



CS 6804: MULTIMODAL VISION

Chris Thomas

Department of Computer Science

Virginia Tech

January 23, 2023

COURSE UPDATES

- Thanks to those who submitted their topic preferences
- Office hours update: *Tuesdays 12:00 – 2:00 PM*
- Schedule has been updated for next couple weeks
- **Please check to see if you are assigned to present as soon as possible**
- **Reminder: First paper review due tomorrow, 10:00 PM on Canvas**
 - Cedric will be presenting VQA

PLAN FOR TODAY

- People from varying backgrounds / familiarity with vision and deep learning are registered for this course
 - Requests to cover some basics in the topic submissions
 - **Thank you for being honest!**
 - Adjusting the schedule / topics to ease into the more complex topics to provide adequate background for everyone
- Today we'll be going over some basics of computer vision and deep learning to bring everyone closer
- Plan for today:
 - Introduce some basics of machine learning
 - Discuss how those concepts translate to deep learning
 - Motivate convolutional neural networks (CNNs)
 - CNN details

GROUP PROJECT

- Advise you to begin forming your group projects as soon as possible
- Would a discussion board or something on Canvas help facilitate this?
 - If so, happy to create one to help match you with potential partners
- You might not know the basics yet, but just start thinking about what ideas you want to tackle:
 - Images and audio? Video and text?
 - E.g. Retrieving images from text? Generating images from text?
 - [How to come up with new research ideas?](#)


VISION BASICS

- Nowadays, computer vision relies on **machine learning** to reason about visual content
 - Most rely on a type of artificial neural network which we will discuss
- Most ML algorithms all follow the same basic formulation:
 - We wish to learn a mapping from input to output: $f: x \rightarrow y$
 - x is the input (image, text, etc.)
 - y is the output {cat, dog}, {1, 6, ...}, etc.
 - f : this is the prediction function
- Basic example: Predicting whether an e-mail is spam or not

WARM-UP MACHINE LEARNING EXAMPLE

- We wish to predict whether an e-mail is spam or not

Not Spam

Sebring, Tracy 
To: Batra, Dhruv
ECE 4424 proposal

January 21, 2015 2:53 PM
[Hide Details](#)

CUSP has approved ECE 4424 with the following changes: Can you please provide a clean copy of the proposal with these items addressed? (see below)
Thanks!!!
Tracy

Spam

nadia bamba
To: undisclosed recipients ;
Reply-To: nadia bamba
From Miss Nadia BamBa,

January 19, 2015 5:57 AM
[Hide Details](#)

From Miss Nadia BamBa,

Greeting, Permit me to inform you of my desire of going into business relationship with you. I am Nadia BamBa the only Daughter of late Mr and Mrs James BamBa, My father was a director of cocoa merchant in Abidjan, the economic capital of Ivory Coast before he was poisoned to death by his business associates on one of their outing to discuss on a business deal. When my mother died on the 21st October 2002, my father took me very special because i am motherless.

Before the death of my father in a private hospital here in Abidjan, He secretly called me on his bedside and told me that he had a sum of \$6, 8000.000(SIX Million EIGHT HUNDRED THOUSAND), Dollars) left in a suspense account in a Bank here in Abidjan, that he used my name as his first Daughter for the next of kin in deposit of the fund.

He also explained to me that it was because of this wealth and some huge amount of money That his business associates supposed to balance him from the deal they had that he was poisoned by his business associates, that I should seek for a God fearing foreign partner in a country of my choice where I will transfer this money and use it for investment purposes, (such as real estate Or Hotel management).please i am honourably seeking your assistance in the following ways.

- 1) To provide a Bank account where this money would be transferred to.
- 2) To serve as the guardian of this Money since I am a girl of 19 years old.
- 3)Your private phone number's and your family background' s that we can know each other more.

SIMPLE STRATEGY: COUNTING WORD OCCURRENCES

nadia bamba

To: undisclosed recipients ;

Reply-To: nadia bamba

From Miss Nadia BamBa,

From Miss Nadia BamBa,

Greeting, Permit me to inform you of my desire of going i
Nadia BamBa the only Daughter of late Mr and Mrs Jame
cocoa merchant in Abidjan, the economic capital of Ivory
his business associates on one of their outing to discus c
on the 21st October 2002, my father took me very speci

Before the death of my father in a private hospital here in
bedside and told me that he had a sum of \$6, 8000.000(S
Dollars) left in a suspense account in a Bank here in Abic
Daughter for the next of kin in deposit of the fund.

Sebring, Tracy ✉

To: Batra, Dhruv

ECE 4424 proposal

CUSP has approved ECE 4424 with the following changes: Can
copy of the proposal with these items addressed? (see below)

Thanks!!!

Tracy

This is X

$$\begin{pmatrix} \text{free} & 100 \\ \text{money} & 2 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

Counts of all words
in vocabulary

$$\begin{pmatrix} \text{free} & 1 \\ \text{money} & 1 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

Counts of all words
in vocabulary

This is Y



= 1 or 0?

WEIGH COUNTS AND SUM TO GET PREDICTION

nadia bamba

To: undisclosed recipients; ;

Reply-To: nadia bamba

From Miss Nadia BamBa,

From Miss Nadia BamBa,

Greeting, Permit me to inform you of Nadia BamBa the only Daughter of k cocoa merchant in Abidjan, the econ his business associates on one of th on the 21st October 2002, my father

Before the death of my father in a p bedside and told me that he had a s Dollars) left in a suspense account in Daughter for the next of kin in depos

$$\begin{pmatrix} \text{free} & 100 \\ \text{money} & 2 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

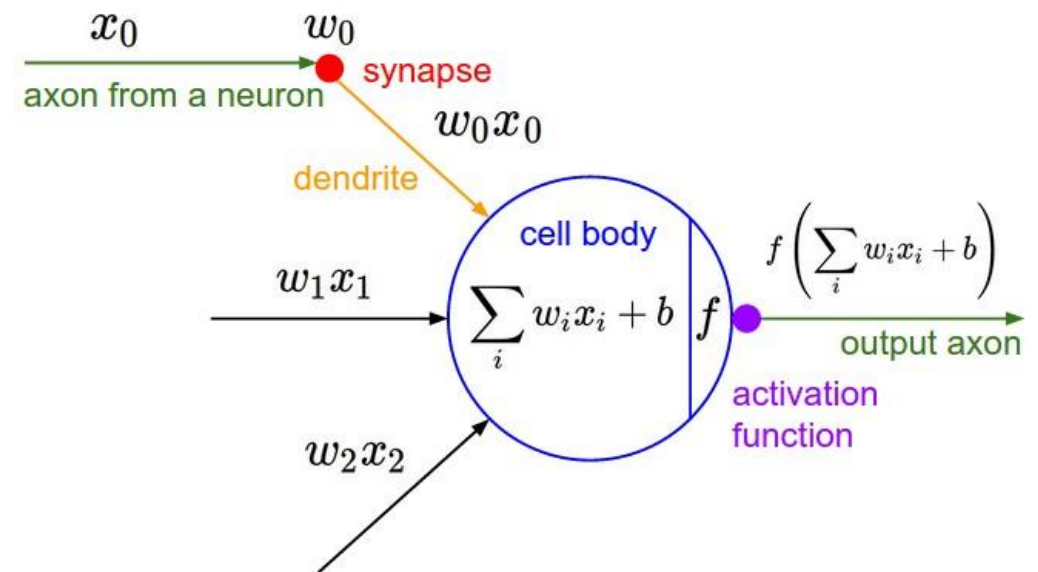
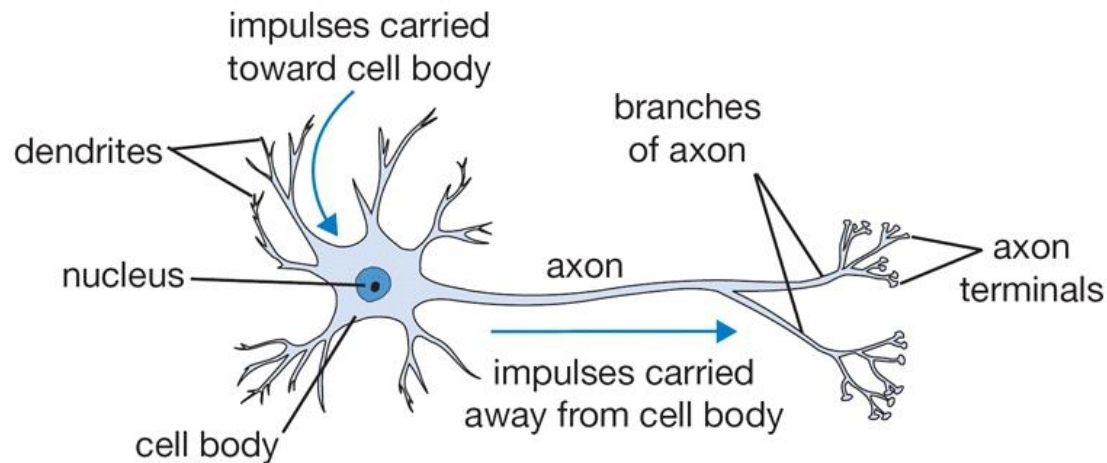
$$\begin{pmatrix} 100 \times 0.2 \\ 2 \times 0.3 \\ \vdots \\ 2 \times 0.3 \\ \vdots \end{pmatrix}$$



This is a *linear classifier*

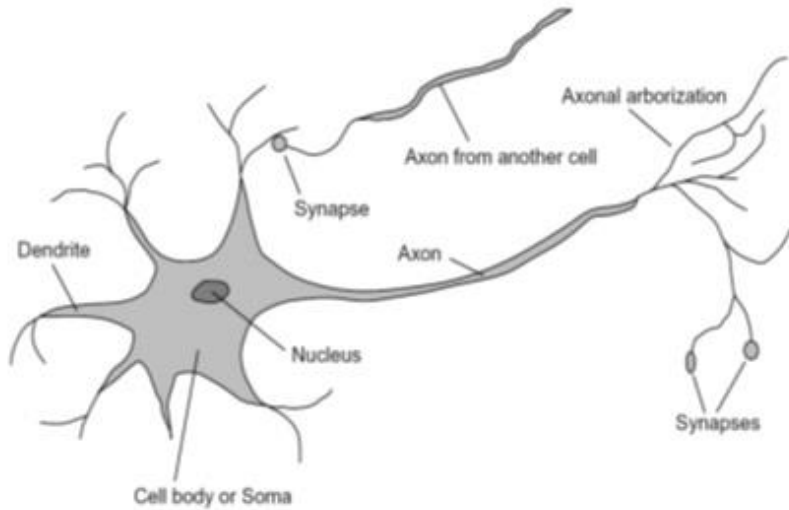
NEURAL NET INSPIRATION: NEURON CELLS

- Neurons
 - accept information from multiple inputs (dendrites)
 - transmit information to other neurons (axons)
- Apply some function to the set of inputs at each node
- If output of function over threshold, neuron “fires”



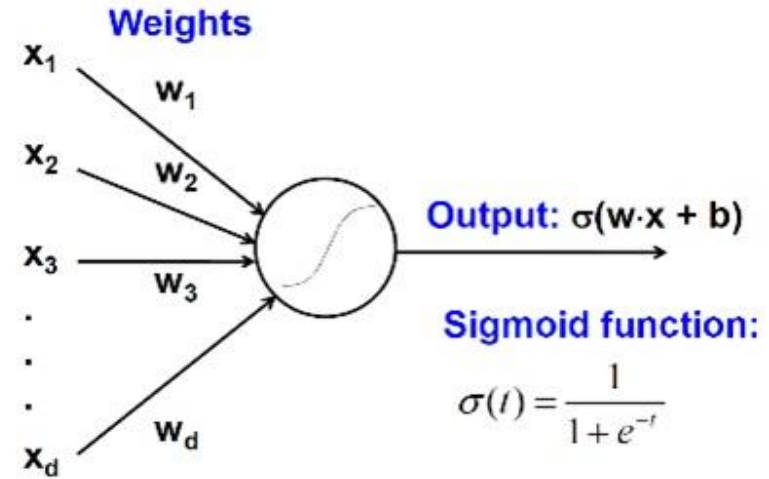
Text: HKUST, figures: Andrej Karpathy

BIOLOGICAL ANALOG



A biological neuron

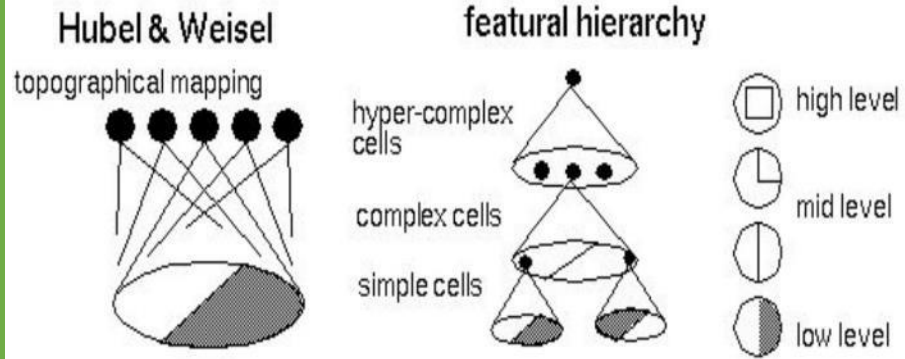
Input



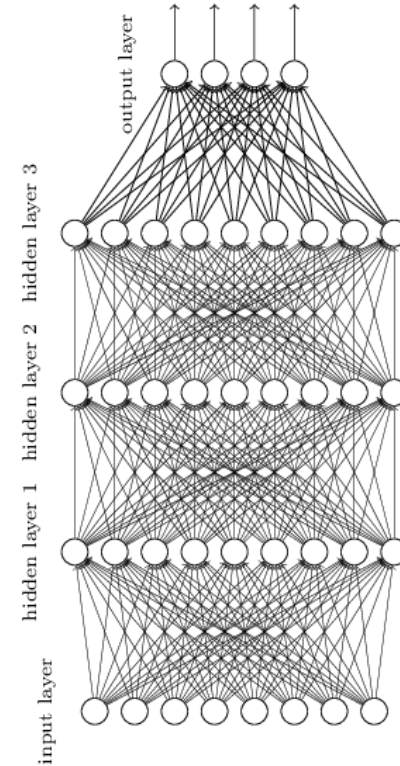
An artificial neuron



BIOLOGICAL ANALOG



Hubel and Weisel's architecture

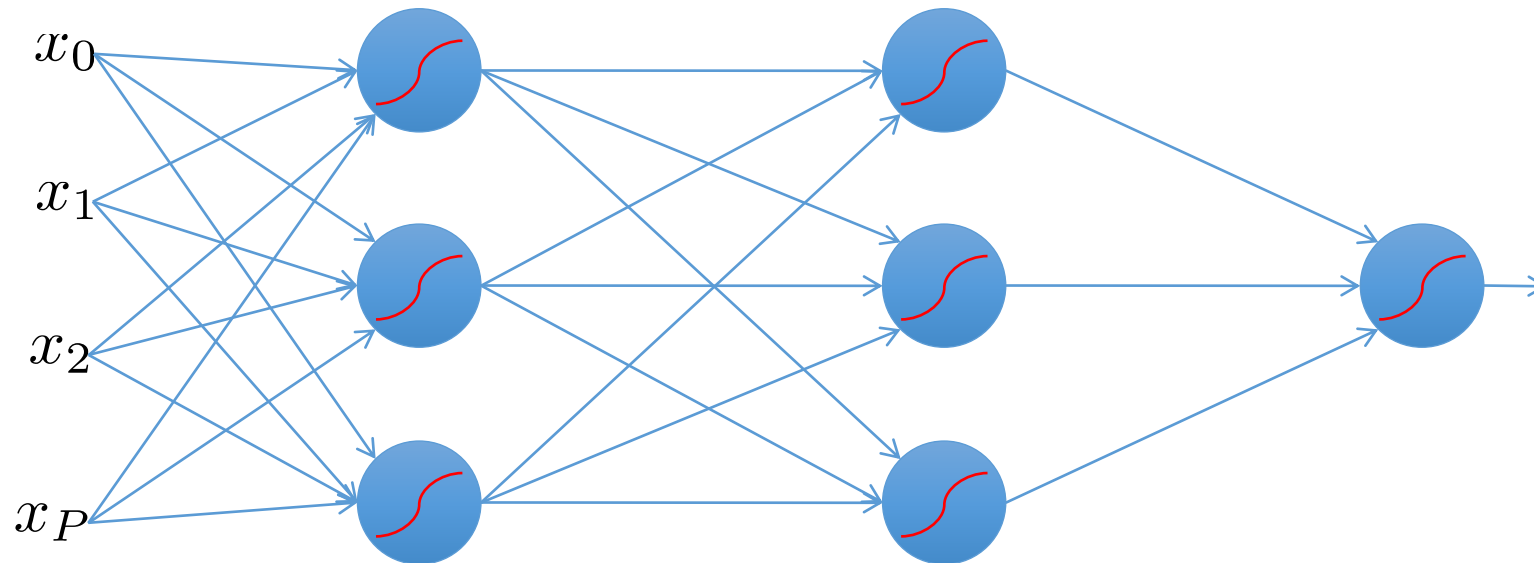


Multi-layer neural network



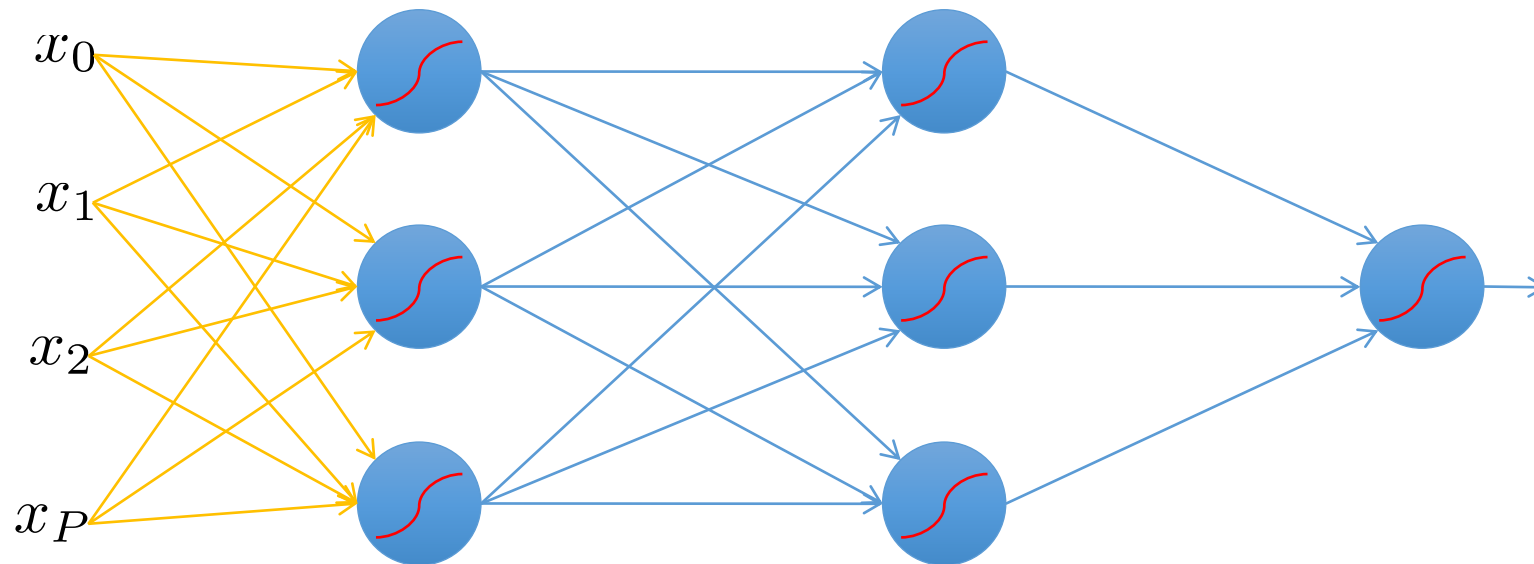
FEED-FORWARD NETWORKS

- Cascade neurons together
- Output from one layer is the input to the next
- Each layer has its own sets of weights



