



HCMUTE

# Artificial Neural Network

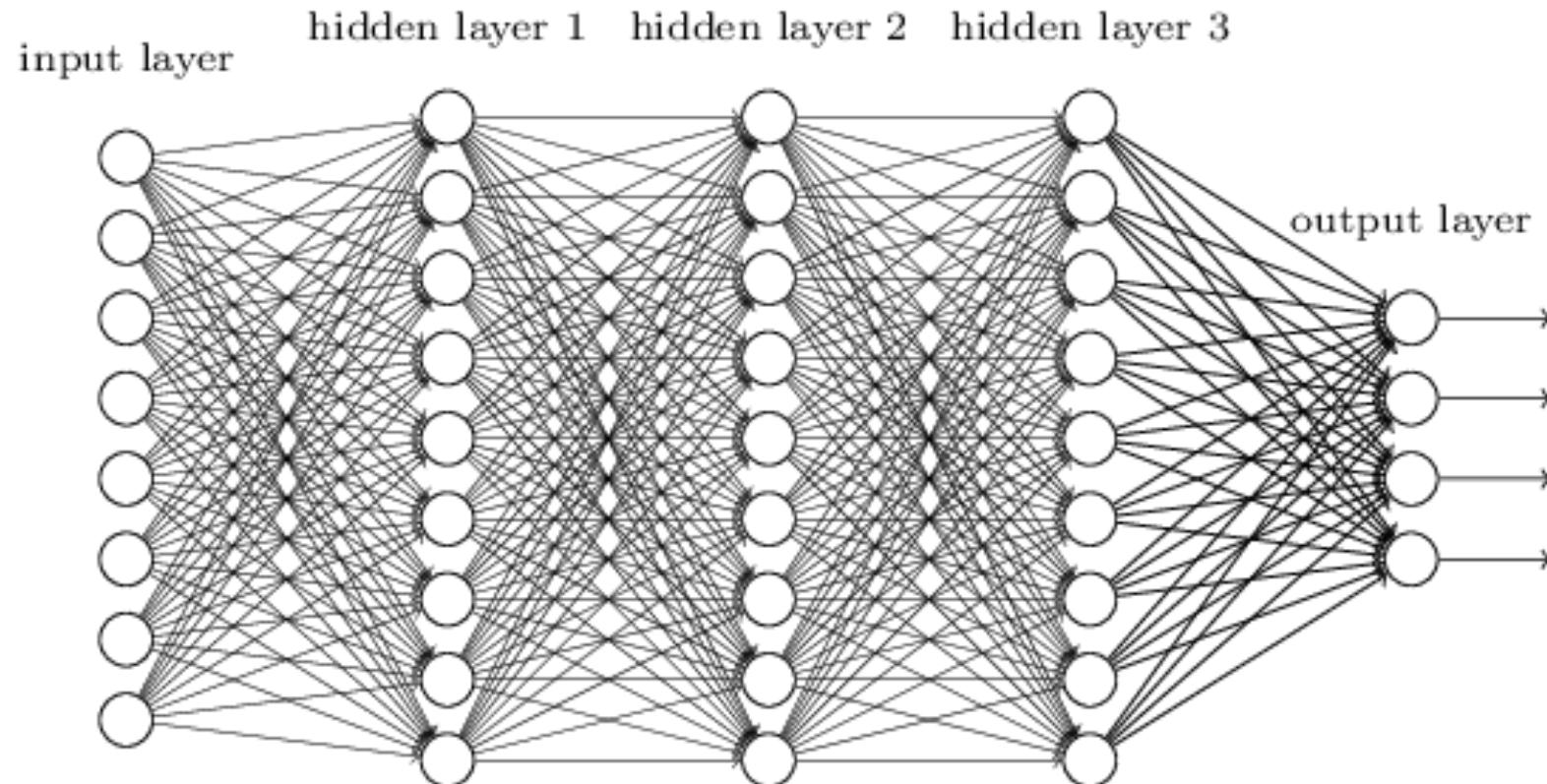


Dr. Trần Vũ Hoàng



# Smaller Network: CNN

- We know it is **good** to learn a **small model**.
- From this fully connected model, do we really need all the edges?
- Can some of these be shared?



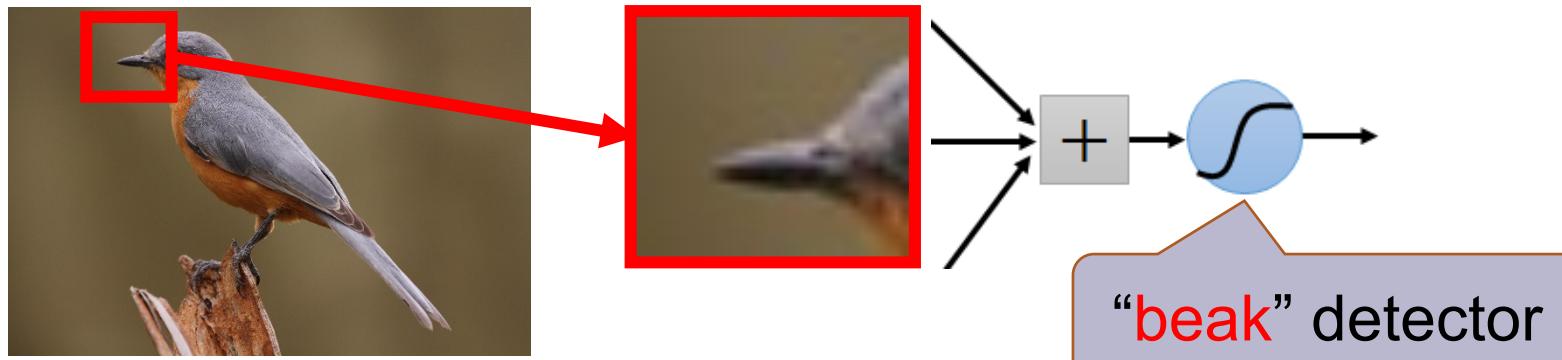


HCMUTE

# Consider learning an image:

- Some patterns are much smaller than the whole image

Can represent a small region with fewer parameters

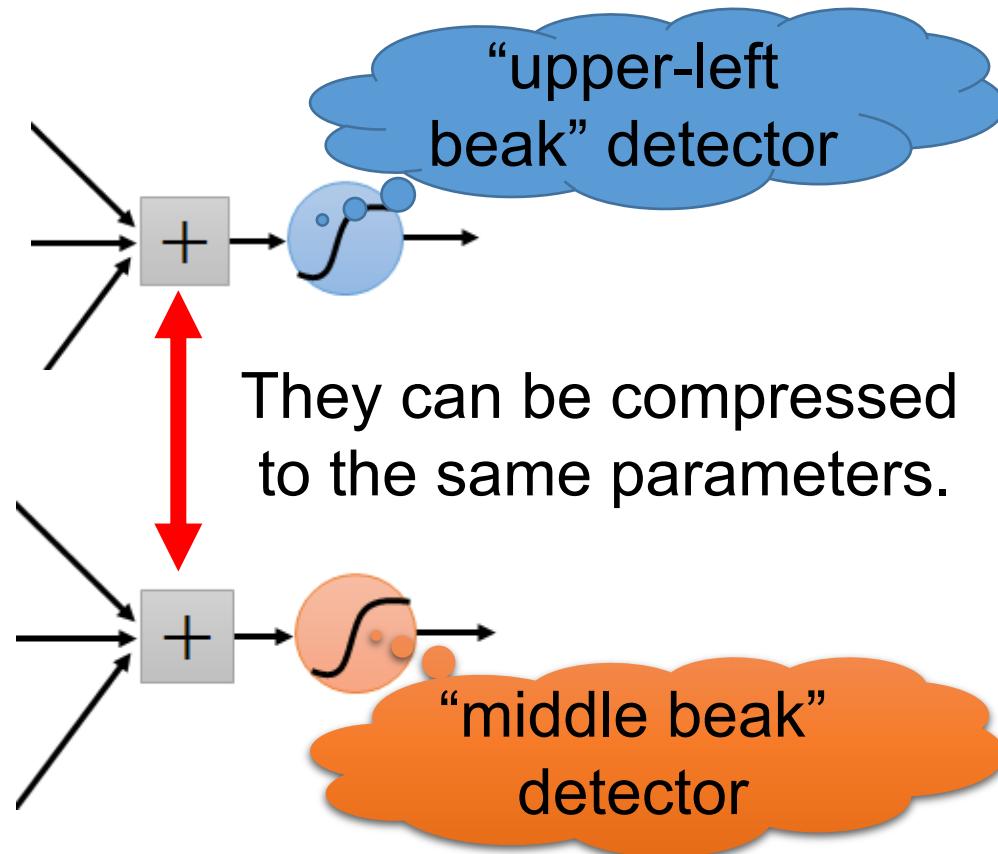




# Consider learning an image:

Same pattern appears in different places:  
They can be compressed!

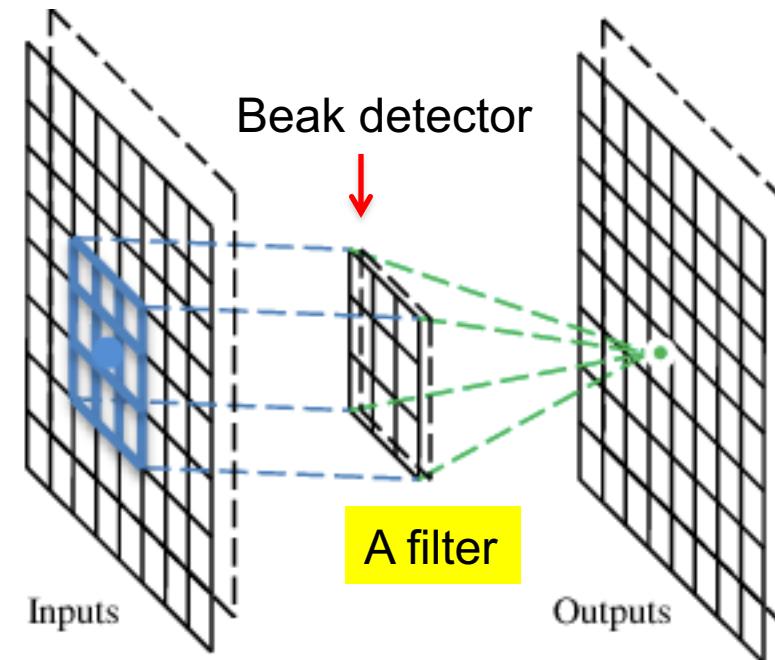
What about training a lot of such “small” detectors  
and each detector must “move around”.





# A convolutional layer

A CNN is a neural network with some convolutional layers (and some other layers). A convolutional layer has a number of filters that does convolutional operation.





# Convolution

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

These are the network parameters to be learned.

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

Filter 1

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

Filter 2

: :

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



HCMUTE

# 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

Dot  
product



6 x 6 image



HCMUTE

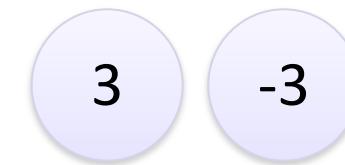
# Convolution

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

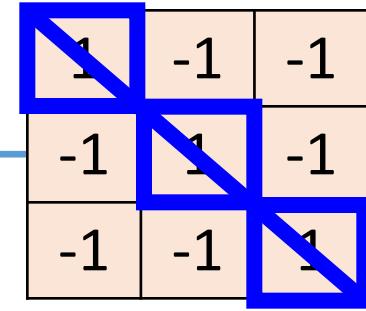


6 x 6 image

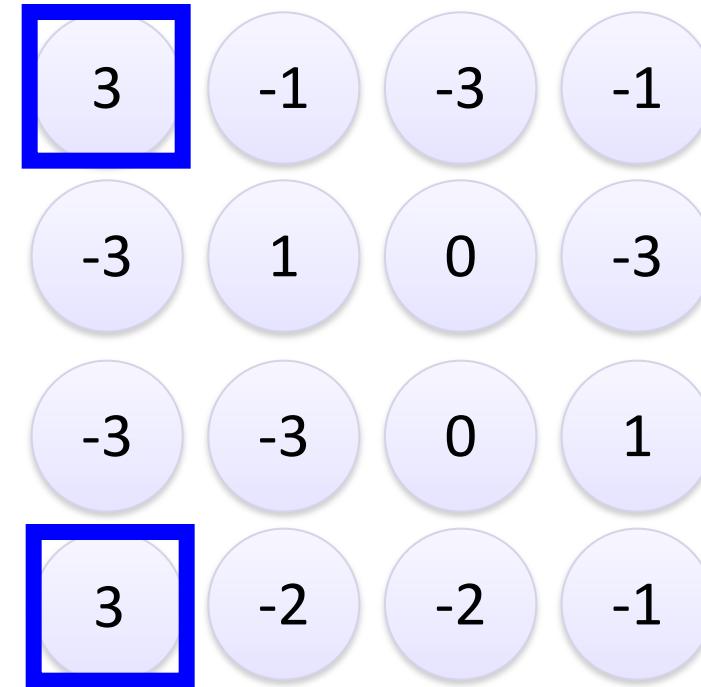
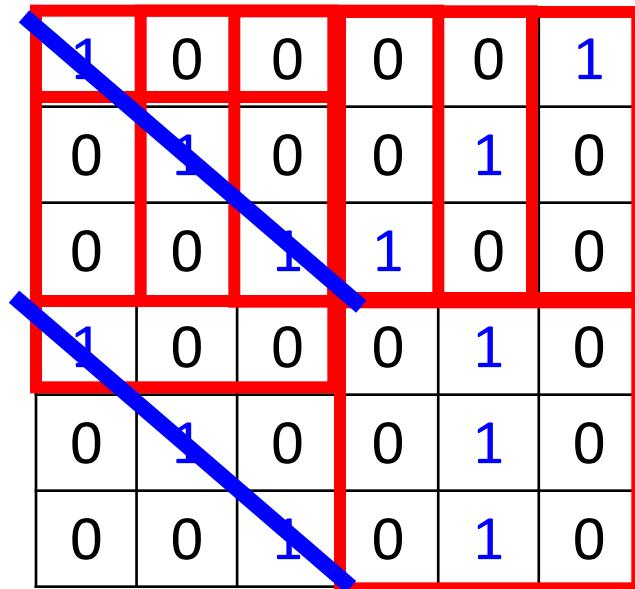


