Introduction to Machine Learning

Lecture 15: Convolutional Neural Network (CNN)

Nov 26, 2024

Jie Wang

Machine Intelligence Research and Applications Lab

Department of Electronic Engineering and Information Science (EEIS)

http://staff.ustc.edu.cn/~jwangx/

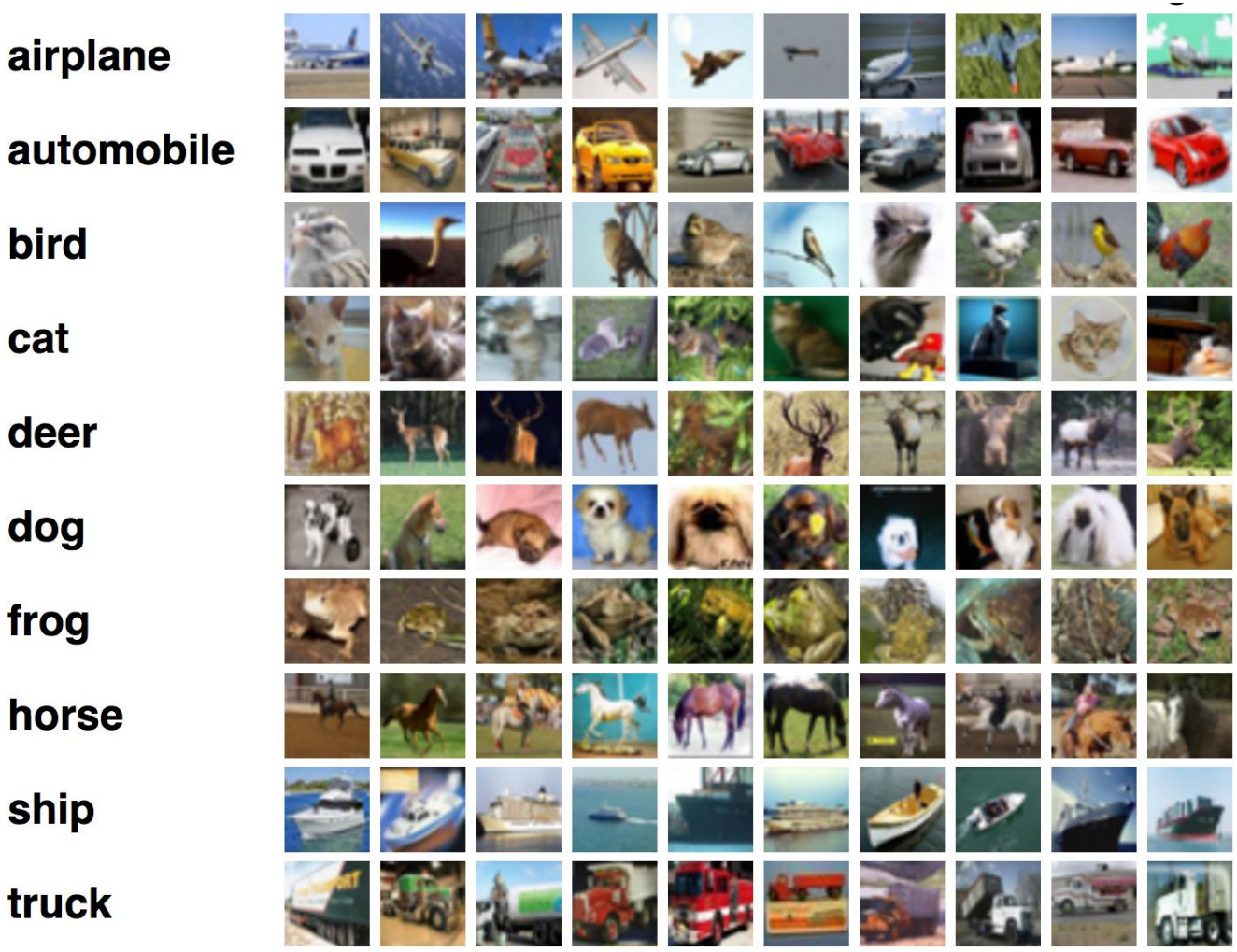
jiewangx@ustc.edu.cn

Contents

- Introduction
- Network Layers of CNN
- Learning a CNN
- Examples of CNN Architectures
- More Applications

Introduction

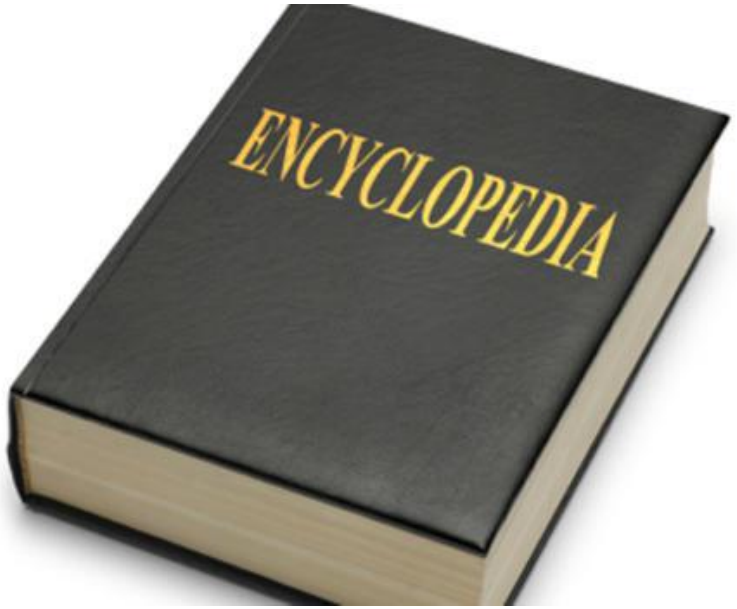
Image Classification

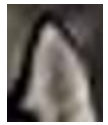
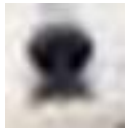
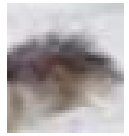


- Identifying objects from various scenes is a easy task for human
- However, it is difficult for human to describe (precisely) how he/she can do it

https://medium.com/@tifa2up/image-classification-using-deep-neural-networks-a-beginner-friendly-approach-using-tensorflow-94b0a090ccd4

Features

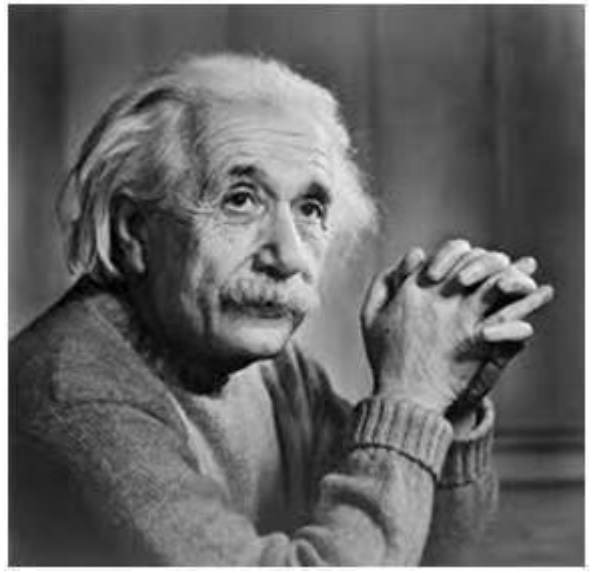


≈  +  +  +  +  + ... + Relative location

Features

- How to extract discriminative features?
  - Hand-crafted features: HOG, SIFT, SURF etc
  - Learned features: hidden states of DNN

Hand-crafted Features – Sobel Operator



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

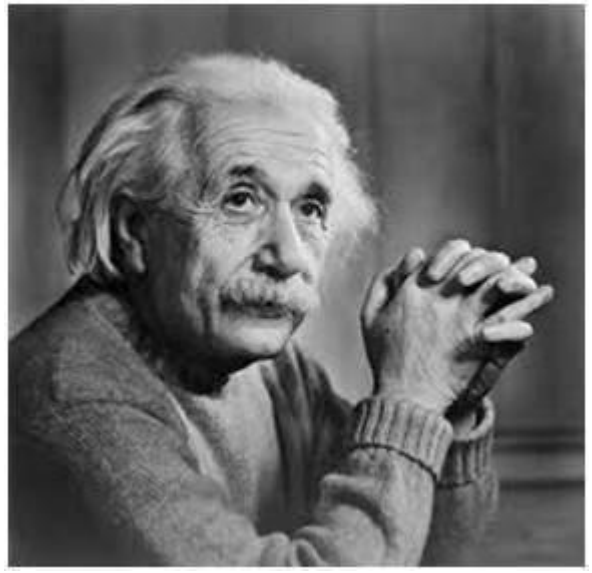
Vertical Mask

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

Horizontal Mask

https://www.tutorialspoint.com/dip/sobel_operator.htm

Hand-crafted Features – Robinson Compass Mask



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

South West Direction Mask

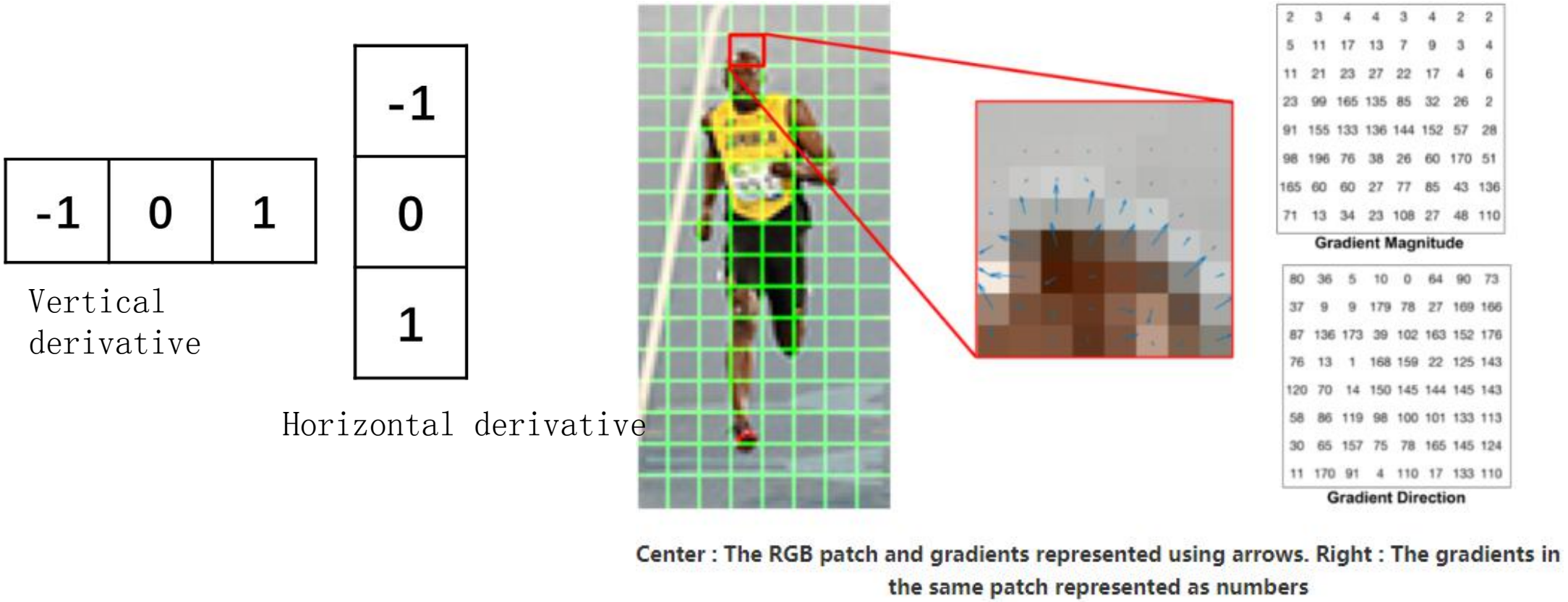
-2	-1	0
-1	0	1
0	1	2

North East Direction Mask

https://www.tutorialspoint.com/dip/robinson_compass_mask.

Hand-crafted Features – HOG

- Histograms of Oriented Gradients



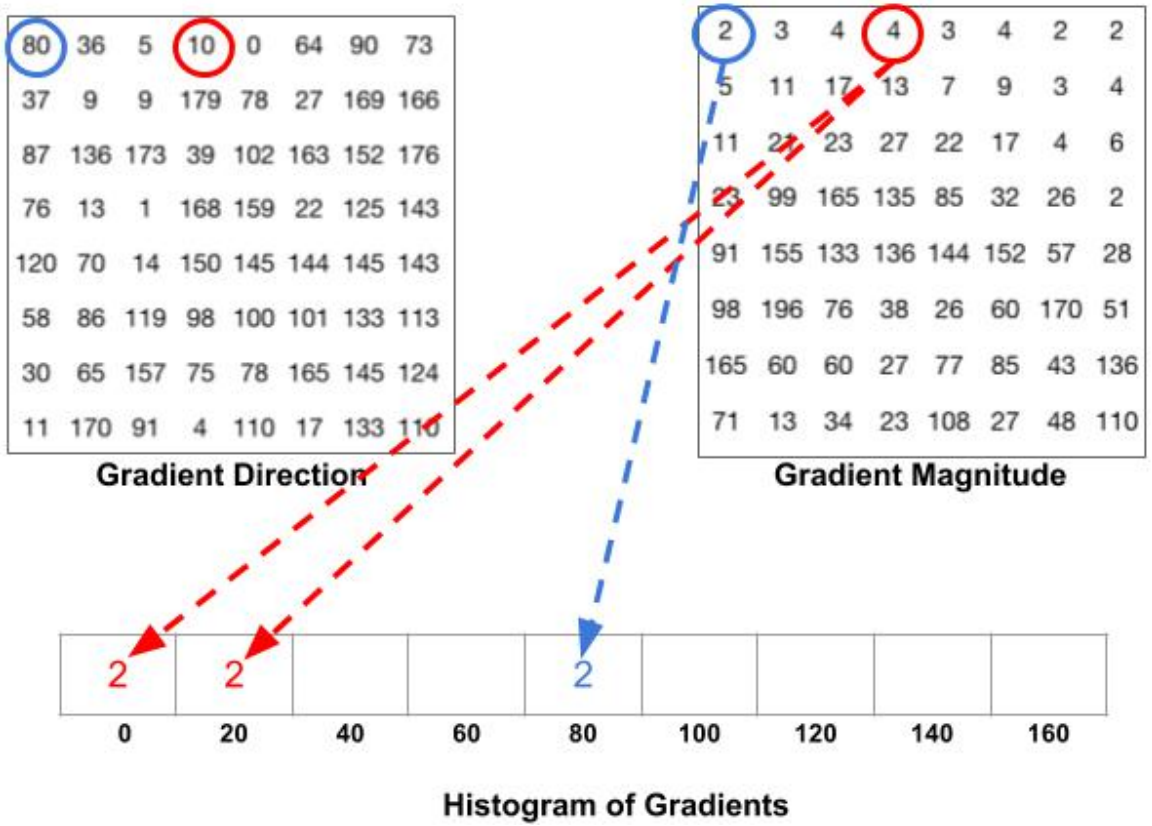
Center : The RGB patch and gradients represented using arrows. Right : The gradients in the same patch represented as numbers

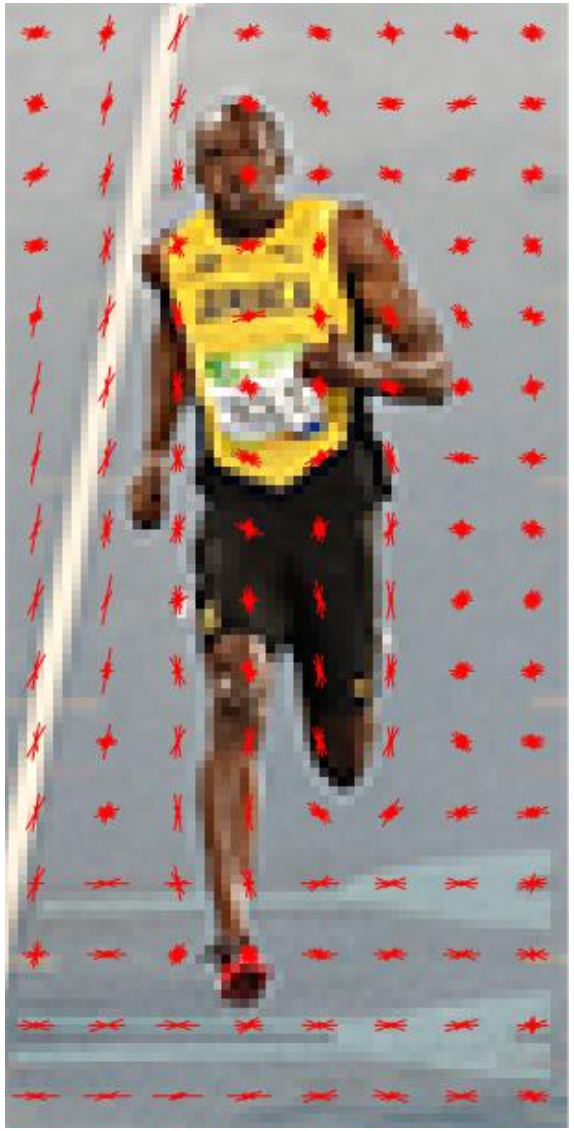
https://www.learnopencv.com/histogram-of-oriented-gradients/

Dalal, Navneet, Triggs, et al. Histograms of Oriented Gradients for Human Detection.

Hand-crafted Features – HOG

- Histograms of Oriented Gradients



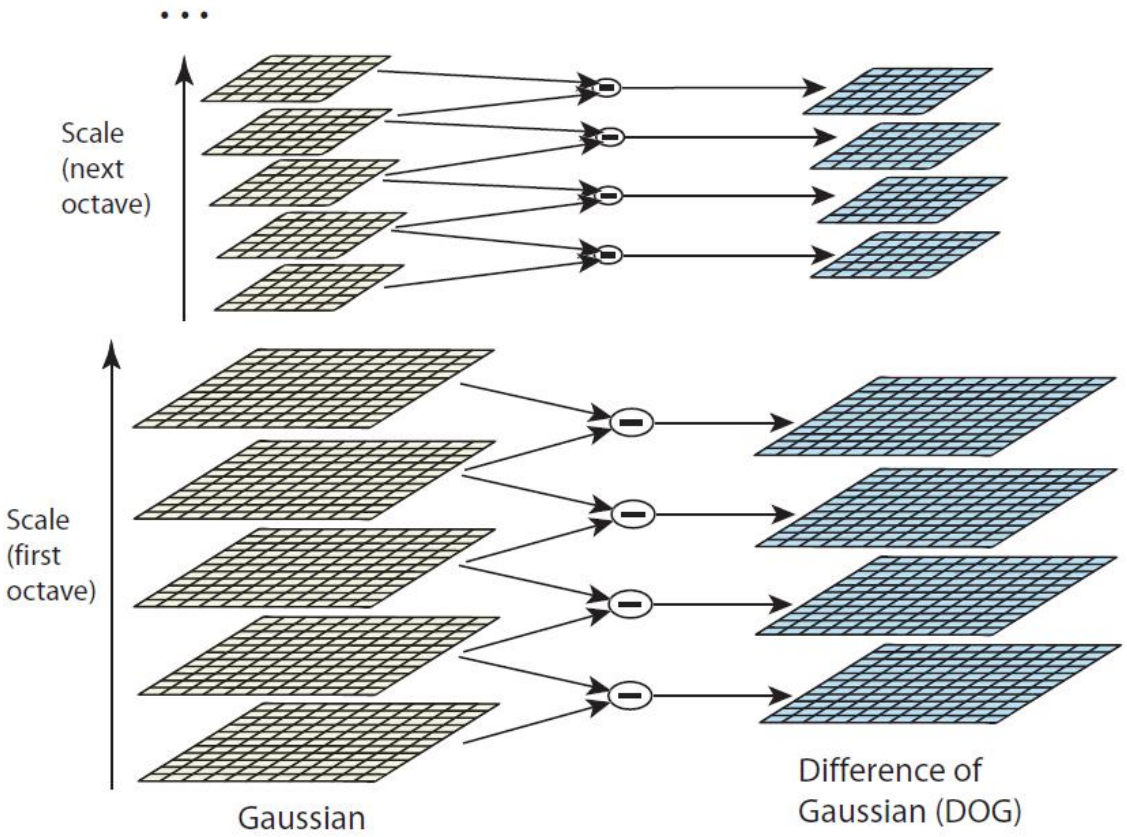


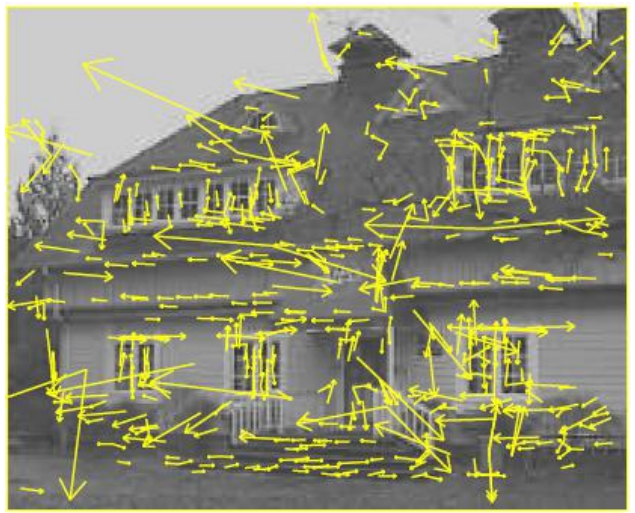
https://www.learnopencv.com/histogram-of-oriented-gradients/

Dalal, Navneet, Triggs, et al. Histograms of Oriented Gradients for Human Detection. CVPR, 2005

