

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

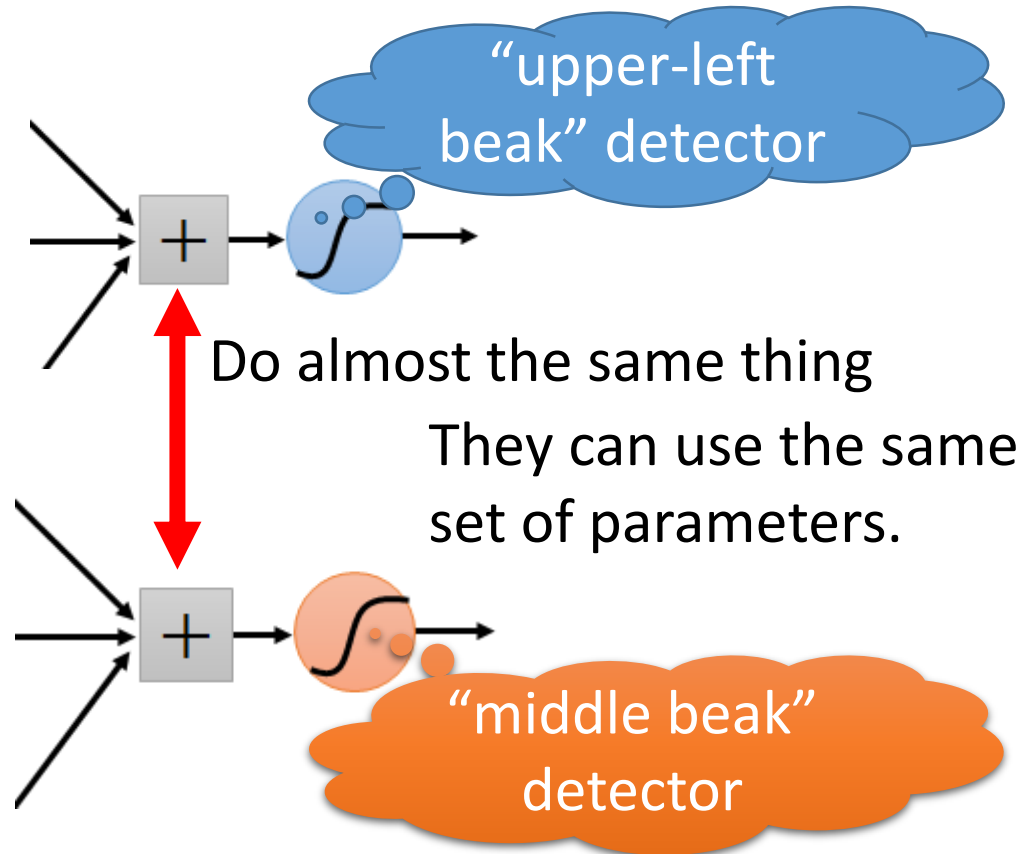
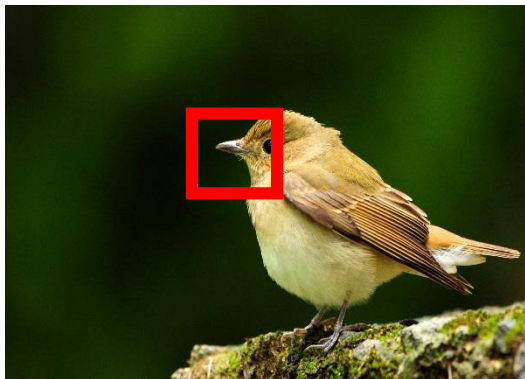
A neuron does not have to see the whole image to discover the pattern.

Connecting to small region with less parameters



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

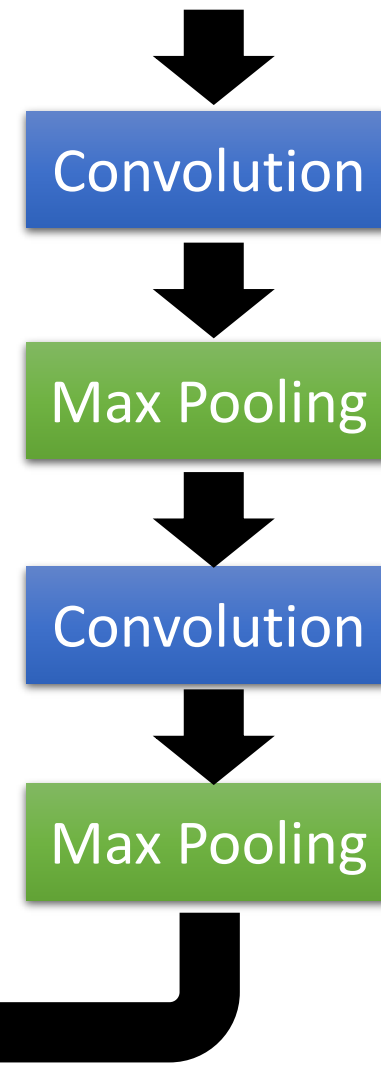
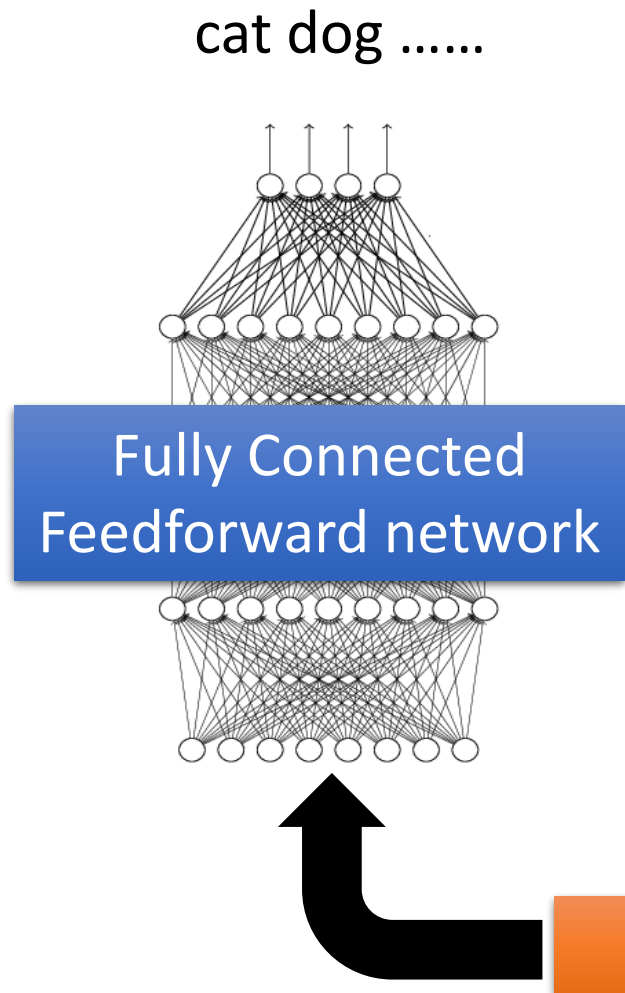


We can subsample the pixels to make image smaller



Less parameters for the network to process the image

The whole CNN



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

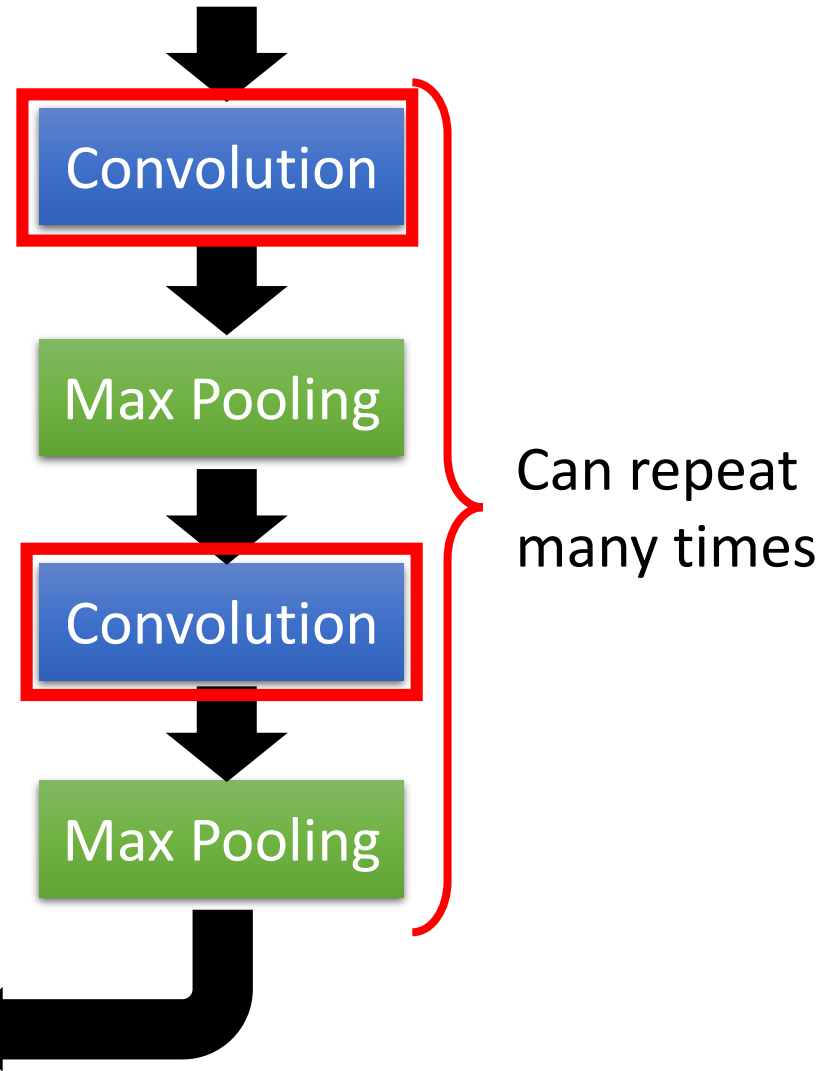
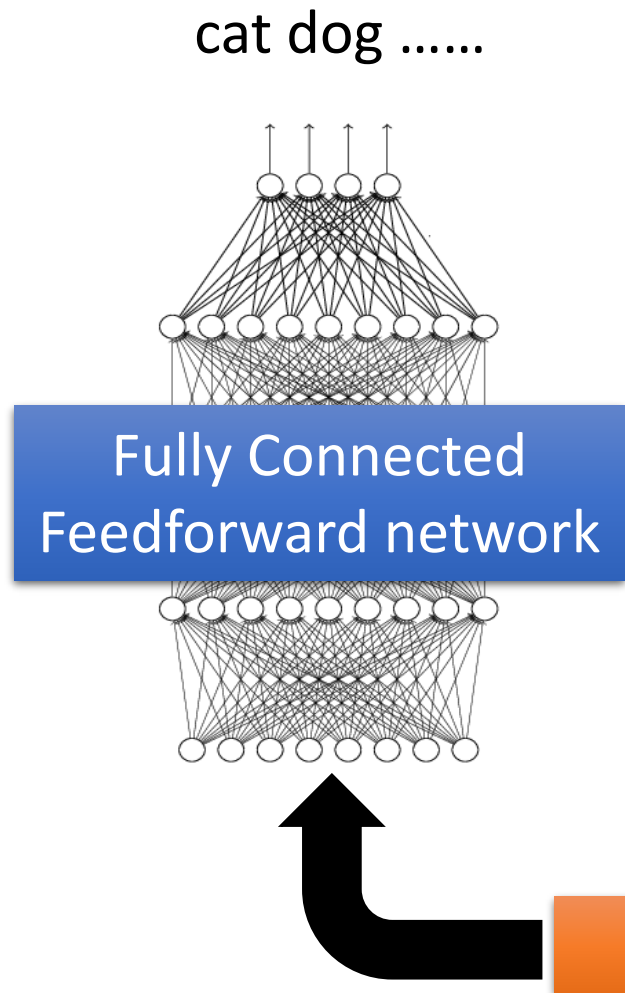
Convolution

Max Pooling

Flatten

Can repeat many times

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

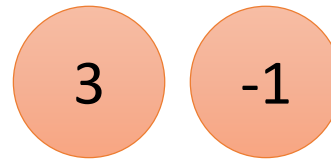
1	-1	-1
-1	1	-1
-1	-1	1

Filter 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



CNN – Convolution

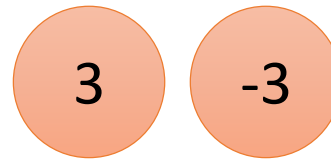
1	-1	-1
-1	1	-1
-1	-1	1

Filter 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

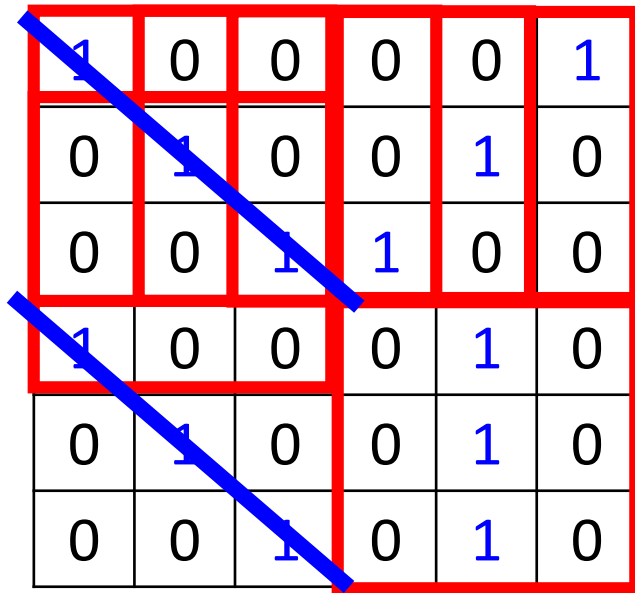
6 x 6 image



We set stride=1 below

CNN – Convolution

stride=1

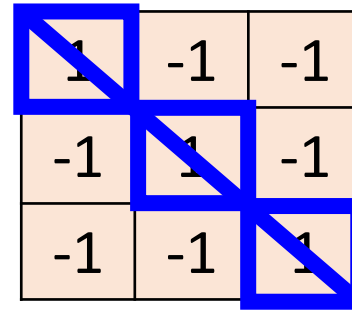


A 6x6 grid representing an image. The values are as follows:

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

A 3x3 red bounding box highlights the top-left corner (rows 1-3, columns 1-3). A blue diagonal line runs from the top-left to the bottom-right of the entire 6x6 grid.

