

Object Recognition: Dataset Challenges and Fundamentals of CNNs

Danna Gurari

University of Colorado Boulder

Fall 2024



Review

- Last lecture:
 - Ways of seeing: image and video acquisition
 - Evolution of computer vision (before versus after 2012)
 - Fundamentals of a neural network architecture
 - Training deep neural networks
- Assignments (Canvas)
 - Reading assignment due earlier today
 - Next two reading assignments due next Monday and Wednesday
- Questions?

Object Recognition: Today's Topics

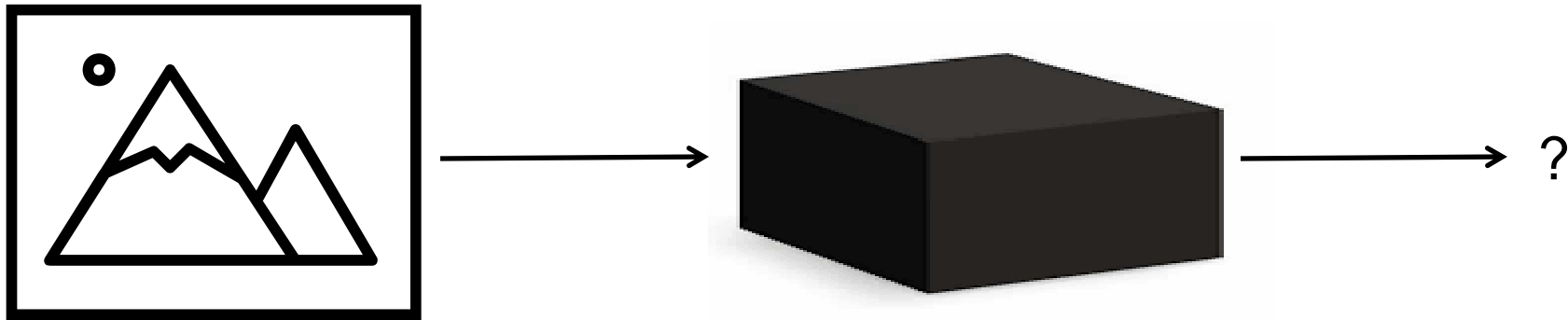
- Problem
- Applications
- Datasets
- Evaluation metric
- A Popular Solution: Convolutional Neural Networks

Object Recognition: Today's Topics

- Problem
- Applications
- Datasets
- Evaluation metric
- A Popular Solution: Convolutional Neural Networks

Object Recognition

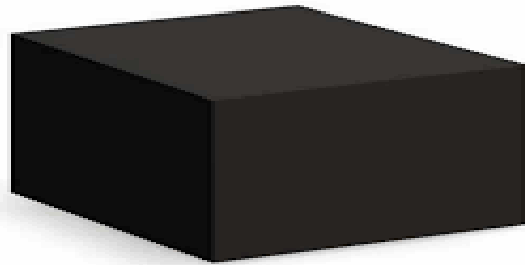
- Given an image, indicate what object(s) are in the image



Object Recognition

e.g.,

INPUT



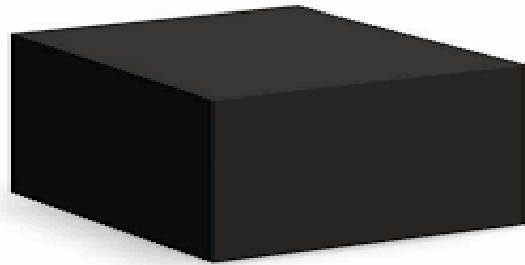
?

OUTPUT

Object Recognition

e.g.,

INPUT



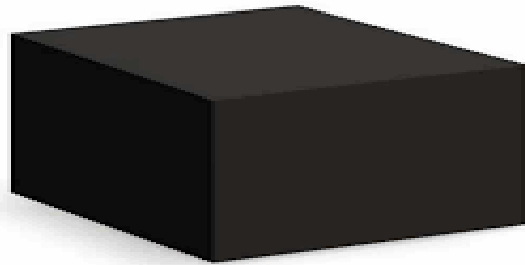
OUTPUT

Sunflower ✓

Object Recognition

e.g.,

INPUT



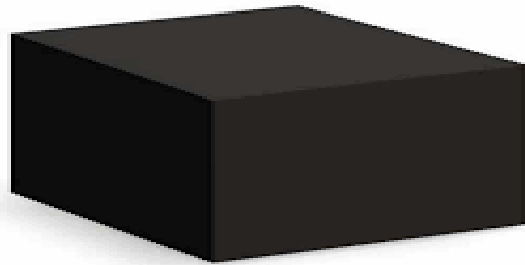
?

OUTPUT

Object Recognition

e.g.,

INPUT



OUTPUT

Weasal

Object Recognition: Today's Topics

- Problem
- Applications
- Datasets
- Evaluation metric
- A Popular Solution: Convolutional Neural Networks

Shopping



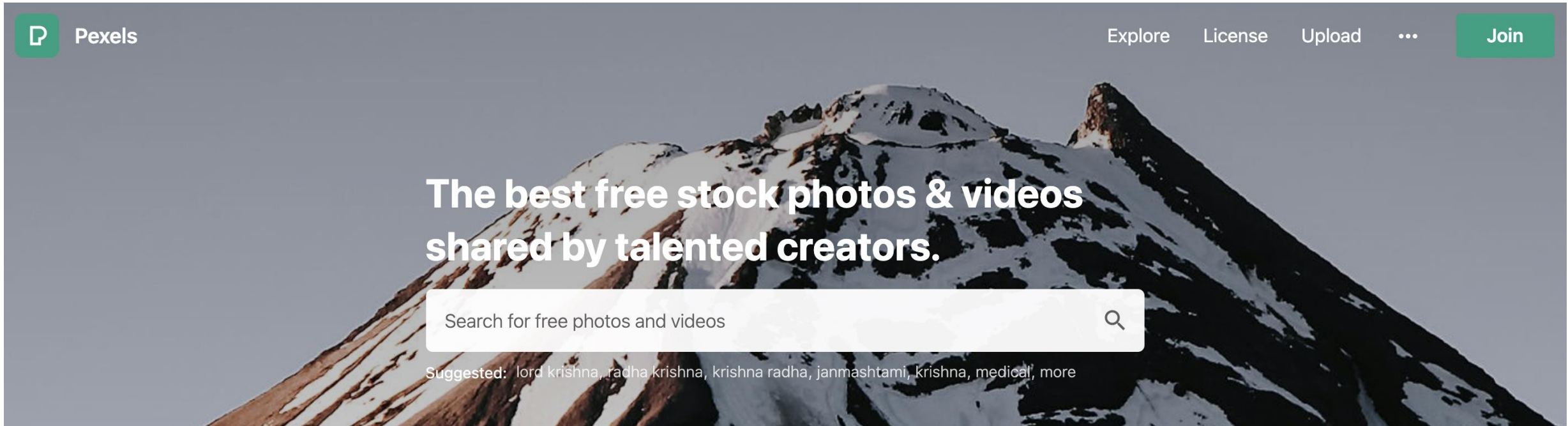
Take a picture of an object and find where to buy it

Photo Organization



Demo: <https://www.youtube.com/watch?v=aBqmWUalnho>
(start video at 1:49)

Image Search



Pexels

[Explore](#)

[License](#)

[Upload](#)



[Join](#)

**The best free stock photos & videos
shared by talented creators.**

Search for free photos and videos



Suggested: lord krishna, radha krishna, krishna radha, janmashtami, krishna, medical, more

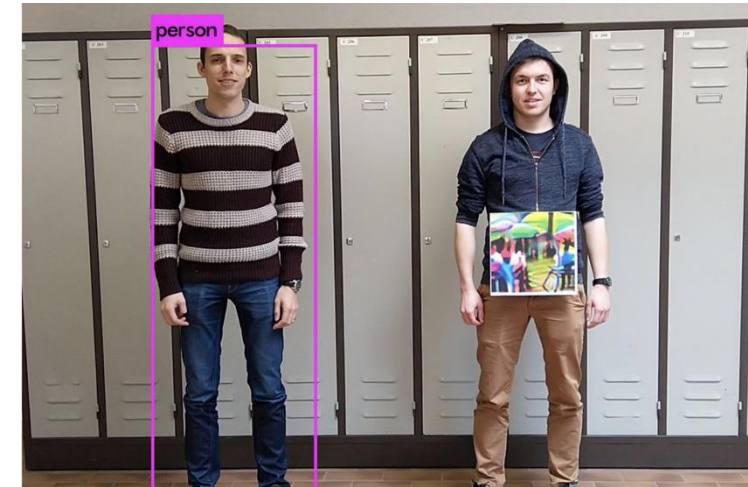
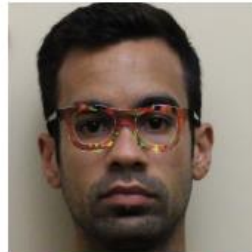
Applications Gone Wrong

- Ethical mistake: people tagged as “gorillas”



<http://www.usatoday.com/story/tech/2015/07/01/google-apologizes-after-photos-identify-black-people-as-gorillas/29567465/>

- Security risk: people mis-recognized or invisible when wearing special designs



<https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf>

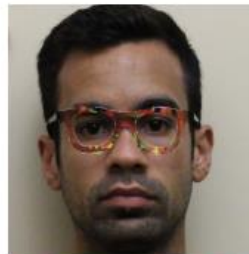
<https://www.theverge.com/2019/4/23/18512472/fool-ai-surveillance-adversarial-example-yolov2-person-detection>

Applications Gone Wrong

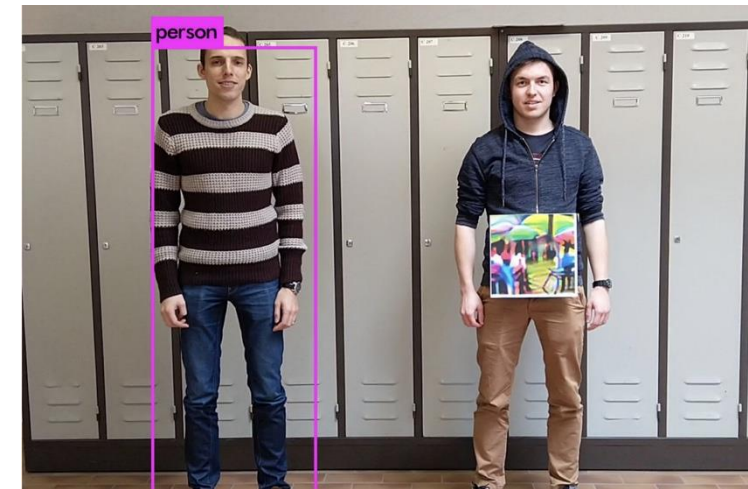
If you were the CEO of the company providing these products, how would you respond to these issues?



<http://www.usatoday.com/story/tech/2015/07/01/google-apologizes-after-photos-identify-black-people-as-gorillas/29567465/>



<https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf>

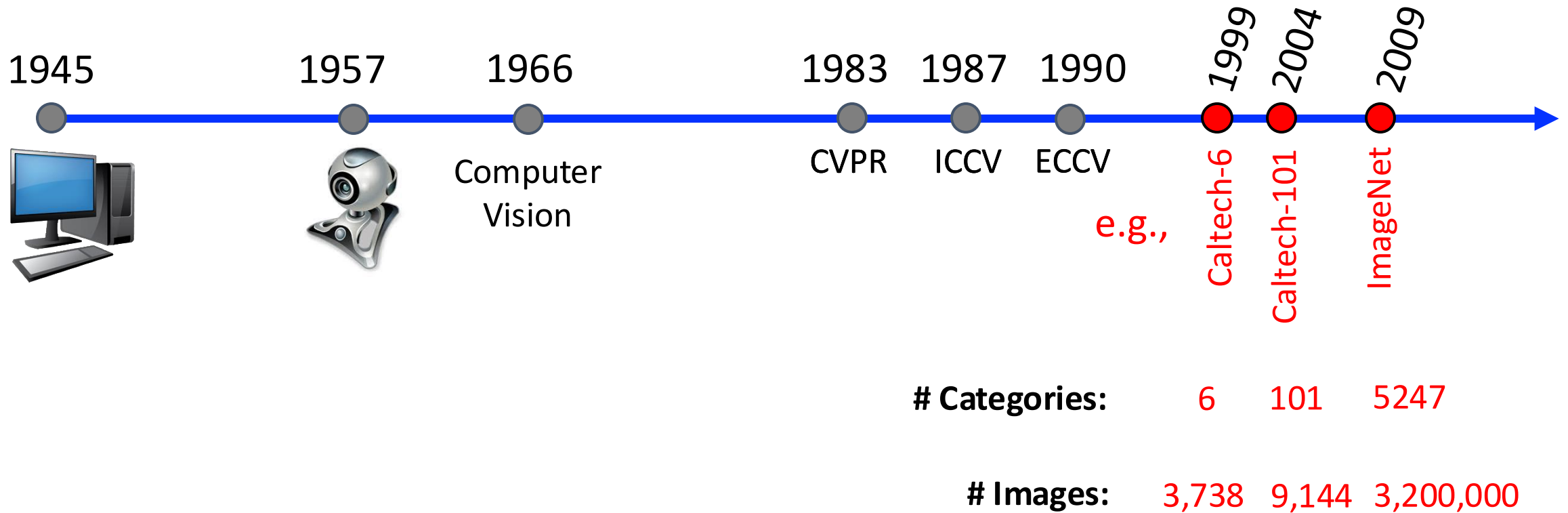


<https://www.theverge.com/2019/4/23/18512472/fool-ai-surveillance-adversarial-example-yolov2-person-detection>

Object Recognition: Today's Topics

- Problem
- Applications
- **Datasets**
- Evaluation metric
- A Popular Solution: Convolutional Neural Networks

Public Datasets Since Early ~2000s



Trend: build bigger datasets

Caltech-6

← → ↻ ⓘ Not Secure | vision.caltech.edu/html-files/archive.html

Computational Vision



Cars 2001 (Rear)

- [Tar file of images](#)
- 526 images of Cars from the rear.
- [Description](#)



Cars 1999 (Rear) 2

- [Tar file of images](#)
- 126 images of Cars from the rear.
- [Description](#)



Motorcycles 2001 (Side)

- [Tar file of images](#)
- 826 images of motorbikes from the side.
- [Description](#)



Airplanes (Side)

- [Tar file of images](#)
- 1074 images of airplanes from the side.
- [Description](#)

- (1) Six categories selected
- (2) Students took pictures and collected images from the web

Computational Vision



Faces 1999 (Front)

- [Tar file of images](#)
- 450 frontal face images of 27 or so unique people.
- [Description](#)

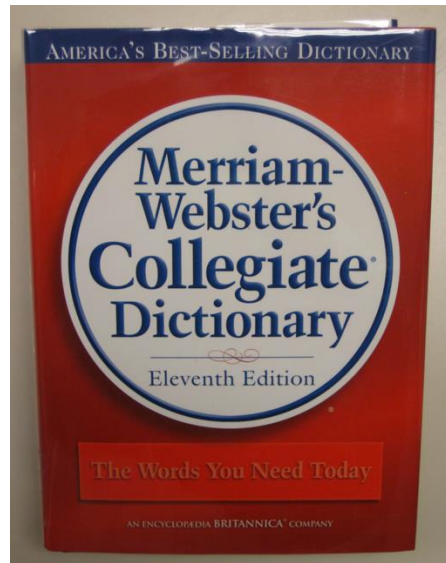


Leaves 1999

- [Tar file of images](#)
- 186 images of 3 species of leaves against cluttered backgrounds.
- [Description](#)

Caltech-101

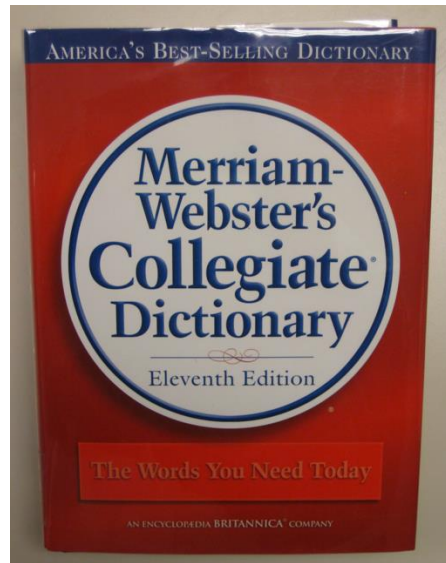
1. Category Selection



Flipped through a dictionary
and chose 101 categories
associated with a drawing

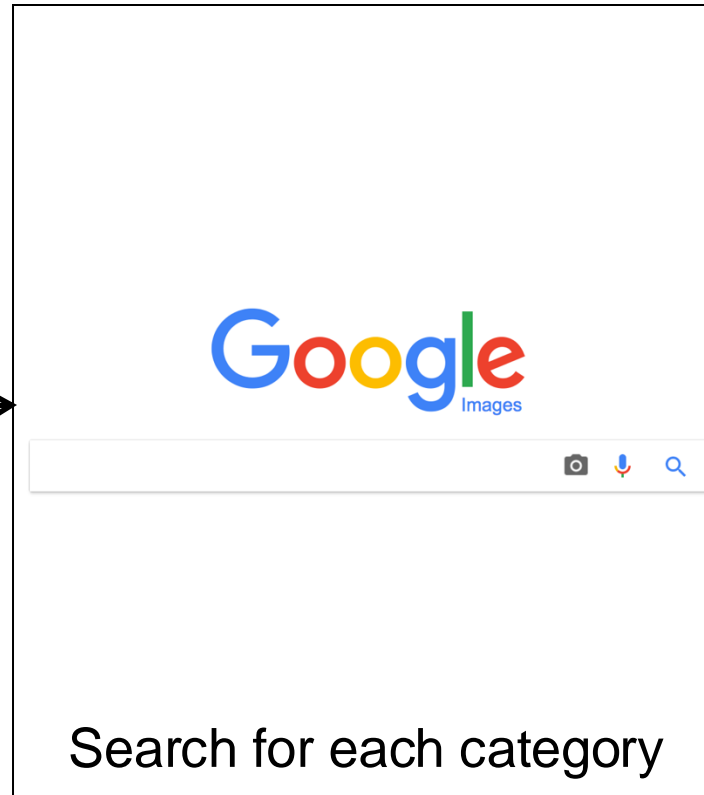
Caltech-101

1. Category Selection



Flipped through a dictionary and chose 101 categories associated with a drawing

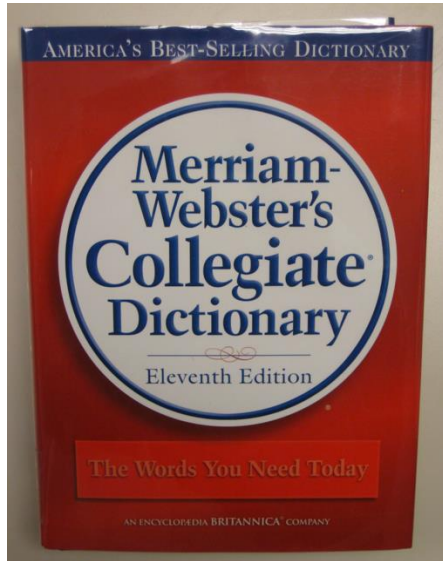
2. Image Collection



Search for each category

Caltech-101

1. Category Selection



Flipped through a dictionary and chose 101 categories associated with a drawing

2. Image Collection



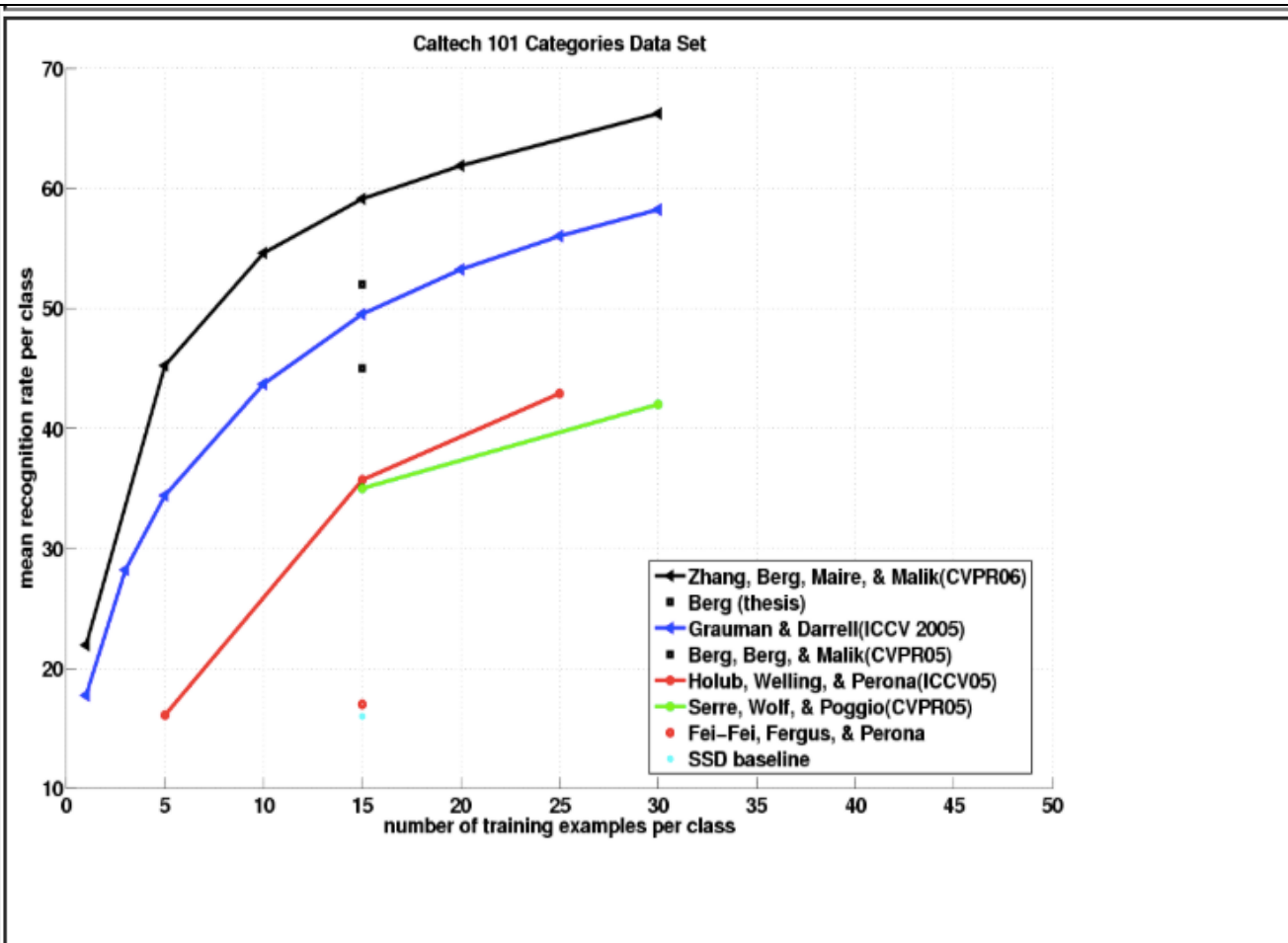
Search for each category

3. Human Verification

- 2 graduate students reviewed & discarded irrelevant images
- Result is 9,144 grayscale 300x200 pixel images with 45-400 images per category

Caltech-101

Progress of algorithms charted



Latest results (March 2006) on the Caltech 101 from a variety of groups. (published results only).

