CS 480/680
Introduction to Machine Learning

Lecture 14
Convolutional Neural Networks

Kathryn Simone

05 November 2024

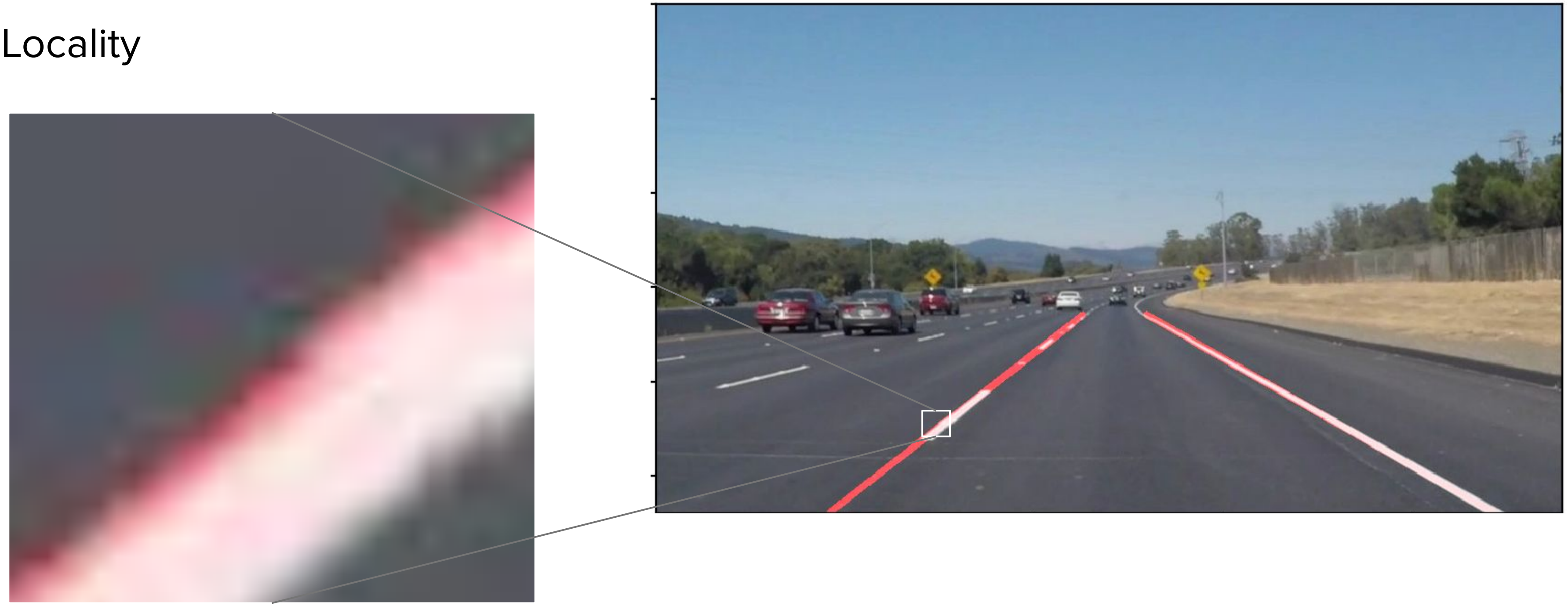
UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

Scaling up Multilayer Perceptrons

- Assume 1 megapixel image: 10^6 features
- One layer with 10^3 units: 10^9 (1 Billion parameters)
- Challenges:
  - Training cost
  - Require data
  - Risks overfitting

What properties of images could be exploited?

Locality



What properties of images could be exploited?

Locality

Spatial Invariance

Edge detection in the visual system

https://www.youtube.com/watch?v=IOHayh06LJ4

The visual system is organized hierarchically

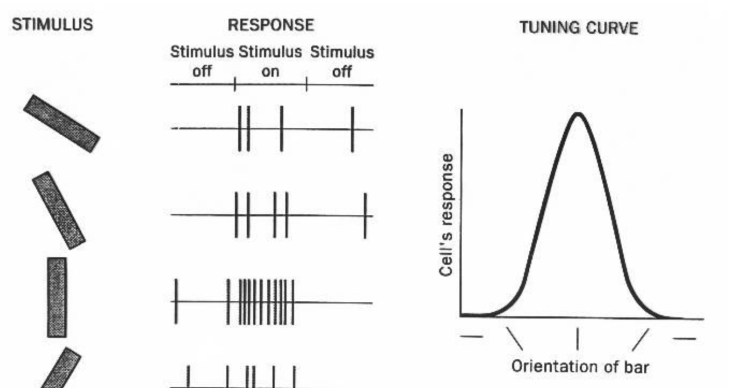
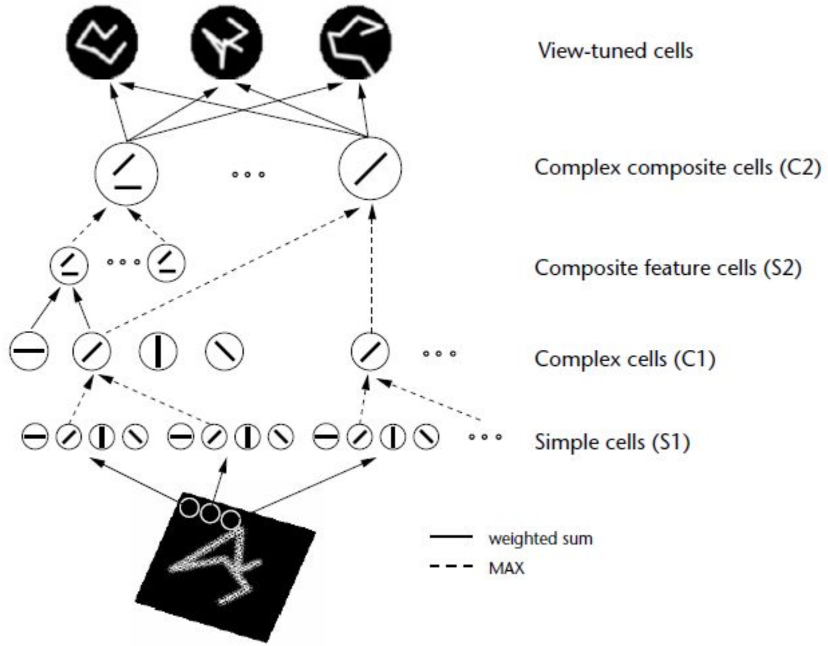


FIGURE 4.8 Response of a single cortical cell to bars presented at various orientations.



Key questions

I. How can a network detect low-level features?

II. How can a network detect higher-level features?

Convolution is a linear operation over two real-valued functions

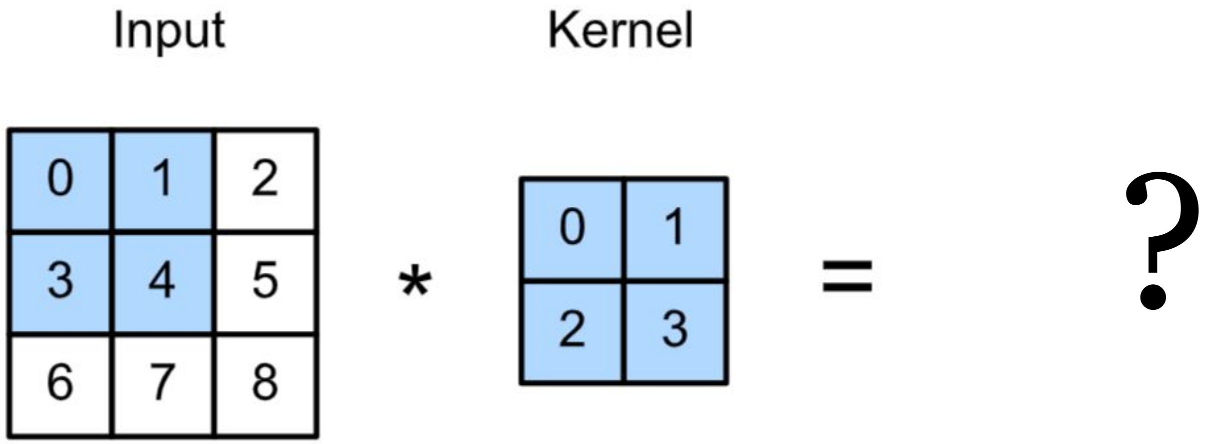
I: Input (image)
K: (Convolution) kernel, or filter
S: Output or feature map

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n)$$

Translation-equivariance:

If $g: I'[i,j] \rightarrow I[i-1,j]$, then $K \square g(I) = g(K \square I)$

Can you compute the output of this convolution?

