Convolutional Networks

ADVANCED ARTIFICIAL INTELLIGENCE

JUCHEOL MOON

1

Convolutional networks

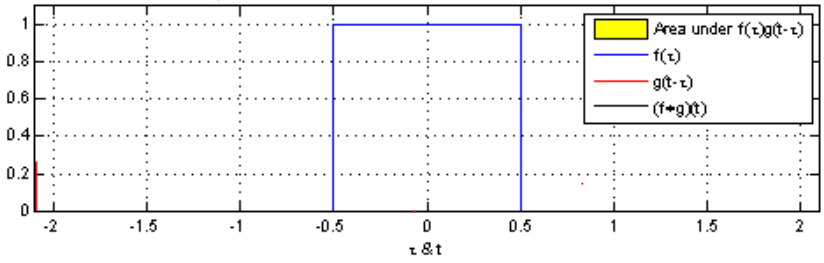
- Convolutional networks also known as convolutional neural networks (CNN), are a specialized kind of neural network for processing data with grid-like topology.
  - Time-series data, which can be thought of as a 1D grid taking samples at regular time intervals
  - Image data, which can be thought of as a 2D grid of pixels.
- The name "convolutional neural network" indicates that the network employs a mathematical operation called **convolution.**

2

2

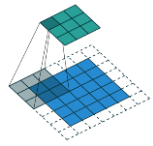
The Convolution Operation

- The convolution of f and g is written $f * g$
  - $(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$
  - f: input
  - g: kernel
  - $(f * g)$: feature map



3

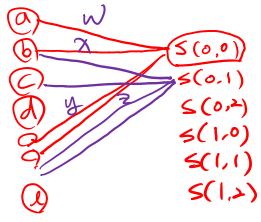
The Convolution Operation

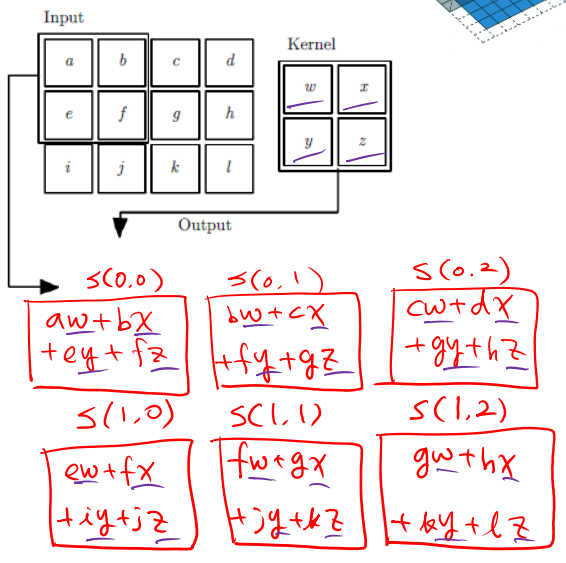


- Discretized two-dimensional
  - $S(i,j) = (I * K)(i,j)$

$= \sum_m \sum_n I(i+m, j+n) K(m,n)$

- I: input
- K: kernel

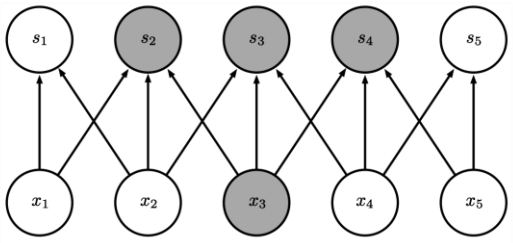




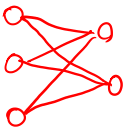
4

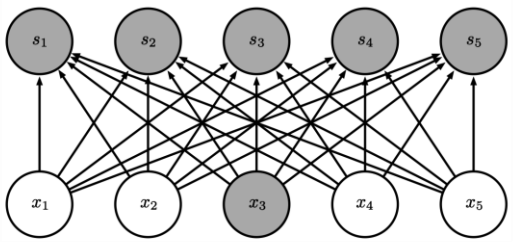
Sparse interactions

- This is accomplished by making the kernel smaller than the input



Sparse connections due to small convolution kernel



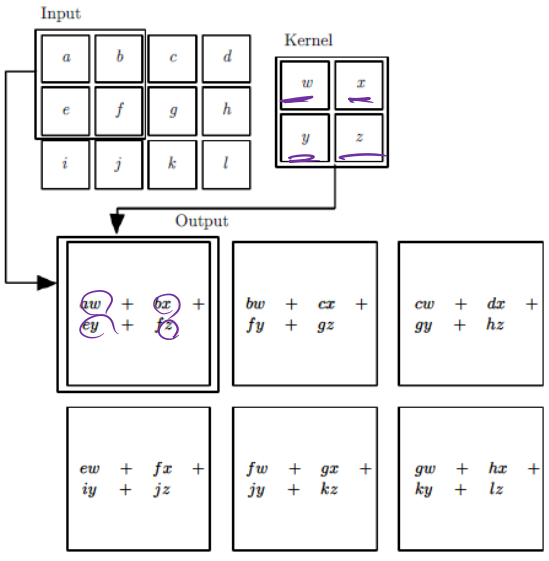


Dense connections

5

Parameter Sharing

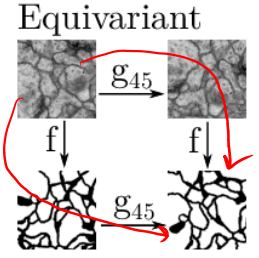
- Using the same parameter for more than one function in a model.
- Each member of the kernel is used at every position of the input

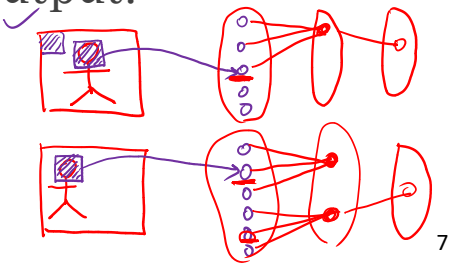


6

Equivariance

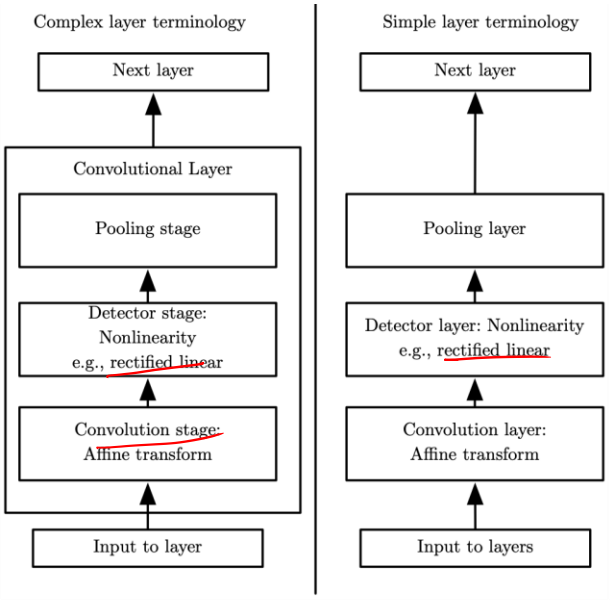
- A function f is equivariant to a function g
  - if $f(g(x)) = g(f(x))$
- With images, convolution creates a 2-D map of where certain features appear in the input.
- If we move the object in the input, its representation will move the same amount in the output.





7

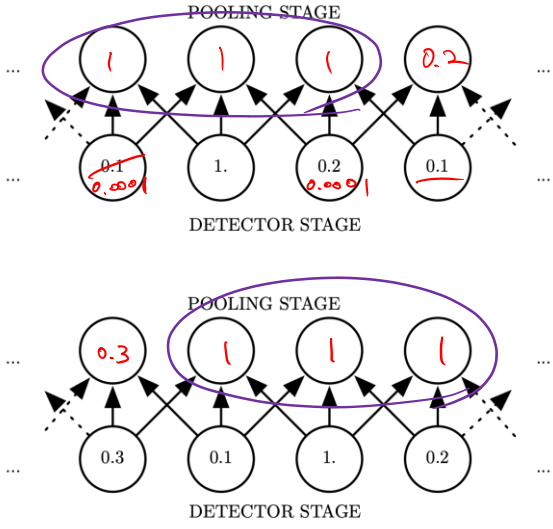
Convolutional Network Components



8

Pooling

- Pooling helps to make the representation become approximately **invariant** to small translations of the input
- Invariance to local translation can be a very useful property if we care more about whether some feature is present than exactly where it is

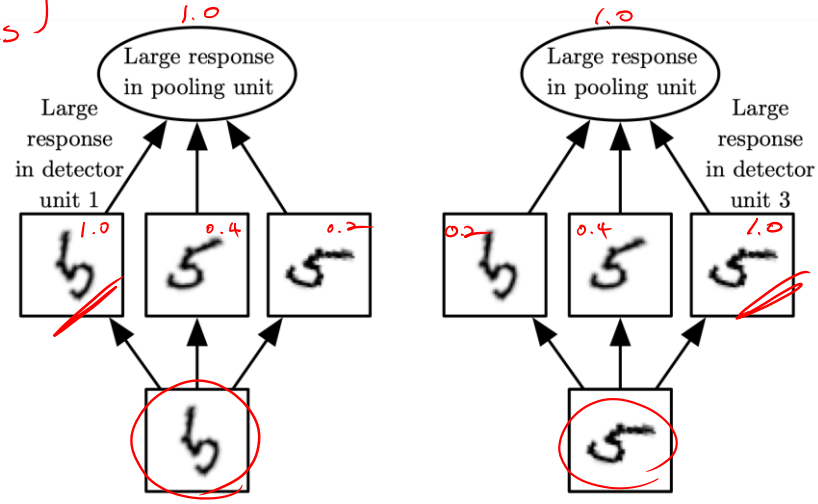


9

9

Invariance to Learned Transformations

- Example of learned invariances
  - All filters (kernels) are intended to detect a hand-written 5.

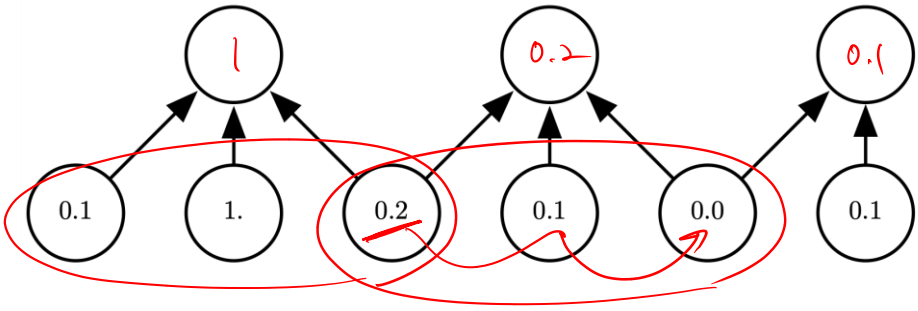


10

10

Pooling with Downsampling

- Max-pooling with a pool width of three and a stride between pools of two
  - This reduces the representation size by a factor of two, which reduces the computational and statistical burden on the next layer

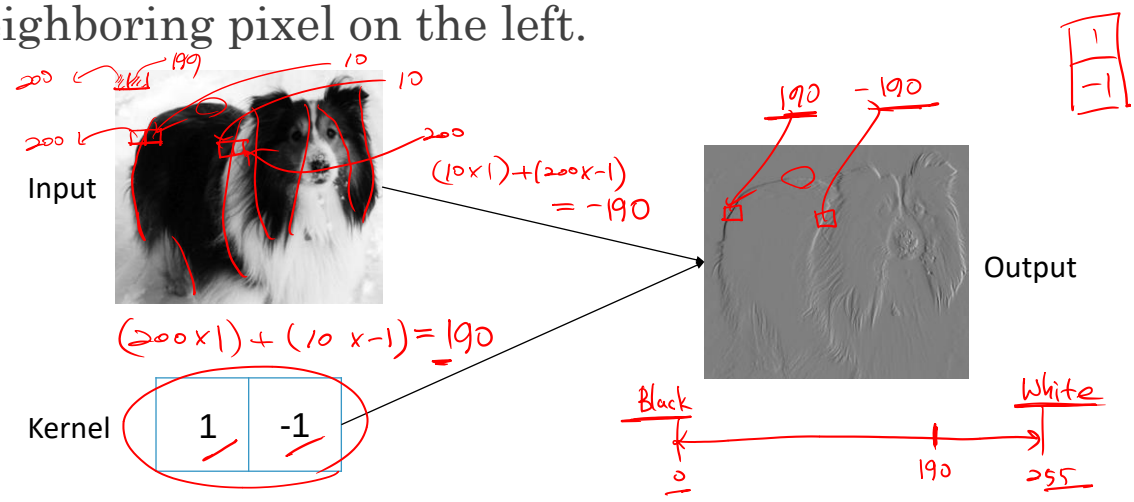


11

11

Edge Detection by Convolution

- The image on the right was formed by taking each pixel in the original image and subtracting the value of its neighboring pixel on the left.

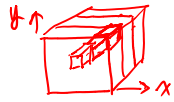


12

12

Variants of the Basic Convolution Function

- Convolution with a single kernel can only extract one kind of feature
- The input is usually not just a grid of real values
  - A color image has a ___R, G, B___ intensity at each pixel
  - When working with images, we usually think of the input and output of the convolution as being 3-D ___tensors___, with one index into the different ___channels___ and two indices into the spatial ___coordinates___ of each channel.