If you would like to include your algorithm's performance please email us at holub@caltech.edu or greg@vision.caltech.edu with a citation and your results. Thanks!

We are also interested in the time it takes to run your algorithm. Both during the training and during the classification stage

Plot courtesy of Hao Zhang.

Update by holub, April 2006.

ImageNet

After creating Caltech-101 and finishing her PhD, Fei-Fei Li began her career as an Assistant Professor creating ImageNet.

Hear her tell her story:



Video URL (5:44 – 9:35):

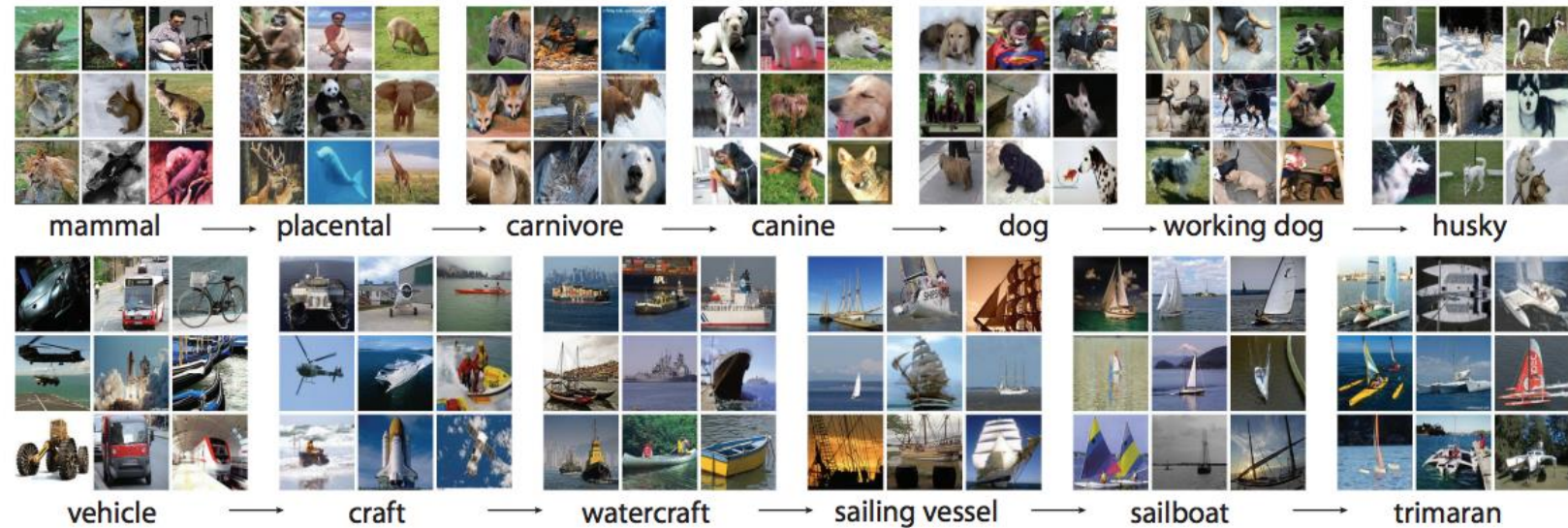
<https://www.youtube.com/watch?v=40riCqvRoMs>

ImageNet

1. Category Selection

~10% of concepts (synonym sets) in WordNet taxonomy

e.g., two root-to-leaf branches of ImageNet with nine examples for each “synonym set”



ImageNet

1. Category Selection

~10% of concepts (synonym sets) in WordNet taxonomy

2. Image Collection

flickr

(& more search engines)

Query expansion:

- Augment queries
- Translate queries to different languages

ImageNet

Key Insight: use crowdsourcing to recruit many people to verify images

1. Category Selection

~10% of concepts (synonym sets) in WordNet taxonomy

2. Image Collection

flickr

(& more search engines)

Query expansion:

- Augment queries
- Translate queries to different languages

3. Human Verification

- Humans verify if image contains queried object

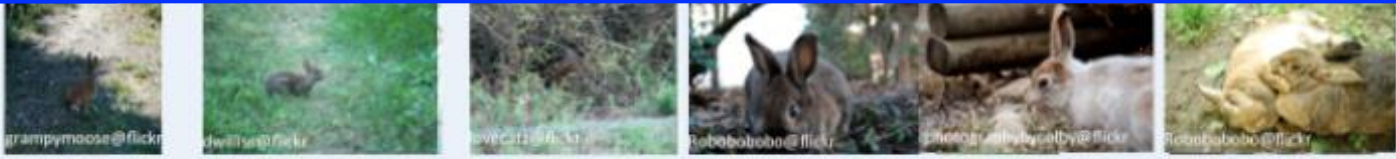
- Use majority vote decision from multiple humans to support high quality results

ImageNet: Crowdsourcing Task


Definition of the target synonym set with link to Wikipedia.

Main Instructions


Good Examples (mouse over to enlarge):



Bad Examples (COMMON MISTAKES)



Please click on the images that contain rabbit



< page 1 of 6 > Submit

Submit button will be enabled on the final page.

ImageNet: Crowdsourcing Platform

Mechanical Turk is a marketplace for work.

We give businesses and developers access to an on-demand, scalable workforce.
Workers select from thousands of tasks and work whenever it's convenient.

400,794 HITS available. [View them now.](#)

Make Money by working on HITS

HITS - *Human Intelligence Tasks* - are individual tasks that you work on. [Find HITS now.](#)

As a Mechanical Turk Worker you:

- Can work from home
- Choose your own work hours
- Get paid for doing good work



or [learn more about being a Worker](#)

Get Results from Mechanical Turk Workers

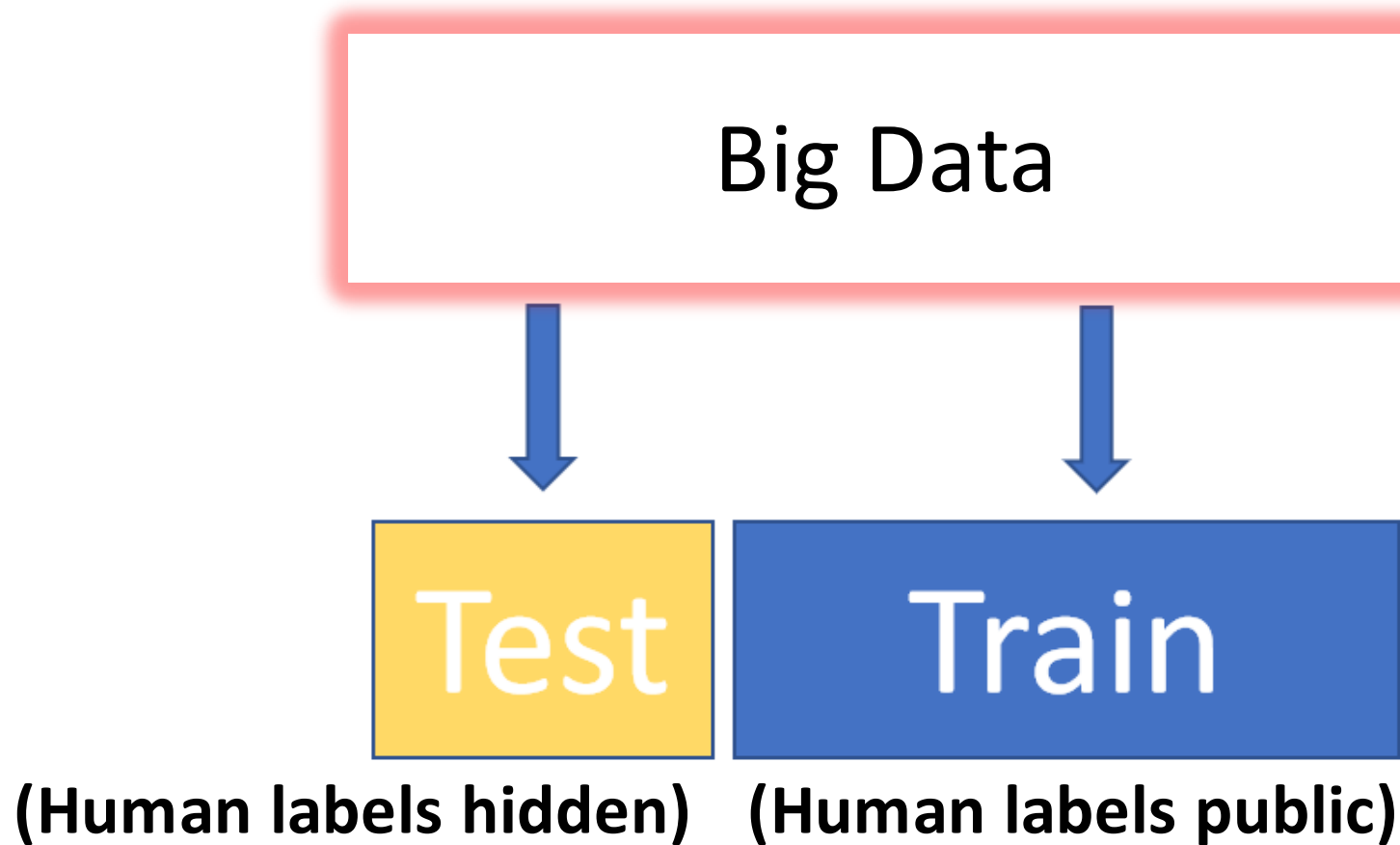
Ask workers to complete HITS - *Human Intelligence Tasks* - and get results using Mechanical Turk. [Get Started.](#)

As a Mechanical Turk Requester you:

- Have access to a global, on-demand, 24 x 7 workforce
- Get thousands of HITS completed in minutes
- Pay only when you're satisfied with the results

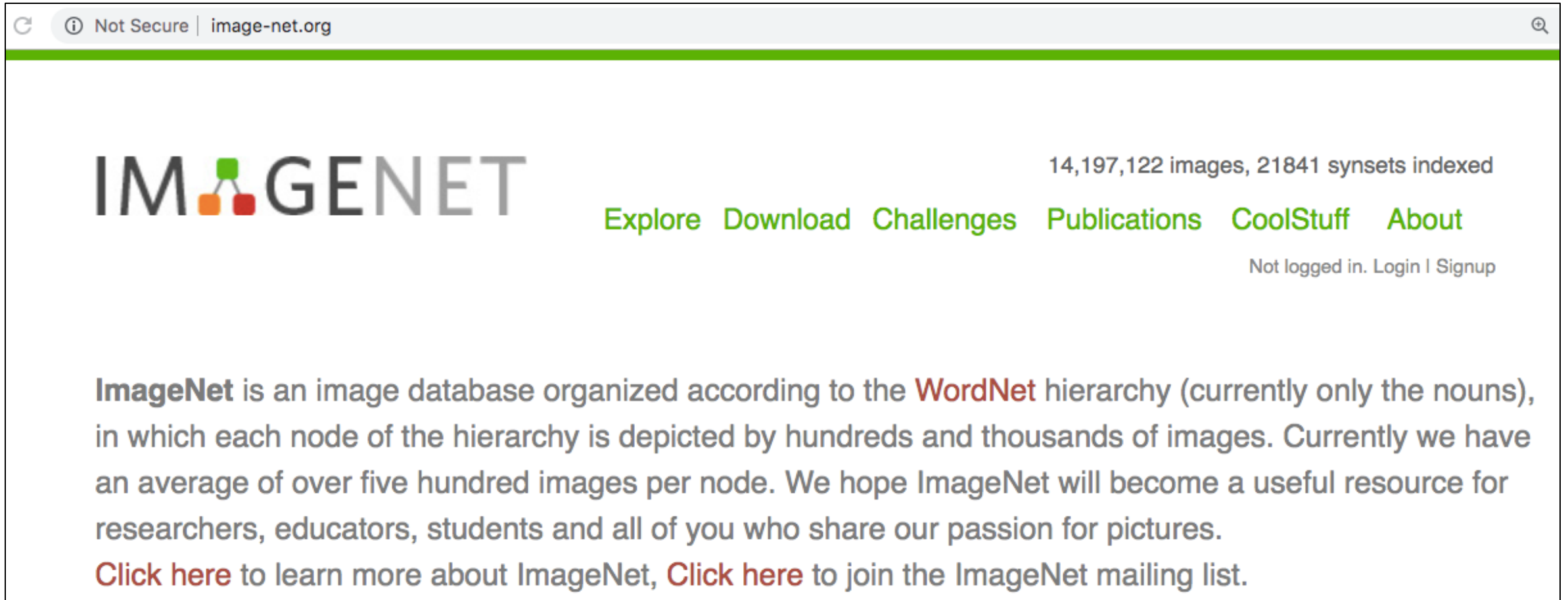


ImageNet: Challenge



Winner: highest scoring method on the hidden test set

ImageNet: Website with Evaluation Server



The screenshot shows the homepage of the ImageNet website. At the top, the browser address bar displays "image-net.org" with a "Not Secure" warning. The main header features the "IMAGENET" logo, where the "A" is replaced by a small tree diagram with three colored nodes (green, orange, red). To the right of the logo, it states "14,197,122 images, 21841 synsets indexed". Below this, a navigation menu includes links for "Explore", "Download", "Challenges", "Publications", "CoolStuff", and "About". A user status indicator shows "Not logged in. Login | Signup". The main content area contains a paragraph describing ImageNet as an image database organized by the WordNet hierarchy, and two red links: "Click here" to learn more and "Click here" to join the mailing list.

image-net.org

IMAGENET

14,197,122 images, 21841 synsets indexed

[Explore](#) [Download](#) [Challenges](#) [Publications](#) [CoolStuff](#) [About](#)

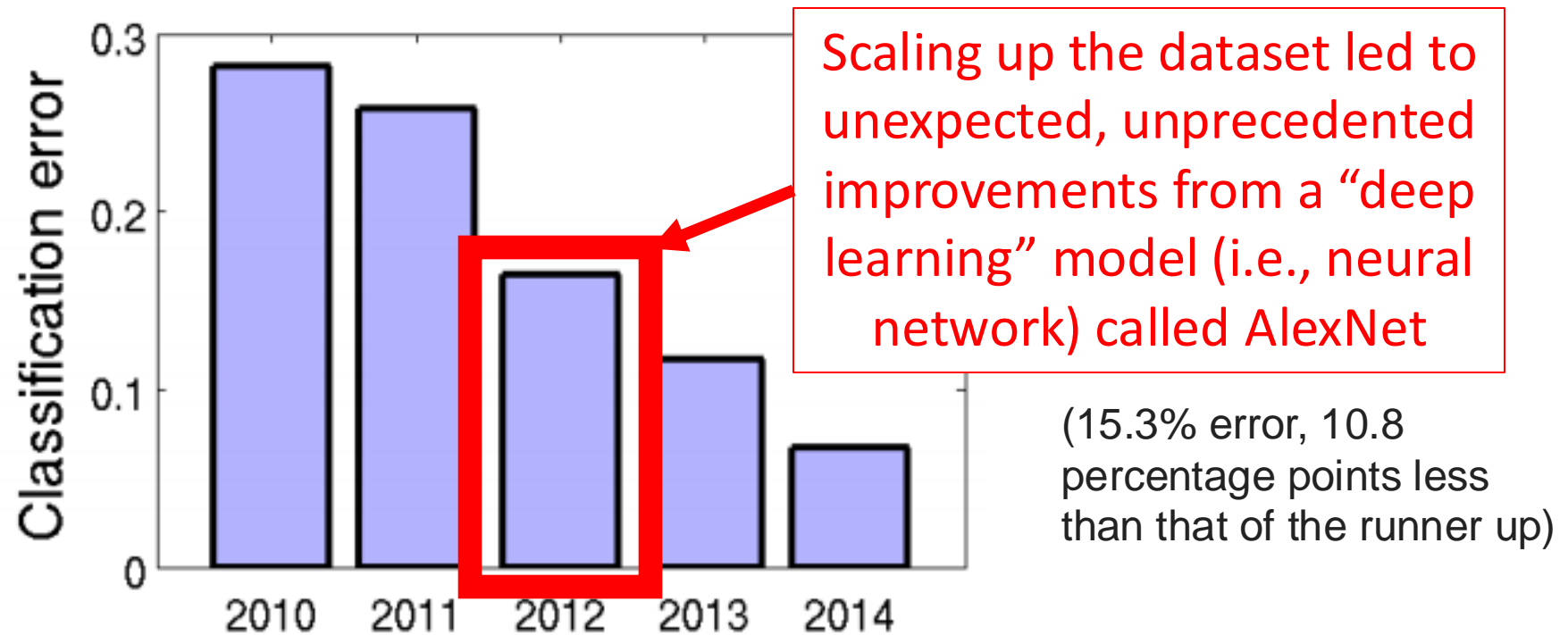
Not logged in. [Login](#) | [Signup](#)

ImageNet is an image database organized according to the **WordNet** hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently we have an average of over five hundred images per node. We hope ImageNet will become a useful resource for researchers, educators, students and all of you who share our passion for pictures.

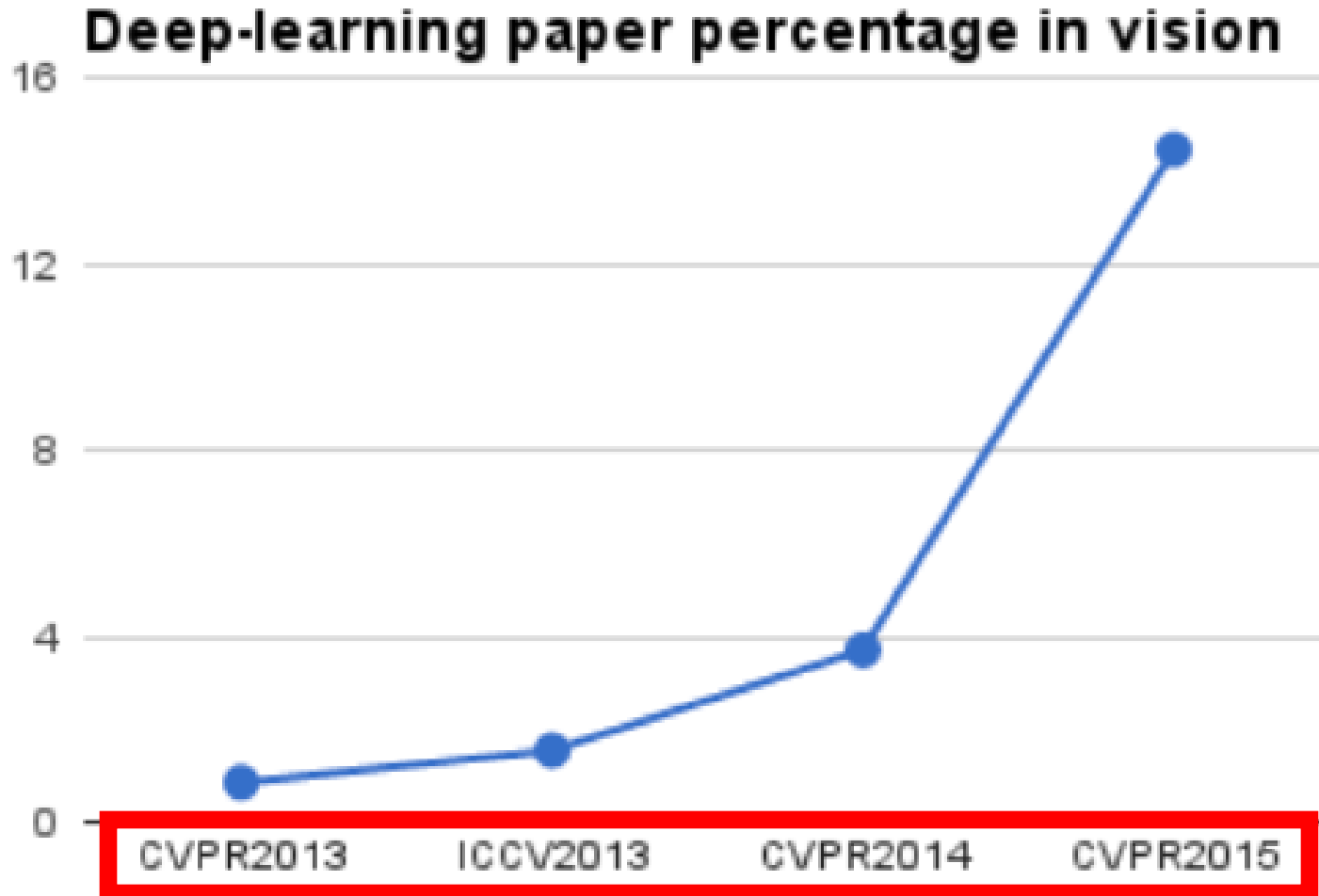
[Click here](#) to learn more about ImageNet, [Click here](#) to join the ImageNet mailing list.