6 x 6 image

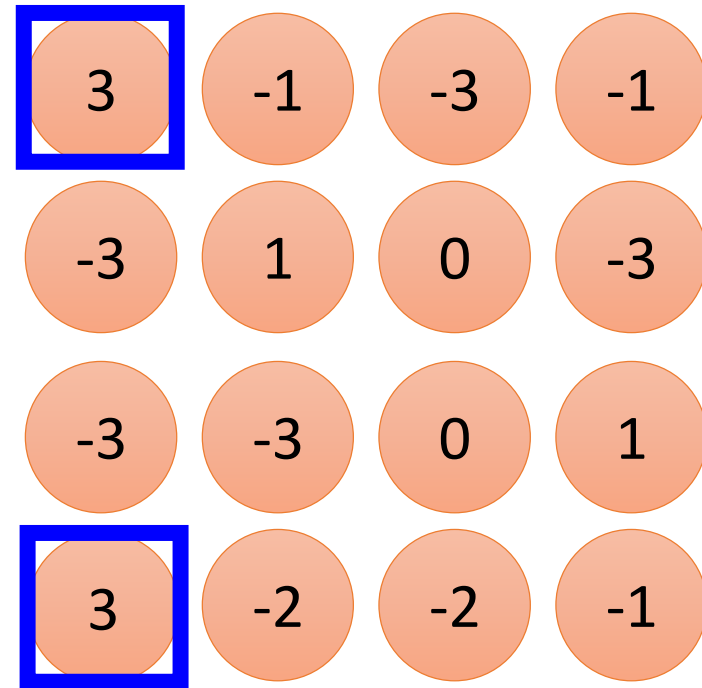


A 3x3 grid representing Filter 1. The values are as follows:

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

A blue diagonal line runs from the top-left to the bottom-right of the 3x3 grid.

Filter 1



A 4x4 grid of orange circles containing numerical values. The values are as follows:

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

Two blue bounding boxes highlight the top-left and bottom-left circles, both containing the value 3.

Property 2

CNN – Convolution

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

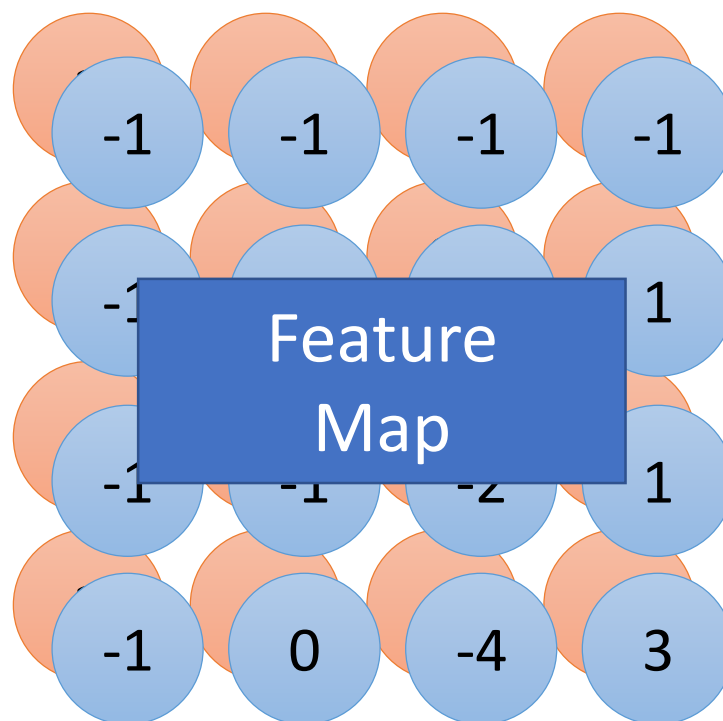
Filter 2

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

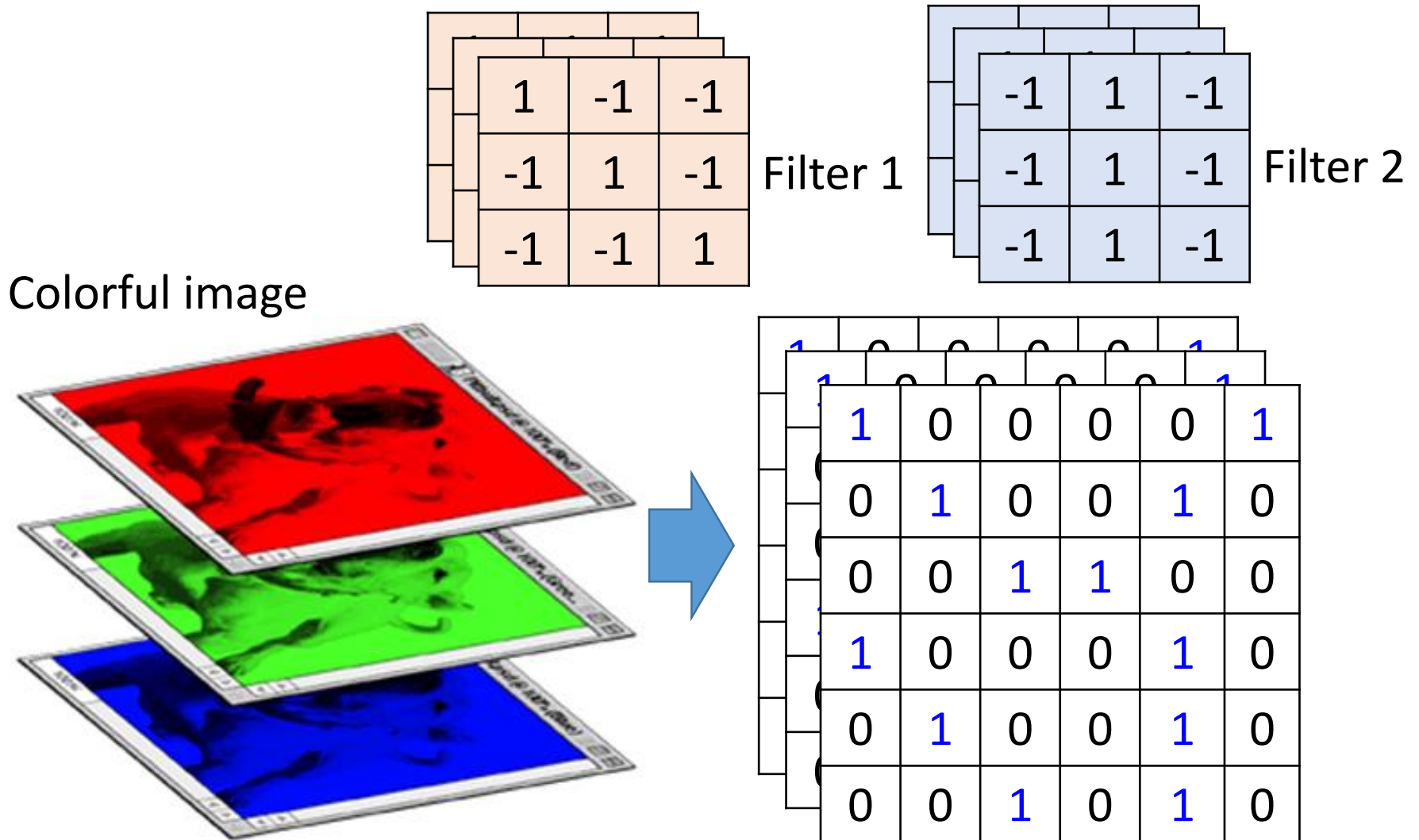
Do the same process for every filter



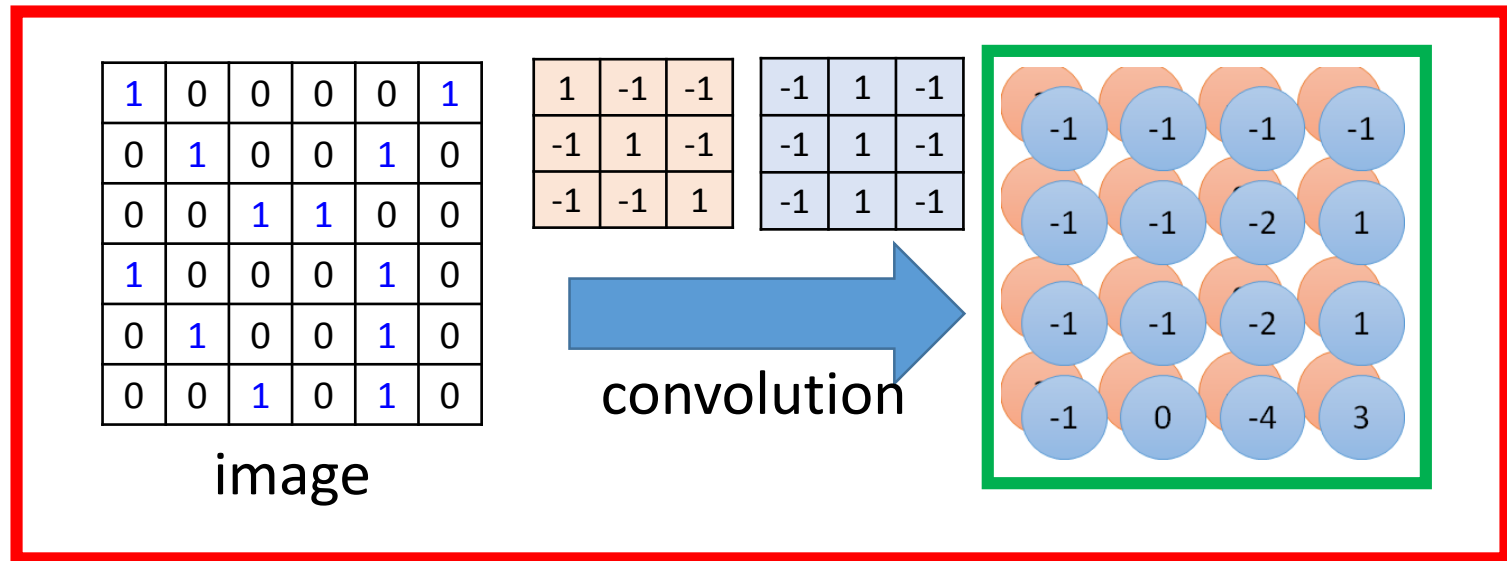
4 x 4 image

如何理解feature map?
每一个feature map就是对图像从一个不一样的角度去描述,
就像我们描述一个人时,也会从身高体重等多个角度去描述一样。
多个feature map就是从多个角度来描述图片。