HCMUTE

# Convolution



stride=1





HCMUTE

# Convolution

stride=1

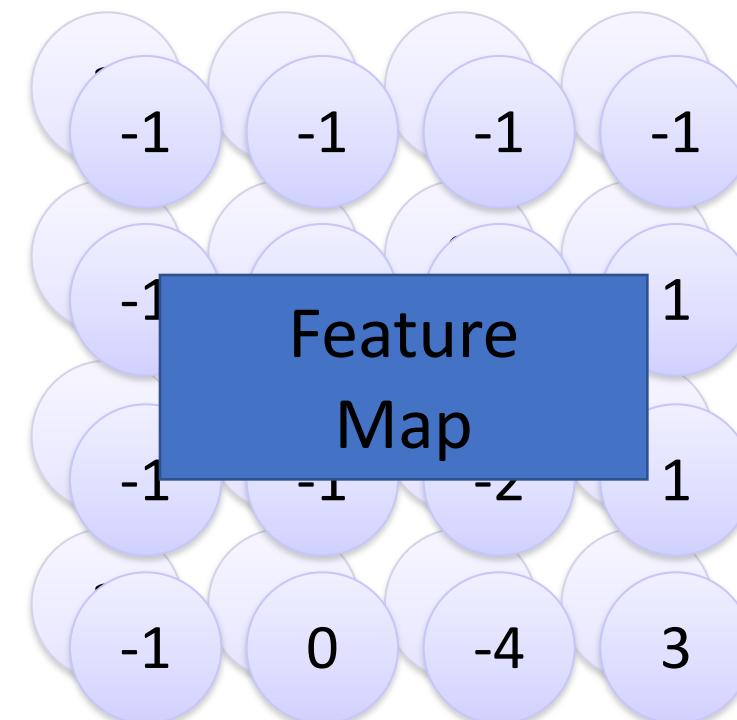
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

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

Filter 2

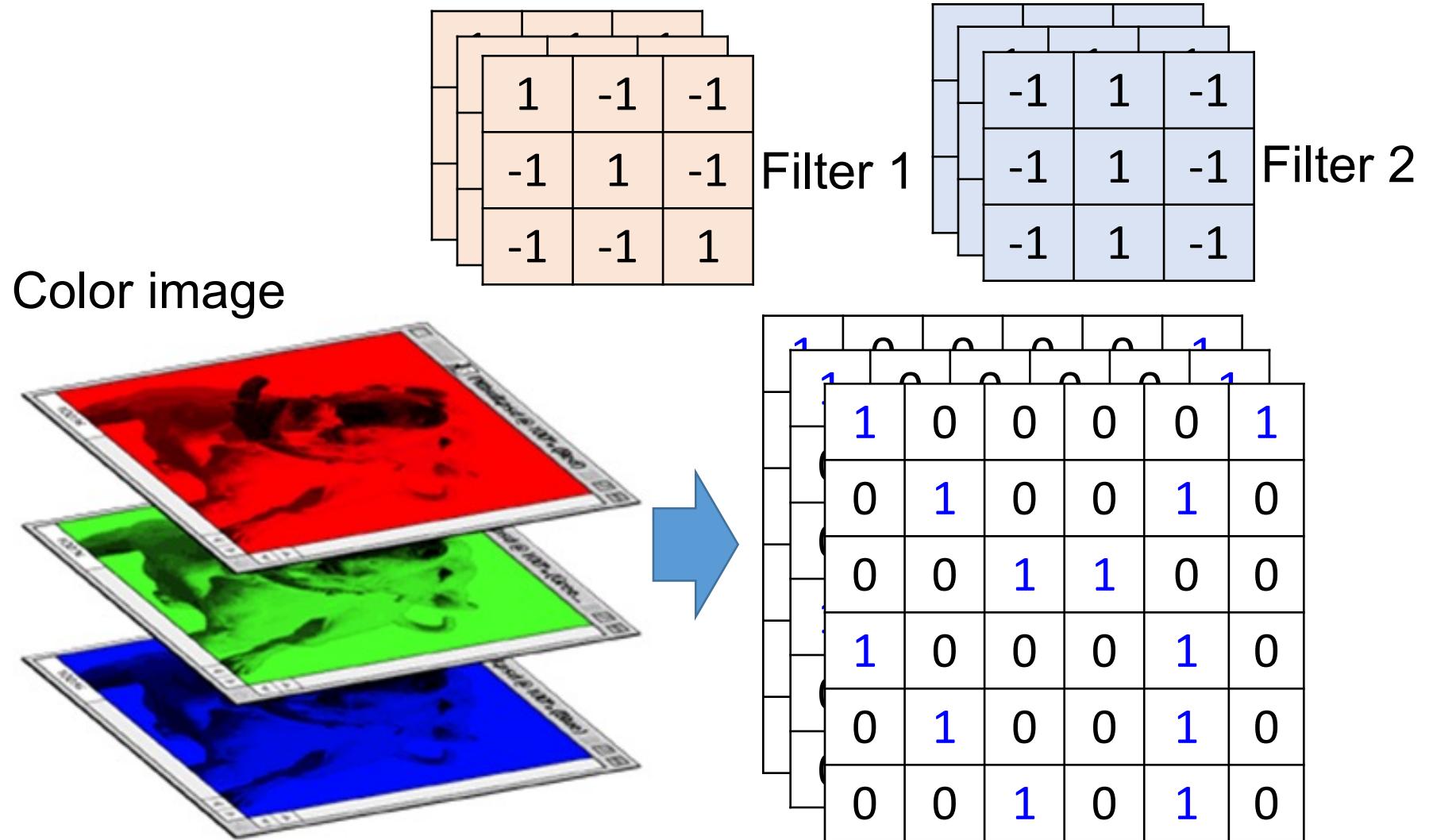
Repeat this for each filter



Two 4 x 4 images  
Forming 2 x 4 x 4 matrix



# Color image: RGB 3 channels





# Convolution v.s. Fully Connected

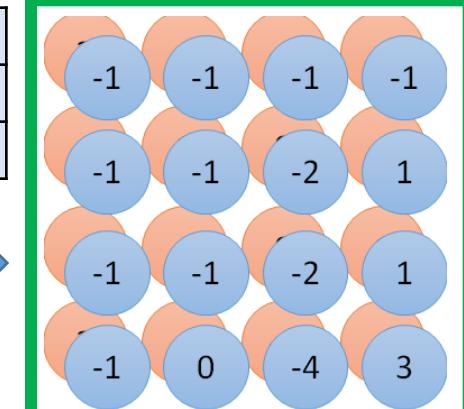
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

image

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

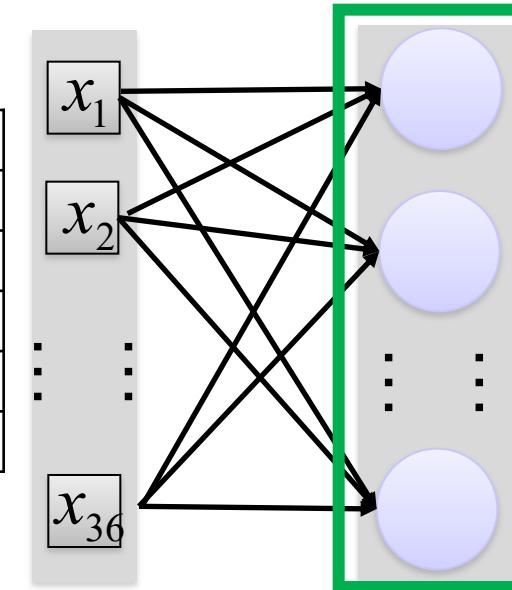
-1	1	-1
-1	1	-1
-1	1	-1

convolution



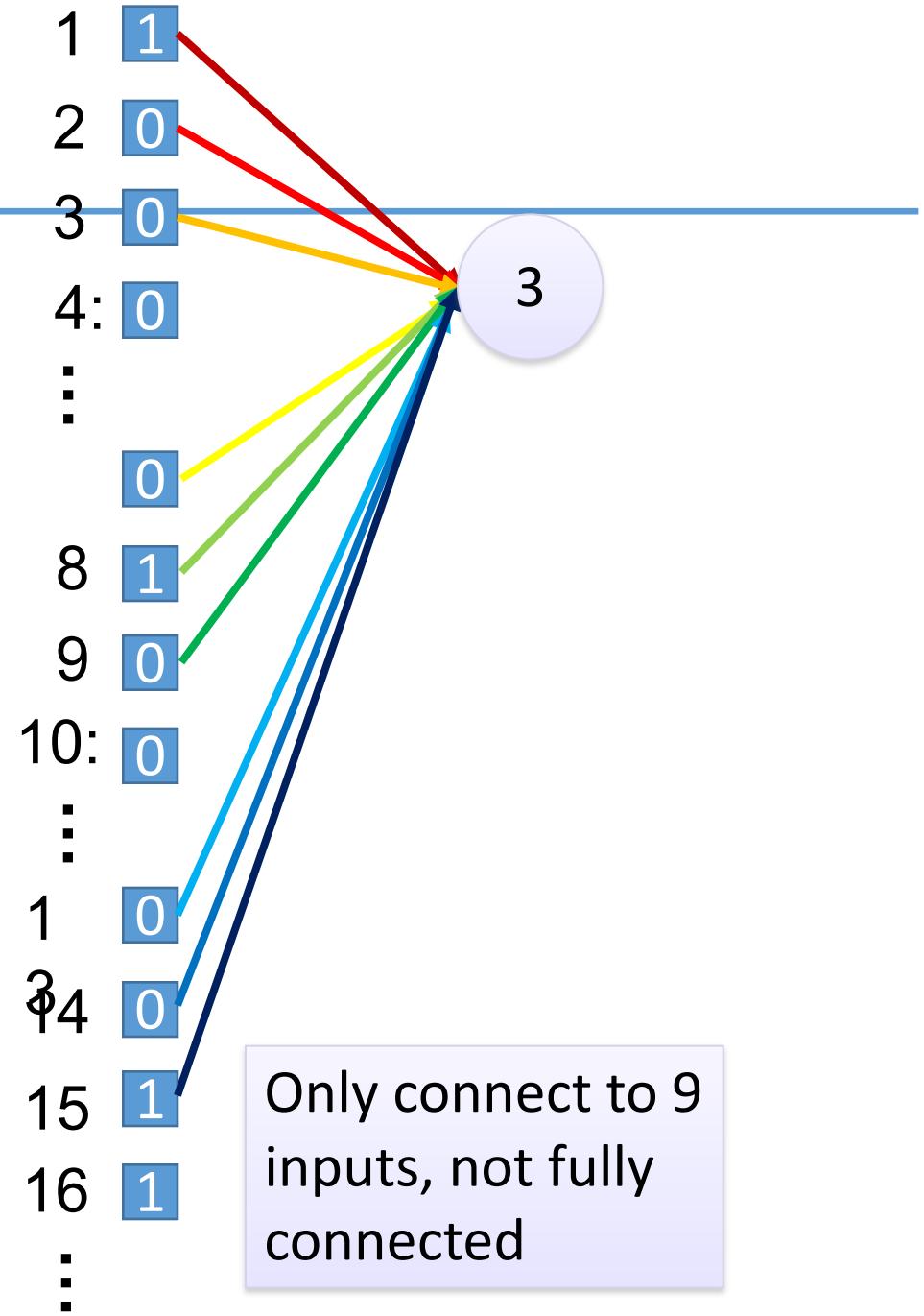
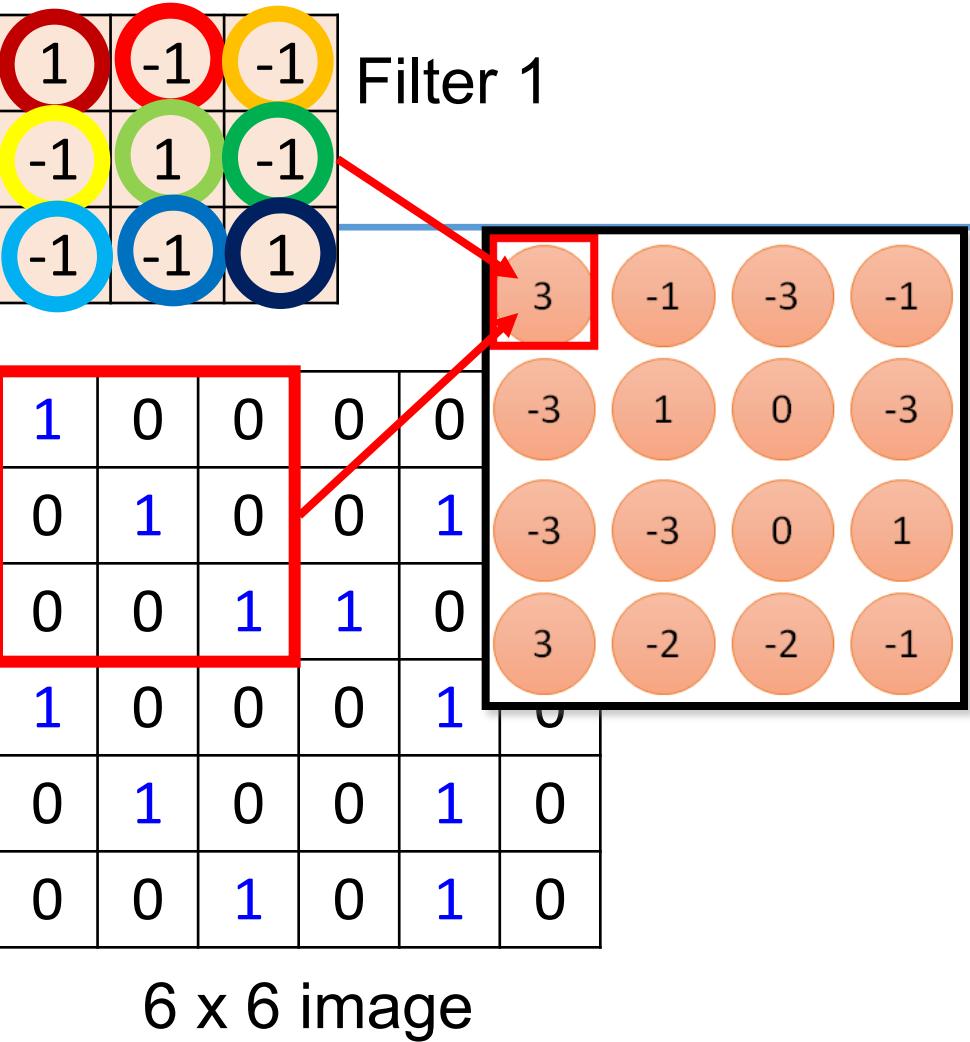
Fully-connected