ImageNet: Catalyst for Revolution

Progress of models on ImageNet

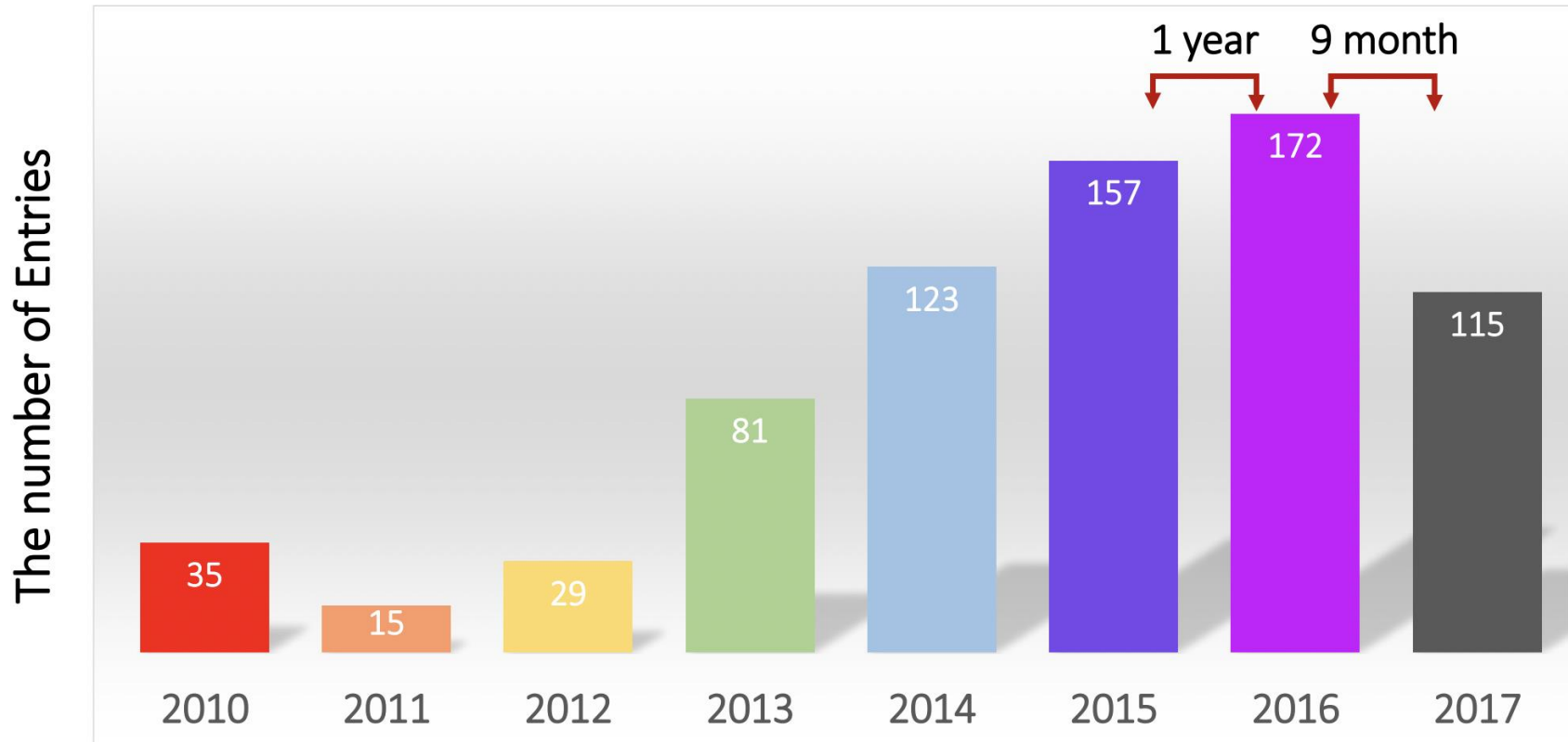


ImageNet: Catalyst for Revolution



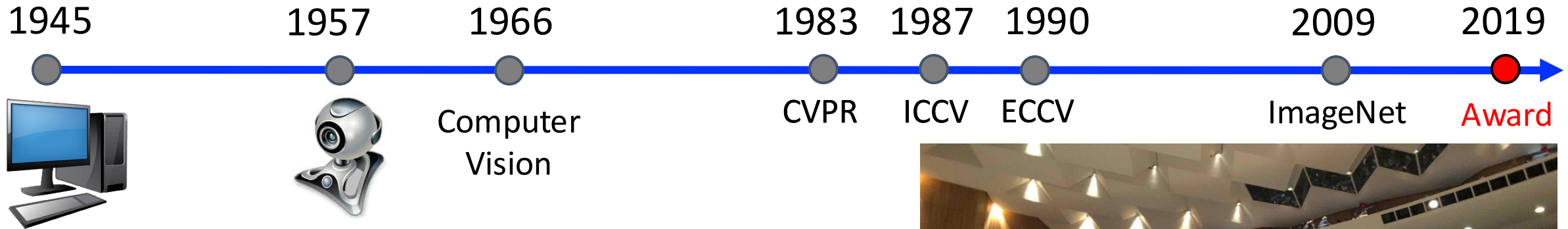
Inspired, many more researchers in the computer vision community focused on neural networks and discovered they succeed for many more vision problems!

ImageNet: Challenge Engagement



- 727 entries (plus an entry from Baidu that famously was kicked out in 2015 for cheating)
- Labor cost ~\$110 million: assuming 3 people contribute to each entry and \$50k cost per person

ImageNet Impact Recognized



PAMI Longuet-Higgins Prize

Retrospective Most Impactful Paper from CVPR 2009

ImageNet: A large-scale hierarchical image database

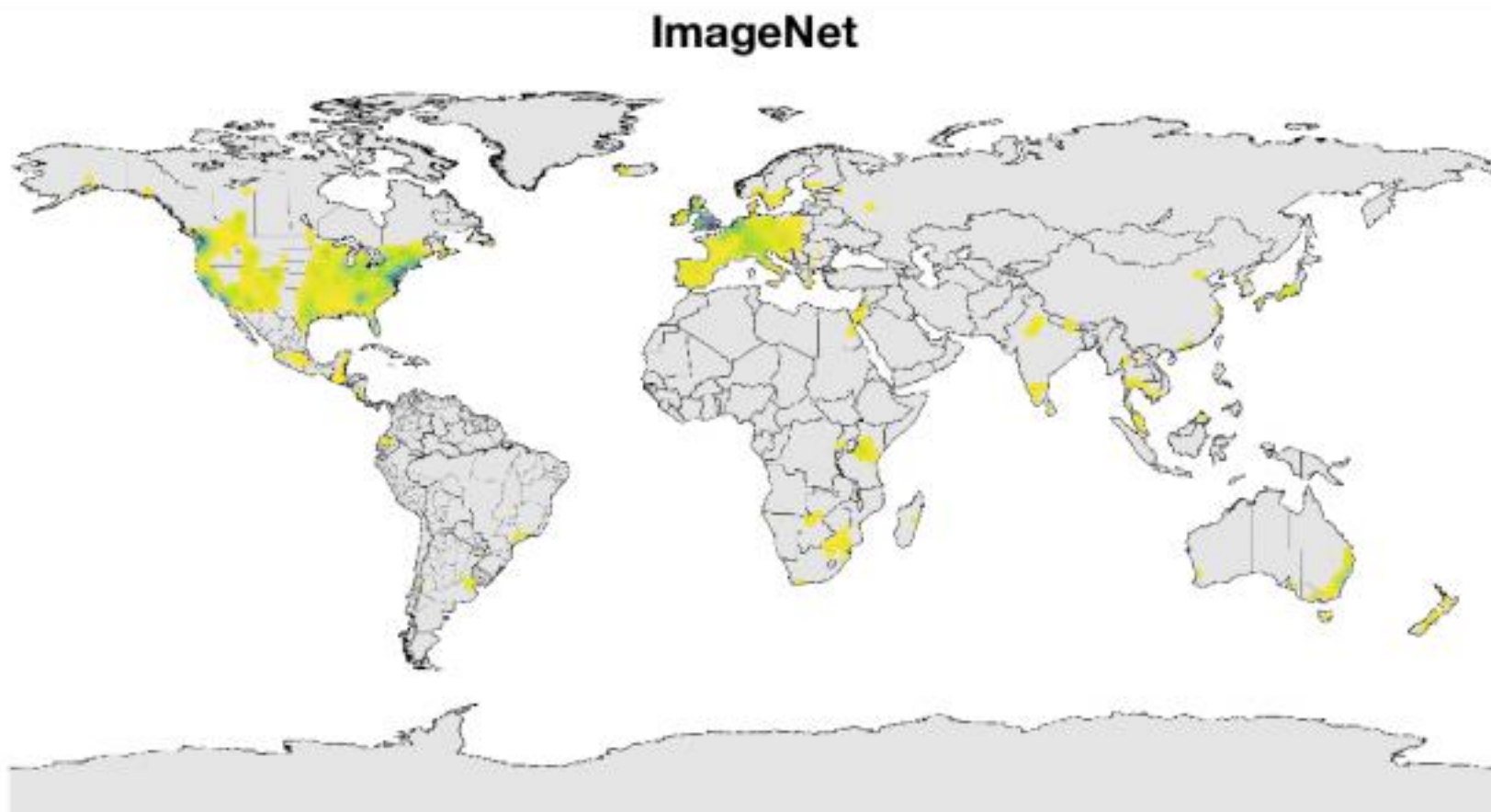
Jia Deng, Wei Dong, Richard Socher,
Li-Jia Li, Kai Li, and Li Fei-Fei



<https://syncedreview.com/2019/06/18/cvpr-2019-attracts-9k-attendees-best-papers-announced-imagenet-honoured-10-years-later/>

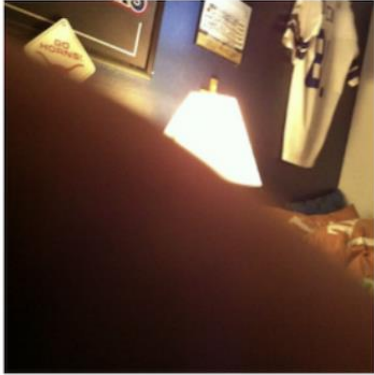



ImageNet: Great Start...

Geographical origins of images in the ImageNet using Flickr metadata:



ImageNet: Great Start...

Differences between images in ImageNet and those taken by blind photographers:

			
<p>Labels: Table lamp, lampshade, and T-shirt</p>	<p>Labels: Table lamp, lampshade, and studio couch</p>	<p>Labels: Table lamp</p>	<p>Labels: Table lamp</p>
<p>Quality issues: Obscured and Framing</p>	<p>Quality issues: Blur</p>	<p>Quality issues: N/A</p>	<p>Quality issues: N/A</p>
	<p>VizWiz-Classification</p>		<p>ImageNet</p>

Object Recognition: Today's Topics

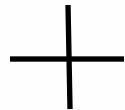
- Problem
- Applications
- Datasets
- **Evaluation metric**
- A Popular Solution: Convolutional Neural Networks

Goal: Design Models that **Generalize Well** to New, Previously Unseen Examples

Apply model on “**test set**” to measure generalization error



Prediction Model



Input:



Label:

?

?



?

Evaluation Metric for ImageNet Challenge

Assumption: 1 ground truth label per image

Top N error: average over all test images using this rule per image:

- * 0 if any of N predictions match the ground truth
- * 1 otherwise

e.g., top 5 error

Steel drum



Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle



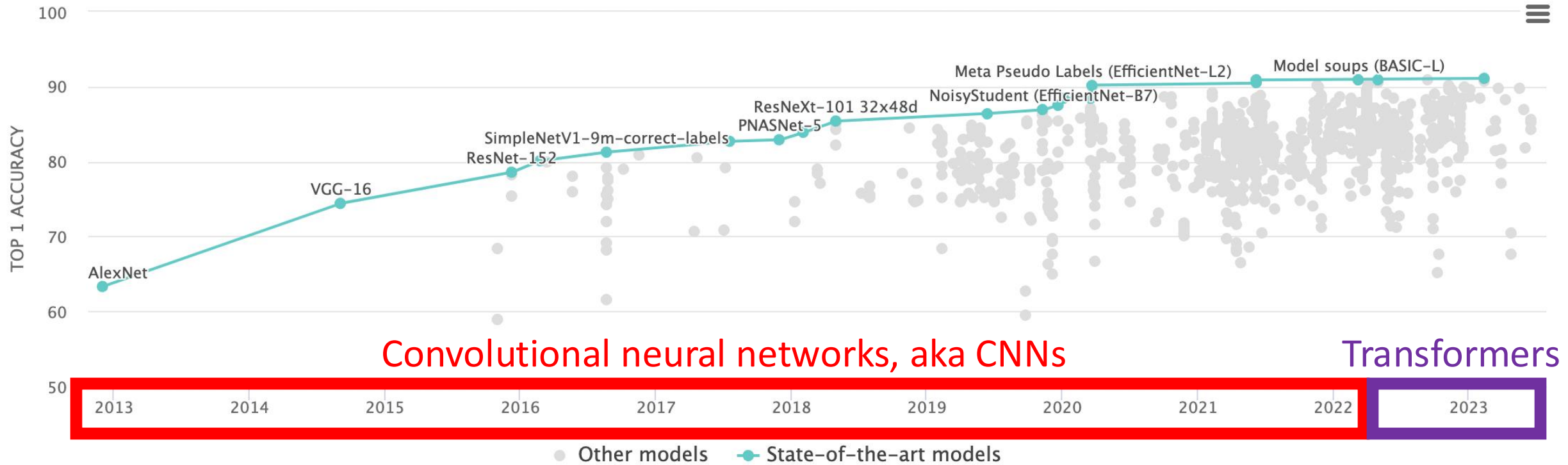
Output:
Scale
T-shirt
Giant panda
Drumstick
Mud turtle



Object Recognition: Today's Topics

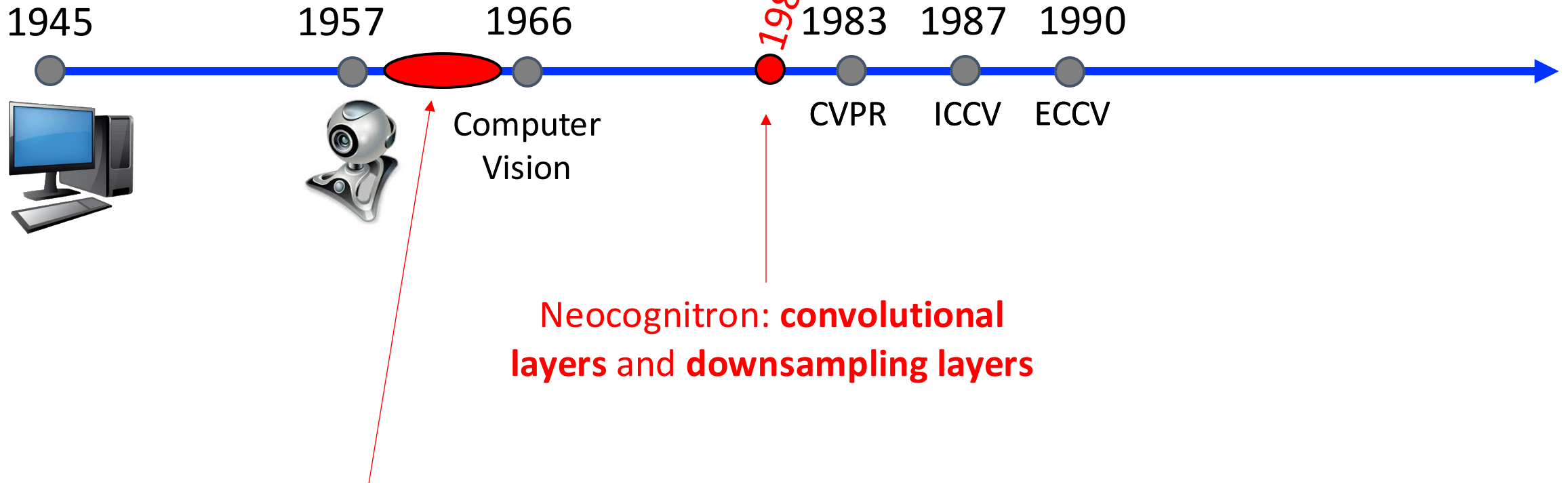
- Problem
- Applications
- Datasets
- Evaluation metric
- A Popular Solution: Convolutional Neural Network

ImageNet Challenge Winners Over Time



CNN Origins

Nobel Prize in Physiology and
Medicine to Hubel and Weisel



Neuroscientific experiments by Hubel & Weisel to understand how mammalian vision system works

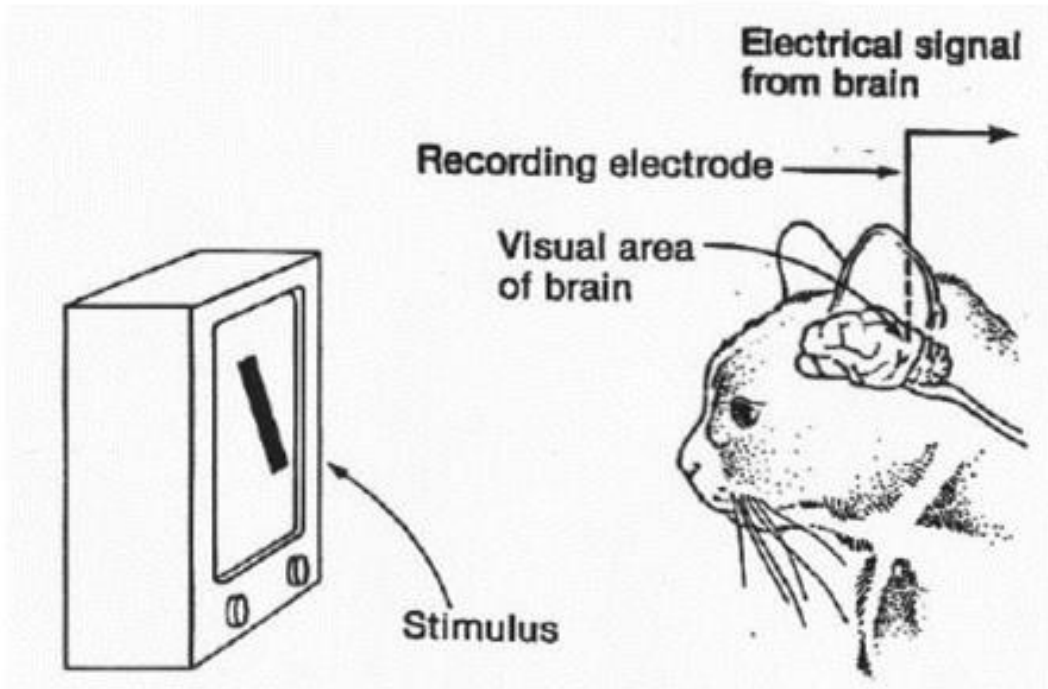
CNN Motivation: How Vision System Works



Nobel Prize winning insight from Hubel and Wiesel...

CNN Motivation: How Vision System Works

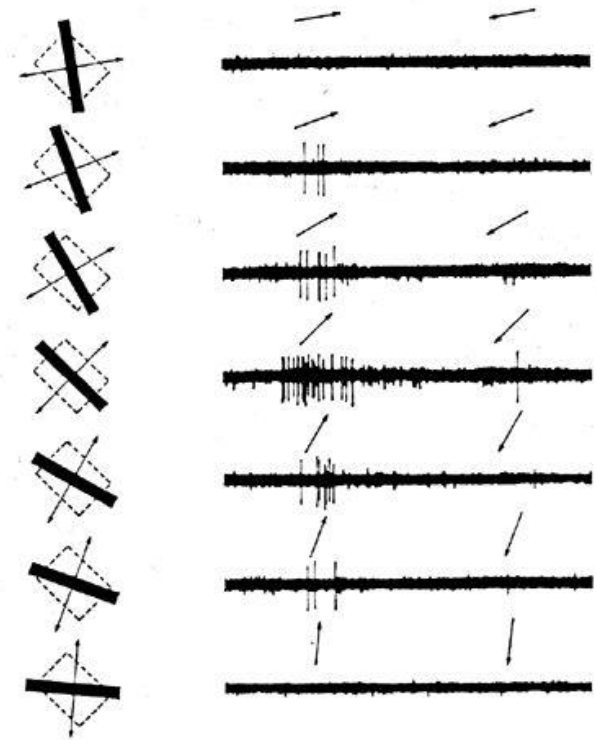
Experiment Set-up:



<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 is shown in certain orientations

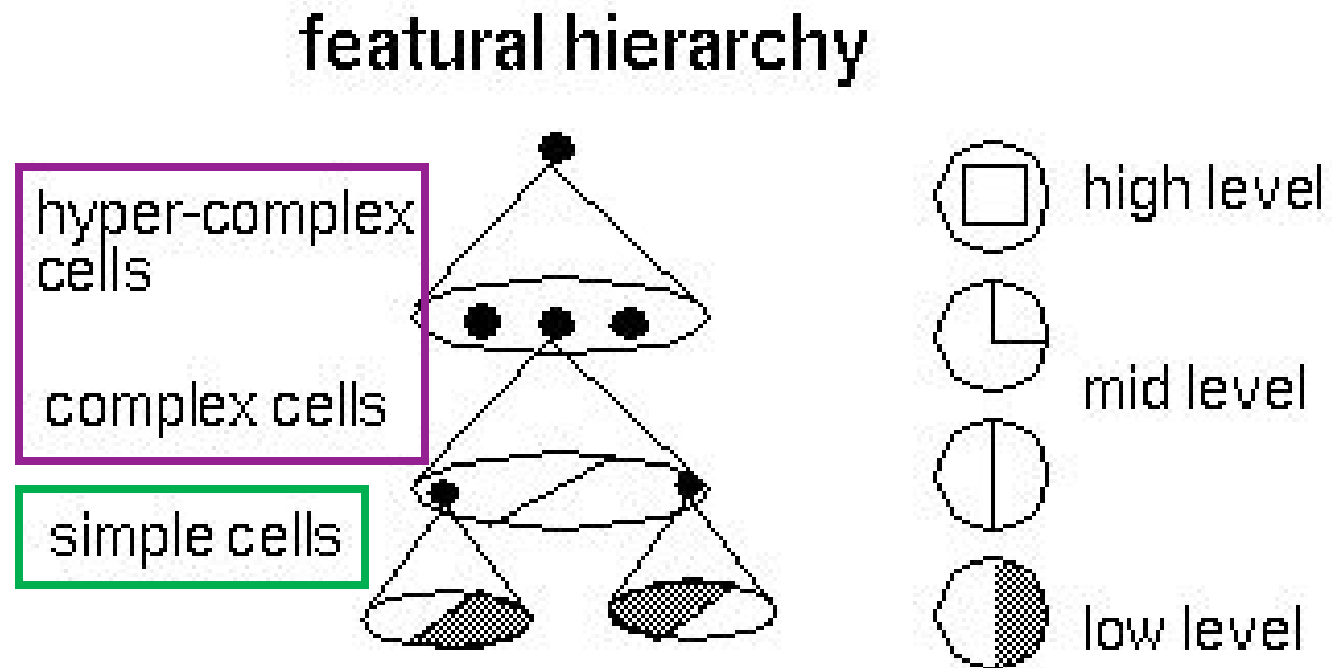
V1 physiology:
direction
selectivity



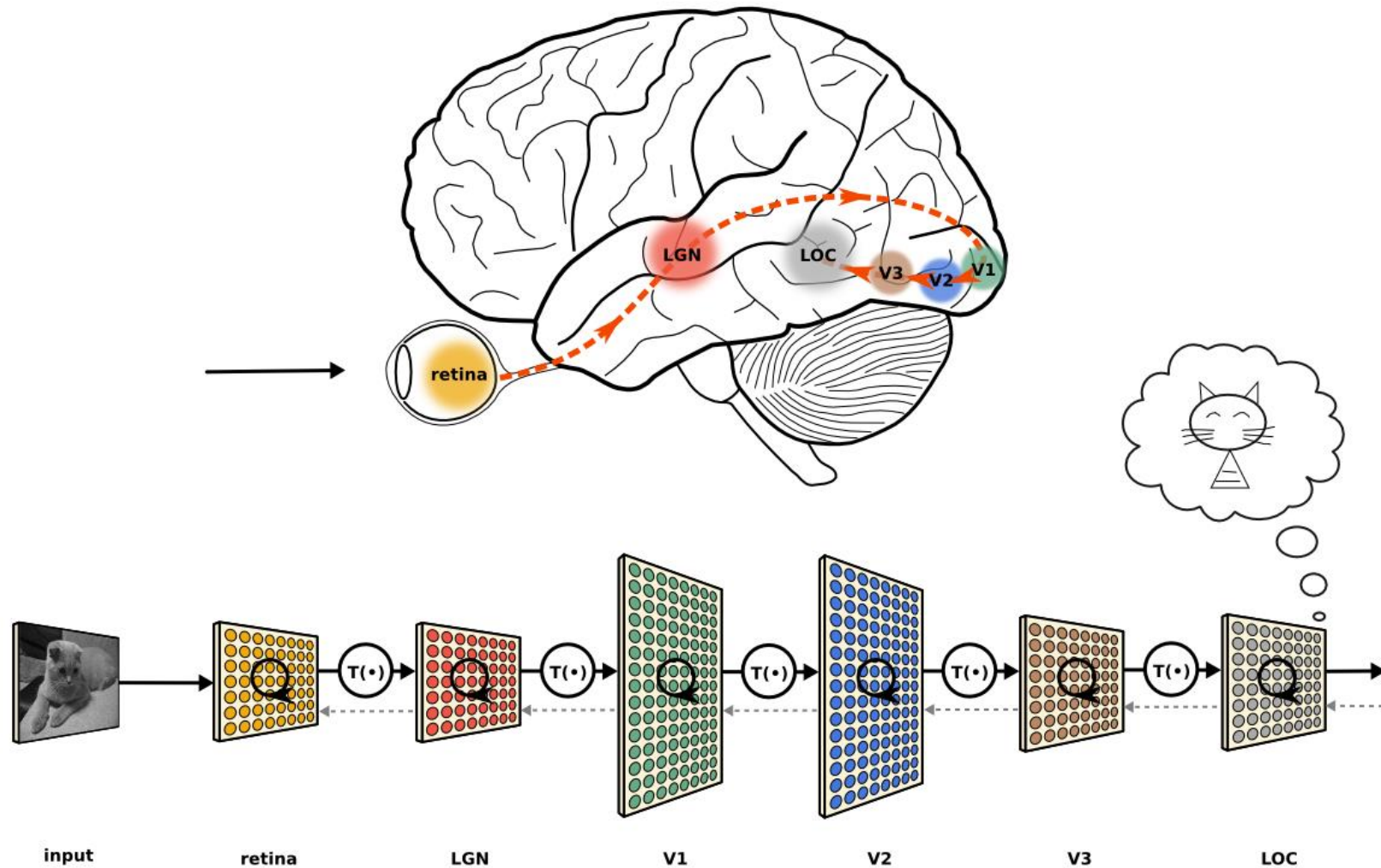
<https://www.cns.nyu.edu/~david/courses/perception/lecturenotes/V1/Ign-V1.html>

