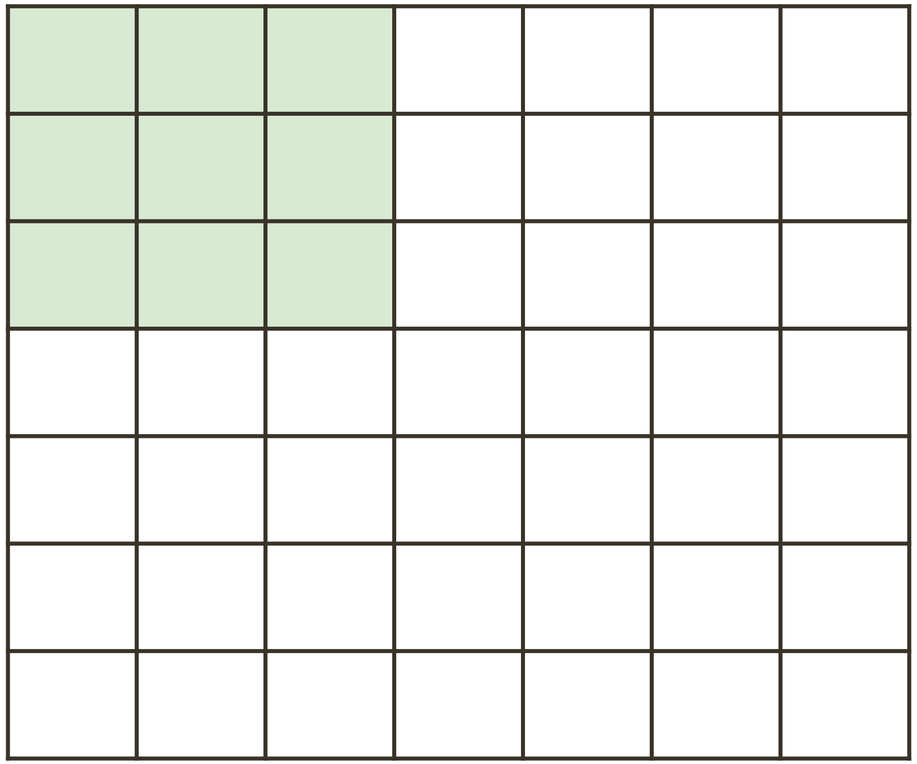
7 x 7 input image (spatially)
3 x 3 filter

=> **5 x 5 output**

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

Convolutional Layer: Closer Look at **Spatial Dimensions**

7 width



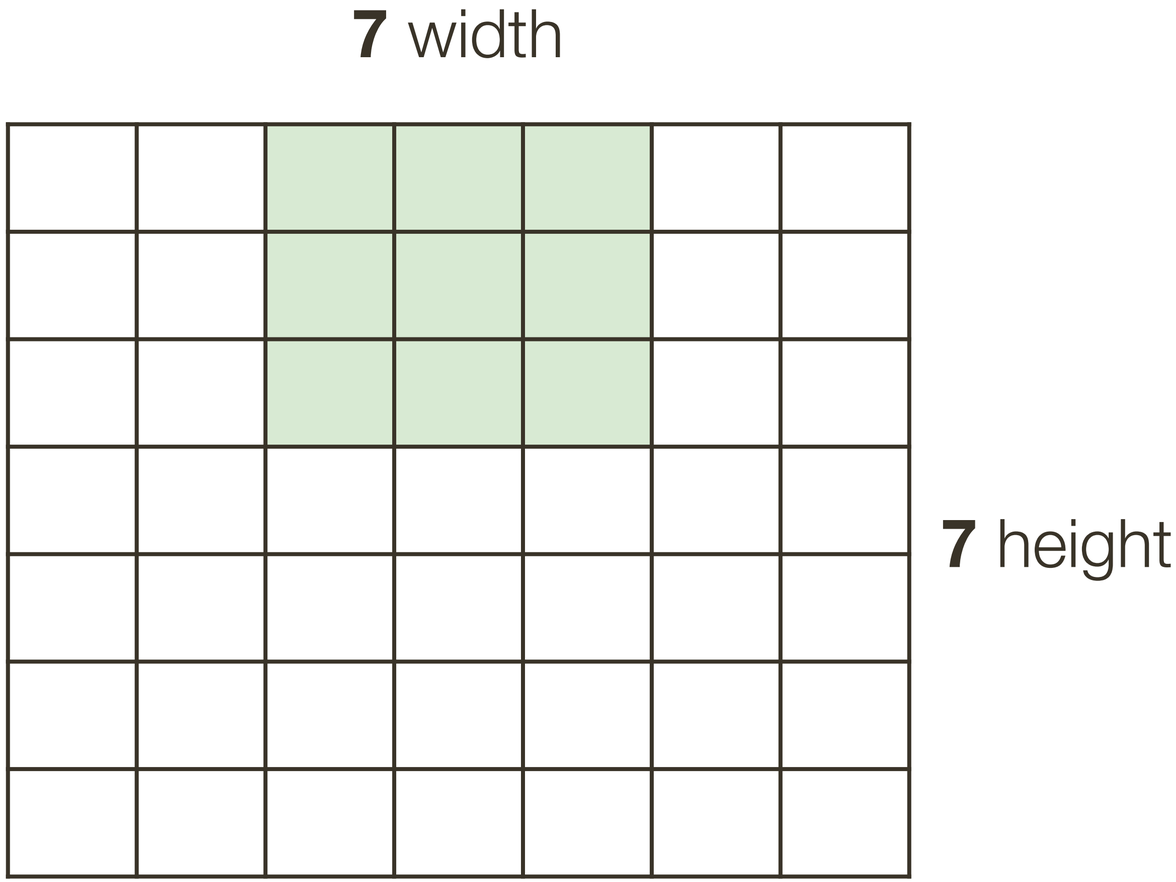
7 height

7 x 7 input image (spatially)
3 x 3 filter

(applied with **stride 2**)

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

Convolutional Layer: Closer Look at **Spatial Dimensions**

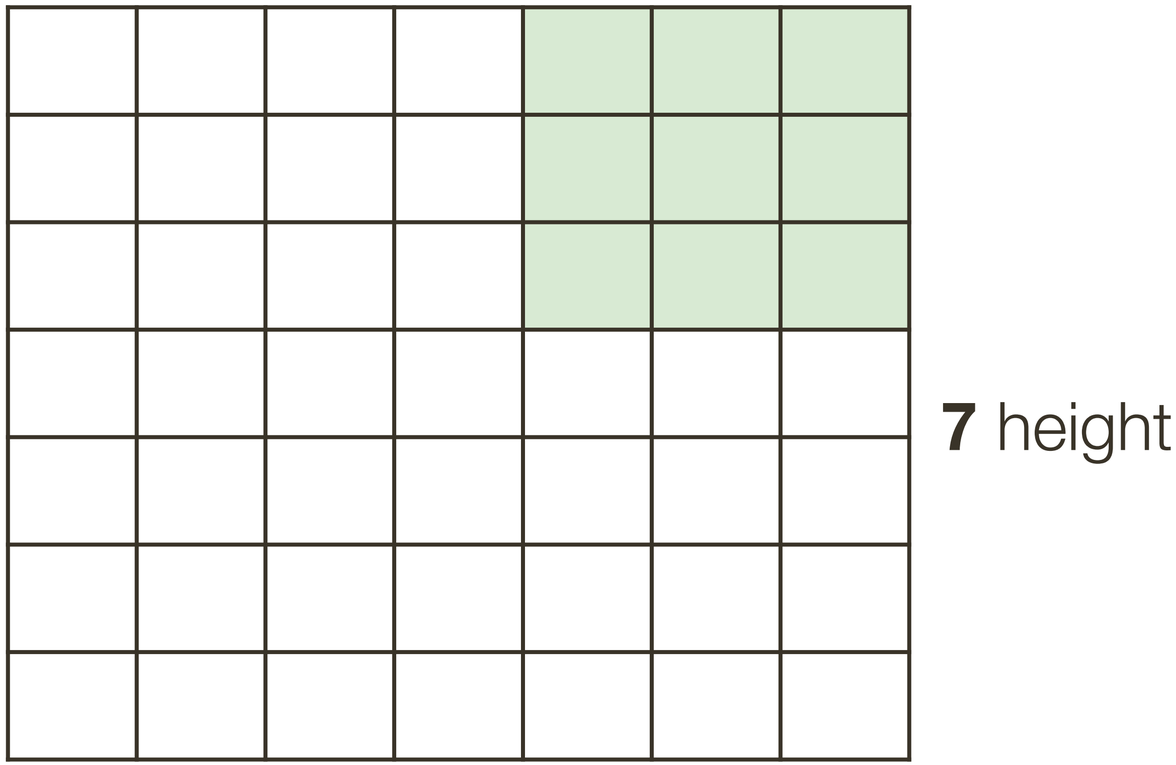


7 x 7 input image (spatially)
3 x 3 filter

(applied with **stride 2**)

