AI and Deep Learning

CNN, Convolutional Neural Network

Jeju National University

Yungcheol Byun

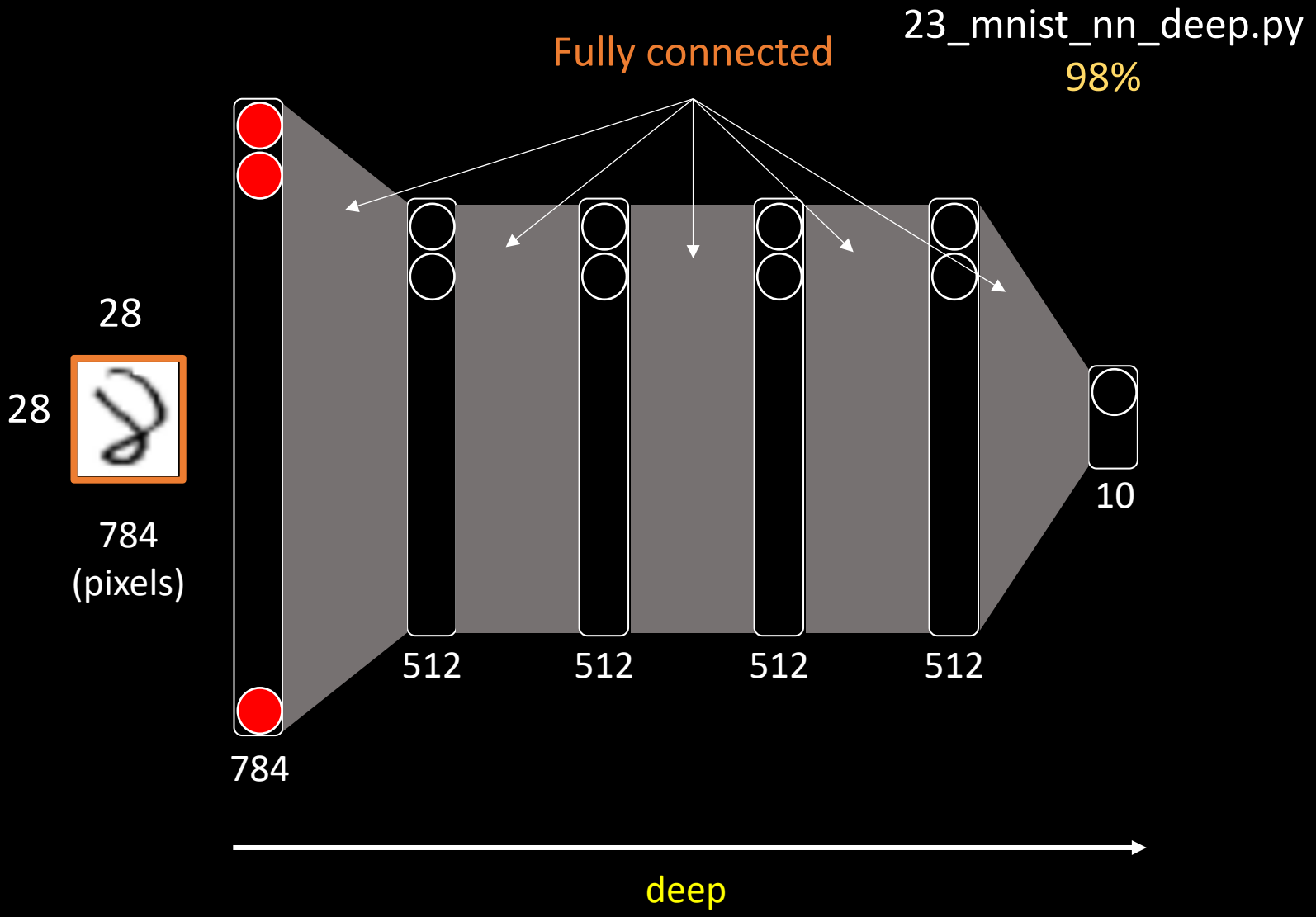
Agenda

- Drawbacks of FCNN
- Feature and Filter
- Activation Maps
- Pooling
- Theoretical Background
- Case Study

Drawbacks of Fully Connected NN

23_mnist_nn_deep.py

98%

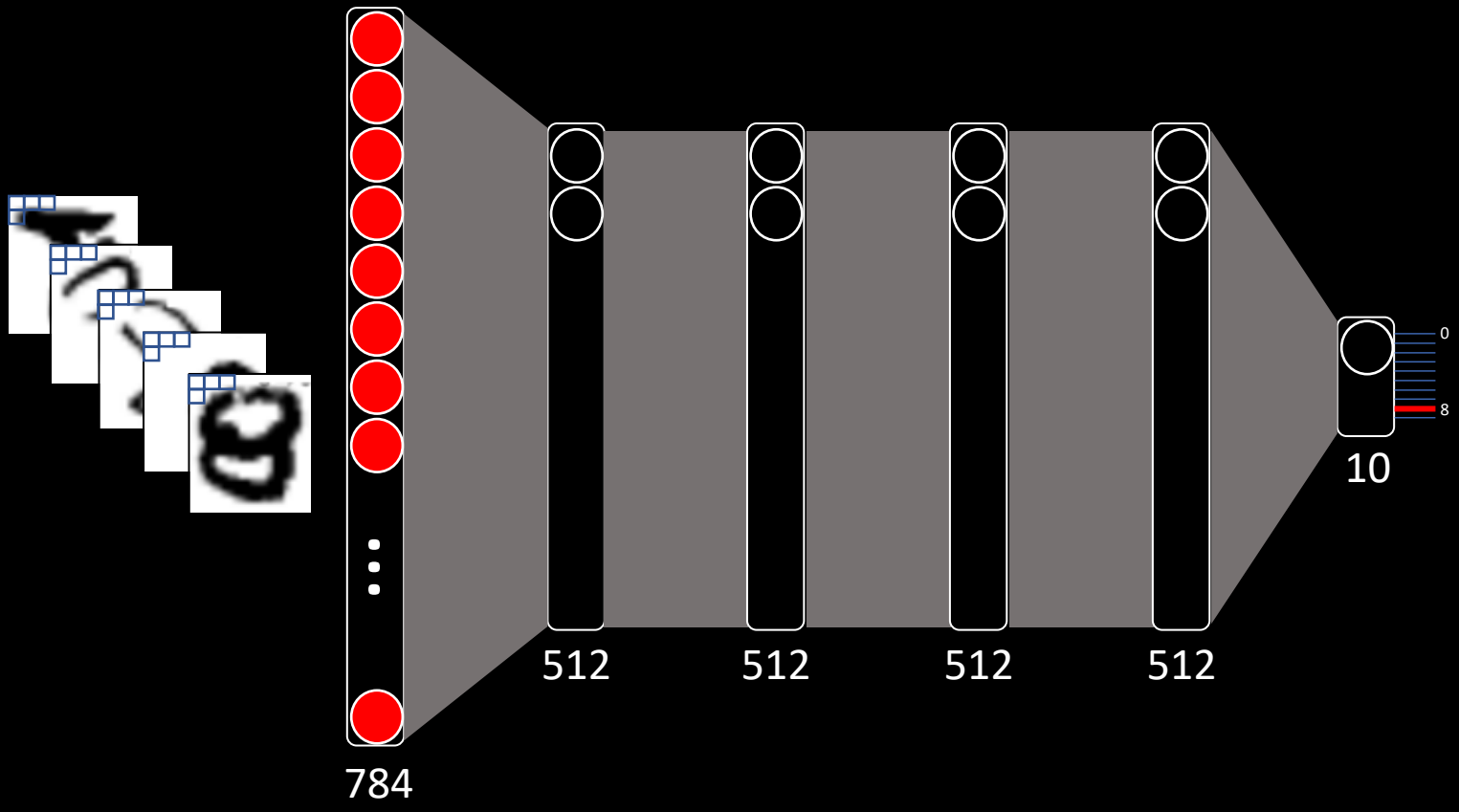


- Various shapes - size, location, skewness, distortion


4 → 9
116
2 → 7
322
5 → 3
341
6 → 0
446
3 → 5
450
8 → 2
583
4 → 9
741
3 → 5
939
8 → 9
948
1 → 3
957
6 → 0
1015
4 → 6
1113
7 → 2
1227
9 → 4
1233
9 → 5
1248
7 → 1
1261
8 → 2
1365
5 → 3
1394
8 → 7
1531
9 → 5
1710
5 → 2
1738
8 → 3
1879
9 → 4
1902
5 → 3
2036
4 → 9
2054
2 → 0
2099
6 → 0
2119
4 → 9
2131
9 → 8
2294
9 → 1
2388
9 → 4
2407
2 → 0
2463
2 → 4
2489
5 → 3
2598
6 → 1
2655
8 → 0
2897
9 → 5
2940
3 → 5
2954
6 → 0
3031
8 → 9
3290
6 → 0
3423
6 → 4
3521
5 → 0
3559
8 → 9
3728
4 → 6
3781
7 → 8
3808
1 → 3
3907
7 → 1
4028
9 → 8
4079
1 → 7
4202
2 → 7
4206
9 → 7
4225
7 → 3
4239
2 → 8
4249
9 → 5
4285
3 → 2
4444
6 → 8
4572
3 → 5
4741
9 → 8
4762
8 → 0
4808
6 → 0
4815
8 → 6
4880
8 → 2
5750
5 → 3
5938
3 → 8
5956
3 → 0
6012
8 → 9
6556
1 → 8
6573
7 → 1
6577
8 → 3
6626
2 → 1
8060
8 → 5
8409
7 → 2
9010
7 → 2
9016
6 → 4
9680
9 → 7
9693
5 → 6
9730
4 → 9
9793
2 → 7
9840

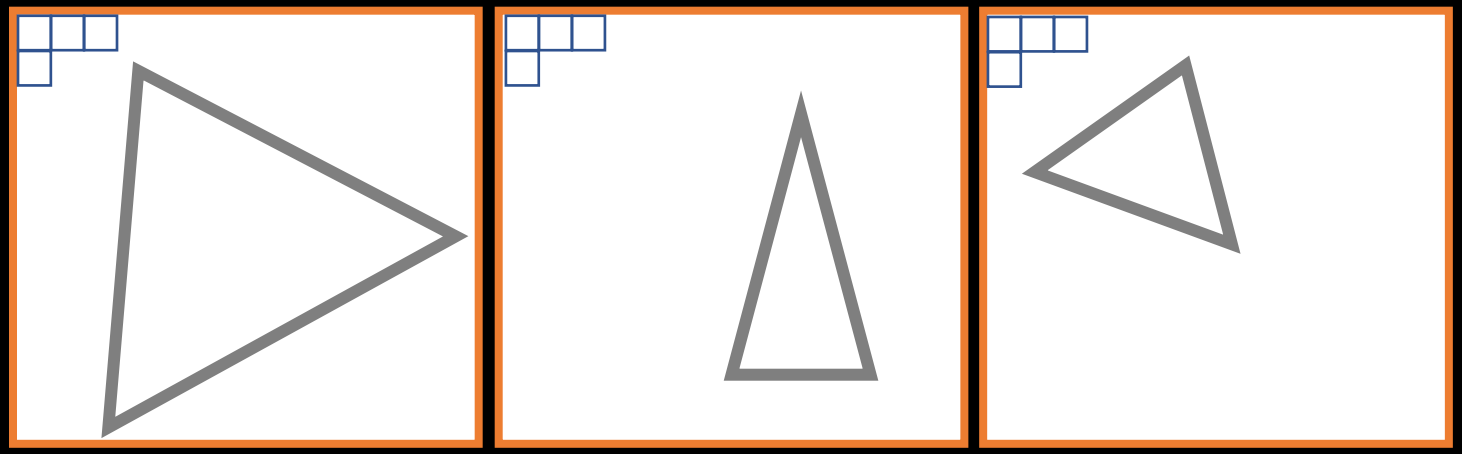
Many variations of eight(8)
from writing styles

Same digits but different sets of values as inputs

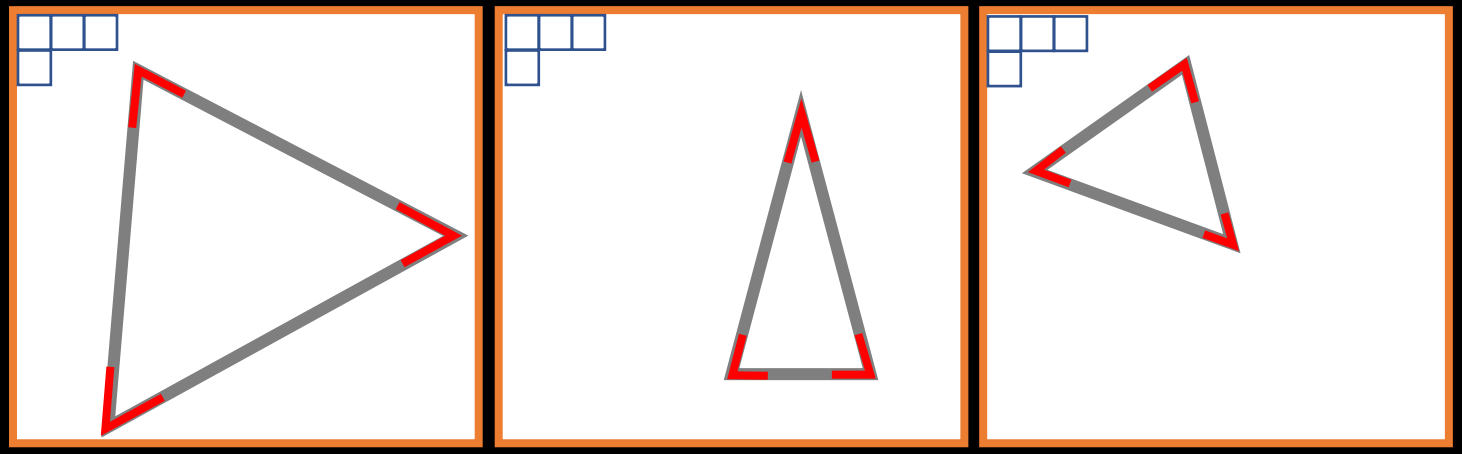


Any common features?

Any common features?



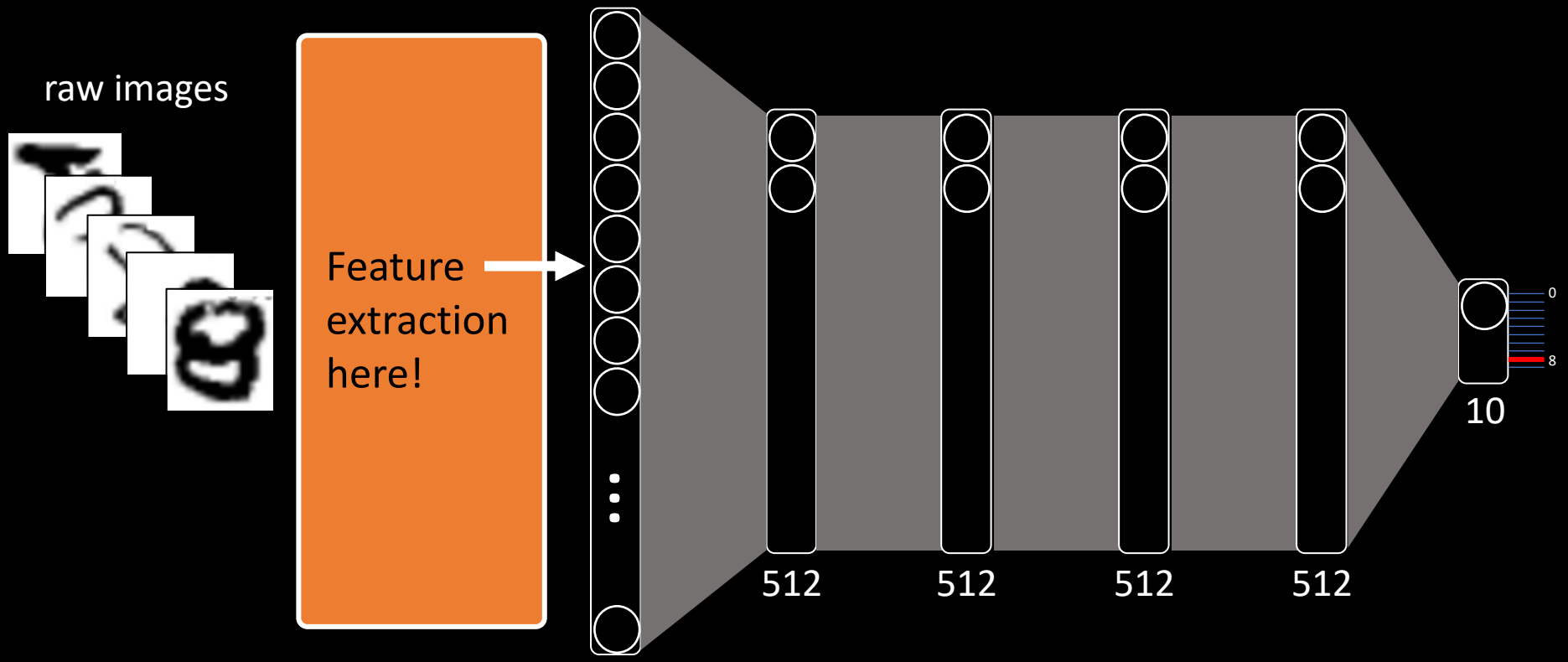
Any common features?



Redundant information

Any common features?

Feature extraction for the FCNN

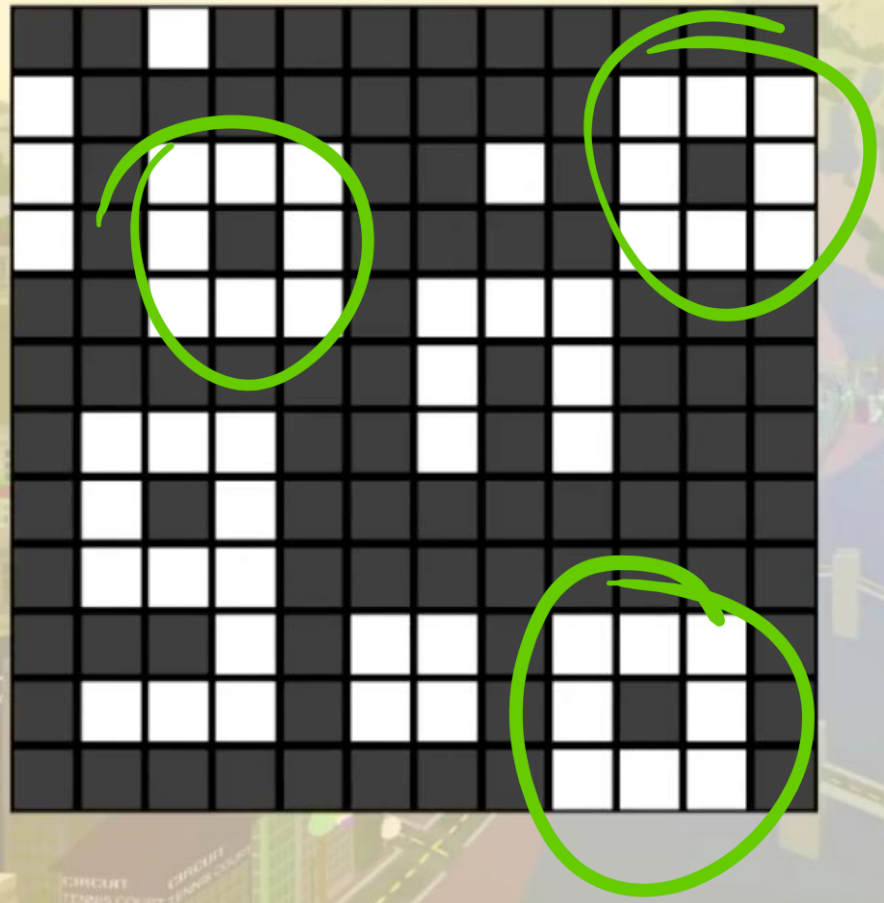


Instead of using the raw image as input,
we extract features first and use them as input for the FCNN.

How to extract features

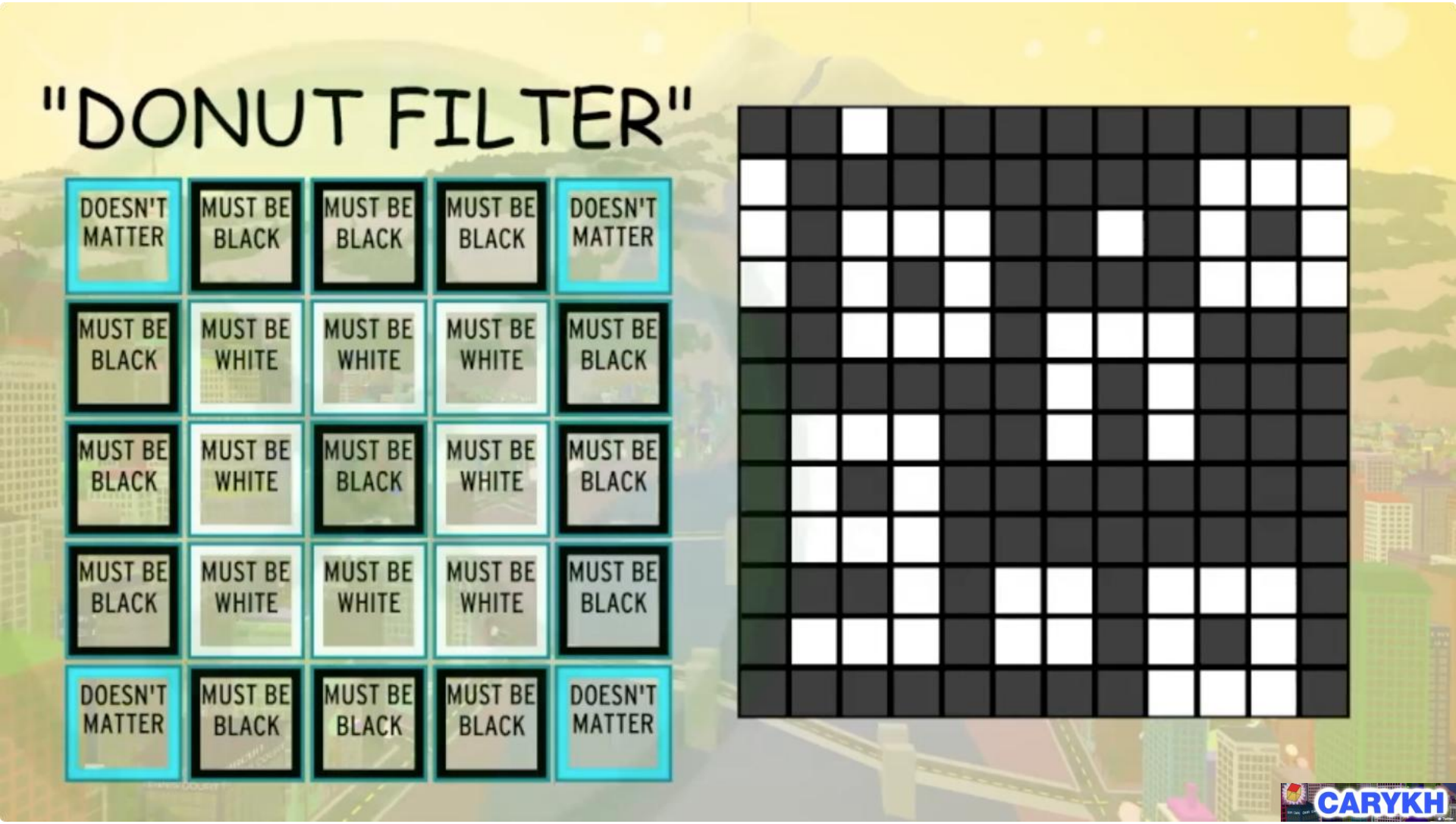
Doughnut features

도넛 특징을 찾아라!