CNN Motivation: How Vision System Works

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



CNN Motivation: How Vision System Works



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:

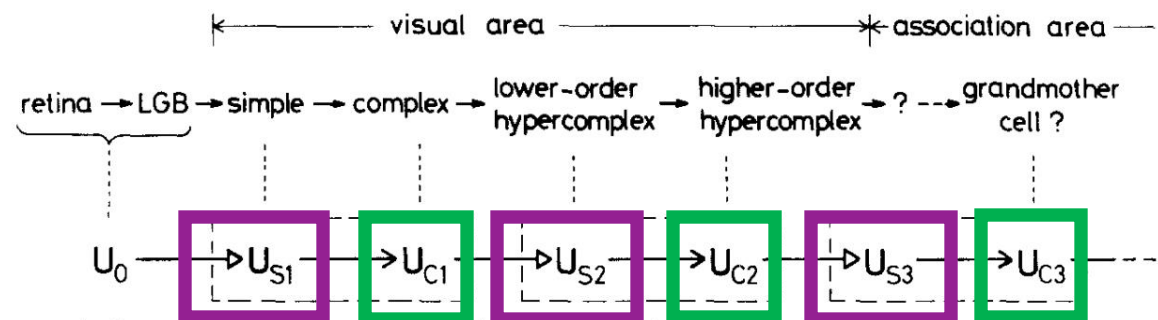


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron

1. Convolutional layers

→ modifiable synapses

→ unmodifiable synapses

2. Pooling Layers

(Modern networks also alternate between these two types of layers!)

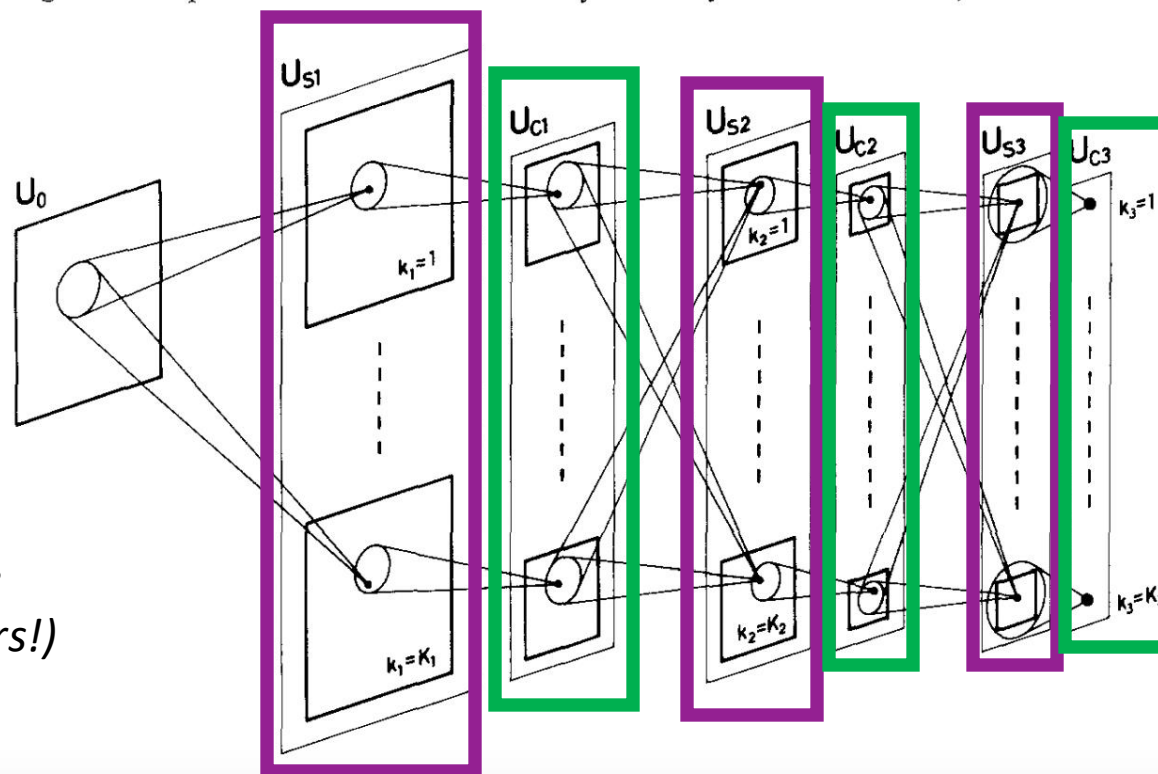
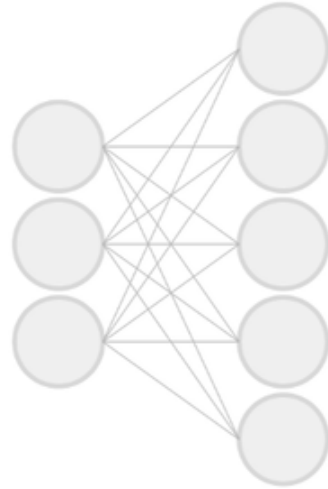


Fig. 2. Schematic diagram illustrating the interconnections between layers in the neocognitron
Fukushima, 1980

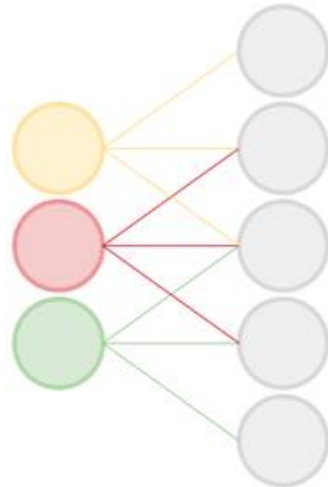
Key Ingredient 1: Convolutional Layers

Fully-connected:



Rather than have each node provide input to each node in the next layer...

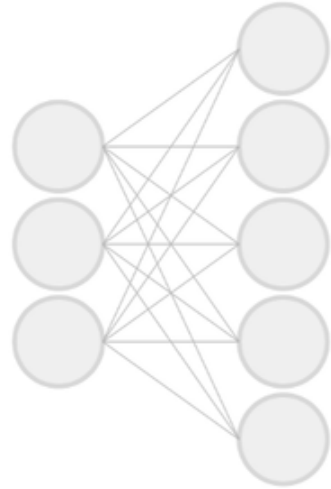
Convolutional:



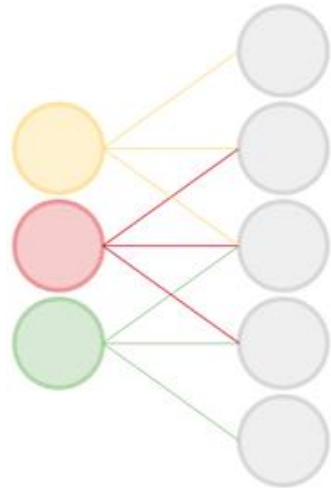
each node receives input only from a small neighborhood in previous layer (and there is parameter sharing)

Key Ingredient 1: Convolutional Layers

Fully-connected:

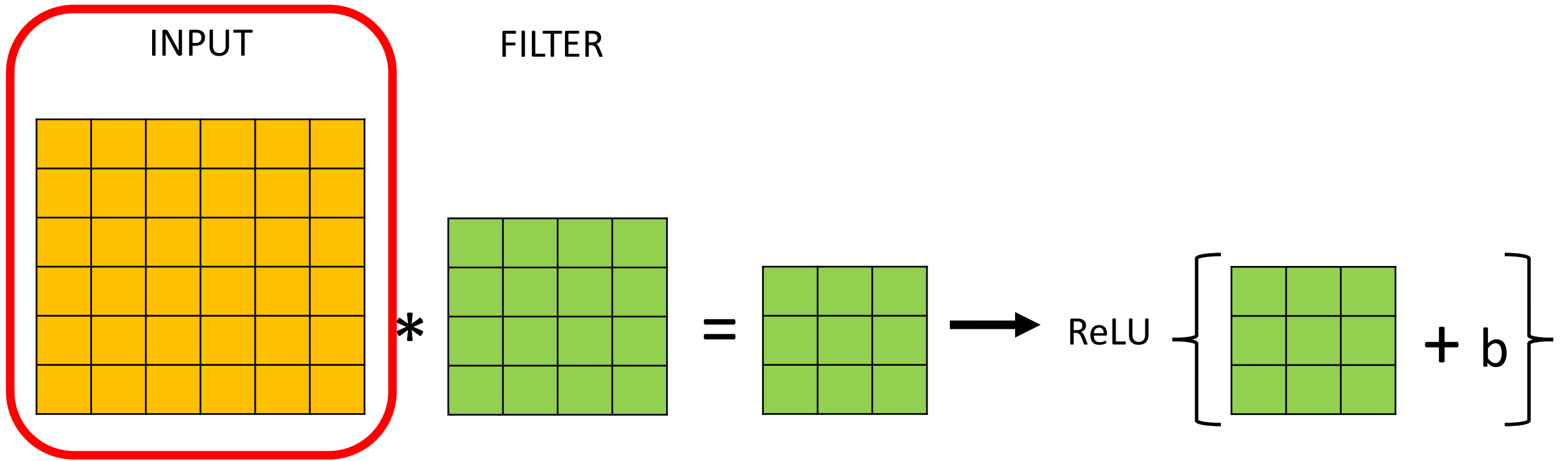


Convolutional:



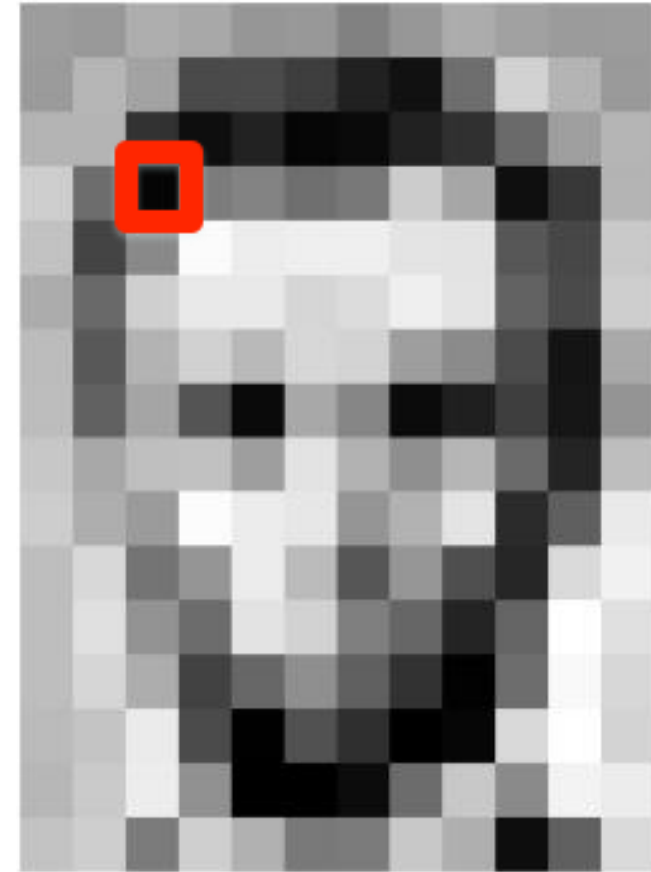
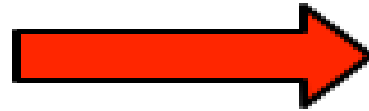
Convolutional layers dramatically reduce number of model parameters!

Key Ingredient 1: Convolutional Layers

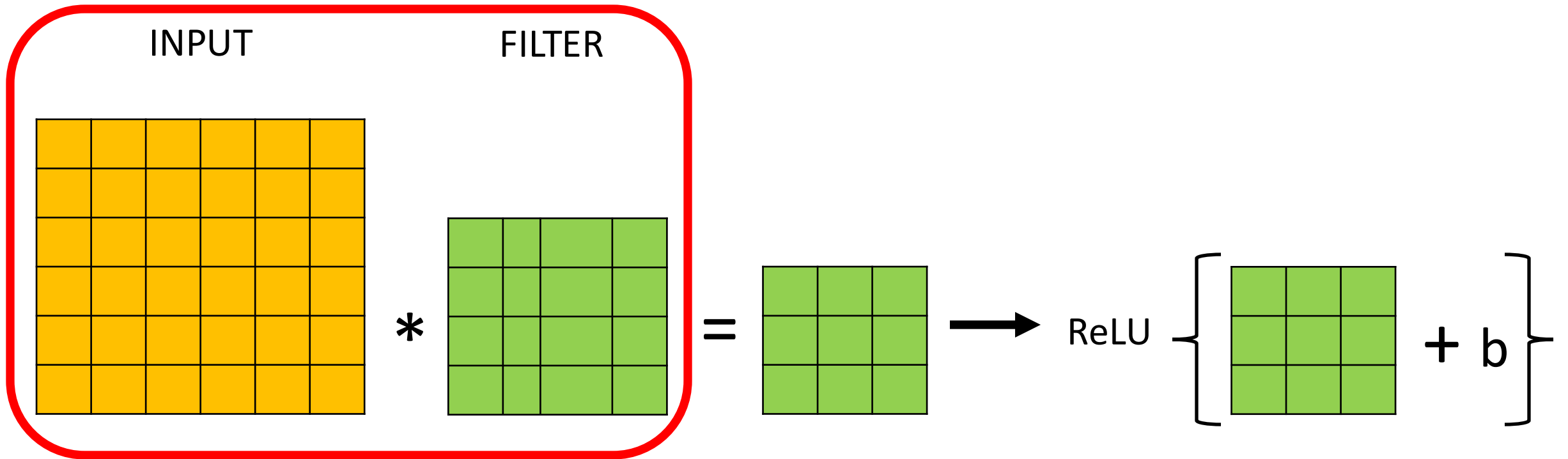


Recall: Image Representation (8-bit Grayscale)

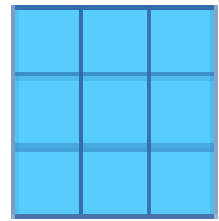
157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	105	5	14	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218



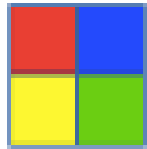
Key Ingredient 1: Convolutional Layers



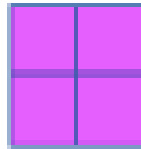
Convolution: Applies Linear Filter (e.g., 2D)



Input

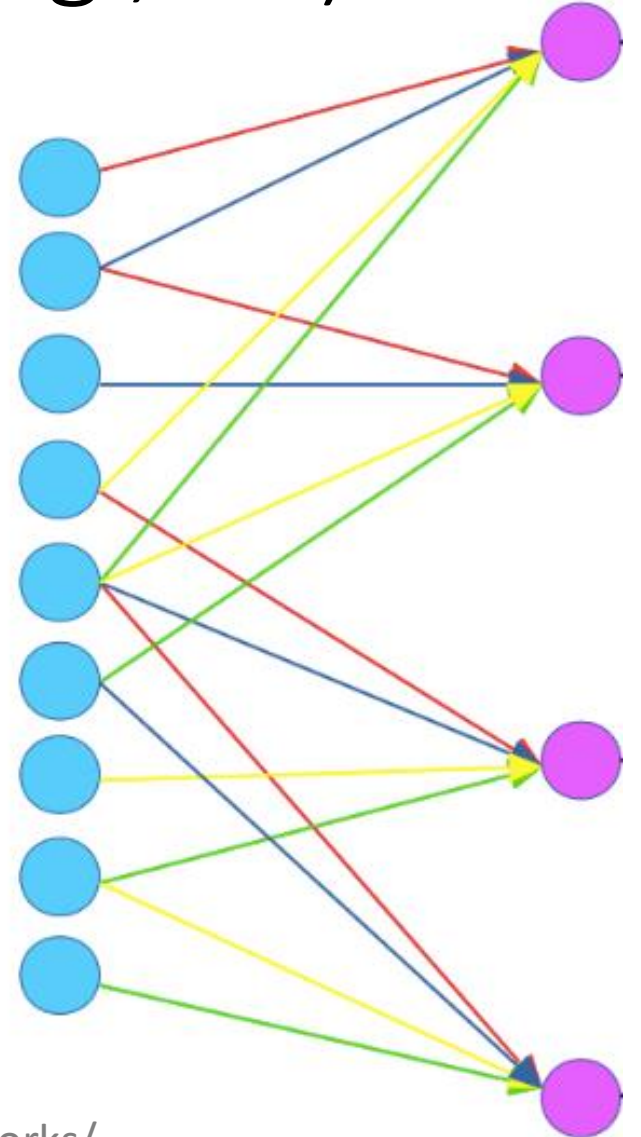


Filter
(aka – Kernel)