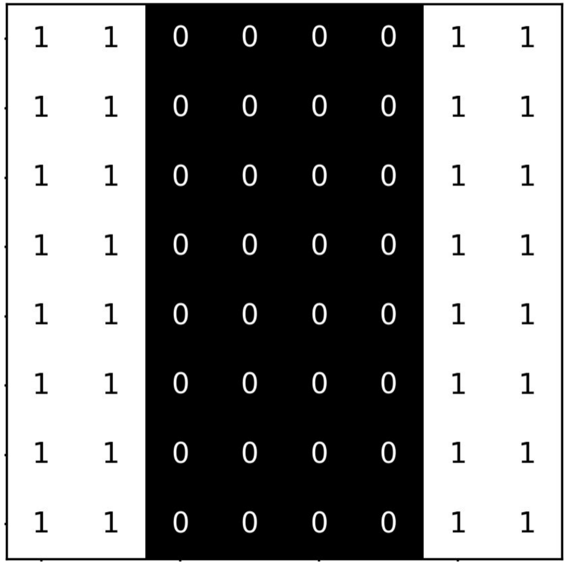


Convolution kernels *filter* for features in the input

$K = \begin{bmatrix} 1 & -1 \end{bmatrix}$

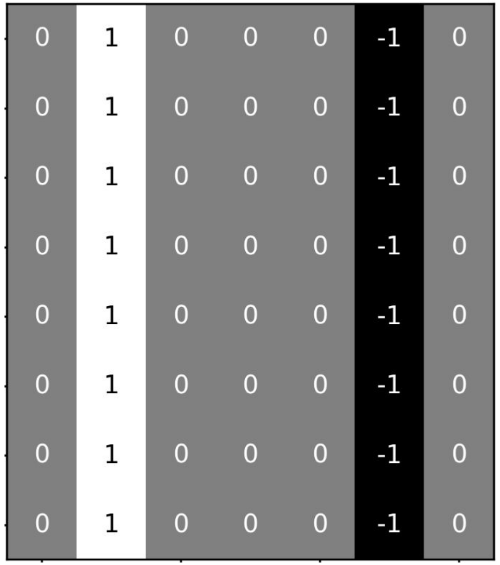
$K = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

Input



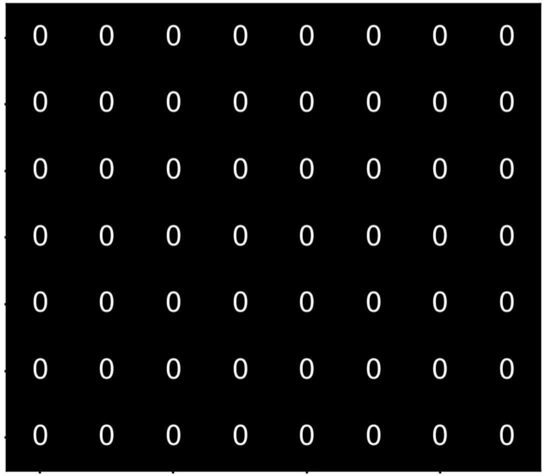
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1

Output, Filter 1



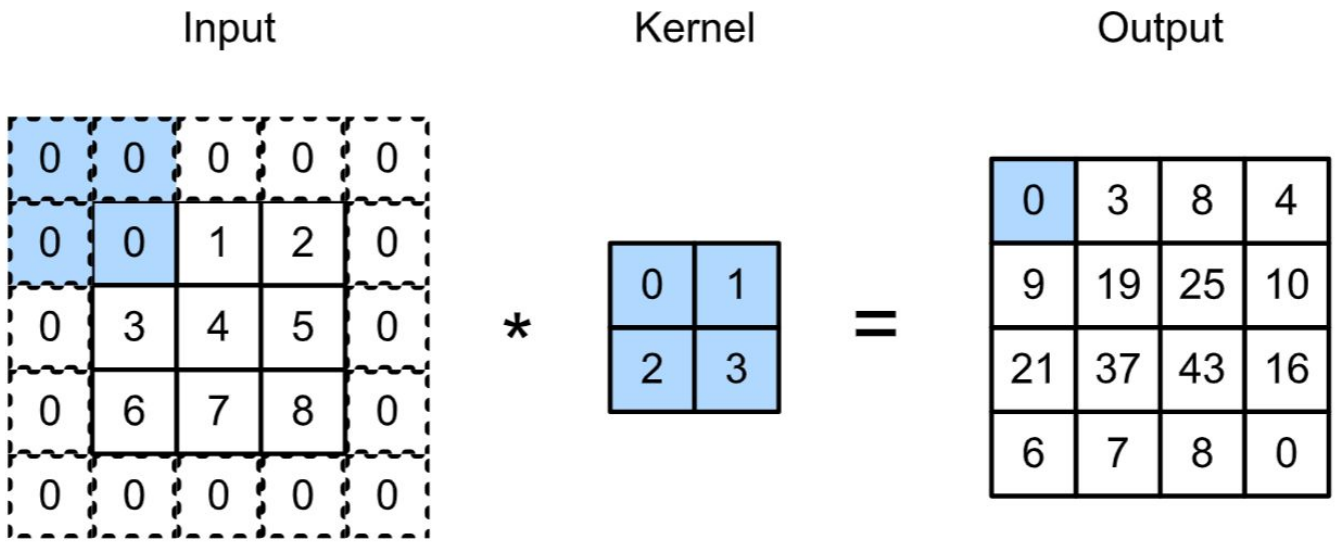
0	1	0	0	0	-1	0
0	1	0	0	0	-1	0
0	1	0	0	0	-1	0
0	1	0	0	0	-1	0
0	1	0	0	0	-1	0
0	1	0	0	0	-1	0
0	1	0	0	0	-1	0
0	1	0	0	0	-1	0

Output, Filter 2



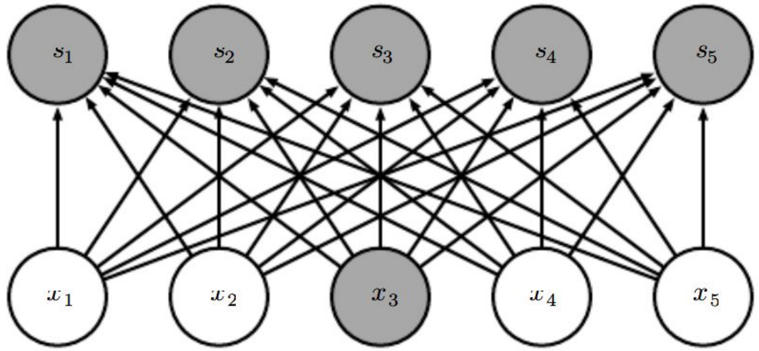
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

