CS 4803 / 7643: Deep Learning

Topics:

Convolutional Neural Networks (Part 1)

Peter Anderson

Georgia Tech

Admin

- PS1 & HW1 are out!
  - Due **Friday Feb 15 11:55pm**

Recall: Linear Classifier

$$\underset{10\text{x}1}{\boxed{f(x,W)}} = \underset{10\text{x}3072}{\boxed{W}}\,\overset{3072\text{x}1}{\boxed{x}} + \underset{10\text{x}1}{\boxed{b}}$$

Image

Array of **32x32x3** numbers
(3072 numbers total)

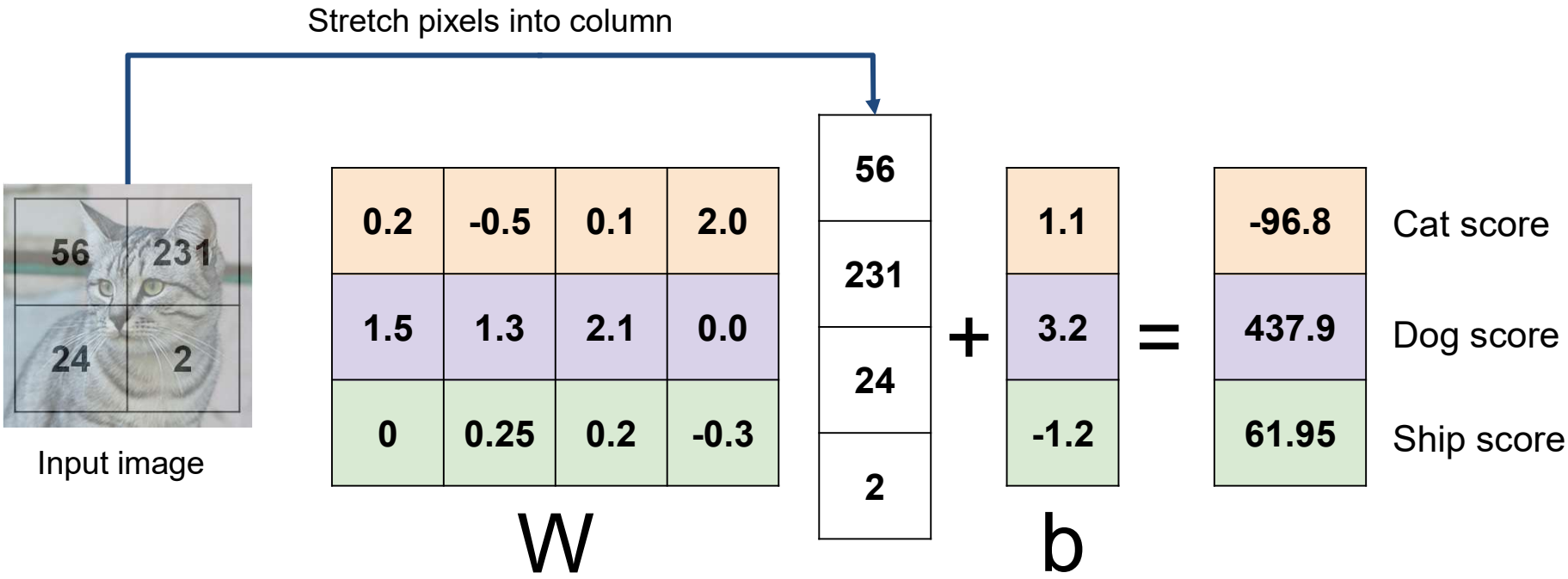
$f(\mathbf{x},\mathbf{W})$

10 numbers giving
class scores

W

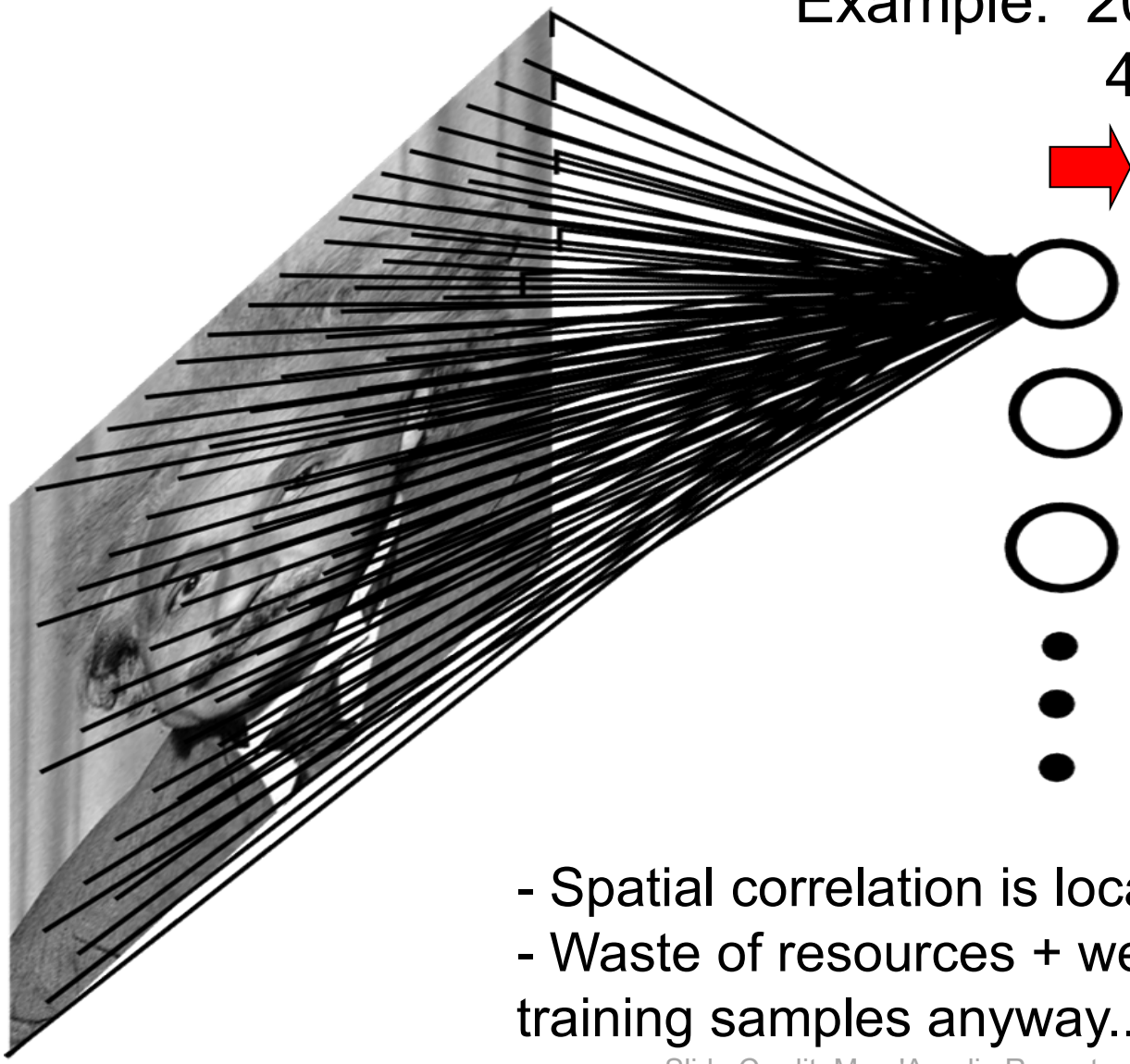
parameters
or weights

Example with an image with 4 pixels, and 3 classes (cat/dog/ship)



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Fully Connected Layer

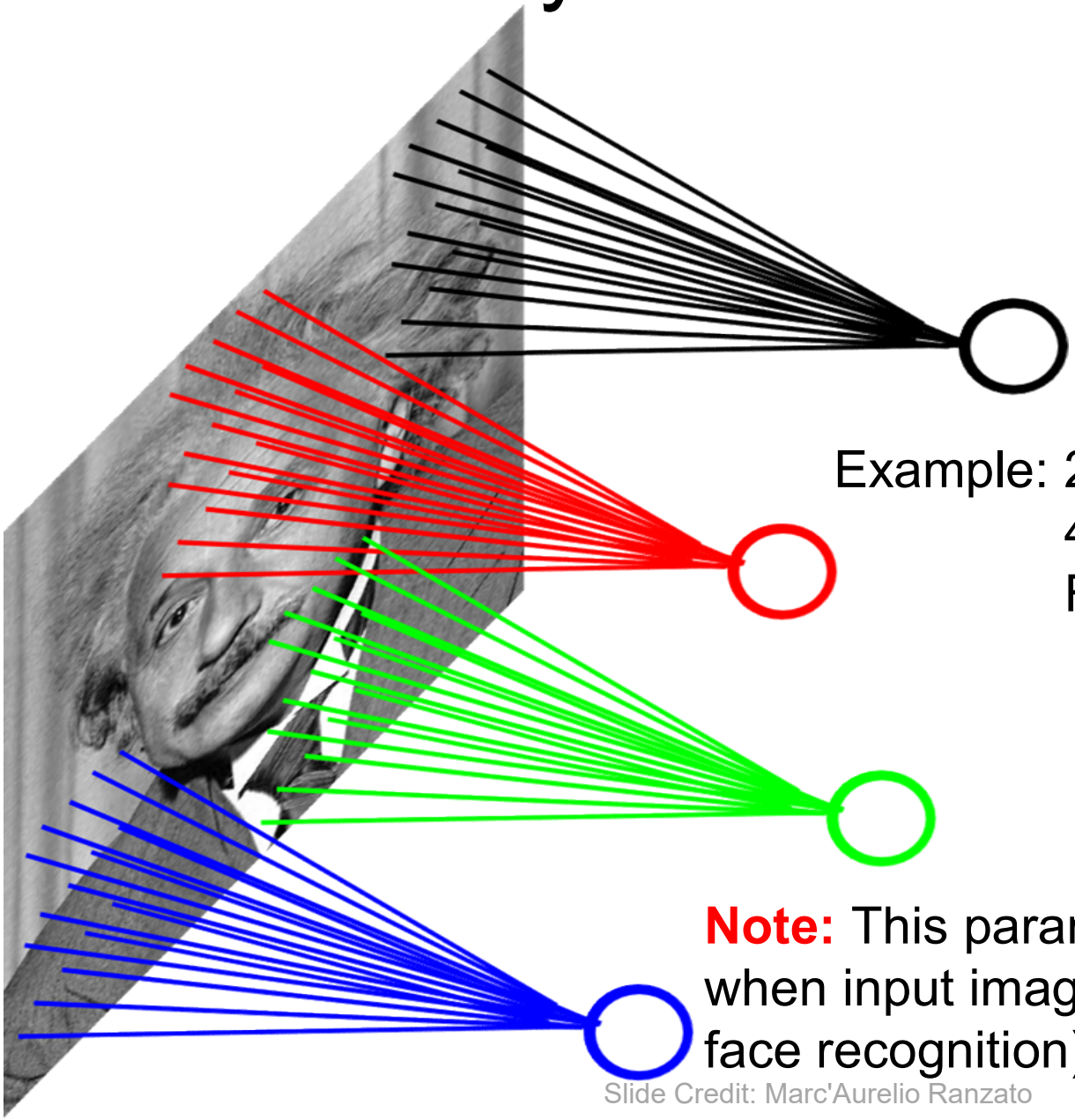


Example: 200x200 image
40K hidden units
~2B parameters!!!

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

Slide Credit: Marc'Aurelio Ranzato

Locally Connected Layer

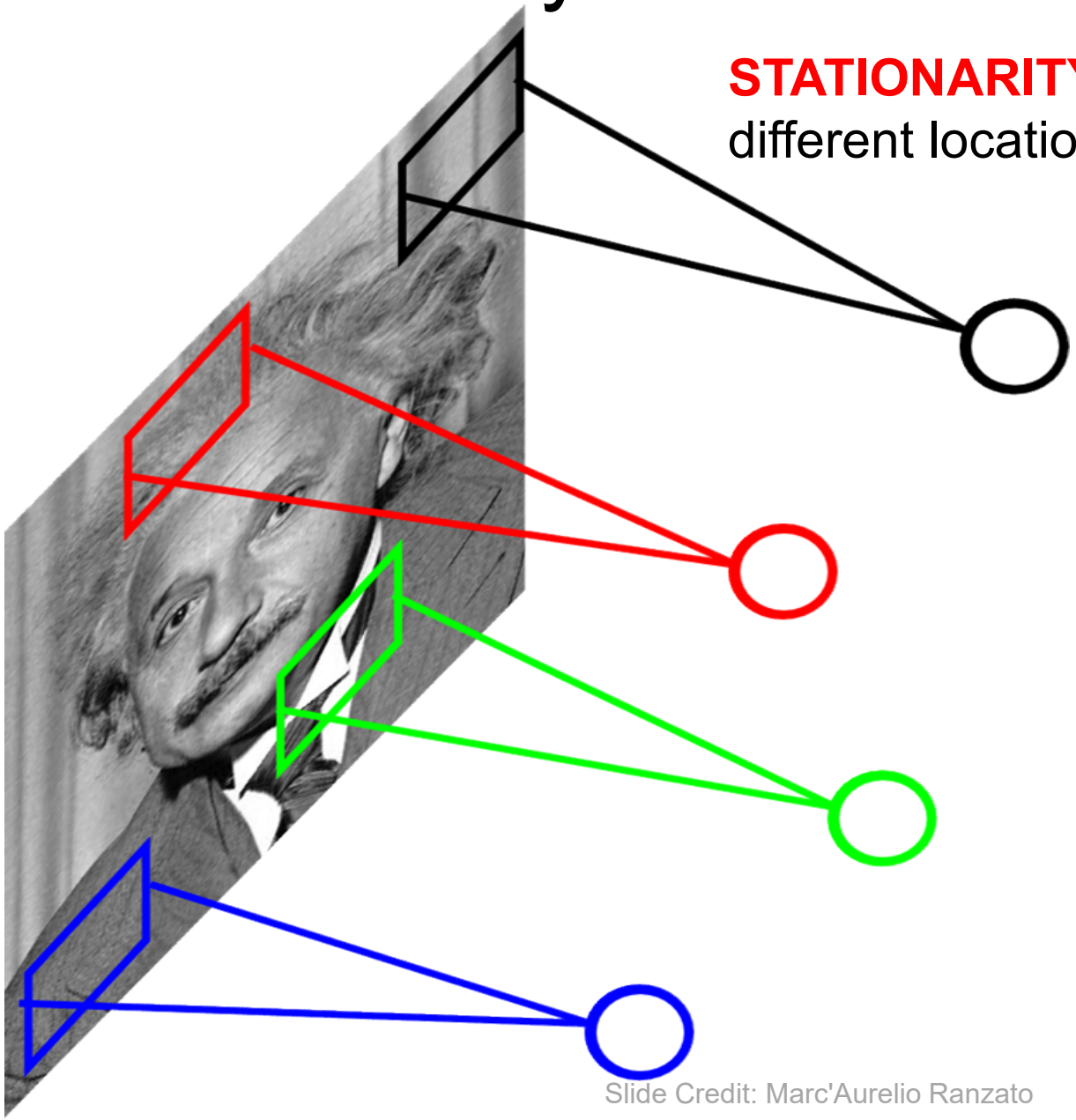


Example: 200x200 image
40K hidden units
Filter size: 10x10
4M parameters

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

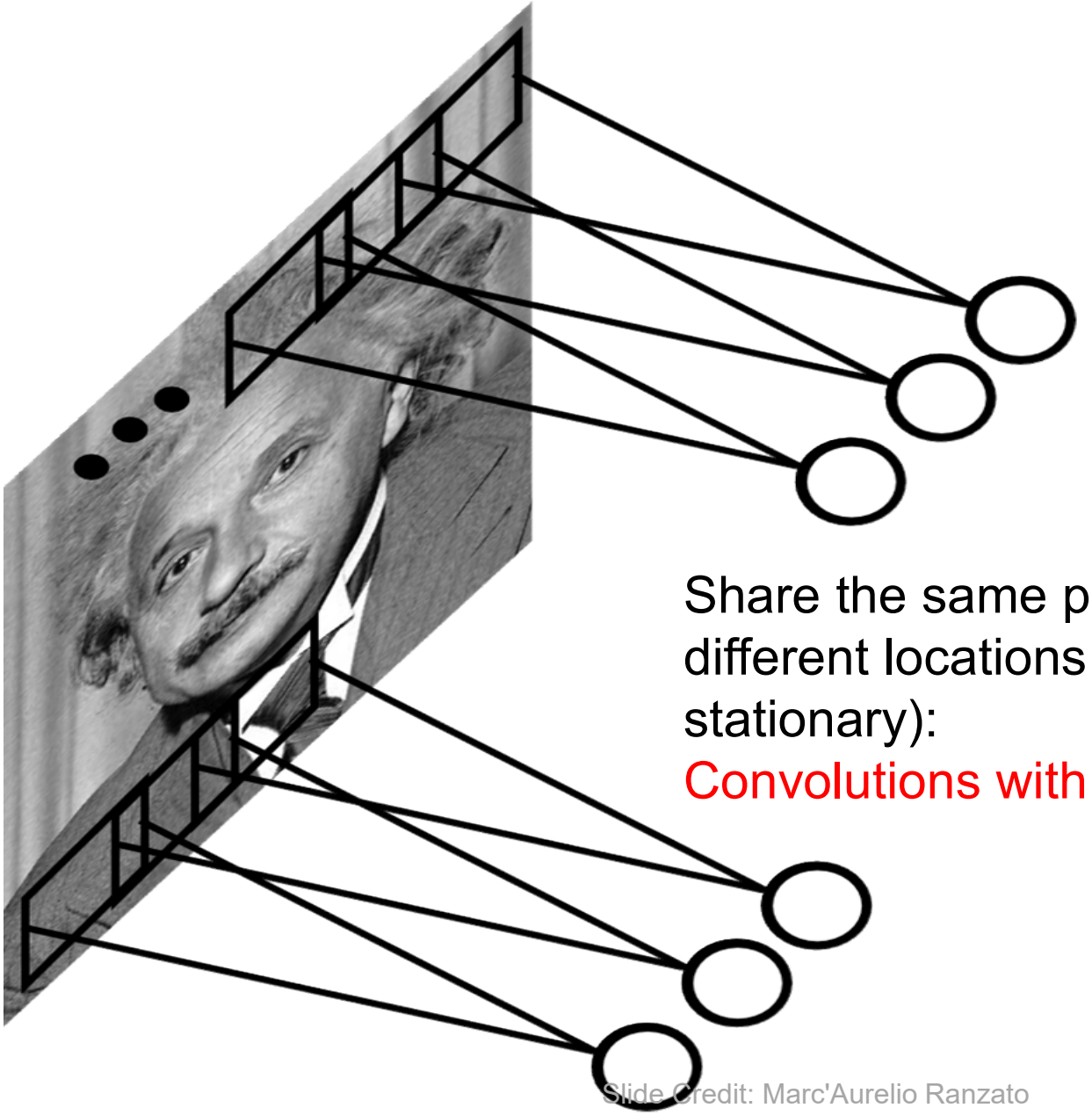
Slide Credit: Marc'Aurelio Ranzato

Locally Connected Layer



STATIONARITY? Statistics is similar at different locations

Convolutional Layer



Share the same parameters across different locations (assuming input is stationary):
Convolutions with learned kernels

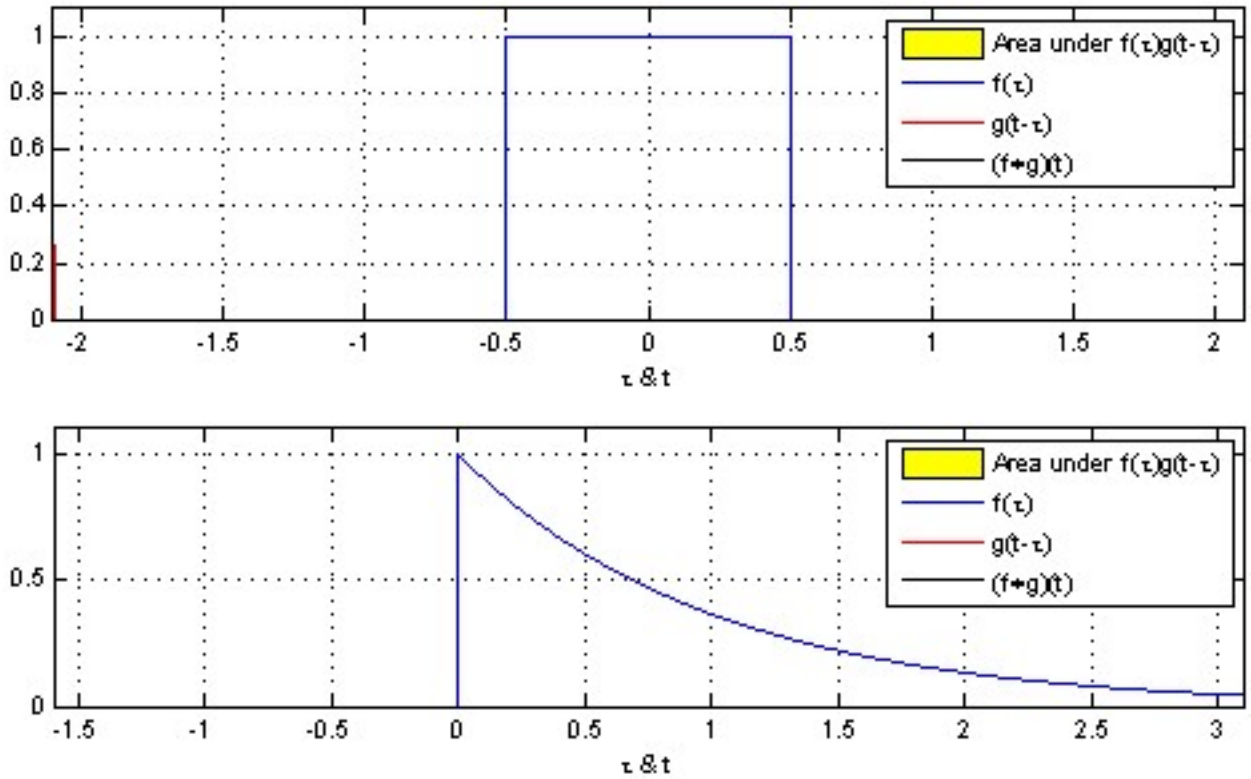
Convolutions!