Hand-crafted Features – SIFT

- Scale-Invariant Feature Transform

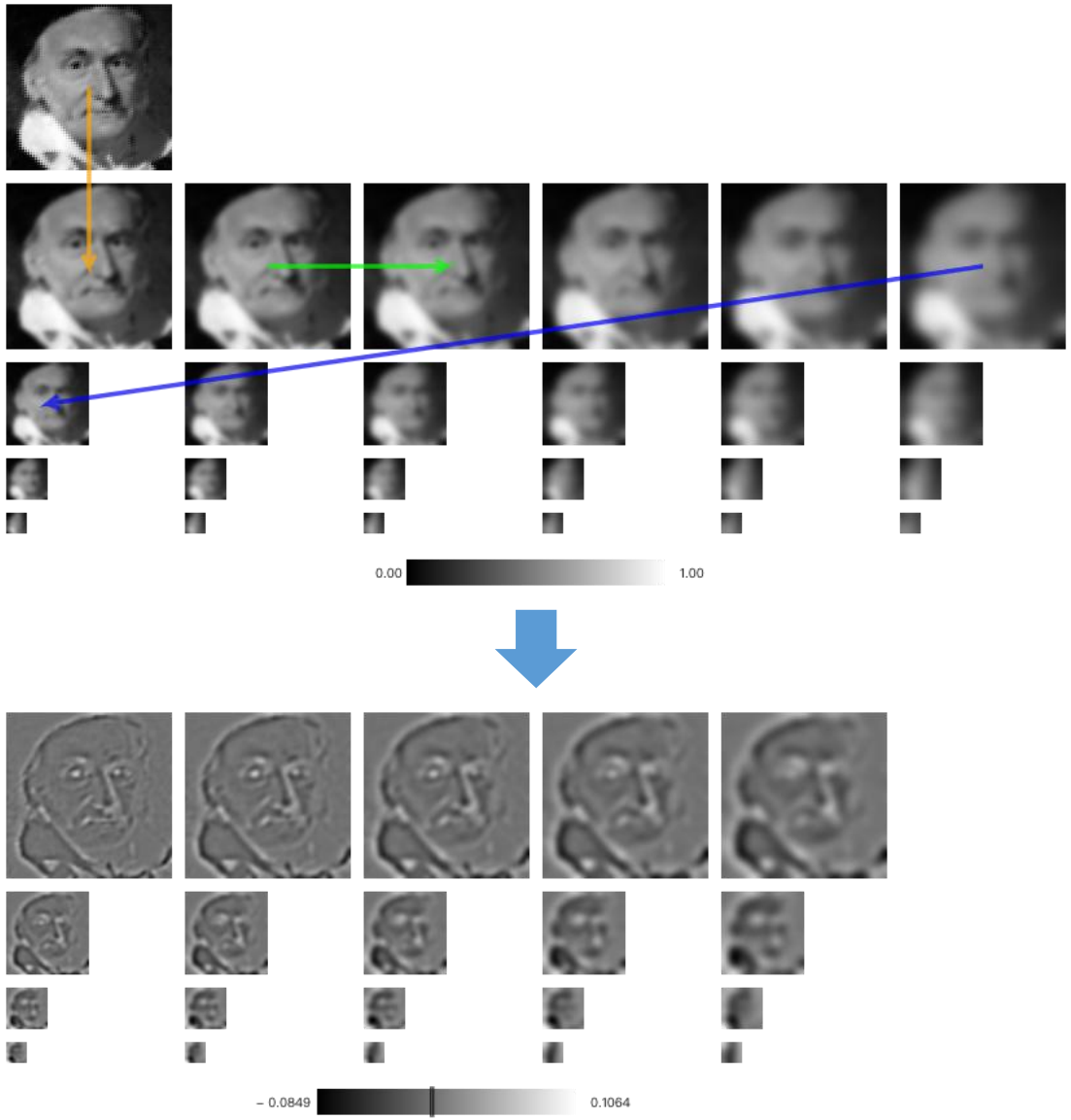


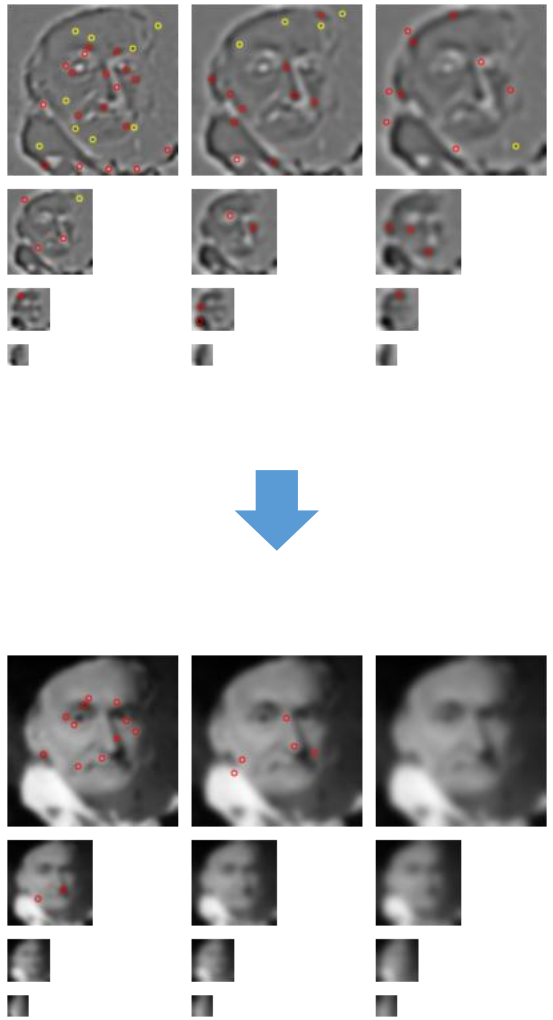


Lowe D G. Distinctive Image Features from Scale-Invariant Keypoints[C]// International Journal of Computer Vision. 2004:91-110.

http://weitz.de/sift/

Hand-crafted Features – SIFT

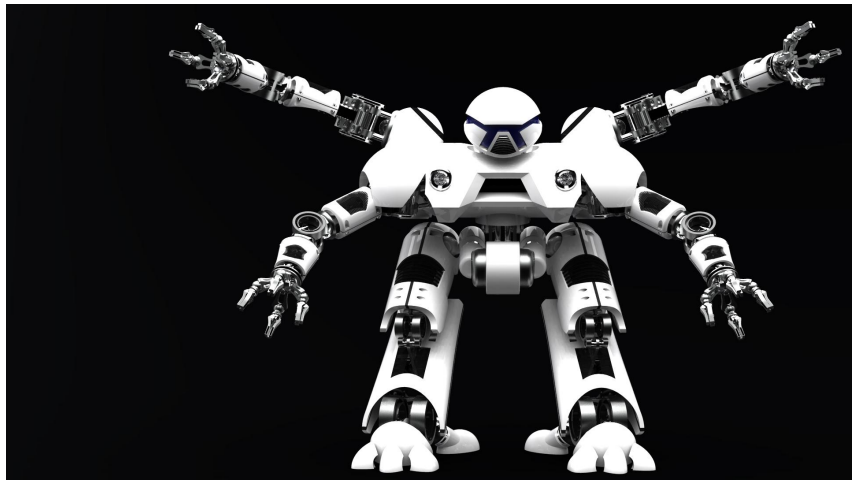




http://weitz.de/sift/

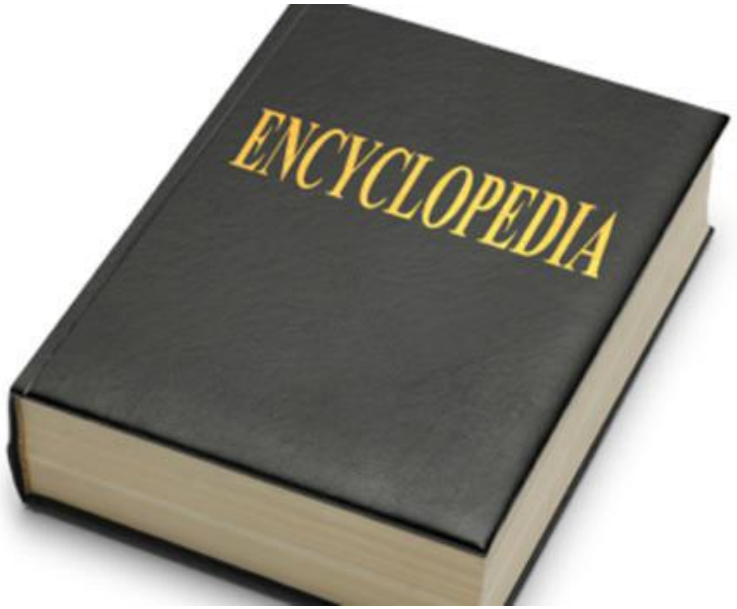
Hand-crafted Features vs Learned Features

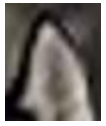
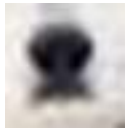
- Hand-crafted Features
  - Challenging to design
  - Require expert domain knowledge
  - Not flexible



- Learned Features
  - Automatically learned by machines
  - No need of expert domain knowledge
  - Data-driven

Learned Features



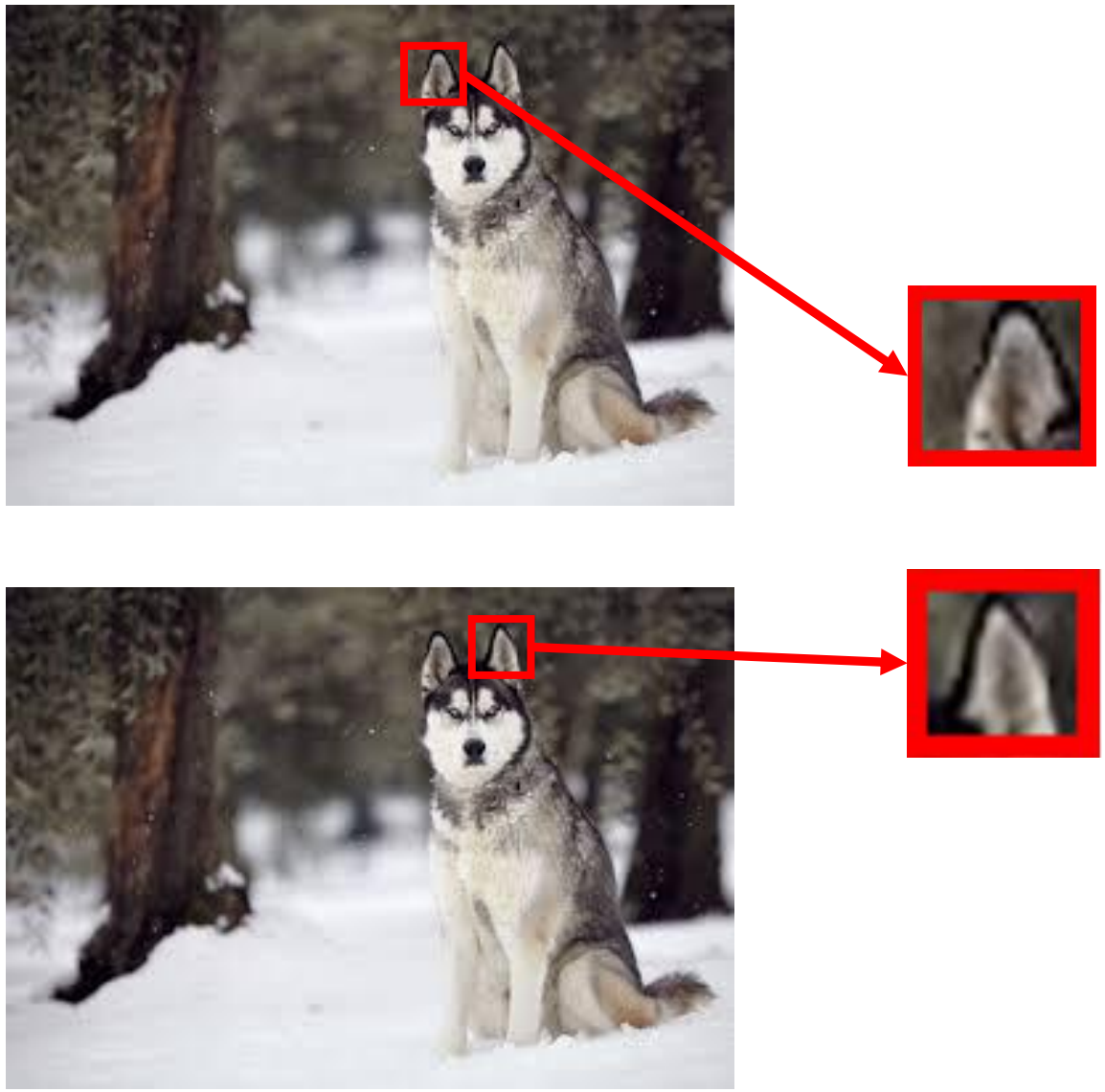
≈  +  +  +  +  + ... + Relative location

- We can design a new neural network
  - It can detect different patterns
  - It can detect similar patterns in different regions
  - It can roughly preserve the spatial information

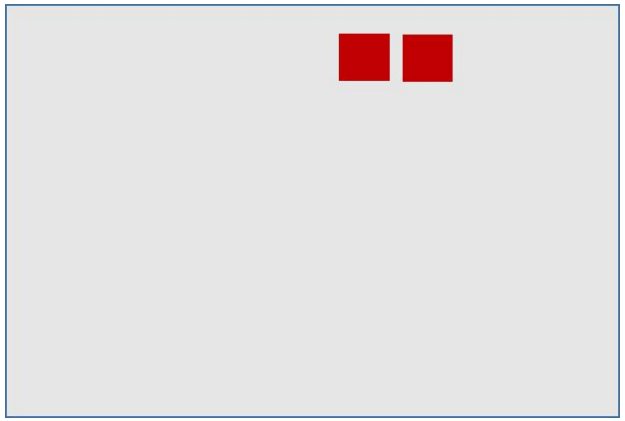
Detect different patterns

- Most patterns are much smaller in view of the whole image
  - A filter does not have to see the whole image to discover the pattern
    - One filter connects to small region with less parameters at a time

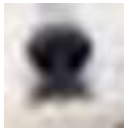
Detect the similar patterns in different regions

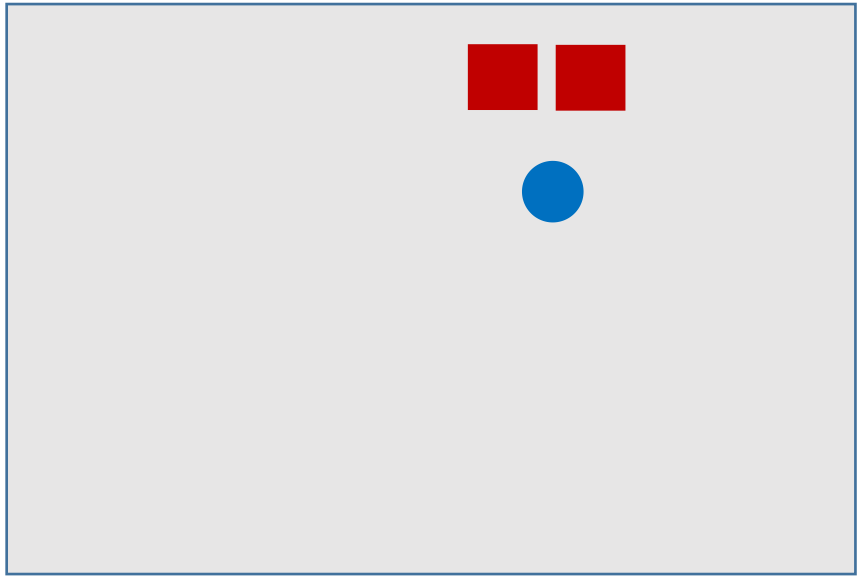


- The similar patterns appear in different regions
  - We can use one filter to detect similar patterns
    - One filter uses the same set of parameters for different regions



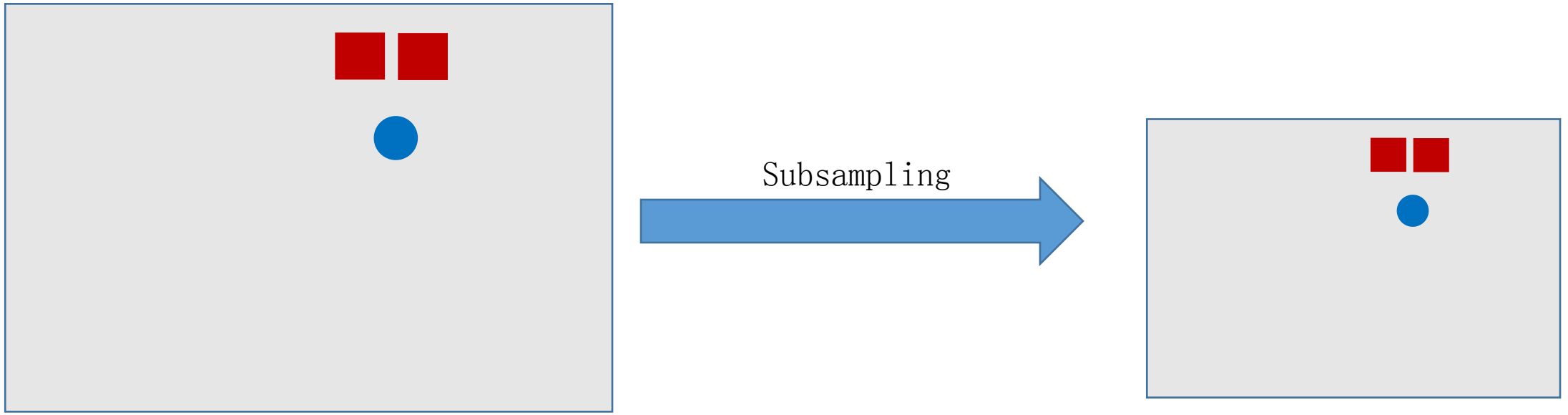
The spatial information





- For many vision tasks, the detected spatial information can be redundant

Roughly preserve the spatial information



- The relative location of patterns will be preserved after subsampling
  - Using subsampling to lessen the parameters

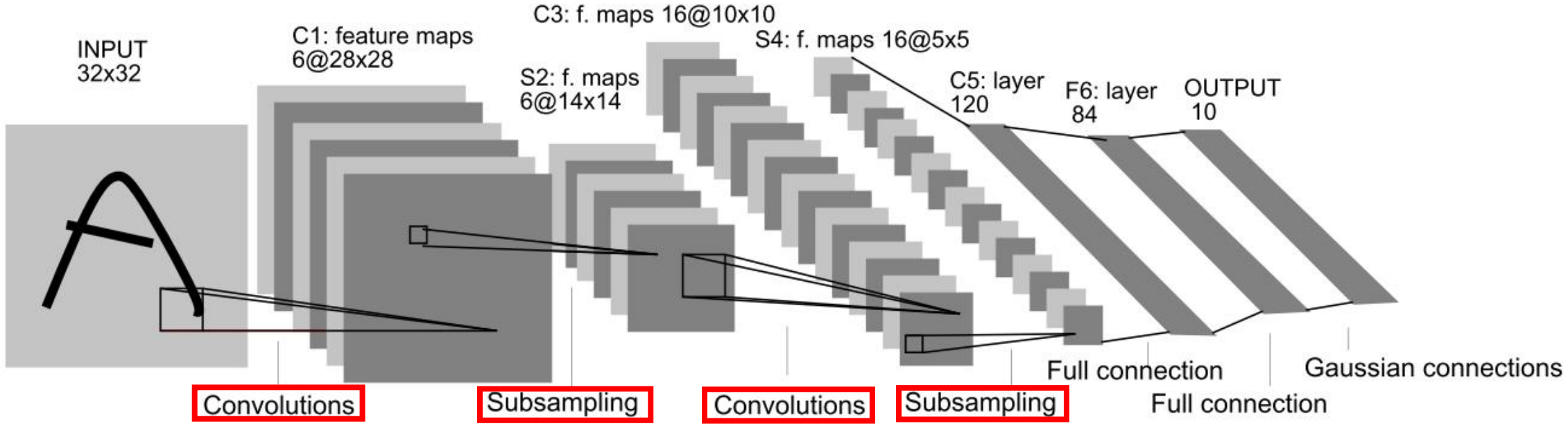
Convolutional Neural Networks (CNN)

- Try to construct a new neural network
  - One filter connects to small region with less parameters at a time
  - One filter uses the same set of parameters for different regions

  } Convolution

  - Using subsampling to lessen the parameters

  } Pooling (Subsampling)

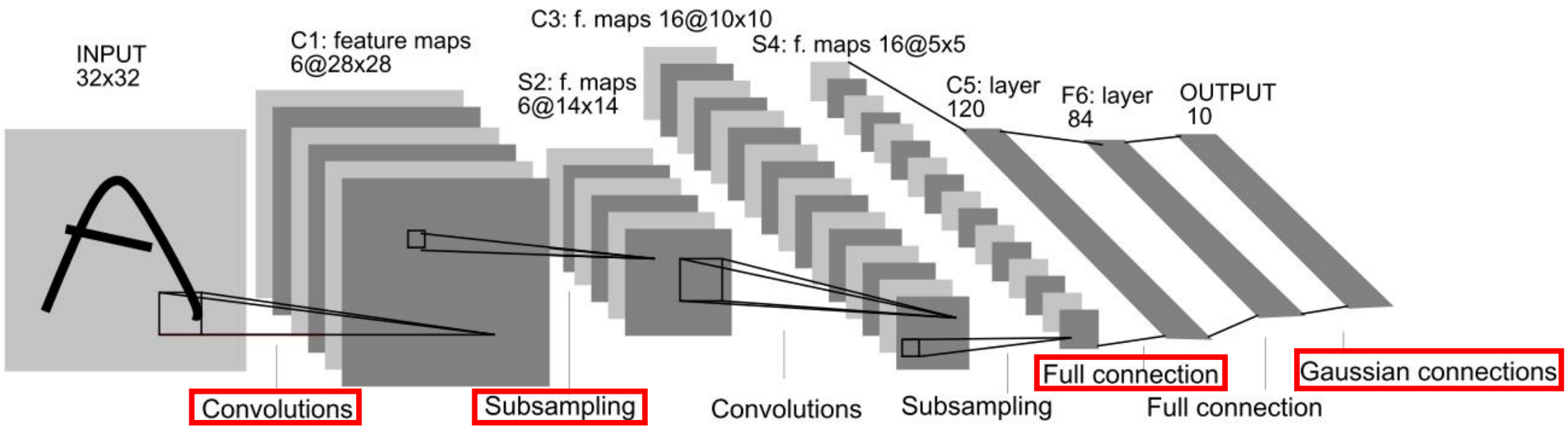


Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE, vol. 86, no. 11, pp. 2278 - 2324, Feb. 1998

Network Layers of CNN

How does CNN work

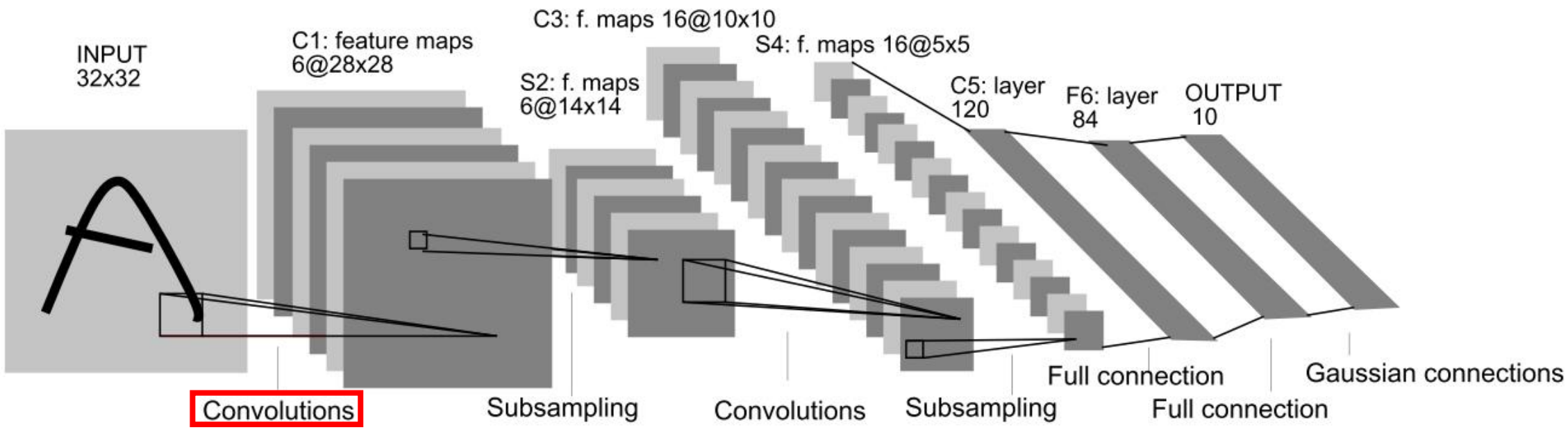
- A simple but important CNN – LeNet 5



- Given an input image, how to predict its label?