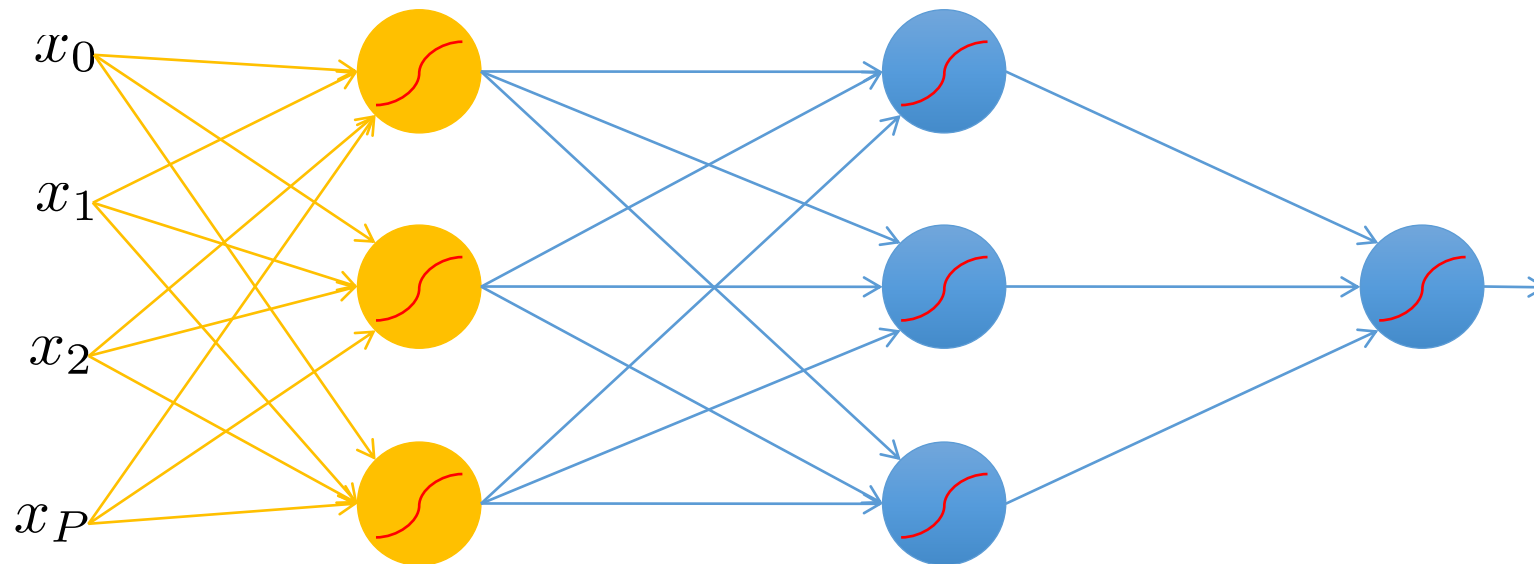
FEED-FORWARD NETWORKS

- Inputs multiplied by initial set of weights



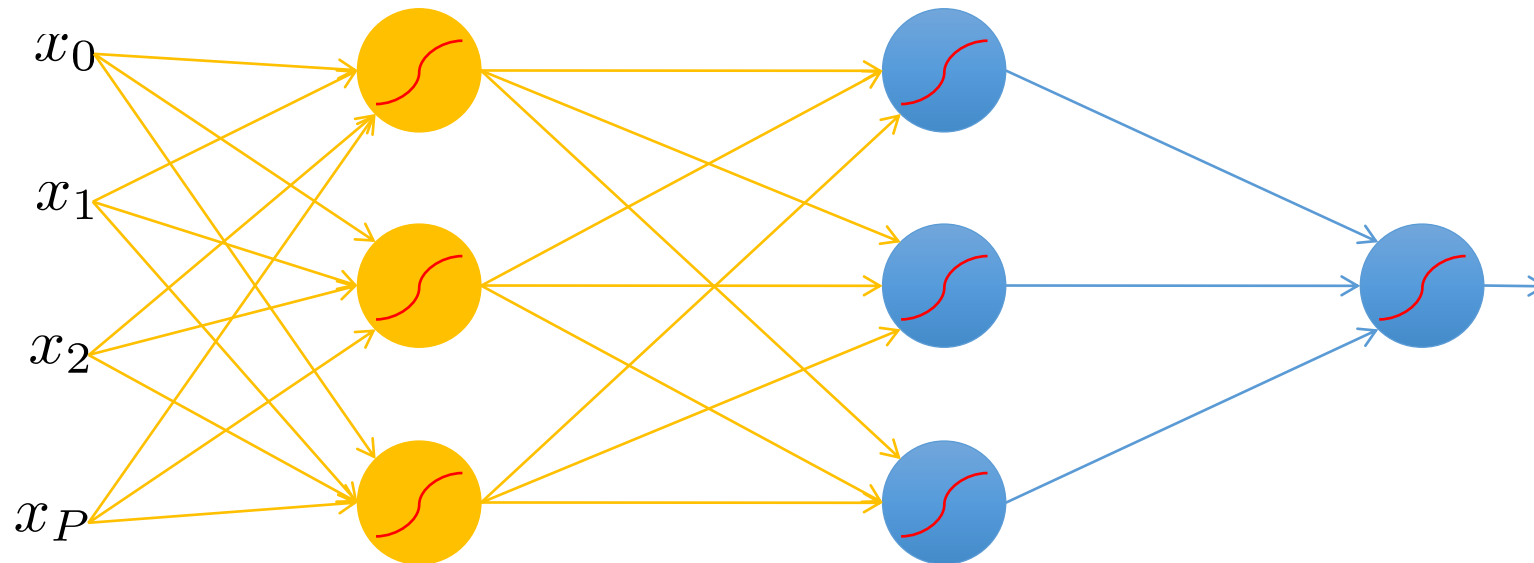
FEED-FORWARD NETWORKS

- Intermediate “predictions” computed at first hidden layer



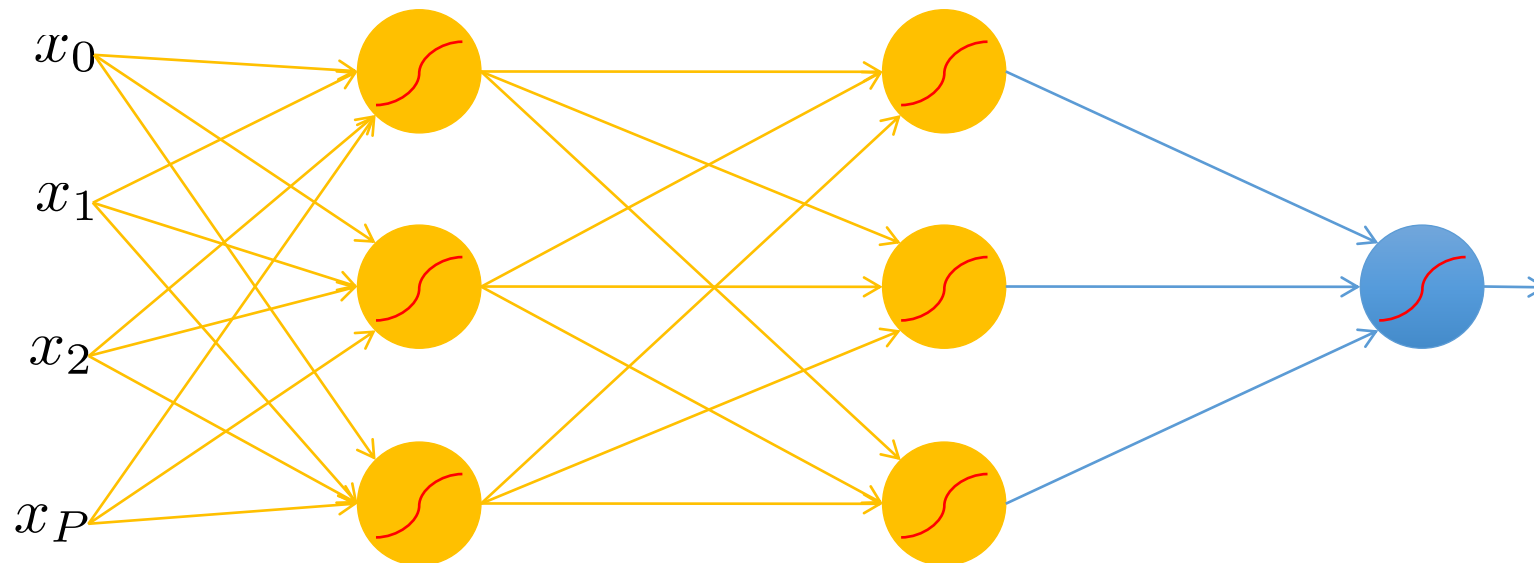
FEED-FORWARD NETWORKS

- Intermediate predictions multiplied by second layer of weights
- Predictions are fed forward through the network to classify



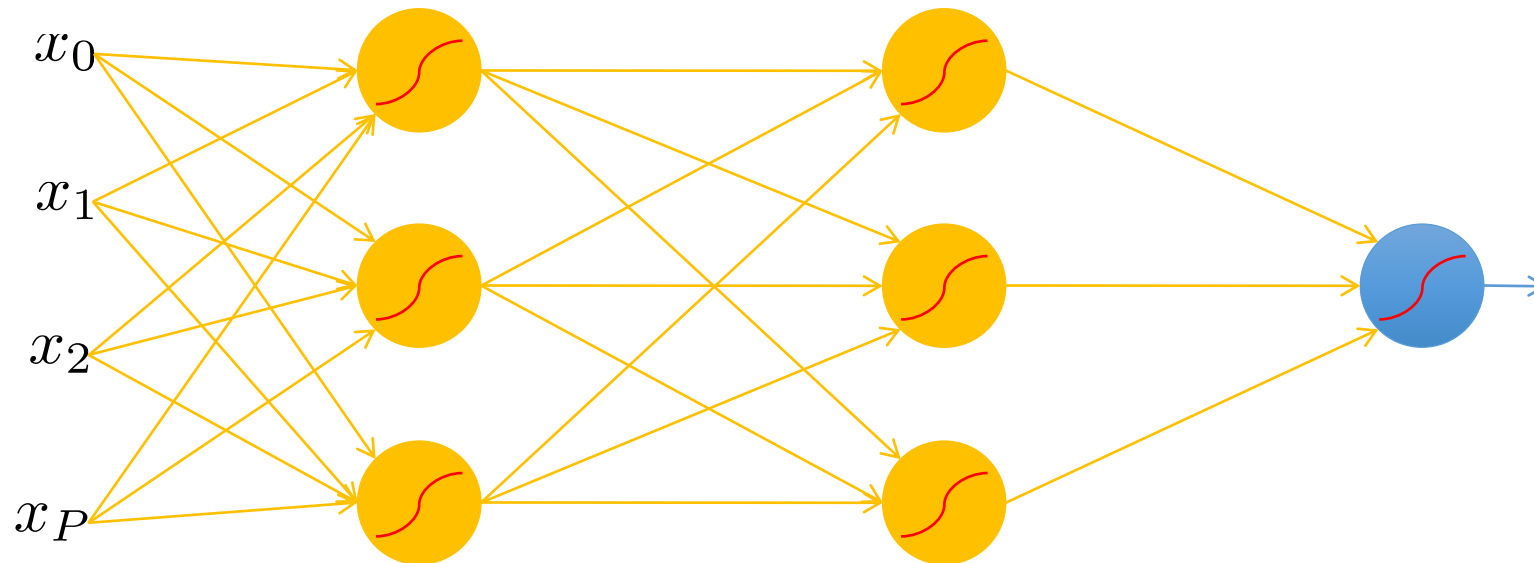
FEED-FORWARD NETWORKS

- Compute second set of intermediate predictions



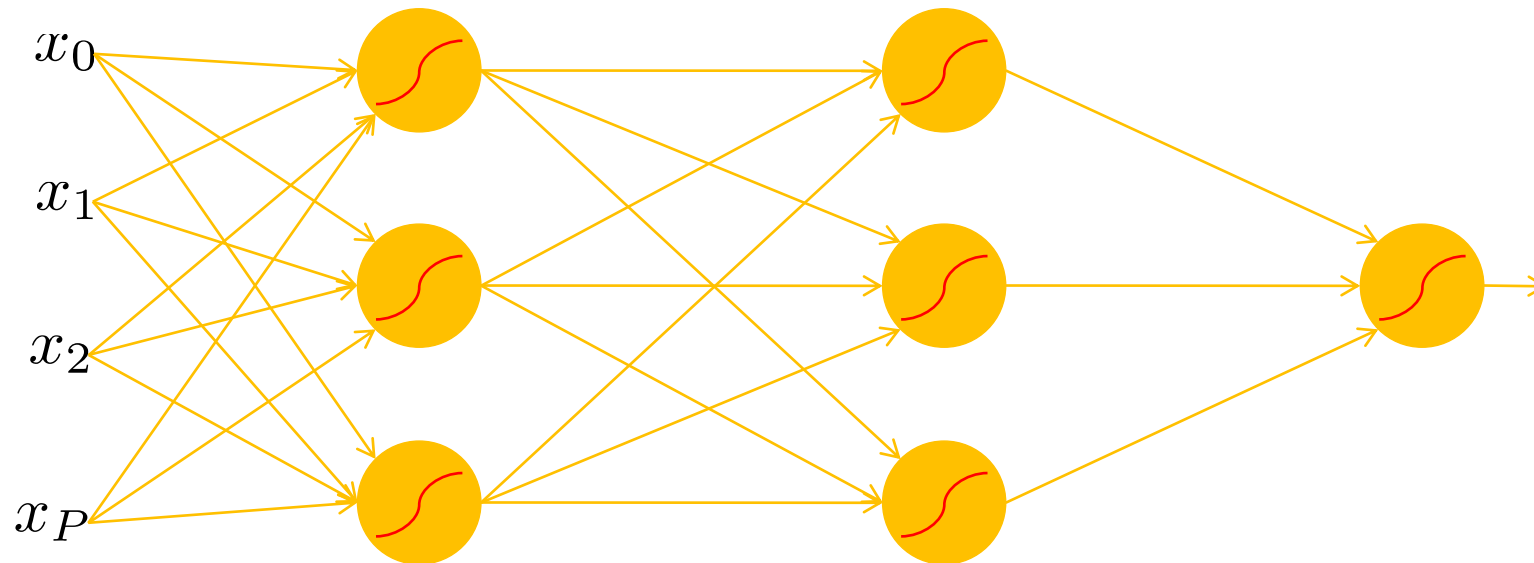
FEED-FORWARD NETWORKS

- Multiply by final set of weights



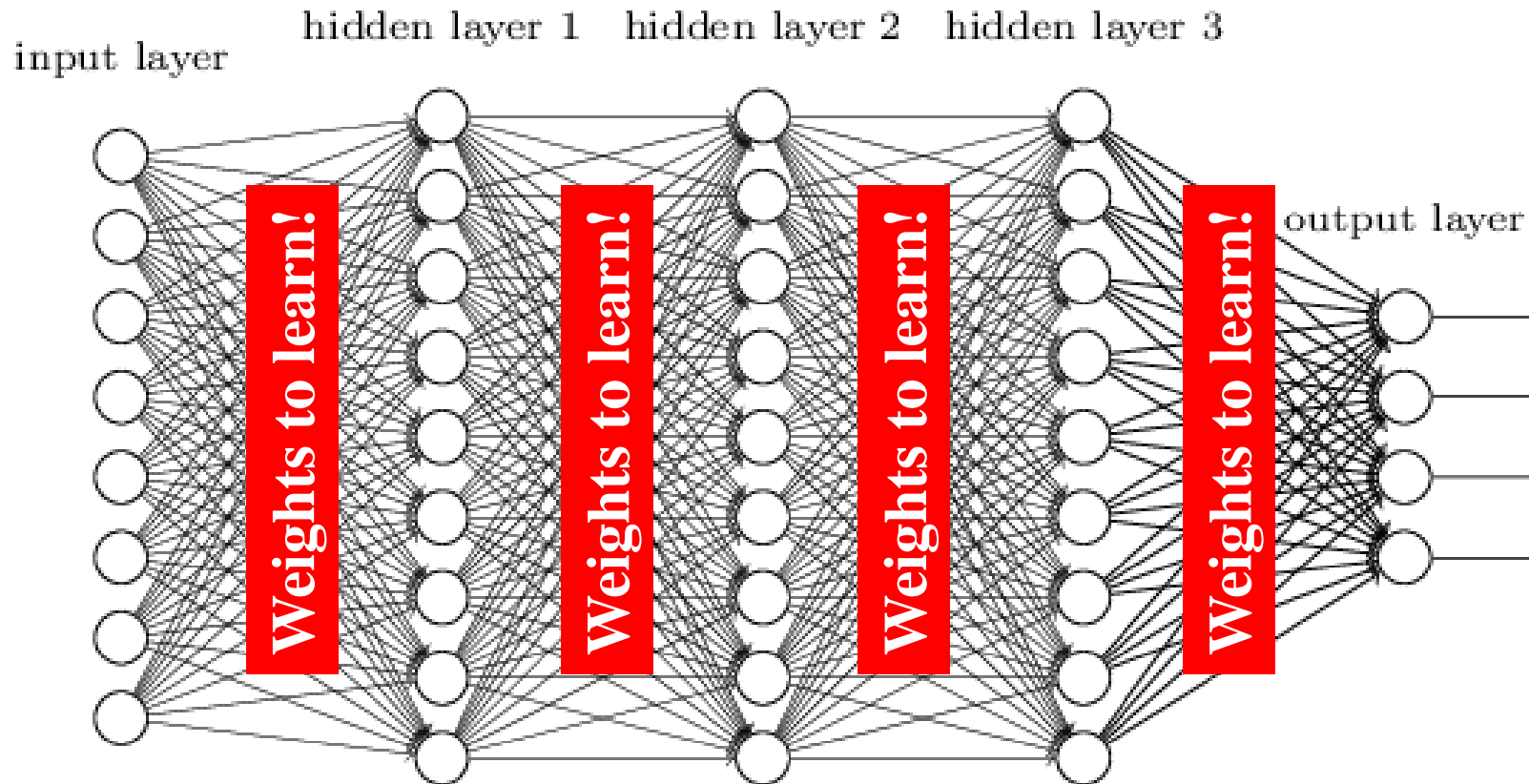
FEED-FORWARD NETWORKS

- Compute output (e.g. probability of a particular class being present in the sample)



DEEP NEURAL NETWORKS

- Lots of hidden layers
- Depth = power (usually)

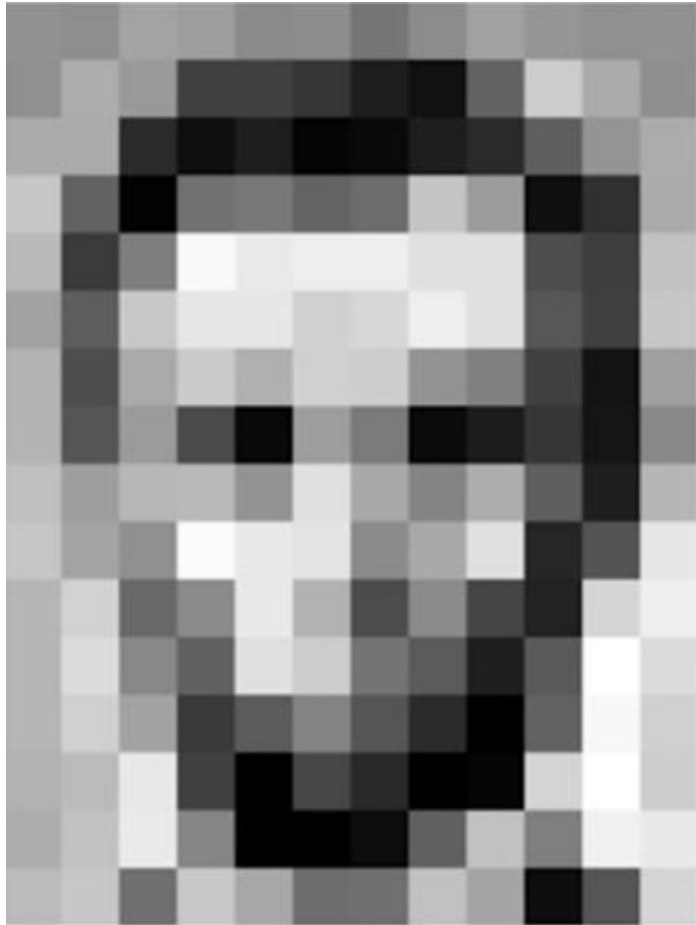


HOW DO WE LEARN THE WEIGHTS?

- There is no feasible closed-form solution for the weights
 - You can't directly set up a system $A*w = b$ and solve for w given the complexity of the network
 - Modern networks contain millions (or even billions) of parameters to learn
- Instead, we iteratively find a set of weights that produce the outputs we want on a set of data used for **training**
- Let's consider a basic image classification problem

WHAT COMPUTERS “SEE”

- An image is a matrix of numbers (e.g. $h \times w \times 3$ (for an RGB image))



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	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	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	86	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

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	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	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	86	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

CLASSIFICATION GOAL

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Example dataset: **CIFAR-10**

10 labels

50,000 training images

each image is **32x32x3**

10,000 test images.