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

Convolutional Layer: Closer Look at **Spatial Dimensions**



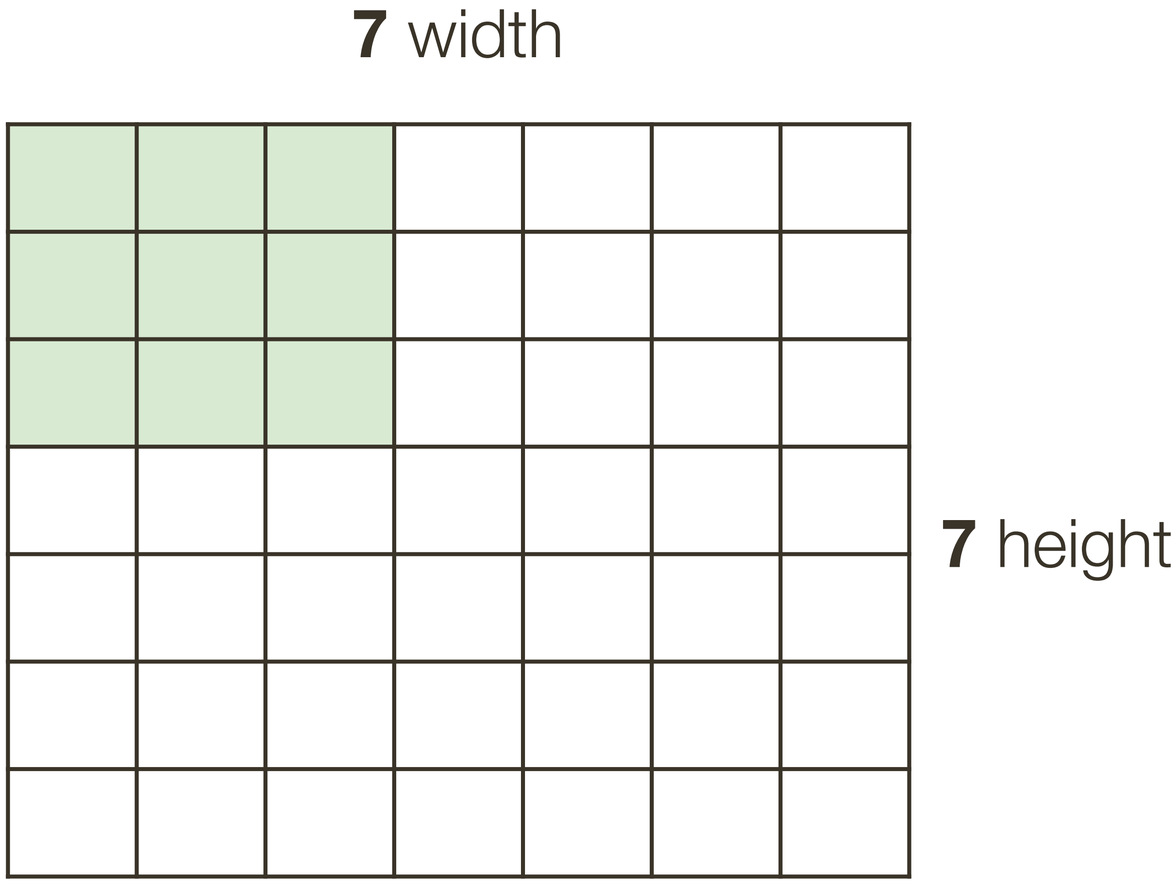
7 x 7 input image (spatially)
3 x 3 filter

(applied with **stride 2**)

=> **3 x 3 output**

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

Convolutional Layer: Closer Look at **Spatial Dimensions**



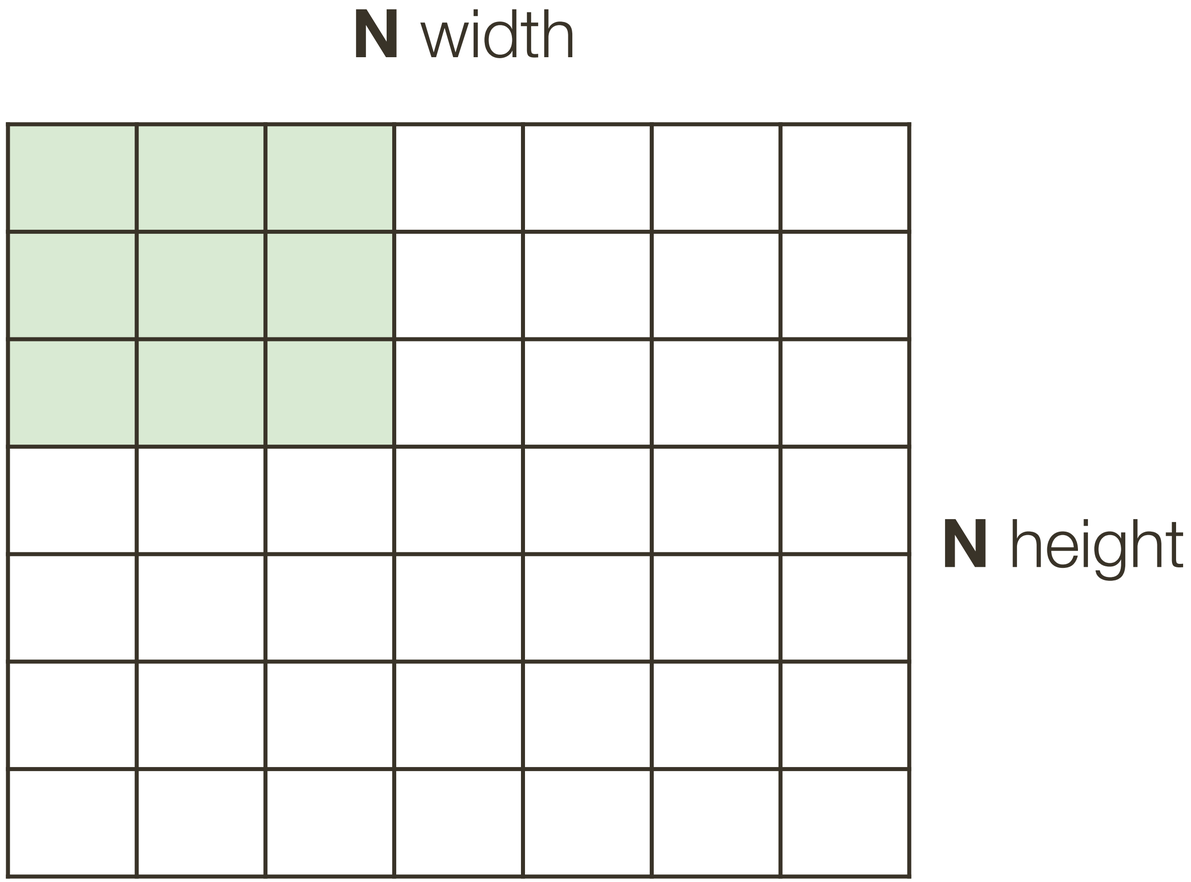
7 x 7 input image (spatially)
3 x 3 filter

(applied with **stride 3**)