How does CNN work

- A simple but important CNN – LeNet 5



CNN – Convolution Layer

- **Input: Image**

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

6 × 6 image

- **Filters**

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

Filter 1

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

Filter 2

Those are parameters to be learned!

CNN – Convolution Layer

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

6×6 image

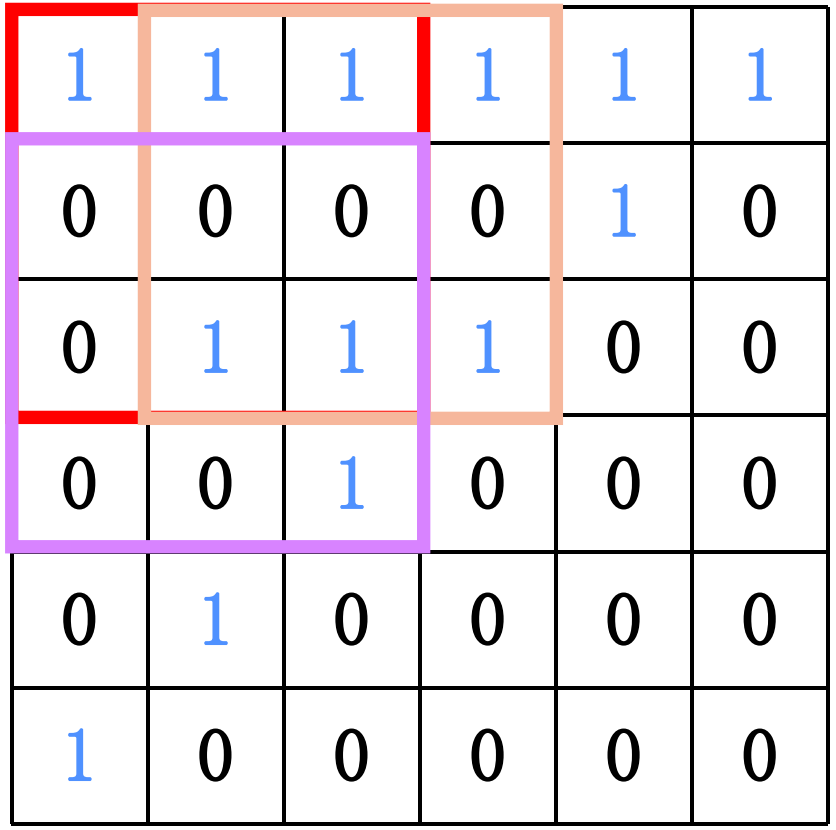
Filter 1

1	1	1
−1	1	−1
1	−1	−1

- One filter Connects to small region with less parameters at a time

$$1 \times 1 + 1 \times 1 + 1 \times 1$$
$$+0 \times (-1) + 0 \times 1 + 0 \times (-1)$$
$$+0 \times 1 + 1 \times (-1) + 1 \times (-1)$$

CNN – Convolution Layer



6 × 6 image

Stride = 1

Filter 1

1	1	1
−1	1	−1
1	−1	−1

1 2 −1

- One filter uses the same set of parameters for different regions

CNN – Convolution Layer

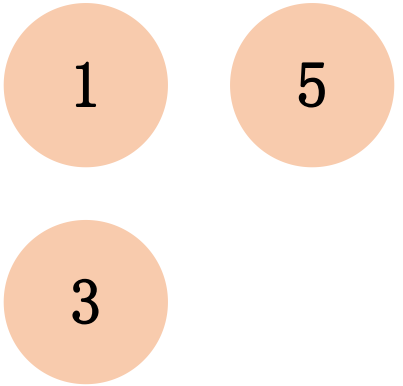
1	1	1	1	1	1
0	0	0	0	1	0
0	1	1	1	0	0
0	0	1	0	0	0
0	1	0	0	0	0
1	0	0	0	0	0

6 × 6 image

Stride = 3

Filter 1

1	1	1
−1	1	−1
1	−1	−1



- One filter uses the same set of parameters for different regions

CNN – Convolution Layer

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

6 × 6 image

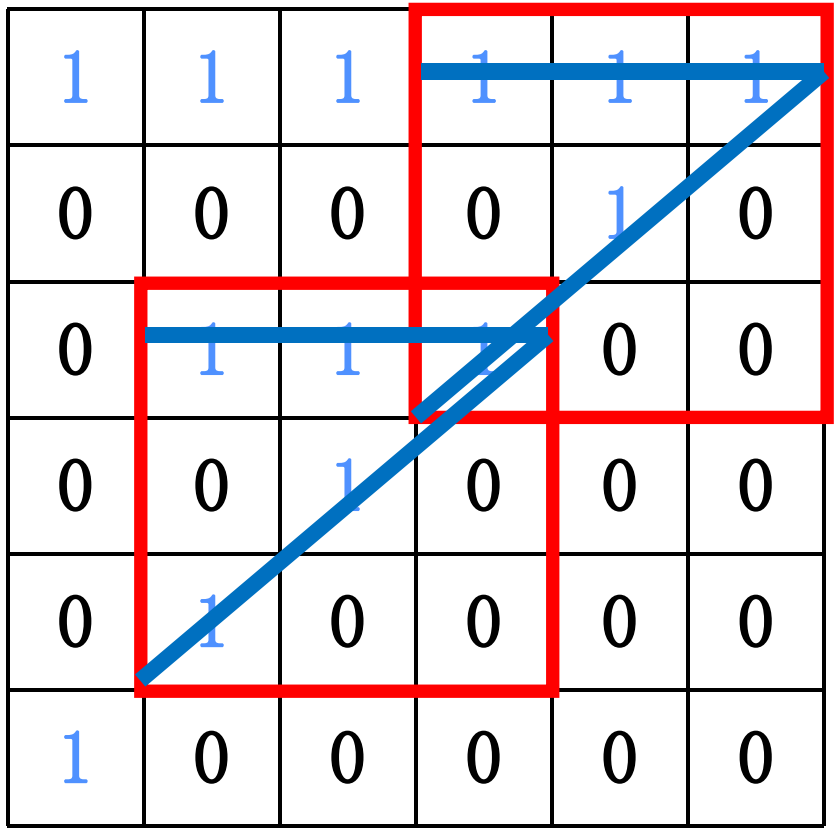
Stride = 1

1	1	1
−1	1	−1
1	−1	−1

Filter 1

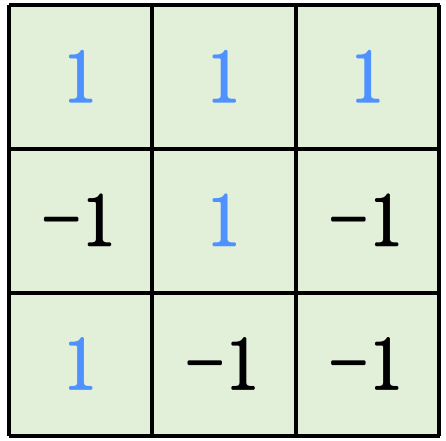
1	2	2	5
−1	−2	2	0
0	5	1	1
3	0	1	0

CNN – Convolution Layer

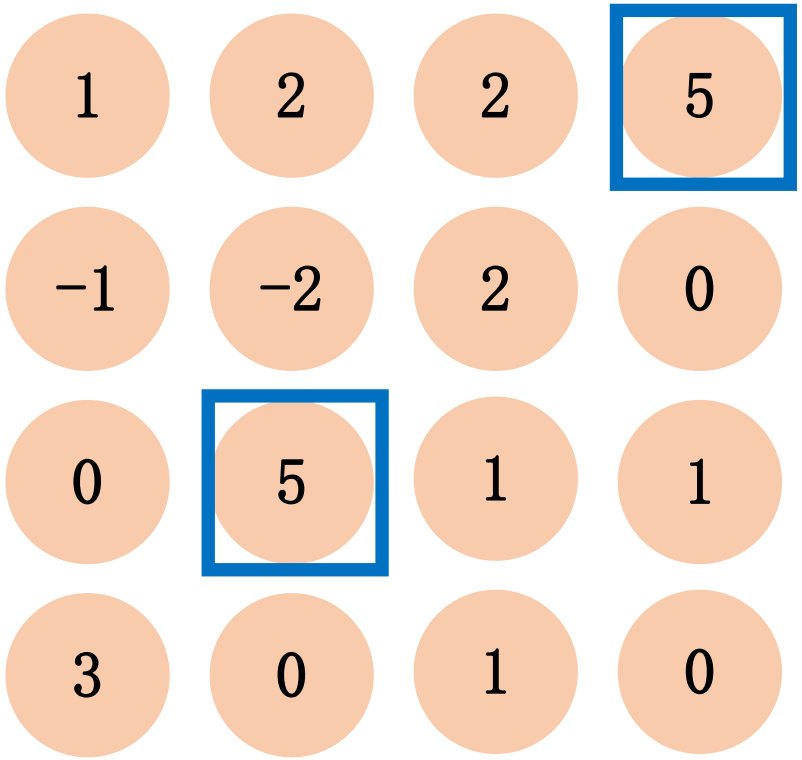


6 × 6 image

Stride = 1



Filter 1



CNN – Convolution Layer

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

6 × 6 image

Stride = 1

1	1	1
−1	1	−1
1	−1	−1

Filter 1

1	2	2	5
−1	−2	2	0
0	5	1	1
3	0	1	0

CNN – Convolution Layer

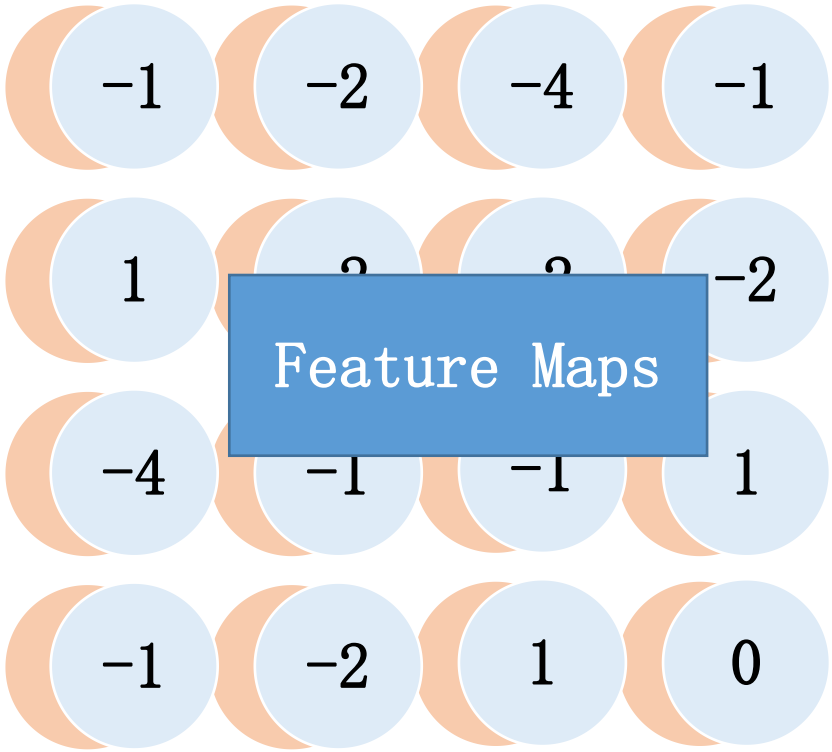
1	1	1	1	1	1
0	0	0	0	1	0
0	1	1	1	0	0
0	0	1	0	0	0
0	1	0	0	0	0
1	0	0	0	0	0

6 × 6 image

Stride = 1

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

Filter 2



2 × 4 × 4 Images

Ignore the bias and activation function!

CNN – Convolution Layer

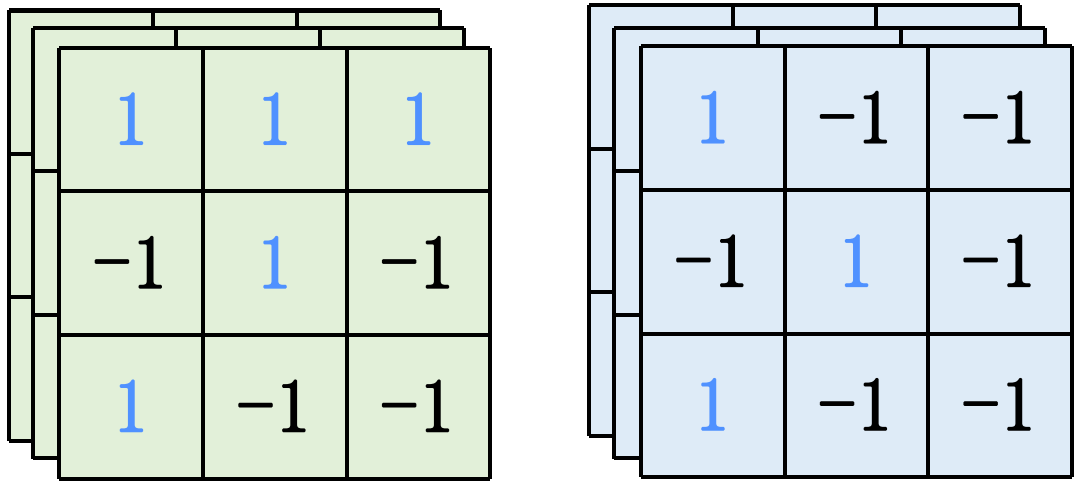
1. How to deal with the color images?

2. How many parameters in the convolution layer?

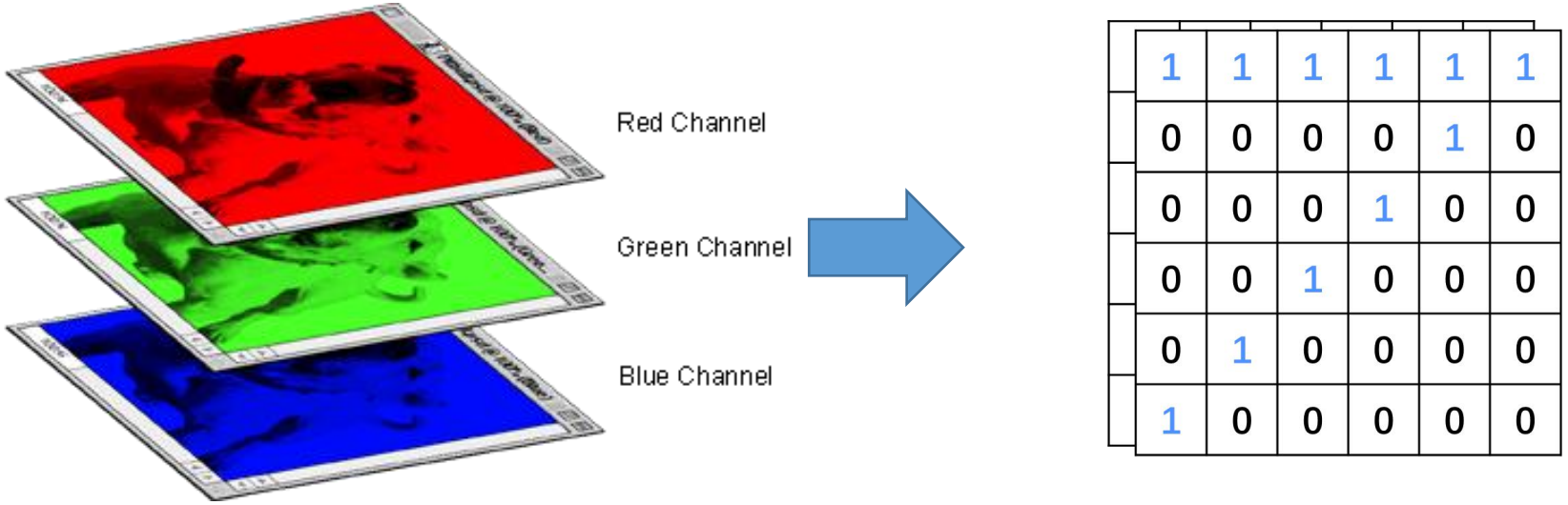
3. What is the size of the feature maps?

CNN – Convolution Layer

- How to deal with the color images?



colorful images

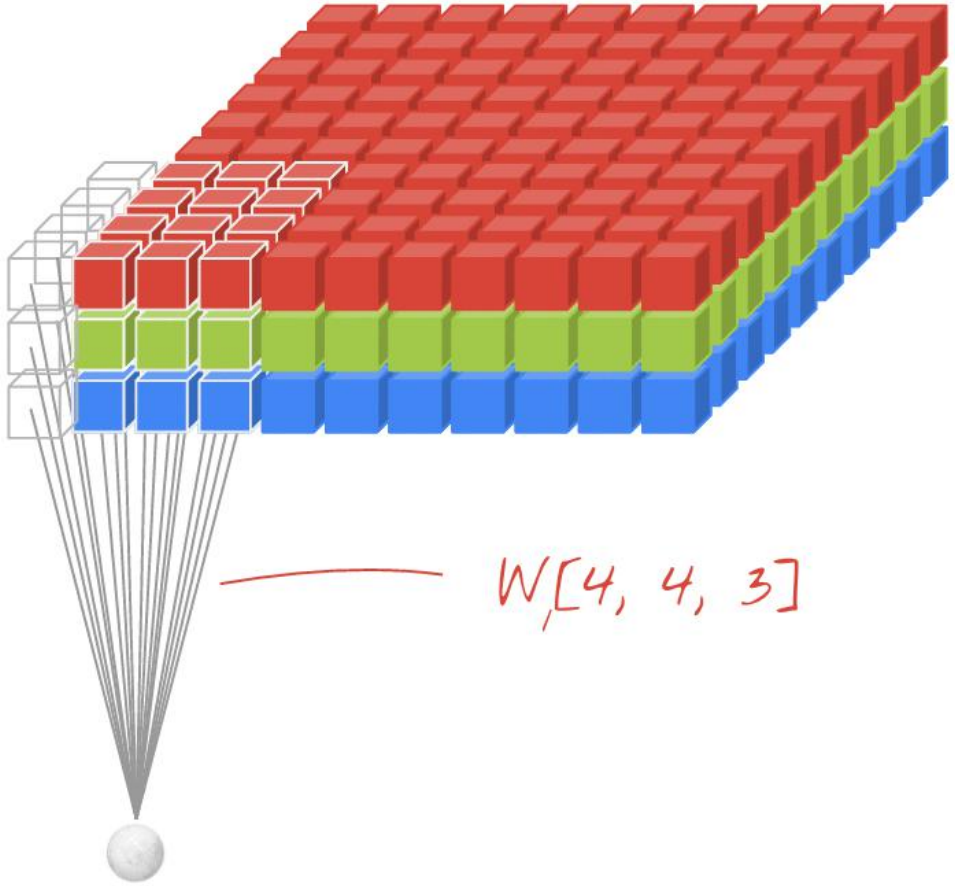


Feature maps is also a “colorful images”!

Image from: https://stackoverflow.com/questions/14113772/what-does-it-mean-to-change-the-color-channel

CNN – Convolution Layer

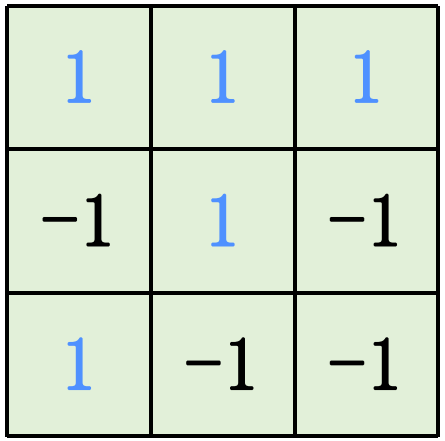
- How to deal with the colorful images?



https://sites.google.com/site/nttrungmtwiki/home/it/data-science---python/tensorflow/tensorflow-and-deep-learning-part-

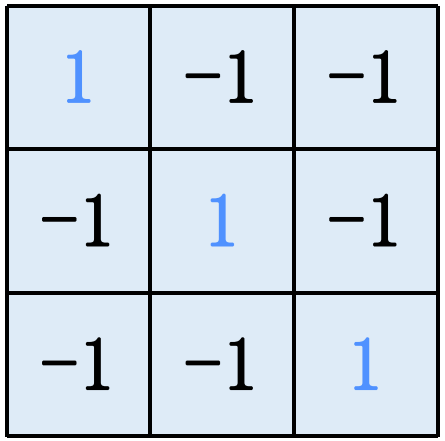
CNN – Convolution Layer

- How many parameters in the convolution layer?



