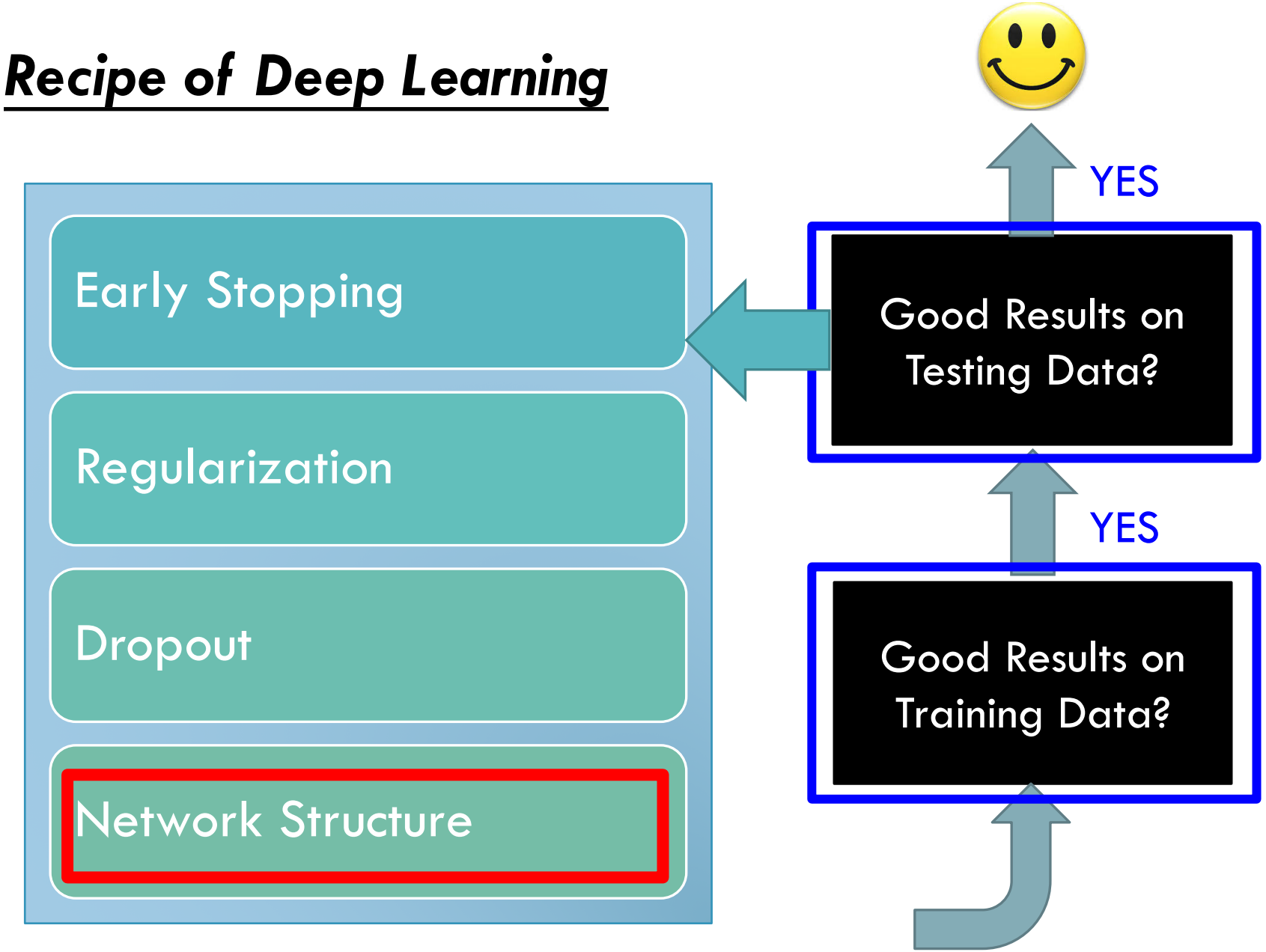
CONVOLUTIONAL NEURAL NETWORK

Narges Norouzi

Recipe of Deep Learning



VARIANTS OF NEURAL NETWORKS

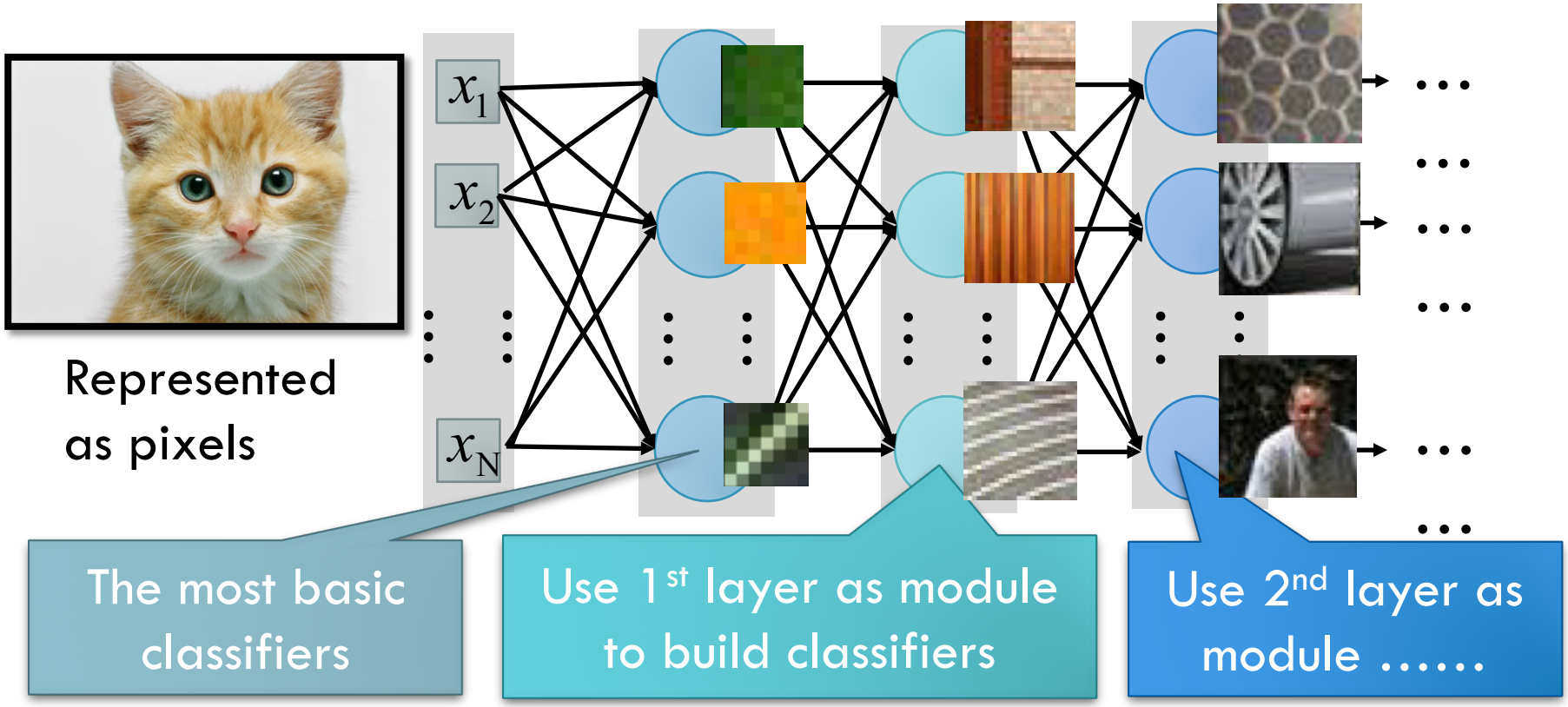
Convolutional Neural Network (CNN)

Widely used in image processing

Recurrent Neural Network (RNN)

WHY CNN FOR IMAGE?

[Zeiler, M. D., *ECCV 2014*]



Can the network be simplified by considering the properties of images?