CLASSIFICATION SCORES



[32x32x3]

array of numbers 0...1
(3072 numbers total)

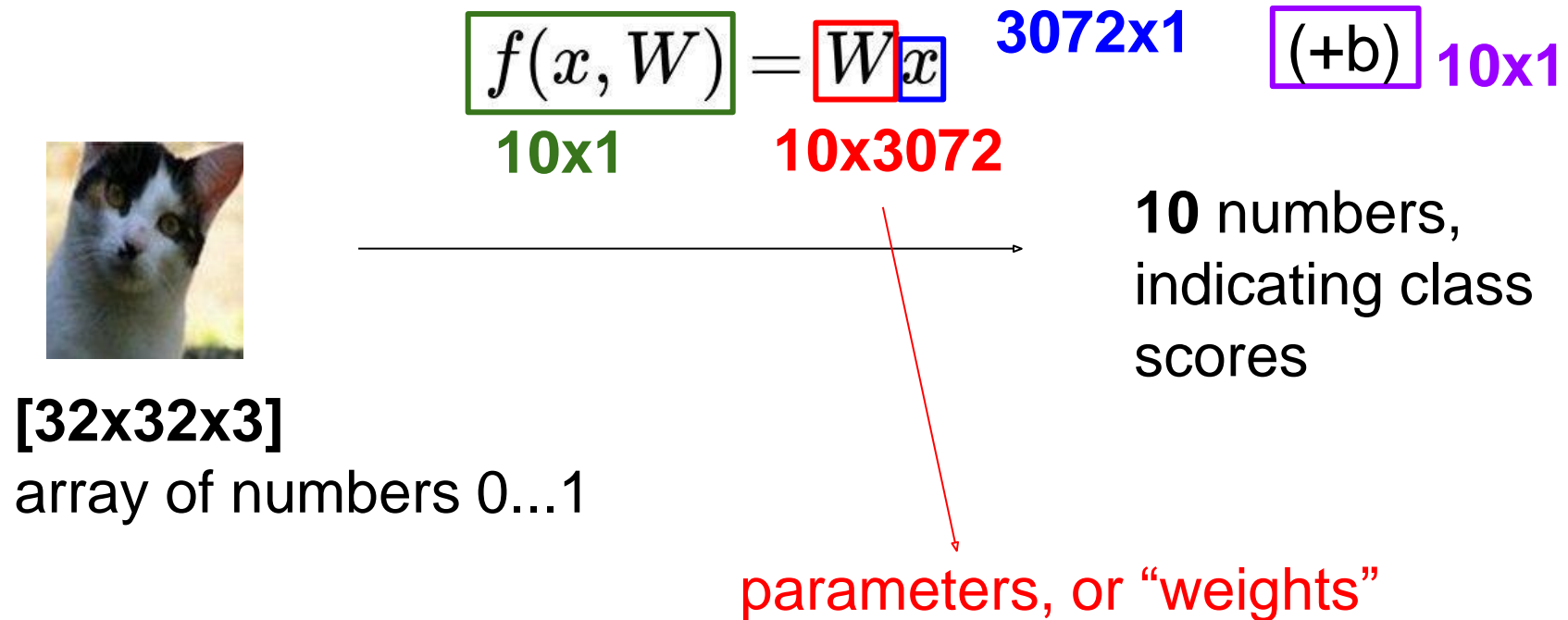
$$f(x, W) = Wx$$

$f(x, W)$



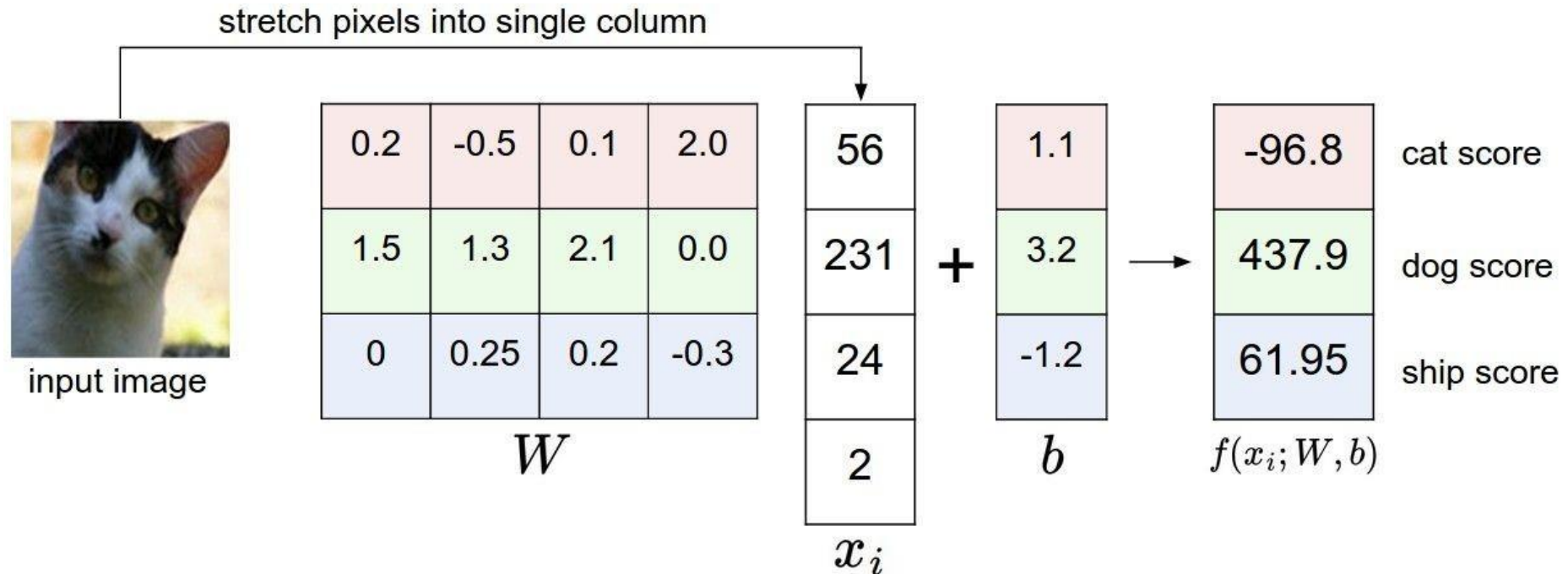
10 numbers,
indicating class
scores

LINEAR CLASSIFIER



LINEAR CLASSIFIER

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



LINEAR CLASSIFIER

Going forward: Loss function/Optimization



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1

TODO:

1. Define a **loss function** that quantifies our unhappiness with the scores across the training data.
2. Come up with a way of efficiently finding the parameters that minimize the loss function.
(optimization)

LINEAR CLASSIFIER

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1

LINEAR CLASSIFIER: HINGE LOSS

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1

Hinge loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the
scores vector: $s = f(x_i, W)$

the loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Want: $s_{y_i} \geq s_j + 1$

i.e. $s_j - s_{y_i} + 1 \leq 0$

If true, loss is 0

If false, loss is magnitude of violation

LINEAR CLASSIFIER: HINGE LOSS

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1
Losses:	2.9		

Hinge loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the
scores vector: $s = f(x_i, W)$

the loss has the form:

$$\begin{aligned} L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\ &= \max(0, 5.1 - 3.2 + 1) \\ &\quad + \max(0, -1.7 - 3.2 + 1) \\ &= \max(0, 2.9) + \max(0, -3.9) \\ &= 2.9 + 0 \\ &= 2.9 \end{aligned}$$

LINEAR CLASSIFIER: HINGE LOSS

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1
Losses:	2.9	0	

Hinge loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the
scores vector: $s = f(x_i, W)$

the loss has the form:

$$\begin{aligned} L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\ &= \max(0, 1.3 - 4.9 + 1) \\ &\quad + \max(0, 2.0 - 4.9 + 1) \\ &= \max(0, -2.6) + \max(0, -1.9) \\ &= 0 + 0 \\ &= 0 \end{aligned}$$

LINEAR CLASSIFIER: HINGE LOSS

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1
Losses:	2.9	0	12.9

Hinge loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the
scores vector: $s = f(x_i, W)$

the loss has the form:

$$\begin{aligned} L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\ &= \max(0, 2.2 - (-3.1) + 1) \\ &\quad + \max(0, 2.5 - (-3.1) + 1) \\ &= \max(0, 5.3 + 1) \\ &\quad + \max(0, 5.6 + 1) \\ &= 6.3 + 6.6 \\ &= 12.9 \end{aligned}$$

LINEAR CLASSIFIER: HINGE LOSS

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1
Losses:	2.9	0	12.9

Hinge loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the
scores vector: $s = f(x_i, W)$

the loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

and the full training loss is the mean
over all examples in the training data:

$$L = \frac{1}{N} \sum_{i=1}^N L_i$$

$$L = (2.9 + 0 + 12.9) / 3 \\ = 15.8 / 3 = 5.3$$

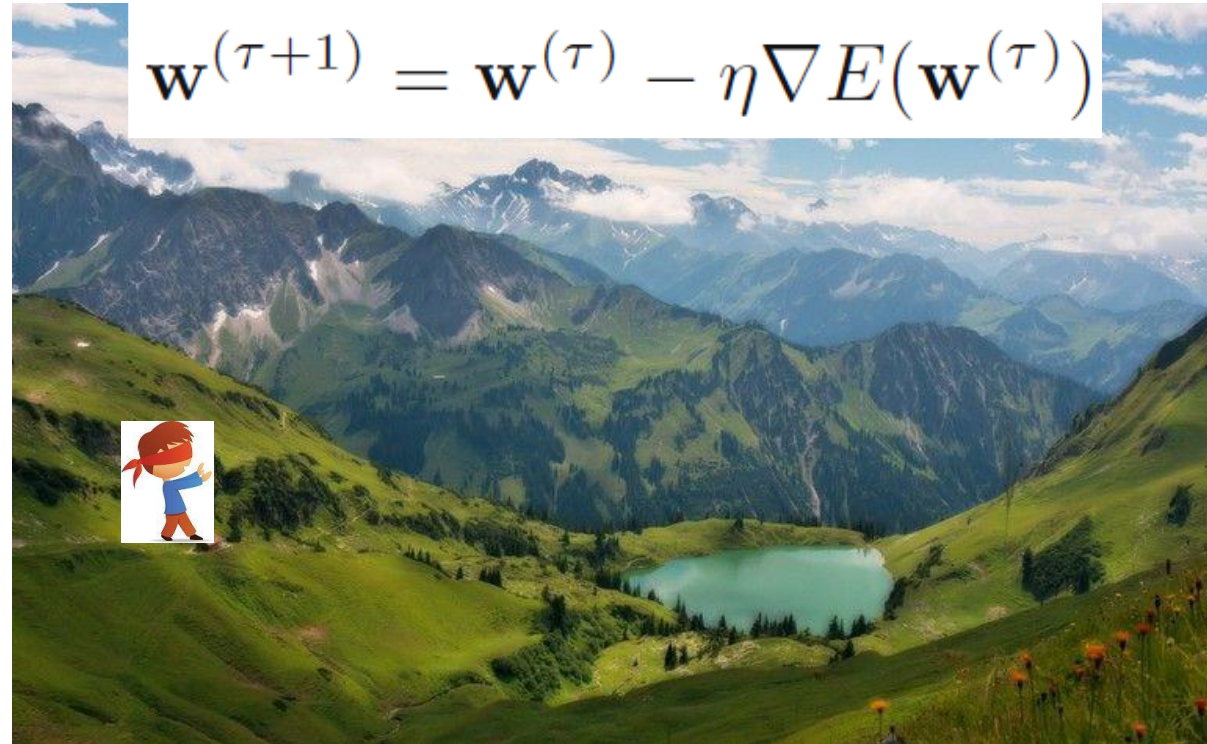
TO MINIMIZE LOSS, USE GRADIENT DESCENT

- Take *small* steps in the direction of lower loss at each time step
 - Show different training sample(s) at each time step

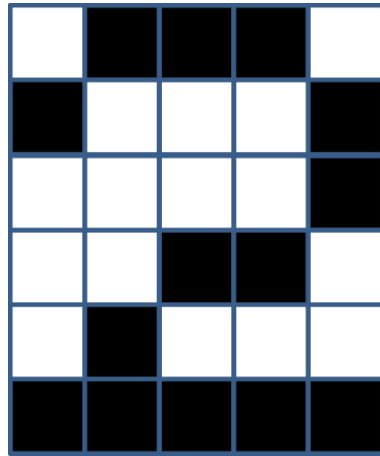


TO MINIMIZE LOSS, USE GRADIENT DESCENT

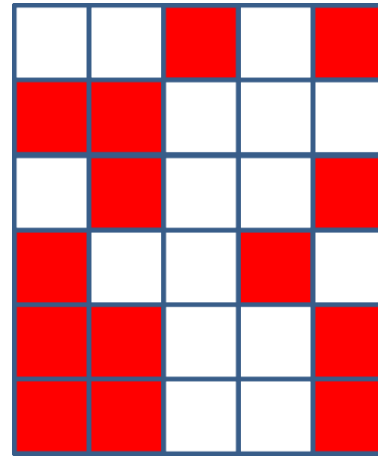
- Take *small* steps in the direction of lower loss at each time step
 - η is the *learning rate* (controls how big of a step you take)
 - $\nabla E(\mathbf{w}^{(\tau)})$ is the gradient of the loss with respect to the current weights



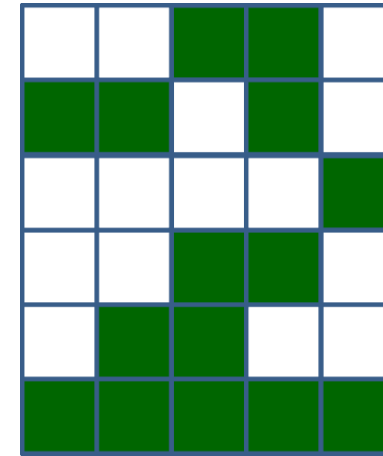
THE WEIGHTS LOOK FOR PATTERNS



$$y = \begin{cases} 1 & \text{if } \sum w_i x_i \geq T \\ 0 & \text{else} \end{cases}$$



Correlation = 0.57

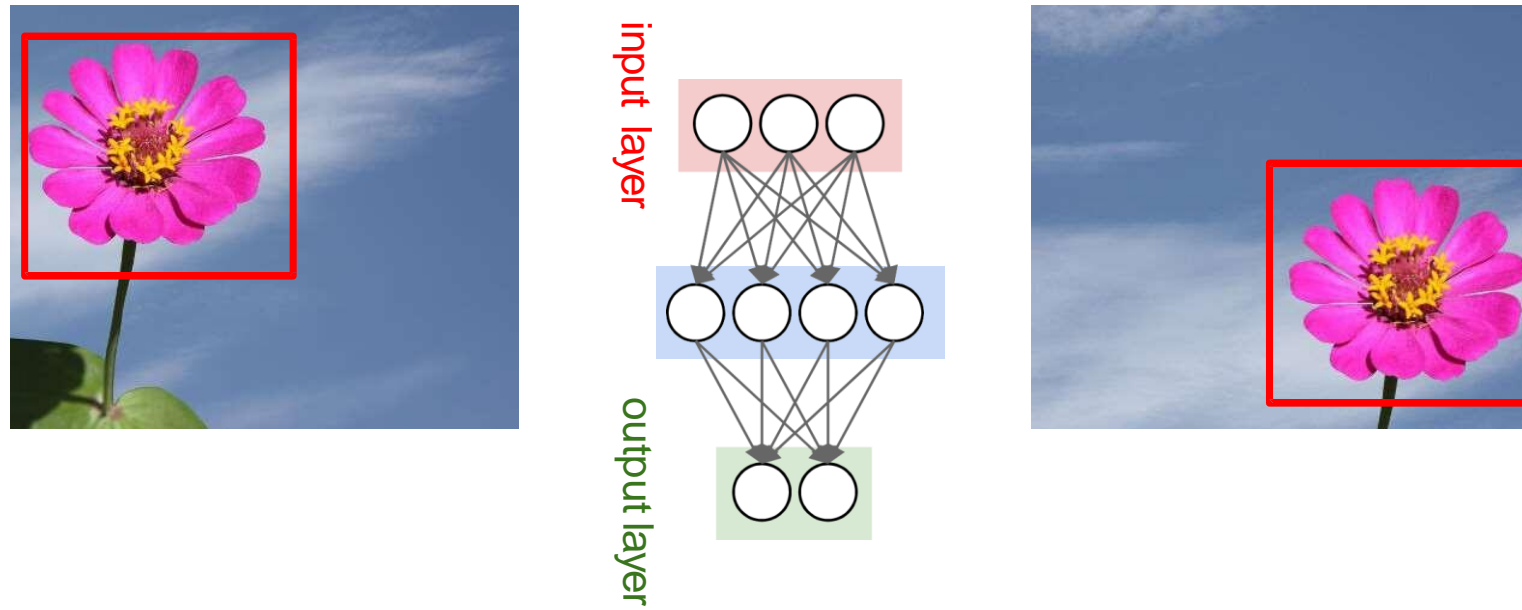


Correlation = 0.82



- The green pattern (input) looks more like the weights (black – on left) than the red pattern
 - The green pattern is more *correlated* with the weights

SHORTCOMINGS OF WHAT WE HAVE DISCUSSED



- Will a NN that recognizes the left image as a flower also recognize the one on the right as a flower?
- Need a network that will recognize the flower regardless of its spatial location in the image

REAL-WORLD IMAGES ARE CHALLENGING



Illumination



Object pose



Clutter



Occlusions



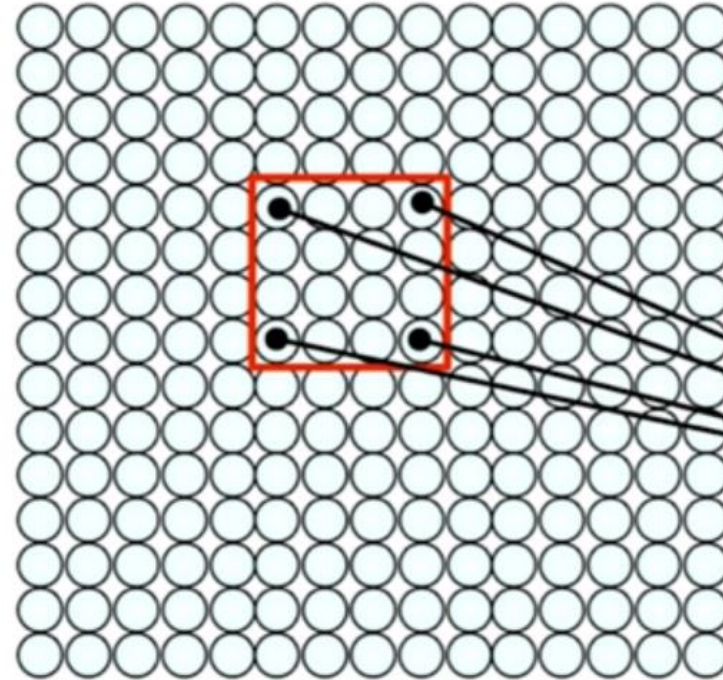
**Intra-class
appearance**



Viewpoint

PRESERVING SPATIAL STRUCTURE

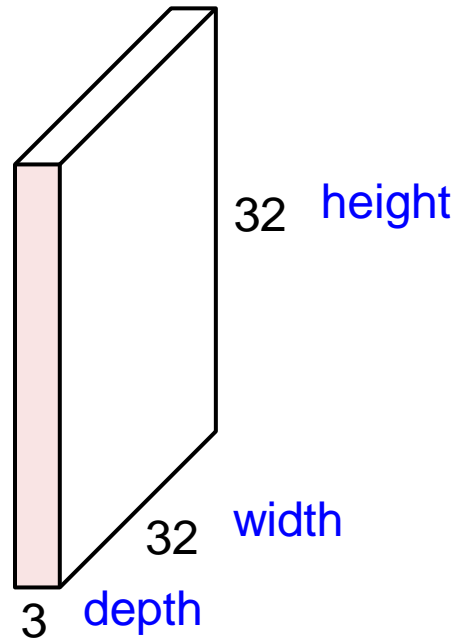
Input: 2D image.
Array of pixel values



Idea: connect patches of input
to neurons in hidden layer.
Neuron connected to region of
input. Only "sees" these values.