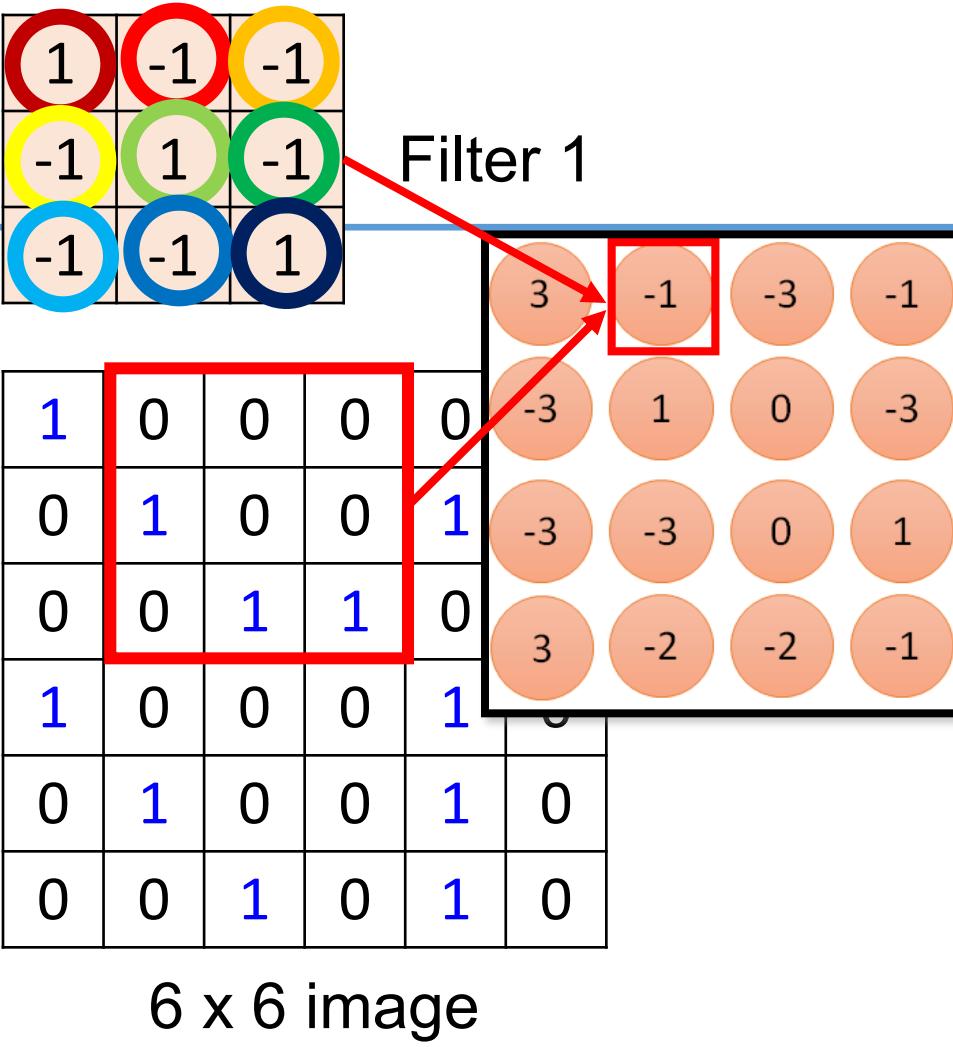
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0





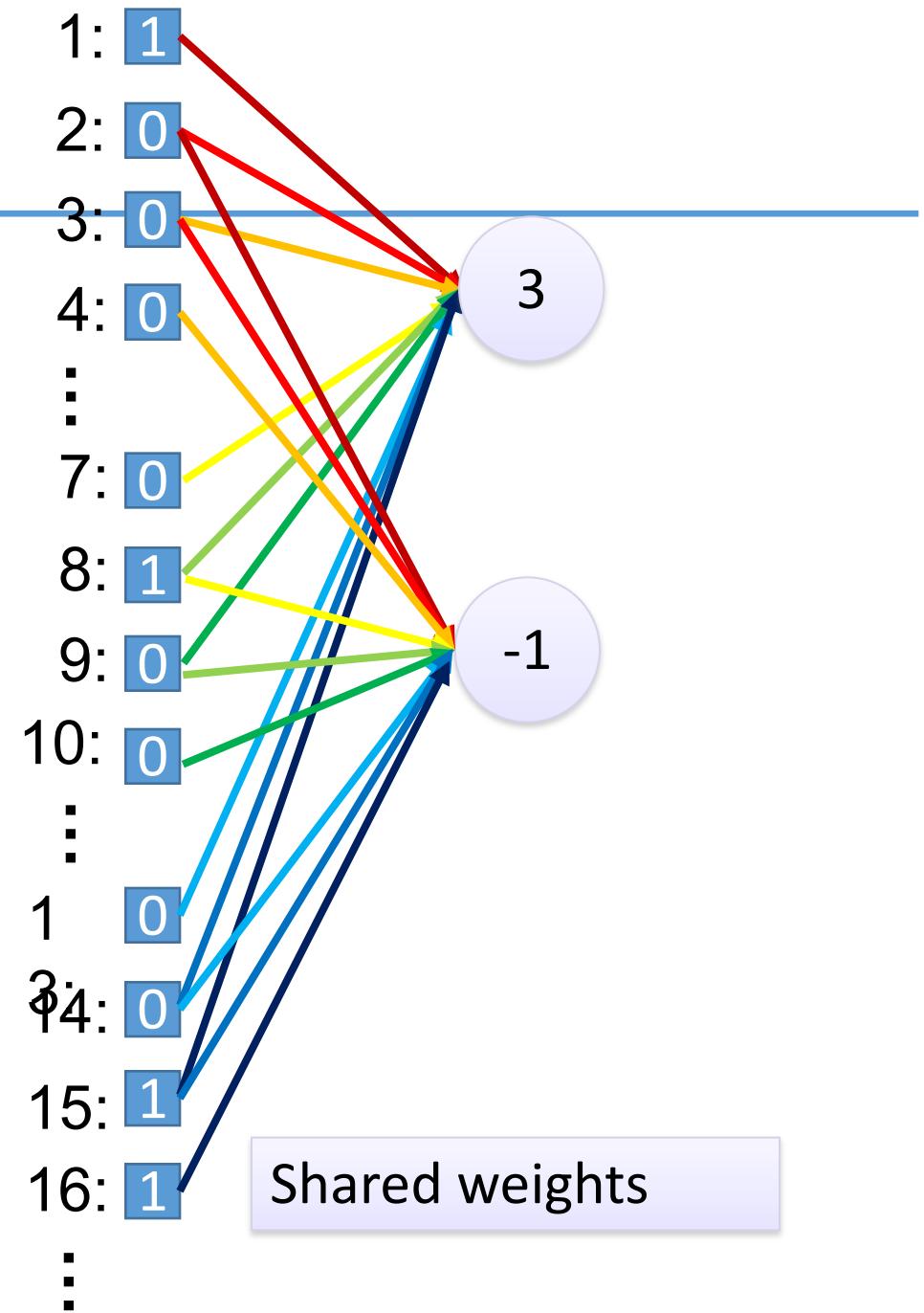
HCMUTE





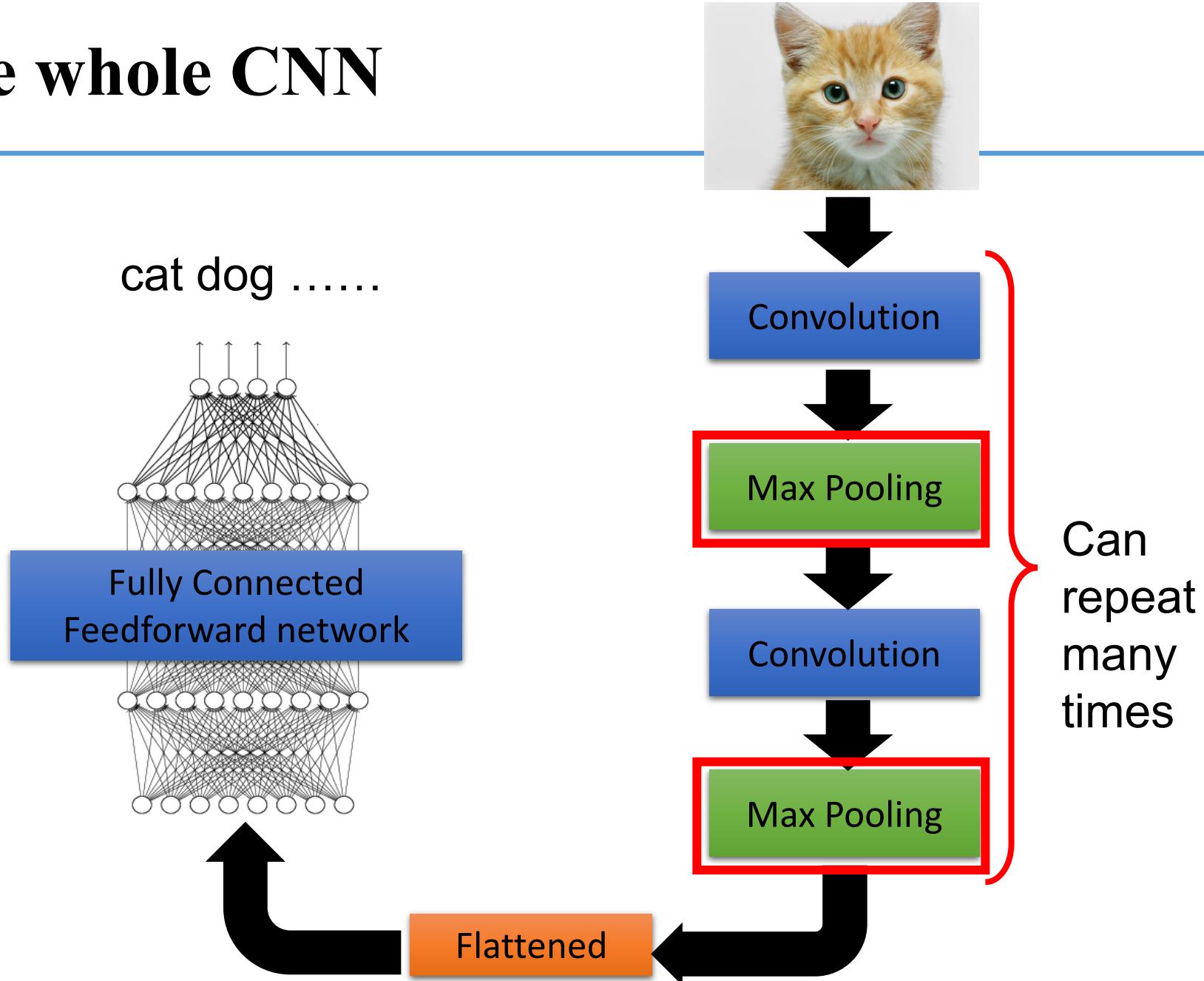
Fewer parameters

Even fewer parameters





# The whole CNN





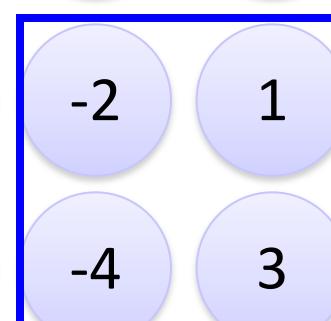
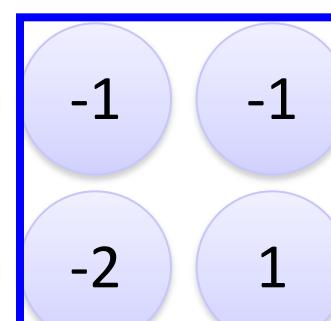
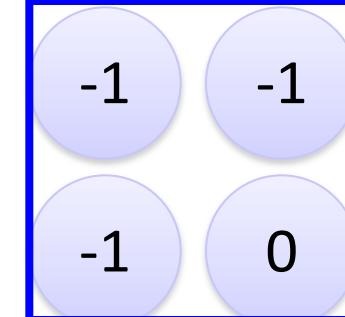
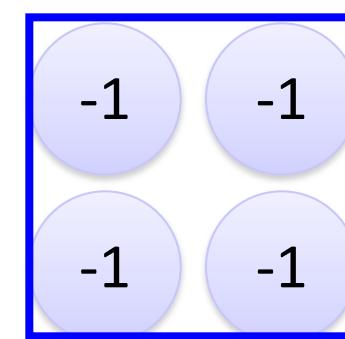
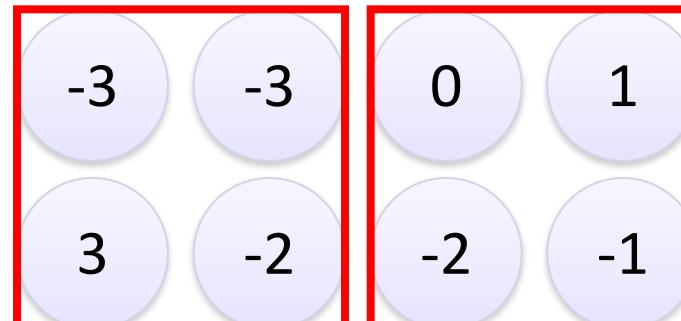
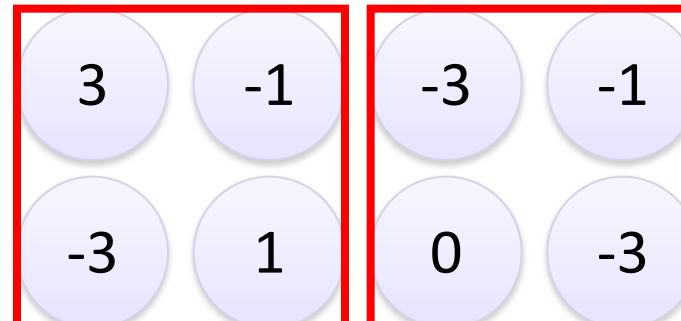
# Max Pooling

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

Filter 1

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

Filter 2





# Why Pooling?

- Subsampling pixels will not change the object

bird



Subsampling

bird



We can subsample the pixels to make image smaller



**fewer parameters** to characterize the image



# A CNN compresses a fully connected network in two ways:

---

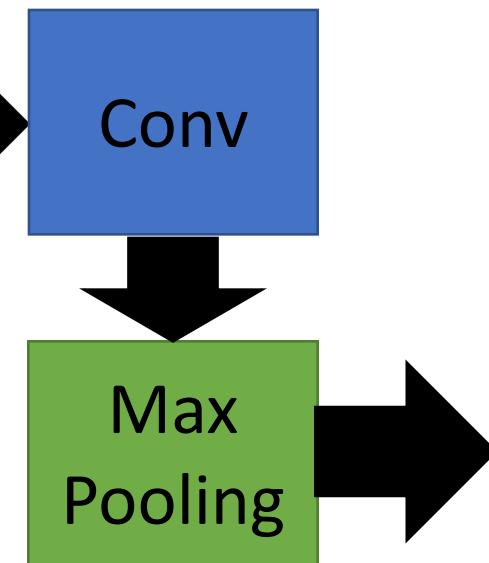
- Reducing number of connections
- Shared weights on the edges
- Max pooling further reduces the complexity