- Math vs. CS vs. programming viewpoints

Convolutions for mathematicians

- On operation on two functions x and w to produce a third function y
- E.g. input $x(t)$ and kernel or weighting function $w(\tau)$

$$
\begin{aligned}
y(t) &= (x * w)(t) \\
&= (w * x)(t) \\
&= \int_{\tau=-\infty}^{\infty} x(t-\tau)w(\tau)d\tau \\
&= \int_{\tau=-\infty}^{\infty} x(\tau)w(t-\tau)d\tau
\end{aligned}
$$



Convolutions for mathematicians

- One dimension

$$y(t) = (x * w)(t)$$
$$= \int_{\tau=-\infty}^{\infty} x(t-\tau)w(\tau)d\tau$$

- Two dimensions

$$y(t_1, t_2) = (x * w)(t_1, t_2)$$
$$= \int_{\tau=-\infty}^{\infty} \int_{\sigma=-\infty}^{\infty} x(t_1 - \tau, t_2 - \sigma)w(\tau, \sigma)\, d\tau d\sigma$$

Convolutions for computer scientists

- Discrete data, e.g. a $W \times H$ image I and a $K \times K$ kernel ω

$$\mathrm{S}(i,j) = (I * \omega)(i,j)$$

$$= \sum_{m=0}^{H} \sum_{n=0}^{W} I(m,n)\omega(i-m, j-n)$$

Convolutions for programmers

$$\mathsf{S}(i,j) = (I * \omega)(i,j)$$

$$= \sum_{m=0}^{H} \sum_{n=0}^{W} I(m,n)\omega(i-m, j-n)$$

- Iterate over the kernel instead of the image

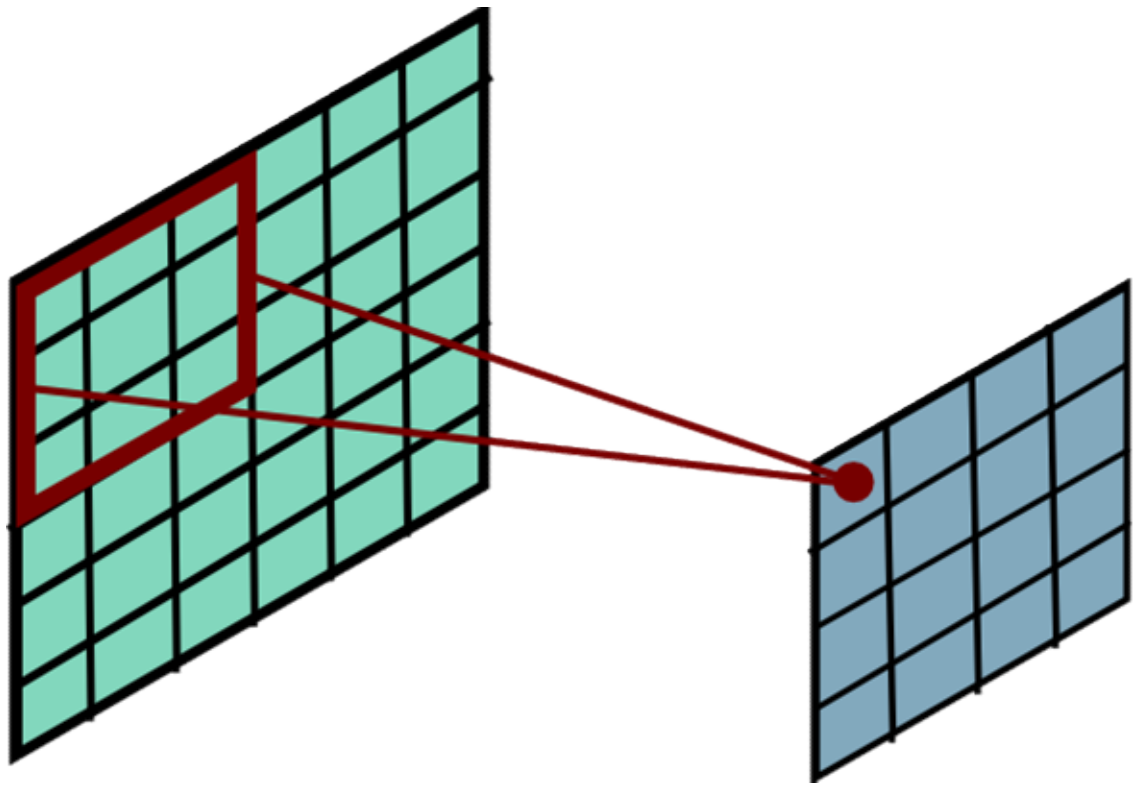
$$= \sum_{m=0}^{K} \sum_{n=0}^{K} I(i-m, j-n)\omega(m,n)$$

- Implement cross-correlation instead of convolution

$$= \sum_{m=0}^{K} \sum_{n=0}^{K} I(i+m, j+n)\omega(m,n)$$

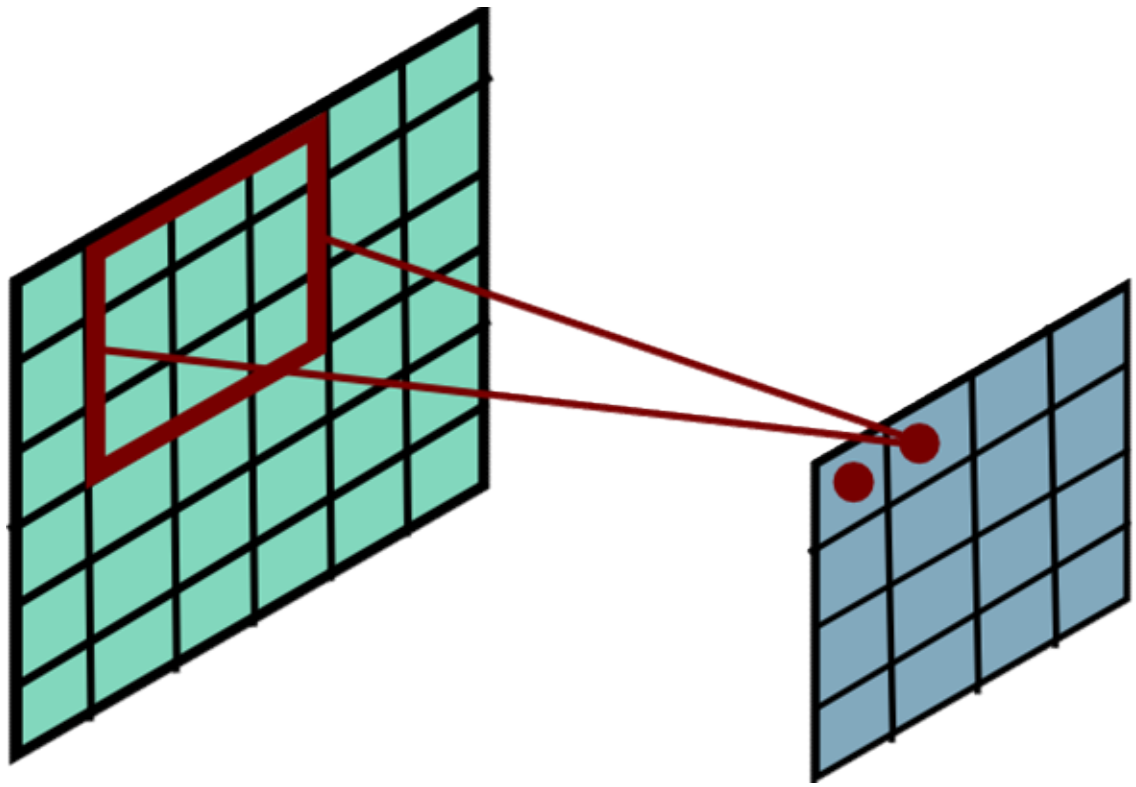
- Later - implementation as matrix multiplication

Convolutional Layer



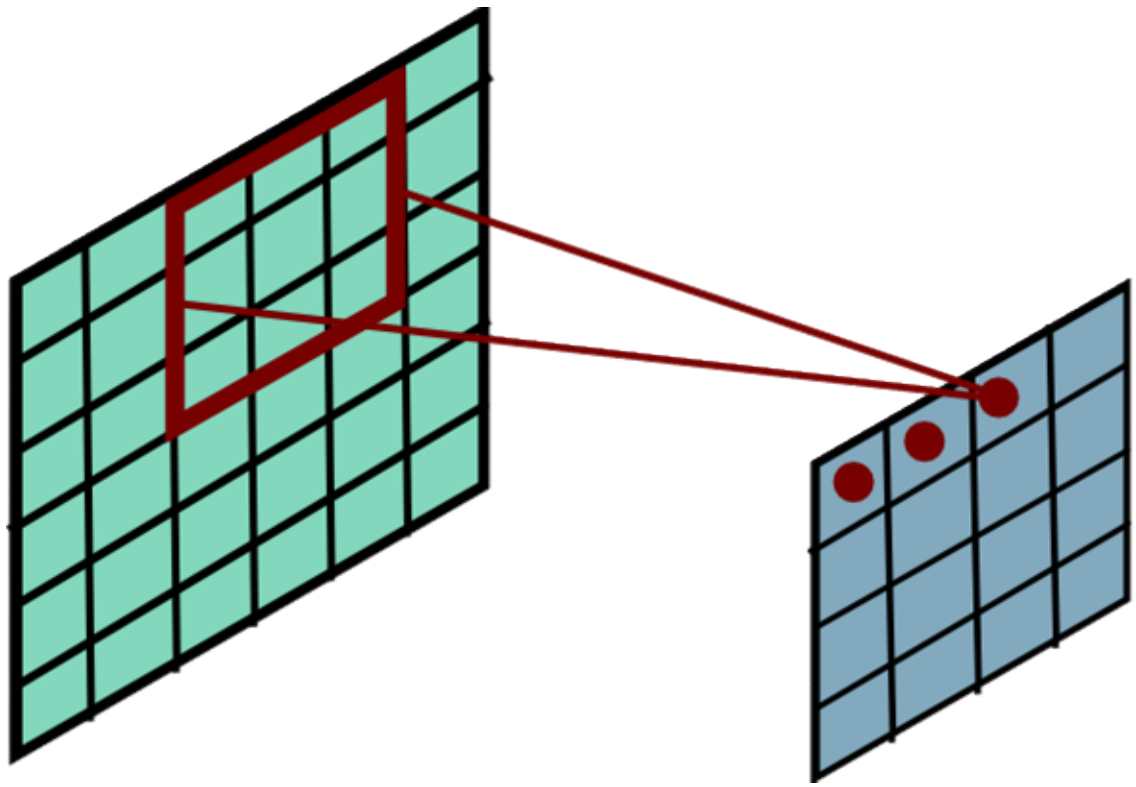
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



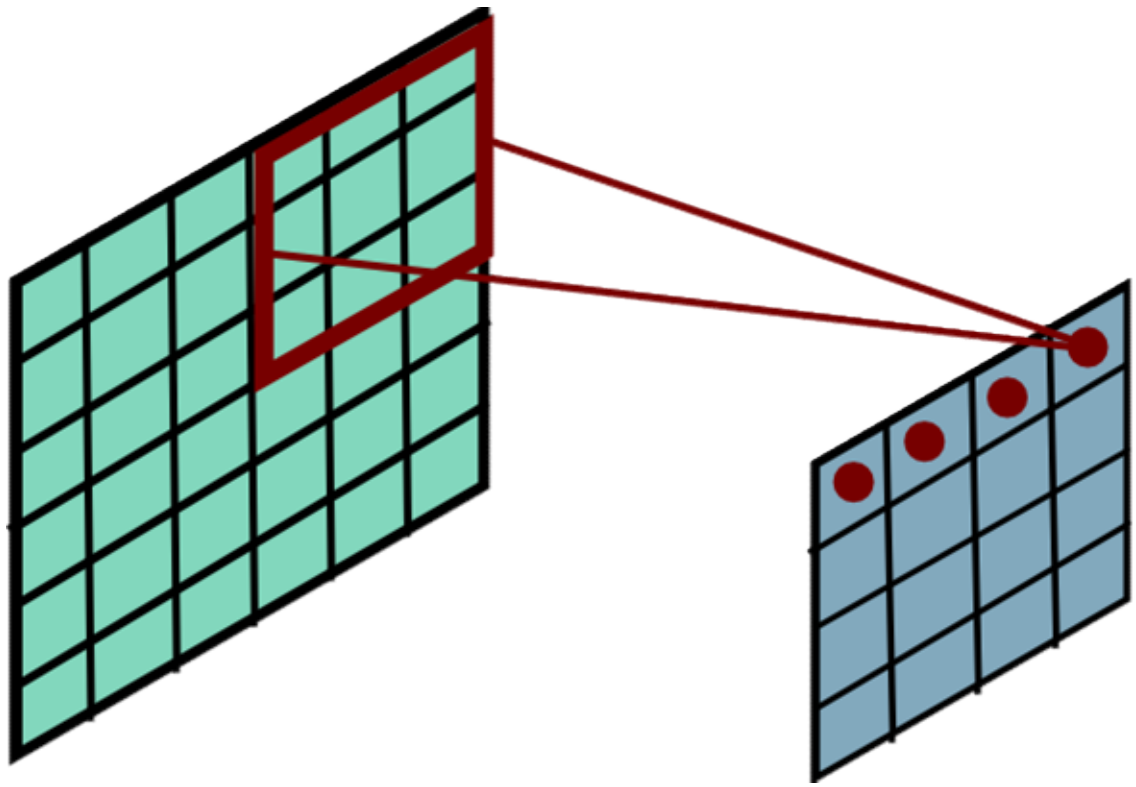
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



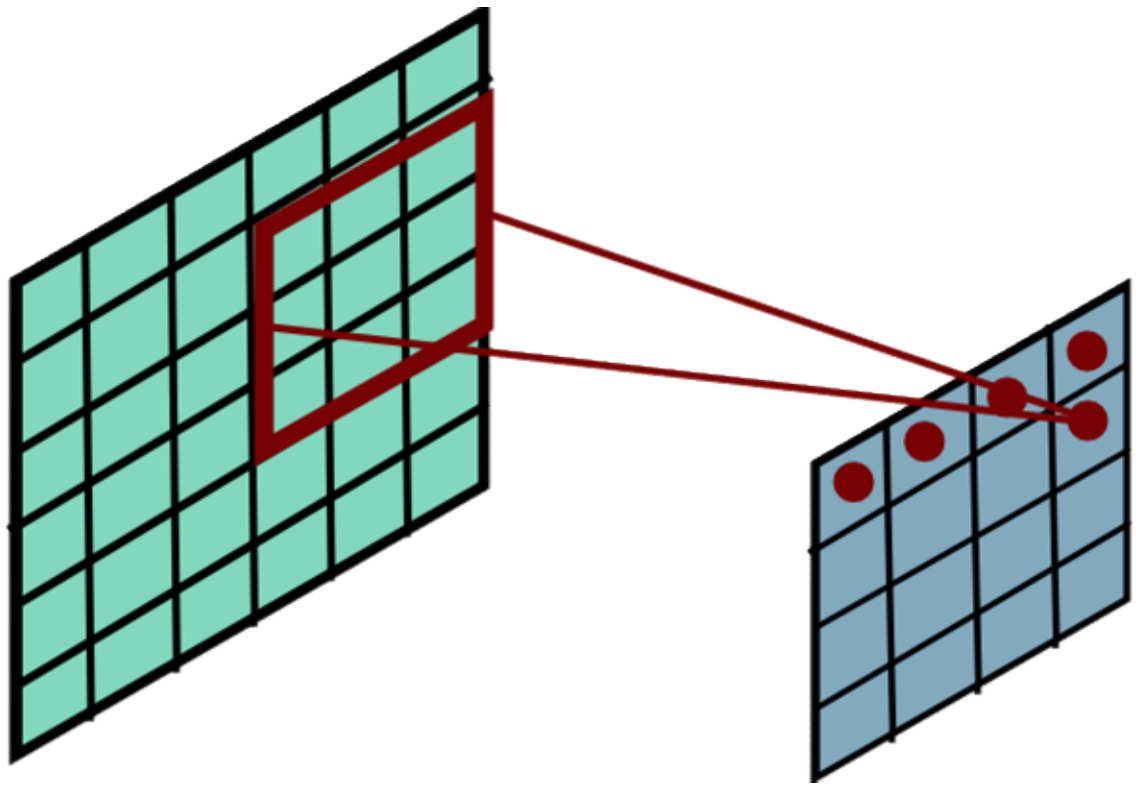
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



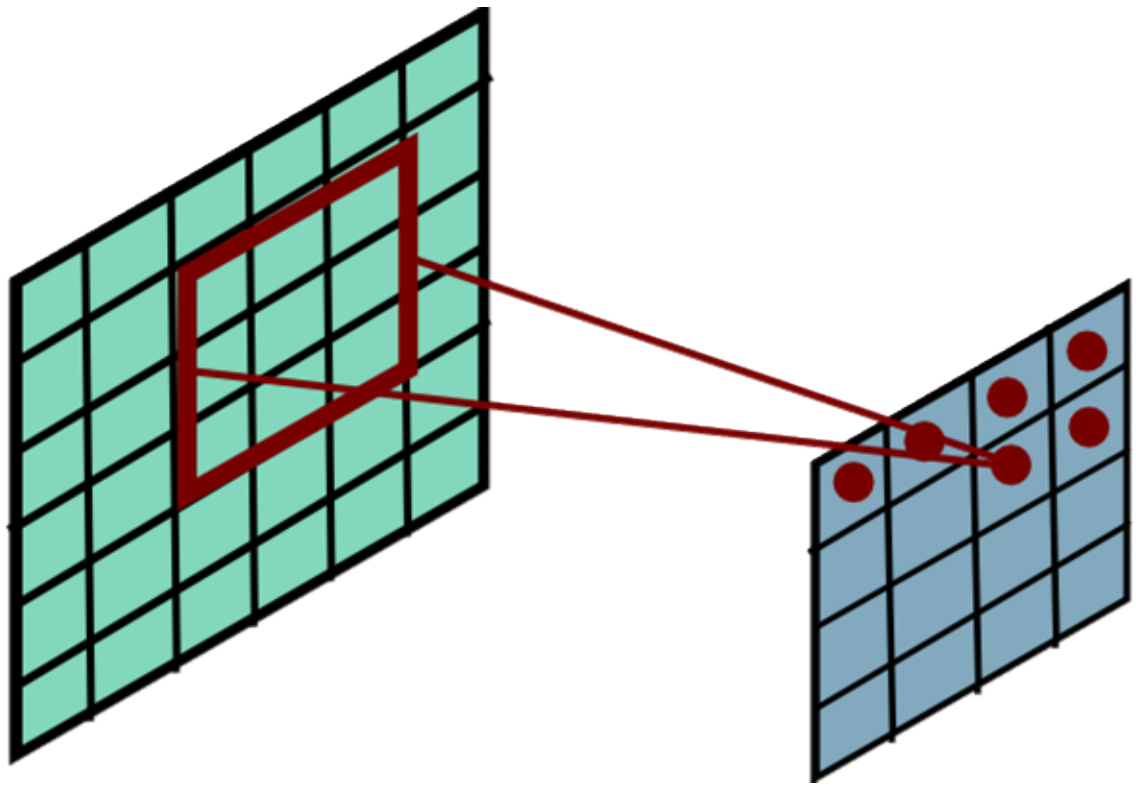
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