Filter 1

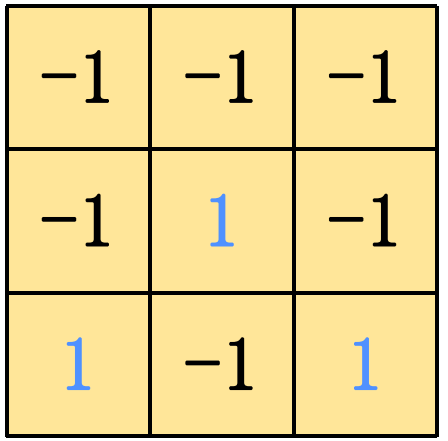
$9 = 3 \times 3$



Filter 2

$9 = 3 \times 3$

……



Filter N

$9 = 3 \times 3$

There are $9N$ parameters

The power of sharing weights!

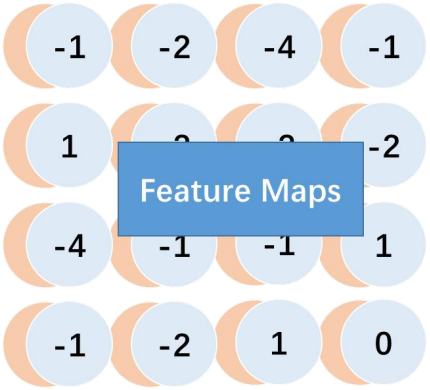
CNN – Convolution Layer

- What is the size of the feature maps?

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

6×6 image

2 filters
Filter size:
3×3
Stride:1



$2 \times 4 \times 4$ image

Input image: $C \times H \times W$

Filters: $N \times n \times n$

Stride: s

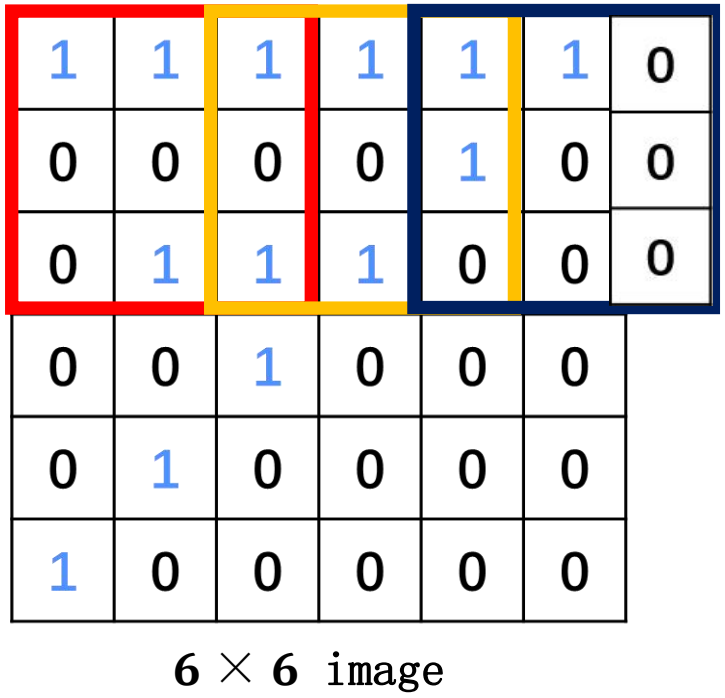
Feature maps=? $N \times \left(\frac{H-n}{s}+1\right) \times \left(\frac{W-n}{s}+1\right)$

May not divisible!

CNN – Convolution Layer

- Padding

Stride = 2



6 × 6 image

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

- Two ways

1. Ignore the extra pixels

2. Fill with zero

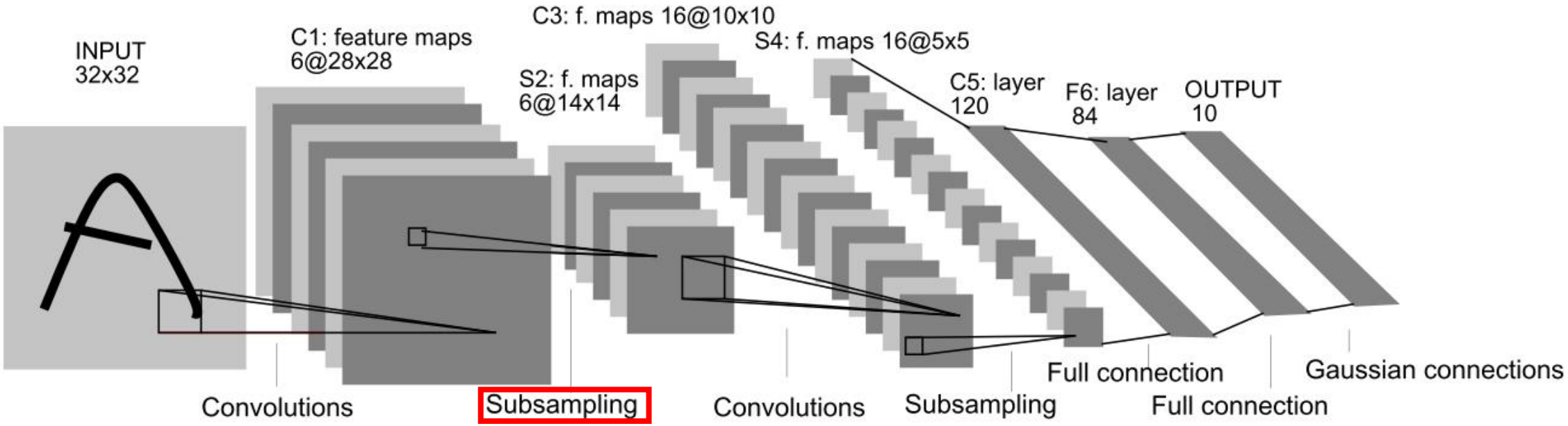
   Add P_H and P_W zeros

$$\text{Feature maps} = N \times \left(\frac{H - n + P_H}{s} + 1\right) \times \left(\frac{W - n + P_W}{s} + 1\right)$$

参考 cs231n 2017 lecture5, page62和 https://www.tensorflow.org/api_guides/python/nn#Notes_on_SAME_Convolution_Padding

CNN – Pooling Layer

- LeNet 5



CNN – Pooling Layer

- The output from the convolution layer can be huge

$$\left.\begin{array}{l}\text{Input image: } 1 \times 96 \times 96 \\ \text{Filters: } 400 \times 8 \times 8 \\ \text{Stride: } 1\end{array}\right\} \text{Feature maps} = 400 \times (96 - 8 + 1) \times (96 - 8 + 1)$$

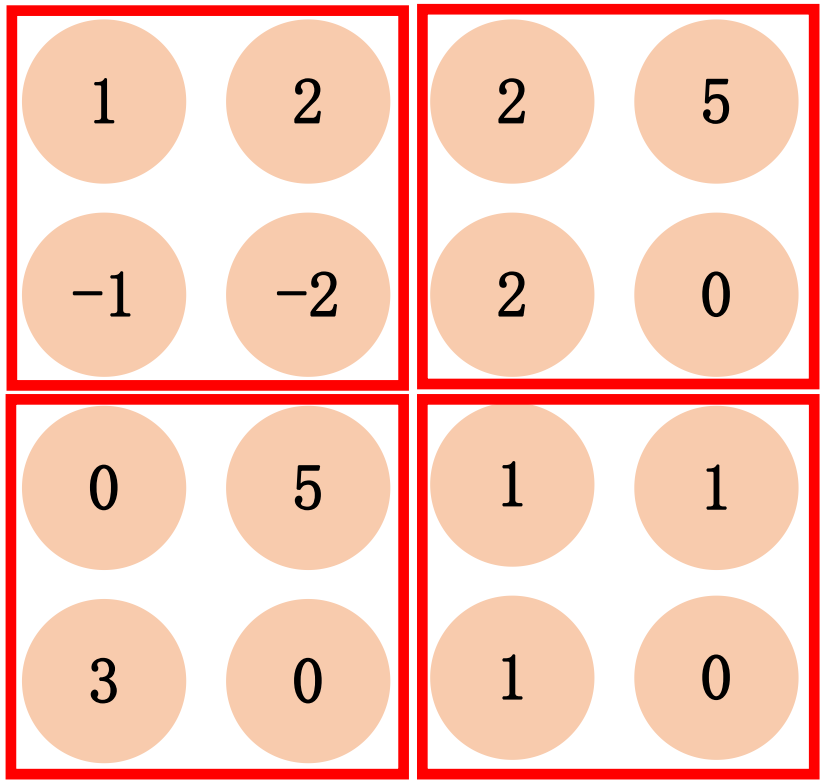
- Output of the convolution layer

$$3168400 = 400 \times (96 - 8 + 1) \times (96 - 8 + 1)$$

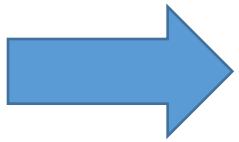
  - Hard to train
  - Overfitting

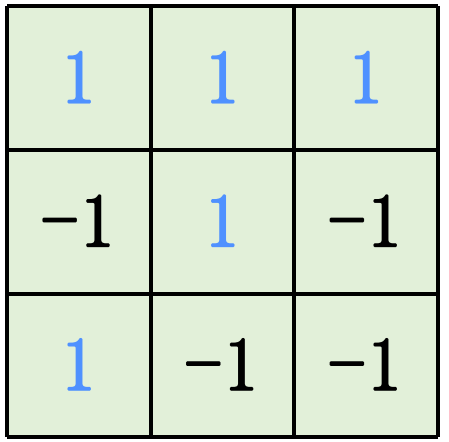
CNN – Pooling Layer

- Max Pooling

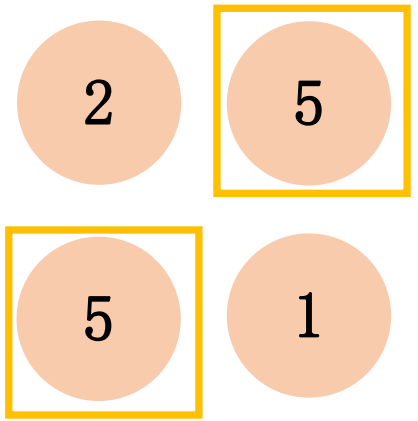


Convolved Feature





Filter 1

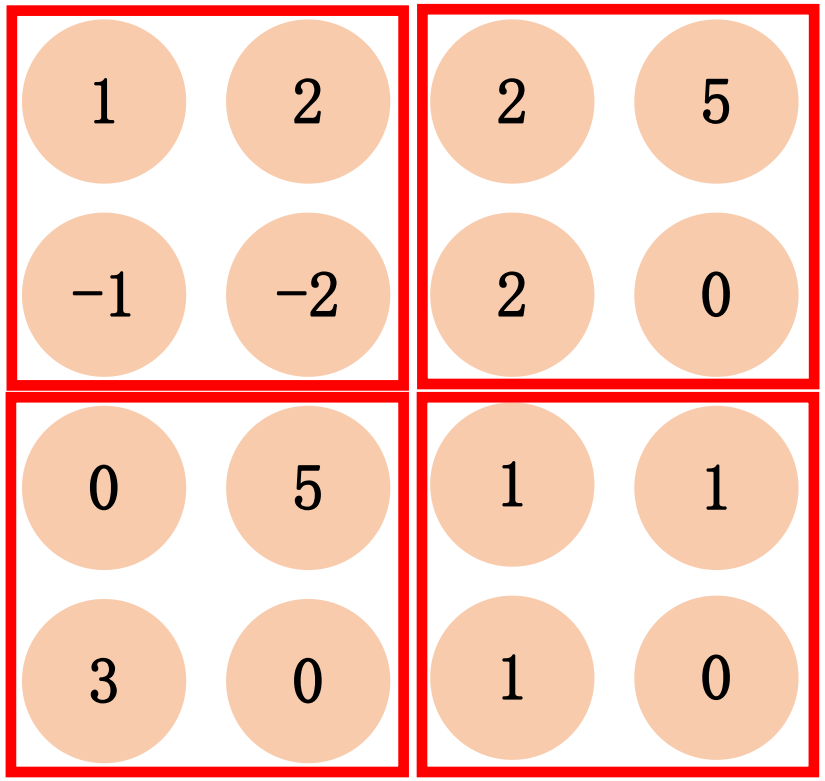


Pooled Feature

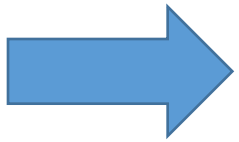
- Using subsampling to lessen the parameters