IMAGE

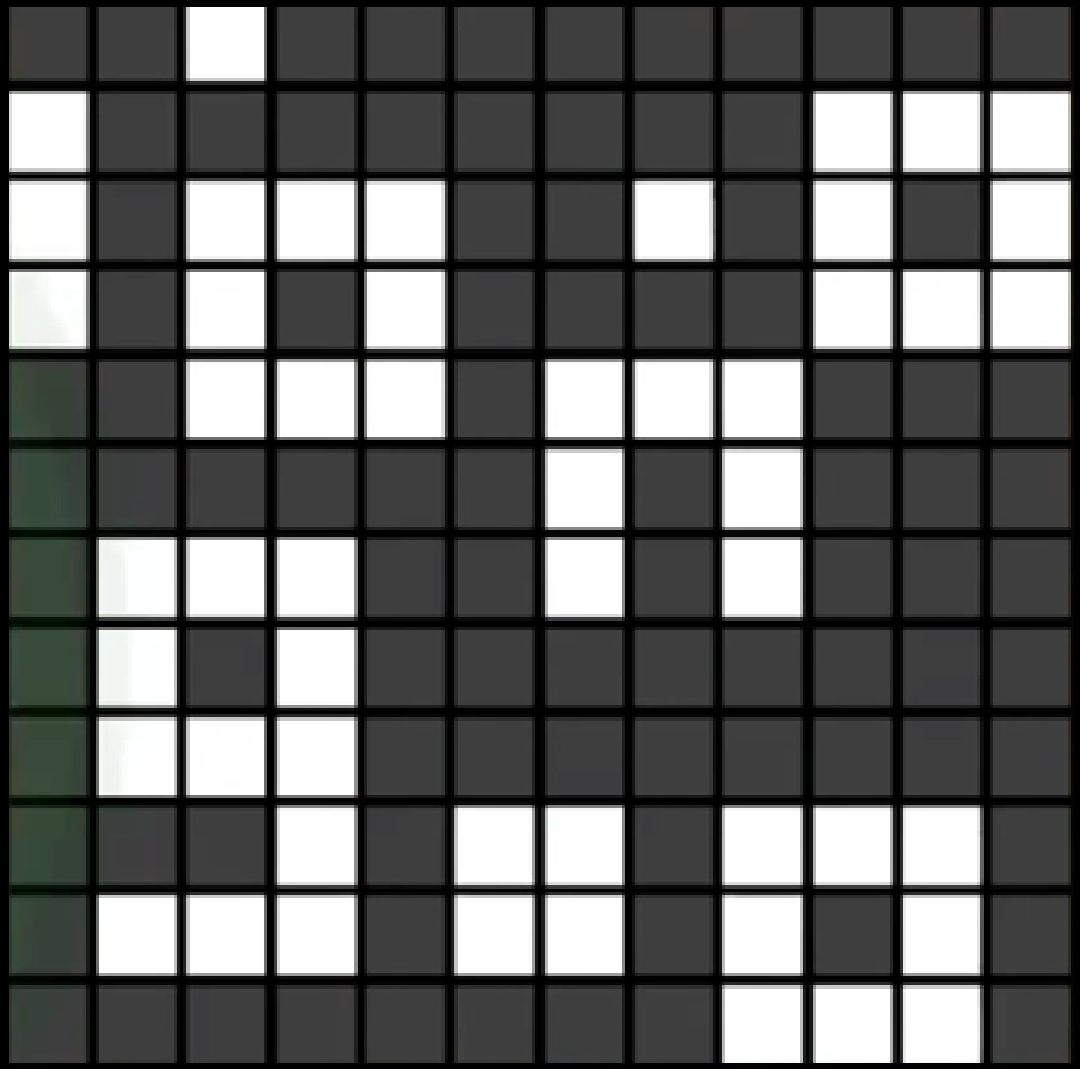
Filter to extract the doughnut feature



How to extract doughnut features

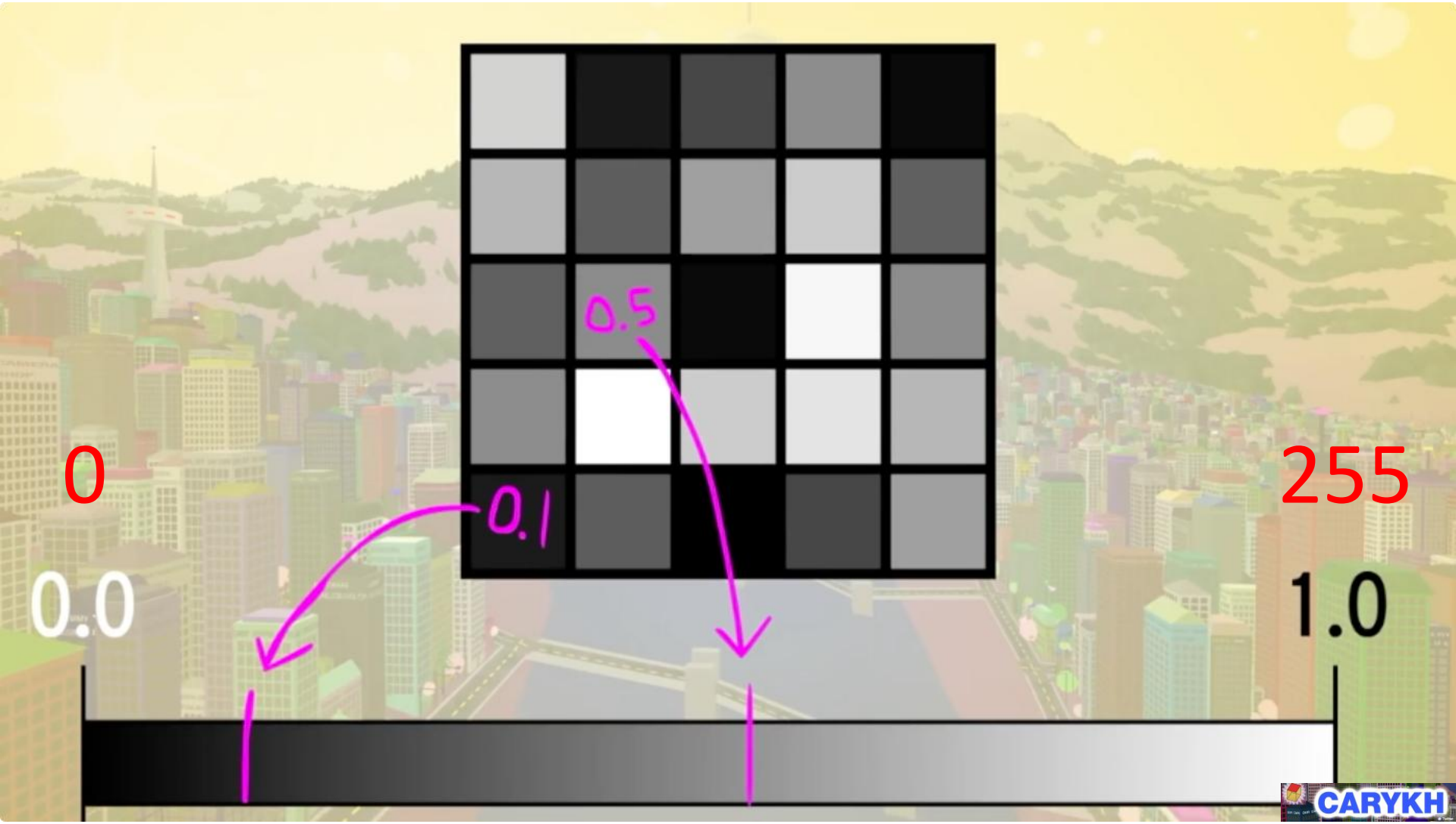
Out-of-bounds pixels are considered as black pixels.

Black & White



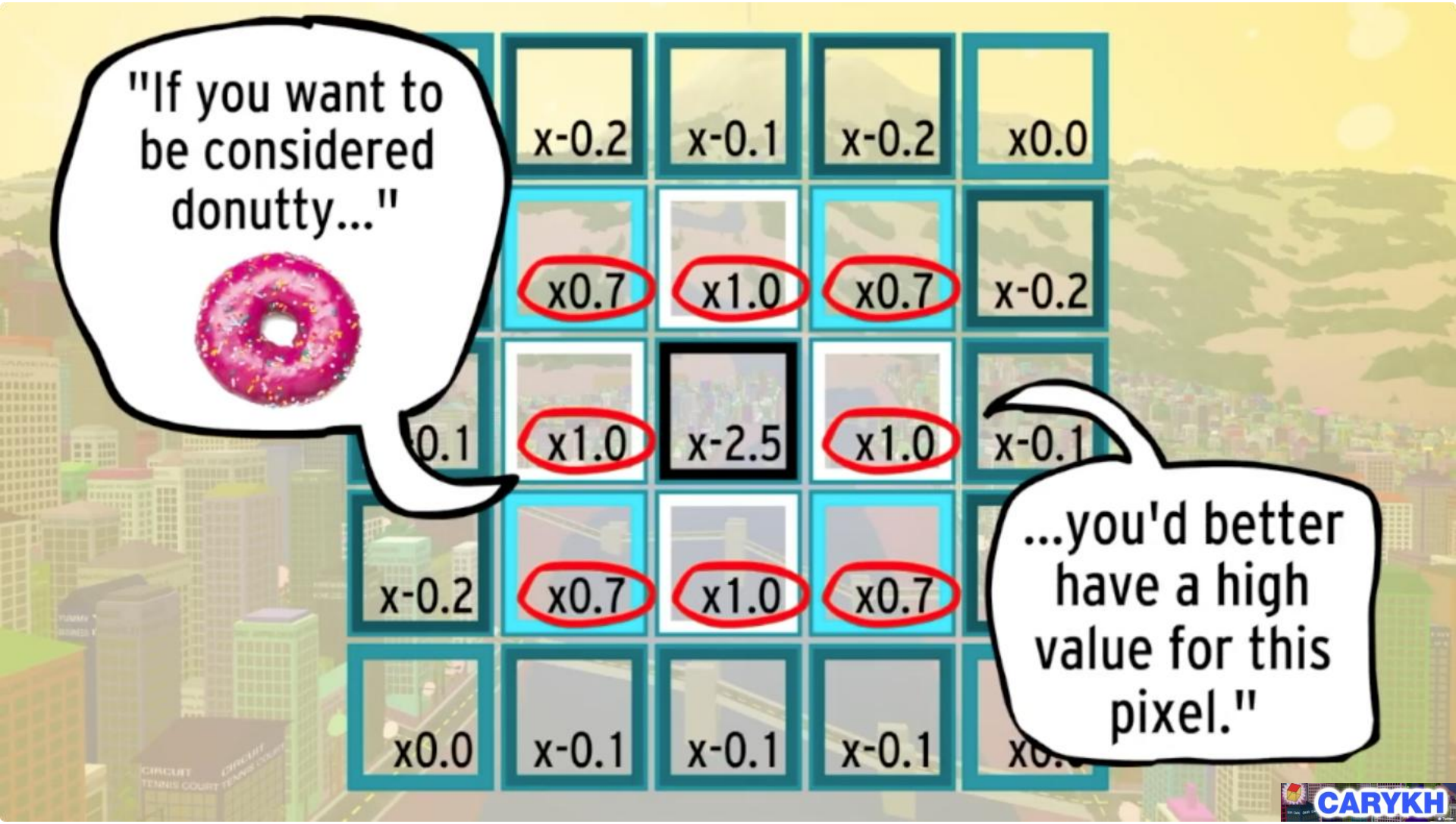
Real images are color or gray

An example of a gray image

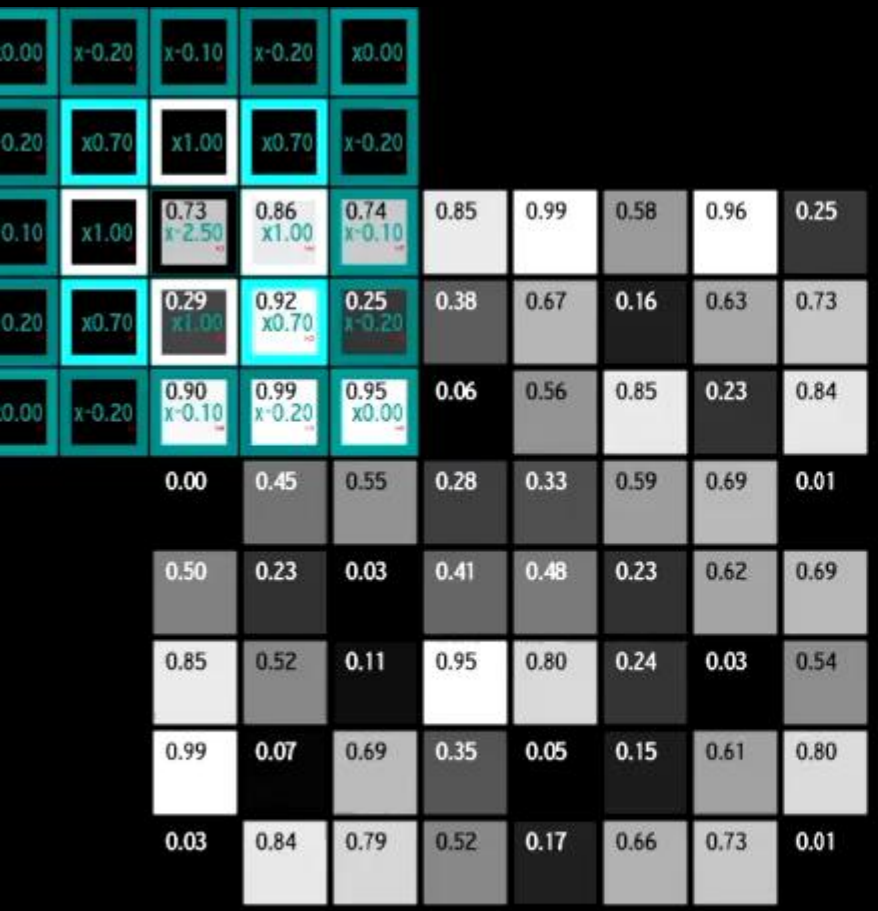



DONUT FILTER
DOESN'T MATTER
MUST BE BLACK
MUST BE WHITE
New improved filter
x0.0
x-0.2
x-0.1
x0.7
x1.0
x-2.5
"set of multipliers"

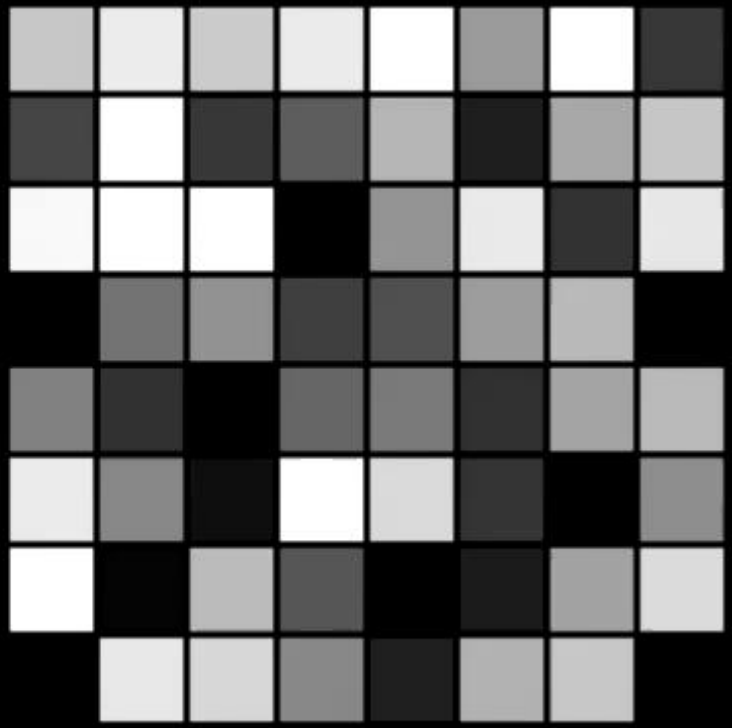
New improved filter



Input Image (입력 이미지)



How donutty is each pixel?

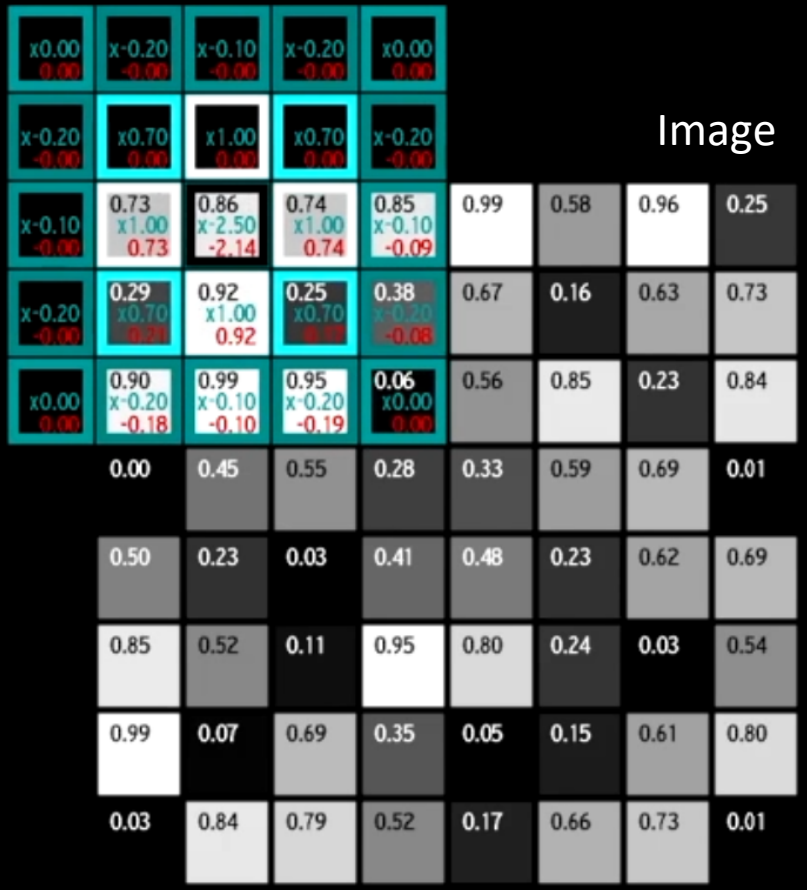


Convolution?

dot product with the image

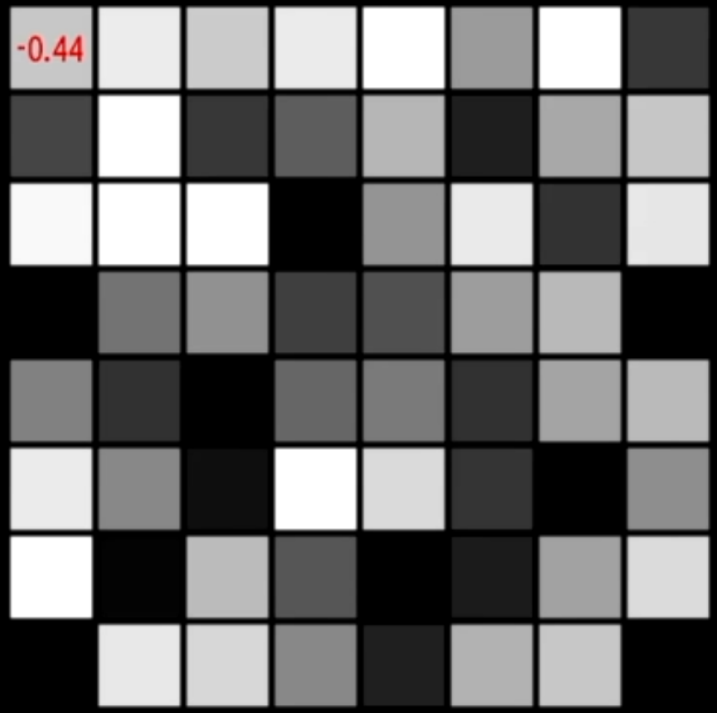
and slide(move) the filter spatially to the next right pixel.

Filter

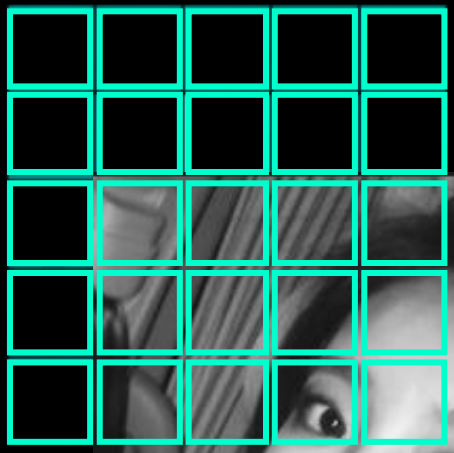


Image

How donutty is each pixel?