13

Convolution with Stride

- We may want to skip over some positions of the kernel in order to reduce the computational cost
  - We can think of this as downsampling the output of the full convolution function

Input

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

Kernel

x	y
w	z

$(4\times4) \rightarrow (3\times3) \rightarrow \cdots (1\times1)$

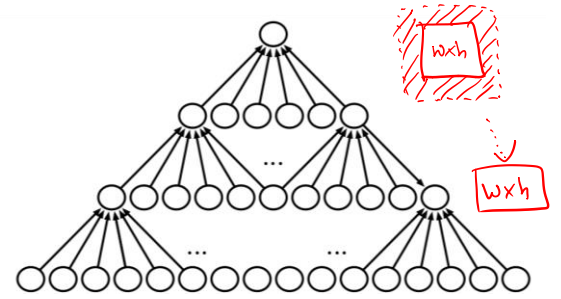
Output

$ax+by+ew+fz$	$cx+dy+gw+hz$
$ix+jy+mw+nz$	$kx+ly+ow+pz$

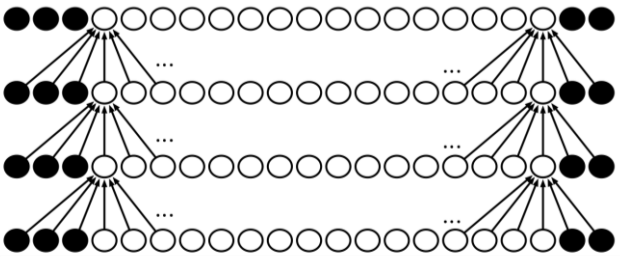
14

Zero Padding Controls Size

- Without zero padding, the width of the representation shrinks by one pixel less than the kernel width
- Zero padding the input allows us to control the kernel width and the size of the output independently



Without zero padding



With zero padding

15

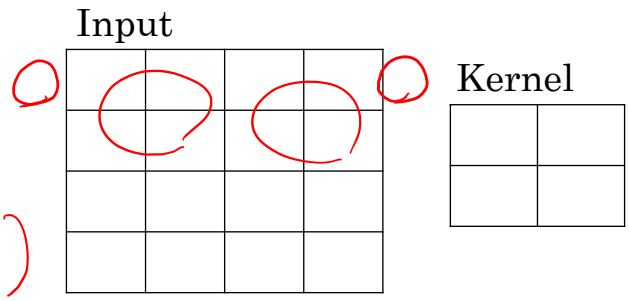
Valid and Same padding in keras

- "valid" means no padding
  - The size of the output shrinks at each layer
  - As layers are added, the spatial dimension of the network will eventually drop to 1 x 1
- "same" results in padding the input such that the output has the same length as the original input.
  - The network can contain as many convolutional layers as the available hardware can support

16

Dimensions

- Let padding P and stride S
- The dimension of an input
  - $H \times W$
- The dimension of a kernel (filter)
  - $FH \times FW$
- The dimension of an output
  - $OH = \left\lfloor \frac{H + 2P - FH}{S} \right\rfloor + 1$
  - $OW = \left\lfloor \frac{W + 2P - FW}{S} \right\rfloor + 1$

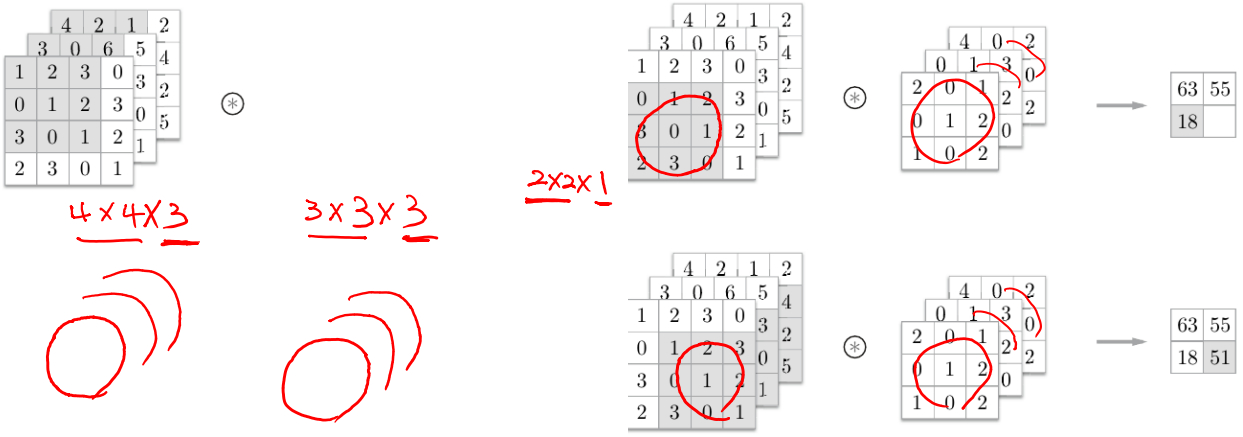


W=4 S=2 OW=2

17

Convolutions on multi-channel

- 3D tensor input and kernel
  - Assume $p = 0$ and $s = 1$



18

Convolutions on multi-channel

- 3D tensor input and kernel
  - Assume $p = 0$ and $s = 1$

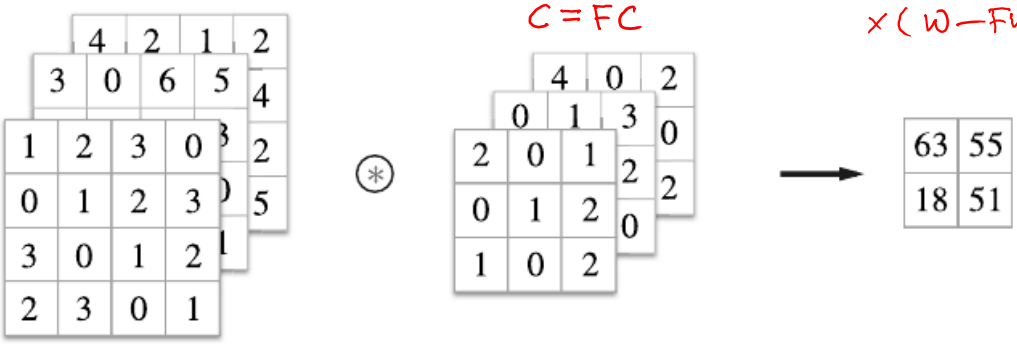
$OH = \left\lfloor \frac{H + 2p - FH}{s} \right\rfloor + 1$

Input
$H \times W \times C$

Kernel
$FH \times FW \times FC$

$C = FC$

Output
$(H - FH + 1) \times (W - FW + 1) \times 1$



19

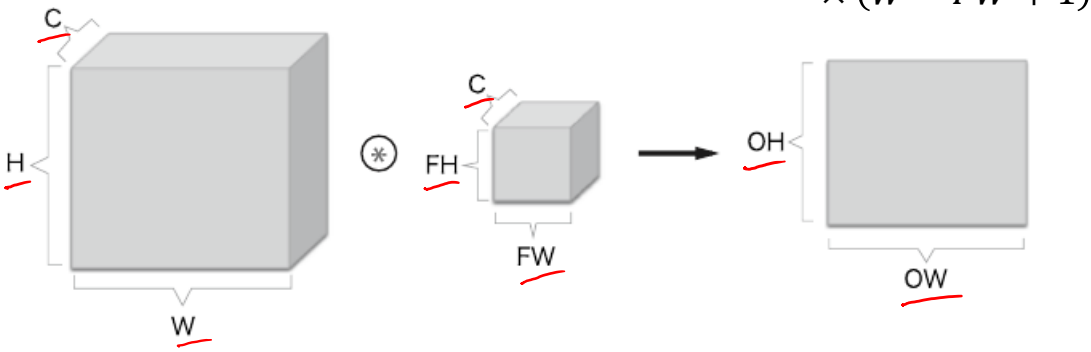
Convolutions on multi-channel

- 3D tensor input and kernel
  - Assume $p = 0$ and $s = 1$

Input
$H \times W \times C$

Kernel
$FH \times FW \times FC$

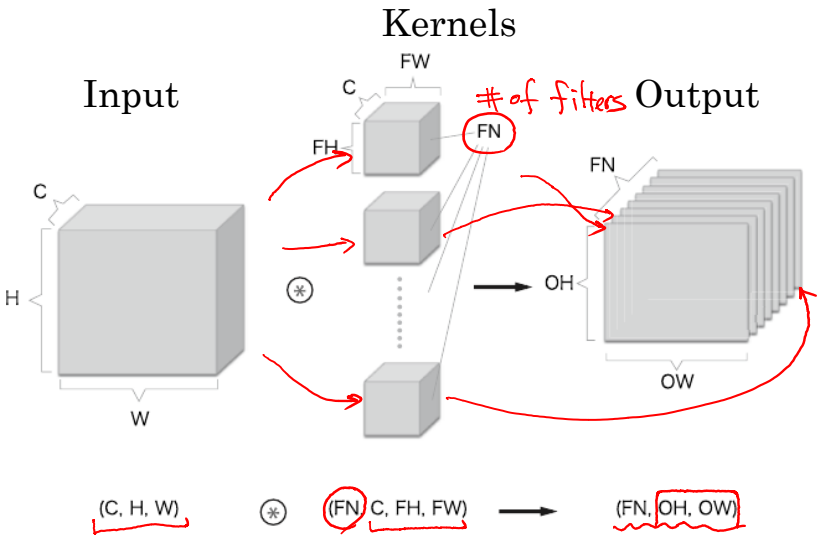
Output
$(H - FH + 1) \times (W - FW + 1)$



20

Convolutions on multi-channel / multi-kernels

- 3D tensor input and kernels (filters)

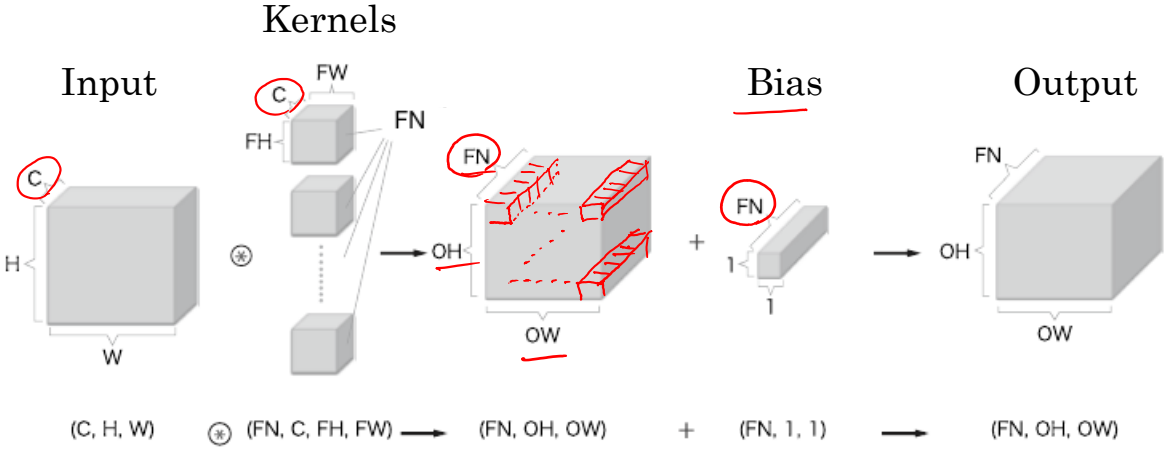


21

21

Convolutions on multi-channel / multi-kernels

- 3D tensor input and kernel (filter)

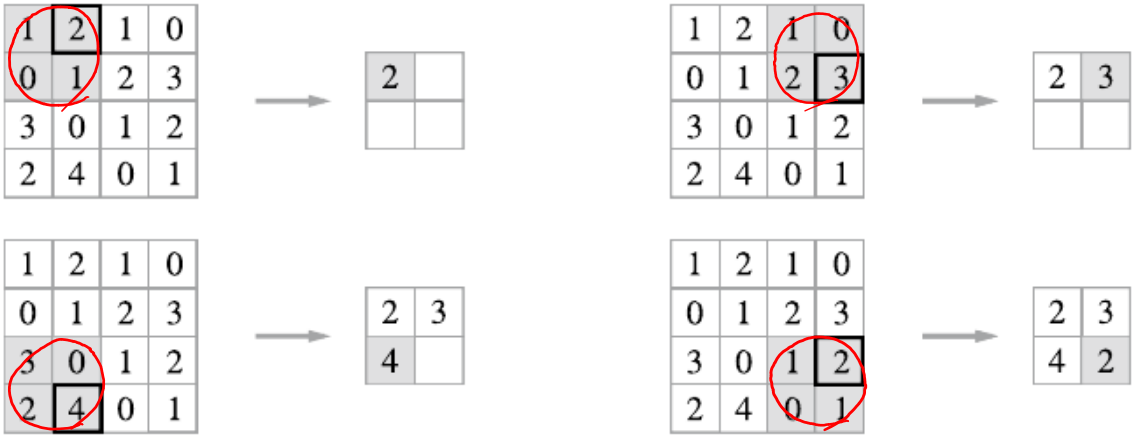


22

22

Max Pooling

- No parameters to be trained

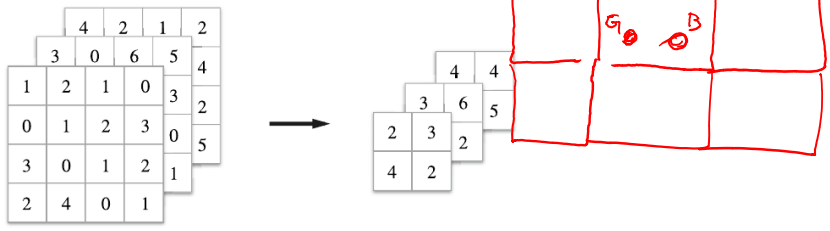


23

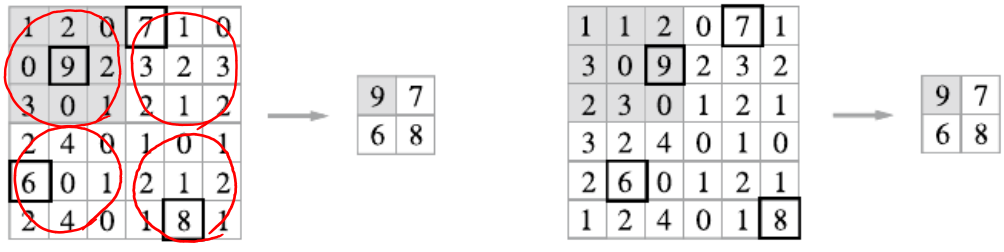
23

Max Pooling

- Same number of channels



- Invariant to small translations of the input

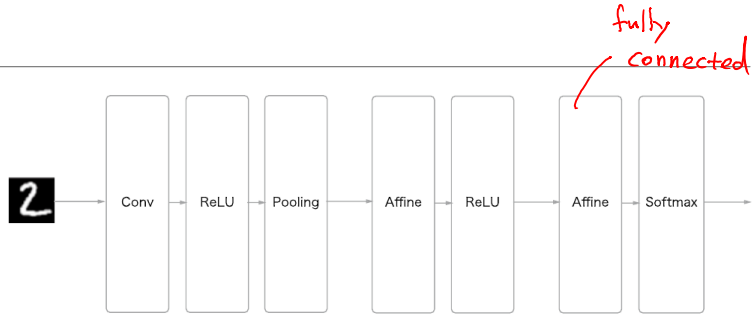


24

24

CNN example

- Design
  - Input: 28x28x1
  - Conv: 5x5x1x30
    - Padding: 0
    - Stride: 1
  - Pooling: 2x2x1x30
    - Stride: 2

