Convolutional Neural Network

Hung-yi Lee

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

Why CNN for Image

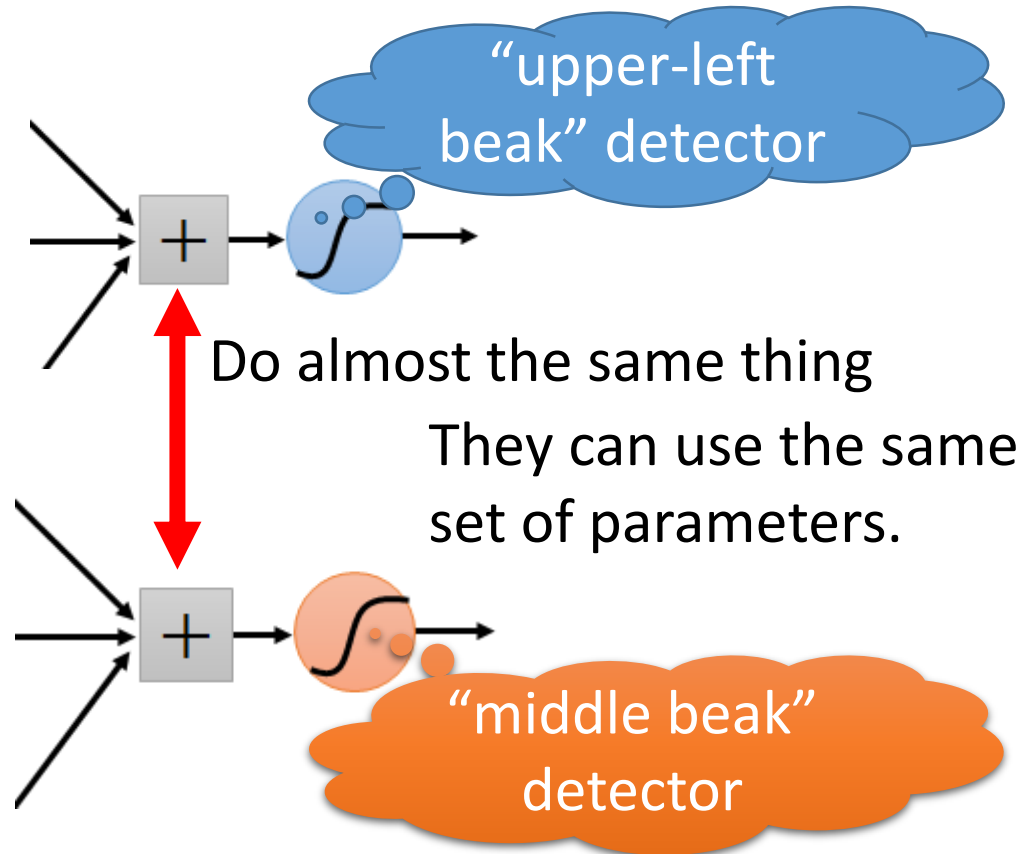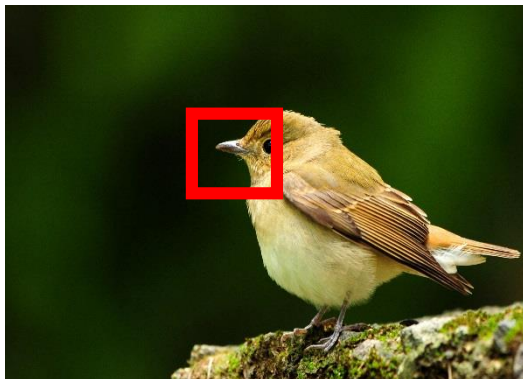- Some patterns are much smaller than the whole image

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

Connecting to small region with less parameters

"beak" detector

Why CNN for Image

- The same patterns appear in different regions.

"upper-left beak" detector

Do almost the same thing

They can use the same set of parameters.

"middle beak" detector

Why CNN for Image

- Subsampling the pixels will not change the object
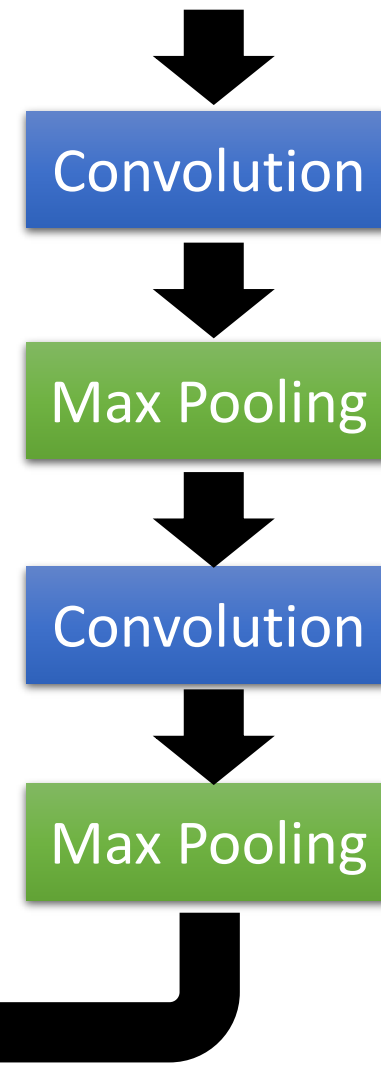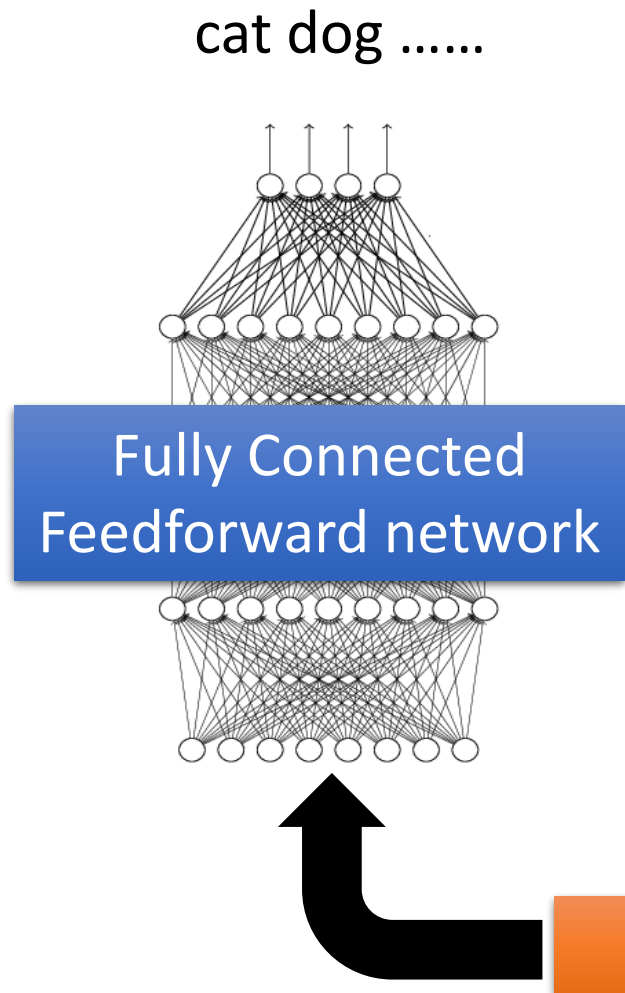
bird

subsampling

bird

We can subsample the pixels to make image smaller

➡ Less parameters for the network to process the image

The whole CNN

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
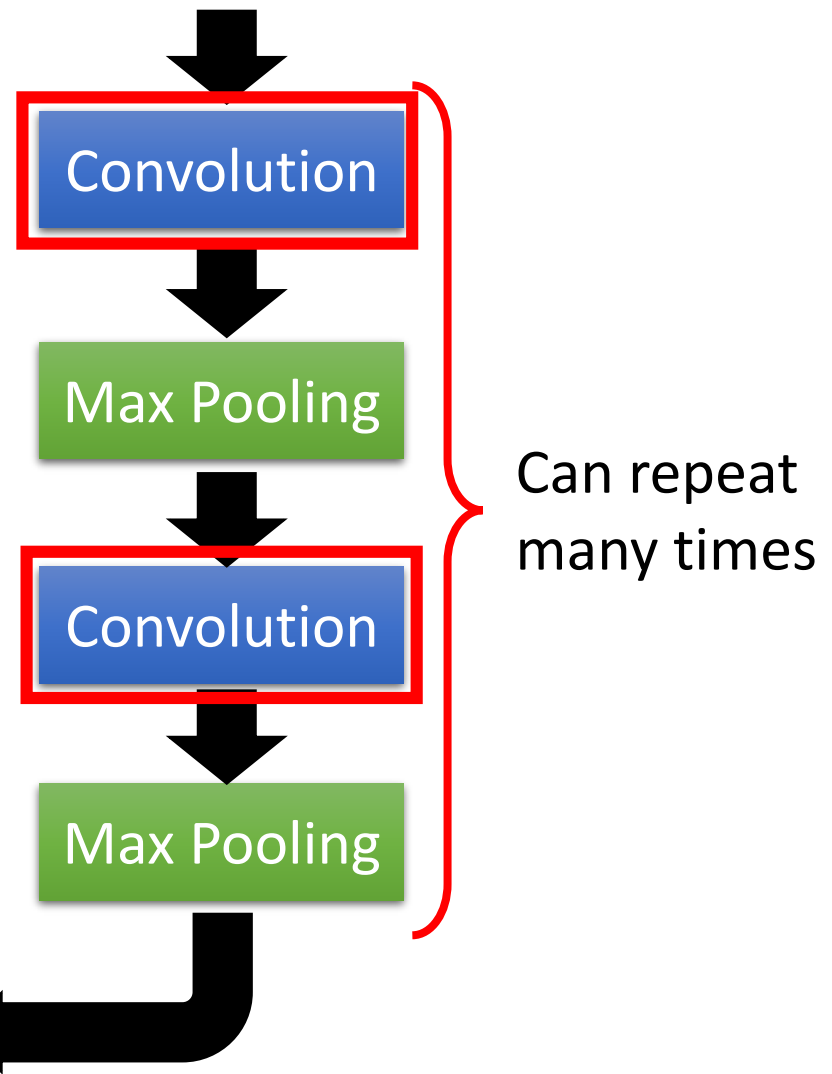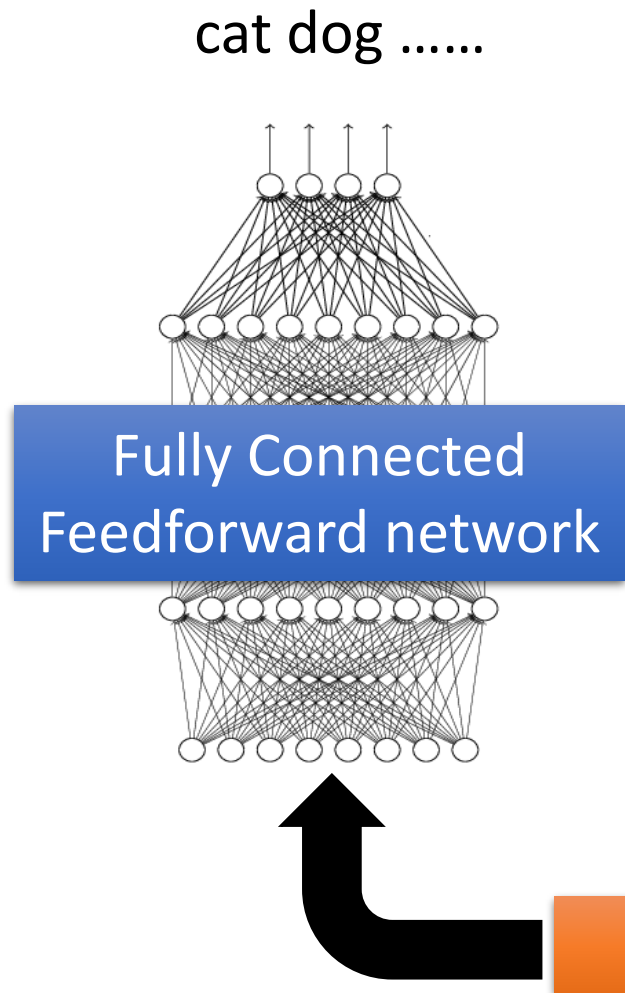
Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