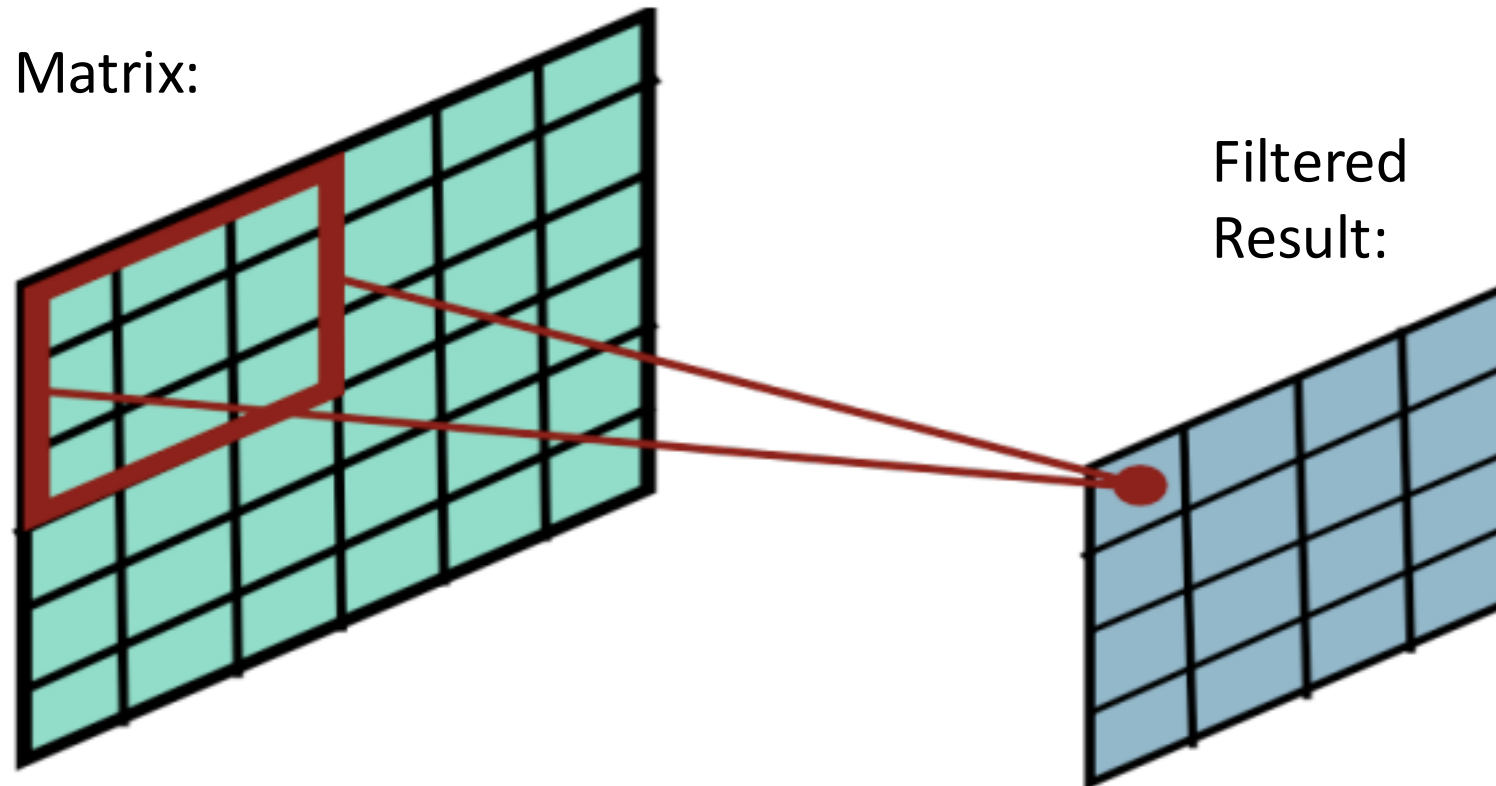
Feature
Map

Way to Interpret
Neural Network



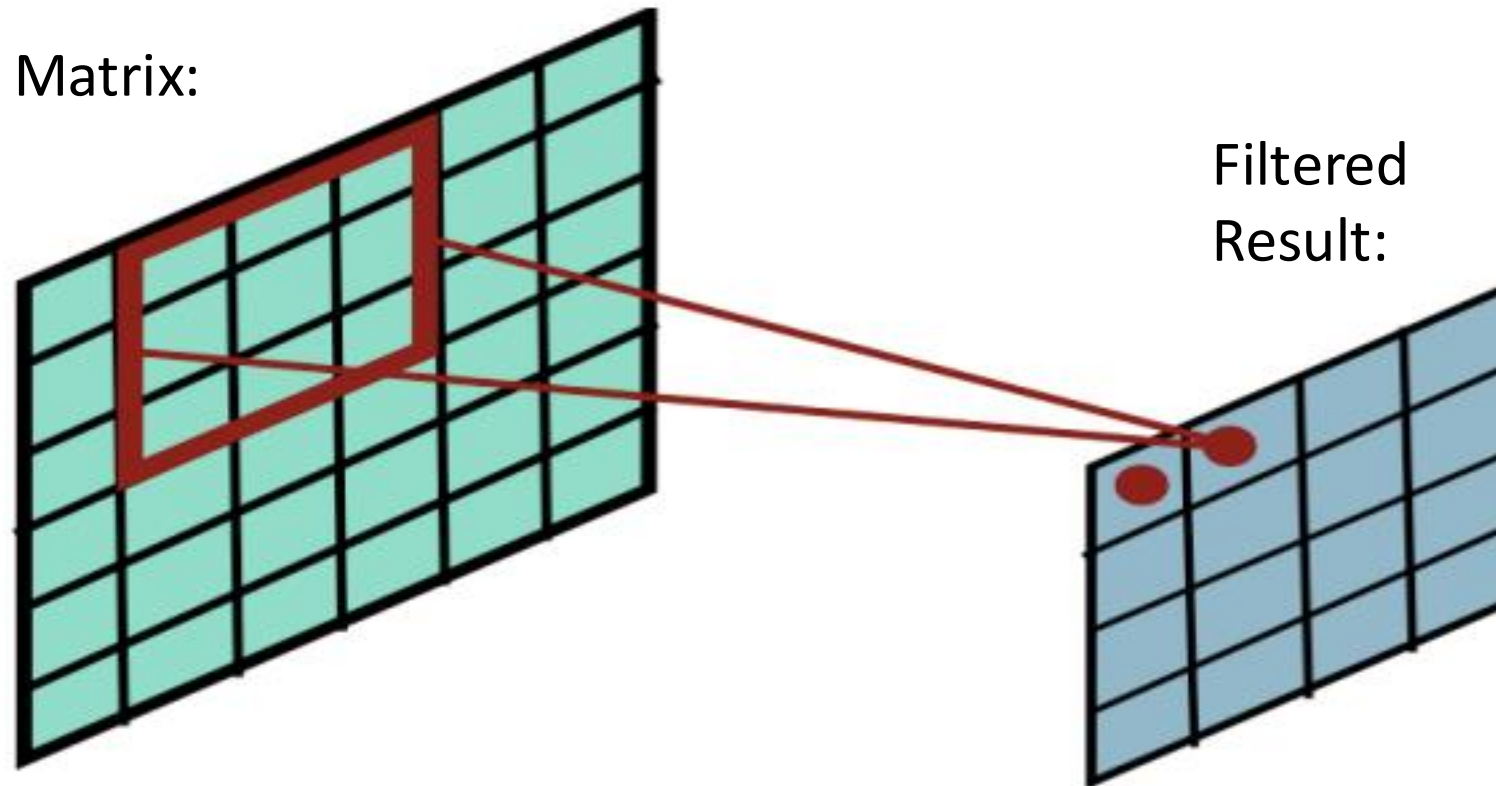
- Compute a **function of local neighborhood** for each location in matrix
- A **filter** specifies the function for how to combine neighbors' values

2D Filtering



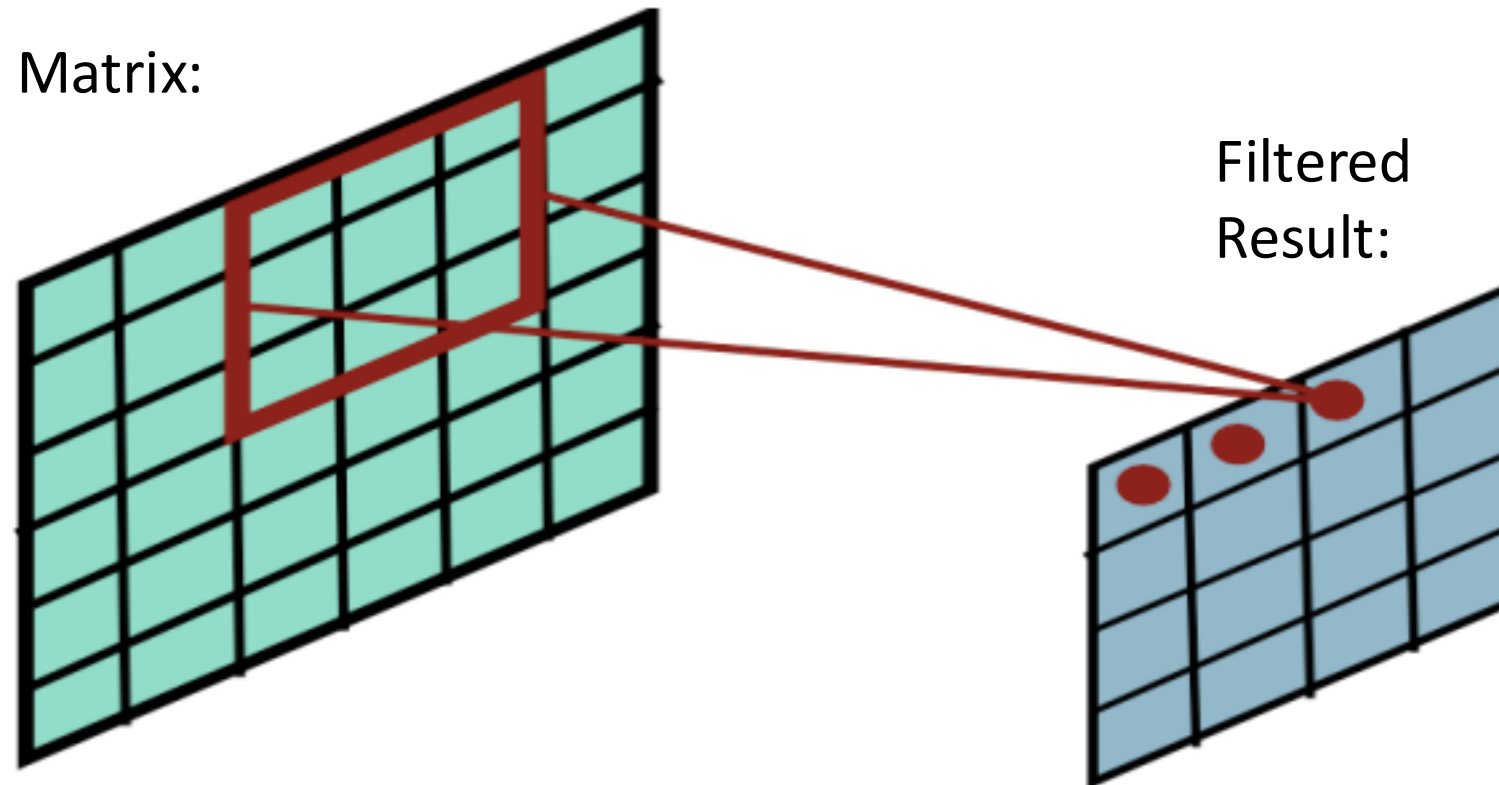
Slides filter over the matrix and computes dot products

2D Filtering



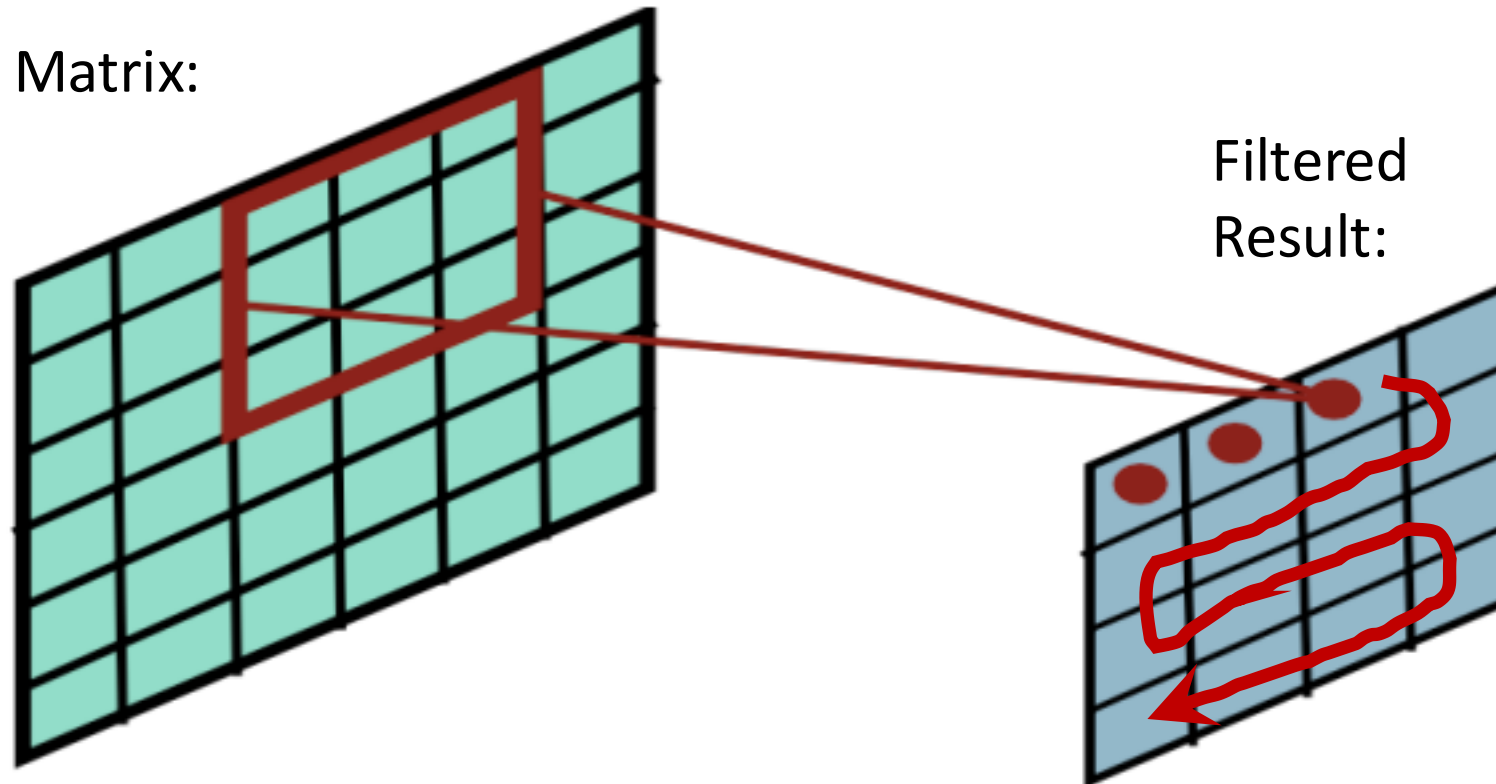
Slides filter over the matrix and computes dot products

2D Filtering



Slides filter over the matrix and computes dot products

2D Filtering



Slides filter over the matrix and computes dot products

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

?	?	?
?	?	?
?	?	?

$$\text{Product} = 1*1 + 1*0 + 1*1 + 0*0 + 1*1 + 1*0 + 0*1 + 0*1 + 0*0 + 0*0 + 1*1$$

$$\text{Product} = 4$$

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

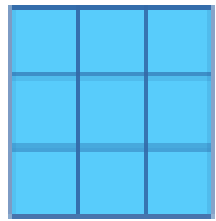
1	0	1
0	1	0
1	0	1

Feature Map

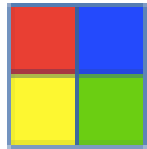
4	3	4
2	4	3
2	3	4

Convolutional Layer

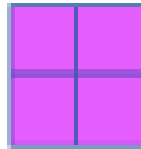
- Many neural network libraries use “convolution” interchangeably with “cross correlation”; for mathematicians, these are technically different
- Examples in these slides show the “cross-correlation” function



Input

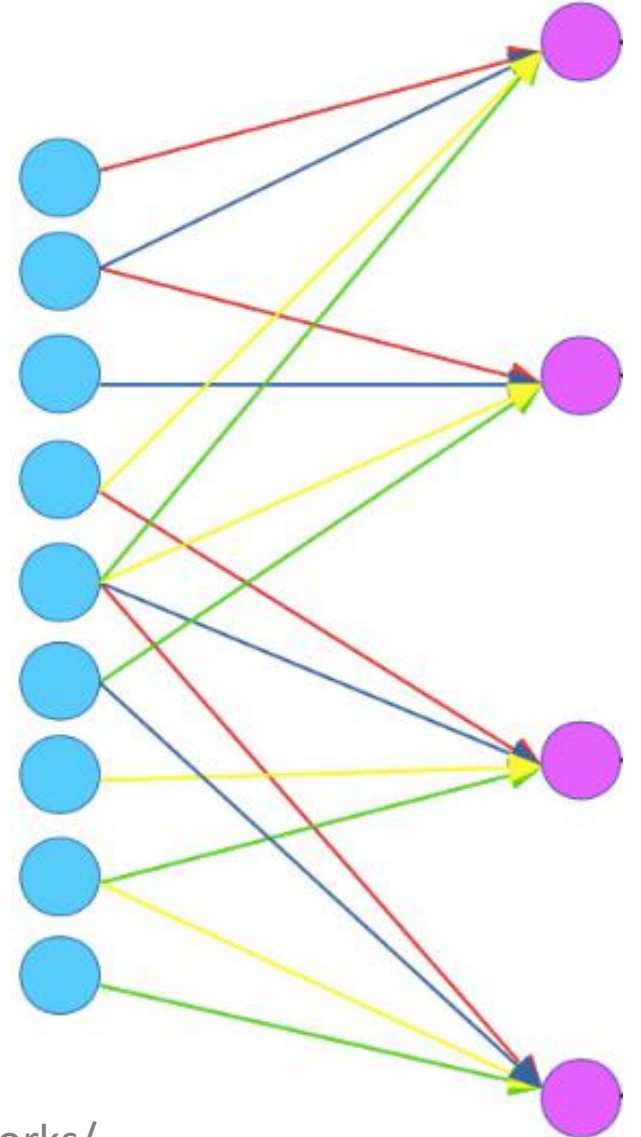


Filter
(aka – Kernel)

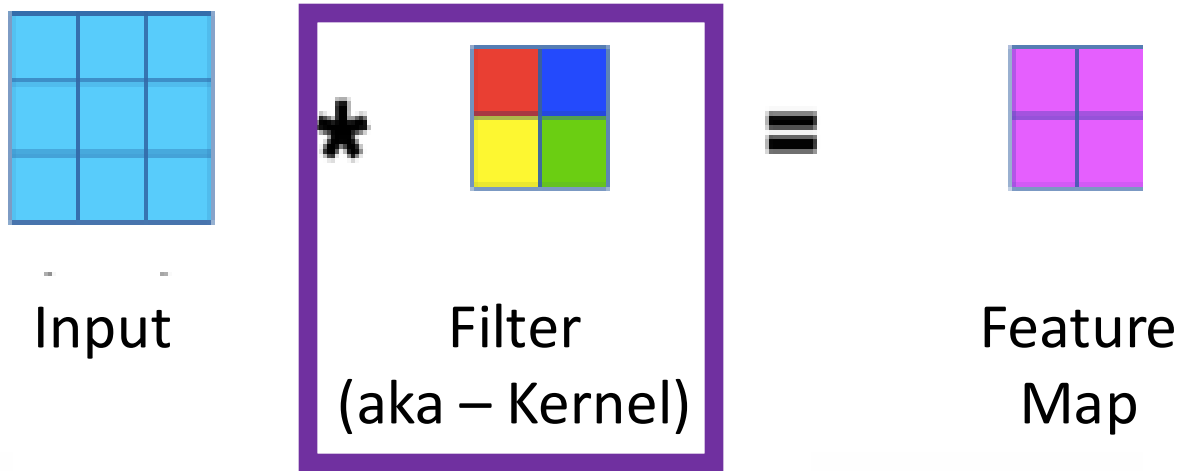


Feature
Map

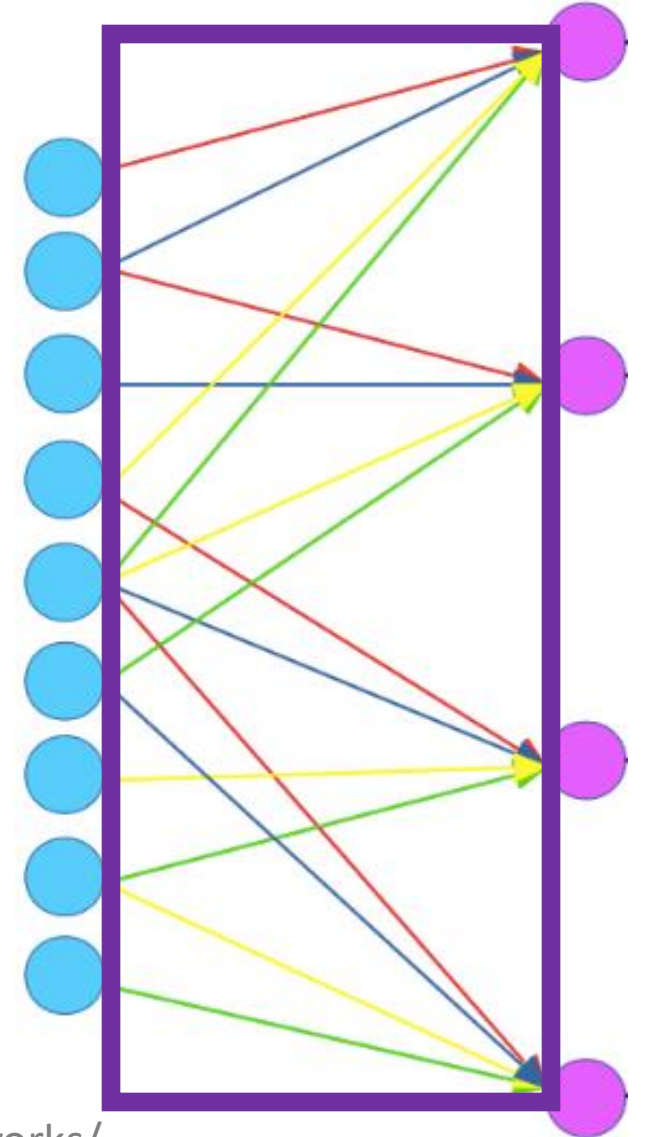
Way to Interpret
Neural Network



Convolutional Layer: Parameters to Learn

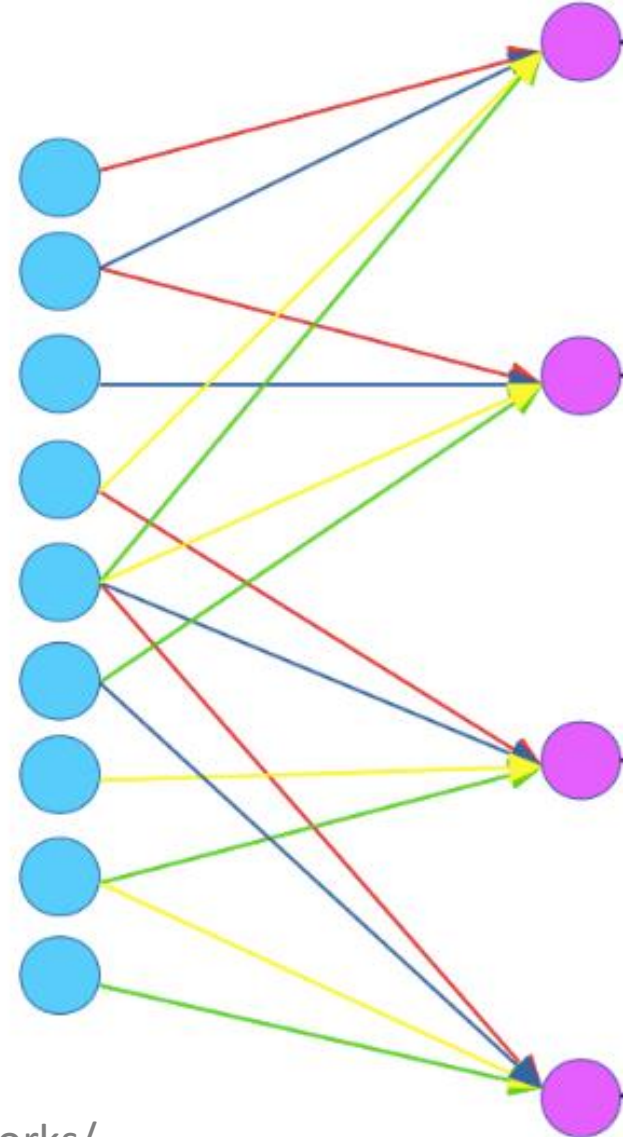


Way to Interpret
Neural Network



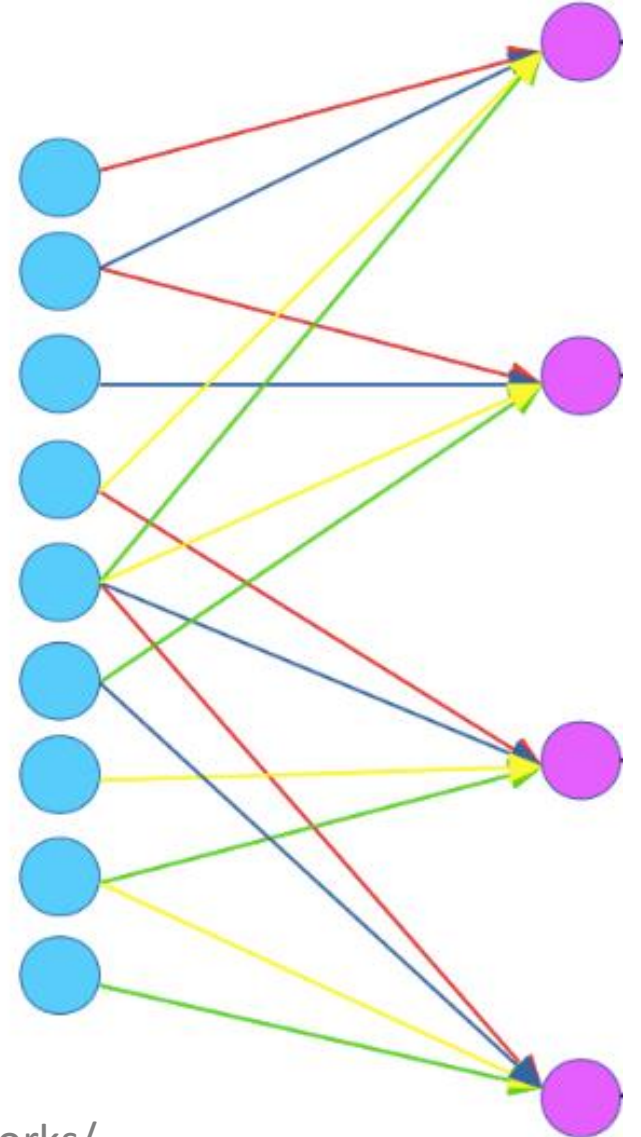
Convolutional Layer: Parameters to Learn

- 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)

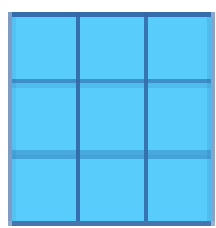


Convolutional Layer: Parameters to Learn

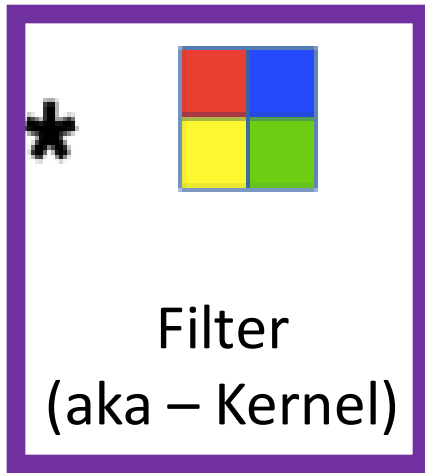
Neocognitron hard-coded filter values...
filter values are learned for CNNs



Convolutional Layer: What Can Filters Do?