WHY CNN FOR IMAGE

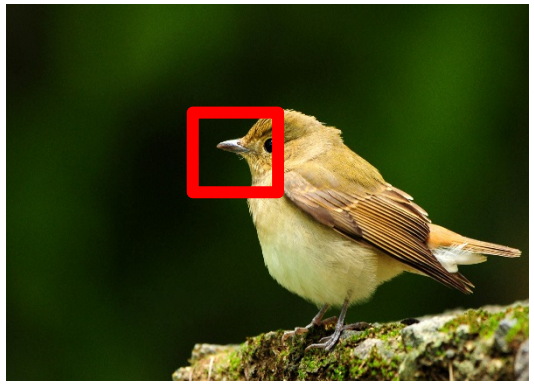
- Some patterns are much smaller than the whole image

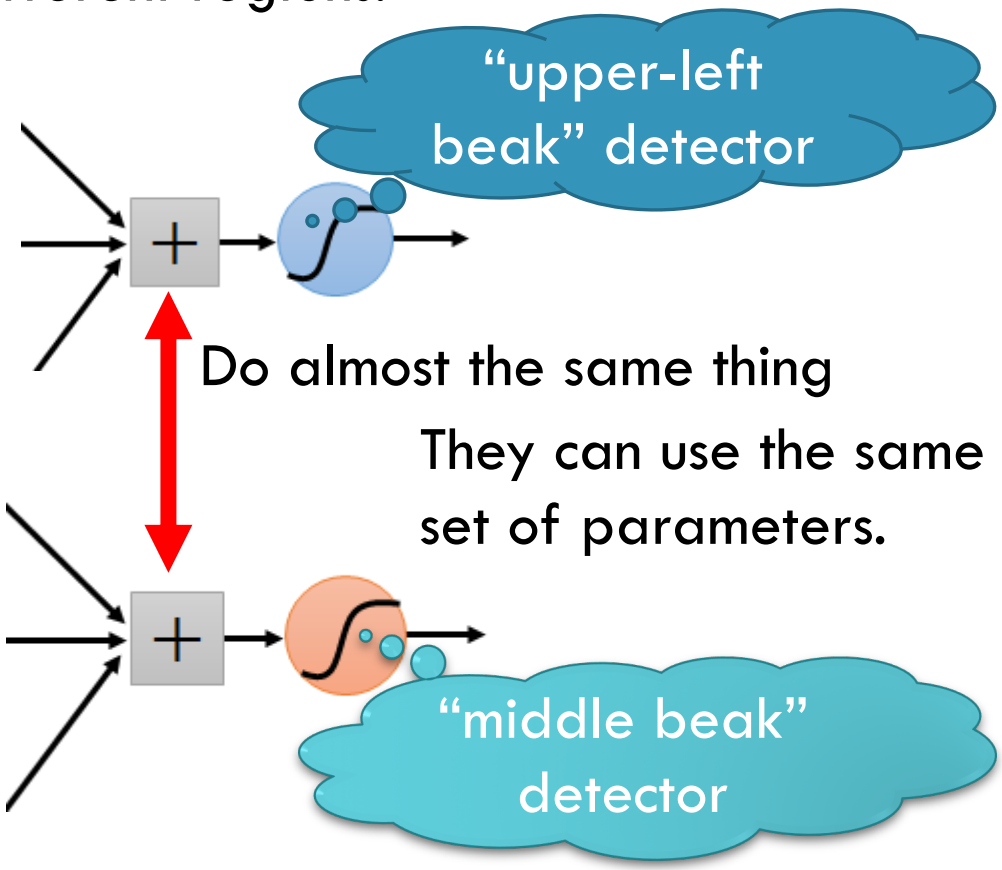
A neuron does not have to see the whole image to discover the pattern.

Connecting to small region with less parameters

WHY CNN FOR IMAGE

- The same patterns appear in different regions.





WHY CNN FOR IMAGE

- Subsampling the pixels will not change the object

bird

subsampling

bird

We can subsample the pixels to make image smaller

→ Less parameters for the network to process the image

THE WHOLE CNN

Convolution

Max Pooling

Convolution

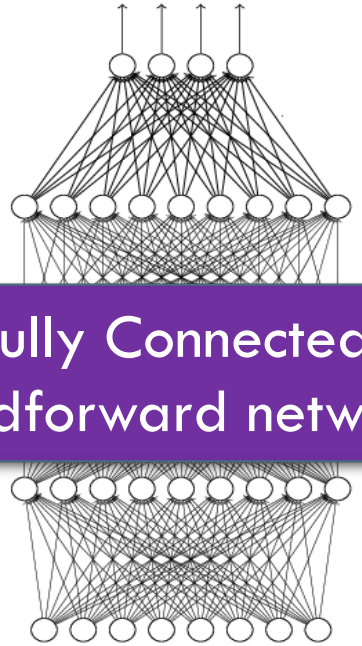
Max Pooling

Can repeat many times

Flatten

Fully Connected Feedforward network

cat dog ……



THE WHOLE CNN

Property 1

- Some patterns are much smaller than the whole image

Property 2

- The same patterns appear in different regions.

Property 3

- Subsampling the pixels will not change the object

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

THE WHOLE CNN

Convolution

Max Pooling

Convolution

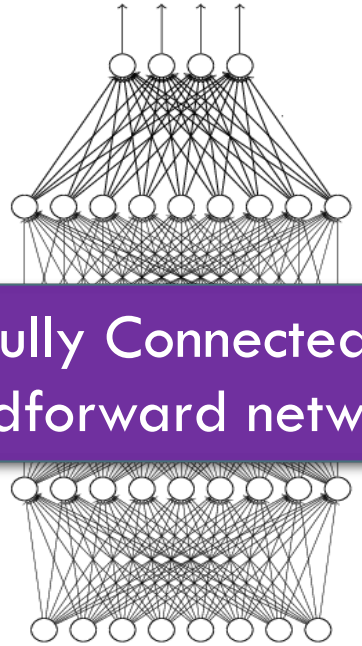
Max Pooling

Can repeat many times

Flatten

Fully Connected Feedforward network

cat dog ……



CNN – CONVOLUTION

These are the network parameters to be learned.

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

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

Matrix

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

Matrix

⋮ ⋮

Property 1

Each filter detects a small pattern (3 x 3).

CNN – CONVOLUTION

stride=1

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

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

3 -1

CNN – CONVOLUTION

If stride=2

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

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

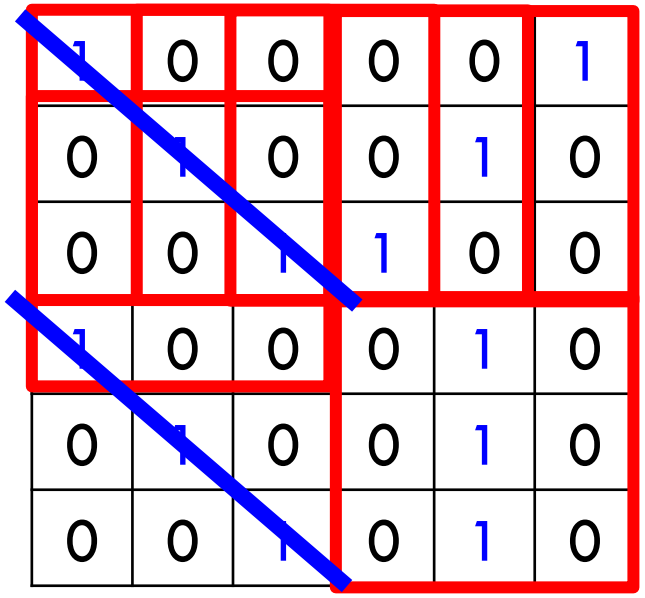
Filter 1

3 -3

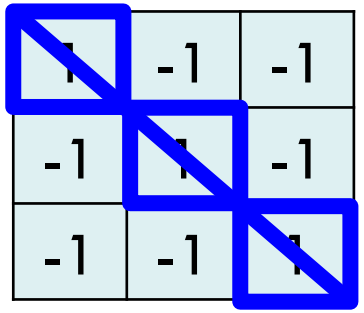
We set stride=1 below

CNN – CONVOLUTION

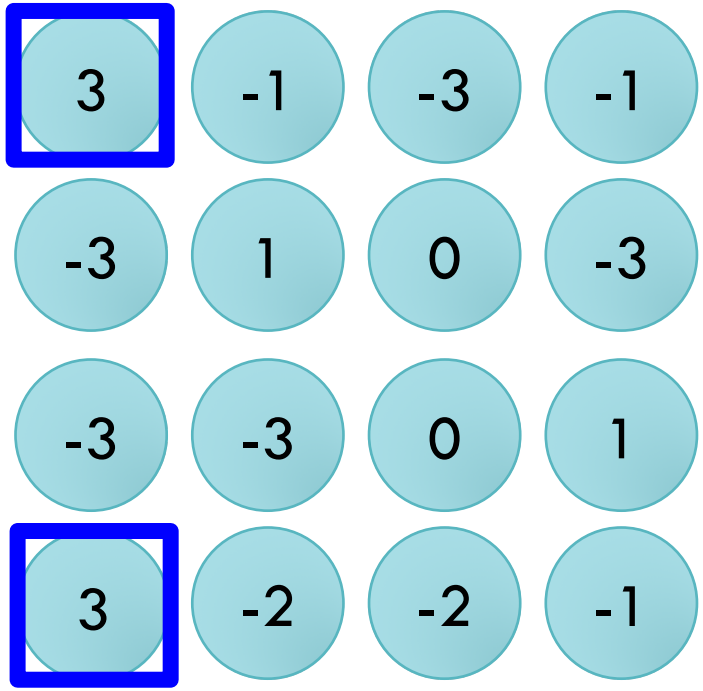
stride=1



6 x 6 image



Filter 1



Property 2

CNN – CONVOLUTION

stride=1

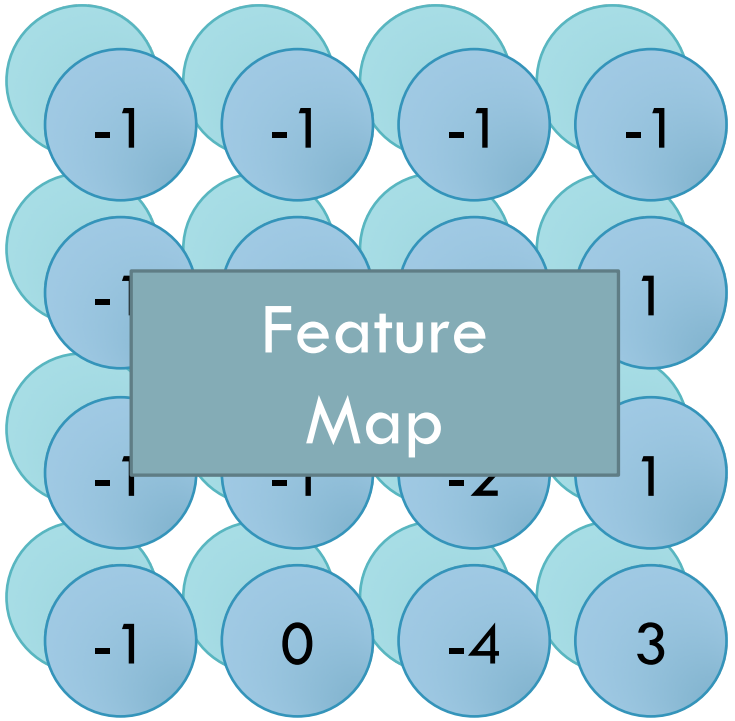
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