The whole CNN

CNN – Convolution

Those 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

Filter 1

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

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

3   -1
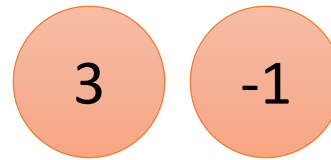
CNN – Convolution

Filter 1

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

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

3   -3
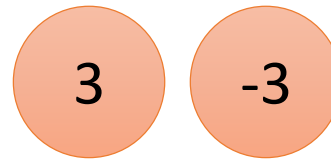
We set stride=1 below

CNN – Convolution

Filter 1

stride=1

6 x 6 image

Property 2

CNN – Convolution

Filter 2

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

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

Do the same process for every filter

-1	-1	-1	-1
-1			1
-1		-2	1
-1	0	-4	3

Feature Map
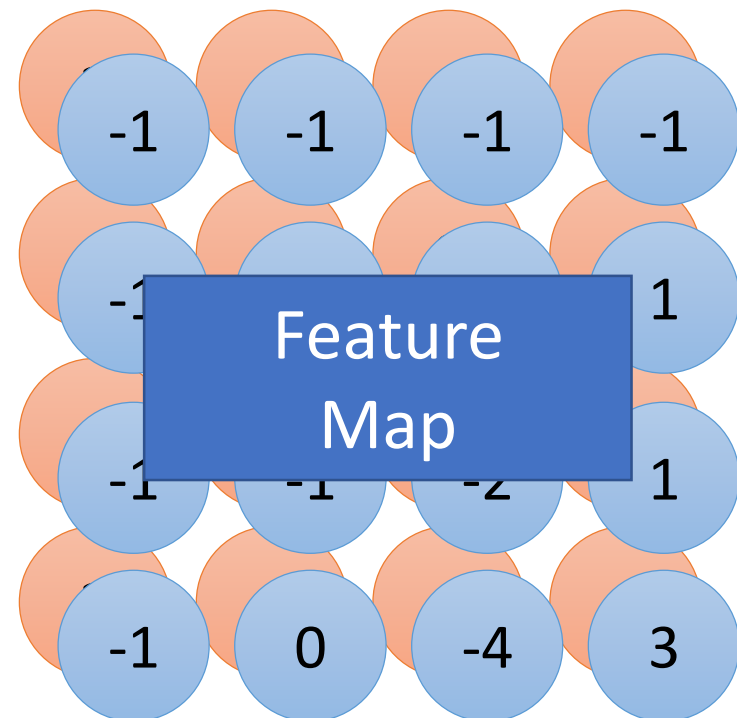
4 x 4 image

CNN – Colorful image

Colorful image

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

Filter 1

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

Filter 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
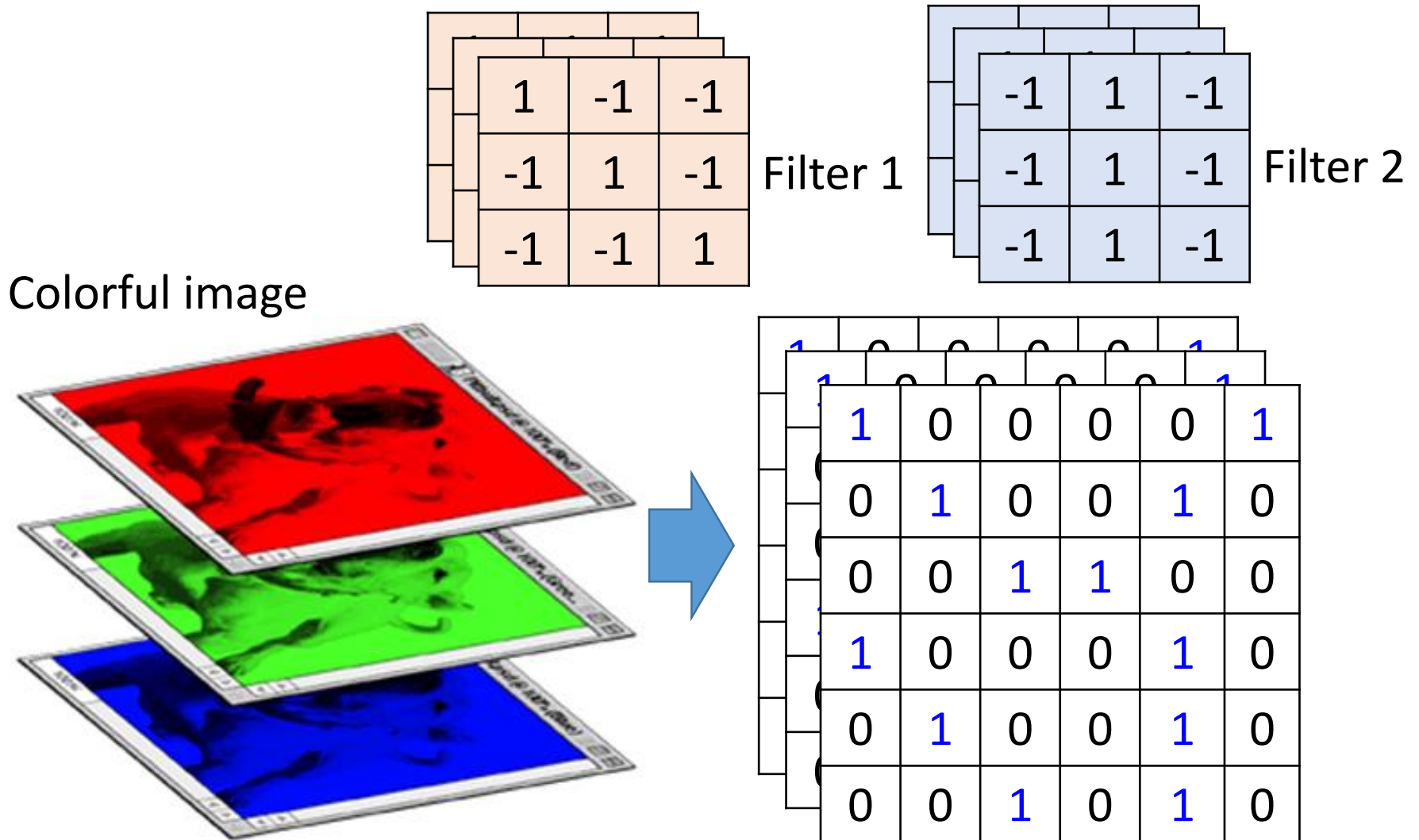
Convolution v.s. Fully Connected
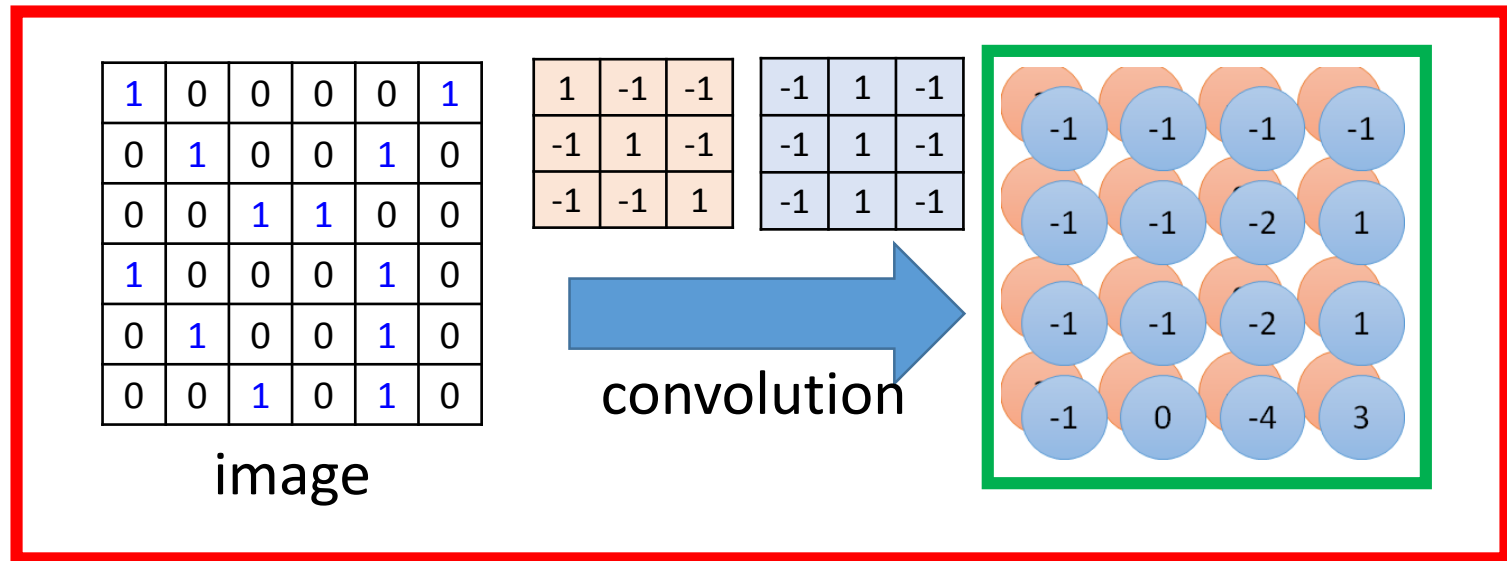
image

convolution

Fully-connected

Filter 1

6 x 6 image

Less parameters!

1: 1
2: 0
3: 0
4: 0
⋮
7: 0
8: 1
9: 0
10: 0
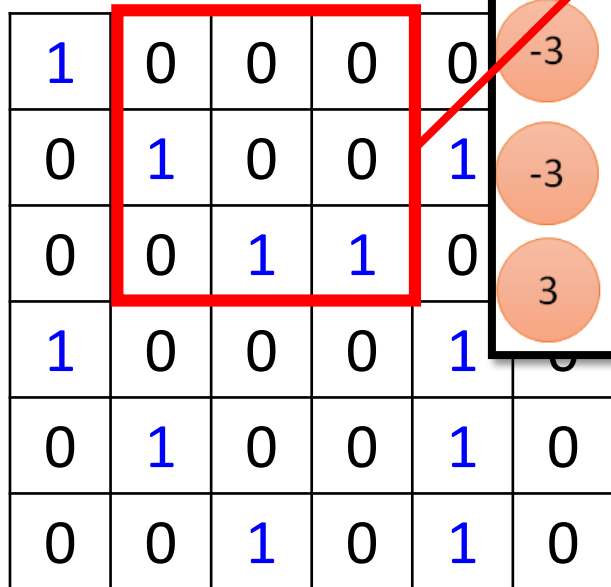⋮
13: 0
14: 0
15: 1
16: 1
⋮