Does not fit! **Cannot apply** 3 x 3 filter on 7 x 7 image with stride 3

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

Convolutional Layer: Closer Look at **Spatial Dimensions**



N x N input image (spatially)
F x F filter

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

Example: N = 7, F = 3

stride 1 => (7-3)/1+1 = 5
stride 2 => (7-3)/2+1 = 3
stride 3 => (7-3)/3+1 = **2.33**

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

Convolutional Layer: **Border padding**

7 width

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

7 height

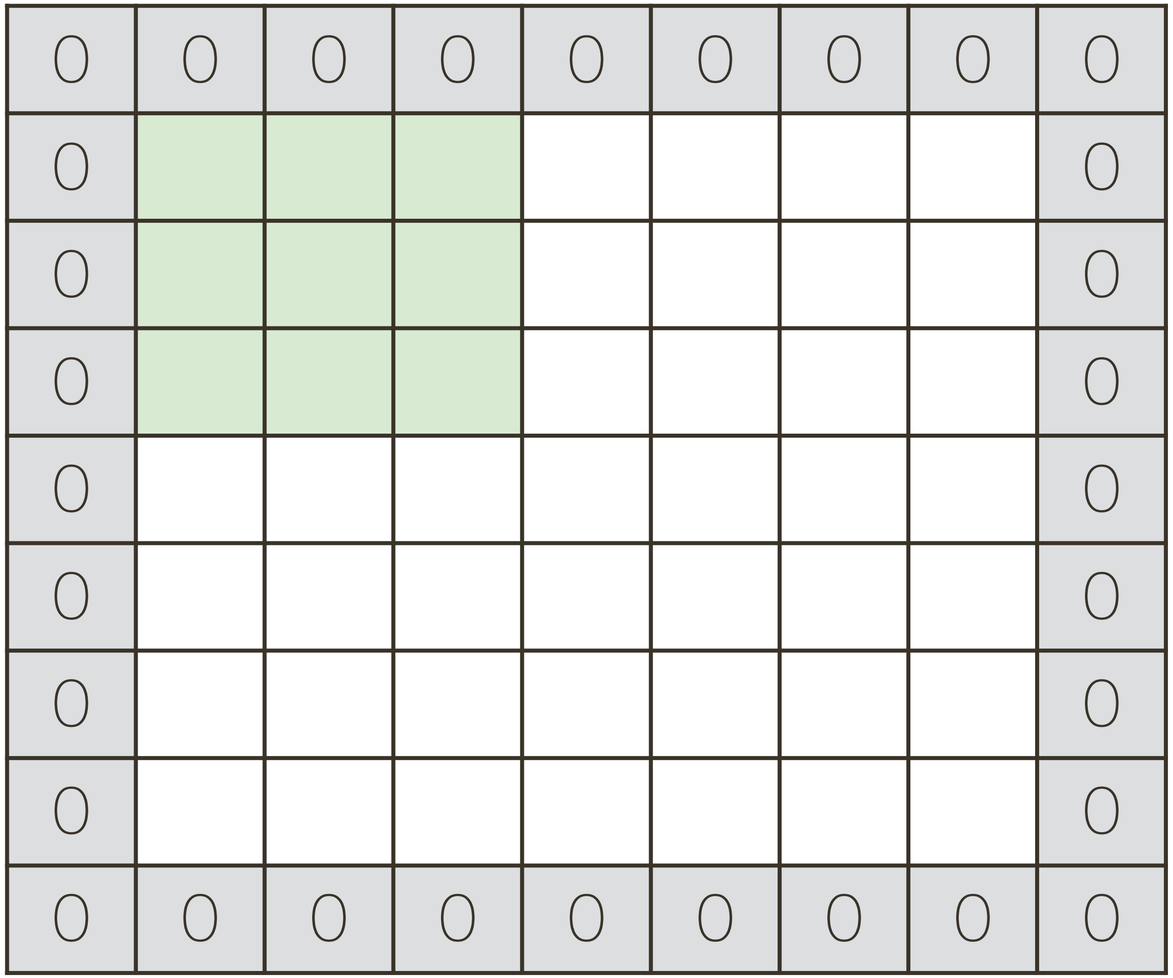
7 x 7 input image (spatially)
3 x 3 filter
(applied with **stride 1**)

pad with 1 pixel border

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

Convolutional Layer: **Border padding**

7 width



7 height

7 x 7 input image (spatially)
3 x 3 filter
(applied with **stride 1**)

pad with 1 pixel border

Output size: 7 x 7

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

Convolutional Layer: **Border padding**

7 width

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

7 height

7 x 7 input image (spatially)
3 x 3 filter
(applied with **stride 3**)

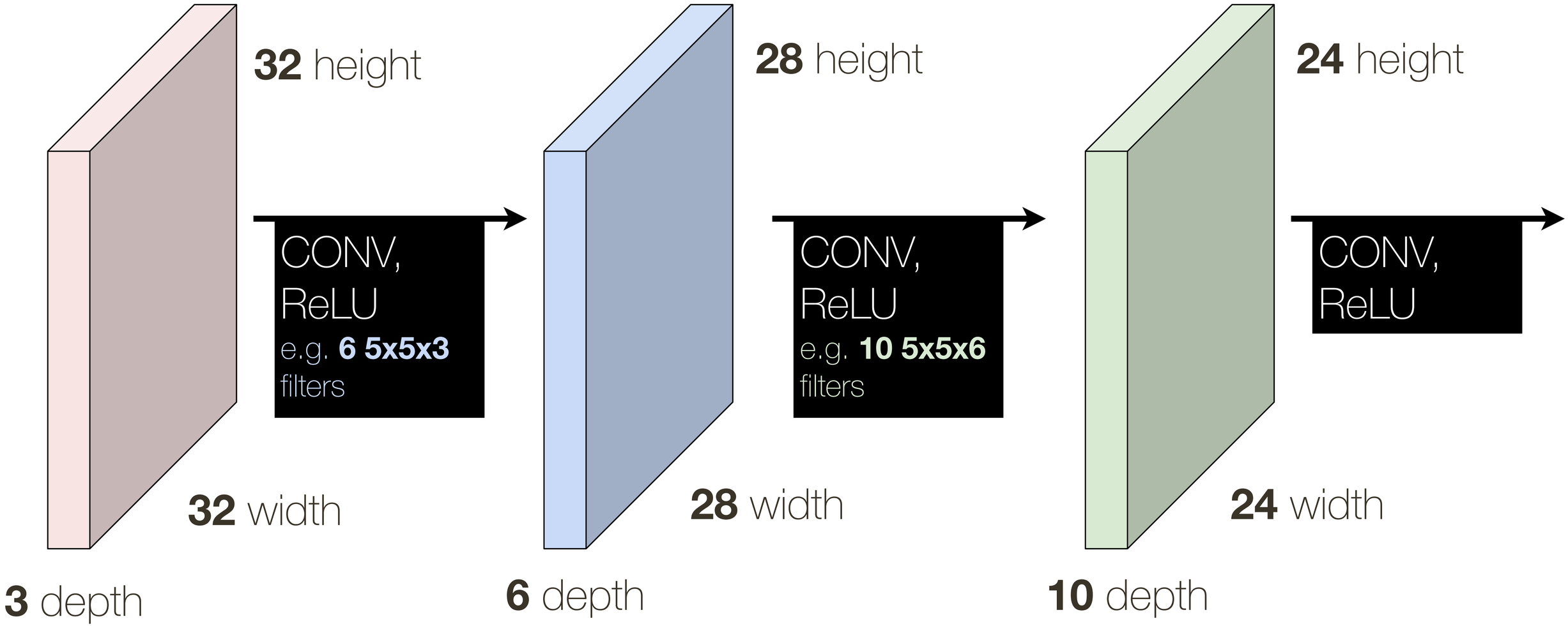
pad with 1 pixel border

Example: N = 7, F = 3

stride 1 => (9-3)/1+1 = 7
stride 2 => (9-3)/2+1 = 4
stride 3 => (9-3)/3+1 = **3**

Convolutional Neural Network (ConvNet)