Do the same process for every filter

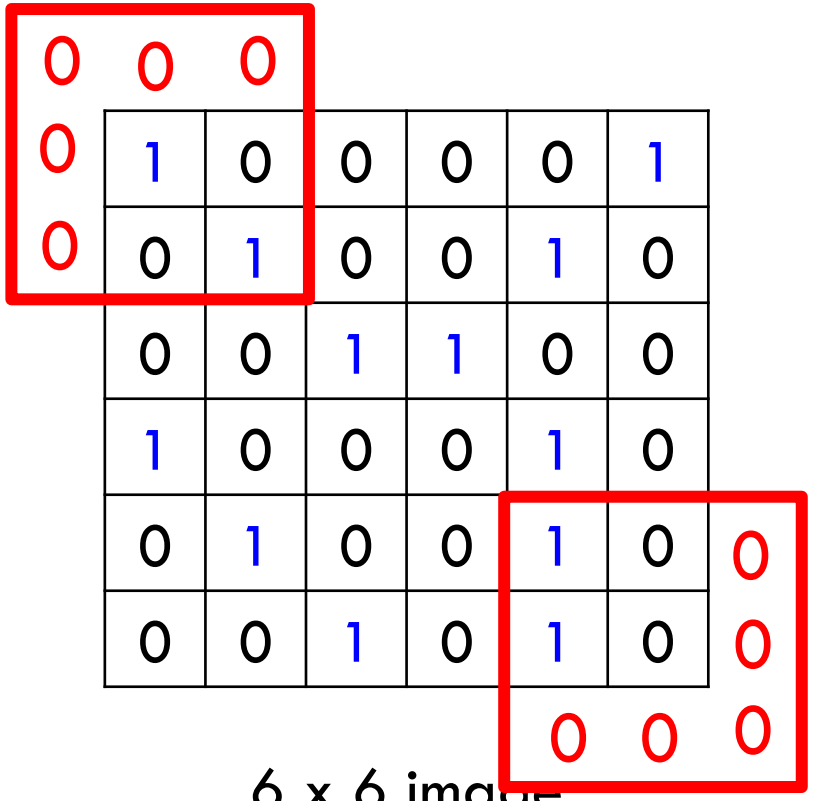


4 x 4 image

CNN – ZERO PADDING

1	-1	-1
-1	1	-1
-1	-1	1

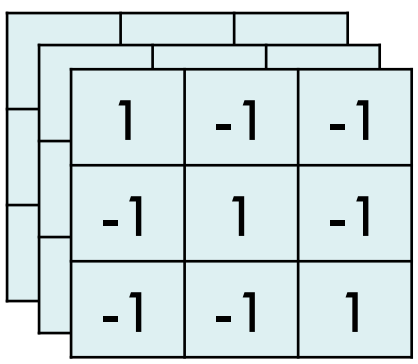
Filter 1



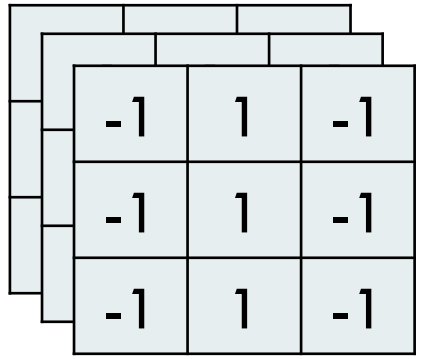
You will get another 6 x 6 images in this way