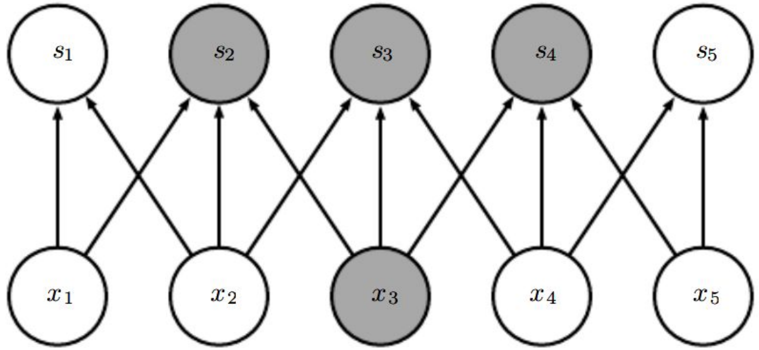
Padding prevents loss of input pixels



Dive into Deep Learning Section 7.2

A filter is realized in a CNN through sparse weights





“Fully-Connected” Neural Network

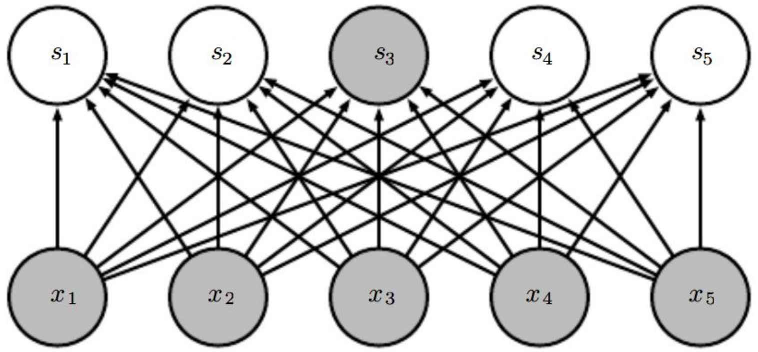
- Every output interacts with every input

Convolutional Neural Network

- Only three outputs are affected by a given input

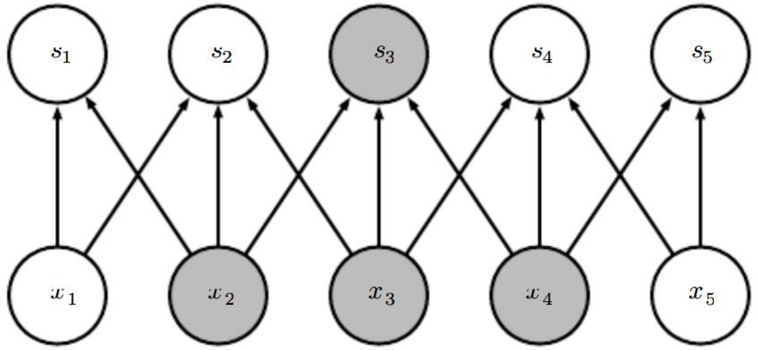
Deep Learning Section 9.2

A filter is realized in a CNN through sparse weights



“Fully-Connected” Neural Network

- Every output interacts with every input

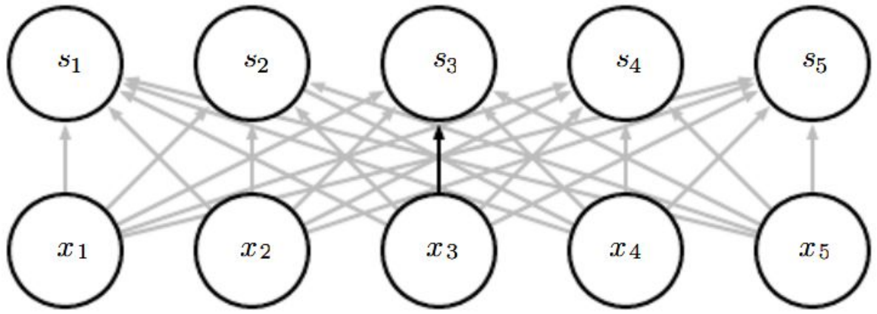


Convolutional Neural Network

- Only three inputs affect an output

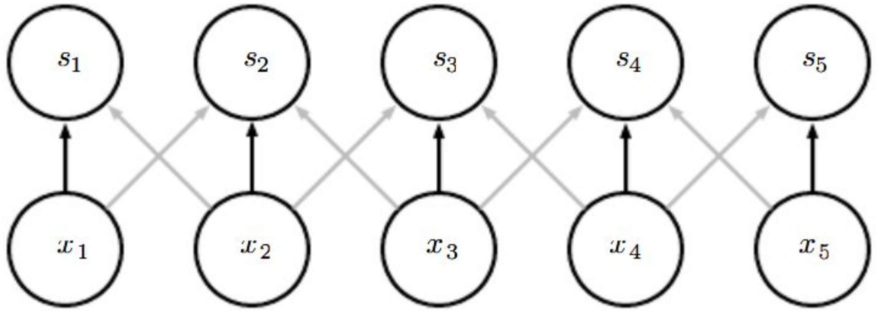
Deep Learning Section 9.2

Parameter sharing achieves translation-equivariance



“Fully-Connected” Neural Network

- Interactions have distinct parameters

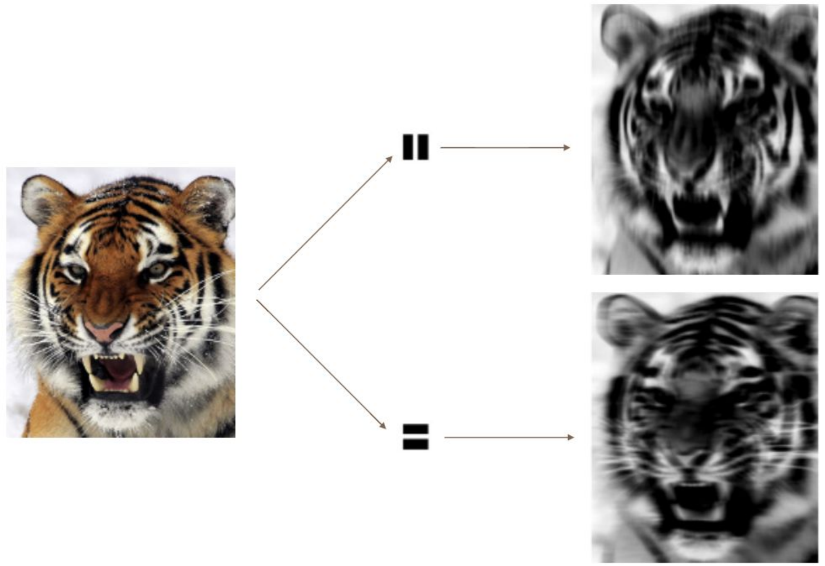


Convolutional Neural Network

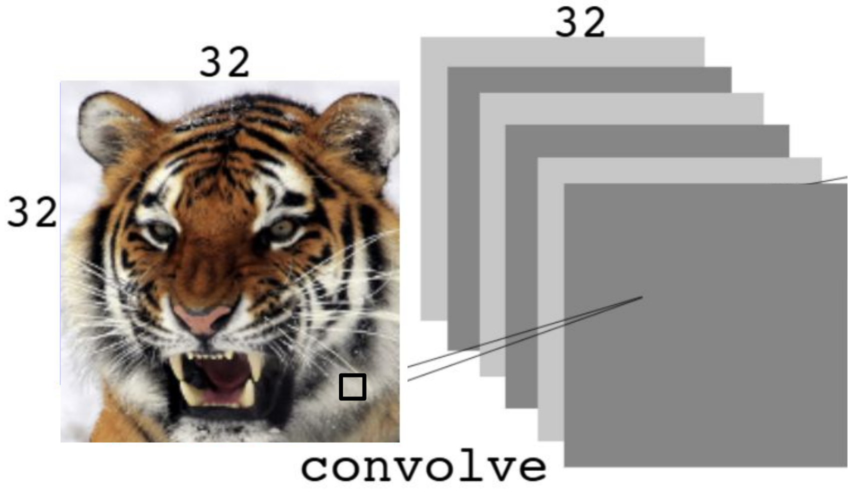
- Parameters are shared

Deep Learning Section 9.2

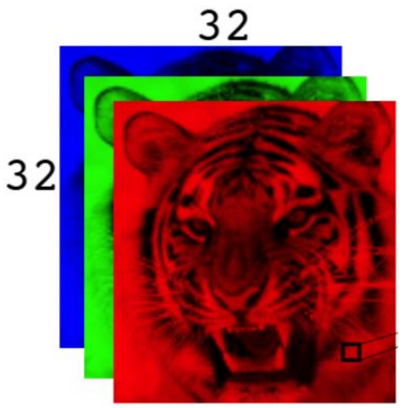
The network must be able to detect a variety of patterns