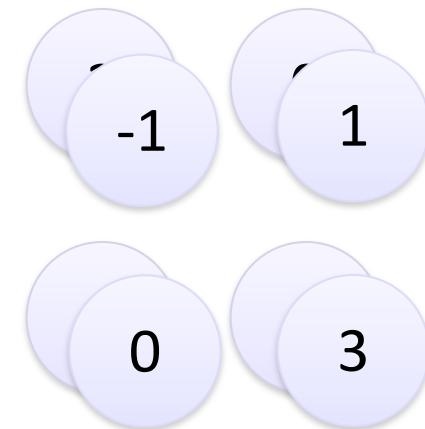
# Max Pooling

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



New image  
but smaller



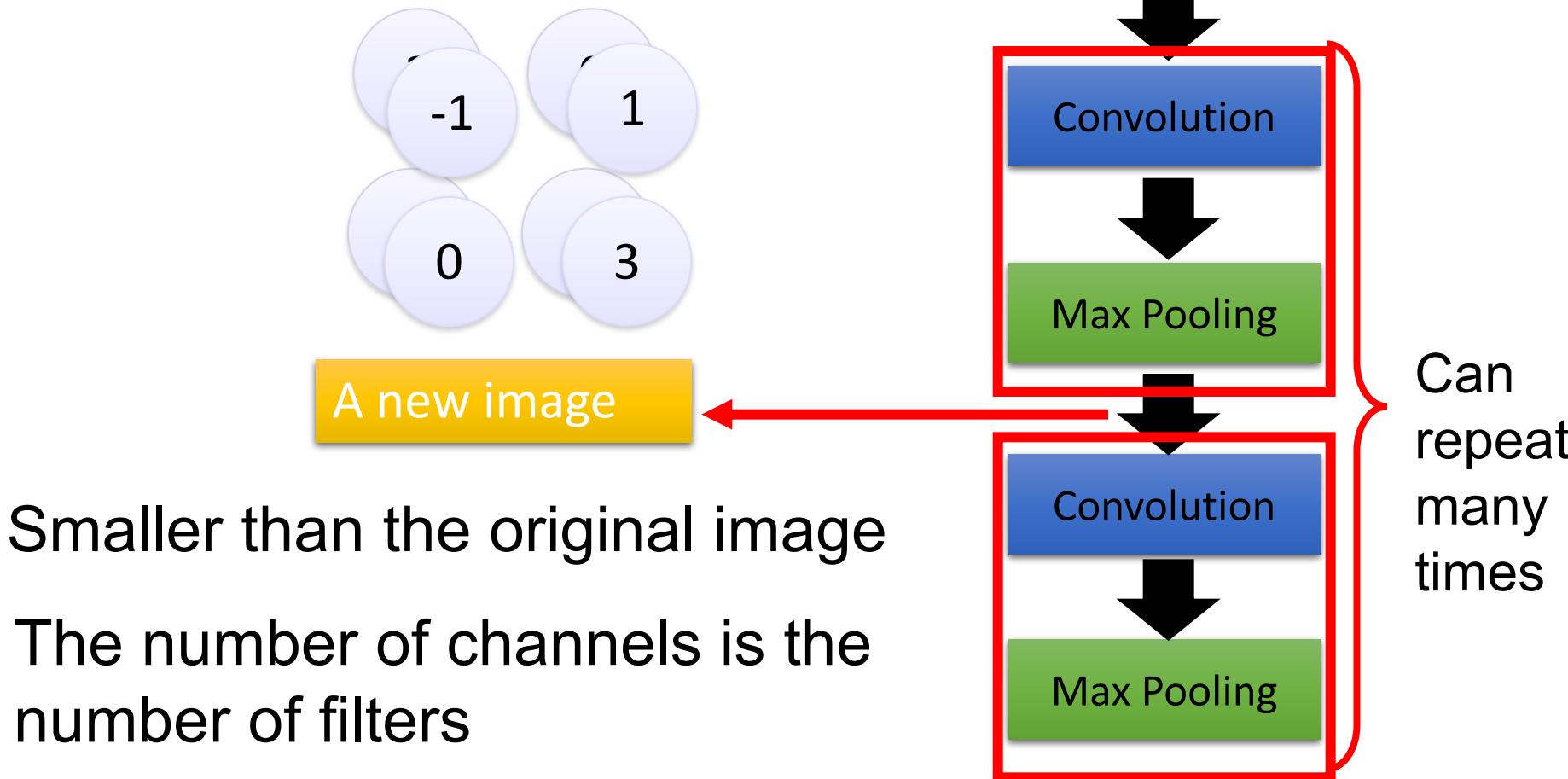
2 x 2 image

Each filter  
is a channel



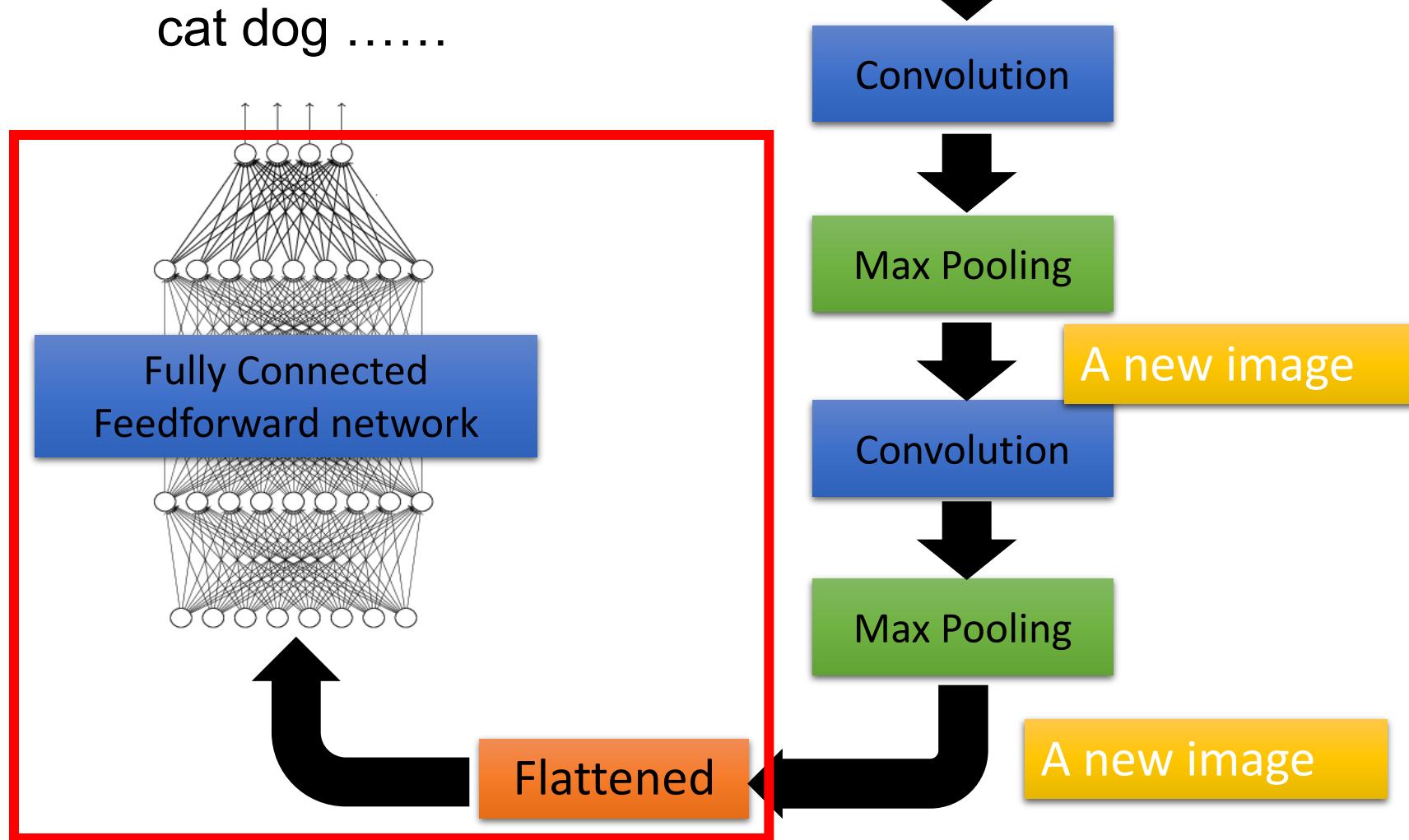
HCMUTE

# The whole CNN



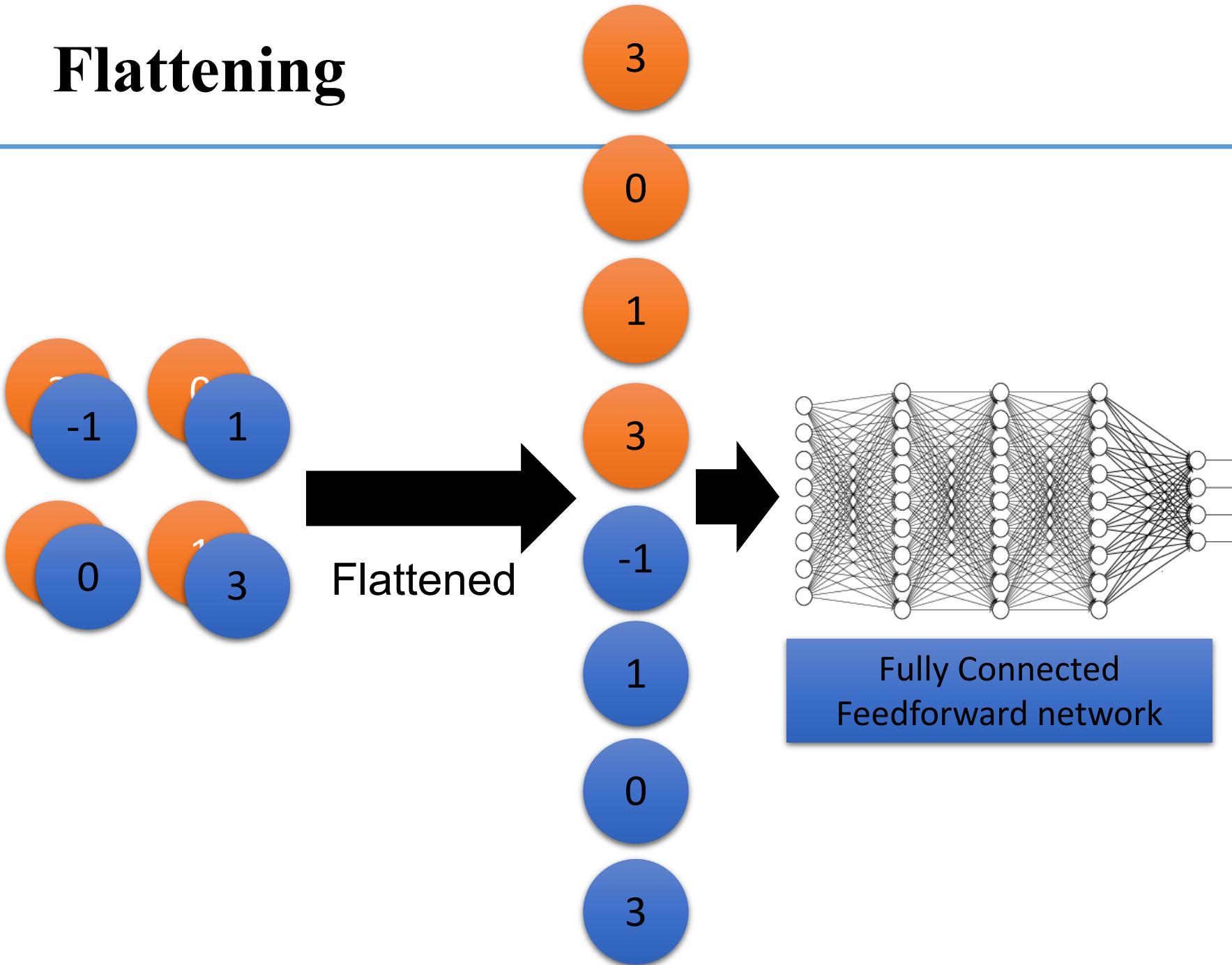


# The whole CNN





# Flattening





# 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)) )
```

$$\begin{matrix} 1 & -1 & 1 \\ -1 & \color{blue}{1} & -1 \\ -1 & -1 & 1 \\ -1 & 1 & -1 \end{matrix}$$

Input\_shape = ( 28 , 28 , 1)

28 x 28 pixels

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

$$\begin{matrix} 3 & -1 \\ -3 & 1 \end{matrix}$$



There are  
25 3x3  
filters.

1: black/white, 3: RGB

input



Convolution



Max Pooling



Convolution

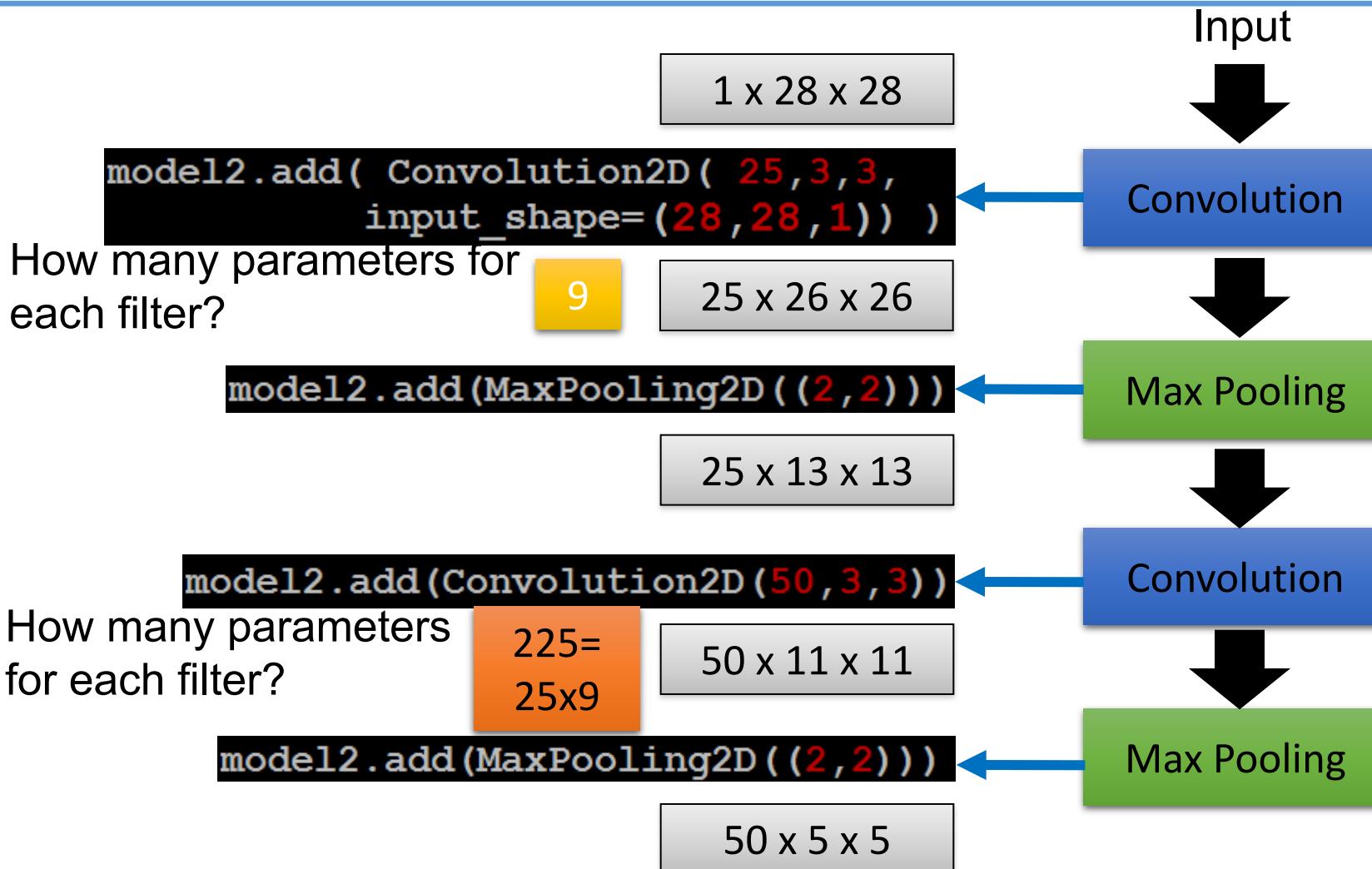


Max Pooling



# CNN in Keras

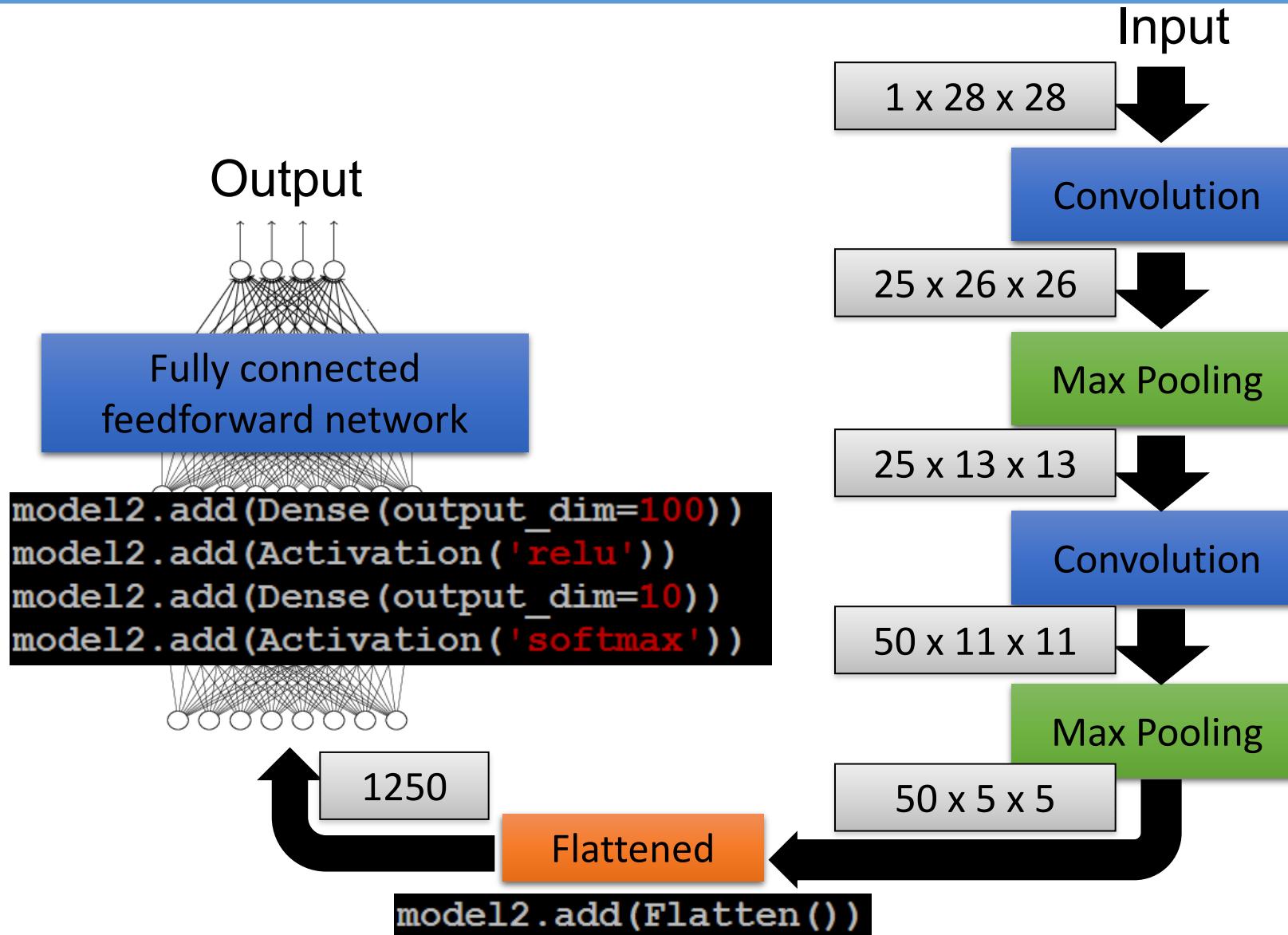
Only modified the *network structure* and *input format* (vector -> 3-D array)





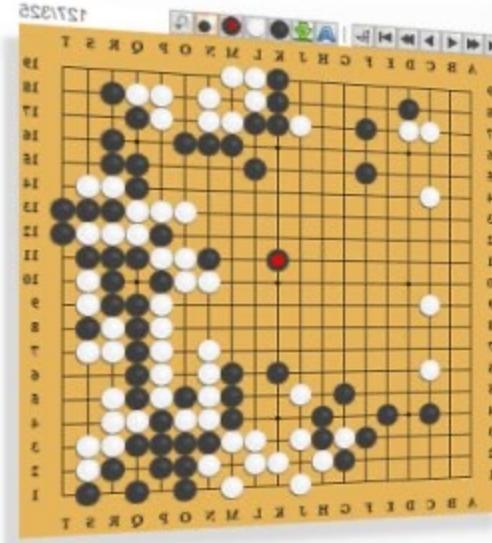
# CNN in Keras

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





# AlphaGo

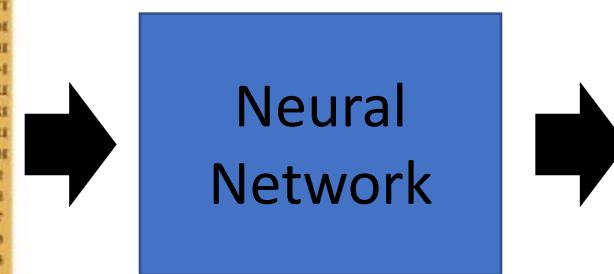


19 x 19 matrix

Black: 1

white: -1

none: 0



Next move  
(19 x 19  
positions)

Fully-connected feedforward network  
can be used

But CNN performs much better



# AlphaGo's policy network

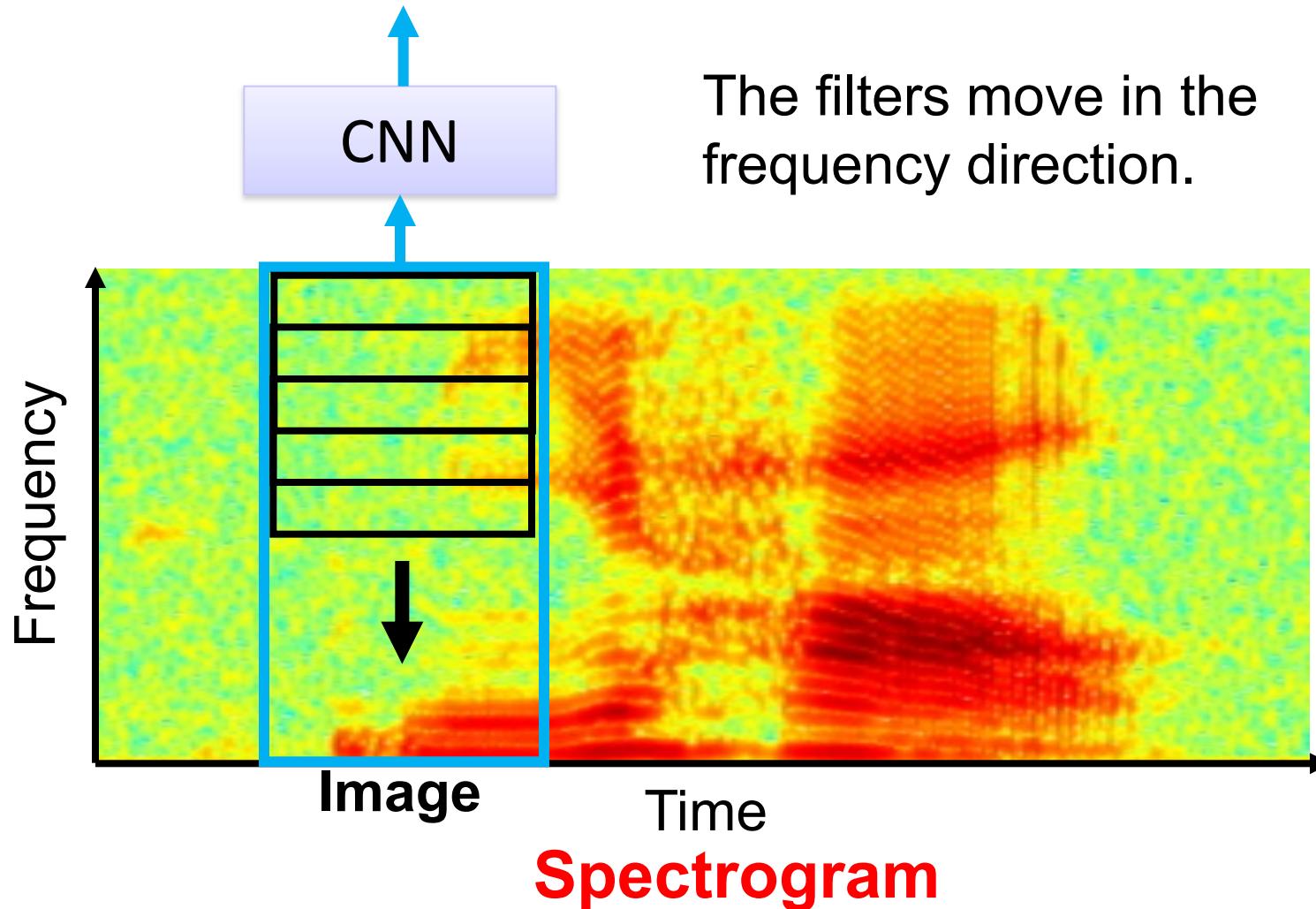
The following is quotation from their Nature article:

Note: AlphaGo does not use Max Pooling.