→ Zero padding

CNN – COLORFUL IMAGE

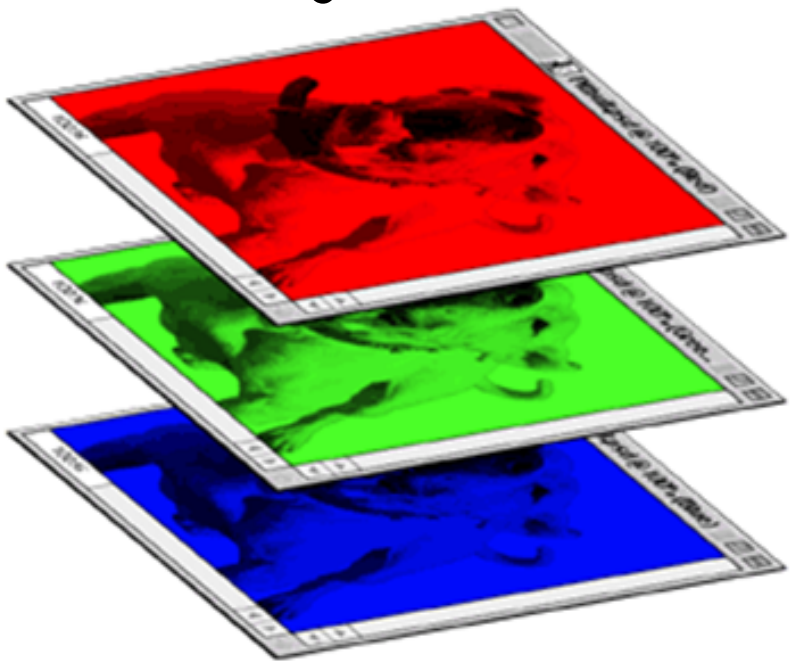


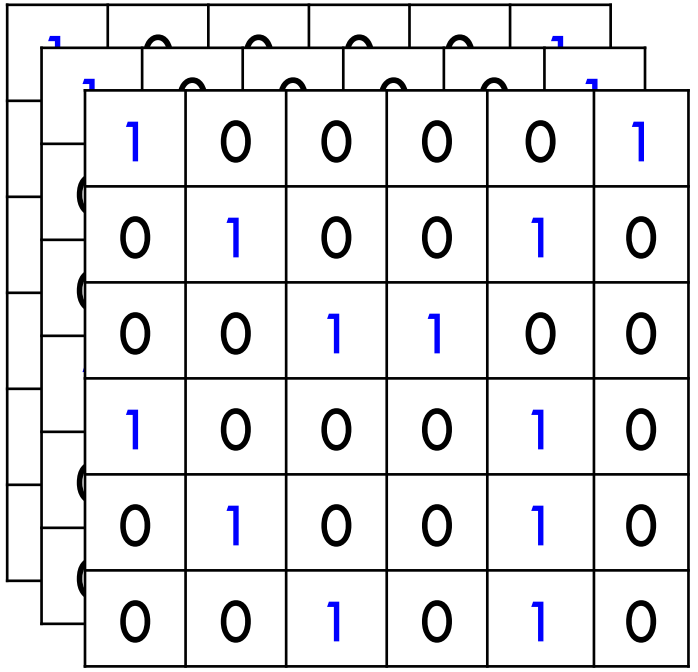
Filter 1



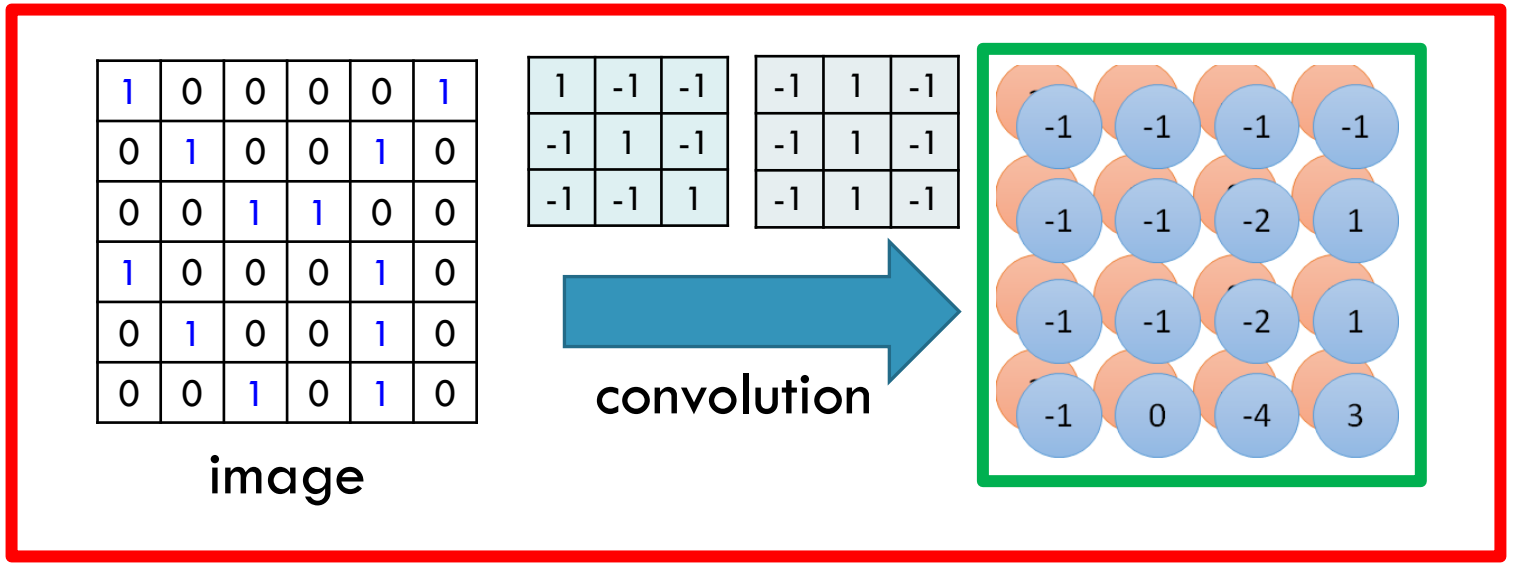
Filter 2

Colorful image

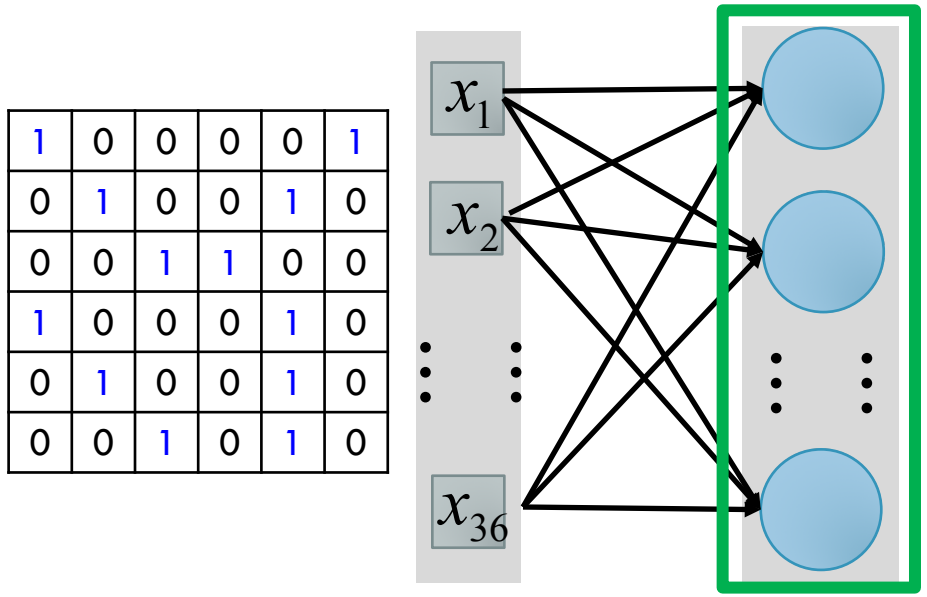


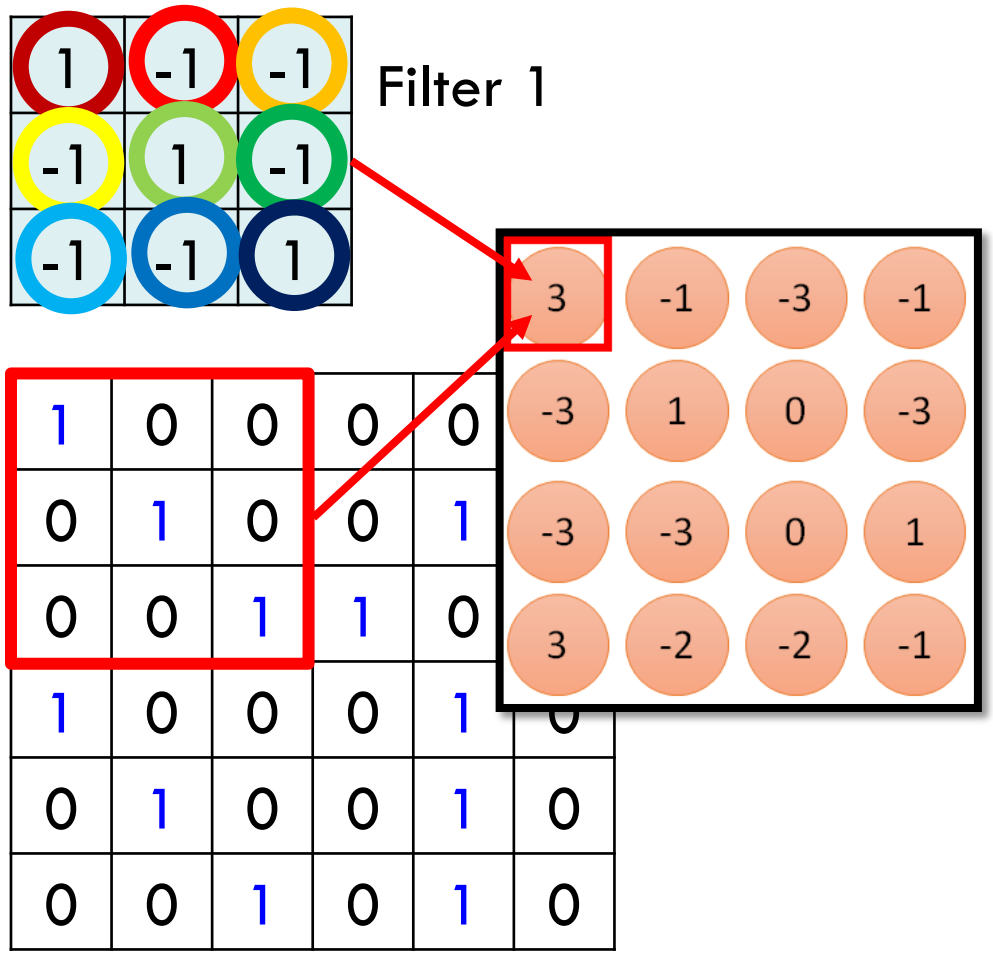