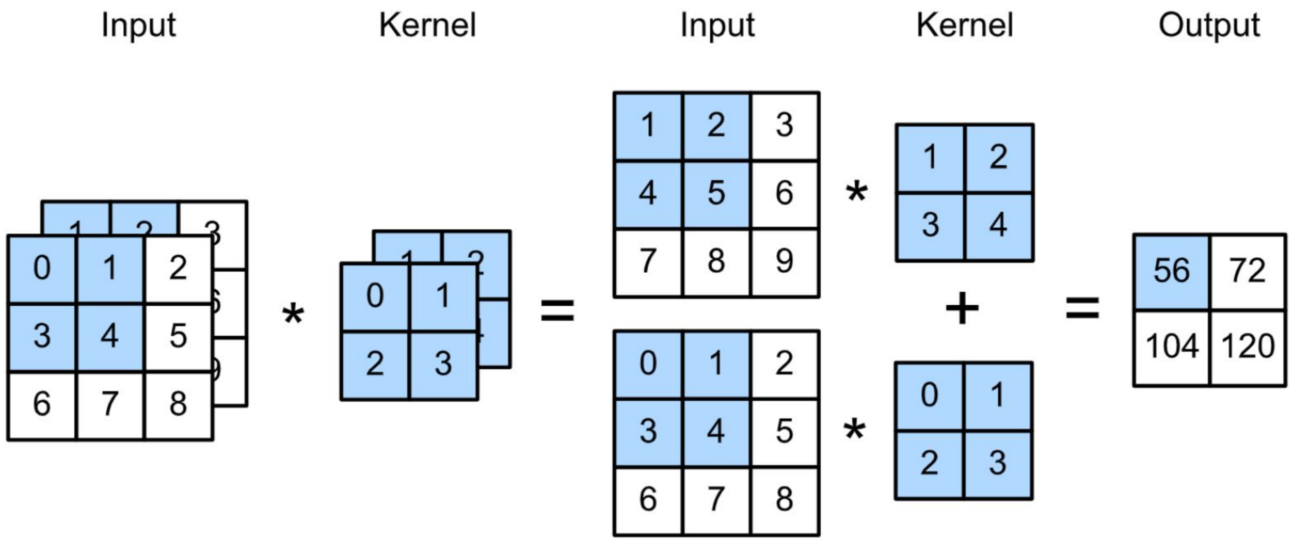


A convolutional *stage* generates several feature maps



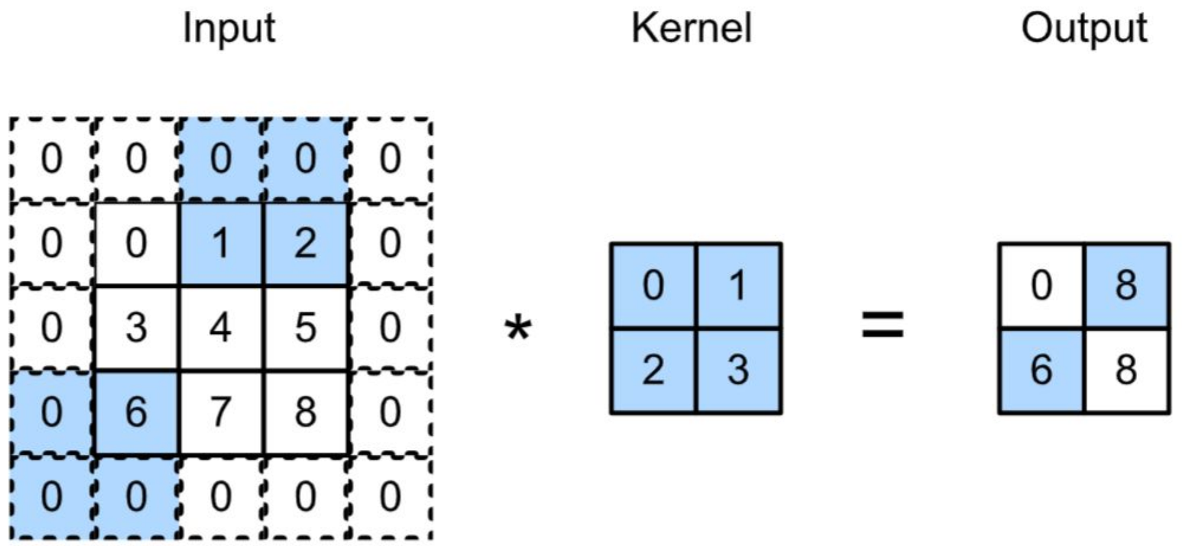
Convolution kernels with input multiple channels





Dive into Deep Learning Section 7.2

Stride: Trade off spatial resolution for output channel depth



Dimensionality and parameters of a convolution stage

Input:

$$W_{IN} \times H_{IN} \times D_{IN}$$

Hyperparameters:

- Number of filters N_k
- Kernel Size $F \times F \times D_{IN}$
- Padding P
- Stride S

Output:

$$W_{OUT} \times H_{OUT} \times D_{OUT}$$

$$W_{OUT} = (W_{IN} + 2P - F)/S + 1$$
$$H_{OUT} = (H_{IN} + 2P - F)/S + 1$$
$$D_{OUT} = N_k$$

# Trainable parameters:

	$N_k \times F \times F \times D_{IN}$	weights
+	N_k	biases

Adapted from Gautam Kamath’s lecture notes

Convolution is translation equivariant but not invariant

Cat

Cat

Convolution is not invariant to other transformations

Top: Introduction to Statistical Learning Section 10.3
Bottom: Deep Learning, Section 9.10

Key questions

I. How can a network detect low-level features?

II. **How can a network detect higher-level features?**

Recall simple and complex cells

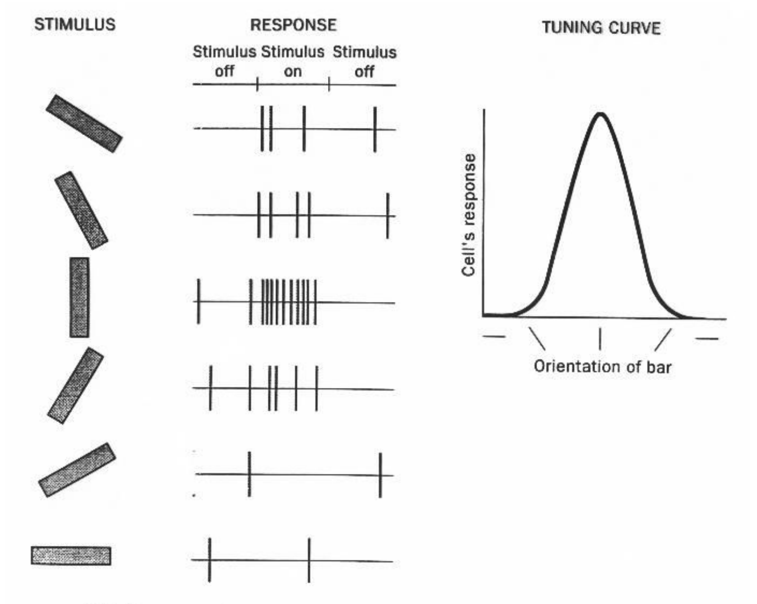
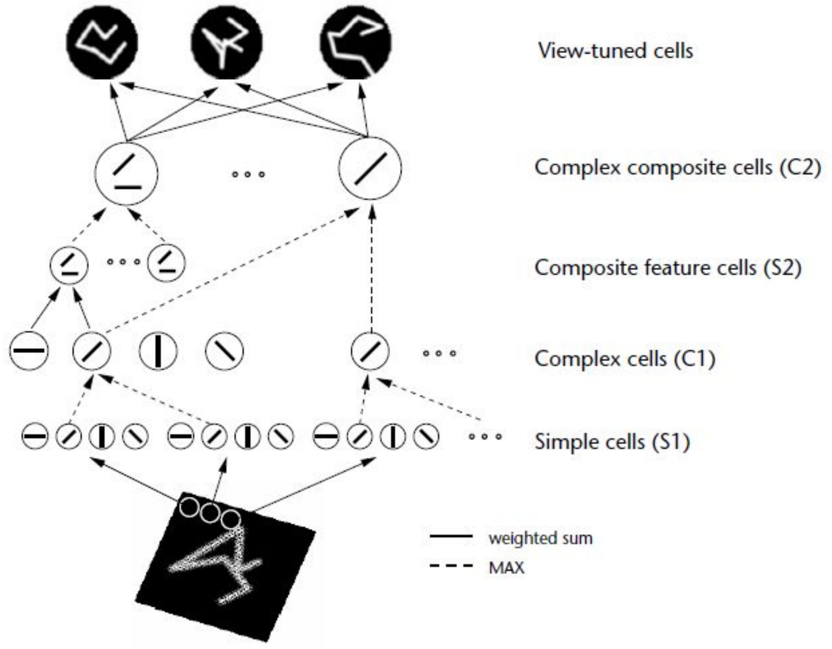
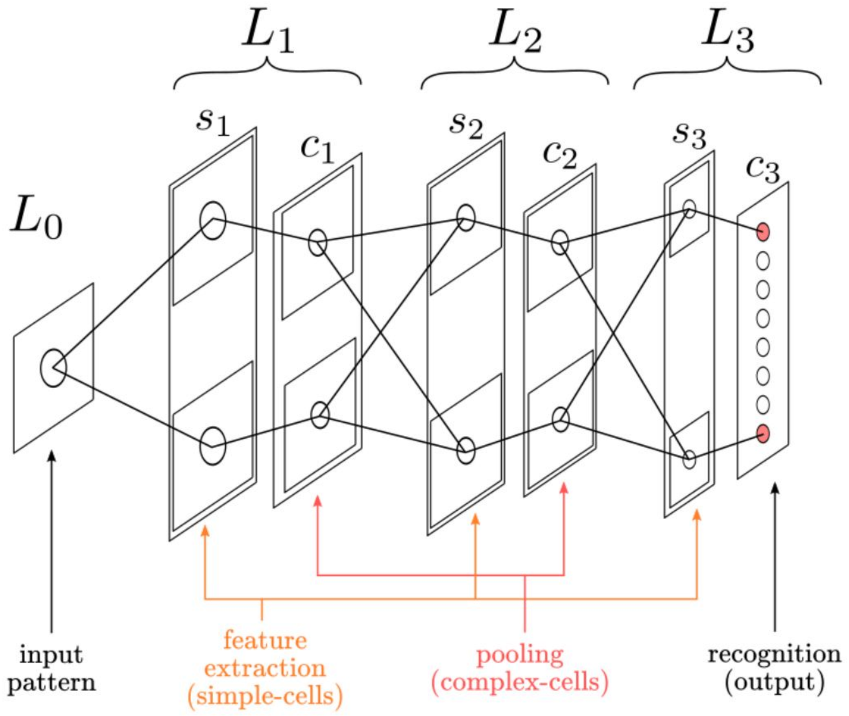