3

Only connect to 9 input, not fully connected
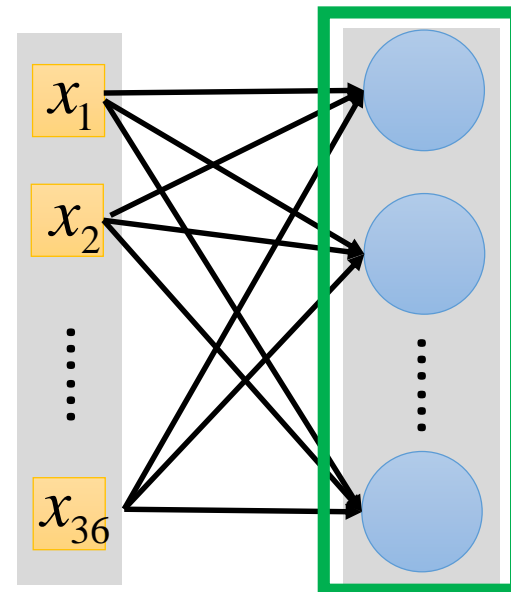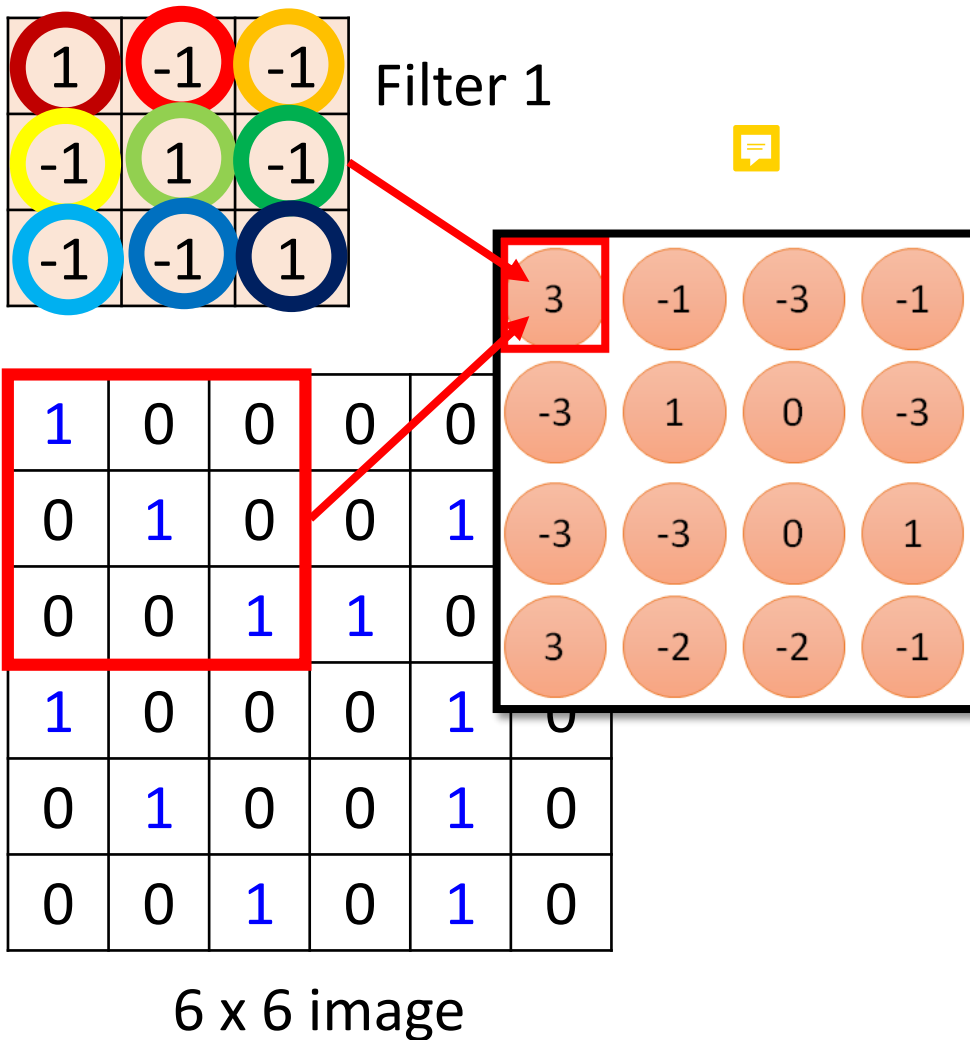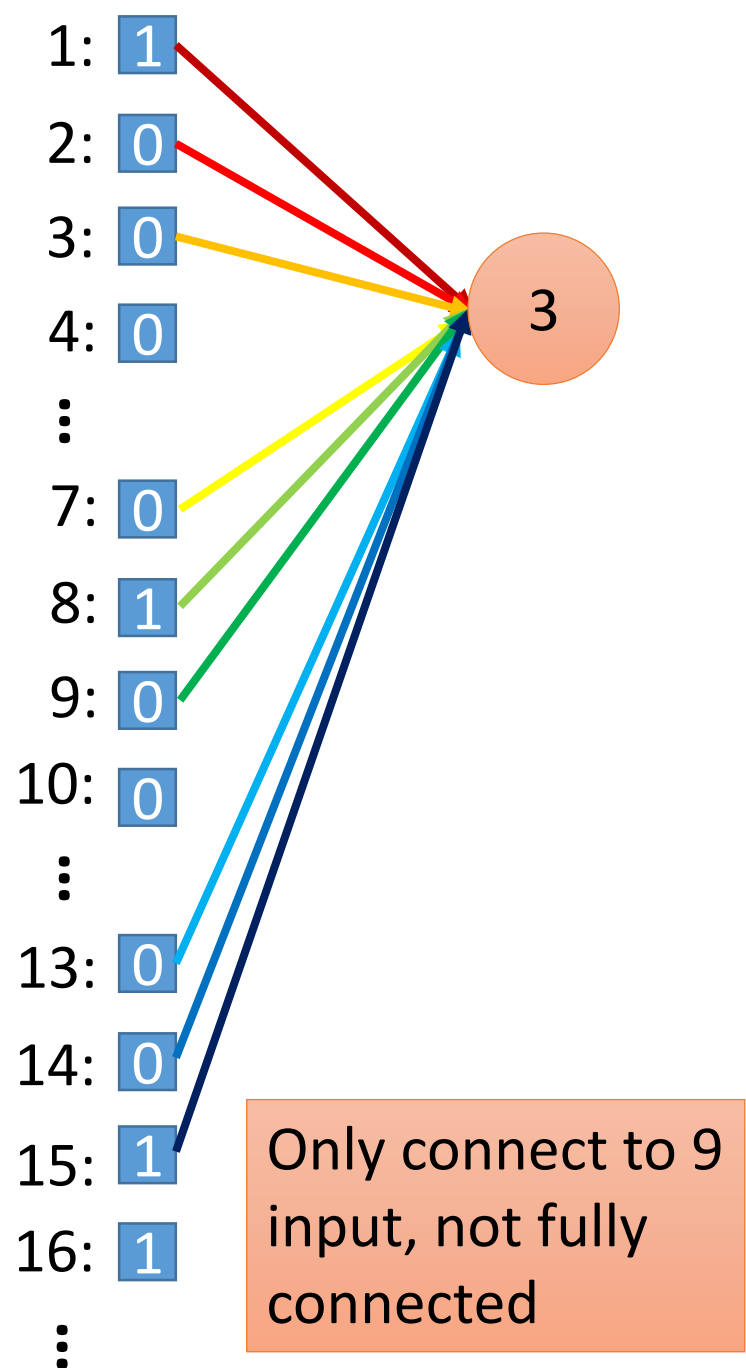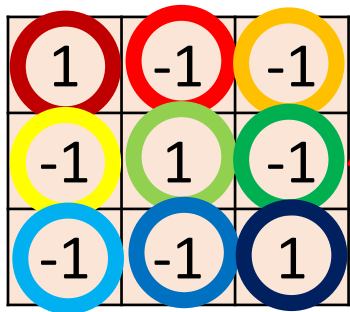
Filter 1

6 x 6 image

Less parameters!

Even less parameters!

1: 1
2: 0
3: 0
4: 0
⋮
7: 0
8: 1
9: 0
10: 0
⋮
13: 0
14: 0
15: 1
16: 1
⋮

3

-1

Shared weights

The whole CNN

cat dog ......

Fully Connected Feedforward network

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times
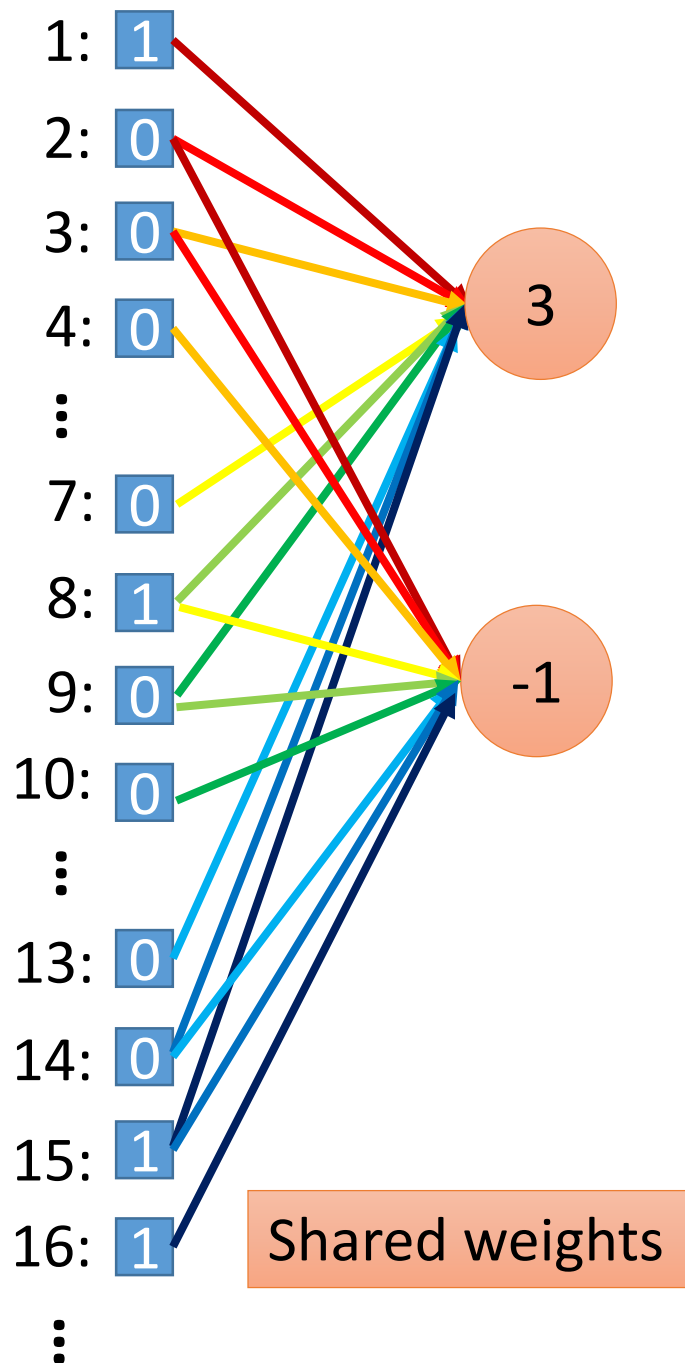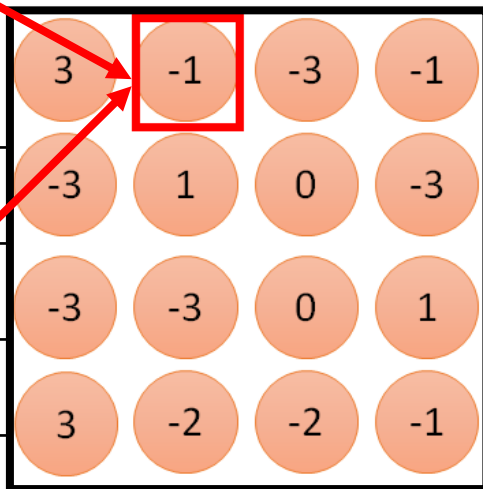
Flatten
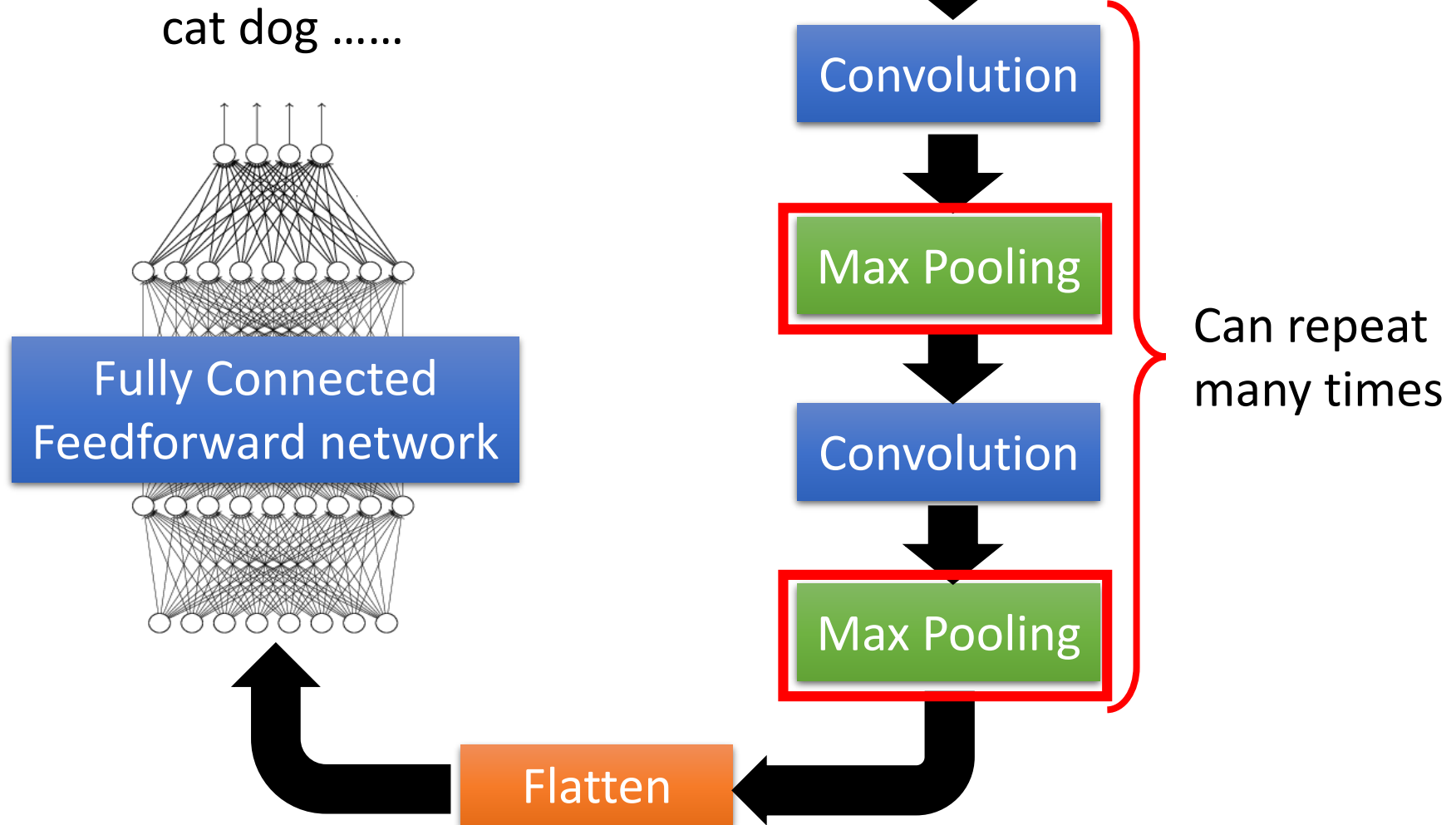
CNN – Max Pooling