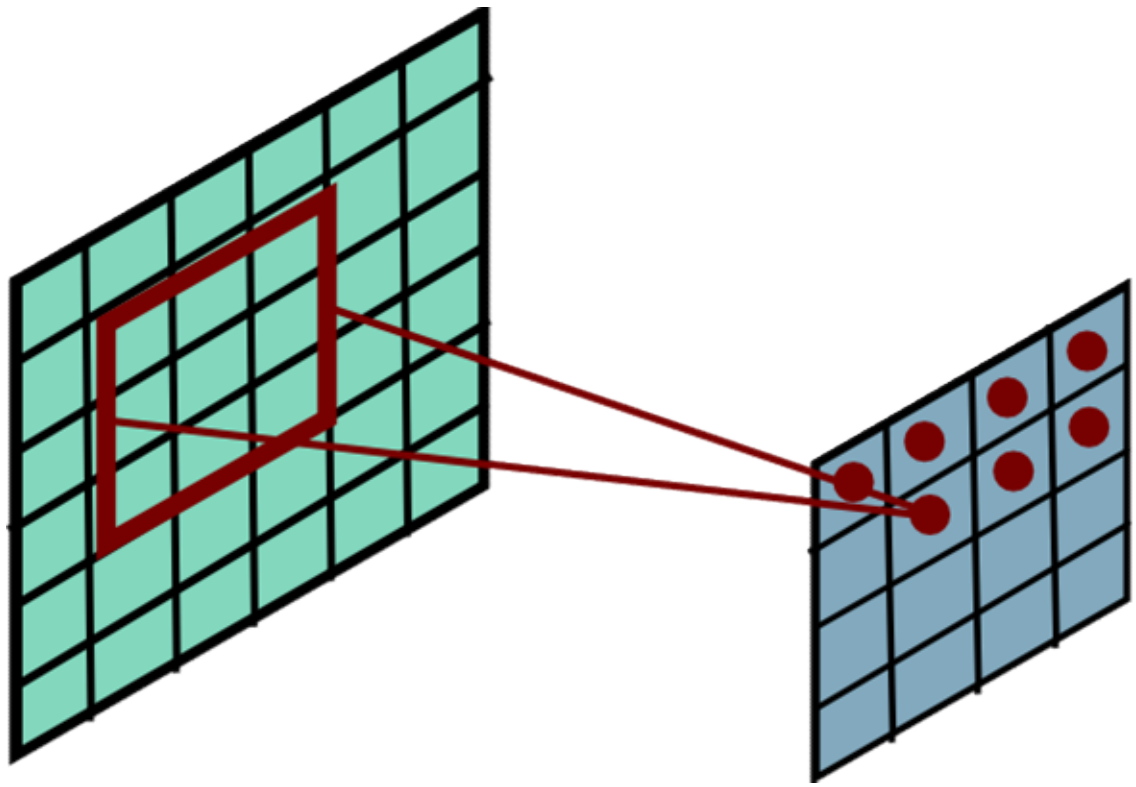
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



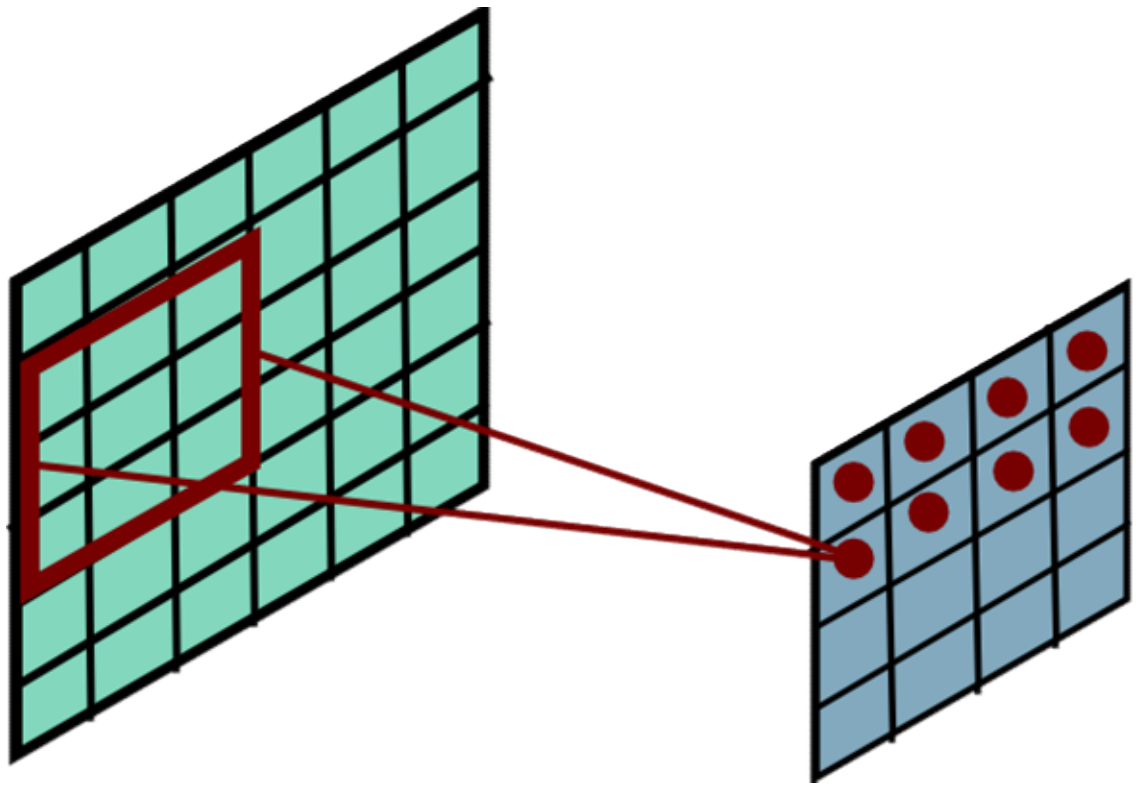
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



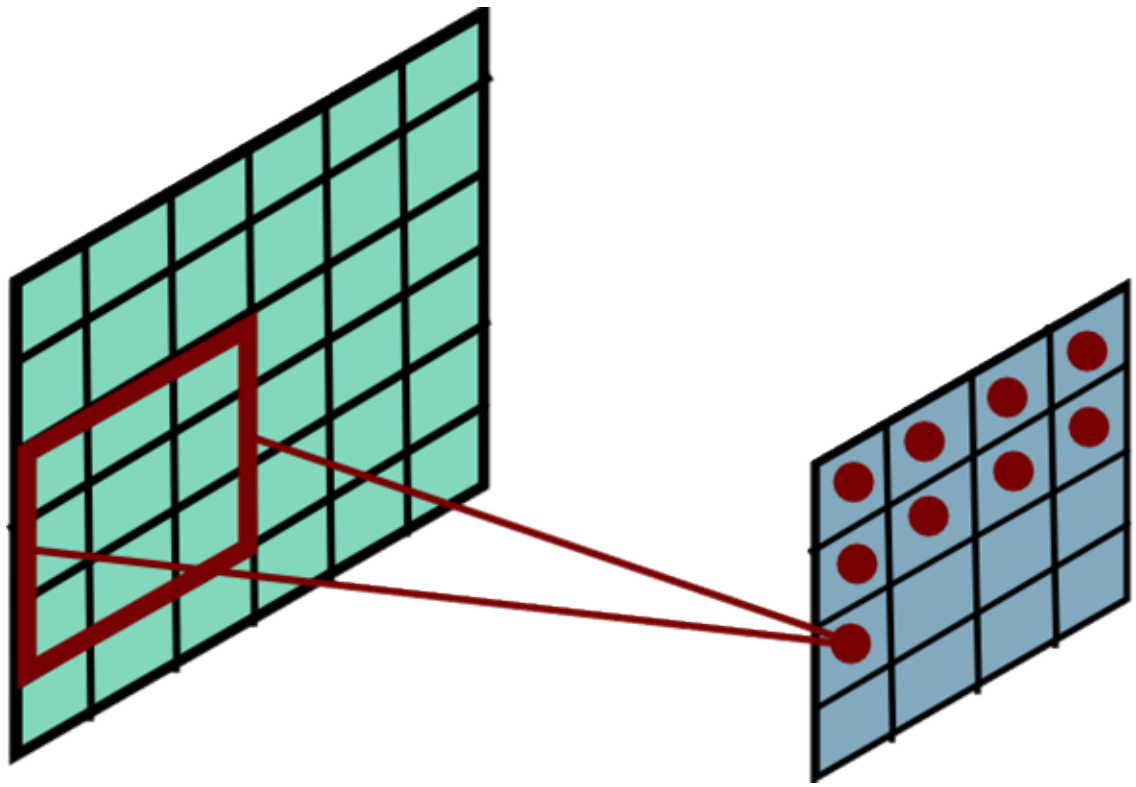
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



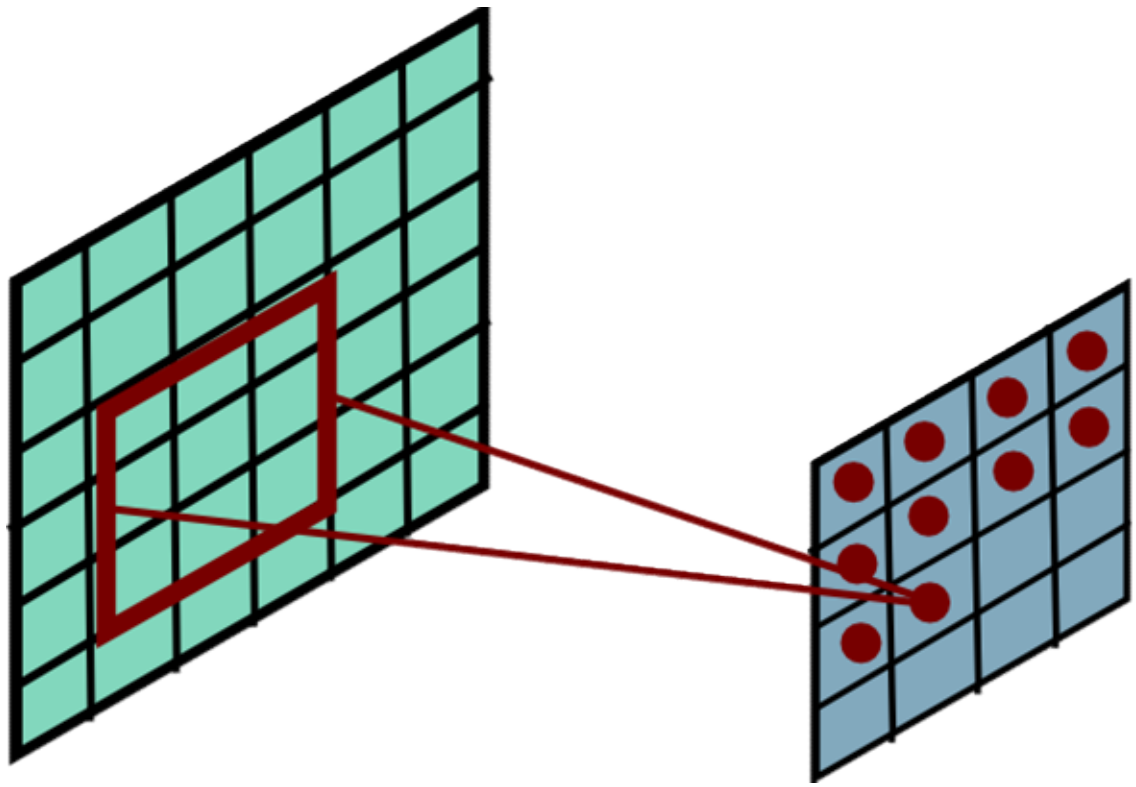
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



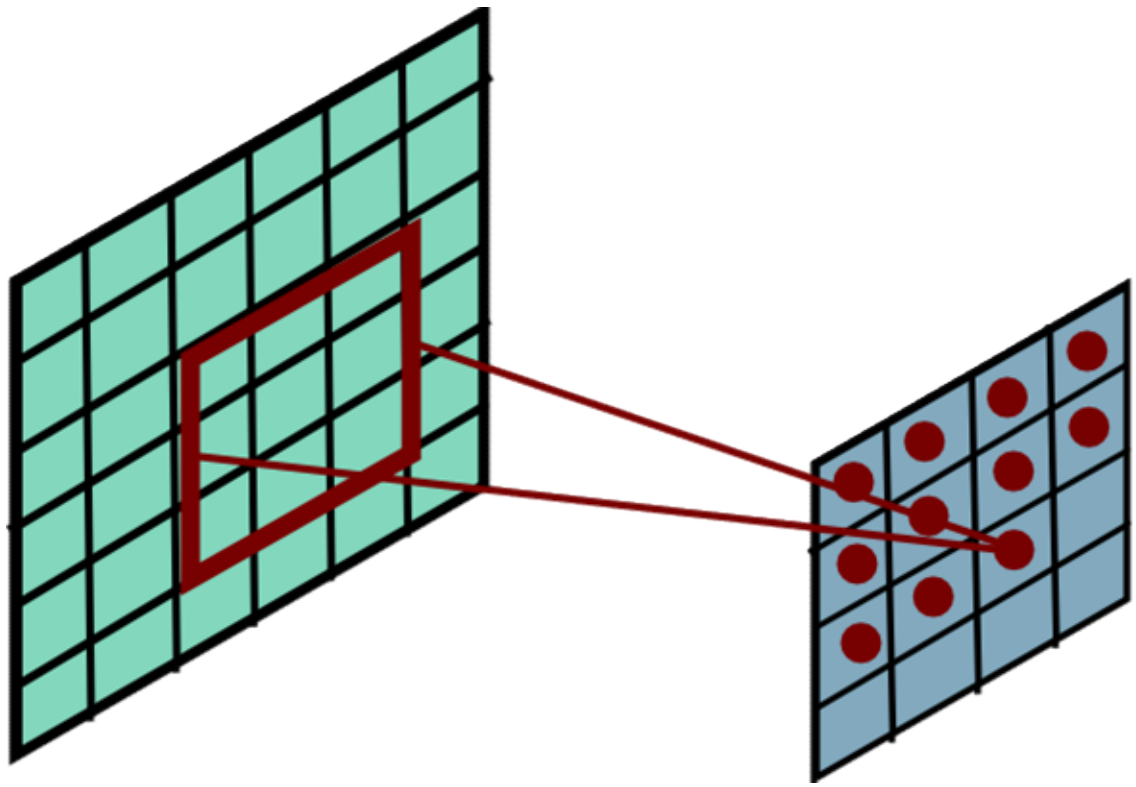
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



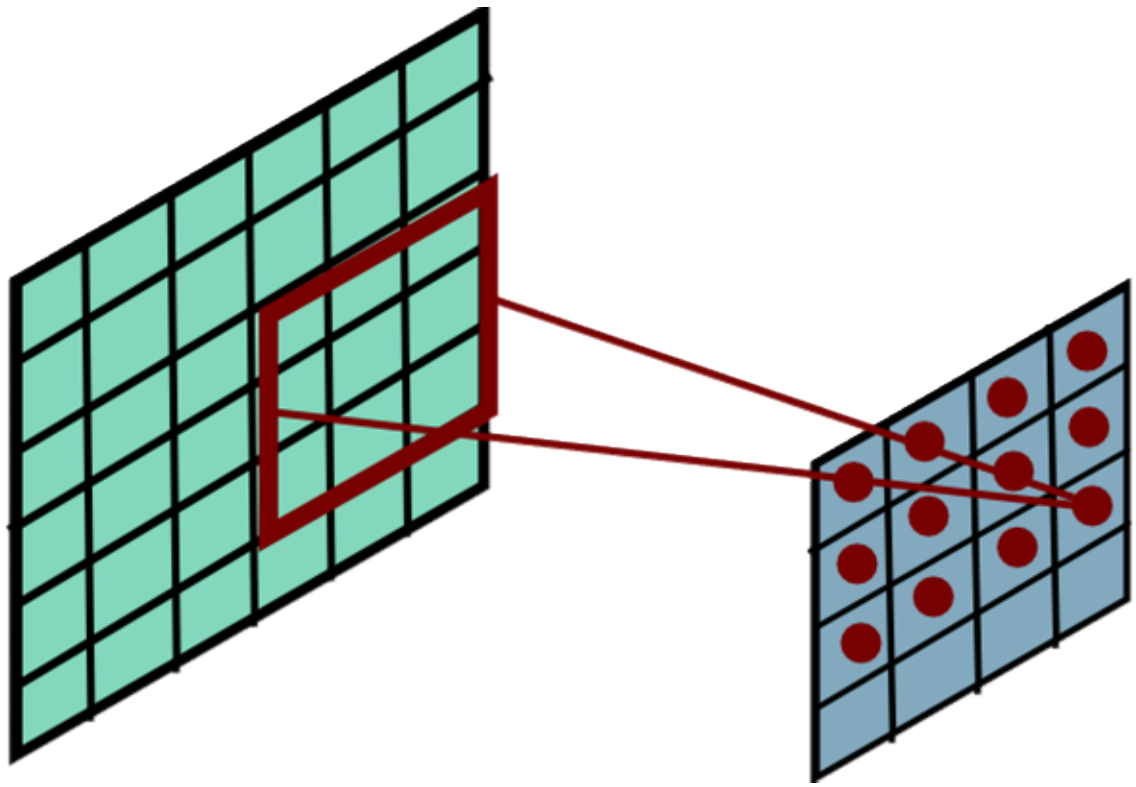
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



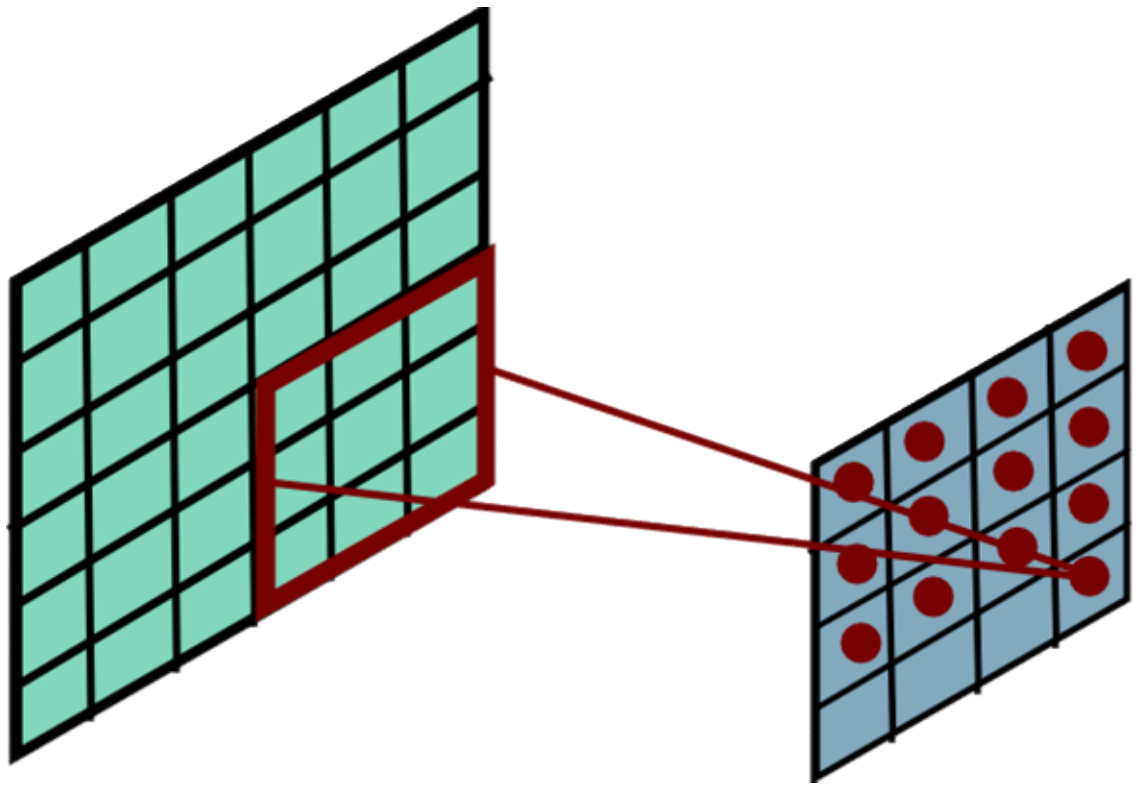
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