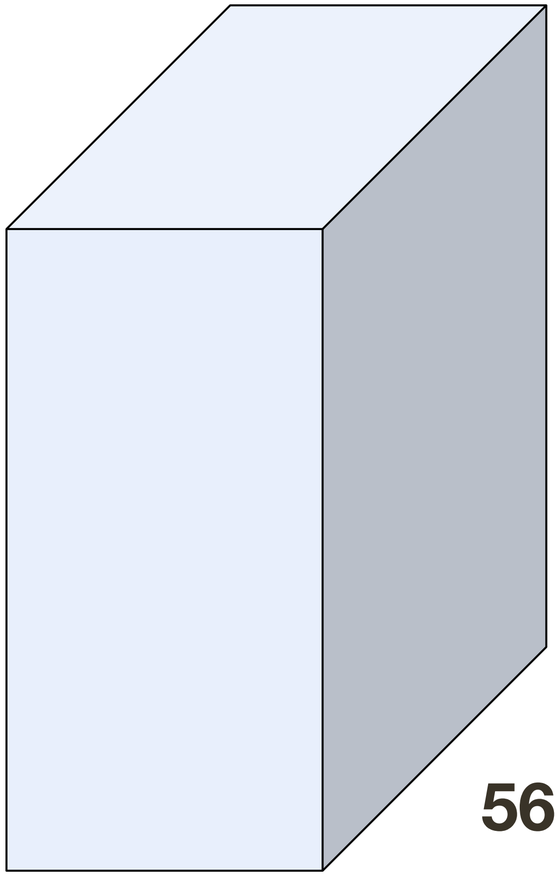
With padding we can achieve no shrinking (32 -> 28 -> 24); shrinking quickly (which happens with larger filters) doesn't work well in practice



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

Convolutional Layer: **1x1 convolutions**

56 x 56 x 64 **image**



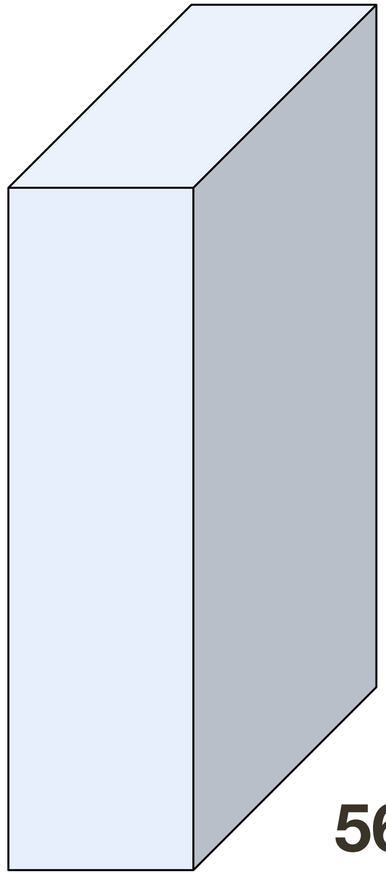
56 height

56 width

64 depth

32 **filters** of size, 1 x 1 x 64

56 x 56 x 32 **image**



56 height

56 width

32 depth

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

Convolutional Layer **Summary**

Accepts a volume of size: $W_i \times H_i \times D_i$

Convolutional Layer **Summary**

Accepts a volume of size: $W_i \times H_i \times D_i$

Requires hyperparameters:

- Number of filters: K (for typical networks $K \in \{32, 64, 128, 256, 512\}$)
- Spatial extent of filters: F (for a typical networks $F \in \{1, 3, 5, ...\}$)
- Stride of application: S (for a typical network $S \in \{1, 2\}$)
- Zero padding: P (for a typical network $P \in \{0, 1, 2\}$)

Convolutional Layer **Summary**

Accepts a volume of size: $W_i \times H_i \times D_i$

Requires hyperparameters:

- Number of filters: K (for typical networks $K \in \{32, 64, 128, 256, 512\}$)
- Spatial extent of filters: F (for a typical networks $F \in \{1, 3, 5, ...\}$)
- Stride of application: S (for a typical network $S \in \{1, 2\}$)
- Zero padding: P (for a typical network $P \in \{0, 1, 2\}$)

Produces a volume of size: $W_o \times H_o \times D_o$

Convolutional Layer **Summary**

Accepts a volume of size: $W_i \times H_i \times D_i$

Requires hyperparameters:

- Number of filters: K (for typical networks $K \in \{32, 64, 128, 256, 512\}$)
- Spatial extent of filters: F (for a typical networks $F \in \{1, 3, 5, ...\}$)
- Stride of application: S (for a typical network $S \in \{1, 2\}$)
- Zero padding: P (for a typical network $P \in \{0, 1, 2\}$)

Produces a volume of size: $W_o \times H_o \times D_o$

$$W_o = (W_i - F + 2P)/S + 1 \qquad H_o = (H_i - F + 2P)/S + 1 \qquad D_o = K$$

Convolutional Layer **Summary**

Accepts a volume of size: $W_i \times H_i \times D_i$

Requires hyperparameters:

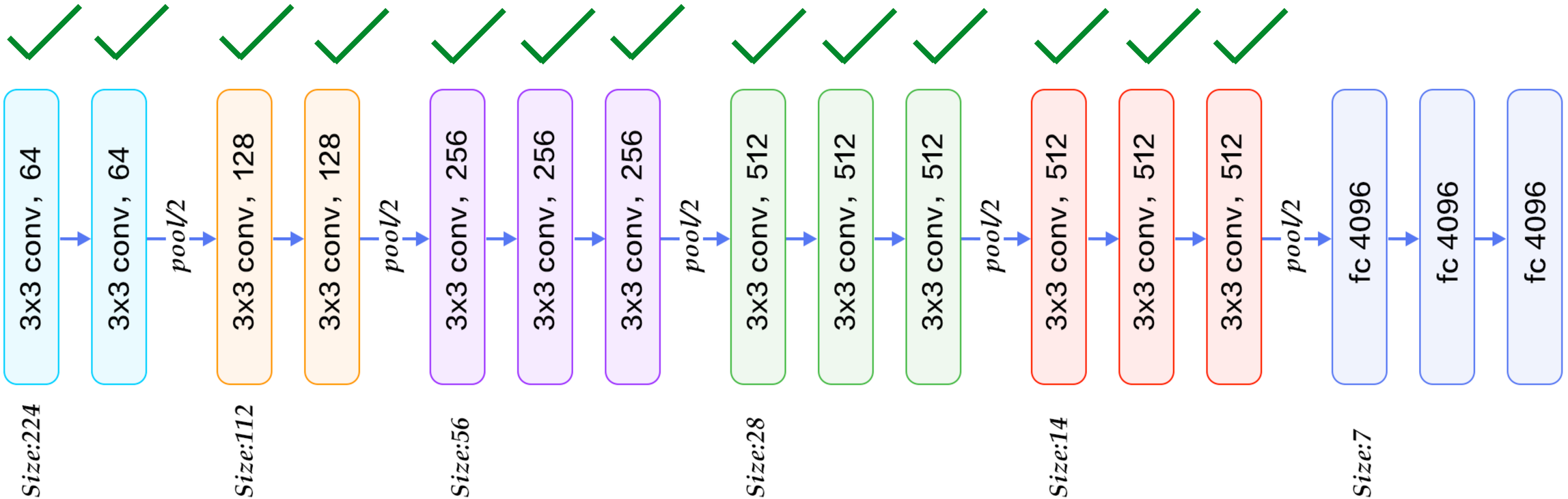
- Number of filters: K (for typical networks $K \in \{32, 64, 128, 256, 512\}$)
- Spatial extent of filters: F (for a typical networks $F \in \{1, 3, 5, ...\}$)
- Stride of application: S (for a typical network $S \in \{1, 2\}$)
- Zero padding: P (for a typical network $P \in \{0, 1, 2\}$)