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

Filter 1

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

Filter 2

3	-1
-3	1

-3	-1
0	-3

-3	-3
3	-2

0	1
-2	-1

-1	-1
-1	-1

-1	-1
-2	1

-1	-1
-1	0

-2	1
-4	3

CNN – Max Pooling

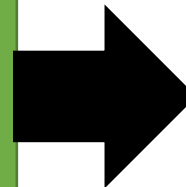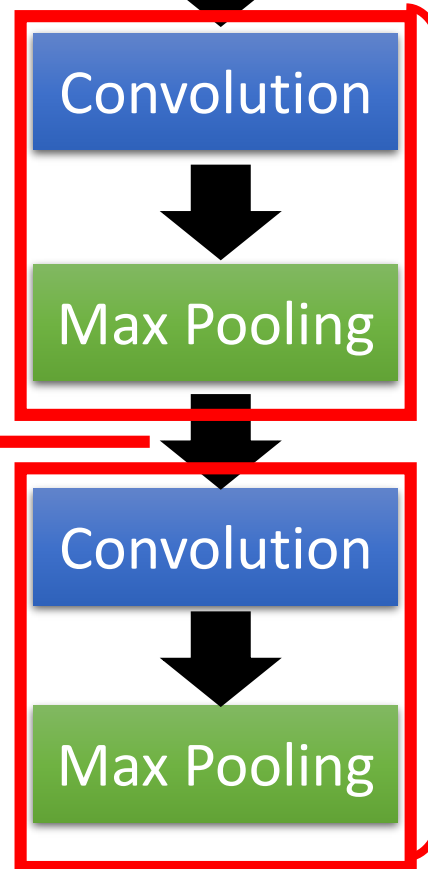| 1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image
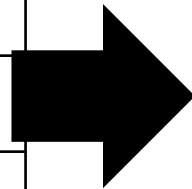
Conv

Max Pooling

New image but smaller

-1	1
0	3
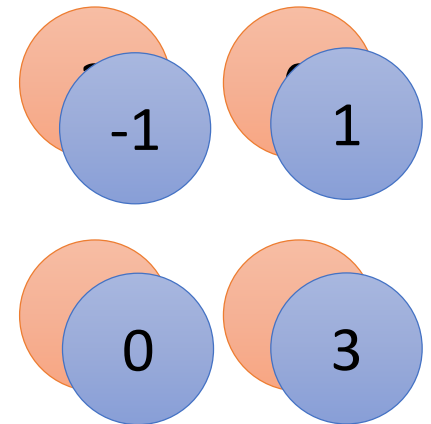
2 x 2 image

Each filter is a channel

The whole CNN

-1	1
0	3

A new image
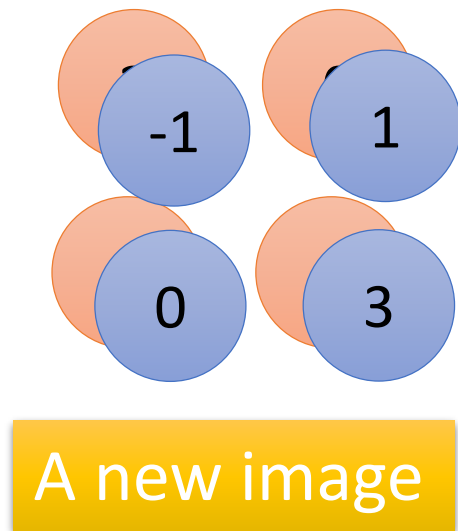
Smaller than the original image

The number of the channel is the number of filters

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

The whole CNN

cat dog ......
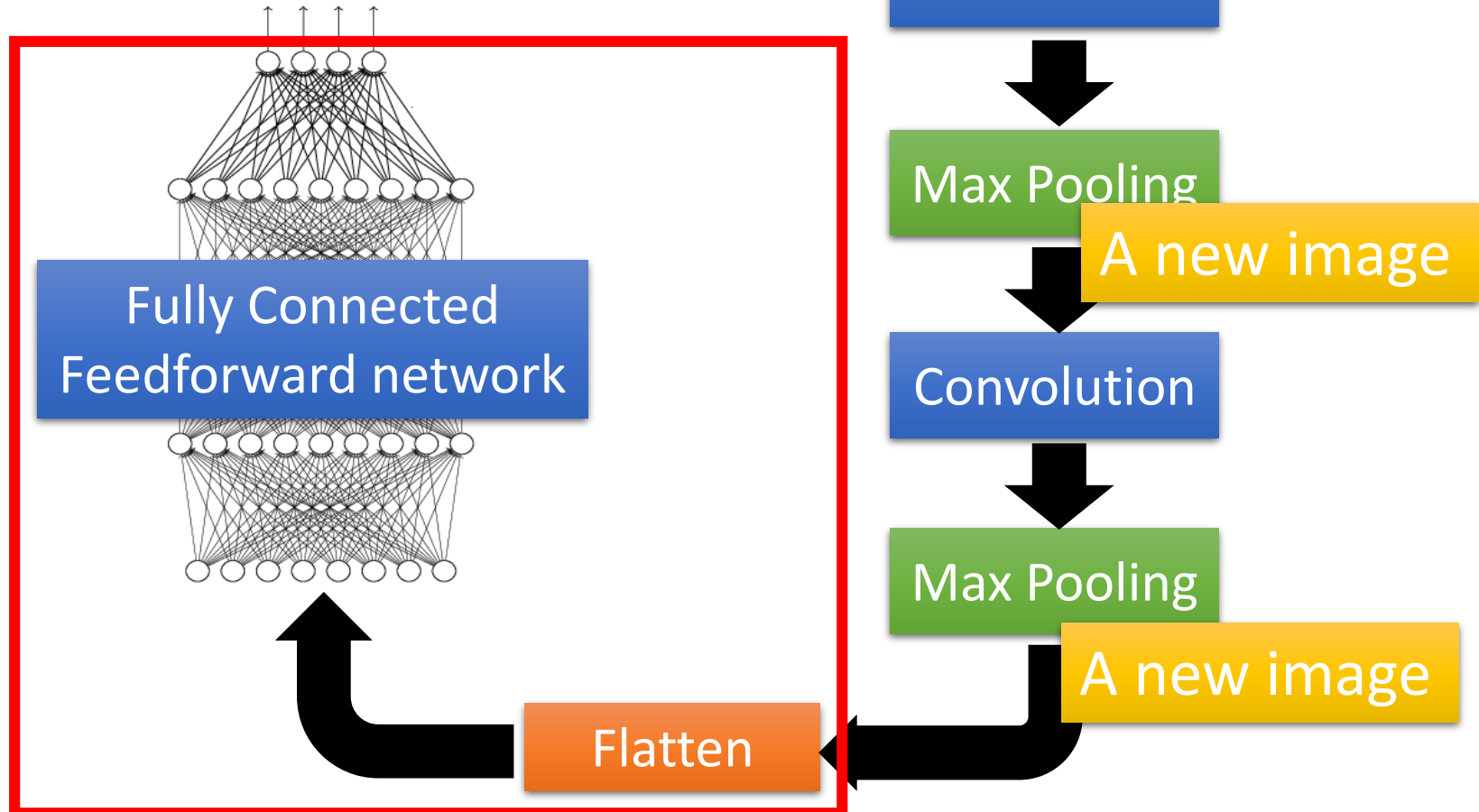
Flatten
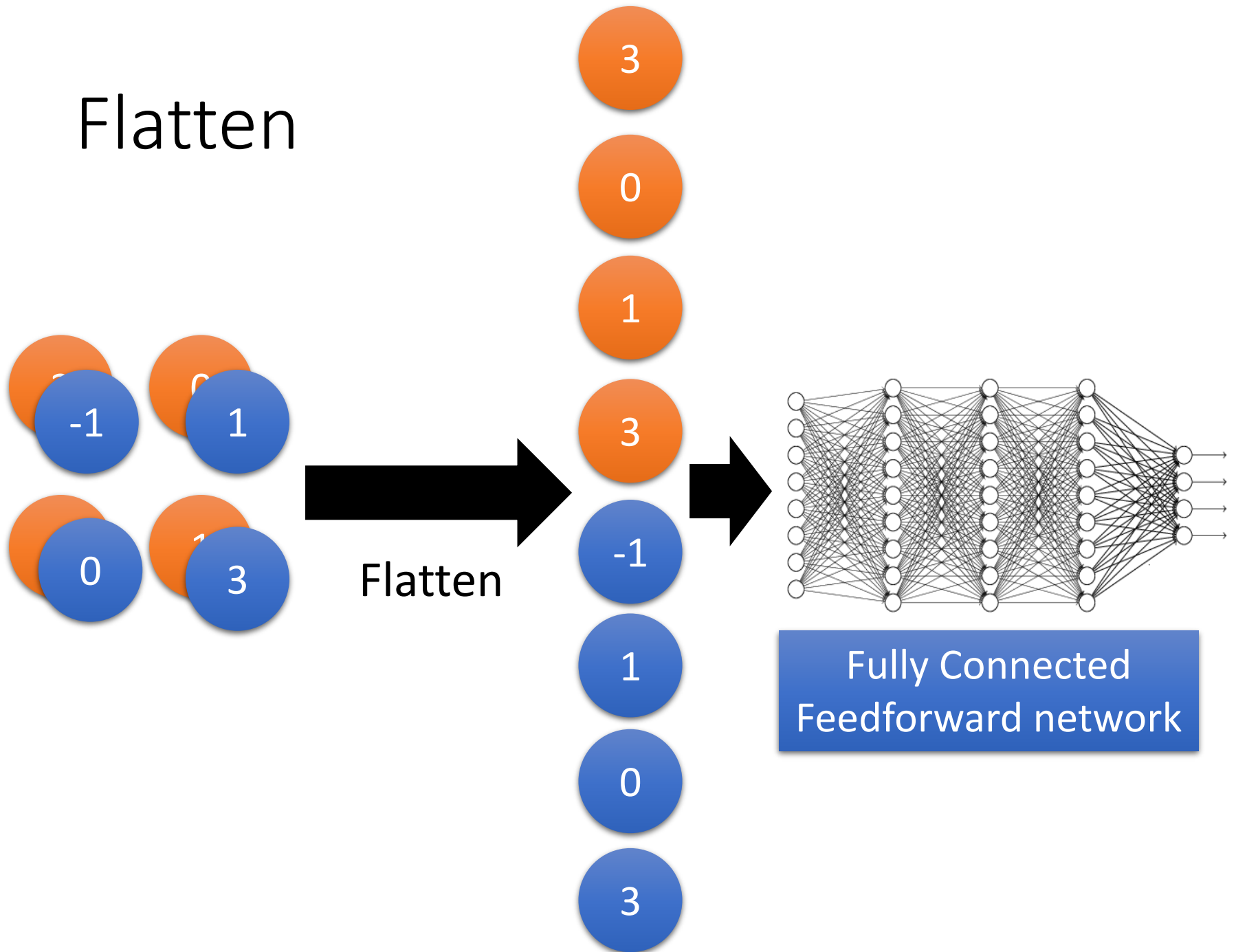
Flatten

Fully Connected Feedforward network

CNN in Keras

input

Convolution

Max Pooling

Convolution

Max Pooling

```
model2.add( Convolution2D( 25,3,3,
            input_shape=(28,28,1)) )
```

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

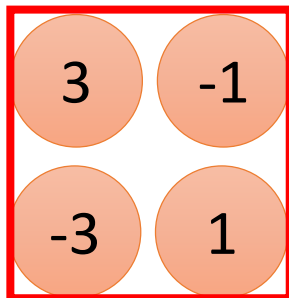|		
-1	1	-1
-1	1	-1
-1	1	-1

......

There are 25 3x3 filters.

Input_shape = (28 , 28 , 1)

28 x 28 pixels   1: black/white, 3: RGB
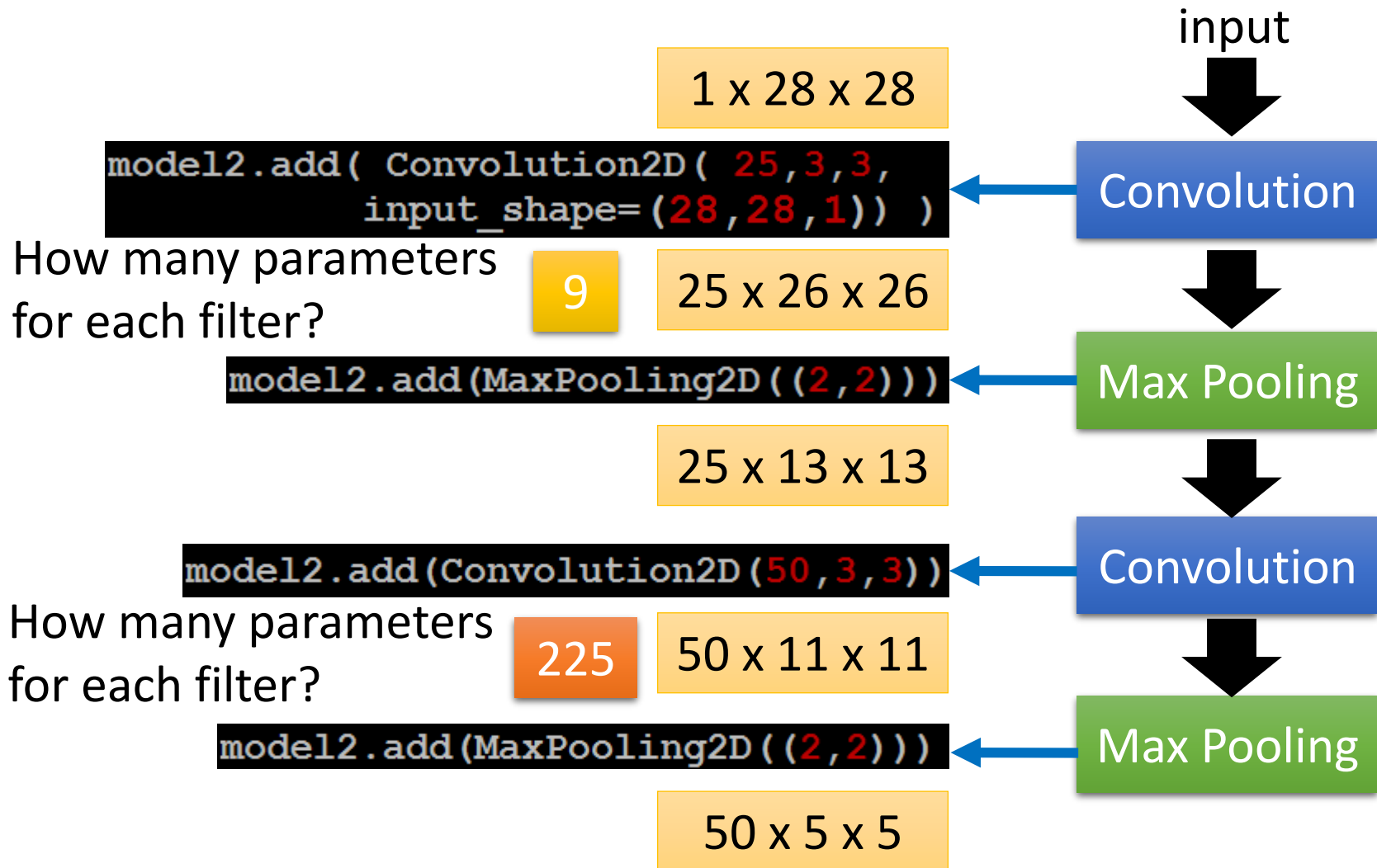
```
model2.add(MaxPooling2D((2,2)))
```