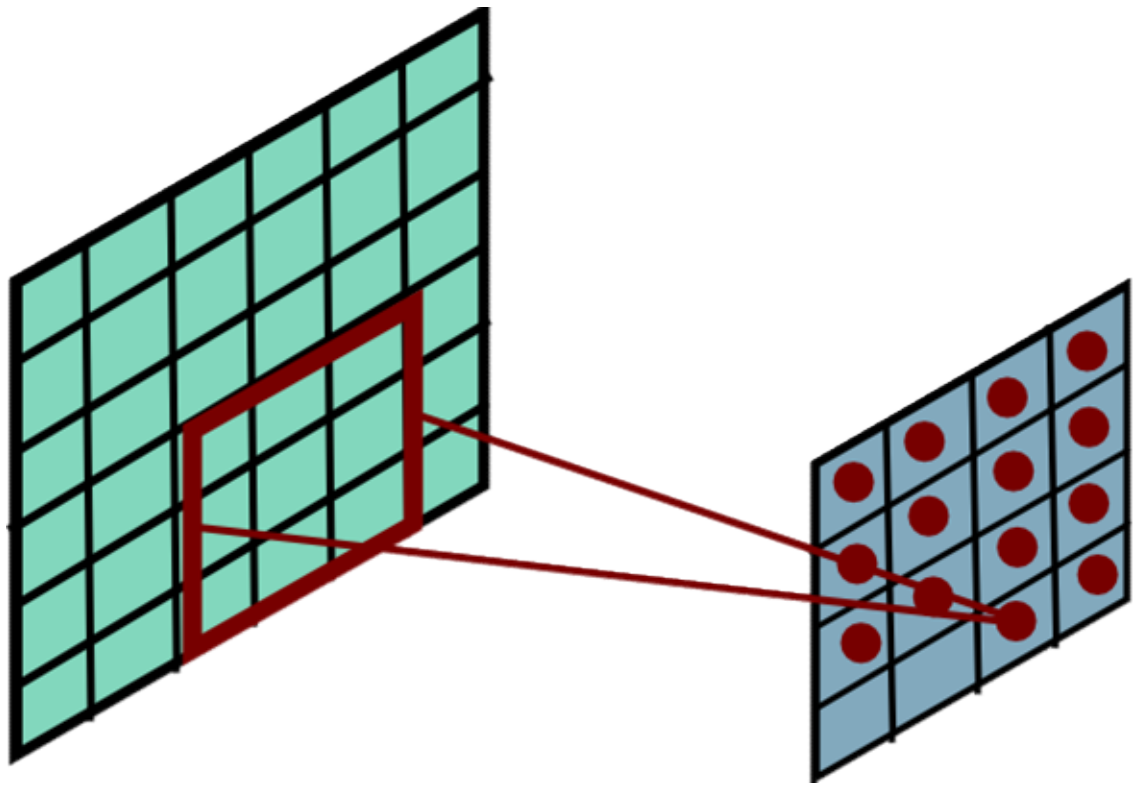
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



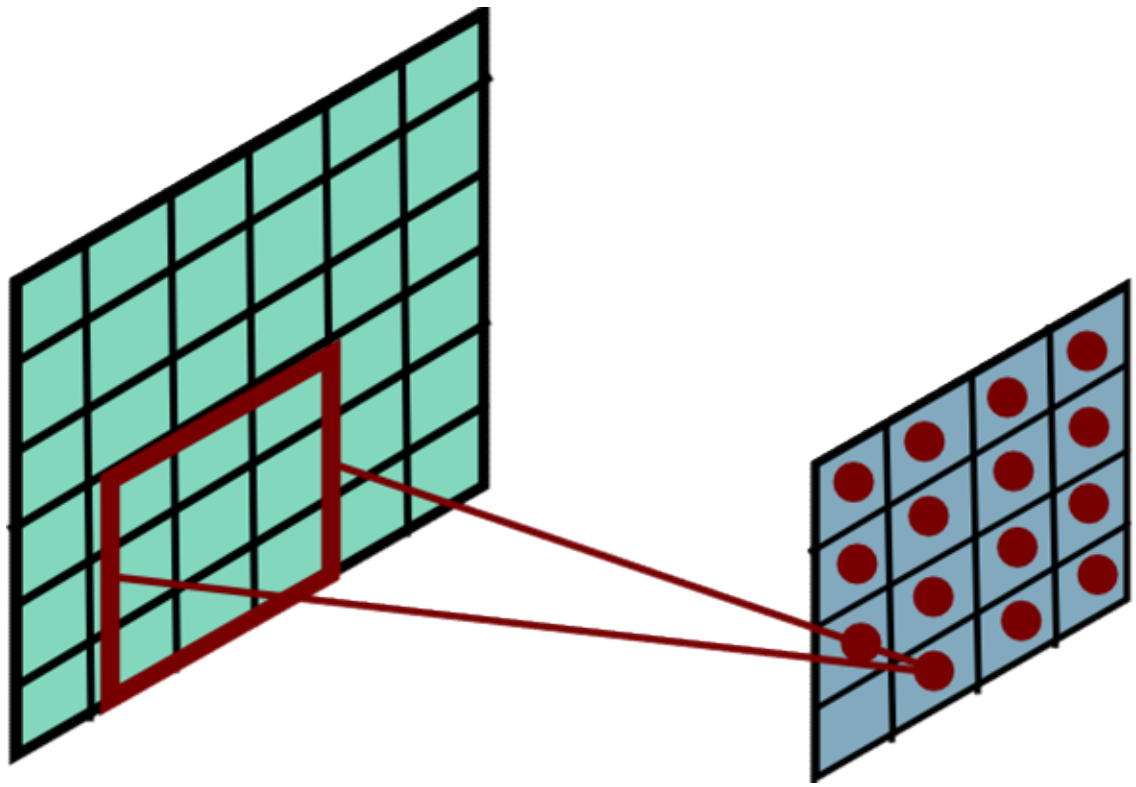
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



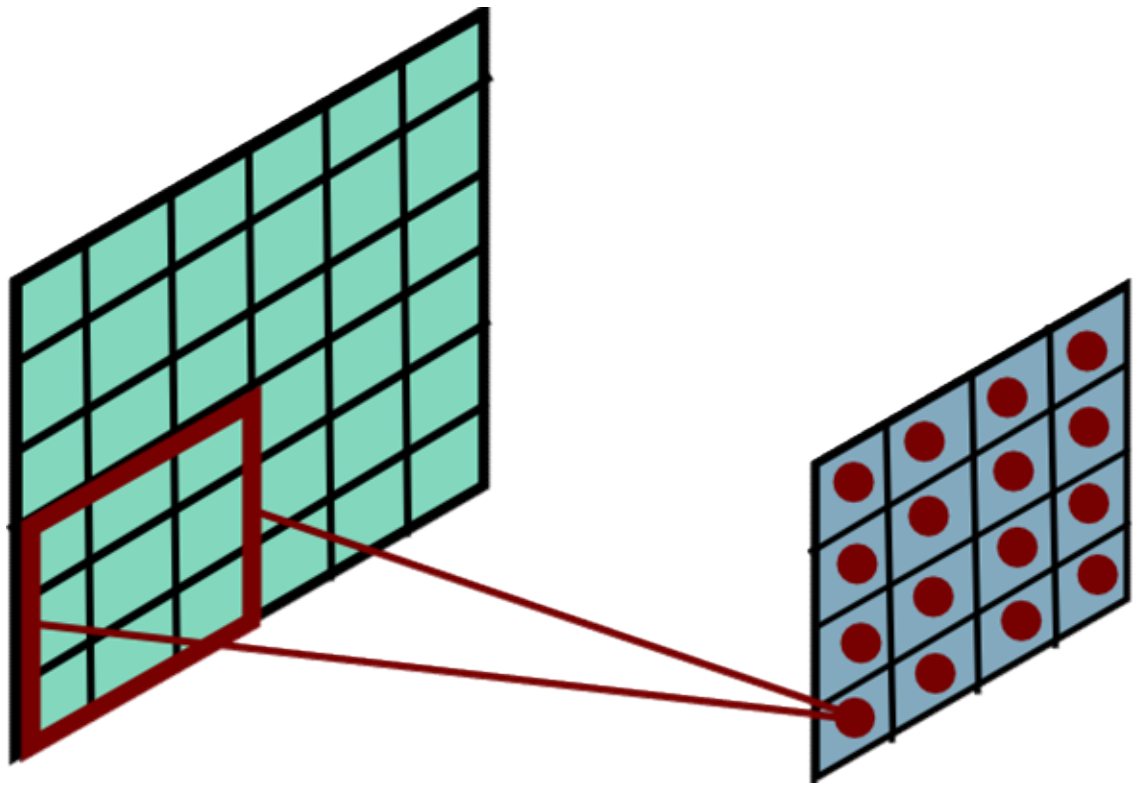
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



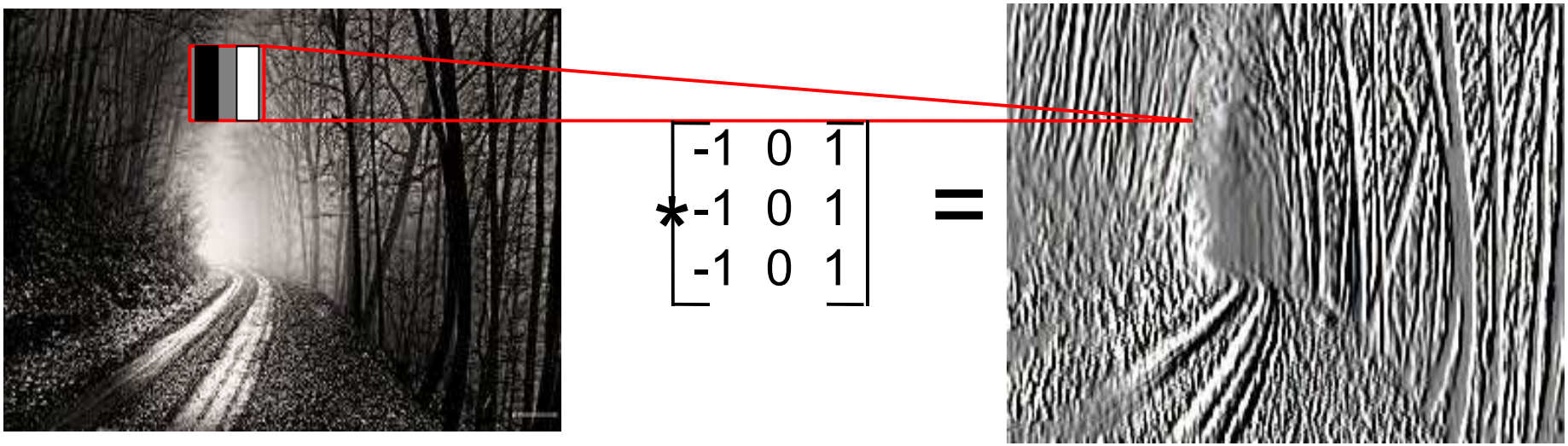
Mathieu et al. “Fast training of CNNs through FFTs” ICLR 2014

Slide Credit: Marc'Aurelio Ranzato

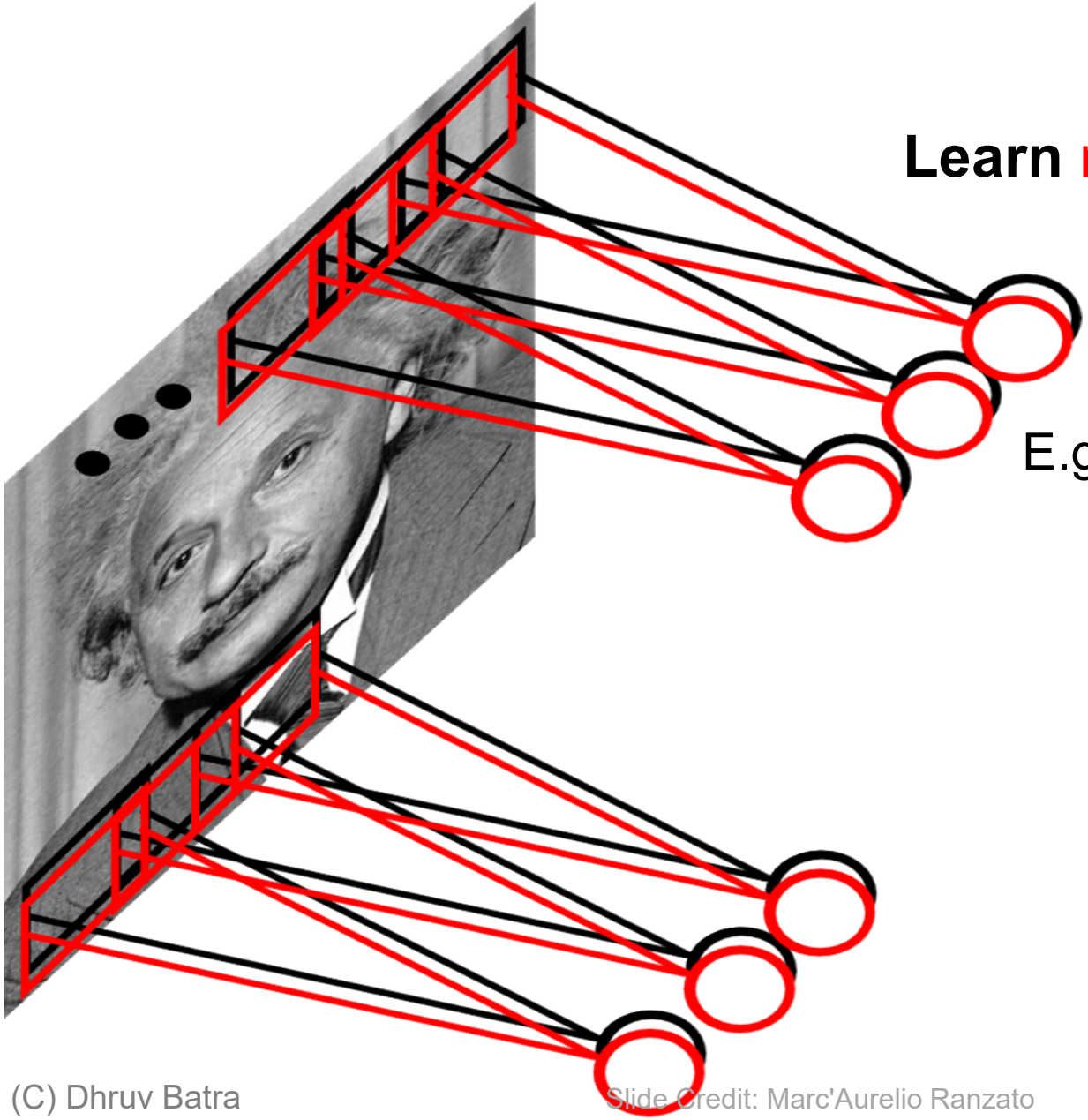
Convolution Explained

- http://setosa.io/ev/image-kernels/
- https://github.com/bruckner/deepViz

Convolutional Layer



Convolutional Layer

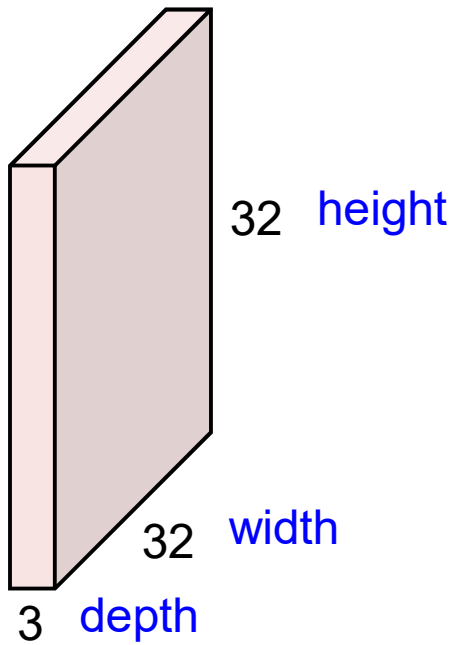


Learn multiple filters.

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

Convolution Layer

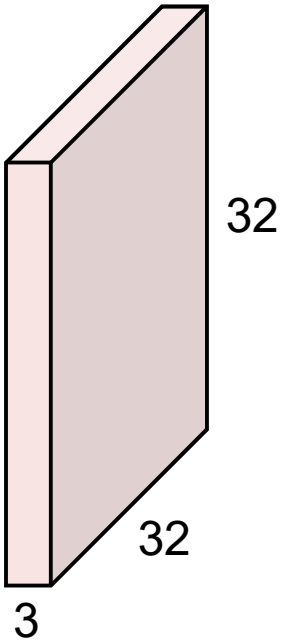
32x32x3 image -> preserve spatial structure



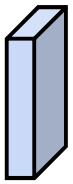
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Convolution Layer

32x32x3 image



5x5x3 filter

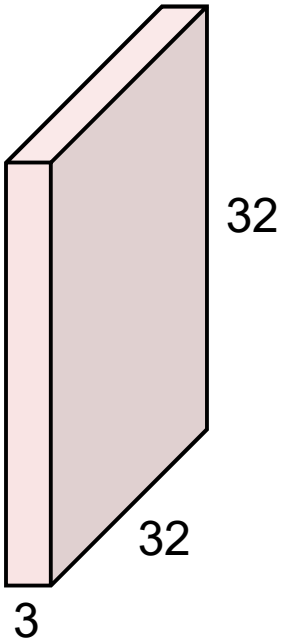


Convolve the filter with the image
i.e. “slide over the image spatially,
computing dot products”