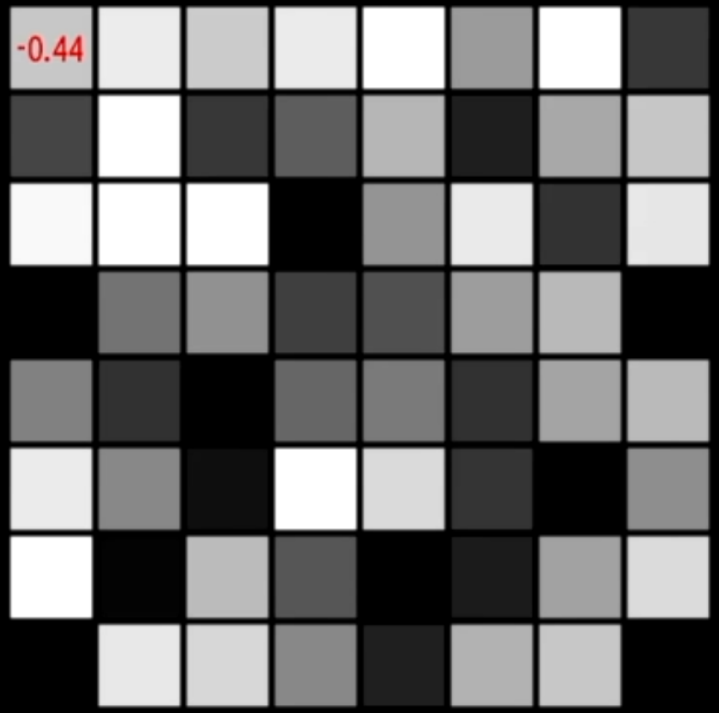


Filter

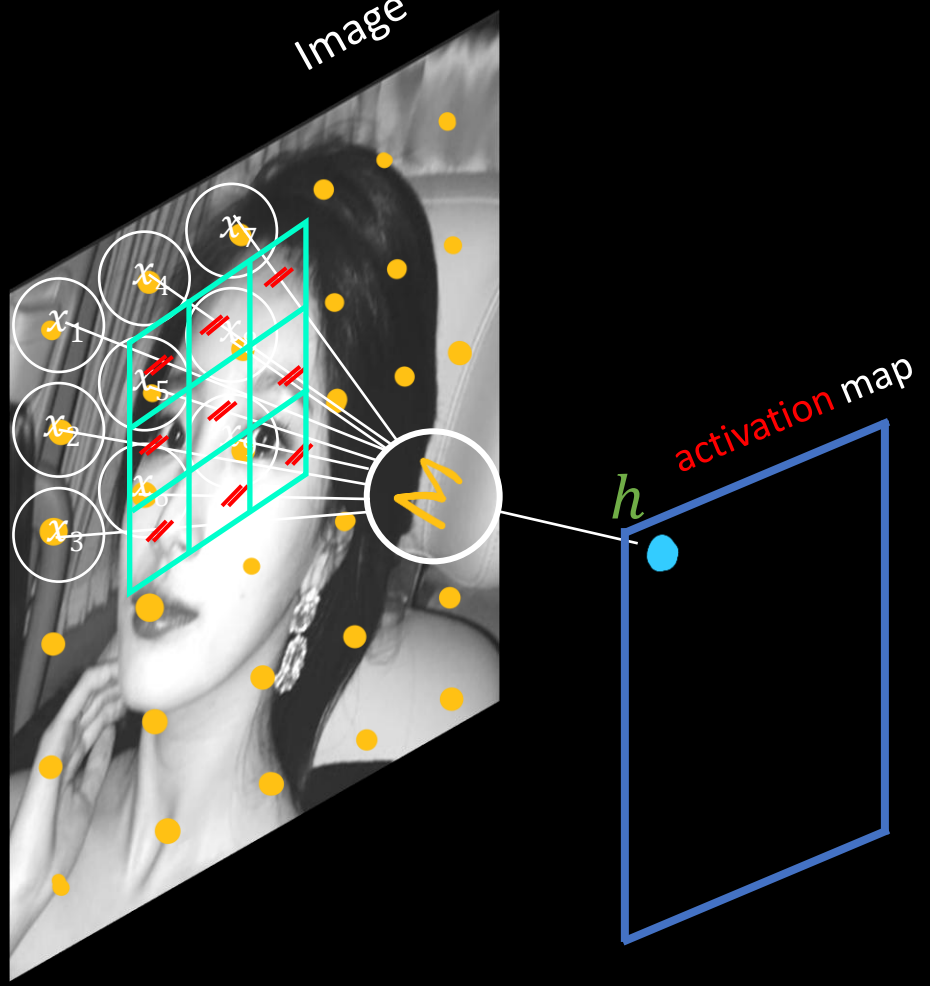


Image

activation map



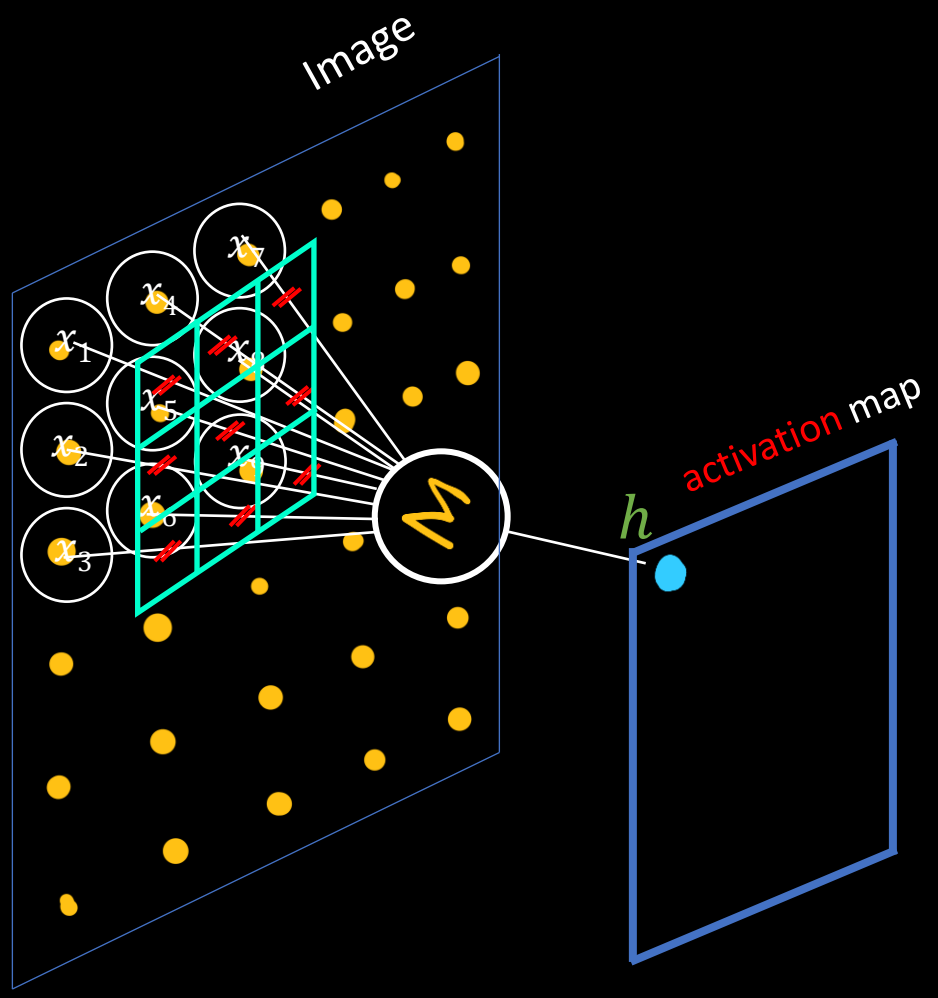
Convolution in 3D view



9 connections = 9 synaps (parameters)

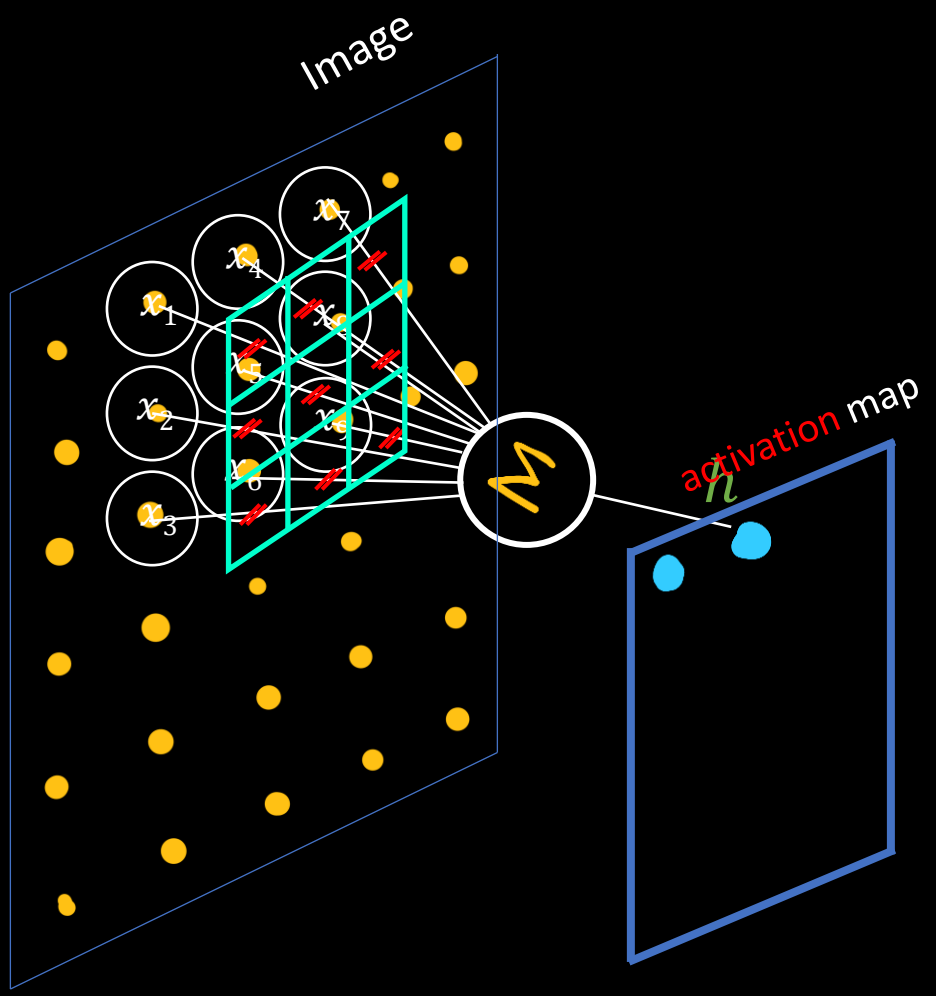
So, what is a filter?

Convolution in 3D view



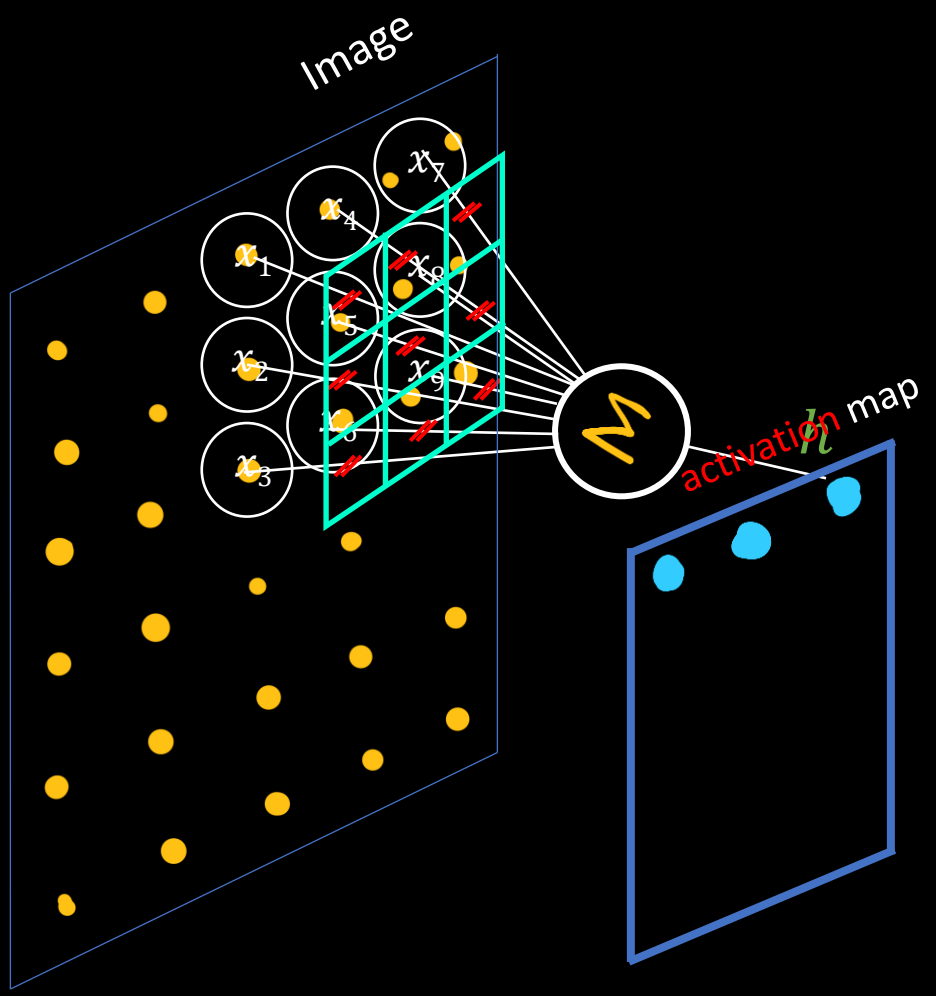
9 connections = 9 synaps (parameters)

Convolution in 3D view



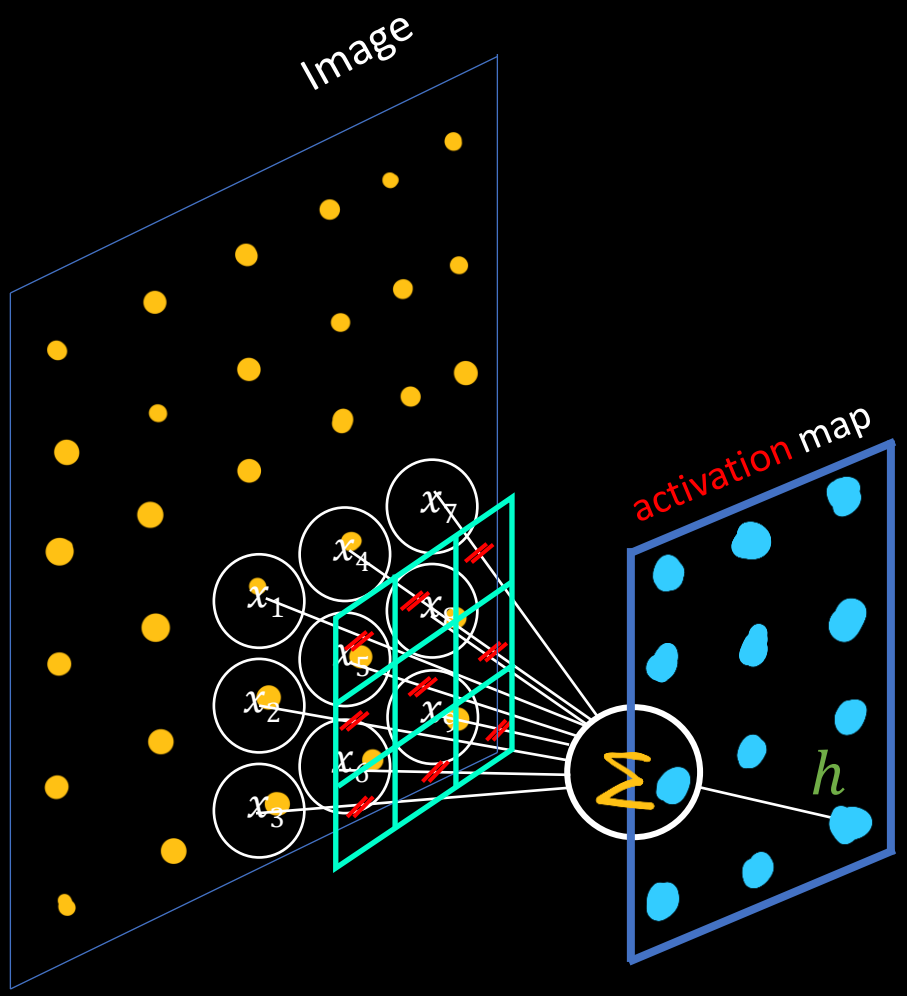
9 connections = 9 synaps (parameters)

Convolution in 3D view



9 connections = 9 synaps (parameters)

Convolution in 3D view

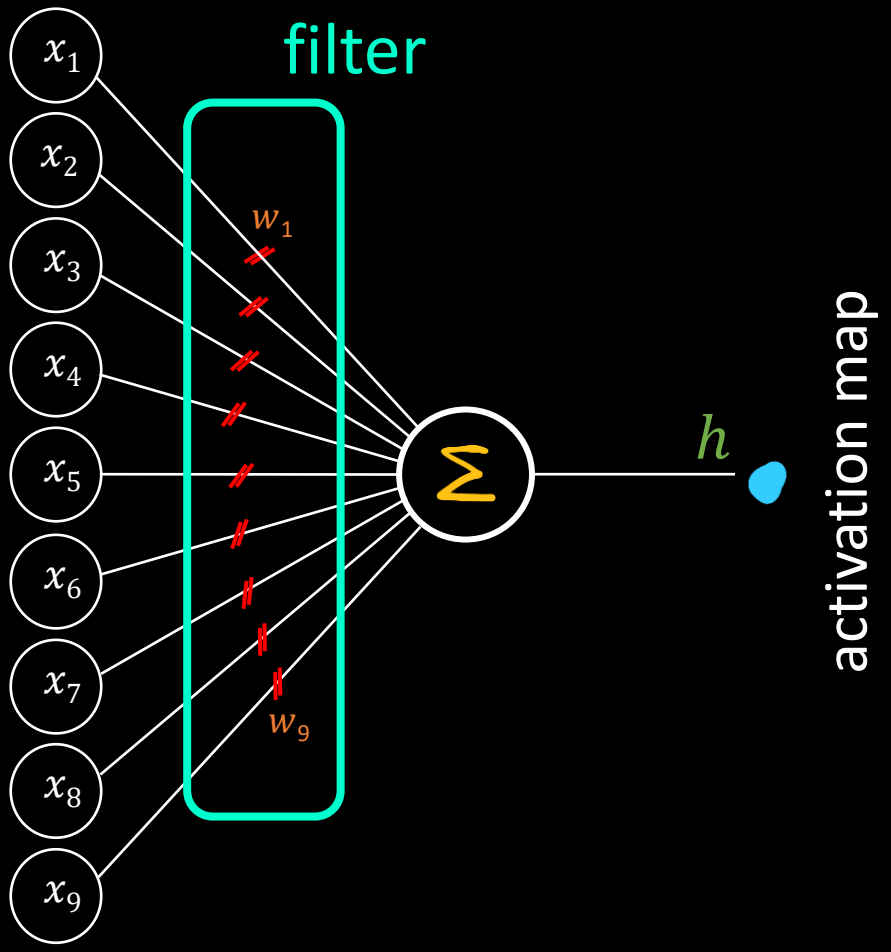


9 connections = 9 synaps (parameters)

A filter

- a group of synapses($\boldsymbol{w}$)
- parameter tuning with back-propagation $\rightarrow$ automatic filter setting
- sharing the same synapses while **covolution**(dot-product & shifting)
- 1 filter $\rightarrow$ 1 activation map

Filter(2D view)



