Convolutional Neural Networks

Danna Gurari
University of Colorado Boulder
Spring 2022

https://home.cs.colorado.edu/~DrG/Courses/NeuralNetworksAndDeepLearning/AboutCourse.html

Review

- Last class:
  - Universal approximation theorem
  - Selecting model capacity: avoid overfitting and underfitting
  - Selecting model hyperparameters
  - Learning efficiently: optimization methods
  - Programming tutorial

- Assignments (Canvas):
  - Lab assignment 1 due earlier today
  - Problem set 2 due next week

- Questions?

Today’s Topics

- Neural Networks for Spatial Data
- History of Convolutional Neural Networks (CNNs)
- CNNs – Convolutional Layers
- CNNs – Pooling Layers

Today's Topics

- Neural Networks for Spatial Data
- History of Convolutional Neural Networks (CNNs)
- CNNs – Convolutional Layers
- CNNs – Pooling Layers

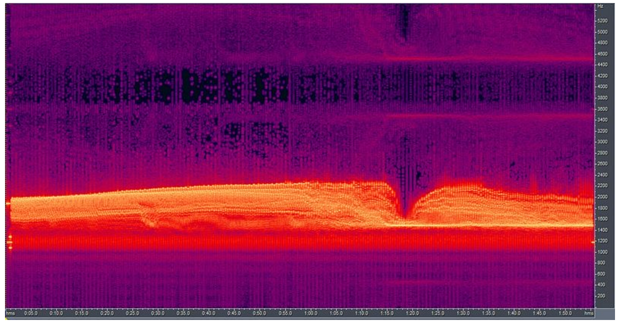
What is Spatial Data?

- Data where the order matters; e.g.,

2D

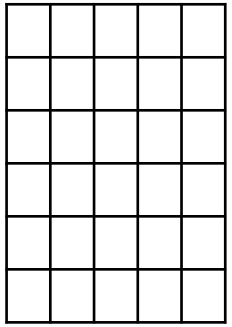
Images

Audio (spectrogram)



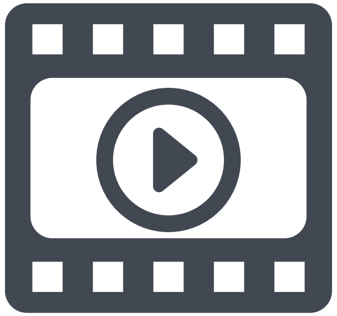
Text (word embeddings)

I
like
learning
about
deep
learning



3D

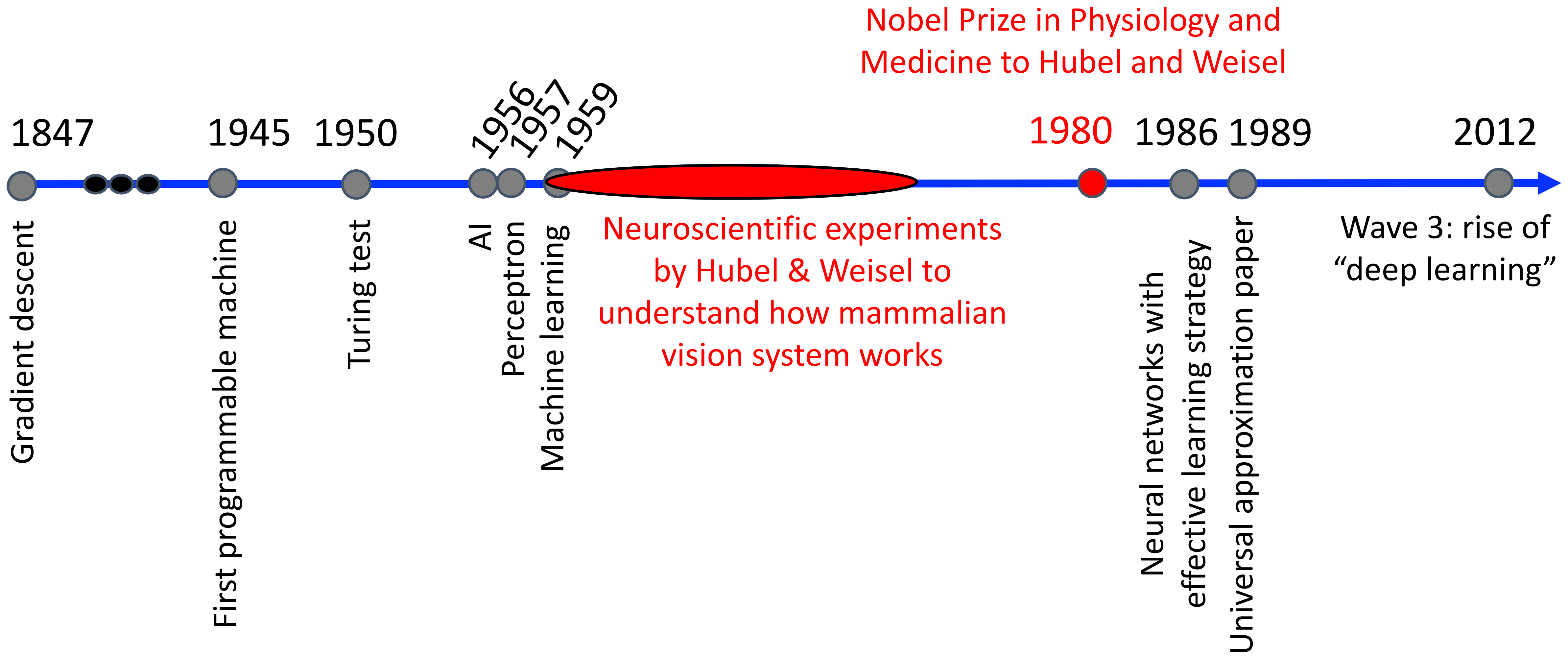
Video



Today’s Topics

- Neural Networks for Spatial Data
- History of Convolutional Neural Networks (CNNs)
- CNNs – Convolutional Layers
- CNNs – Pooling Layers

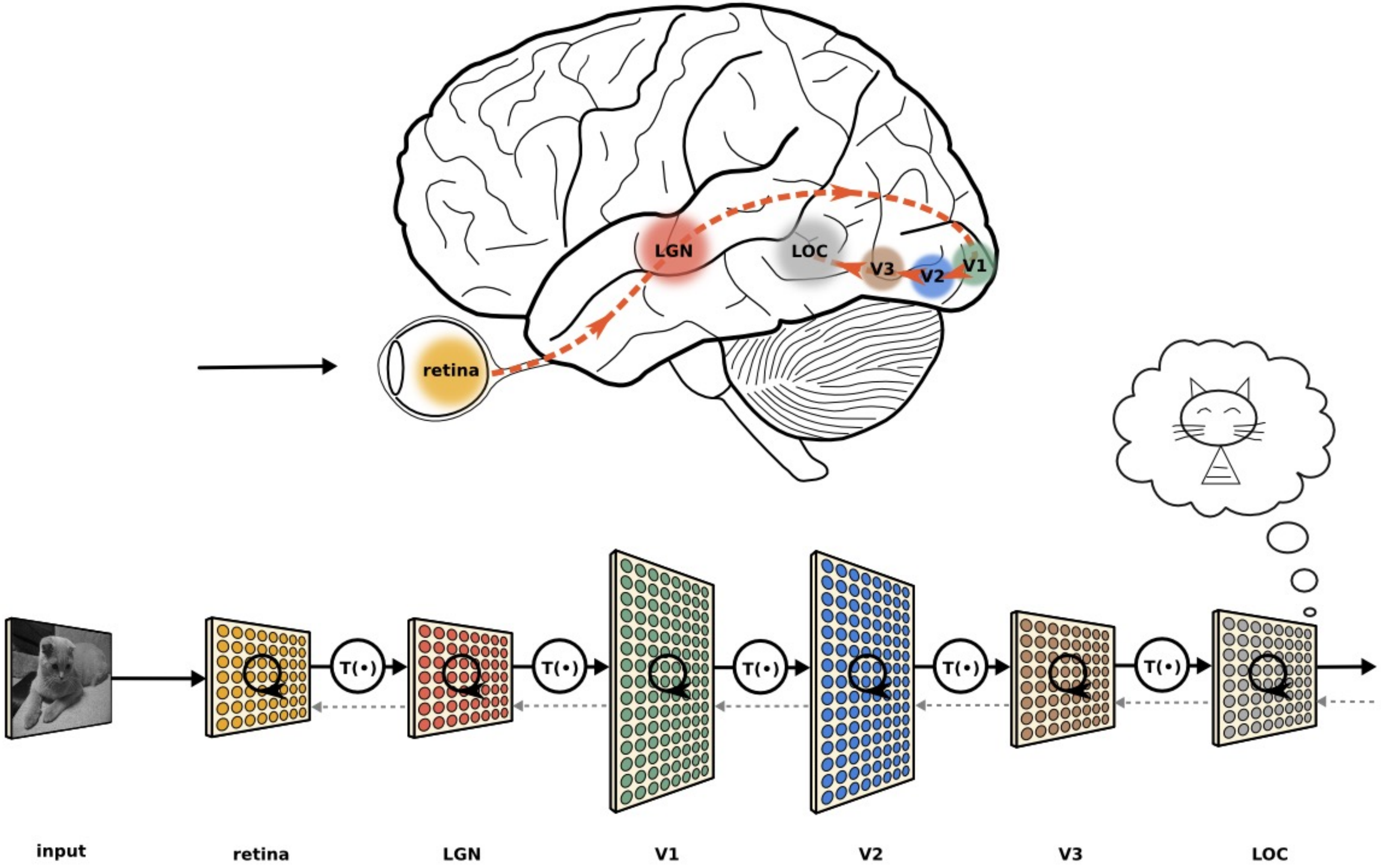
Historical Context: Inspiration



Motivation: How Vision System Works

Image Source: https://braintour.harvard.edu/archives/portfolio-items/hubel-and-wiesel

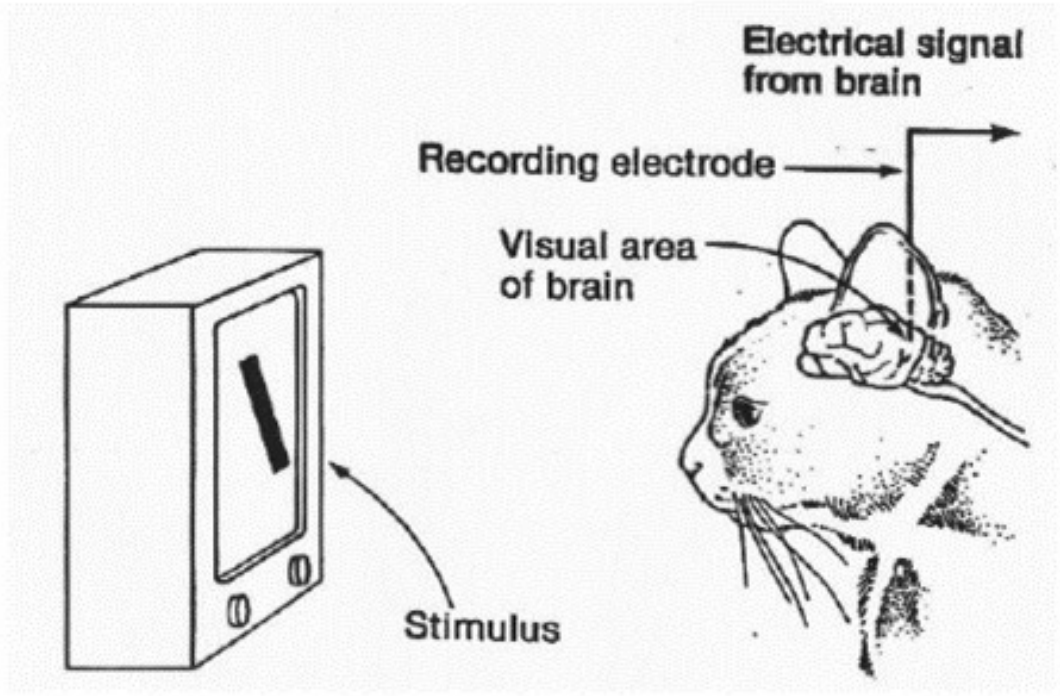
Motivation: How Vision System Works



https://neuwritesd.files.wordpress.com/2015/10/visual_stream_small.png

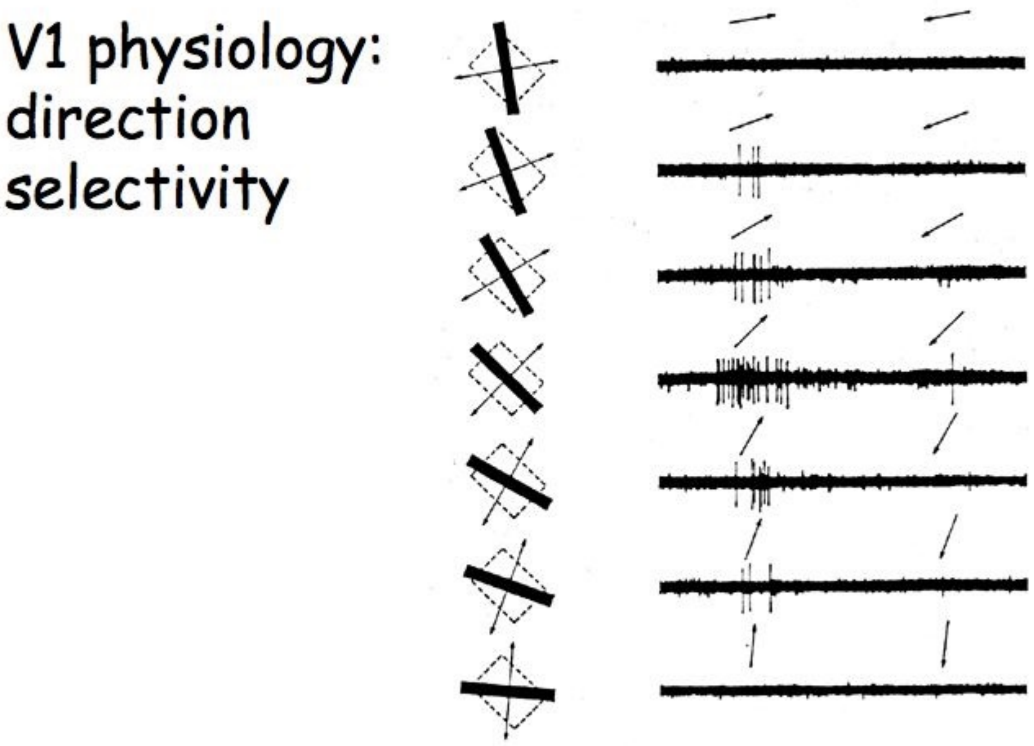
Motivation: How Vision System Works

Experiment Set-up:



https://www.esantus.com/blog/2019/1/31/convolutional-neural-networks-a-quick-guide-for-newbies

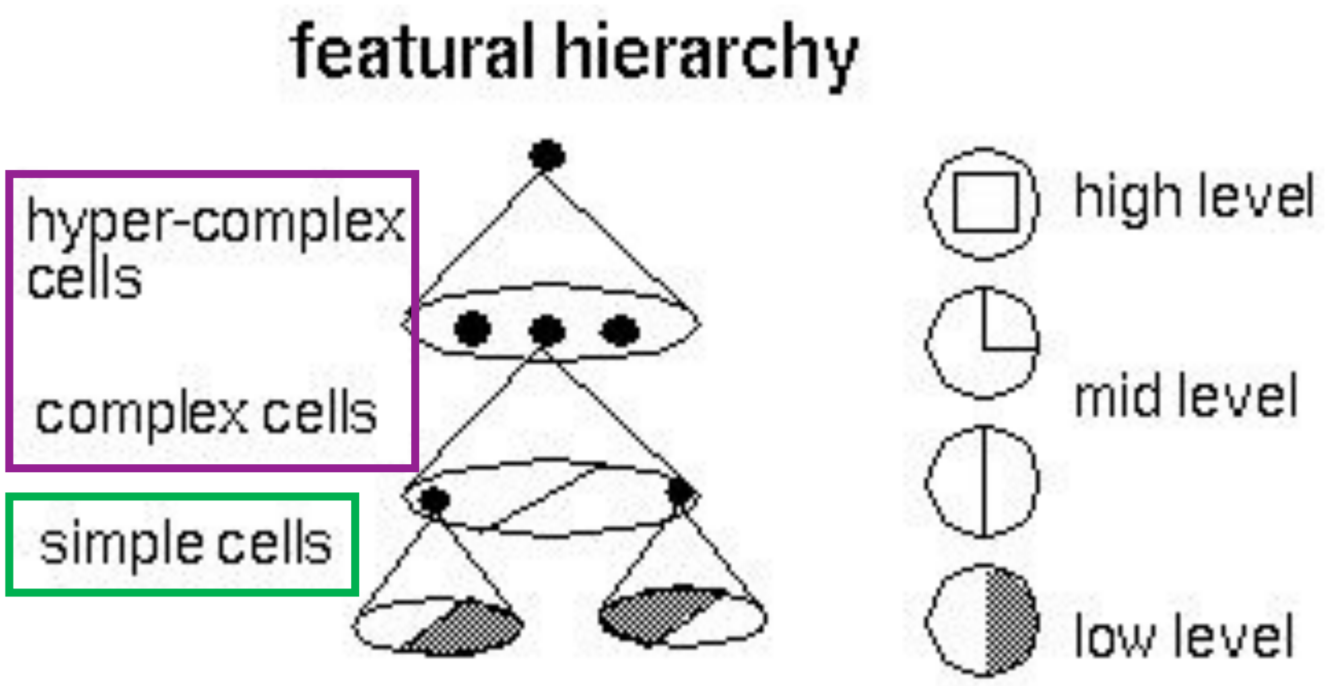
Key Finding: initial neurons responded strongly only when light was shown in certain orientations



https://www.cns.nyu.edu/~david/courses/perception/lecturenotes/V1/lgn-V1.html

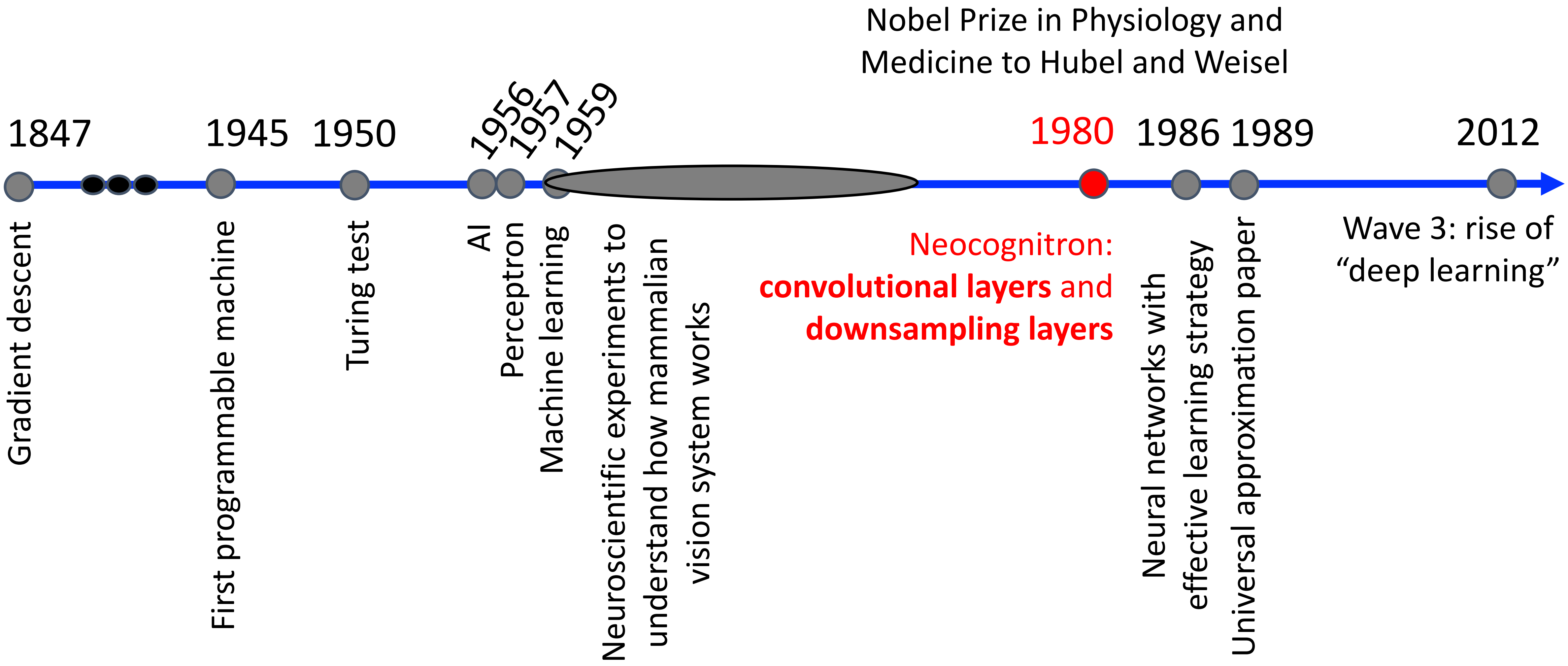
Motivation: How Vision System Works

Key Idea: cells are organized as a hierarchy of feature detectors, with higher level features responding to patterns of activation in lower level cells



Source: https://bruceoutdoors.files.wordpress.com/2017/08/hubel.jpg

Historical Context: Key Ingredients



Neocognitron: Key Ingredients

http://personalpage.flsi.or.jp/fukushima/index-e.html

"In this paper, we discuss how to synthesize a neural network model in order to endow it an ability of pattern recognition like a human being... the network acquires a similar structure to the hierarchy model of the visual nervous system proposed by Hubel and Wiesel."