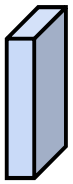
Convolution Layer

Filters always extend the full depth of the input volume

32x32x3 image

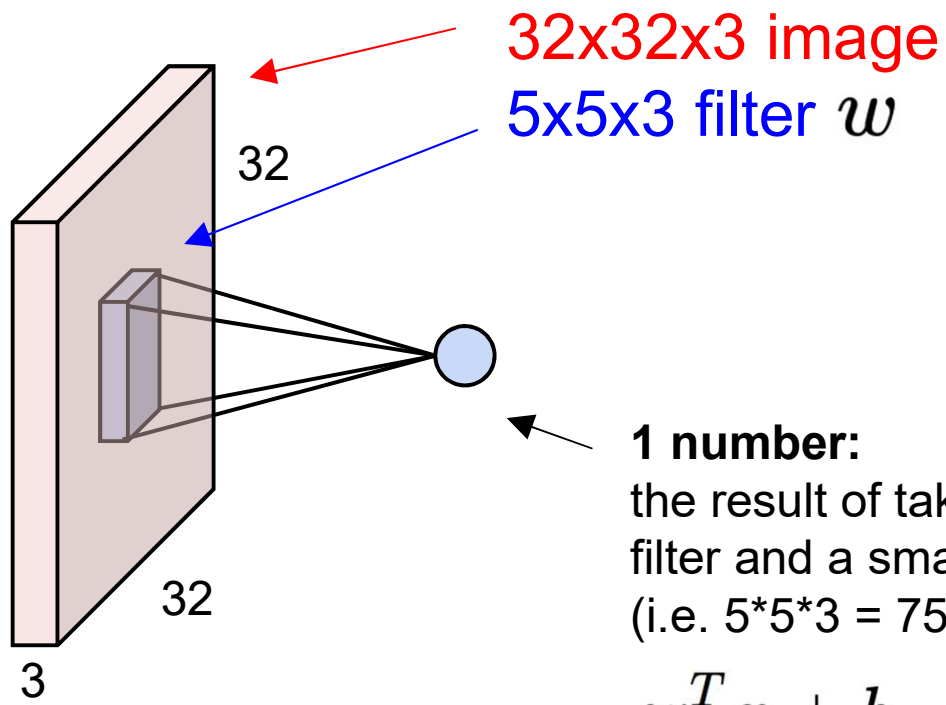


5x5x3 filter



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

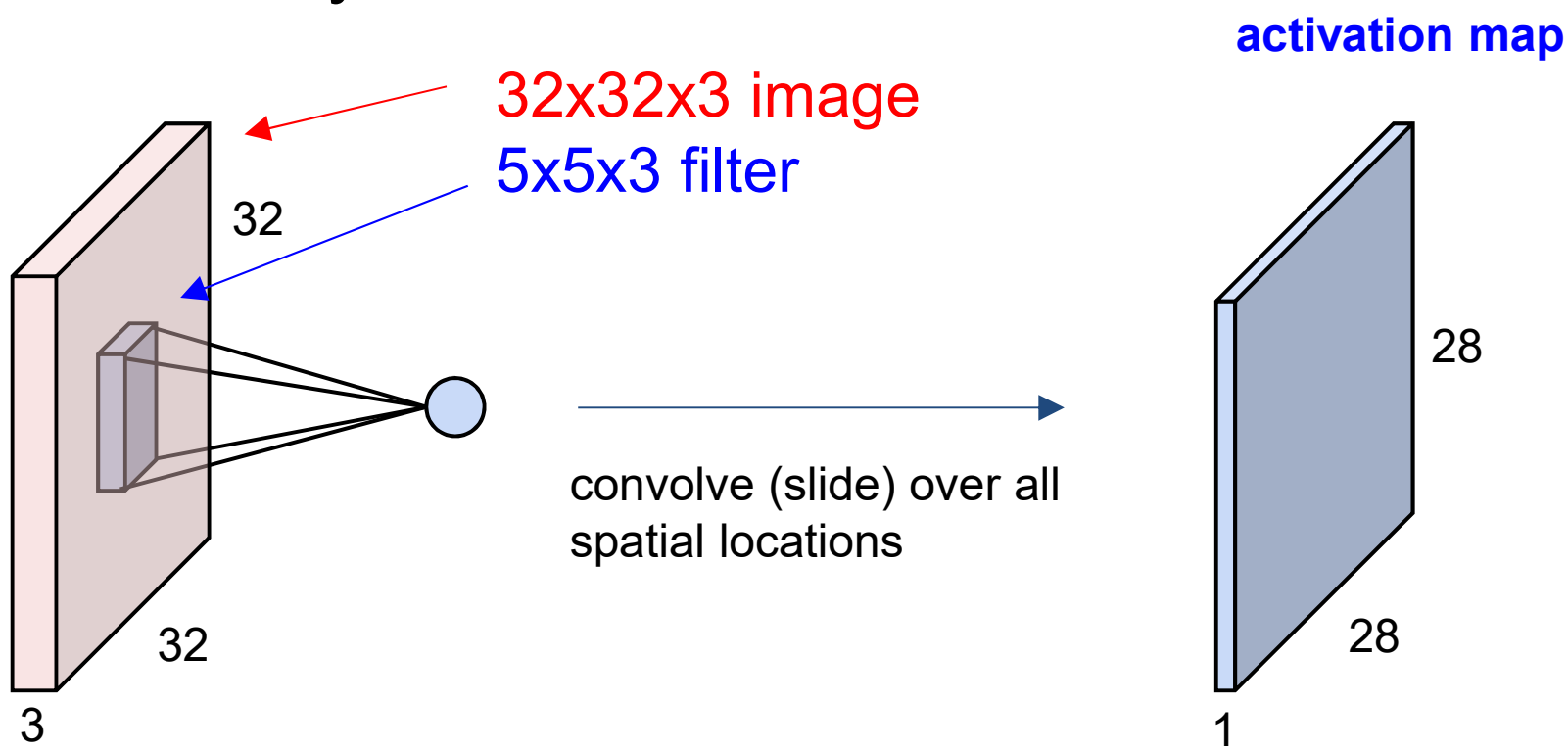
Convolution Layer



1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

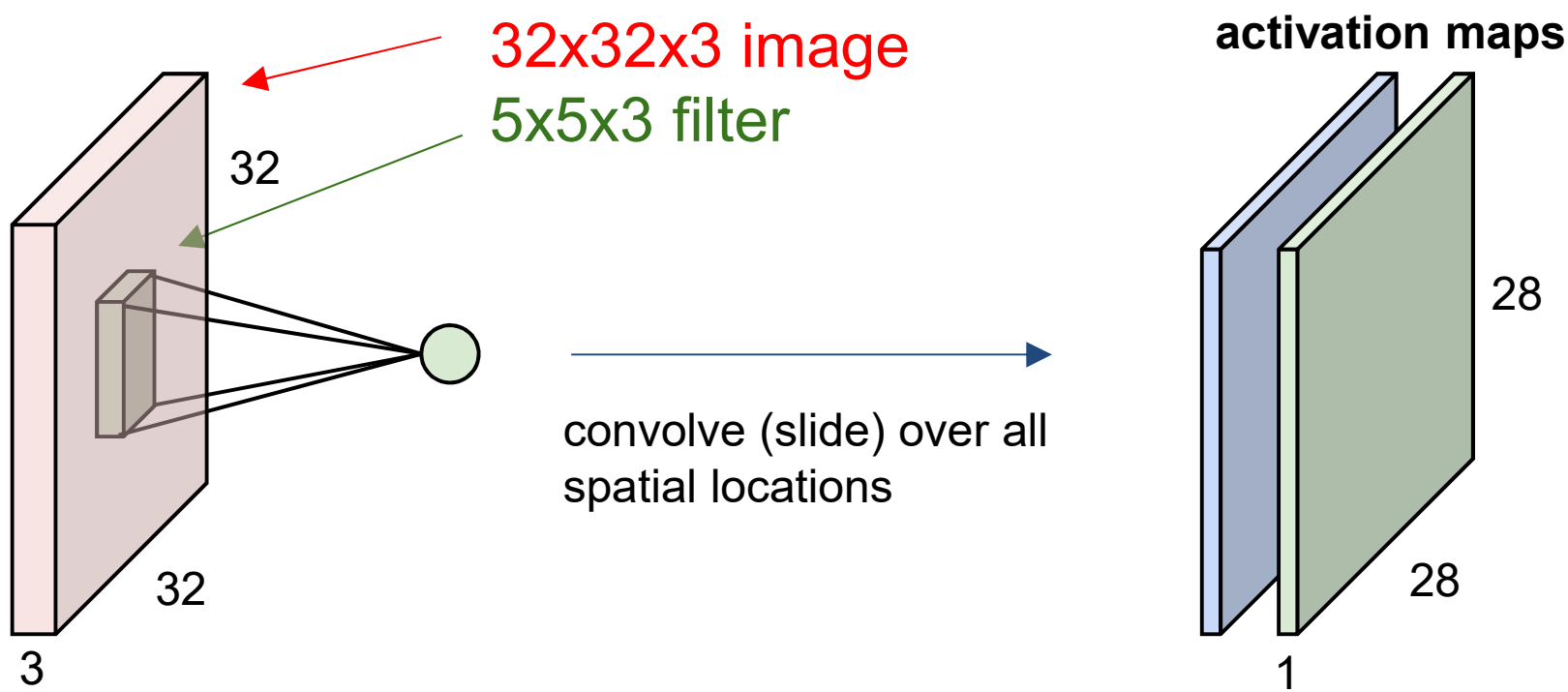
Convolution Layer



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

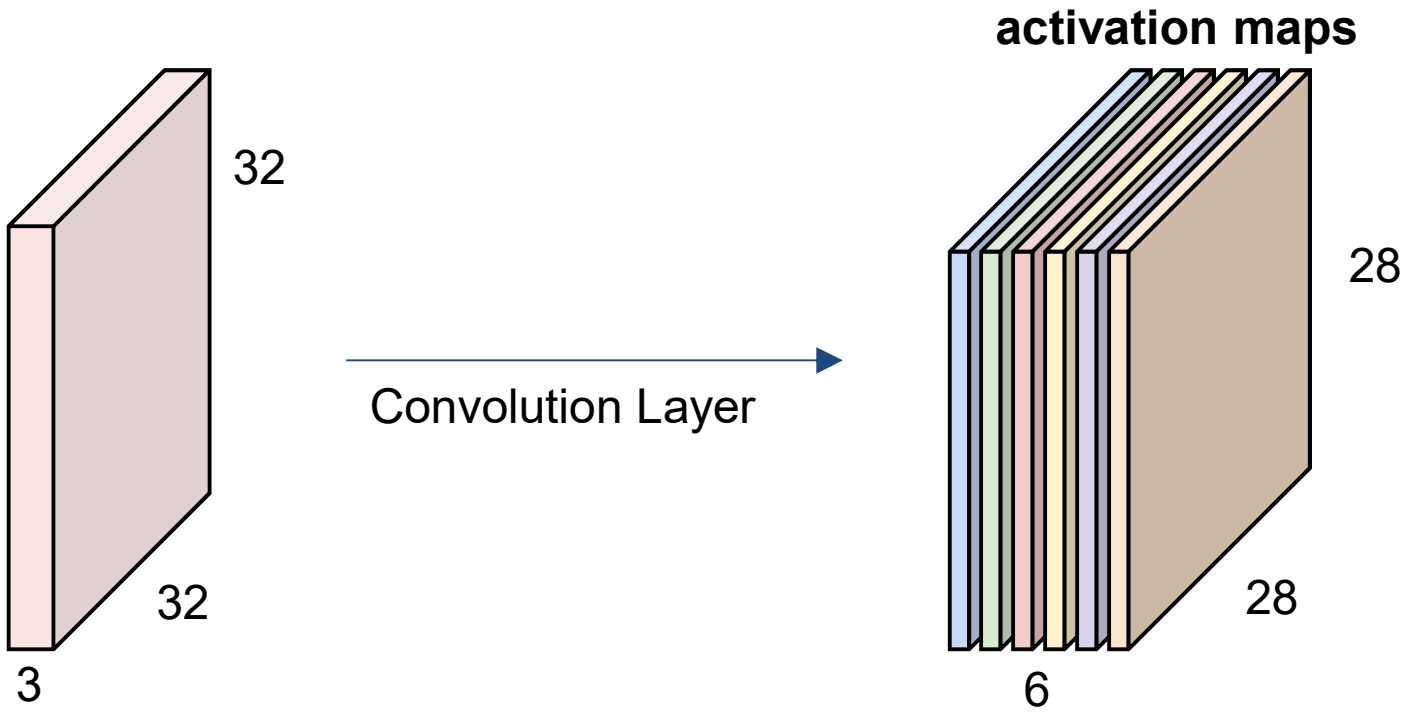
Convolution Layer

consider a second, green filter



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

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!

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Convolutions for programmers

$$\mathrm{S}(i,j) = (I * \omega)(i,j)$$

$$= \sum_{m=0}^{K} \sum_{n=0}^{K} I(i+m, j+n)\omega(m,n)$$

- Updated for multi channel inputs

$$= \sum_{m=0}^{K} \sum_{n=0}^{K} I(i+m, j+n, :)\omega(m,n,:)$$

- Note: 3 loops just for *one* output value of *one* filter, actually we will need 6 nested loops…

Convolutions for programmers

- Forward pass with input depth D and P filters, in pseudocode (ignoring boundary conditions):

```
for i in 1..W
    for j in 1..H
        for m in 1..K
            for n in 1..K
                for p in 1..P
                    for d in 1..D
                        output(i, j, p) += input(i+m, j+n, d) * filter(p, m, n, d)
                    end
                end
            end
        end
    end
end
```

Im2Col

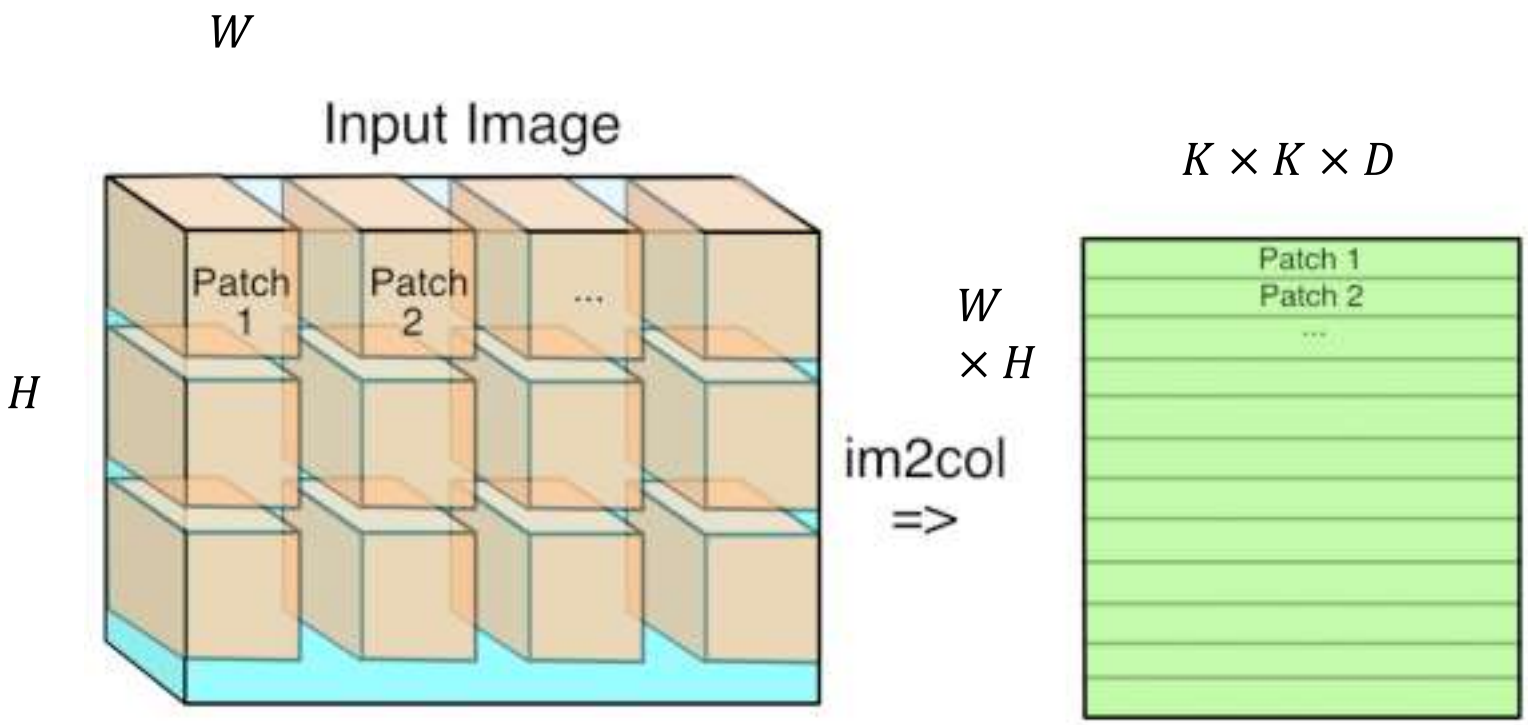
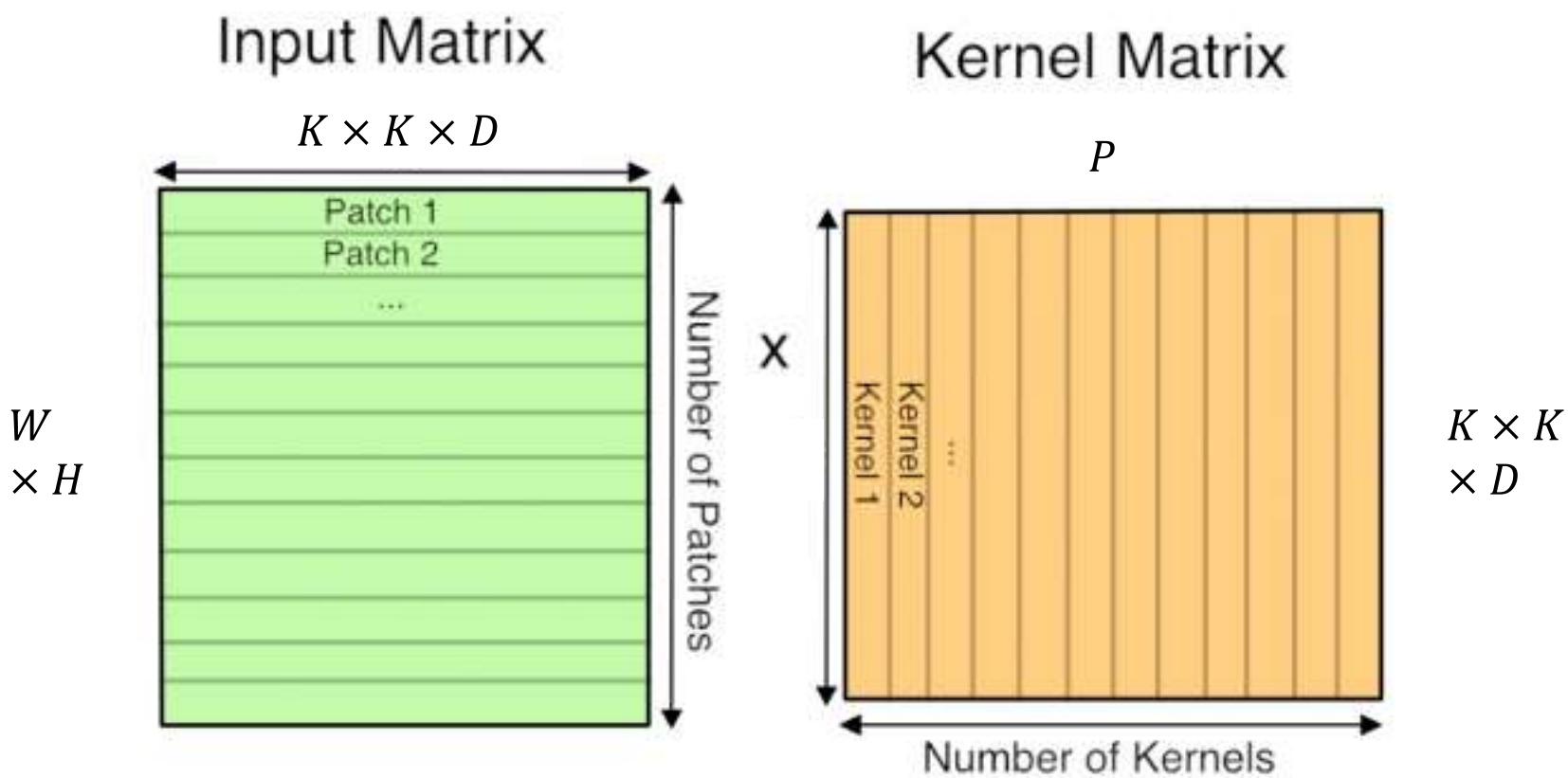


 Figure Credit: https://petewarden.com/2015/04/20/why-gemm-is-at-the-heart-of-deep-learning/

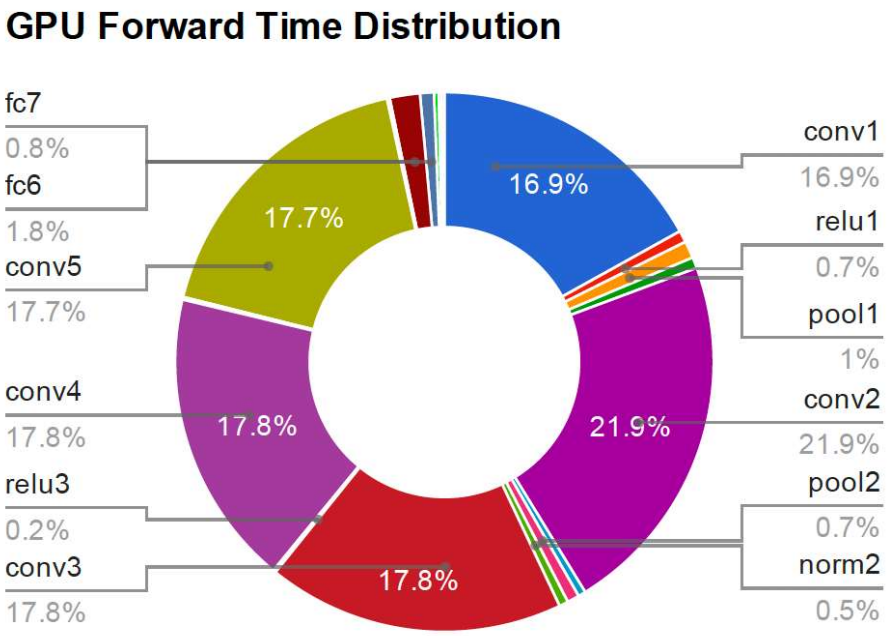
GEMM



Output: $W \times H$ by P

 Figure Credit: https://petewarden.com/2015/04/20/why-gemm-is-at-the-heart-of-deep-learning/

Time Distribution of AlexNet



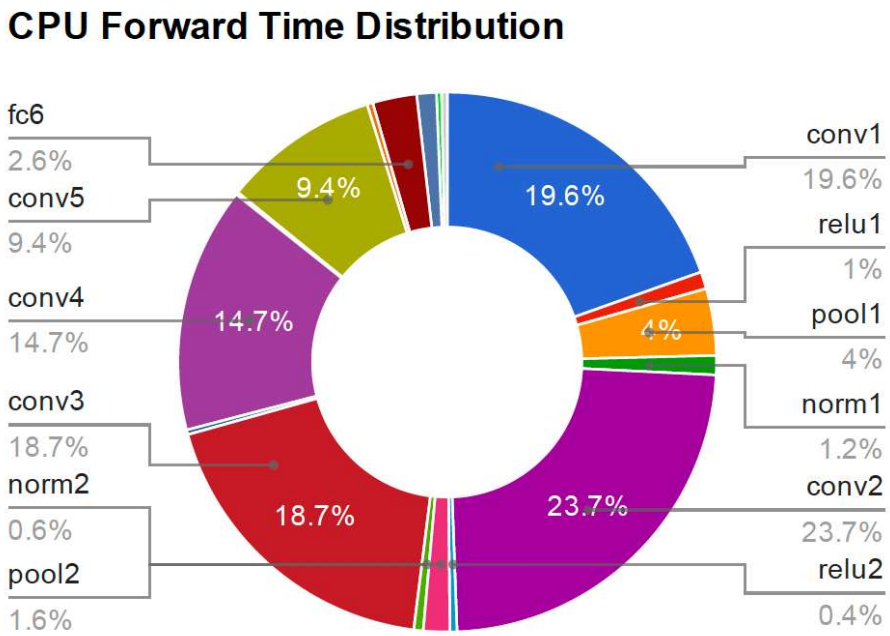
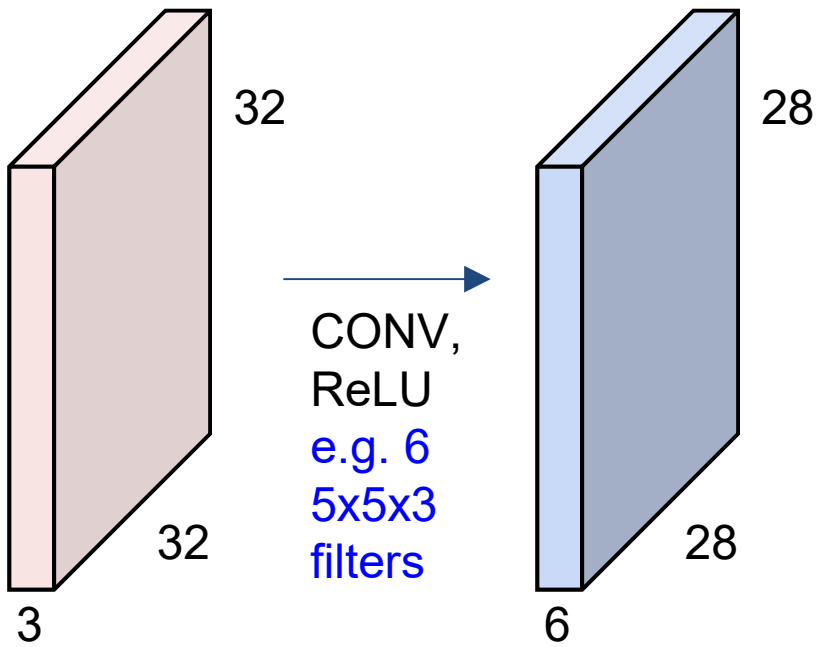