CNN – Colorful image

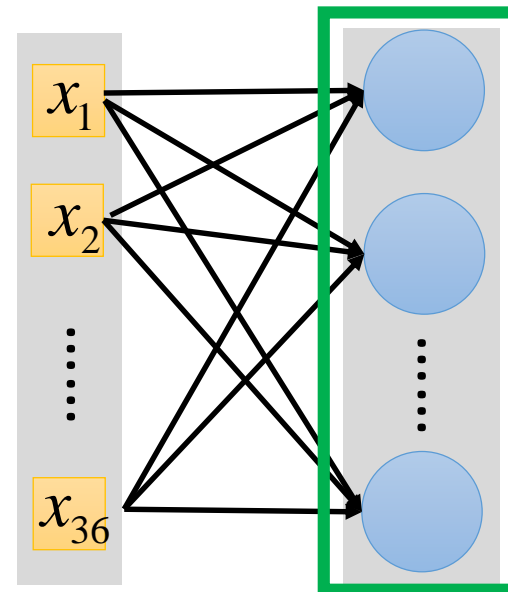


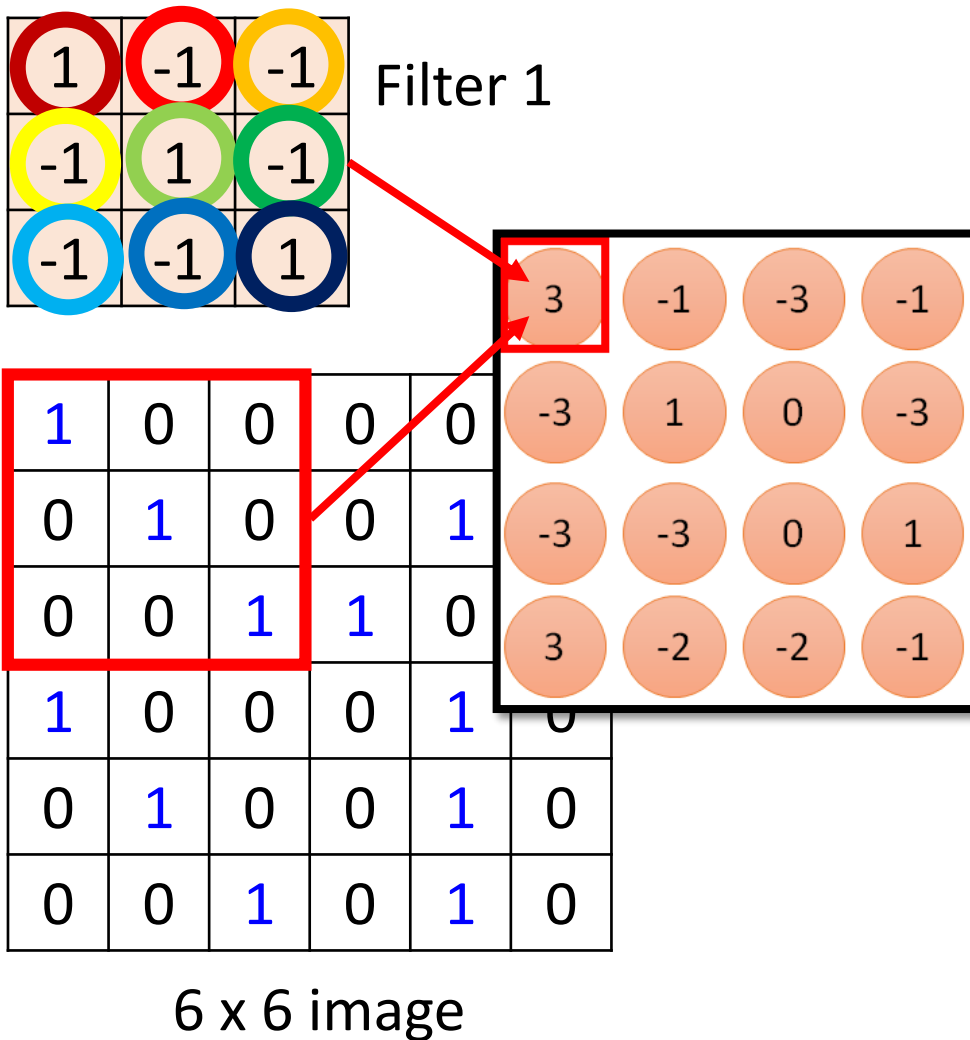
Convolution v.s. Fully Connected



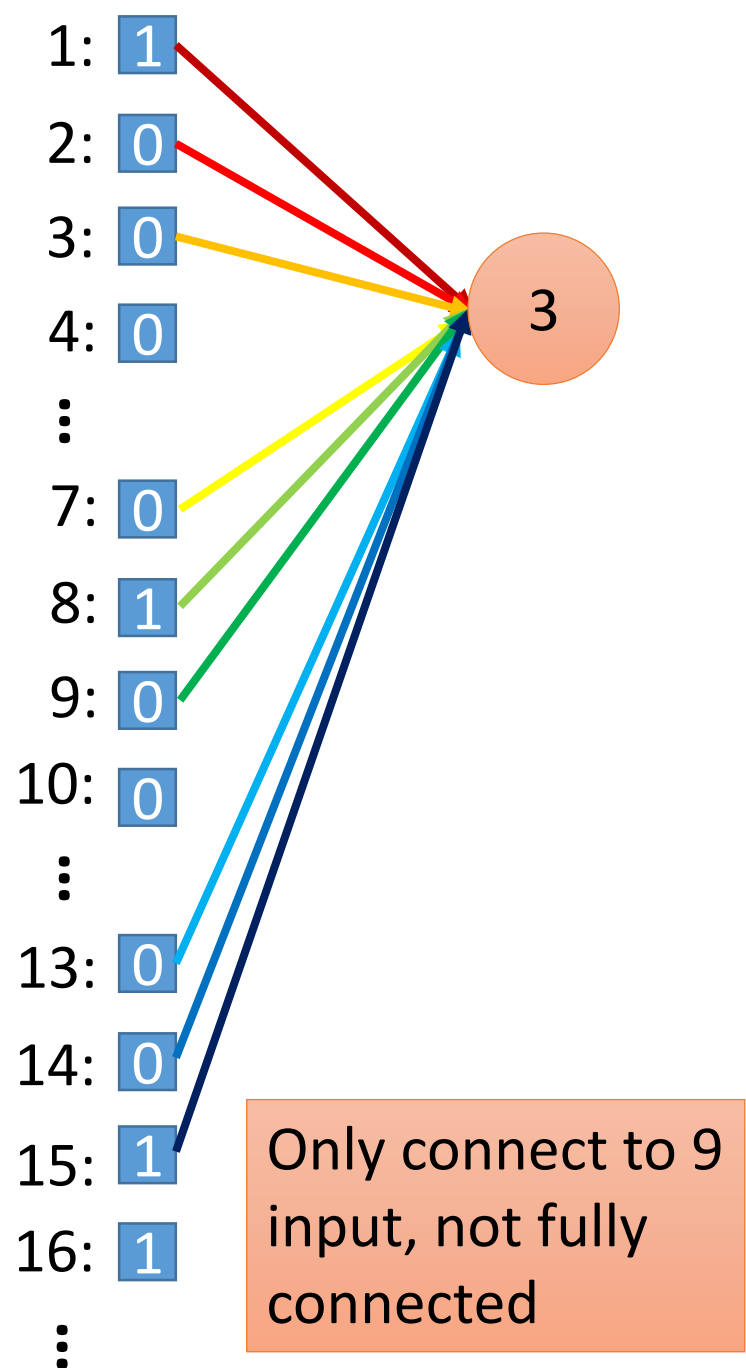
Fully-
connected

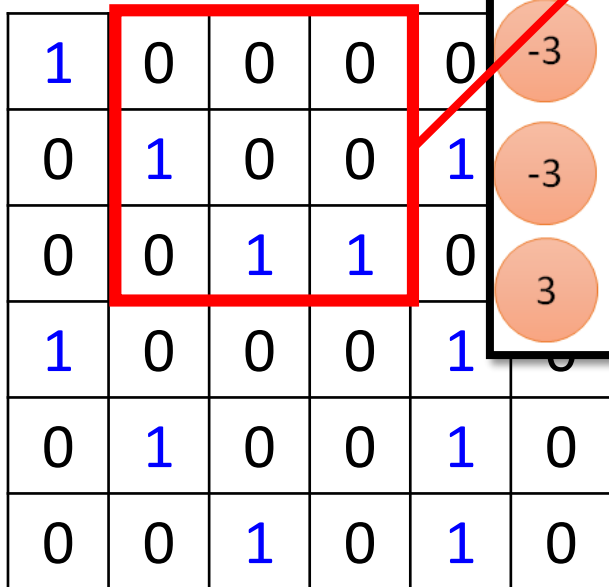
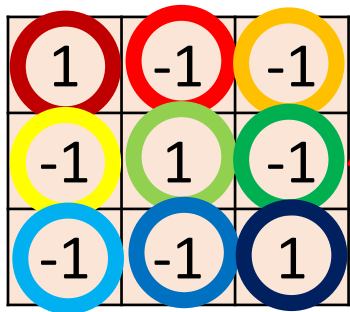
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0





Less parameters!

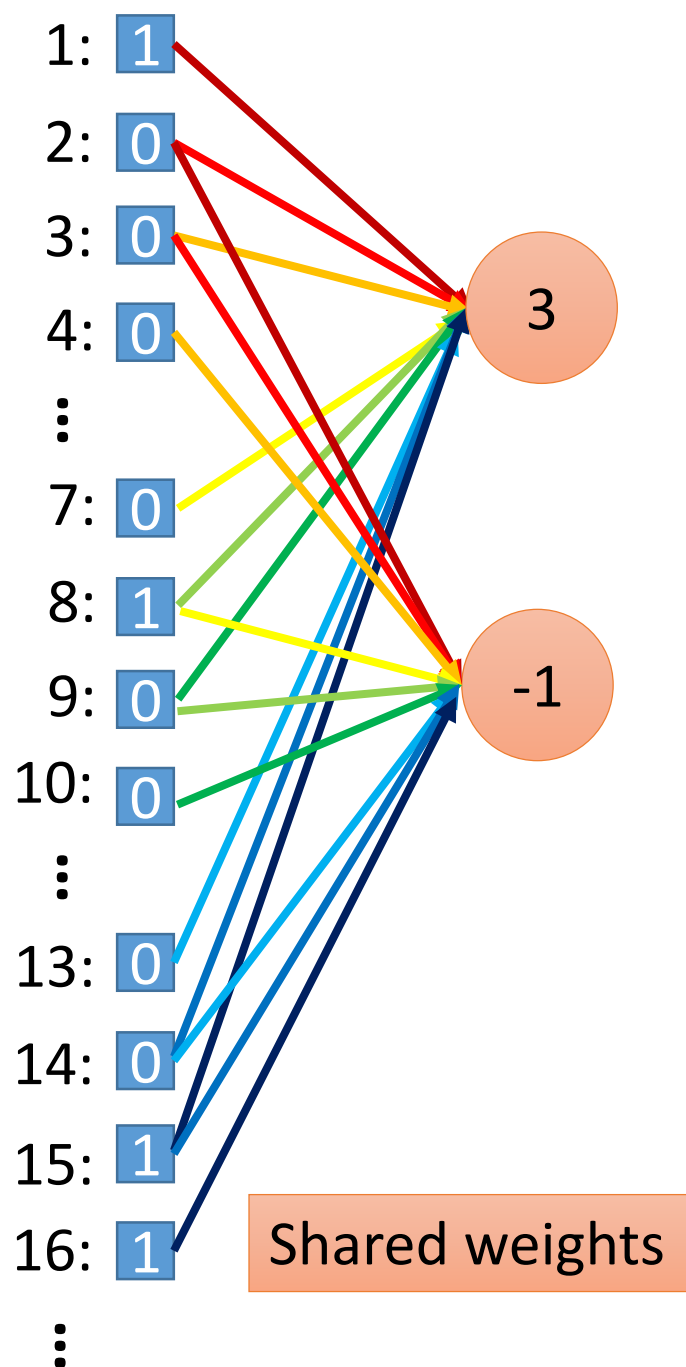
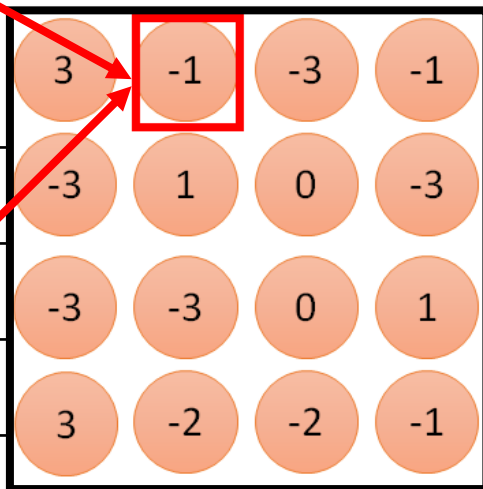




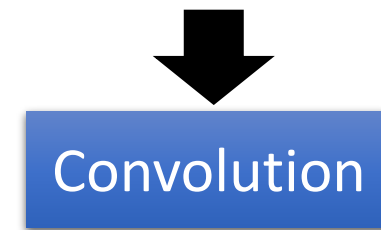
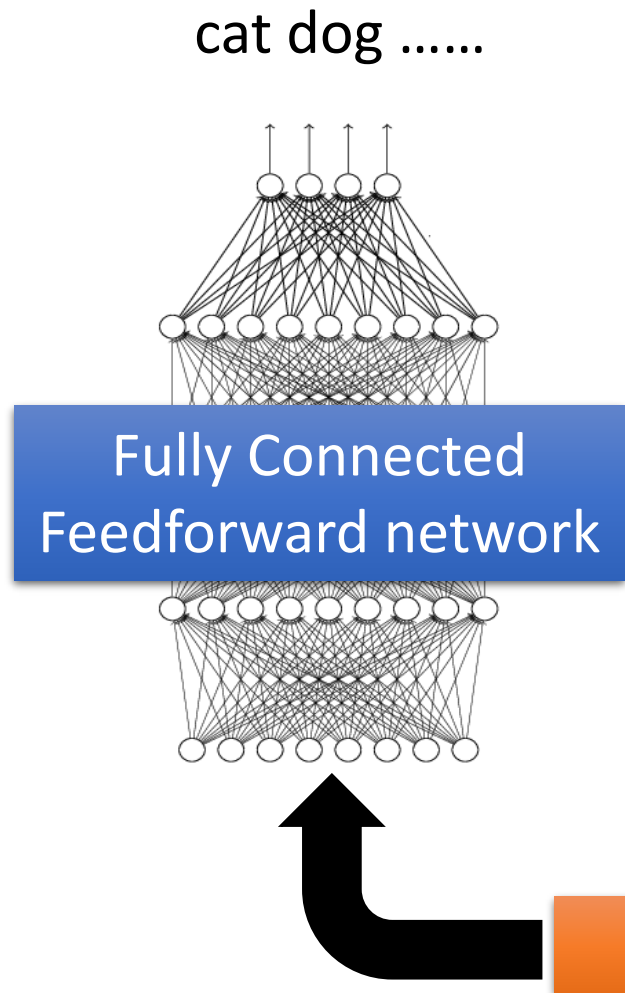
6 x 6 image

Less parameters!

Even less parameters!