CNN – Pooling Layer

- Average Pooling

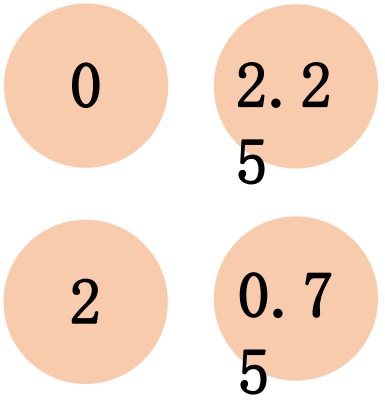


Convolved Feature



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

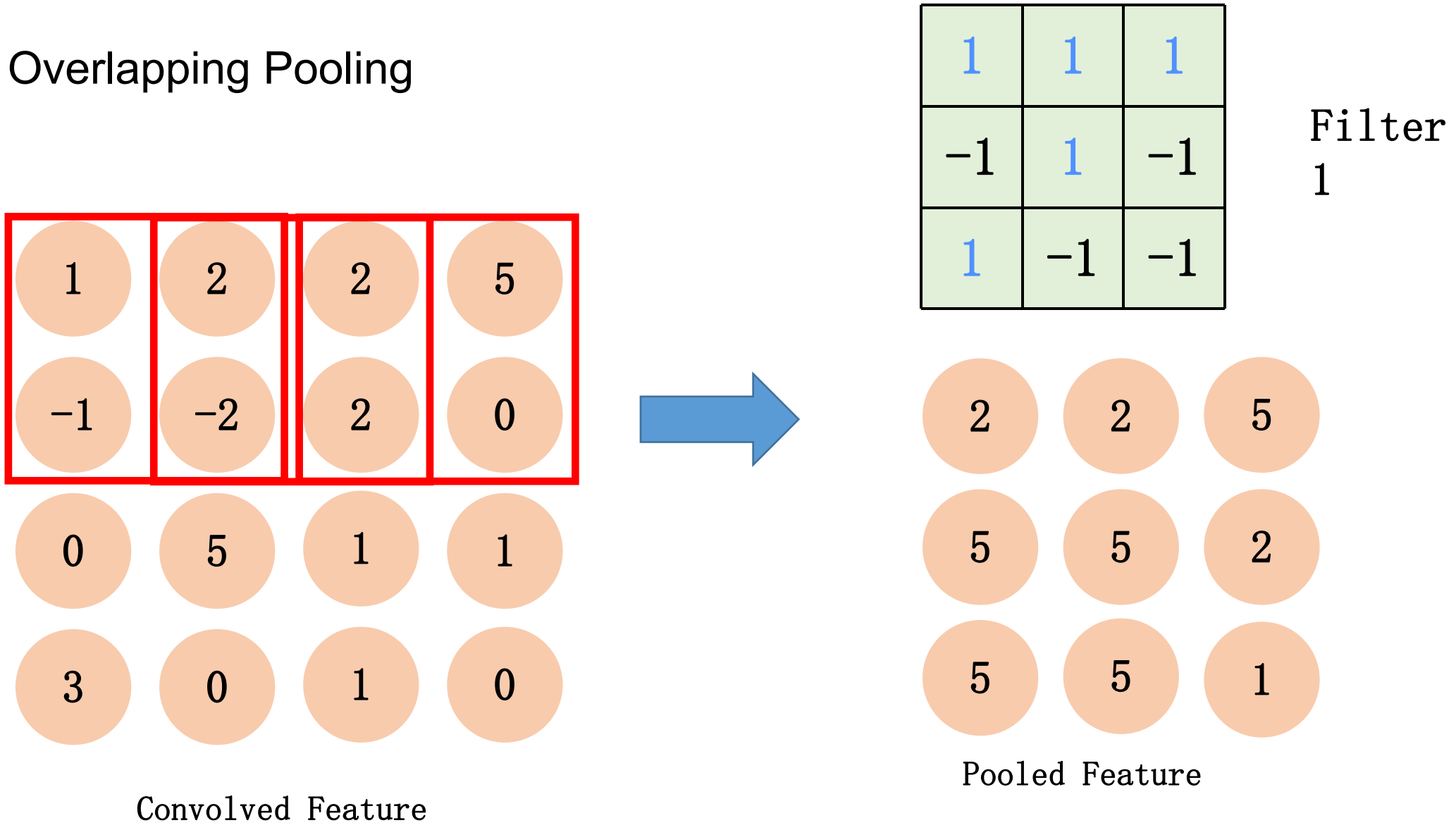
Filter 1



Pooled Feature

CNN – Pooling Layer

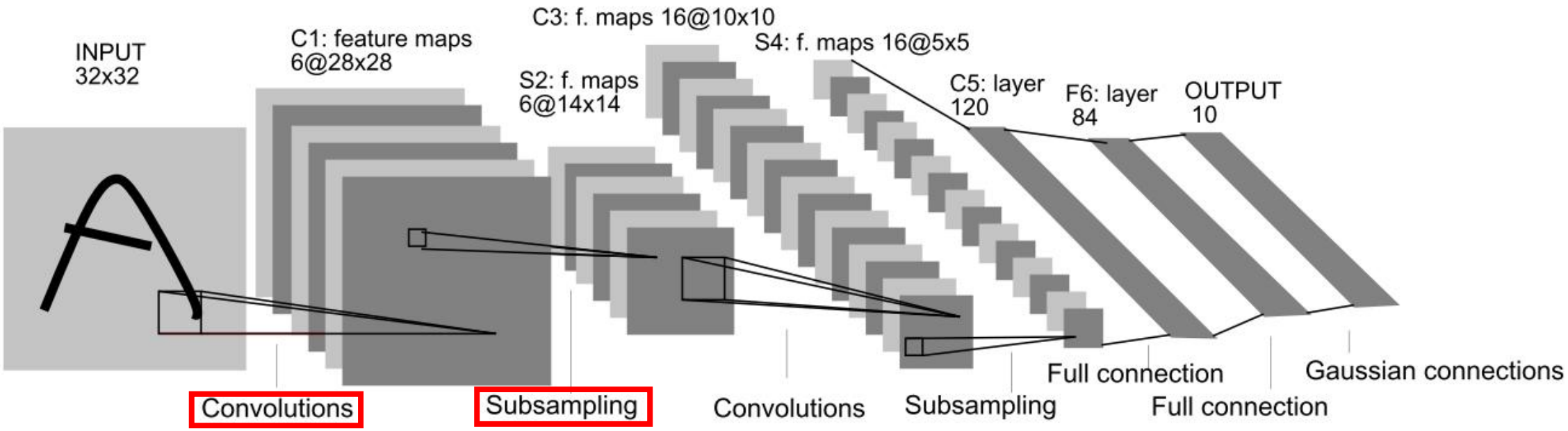
- Overlapping Pooling



[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural

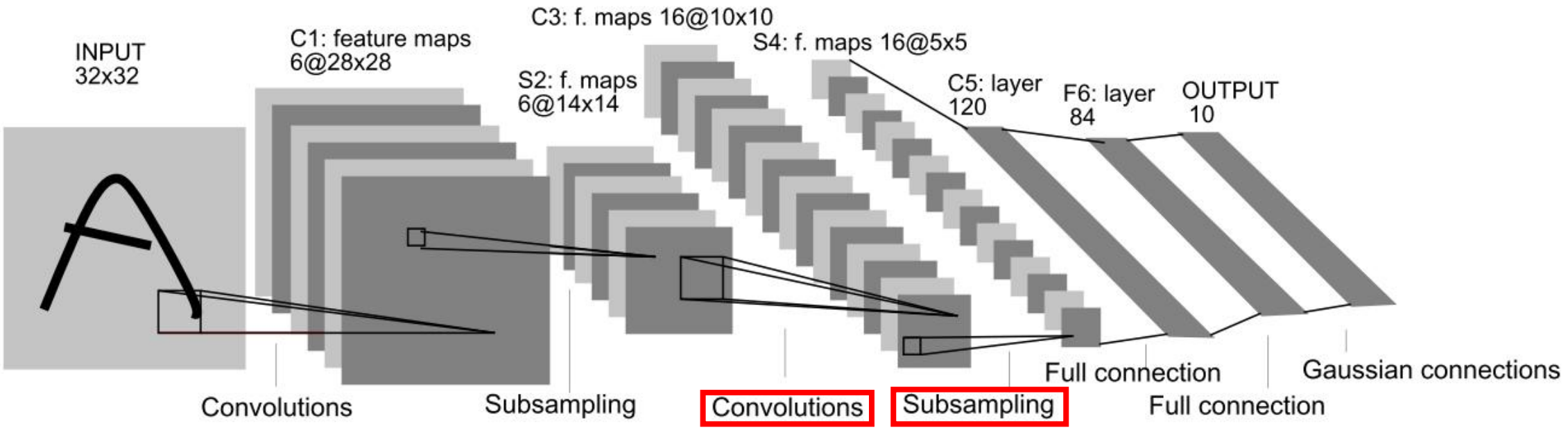
CNN - Convolution + Pooling

- LeNet 5



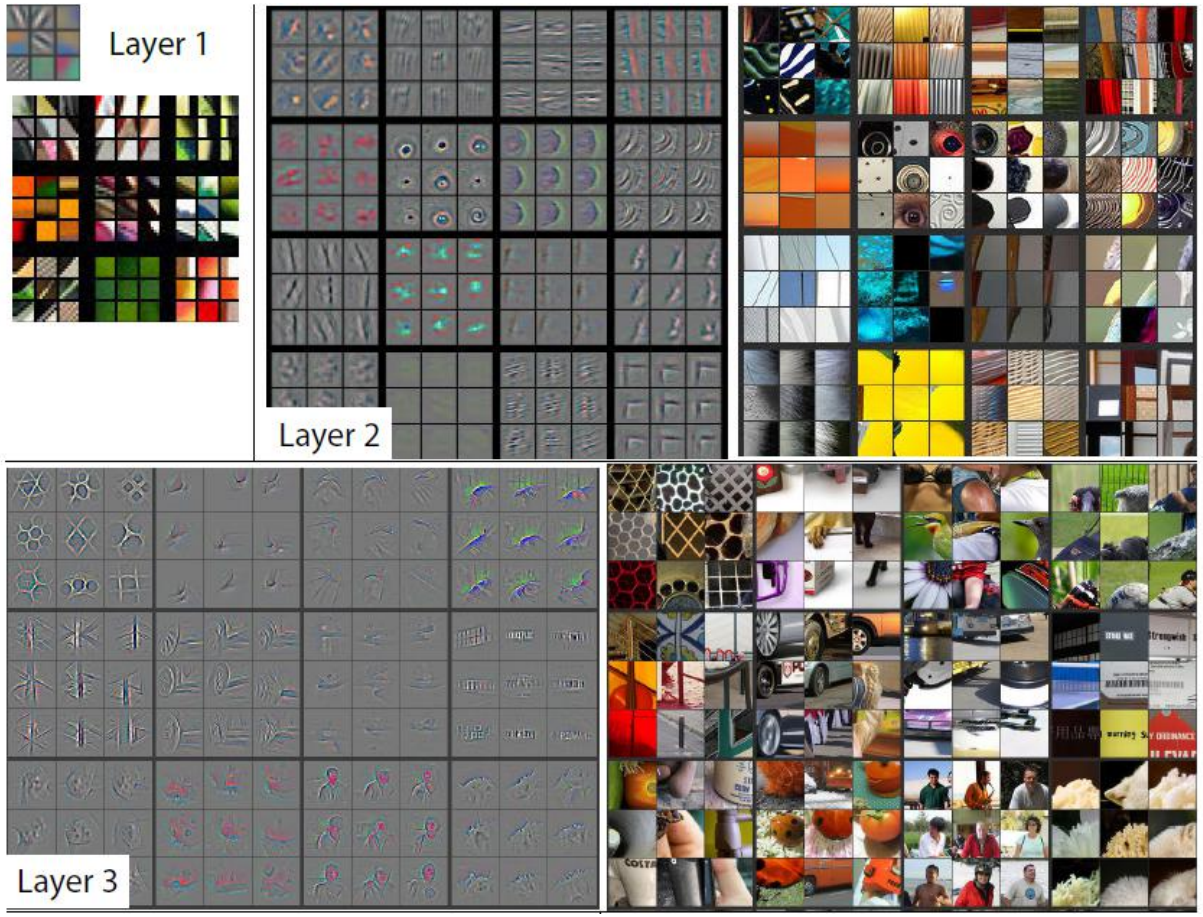
CNN - Convolution + Pooling

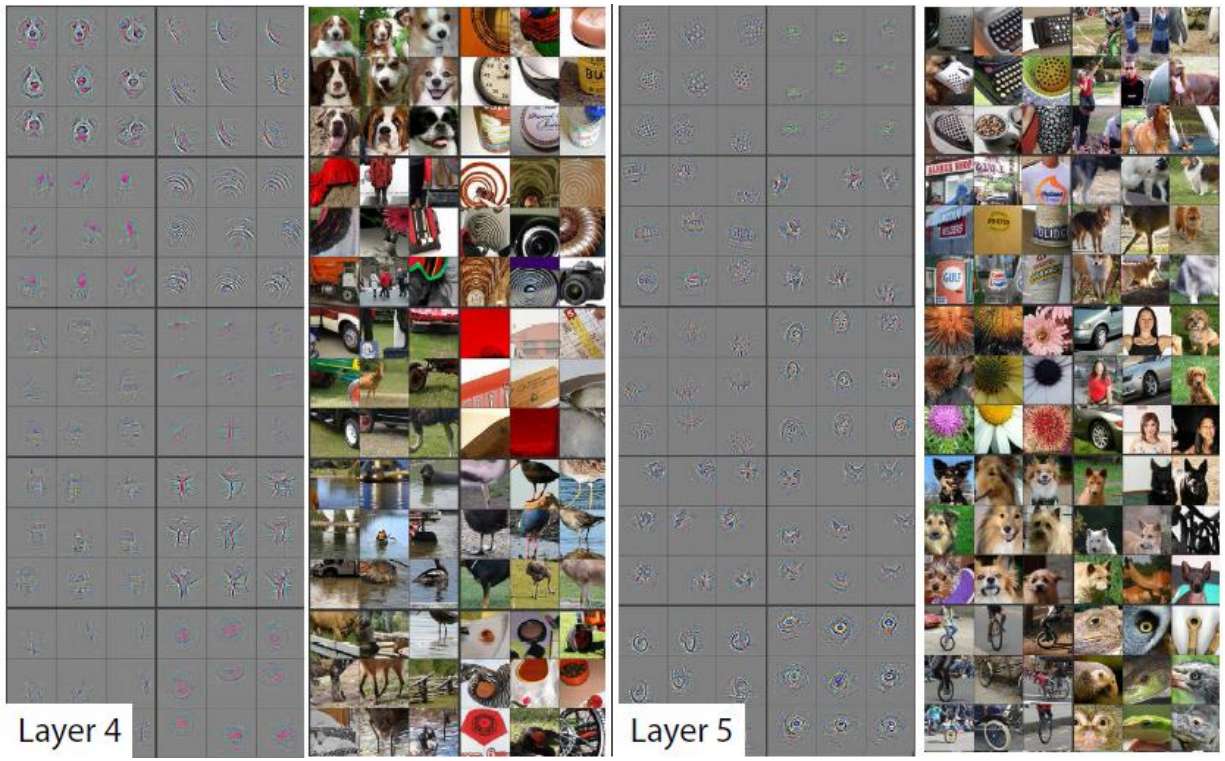
- LeNet 5



- Why more convolution and pooling layer?

Learned Features by CNN





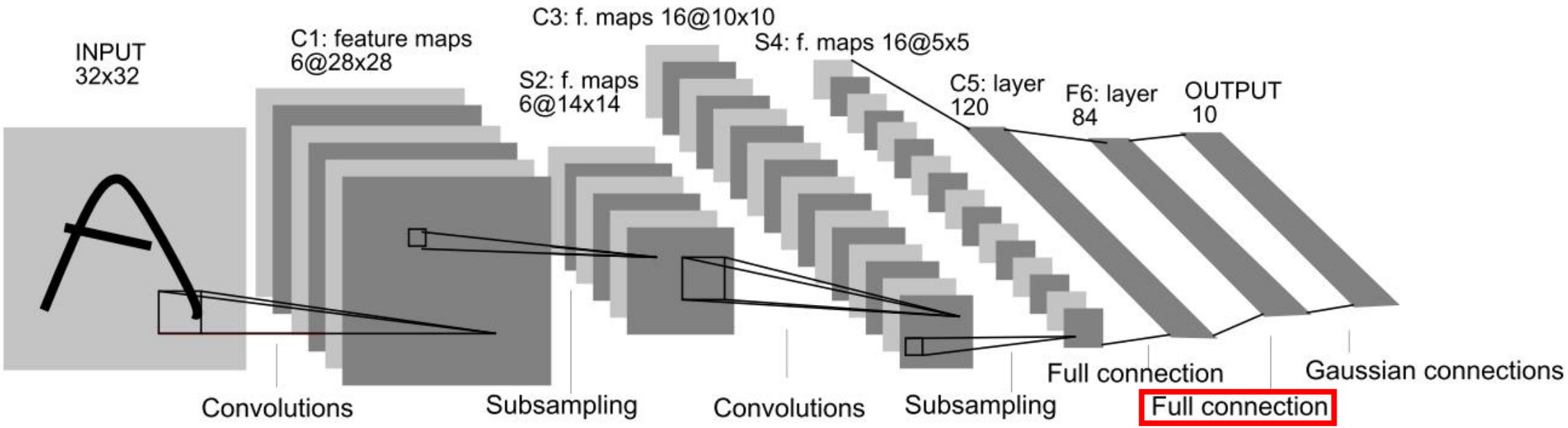
More details in
https://www.youtube.com/watch?v=ghEmQSxT6tw

- Deeper layers, more specific features

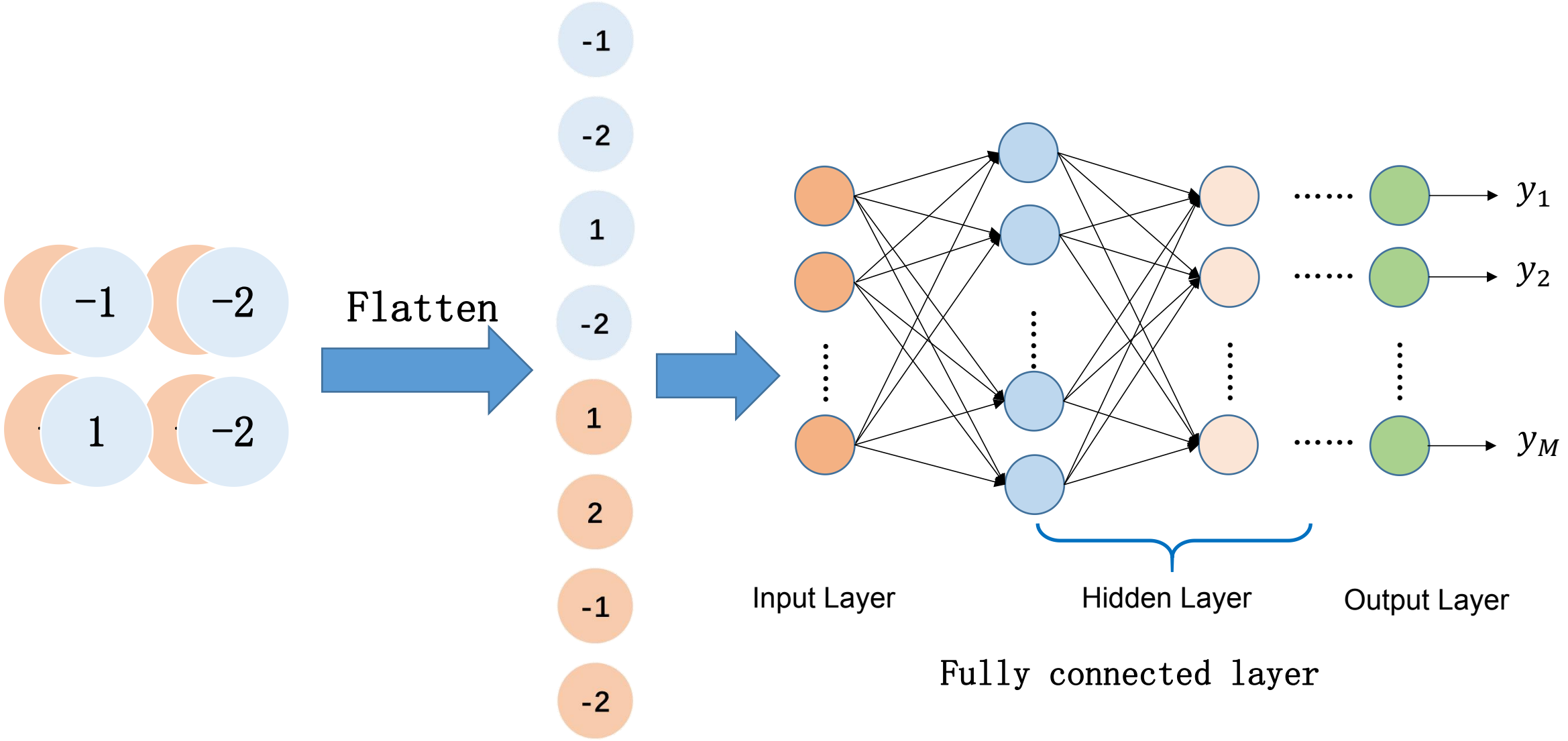
Zeiler M D, Fergus R. Visualizing and Understanding Convolutional Networks[J]. 2013, 8689:818-833.

CNN – Fully connected layer

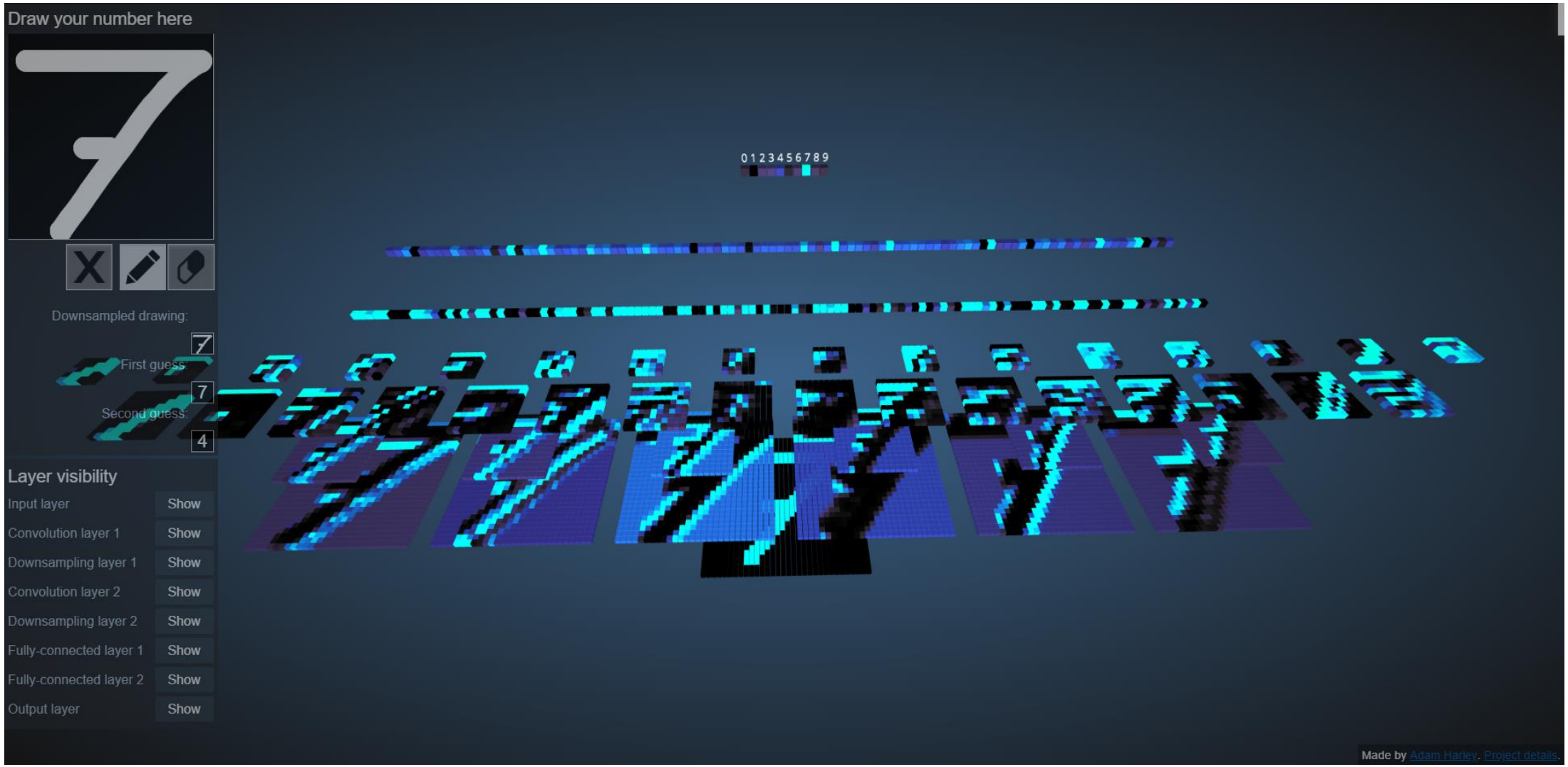
- LeNet 5



CNN – Fully connected layer



CNN – Visualization



3D convolutional network visualization http://scs.ryerson.ca/~aharley/vis/conv/

A. W. Harley, "An Interactive Node-Link Visualization of Convolutional Neural Networks," in ISVC, pages 867-

CNN – Hyperparameters

- Convolution layers
  - Number of filters
  - Size of filters
  - Stride
- Pooling layers
  - Window size
  - Window stride
- Fully connected layers
  - Number of layers
  - Number of neurons

CNN – Traditional Methods

- Comparison

	Traditional Methods	CNN
Filters	Manually design	Learn automatically
Layers	Few	Can be quite some
Features	Low level features	From low to high level features

CNN is more powerful!

Learning a CNN

Loss functions

- Mean squared error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2$$

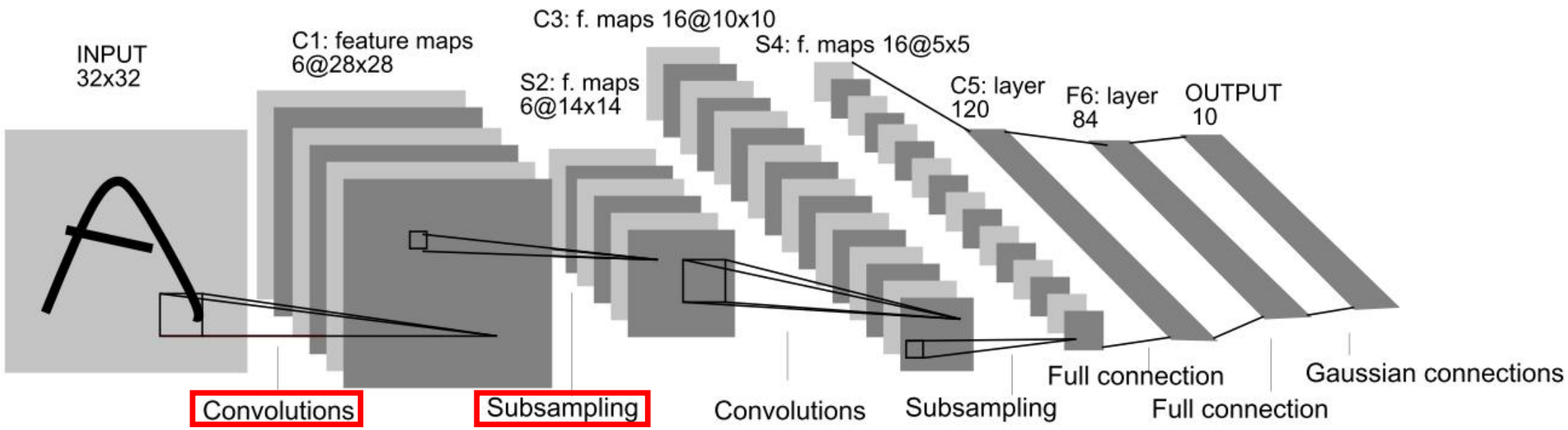
- Cross entropy loss

$$\text{Loss} = -\sum_{i=1}^{n} Y_i \log p_i$$

- User defined loss

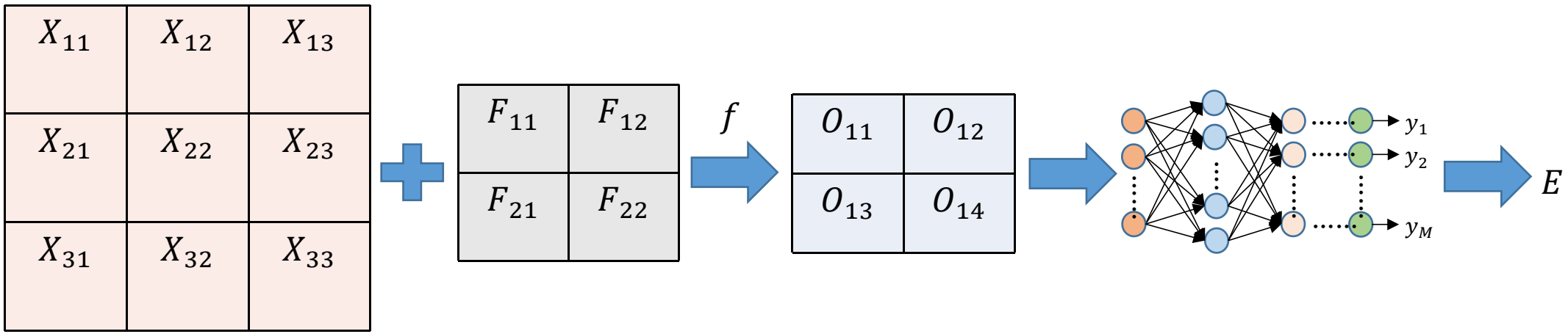
Backpropagation

- A simple but important CNN – LeNet 5



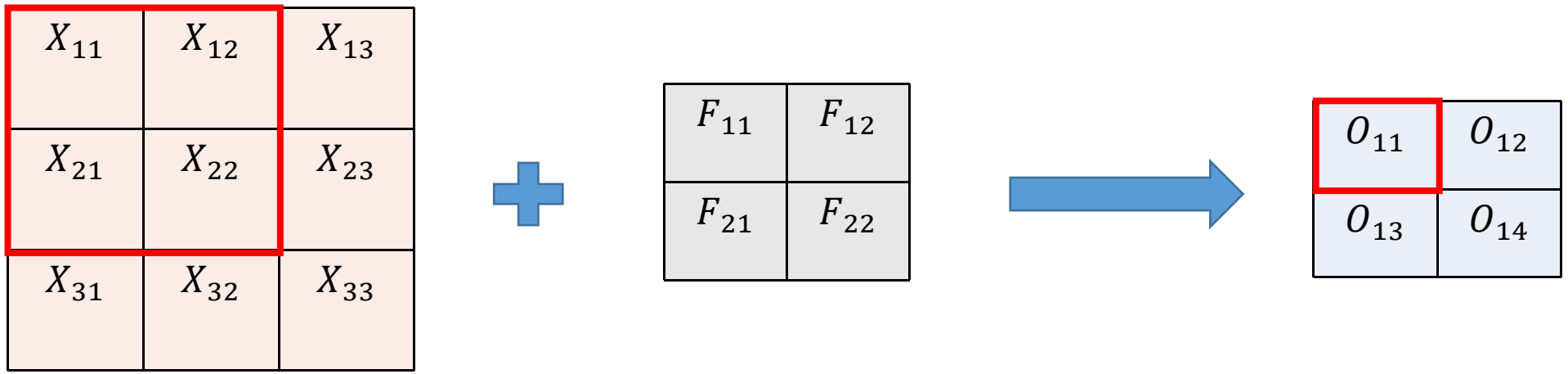
- Back-propagation – Chain rule
- How to compute the gradients of convolution layers and pooling layers?