- Fukushima, Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. Biological Cybernetics, 1980.

Neocognitron: Key Ingredients

Cascade of simple and complex cells:

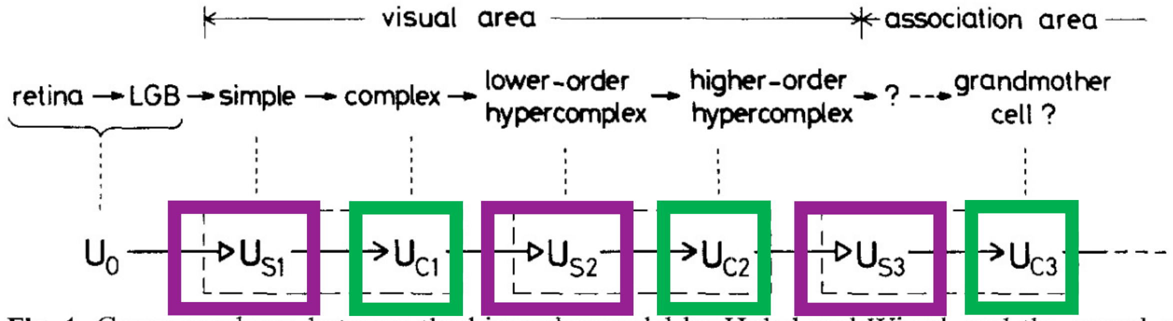


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron

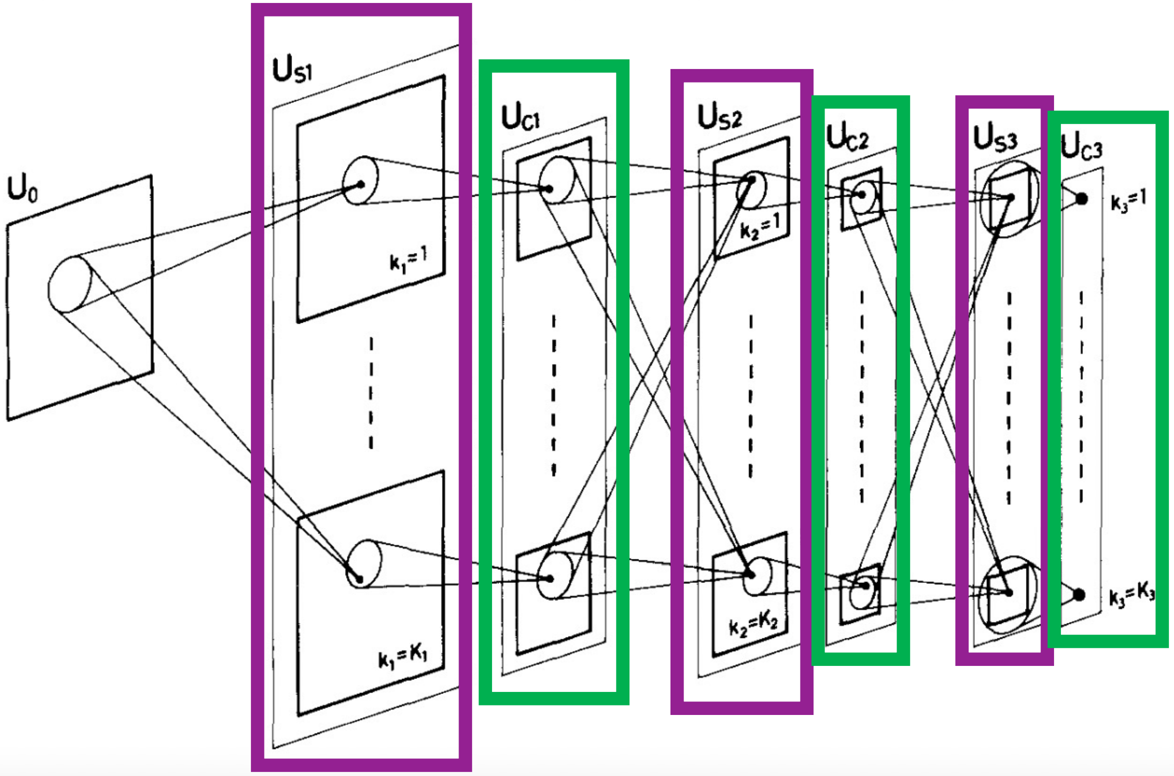
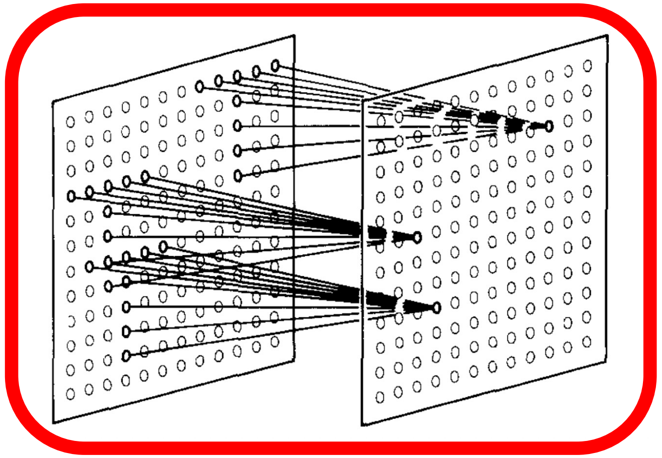


Fig. 2. Schematic diagram illustrating the interconnections between layers in the neocognitron

Fukushima, 1980.

Neocognitron: Key Ingredients

Simple cells extract local features using a sliding filter:



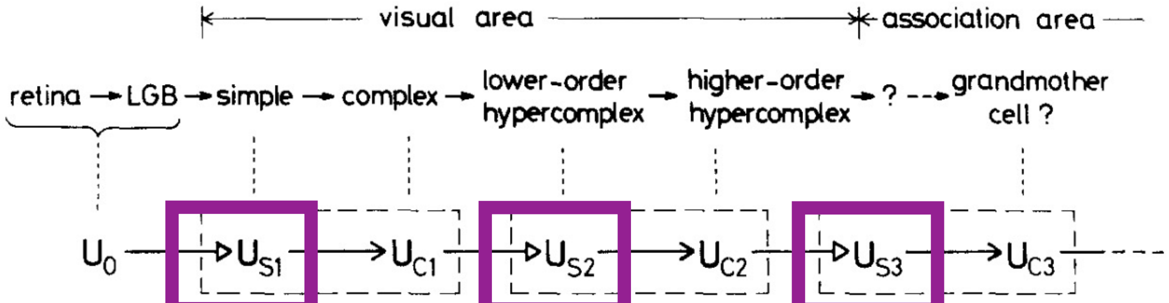


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron

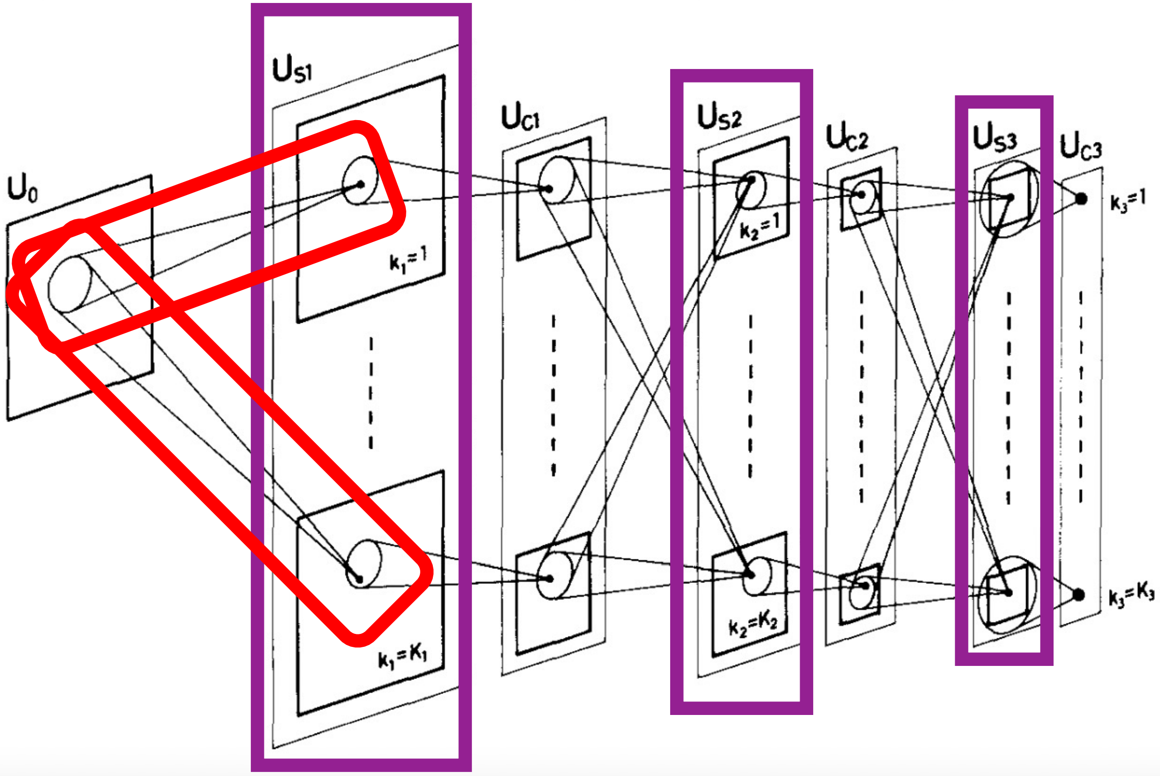


Fig. 2. Schematic diagram illustrating the interconnections between layers in the neocognitron

Fukushima, 1980.

Neocognitron: Key Ingredients

Complex cells fire when any part of the local region is the desired pattern

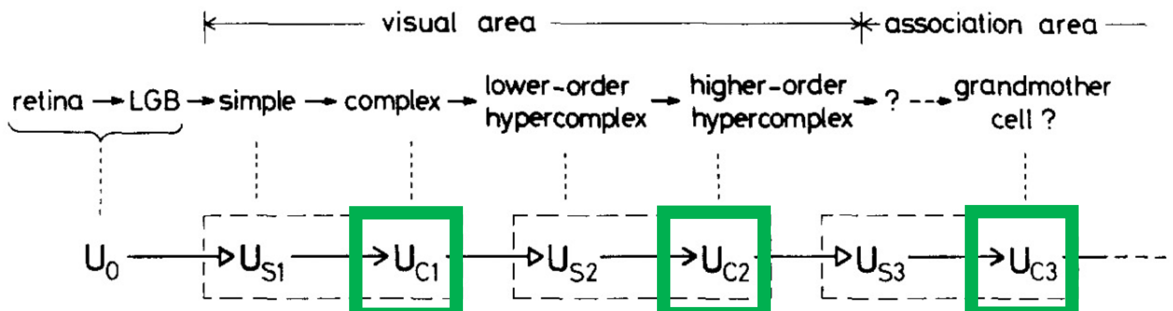


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron

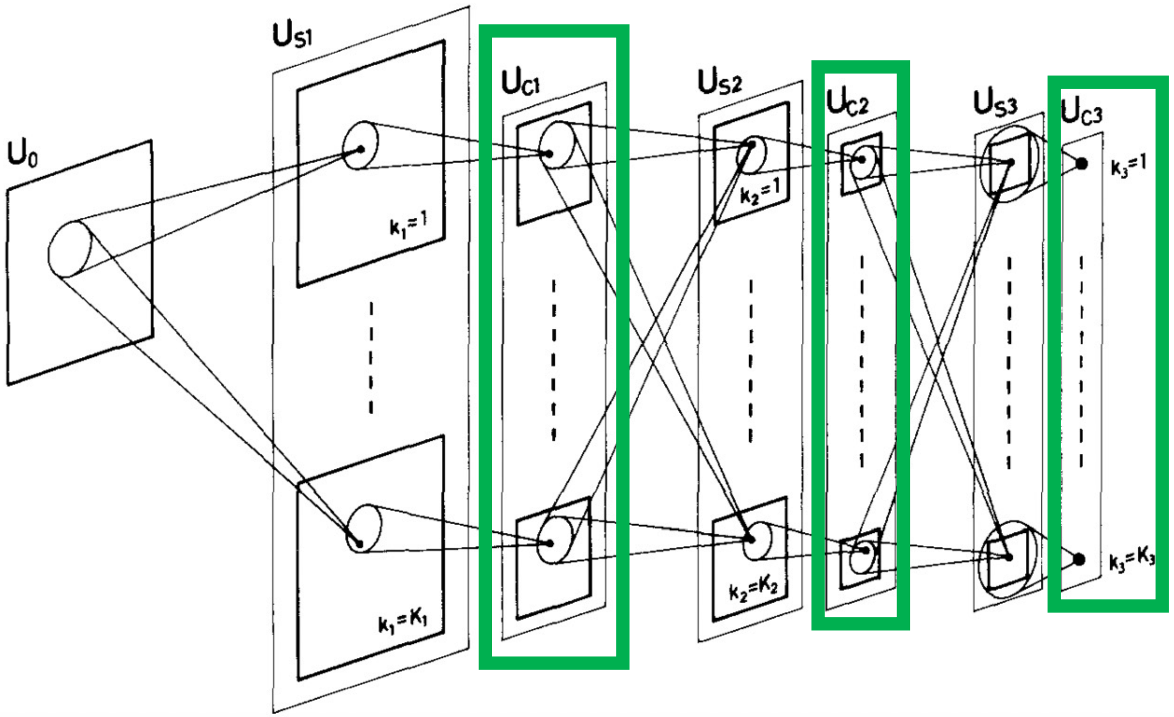


Fig. 2. Schematic diagram illustrating the interconnections between layers in the neocognitron

Fukushima, 1980.

Neocognitron: Key Ingredients

1. ~ Convolutional layers

modifiable synapses

unmodifiable synapses

2. ~ Pooling Layers

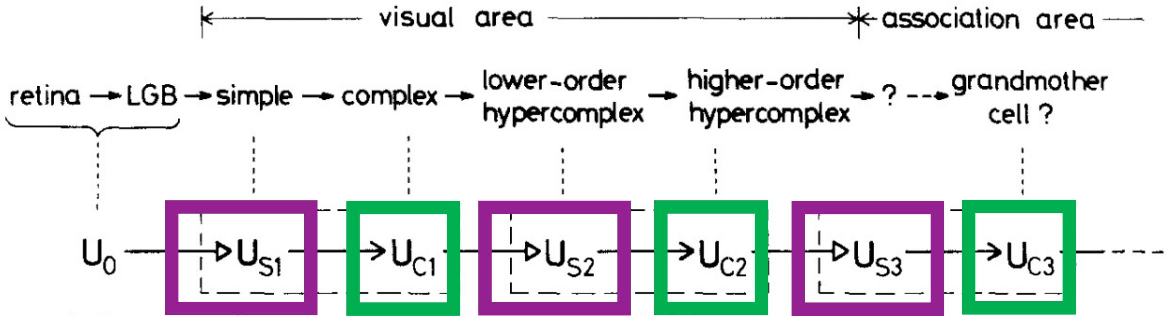


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron

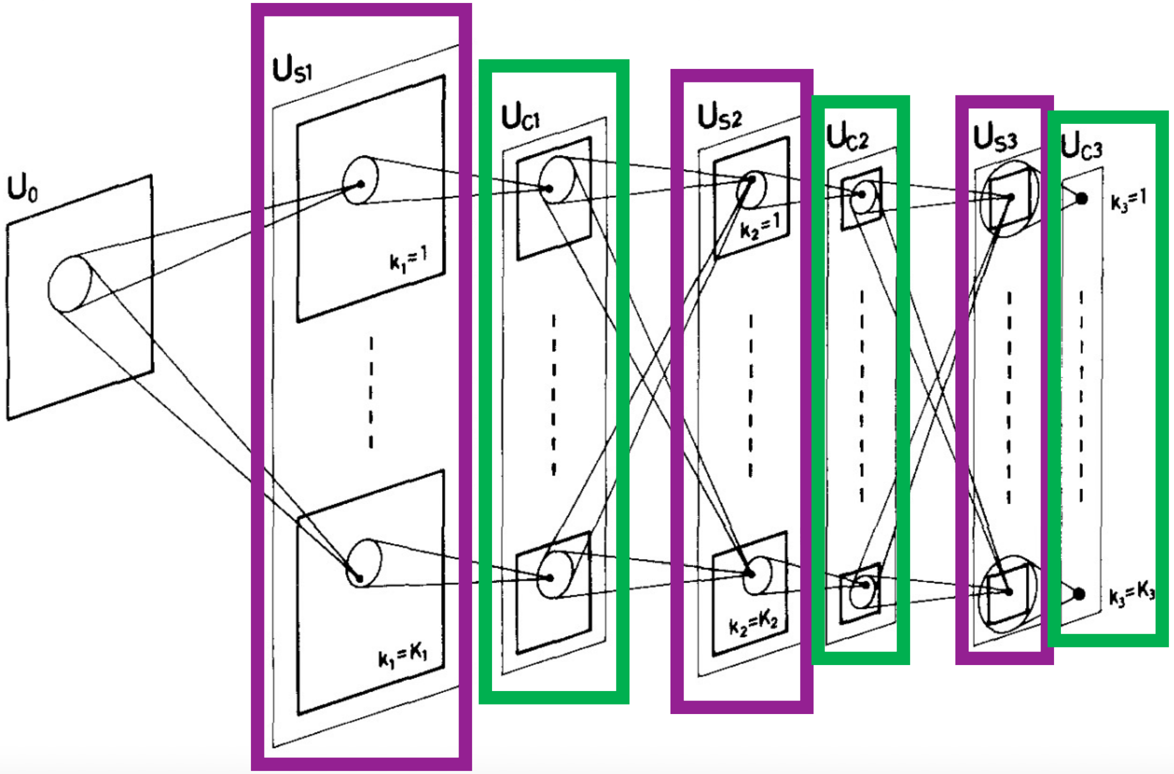


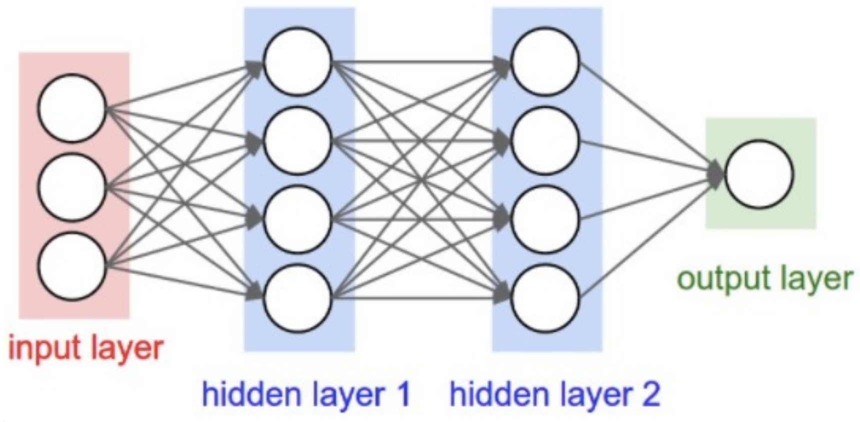
Fig. 2. Schematic diagram illustrating the interconnections between layers in the neocognitron

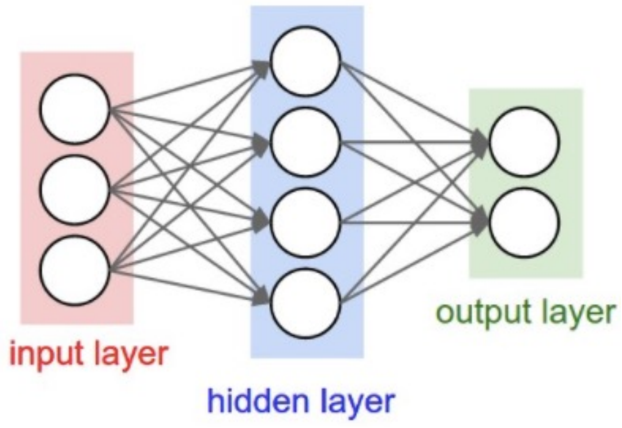
Fukushima, 1980.