The whole CNN



Can repeat many times

A red bracket on the right side of the diagram groups the 'Convolution' and 'Max Pooling' layers of the first stage, indicating that this sequence of operations can be repeated multiple times in a CNN architecture.



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

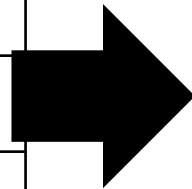
3	-1	-3	-1
-3	1	0	-3
-3	-3	0	1
3	-2	-2	-1

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

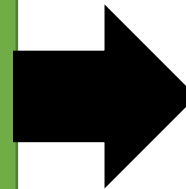
6 x 6 image



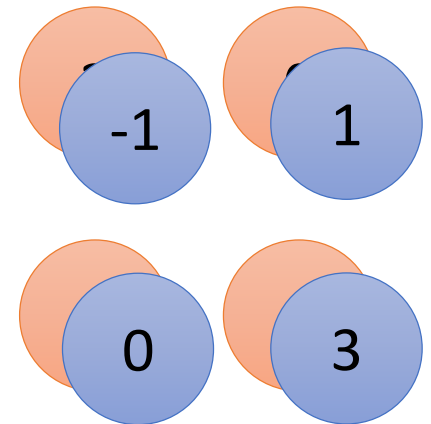
Conv



Max
Pooling



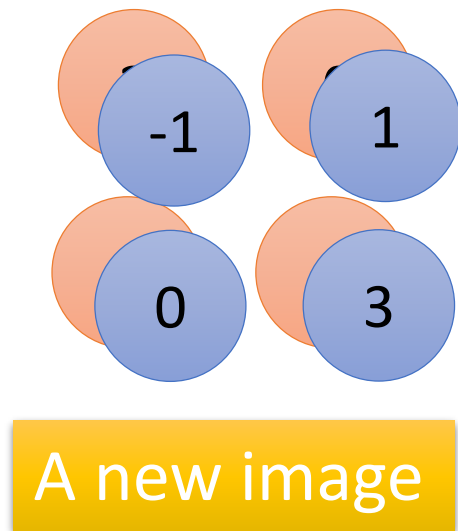
New image
but smaller



2 x 2 image

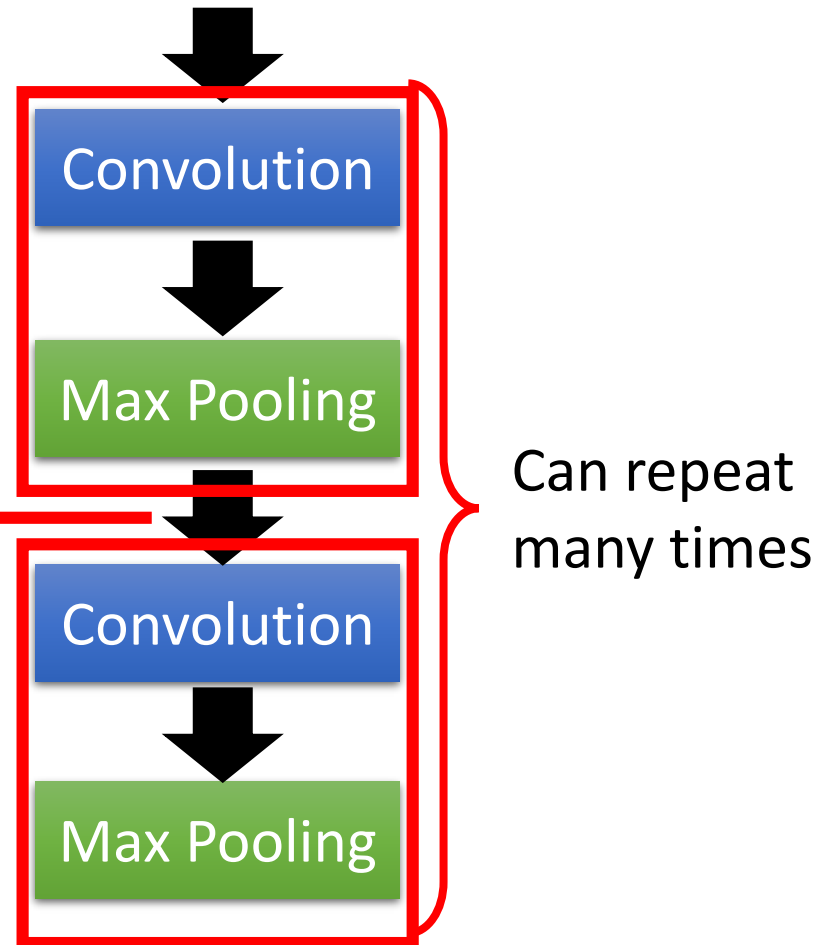
Each filter
is a channel

The whole CNN



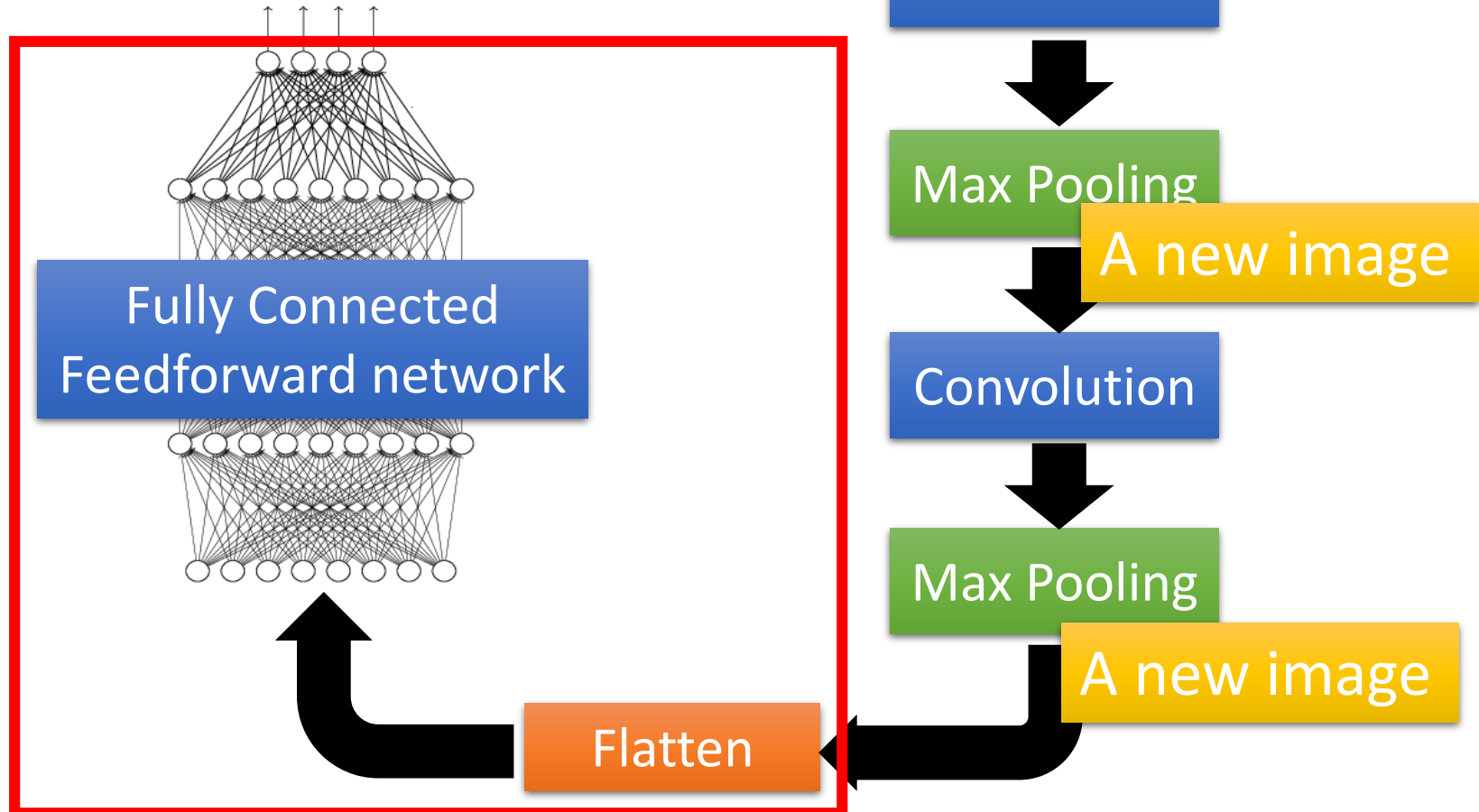
Smaller than the original image

The number of the channel is the number of filters

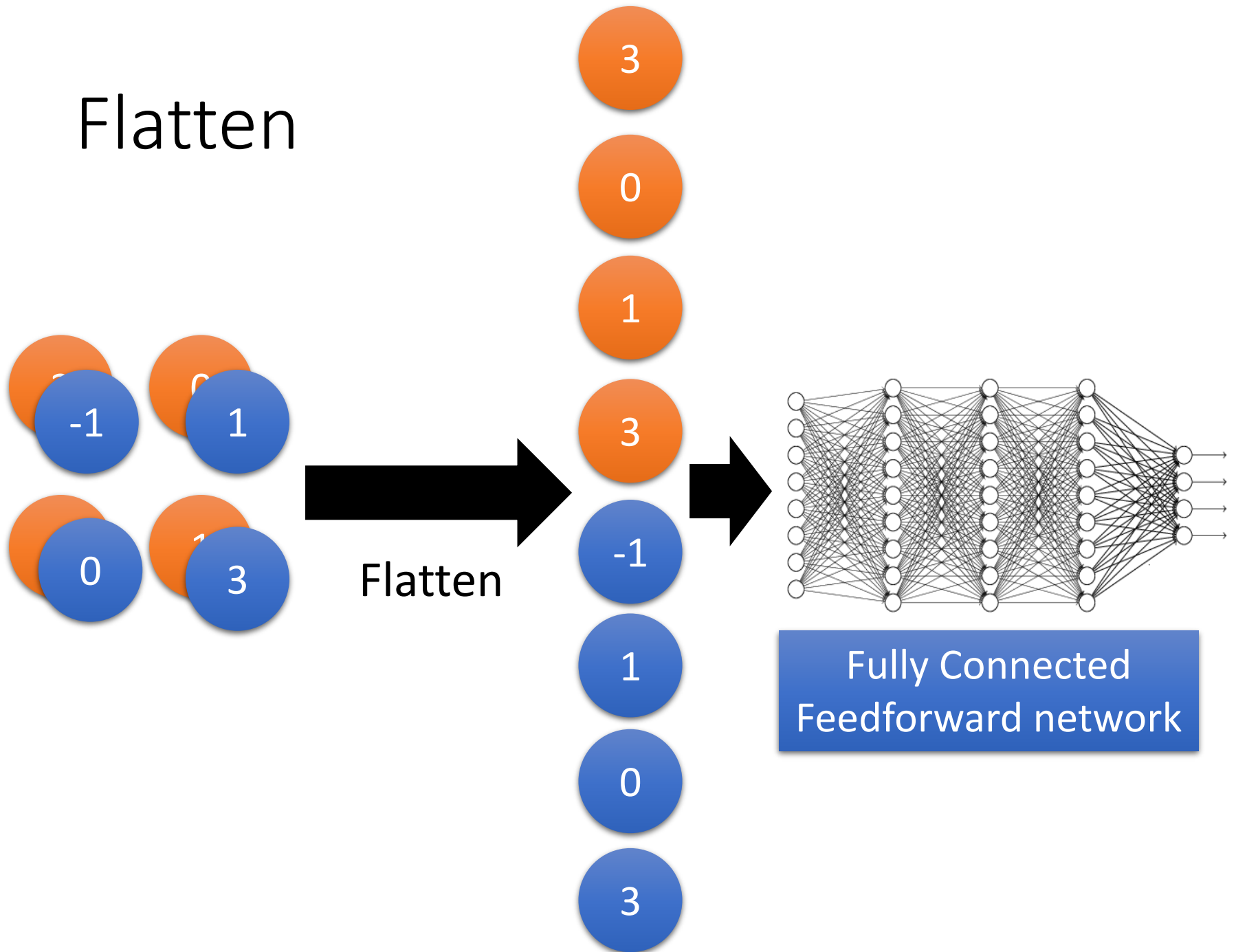


The whole CNN

cat dog



Flatten



CNN in Keras

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

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

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

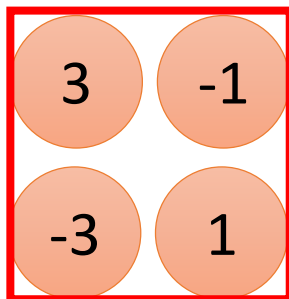
-1	1	-1
-1	1	-1
-1	1	-1

..... There are 25
3x3 filters.