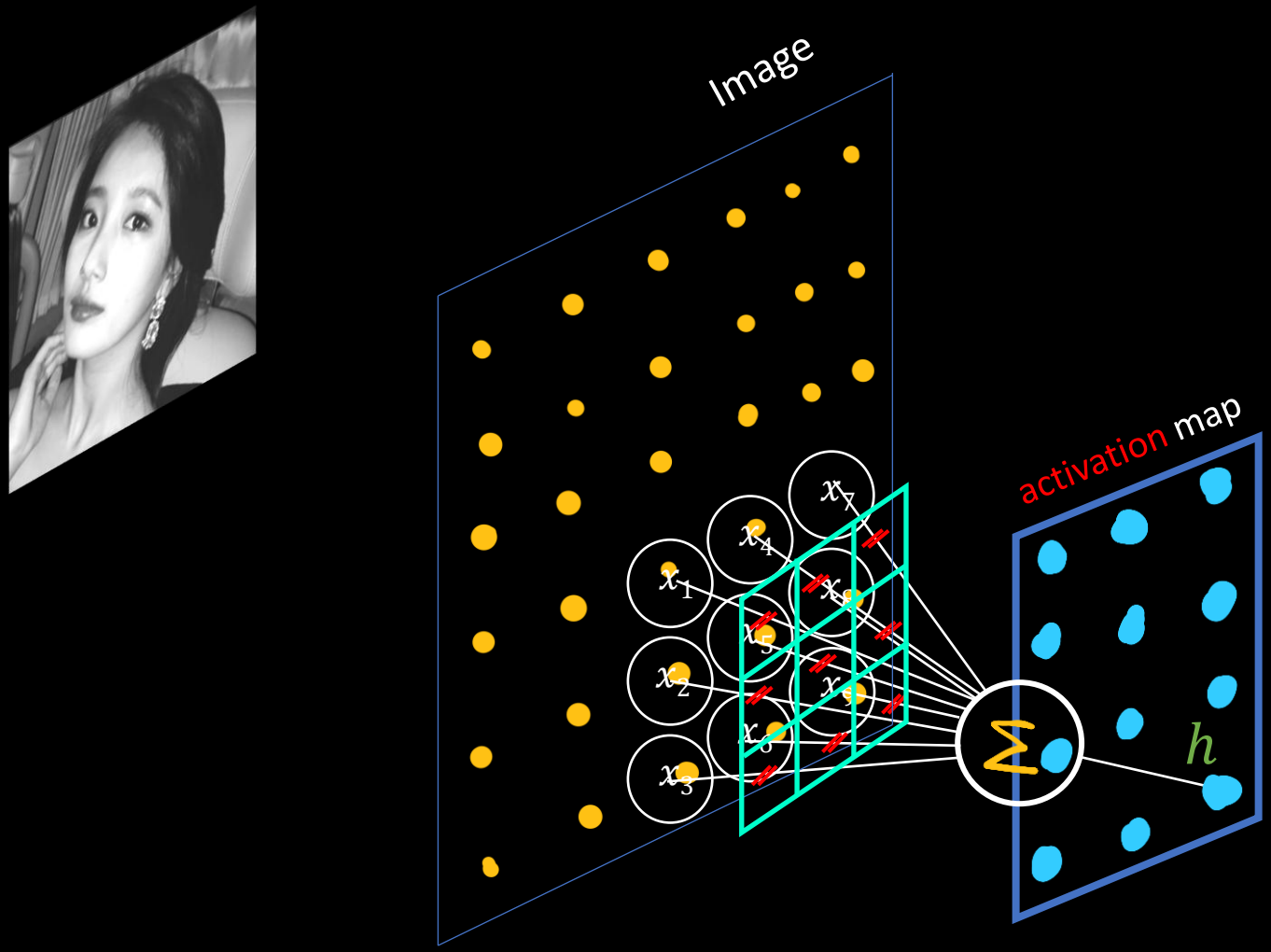


Image
activation map
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
Σ
h

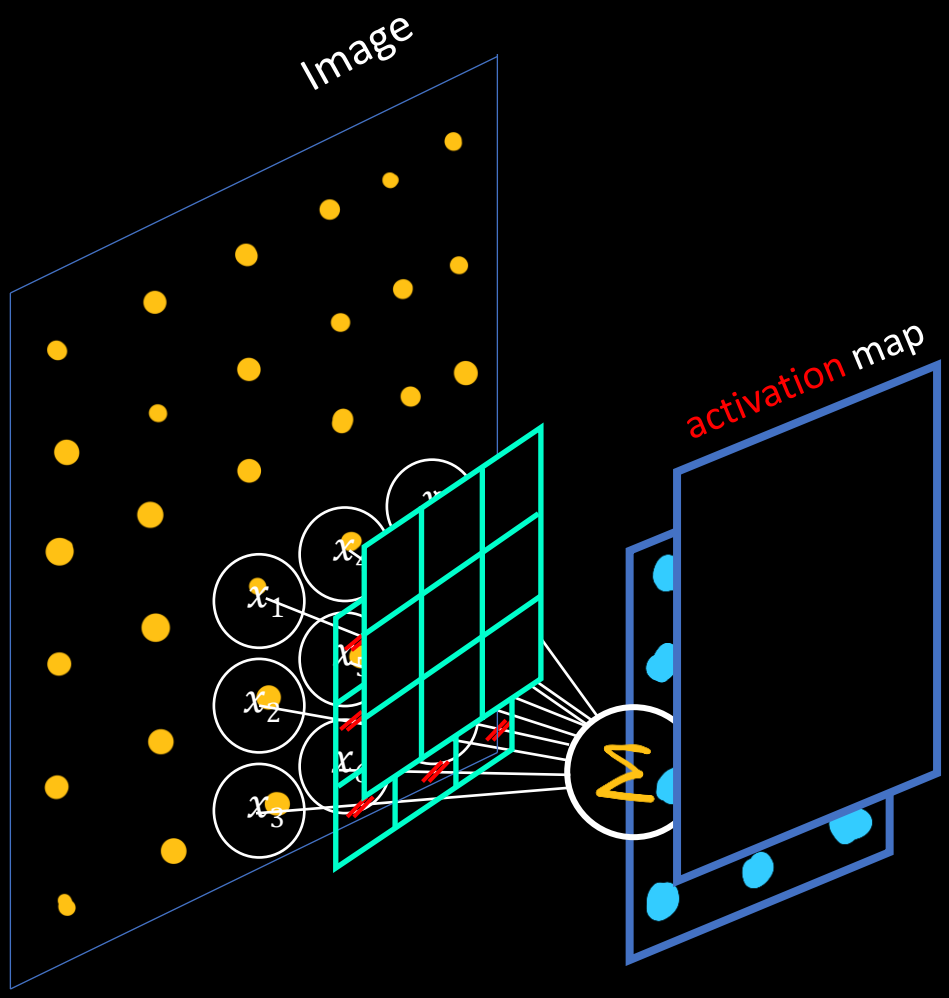


Image
activation map
x_1
x_2
x_3
Σ

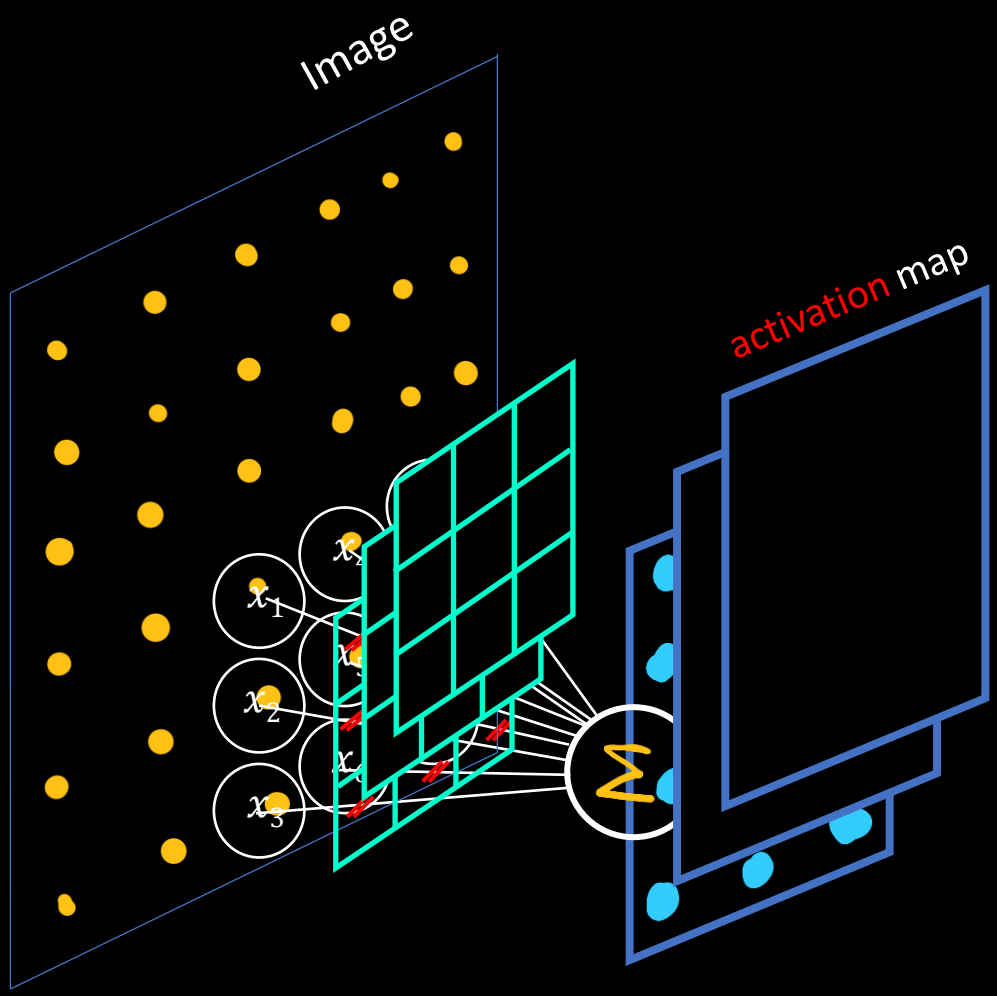


Image
activation map
x_1
x_2
x_3
Σ

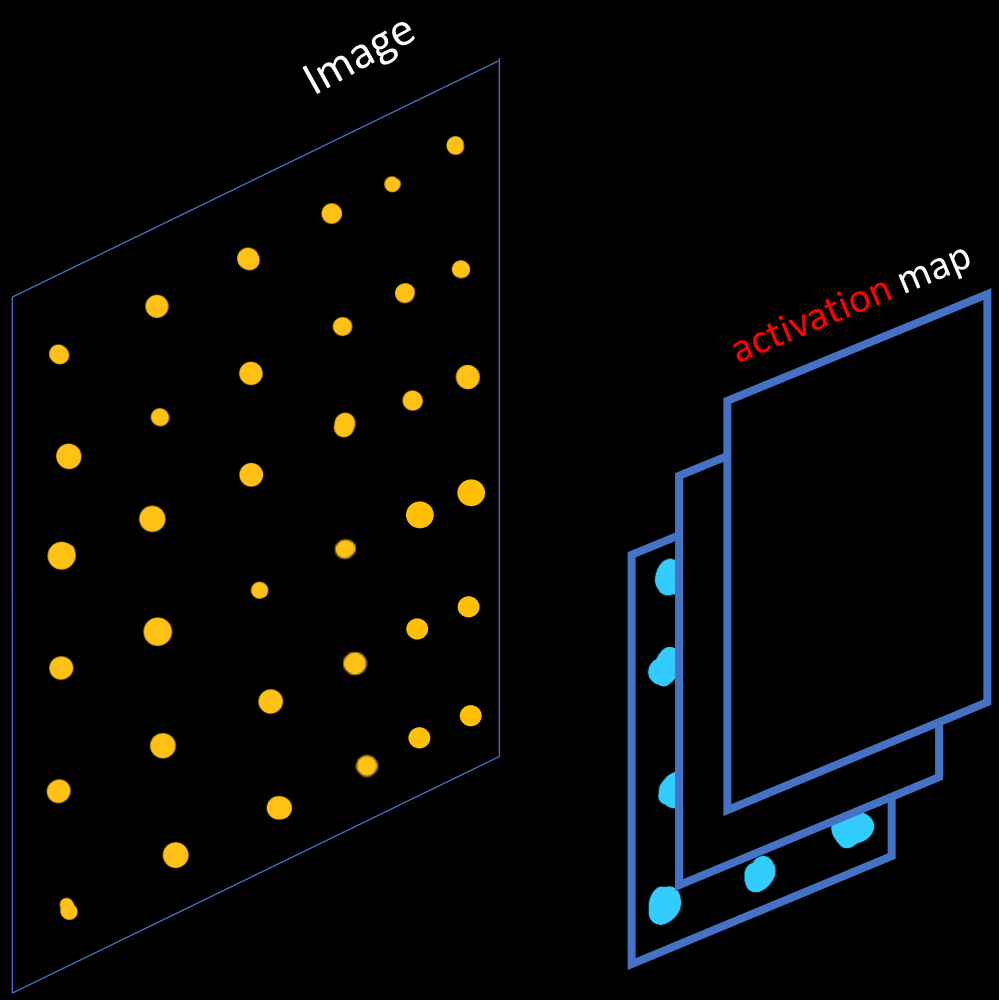
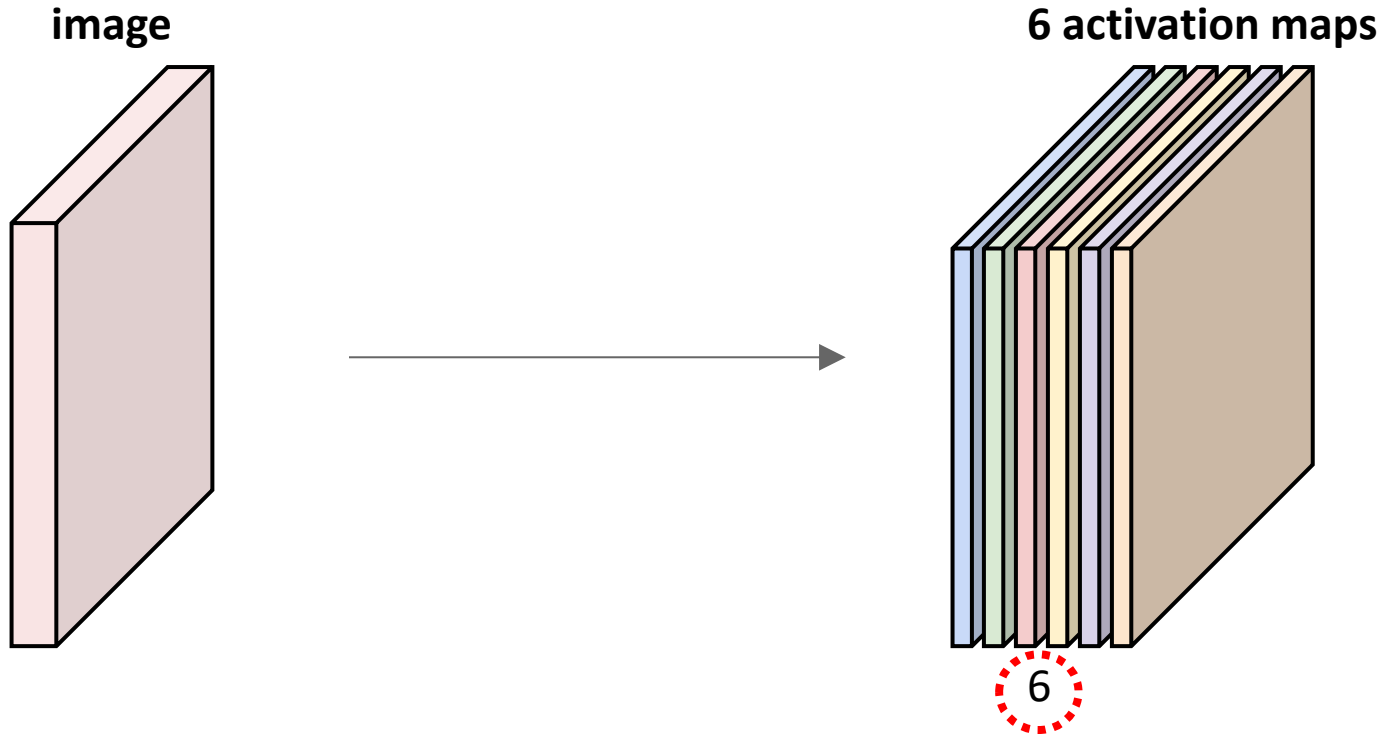
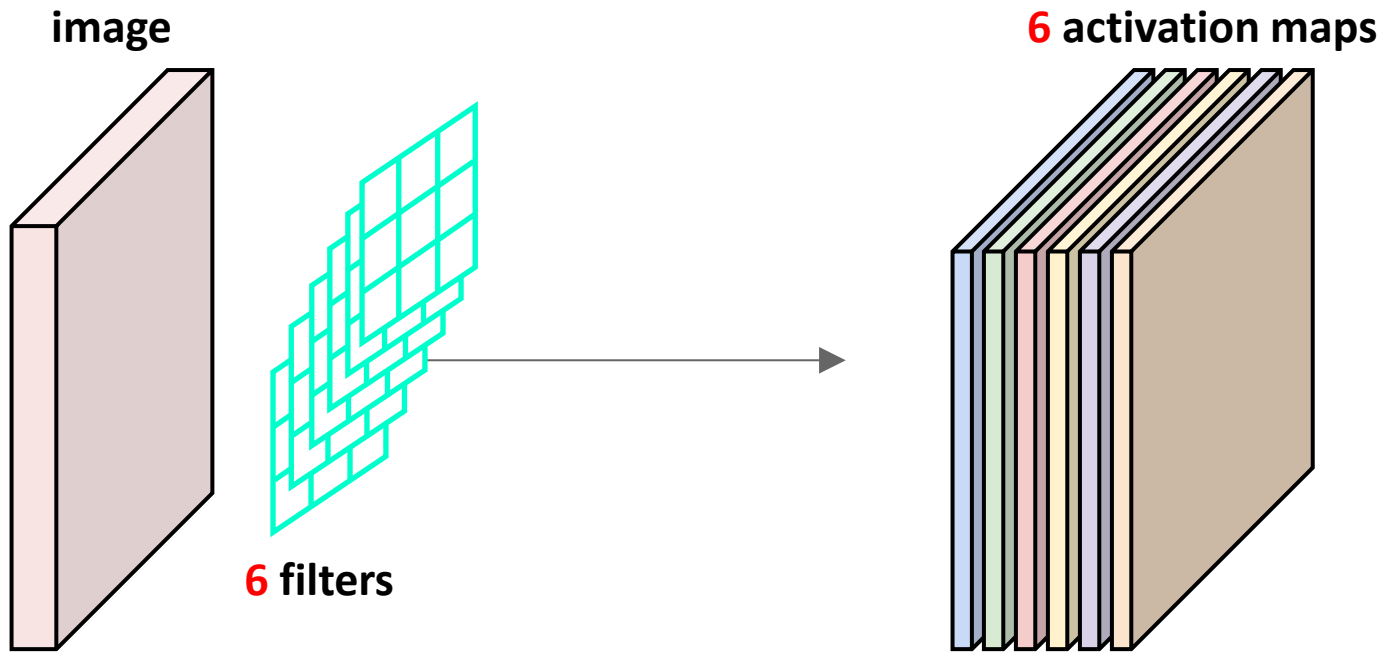


Image
activation map



Where are the filter, and how many?

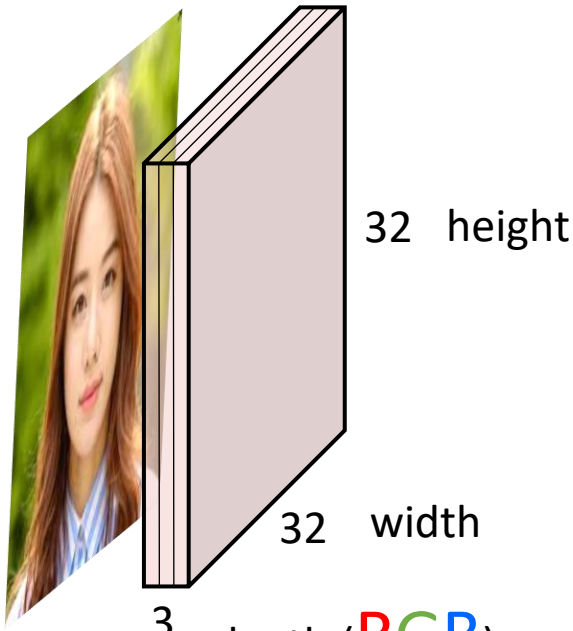


How many filters and where are they?


x1.0
x-1.5
x1.0
x-0.4
x0.7
x1.0
x0.7
x-0.6
x-0.4
x-0.4
x-0.4
x0.0
0.73 0.86 0.74 0.85 0.99 0.58 0.96 0.25
0.29 0.92 0.25 0.38 0.67 0.16 0.63 0.73
0.90 0.99 0.95 0.06 0.56 0.85 0.23 0.84
0.00 0.45 0.55 0.28 0.33 0.59 0.69 0.01
0.50 0.23 0.03 0.41 0.48 0.23 0.62 0.69
0.85 0.52 0.11 0.95 0.80 0.24 0.03 0.54
0.99 0.07 0.69 0.35 0.05 0.15 0.61 0.80
0.03 0.84 0.79 0.52 0.17 0.66 0.73 0.01

Convolution Layer

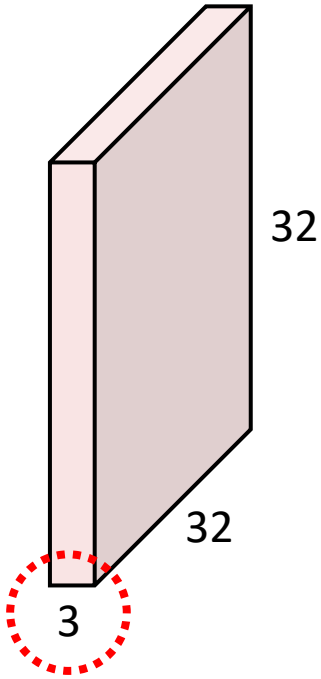
32x32x3 pixels image



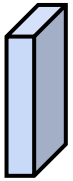
Convolution Layer

Filters always extend the full depth of the input volume

32x32x3 image

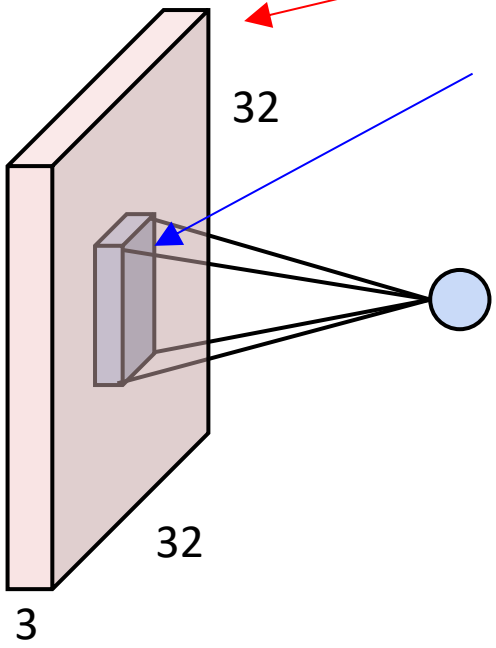


5x5x3 filter



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

Convolution Layer



32x32x3 image

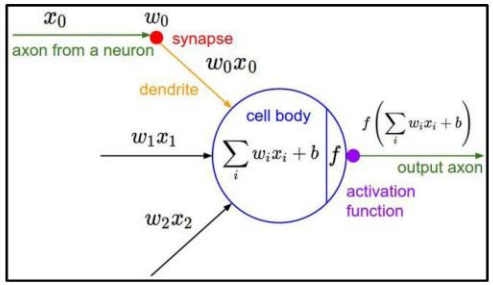
5x5x3 filter w

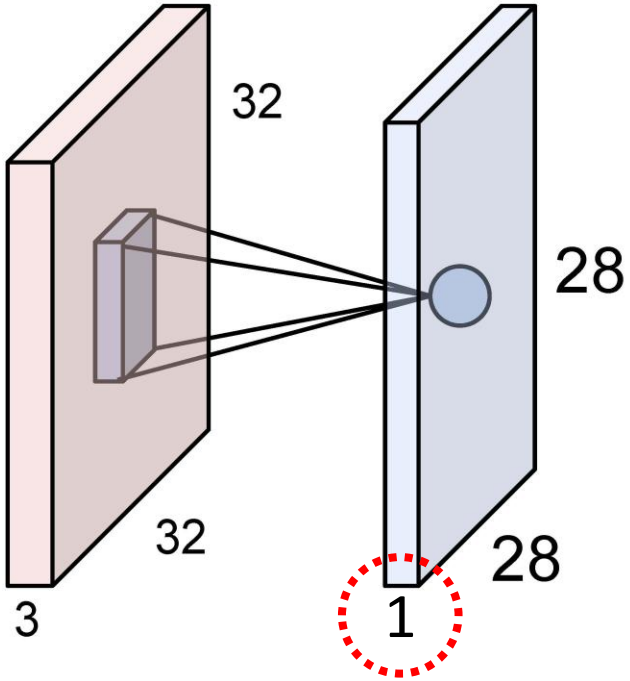
Where are synapses?