Today’s Topics

- Neural Networks for Spatial Data
- History of Convolutional Neural Networks (CNNs)
- CNNs – Convolutional Layers
- CNNs – Pooling Layers

Motivation: Fully-Connected Layers Are Limited

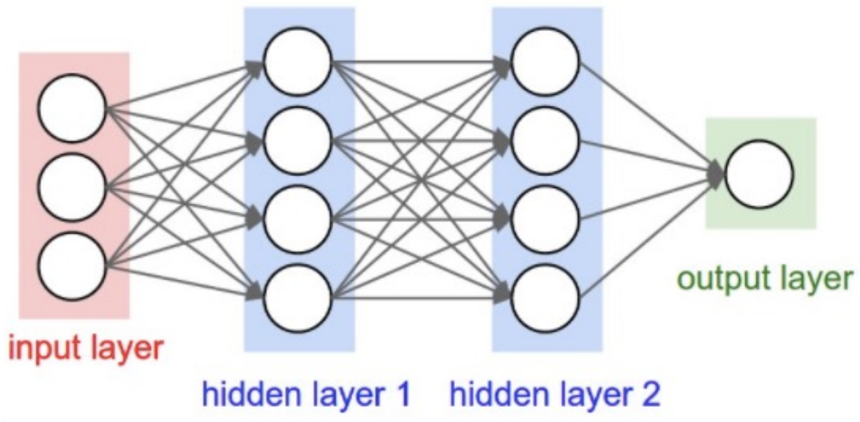


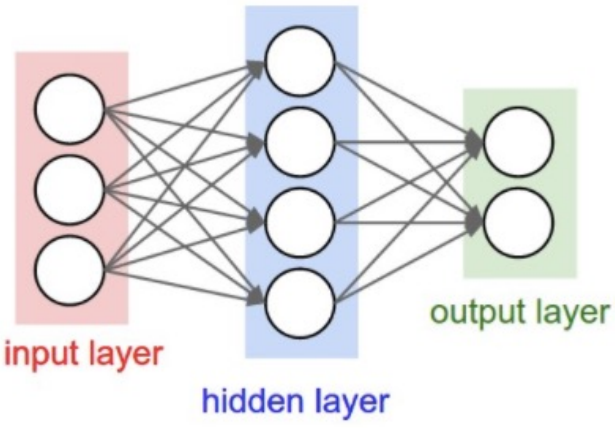


Each node provides input to each node in the next layer

Figure Source: http://cs231n.github.io/neural-networks-1/

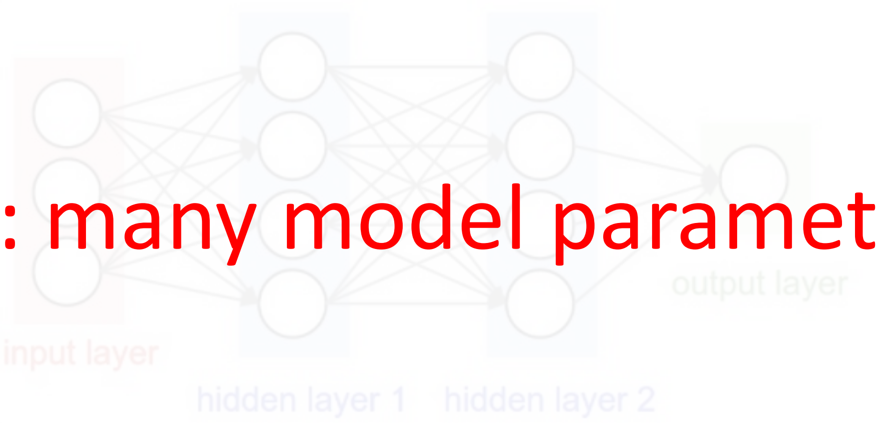
Motivation: Fully-Connected Layers Are Limited

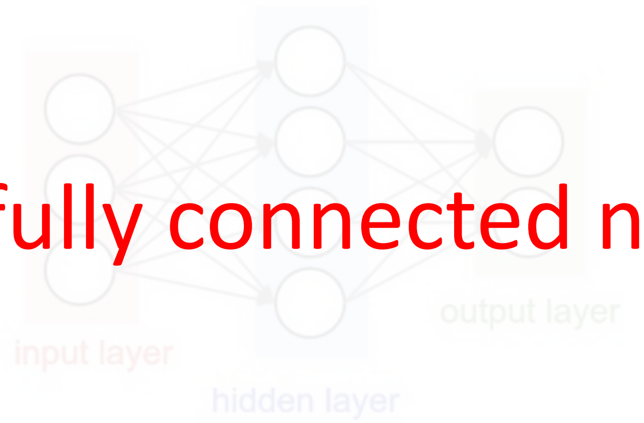




- Assume 2 layer model with 100 nodes per layer
  - e.g., how many weights are in a 640x480 image?
    - 640x480x3x100 + 100x100 + 100x1 = 92,170,100
  - e.g., how many weights are in a 2048X1536 image (3.1 Megapixel image)?
    - 2048x1536x3x100 + 100x100 + 100x1 = 943,728,500

Motivation: Fully-Connected Layers Are Limited

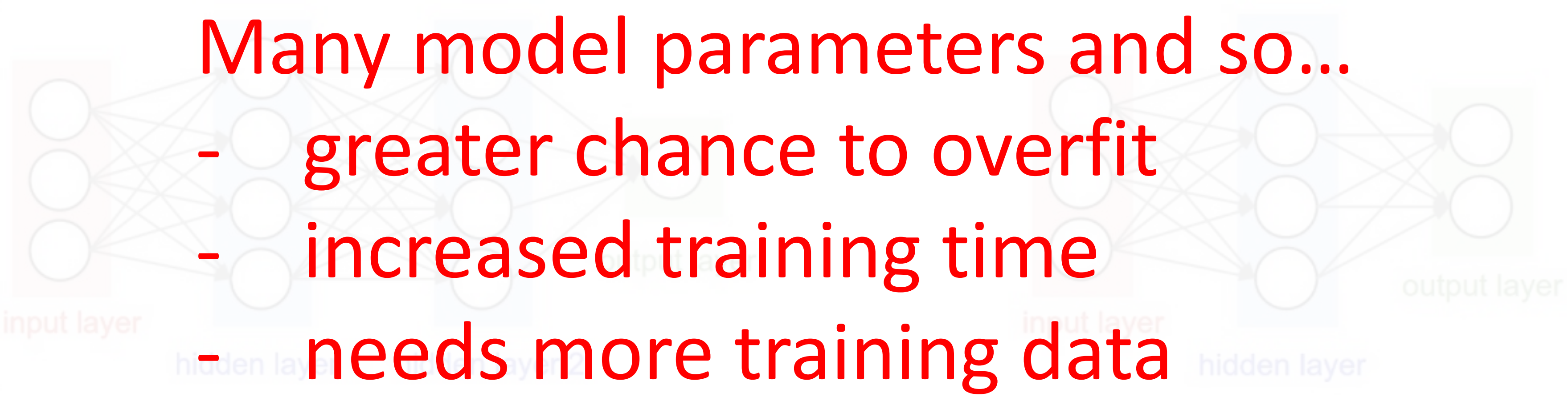




Issue: many model parameters in fully connected networks

- Assume 2 layer model with 100 nodes per layer
  - e.g., how many weights are in a 640x480 image?
    - 640x480x3x100 + 100x100 + 100x1 = 92,170,100
  - e.g., how many weights are in a 2048X1536 image (3.1 Megapixel image)?
    - 2048x1536x3x100 + 100x100 + 100x1 = 943,728,500

Motivation: Fully-Connected Layers Are Limited

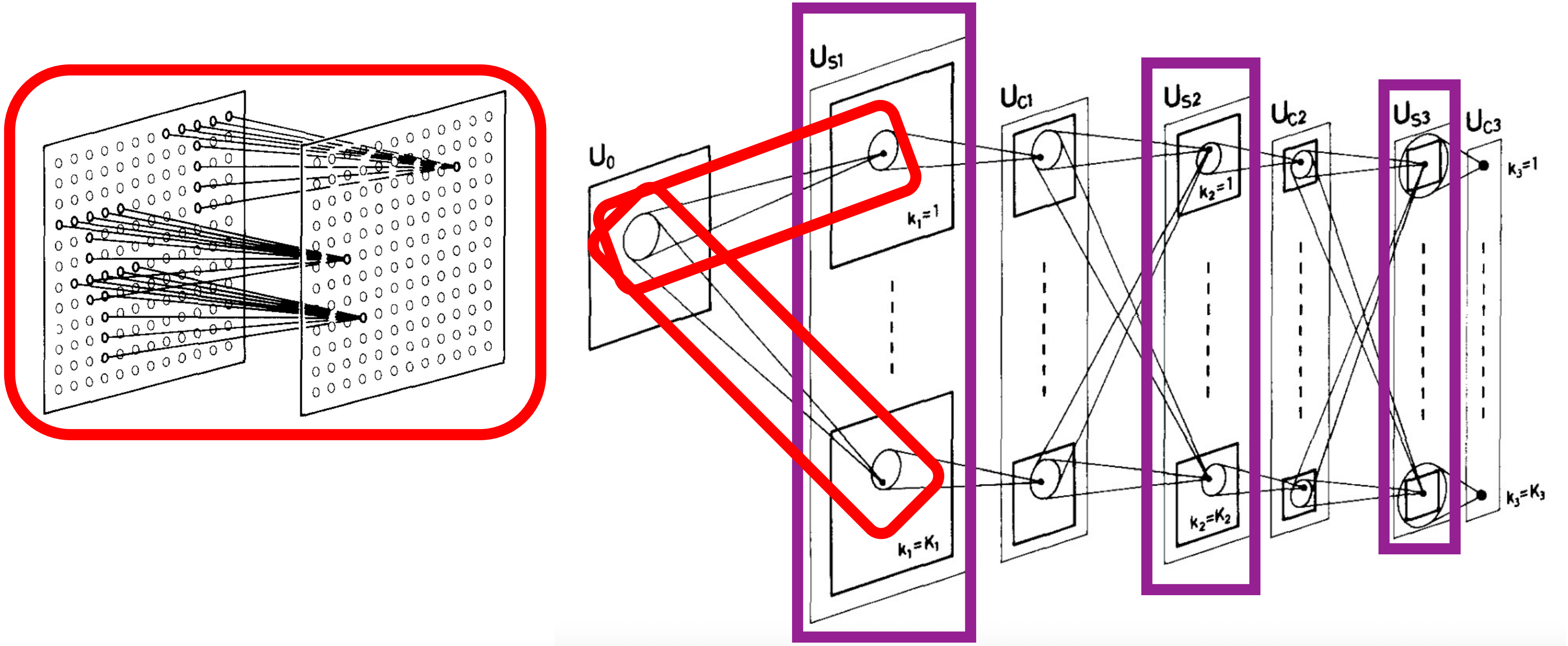


Many model parameters and so…
- greater chance to overfit
- increased training time
- needs more training data

- Assume 2 layer model with 100 nodes per layer
  - e.g., how many weights are in a 640x480 image?
    - 640x480x3x100 + 100x100 + 100x1 = 92,170,100
  - e.g., how many weights are in a 2048X1536 image (3.1 Megapixel image)?
    - 2048x1536x3x100 + 100x100 + 100x1 = 943,728,500

Convolutional Layer (Recall Neocognitron)

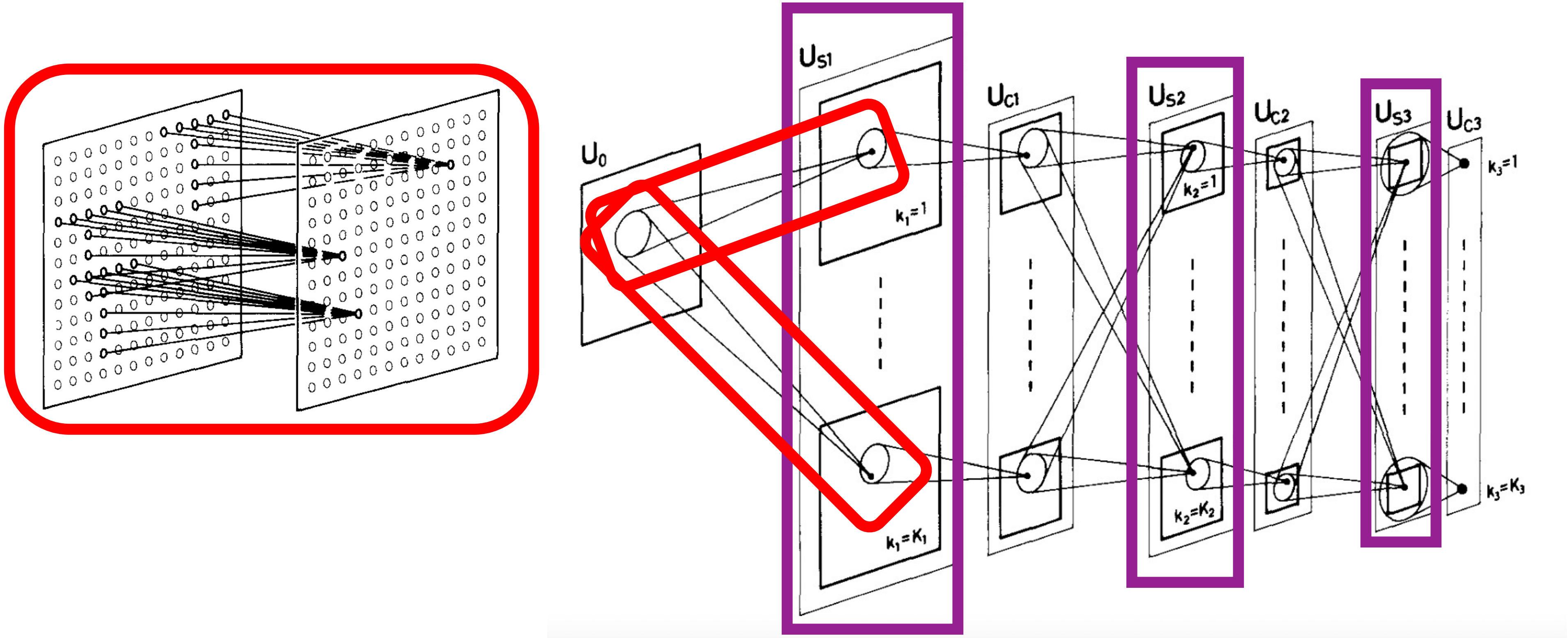
Idea: each node receives input only from a small neighborhood in previous layer and parameter sharing



Fukushima, 1980.

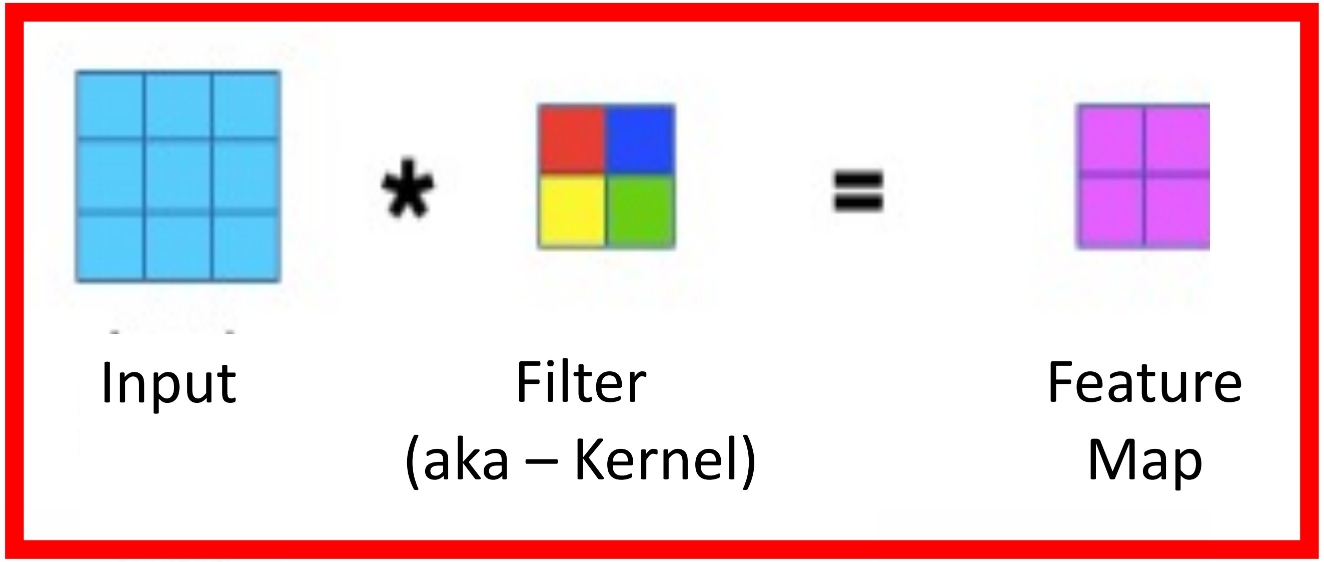
Convolutional Layer (Recall Neocognitron)

To do so, convolutions replace general matrix multiplication used in fully connected layers

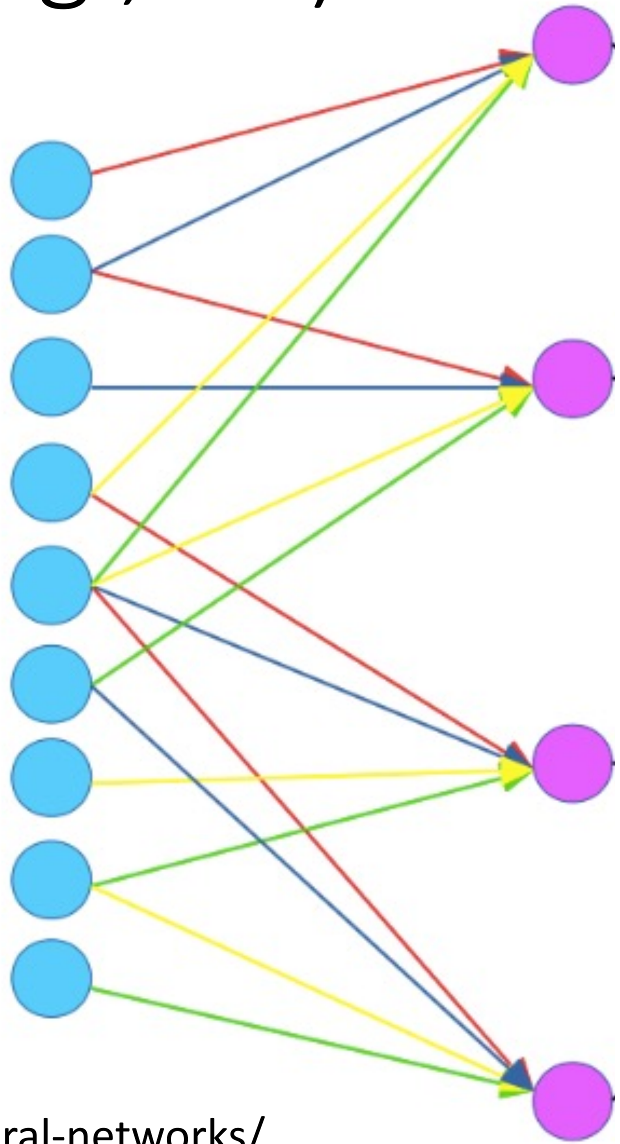


Fukushima, 1980.