3	-1
-3	1

➡

3

CNN in Keras

Only modified the *network structure* and *input format (vector -> 3-D tensor)*

input

1 x 28 x 28

```
model2.add( Convolution2D( 25,3,3,
        input_shape=(28,28,1)) )
```

Convolution

How many parameters for each filter?

9 · 25 x 26 x 26

```
model2.add(MaxPooling2D((2,2)))
```

Max Pooling

25 x 13 x 13

```
model2.add(Convolution2D(50,3,3))
```

Convolution

How many parameters for each filter?

225 · 50 x 11 x 11

```
model2.add(MaxPooling2D((2,2)))
```

Max Pooling

50 x 5 x 5

CNN in Keras

Only modified the **network structure** and **input format (vector -> 3-D tensor)**

output

Fully Connected Feedforward network

```
model2.add(Dense(output_dim=100))
model2.add(Activation('relu'))
model2.add(Dense(output_dim=10))
model2.add(Activation('softmax'))
```

input

1 x 28 x 28

Convolution

25 x 26 x 26

Max Pooling

25 x 13 x 13

Convolution

50 x 11 x 11

Max Pooling

50 x 5 x 5

1250

Flatten

```
model2.add(Flatten())
```
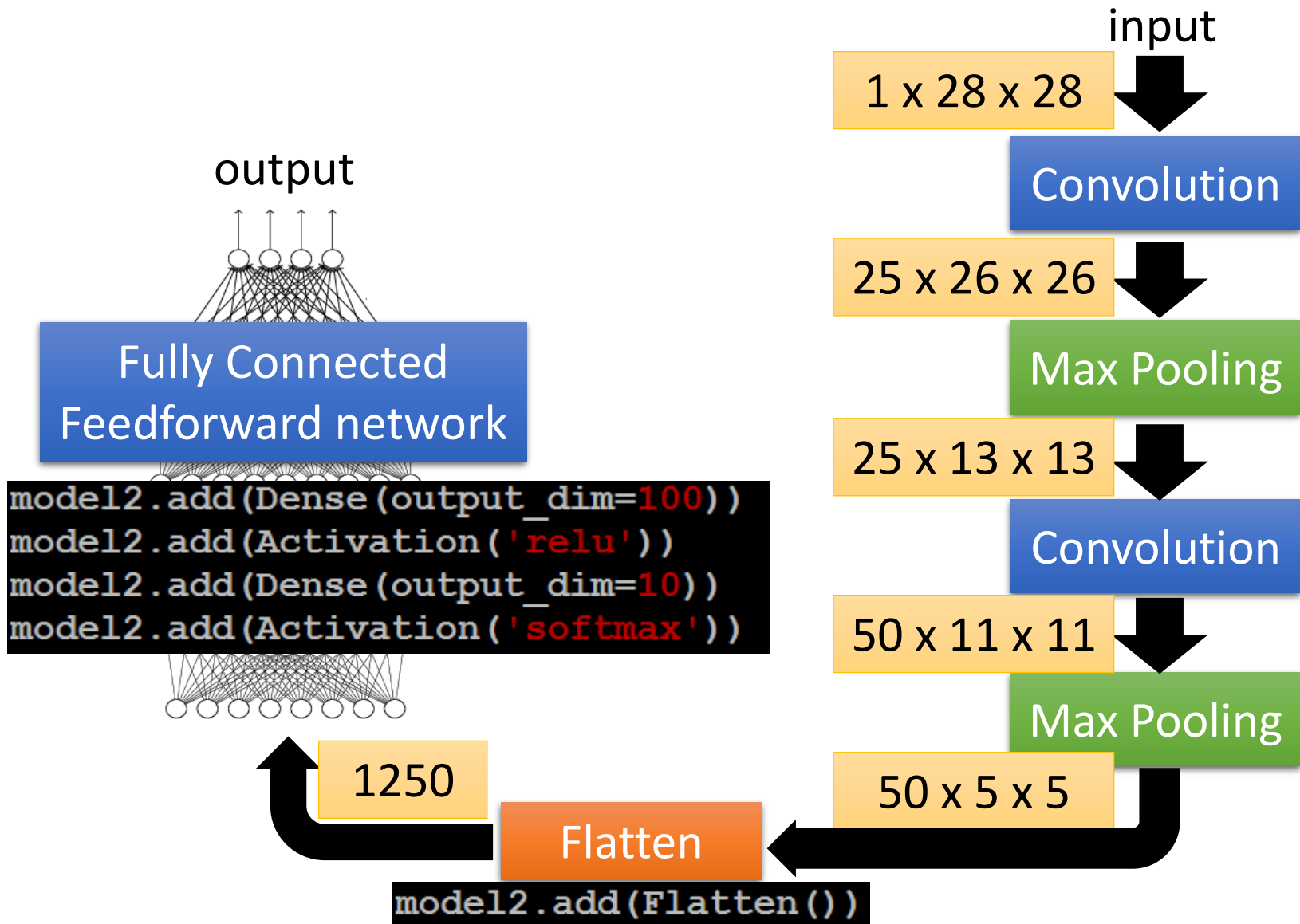
Live Demo

What does machine learn?

http://newsneakernews.wpengine.netdna-cdn.com/wp-content/uploads/2016/11/rihanna-puma-creeper-velvet-release-date-02.jpg

First Convolution Layer

• Typical-looking filters on the trained first layer

11 x 11
(AlexNet)

http://cs231n.github.io/understanding-cnn/
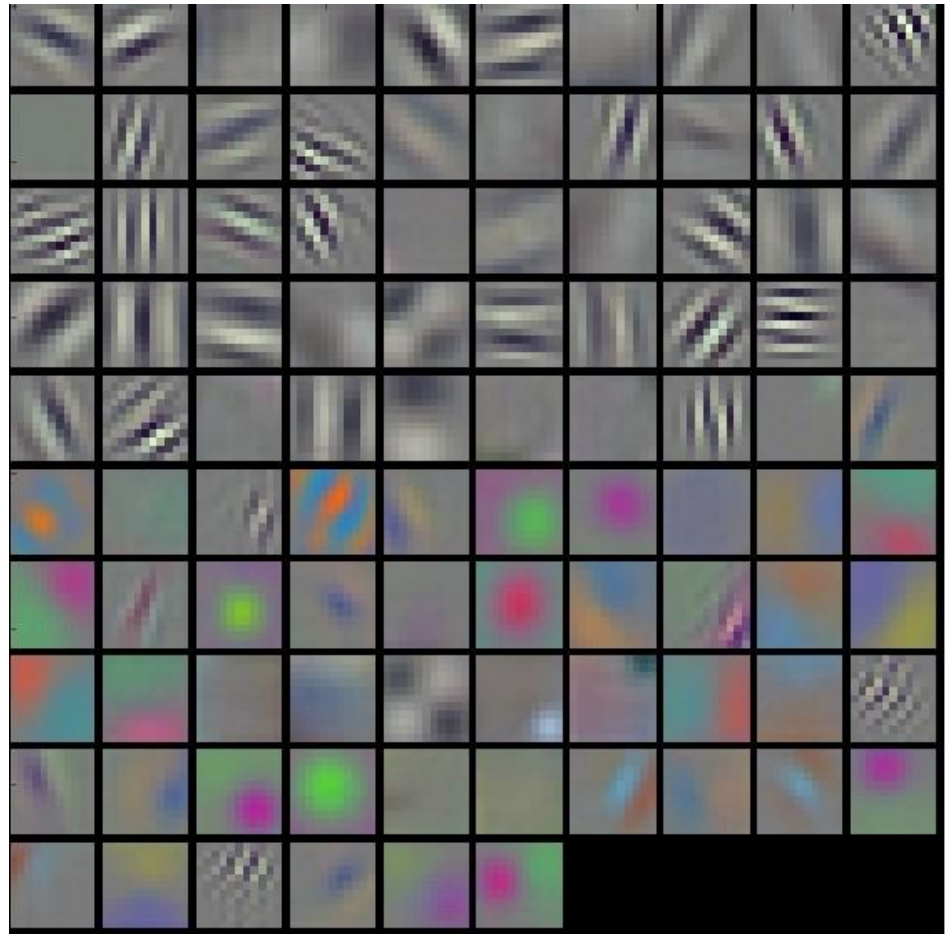
How about higher layers?

- Which images make a specific neuron activate
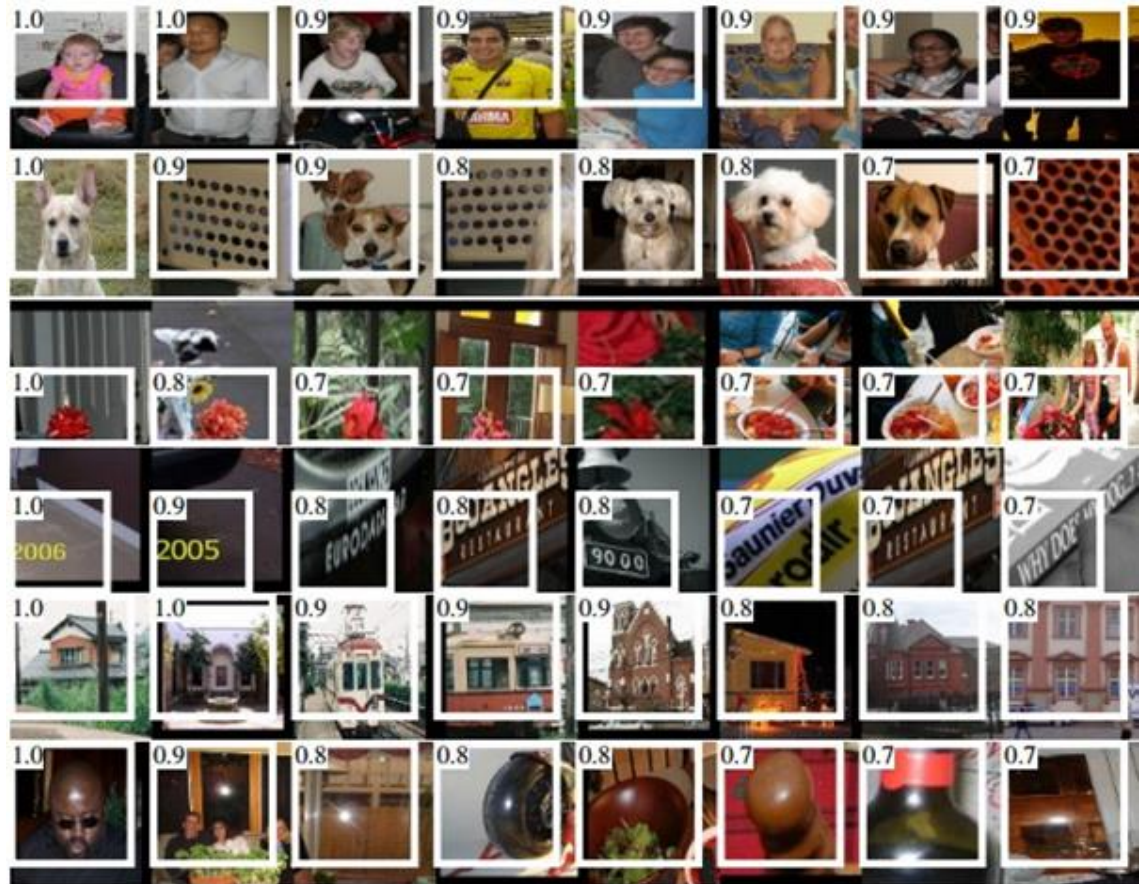
Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR, 2014

What does CNN learn?