FIGURE 4.8 Response of a single cortical cell to bars presented at various orientations.

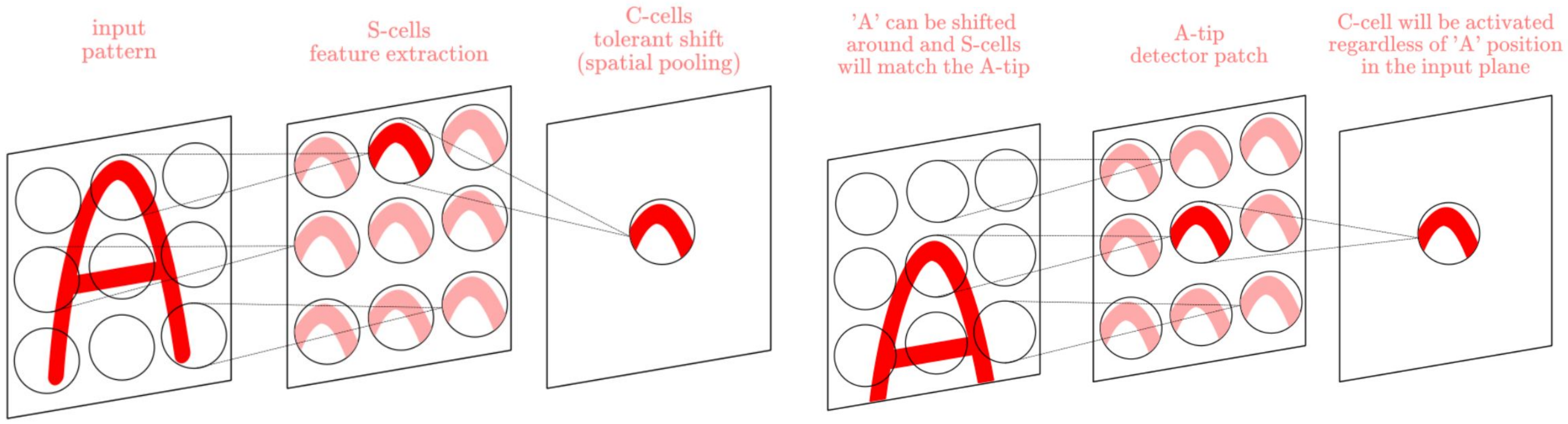


Fukushima’s Neocognitron (1980) and pooling



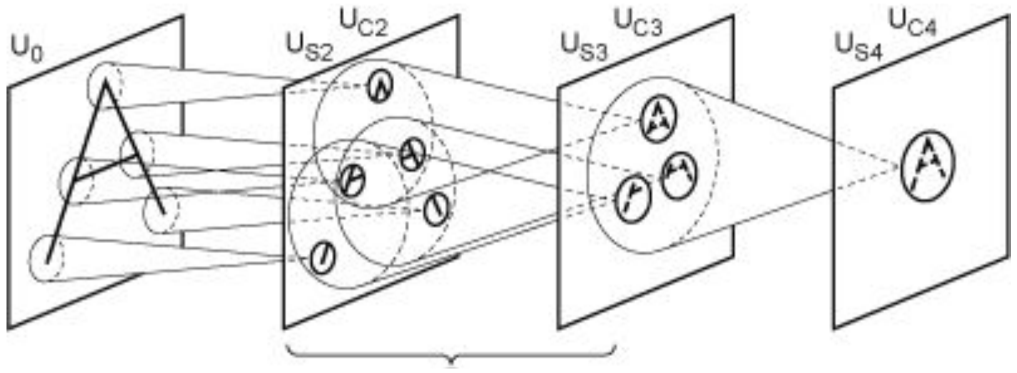
https://pabloinsente.github.io/the-convolutional-network

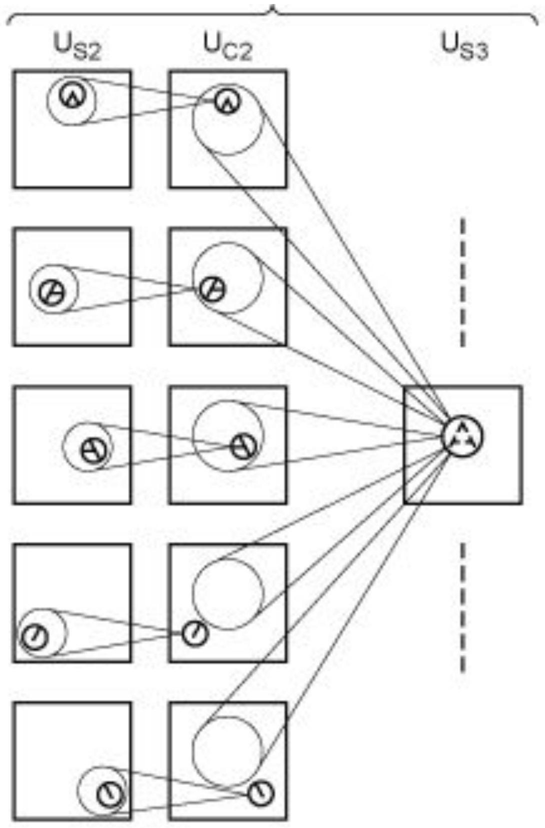
Pooling over spatial regions confers translation invariance


input pattern
S-cells feature extraction
C-cells tolerant shift (spatial pooling)
'A' can be shifted around and S-cells will match the A-tip
A-tip detector patch
C-cell will be activated regardless of 'A' position in the input plane