**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  $23 \times 23$  image, then convolves  $k$  filters of kernel size  $5 \times 5$  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  $21 \times 21$  image, then convolves  $k$  filters of kernel size  $3 \times 3$  with stride 1, again followed by a rectifier nonlinearity. The final layer convolves 1 filter of kernel size  $1 \times 1$  with stride 1, with a different bias for each position, and applies a softmax function. The match version of AlphaGo used  $k=192$  filters; Fig. 2b and Extended Data Table 3 additionally show the results of training with  $k=128, 256$  and  $384$  filters.



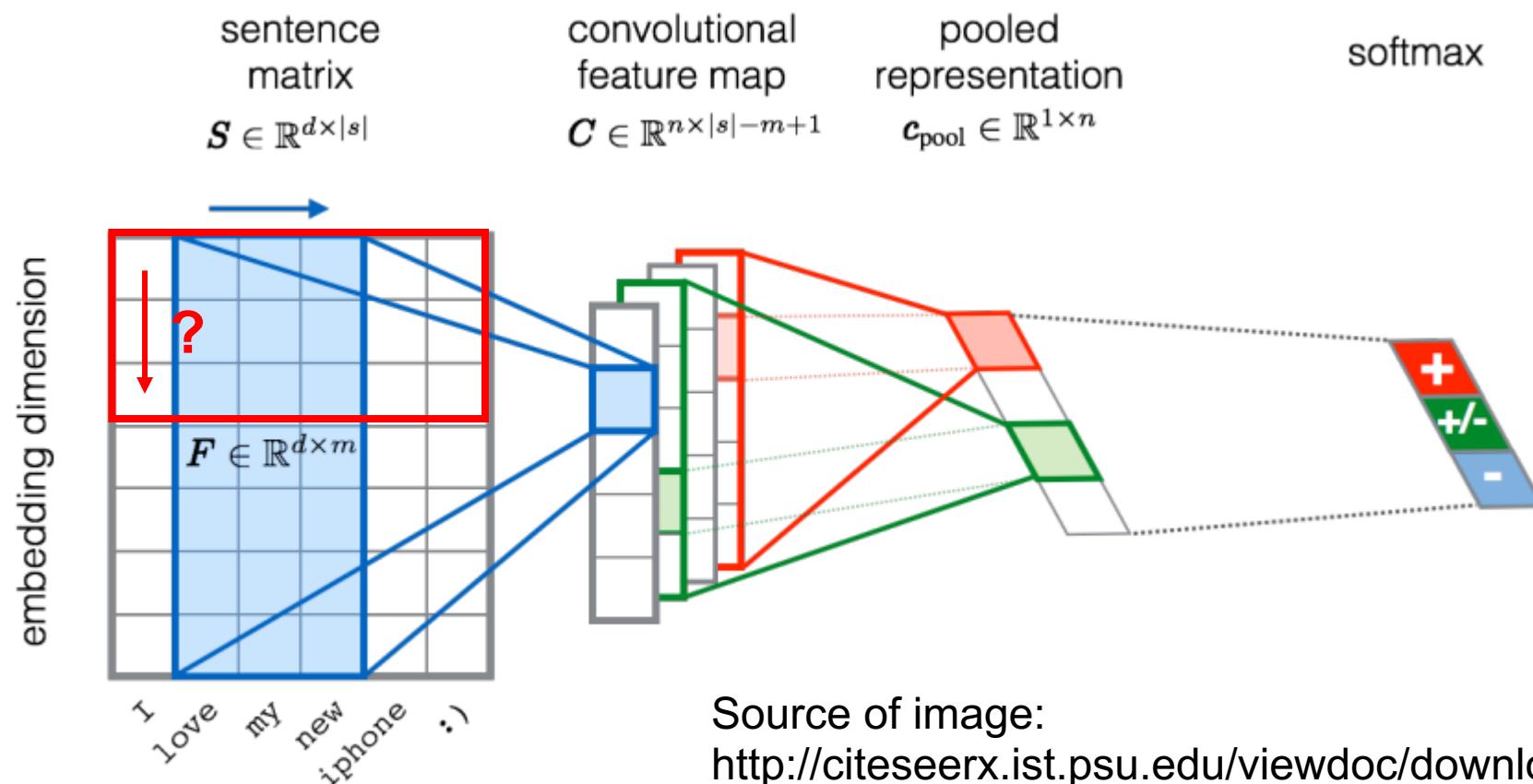
# CNN in speech recognition





# CNN in text classification

HCMUTE

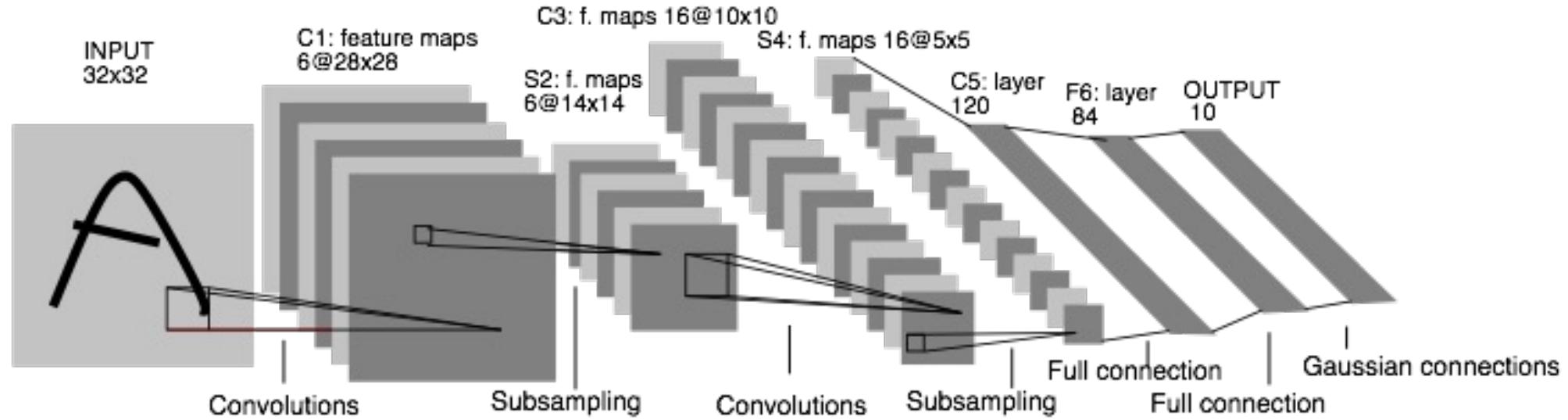


Source of image:  
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.6858&rep=rep1&type=pdf>



# Convolutional Neural Networks: 1998.

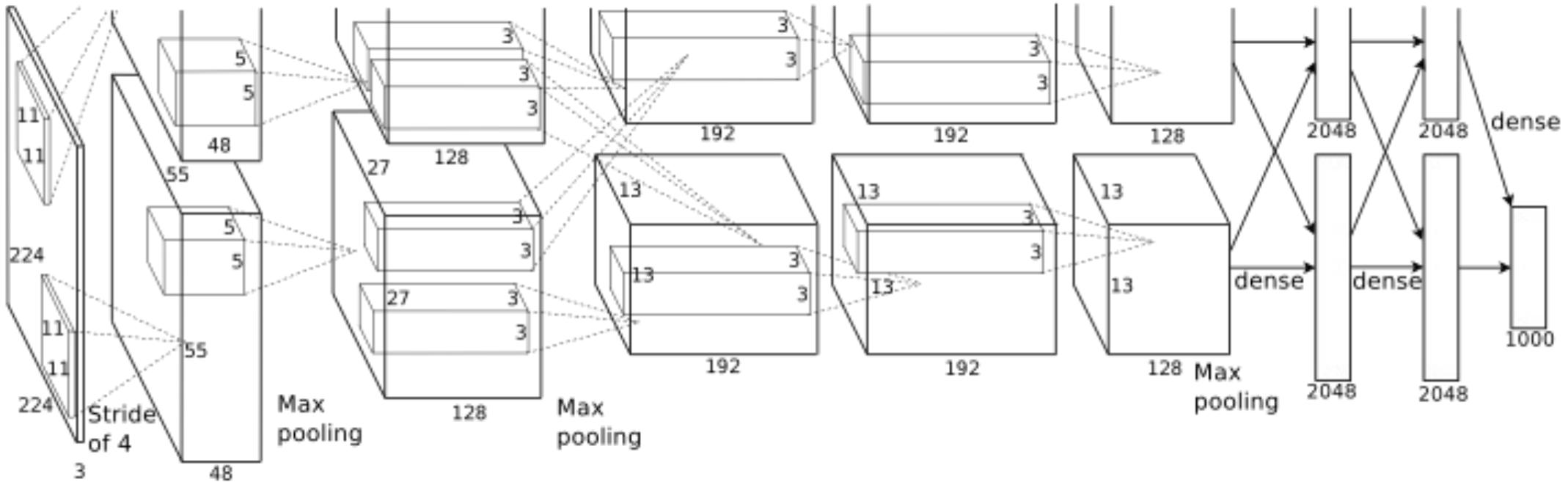
## Input 32\*32. CPU



LeNet: a layered model composed of convolution and subsampling operations followed by a holistic representation and ultimately a classifier for handwritten digits. [ LeNet ]

# Convolutional Neural Networks: 2012.

## Input 224\*224\*3. GPU.



AlexNet: a layered model composed of convolution, subsampling, and further operations followed by a holistic representation and all-in-all a landmark classifier on ILSVRC12. [ AlexNet ]

- + data
- + gpu
- + non-saturating nonlinearity
- + regularization



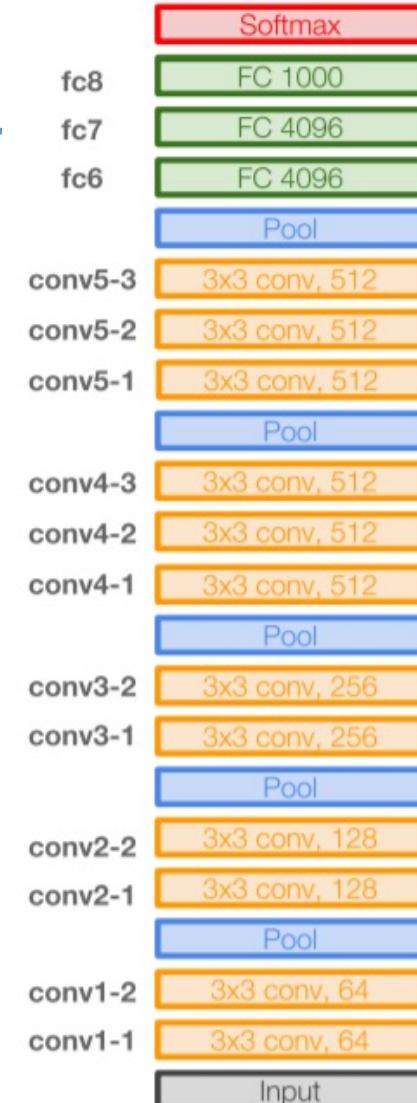