The output of the k-th filter is a 11 x 11 matrix.

Degree of the activation of the k-th filter:

$$a^k = \sum_{i=1}^{11} \sum_{j=1}^{11} a_{ij}^k$$
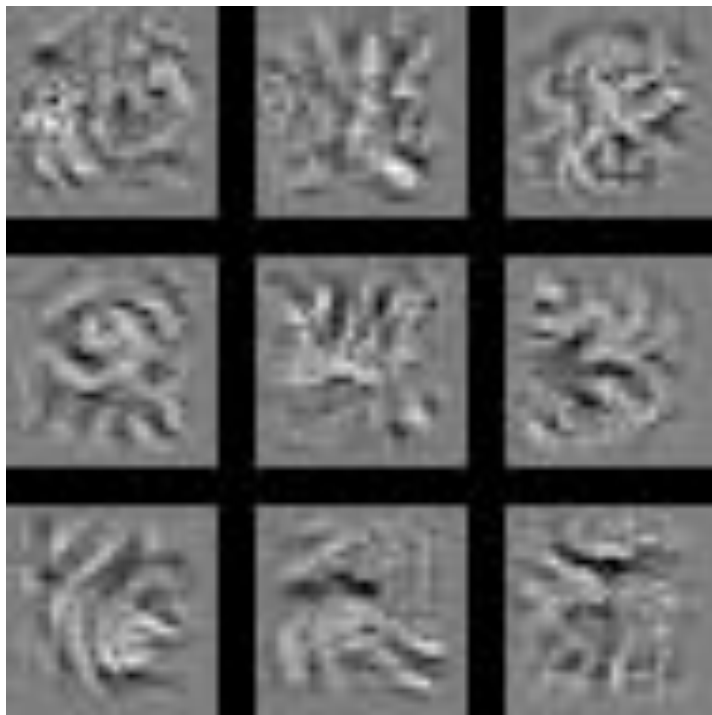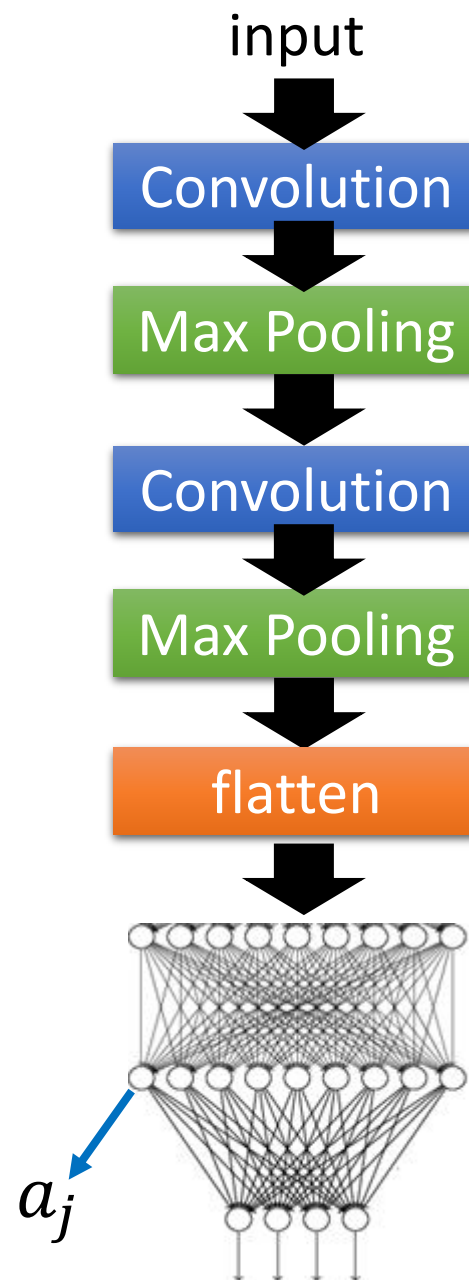
$$x^* = arg \max_x a^k \quad \text{(gradient ascent)}$$

$$\frac{\partial a^k}{\partial x_{ij}}$$

x input

11

11

a_{ij}^k

3 -1 ...... -1

-3 1 ...... -3

3 -2 ...... -1

25 3x3 filters → Convolution

Max Pooling

50 3x3 filters → Convolution

50 x 11 x 11

Max Pooling

What does CNN learn?

The output of the k-th filter is a 11 x 11 matrix.

Degree of the activation of the k-th filter:

$$a^k = \sum_{i=1}^{11} \sum_{j=1}^{11} a_{ij}^k$$

$$x^* = arg \max_x a^k \quad \text{(gradient ascent)}$$

$$\frac{\partial a^k}{\partial x_{ij}}$$

input

25 3x3 filters → Convolution

Max Pooling

50 3x3 filters → Convolution

50 x 11 x 11

Max Pooling
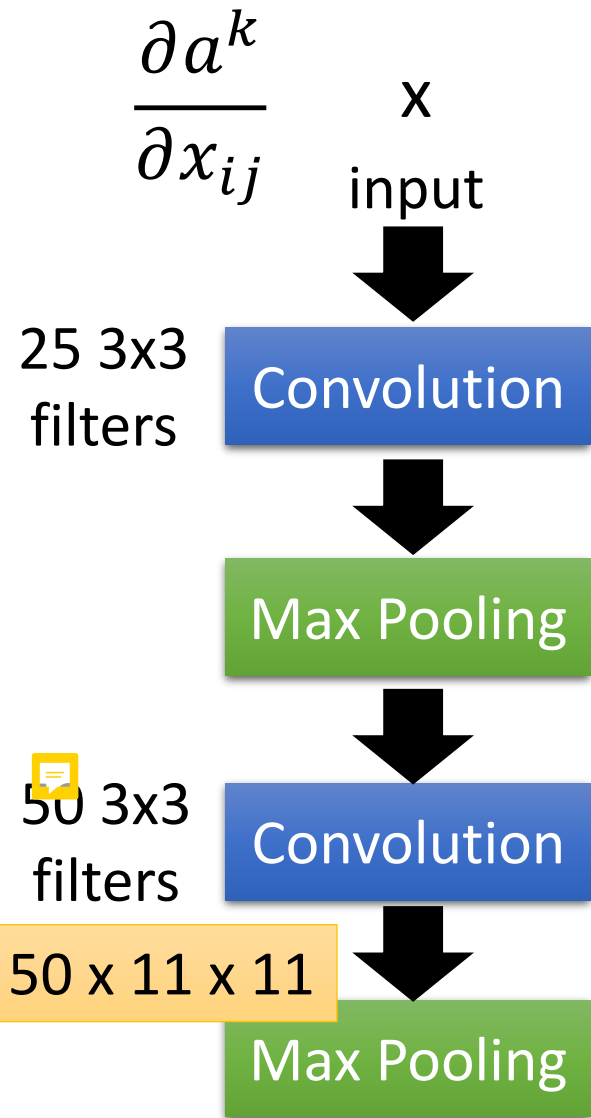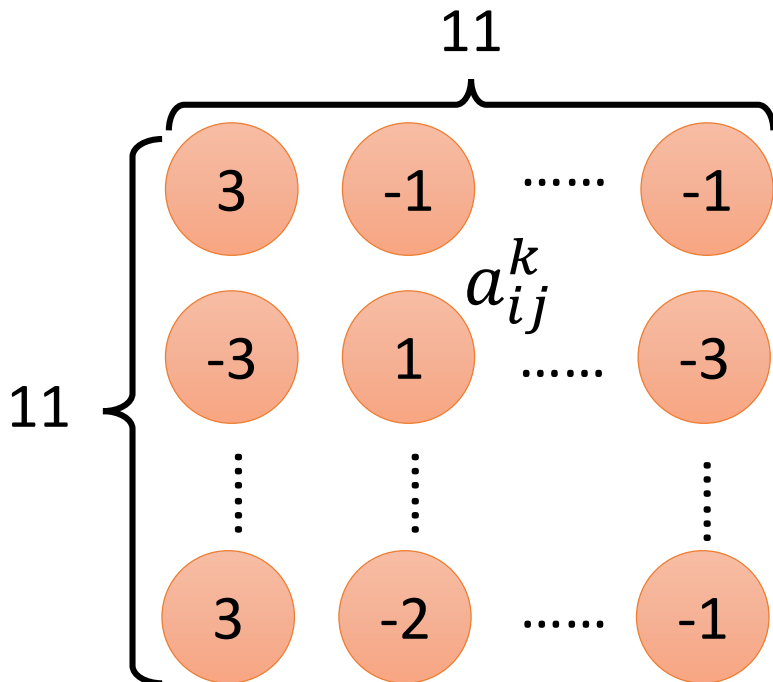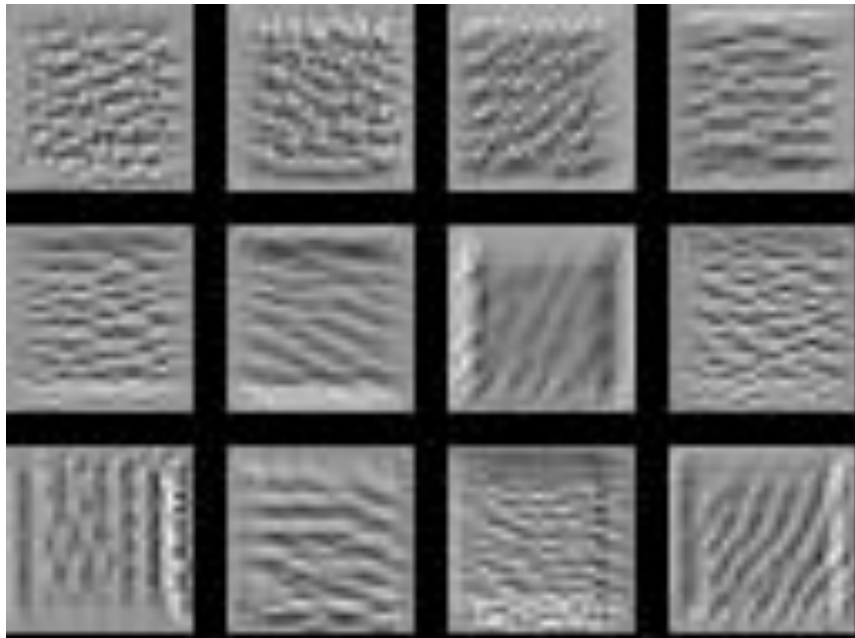
For each filter

What does CNN learn?

input
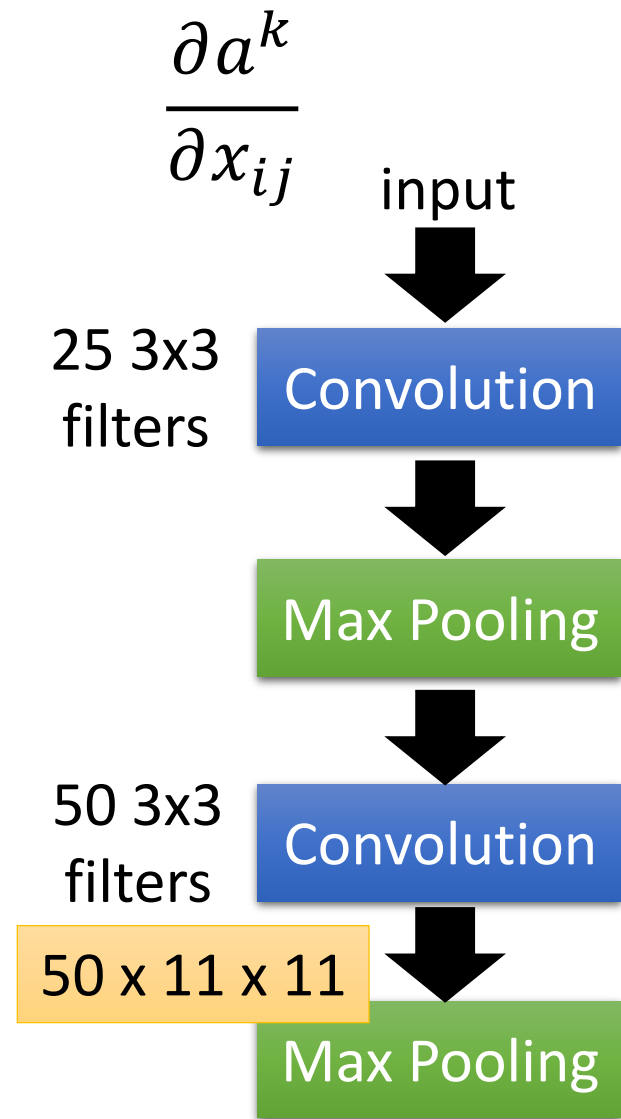
Find an image maximizing the output of neuron:

$$x^* = arg \max_x a^j$$

Each figure corresponds to a neuron

Convolution

Max Pooling

Convolution

Max Pooling

flatten

a_j

What does CNN learn?

input

$$x^* = arg \max_{x} y^i$$

Can we see digits?