Backpropagation – Convolution Layer



- The output of convolution operation $\boldsymbol{O} = f(\boldsymbol{F}, \boldsymbol{X})$
- The loss E
- Assume that we have already computed $\frac{\partial E}{\partial O_{ij}}$, and of course all partial derivatives of latter layers

Backpropagation – Convolution Layer



3×3 Input

2×2 Filter

2×2 Output

$$\boldsymbol{O} = f(\boldsymbol{F}; \boldsymbol{X})$$

$$O_{11} = F_{11}X_{11} + F_{12}X_{12} + F_{21}X_{21} + F_{22}X_{22}$$

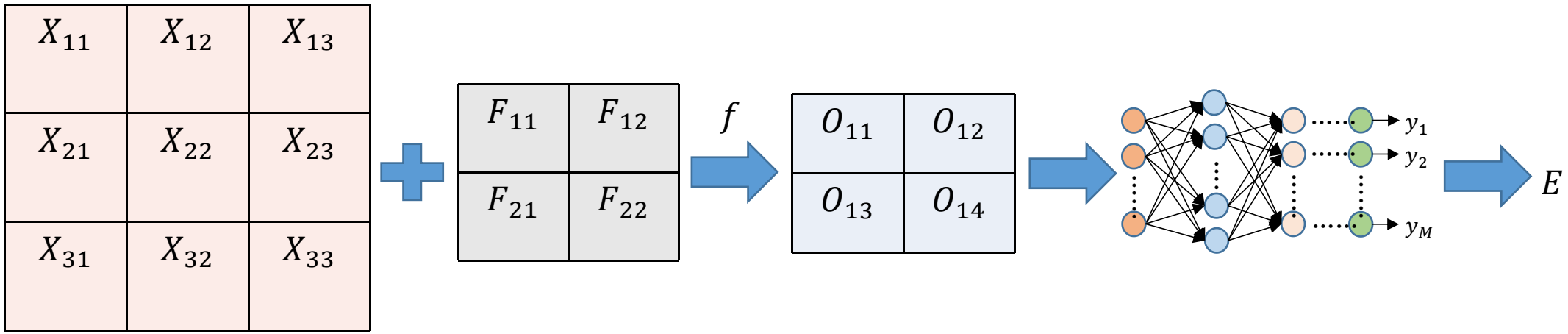
$$O_{12} = F_{11}X_{12} + F_{12}X_{13} + F_{21}X_{22} + F_{22}X_{23}$$

$$O_{21} = F_{11}X_{21} + F_{12}X_{22} + F_{21}X_{31} + F_{22}X_{32}$$

$$O_{22} = F_{11}X_{22} + F_{12}X_{23} + F_{21}X_{32} + F_{22}X_{33}$$

No bias!

Backpropagation – Convolution Layer



- How to compute $\frac{\partial E}{\partial F_{11}}$?

Chain Rule

$$\frac{\partial E}{\partial F_{11}} = \left(\frac{\partial E}{\partial \boldsymbol{O}}\right)^T \frac{\partial \boldsymbol{O}}{\partial F_{11}}$$

$$\frac{\partial E}{\partial \boldsymbol{O}} = \nabla E(O_{11}, O_{12}, O_{21}, O_{22})$$

$$= \left(\frac{\partial E}{\partial O_{11}}, \frac{\partial E}{\partial O_{12}}, \frac{\partial E}{\partial O_{21}}, \frac{\partial E}{\partial O_{22}}\right)^T$$

$$\boldsymbol{O} = f(\boldsymbol{F}, \boldsymbol{X}) = (O_{11}, O_{12}, O_{21}, O_{22})^T \triangleq \left(f_1(\boldsymbol{F}, \boldsymbol{X}), f_2(\boldsymbol{F}, \boldsymbol{X}), f_3(\boldsymbol{F}, \boldsymbol{X}), f_4(\boldsymbol{F}, \boldsymbol{X})\right)^T$$

$$\frac{\partial \boldsymbol{O}}{\partial F_{11}} = \left(\frac{\partial f_1}{\partial F_{11}}, \frac{\partial f_2}{\partial F_{11}}, \frac{\partial f_3}{\partial F_{11}}, \frac{\partial f_4}{\partial F_{11}}\right)^T$$

Backpropagation – Convolution Layer

$$\frac{\partial E}{\partial F_{11}} = \frac{\partial E}{\partial O_{11}}\frac{\partial f_1}{\partial F_{11}} + \frac{\partial E}{\partial O_{12}}\frac{\partial f_2}{\partial F_{11}} + \frac{\partial E}{\partial O_{21}}\frac{\partial f_3}{\partial F_{11}} + \frac{\partial E}{\partial O_{22}}\frac{\partial f_4}{\partial F_{11}}$$

$$\frac{\partial E}{\partial F_{12}} = \frac{\partial E}{\partial O_{11}}\frac{\partial f_1}{\partial F_{12}} + \frac{\partial E}{\partial O_{12}}\frac{\partial f_2}{\partial F_{12}} + \frac{\partial E}{\partial O_{21}}\frac{\partial f_3}{\partial F_{12}} + \frac{\partial E}{\partial O_{22}}\frac{\partial f_4}{\partial F_{12}}$$

$$\frac{\partial E}{\partial F_{21}} = \frac{\partial E}{\partial O_{11}}\frac{\partial f_1}{\partial F_{21}} + \frac{\partial E}{\partial O_{12}}\frac{\partial f_2}{\partial F_{21}} + \frac{\partial E}{\partial O_{21}}\frac{\partial f_3}{\partial F_{21}} + \frac{\partial E}{\partial O_{22}}\frac{\partial f_4}{\partial F_{21}}$$

$$\frac{\partial E}{\partial F_{22}} = \frac{\partial E}{\partial O_{11}}\frac{\partial f_1}{\partial F_{22}} + \frac{\partial E}{\partial O_{12}}\frac{\partial f_2}{\partial F_{22}} + \frac{\partial E}{\partial O_{21}}\frac{\partial f_3}{\partial F_{22}} + \frac{\partial E}{\partial O_{22}}\frac{\partial f_4}{\partial F_{22}}$$

Backpropagation – Convolution Layer

$$\boldsymbol{O} = f(\boldsymbol{F}; \boldsymbol{X})$$

$$f_1(\boldsymbol{X}) = O_{11} = F_{11}X_{11} + F_{12}X_{12} + F_{21}X_{21} + F_{22}X_{22}$$

$$f_2(\boldsymbol{X}) = O_{12} = F_{11}X_{12} + F_{12}X_{13} + F_{21}X_{22} + F_{22}X_{23}$$

$$f_3(\boldsymbol{X}) = O_{21} = F_{11}X_{21} + F_{12}X_{22} + F_{21}X_{31} + F_{22}X_{32}$$

$$f_4(\boldsymbol{X}) = O_{22} = F_{11}X_{22} + F_{12}X_{23} + F_{21}X_{32} + F_{22}X_{33}$$

$$\frac{\partial f_1}{\partial F_{11}} = X_{11} \quad \frac{\partial f_2}{\partial F_{11}} = X_{12} \quad \frac{\partial f_3}{\partial F_{11}} = X_{21} \quad \frac{\partial f_4}{\partial F_{11}} = X_{22}$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$\frac{\partial f_1}{\partial F_{22}} = X_{22} \quad \frac{\partial f_2}{\partial F_{22}} = X_{23} \quad \frac{\partial f_3}{\partial F_{22}} = X_{32} \quad \frac{\partial f_4}{\partial F_{22}} = X_{33}$$

Backpropagation – Convolution Layer

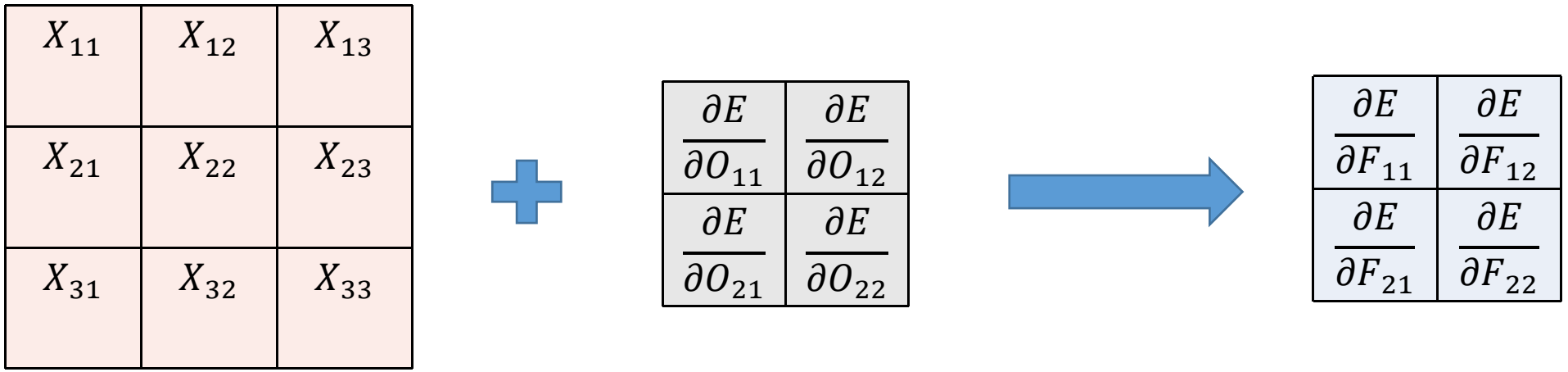
$$\frac{\partial E}{\partial F_{11}} = \frac{\partial E}{\partial O_{11}} X_{11} + \frac{\partial E}{\partial O_{12}} X_{12} + \frac{\partial E}{\partial O_{21}} X_{21} + \frac{\partial E}{\partial O_{22}} X_{22}$$

$$\frac{\partial E}{\partial F_{12}} = \frac{\partial E}{\partial O_{11}} X_{12} + \frac{\partial E}{\partial O_{12}} X_{13} + \frac{\partial E}{\partial O_{21}} X_{22} + \frac{\partial E}{\partial O_{22}} X_{23}$$

$$\frac{\partial E}{\partial F_{21}} = \frac{\partial E}{\partial O_{11}} X_{21} + \frac{\partial E}{\partial O_{12}} X_{22} + \frac{\partial E}{\partial O_{21}} X_{31} + \frac{\partial E}{\partial O_{22}} X_{32}$$

$$\frac{\partial E}{\partial F_{22}} = \frac{\partial E}{\partial O_{11}} X_{22} + \frac{\partial E}{\partial O_{12}} X_{23} + \frac{\partial E}{\partial O_{21}} X_{32} + \frac{\partial E}{\partial O_{22}} X_{33}$$

Backpropagation – Convolution Layer



$$\frac{\partial E}{\partial F_{11}} = \frac{\partial E}{\partial O_{11}} X_{11} + \frac{\partial E}{\partial O_{12}} X_{12} + \frac{\partial E}{\partial O_{21}} X_{21} + \frac{\partial E}{\partial O_{22}} X_{22}$$

$$\frac{\partial E}{\partial F_{12}} = \frac{\partial E}{\partial O_{11}} X_{12} + \frac{\partial E}{\partial O_{12}} X_{13} + \frac{\partial E}{\partial O_{21}} X_{22} + \frac{\partial E}{\partial O_{22}} X_{23}$$

$$\frac{\partial E}{\partial F_{21}} = \frac{\partial E}{\partial O_{11}} X_{21} + \frac{\partial E}{\partial O_{12}} X_{22} + \frac{\partial E}{\partial O_{21}} X_{31} + \frac{\partial E}{\partial O_{22}} X_{32}$$

$$\frac{\partial E}{\partial F_{22}} = \frac{\partial E}{\partial O_{11}} X_{22} + \frac{\partial E}{\partial O_{12}} X_{23} + \frac{\partial E}{\partial O_{21}} X_{32} + \frac{\partial E}{\partial O_{22}} X_{33}$$

Backpropagation – Convolution Layer

- How to compute $\frac{\partial E}{\partial X_{11}}$?

Chain Rule

$$\frac{\partial E}{\partial X_{11}} = \left(\frac{\partial E}{\partial \boldsymbol{O}}\right)^T \frac{\partial \boldsymbol{O}}{\partial X_{11}}$$

$$\frac{\partial E}{\partial \boldsymbol{O}} = \nabla E(O_{11}, O_{12}, O_{21}, O_{22})$$

$$= \left(\frac{\partial E}{\partial O_{11}}, \frac{\partial E}{\partial O_{12}}, \frac{\partial E}{\partial O_{21}}, \frac{\partial E}{\partial O_{22}}\right)^T$$

$$\boldsymbol{O} = f(\boldsymbol{F}, \boldsymbol{X}) = (O_{11}, O_{12}, O_{21}, O_{22})^T \triangleq \left(f_1(\boldsymbol{F}, \boldsymbol{X}), f_2(\boldsymbol{F}, \boldsymbol{X}), f_3(\boldsymbol{F}, \boldsymbol{X}), f_4(\boldsymbol{F}, \boldsymbol{X})\right)^T$$

$$\frac{\partial \boldsymbol{O}}{\partial X_{11}} = \left(\frac{\partial f_1}{\partial X_{11}}, \frac{\partial f_2}{\partial X_{11}}, \frac{\partial f_3}{\partial X_{11}}, \frac{\partial f_4}{\partial X_{11}}\right)^T$$

Backpropagation – Convolution Layer

$$\boldsymbol{O} = f(\boldsymbol{F}; \boldsymbol{X})$$

$$f_1(\boldsymbol{X}) = O_{11} = F_{11}X_{11} + F_{12}X_{12} + F_{21}X_{21} + F_{22}X_{22}$$

$$f_2(\boldsymbol{X}) = O_{12} = F_{11}X_{12} + F_{12}X_{13} + F_{21}X_{22} + F_{22}X_{23}$$

$$f_3(\boldsymbol{X}) = O_{21} = F_{11}X_{21} + F_{12}X_{22} + F_{21}X_{31} + F_{22}X_{32}$$

$$f_4(\boldsymbol{X}) = O_{22} = F_{11}X_{22} + F_{12}X_{23} + F_{21}X_{32} + F_{22}X_{33}$$

$$\frac{\partial E}{\partial X_{11}} = \frac{\partial E}{\partial O_{11}}F_{11} + \frac{\partial E}{\partial O_{12}}0 + \frac{\partial E}{\partial O_{21}}0 + \frac{\partial E}{\partial O_{22}}0$$

$$\frac{\partial E}{\partial X_{12}} = \frac{\partial E}{\partial O_{11}}F_{12} + \frac{\partial E}{\partial O_{12}}F_{11} + \frac{\partial E}{\partial O_{21}}0 + \frac{\partial E}{\partial O_{22}}0$$

$$\frac{\partial E}{\partial X_{13}} = \frac{\partial E}{\partial O_{11}}0 + \frac{\partial E}{\partial O_{12}}F_{12} + \frac{\partial E}{\partial O_{21}}0 + \frac{\partial E}{\partial O_{22}}0$$

$$\frac{\partial E}{\partial X_{21}} = \frac{\partial E}{\partial O_{11}}F_{21} + \frac{\partial E}{\partial O_{12}}0 + \frac{\partial E}{\partial O_{21}}F_{11} + \frac{\partial E}{\partial O_{22}}0$$

$$\frac{\partial E}{\partial X_{22}} = \frac{\partial E}{\partial O_{11}}F_{22} + \frac{\partial E}{\partial O_{12}}F_{21} + \frac{\partial E}{\partial O_{21}}F_{12} + \frac{\partial E}{\partial O_{22}}F_{11}$$

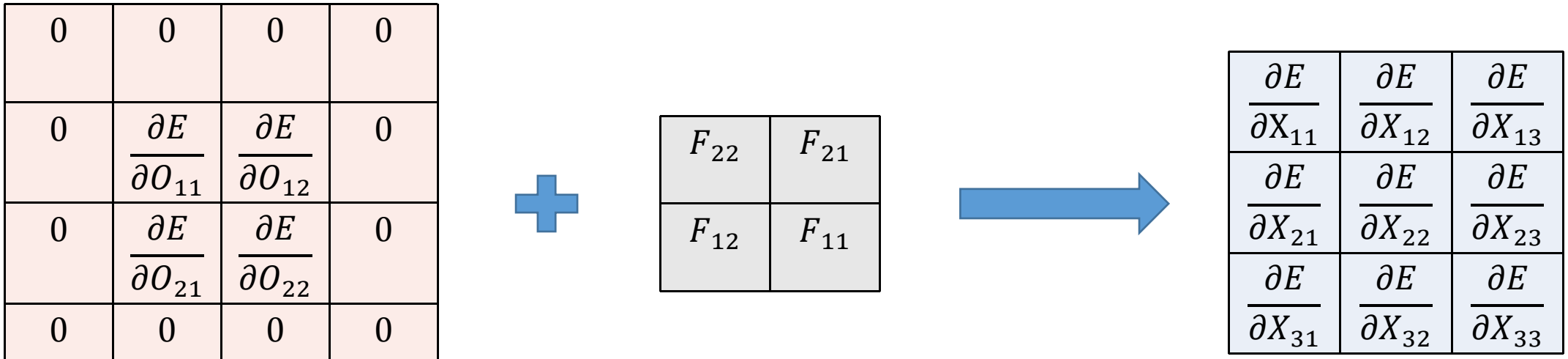
$$\frac{\partial E}{\partial X_{23}} = \frac{\partial E}{\partial O_{11}}0 + \frac{\partial E}{\partial O_{12}}F_{22} + \frac{\partial E}{\partial O_{21}}0 + \frac{\partial E}{\partial O_{22}}F_{11}$$

$$\frac{\partial E}{\partial X_{31}} = \frac{\partial E}{\partial O_{11}}0 + \frac{\partial E}{\partial O_{12}}0 + \frac{\partial E}{\partial O_{21}}F_{21} + \frac{\partial E}{\partial O_{22}}0$$

$$\frac{\partial E}{\partial X_{32}} = \frac{\partial E}{\partial O_{11}}0 + \frac{\partial E}{\partial O_{12}}0 + \frac{\partial E}{\partial O_{21}}F_{22} + \frac{\partial E}{\partial O_{22}}F_{21}$$

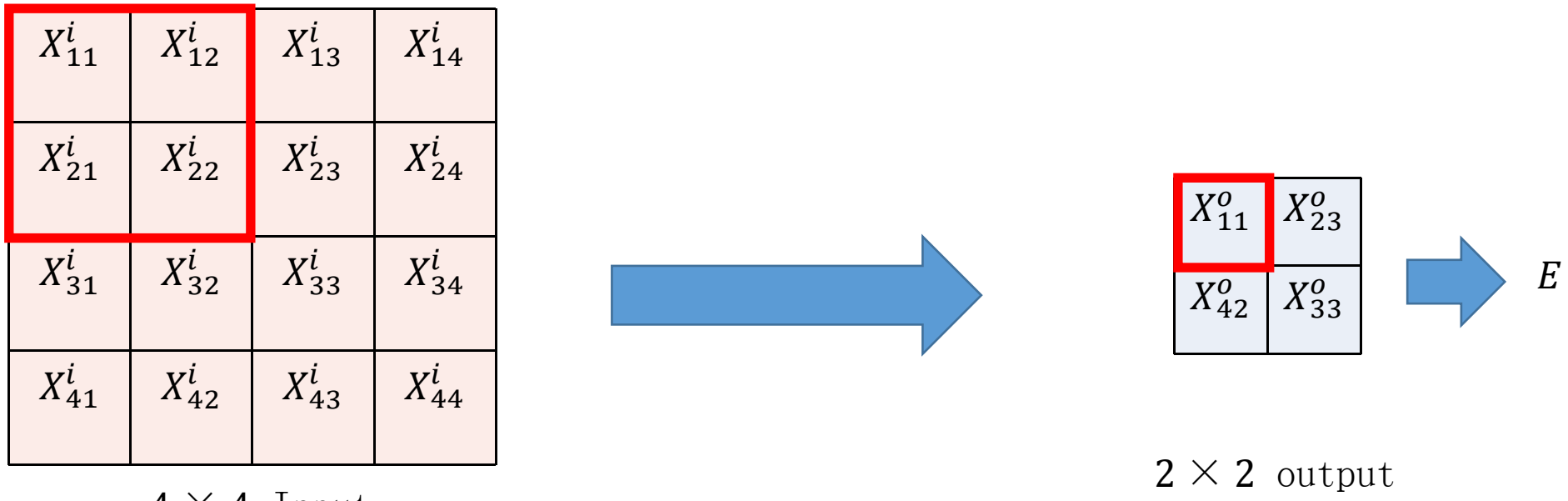
$$\frac{\partial E}{\partial X_{33}} = \frac{\partial E}{\partial O_{11}}0 + \frac{\partial E}{\partial O_{12}}0 + \frac{\partial E}{\partial O_{21}}0 + \frac{\partial E}{\partial O_{22}}F_{22}$$