https://pabloinsente.github.io/the-convolutional-network

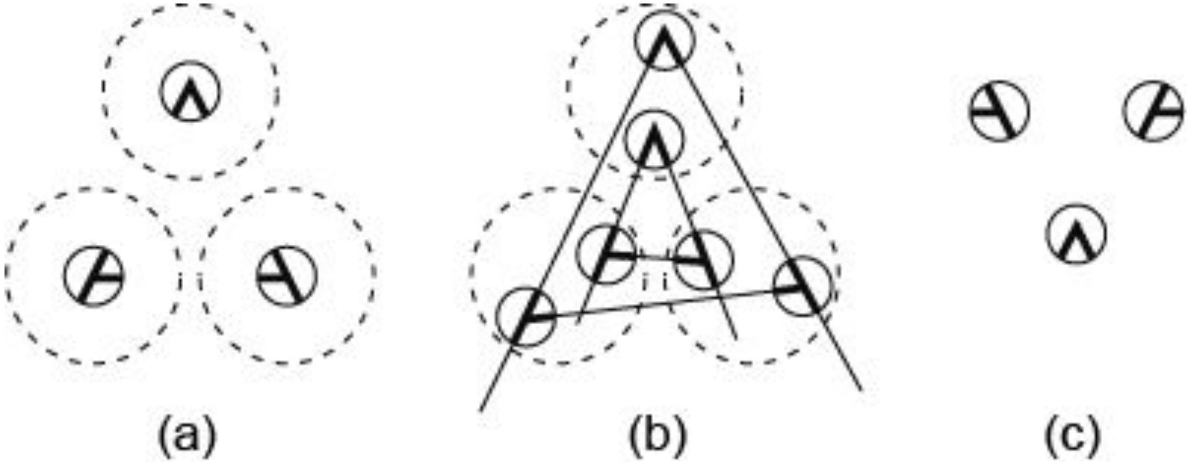
Pooling over feature maps yields latent object representations





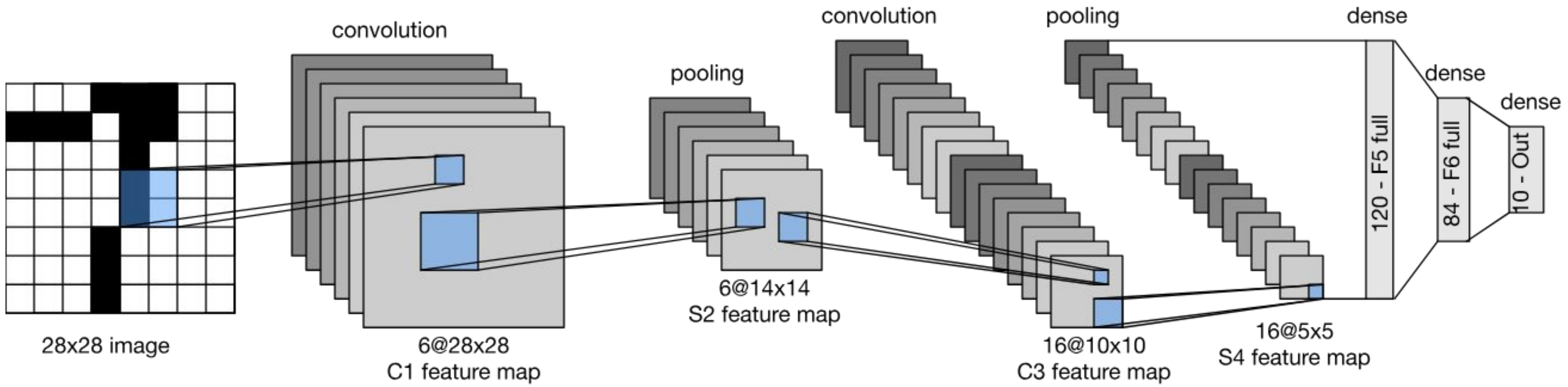
http://www.scholarpedia.org/article/Neocognitron

Invariance to small translation confers invariance to distortion

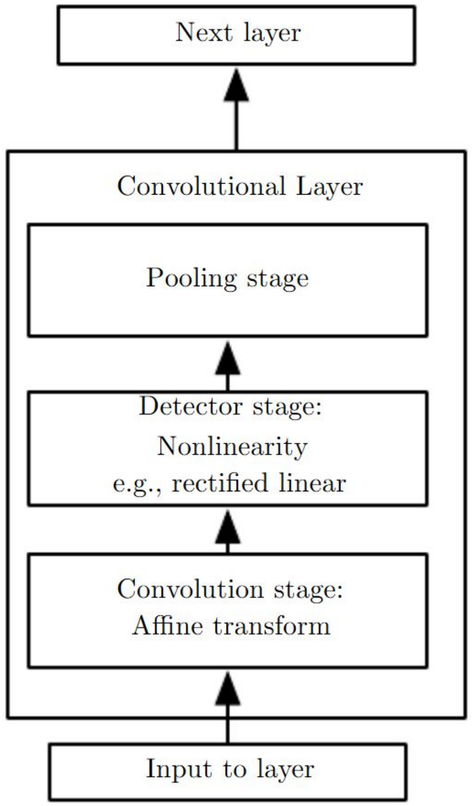


http://www.scholarpedia.org/article/Neocognitron

From Neocognitron to CNN (LeCun's LeNet, 1989)



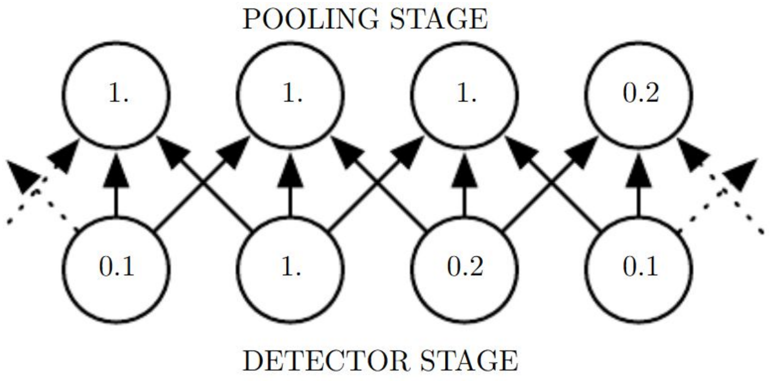
A *convolutional layer* may apply multiple nonlinearities