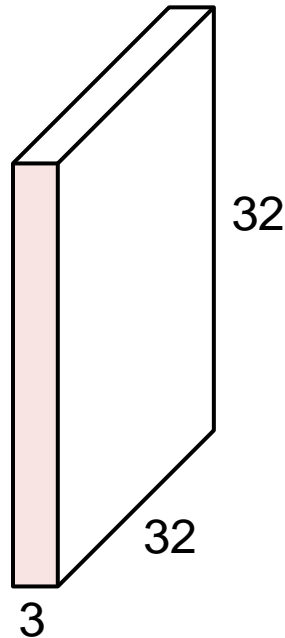
CONVOLUTIONS: MORE DETAIL

32x32x3 image

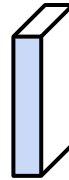


CONVOLUTIONS: MORE DETAIL

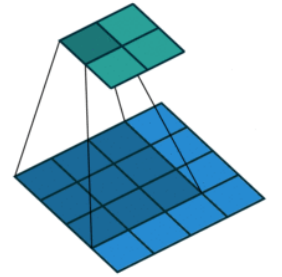
32x32x3 image



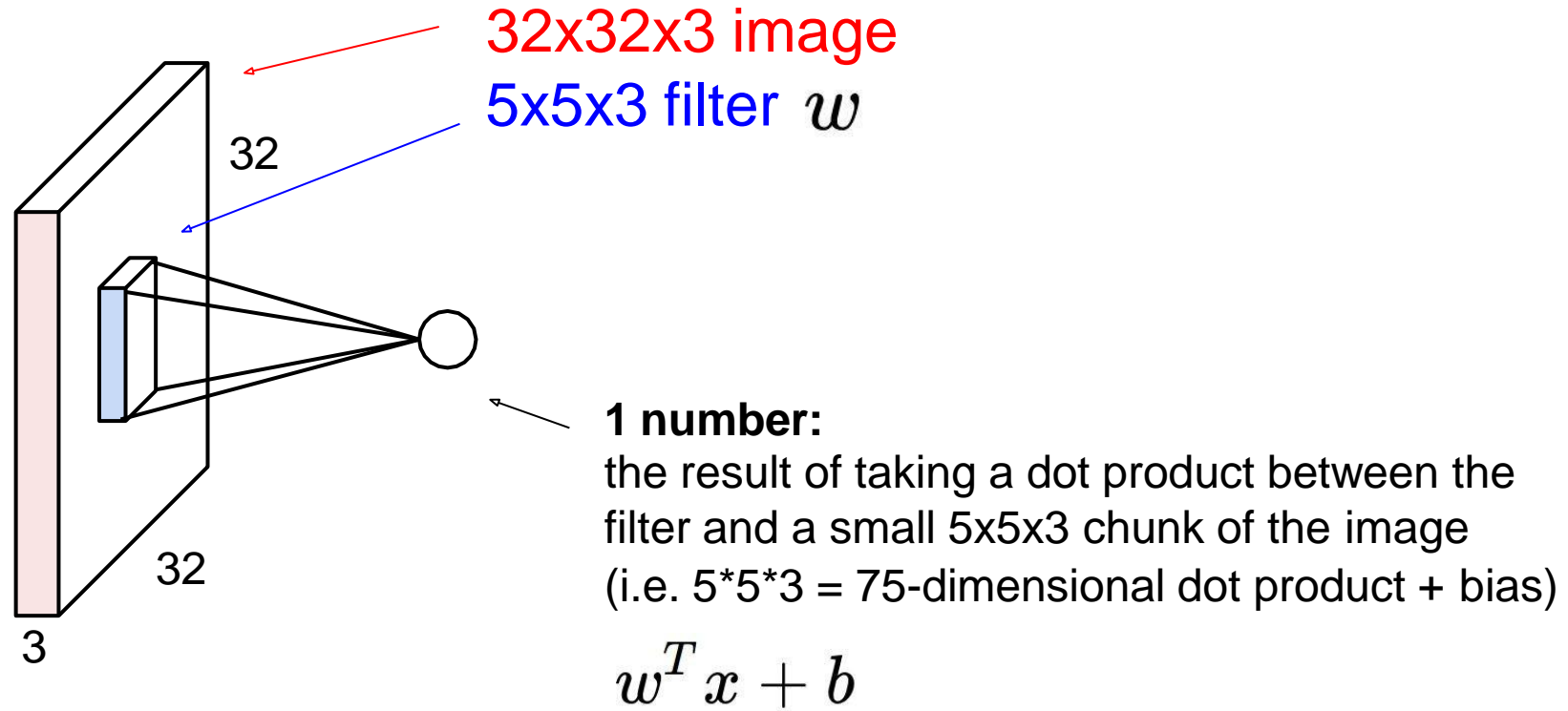
5x5x3 filter



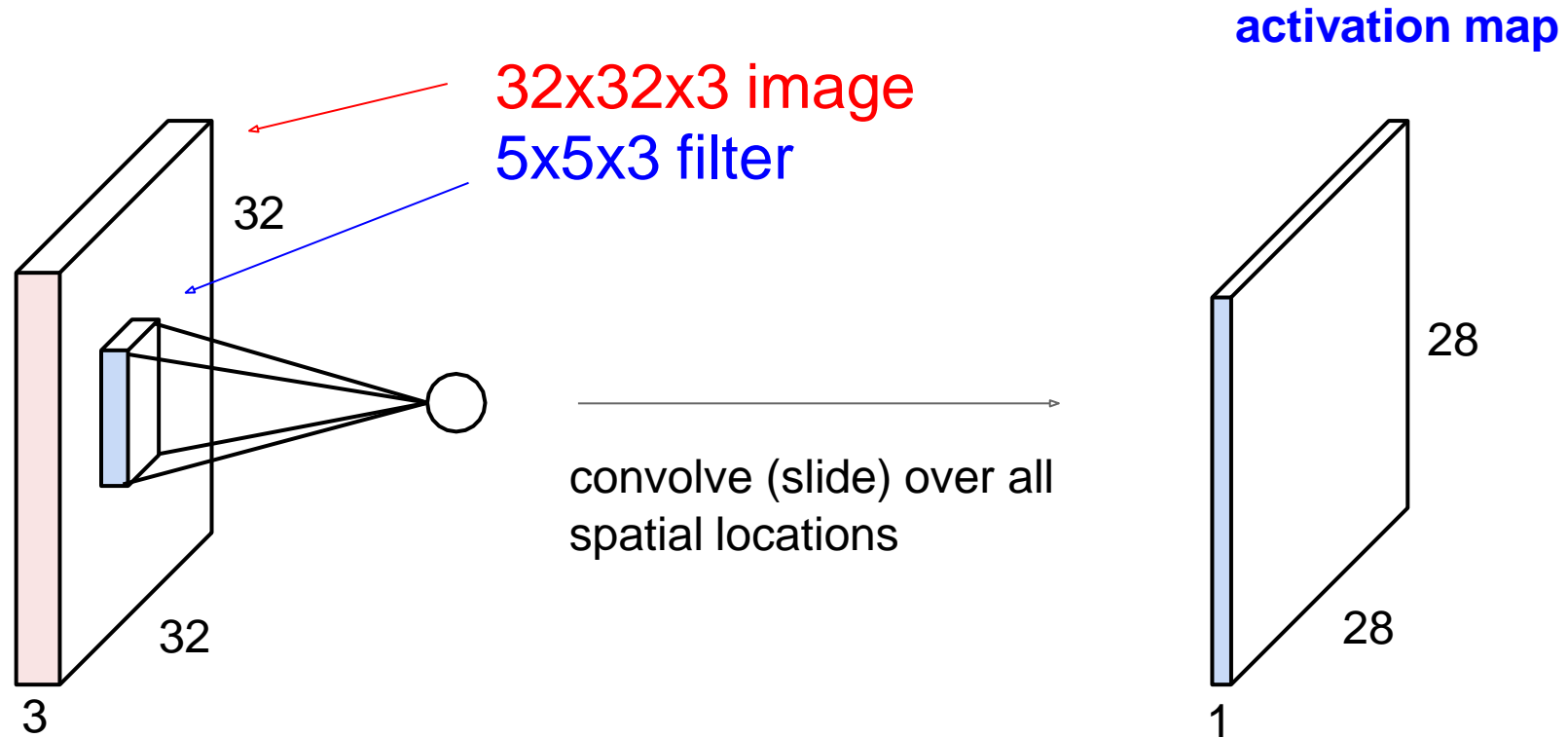
Convolve the filter with the image
i.e. “slide over the image spatially,
computing dot products”



CONVOLUTIONAL LAYER



CONVOLUTIONAL LAYER



MANUALLY DEFINED FILTER MAPS

- Effect of convolving manually handcrafted filters on an image
- In a convolutional network, these filters are automatically learned!
 - The filter parameters are the weights



Original



Sharpen



Edge Detect



"Strong" Edge
Detect

Example: box filter

$g[\cdot, \cdot]$

	1	1	1
1	1	1	1
9	1	1	1

IMAGE FILTERING

$$g[\cdot, \cdot] \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot, \cdot]$

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	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$h[\cdot, \cdot]$

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

IMAGE FILTERING

$$g[\cdot, \cdot] \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot, \cdot]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[\cdot, \cdot]$

	0	10							

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

IMAGE FILTERING

$$g[\cdot, \cdot] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot, \cdot]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[\cdot, \cdot]$

	0	10	20						

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

IMAGE FILTERING

$$g[\cdot, \cdot] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot, \cdot]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[\cdot, \cdot]$

	0	10	20	30					

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

IMAGE FILTERING

$$g[\cdot, \cdot] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot, \cdot]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[\cdot, \cdot]$

	0	10	20	30	30				

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

IMAGE FILTERING

$$g[\cdot, \cdot] \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot, \cdot]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[\cdot, \cdot]$

	0	10	20	30	30				
						?			
				50					

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

IMAGE FILTERING

$$g[\cdot, \cdot] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot, \cdot]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[\cdot, \cdot]$

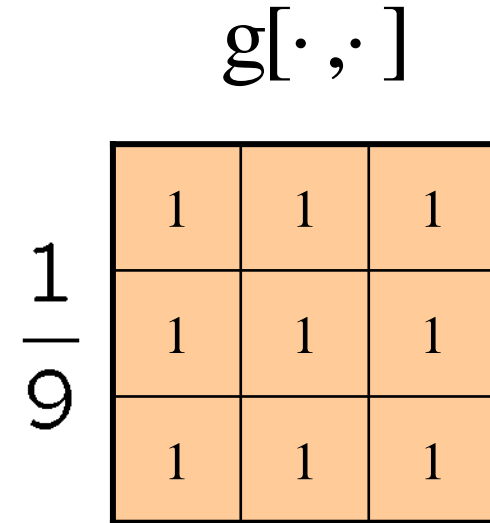
	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

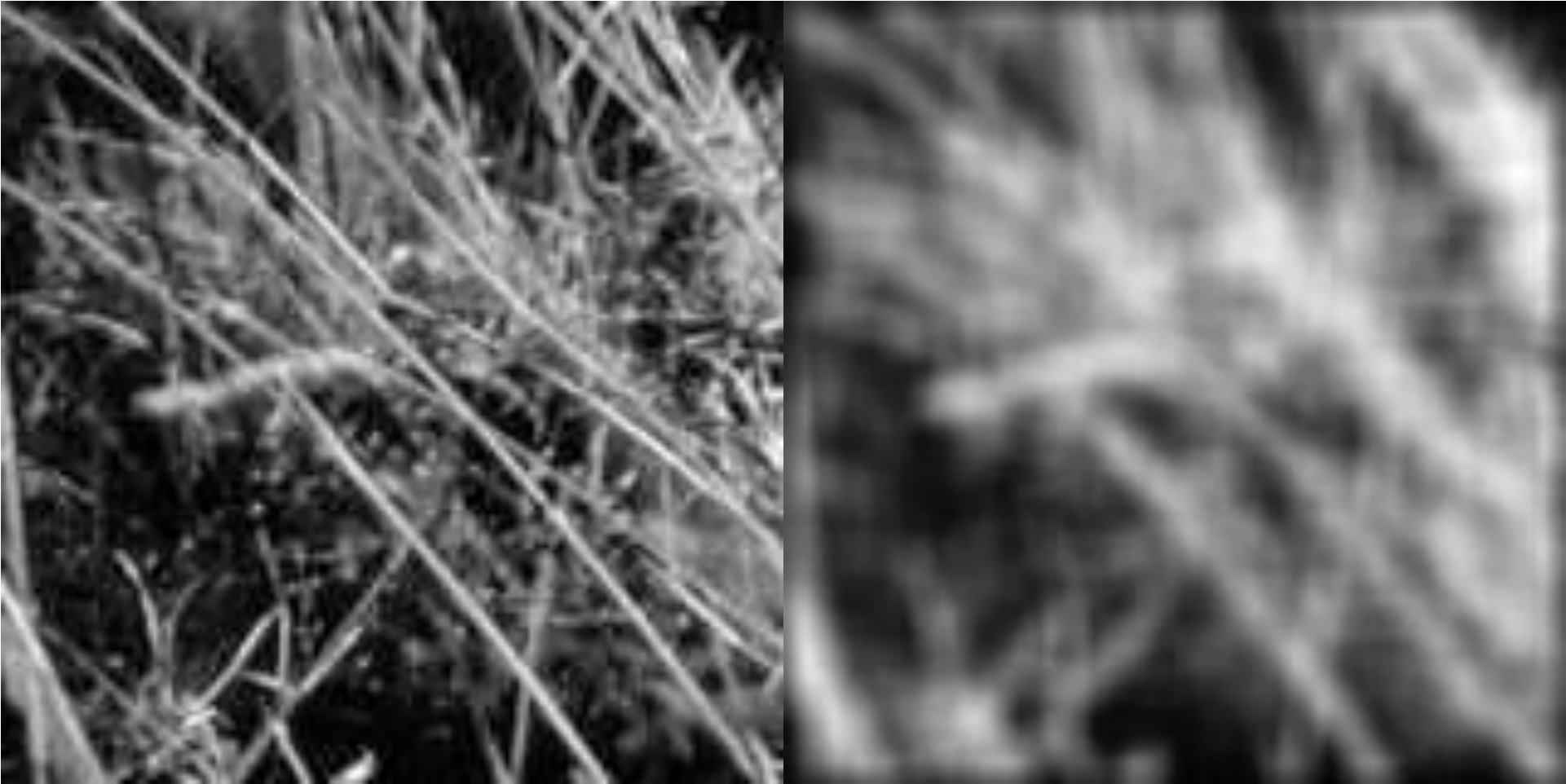
BOX FILTER

What does it do?

- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)



SMOOTHING WITH BOX FILTER



PRACTICE WITH LINEAR FILTERS



Original

0	0	0
0	1	0
0	0	0

?

PRACTICE WITH LINEAR FILTERS



Original

0	0	0
0	1	0
0	0	0



Filtered
(no change)

PRACTICE WITH LINEAR FILTERS



Original

0	0	0
0	0	1
0	0	0

?

PRACTICE WITH LINEAR FILTERS



Original

0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

PRACTICE WITH LINEAR FILTERS



Original

0	0	0
0	2	0
0	0	0

-

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

?

(Note that filter sums to 1)