Backpropagation – Convolution Layer



Backpropagation – Pooling Layer

- Max Pooling



- Only need to compute the gradients of the inputs

$$\frac{\partial E}{\partial X_{11}^i} = \frac{\partial E}{\partial X_{11}^o} \qquad \frac{\partial E}{\partial X_{12}^i} = 0 \qquad \frac{\partial E}{\partial X_{21}^i} = 0 \qquad \frac{\partial E}{\partial X_{22}^i} = 0$$

$$\frac{\partial E}{\partial X_{13}^i} = 0 \qquad \frac{\partial E}{\partial X_{14}^i} = 0 \qquad \frac{\partial E}{\partial X_{23}^i} = \frac{\partial E}{\partial X_{23}^o} \qquad \frac{\partial E}{\partial X_{24}^i} = 0$$

$$\vdots$$

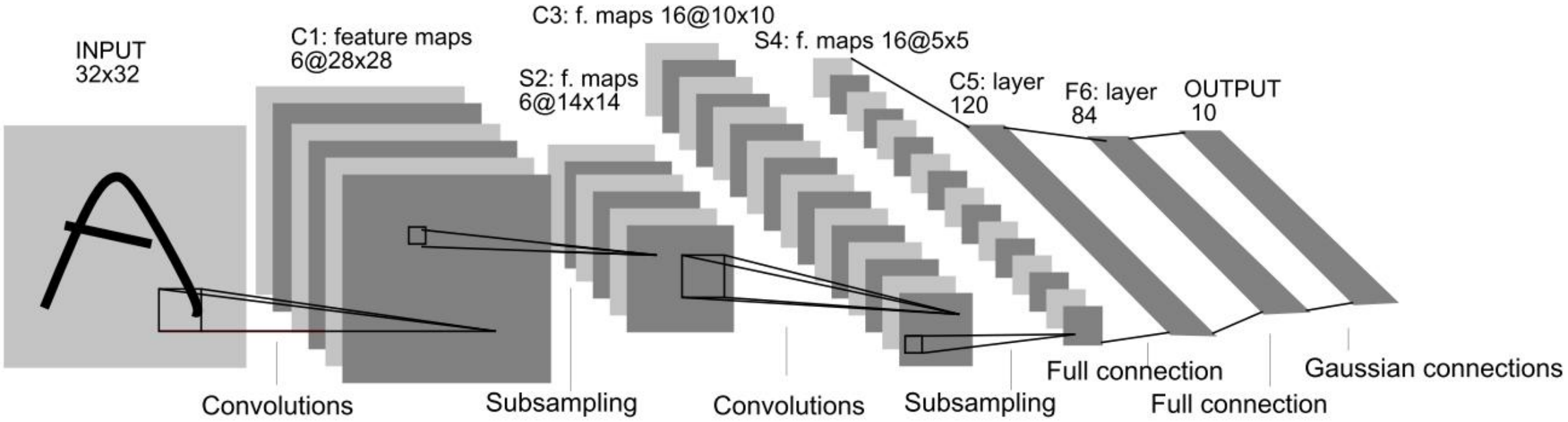
We assign the error to where it comes from

Examples of CNN Architecture

LeNet (1998)

- Gradient-based learning applied to document recognition

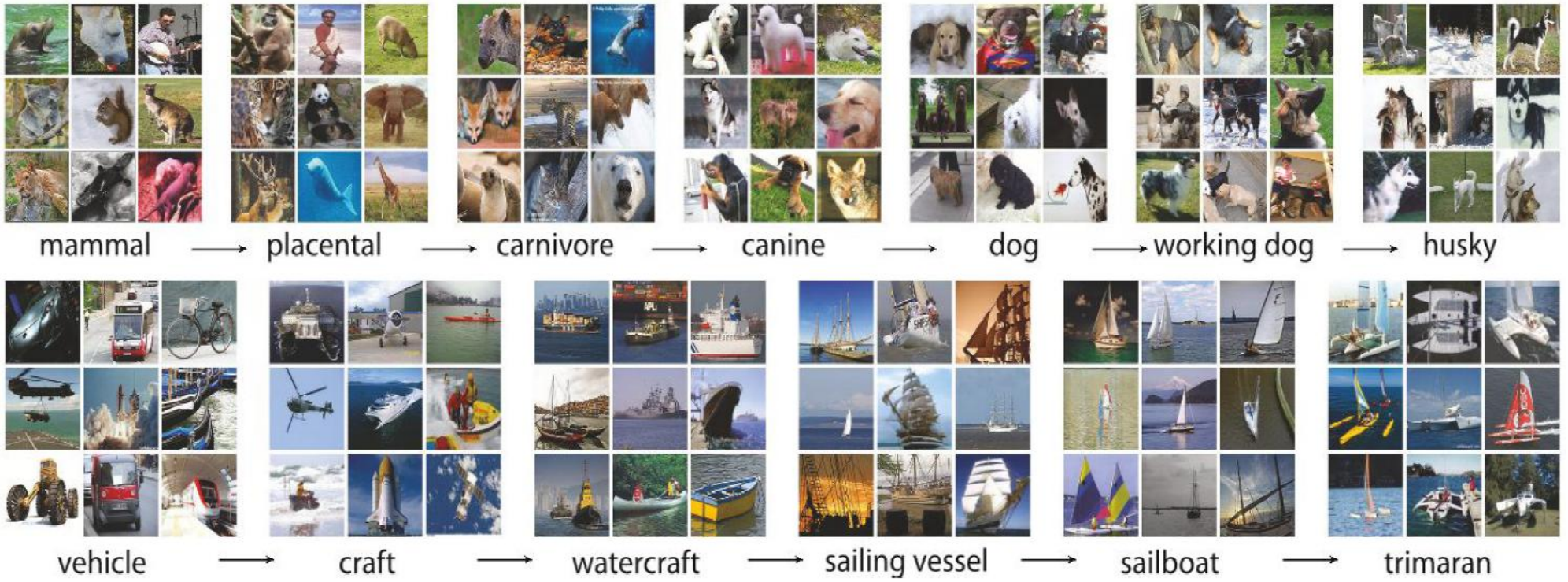
[Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner 1998]



LeNet-5 5 layers

ImageNet (Benchmark dataset)

- ImageNet
  - About 1.5×10^7 images, 2.2×10^4 categories
  - An image database organized according to the WordNet hierarchy



J Deng, W Dong, R Socher, LJ Li, K Li, L Fei-Fei. ImageNet: A large-scale hierarchical image. *CVPR*, 2009.

ISVRC

Steel drum

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

✔

Output:
Scale
T-shirt
Giant panda
Drumstick
Mud turtle

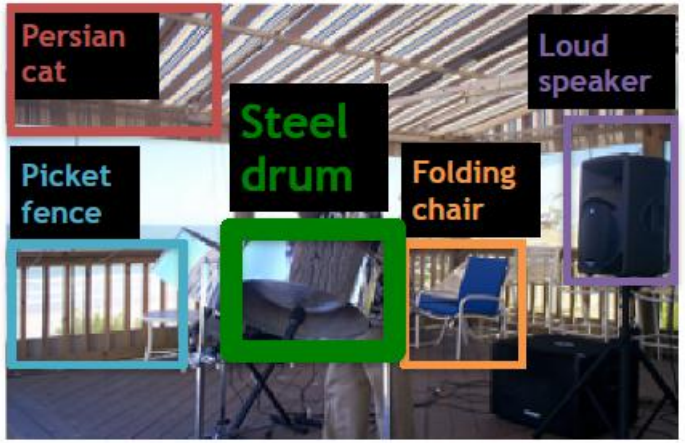
✗

Classification Task

Steel drum

✔

Output



Classification + Localization Task

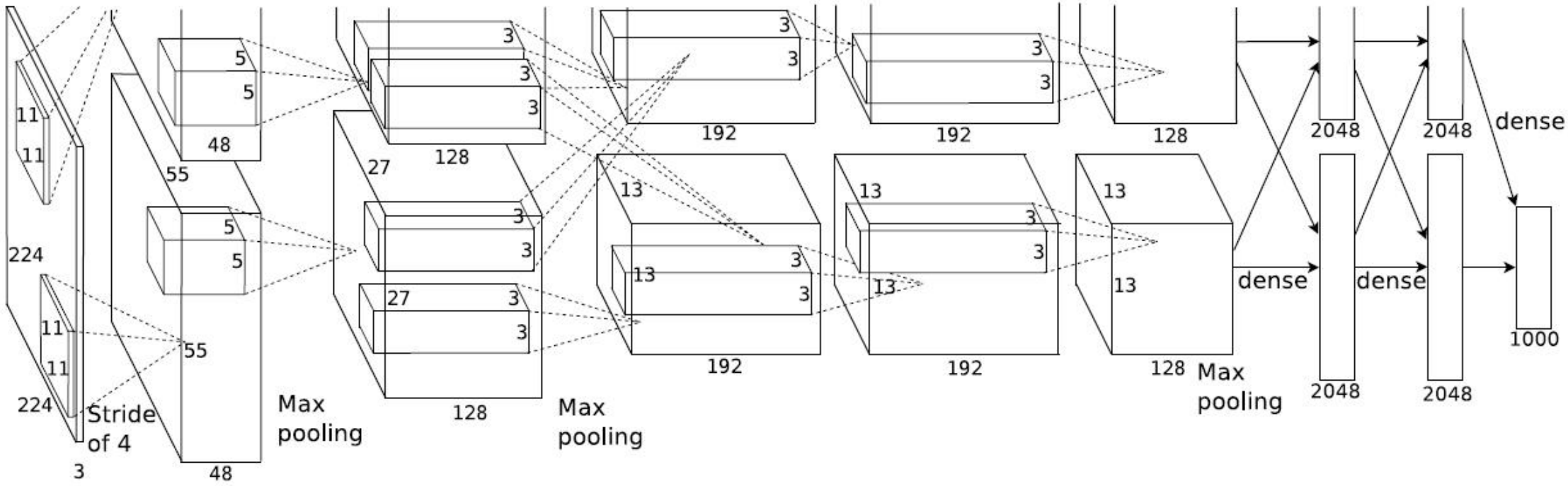
Deng et al. 2009, Russakovsky et al. 2015

Example credit: Fei-Fei Li and Jia Deng

AlexNet (2012)

- ImageNet Classification with Deep Convolutional Neural Networks

[Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton 2012]



AlexNet 8 layers Top-5 Error rate: 16.4%

VGGNet (2014)

- Very Deep Convolutional Networks for Large-Scale Image Recognition

[Karen Simonyan, and Andrew Zisserman 2014]

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 conv3-256 **conv1-256**	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**
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**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG-16 & VGG-19

16 layers (trainable)

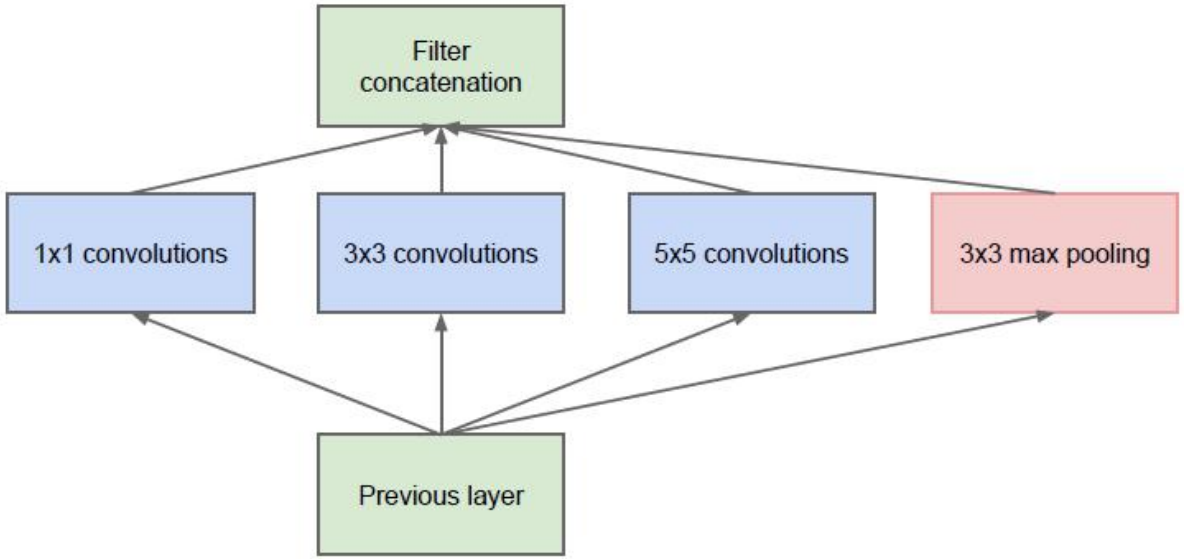
19 layers (trainable)

Top-5 Error rate: 7.3% (VGG-19)

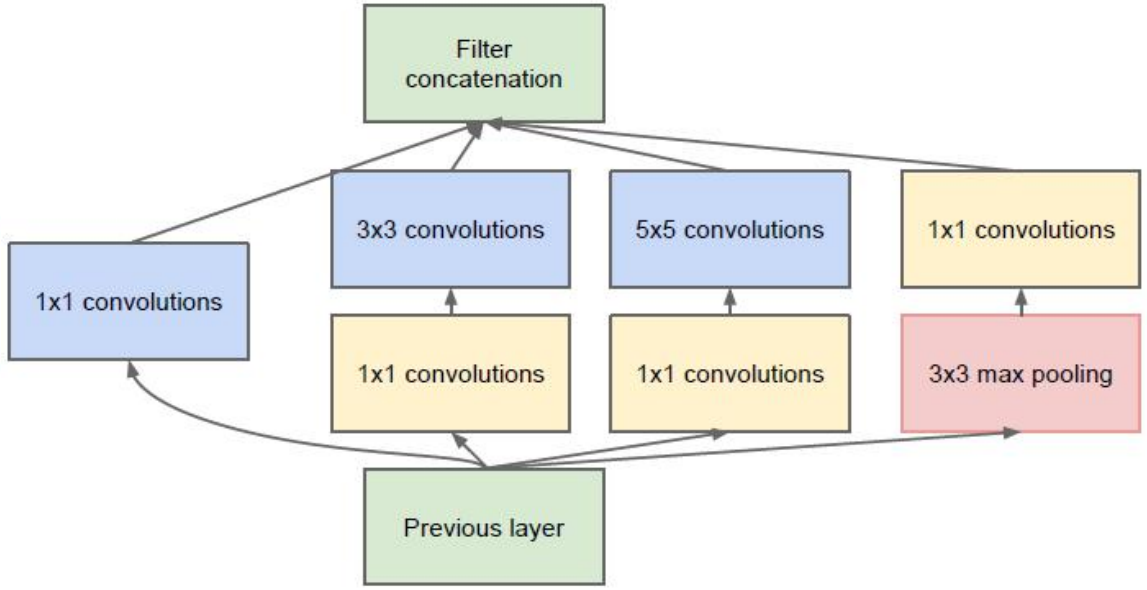
GoogLeNet (2014)

- Going Deeper with Convolutions

[Karen Simonyan, Andrew Zisserman 2014]



(a) Inception module, naïve version

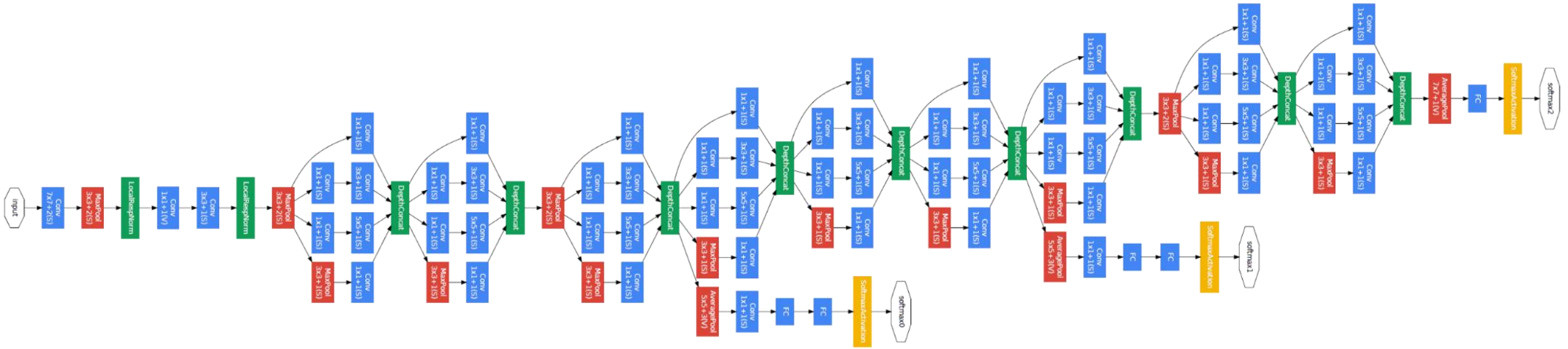


(b) Inception module with dimensionality reduction

GoogLeNet (2014)

- Going Deeper with Convolutions

[Karen Simonyan, Andrew Zisserman 2014]



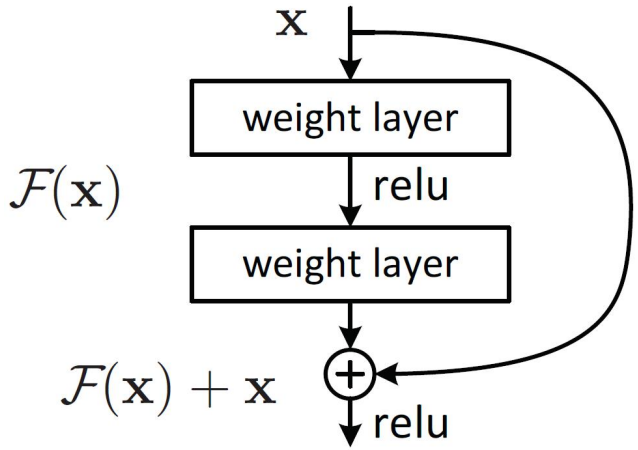
GoogLeNet

Only 3 layers??? 22 layers!!! Top-5 Error rate: 6.7%

ResNet

- Deep Residual Learning for Image Recognition

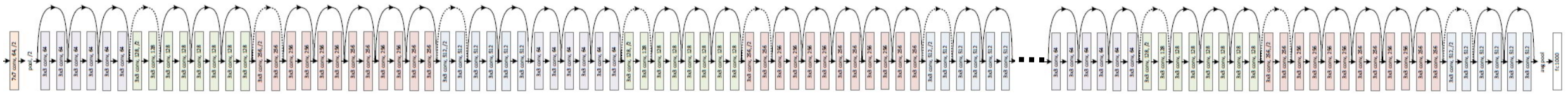
[Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun 2015]



x

identity

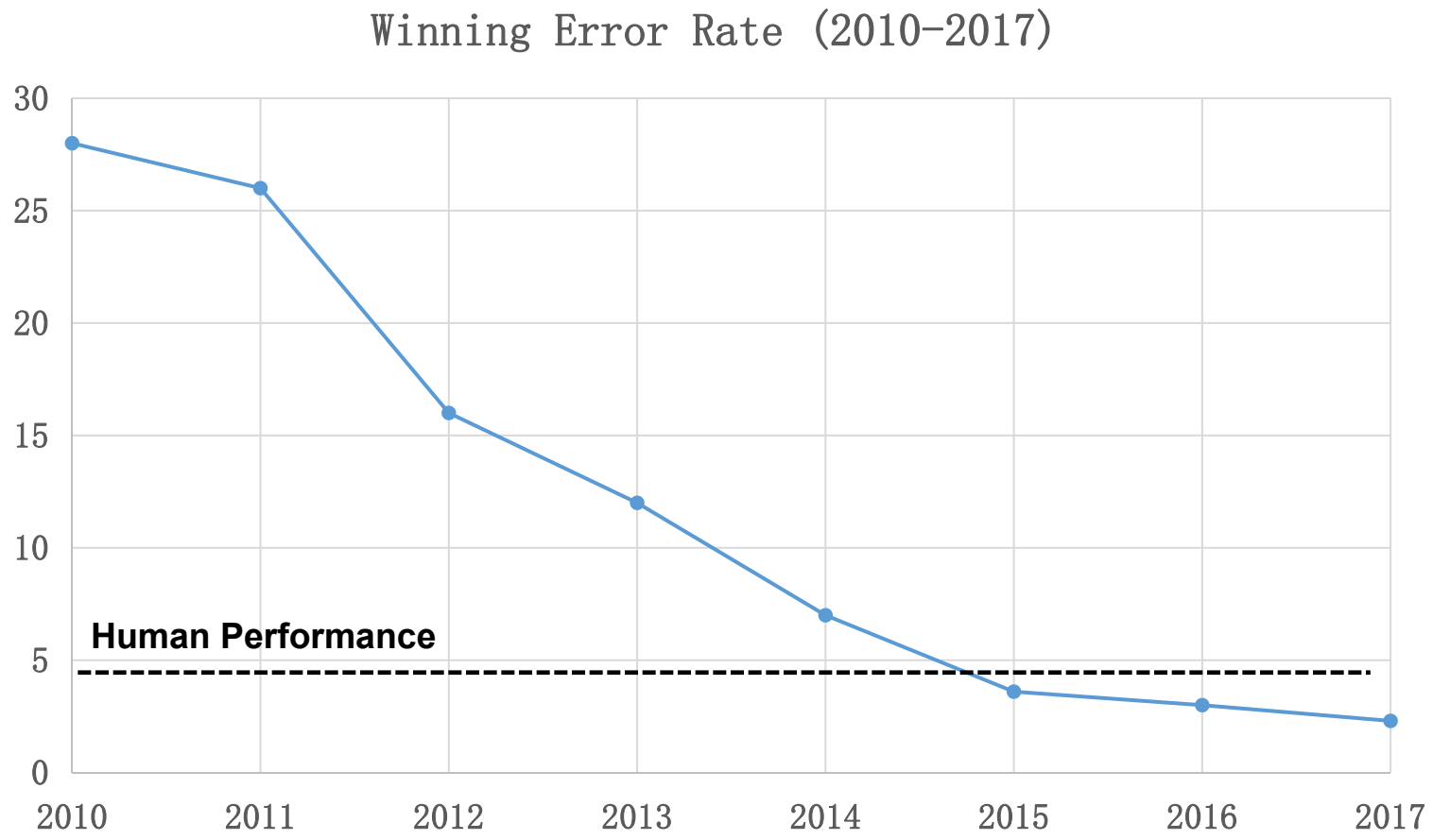
Residual learning: a building block



ResNet

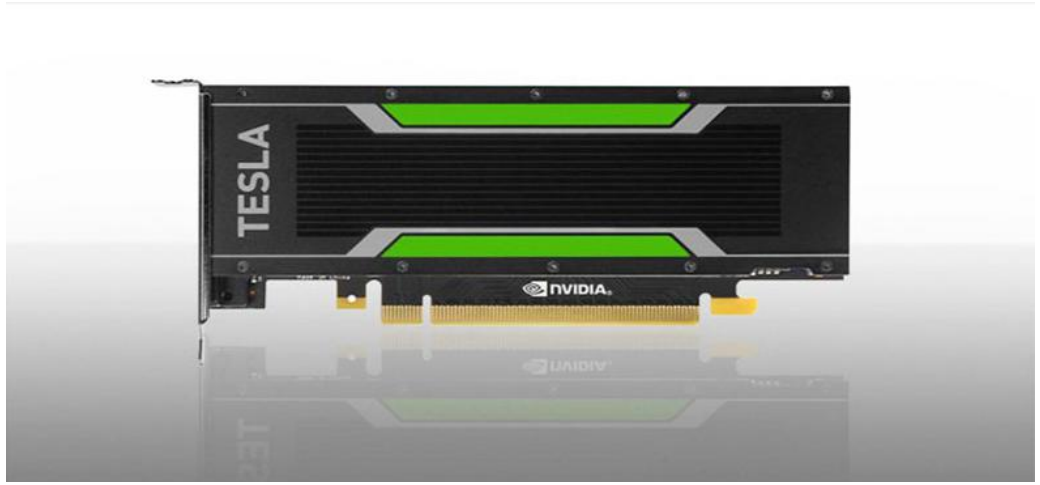
Only 2 layers??? Of course NOT!!! 152 layers!!! Top-5 Error rate: 3.57%