Input_shape = (28, 28, 1)

28 x 28 pixels 1: black/white, 3: RGB

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



input

Convolution

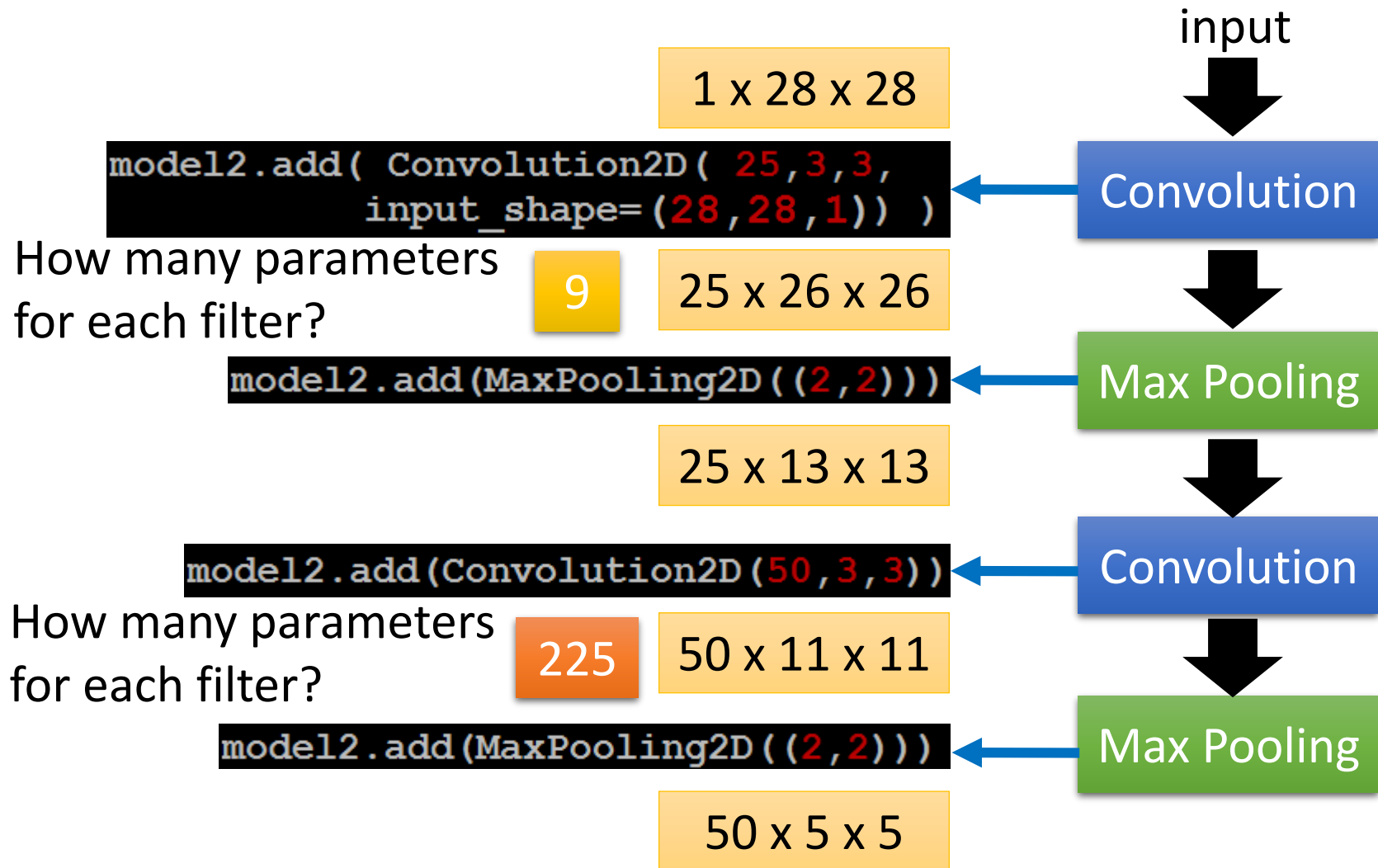
Max Pooling

Convolution

Max Pooling

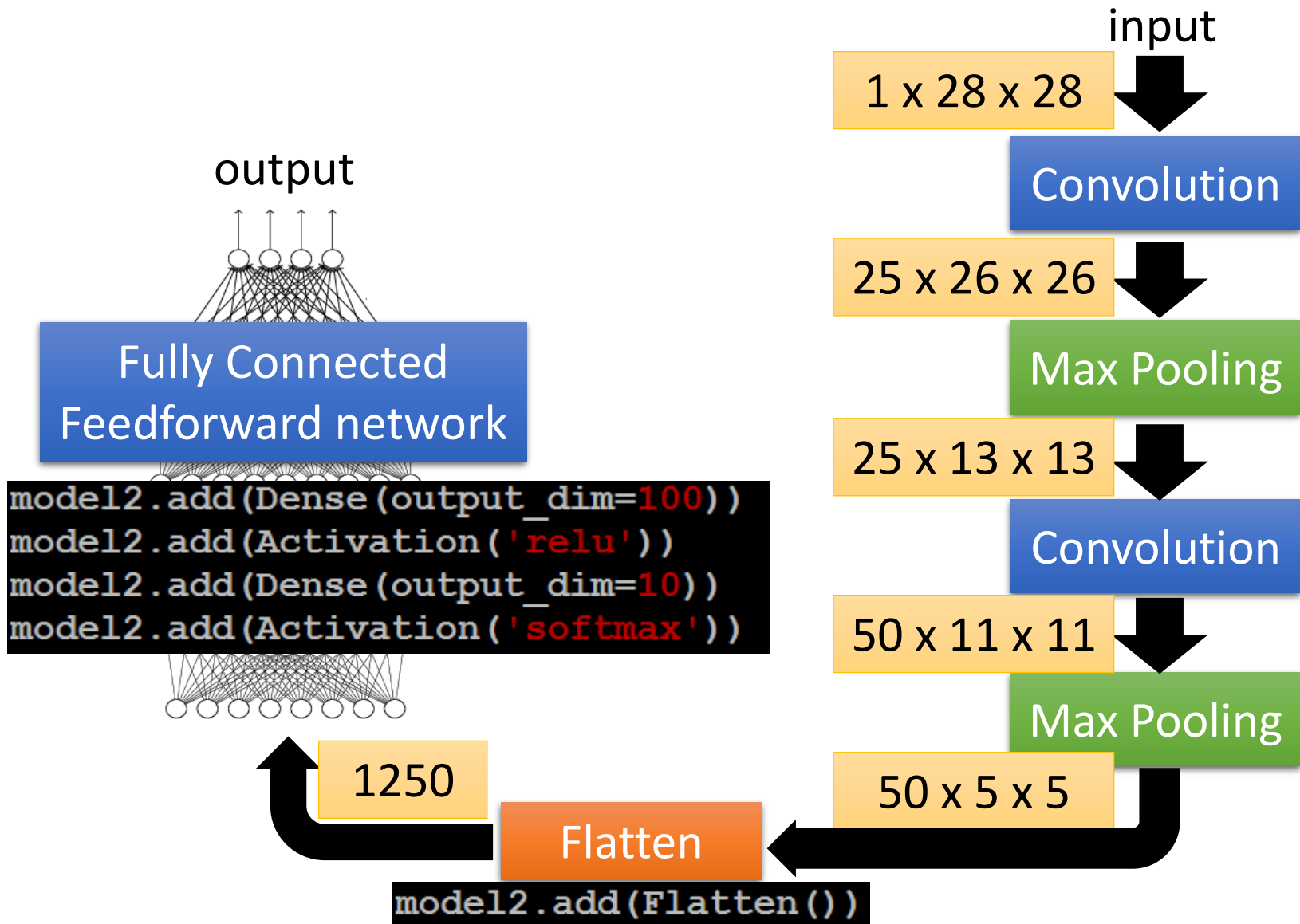
CNN in Keras

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



CNN in Keras

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



Live Demo

What does machine learn?

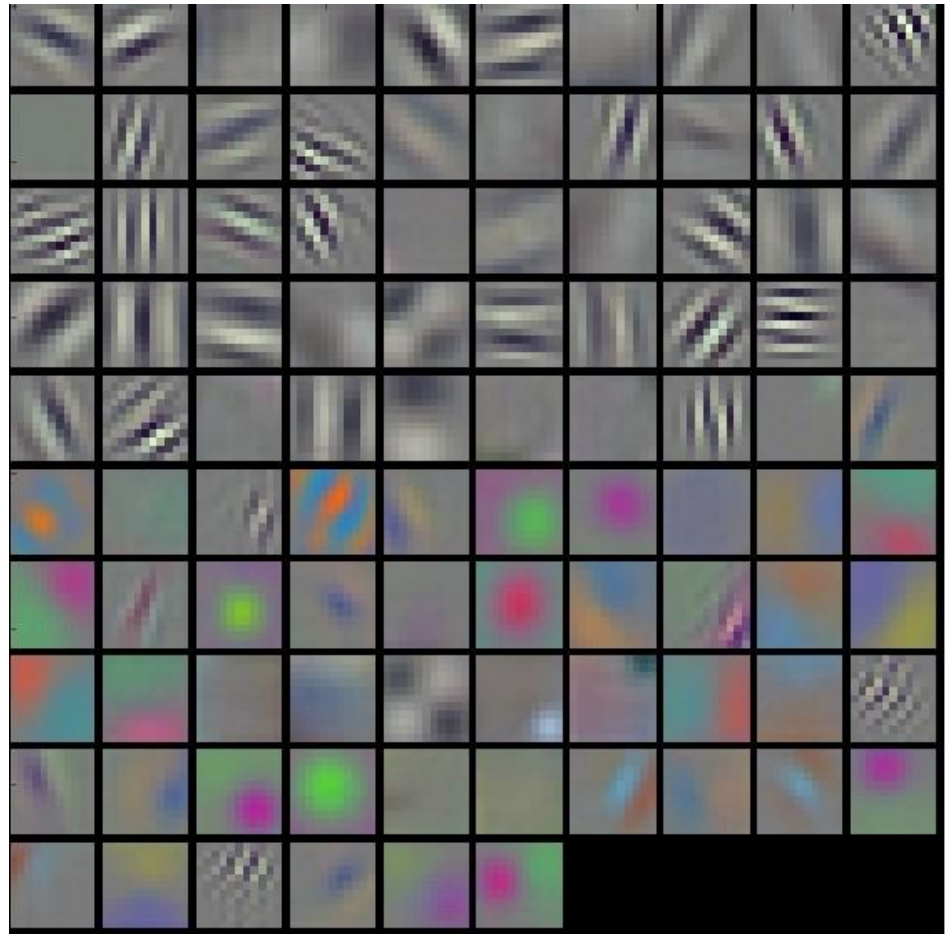


<http://newsneakernews.wpengine.netdna-cdn.com/wp-content/uploads/2016/11/rihanna-puma-creeper-velvet-release-date-02.jpg>

First Convolution Layer

- Typical-looking filters on the trained first layer

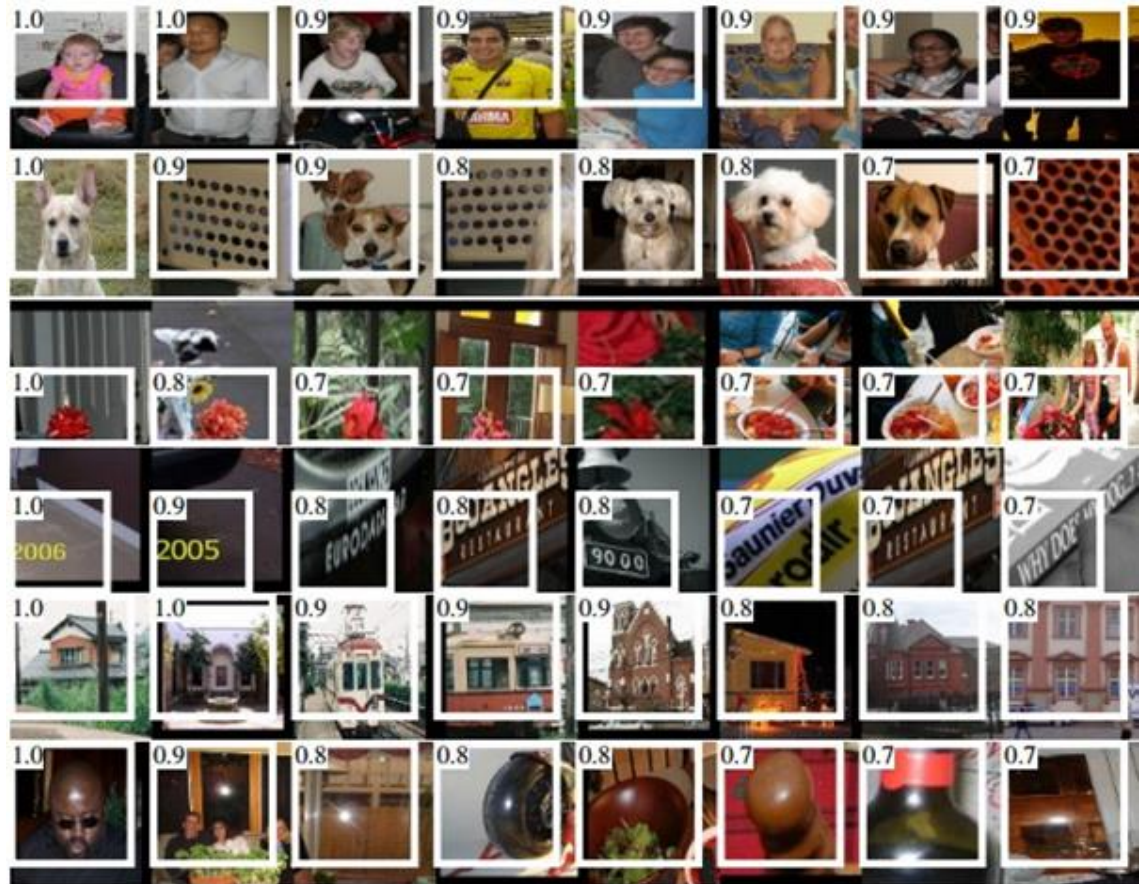
11 x 11
(AlexNet)