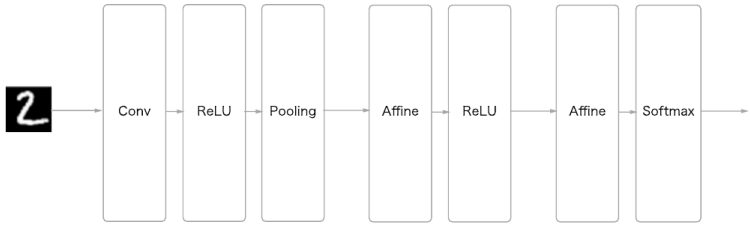


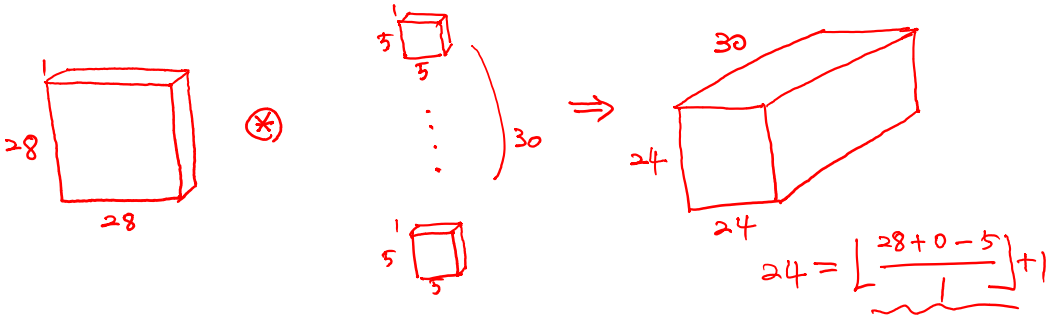
25

CNN example

$$OH = \left\lfloor \frac{H + 2P - FH}{S} \right\rfloor + 1$$

- Design
  - Input: 28 × 28 × 1
  - Conv: 5 × 5 × 1 × 30
    - Padding: 0
    - Stride: 1

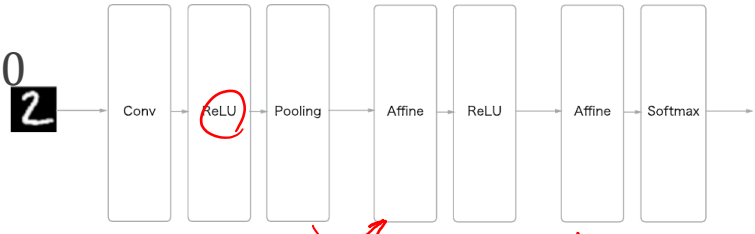


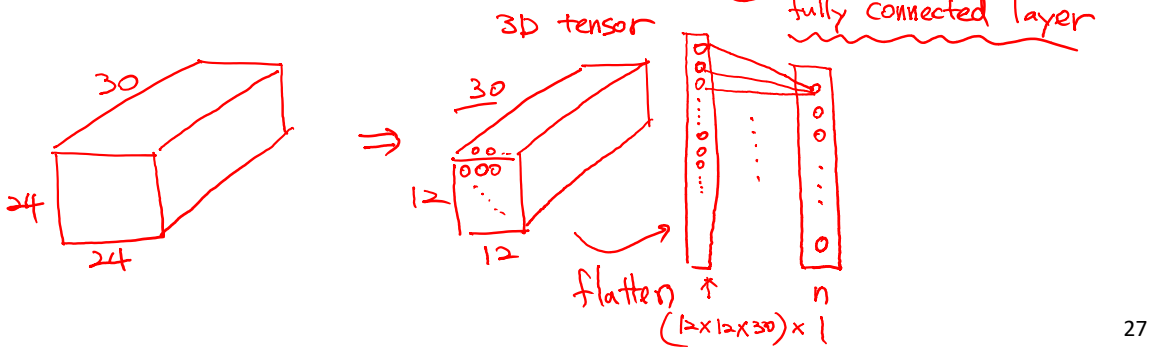


26

CNN example

- Design
  - Pooling: $2 \times 2 \times 1 \times 30$
    - Stride: 2



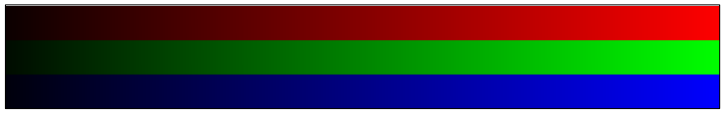


27

Bitmap

- Image pixels are generally stored with a variable number of bits per pixel which identify its color, the color depth.
  - Grayscale 8 bits
    - 0: Black, $2^8 - 1 = 255$: White

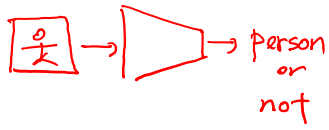
  - RGB 24 bits

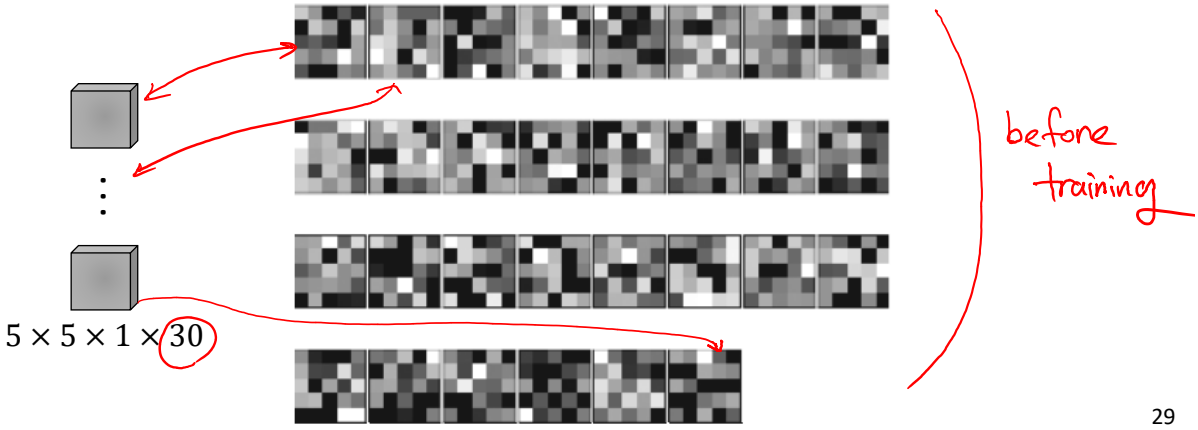


28

Visualize CNN

- Conv: $5 \times 5 \times 1 \times 30$
- Parameters: random initial numbers
  - Normalize between 0 ~ 255

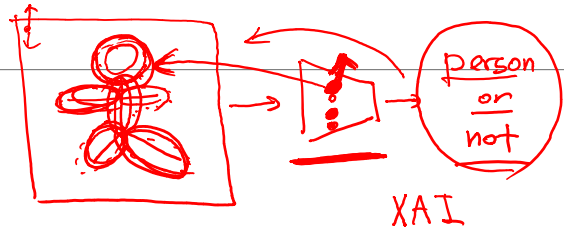


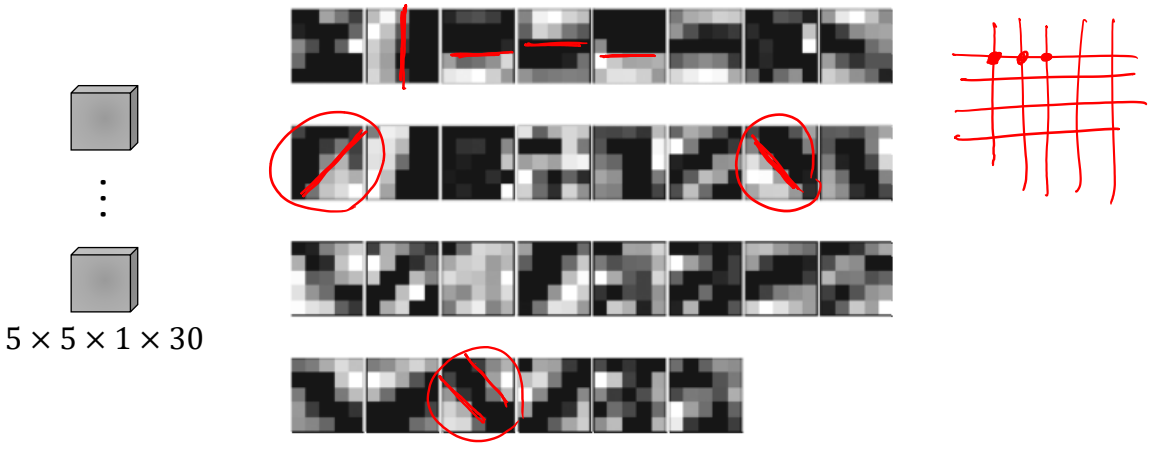


29

Visualize CNN

- Conv: $5 \times 5 \times 1 \times 30$
- Parameters: after training
  - Normalize between 0 ~ 255

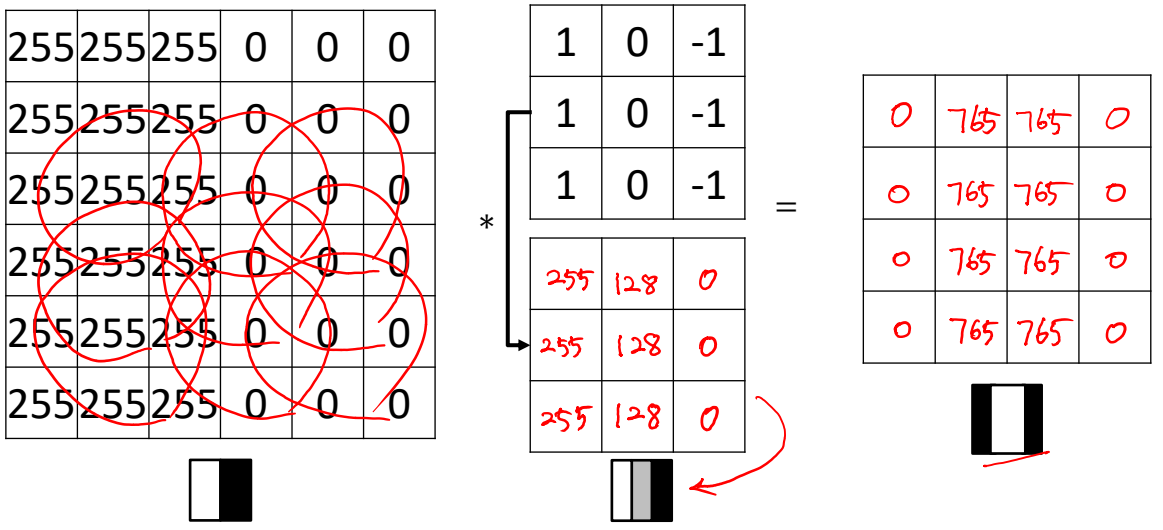




30

Vertical edge detection

- How CNN detect vertical edge?