Convolution v.s. Fully Connected

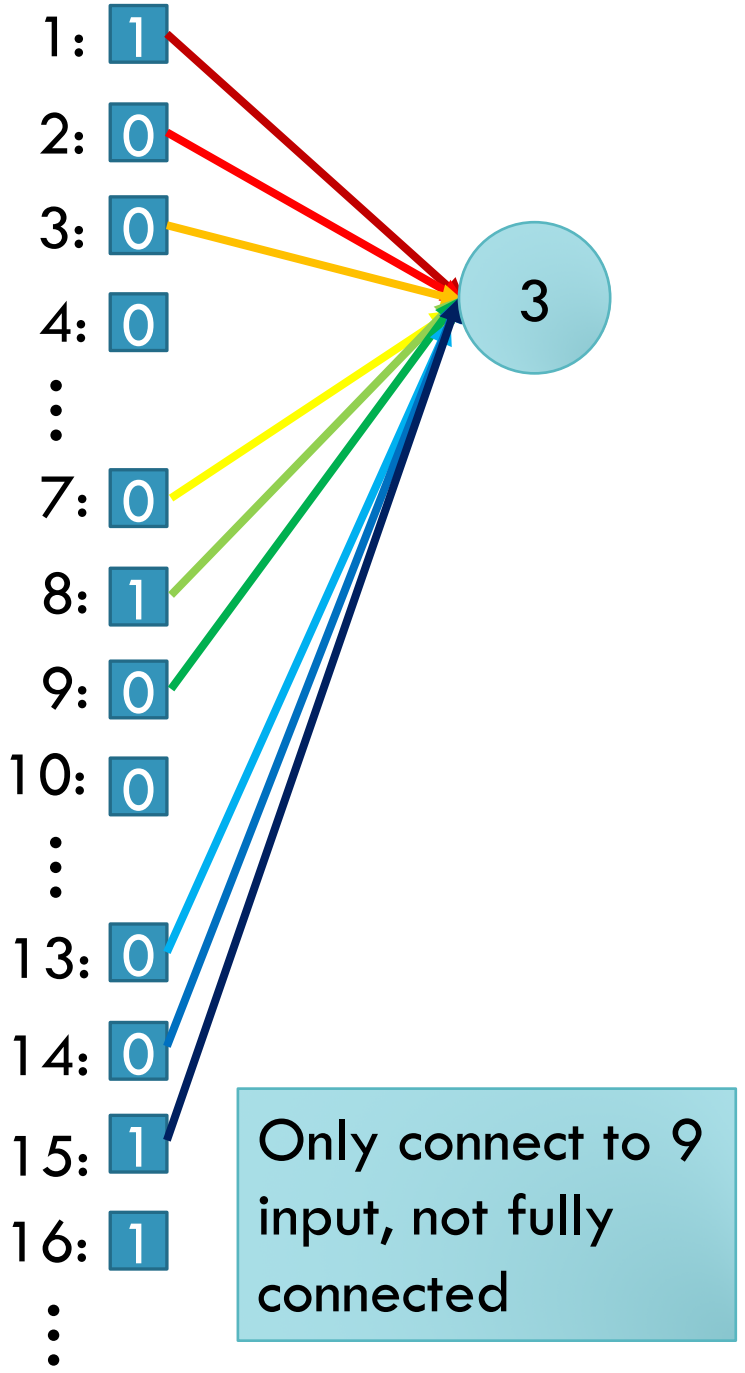


Fully-connected

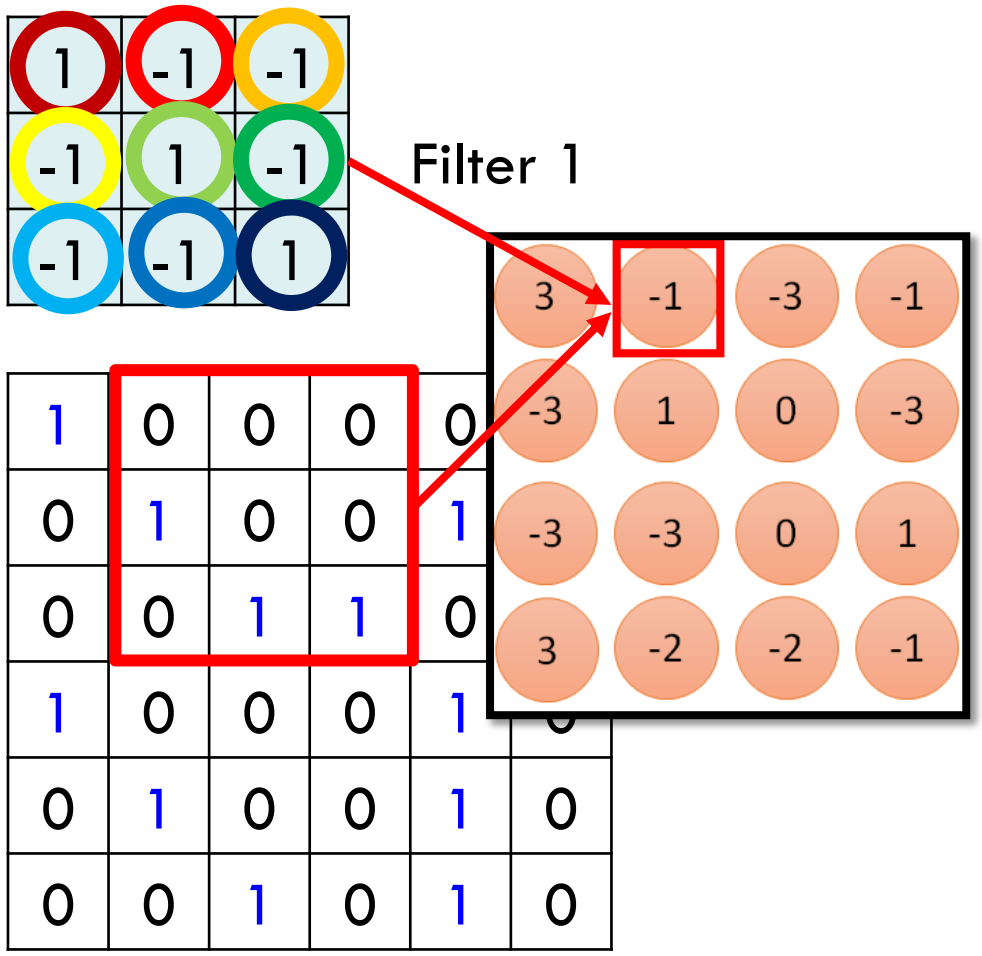




Less parameters!

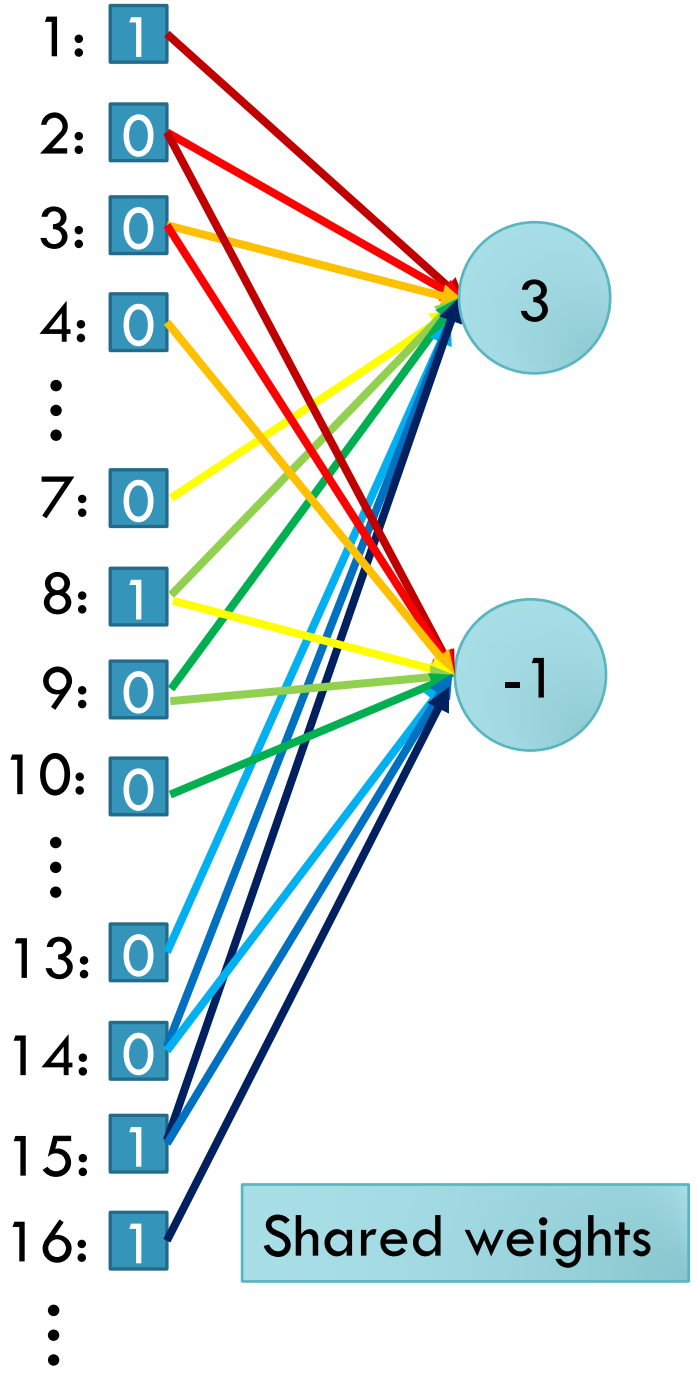


Only connect to 9 input, not fully connected



Less parameters!

Even less parameters!