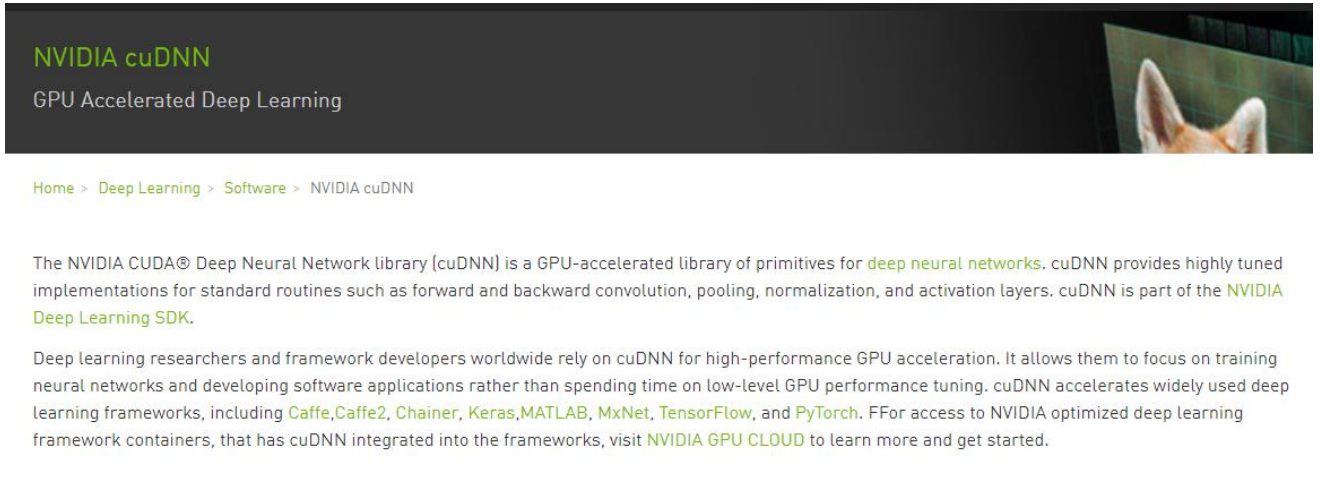
ResNet


Winning Error Rate (2010-2017)
30
25
20
15
10
5
0
Human Performance
2010
2011
2012
2013
2014
2015
2016
2017

Hardware

- Trend
  - Deeper than deeper
  - Need a great number of computation resources





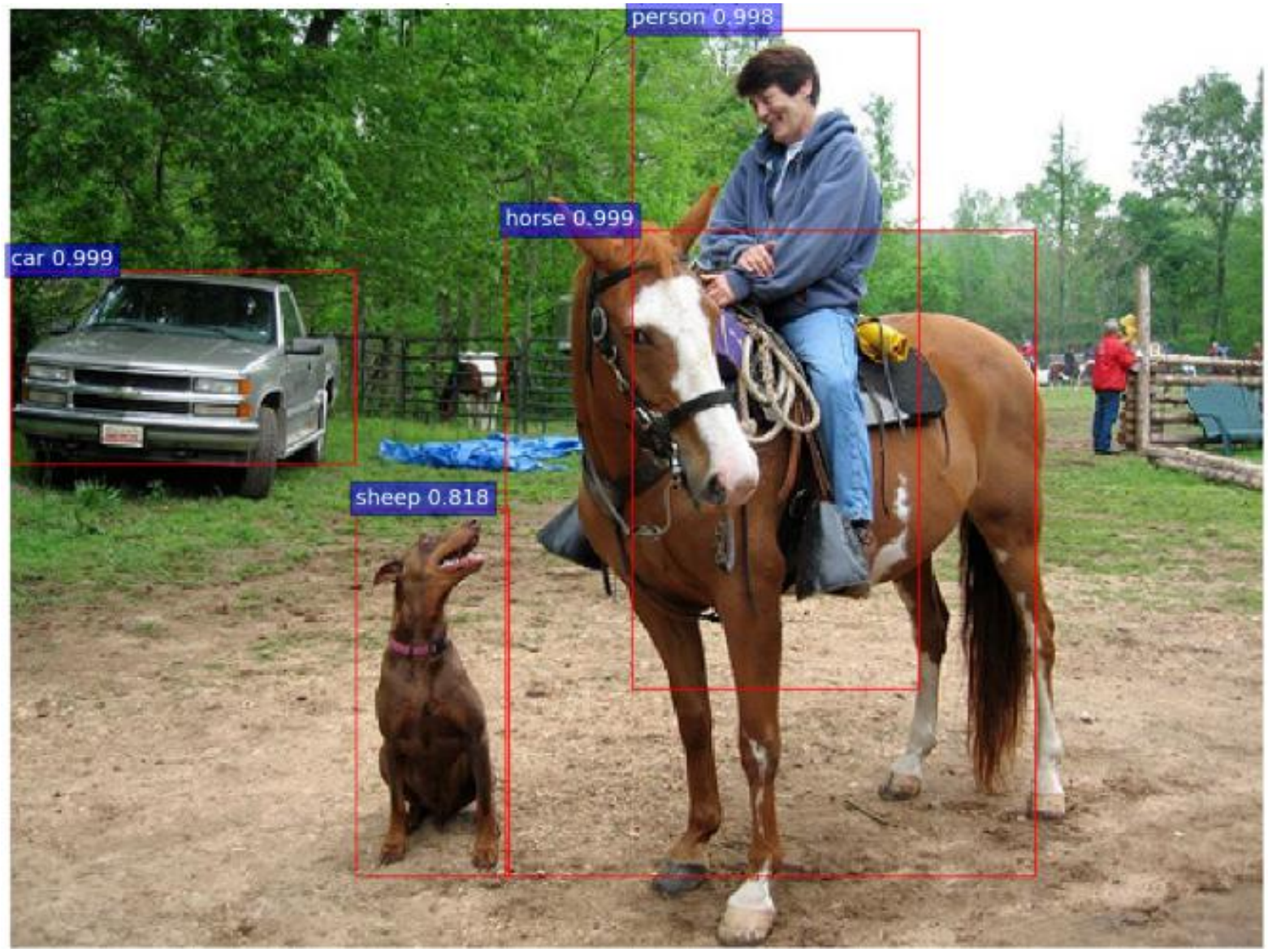
Hardware

Thanks to NVIDIA!

Software

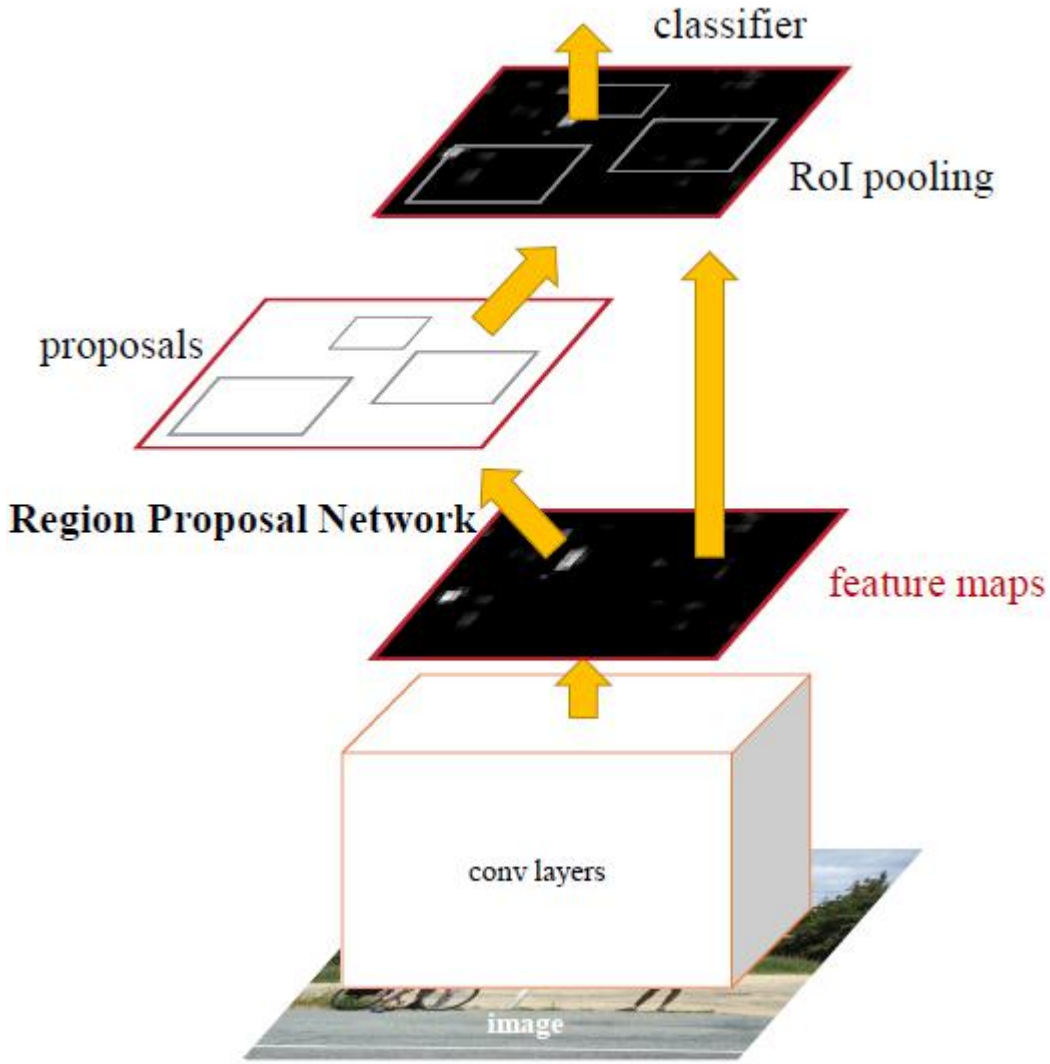
More Applications

Applications – Object Detection



Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. NIPS 2015.

Applications – Object Detection

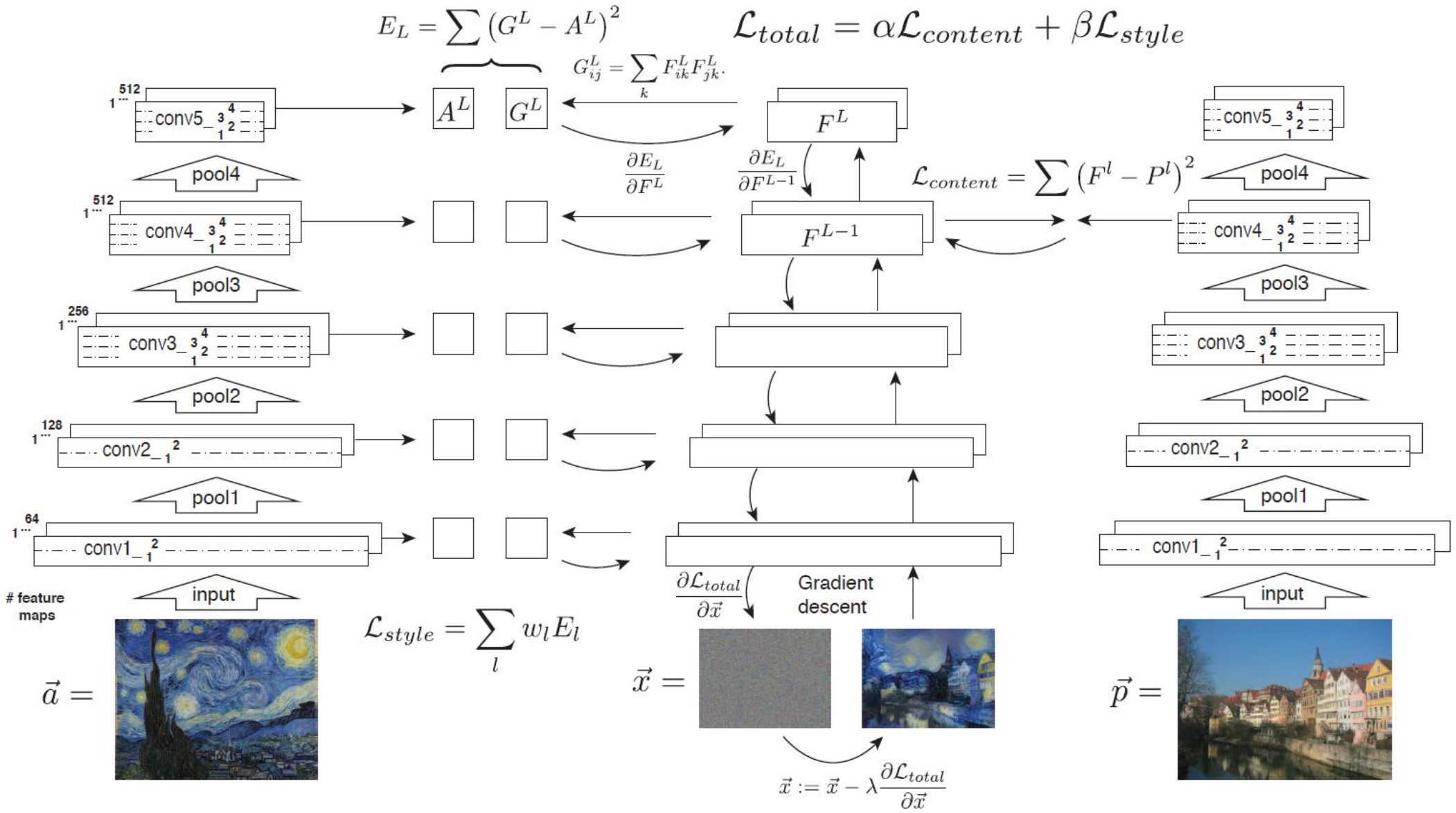


Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. NIPS 2015.

Applications – Style Transfer

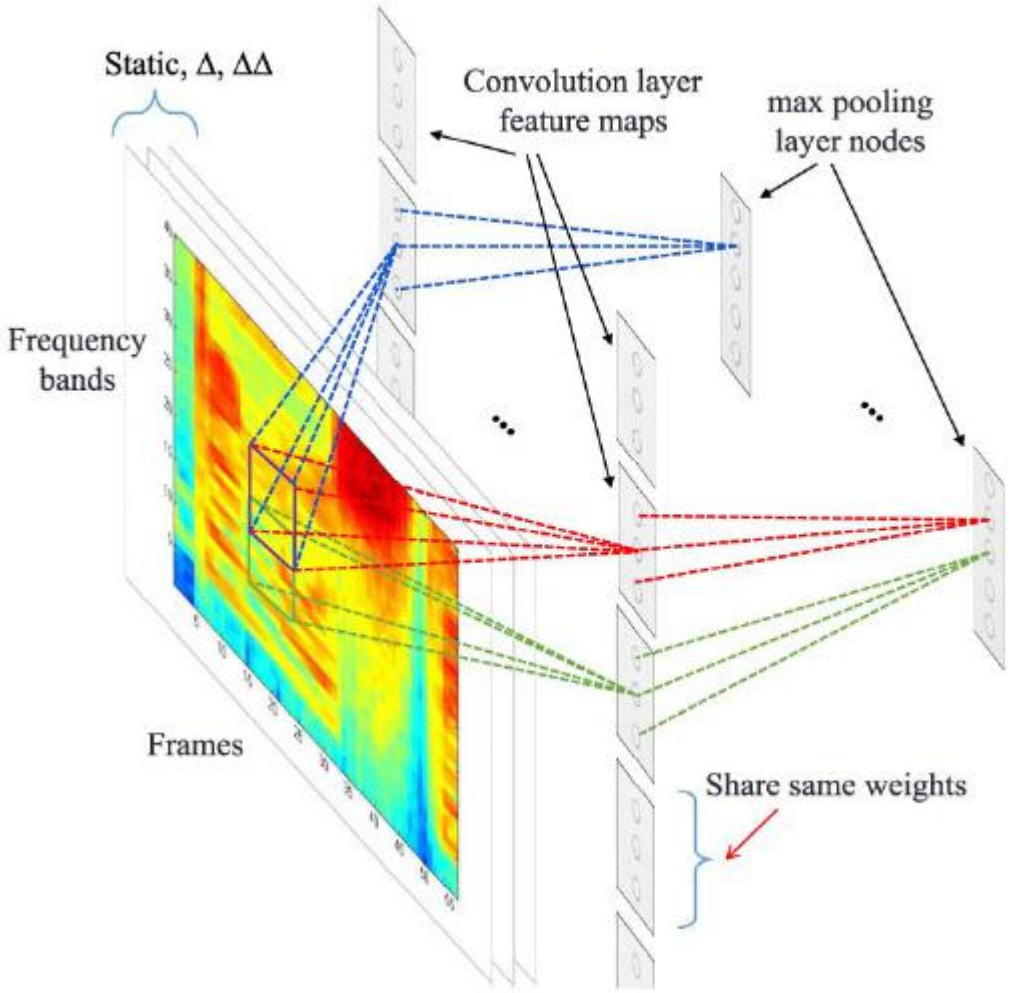
Gatys L A, Ecker A S, Bethge M. Image Style Transfer Using Convolutional Neural Networks. CVPR, 2016.

Applications – Style Transfer



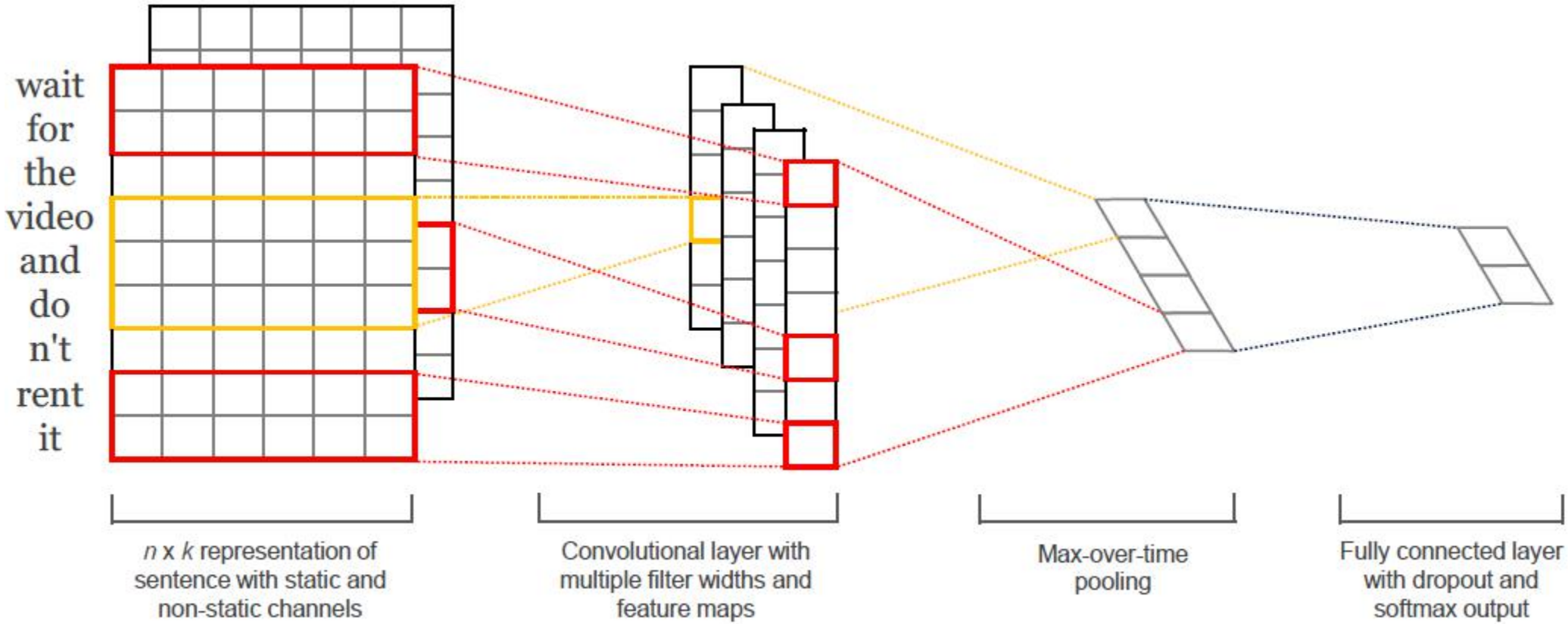
Gatys L A, Ecker A S, Bethge M. Image Style Transfer Using Convolutional Neural Networks. CVPR, 2016.

Applications – Speech Recognition



Abdel-Hamid O, Mohamed A R, Jiang H, et al. Convolutional Neural Networks for Speech Recognition[J]. IEEE/ACM Transactions on Audio Speech & Language Processing, 2014,

Applications – Text Classification



Yoon Kim. Convolutional Neural Networks for Sentence Classification. EMNLP, 2014.

Materials

- Paper: Gradient-based Learning Applied to Document Recognition

  Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, 1998.

- Paper: Deep Learning

  Y. Lecun, Y. Bengio, and G. Hinton, *Nature*, 2015.

- Course: Convolutional Neural Networks for Visual Recognition

  http://cs231n.stanford.edu/

- Tool: CNN Visualization

  https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html

Questions