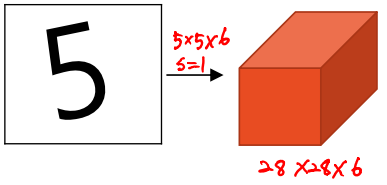
31

LeNet (1998)

- Input: $32 \times 32 \times 3$
- Conv1: $5 \times 5 \times 3 \times 6 \rightarrow 5 \times 5 \times 6$

$$OH = \left\lfloor \frac{H + 2P - FH}{S} \right\rfloor + 1$$

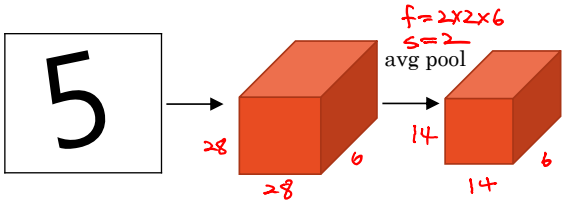
  - Stride: 1 , Padding: 0
  - Output: $28 \times 28 \times 6$



32

LeNet (1998)

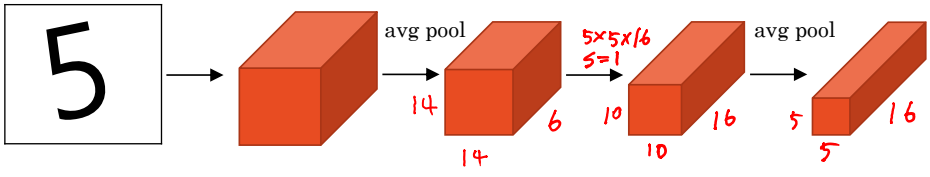
- Pool1: 2 x 2 x 6
  - Stride: 2
  - Output: 14 x 14 x 6



33

LeNet (1998)

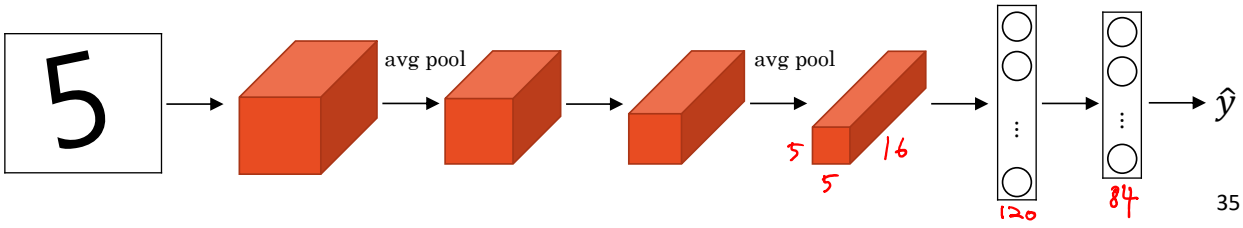
- Conv2: 5 x 5 x 16
  - Stride: 1 , Padding: 0
  - Output: 10 x 10 x 16
- Pool2: 5 x 5 x 16



34

LeNet (1998)

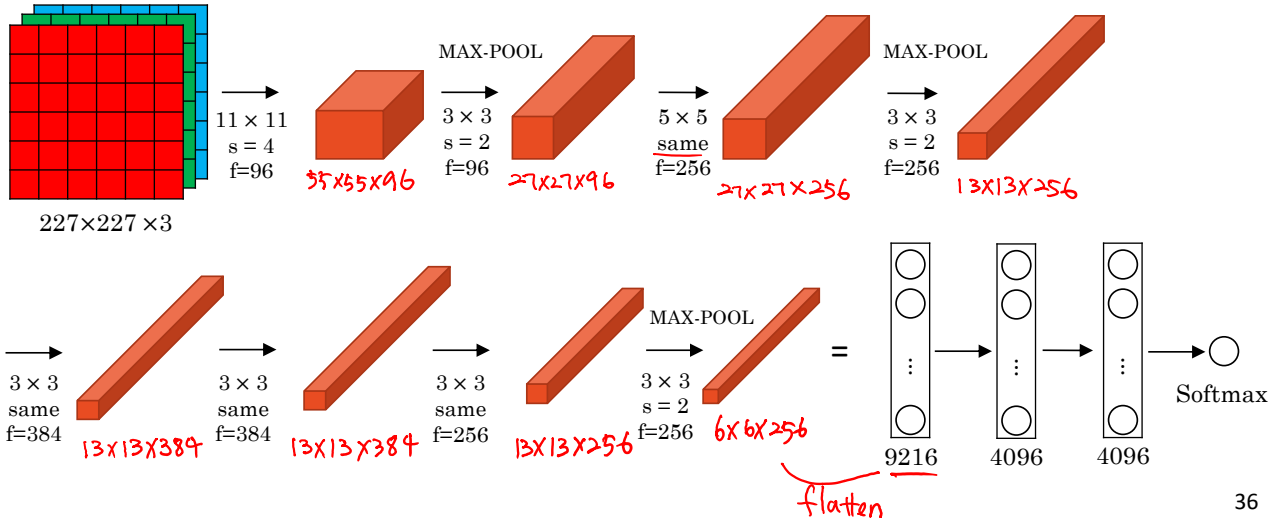
- Pool2: 5 × 5 × 16
- H3: 120
- H4: 84
- Softmax



35

AlexNet (2012)

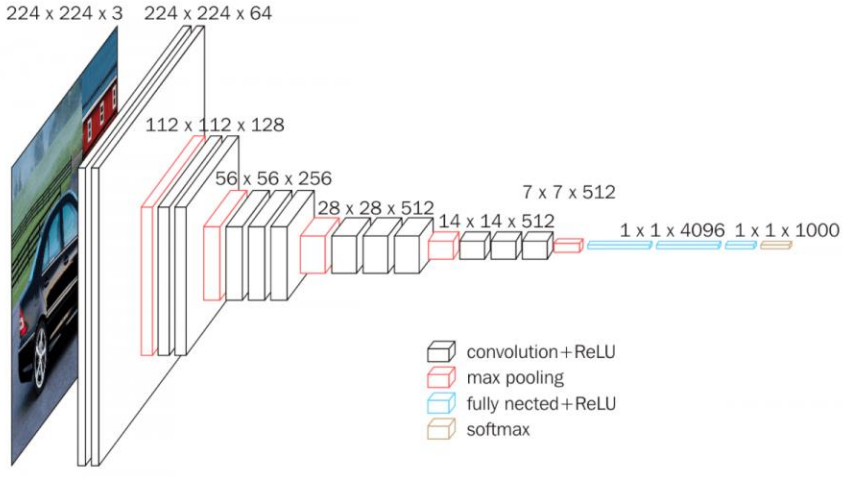
- Upgrade version of LeNet



36

VGG-16 (2015)

```
model.add(Conv2D(filters=64,kernel_size=(3,3),padding="same", activation="relu"))
model.add(Conv2D(filters=64,kernel_size=(3,3),padding="same", activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
```

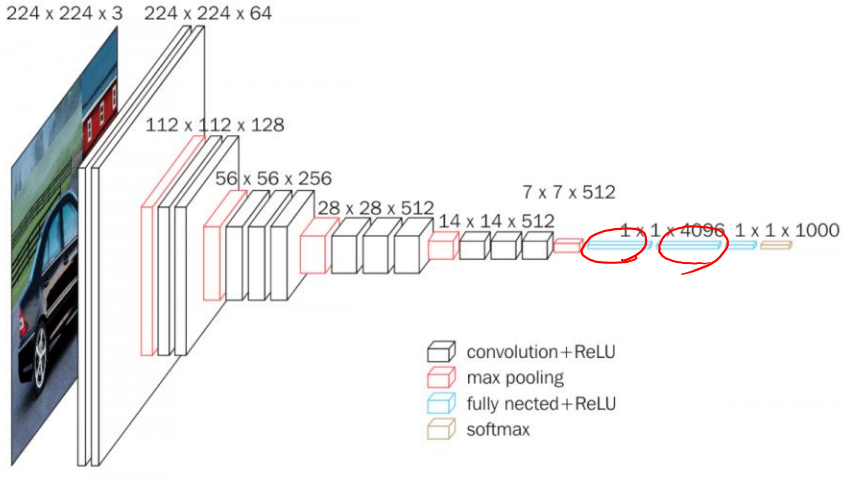


37

37

VGG-16 (2015)

```
model.add(Flatten())
model.add(Dense(units=4096,activation="relu"))
model.add(Dense(units=4096,activation="relu"))
```

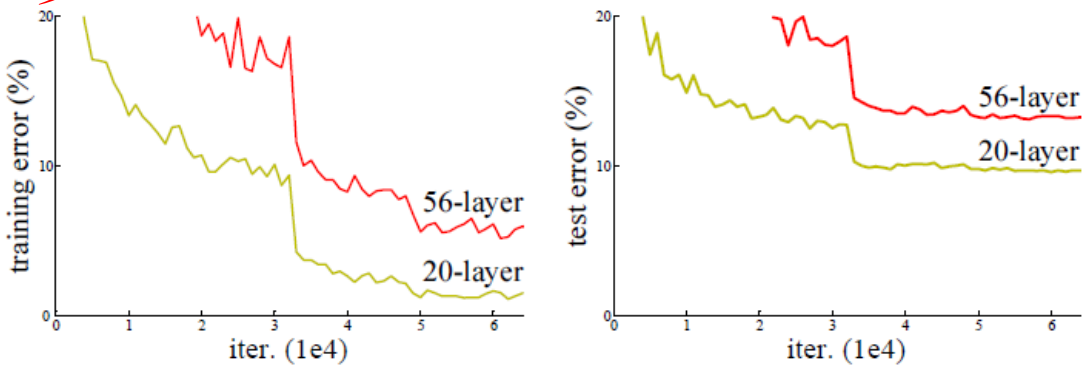


38

38

Deep Residual Learning

- Deeper neural networks are better in general?
  - Adding more layers to a suitably deep model leads to **(higher/lower)** training and test error

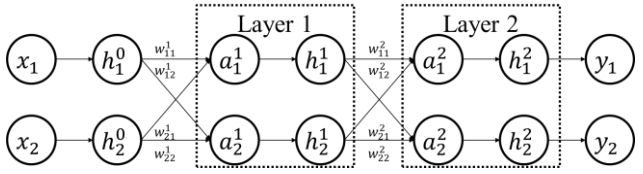


43

43

Deep Neural Networks

- Can a DNN learn an identity function? $\vec{y} = \vec{x}$
  - Yes, but it's not easy.
  - $y_1 = x_1$



$= h_1^2 = g(a_1^2)$

$\hookrightarrow w_{11}^2 h_1^1 + w_{21}^2 h_2^1$

$g(a_1^1)$ $g(a_2^1)$ $w_{21}^1 h_1^0 + w_{22}^1 h_2^0$

$\llcorner w_{11}^1 h_1^0 + w_{21}^1 h_2^0$

x_1 x_2 x_1 x_2

$= g\left(w_{11}^2 g(w_{11}^1 x_1 + w_{21}^1 x_2) + w_{21}^2 g(w_{21}^1 x_1 + w_{22}^1 x_2)\right) = x_1$

44

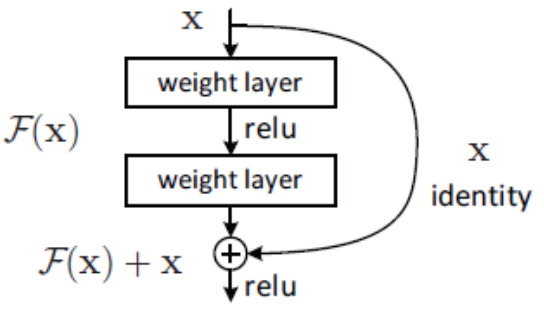
44

Deep Residual Network

- Shortcut Connections
  - $a^k =$ $w^k h^{k-1} + b^{k-1}$
  - Plain block
    - $h^k =$ $g(a^k)$
  - Residual block
    - $h^k =$ $g(a^k + h^{k-1})$

$= g((w^k h^{k-1} + b^{k-1}) + h^{k-1})$

$w \to 0, b \to 0$

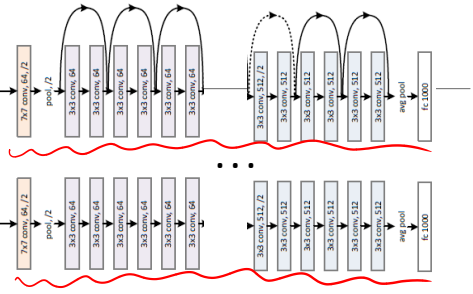


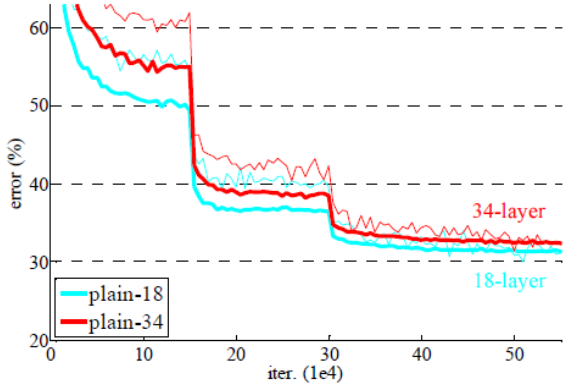
45

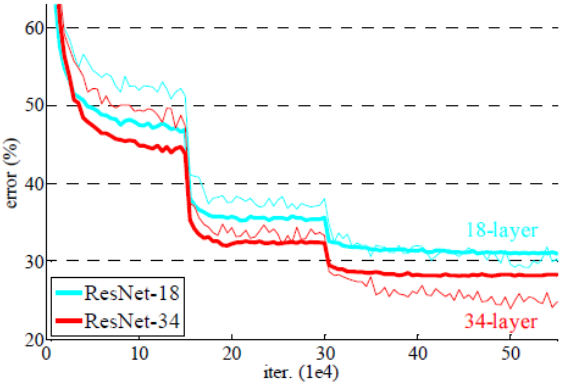
45

Deep Residual Network

- Experiments
  - Plain net of 18 and 34
  - Residual net of 18 and 34







46

46

ILSVRC

- The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) evaluates algorithms for object detection and image classification at large scale.