THE WHOLE CNN

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

Fully Connected Feedforward network

cat dog ……

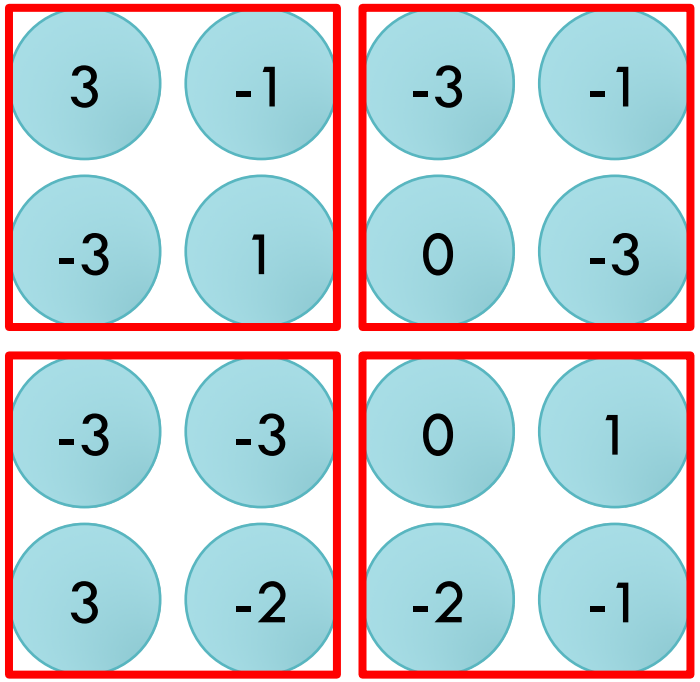
CNN – MAX POOLING

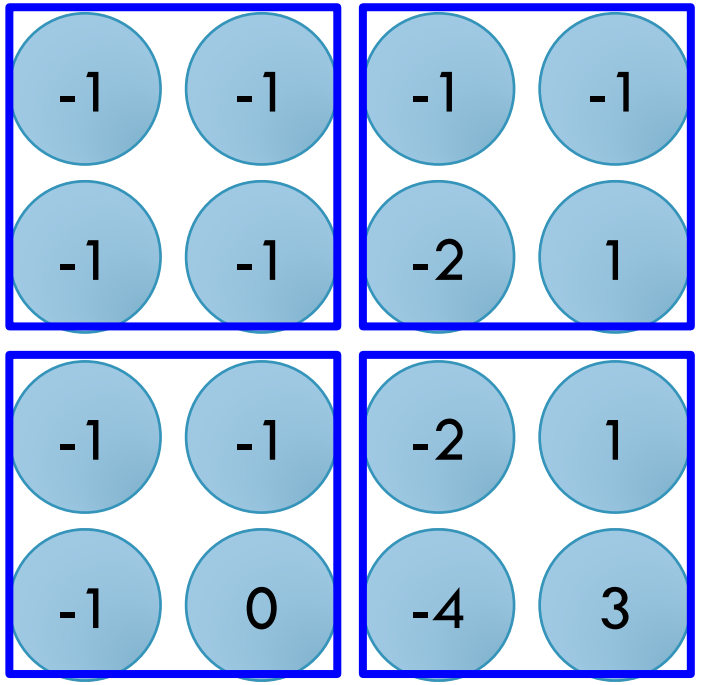
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

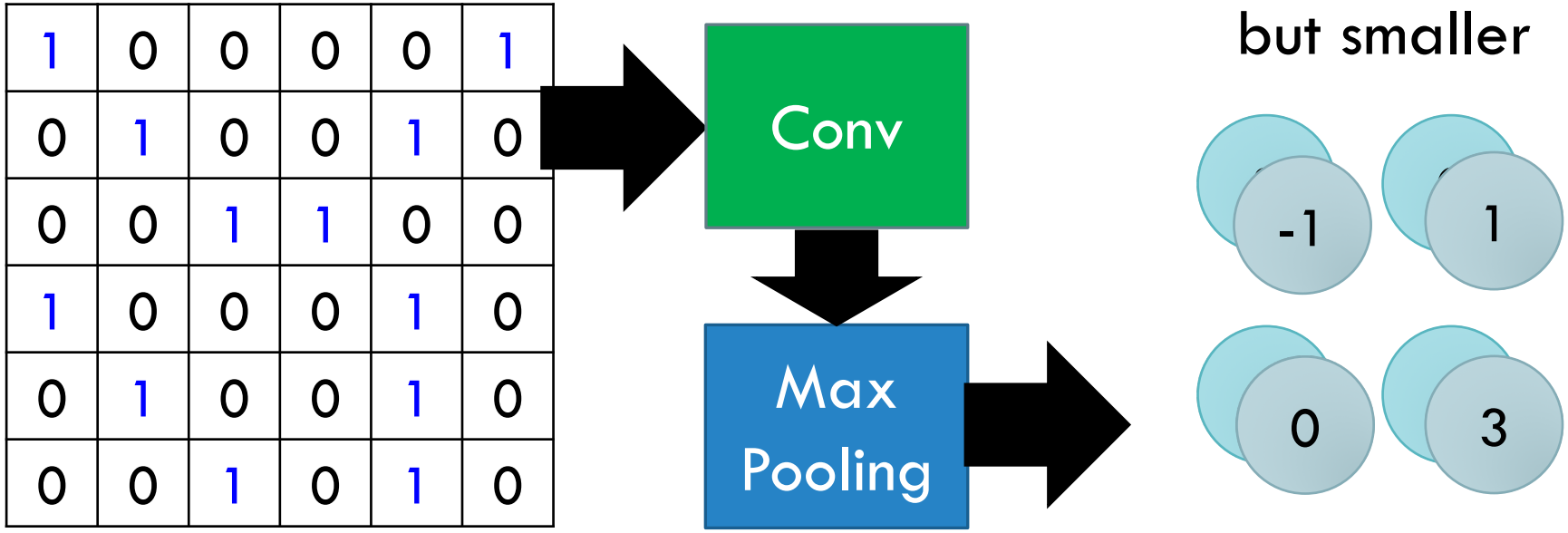
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

