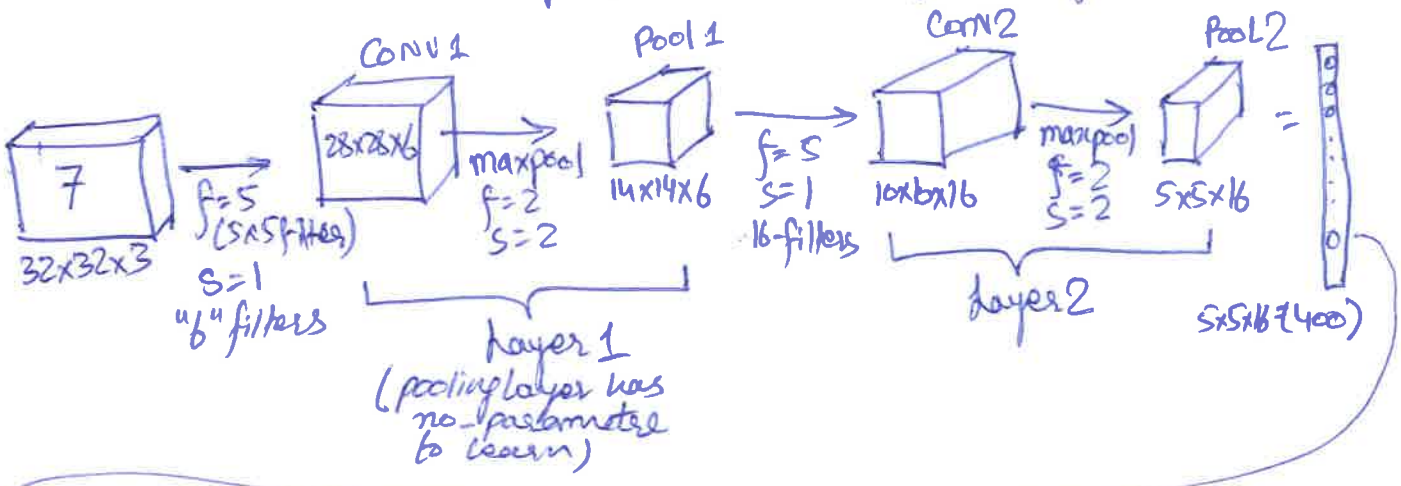
(1)

~~Image~~ Handwritten digits recognition



→ FullyConnected3 (FC3)

$W^{[3]} = (120 \times 400)$
$b^{[3]} = (120 \times 1)$

$W^{[4]} = (84 \times 120)$
$b^{[4]} = 84 \times 1$

→ Softmax (10 outputs)

Generally

1) $n_H, n_W \downarrow$, $n_C \uparrow$ } across layers
2) This pattern is followed
conv-pool-conv-pool

	Activation shape	Activation size	# of Parameters	
Input	(32,32,3)	$a^{[0]} = 3072$ (32×32×3)	0	
Conv1 (f=5, s=1)	(28,28,8)	6272 (28×28×8)	208	(5×5 filter × 3 channel + 8 bias = 5×5×3 + 8 = 208)
Pool1	(14,14,8)	1568	0	
Conv2 (f=5, s=1)	(10×10×16)	1600	416	(5×5 filter × 16 channel + 16 bias = (5×5×16) + 16 = 416)
Pool2	(5,5,16)	400	0	
FC3	(120,1)	120	48001	(120×400 + 1 bias = 48001)
FC4	(84,1)	84	10,081	(84×120 + 1 bias = 10,081)
Softmax	(10,1)	10	841	(10×84 + 1 bias = 841)

Neural Net Example

Very similar to LENET-5 (YANN LECUN)

W1-L9

MAX Pooling Example

And Avg

W1-L10

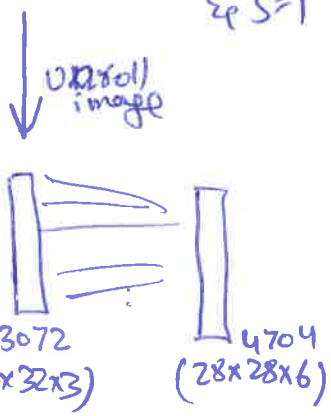
How Convolution reduces # of params?