Convolution

Max Pooling

Convolution

Max Pooling

flatten

0	1	2
3	4	5
6	7	8
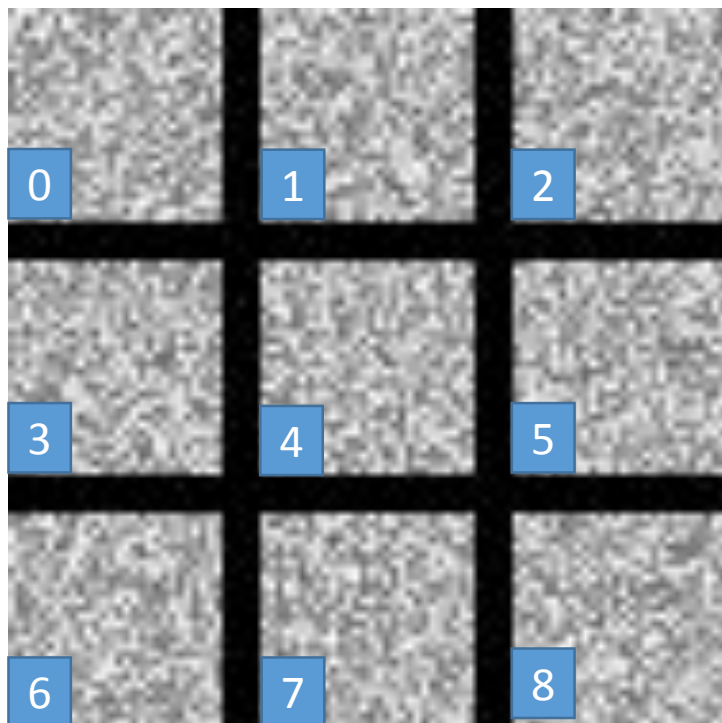
Deep Neural Networks are Easily Fooled
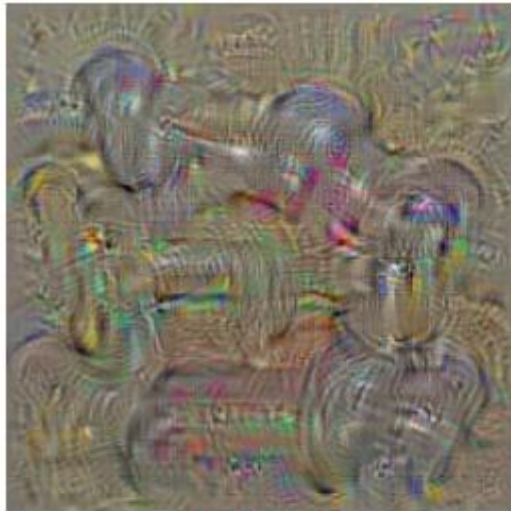https://www.youtube.com/watch?v=M2IebCN9Ht4
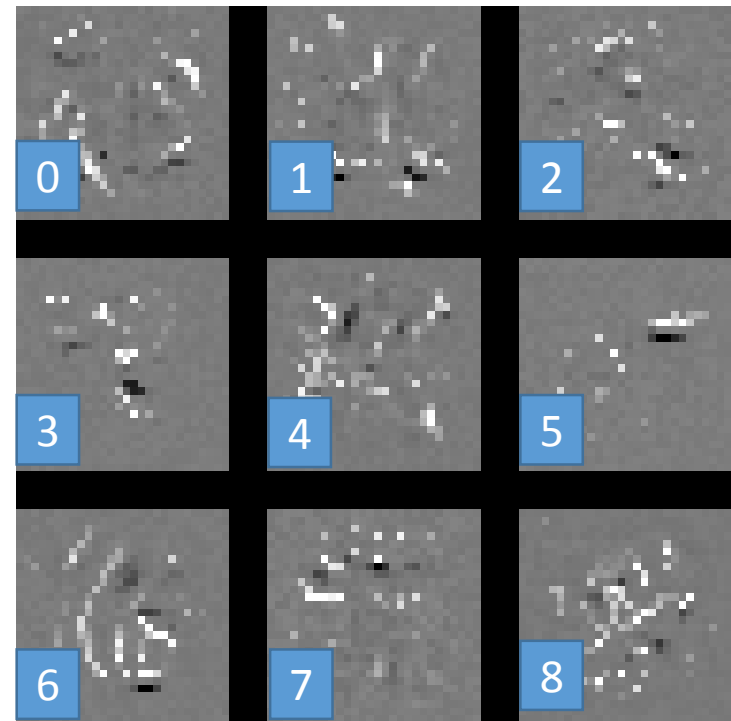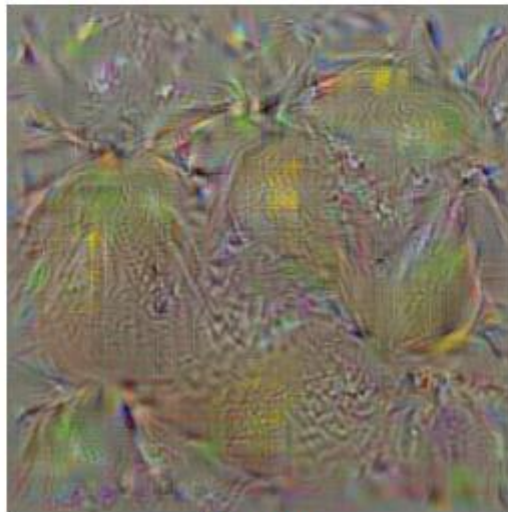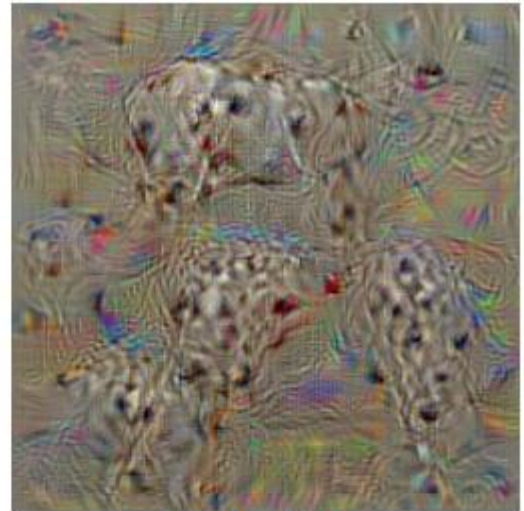
y_i

What does CNN learn?

$$x^* = arg \max_{x} y^i$$

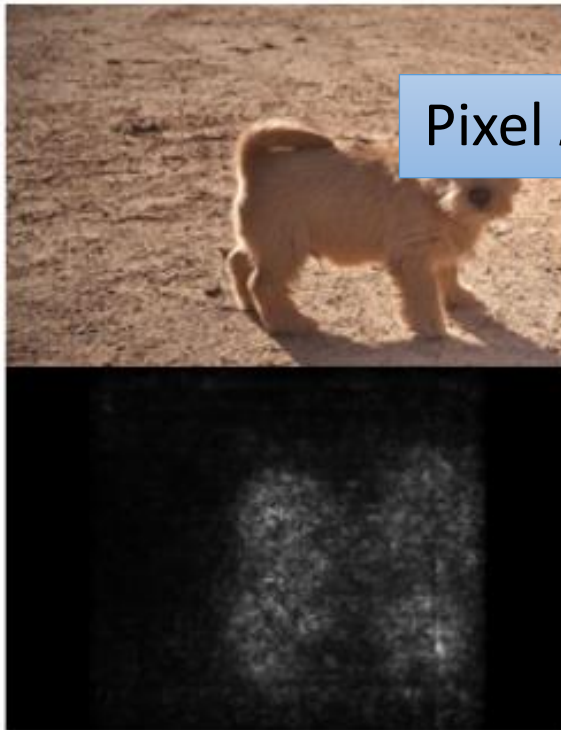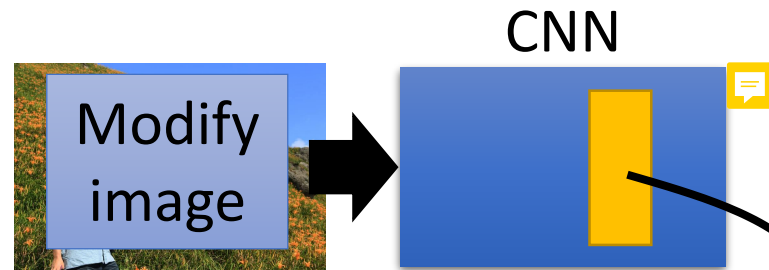$$x^* = arg \max_{x} \left(y^i - \boxed{\sum_{i,j} |x_{ij}|} \right)$$

Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR, 2014

$$\left|\frac{\partial y_k}{\partial x_{ij}}\right|$$

y_k: the predicted class of the model

Pixel x_{ij}
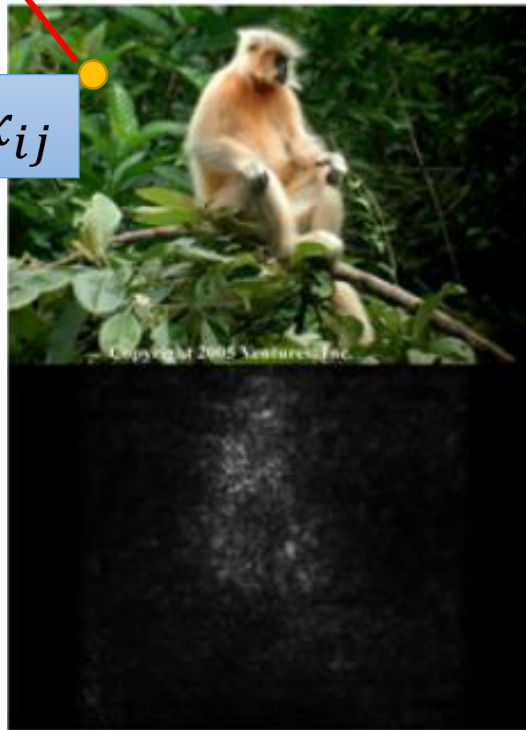
Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR, 2014

True Label: Pomeranian
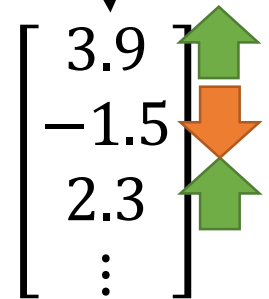
True Label: Car Wheel

True Label: Afghan Hound

Reference: Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014* (pp. 818-833)

Deep Dream

CNN

Modify image

- Given a photo, machine adds what it sees ......

$$\begin{bmatrix} 3.9 \\ -1.5 \\ 2.3 \\ \vdots \end{bmatrix}$$

CNN exaggerates what it sees

http://deepdreamgenerator.com/

Deep Dream

- Given a photo, machine adds what it sees ……

http://deepdreamgenerator.com/

Deep Style

- Given a photo, make its style like famous paintings

https://dreamscopeapp.com/

Deep Style

- Given a photo, make its style like famous paintings

https://dreamscopeapp.com/
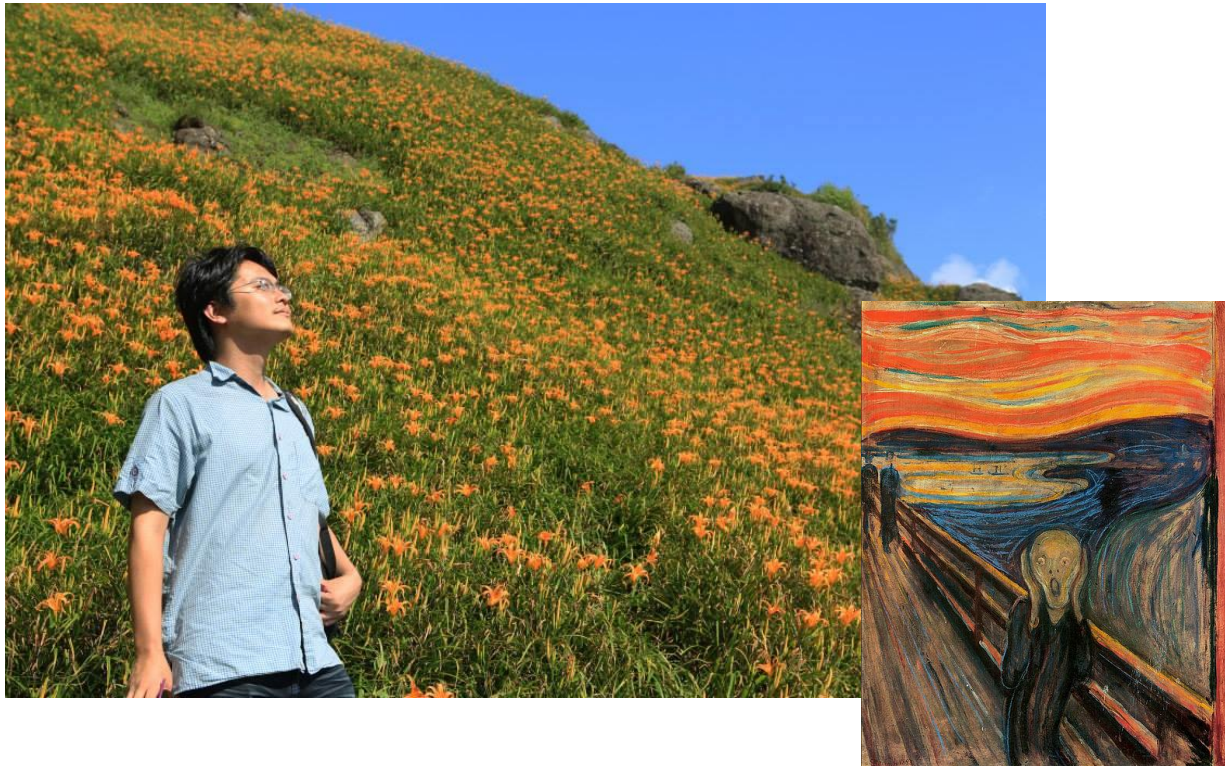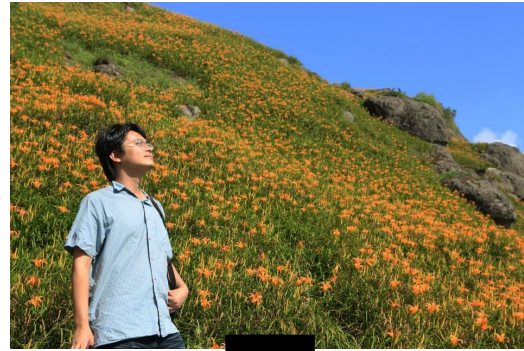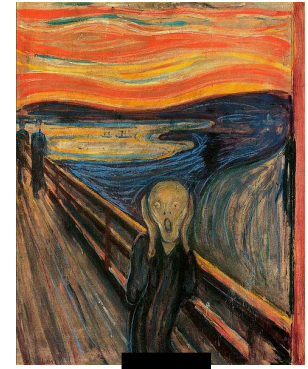
Deep Style