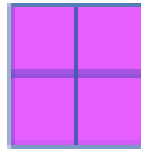


Input



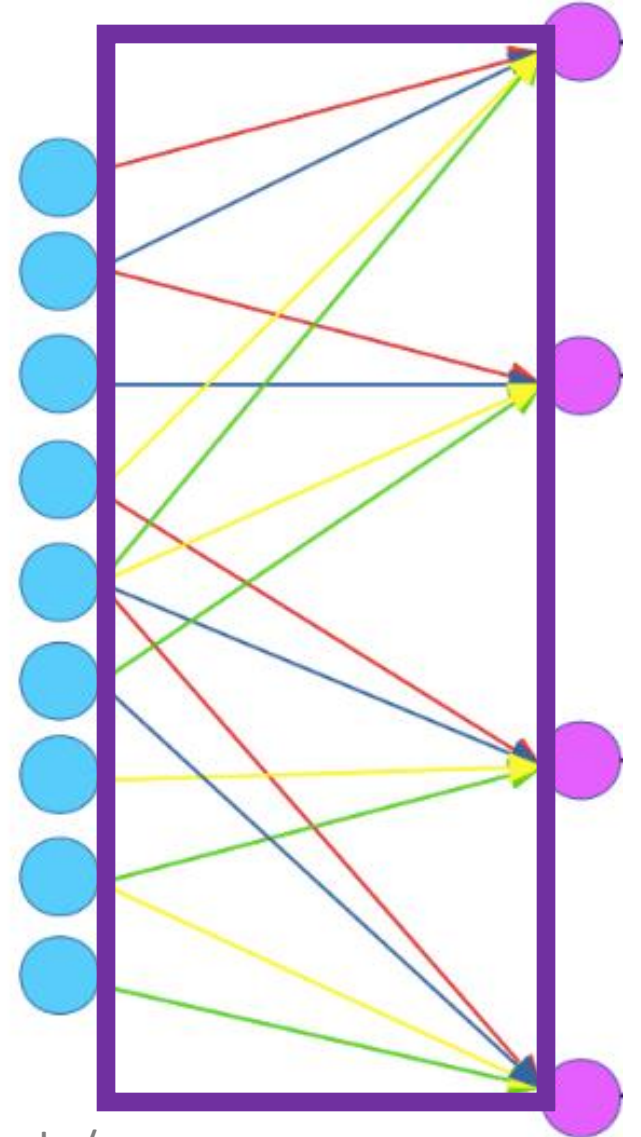
Filter
(aka – Kernel)

=



Feature
Map

Way to Interpret
Neural Network



Convolutional Layer: What Can Filters Do?

Filter



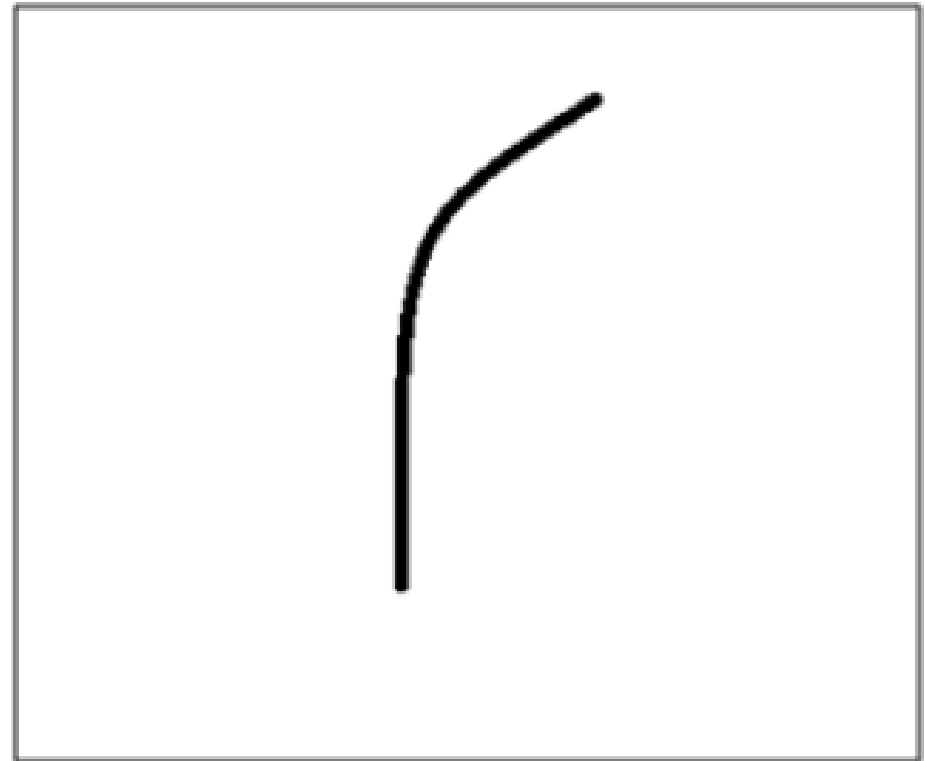
Convolutional Layer: What Can Filters Do?

- 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



Convolutional Layer: What Can Filters 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 = $(50 \times 30) + (20 \times 30) + (50 \times 30) + (50 \times 30) + (50 \times 30)$

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

Convolutional Layer: What Can Filters 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	30	0	0	0
0	0	0	0	0	0	0

Weighted Sum = ?

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

Convolutional Layer: What Can Filters Do?

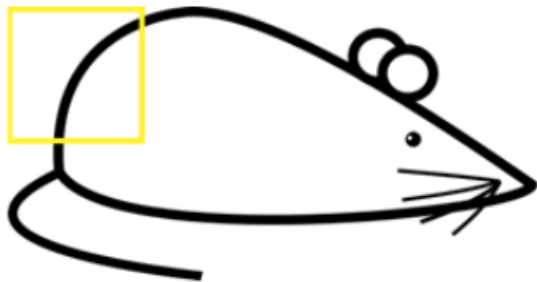
This Filter is a Curve Detector!

- e.g.,

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








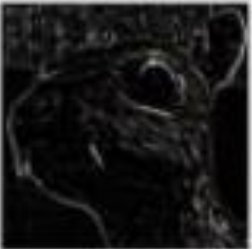

Filter Overlaid on Image (**Big Response!**)




Filter Overlaid on Image (**Small Response!**)



Convolutional Layer: What Can Filters Do?

	Filter	Feature Map		Filter	Feature Map
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$				
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$		Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$		Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$		Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Convolutional Layer: What Can Filters Do?



Filter:
Sharpen

Image:
Bell

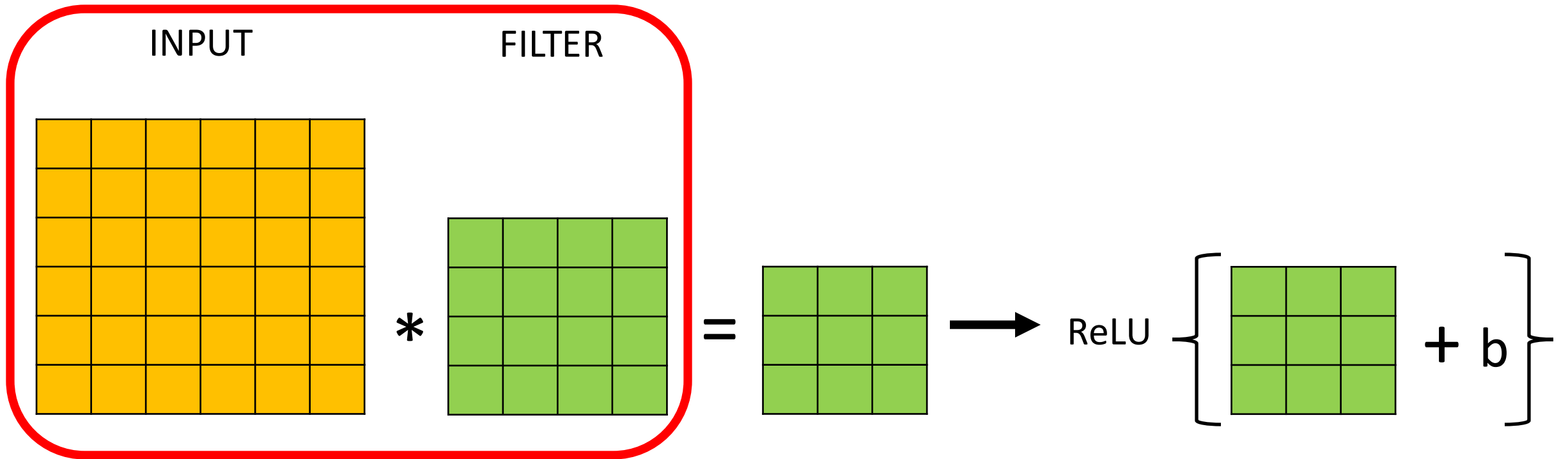
0	-3	0
-3	21	-3
0	-3	0

Divisor: 9

The Matrix

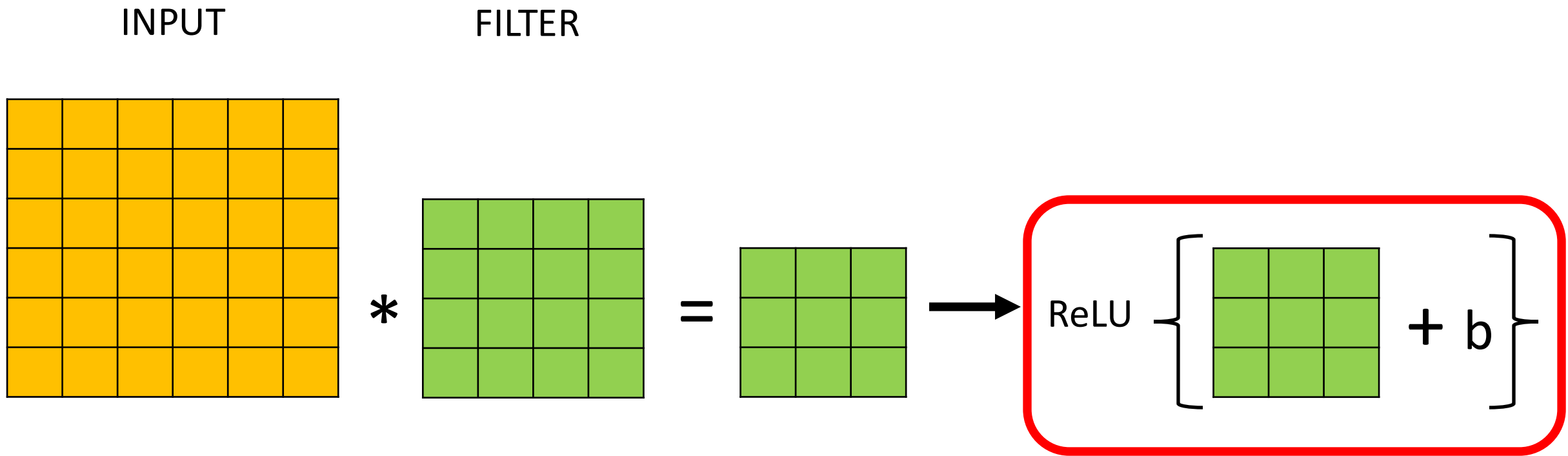
Demo: <http://beej.us/blog/data/convolution-image-processing/>

Key Ingredient 1: Convolutional Layers



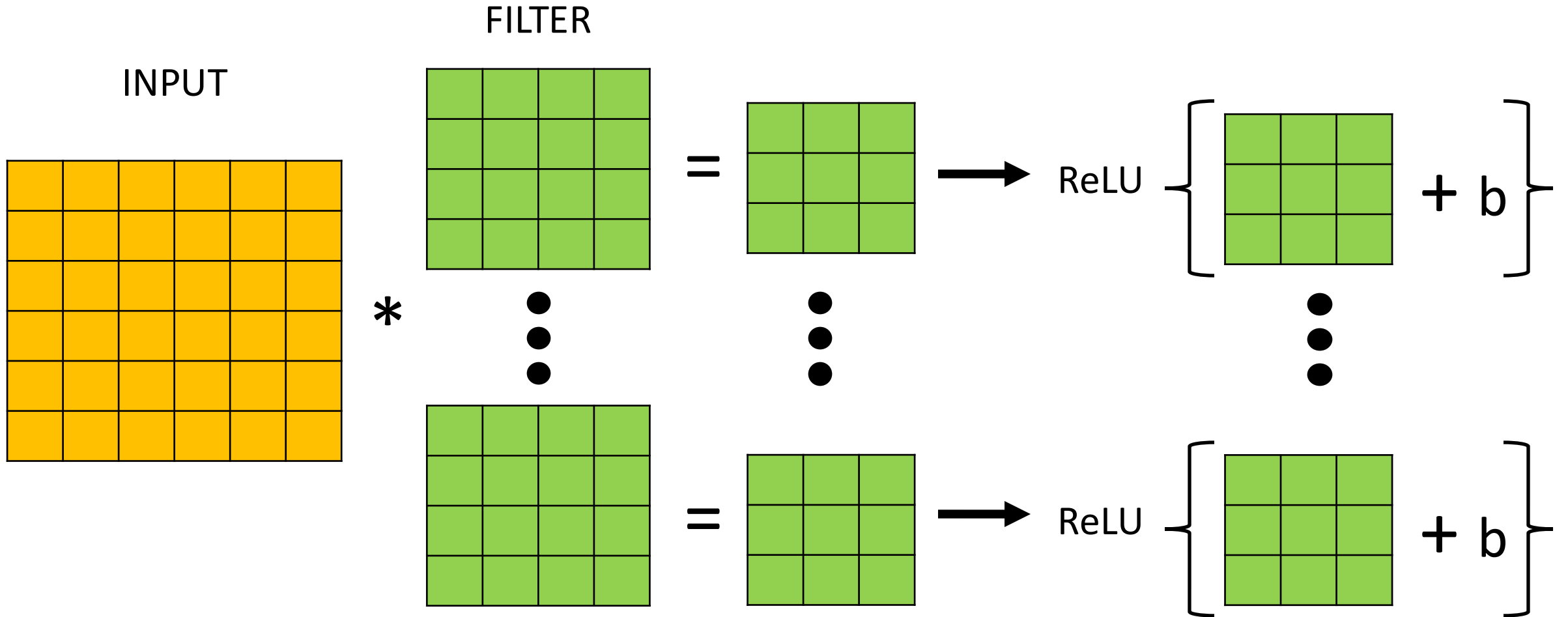
Can choose filters of any size to support feature learning!

Key Ingredient 1: Convolutional Layers



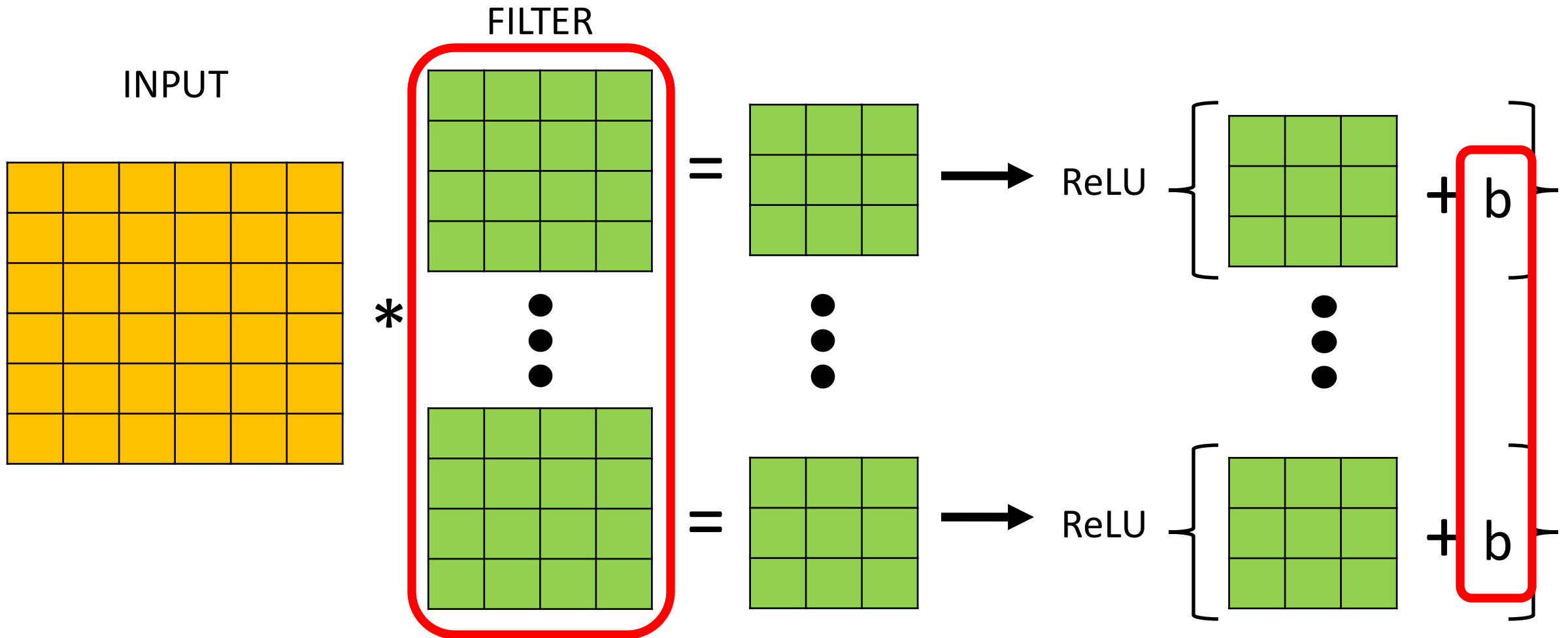
Filtered results are passed, with a bias term, through an activation function to create **activation/feature maps**

Key Ingredient 1: Convolutional Layers



Can have multiple filters (with a unique bias parameter per filter)

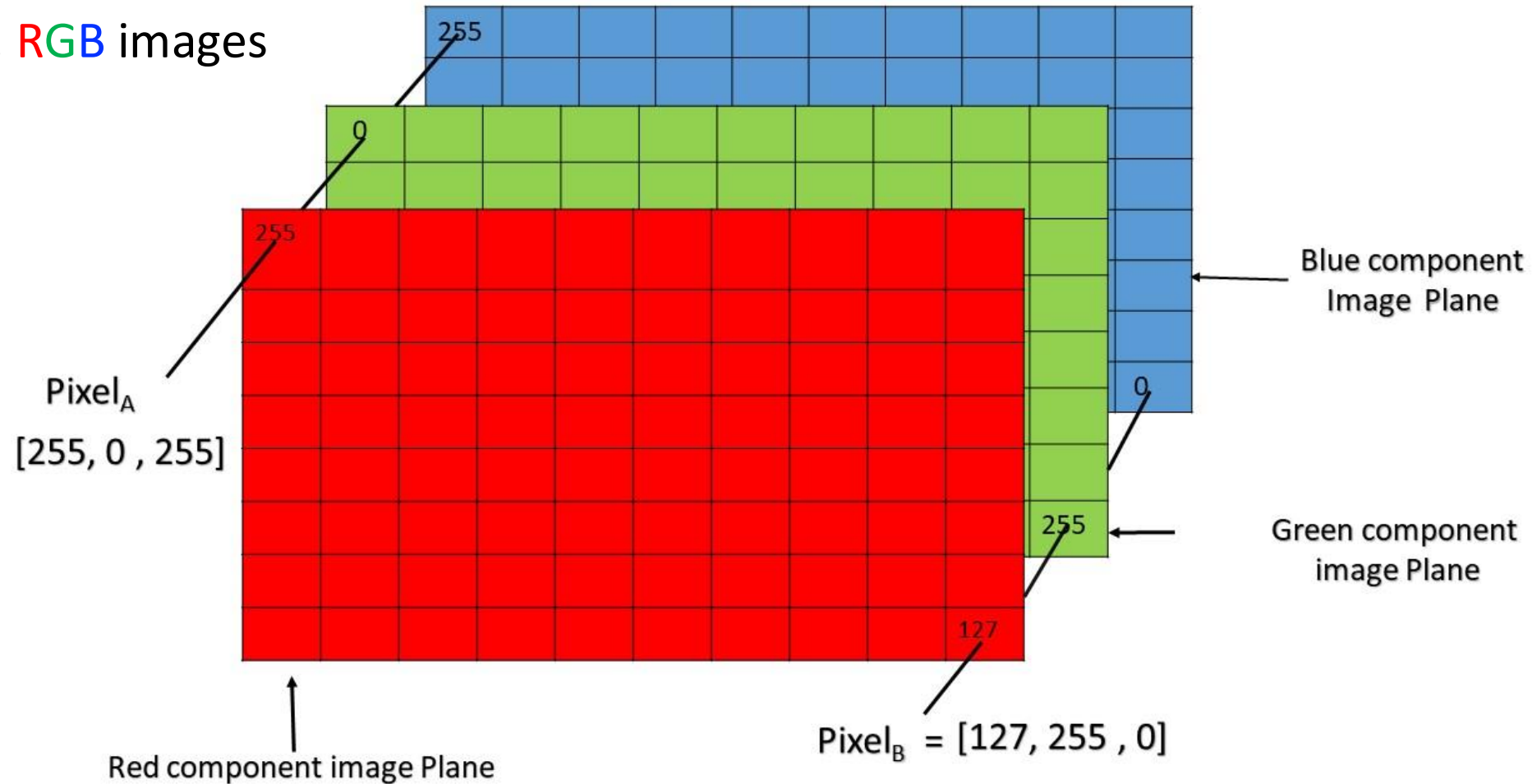
Key Ingredient 1: Convolutional Layer Summary