Produces a volume of size: $W_o \times H_o \times D_o$

$$W_o = (W_i - F + 2P)/S + 1 \qquad H_o = (H_i - F + 2P)/S + 1 \qquad D_o = K$$

Number of total learnable parameters: $(F \times F \times D_i) \times K + K$

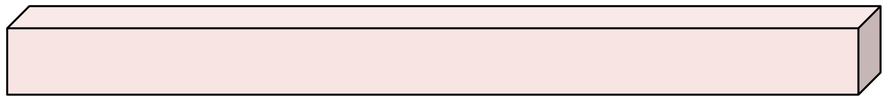
Convolutional Neural Networks



VGG-16 Network

CNNs: Reminder Fully Connected Layers

Input



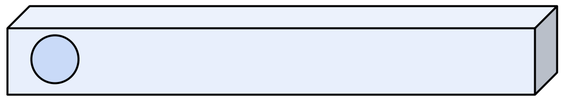
3072

(32 x 32 x 3 image -> stretches to 3072 x 1)

$\mathbf{W}^T\mathbf{x} + \mathbf{b}$, where $\mathbf{W} \in \mathbb{R}^{10\times3072}$

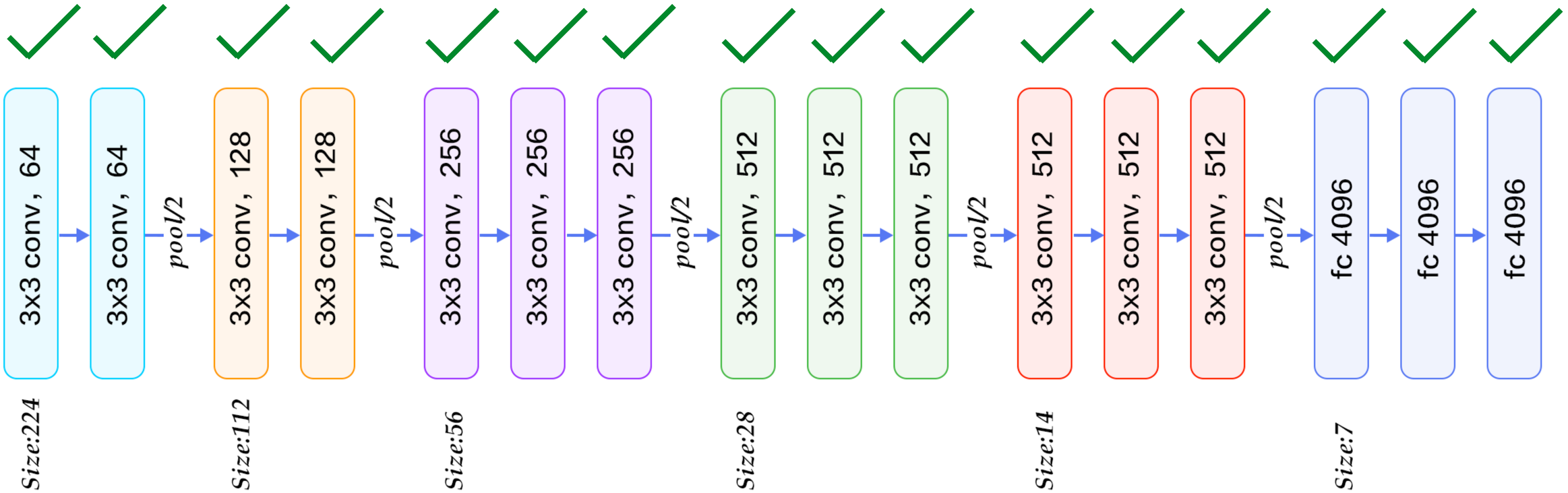
each neuron looks at the full input volume

Activation



10

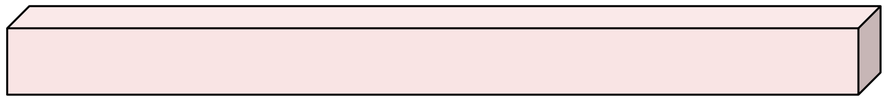
Convolutional Neural Networks



VGG-16 Network

CNNs: Reminder Fully Connected Layers

Input



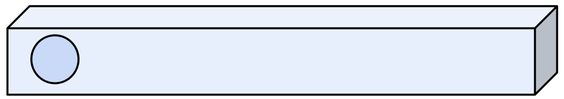
25,088

(7 x 7 x 512 image -> stretches to 25,088 x 1)

$\mathbf{W}^T\mathbf{x} + \mathbf{b}$, where $\mathbf{W} \in \mathbb{R}^{25,088 \times 4,096}$

each neuron looks at the full input volume

Activation

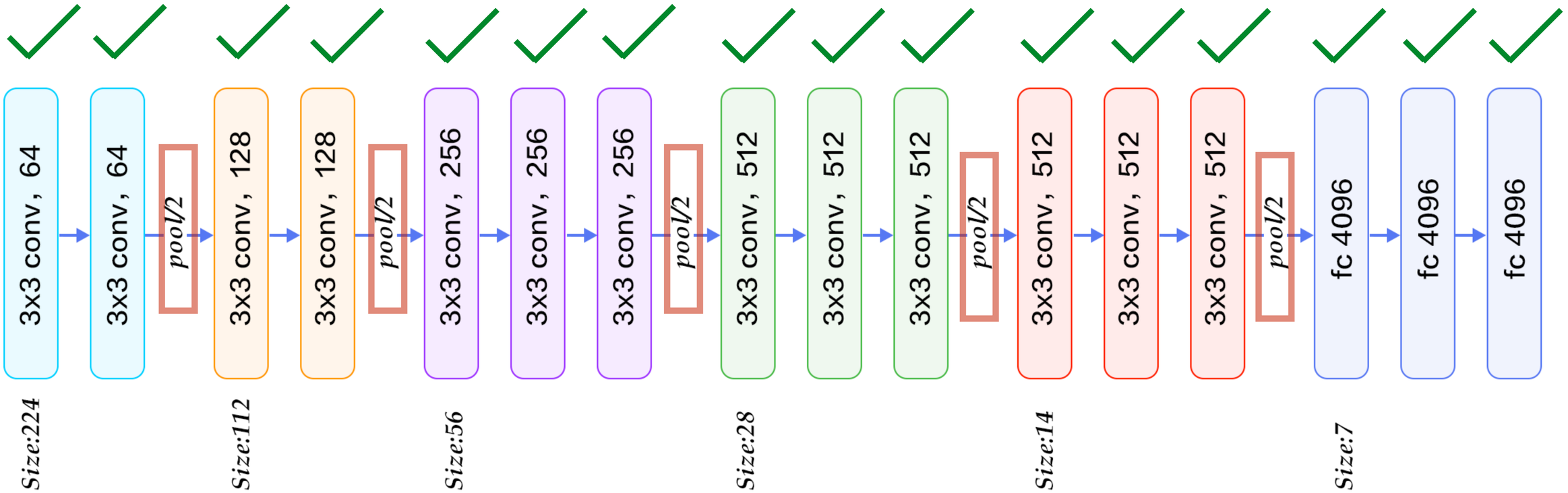


4,096

102,760,448 parameters!