HCMUTE

# VGGNet

- 16 layers
- Only 3\*3 convolutions
- 138 million parameters



AlexNet

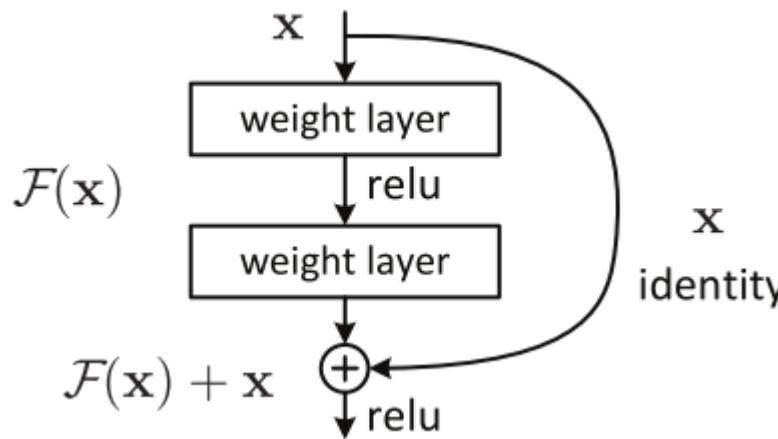


VGG16



HCMUTE

# ResNet



- 152 layers
- ResNet50

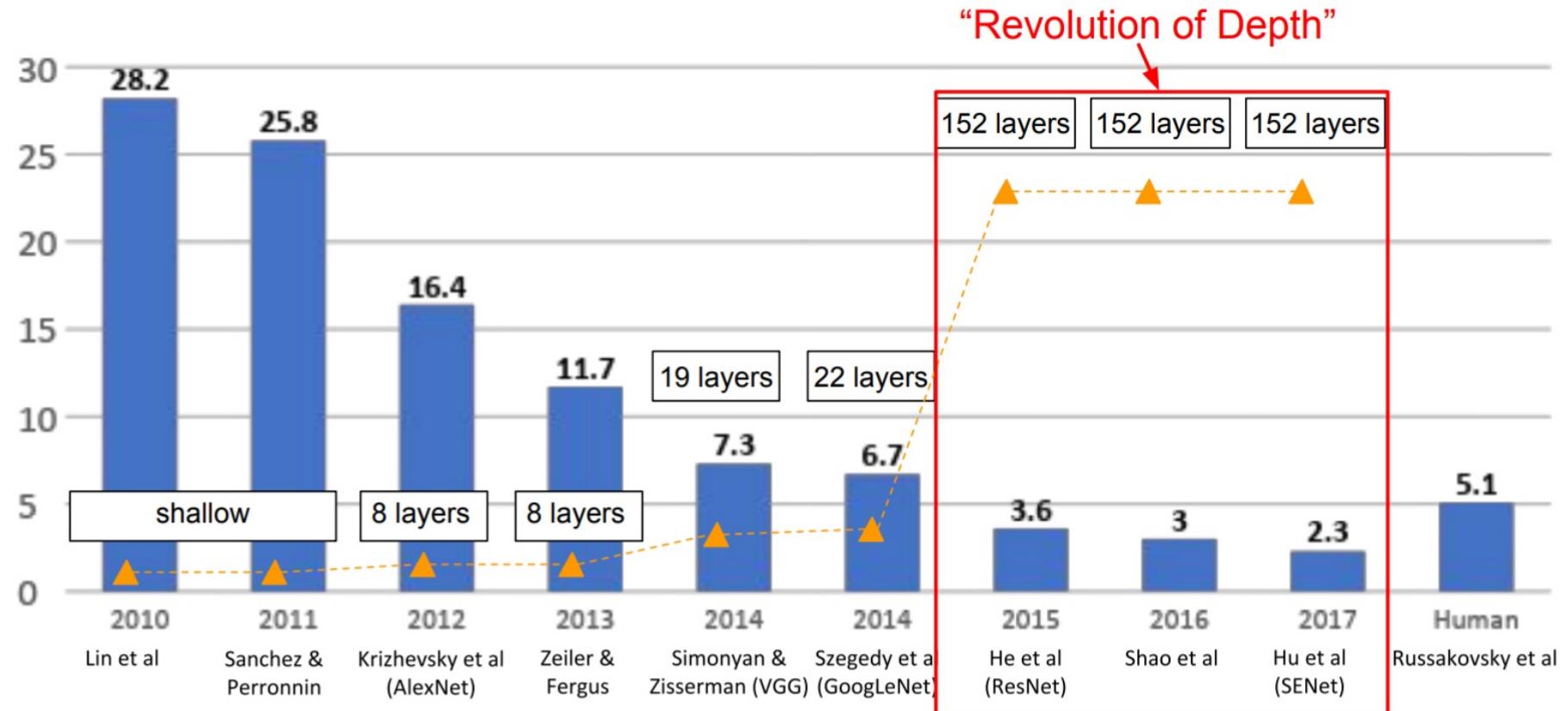




# The popular CNN

- LeNet, 1998
- AlexNet, 2012
- VGGNet, 2014
- ResNet, 2015

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

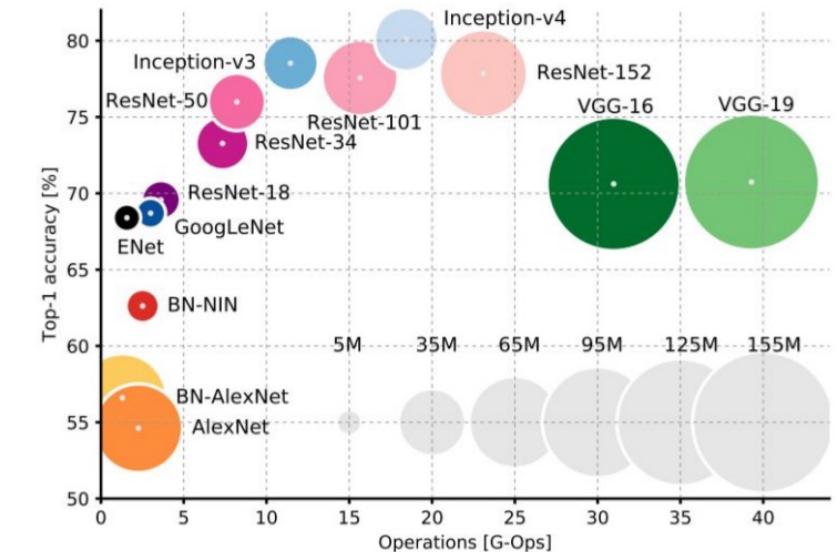
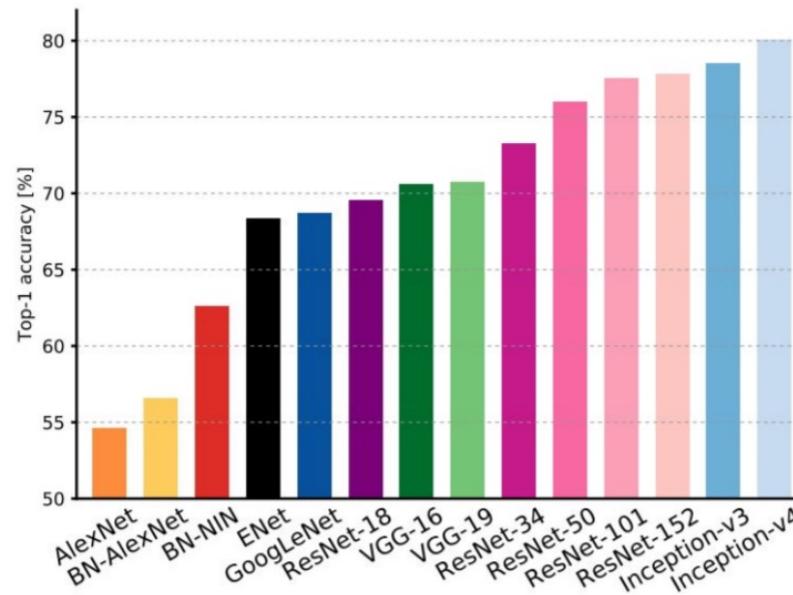




# Computational complexity

- The memory bottleneck
- GPU, a few GB

Comparing complexity...



An Analysis of Deep Neural Network Models for Practical Applications, 2017.



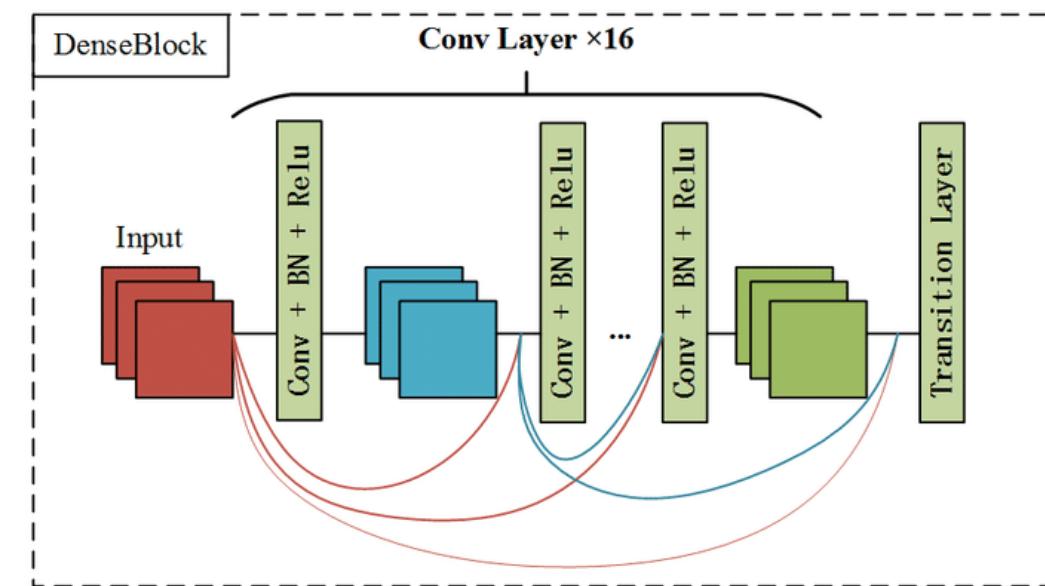
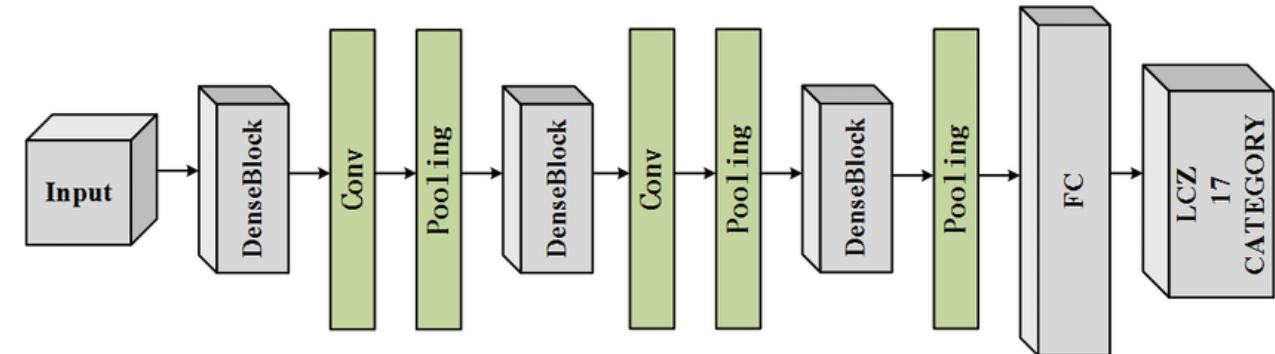
# CNN applications

- Transfer learning
- Fine-tuning the CNN
  - Keep some early layers
    - Early layers contain more generic features, edges, color blobs
    - Common to many visual tasks
  - Fine-tune the later layers
    - More specific to the details of the class
- CNN as feature extractor
  - Remove the last fully connected layer
  - A kind of descriptor or CNN codes for the image
  - AlexNet gives a 4096 Dim descriptor



# CNN classification/recognition nets

- CNN layers and fully-connected classification layers
- From ResNet to DenseNet
  - Densely connected
  - Feature concatenation





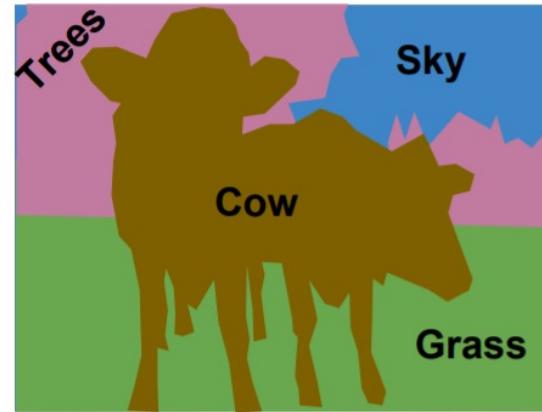
# Fully convolutional nets: semantic segmentation

- Classification/recognition nets produce ‘non-spatial’ outputs
  - the last fully connected layer has the fixed dimension of classes, throws away spatial coordinates