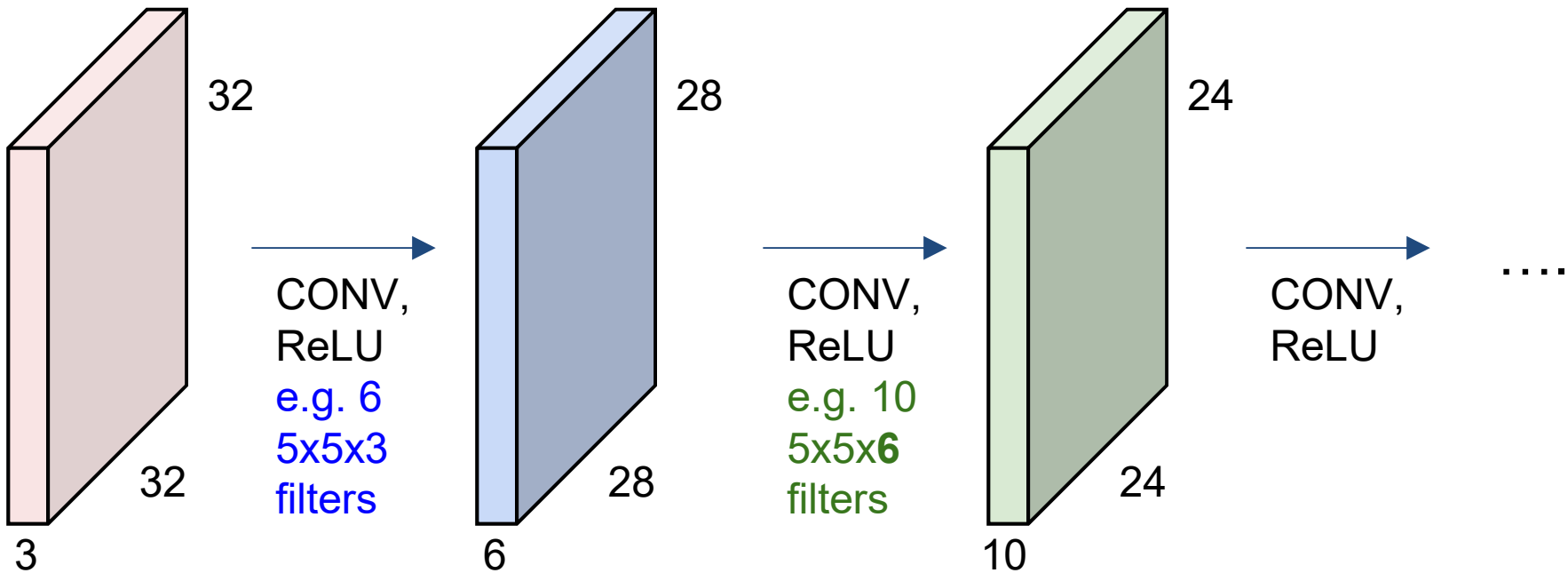
Figure Credit: Yangqing Jia, PhD Thesis

Preview: ConvNet is a sequence of Convolution Layers, interspersed with activation functions



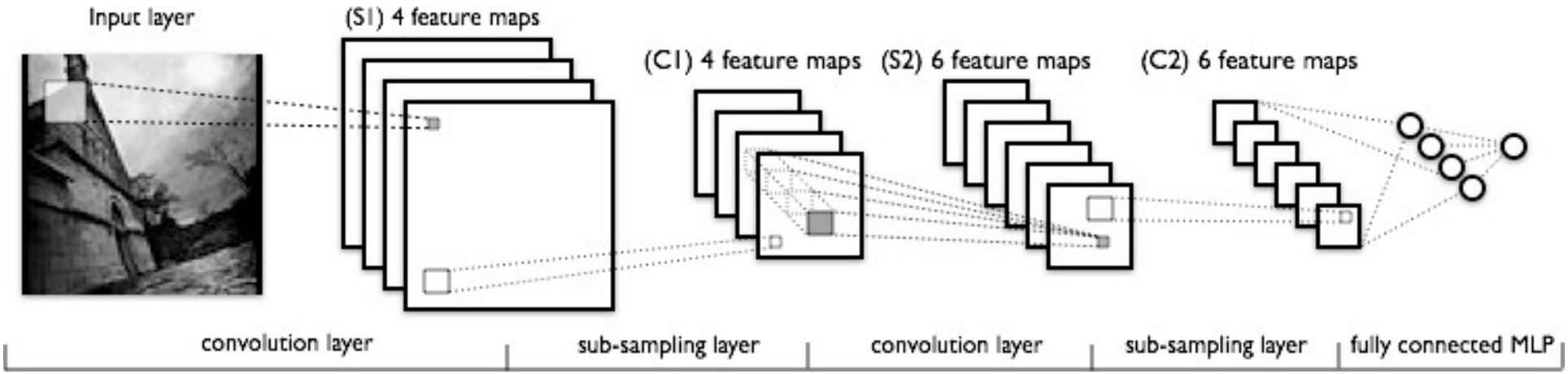
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Preview: ConvNet is a sequence of Convolutional Layers, interspersed with activation functions



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Convolutional Neural Networks



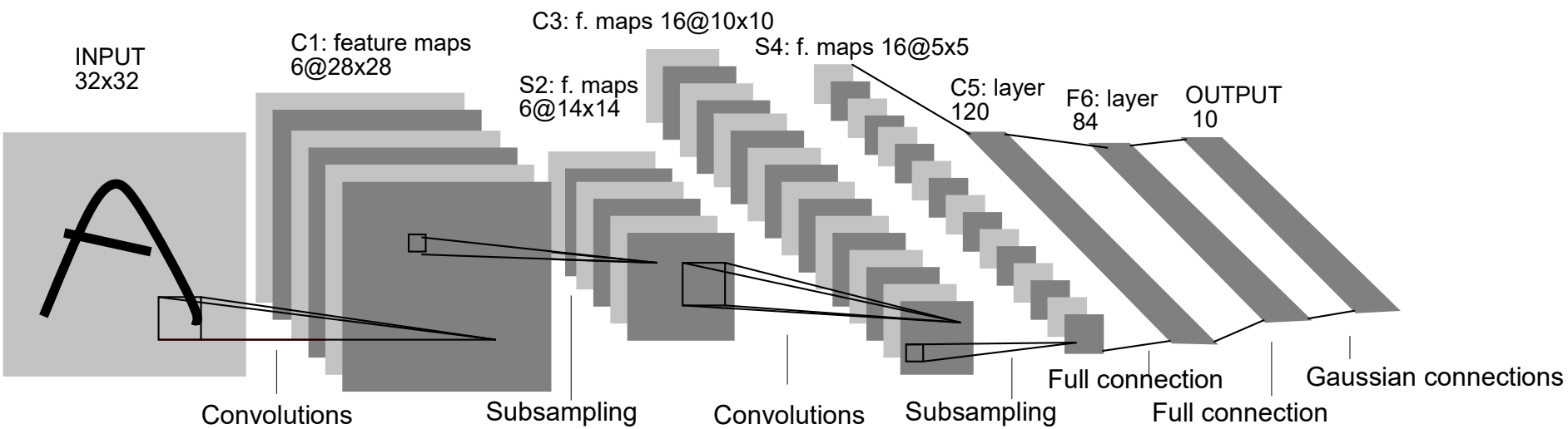
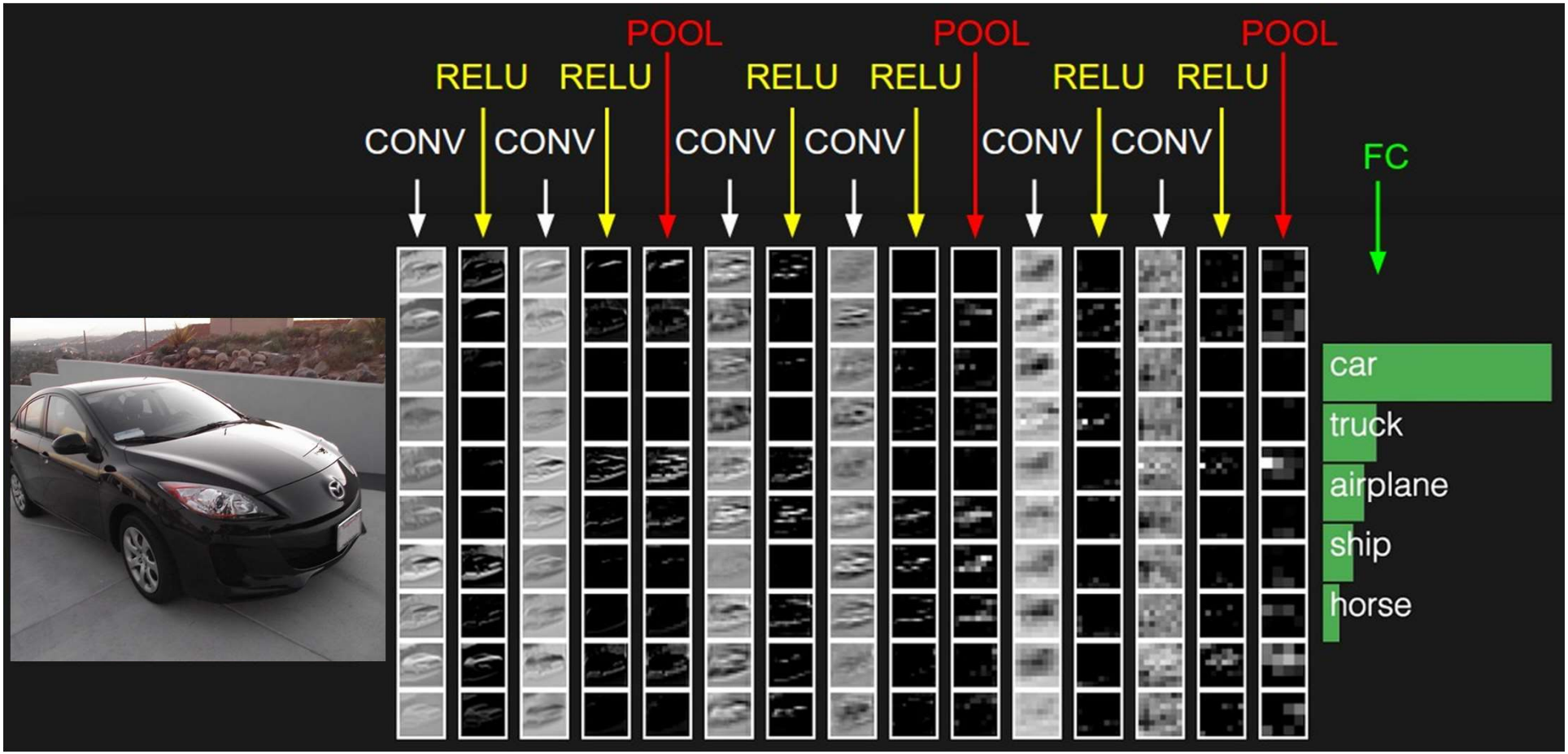


Image Credit: Yann LeCun, Kevin Murphy

preview:

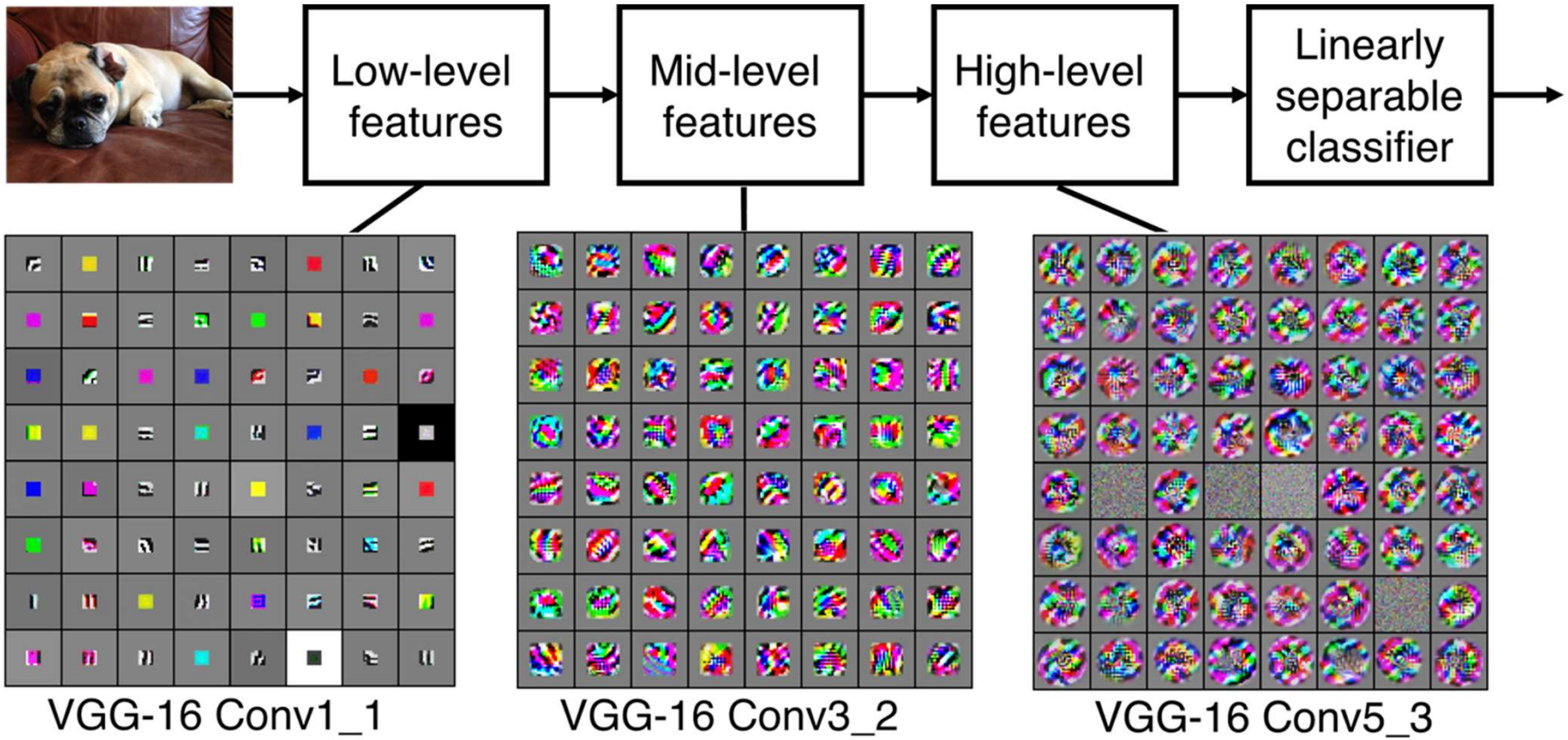


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

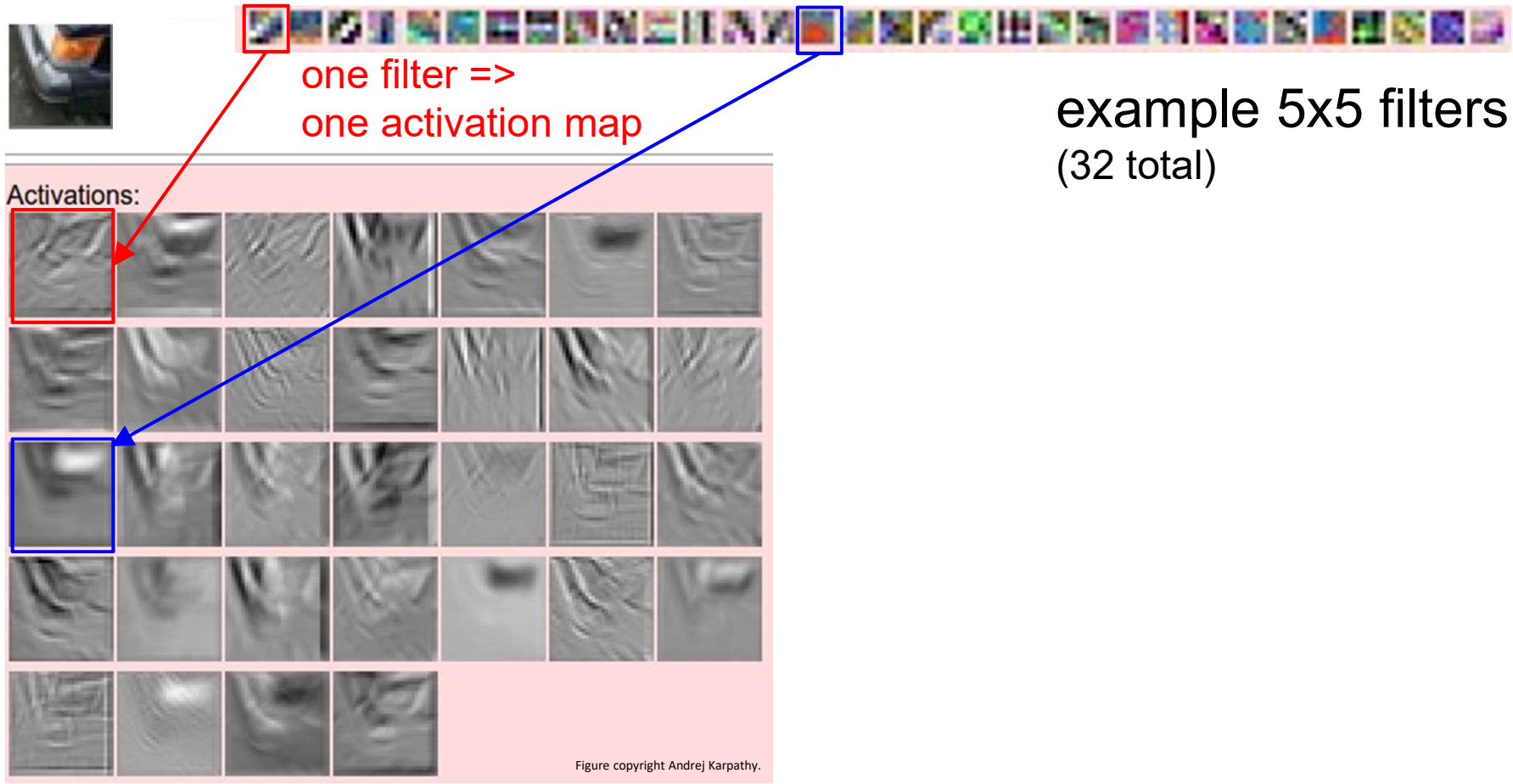
Preview

[Zeiler and Fergus 2013]

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



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n



example 5x5 filters
(32 total)

Visualizing Learned Filters

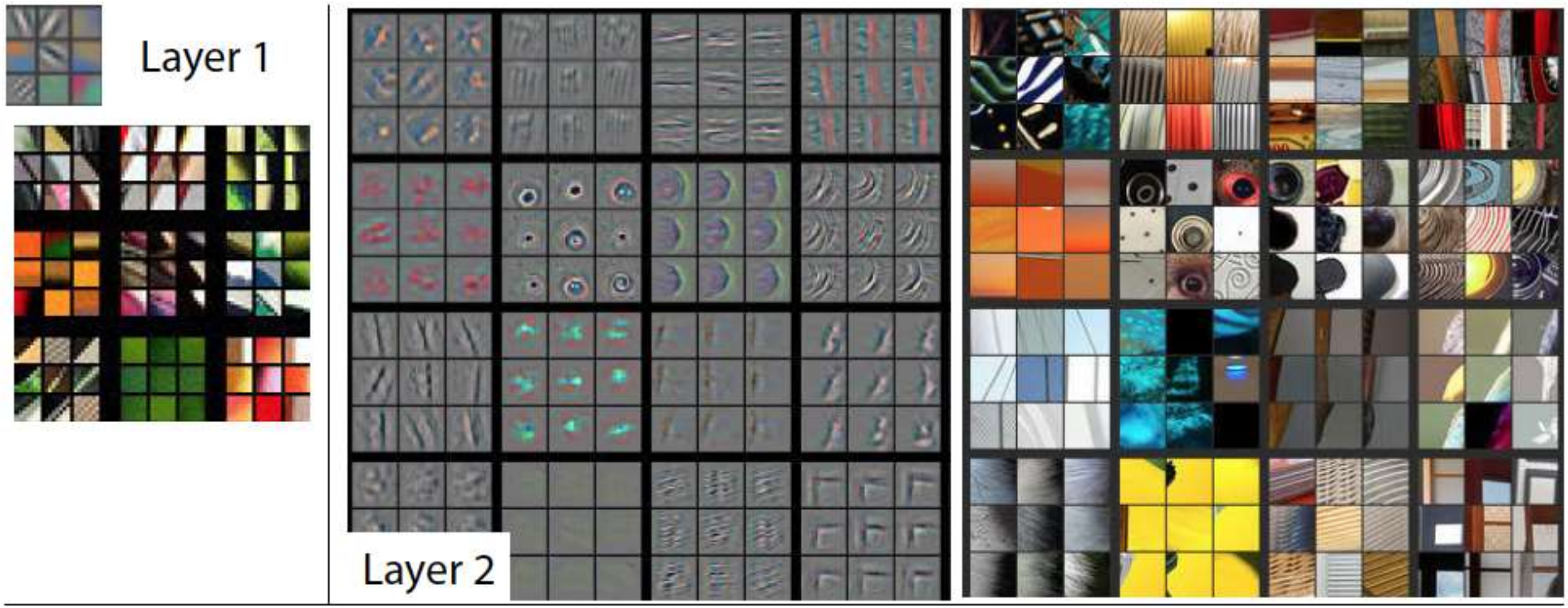


Figure Credit: [Zeiler & Fergus ECCV14]