PRACTICE WITH LINEAR FILTERS



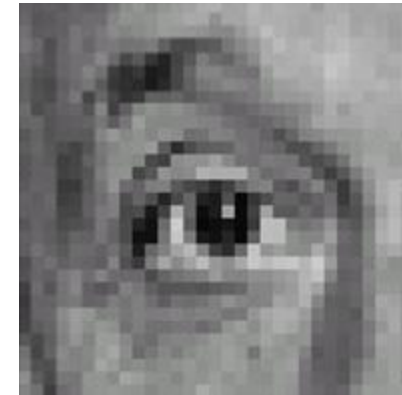
Original

0	0	0
0	2	0
0	0	0

−

$\frac{1}{9}$

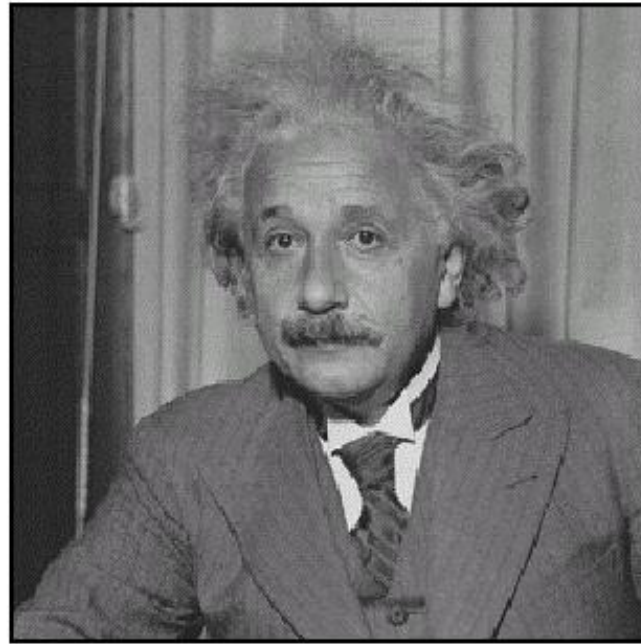
1	1	1
1	1	1
1	1	1



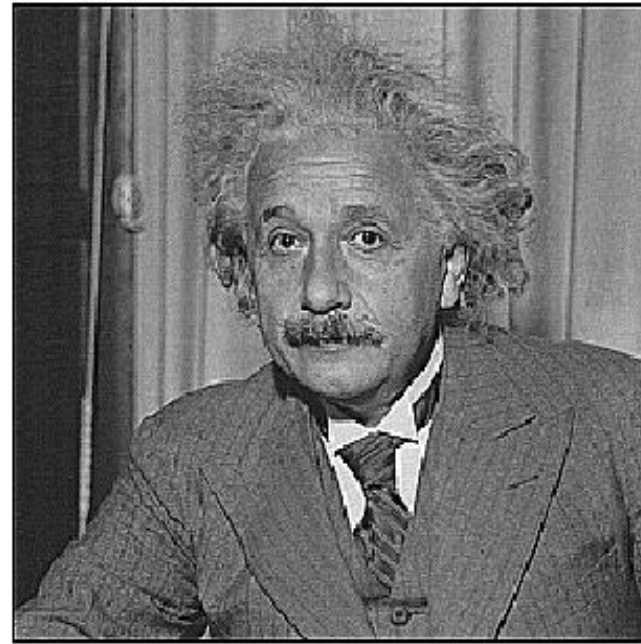
Sharpening filter

- Accentuates differences with local average

SHARPENING

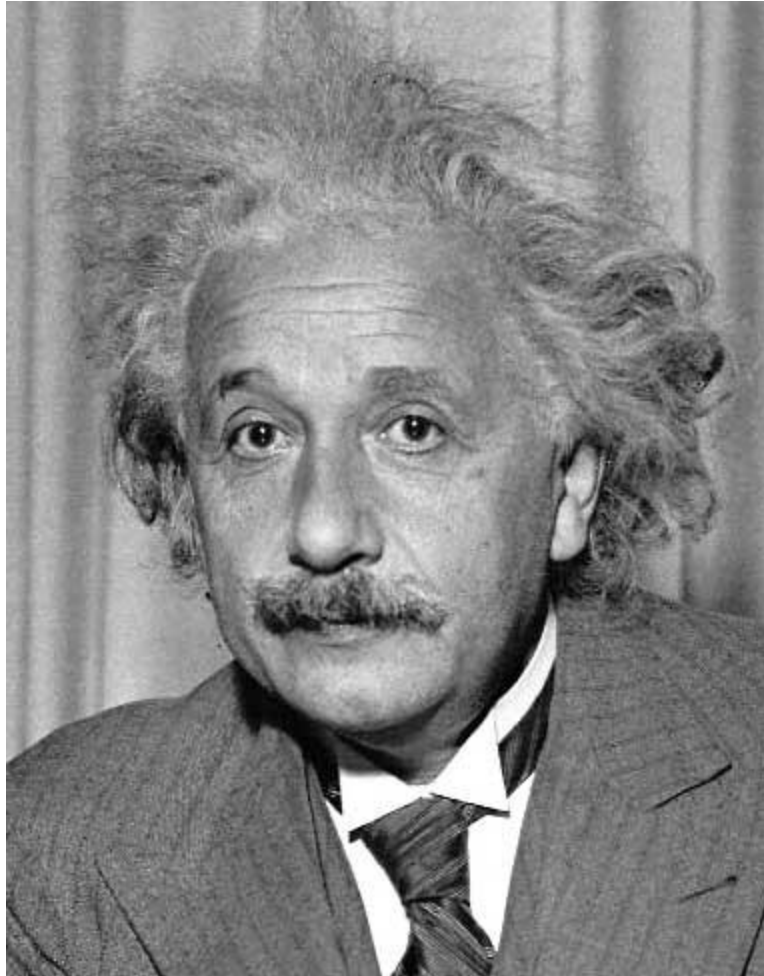


before



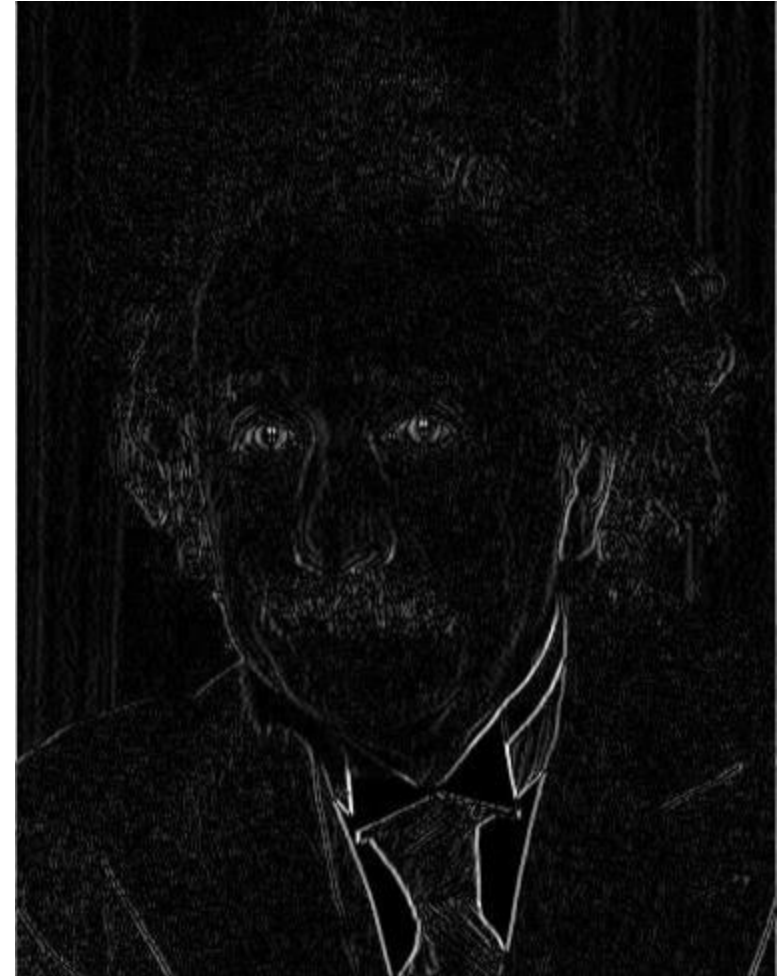
after

OTHER FILTERS



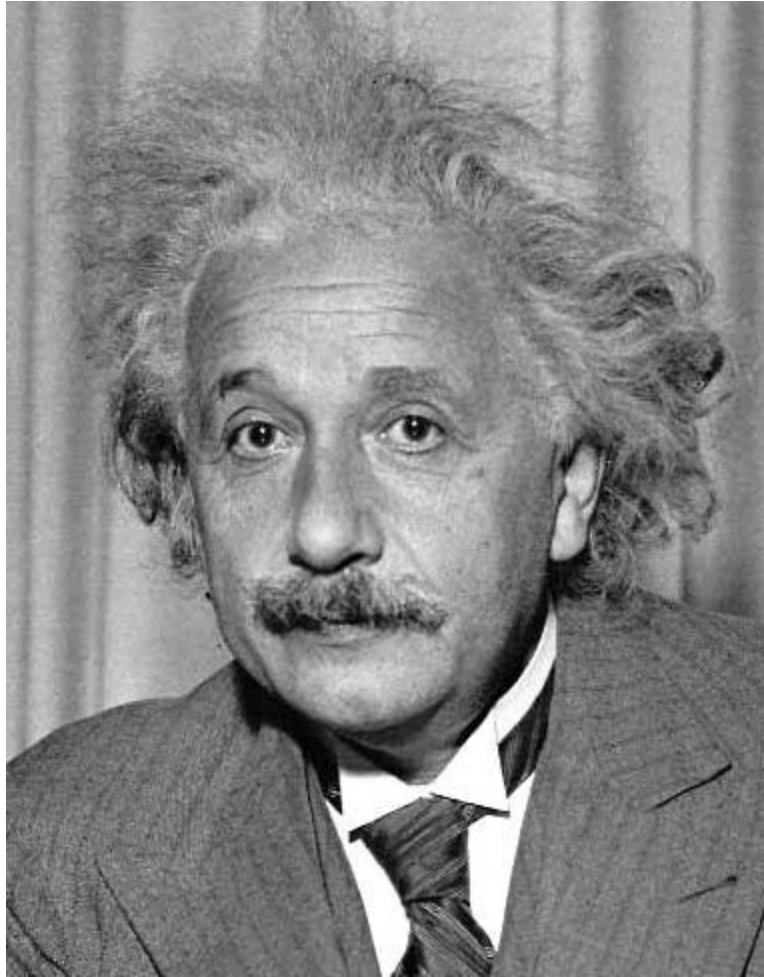
1	0	-1
2	0	-2
1	0	-1

Sobel



Vertical Edge
(absolute value)

OTHER FILTERS



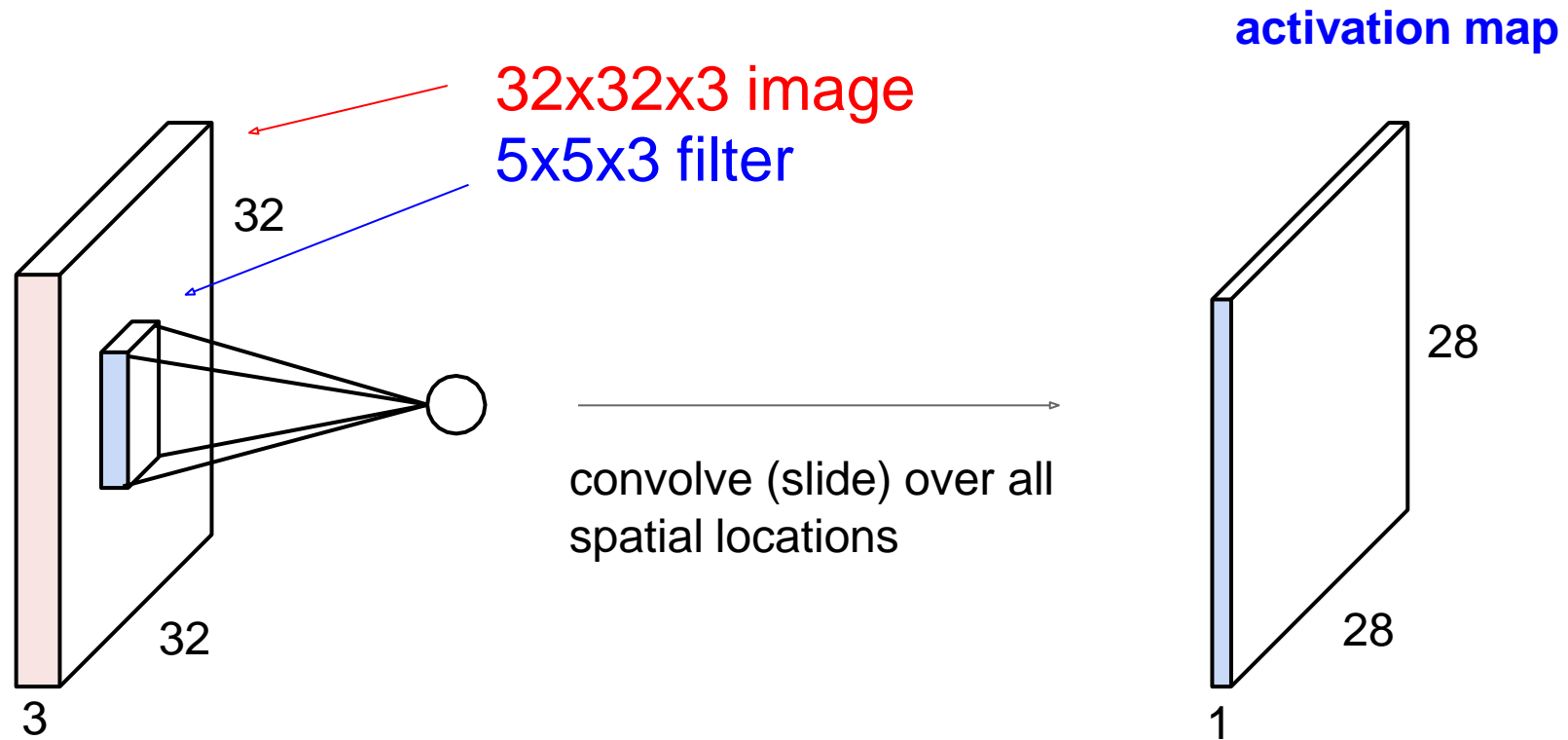
1	2	1
0	0	0
-1	-2	-1

Sobel



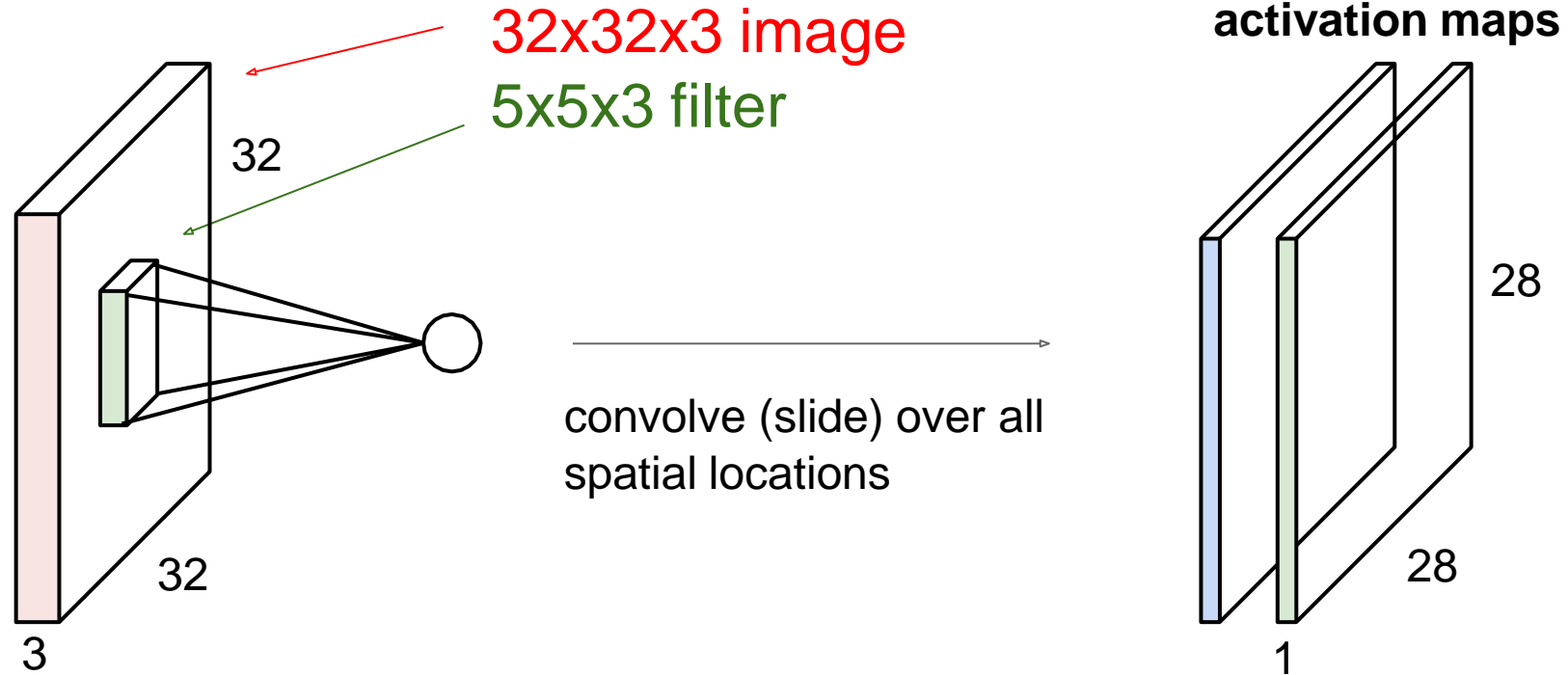
Horizontal Edge
(absolute value)

CONVOLUTIONAL LAYER



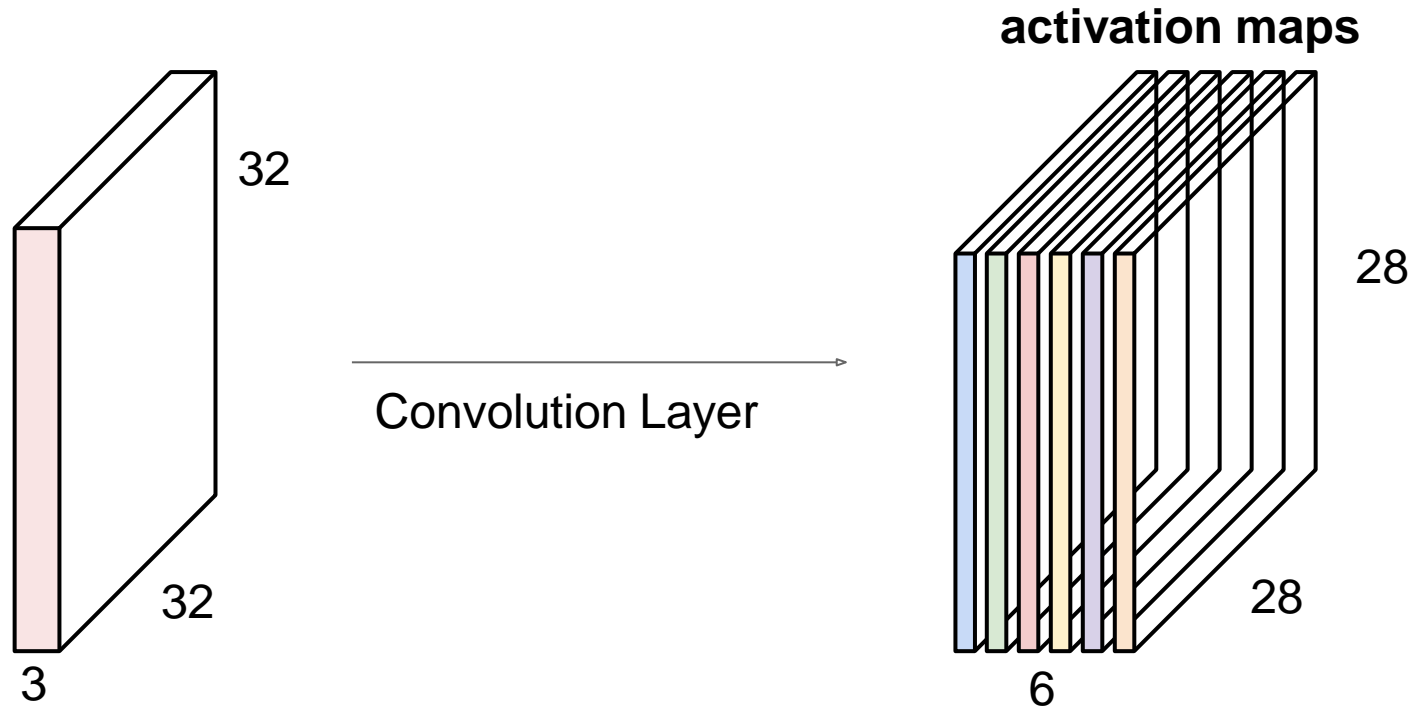
CONVOLUTIONAL LAYER

consider a second, green filter



CONVOLUTIONAL LAYER

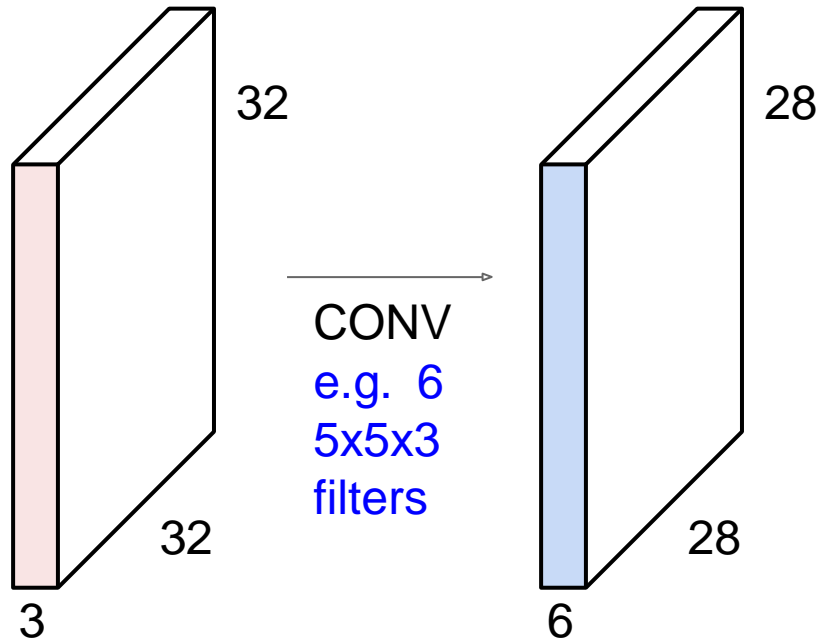
- For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



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

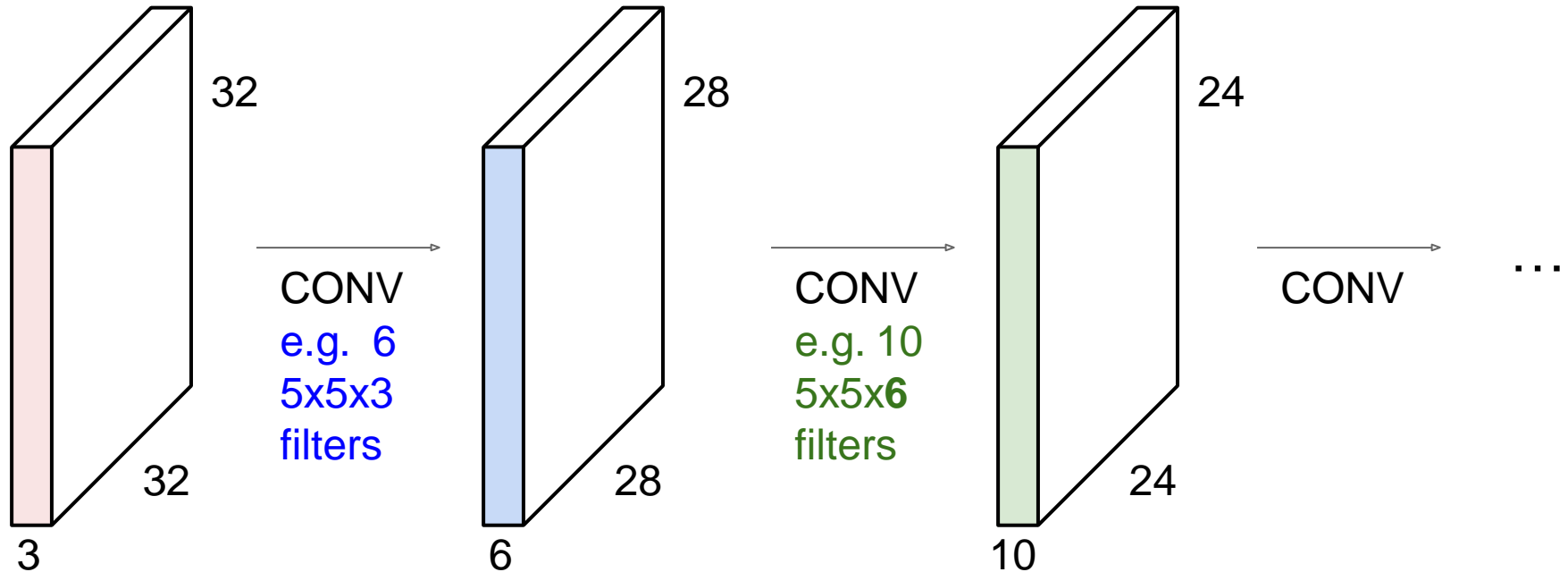
CONVOLUTIONAL NEURAL NETWORK

- A convolutional neural network (CNN) is a sequence of convolution layers, interspersed with activation functions



CONVOLUTIONAL NEURAL NETWORK

- A convolutional neural network (CNN) is a sequence of convolution layers, interspersed with activation functions

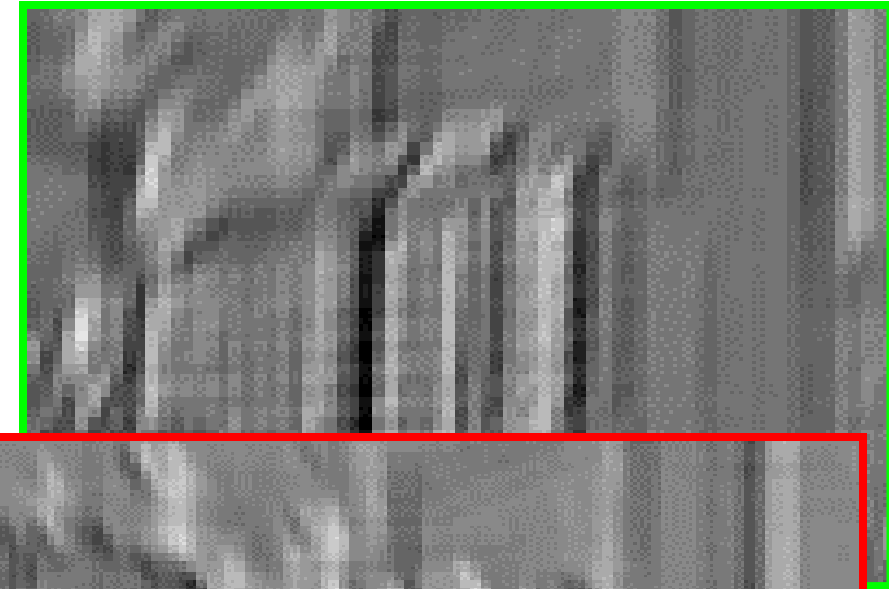
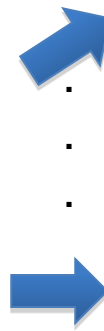


WHAT IS A CONVOLUTION?