- Fully convolutional nets output maps as well



# Semantic segmentation

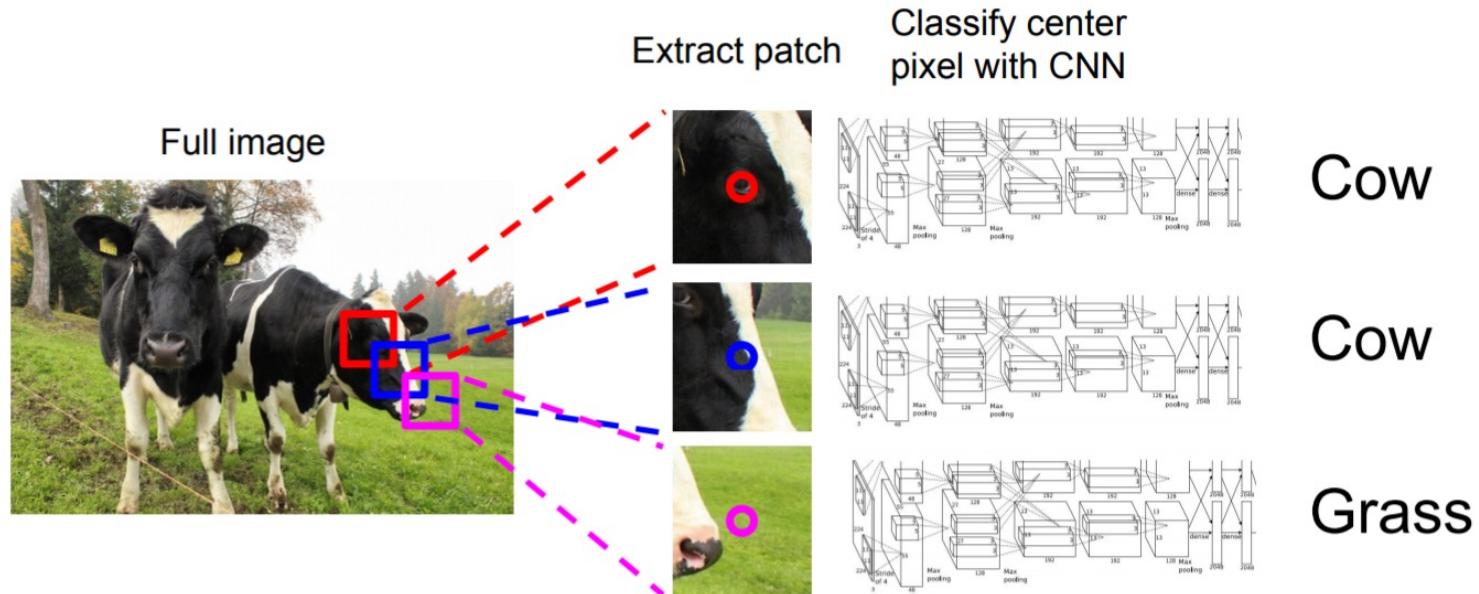
[This image is CC0 public domain](#)





HCMUTE

# Using sliding windows for semantic segmentation



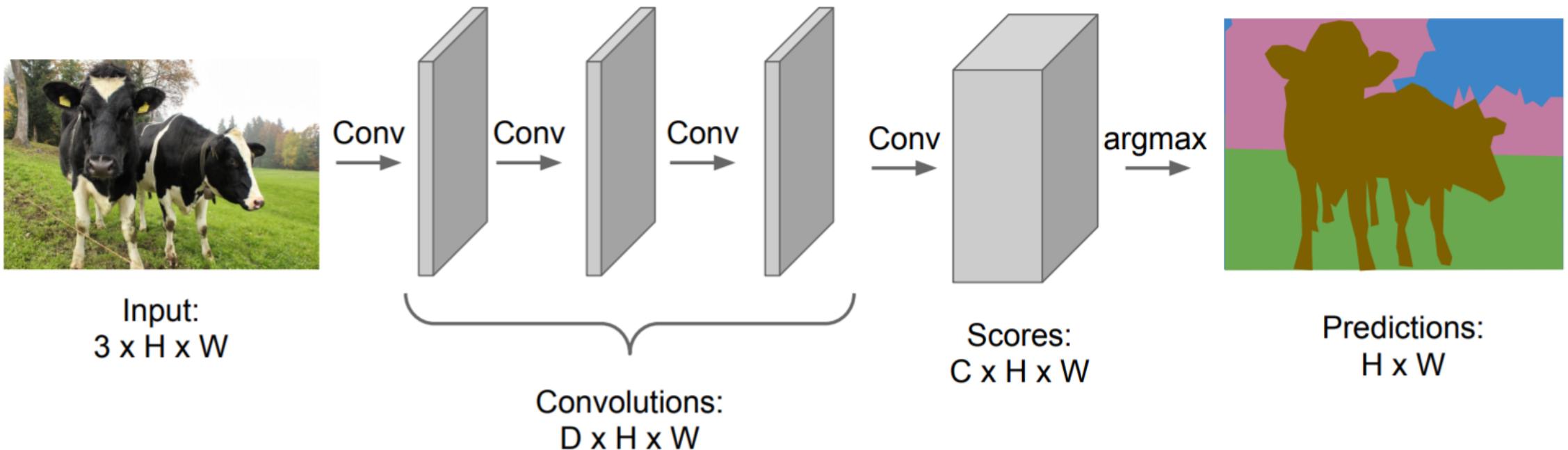
Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013

Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014



# Fully convolutional

Design a network as a bunch of convolutional layers  
to make predictions for pixels all at once!





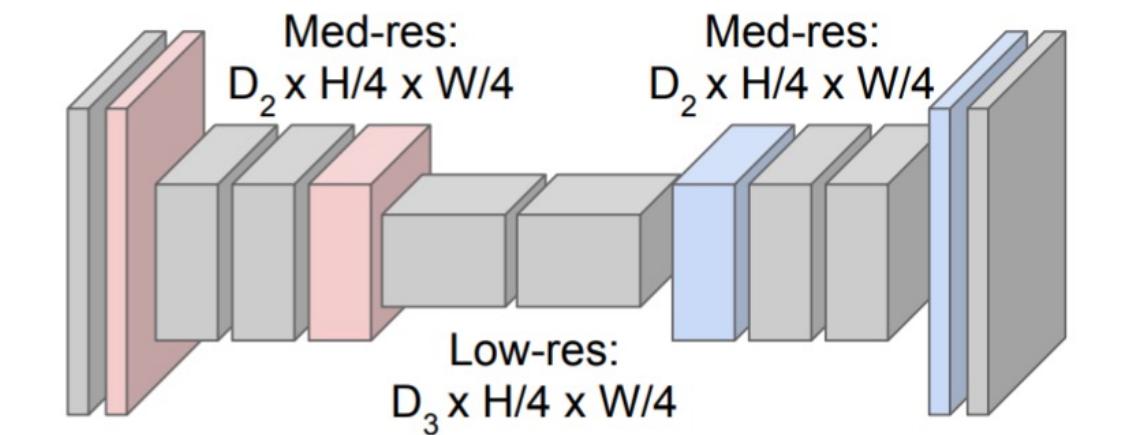
# Fully convolutional

**Downsampling:**  
Pooling, strided  
convolution



Input:  
 $3 \times H \times W$

High-res:  
 $D_1 \times H/2 \times W/2$



Design network as a bunch of convolutional layers, with  
**downsampling** and **upsampling** inside the network!

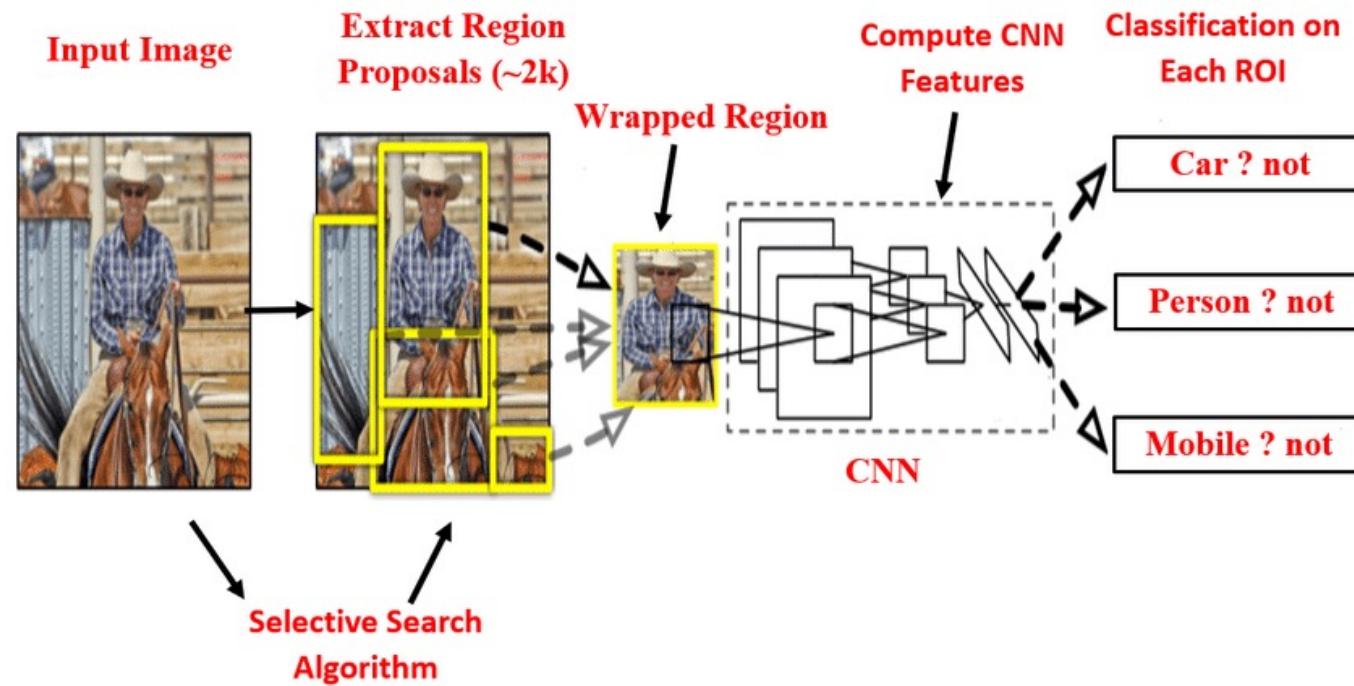
**Upsampling:**  
Unpooling or strided  
transpose convolution



Predictions:  
 $H \times W$

# Detection and segmentation nets: The Mask Region-based CNN (R-CNN):

- Class-independent region (bounding box) proposals
  - From selective search to region proposal net with objectness
- Use CNN to class each region
- Regression on the bounding box or contour segmentation

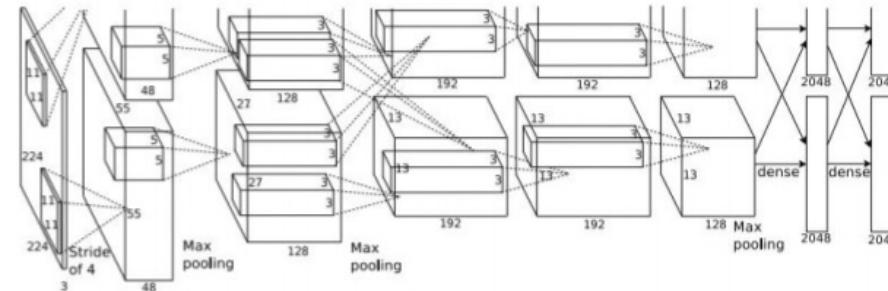
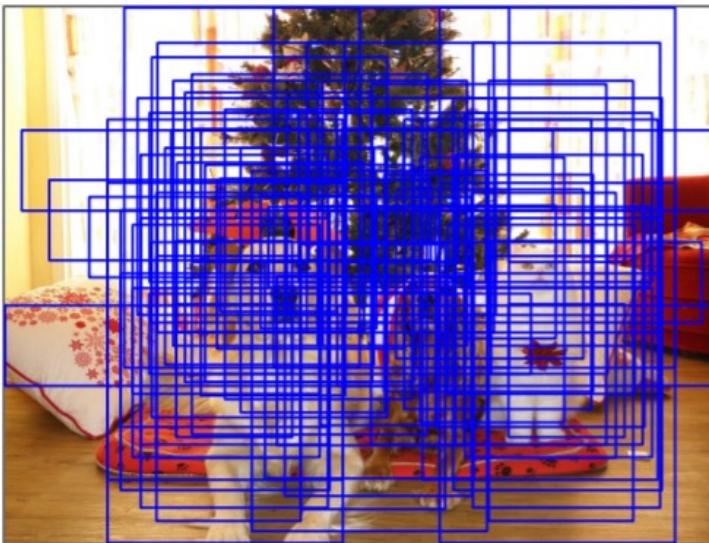




HCMUTE

# Using sliding windows for object detection as classification

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

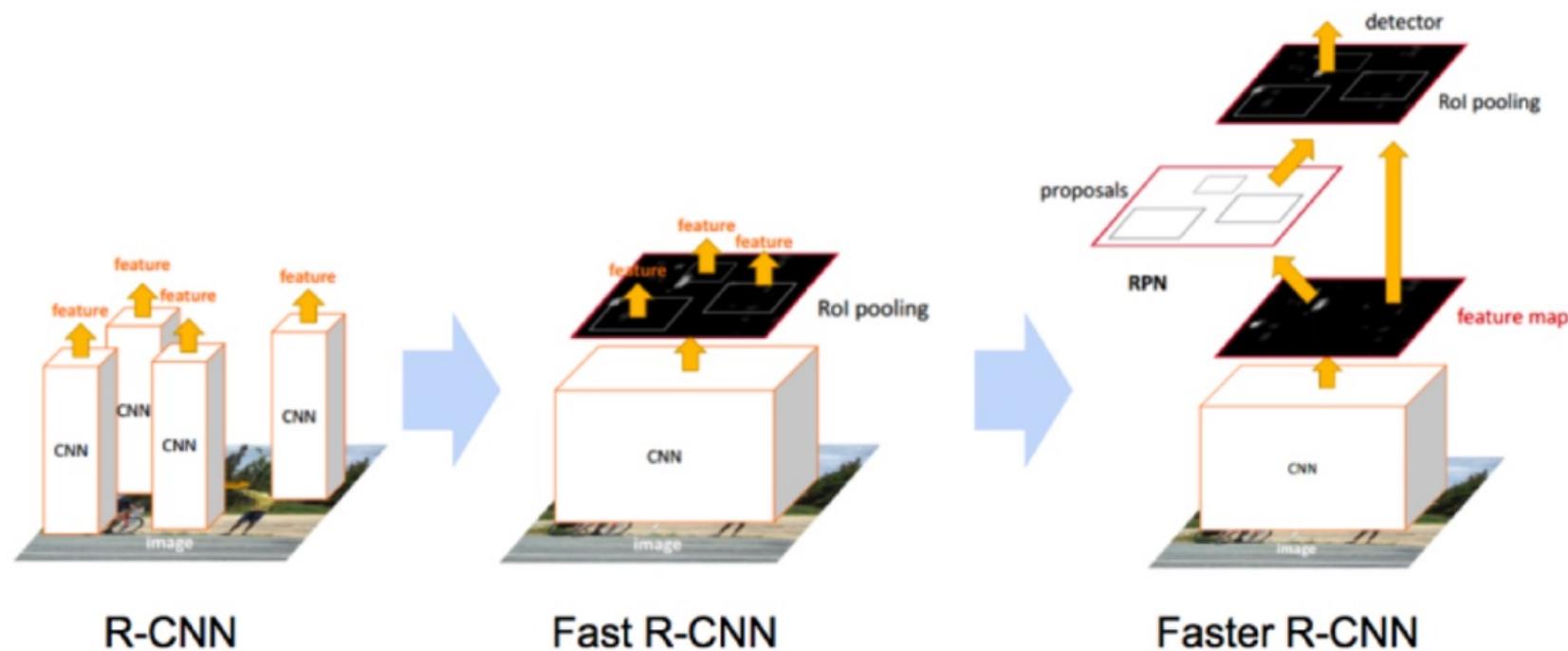