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)

$$w^T x + b$$

The brain/neuron view of CONV Layer





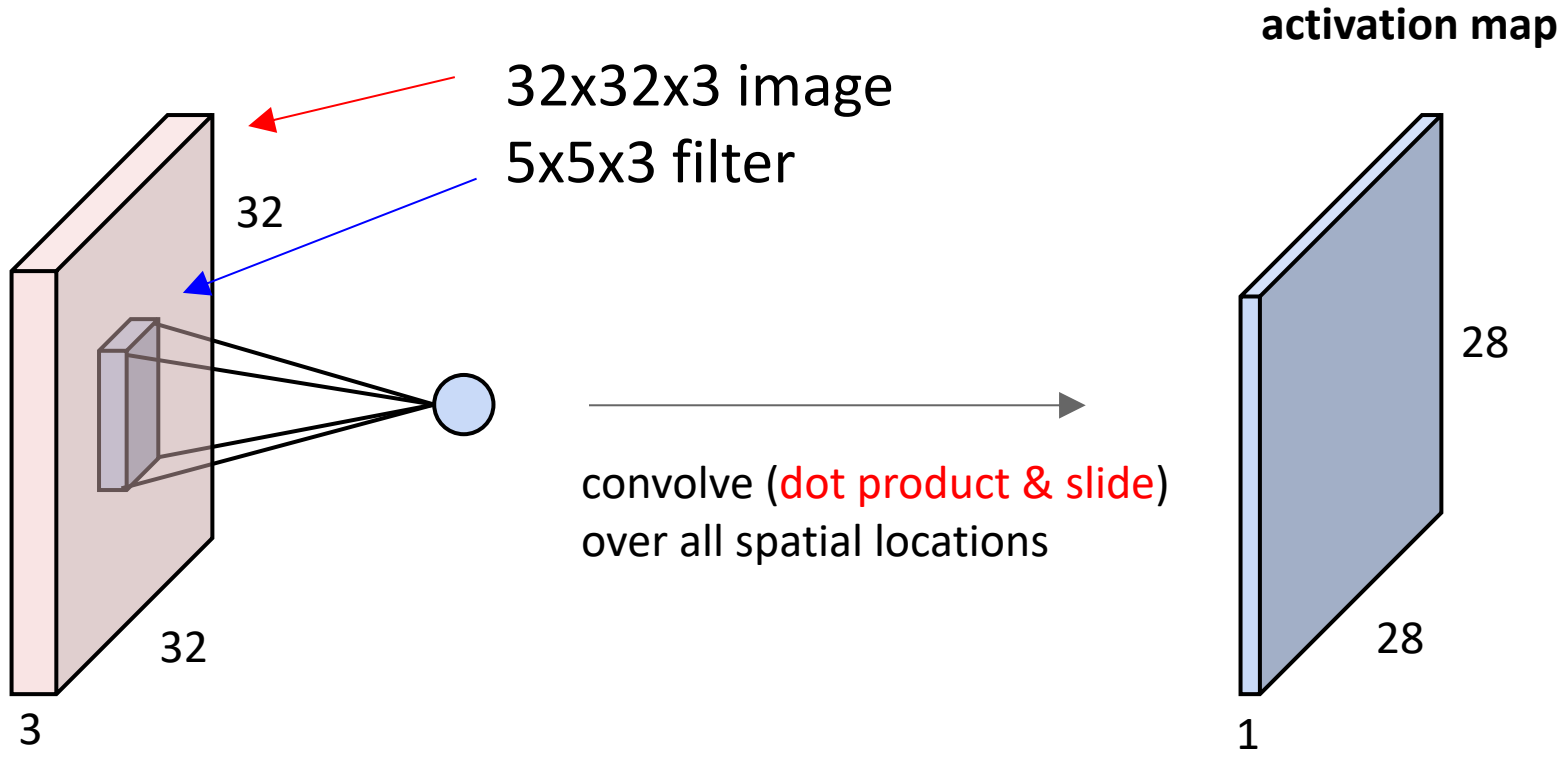
Where are synapses?

An activation map is a 28x28 sheet of neuron outputs:

1. Each is connected to a small region in the input
2. All of them share parameters

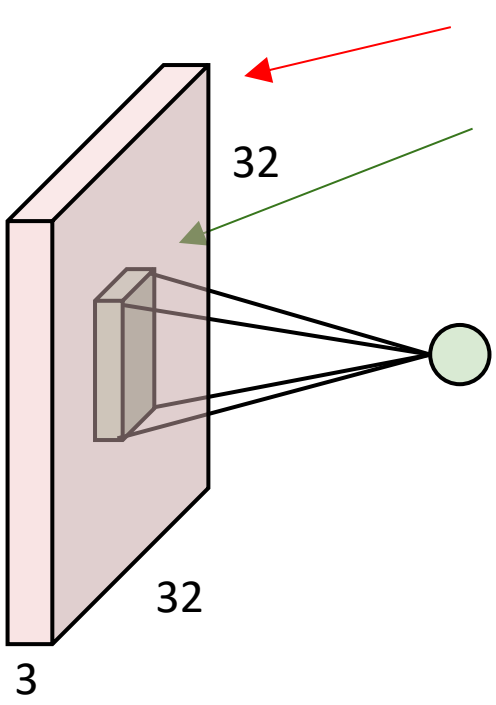
"5x5 filter" -> "5x5 receptive field for each neuron"

Convolution Layer



Convolution Layer

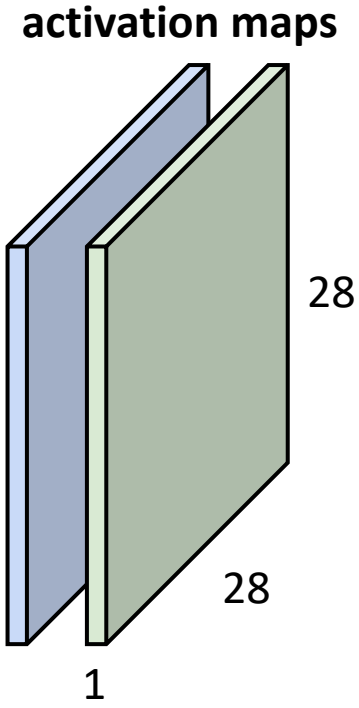
consider a second, green filter



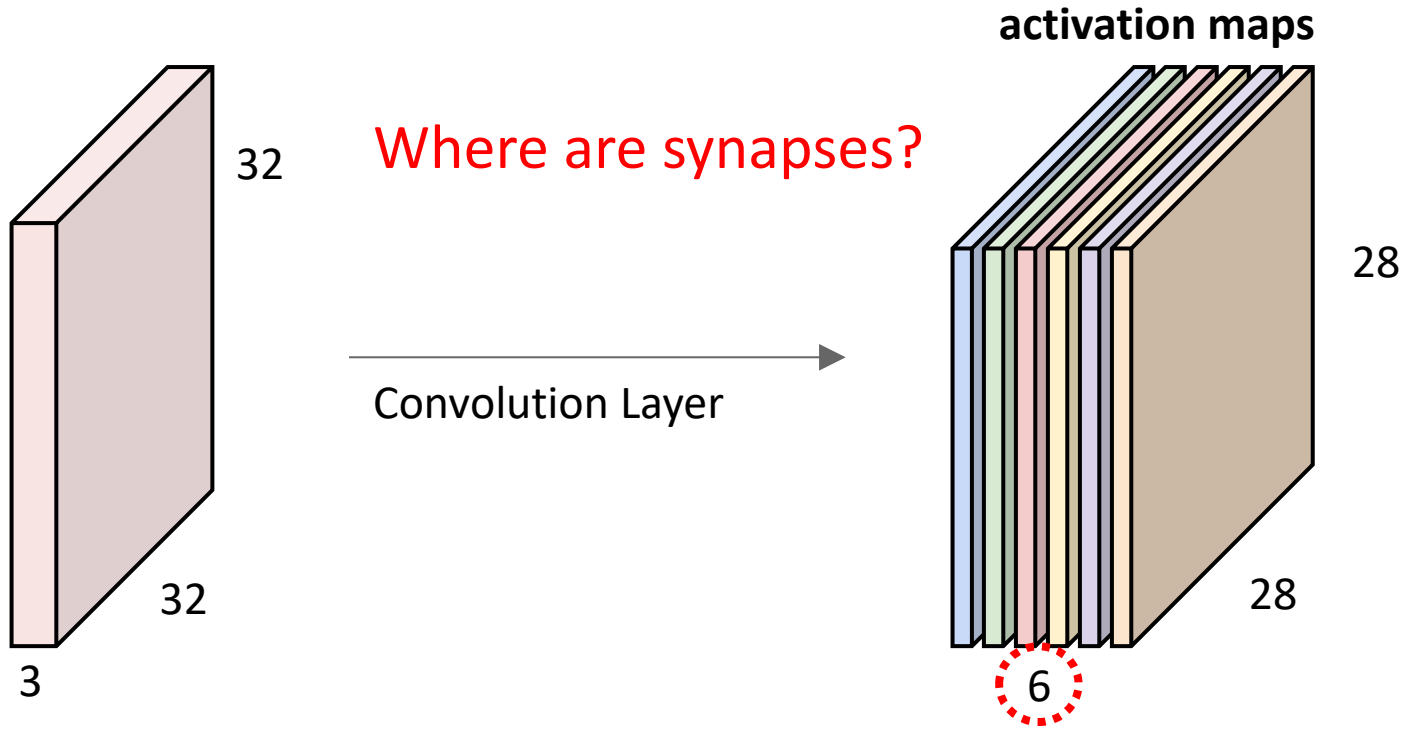
32x32x3 image
5x5x3 filter

Where are synapses?

convolve (slide) over all spatial locations



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

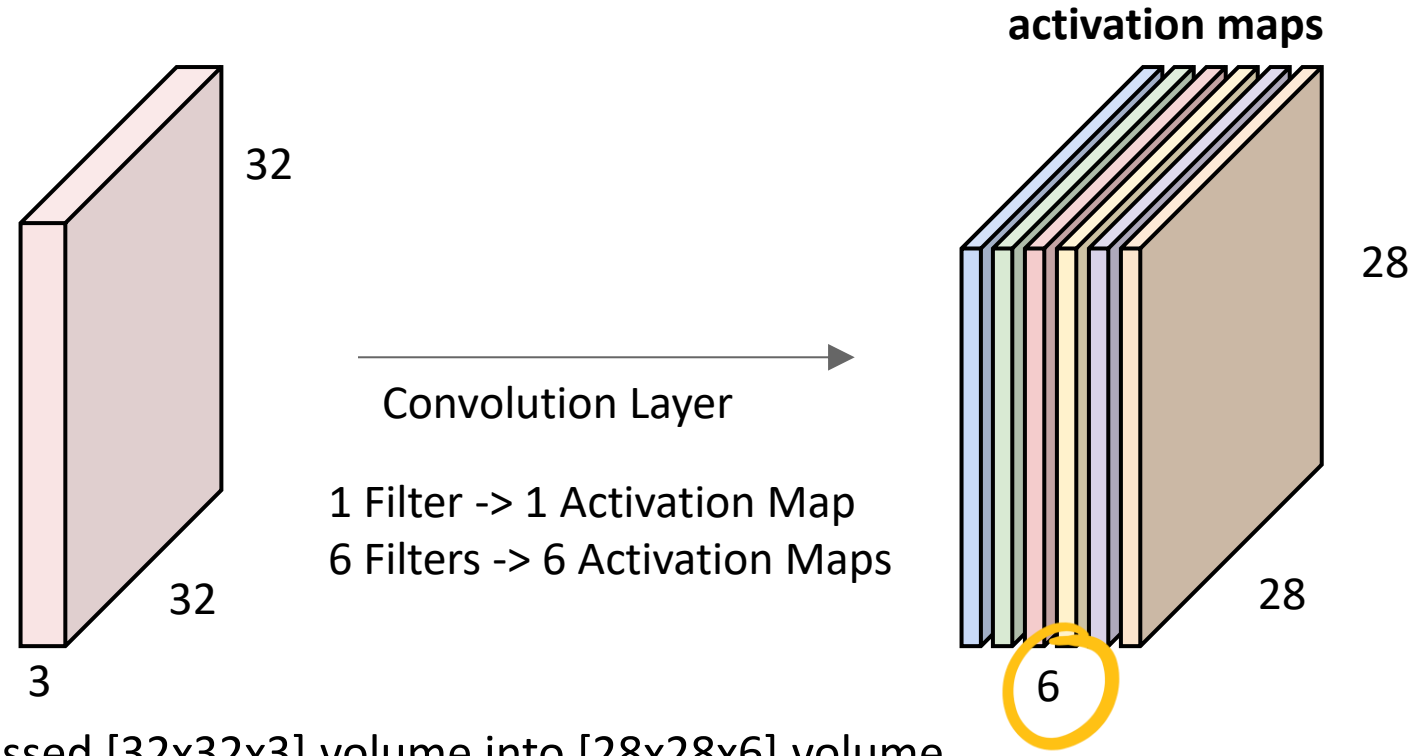


We processed [32x32x3] volume into [28x28x6] volume.

Q: how many parameters would this be if we used a fully connected layer instead?

A: (32*32*3)*(28*28*6) = **14.5M parameters**, **~14.5M multiplies**

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

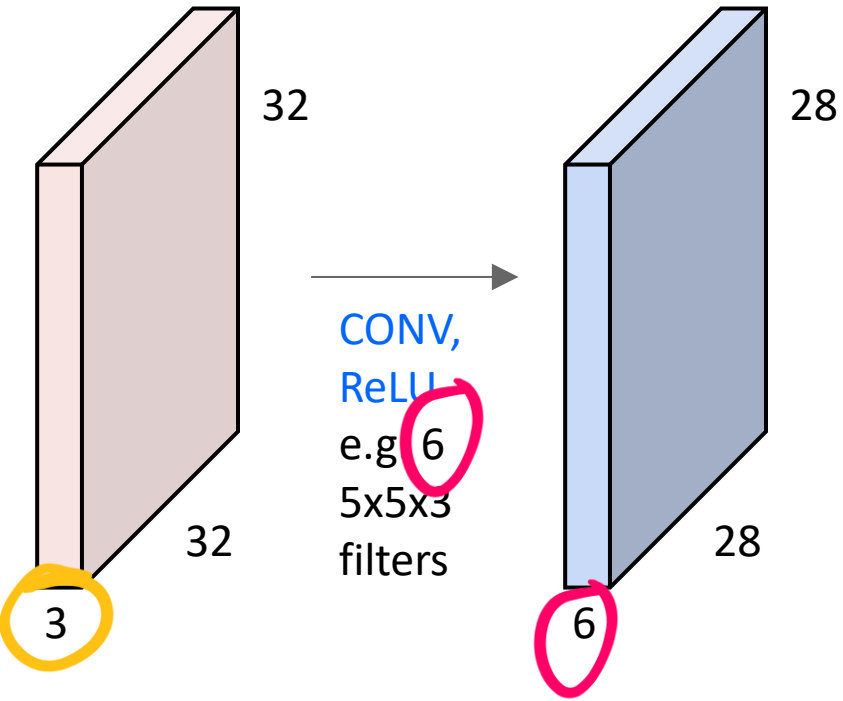


We processed [32x32x3] volume into [28x28x6] volume.

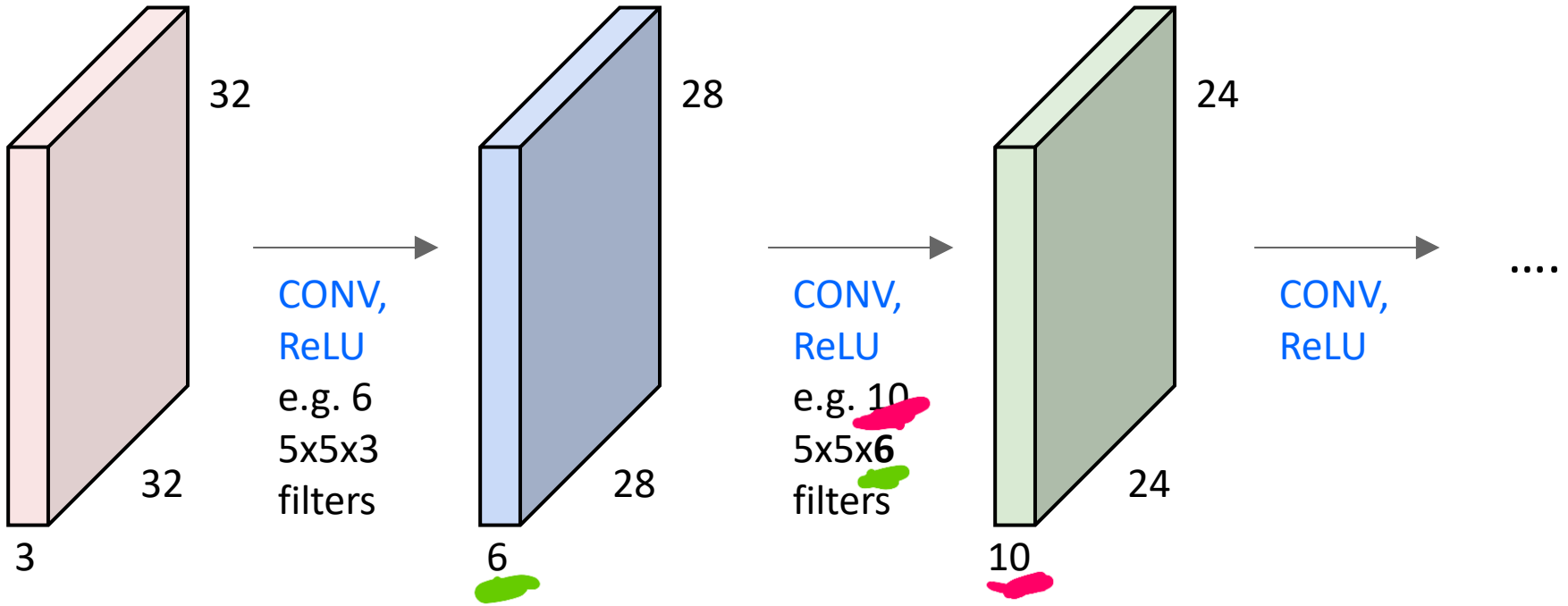
Q: how many parameters are used instead?

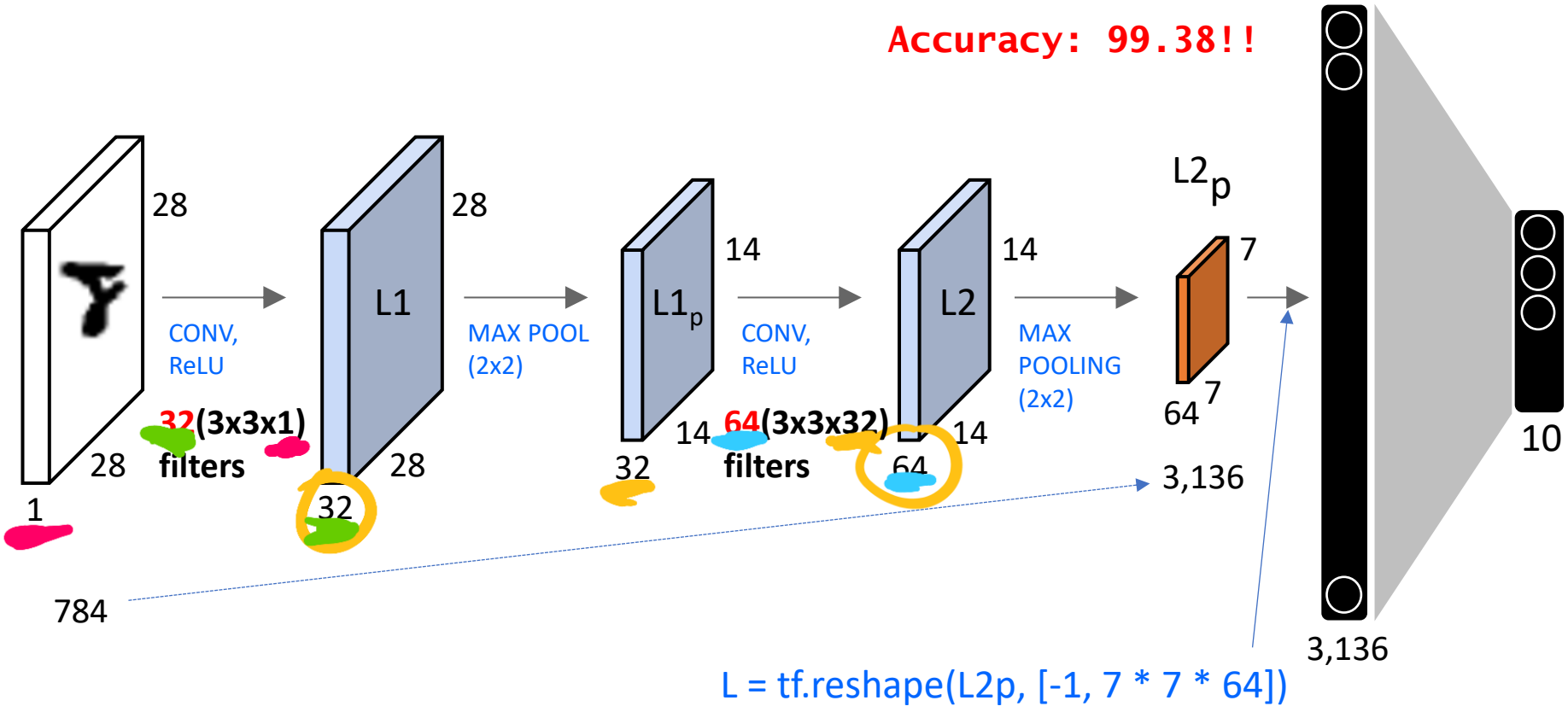
A: (5*5*3)*6 = **450 parameters**, (5*5*3)*(28*28*6) = **~350K multiplies**

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



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



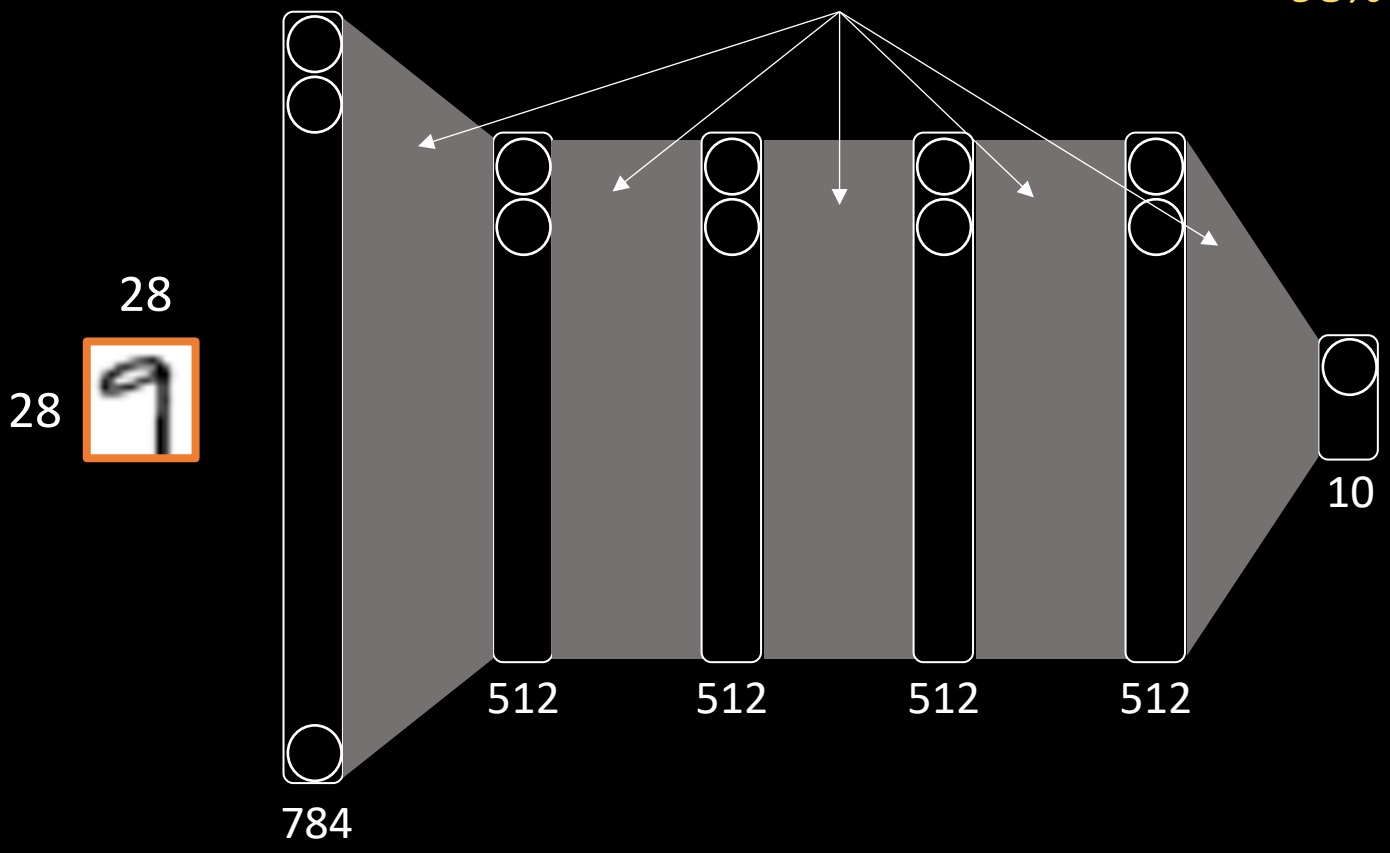


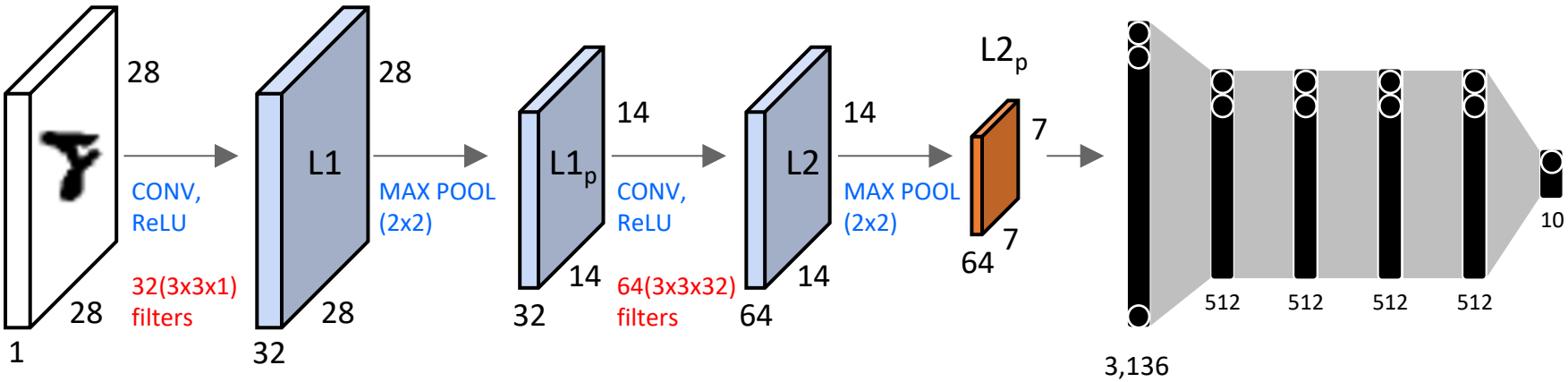
Where are the final features we try to extract?