A Neural Algorithm of Artistic Style

https://arxiv.org/abs/1508.06576

More Application: Playing Go

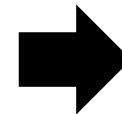**19 x 19 matrix (image)** → **Network** → Next move (19 x 19 positions)

19 x 19 vector

Black: 1

white: -1

none: 0
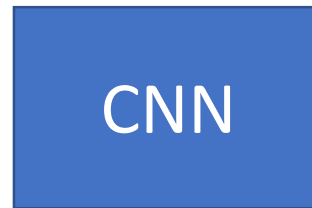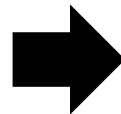
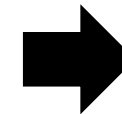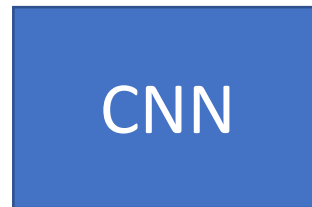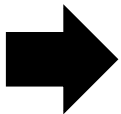Fully-connected feedforward network can be used

But CNN performs much better.

More Application: Playing Go

Training:

record of previous plays

黑: 5之五 ➡ 白: 天元 ➡ 黑: 五之5 …
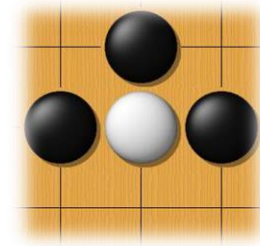

CNN

Target:
"天元" = 1
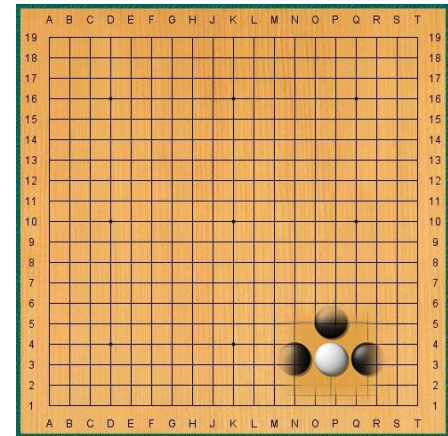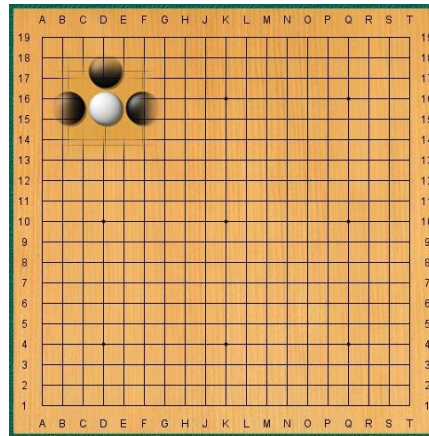else = 0

CNN

Target:
"五之 5" = 1
else = 0

Why CNN for playing Go?

- Some patterns are much smaller than the whole image

    Alpha Go uses 5 x 5 for first layer

- The same patterns appear in different regions.

Why CNN for playing Go?
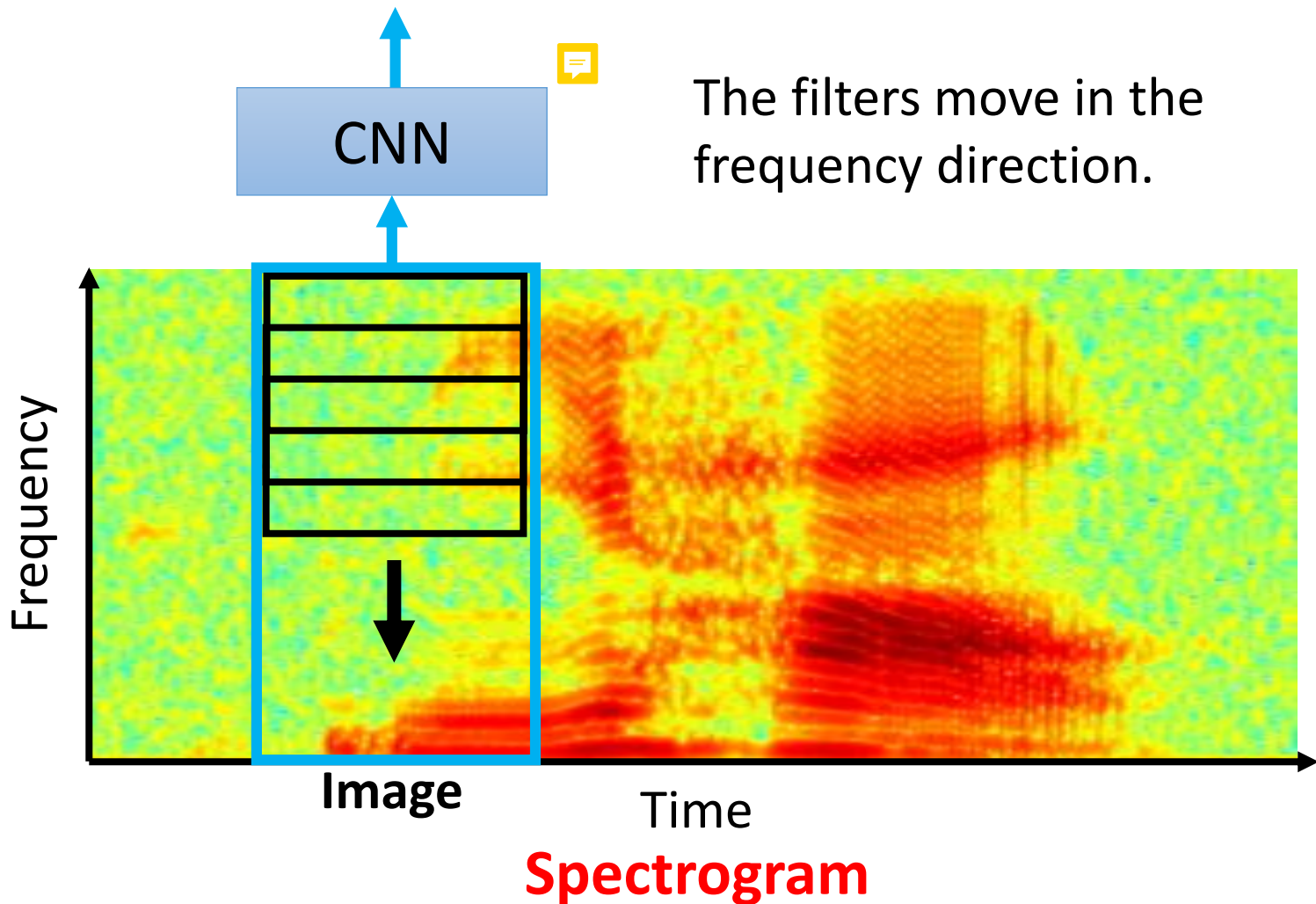
- Subsampling the pixels will not change the object

Max Pooling How to explain this???

Neural network architecture. The input to the policy network is a $19 \times 19 \times 48$ image stack consisting of 48 feature planes. The first hidden layer zero pads the input into a 23×23 image, then convolves k filters of kernel size 5×5 with stride 1 with the input image and applies a rectifier nonlinearity. Each of the subsequent hidden layers 2 to 12 zero pads the respective previous hidden layer into a 21×21 image, then convolves k filters of kernel size 3×3 with stride 1, again followed by a rectifier nonlinearity. The final layer convolves 1 filter of kernel size 1×1 with stride 1, with a different bias for each position, and applies a softmax function. The **Alpha Go does not use Max Pooling ......** Extended Data Table 3 additionally show the results of training with $k = 128$, 256 and 384 filters.
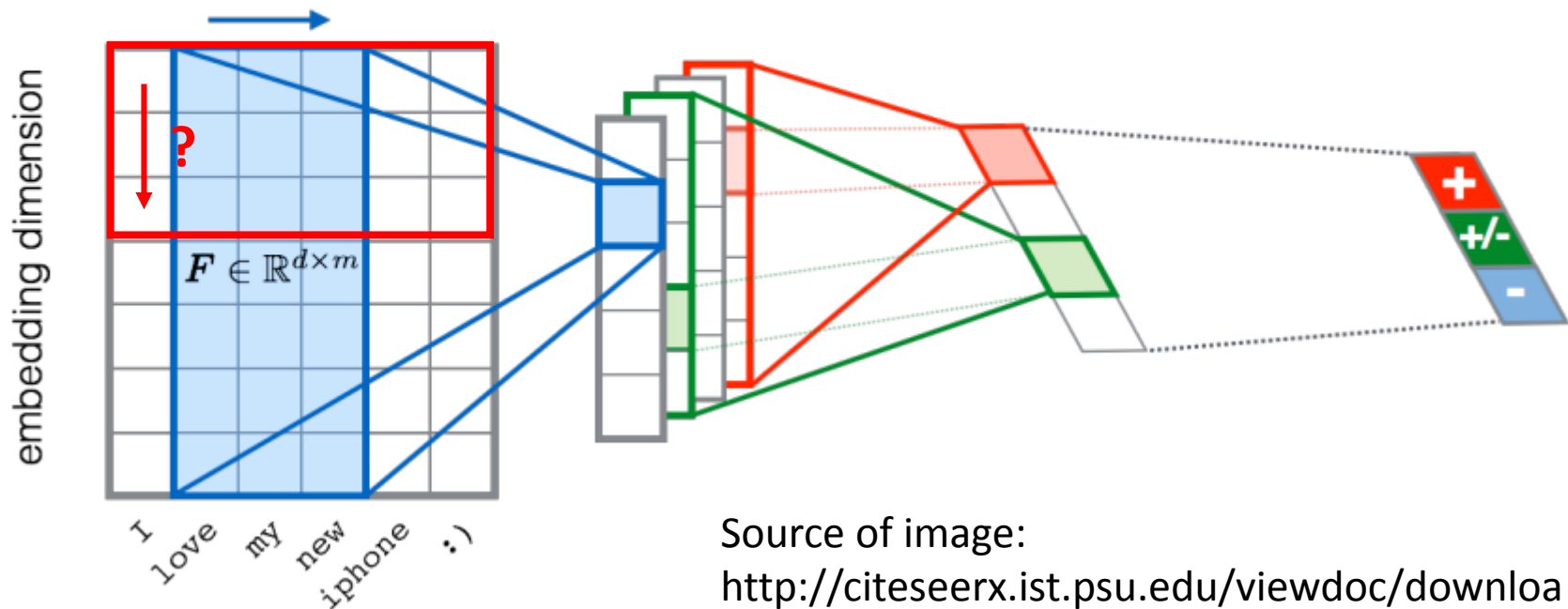
More Application: Speech

The filters move in the frequency direction.

More Appl

- The methods of visualization in these slides
  - https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html
- More about visualization
  - http://cs231n.github.io/understanding-cnn/
- Very cool CNN visualization toolkit
  - http://yosinski.com/deepvis
  - http://scs.ryerson.ca/~aharley/vis/conv/

sentence matrix

$S \in \mathbb{R}^{d \times |s|}$

softmax

embedding dimension

?

$F \in \mathbb{R}^{d \times m}$

I love my new iphone :)

+

+/-

-

Source of image:
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.6858&rep=rep1&type=pdf

Acknowledgment

• 感謝 Guobiao Mo 發現投影片上的打字錯誤

To learn more ……

• How to let machine draw an image
  • PixelRNN
    • https://arxiv.org/abs/1601.06759
  • Variation Autoencoder (VAE)
    • https://arxiv.org/abs/1312.6114
  • Generative Adversarial Network (GAN)
    • http://arxiv.org/abs/1406.2661