Dog? NO  
Cat? YES  
Background? NO

Problem: Need to apply CNN to huge number of locations, scales, and aspect ratios, very computationally expensive!

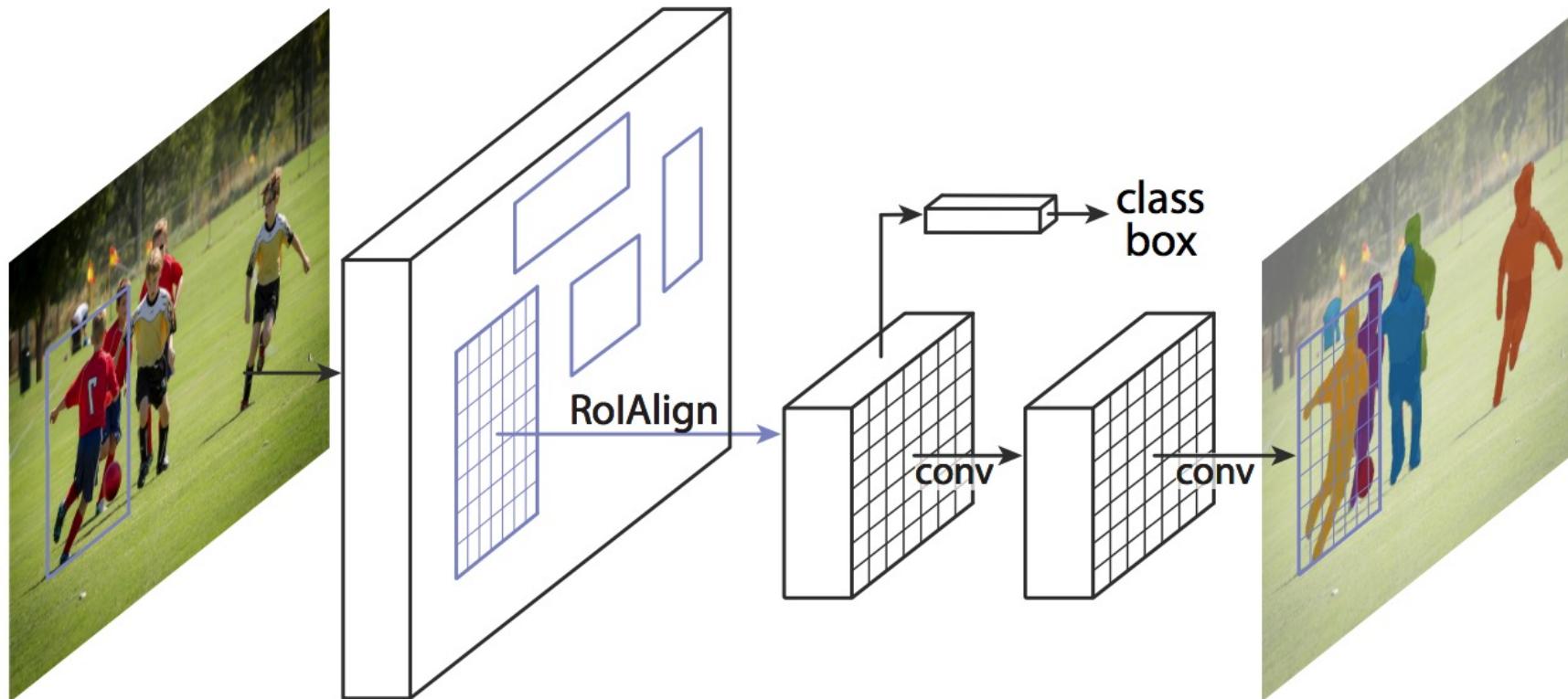
# Detection and segmentation nets: The Mask Region-based CNN (R-CNN):

- Class-independent region (bounding box) proposals
  - From selective search to region proposal net with objectness
- Use CNN to class each region
- Regression on the bounding box or contour segmentation



# Detection and segmentation nets: The Mask Region-based CNN (R-CNN):

- Mask R-CNN: end-to-end
  - Use CNN to make proposals on object/non-object in parallel

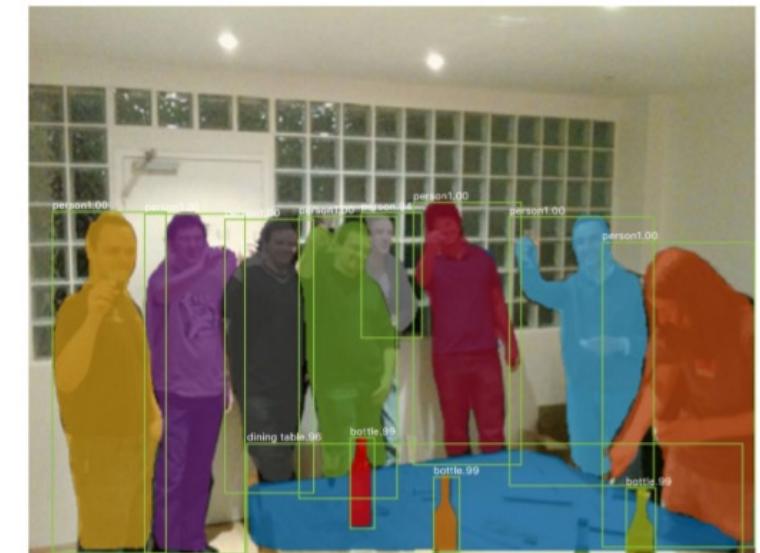
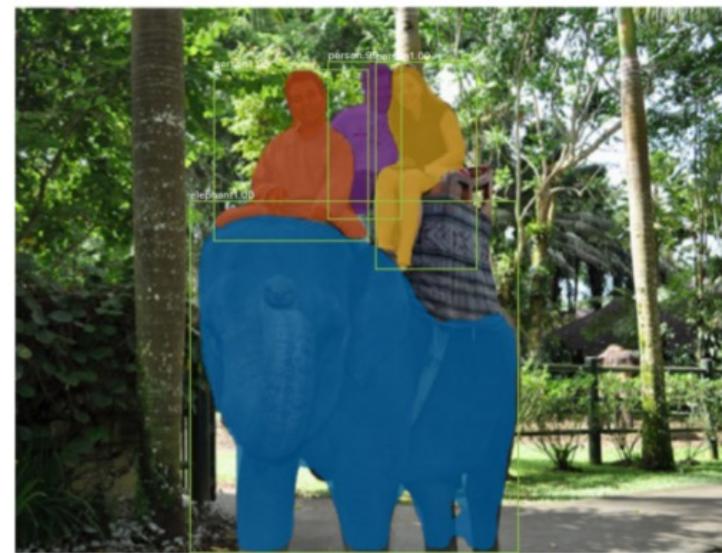




HCMUTE

---

# Excellent results



He et al, "Mask R-CNN", arXiv 2017

Figures copyright Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 2017.

Reproduced with permission.



# Bài tập 11

Trong dữ liệu ex7data.mat chưa dữ liệu lưu dưới dạng dict gồm:

X: 5000x400 là 5000 ảnh nhị phân chữ số viết tay có kích thước 20x20

y: 5000x1 là nhãn của các ảnh tương ứng

Các bạn làm các công việc sau:

- Reshape X về kích thước 5000x1x20x20
- Chia dữ liệu thành 70% train, 30% test (train\_test\_split) đảm bảo tính ngẫu nhiên và đồng đều về nhãn.
- Chia dữ liệu train thành 90% train, 10% val (train\_test\_split) đảm bảo tính ngẫu nhiên và đồng đều về nhãn.
- Xây dựng một mạng CNN cho phù hợp với dữ liệu trên để đạt được hiệu suất tốt nhất
- Show đường cong loss trong quá trình học
- Show độ chính xác trên tập test