Siberian husky

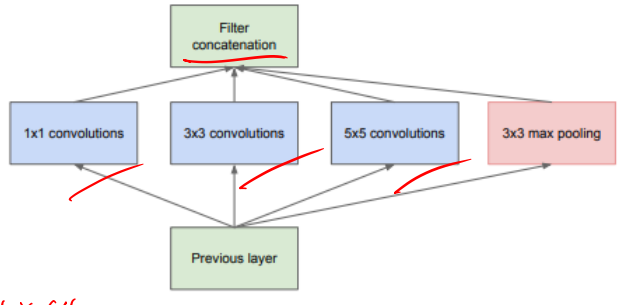
Eskimo dog

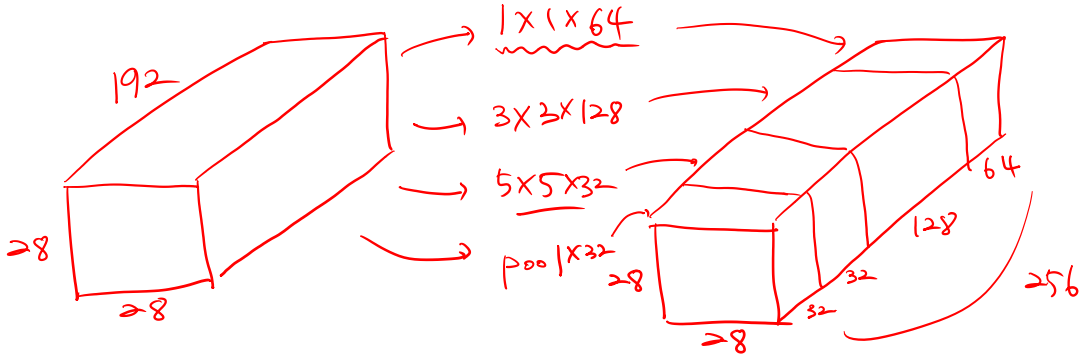
Year	Error
2012	16.4%
2013	11.7%
2014	6.67%

47

GoogLeNet

- Inception architecture

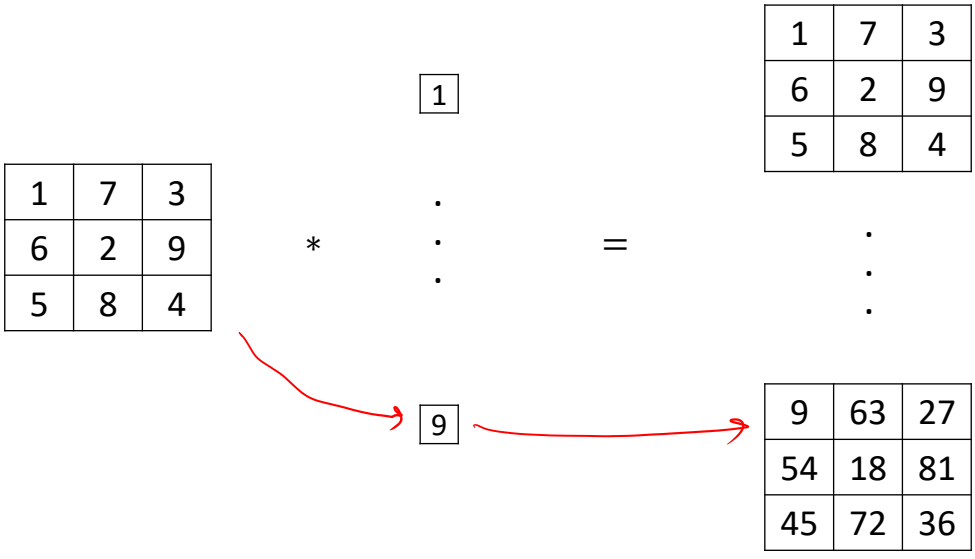




48

1 × 1 Convolution

- Meaning of using 1 × 1 convolution

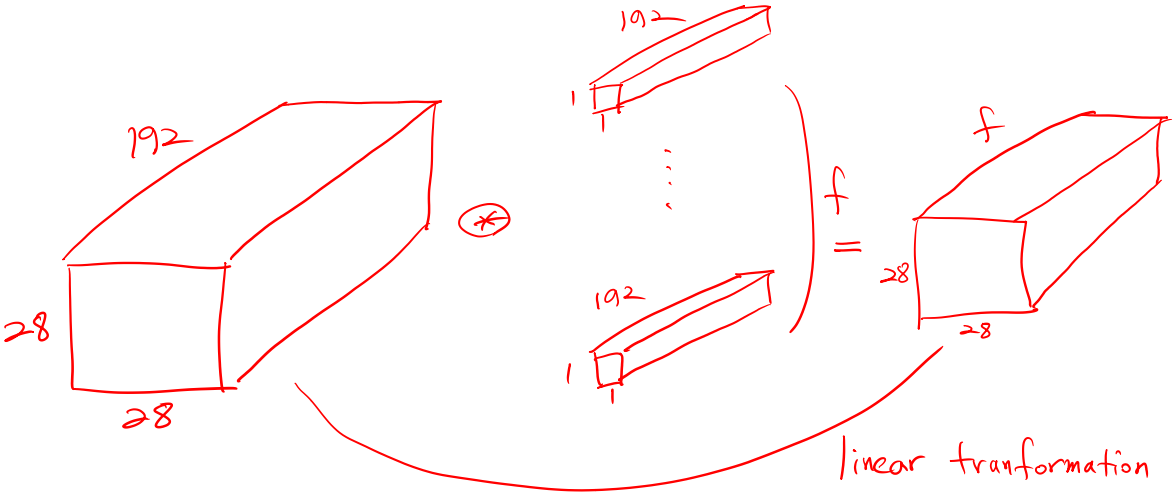


49

49

1 × 1 Convolution

- Depth reduction

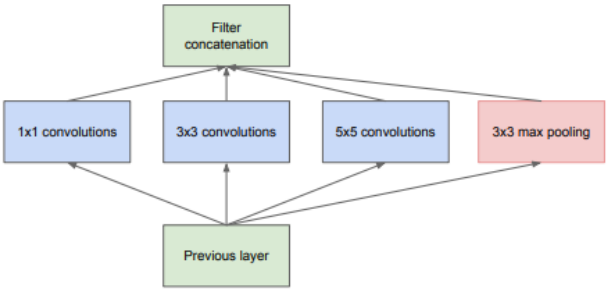


50

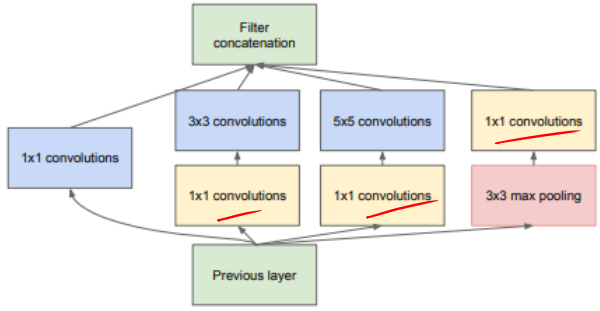
50

The problem of inception network

- Computational cost



Naïve version



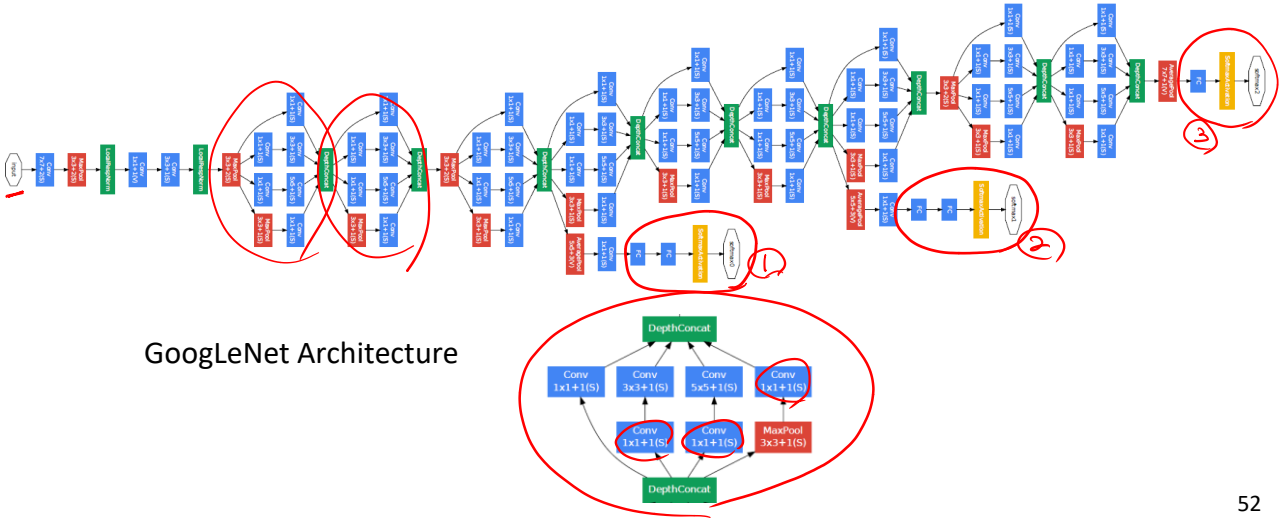
dimension reduction

51

51

ILSVRC 2014

- GoogLeNet



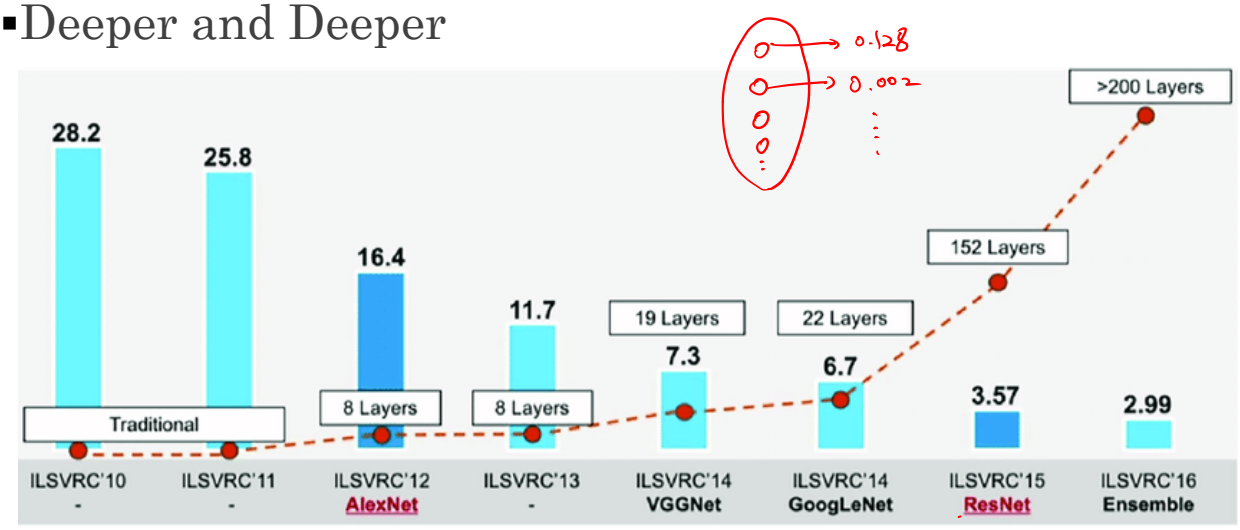
GoogLeNet Architecture

52

52

ILSVRC

- Deeper and Deeper



53

53

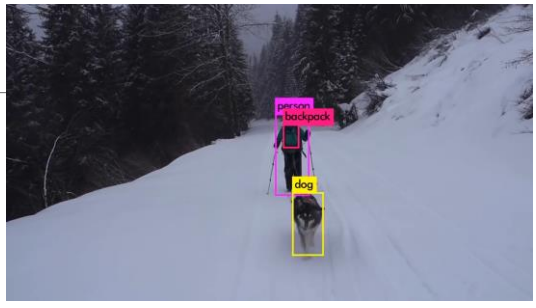
Object Detection

- Detecting instances of semantic objects of a certain class
  - Classification

    Car
  - Detection

    Car

    Location

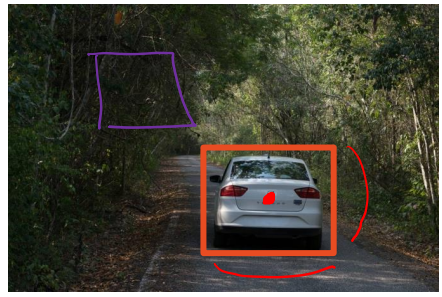


54

54

Object Detection

- Bounding box
  - $\vec{y} = [b_x, b_y, b_h, b_w, c_1, c_2, \cdots, c_n]^T$
  - c_i: 1 if its class is i
- Loss function
  - $L(\vec{y}, \hat{\vec{y}}) = \|\vec{y} - \hat{\vec{y}}\|_2^2$



$[\_, \_, \_, \_, 0.02, 0.123, 0.134, \cdots]$

55

Object Detection

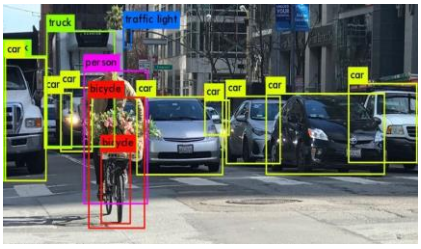
- Just when you think you have the correct answers, things get changed
- Bounding box
  - $\vec{y} = [p_c, b_x, b_y, b_h, b_w, c_1, c_2, \cdots, c_n]^T$
  - p_c: Image has the object
  - c_i: 1 if its class is i

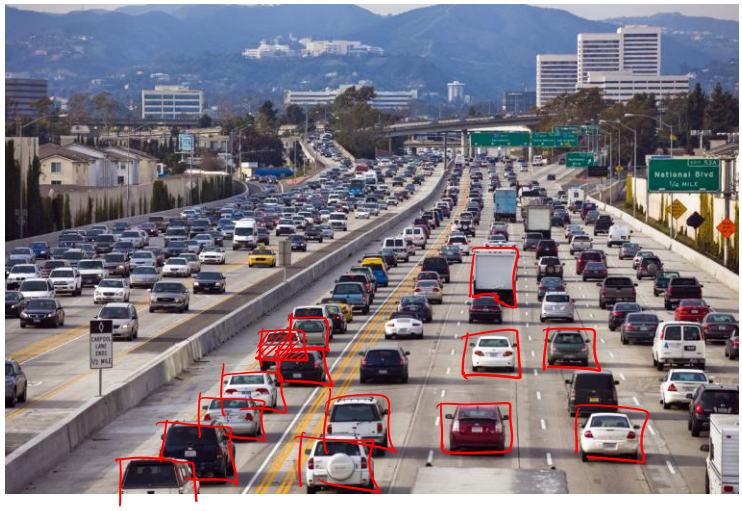
- Loss function
  - $L(\vec{y}, \hat{\vec{y}}) = \begin{cases} \|\vec{y} - \hat{\vec{y}}\|_2^2 & \text{if } p_c = 1 \\ (y_1 - \hat{y}_1)^2 & \text{O.W} \end{cases}$ (p_c, $\hat{p}_c$) $\Rightarrow \hat{p}_c^2 \to 0$

56

Multiple Objects

- In practice, labeling job is painful.