Convolution: Applies Linear Filter (e.g., 2D)



Way to Interpret Neural Network

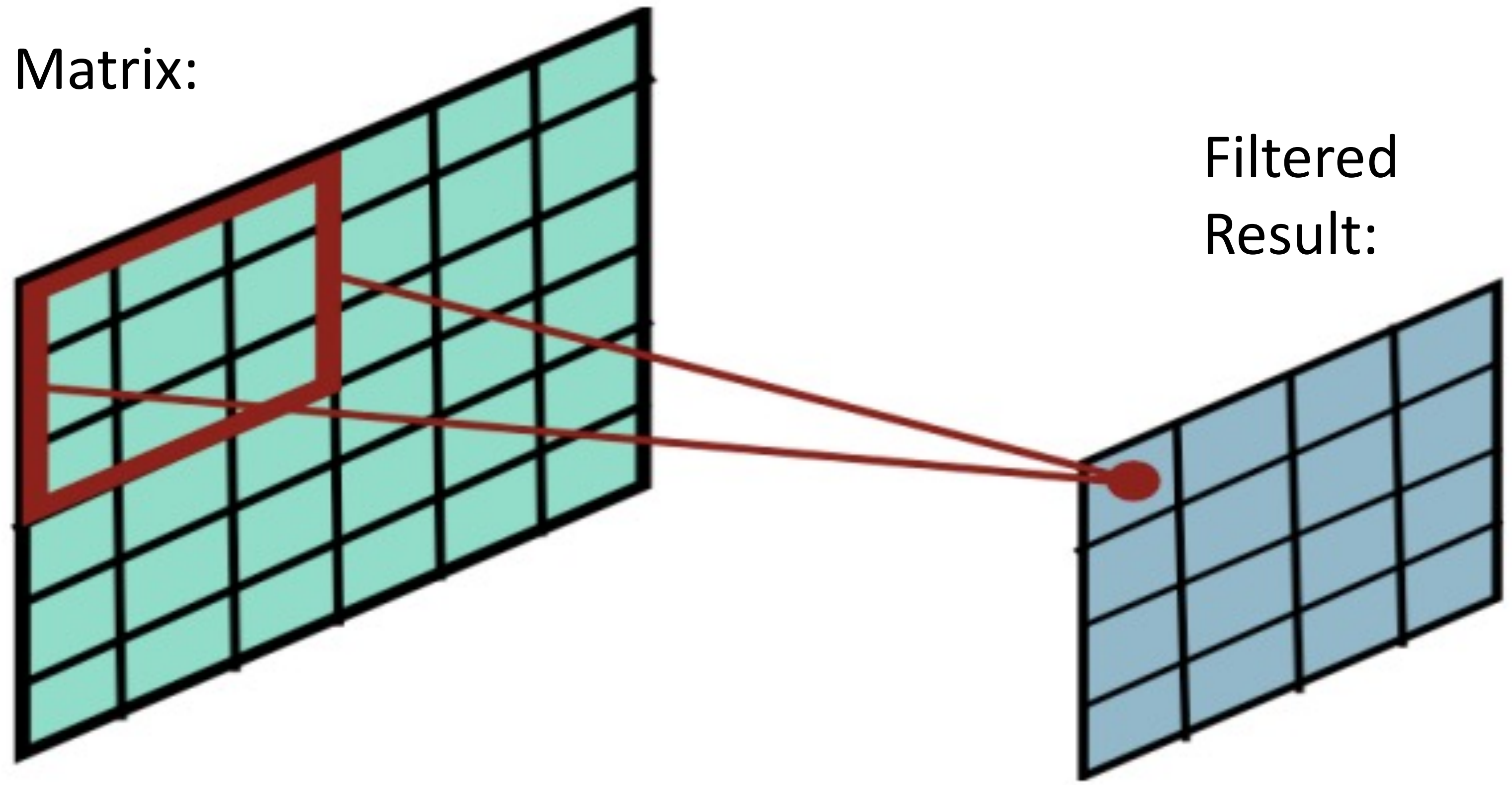


https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

2D Filtering

- Compute a function of local neighborhood for each location in matrix
- A filter specifies the function for how to combine neighbors' values

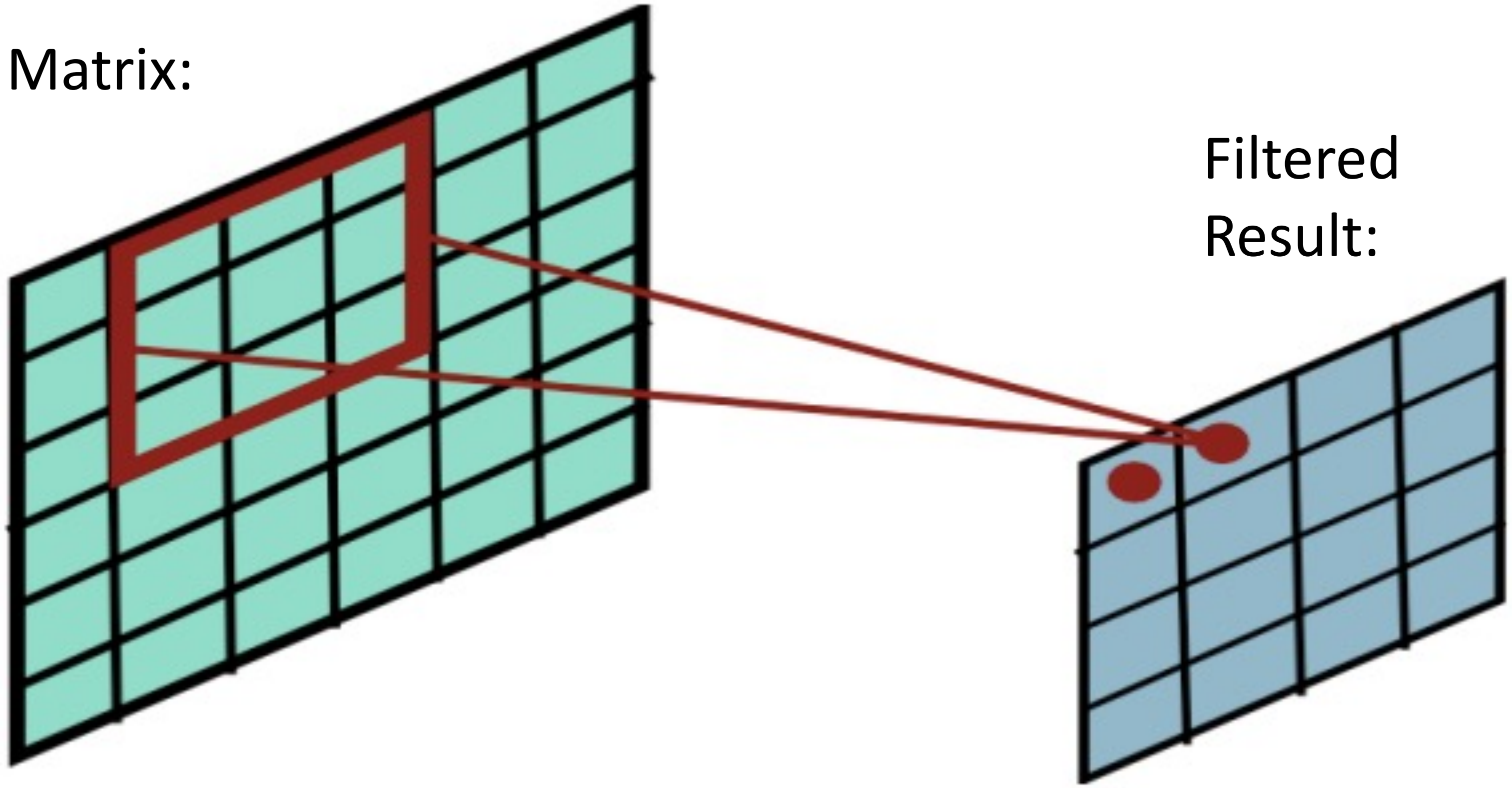
2D Filtering



Slides filter over the matrix and computes dot products

https://people.eecs.berkeley.edu/~jrs/189/lec/cnn.pdf

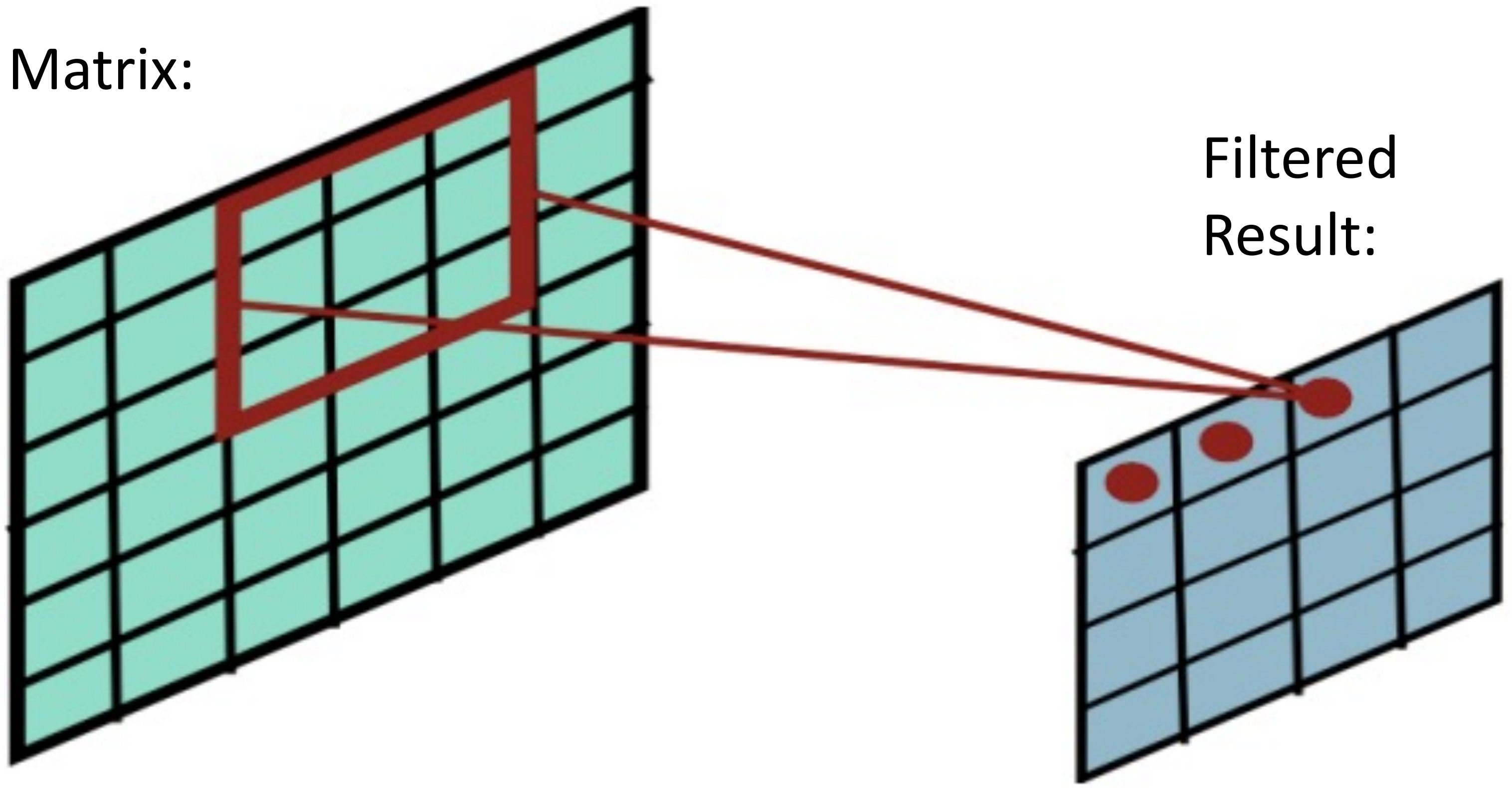
2D Filtering



Slides filter over the matrix and computes dot products

https://people.eecs.berkeley.edu/~jrs/189/lec/cnn.pdf

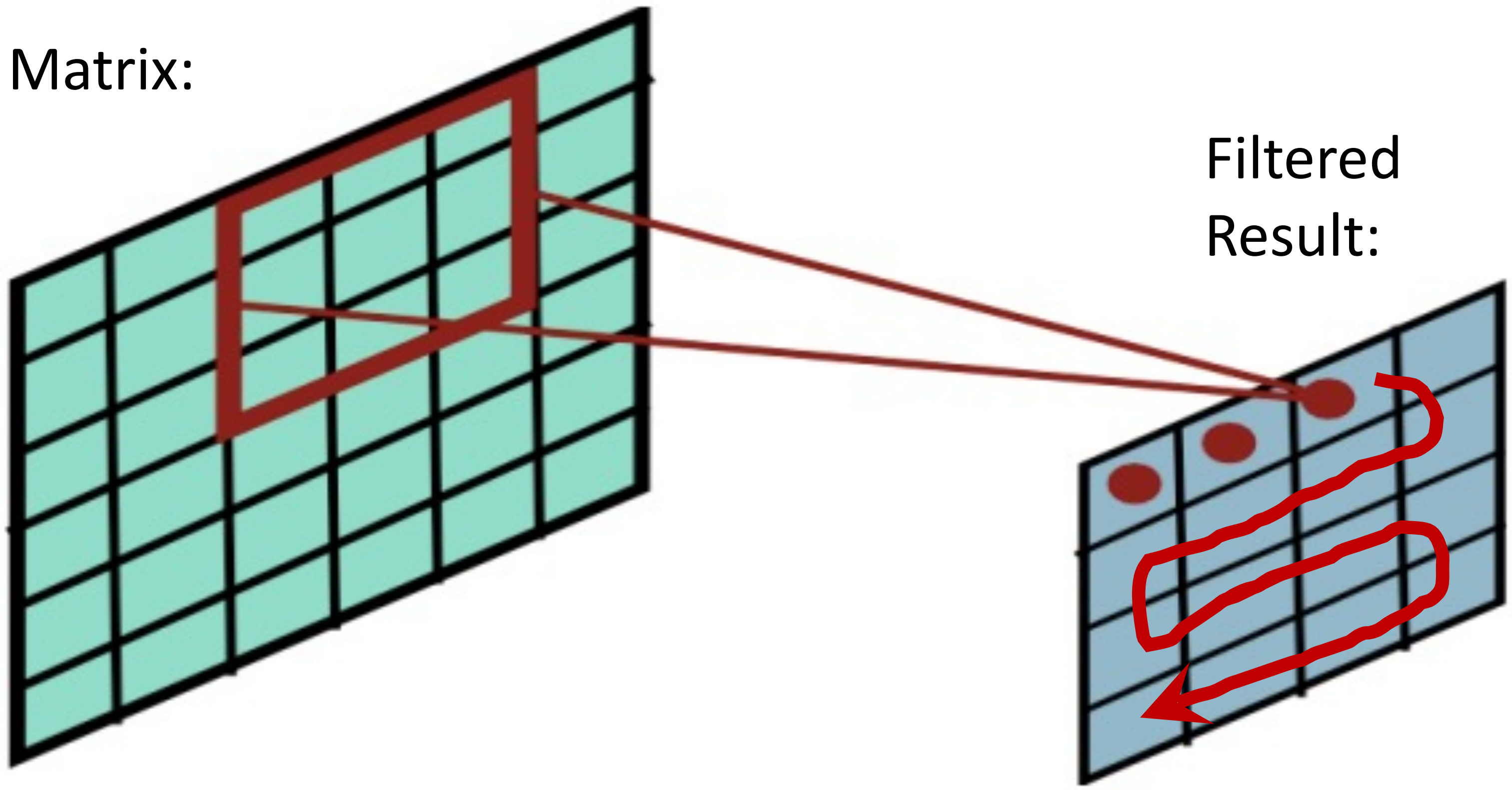
2D Filtering



Slides filter over the matrix and computes dot products

https://people.eecs.berkeley.edu/~jrs/189/lec/cnn.pdf

2D Filtering



Slides filter over the matrix and computes dot products

https://people.eecs.berkeley.edu/~jrs/189/lec/cnn.pdf

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

?	?	?
?	?	?
?	?	?

Dot Product = 1*1 + 1*0 + 1*1 + 0*0 + 1*1 + 1*0 + 0*1 + 0*1 + 0*0 + 0*0 + 1*1

Dot Product = 4

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	?	?
?	?	?
?	?	?

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	3	?
?	?	?
?	?	?

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	3	4
?	?	?
?	?	?

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	3	4
2	?	?
?	?	?

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	3	4
2	4	?
?	?	?

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	3	4
2	4	3
?	?	?

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	3	4
2	4	3
2	?	?

2D Filtering: Toy Example

Input

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	3	4
2	4	3
2	3	?