Visualizing Learned Filters

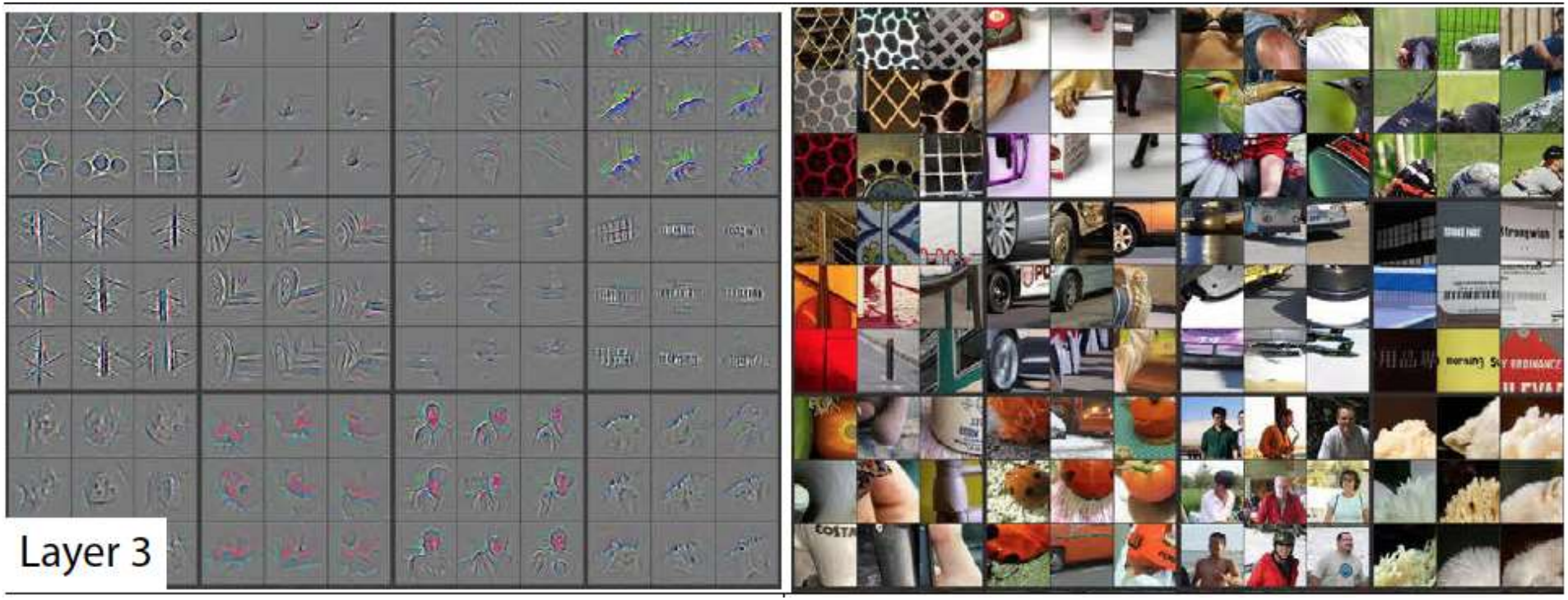


Figure Credit: [Zeiler & Fergus ECCV14]

Visualizing Learned Filters

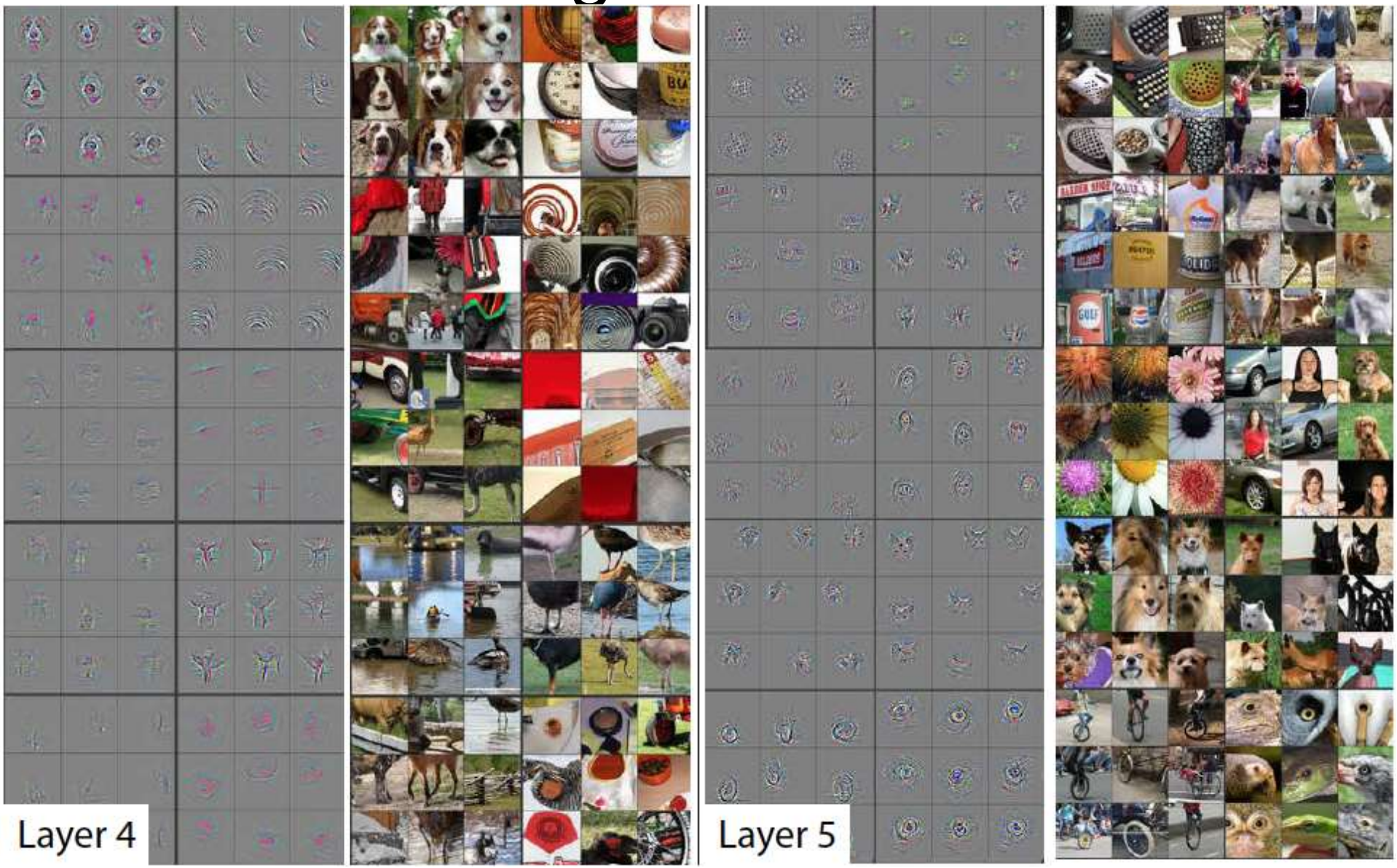
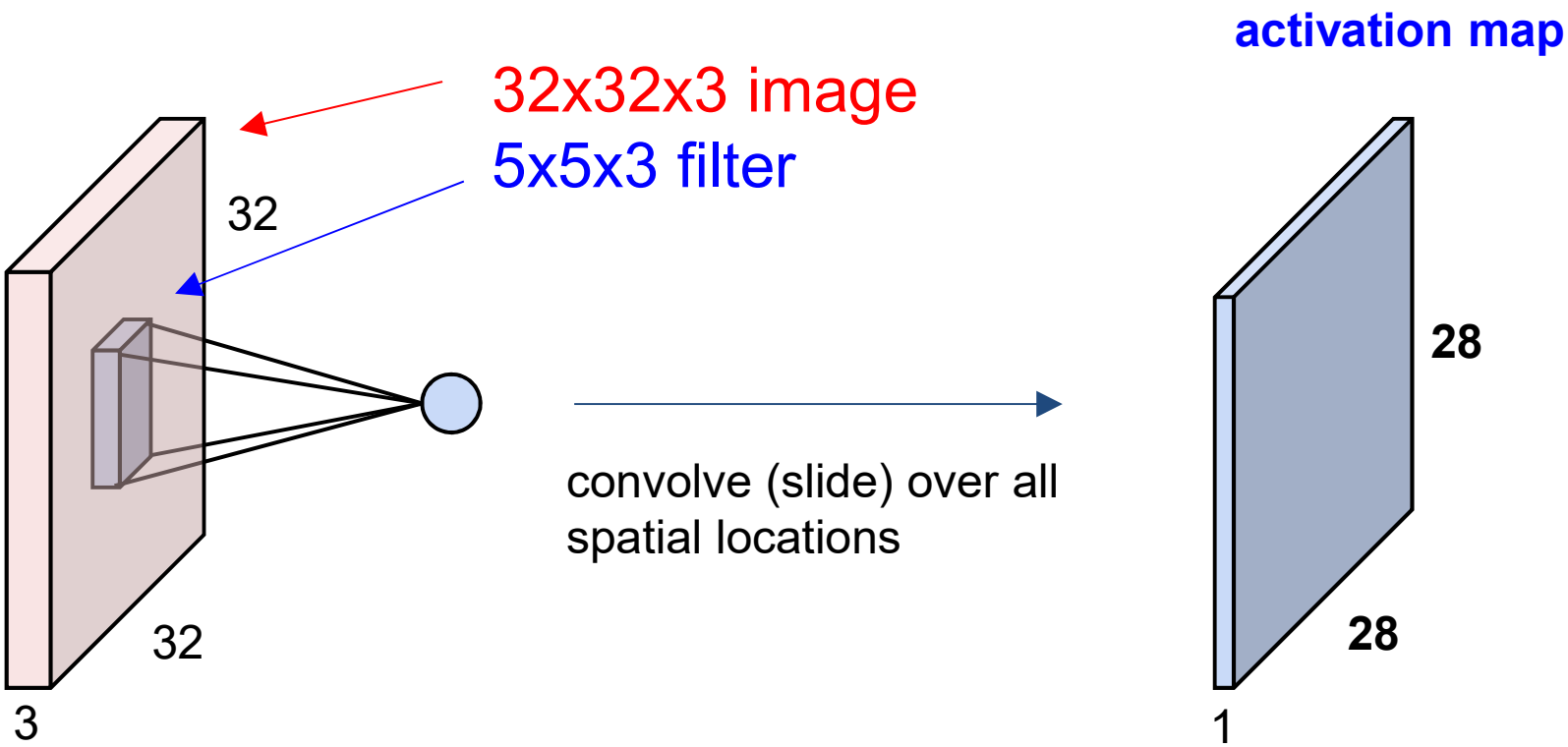


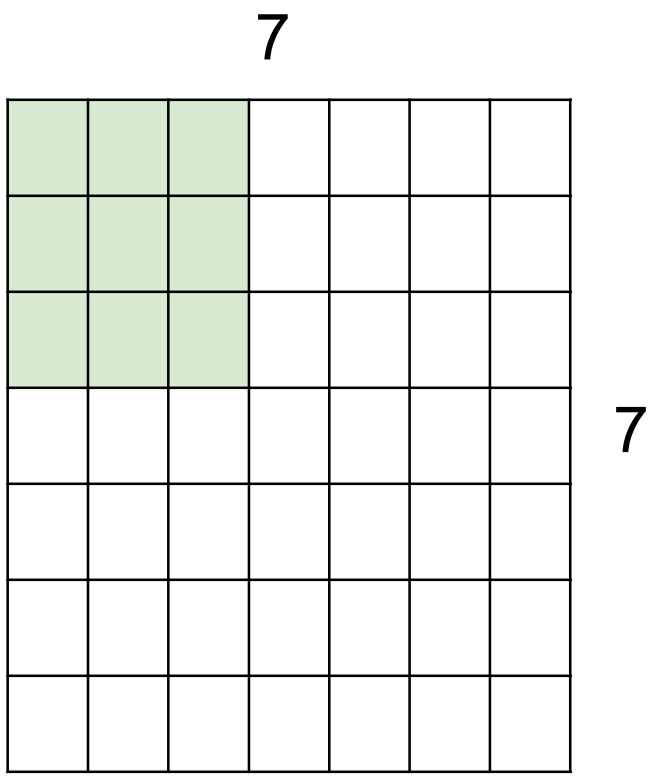
Figure Credit: [Zeiler & Fergus ECCV14]

A closer look at spatial dimensions:



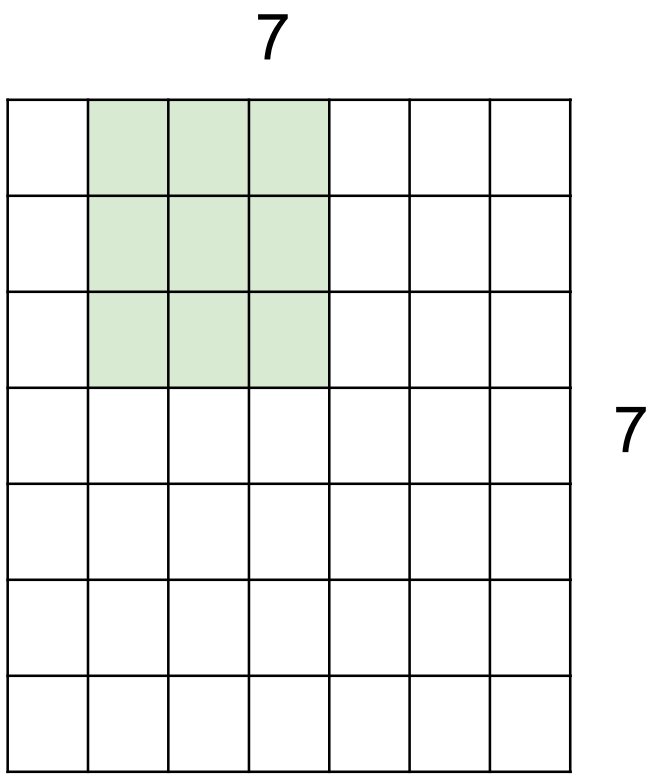
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter

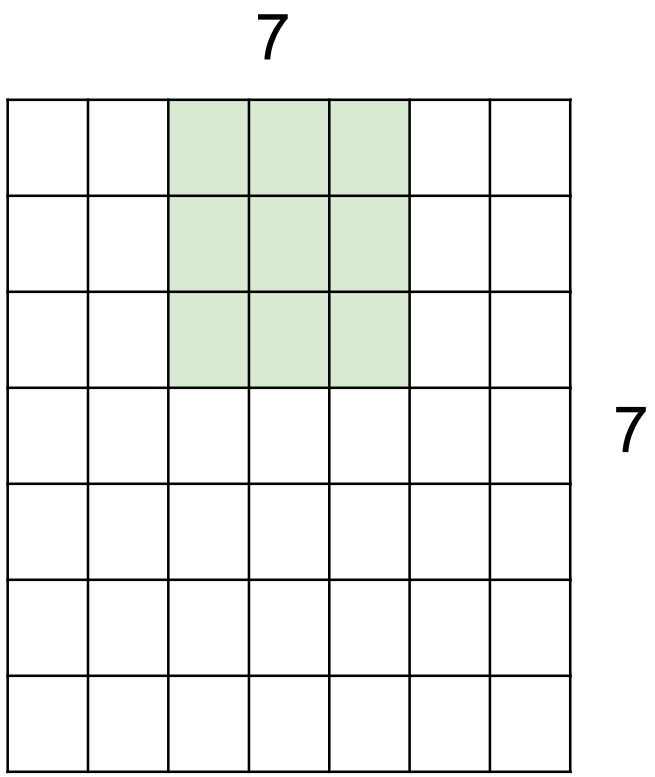
A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter

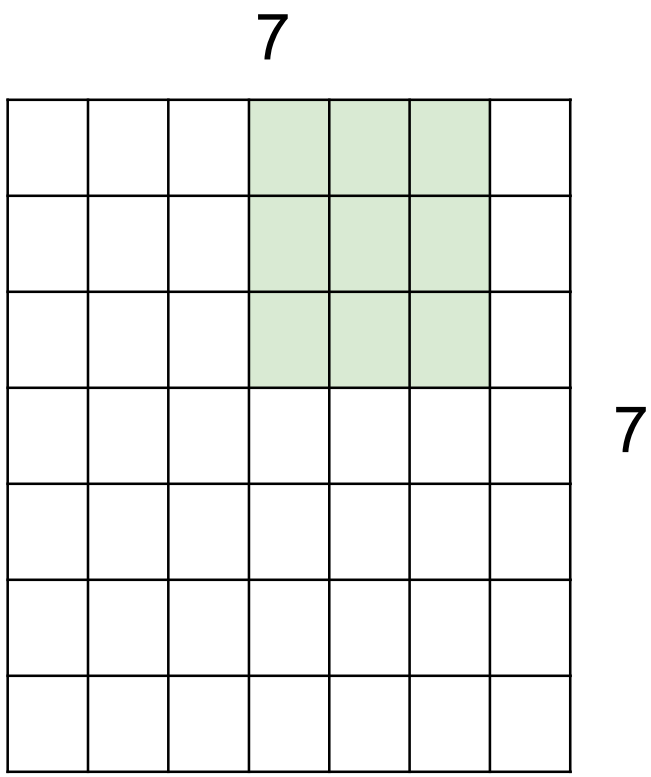
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter

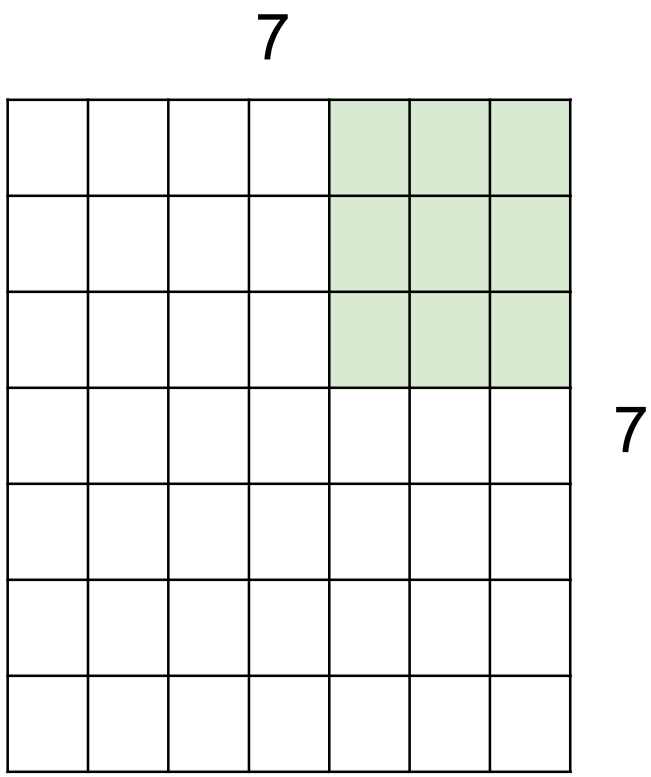
A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

A closer look at spatial dimensions:

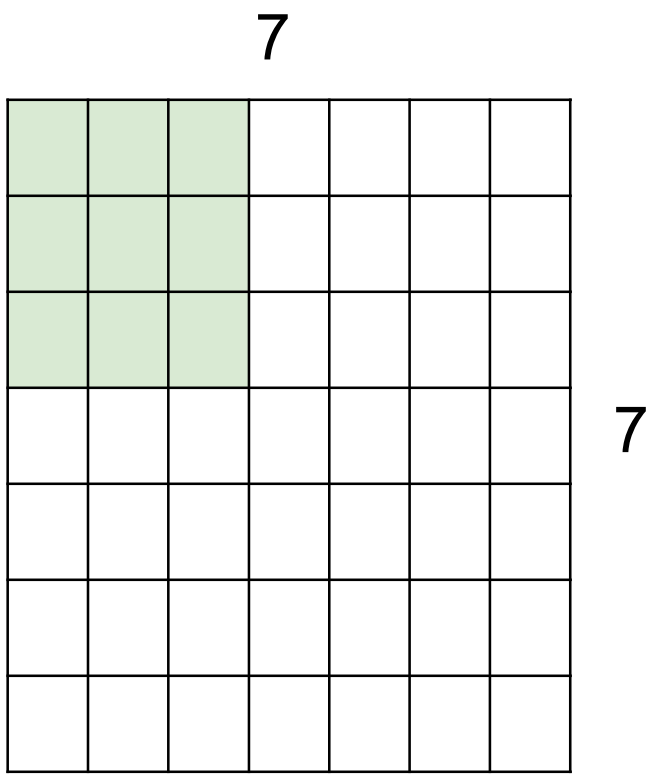


7x7 input (spatially)
assume 3x3 filter

=> 5x5 output

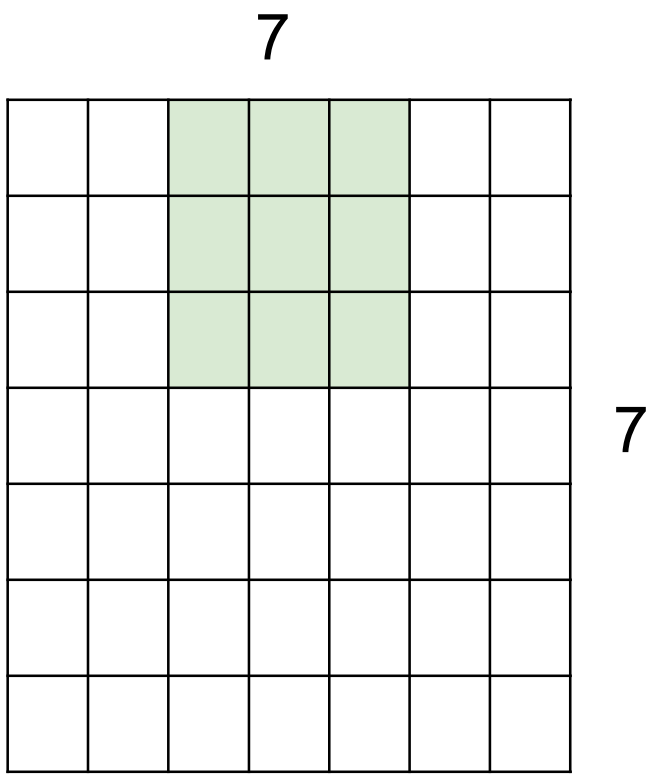
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

A closer look at spatial dimensions:



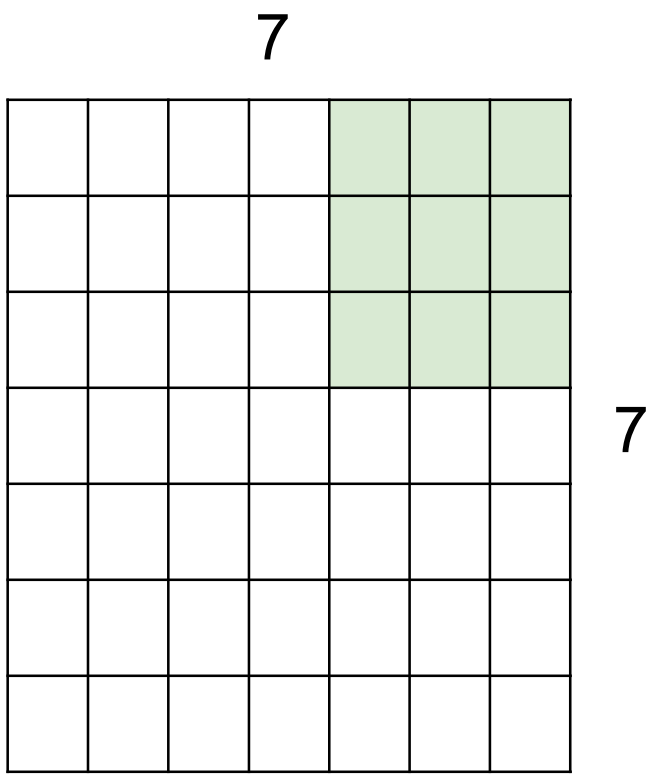
7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**

A closer look at spatial dimensions:



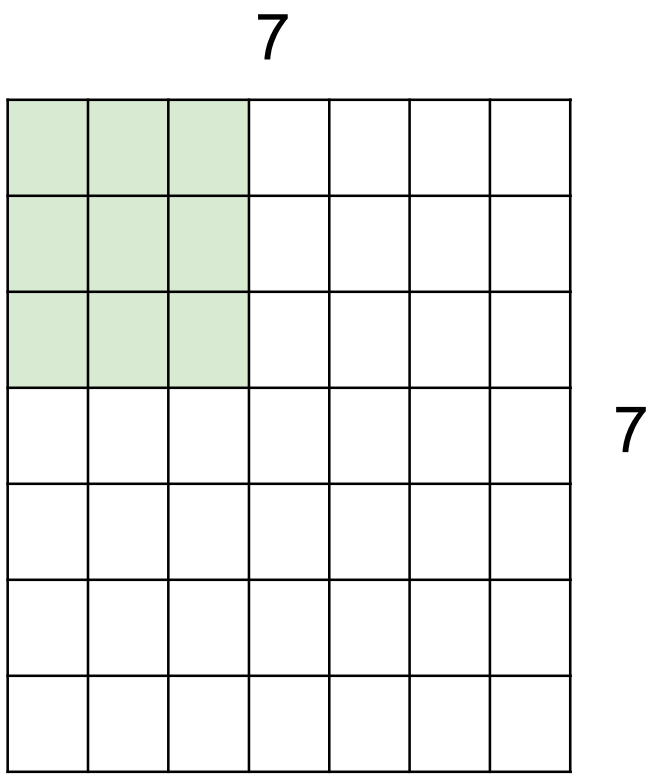
7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**

A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**
=> 3x3 output!

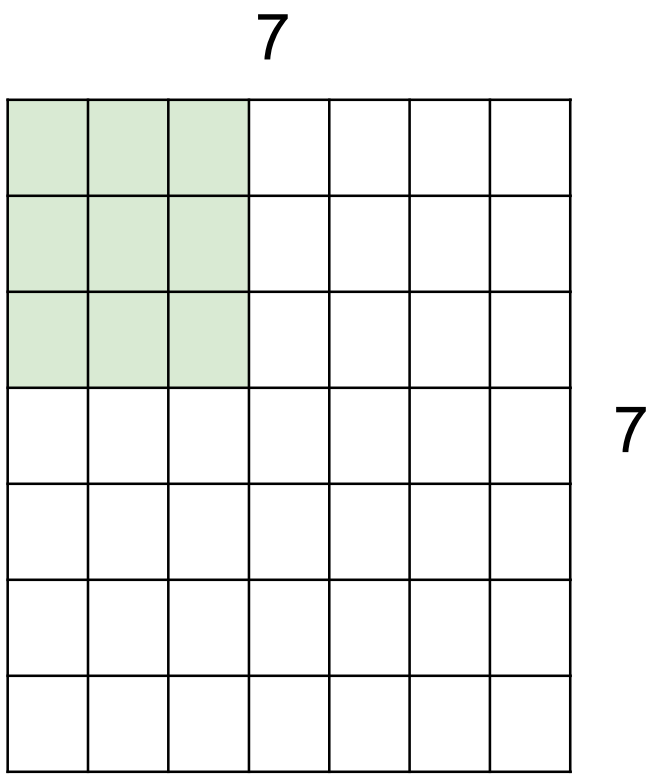
A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter
applied **with stride 3?**

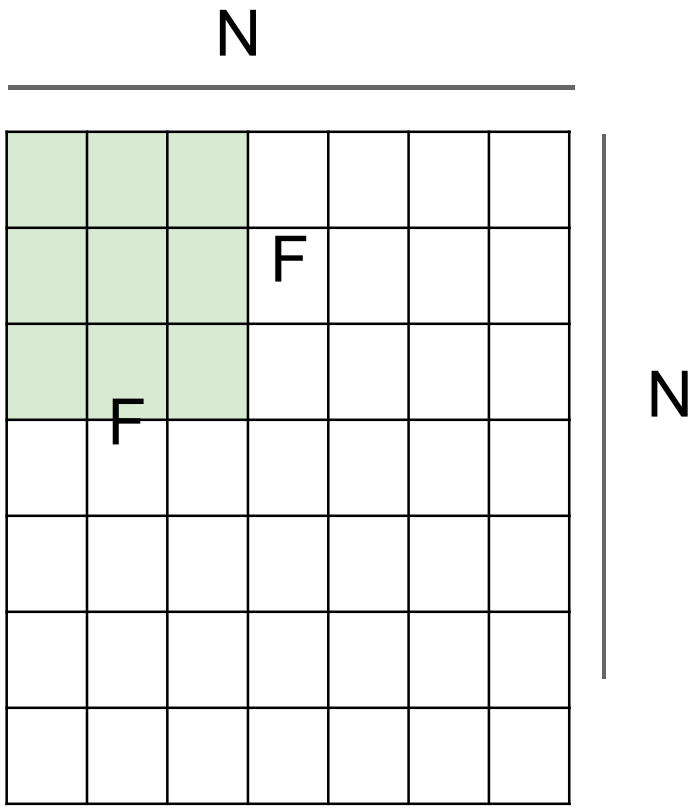
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter
applied **with stride 3?**

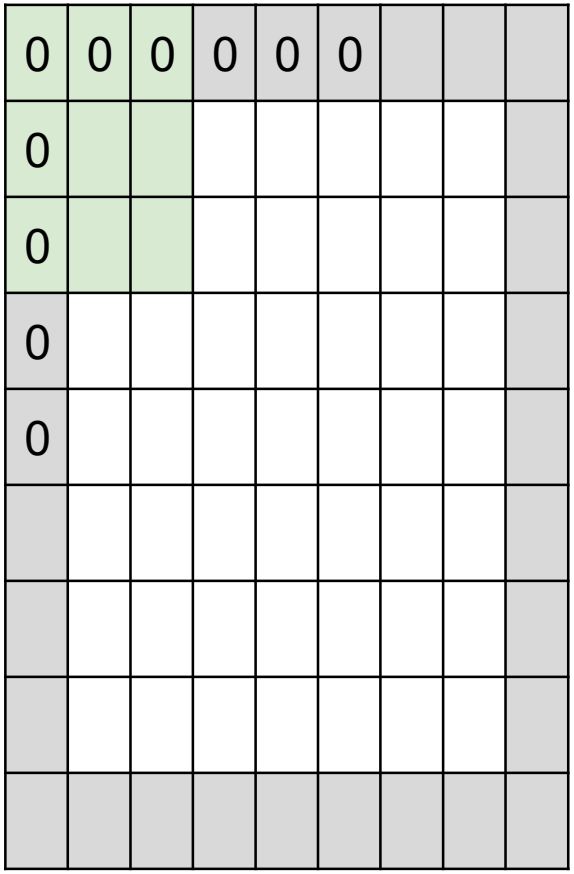
doesn't fit!
cannot apply 3x3 filter on
7x7 input with stride 3.



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

e.g. 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 :\

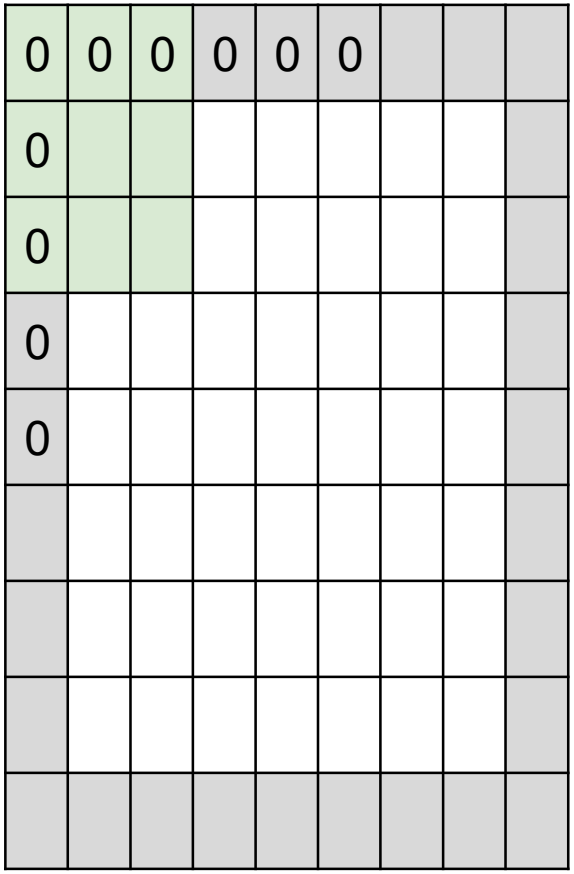
In practice: Common to zero pad the border



e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

(recall:)
(N - F) / stride + 1

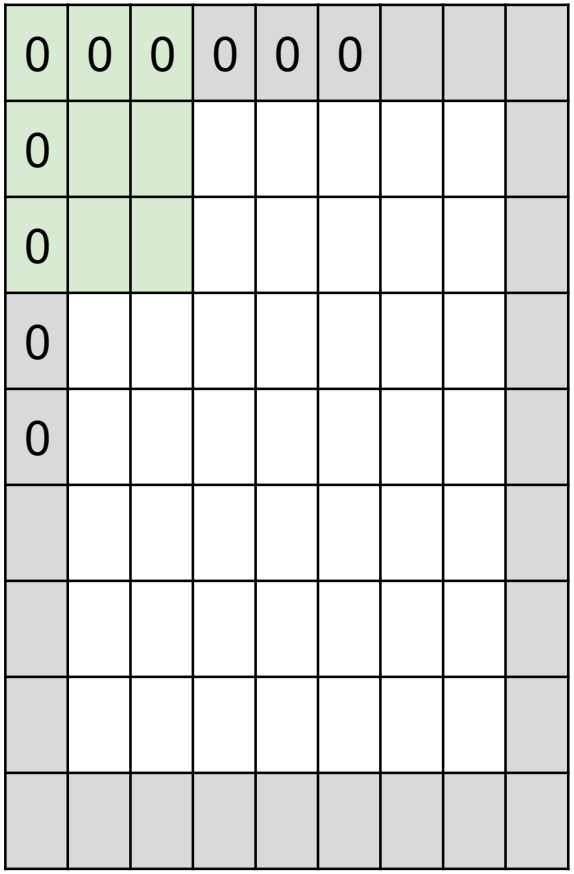
In practice: Common to zero pad the border



e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

7x7 output!

In practice: Common to zero pad the border

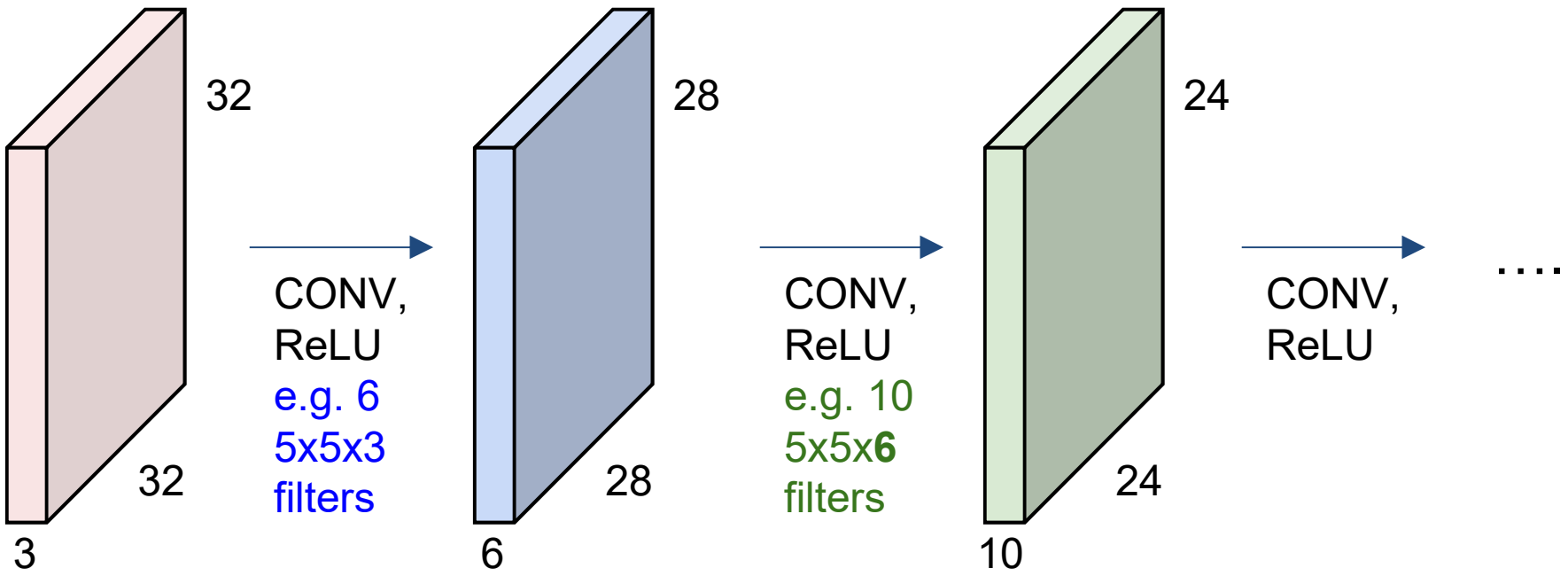


e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

7x7 output!
in general, common to see CONV layers with stride 1, filters of size FxF, and zero-padding with (F-1)/2. (will preserve size spatially)
e.g. F = 3 => zero pad with 1
F = 5 => zero pad with 2
F = 7 => zero pad with 3

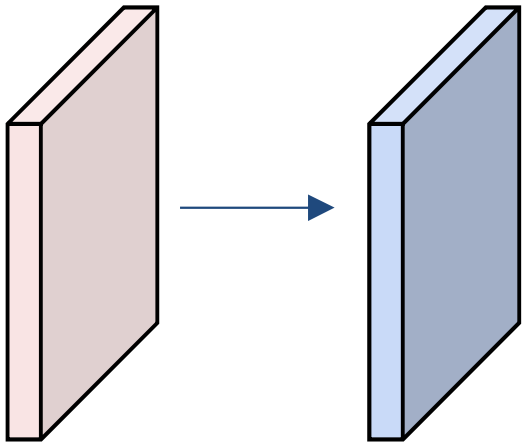
Remember back to…

E.g. 32x32 input convolved repeatedly with 5x5 filters shrinks volumes spatially! (32 -> 28 -> 24 ...). Shrinking too fast is not good, doesn't work well.



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

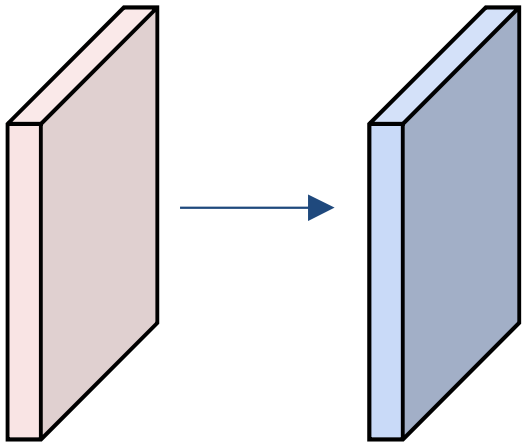
Examples time:



Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Output volume size: ?

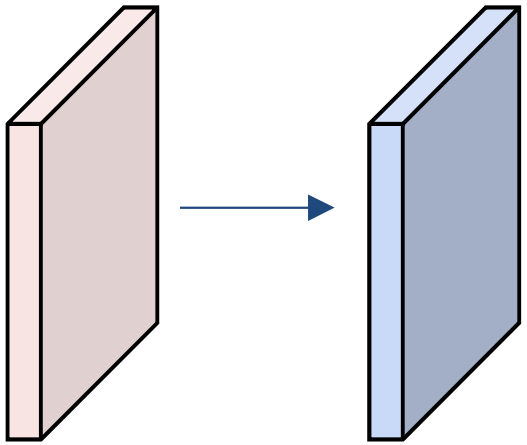
Examples time:



Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Output volume size:
(32+2*2-5)/1+1 = 32 spatially, so
32x32x10

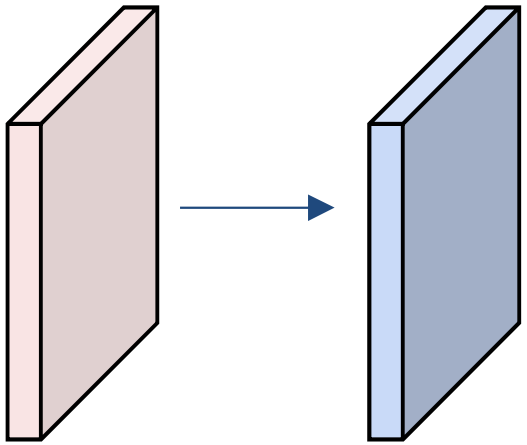
Examples time:



Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?

Examples time:



Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?
each filter has 5*5*3 + 1 = 76 params (+1 for bias)
=> 76*10 = **760**

Summary

This lecture:
- Convolutions
- Stride

Still to come:
- Pooling
- CNN architectures
- Fully connected layers as convolutions
- Backprop in conv layers