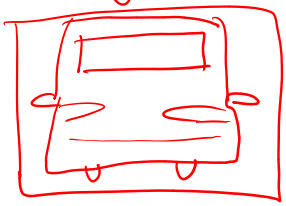


57

Can we utilize CIFAR-10 dataset?

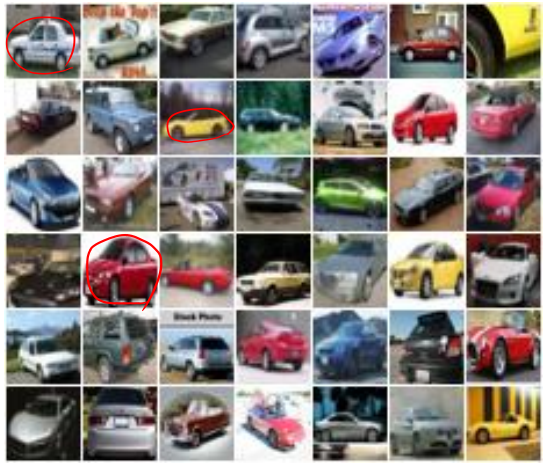
- We need a magic function s.t. $f(\vec{x}) = \hat{\vec{y}}$
  - $\vec{x}$: Image
  - $\hat{\vec{y}}$: Class label + bounding box

Test Image



Car!

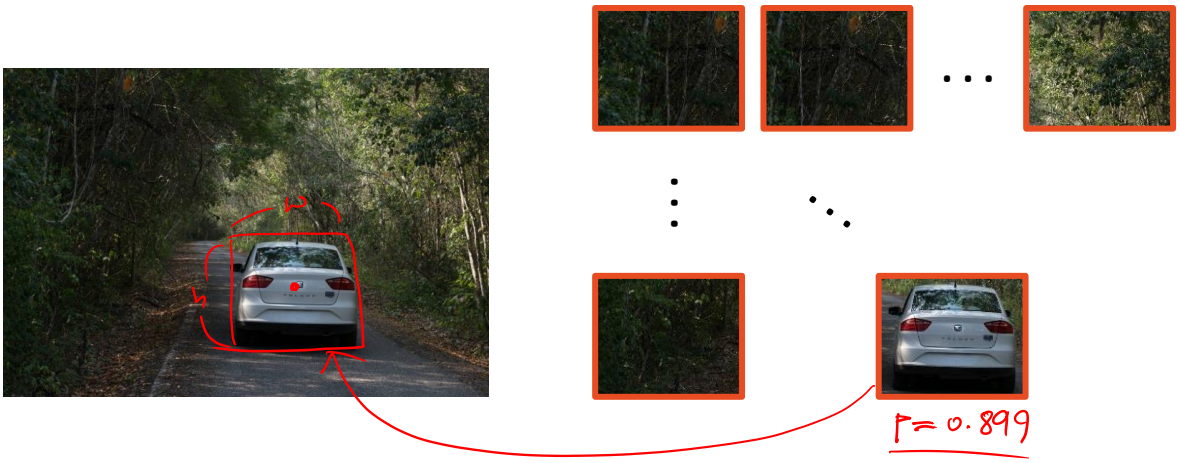
Training Images



58

Sliding window

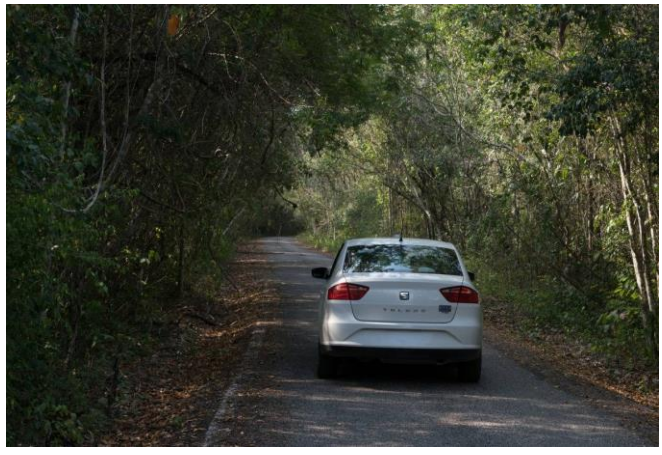
- Slide a box image and classify each image inside a box



59

Sliding window

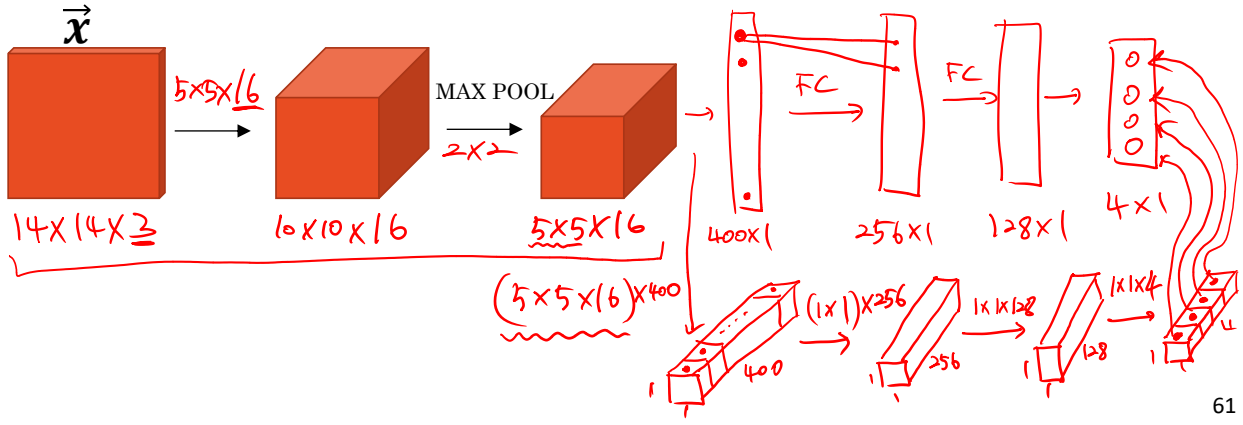
- Questions?
  - Shape of the boxes
  - Size of the boxes
  - Step interval



60

Fully connected layers

- Typical CNN architectures include
  - Convolutional layers (Pooling layers)
  - Fully connected layers (Softmax layer)

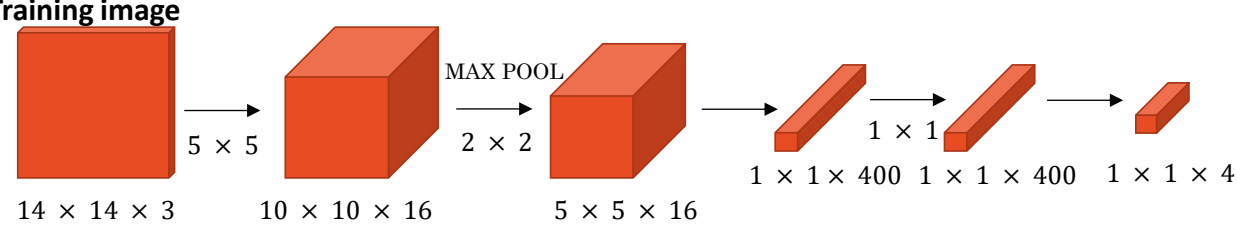


61

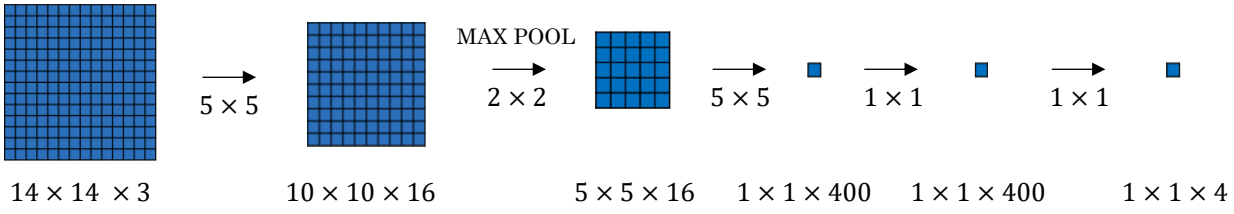
ConvNets for detection

- Fast/cheap implementation of sliding window

Training image



Simplified diagram

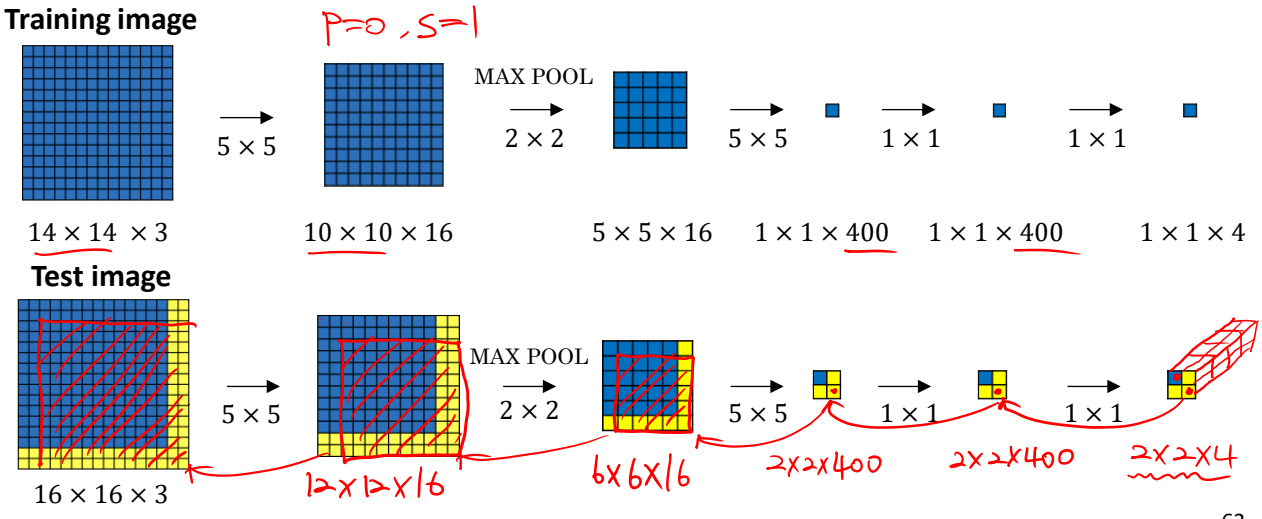


62

ConvNets for detection

$OH = \left\lfloor \frac{W+2P-FW}{S} \right\rfloor + 1$

- Fast/cheap implementation of sliding window

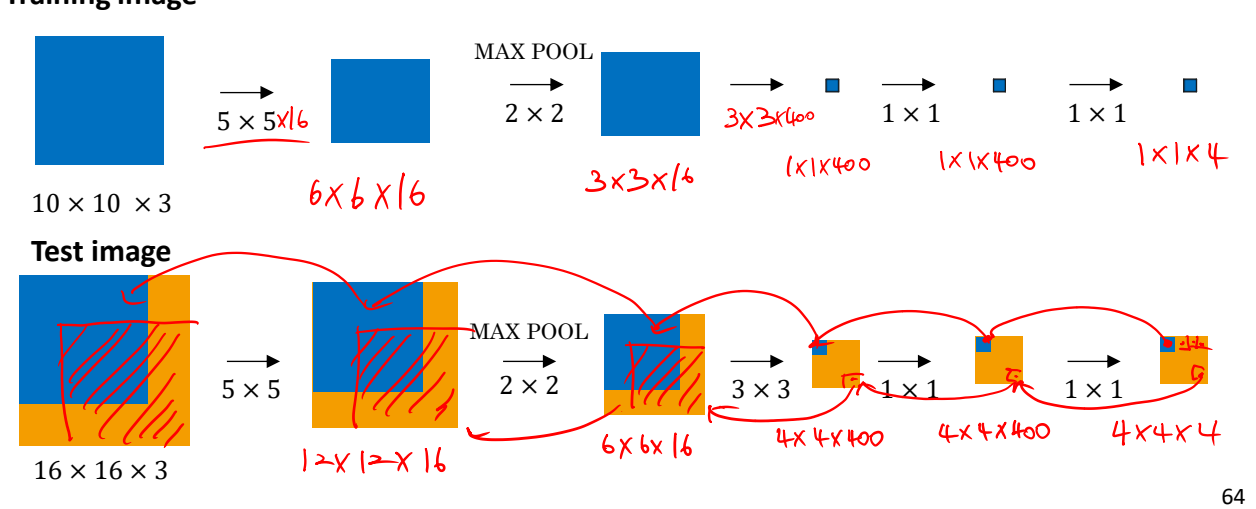


Sermanet, et al. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks

63

ConvNets for detection

- How we can change the window size and step interval?