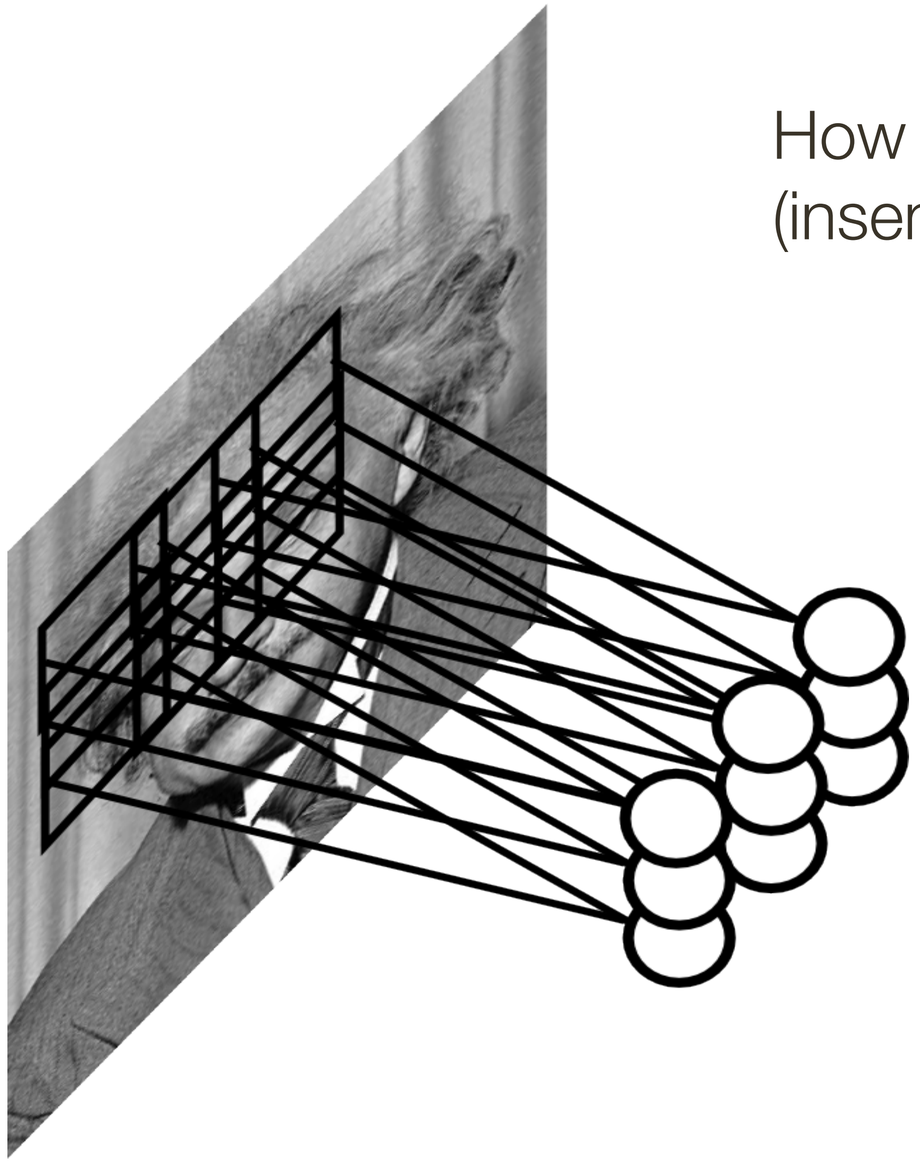
* adopted from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Convolutional Neural Networks



VGG-16 Network

Pooling Layer



Let us assume the filter is an “eye” detector

How can we make detection spatially invariant (insensitive to position of the eye in the image)

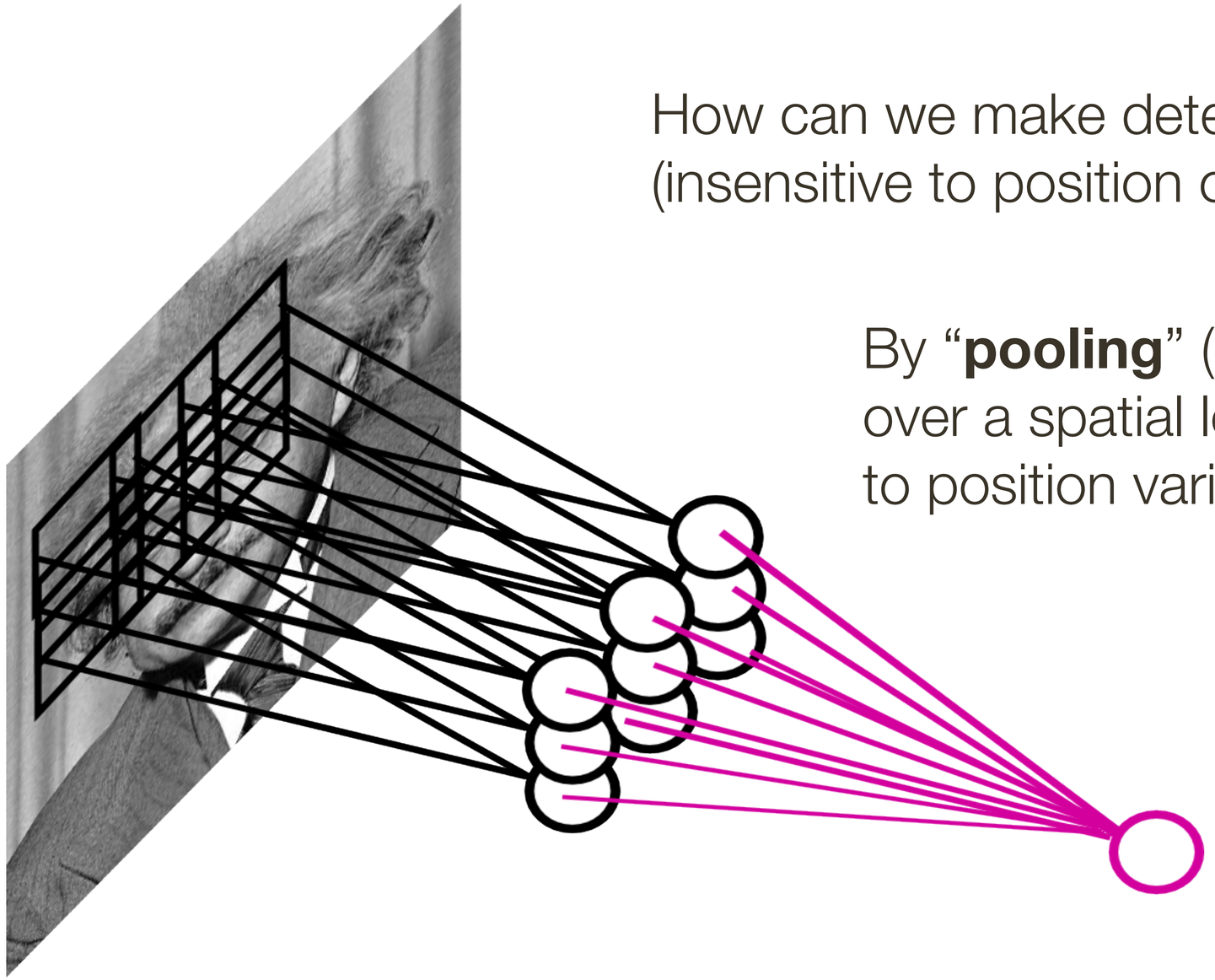
* slide from Marc’Aurelio Renzato

Pooling Layer

Let us assume the filter is an “eye” detector

How can we make detection spatially invariant (insensitive to position of the eye in the image)

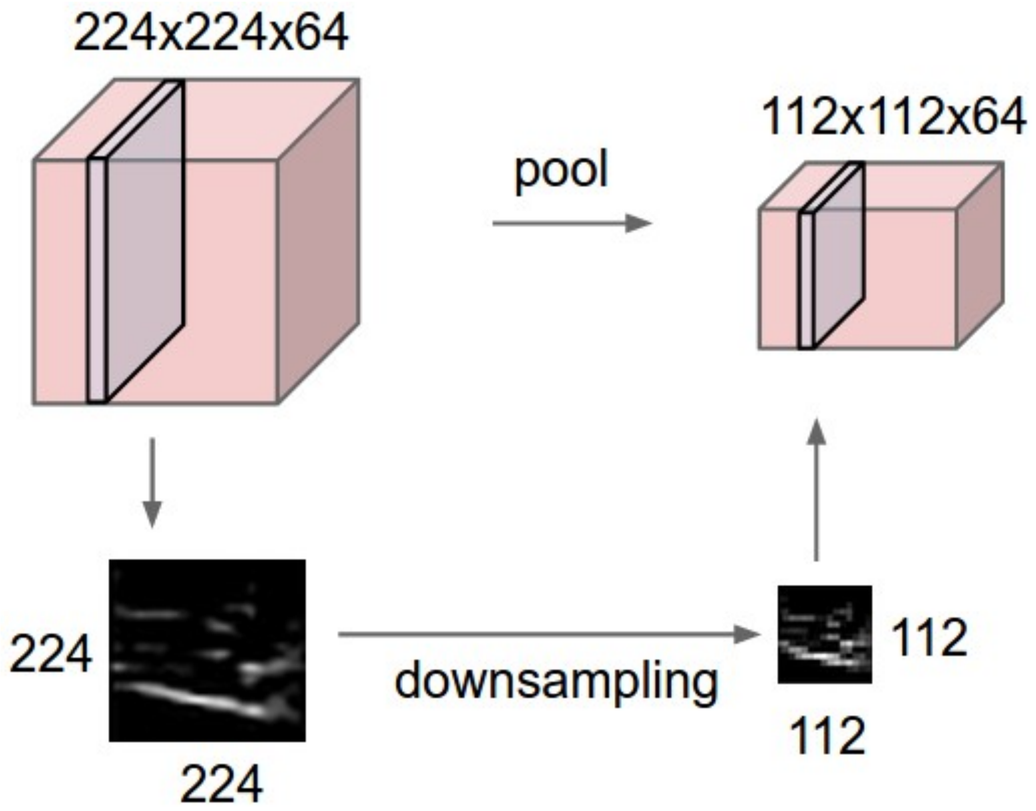
By “**pooling**” (e.g., taking a max) response over a spatial locations we gain robustness to position variations



* slide from Marc’Aurelio Renzato

Pooling Layer

- Makes representation smaller, more manageable and spatially invariant
- Operates over each activation map independently



How many **parameters**?