CONV

* f=5(5x5) filter
* "6" filters
* no-padding
  i.e S=1

# of params here = $(5 \times 5 + 1) \times 6 = 156$ params

FC Network

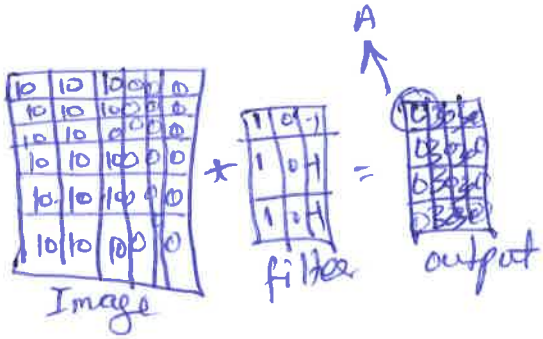


# of params here = $(3072 \times 4704) \approx 14\,M$

Why Convolution

1) Parameter sharing

look at ~~image~~. same set of filter is applied to entire image. (9 parameters)



2) Sparsity of connections

In each layer, each output value (for example "A") depends on only a small number of inputs from image.

W2-L1 CASE Studies

- Classic Networks
  1) LeNet-5
  2) AlexNet
  3) VGG
- ResNet (152-layered N/w)
- Inception NN

} CNN

W2-L2

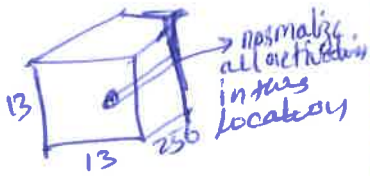
1) LENet5 (60K params) Lecun (98) → Gradient-based learning to document recognition

Advanced Notes

1) Sigmoid/tanh not Relu

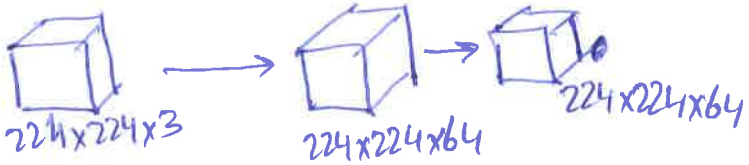
2) AlexNet (Krizhevsky et al 2012. ImageNet classification with deep CNN)
   - Similarity with LeNet-5 (it has 60M params)
   - Used Relu activations
   - Multiple GPUs
   - local response Normalization layer

easy paper than VGG paper than LeNet5



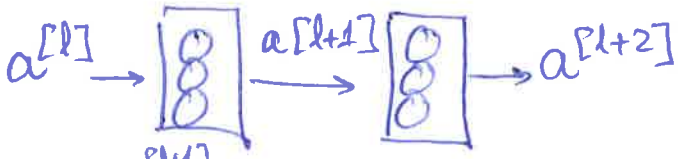
3) VGG-16 → 16-layers ~138M params (Simonyan & Zisserman 2015) Very deep CNN for large-scale image recognition

[Conv64] x2 :



Residual block

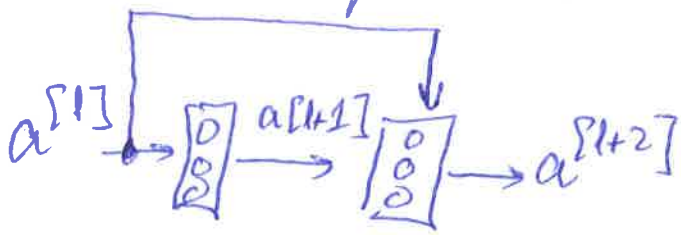
Consider this:



$a^{[l]} \rightarrow$ Linear $\rightarrow$ ReLU $\xrightarrow{a^{[l+1]}}$ Linear $\rightarrow$ ReLU $\rightarrow a^{[l+2]}$