23_mnist_nn_deep.py

98%

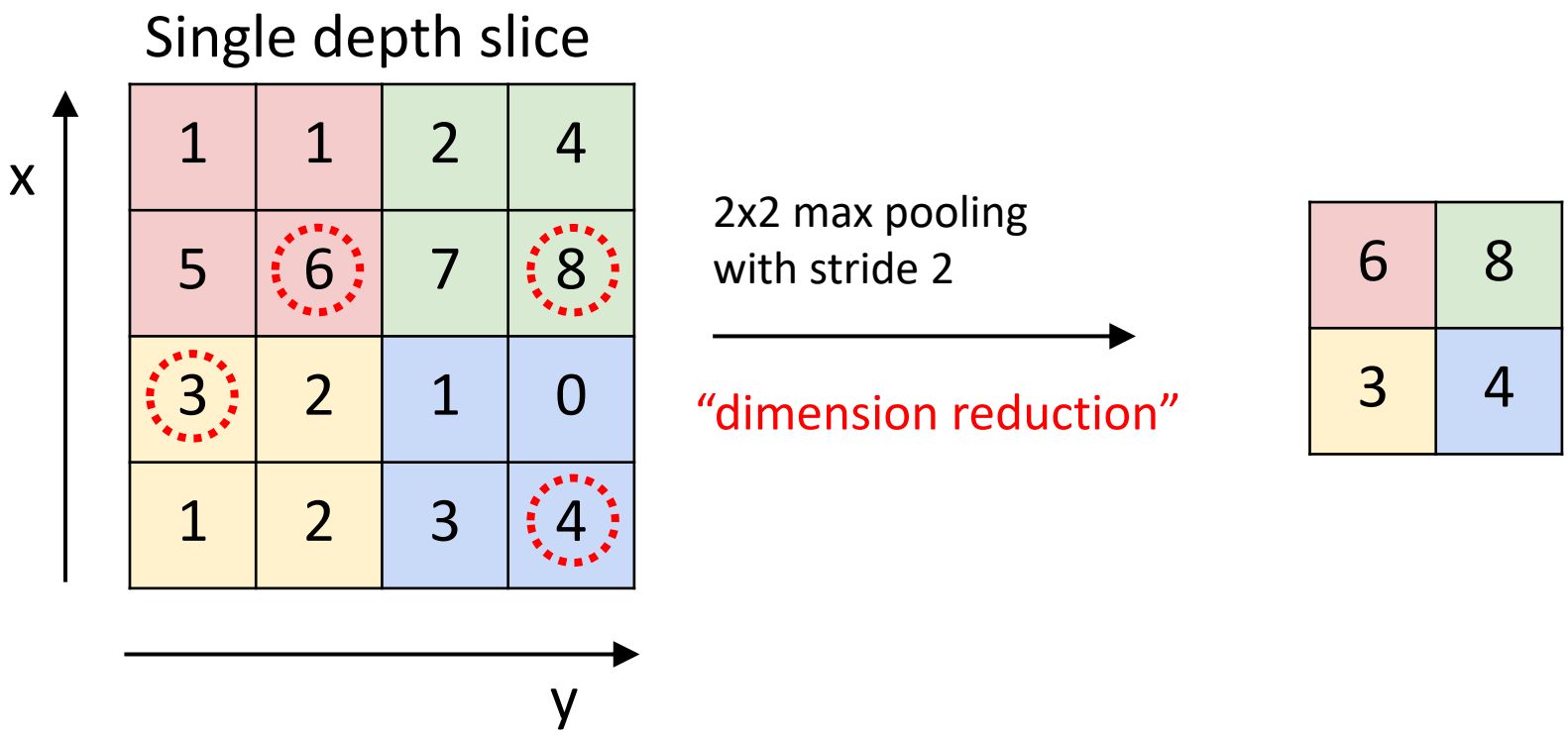
Fully connected





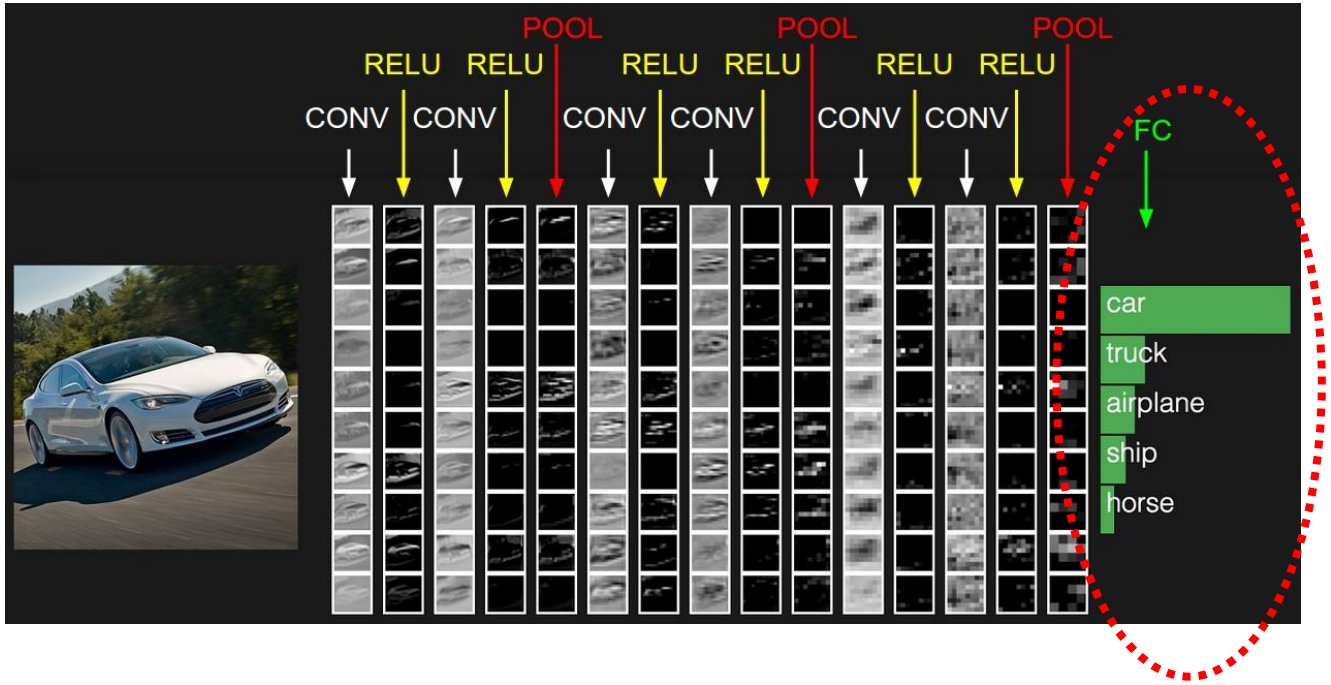
Where are the features we try to extract?

MAX POOLING


Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
2x2 max pooling
with stride 2
"dimension reduction"
6 8
3 4

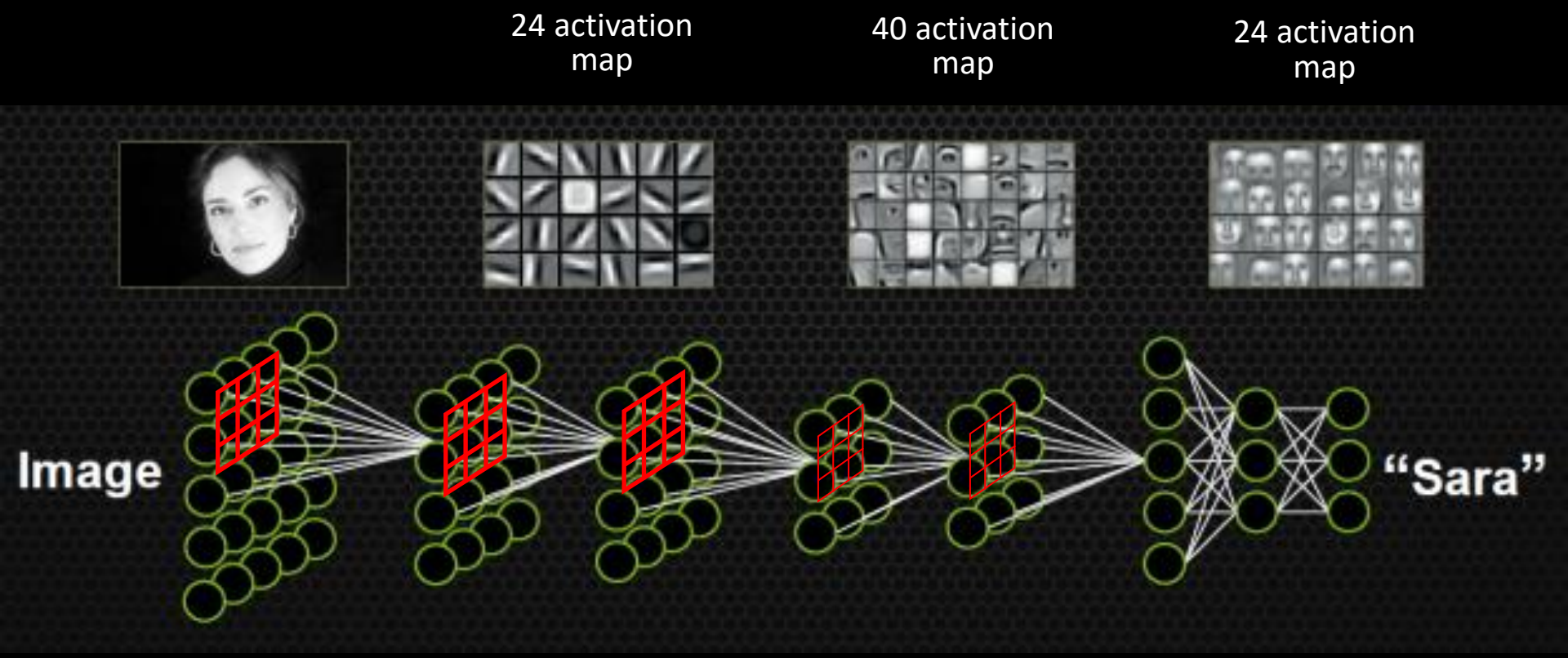
Fully Connected Layer (FC layer)

- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks



10 activation maps

HOW A DEEP NEURAL NETWORK SEES



HOW A DEEP NEURAL NETWORK SEES

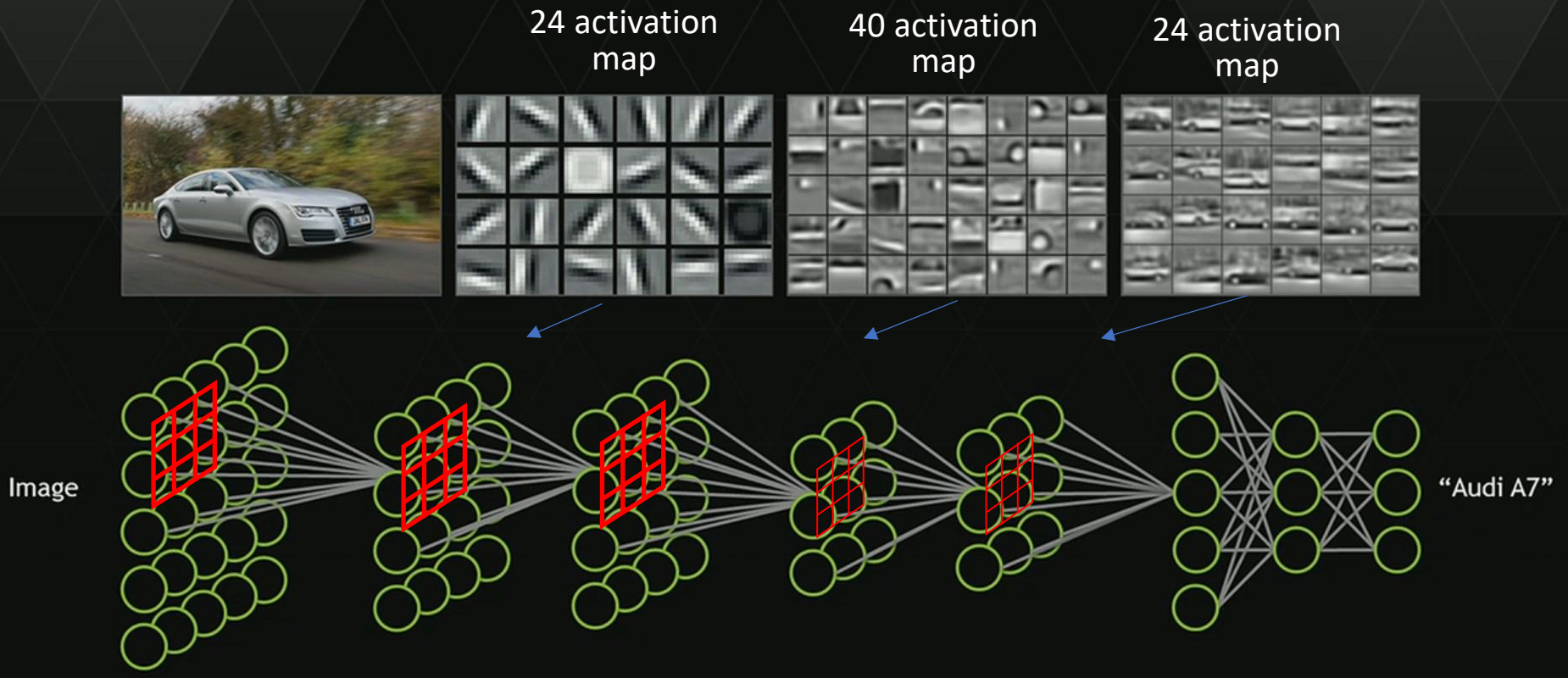
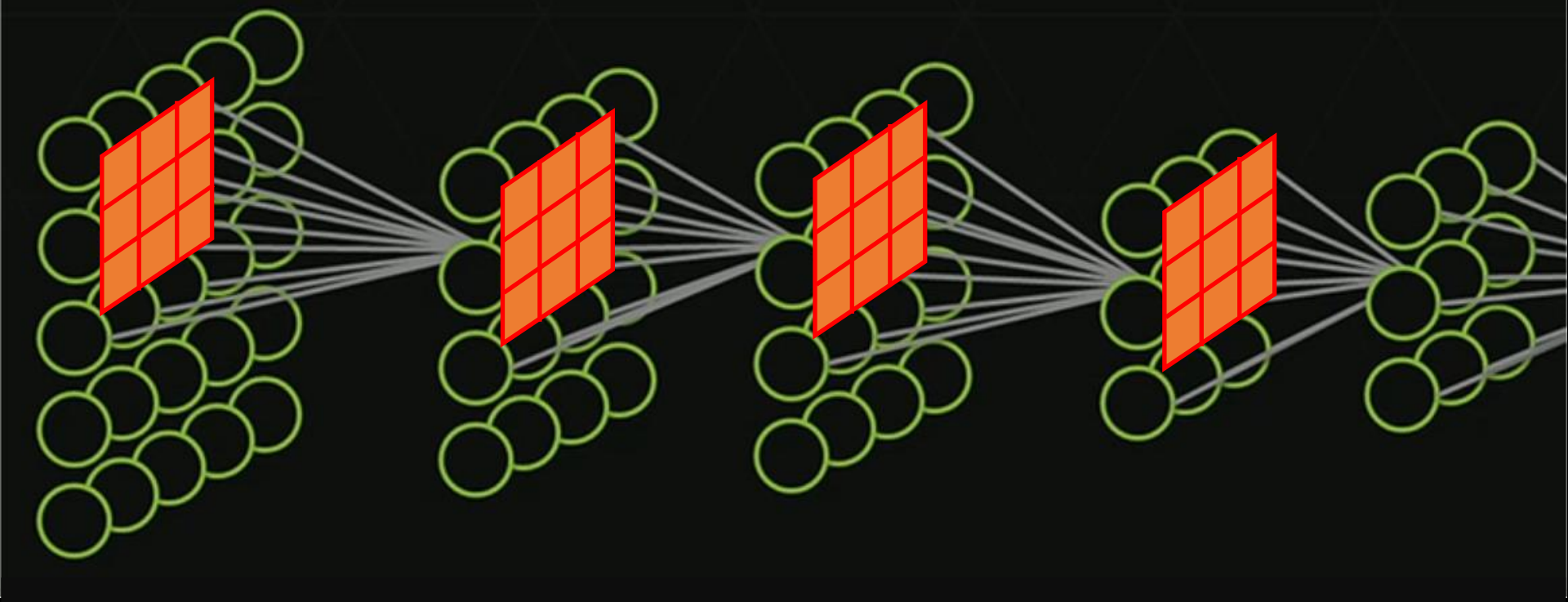


Image source: "Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks" ICML 2009 & Comm. ACM 2011. Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Ng.

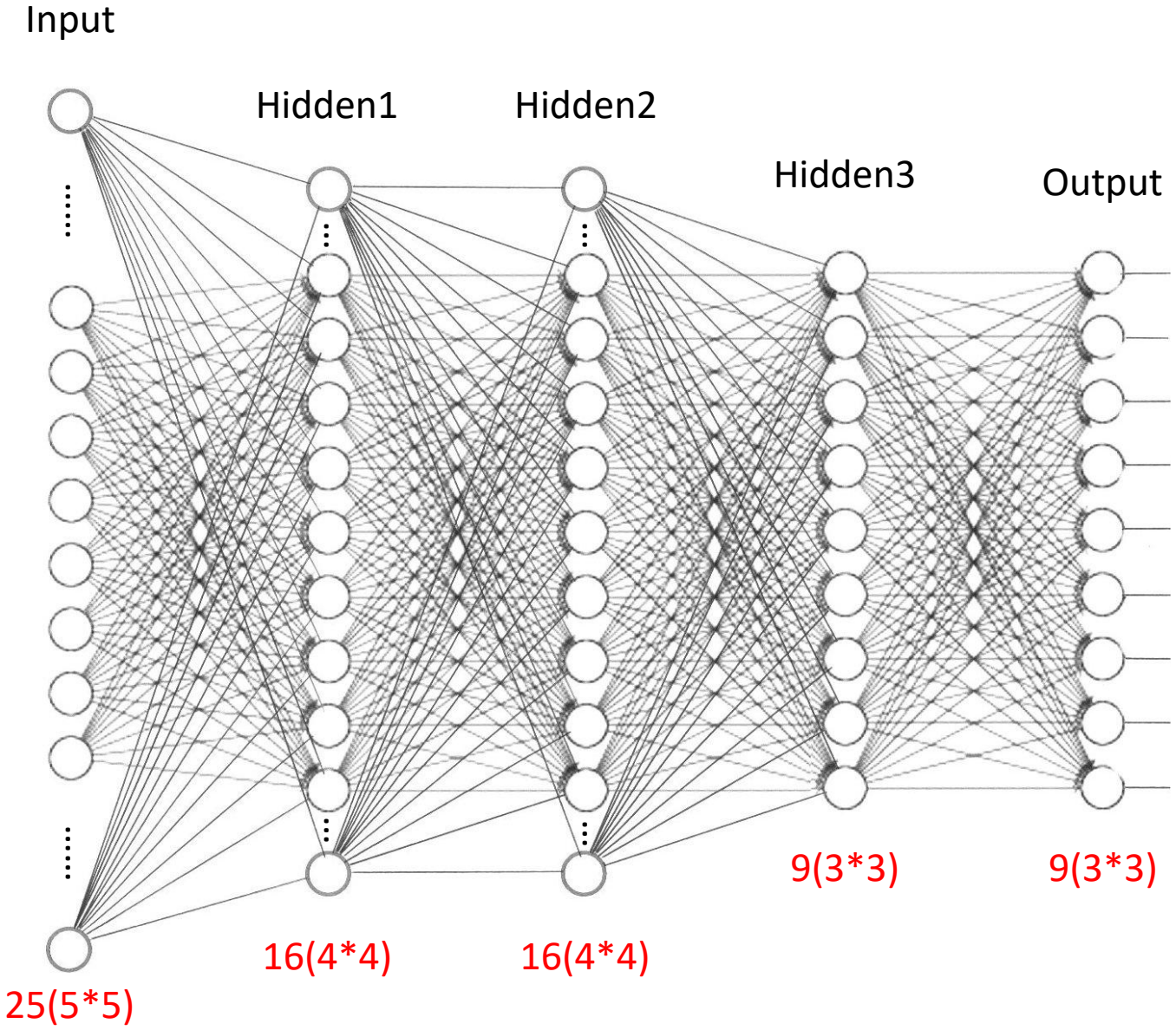
Filter size : 3x3, 1 filter for each convolution,
then how many parameters to be tuned?