<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

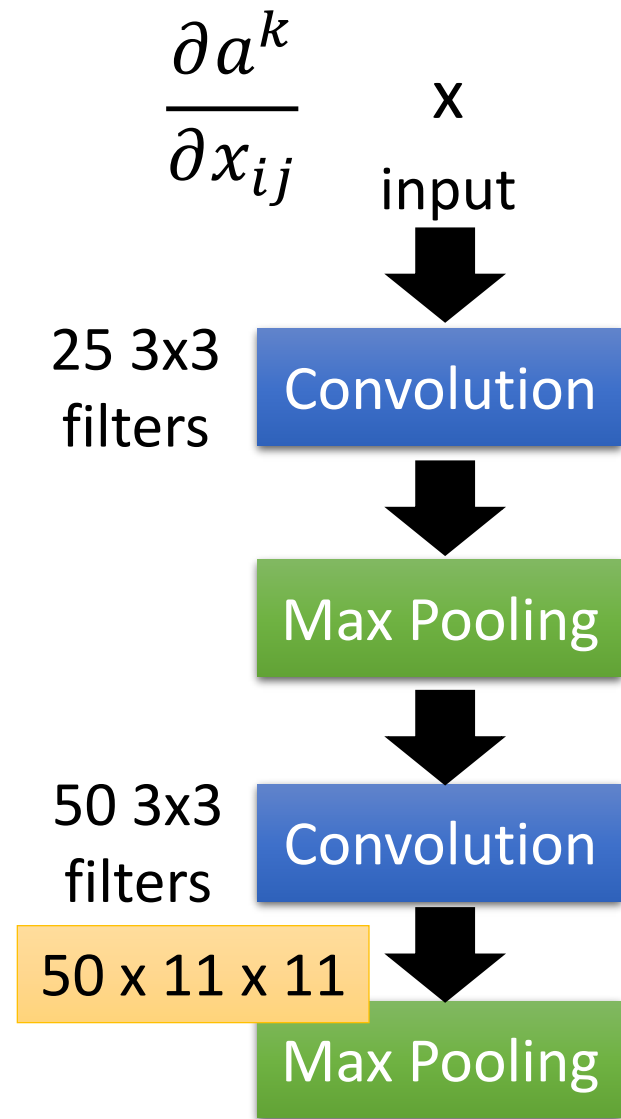
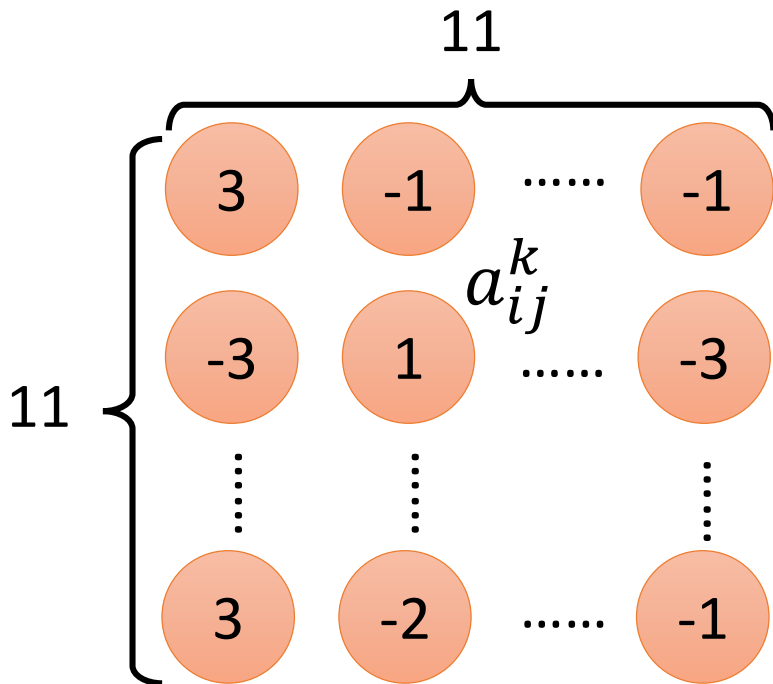
What does CNN learn?

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

Degree of the activation of the k-th filter:

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

$x^* = \arg \max_x a^k$ (gradient ascent)



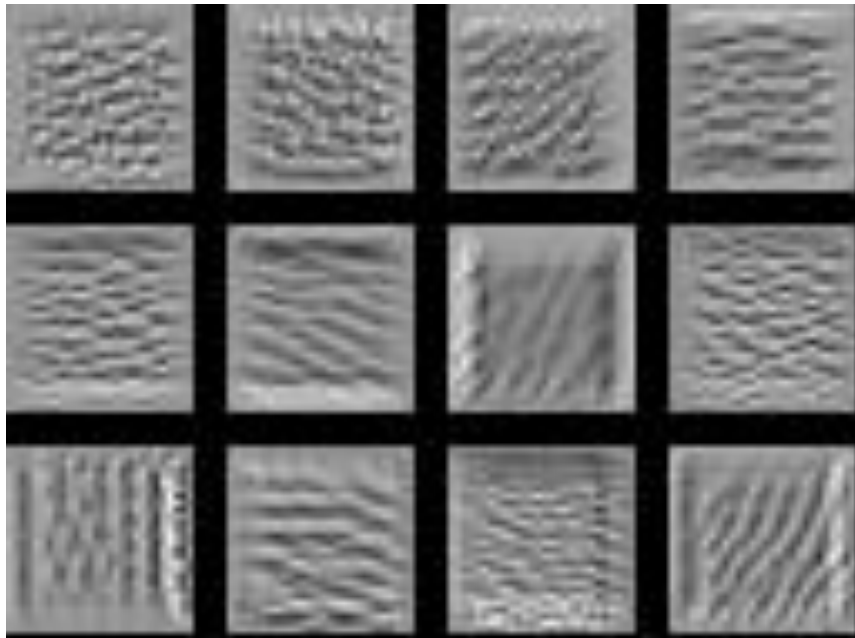
What does CNN learn?

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

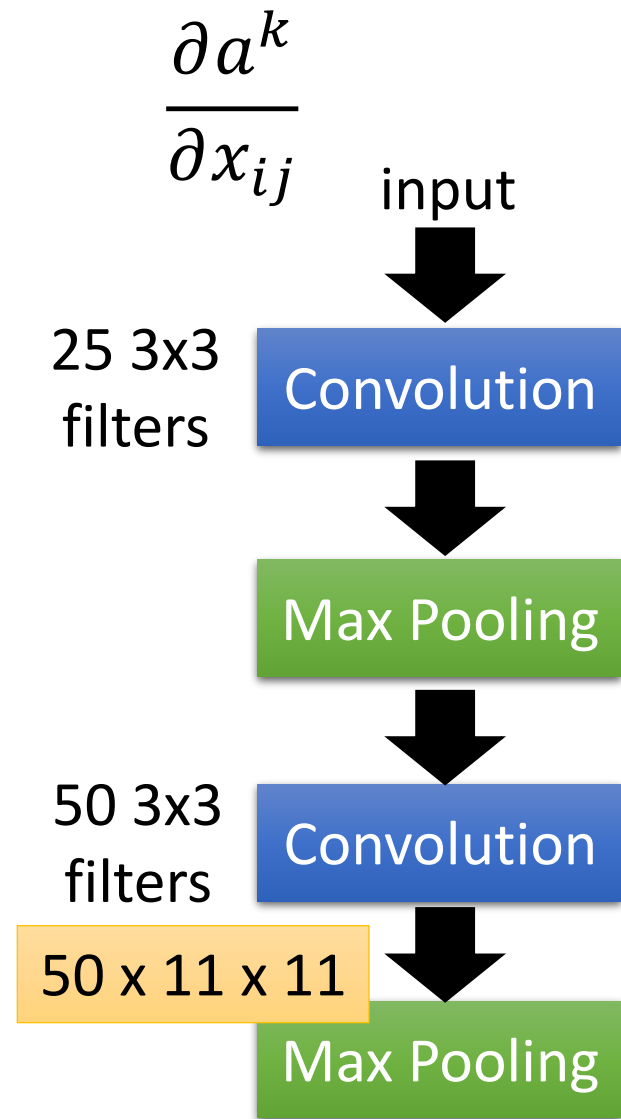
Degree of the activation of the k-th filter:

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

$x^* = \arg \max_x a^k$ (gradient ascent)



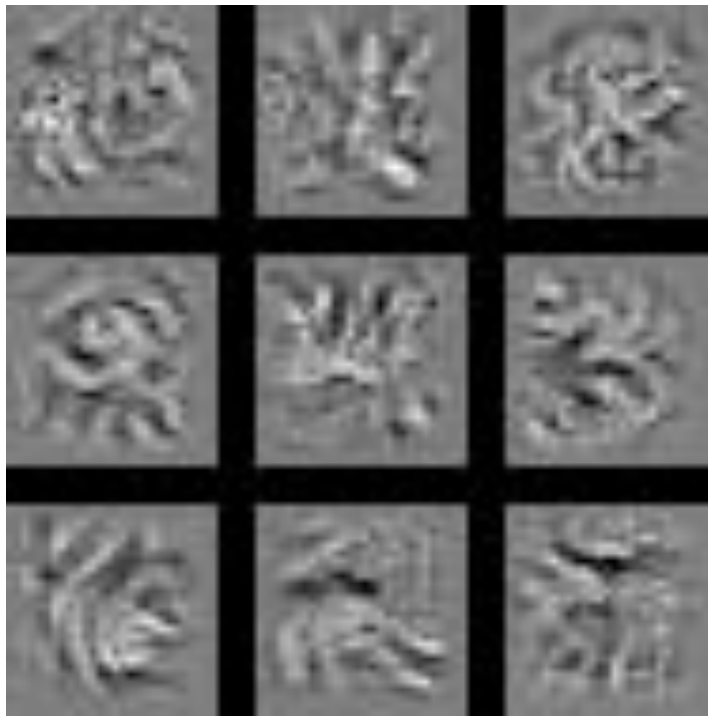
For each filter



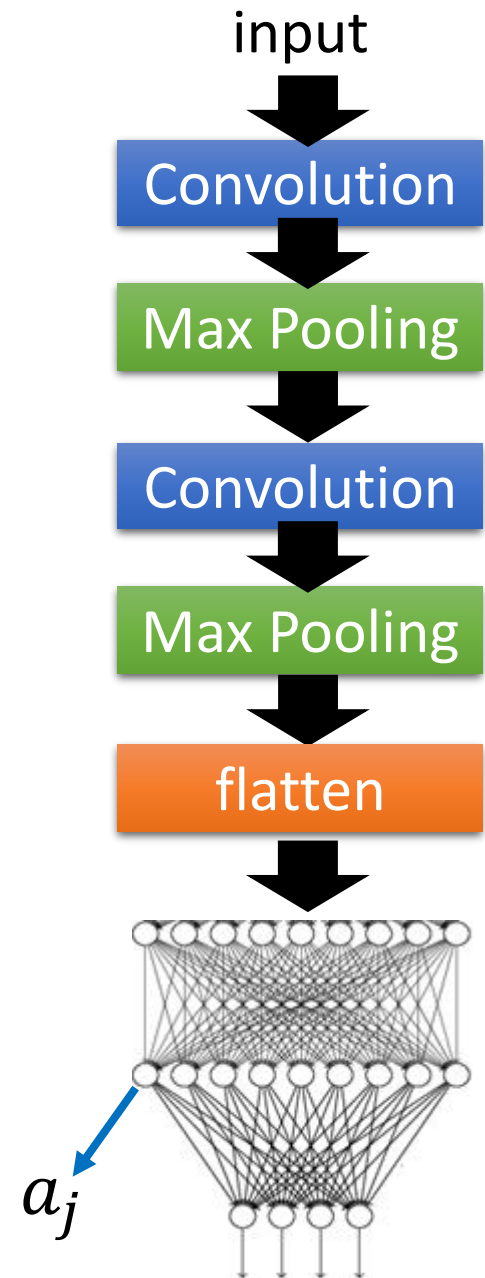
What does CNN learn?