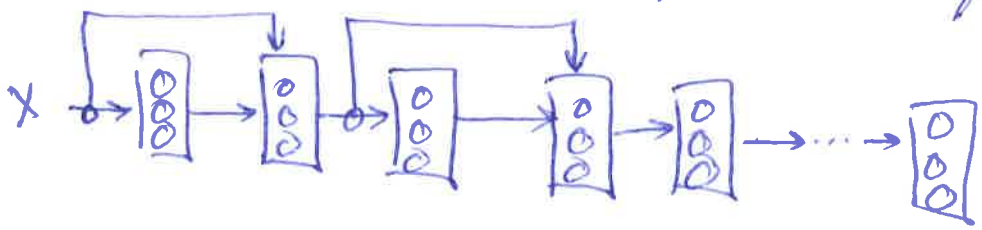
$z^{[l+1]} = W^{[l+1]} a^{[l]} + b^{[l+1]}$ $\quad a^{[l+1]} = g(z^{[l+1]})$ $\quad z^{[l+2]} = W^{[l+2]} a^{[l+1]} + b^{[l+2]}$ $\quad a^{[l+2]} = g(z^{[l+2]})$

Now adding "shortcut or skip connection"

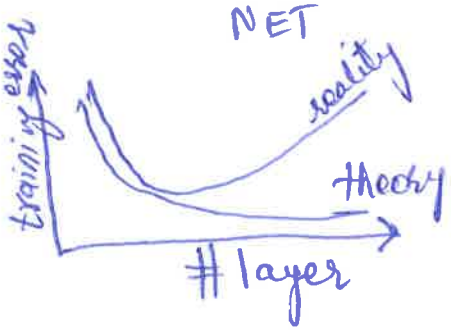


Here $a^{[l+2]} = g(z^{[l+2]} + a^{[l]})$

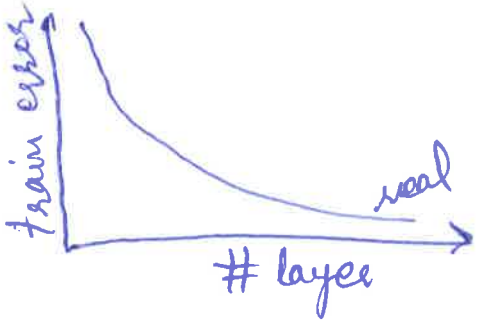
Residual N/w's (He et.al. 2015. Deep residual net for image recognition)



Plain NET



RESNet



W2-24

1) Why ResNet works

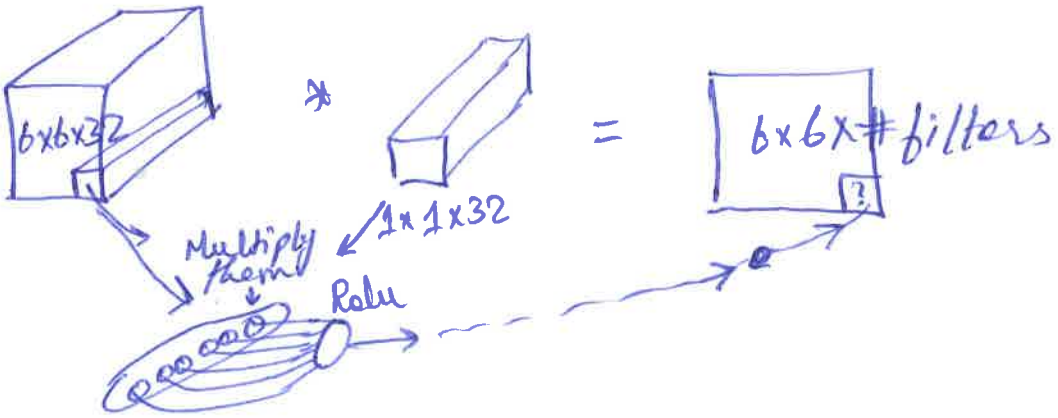
2) See the diagrams of ResNet in Slides
[He et al. 2015 Deep residual networks for image recognition]

W2-25

1x1 Convolution (Lin et al. 2013 Network in Network)

1-channel (no effect)

It makes sense with more than one channel

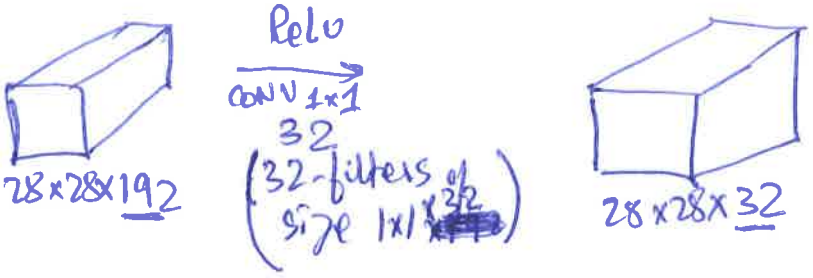


For more filters,

6x6x32 * 1x1x32 (1st filter)
1x1x32 (n-filter)

Here we will have "n" Relu units.