None!

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

Max **Pooling**

activation map

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

max pool with 2 x 2 filter
and stride of 2

6	8
3	4

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

Average **Pooling**

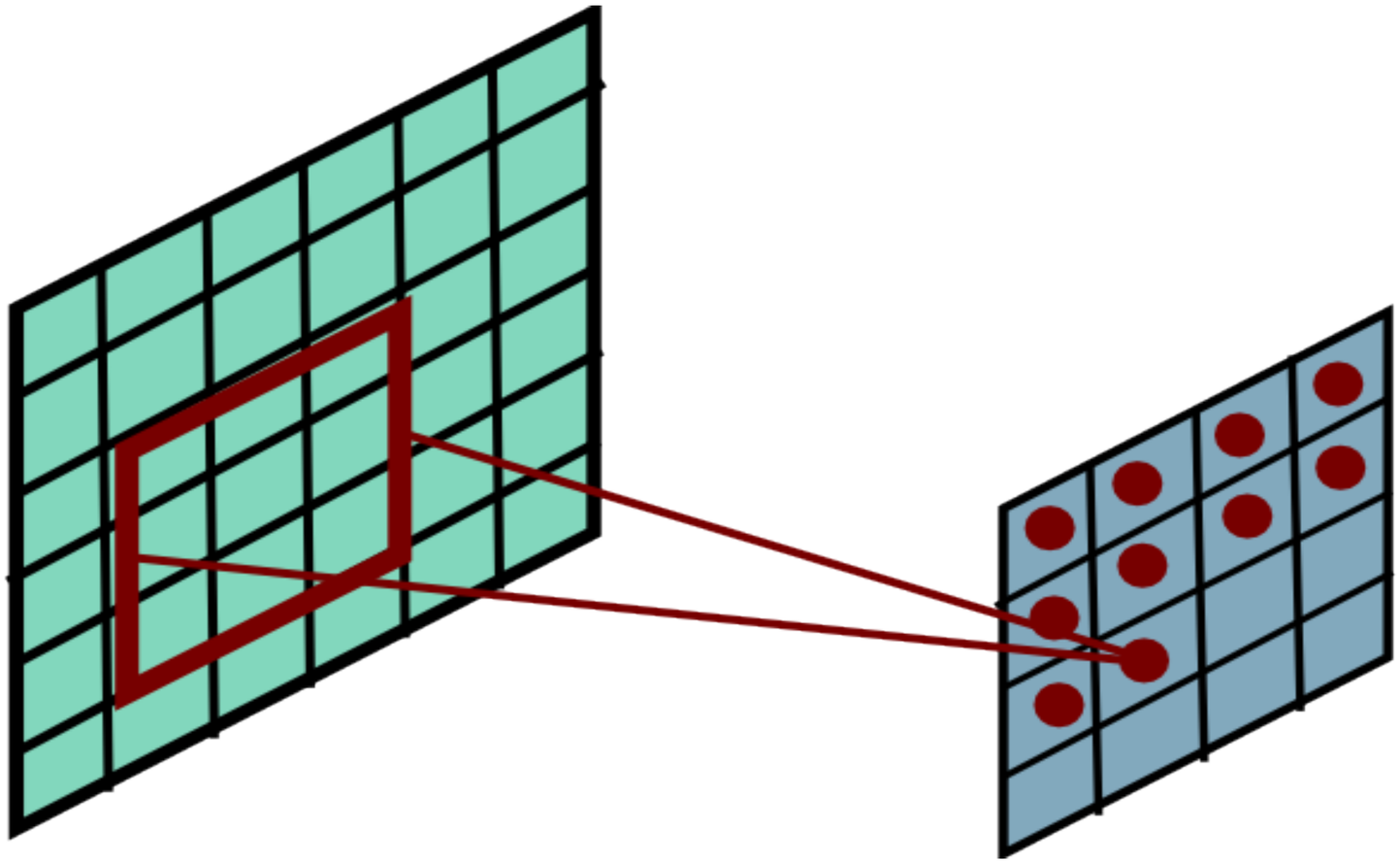
activation map

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

3.25	5.25
2	2

Pooling Layer **Receptive Field**

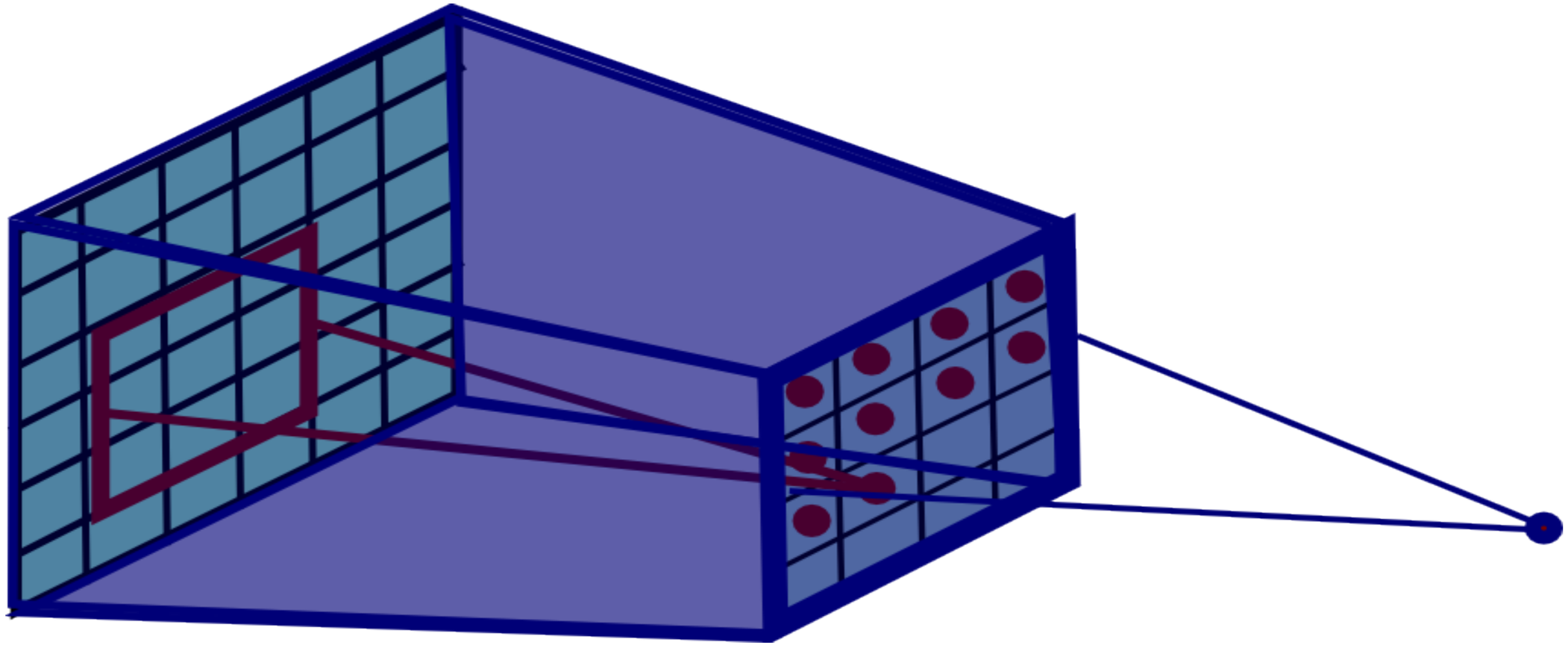
If convolutional filters have size KxK and stride 1, and pooling layer has pools of size PxP, then each unit in the pooling layer depends upon a patch (at the input of the preceding conv. layer) of size: **(P+K-1)x(P+K-1)**