Find an image maximizing the output of neuron:

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

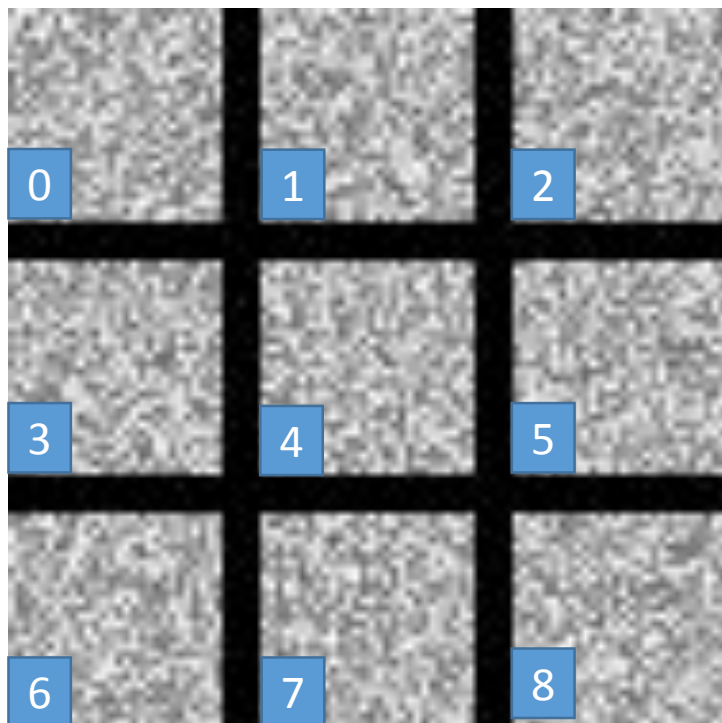


Each figure corresponds to a neuron



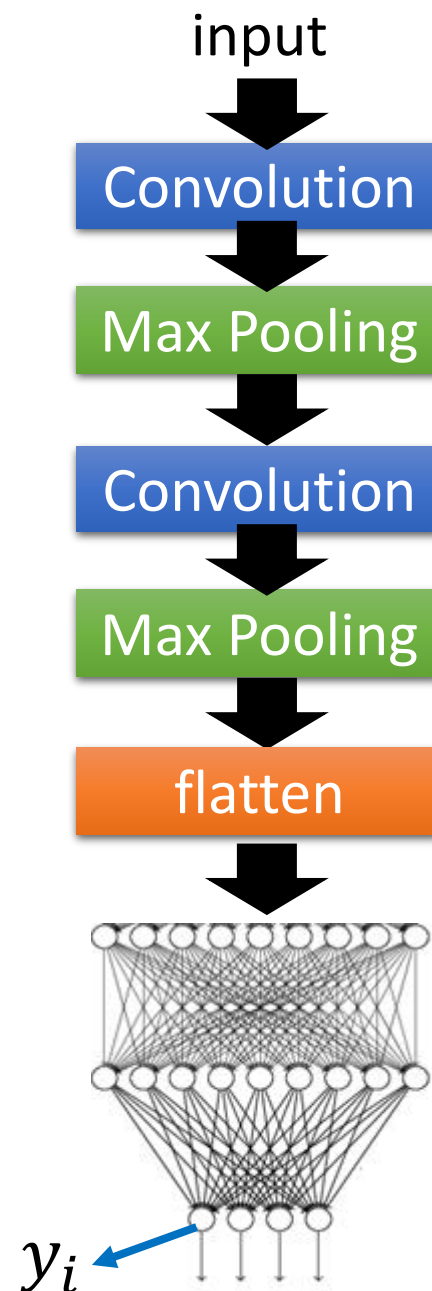
What does CNN learn?

$$x^* = \arg \max_x y^i \quad \text{Can we see digits?}$$



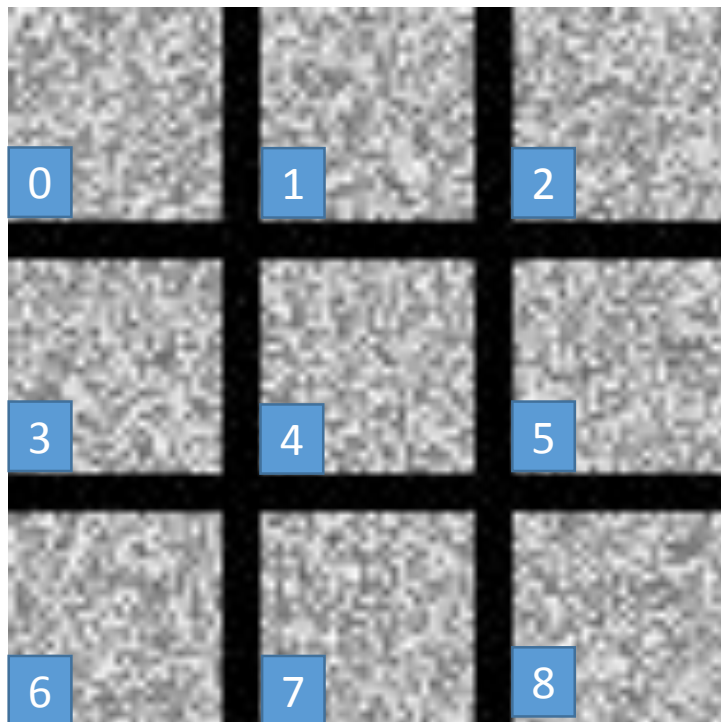
Deep Neural Networks are Easily Fooled

<https://www.youtube.com/watch?v=M2lebCN9Ht4>



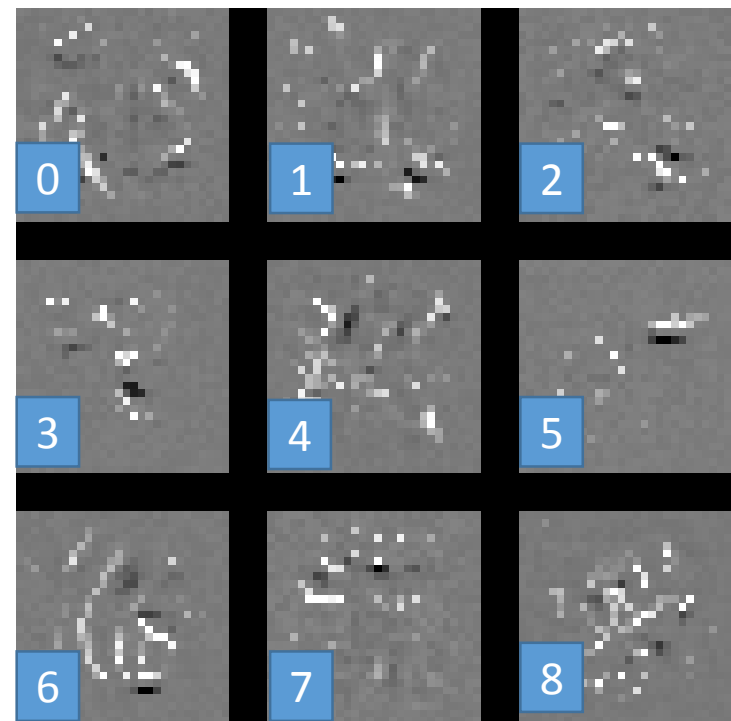
What does CNN learn?

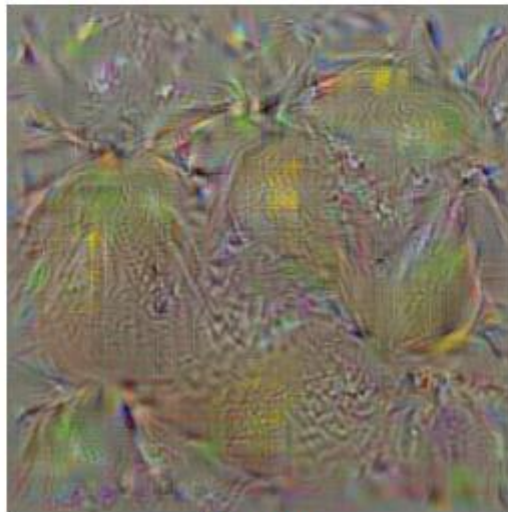
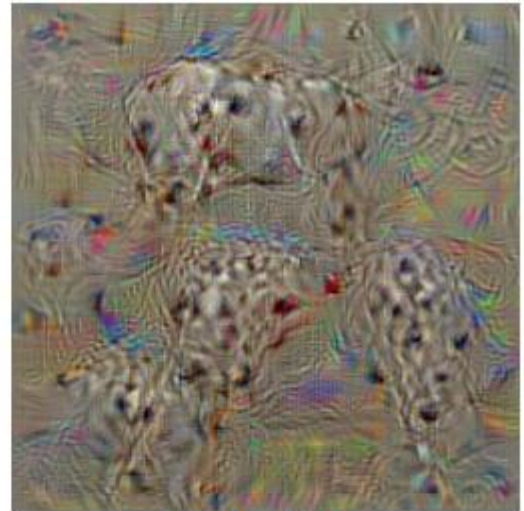
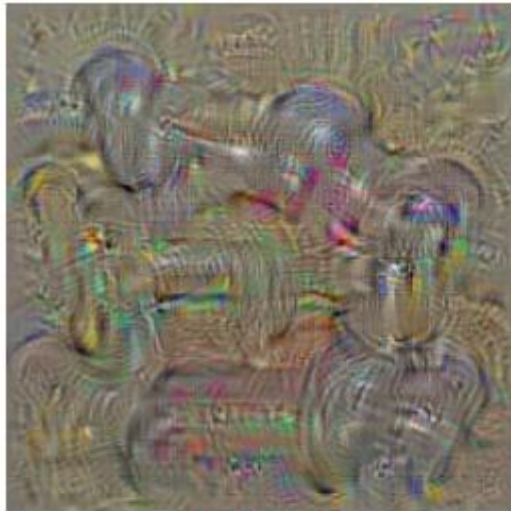
$$x^* = \arg \max_x y^i$$



Over all
pixel values

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





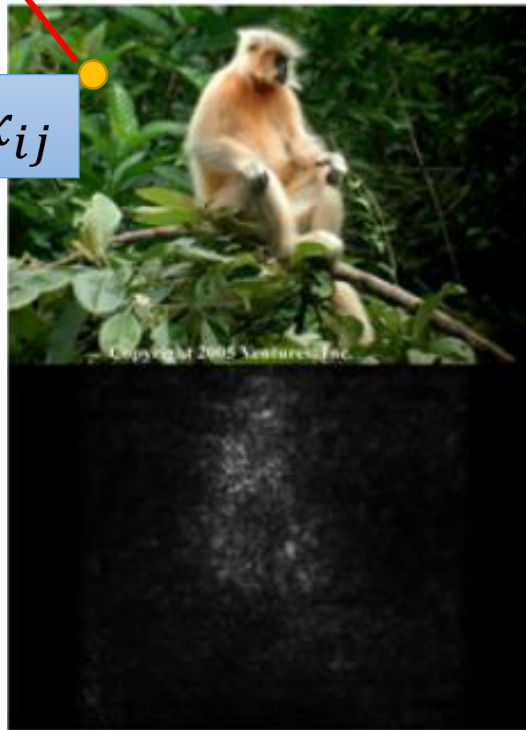
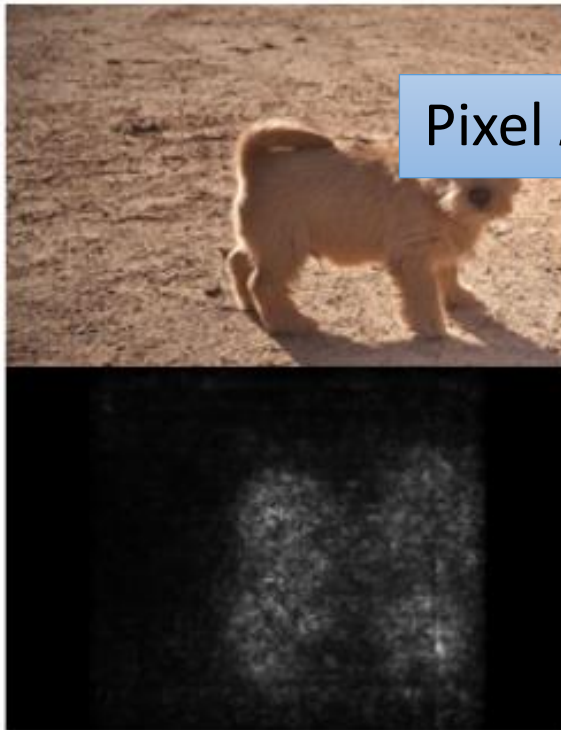
Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR, 2014

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

y_k : the predicted
class of the model



Pixel x_{ij}

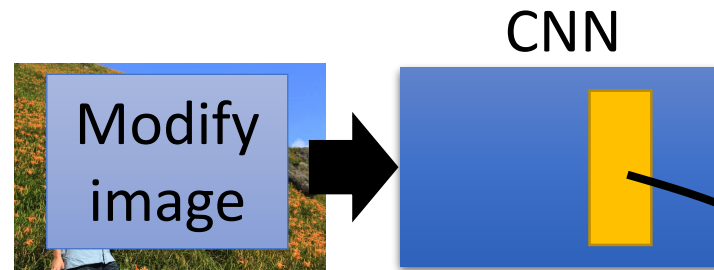


Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR, 2014



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

Deep Dream



- Given a photo, machine adds what it sees



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

Green arrow pointing up next to 3.9
Orange arrow pointing down next to -1.5
Green arrow pointing up next to 2.3

Deep Dream

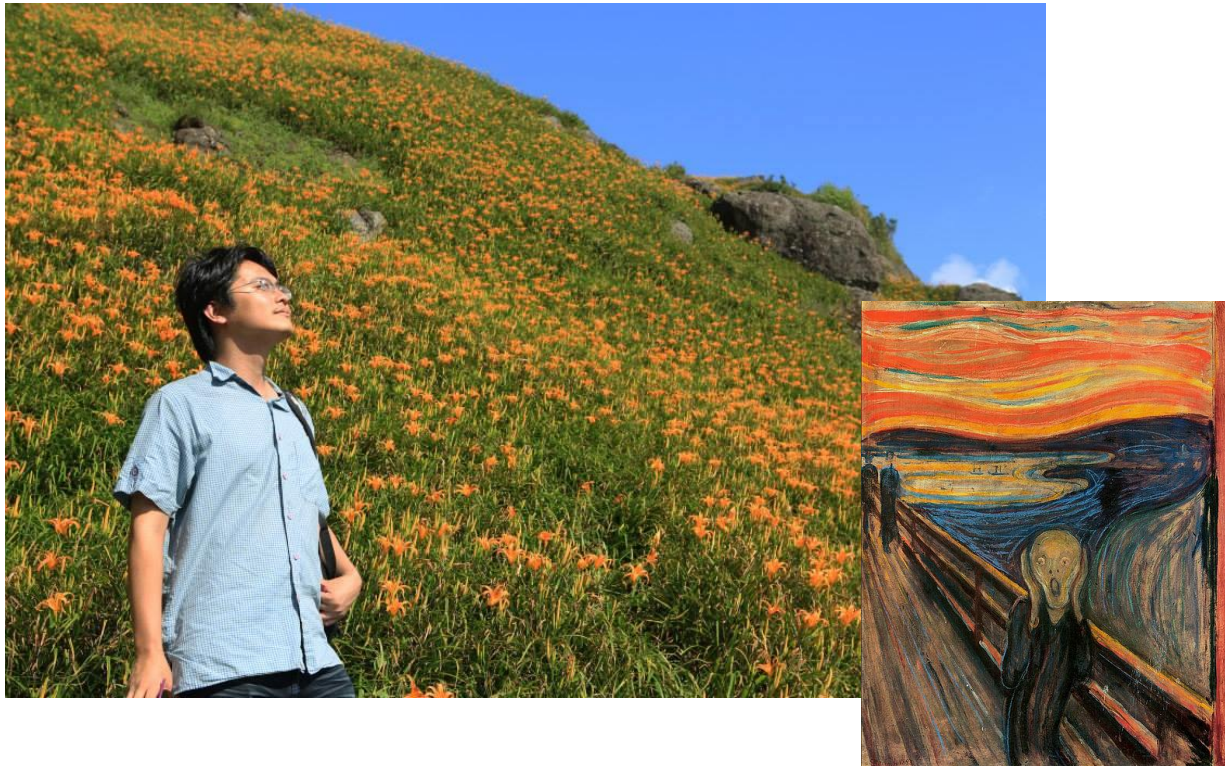
- Given a photo, machine adds what it sees