64

Problem of sliding window detection

- What if the bounding boxes are not matched perfectly?
  - YOLO

65

65

Problem of sliding window detection

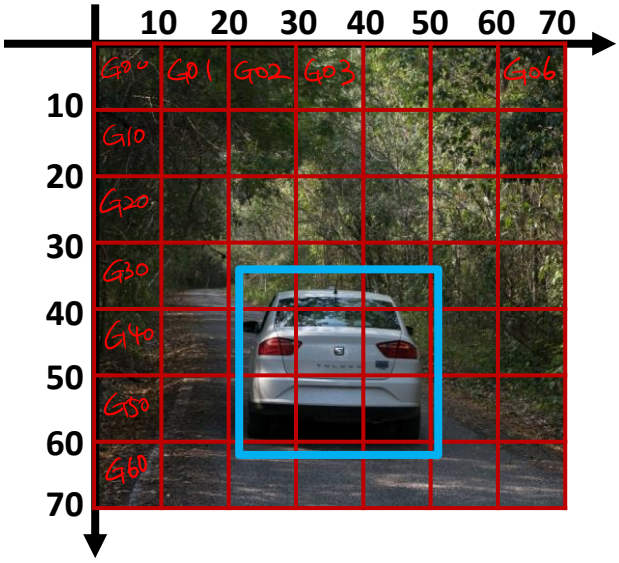
- What if the bounding boxes are not matched perfectly?
  - YOLO (You Only Live Once)

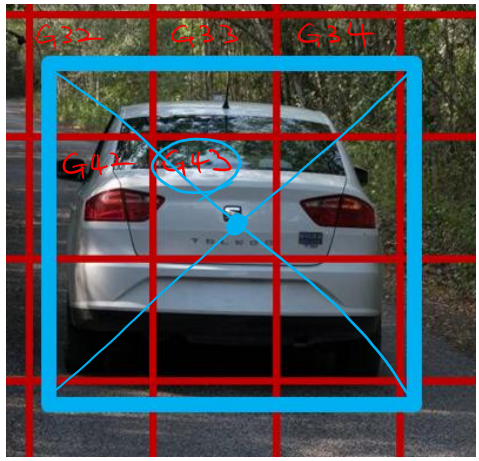
66

66

YOLO

- To us, YOLO stands for "You Only Look Once"

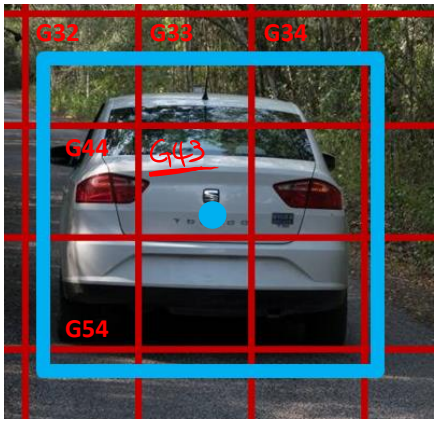


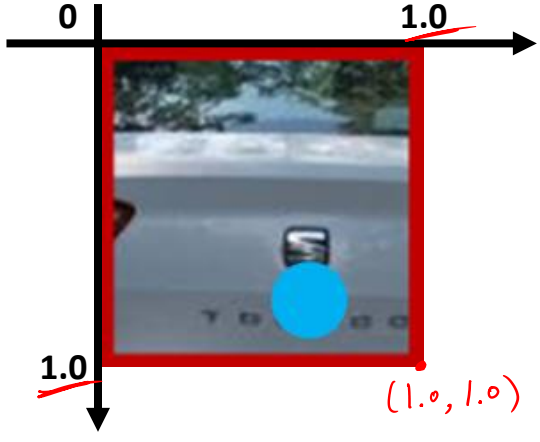


67

YOLO

- Labeling for the training set
  - The object is assigned to only one grid cell which includes the center of the bounding box





68

YOLO

- Labeling for the training set
  - The object is assigned to only one grid cell

$\vec{y} = [P_c, b_x, b_y, b_h, b_w, c_1, c_2 \cdots]^T$ (car)

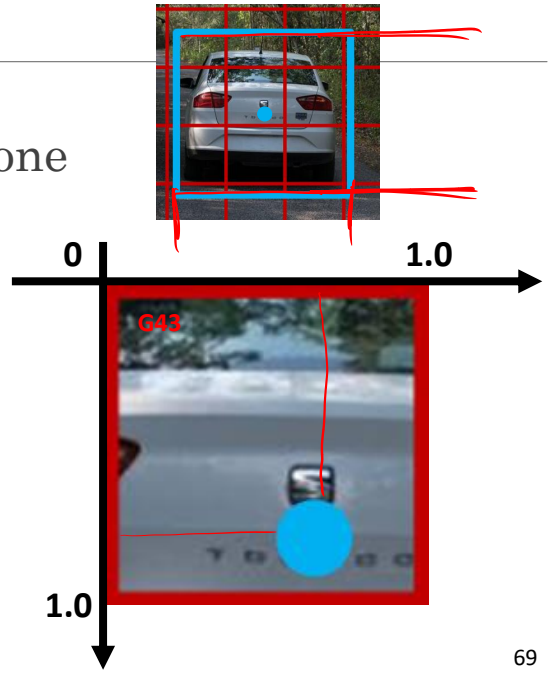
$\vec{y}_{43} = [1, 0.9, 0.8, 2.5, 2.9, 0, 1, \cdots]^T$

$\vec{y}_{42} = [0, ?, ?, ? \cdots]^T$

$\vec{y}_{53} = [0, ?, ?, ? \cdots]^T$

$|\vec{y}| \times 7 \times 7 = |\vec{Y}|$

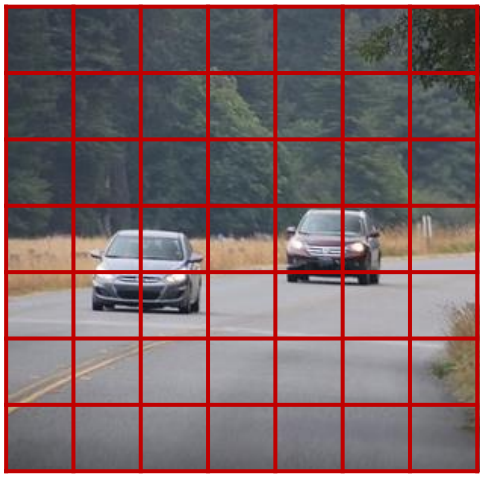
$(5+4)$



69

YOLO

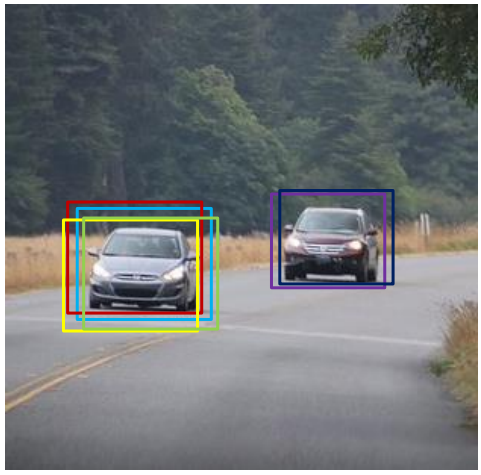
- At test phase
  - Multiple grids may include the same object.



70

YOLO

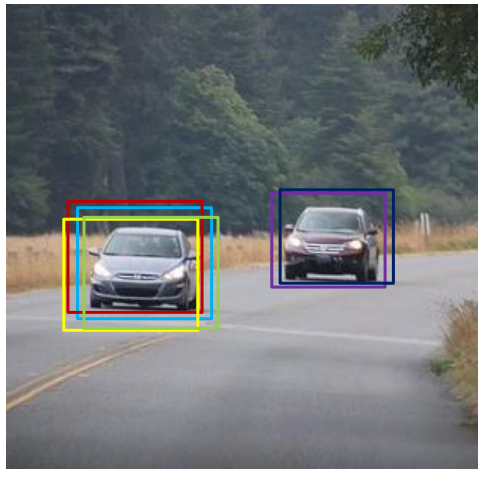
- Evaluation of bounding boxes



71

YOLO

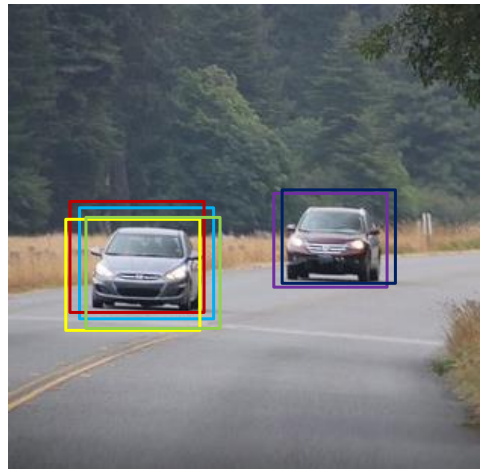
- Non-maximal suppression
  - Drop boxes s.t. $p_c < threshold$
  - Group boxes that are overlapped
    - Based on intersection of union (IoU) of A and B
    - $IoU = \frac{A \cap B}{A \cup B}$
    - For instance, $IoU > 0.5$
  - For each group, keep a box whose p_c is largest



72

YOLO

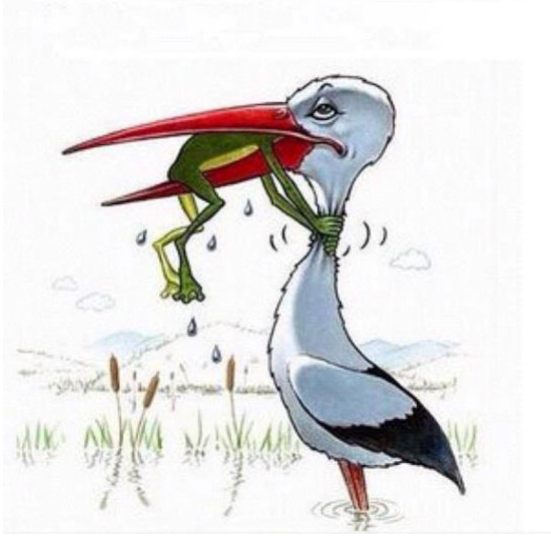
- Non-maximal suppression
  - $\vec{y}_{red} = [0.96, \dots]^T$
  - $\vec{y}_{blue} = [0.97, \dots]^T$
  - $\vec{y}_{green} = [0.94, \dots]^T$
  - $\vec{y}_{yellow} = [0.93, \dots]^T$
  - $\vec{y}_{navy} = [0.95, \dots]^T$
  - $\vec{y}_{purple} = [0.98, \dots]^T$

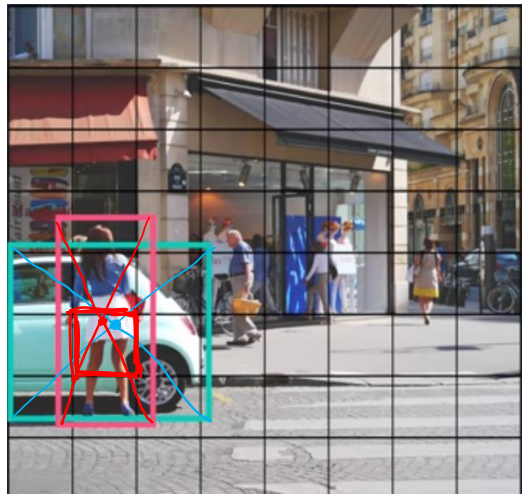


73

Never enough

- It ain't over till it's over





74

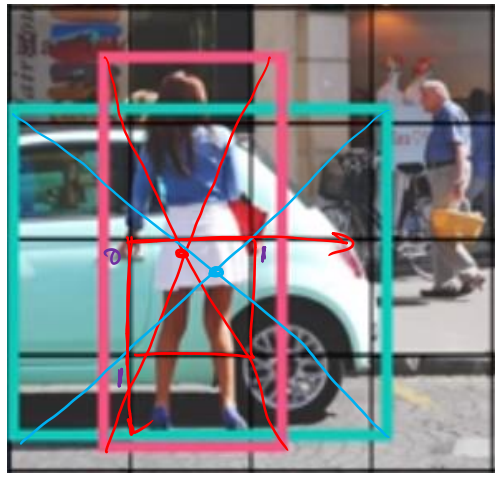
Anchor Boxes

- A grid can have multiple objects.

$$\vec{y} = \begin{bmatrix} p_c, b_x, b_y, b_h, b_w \\ p_c, b_x, b_y, b_h, b_w \\ c_1, c_2, \cdots \end{bmatrix}^T$$

ex) $\vec{y} = \begin{bmatrix} 1, 0.4, 0.1, 3.1, 1.4 \\ 1, 0.8, 0.3, 2.8, 3.1 \\ 0, 0, \cdots 1, \cdots, 1, 0 \end{bmatrix}$

↑ human ↑ car

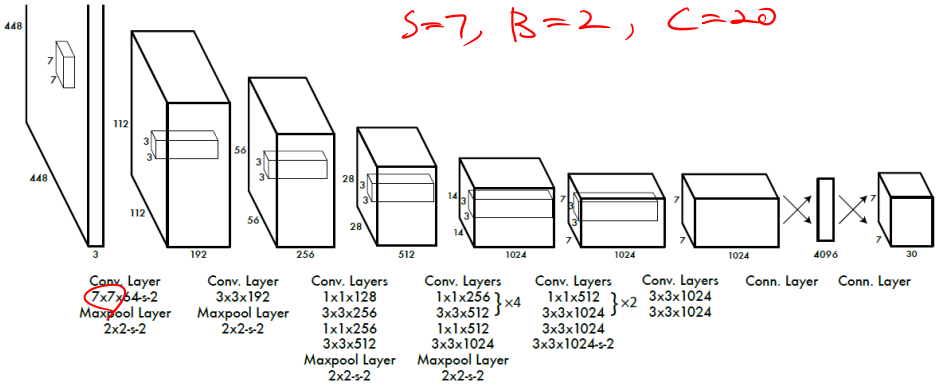


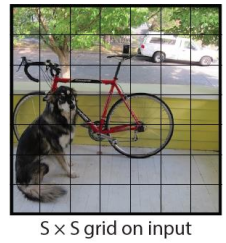
75

YOLO

- YOLO divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B \times 5 + C)$ tensor.