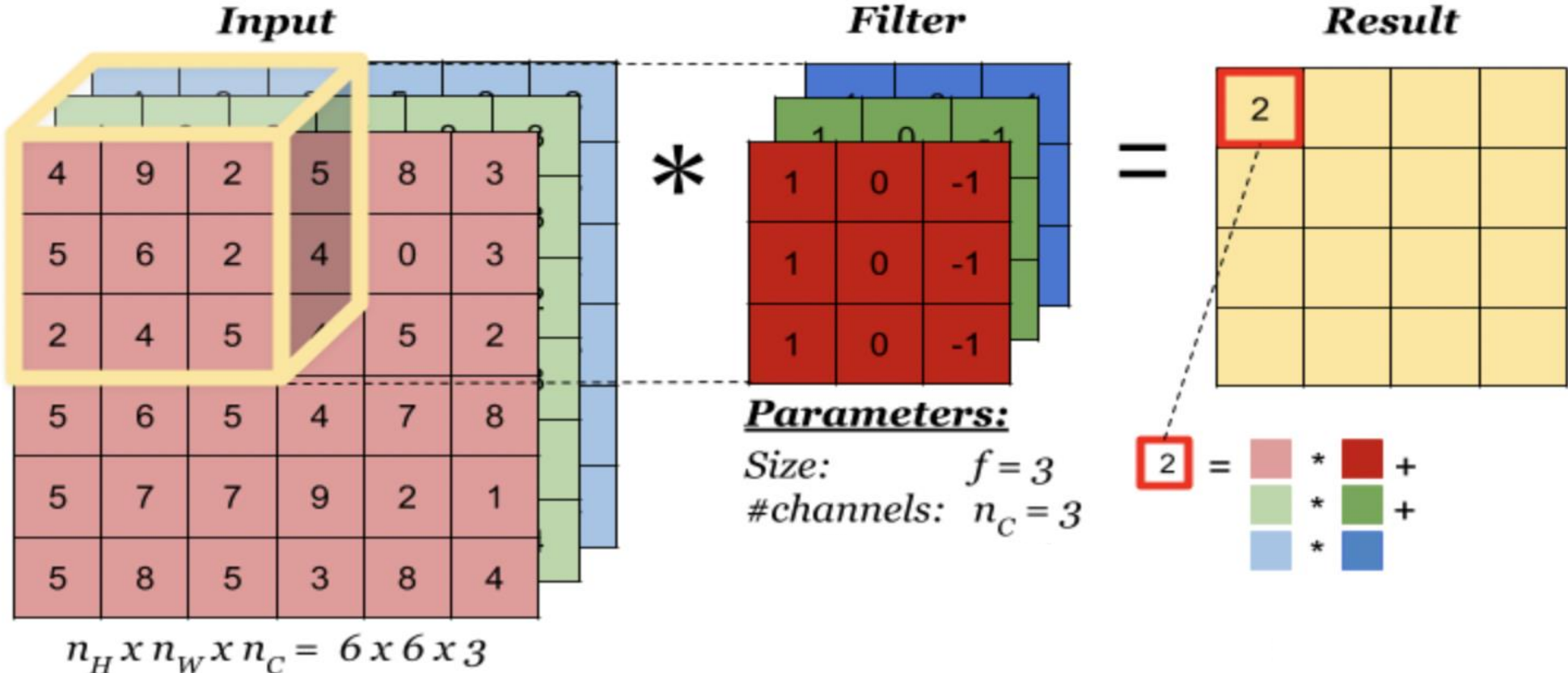
Neural networks learn values for all filters and biases in all layers

How Filters Are Applied to Multi-Channel Inputs

e.g., RGB images



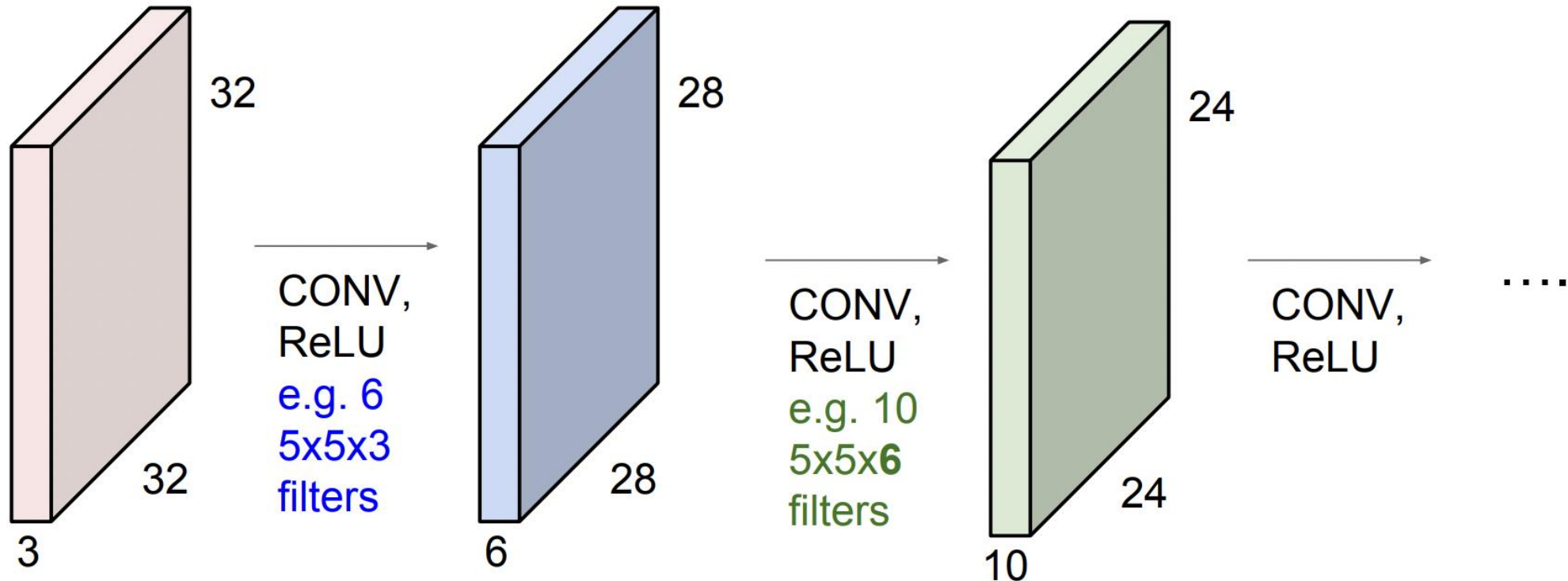
How Filters Are Applied to Multi-Channel Inputs



Number of channels in a filter matches that of the input

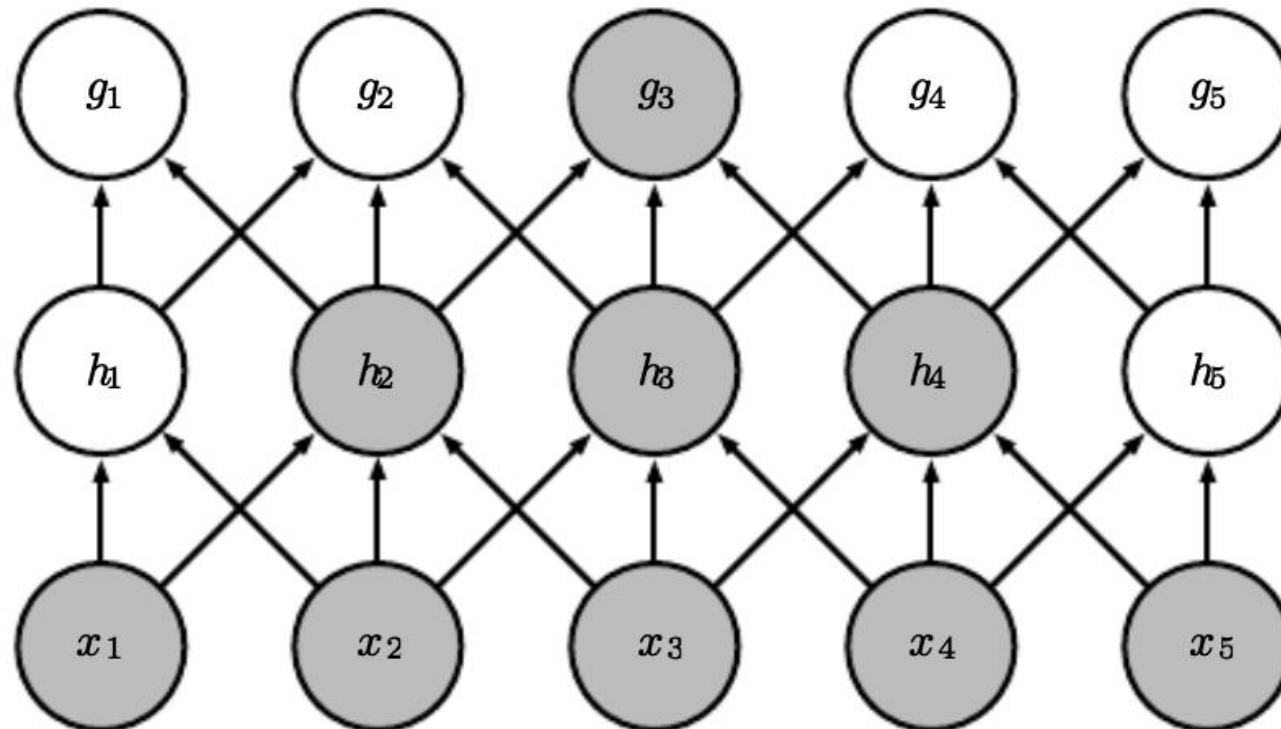
Convolutional Layers Stacked

Can then stack a sequence of convolution layers; e.g.,



Convolutional Layers Stacked

Can then stack a sequence of convolution layers, which leads to identifying patterns in increasingly **larger regions of the input (e.g., pixel) space**:

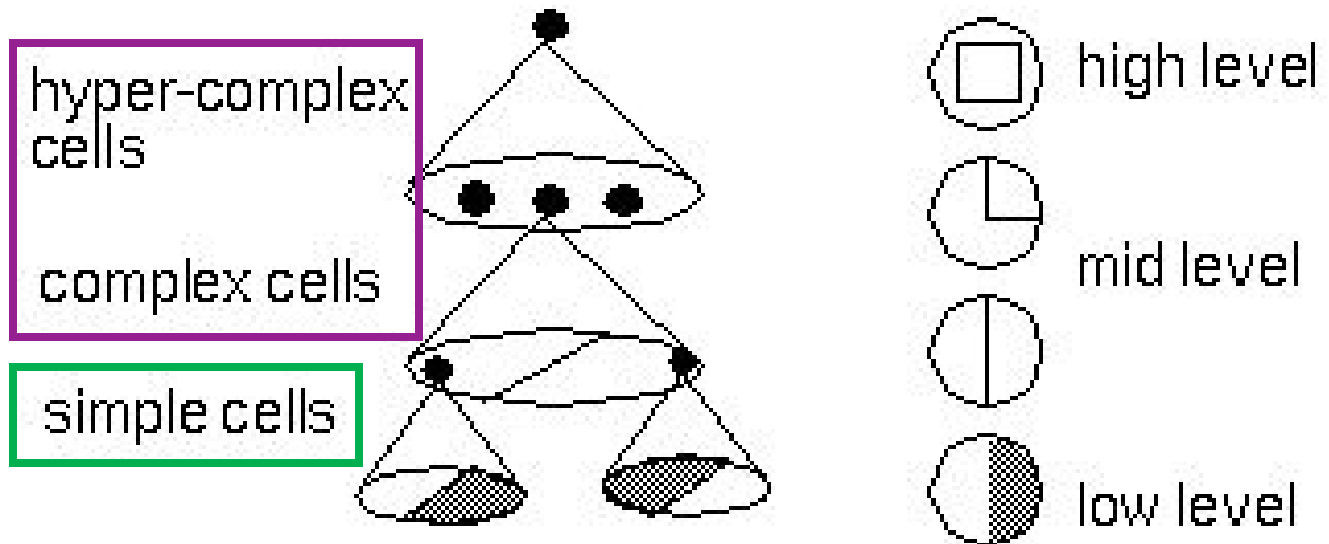


Convolutional Layers Stacked

Can then stack a sequence of convolution layers, which leads to identifying patterns in increasingly **larger regions of the input (e.g., pixel) space** and **mimicking vision system**:

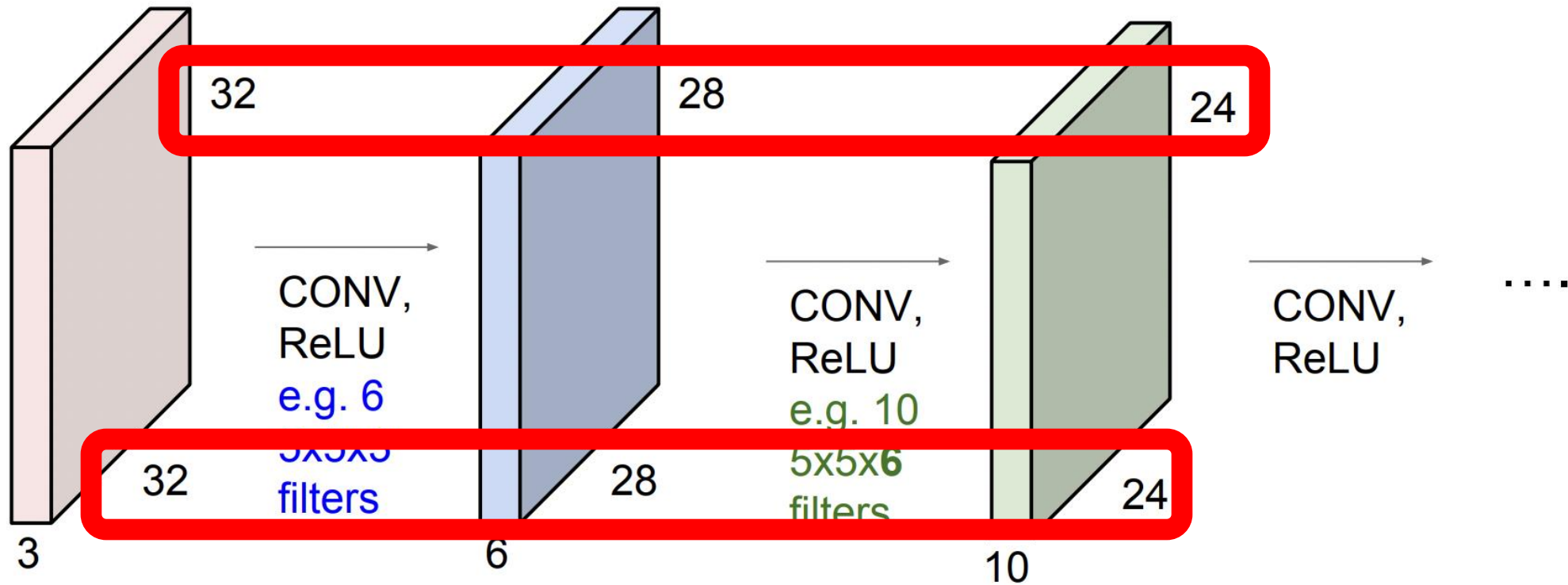
featural hierarchy

Higher level features are constructed by combining lower level features



Problem #1: Input Shrinks

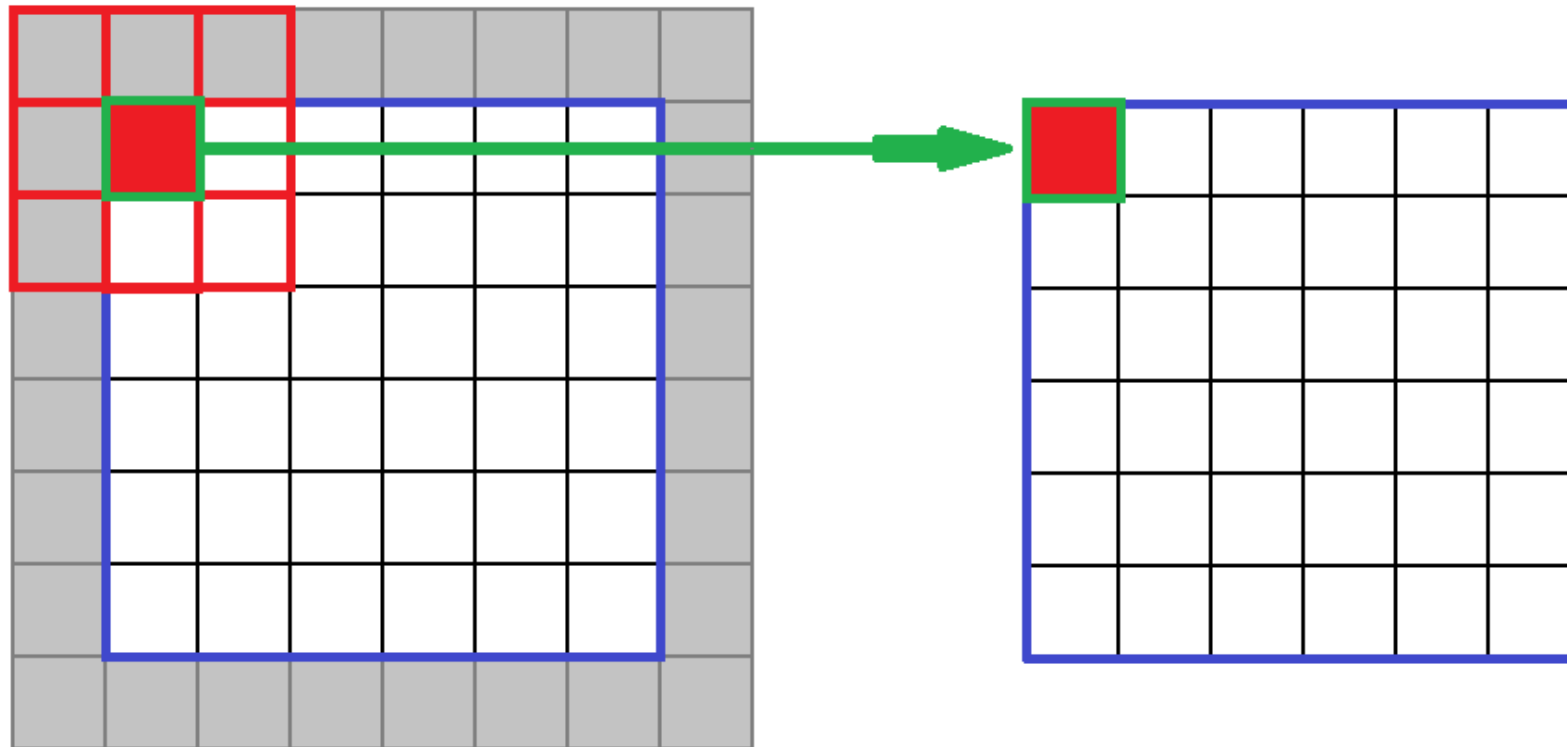
Why do the dimensions shrink with each convolutional layer?



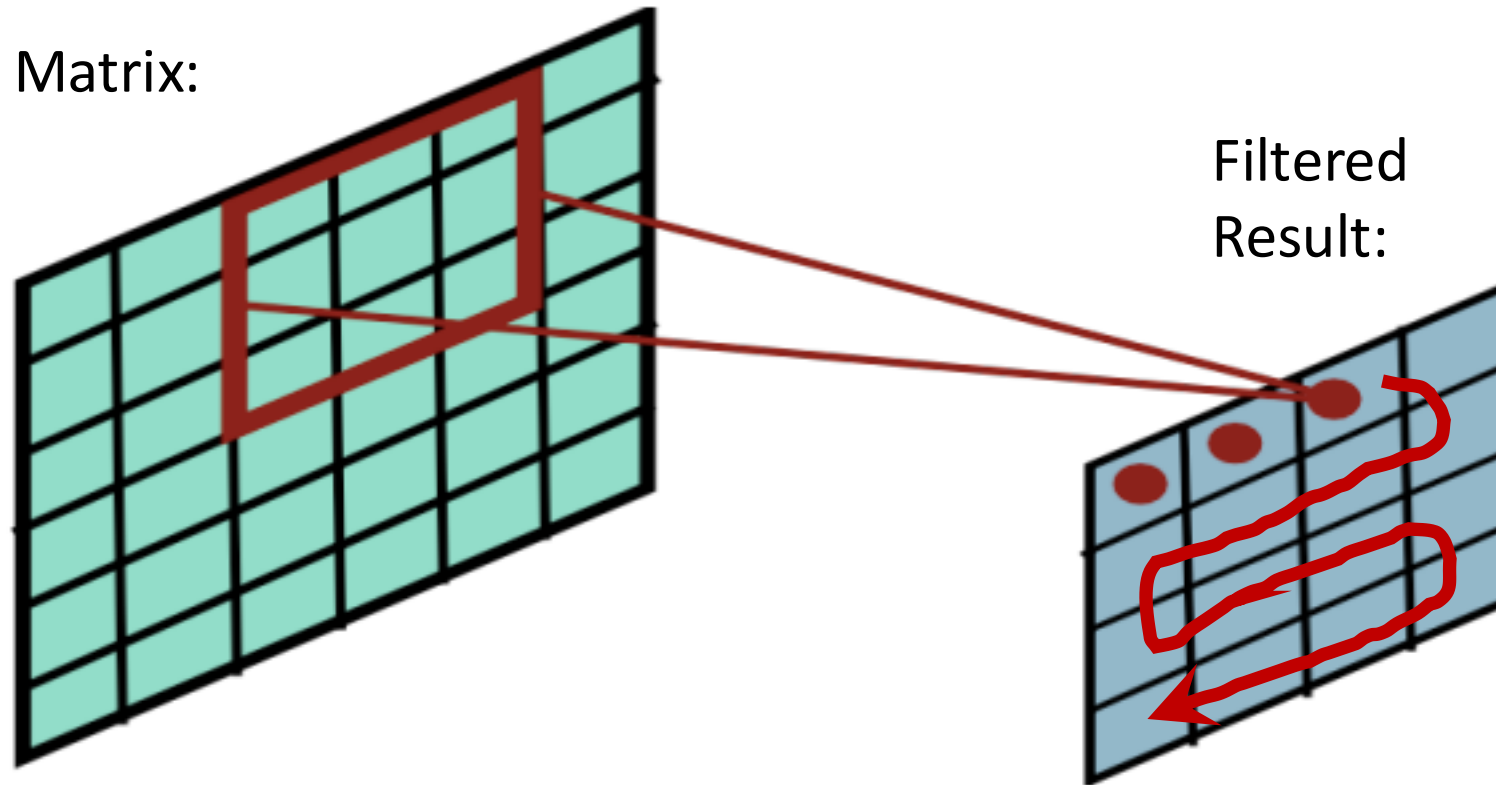
Information is lost around boundary of the input!