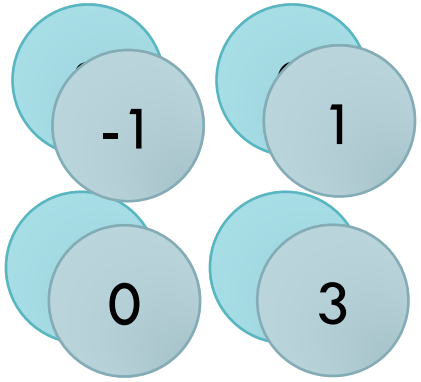




CNN – MAX POOLING



THE WHOLE CNN



A new image

Convolution

Max Pooling

Convolution

Max Pooling

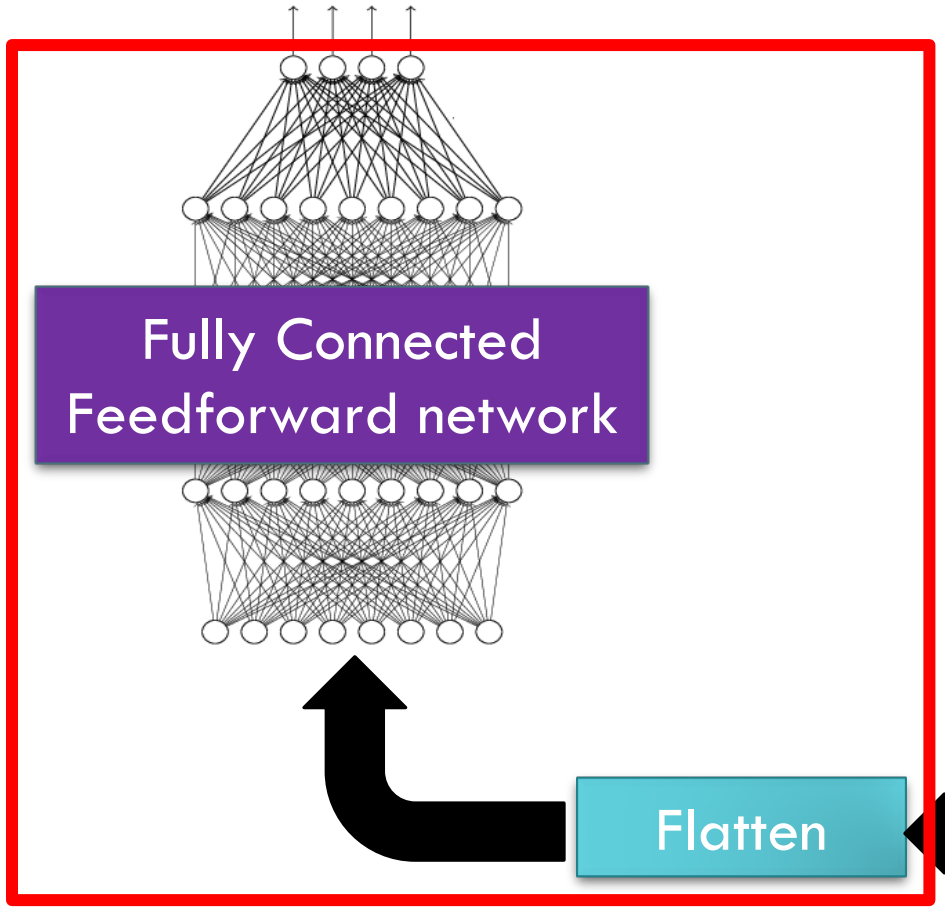
Can repeat many times

Smaller than the original image

The number of the channel is the number of filters

THE WHOLE CNN

cat dog ……



Convolution

Max Pooling

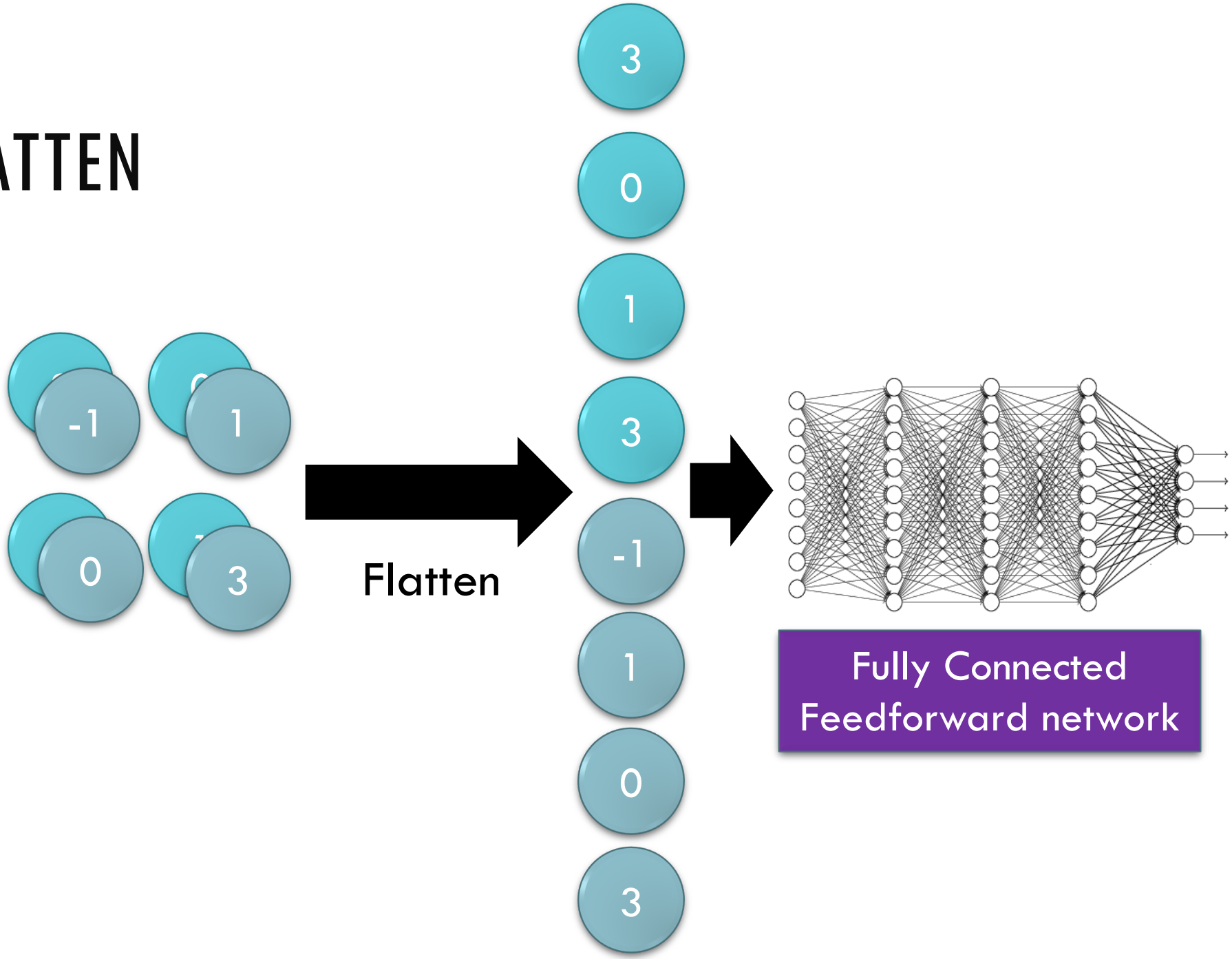
A new image

Convolution

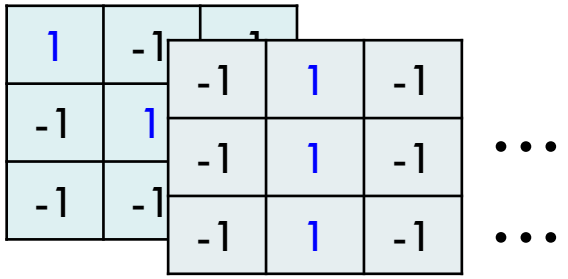
Max Pooling

A new image

FLATTEN


-1
1
0
3
Flatten
3
0
1
3
-1
1
0
3
Fully Connected
Feedforward network

```
model.add(Conv2D(filters = 25,
kernel_size = (3, 3),
padding = 'same',
activation = 'relu',
input_shape = (28, 28, 1)))
```

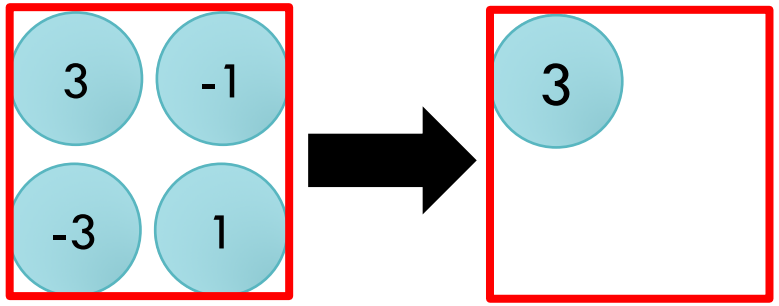


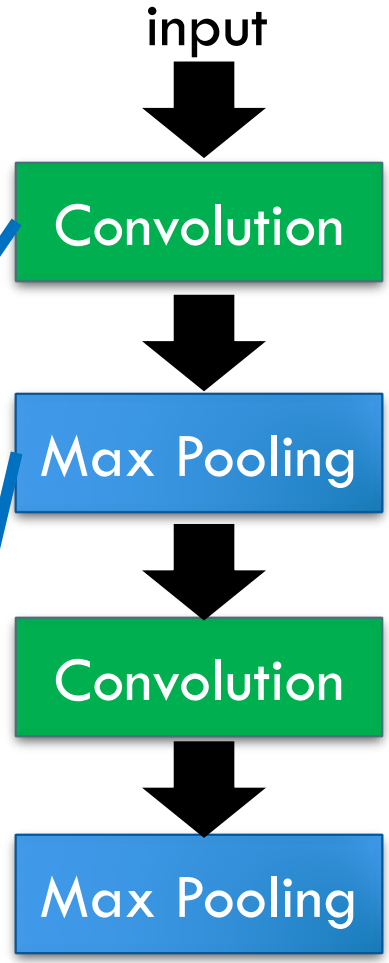
...

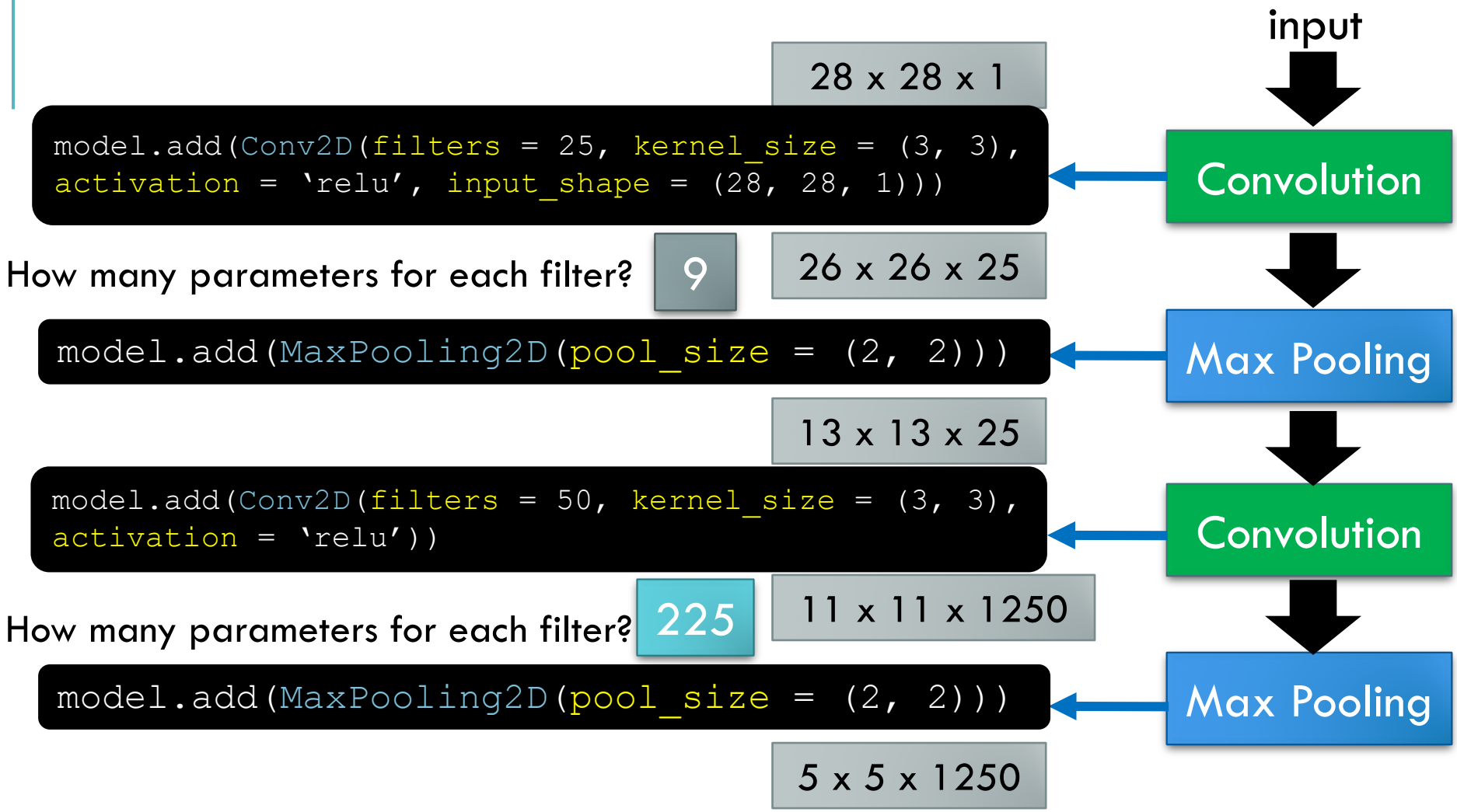
...