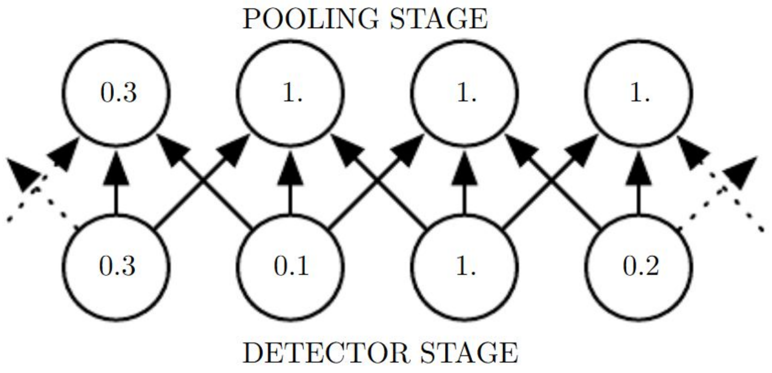


Pooling stage

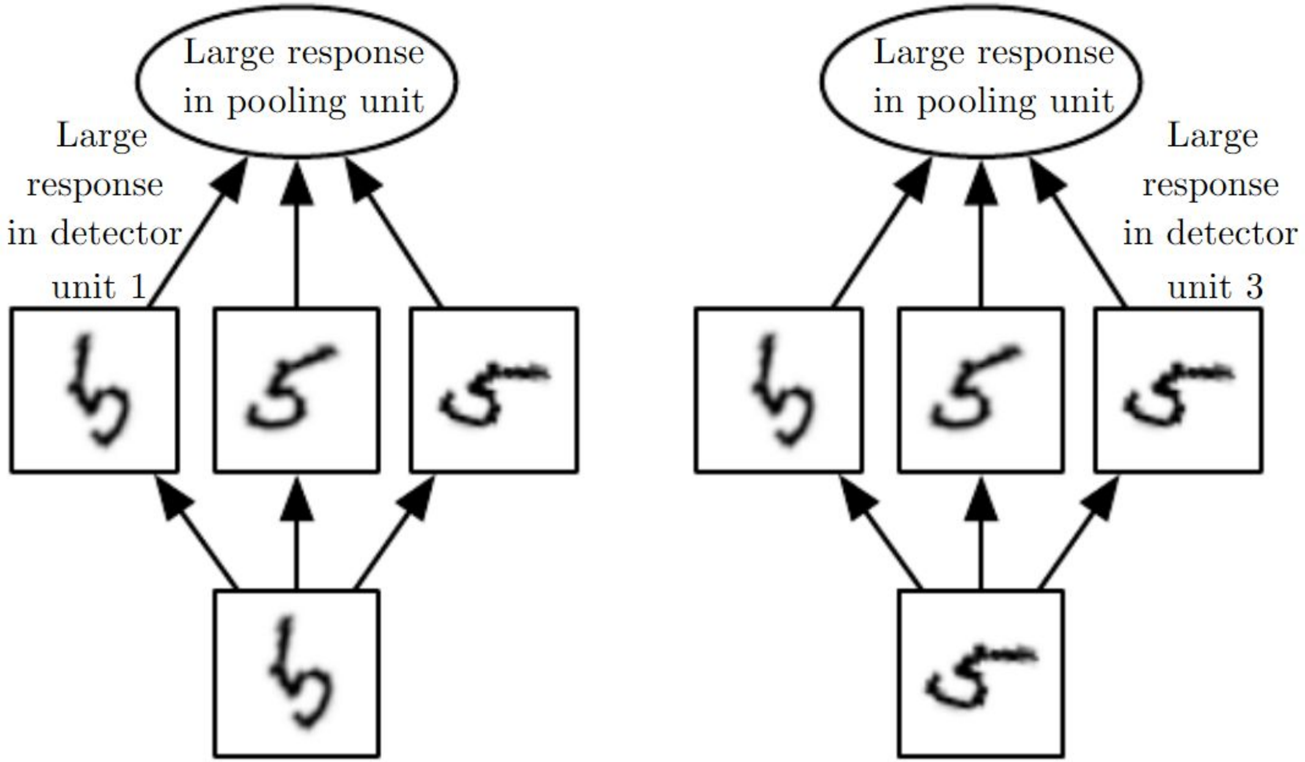
- Summarizes nearby outputs
- Average pooling, L2 norm, Max pooling

Pooling across spatial regions induces some translation *invariance*

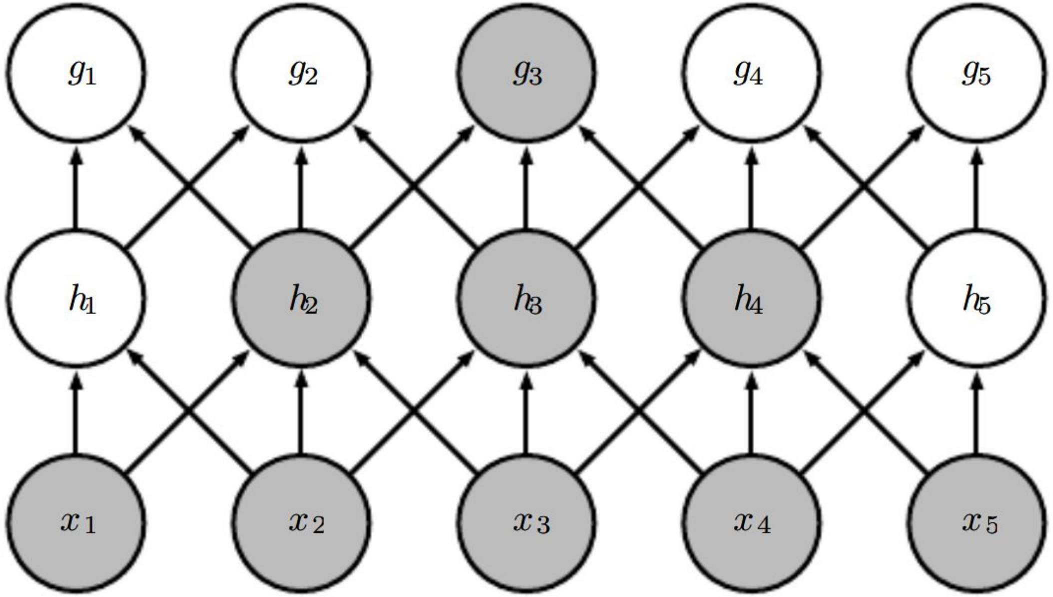




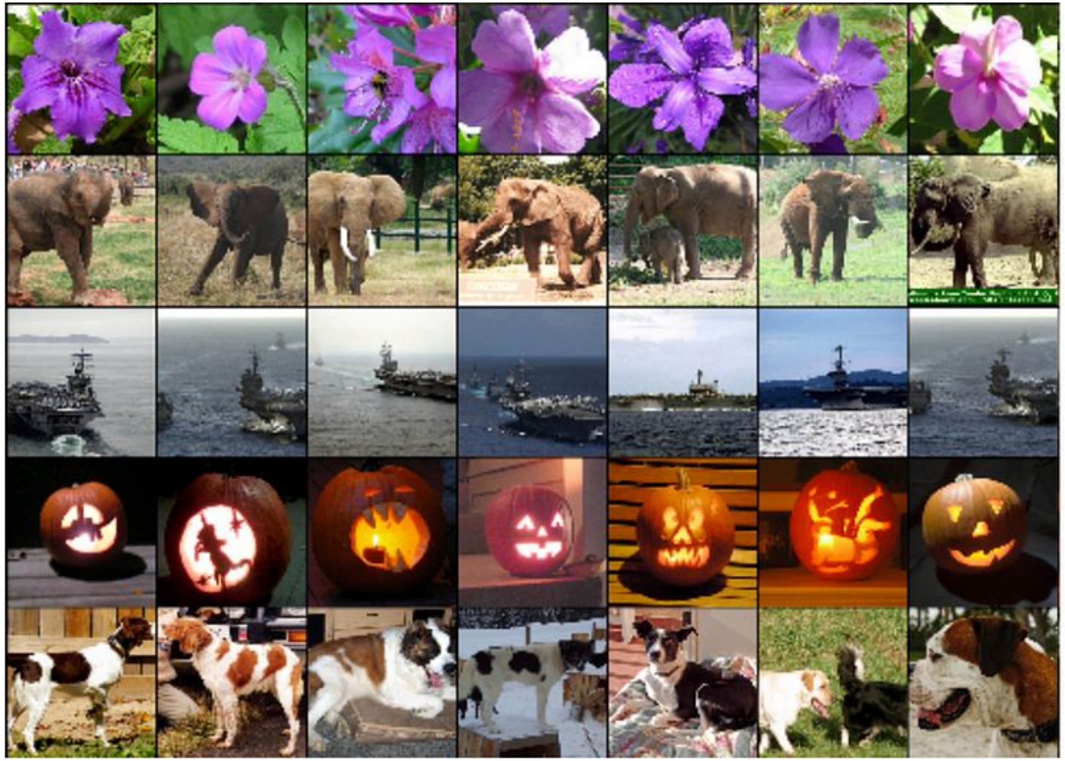
Pooling stages summarize their inputs


Large response in pooling unit
Large response in detector unit 1
Large response in pooling unit
Large response in detector unit 3