Using 1×1 convolution



* Pooling was used to decrease n_H, n_W but 1×1 convolution can be used to increase or decrease number of channels i.e n_c.

28×28×192 —Relu, conv 1×1, 192 filters of size 1×1×192→ 28×28×192

(It preserves # of channels & learn non-linearities)

W2-26

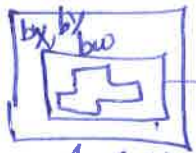
Motivation for inception network.
(Szegedy et al. 2014. Going deeper with convolutions)

W3-L1

Object detection with localization

1- C_1 (Pedestrian)
2- C_2 (car)
3- C_3 (motorcycle)
4- background

Need to output b_x, b_y, b_h, b_w
Class label (1-4)



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \rightarrow \text{if (object / No object)}$$

$$y = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \end{bmatrix} \rightarrow \text{No object} \quad \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{dont care}$$

$$\text{loss}(\hat{y}, y) = (\hat{y}_1 - y_1)^2 + \cdots + (\hat{y}_8 - y_8)^2 \quad \text{(Squared loss)}$$

(p_c … c_3; error)

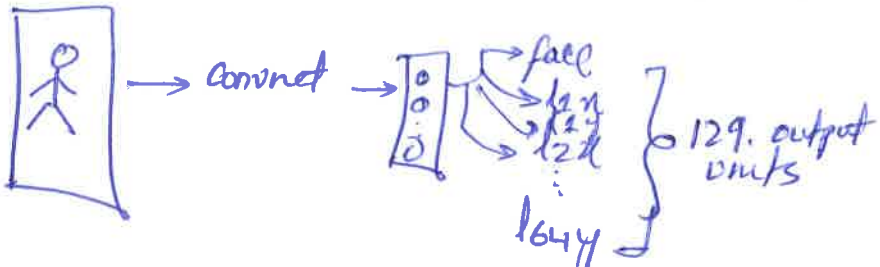
OR $\quad \text{loss}(\hat{y}, y) = \text{logistic regression loss}(y_1) + \text{squared error loss}(b_x, b_y, b_h, b_w) + \text{log likelihood loss for}\ (c_1, c_2, c_3)$

W3-L2

Landmark detection

It outputs (x,y) of image i.e output key points of image

i.e



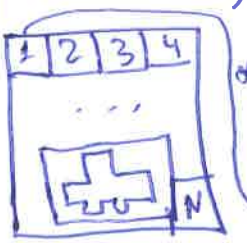
Pose-detection Annotate key positions of person's pose

W3-L3

* Sliding windows detection (used for objection detection)

W3-L4

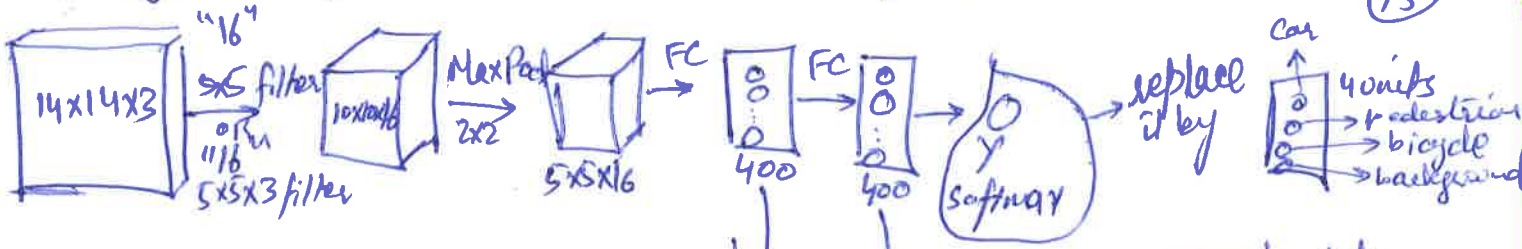
* Sliding window object detection using conv net is slow.