* slide from Marc'Aurelio Renzato

Pooling Layer **Receptive Field**

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)**



* slide from Marc'Aurelio Renzato

Pooling Layer **Summary**

Accepts a volume of size: $W_i \times H_i \times D_i$

Requires hyperparameters:

— Spatial extent of filters: K

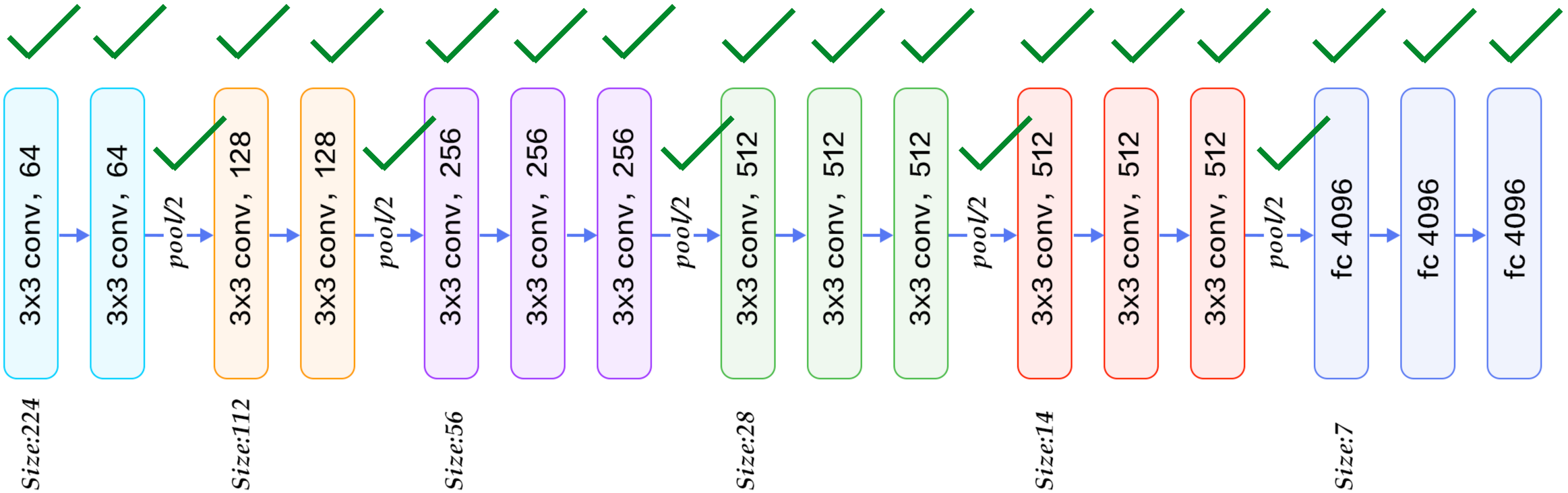
— Stride of application: F

Produces a volume of size: $W_o \times H_o \times D_o$

$$W_o = (W_i - F)/S + 1 \qquad H_o = (H_i - F)/S + 1 \qquad D_o = D_i$$

Number of total learnable parameters: 0

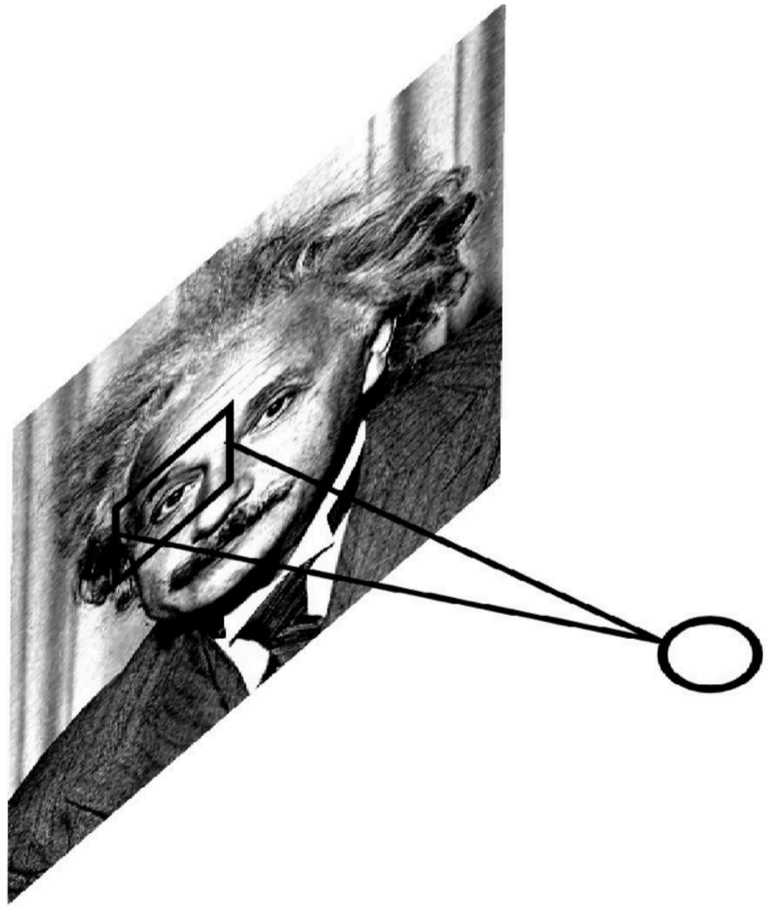
Convolutional Neural Networks

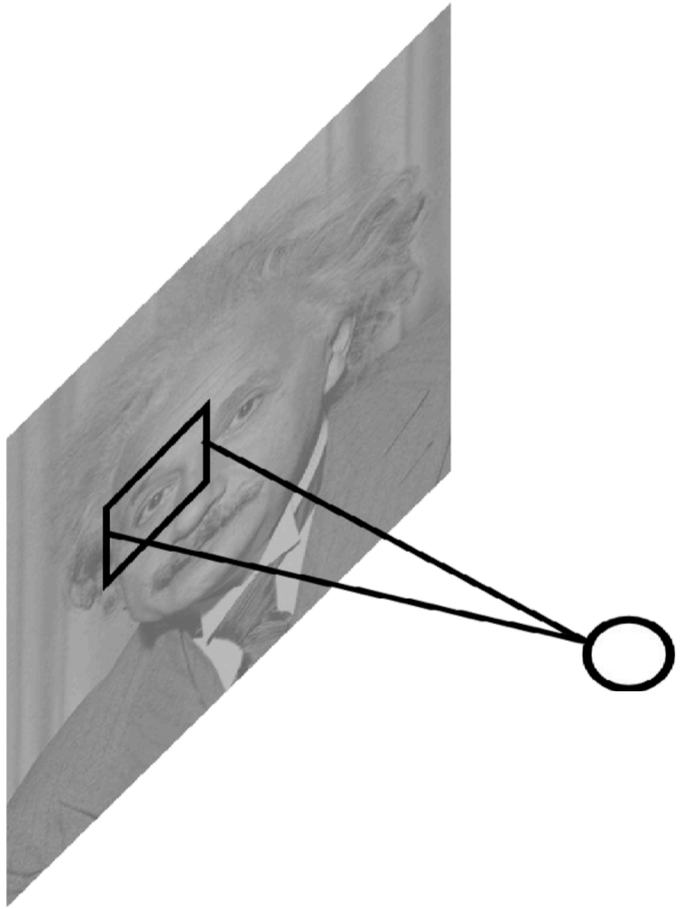


VGG-16 Network

Local **Contrast Normalization** Layer

ensures response is the same in both case (details omitted, no longer popular)





* images from Marc'Aurelio Renzato