$9 * 4 = 36$



If fully connected, $25*16+16*16+16*9+9*9 = 881$

$$\frac{881}{36} = 24$$


Input
Hidden1
Hidden2
Hidden3
Output
25(5*5)
16(4*4)
16(4*4)
9(3*3)
9(3*3)

Some questions about CNN

- Fully connected? (Yes/No)
- Where are the parameters to be tuned?
- We have 2 filters, then?
- Convolution again with the resulting activation map

Learning in CNN

- Parameter tuning in filters
  → filter organizing
- Parameter tuning in fully connected layers

Theoretical backgrounds of CNN

A bit of history:

Hubel & Wiesel,

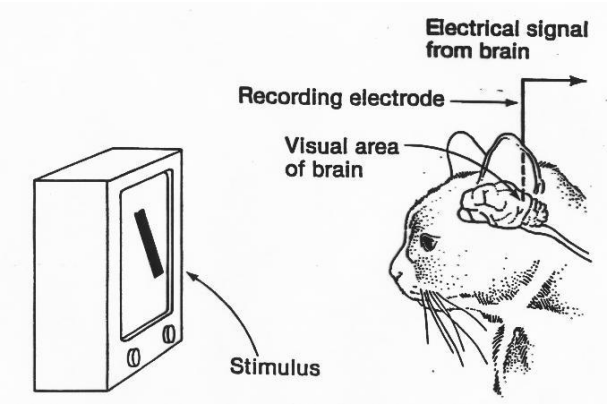
1959

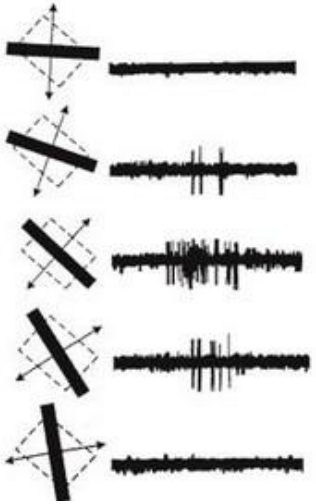
RECEPTIVE FIELDS OF SINGLE NEURONES IN
THE CAT'S STRIATE CORTEX

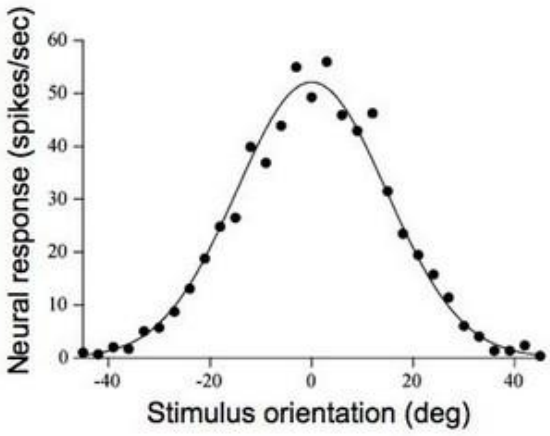
1962

RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX

1968...



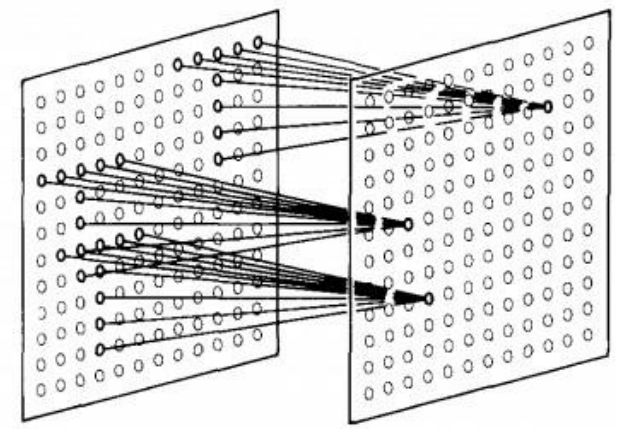


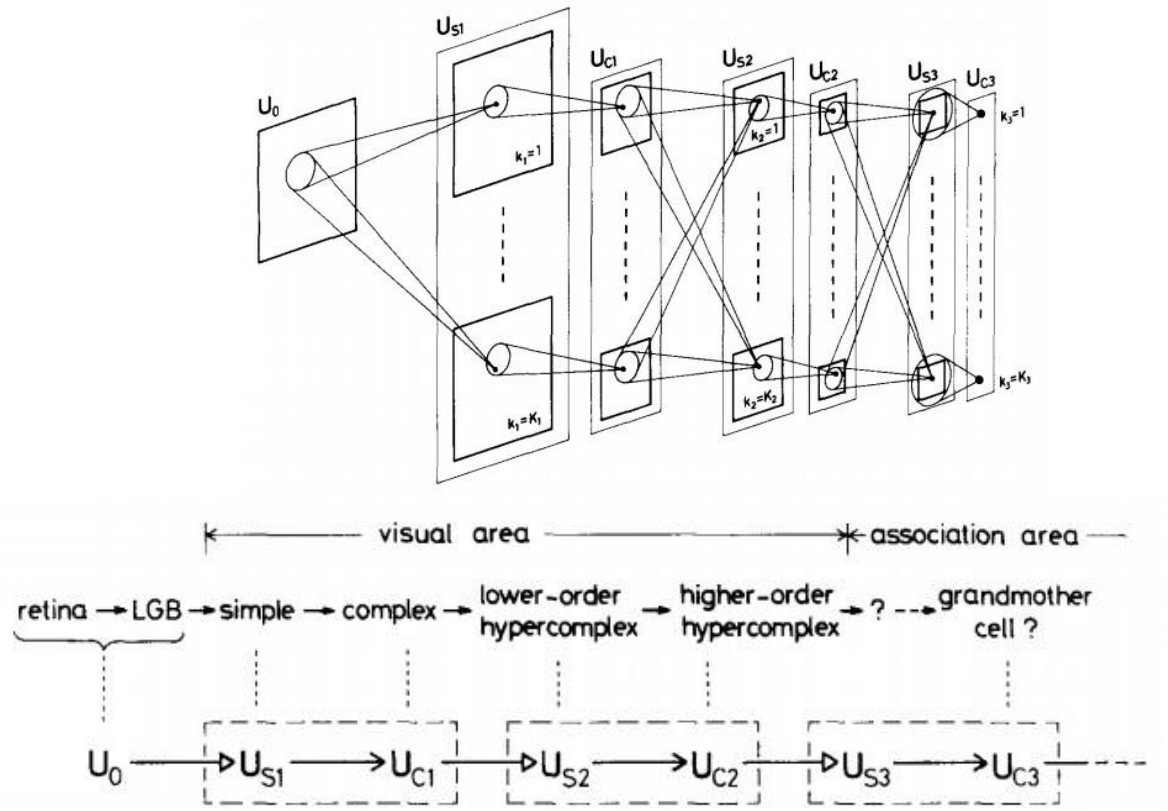


A bit of history:

Neuro-cognitron
[Fukushima 1980]

"sandwich" architecture (SCSCSC…)
simple cells: modifiable parameters
complex cells: perform pooling

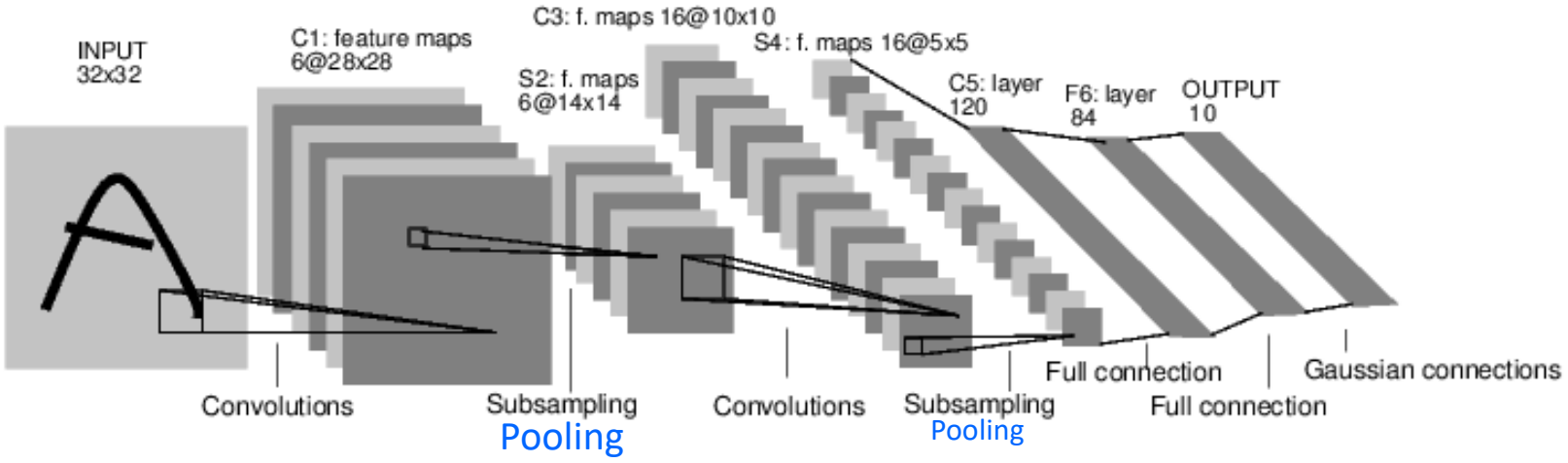




LeNet-5

Gradient-based learning applied to document recognition

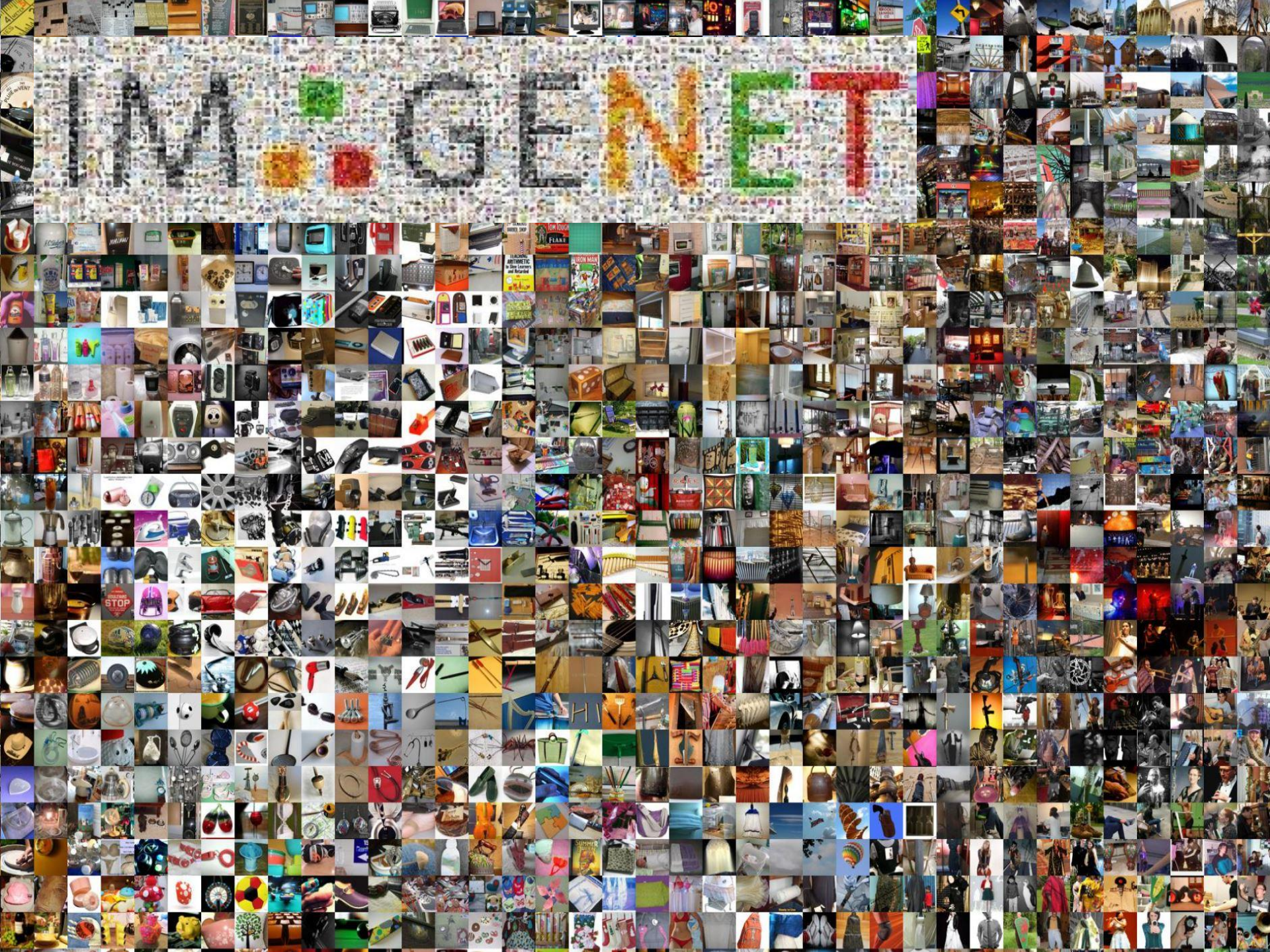
[LeCun, Bottou, Bengio, Haffner 1998]



Convolution-Pooling-Convolution-Pooling-FC

Where are the features we try to extract?

Case Studies

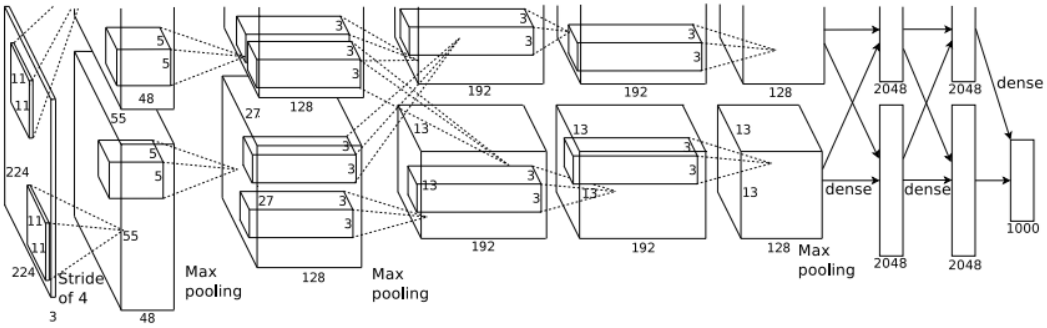

IMAGENET

ImageNet and Competition

- ImageNet: categorization of huge amount of images according to WordNet schema
- database for ILSVRC (ImageNet Large Scale Visual Recognition Competition)
- Largest Computer Vision Challenge showing state-of-the-art performance on the dataset every year
- firstly collected by Deng et al. in 2009
- And it has been organized by Stanford University Vision Lab (prof. Fei-Fei Li) since 2010.

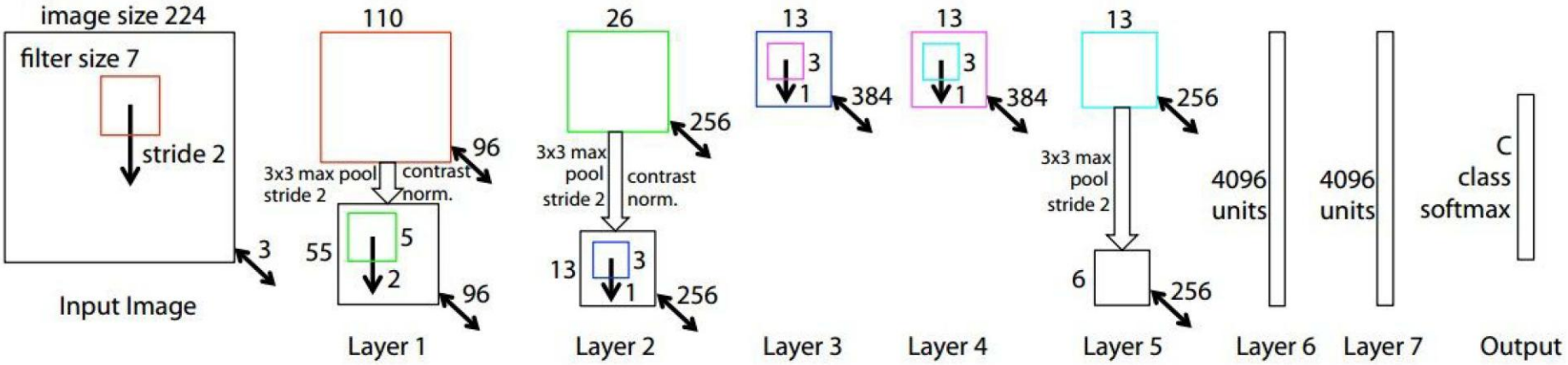
Case Study: AlexNet

[Krizhevsky et al. 2012]



Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

Case Study: ZFNet *[Zeiler and Fergus, 2013]*



AlexNet but:
CONV1: change from (11x11 stride 4) to (7x7 stride 2)
CONV3,4,5: instead of 384, 384, 256 filters use 512, 1024, 512

ImageNet top 5 error: 15.4% -> 14.8%

Case Study: VGGNet

[Simonyan and Zisserman, 2014]

Only 3x3 CONV stride 1, pad 1
and 2x2 MAX POOL stride 2

best model

11.2% top 5 error in ILSVRC 2013
->
7.3% top 5 error

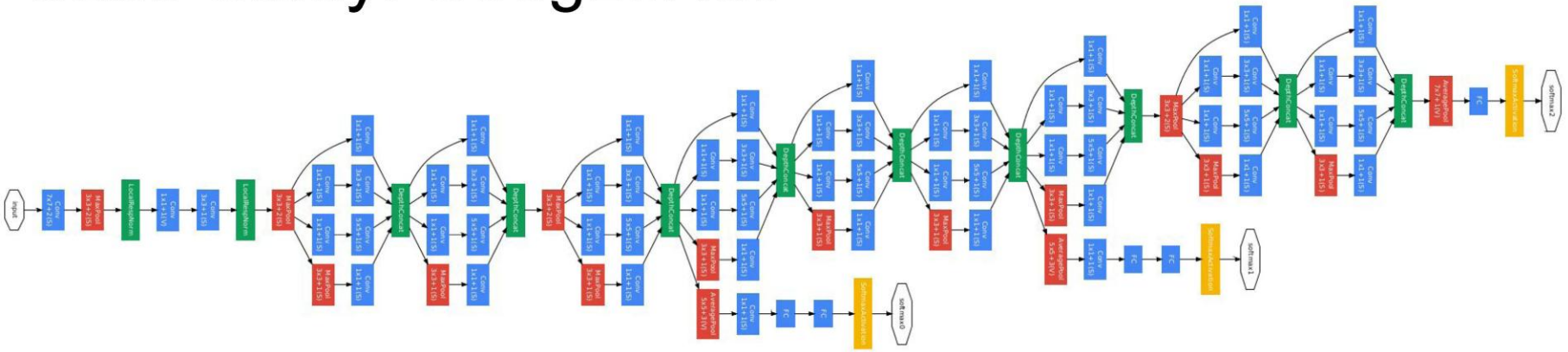
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

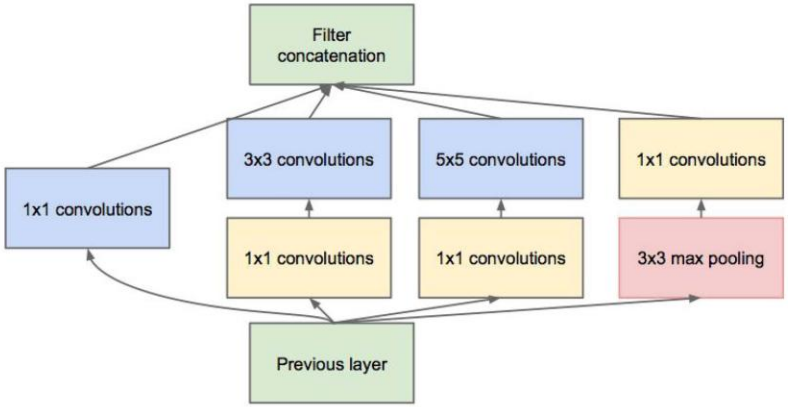
Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

Case Study: GoogLeNet

[Szegedy et al., 2014]