- Weighted moving sum



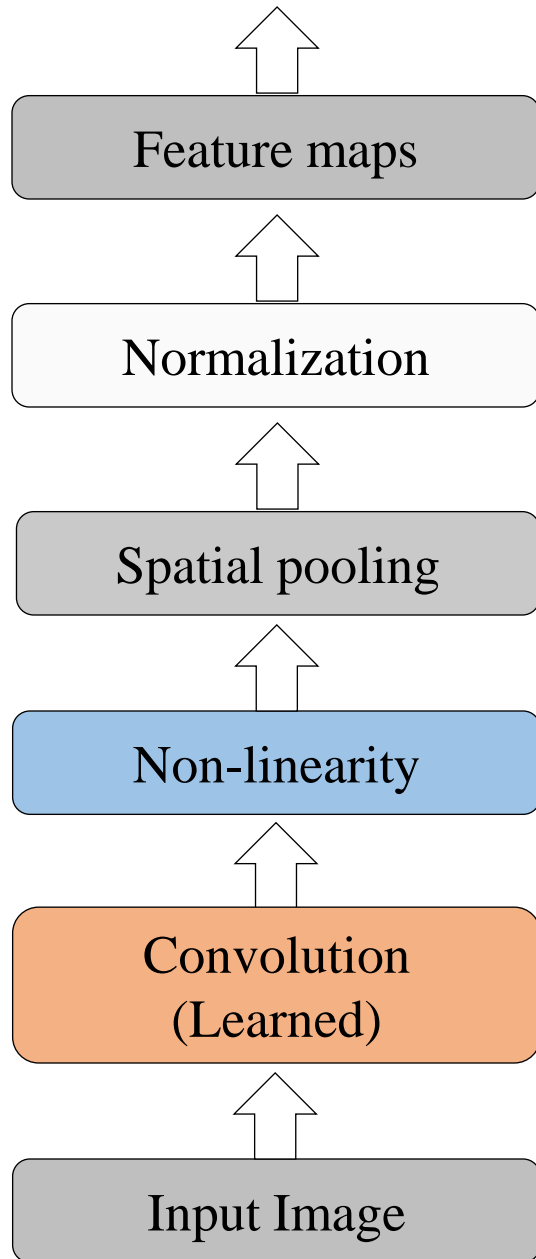
Input



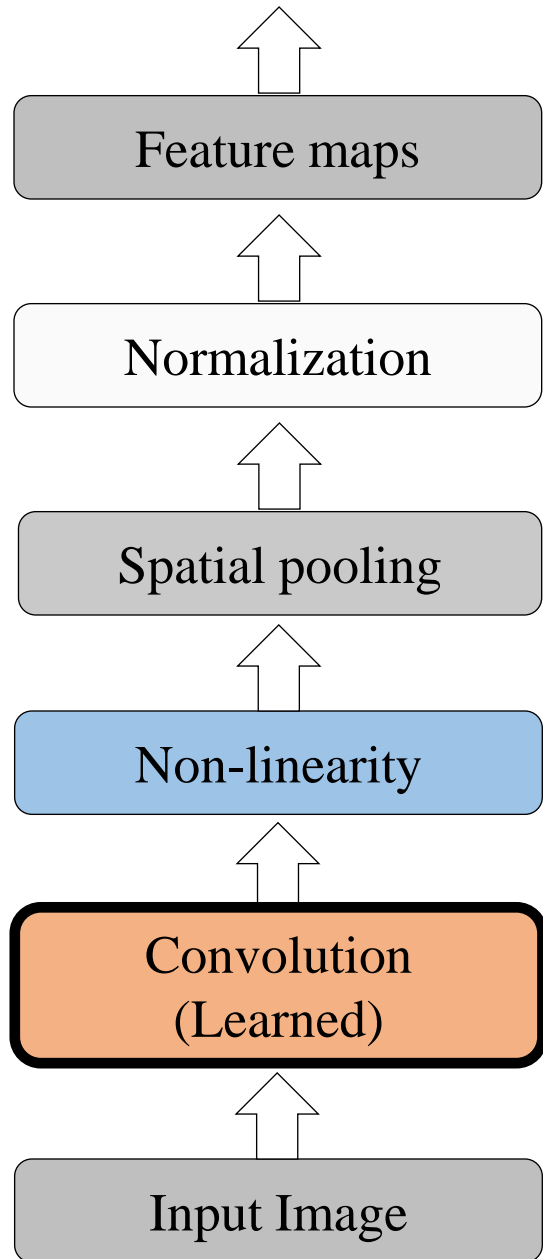
Feature Activation Map

slide credit: S. Lazebnik

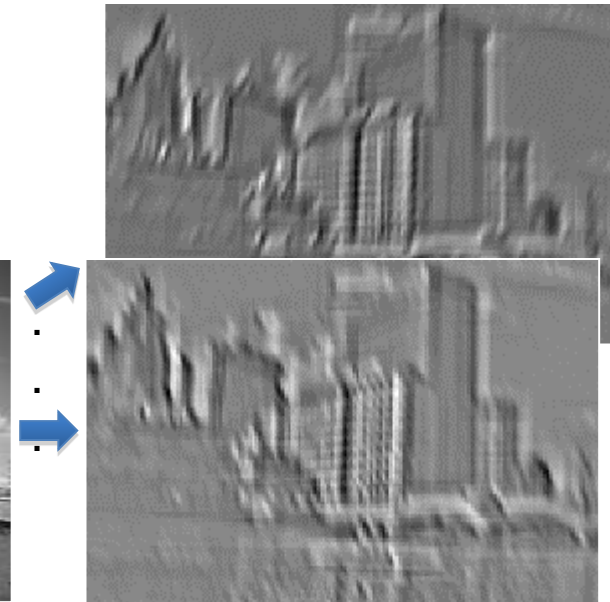
CONVOLUTIONAL NEURAL NETWORKS



CONVOLUTIONAL NEURAL NETWORKS



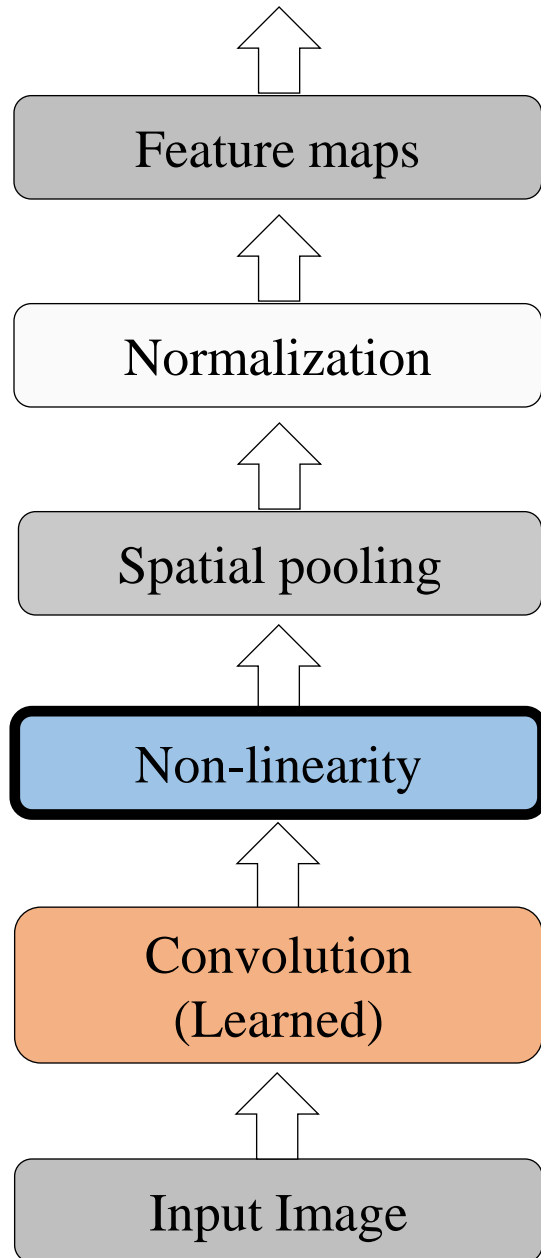
Input



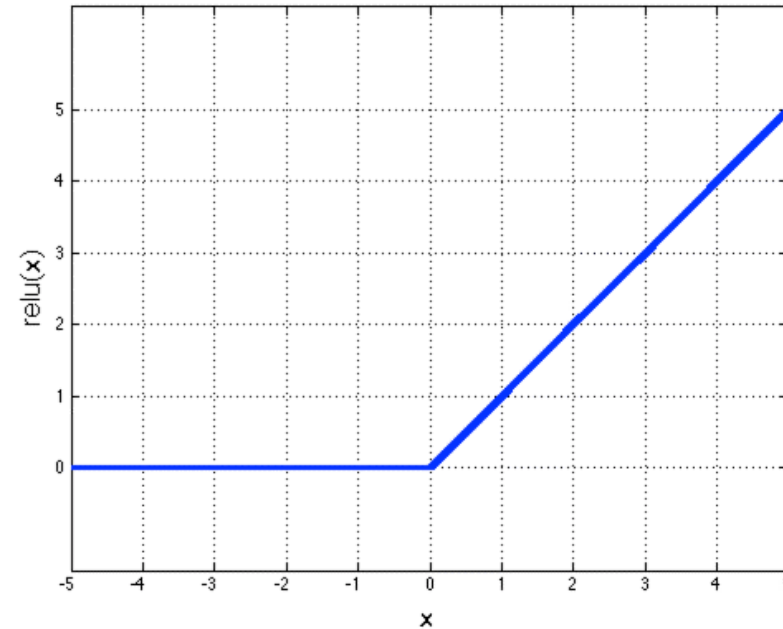
Feature Map

slide credit: S. Lazebnik

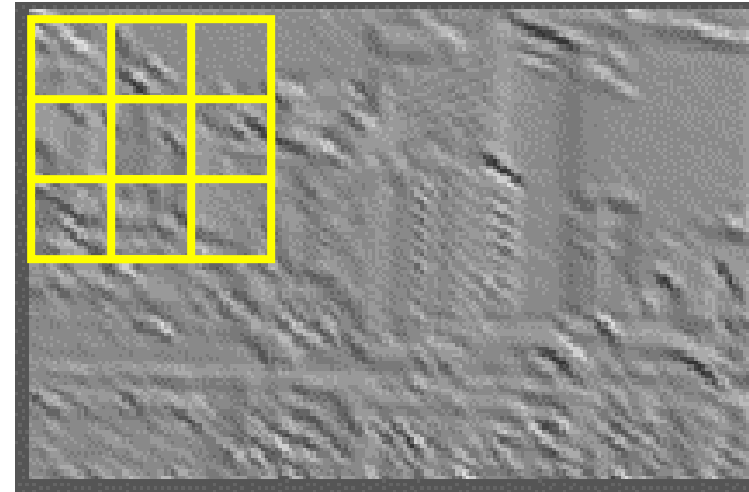
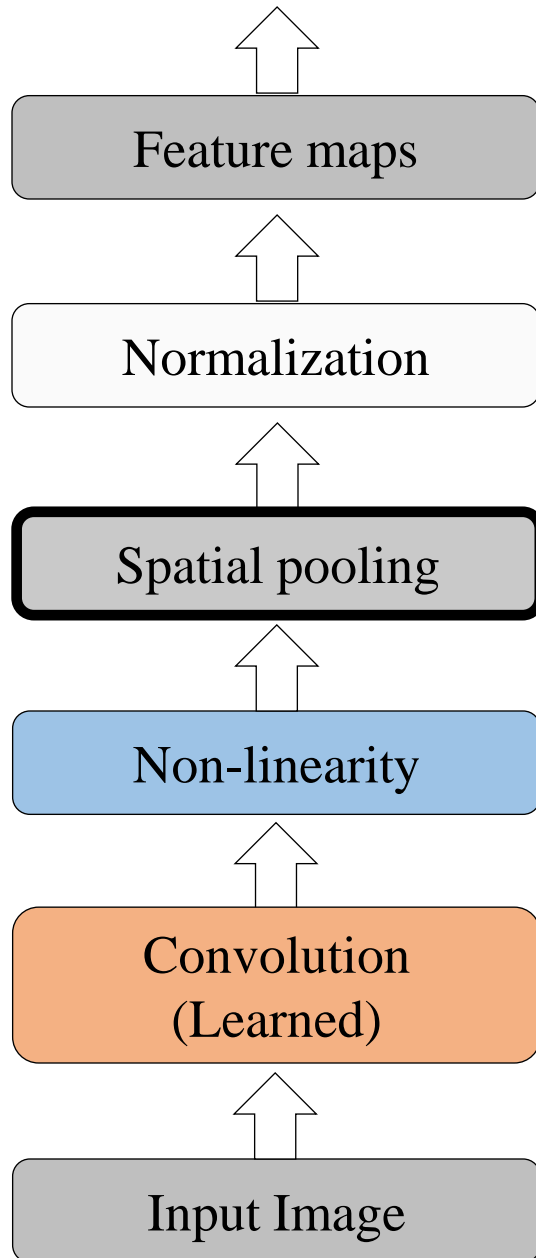
CONVOLUTIONAL NEURAL NETWORKS



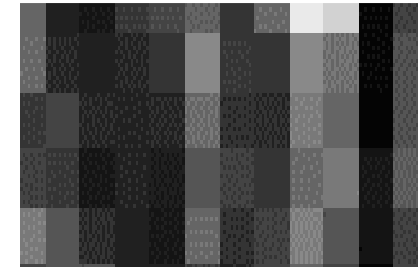
Rectified Linear Unit (ReLU)



CONVOLUTIONAL NEURAL NETWORKS



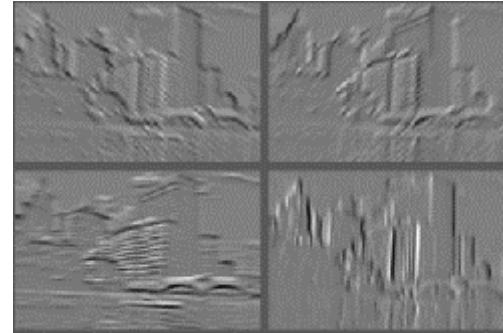
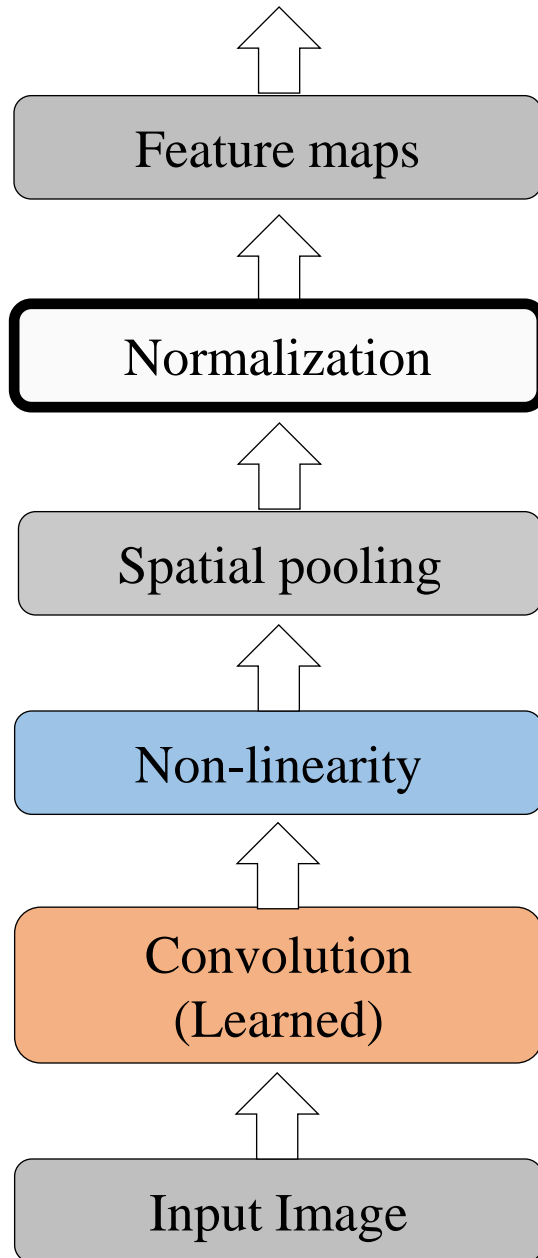
Max pooling



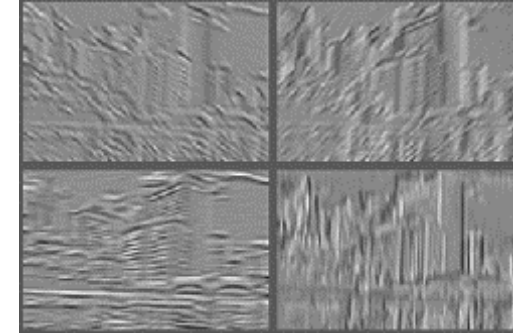
Max-pooling: a non-linear down-sampling

Provide *translation invariance*

CONVOLUTIONAL NEURAL NETWORKS



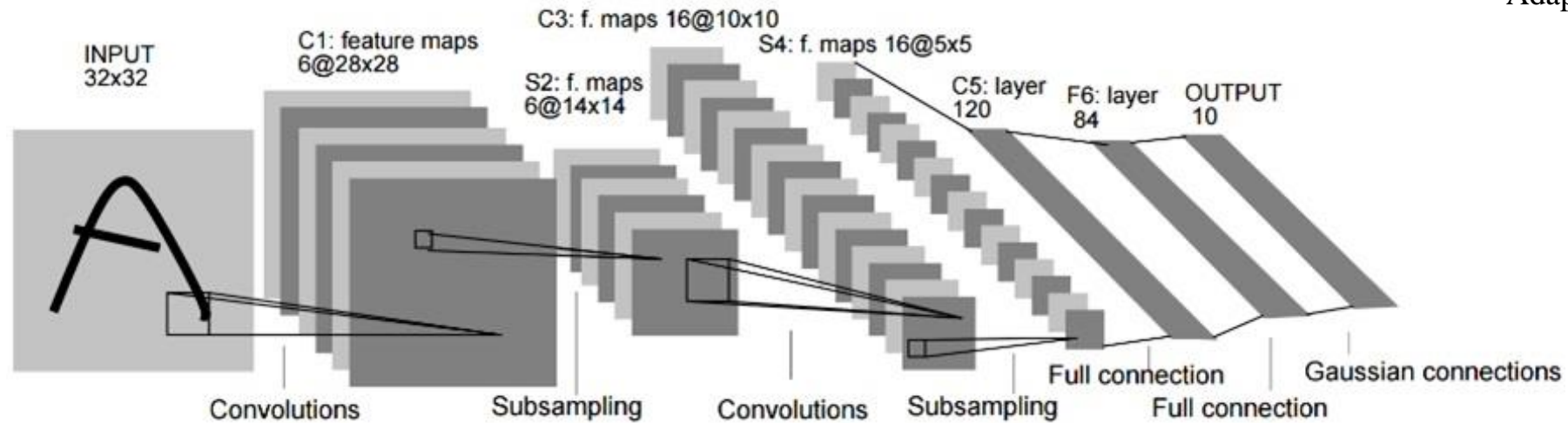
Feature Maps



Feature Maps
After Normalization

STATE OF THE ART? LENET [LECUN ET AL. 1998]