Solution: Control Output Size with **Padding**

- **Padding:** add values at the boundaries



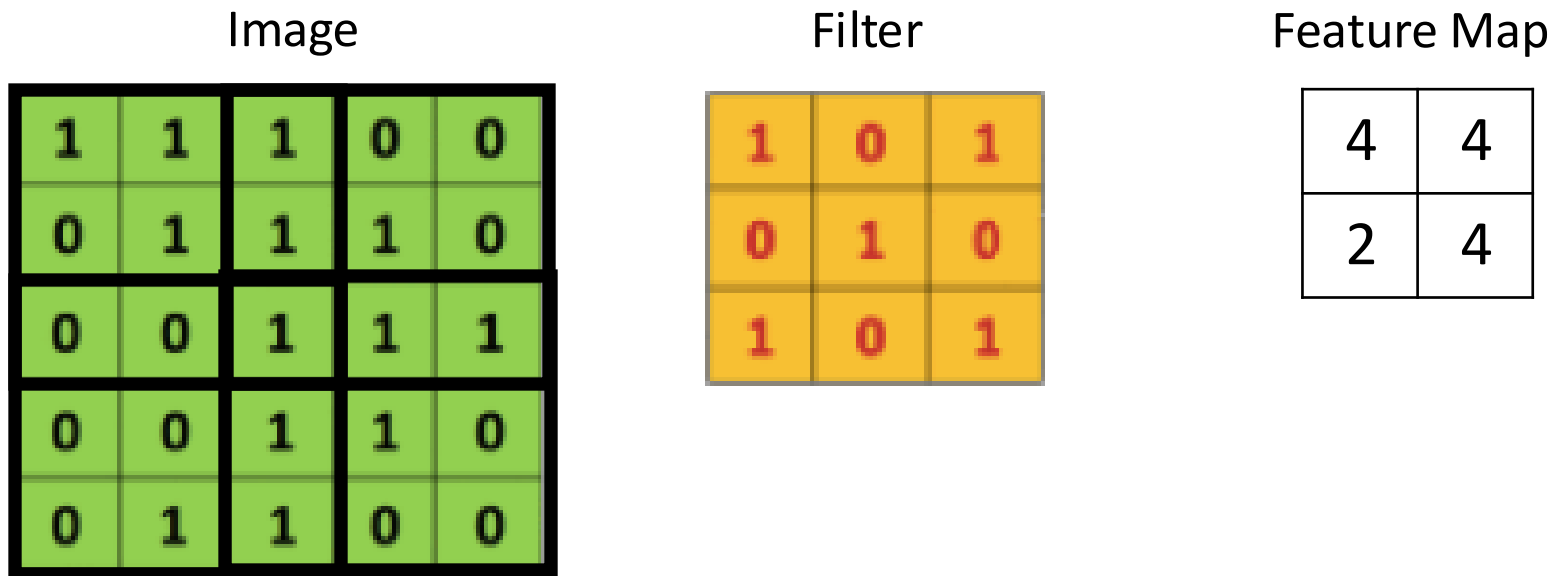
Problem #2: Computation Expensive



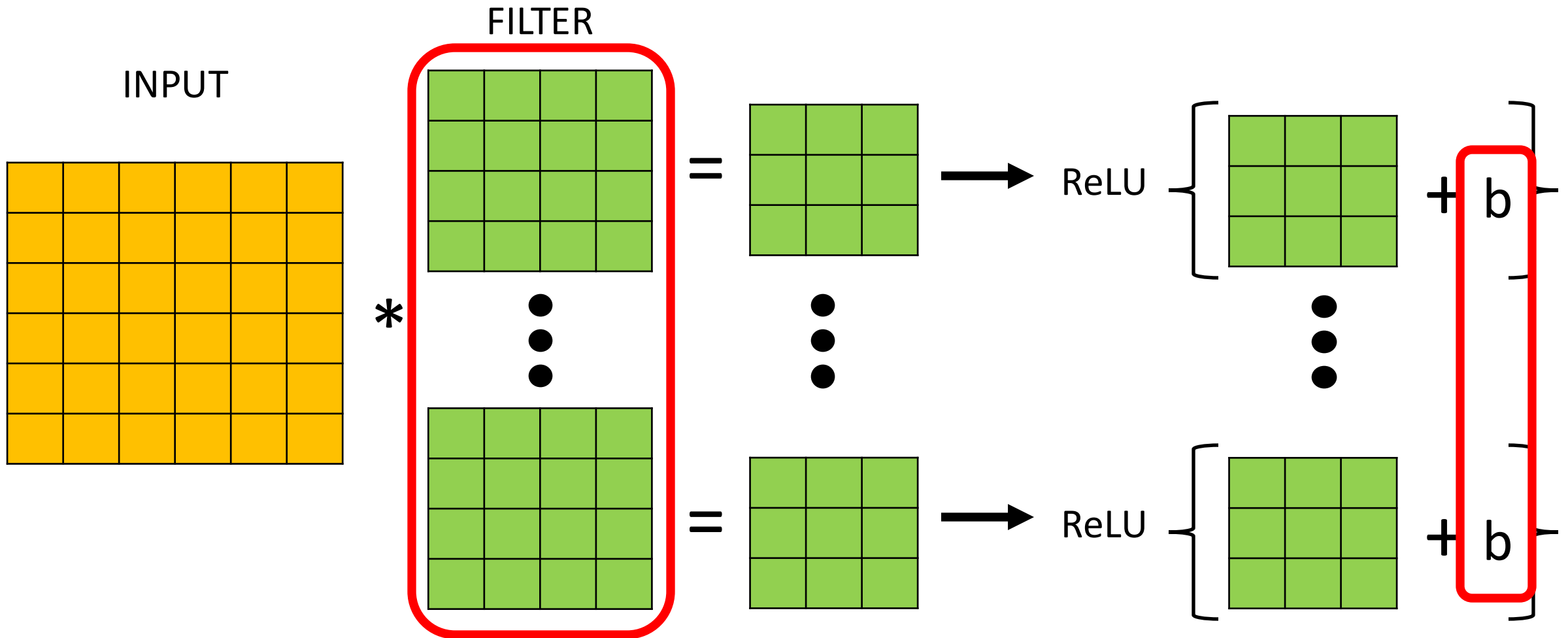
Many computations to slide filter over every point in the matrix and compute dot products

Idea: Reduce Computations with Stride

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



Key Ingredient 1: Convolutional Layer Summary



Neural networks learn values for all filters and biases in all layers

Key Ingredient 2: Pooling Layer

- Summarizes neighborhood; e.g., **max-pooling** partitions input into non-overlapping rectangles and outputs maximum value per chunk

Single depth slice

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

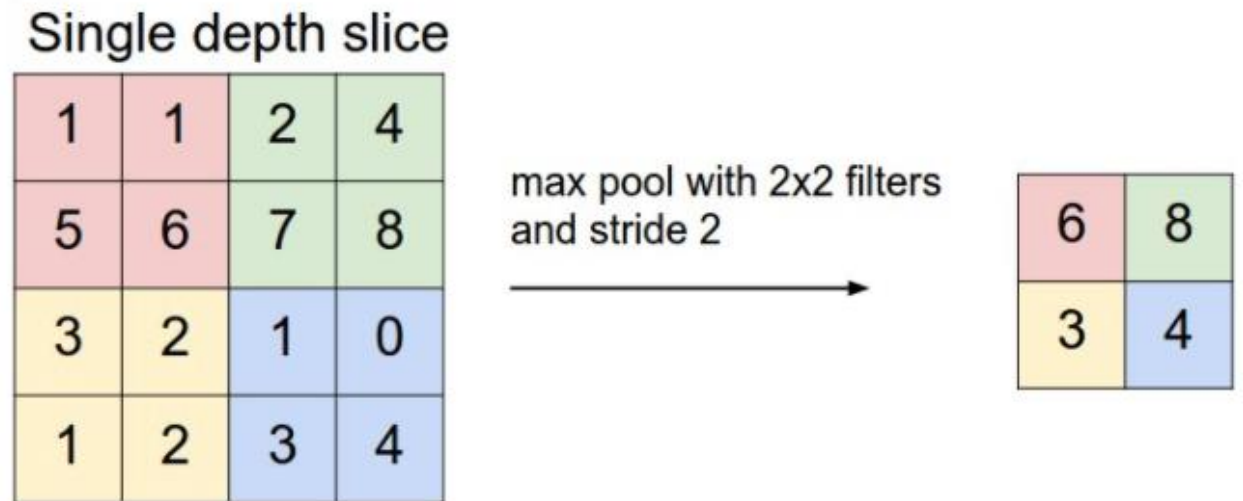
max pool with 2x2 filters
and stride 2



?	?
?	?

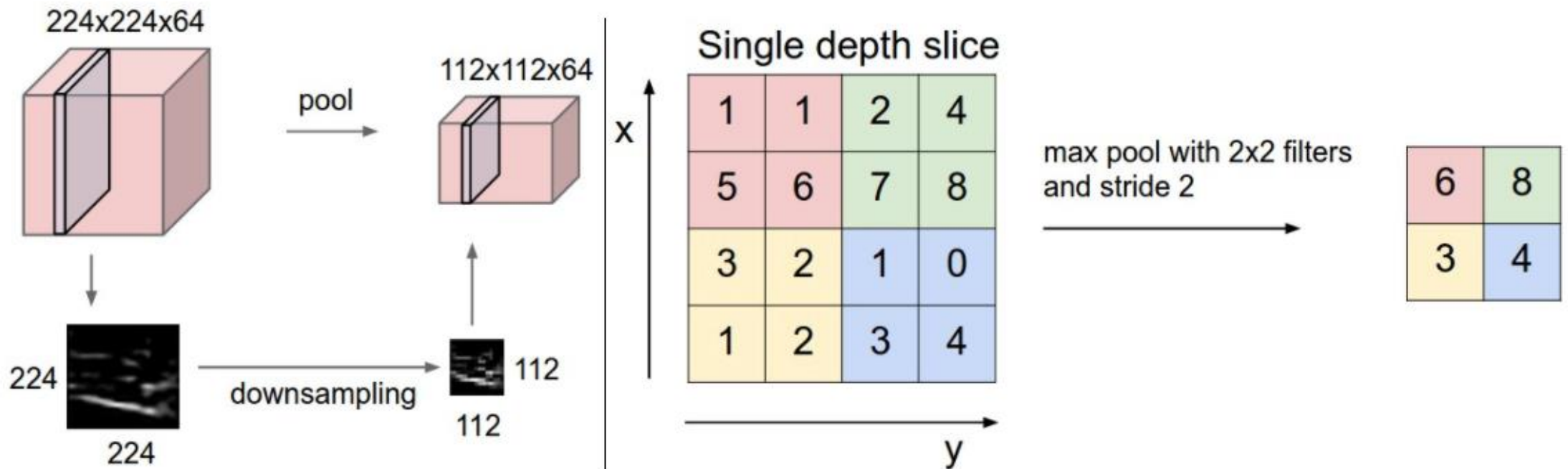
Key Ingredient 2: Pooling Layer

- Summarizes neighborhood; e.g., **max-pooling** partitions input into non-overlapping rectangles and outputs maximum value per chunk



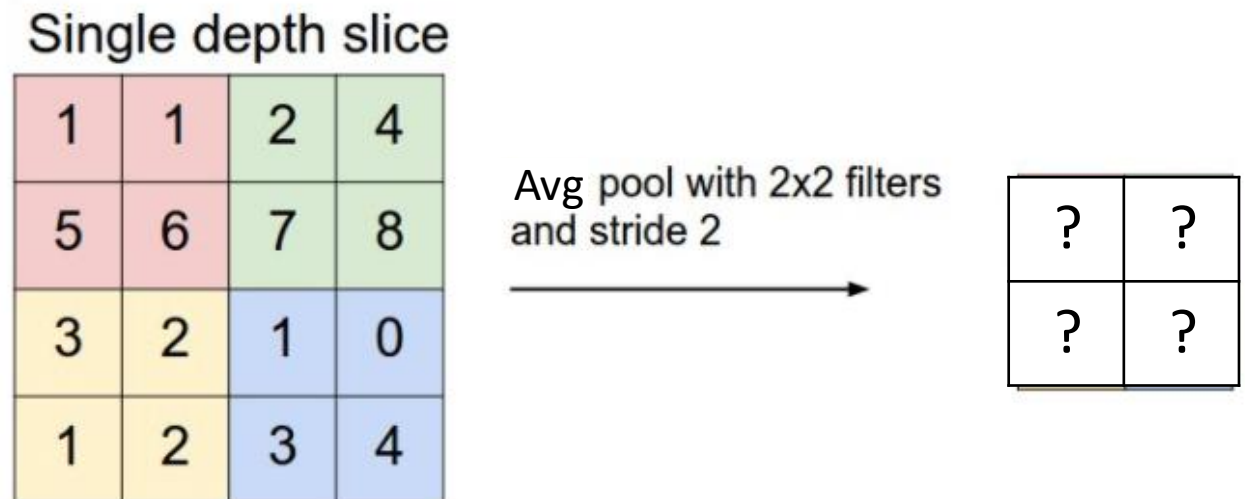
Key Ingredient 2: Pooling Layer

- Summarizes neighborhood; e.g., **max-pooling** partitions input into non-overlapping rectangles and outputs maximum value per chunk



Key Ingredient 2: Pooling Layer

- **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



Key Ingredient 2: Pooling Layer

- **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

Single depth slice

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

Avg pool with 2x2 filters
and stride 2

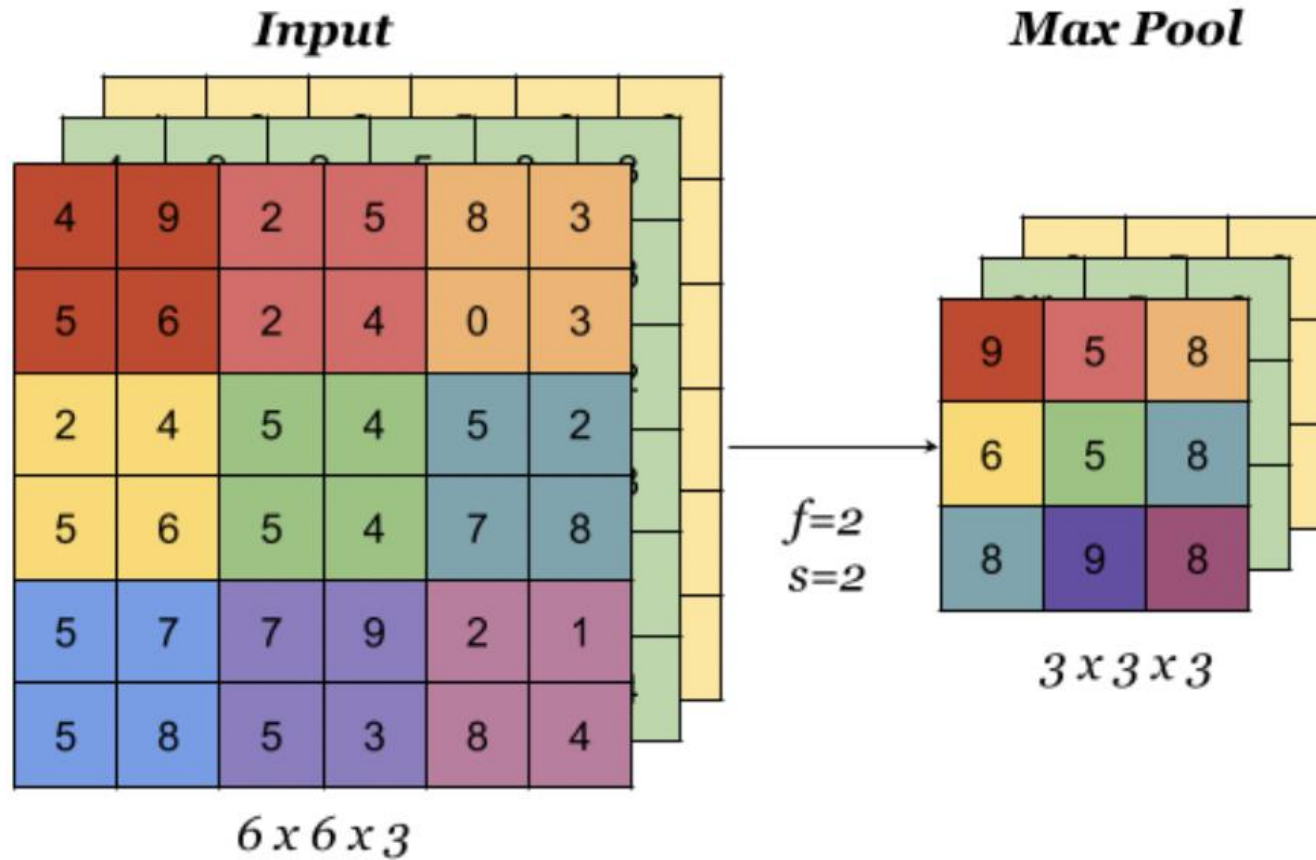


3.25	5.25
2	2

Key Ingredient 2: Pooling Layer

- **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
- And many more pooling options
 - E.g., listed here: <https://pytorch.org/docs/stable/nn.html#pooling-layers>

Pooling for Multi-Channel Input

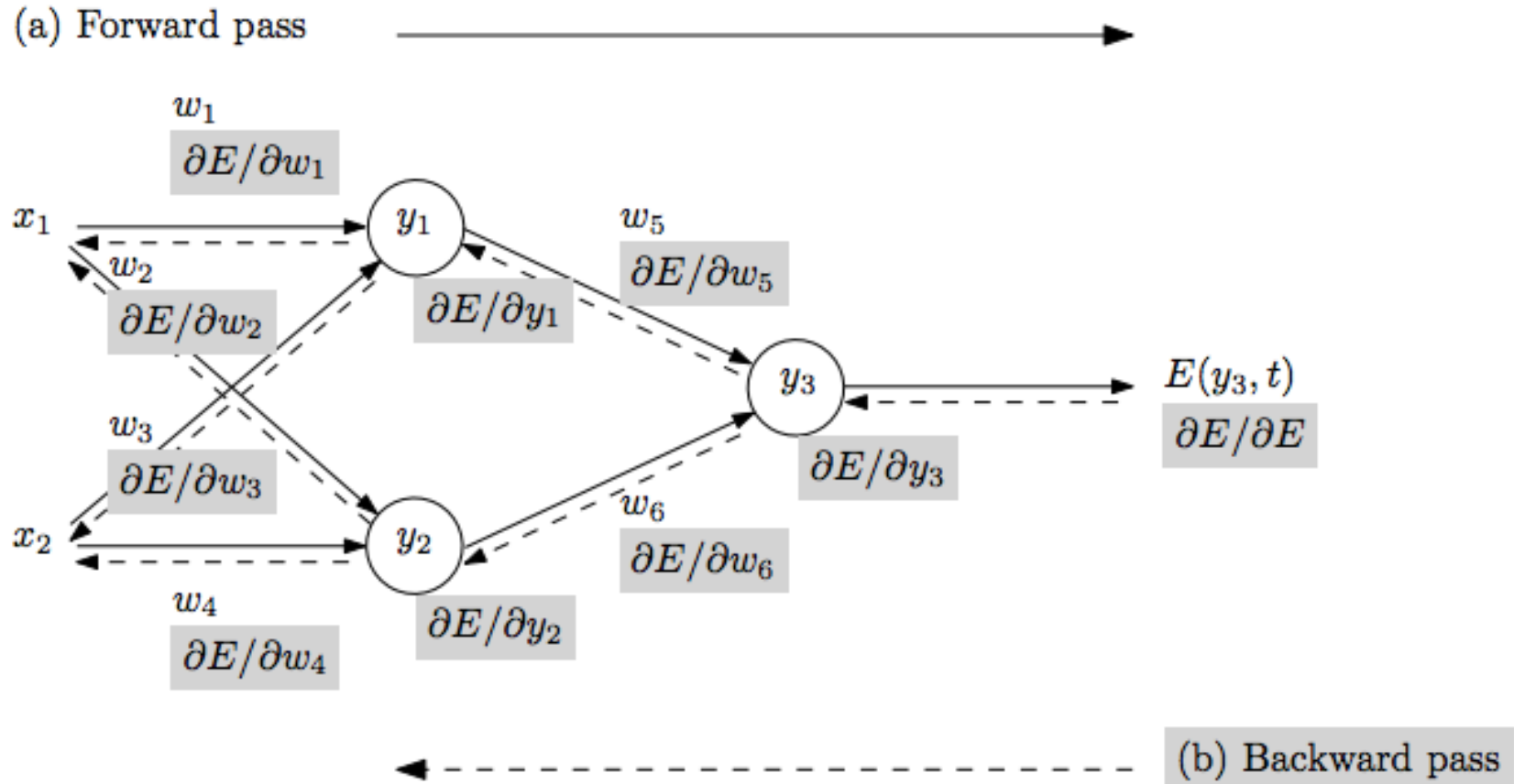


Pooling is applied to each input channel separately

Pooling Layer: Benefits

- Builds in invariance to translations of the input
- Reduces memory requirements
- Reduces computational requirements

Approach to Train CNNs



- Repeat until stopping criterion met:
 1. **Forward pass:** propagate training data through model to make predictions
 2. **Error quantification:** measure dissatisfaction with a model's predictions on training data
 3. **Backward pass:** using predicted output, calculate gradients backward to assign blame to each model parameter; **account for weight sharing by using average of all connections for a parameter**
 4. Update each parameter using calculated gradients

Object Recognition: Today's Topics

- Problem
- Applications
- Datasets
- Evaluation metric
- A Popular Solution: Convolutional Neural Network



The End