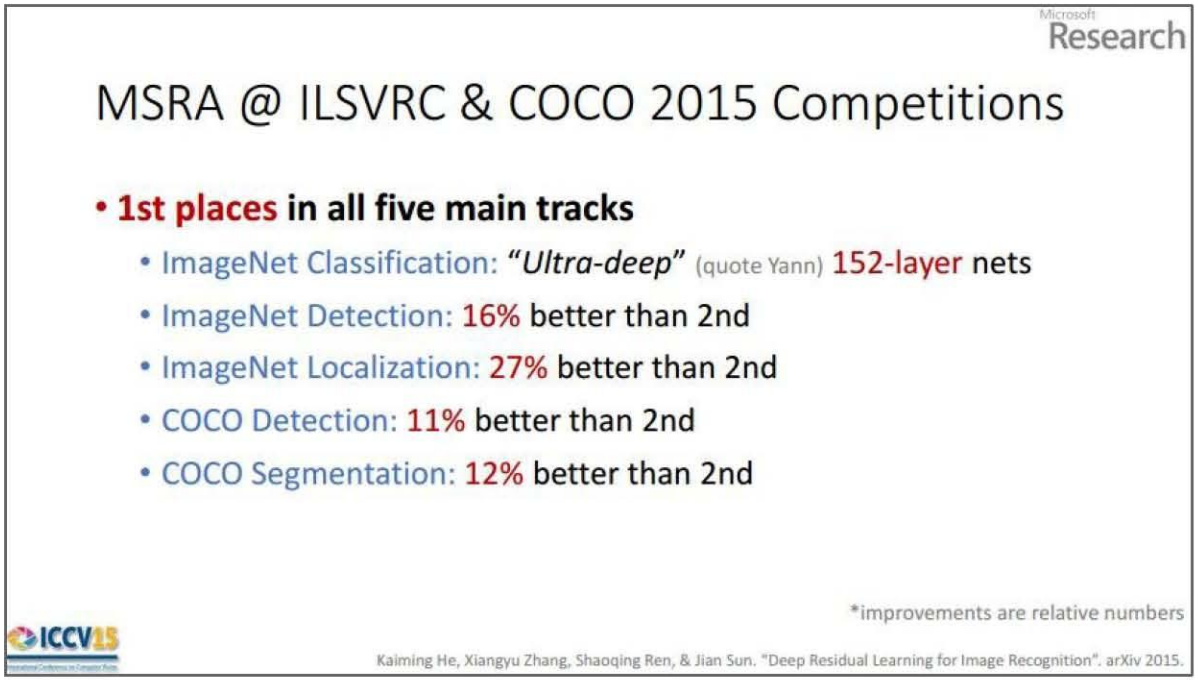


Inception module

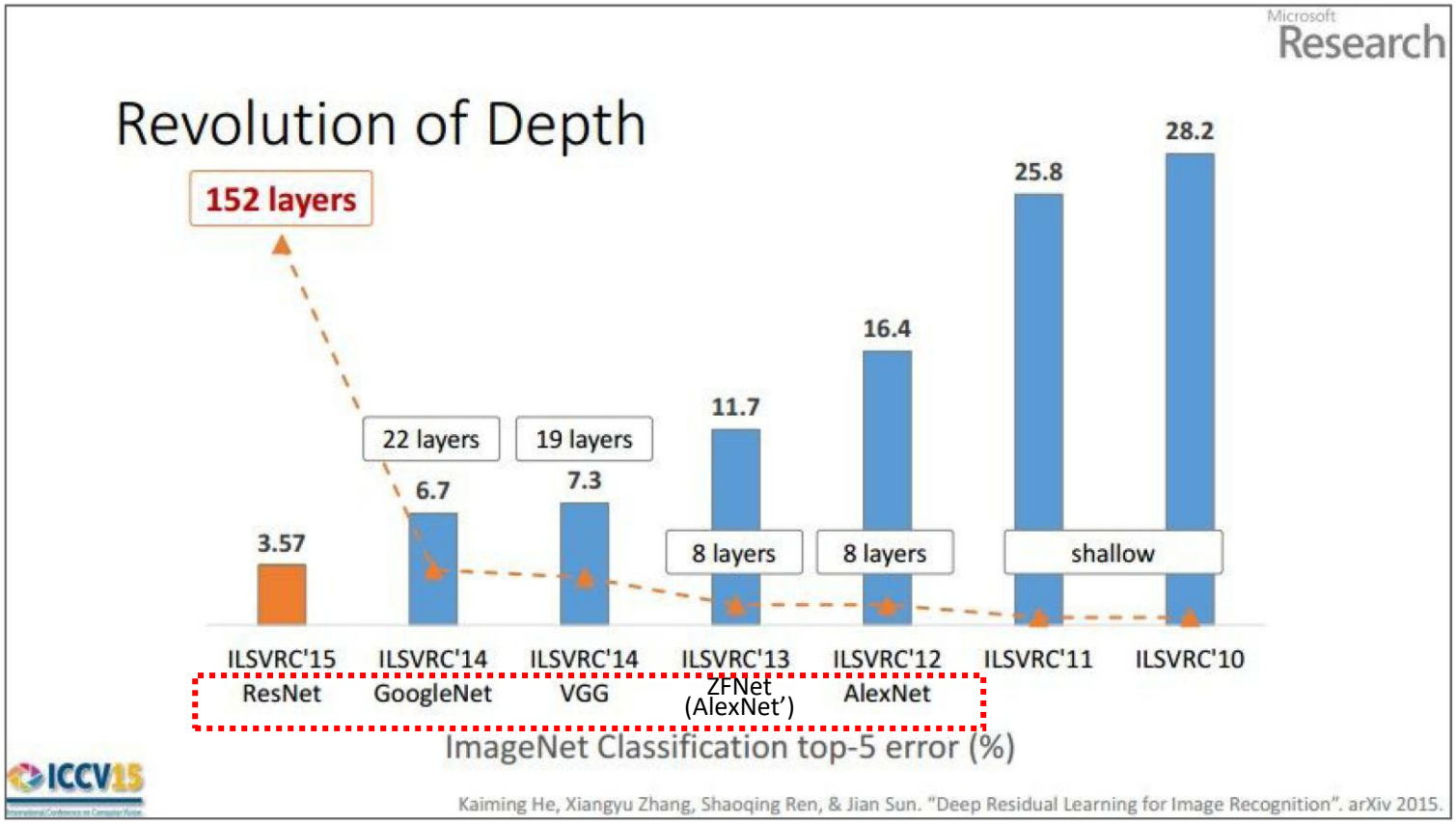
ILSVRC 2014 winner (6.7% top 5 error)

Case Study: ResNet *[He et al., 2015]*

ILSVRC 2015 winner (3.6% top 5 error)



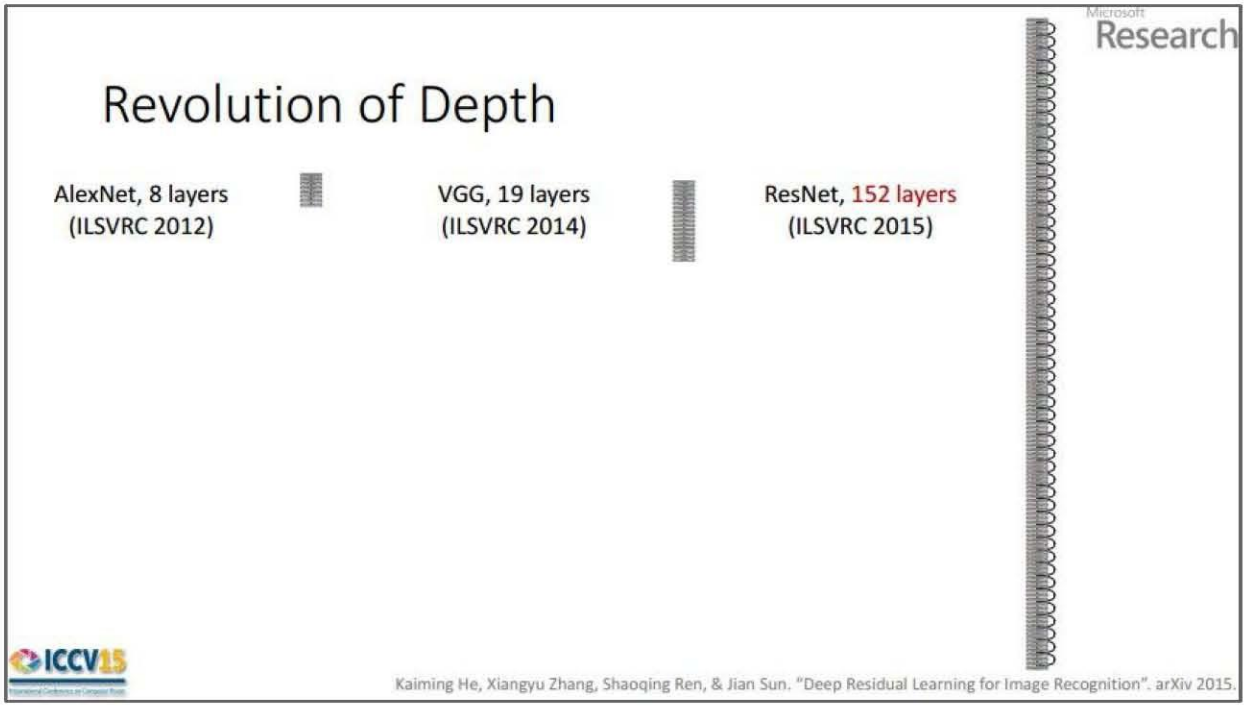
Slide from Kaiming He's recent presentation https://www.youtube.com/watch?v=1PGLj-uKT1w



(slide from Kaiming He's recent presentation)

Case Study: ResNet *[He et al., 2015]*

ILSVRC 2015 winner (3.6% top 5 error)



2-3 weeks of training on 8 GPU machine

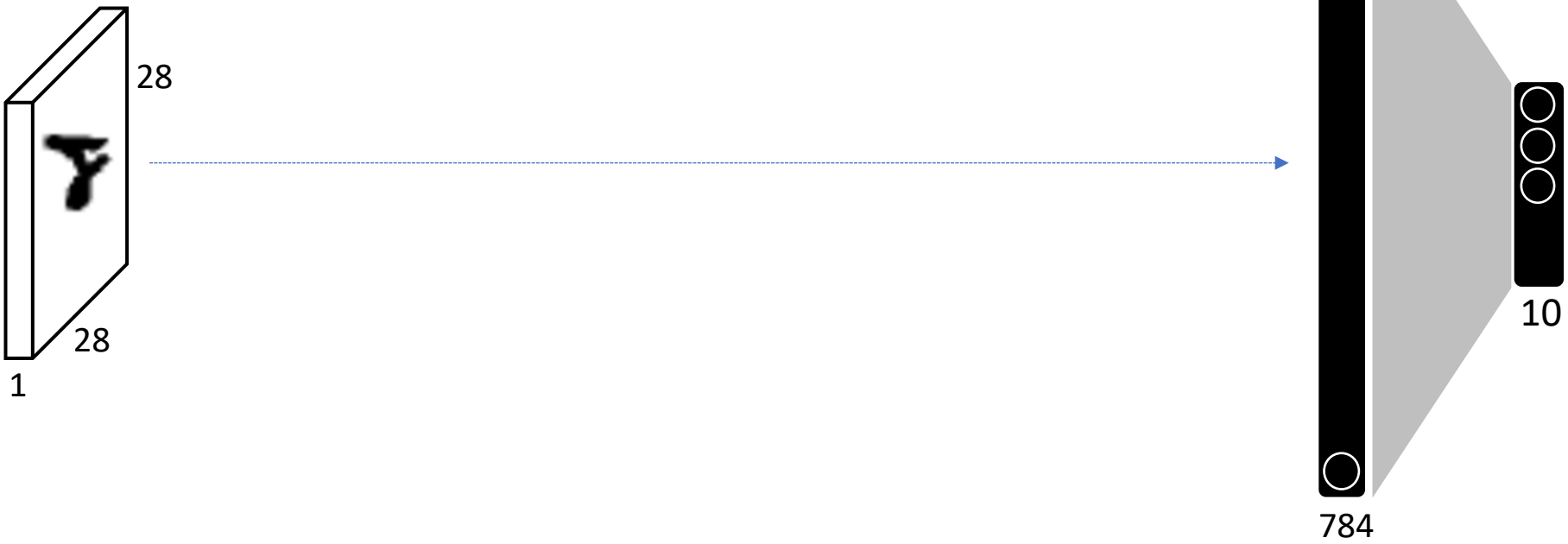
at runtime: faster than a VGGNet! (even though it has 8x more layers)

(slide from Kaiming He's recent presentation)

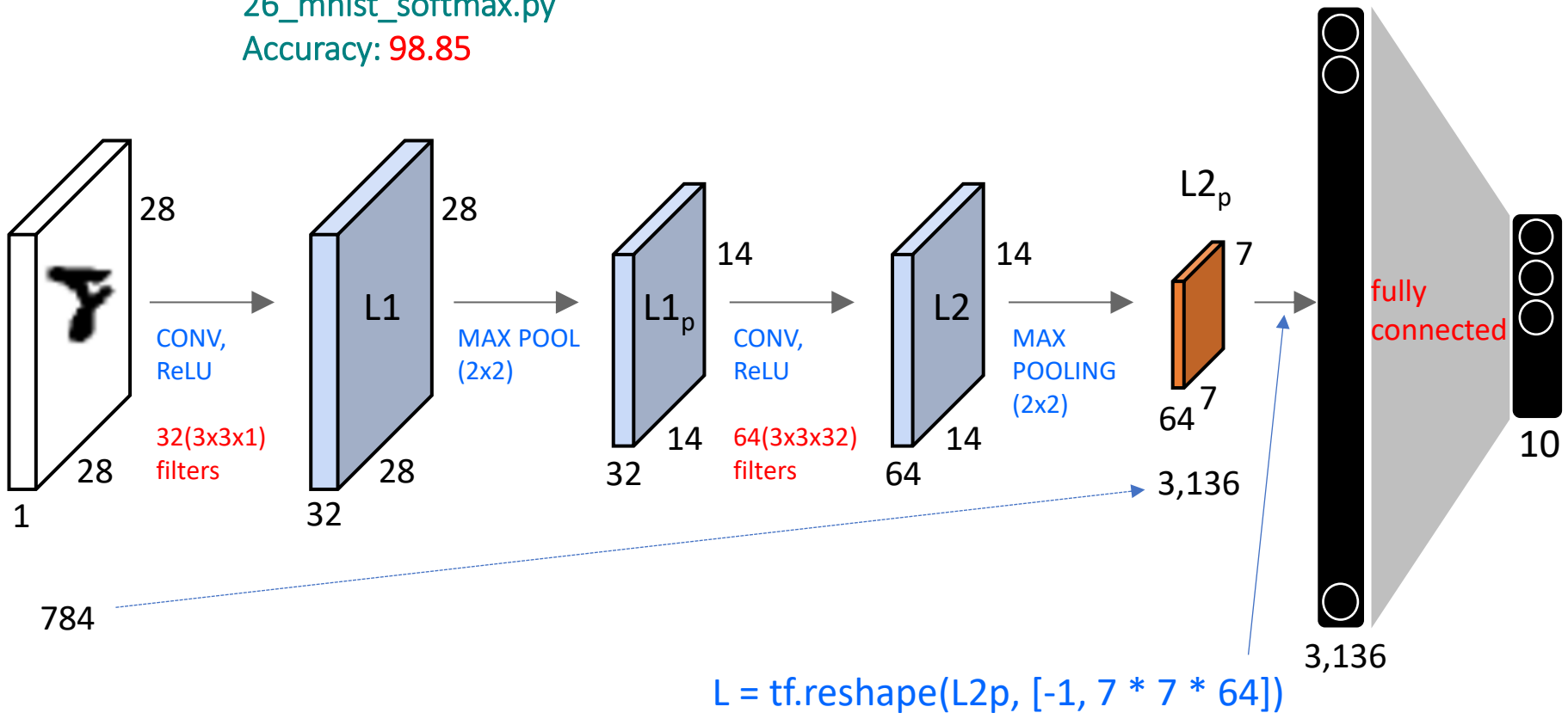
Dot product & shift for feature extraction

CNN, Convolutional Neural Network

20_mnist_softmax.py
Accuracy: 90.23

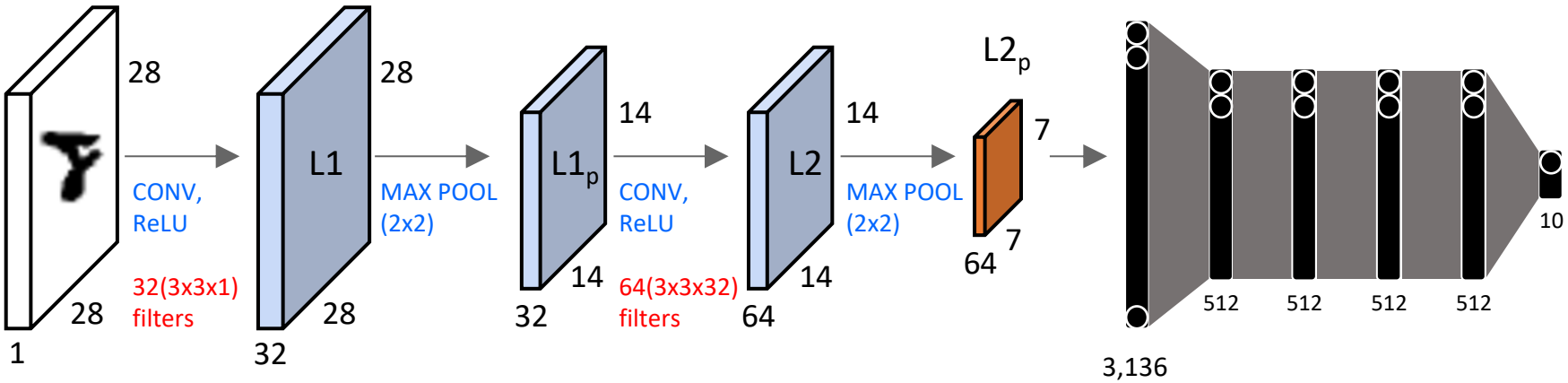


26_mnist_softmax.py
Accuracy: 98.85

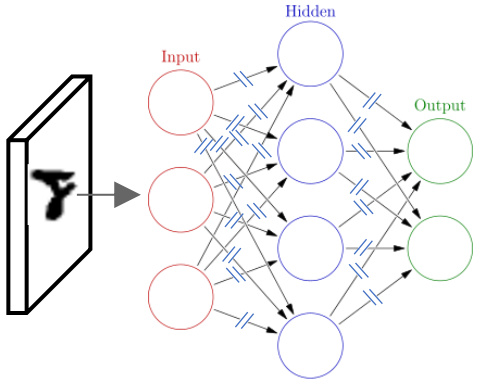


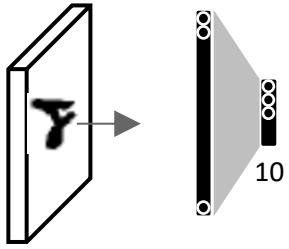
L = tf.reshape(L2p, [-1, 7 * 7 * 64])

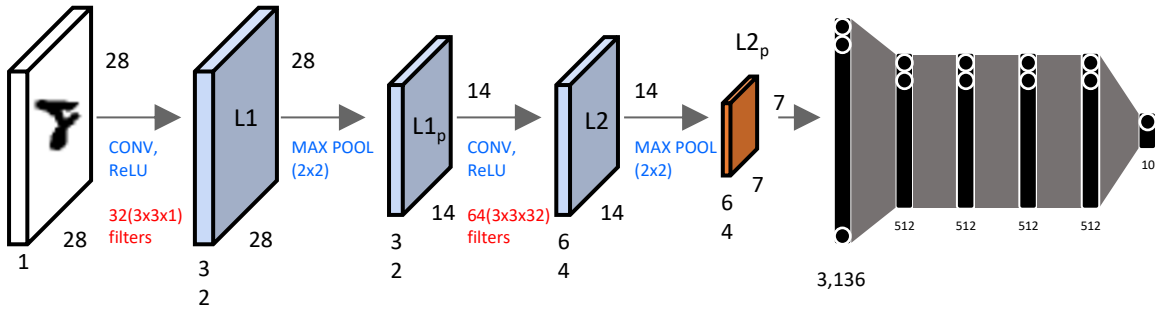
Accuracy: ?????

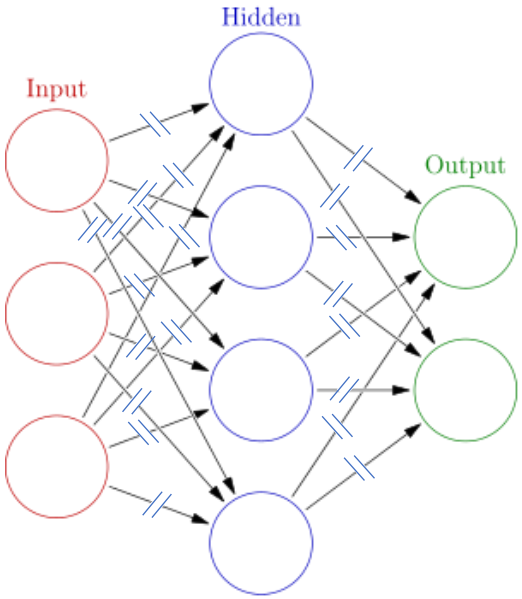

28
28
1
CONV,
ReLU
32(3x3x1)
filters
L1
28
28
32
MAX POOL
(2x2)
L1p
14
14
32
CONV,
ReLU
64(3x3x32)
filters
L2
14
14
64
MAX POOL
(2x2)
L2p
7
7
64
3,136
512
512
512
512
10

Feedforward NN

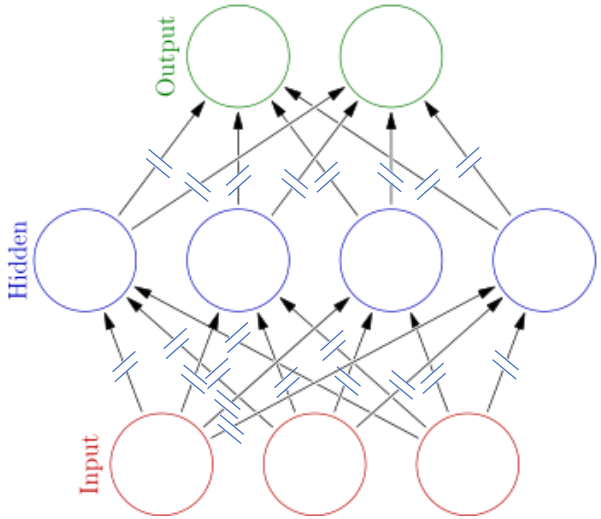




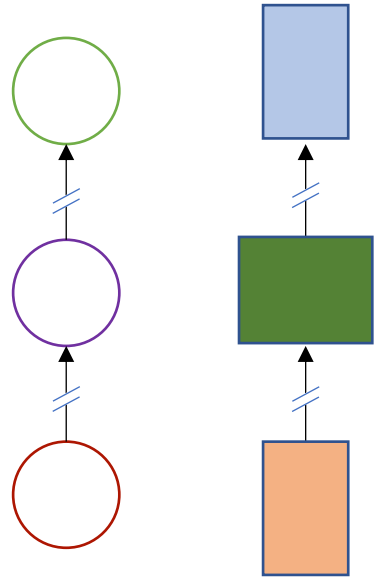




3 layered FCNN

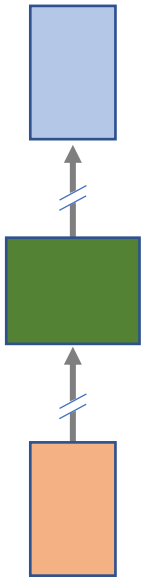


Rotated 90^{o} clockwise



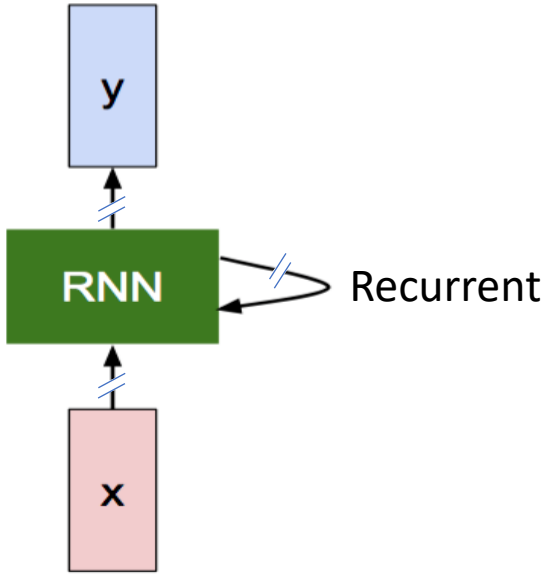
Simplified

Rectangular representation



Simplified NN

What is the arrow?
Where are synapses?
What is learning?



Recurrent NN

Imagine the inside of RNN

Recurrent Neural Networks

RNN

Seq2Seq

Transformer

LLMs

(ChatGPT, Gemini, Claude, etc)