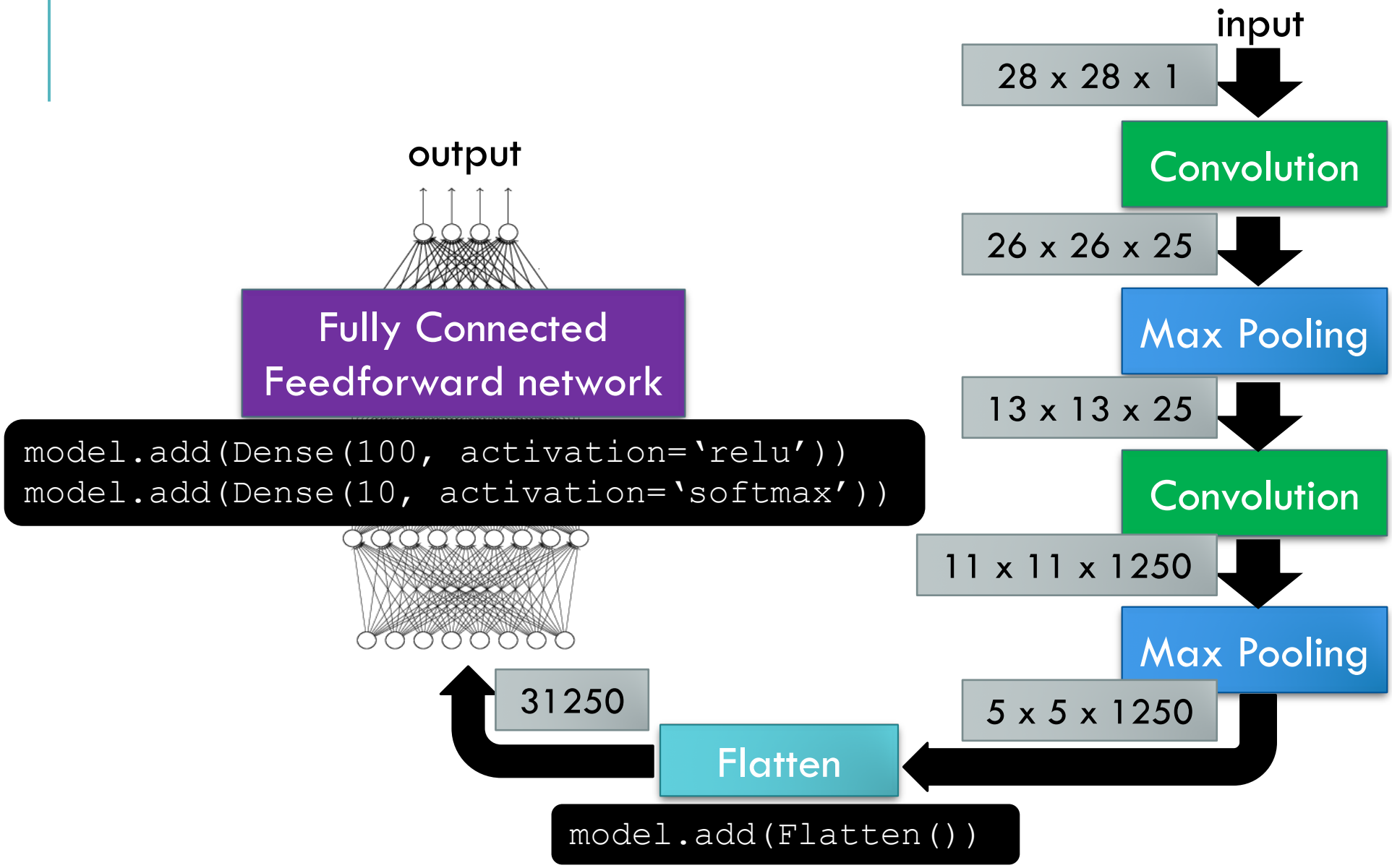
There are 25 3x3 filters.

```
model.add(MaxPooling2D(pool_size = (2, 2)))
```



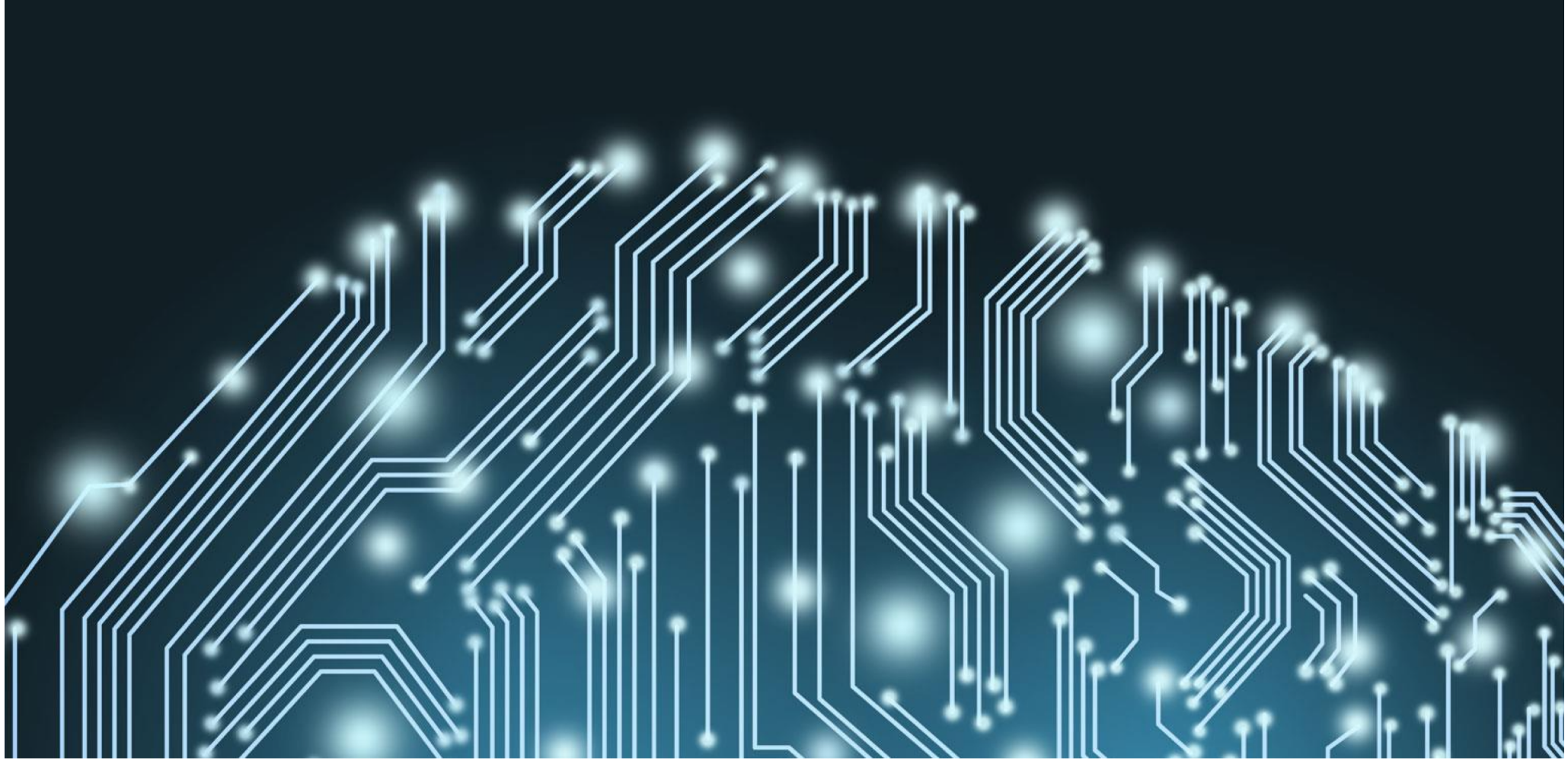



input
28 x 28 x 1
model.add(Conv2D(filters = 25, kernel_size = (3, 3), activation = 'relu', input_shape = (28, 28, 1)))
Convolution
How many parameters for each filter?
9
26 x 26 x 25
model.add(MaxPooling2D(pool_size = (2, 2)))
Max Pooling
13 x 13 x 25
model.add(Conv2D(filters = 50, kernel_size = (3, 3), activation = 'relu'))
Convolution
How many parameters for each filter?
225
11 x 11 x 1250
model.add(MaxPooling2D(pool_size = (2, 2)))
Max Pooling
5 x 5 x 1250


input
28 x 28 x 1
Convolution
26 x 26 x 25
Max Pooling
13 x 13 x 25
Convolution
11 x 11 x 1250
Max Pooling
5 x 5 x 1250
Flatten
model.add(Flatten())
31250
output
Fully Connected
Feedforward network
model.add(Dense(100, activation='relu'))
model.add(Dense(10, activation='softmax'))

CLASS EXERCISE

bit.ly/cnn-15



QUESTIONS?