2D Filtering: Toy Example

Input

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

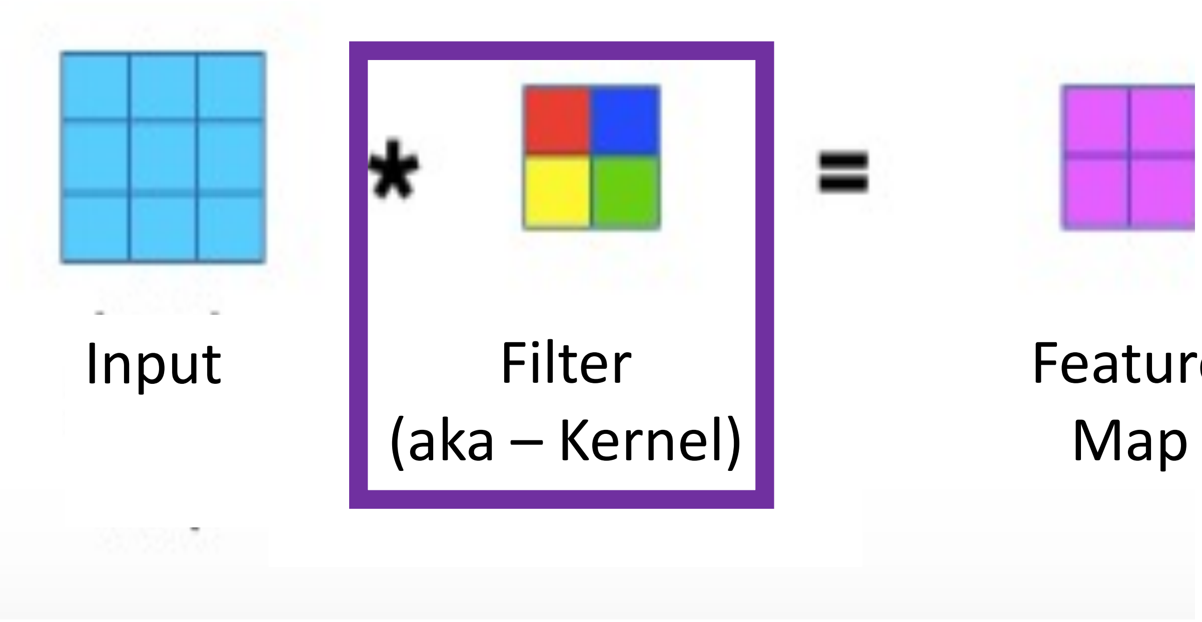
Filter

1	0	1
0	1	0
1	0	1

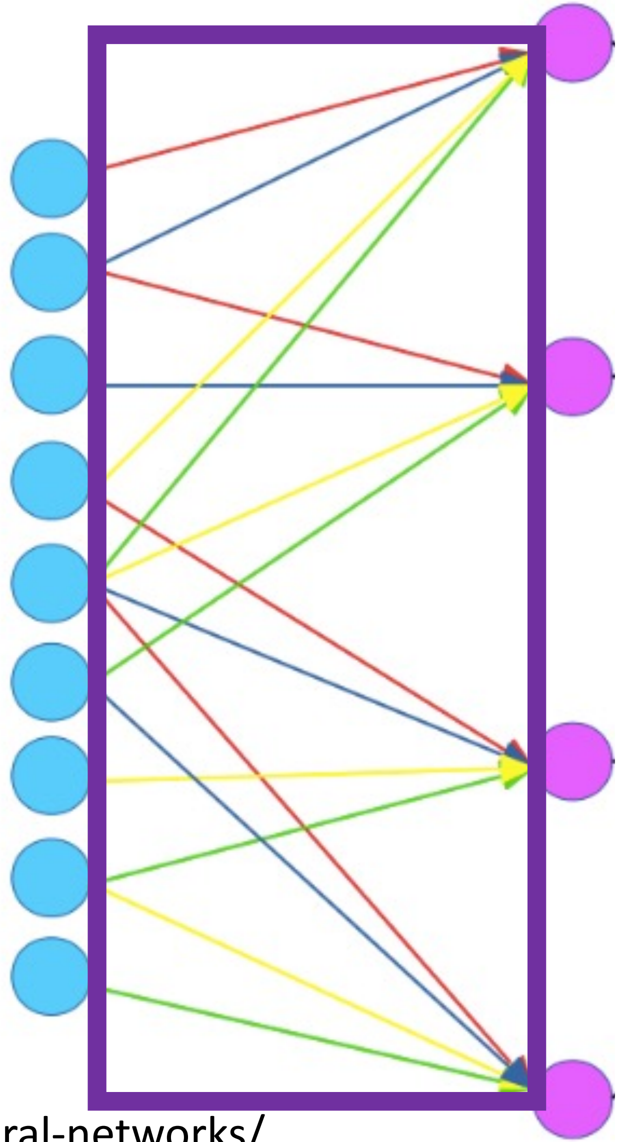
Feature Map

4	3	4
2	4	3
2	3	4

Convolutional Layer: Parameters to Learn



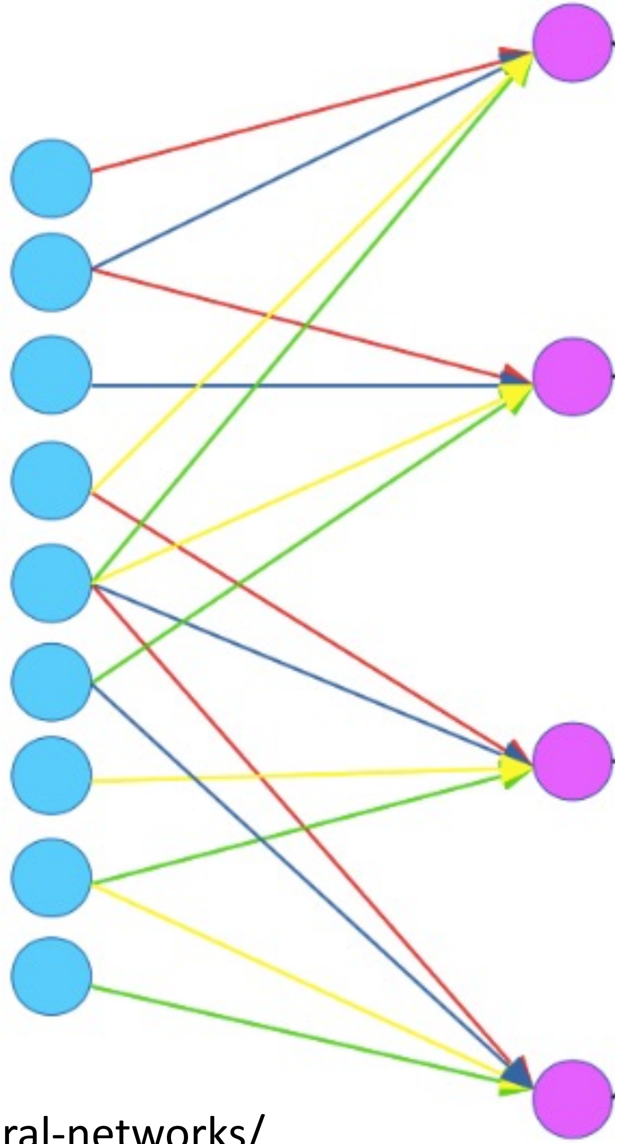
Way to Interpret
Neural Network



https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

Convolutional Layer: Parameters to Learn

- For shown example, how many weights must be learned?
  - 4 (red, blue, yellow, and green values)
- If we instead used a fully connected layer, how many weights would need to be learned?
  - 36 (9 turquoise nodes x 4 magenta nodes)
- For shown example, how many parameters must be learned?
  - 5 (4 weights + 1 bias)
- If we instead used a fully connected layer, how many parameters would need to be learned?
  - 40 (36 weights + 4 bias)

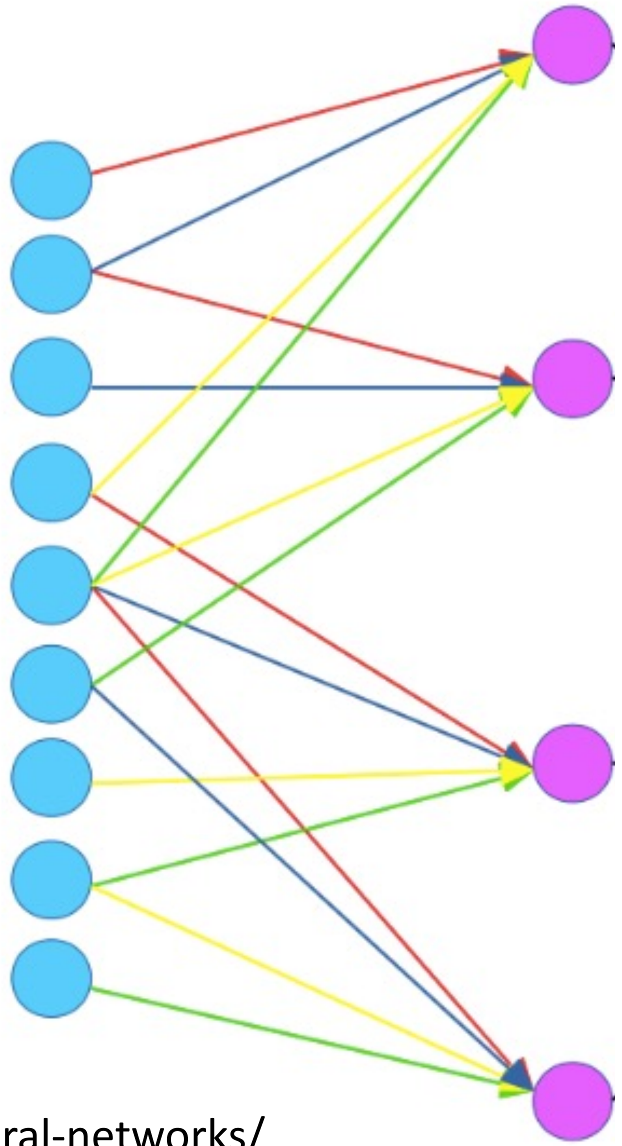


https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

Convolutional Layer: Parameters to Learn

- Parameter sharing significantly reduces number of parameters to learn and so storage requirements

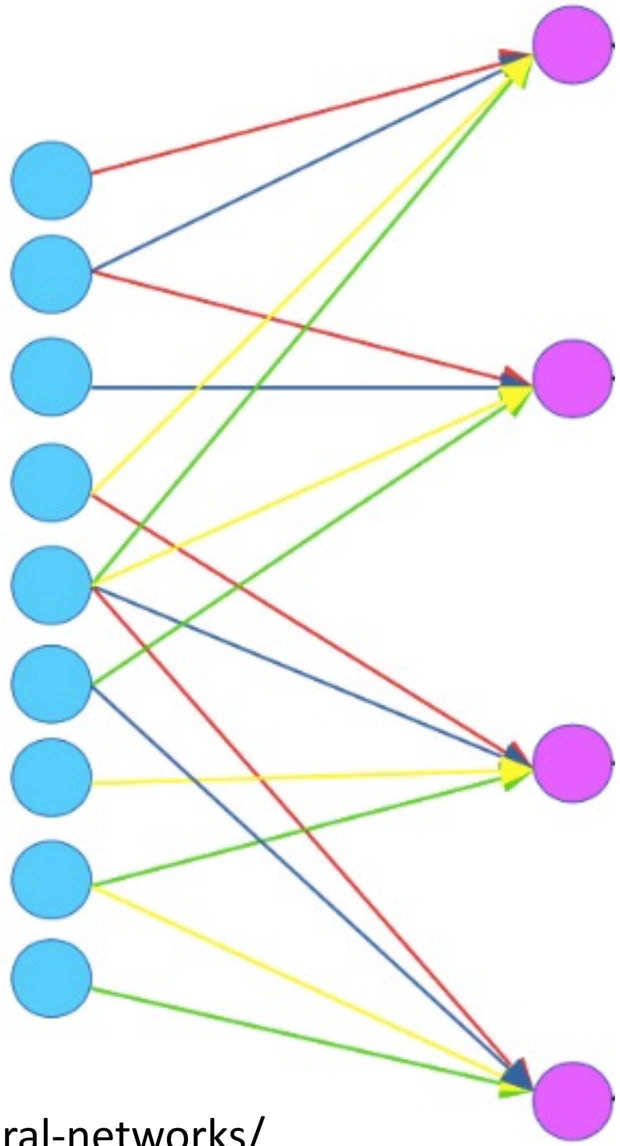
- Sparse (rather than full) connectivity also significantly reduces the number of computational operations required



https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

Convolutional Layer: Parameters to Learn

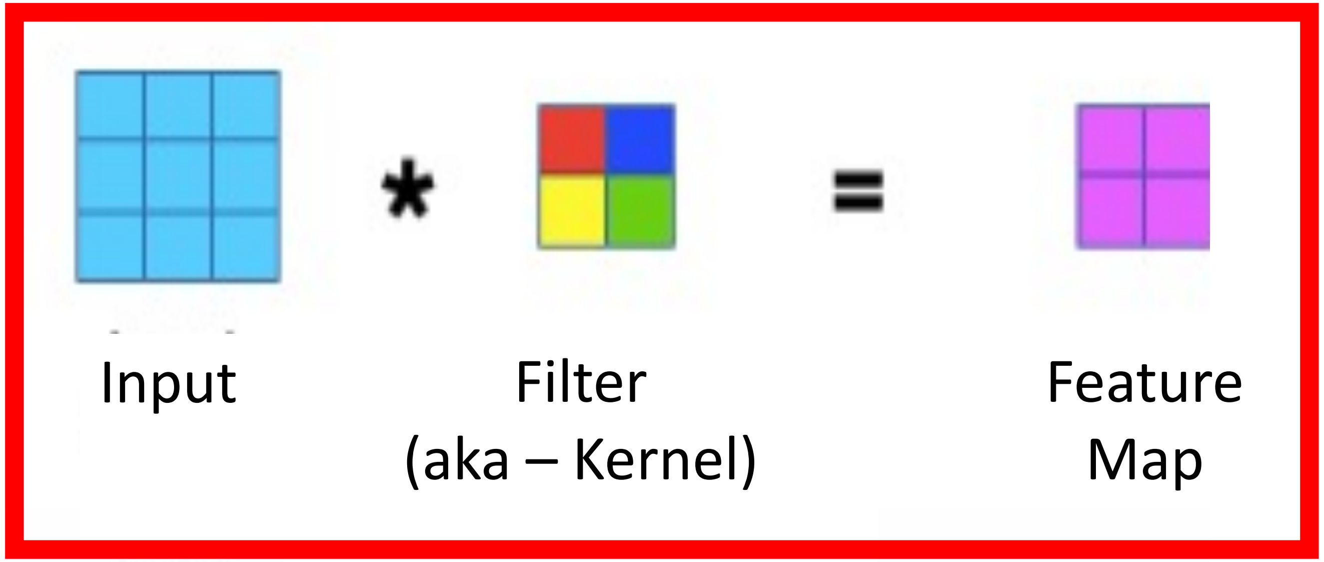
- Neocognitron hard-coded filter values... we will cover models that learn the filter values in the next lecture



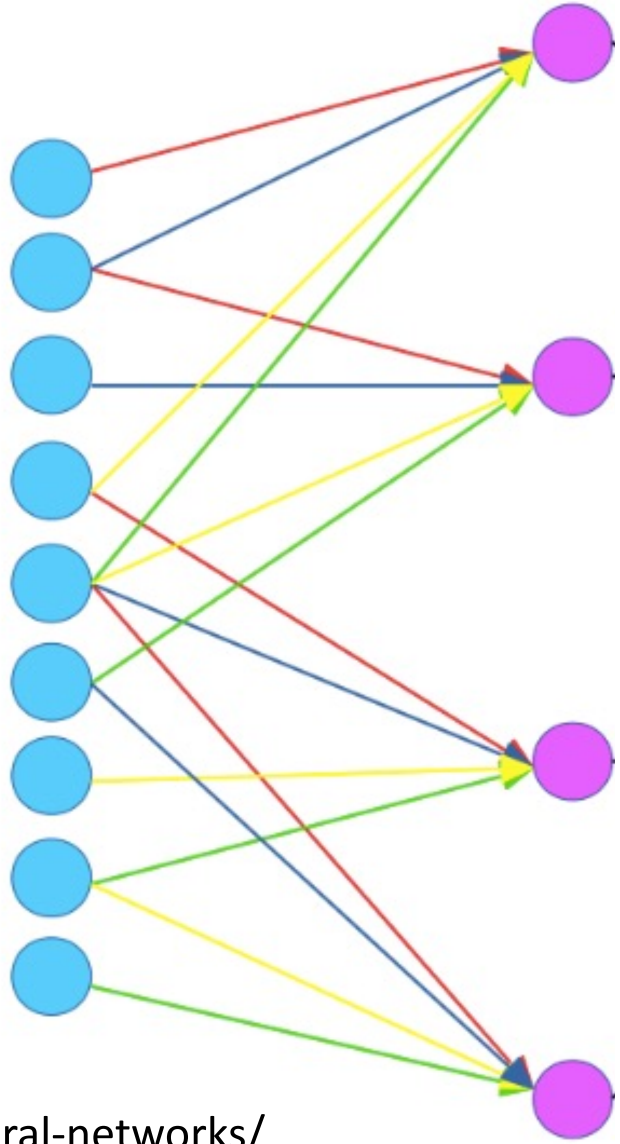
https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

Convolutional Layer

- Many neural network libraries use “convolution” interchangeably with “cross correlation”; for mathematicians, these are technically different
- Examples in these slides show the “cross-correlation” function

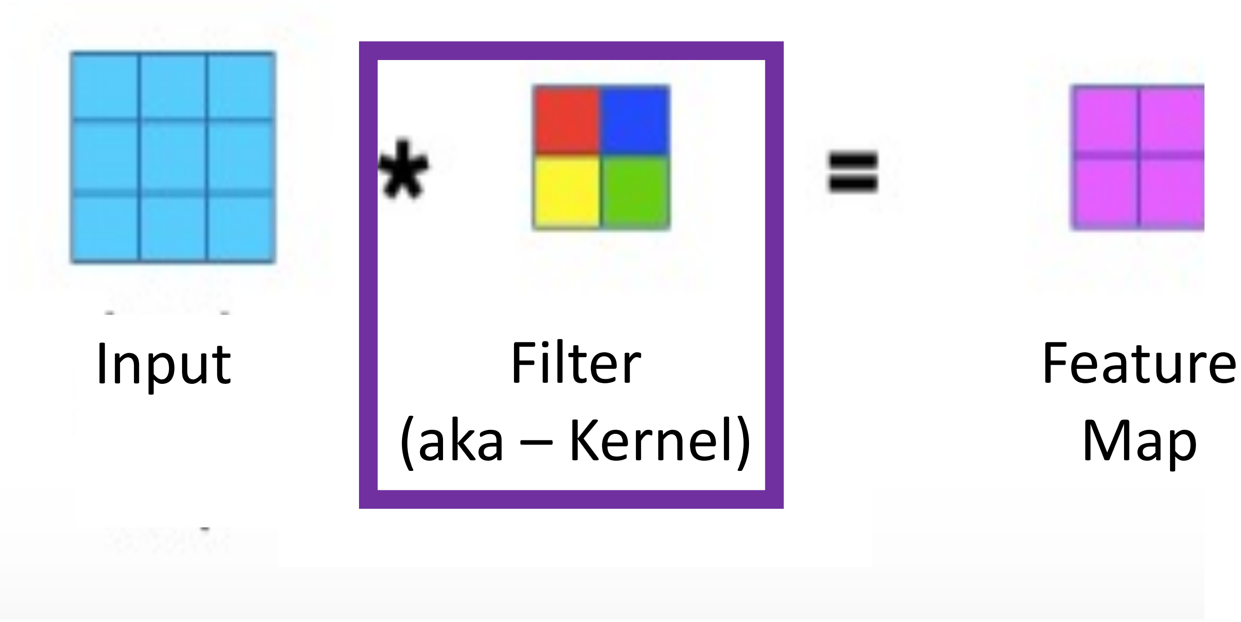


Way to Interpret Neural Network

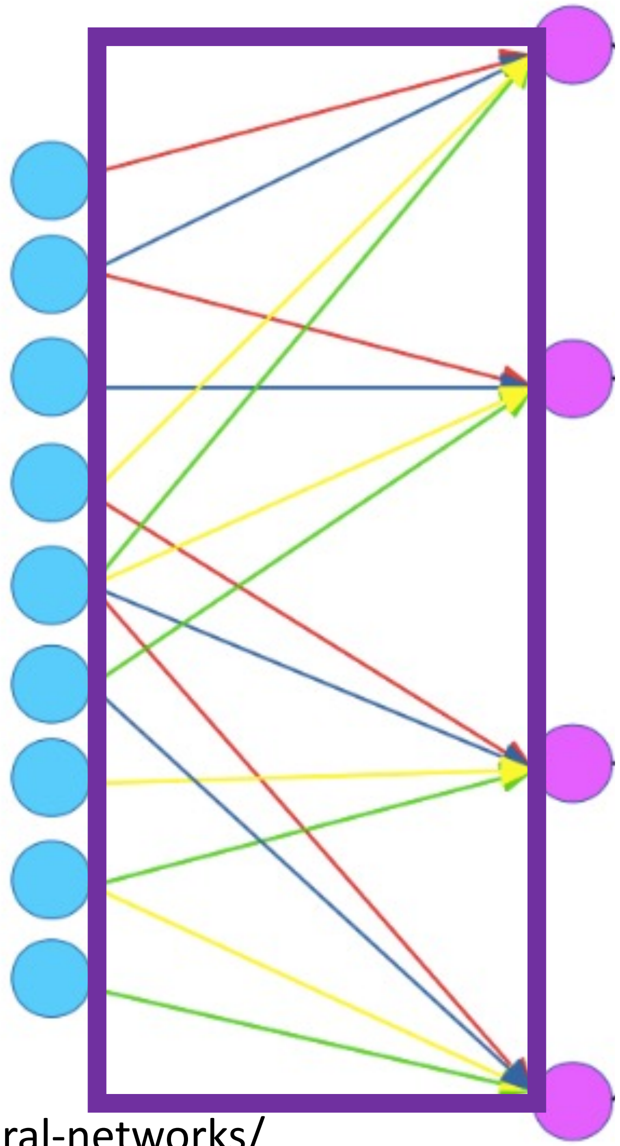


https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

Convolutional Layer: What Does The Filter Do?