Adapted from Jia-Bin Huang



Gradient-based learning applied to document recognition [[LeCun, Bottou, Bengio, Haffner 1998](#)]

LeNet-1 from 1993

CONVOLUTIONS: MORE DETAIL

one filter =>
one activation map

example 5x5 filters
(32 total)

Activations:

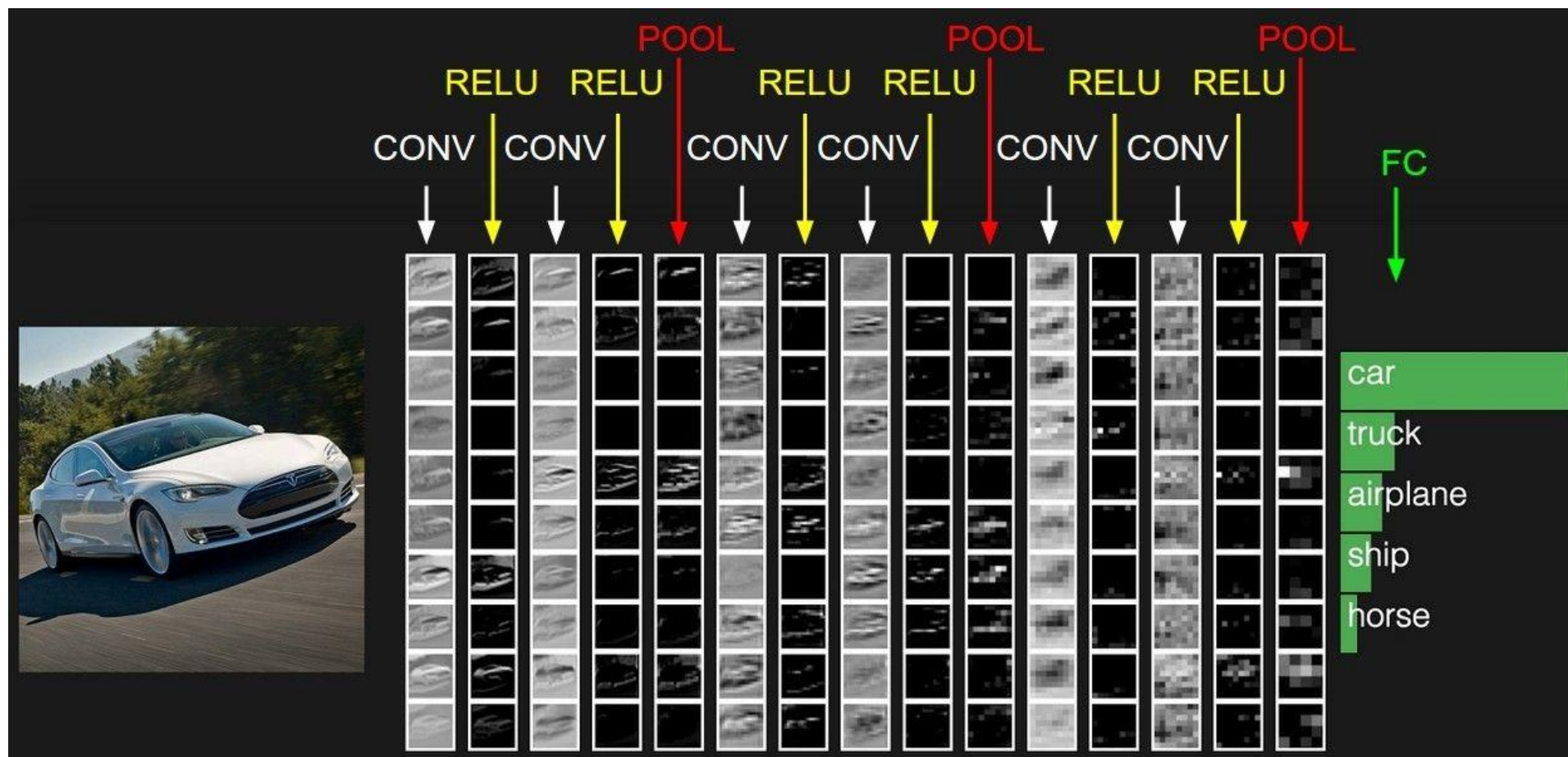
We call the layer convolutional because it is related to convolution of two signals:

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v] F[i + u, j + v]$$

Element-wise multiplication and sum of a filter and the signal (image)

PUTTING IT ALL TOGETHER

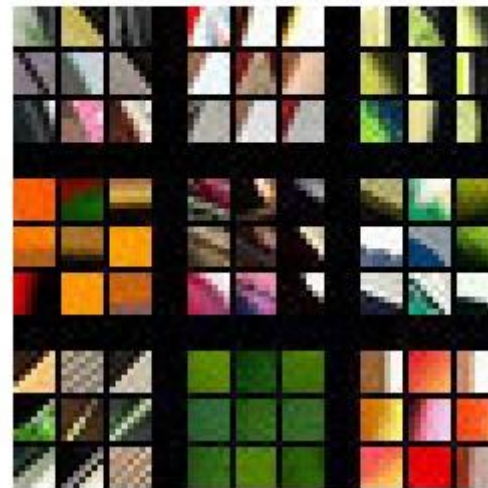
- RELU just means set any negative outputs to 0
- Pool shrinks each activation map by passing a “filter” that just takes the max of all values at each location



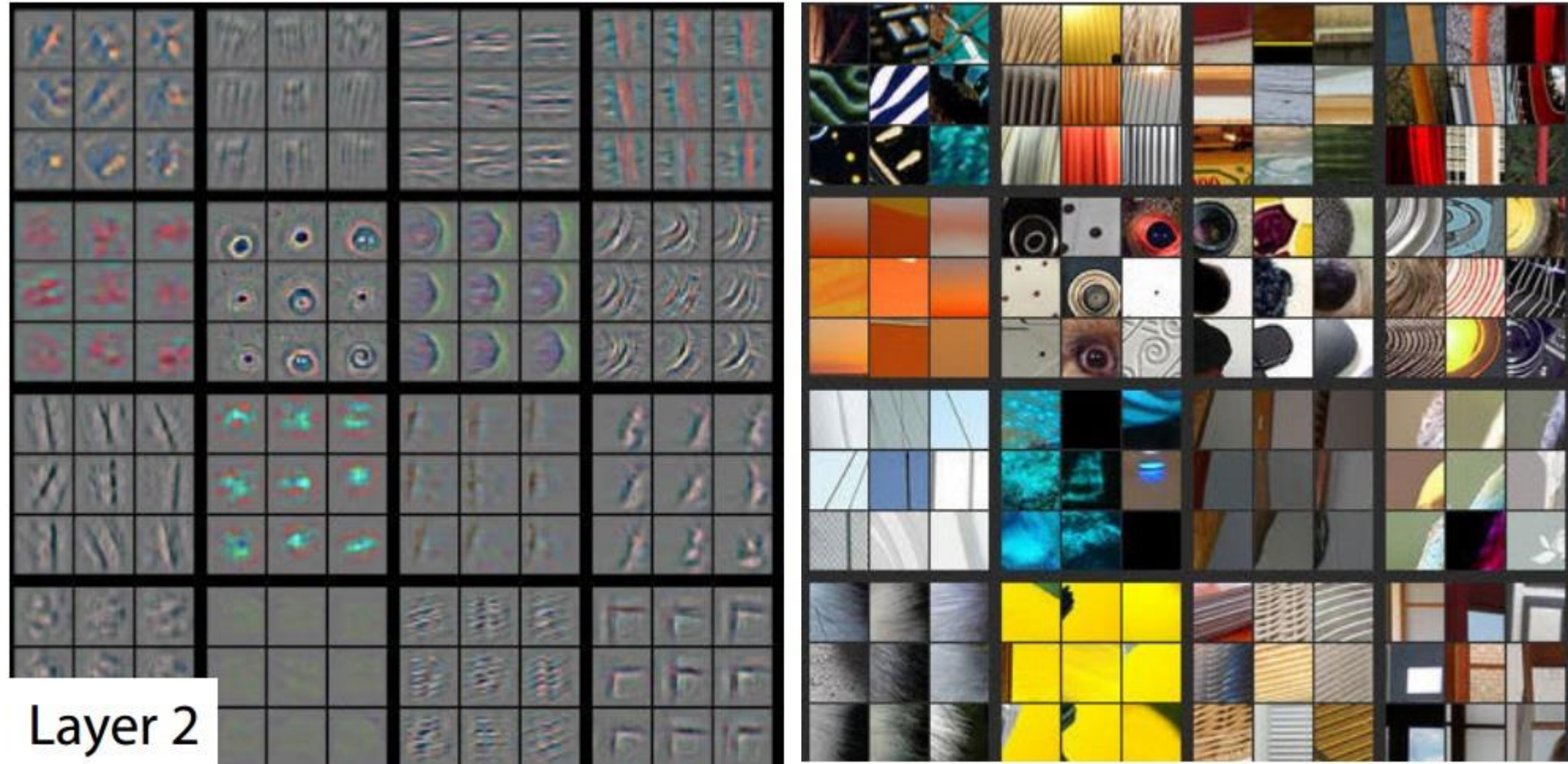
LAYER 1



Layer 1



LAYER 2

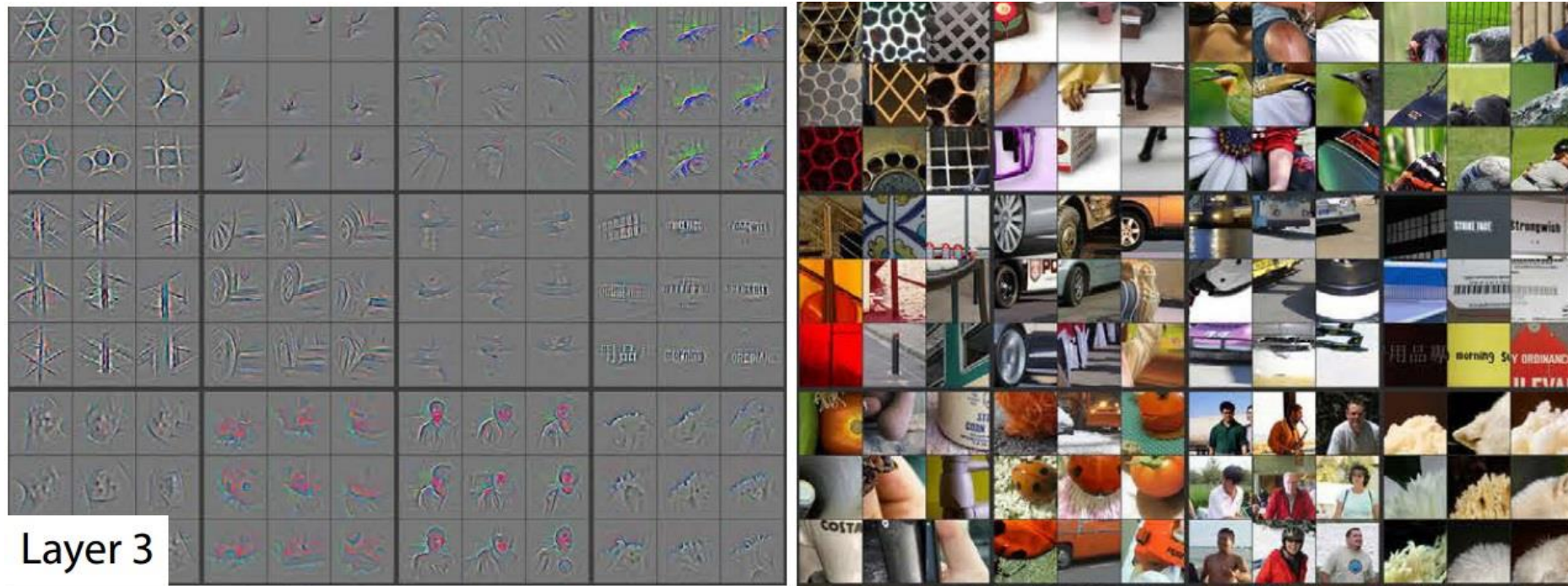


Layer 2

Slide credit: Jia-Bin Huang

Visualizing and Understanding Convolutional Networks [[Zeiler and Fergus, ECCV 2014](#)]

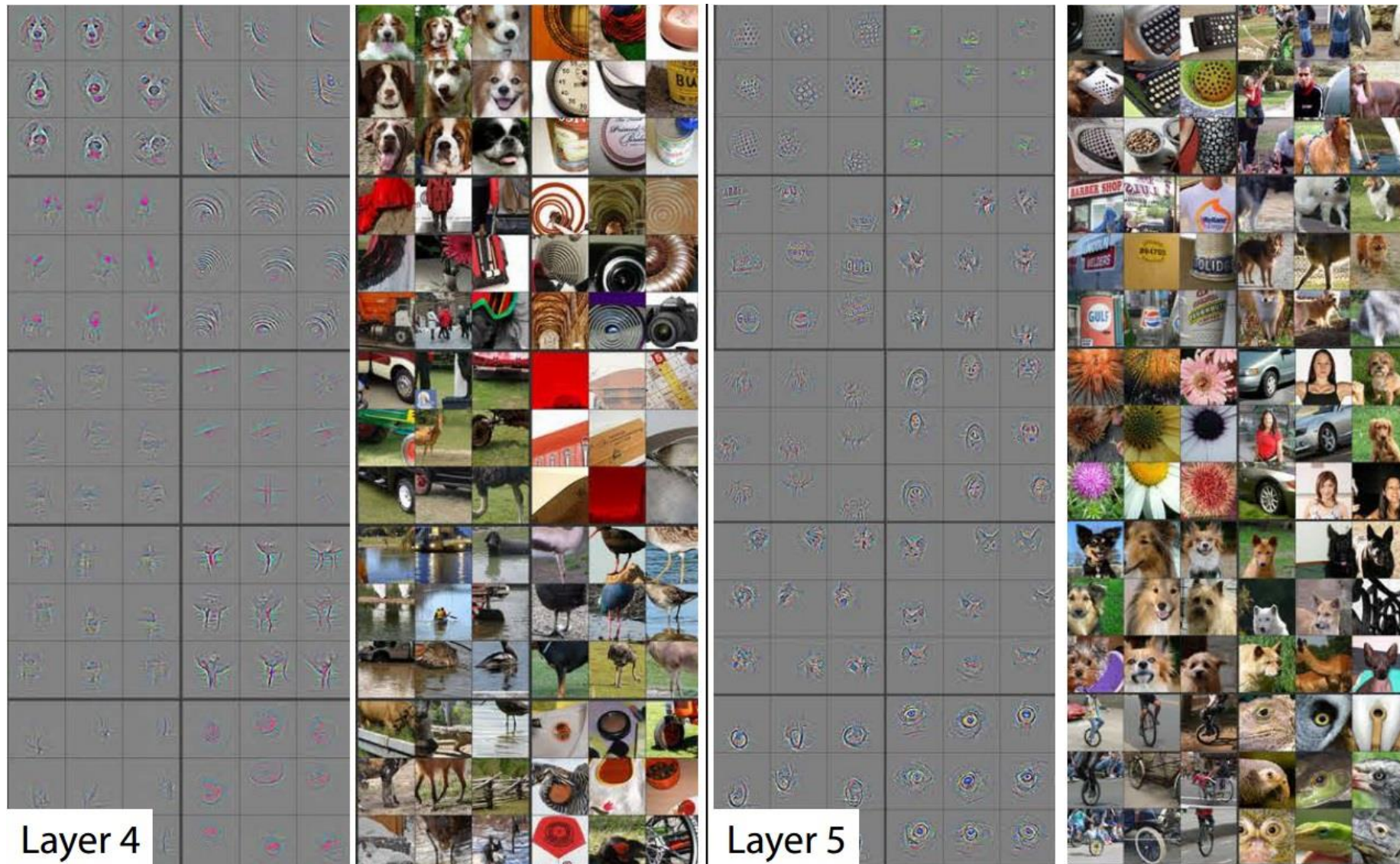
LAYER 3



Slide credit: Jia-Bin Huang

Visualizing and Understanding Convolutional Networks [[Zeiler and Fergus, ECCV 2014](#)]

LAYER 4 AND 5



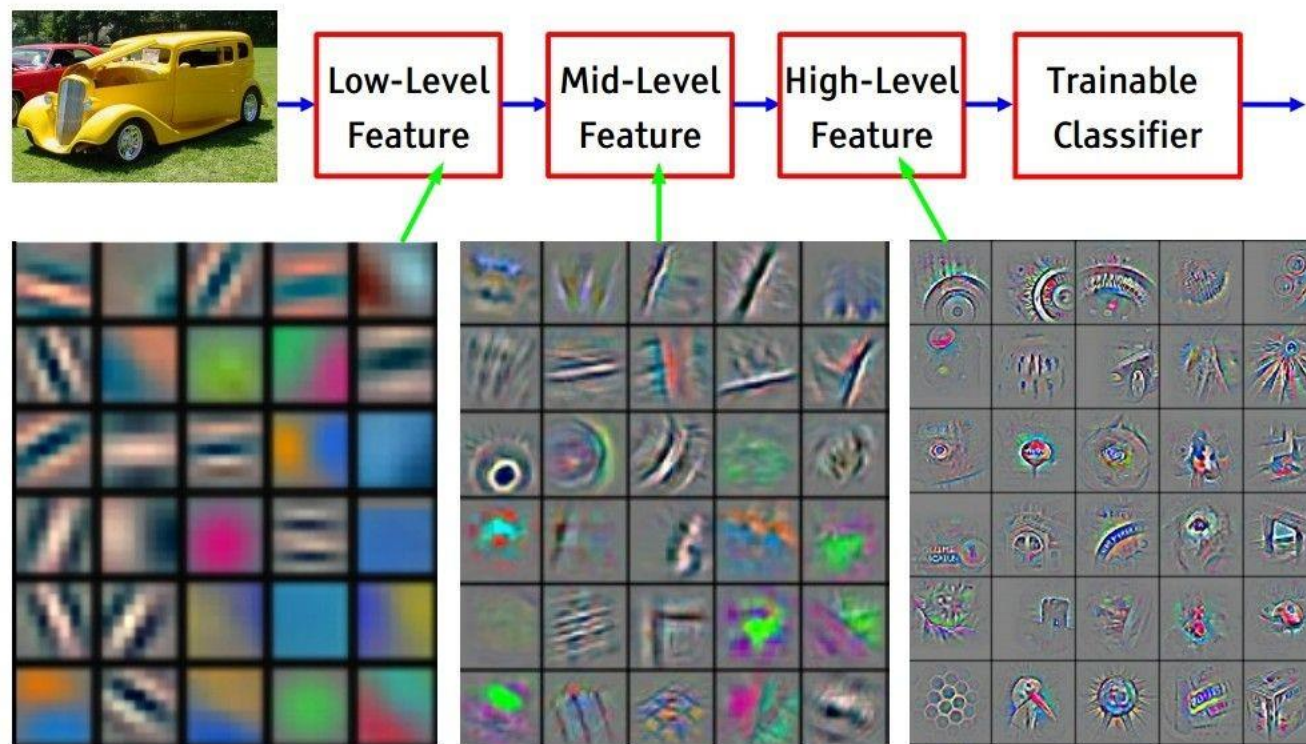
Layer 4

Layer 5

Slide credit: Jia-Bin Huang

Visualizing and Understanding Convolutional Networks [[Zeiler and Fergus, ECCV 2014](#)]

VISUALIZING THE FILTERS CNNs LEARN



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

CASE STUDY: RESNET

[He et al., 2016]

Since deeper usually means better, why not just keep adding more and more layers?



Q: Look at the train and test curves above. What is atypical about these learning curves?

CASE STUDY: RESNET

[He et al., 2016]

Since deeper usually means better, why not just keep adding more and more layers?



56-layer model performs worse on both training and test error
-> The deeper model performs worse, but it's not caused by overfitting!