$S=7, B=2, C=20$





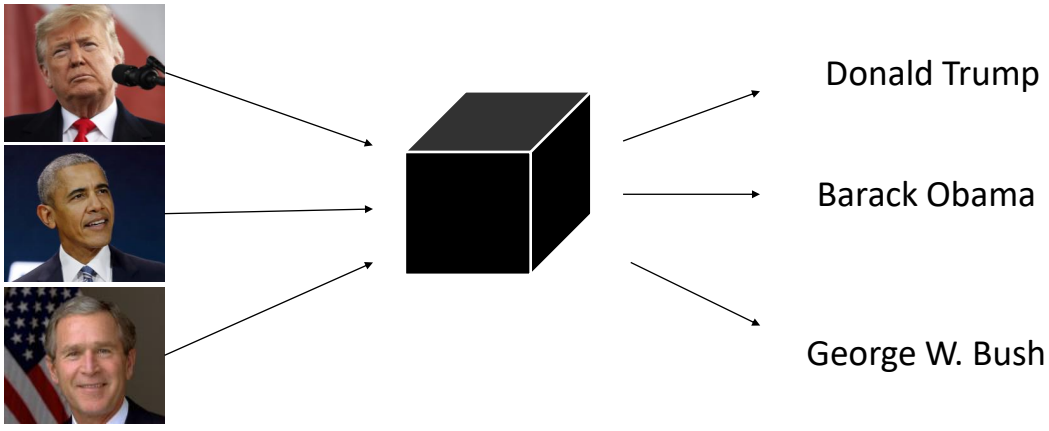
S × S grid on input

76

Face Recognition

- Identifying or verifying a person from visual data.
  - Input: image / video
  - Output: id / name

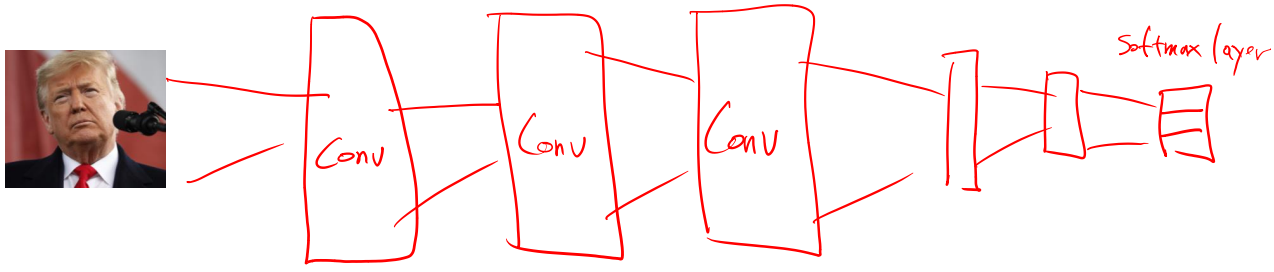
Closed set recognition



77

Face Recognition

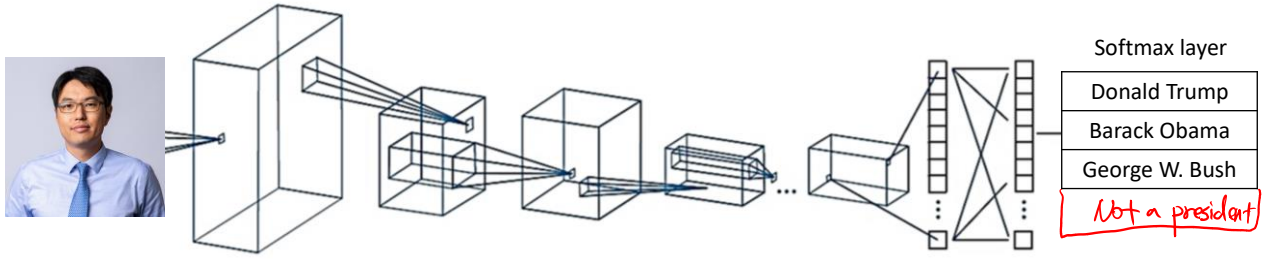
- Assume we have enough labeled data
- Network architecture (1st trial)
  - Convolutional layers (Dense layers)
  - Softmax layer



78

Face Recognition

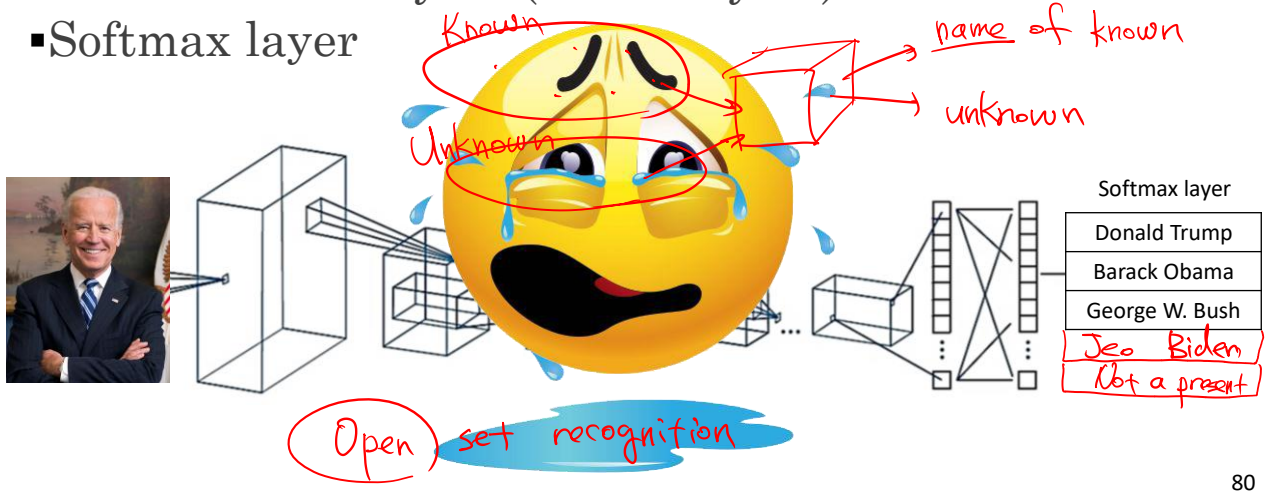
- Assume we have enough labeled data
- Network architecture (2nd trial)
  - Convolutional layers (Dense layers)
  - Softmax layer



79

Face Recognition

- Network architecture (3rd trial)
  - Convolutional layers (Dense layers)
  - Softmax layer



80

Metric based recognition

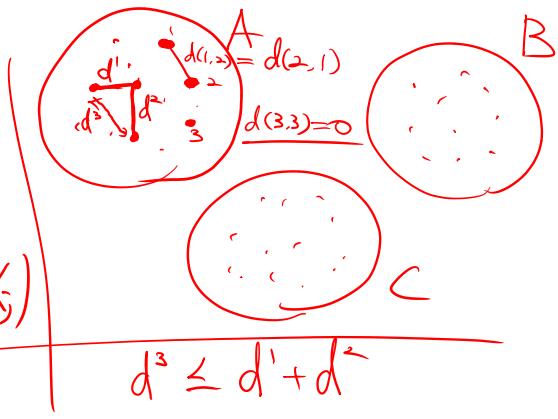
- Let $\vec{X}_i$ is an image of i.
- Assume we have a distance function $d(\vec{X}_i, \vec{X}_j)$ such that

$d(X_i, X_j) \geq 0$

$d(X_i, X_j) = 0 \Leftrightarrow X_i = X_j$

$d(X_i, X_j) = d(X_j, X_i)$

$d(X_i, X_j) \leq d(X_i, X_k) + d(X_k, X_j)$



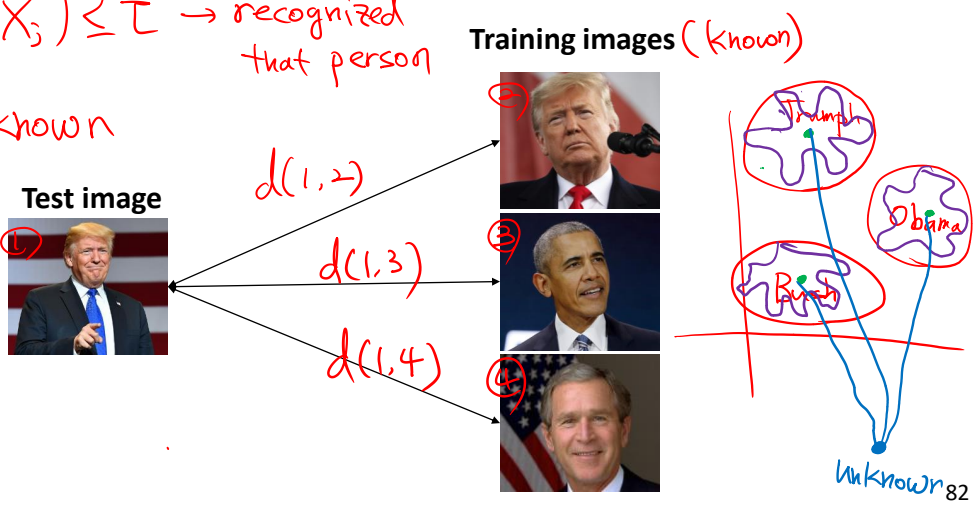
81

Metric based recognition

- Assume we have a metric $d(\vec{X}_i, \vec{X}_j)$

If $d(X_i, X_j) \leq \tau$ → recognized that person

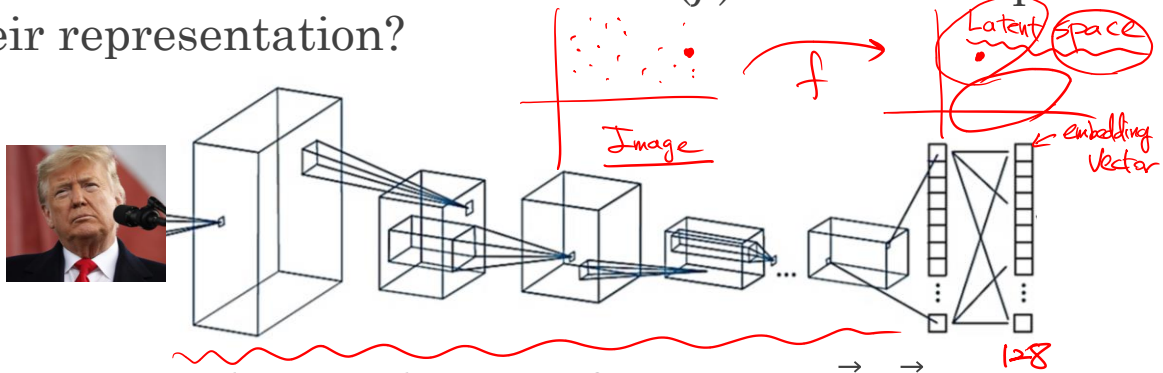
else ⇒ unknown



82

Representation Learning

- Can we utilize a neural network (f) to encode inputs to their representation?



- How can we have a distance function $d(\vec{X}_i, \vec{X}_j)$?

$d(X_i, X_j) = \| f(X_i) - f(X_j) \|_2^2 \quad \| X_i - X_j \|_2^2$

83

Embedding Vector

- We need to train a network to encode inputs such that
  - $d(\vec{X}_i, \vec{X}_j) = \|f(\vec{X}_i) - f(\vec{X}_j)\|_2^2$ is a distance between $\vec{X}_i$ and $\vec{X}_j$

$d(X_1, X_2) < d(X_1, X_3)$
$d(X_1, X_2) < d(X_1, X_4)$
$d(X_3, X_4) < d(X_3, X_1)$
$d(X_3, X_4) < d(X_3, X_2)$
$d(X_1, X_2) \; ? \; d(X_3, X_4)$

84

Triplet Loss

- We want to ensure that an image $\vec{X}^a$ (anchor) of a specific person is closer to all other images $\vec{X}^p$ (positive) of the same person than it is to any image $\vec{X}^n$ (negative) of any other person.

$d(X^a, X^p) < d(X^a, X^n)$ for all pairs

$d(X^a, X^p) + \alpha < d(X^a, X^n)$ α is a margin.

$\|f(X^a) - f(X^p)\|_2^2 + \alpha - \|f(X^a) - f(X^n)\|_2^2 < 0$

85

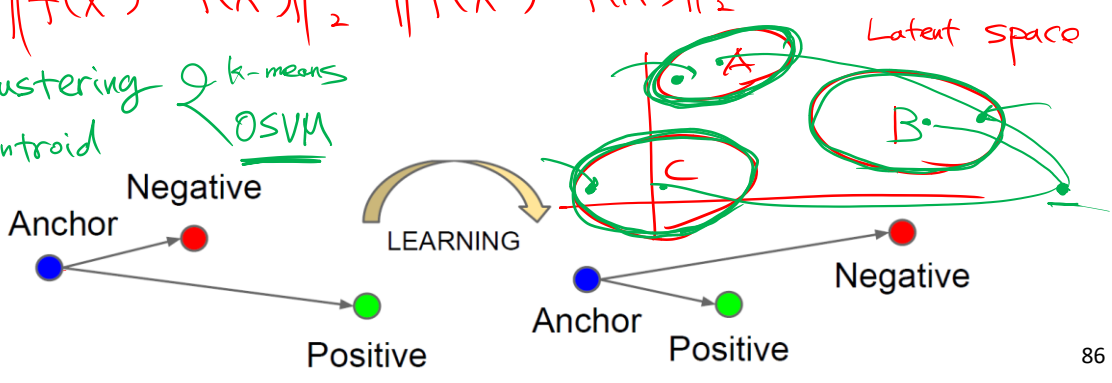
Triplet Loss

- Triplet Loss minimizes the distance between an anchor and a positive and maximizes the distance between the anchor and a negative

$L = \|f(X^a) - f(X^p)\|_2^2 - \|f(X^a) - f(X^n)\|_2^2 + \alpha$

1. Clustering & k-means
2. Centroid

OSVM



86