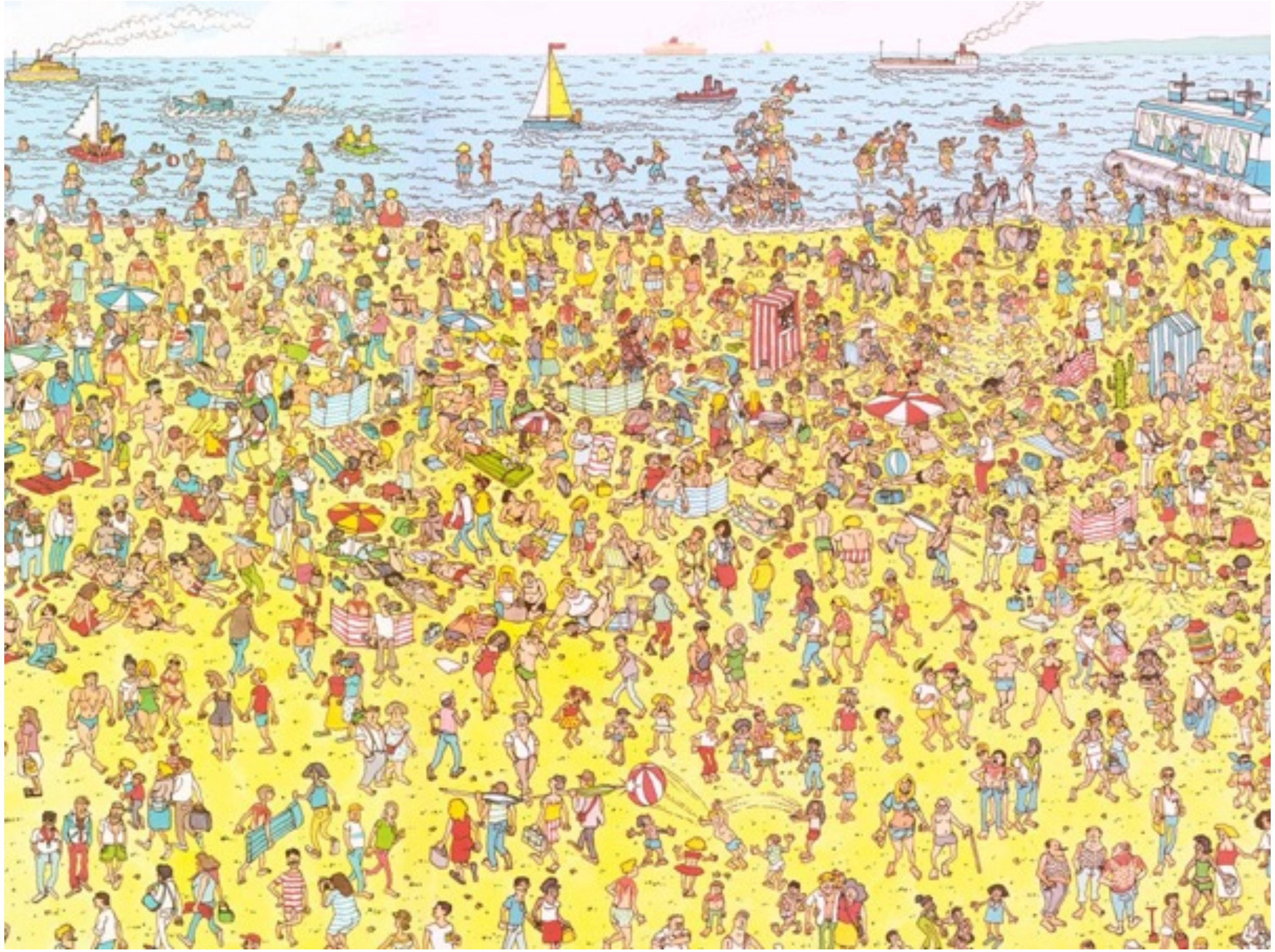
Way to Interpret Neural Network



https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

Filter: What Does It Do?

Filter



Filter: What Does It Do?

- e.g.,

Filter

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Visualization of Filter

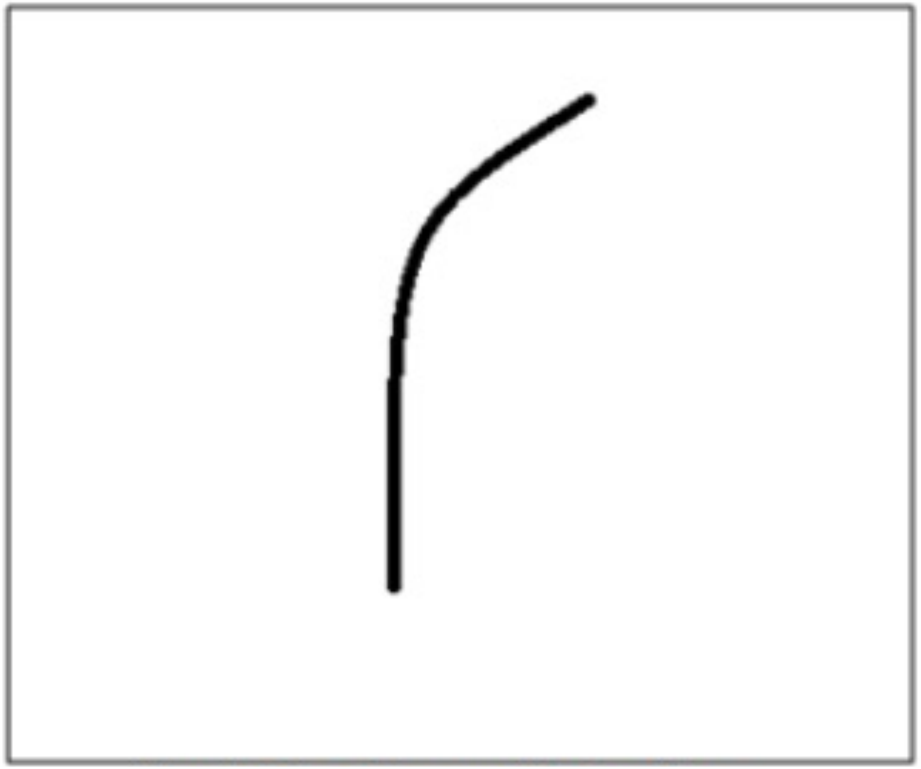
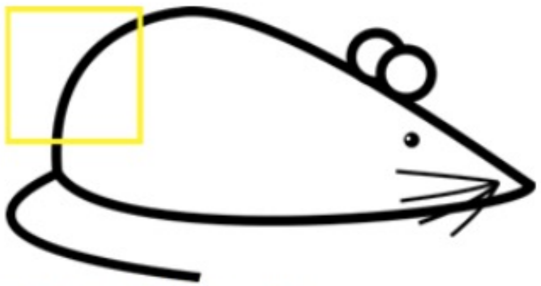


Image Credit: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

Filter: What Does It Do?

- e.g.,

Filter Overlaid on Image



Image

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

*

Filter

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Weighted Sum = ?

Weighted Sum = (50x30) + (20x30) + (50x30) + (50x3) + (50x30)

Weighted Sum = 6600 **(Large Number!!)**

Image Credit: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

Filter: What Does It Do?

- e.g.,

Filter Overlaid on Image

Image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

*

Filter

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Weighted Sum = ?

Weighted Sum = 0 **(Small Number!!)**

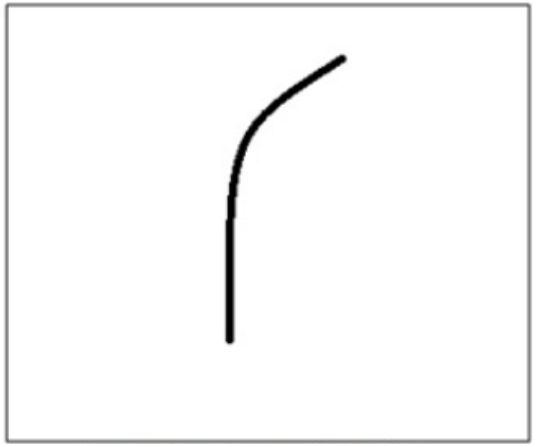
Image Credit: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

Filter: What Does It Do?

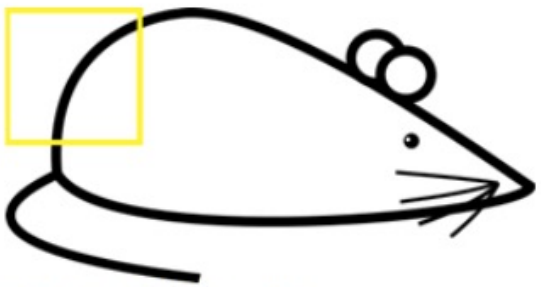
- e.g.,

This Filter is a Curve Detector!

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0



Filter Overlaid on Image (Big Response!)



Filter Overlaid on Image (Small Response!)

Image Credit: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

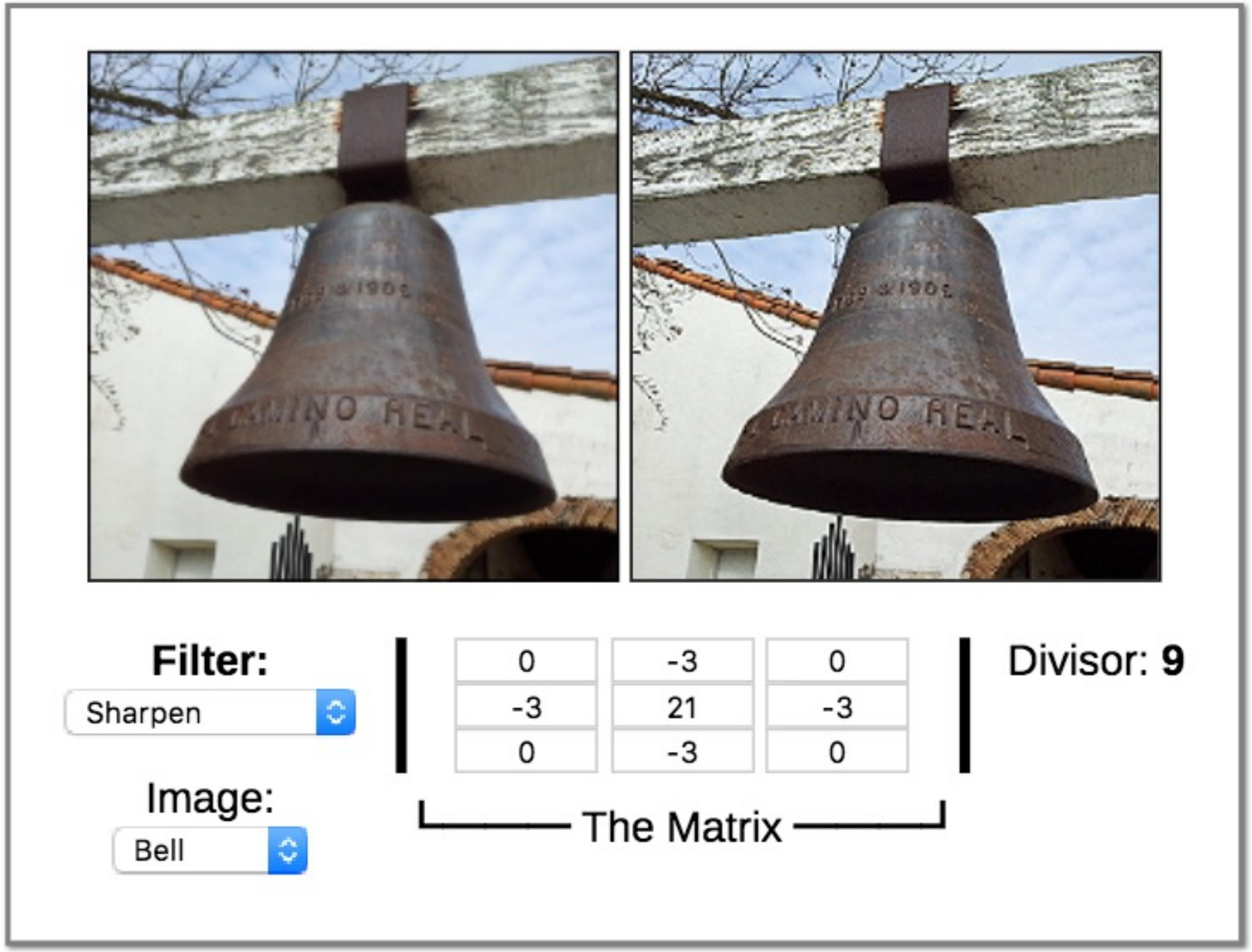
Filters Detect Different Features

	Filter	Feature Map
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

	Filter	Feature Map
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

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

Different Filters Detect Different Features

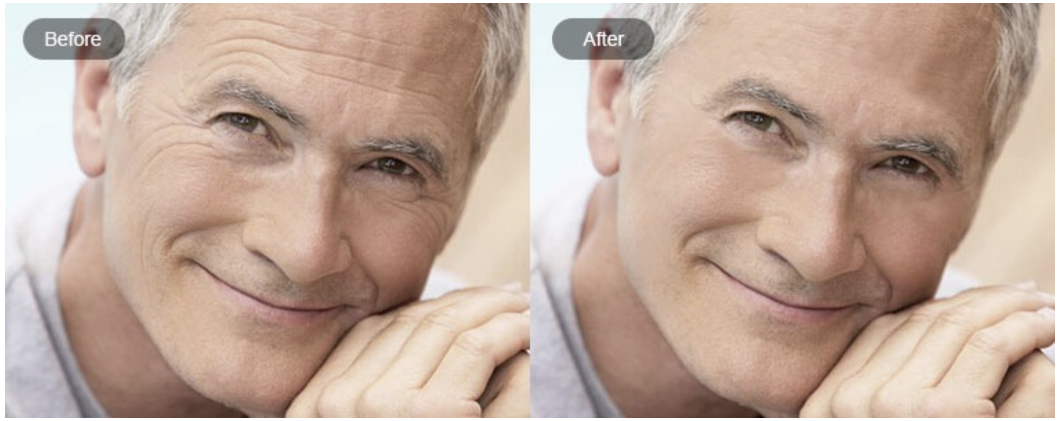


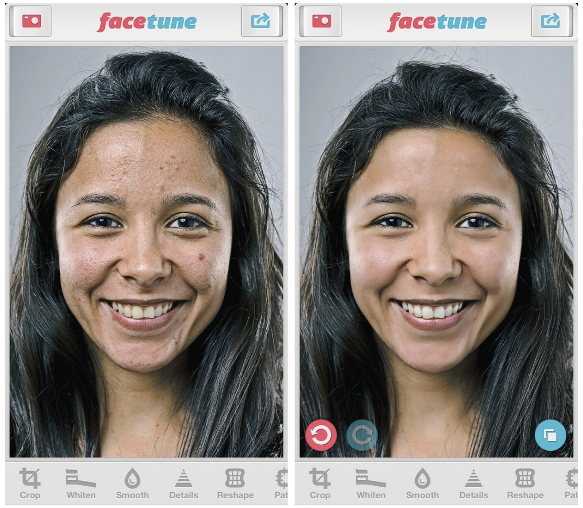
Demo: http://beej.us/blog/data/convolution-image-processing/

Group Discussion

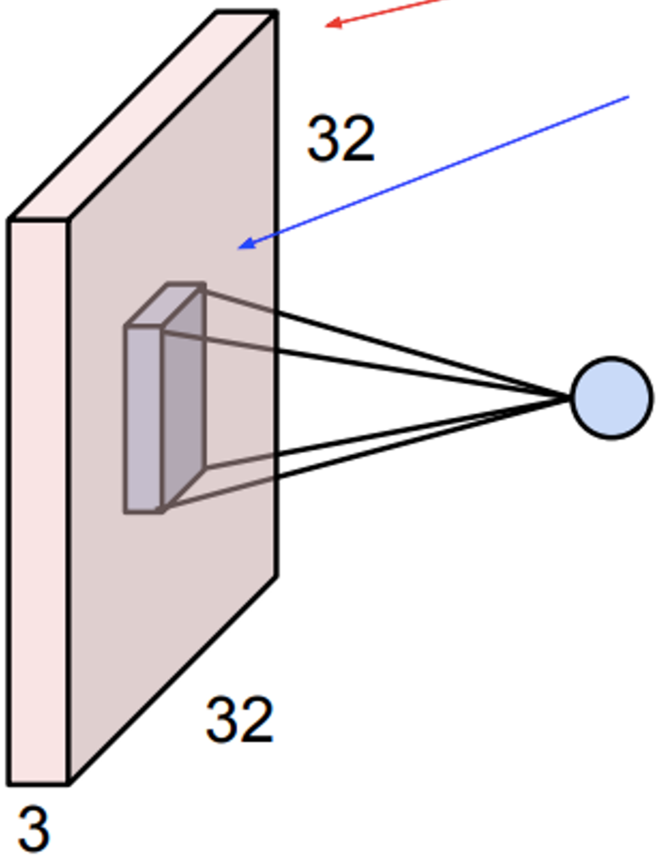
1. How would you design a filter to “brighten” an image?

2. How would you design a filter to remove wrinkles/blemishes?





Convolutional Layer: After Applying Filter, Remember to Introduce Non-Linearity



32x32x3 image

5x5x3 filter w

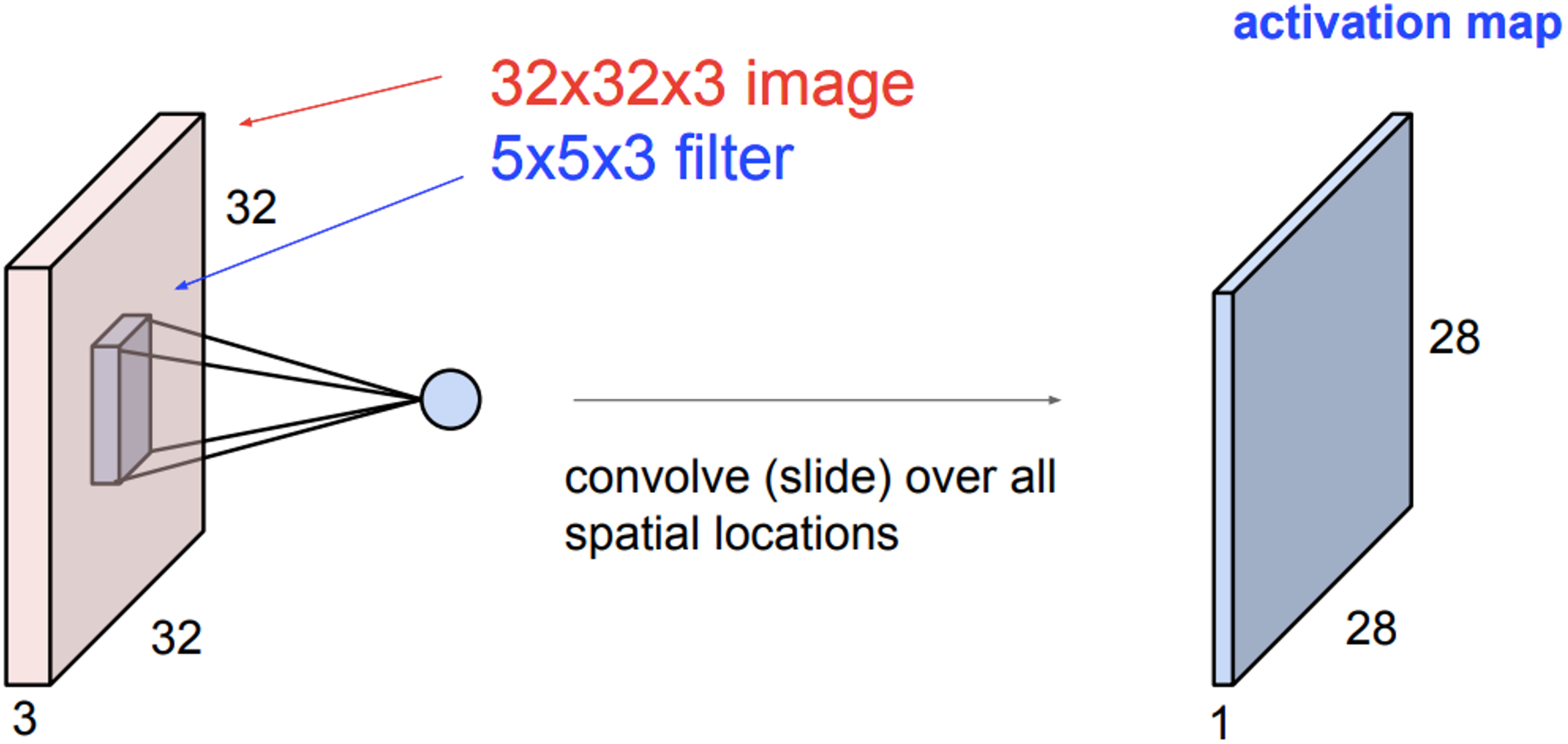
Apply activation function to number; e.g., ReLU