CASE STUDY: RESNET

[He et al., 2016]

Hypothesis: **Deeper models are harder to optimize**

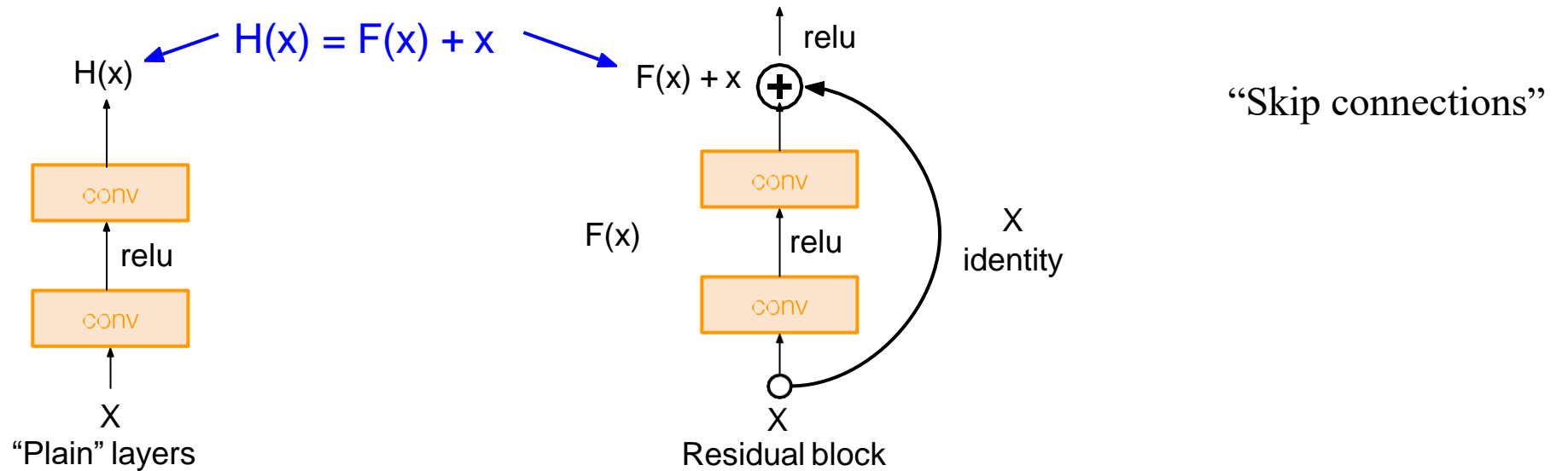
Theoretically, the deeper model should perform as well or better than a shallower model – but in practice this doesn't happen.

Why?

CASE STUDY: RESNET

[He et al., 2016]

Solution: Learn a residual function rather than the underlying function directly

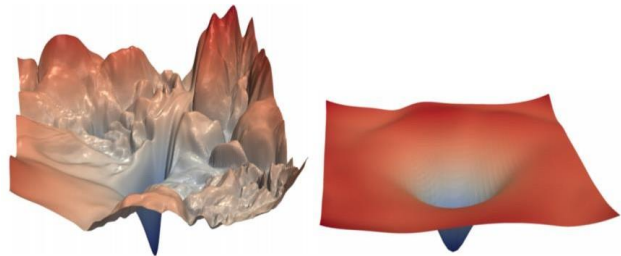


CASE STUDY: RESNET

[He et al., 2016]

Full ResNet architecture:

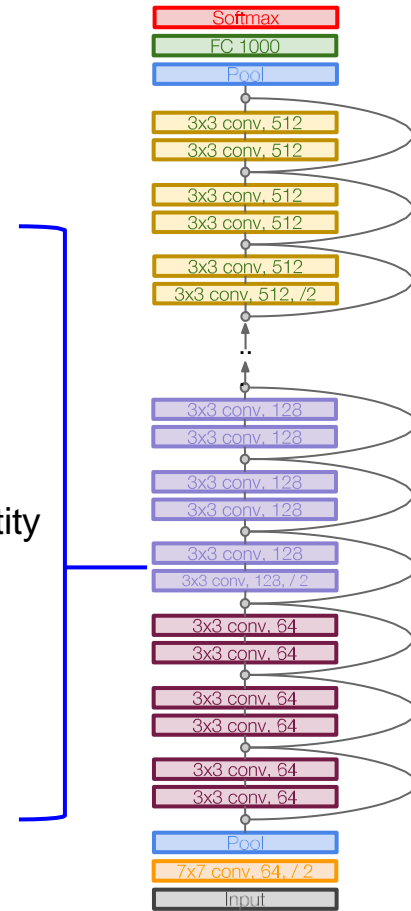
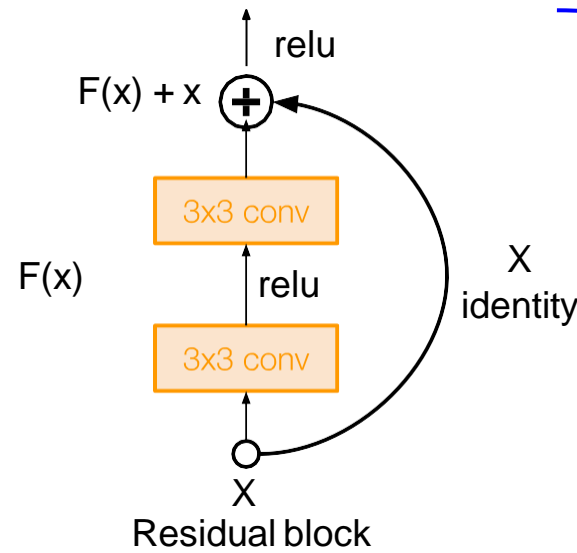
- Stack residual blocks
- Every residual block has two 3x3 conv layers



[no residuals]

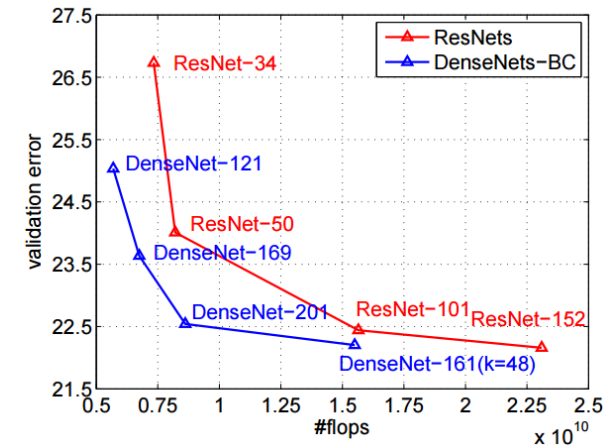
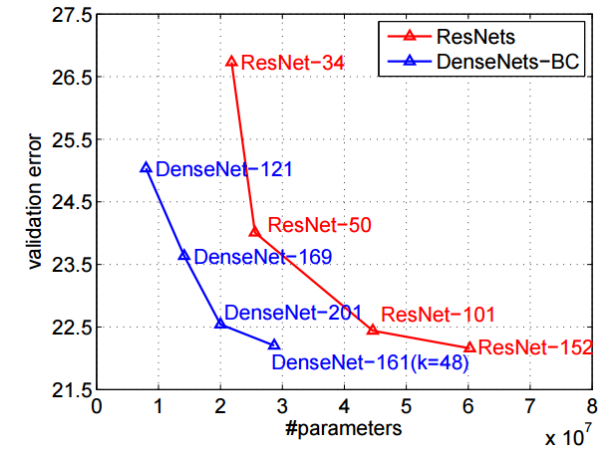
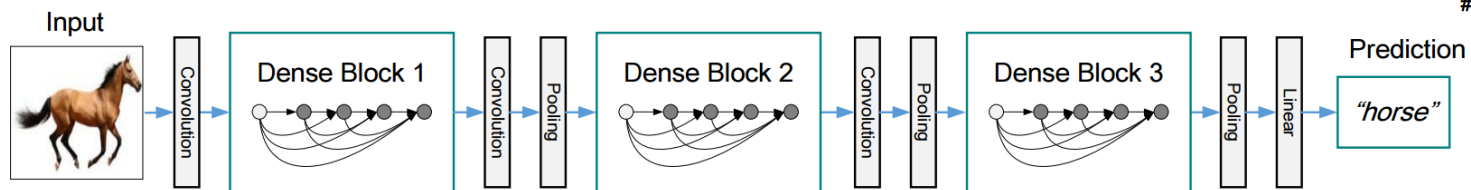
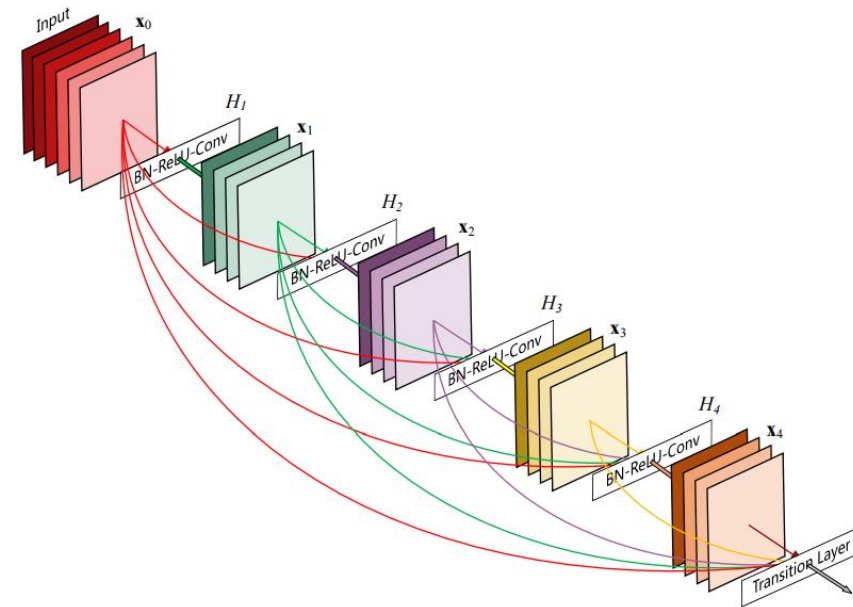
[residuals]

[Loss landscape visualization,
[Li et al., 2018](#), on a ResNet]

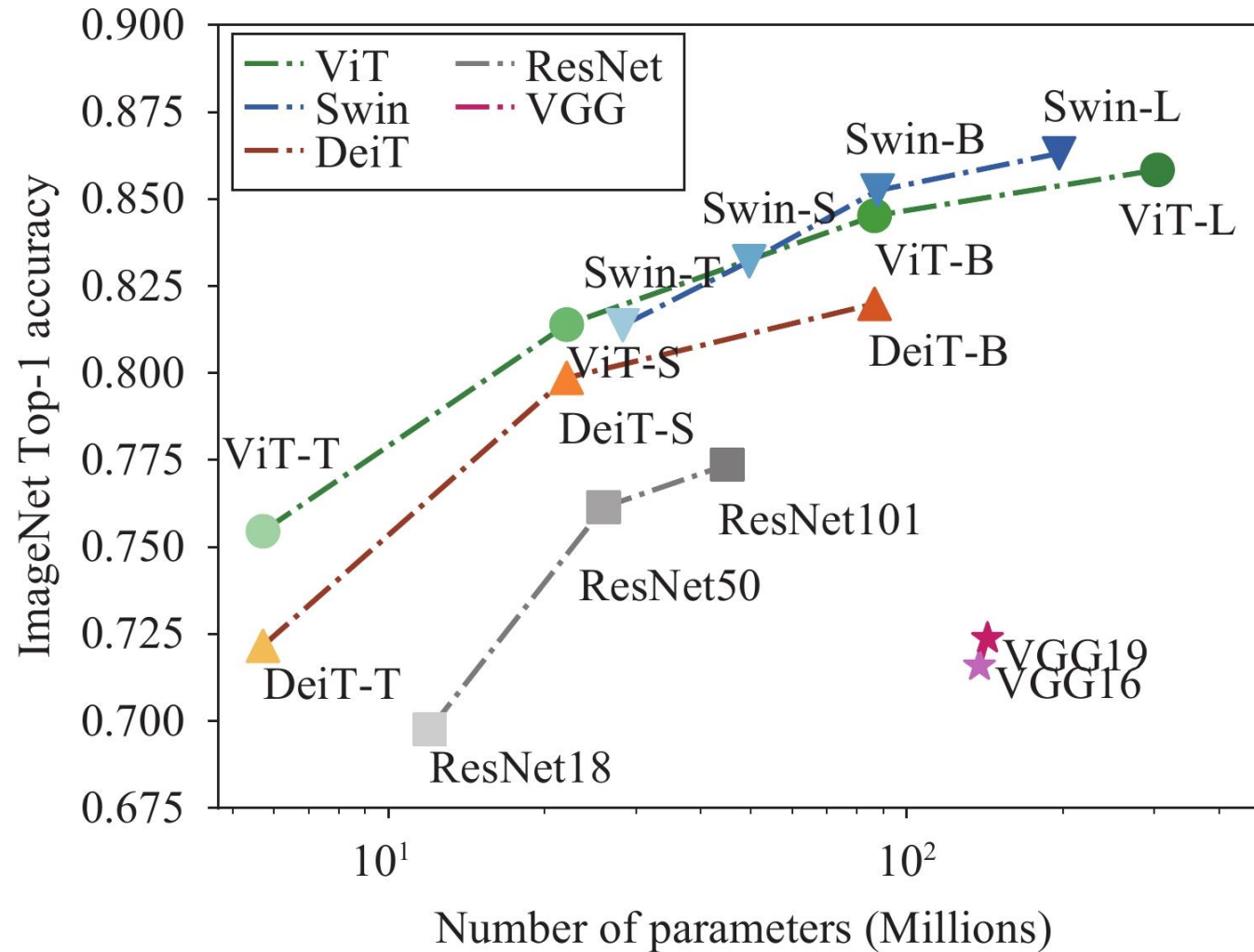


DENSENET

- Shorter connections (like ResNet) help
- Why not just connect them all?



PROGRESS ON IMAGENET

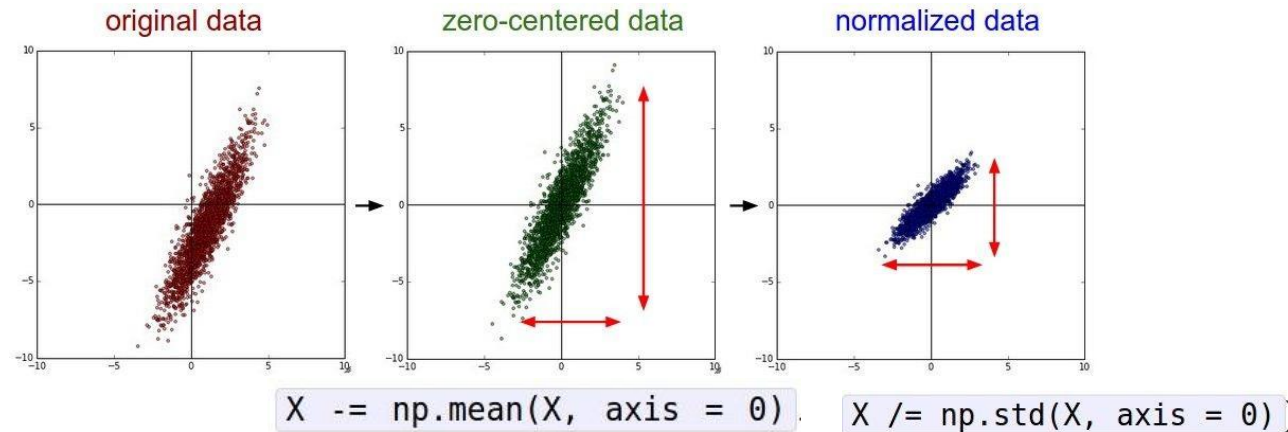


Qiongyi Zhou, Changde Du, Huiguang He. Exploring the Brain-like Properties of Deep Neural Networks: A Neural Encoding Perspective. Machine Intelligence Research, vol. 19, no. 5, pp.439-455, 2022.

Image from Jia-Bin Huang

PRACTICAL TIPS FOR USING CNNs

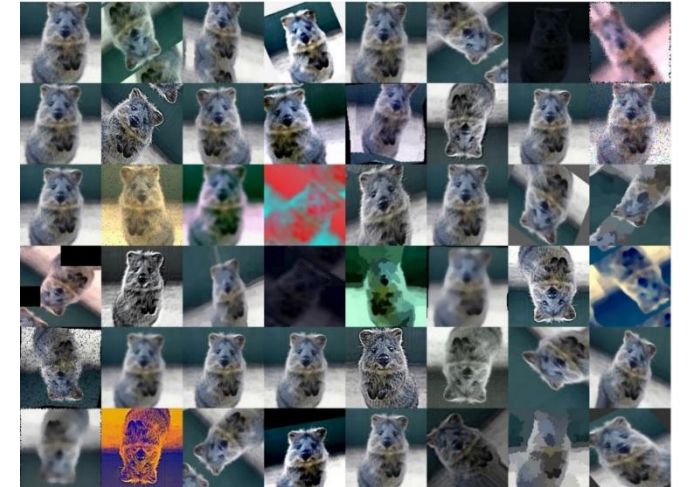
- If you are going to be using a CNN (or later a multimodal transformer), start with weights from a pretrained model (unless you really know what you're doing)
 - Typically ImageNet pretrained
- Usually normalize and resize images before feeding into CNN (and transformers)



- Note: Use the normalization parameters from train set (should come with model)
 - PyTorch may do this for you automatically, but double check to make sure it is happening

MORE PRACTICAL TIPS FOR USING CNNs

- Use minibatching
 - Run multiple images through at once, compute loss and average across all samples in batch
 - Smooths out noise in gradient
- Data augmentation helps prevent overfitting
 - Translation
 - rotation
 - stretching
 - shearing,
 - lens distortions
 - Note: Data augmentations are applied during at training. During test time you may need

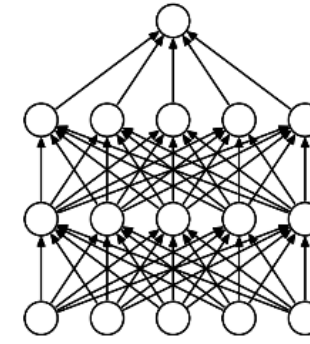


EVEN MORE PRACTICAL TIPS FOR USING CNNs

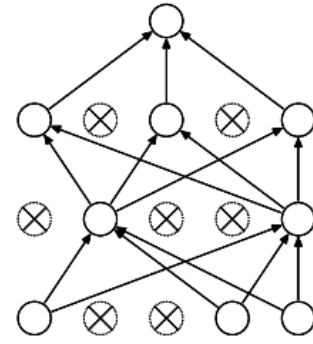
- Initializing the weights for new layers
 - Most of the time you aren't initializing weights and instead initializing from a pretrained model
 - But if you are adding a new layer or classifier, you need to initialize the weights from somewhere
 - What if I just initialize all weights the same as a constant?
 - Usually you initialize the weights randomly
 - In practice: Use Xavier initialization with defaults
 - Other options, but stick with defaults unless you know what you're doing
- Non-linearity activation functions for networks?
 - Stick with RELU or LeakyRELU (unless you know what you're doing)

AND EVEN MORE PRACTICAL TIPS FOR USING CNNs

- Use dropout, especially for fully connected layers
 - Makes neural networks more robust
 - Subsets of neurons must independently be responsible for performance
- Regularization
 - Prevent overfitting by setting a weight decay in optimizer
- Optimizers
 - Can try Adam, SGD with Momentum, RMSProp
 - Learning rate – Try $1e-3$ to $1e-6$
- Batch normalization
 - Recenters / scales intermediate layers
 - Believed to address internal covariate shift



(a) Standard Neural Net



(b) After applying dropout.

COMBINING IMAGE + TEXT (SIMPLE)

- It's common to extract activations from the layers of the network, flatten them, and use them to represent the image
 - These are the image “features”
- Other types of neural networks exist for text, but they operate on the same basic principles
- Using these networks, we can extract features for both images and text and combine them into a *multimodal model*
 - One simple way is just concatenate the features and feed into a NN
- Later, we'll explore more advanced ways of doing this

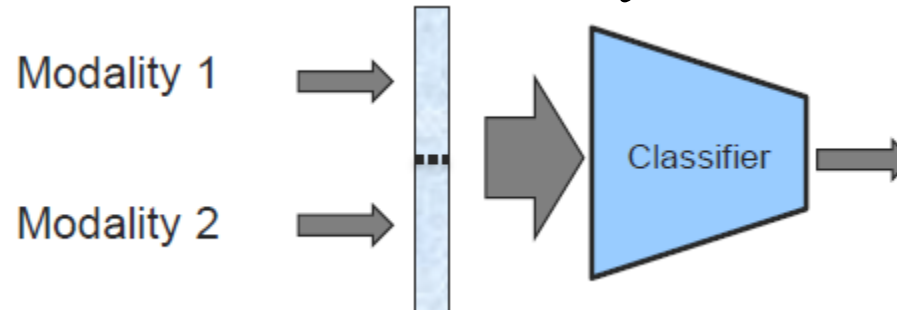


Figure from Tadas Baltrusaitis, CMU