<http://deepdreamgenerator.com/>

Deep Style

- Given a photo, make its style like famous paintings



<https://dreamscopeapp.com/>

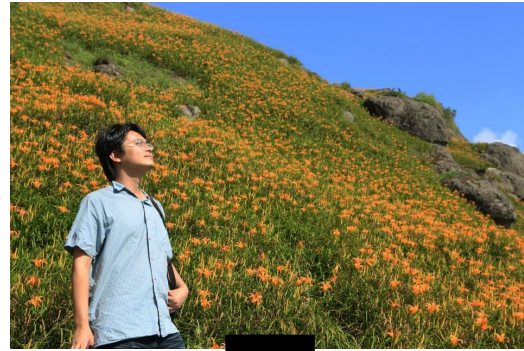
Deep Style

- Given a photo, make its style like famous paintings



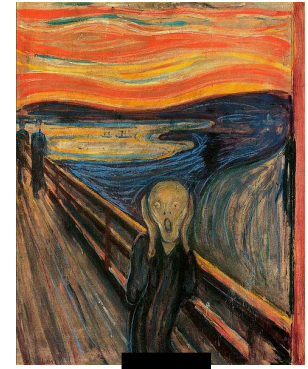
<https://dreamscopeapp.com/>

Deep Style



CNN

content



CNN

style

A Neural
Algorithm of
Artistic Style

<https://arxiv.org/abs/1508.06576>



CNN

?

More Application: Playing Go



Black: 1
white: -1
none: 0



Network



Next move
(19 x 19
positions)

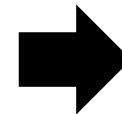
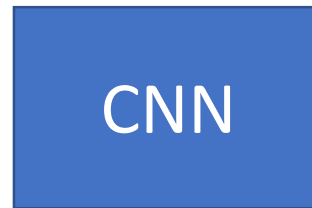
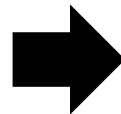
19 x 19 vector

Fully-connected feedforward
network can be used

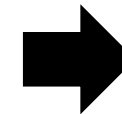
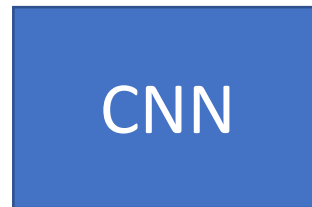
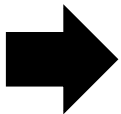
But CNN performs much better.

More Application: Playing Go

Training: record of previous plays 黒: 5之五 → 白: 天元 → 黒: 五之5 ...



Target:
“天元” = 1
else = 0

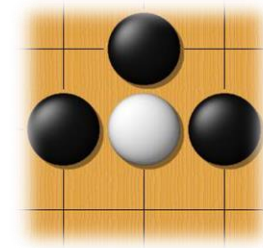


Target:
“五之5” = 1
else = 0

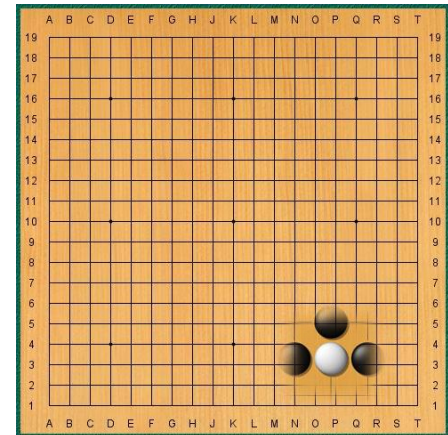
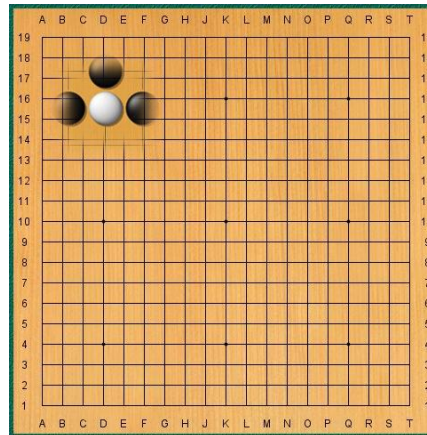
Why CNN for playing Go?

- 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



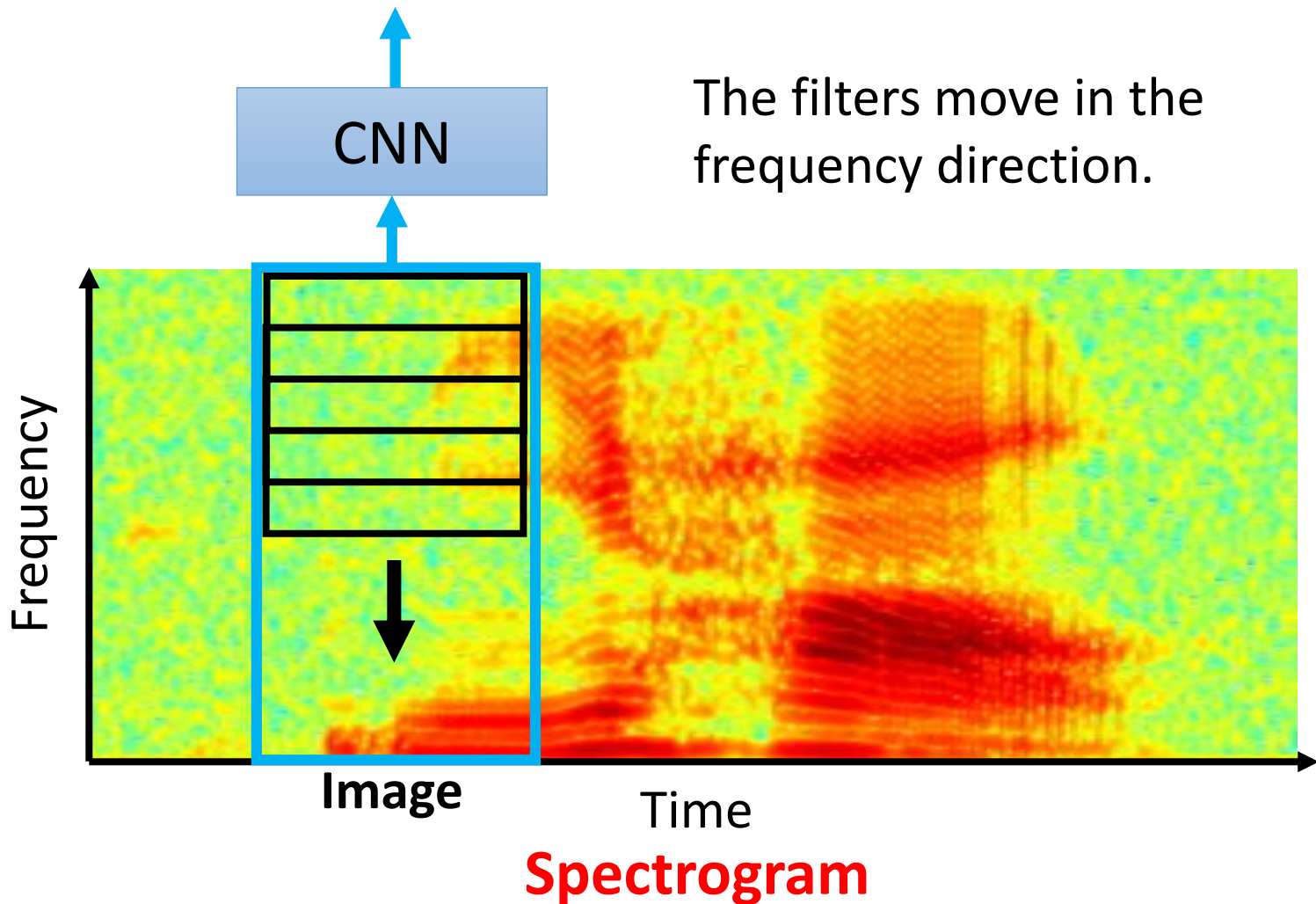
Max Pooling

How to explain this???

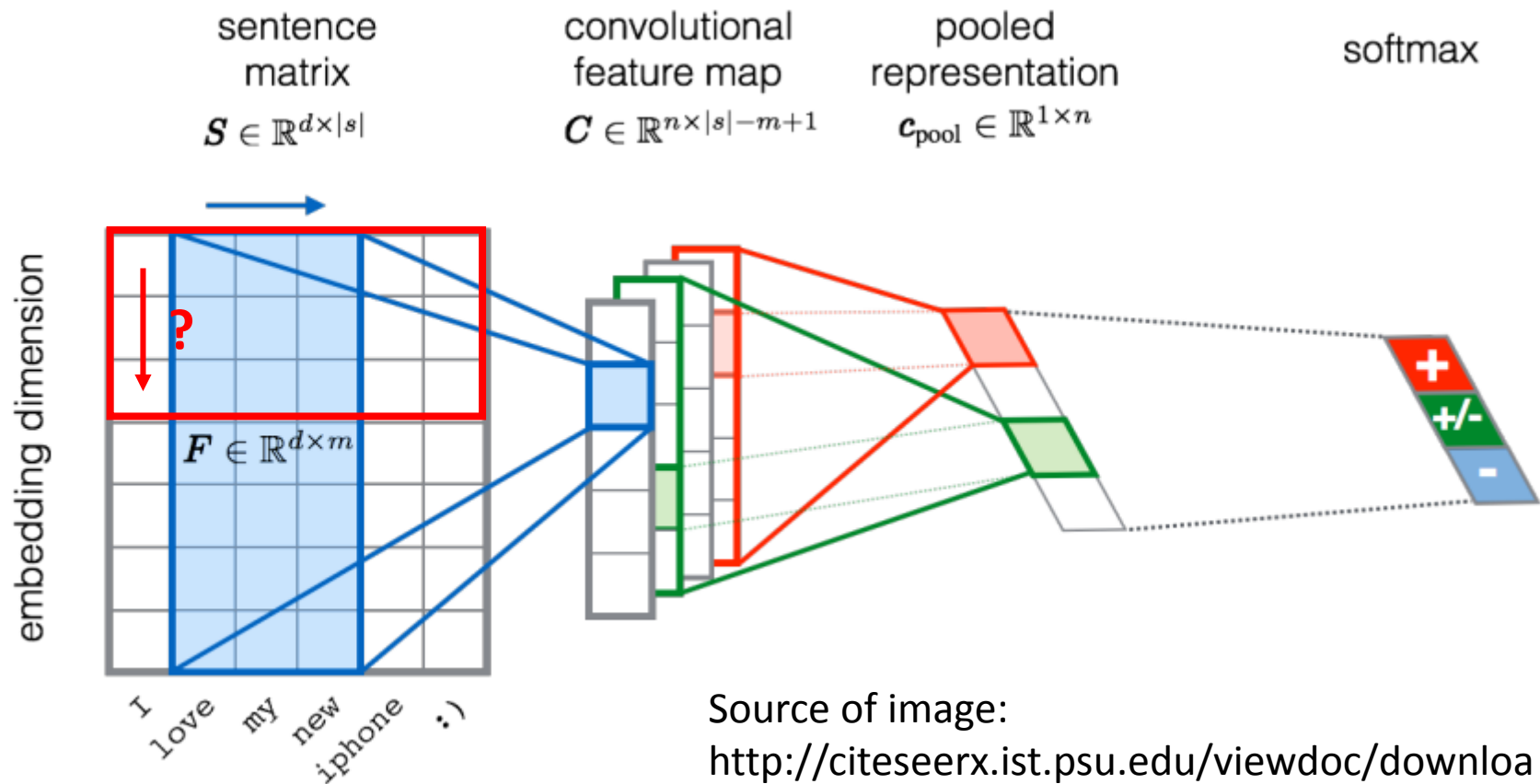
Neural network architecture. The input to the policy network is a $19 \times 19 \times 48$ image stack consisting of 48 feature planes. The first hidden layer zero pads the input into a 23×23 image, then convolves k filters of kernel size 5×5 with stride 1 with the input image and applies a rectifier 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



More Application: Text



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