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)

$$w^T x + b$$

Slide Credit: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

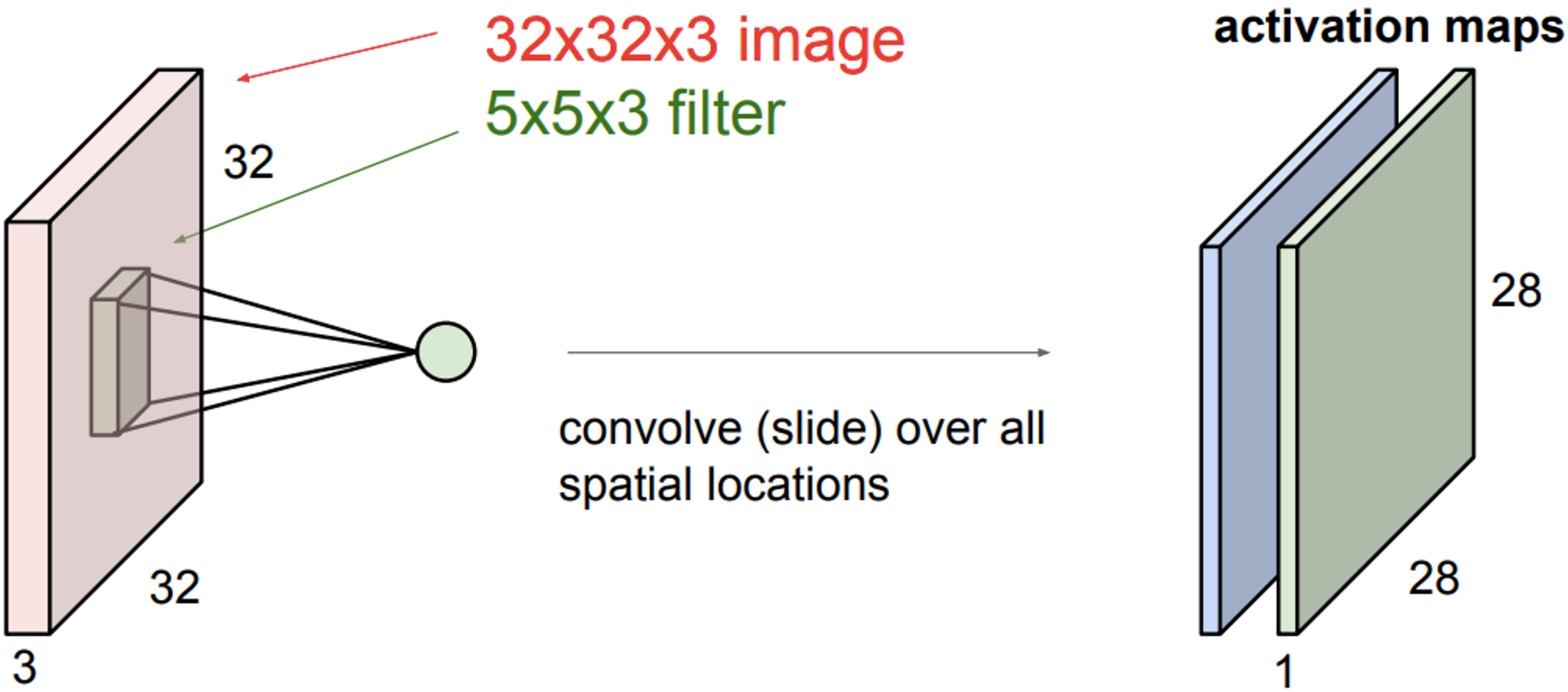
Convolutional Layer: Slide Filter Across Input



Slide Credit: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

consider a second, green filter

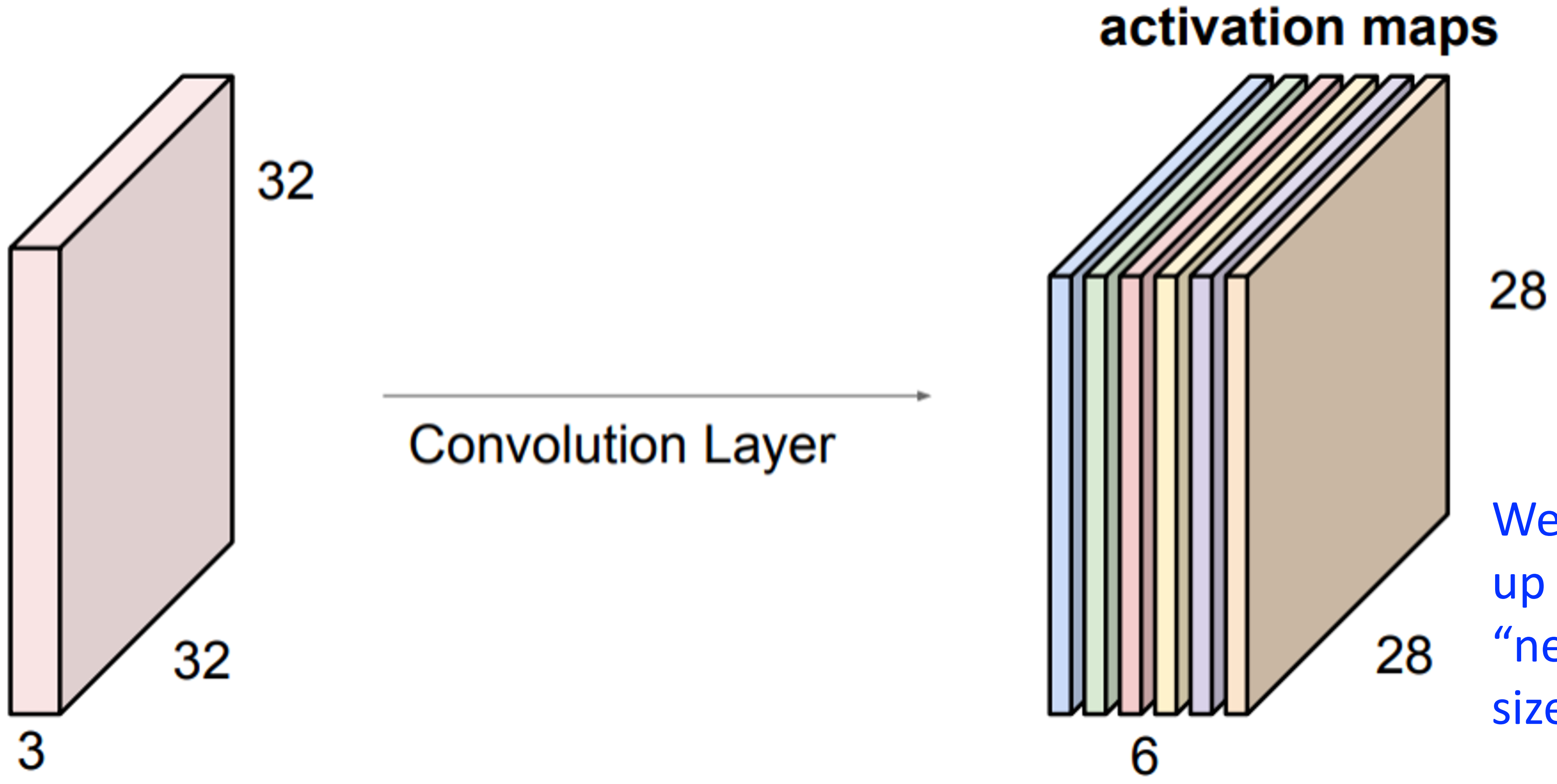
Convolutional Layer: Slide Filter Across Input



Slide Credit: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

Convolutional Layer: Slide Filter Across Input

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

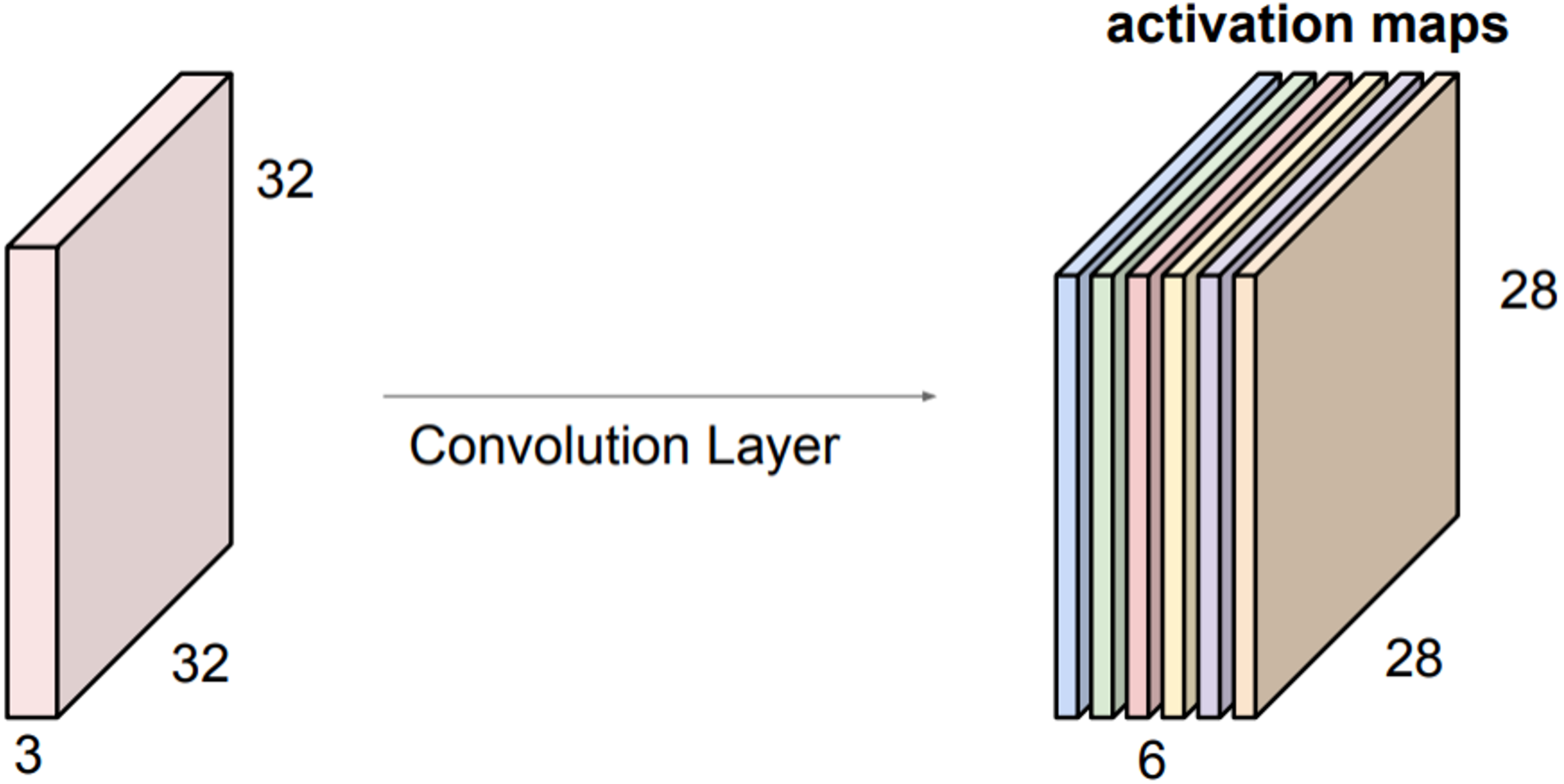


We stack these up to get a "new image" of size 28x28x6!

Slide Credit: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

Convolutional Layer: Parameters to Learn

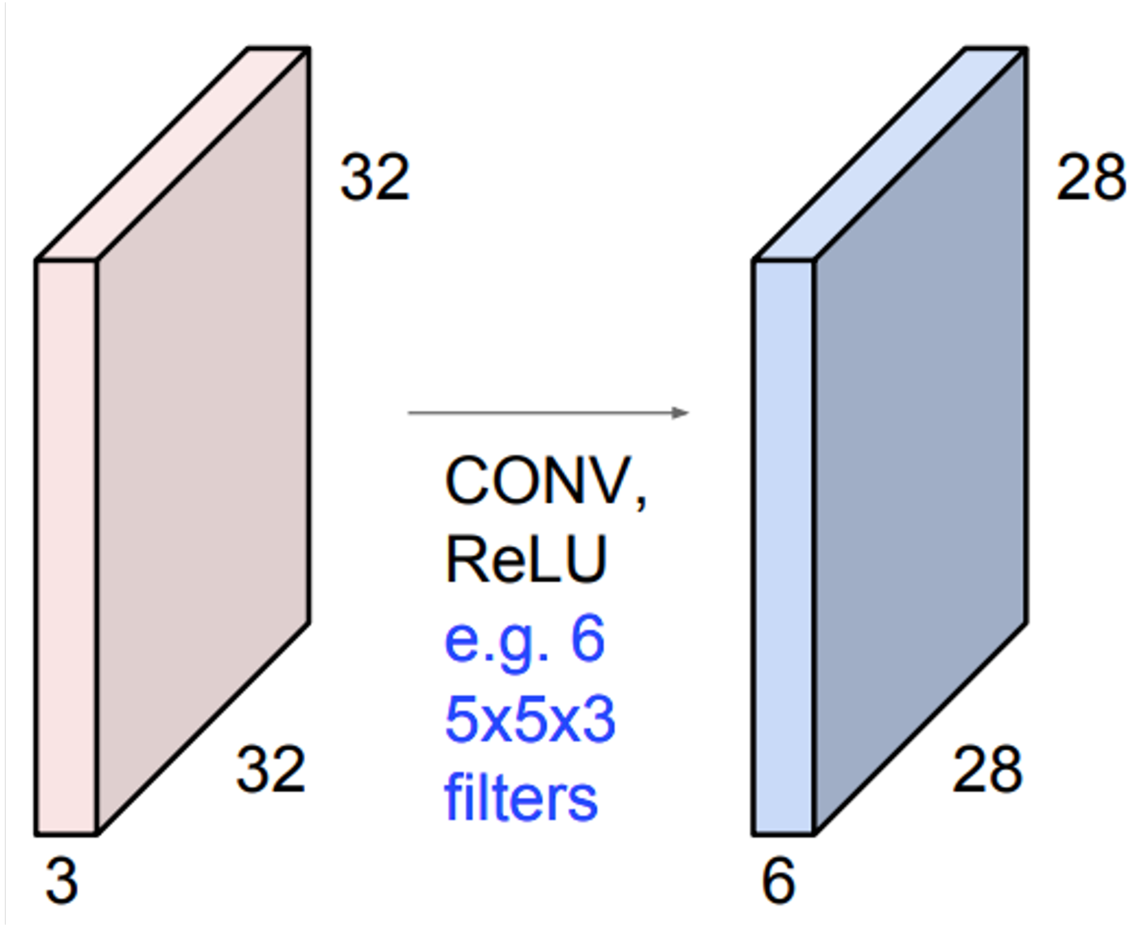
Parameters: bank of filters and biases used to create the activation maps (aka – feature maps)



Slide Credit: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

Convolutional Layers Stacked

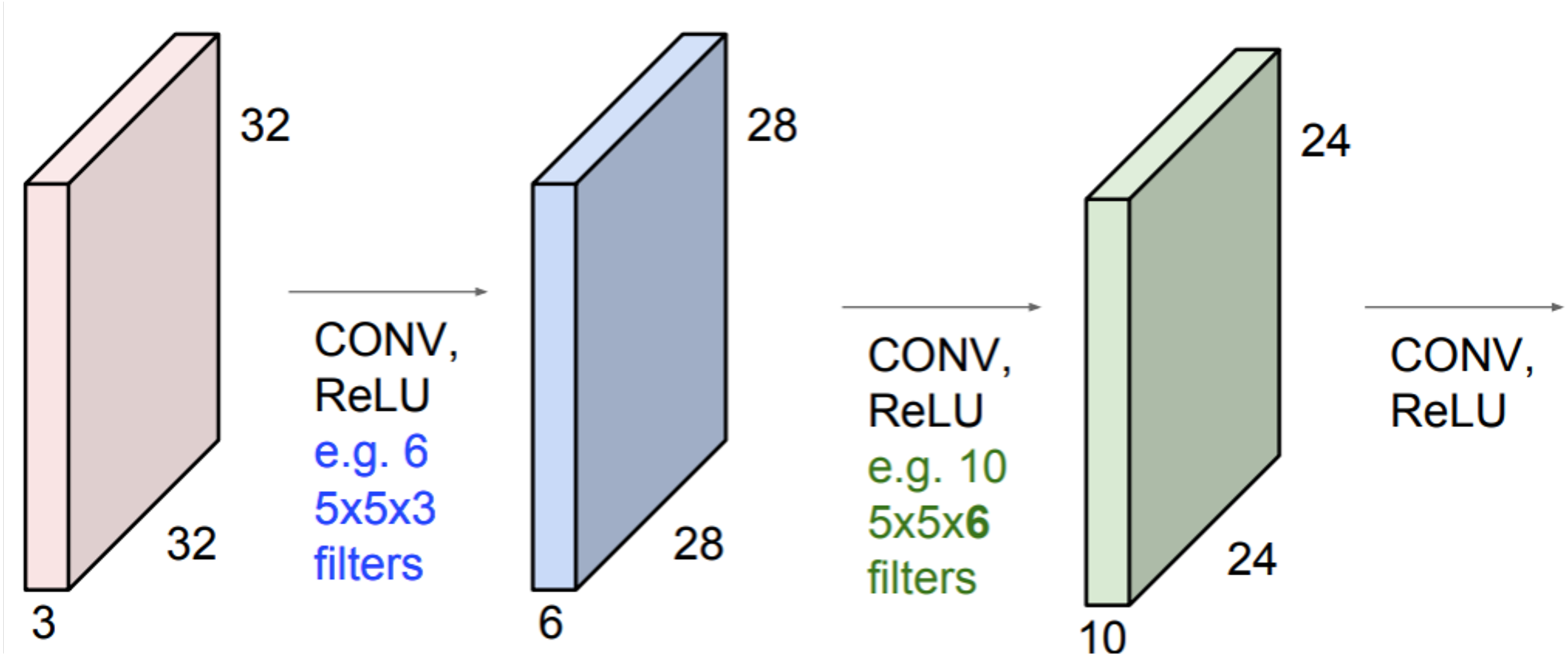
Can then stack a sequence of convolution layers, interspersed with activation functions:



Slide Credit: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

Convolutional Layers Stacked

Can then stack a sequence of convolution layers, interspersed with activation functions:

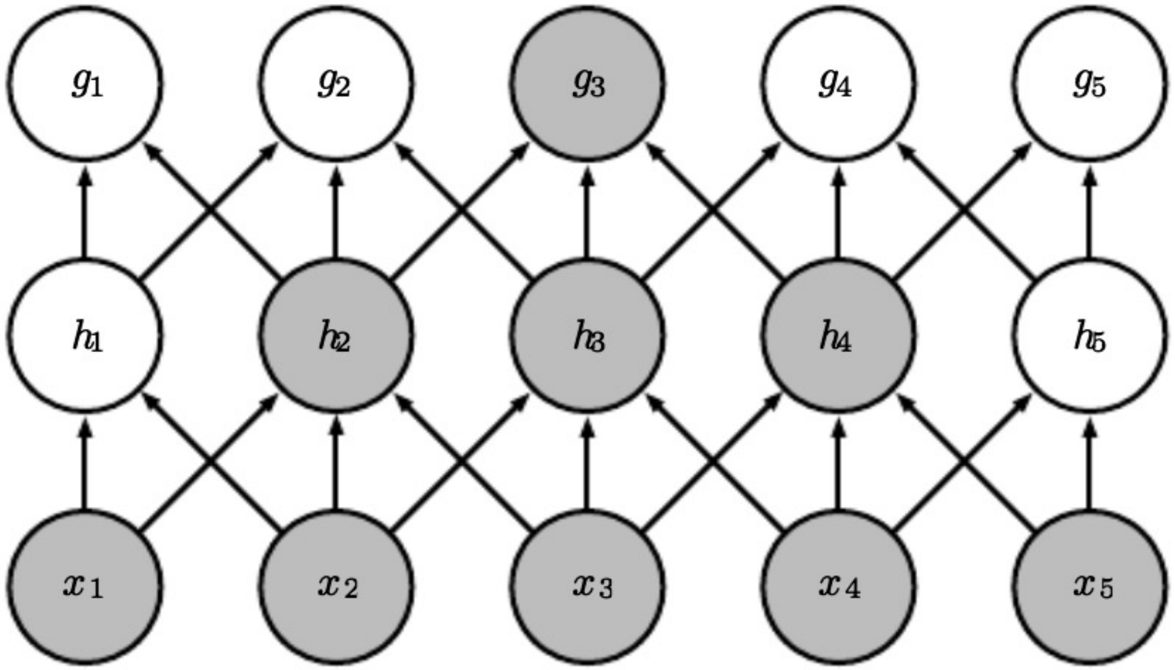


Slide Credit: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

Convolutional Layers Stacked

Can then stack a sequence of convolution layers, interspersed with activation functions:

Stacking many convolutional layers leads to identifying patterns in increasingly **larger regions of the input (e.g., pixel) space.**



https://www.deeplearningbook.org/contents/convnets.html

Convolution: Implementation Details

- **Padding**: add values at the boundaries to control output size

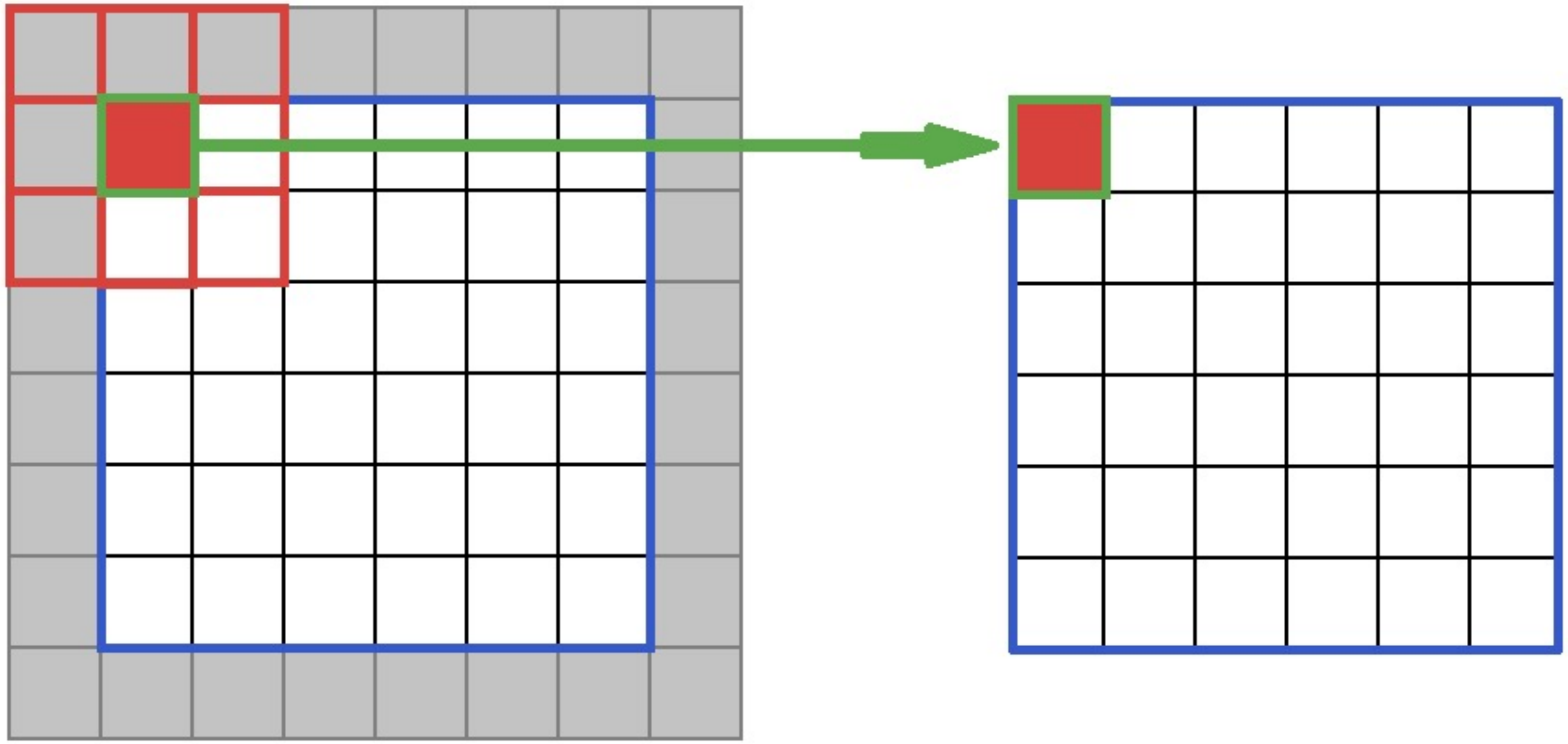


Image Credit: https://software.intel.com/en-us/node/586159

Convolution: Implementation Details

- **Stride**: how many steps taken spatially before applying a filter
  - e.g., 2x2

Image

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

Filter

1	0	1
0	1	0
1	0	1

Feature Map

4	4
2	4

http://deeplearning.net/software/theano/tutorial/conv_arithmetic.html

Convolution: Implementation Details

- Demo:
  - https://theano-pymc.readthedocs.io/en/latest/tutorial/conv_arithmetic.html

Parameters vs Hyperparameters In Convolutional Layers

- Hyperparameters:
  - Number of filters, including height and width of each
  - Strides
  - Padding type

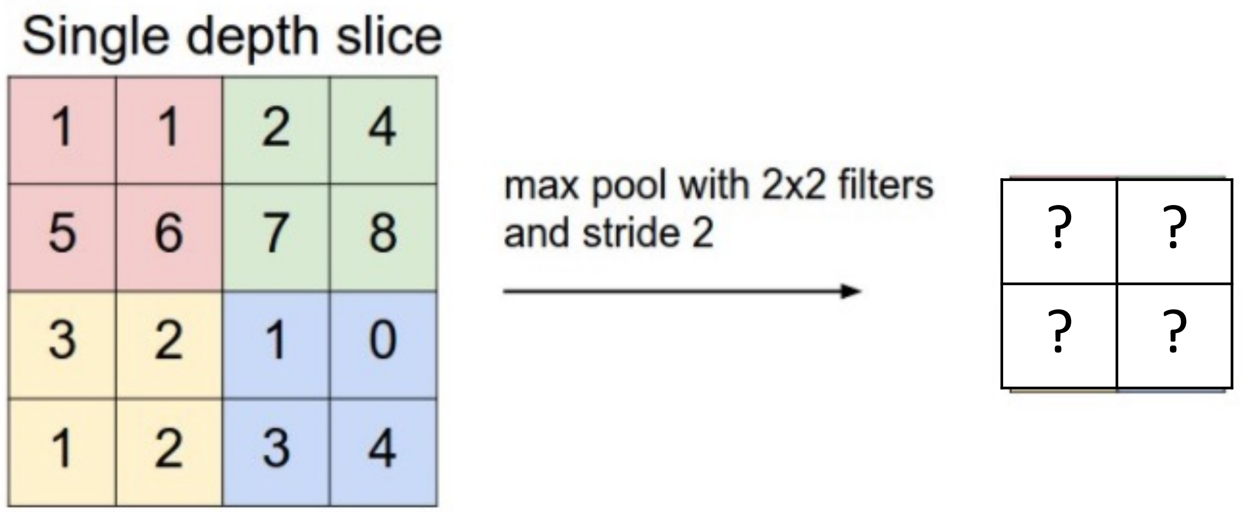
- Parameters
  - Weights
  - Biases

Today’s Topics

- Neural Networks for Spatial Data
- History of Convolutional Neural Networks (CNNs)
- CNNs – Convolutional Layers
- CNNs – Pooling Layers

Pooling Layer: Summarizes Neighborhood

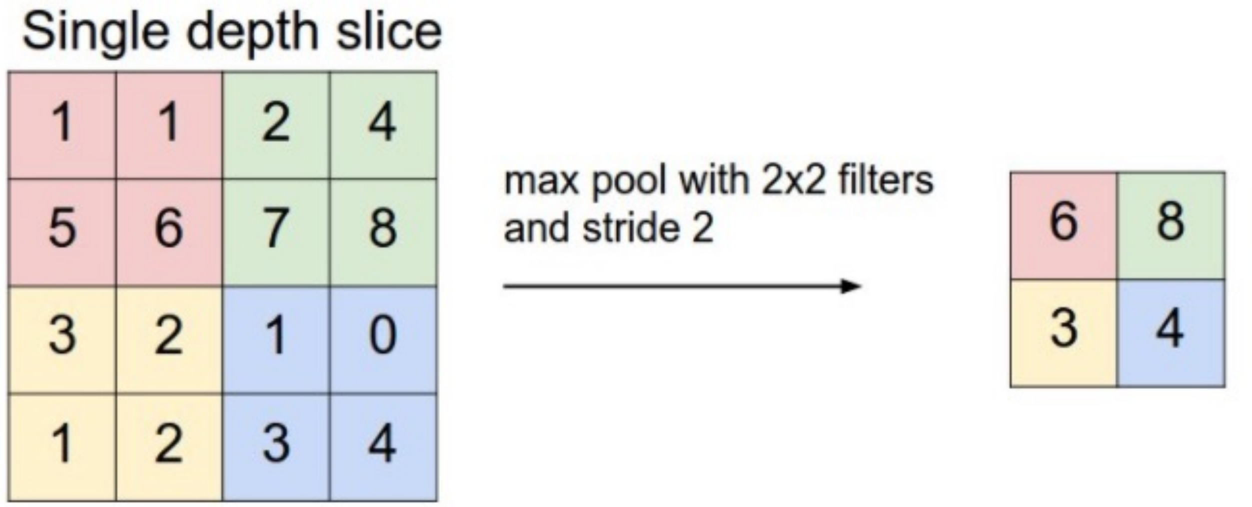
- **Max-pooling**: partitions input into a set of non-overlapping rectangles and outputs the maximum value for each chunk



http://cs231n.github.io/convolutional-networks/#pool

Pooling Layer: Summarizes Neighborhood

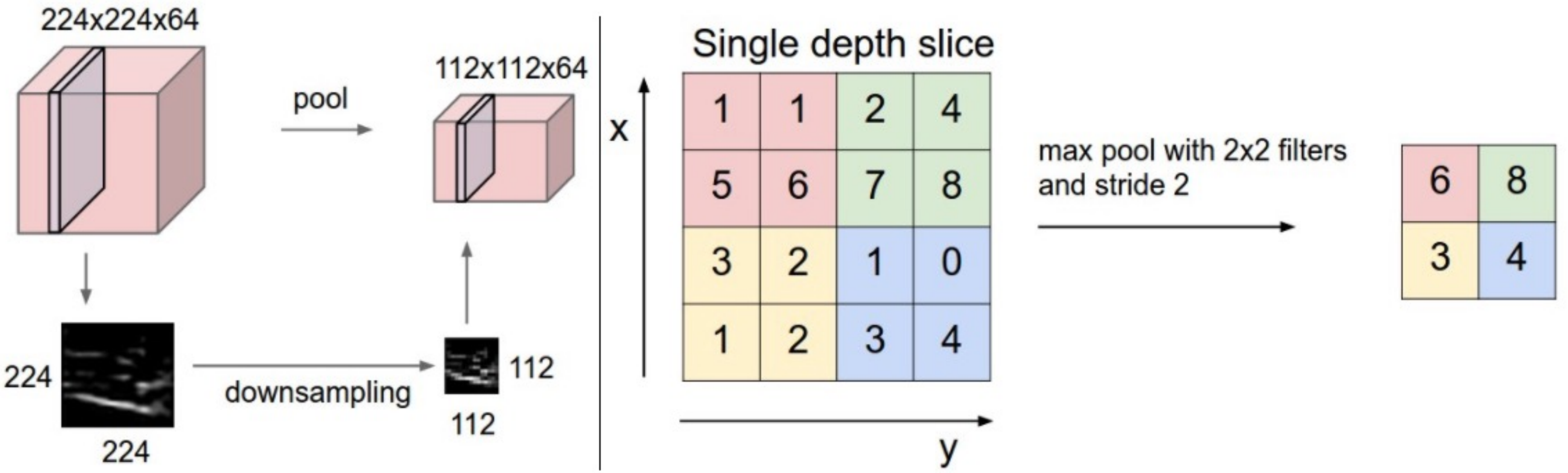
- **Max-pooling**: partitions input into a set of non-overlapping rectangles and outputs the maximum value for each chunk



http://cs231n.github.io/convolutional-networks/#pool

Pooling Layer: Summarizes Neighborhood

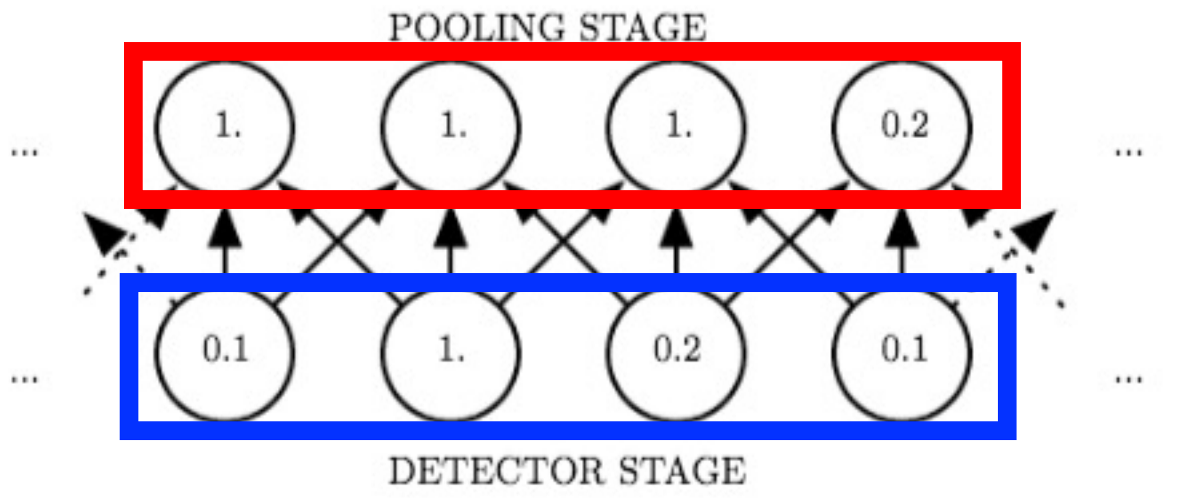
- **Max-pooling**: partitions input into a set of non-overlapping rectangles and outputs the maximum value for each chunk



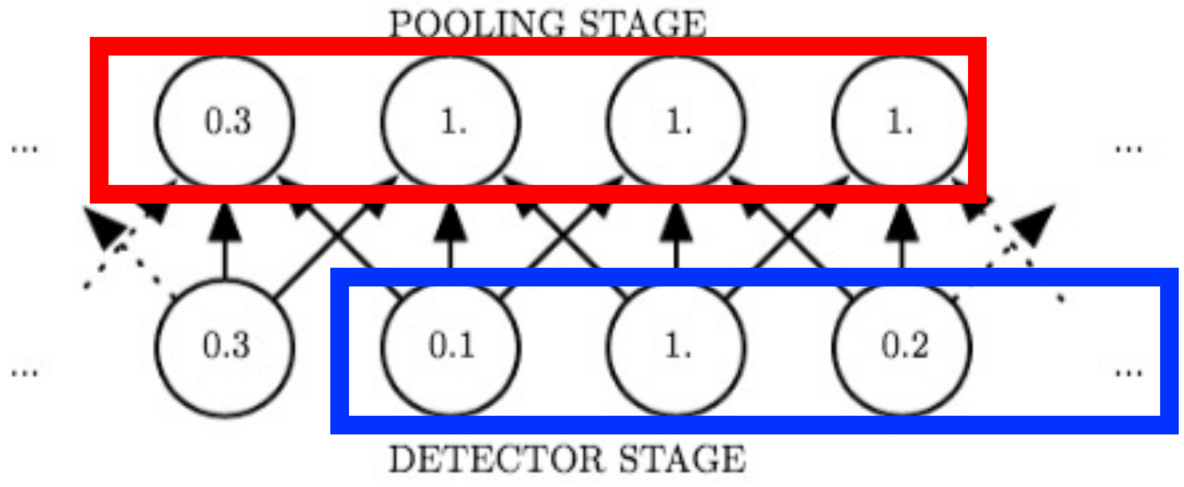
http://cs231n.github.io/convolutional-networks/#pool

Pooling Layer

- Resilient to small translations



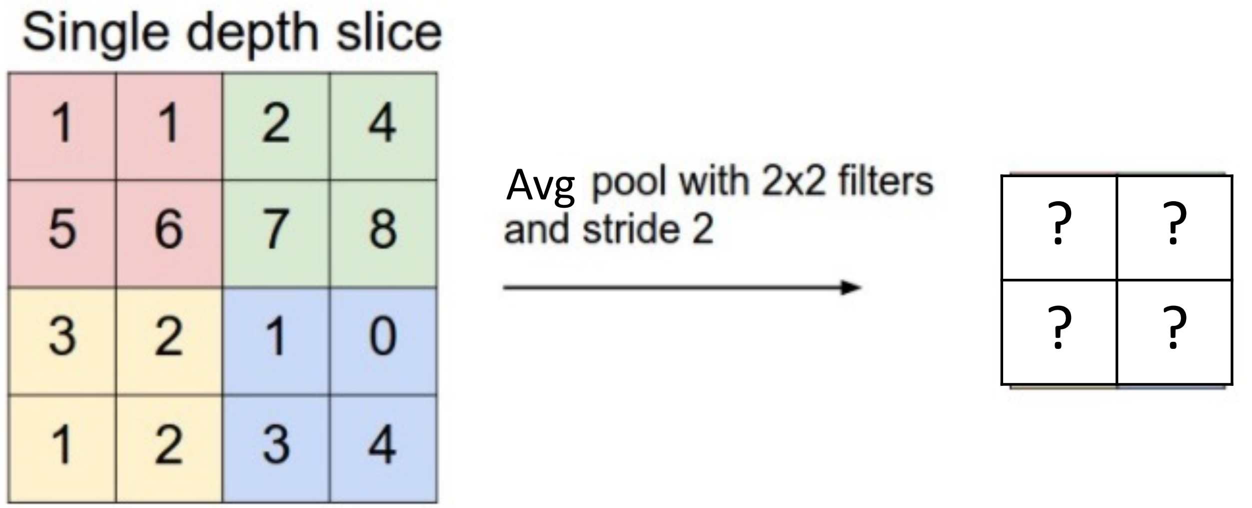
- e.g.,
  - Input: all values change (shift right)
  - Output: only half the values change



https://www.deeplearningbook.org/contents/convnets.html

Pooling Layer: Summarizes Neighborhood

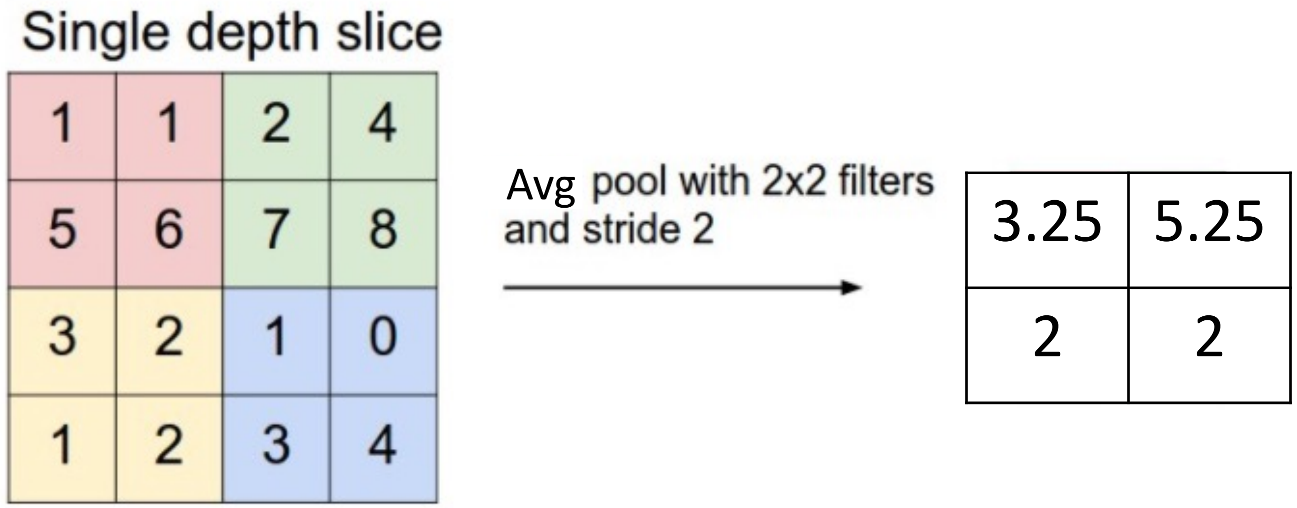
- **Max-pooling***:* partitions input into a set of non-overlapping rectangles and outputs the maximum value for each chunk
- **Average-pooling***:* partitions input into a set of non-overlapping rectangles and outputs the average value for each chunk



http://cs231n.github.io/convolutional-networks/#pool

Pooling Layer: Summarizes Neighborhood

- **Max-pooling**: partitions input into a set of non-overlapping rectangles and outputs the maximum value for each chunk
- **Average-pooling**: partitions input into a set of non-overlapping rectangles and outputs the average value for each chunk



http://cs231n.github.io/convolutional-networks/#pool

Pooling Layer: Benefits

- How many parameters must be learned?
  - None
- Benefits?
  - Builds in invariance to translations of the input
  - Reduces memory requirements
  - Reduces computational requirements

Today’s Topics

- Neural Networks for Spatial Data
- History of Convolutional Neural Networks (CNNs)
- CNNs – Convolutional Layers
- CNNs – Pooling Layers


The End