Deeper layers have indirect interactions with most of the input

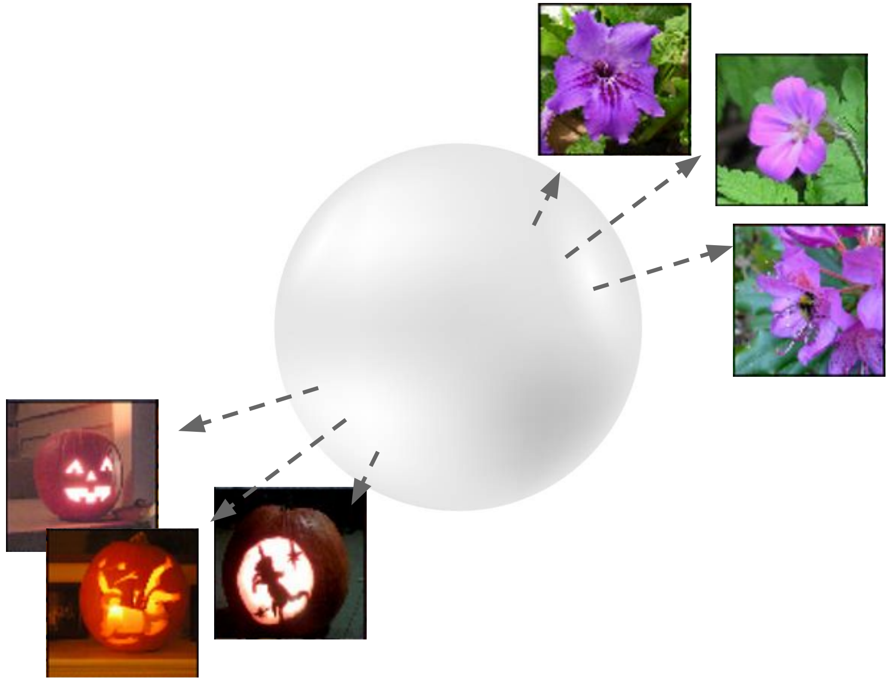


CNNs learn a feature space where inner products are meaningful



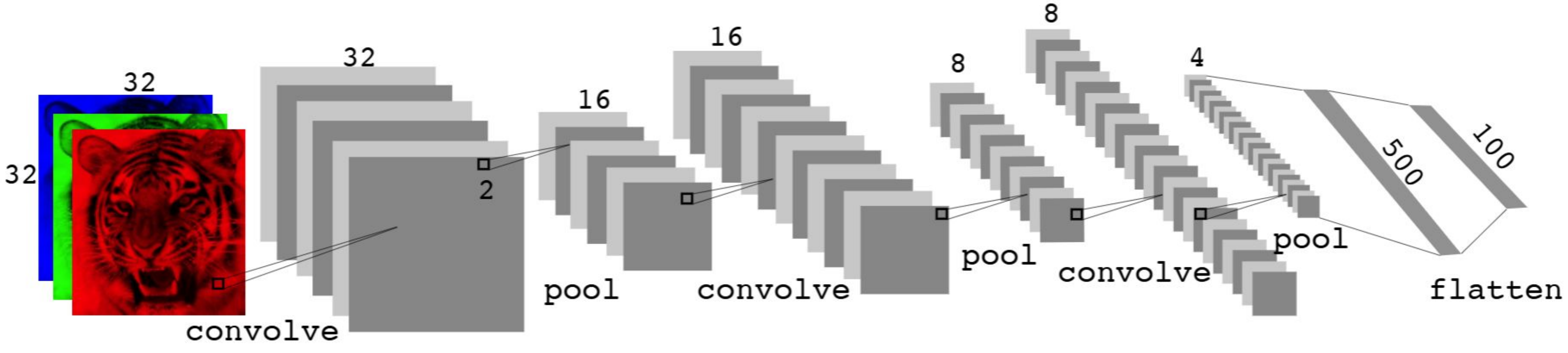
https://cacm.acm.org/research/imagenet-classification-with-deep-convolutional-neural-networks/

CNNs learn a feature space where inner products are meaningful



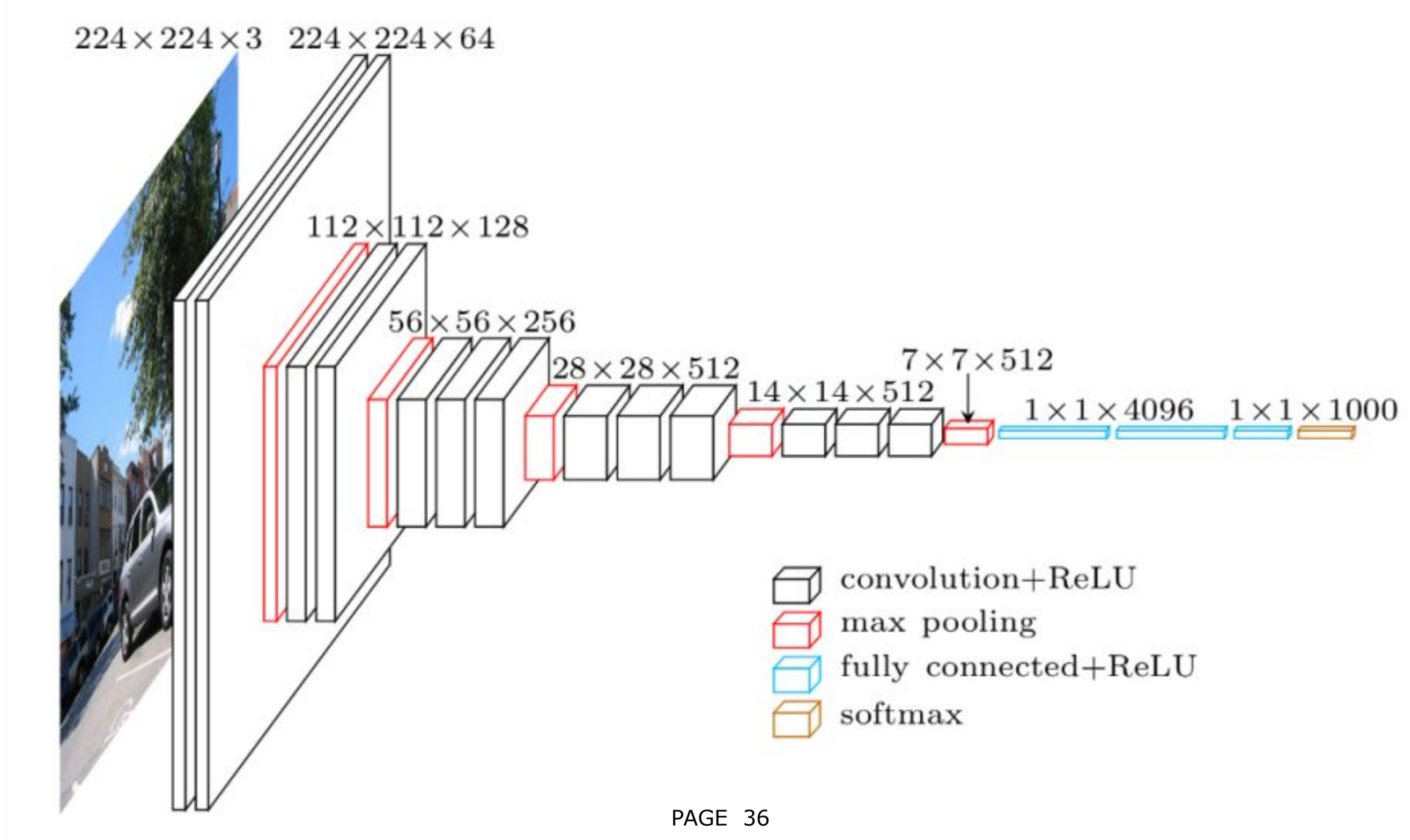
https://cacm.acm.org/research/imagenet-classification-with-deep-convolutional-neural-networks/

Interpreting the architecture of a CNN



Top: Introduction to Statistical Learning Section 10.3

Interpreting the architecture of a CNN



Now that we're at the end of the lecture, you should be able to…

- ★ Compute the grayscale value of an image pixel through **cross-correlation** of a filter and an image.
- ★ Give examples of **low-level filters** used in a CNN for image processing and define their effect with reference to **convolution**.
- ★ Define and explain the role of **parameter sharing** and **local connections** in CNNs.
- ★ Construct a convolutional layer using **convolution, nonlinearity** and **pooling** operations.
- ★ Describe the information processing occurring at each layer of a **Convolutional Neural Network (CNNs)**, given a schematic of its architecture.
- ★ Sketch the **architecture of a CNN** given its verbal description.
- ★ Determine the number of trainable parameters in a convolutional layer given the hyperparameters to set **# of filters, kernel size, padding, stride**.