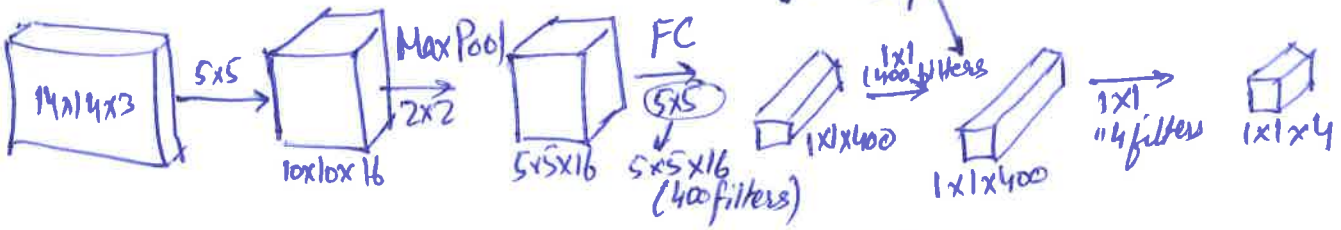
* To detect car with its bounding box, we need to extract image patch 1,2,3,4 as pass it to conv Net. It will output

4 numbers. So, running sliding window over image is expensive as we have to pass "N" patches to conv Net to get the location of car. To fix this problem, lets ~~~~ see convolutional implementation of sliding window.

Turning FC Layer into Convolutional layer

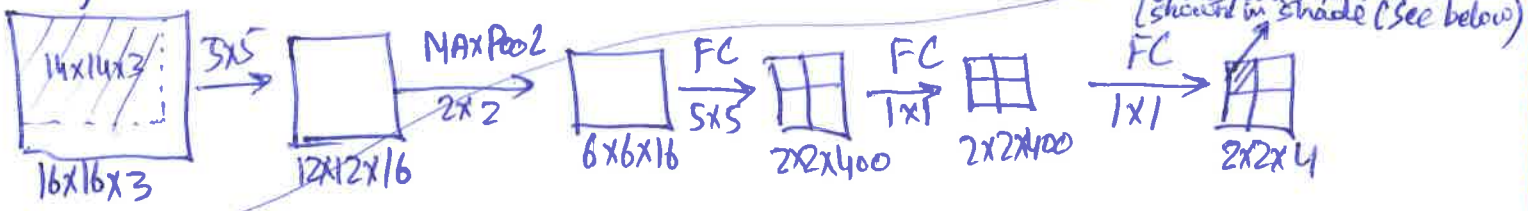


Implement them using convolution



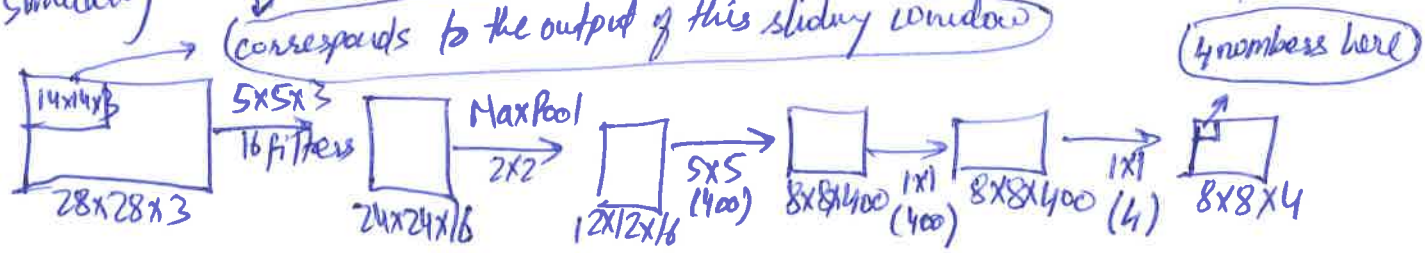
See Sermanet et al. 2014, Overfeat: Integrated recognition, localization & detection using convolutional networks

*Imagine you get an image of size 16x16x3. Show how can we do convolutional sliding window



Four numbers here are the result obtained by running 14x14x3 patch through convnet described above in the shaded region of 16x16x3 volume.

Similarly



Thus through a single pass, we obtain the detection over entire 28x28x3 image, just by implementing kernel.

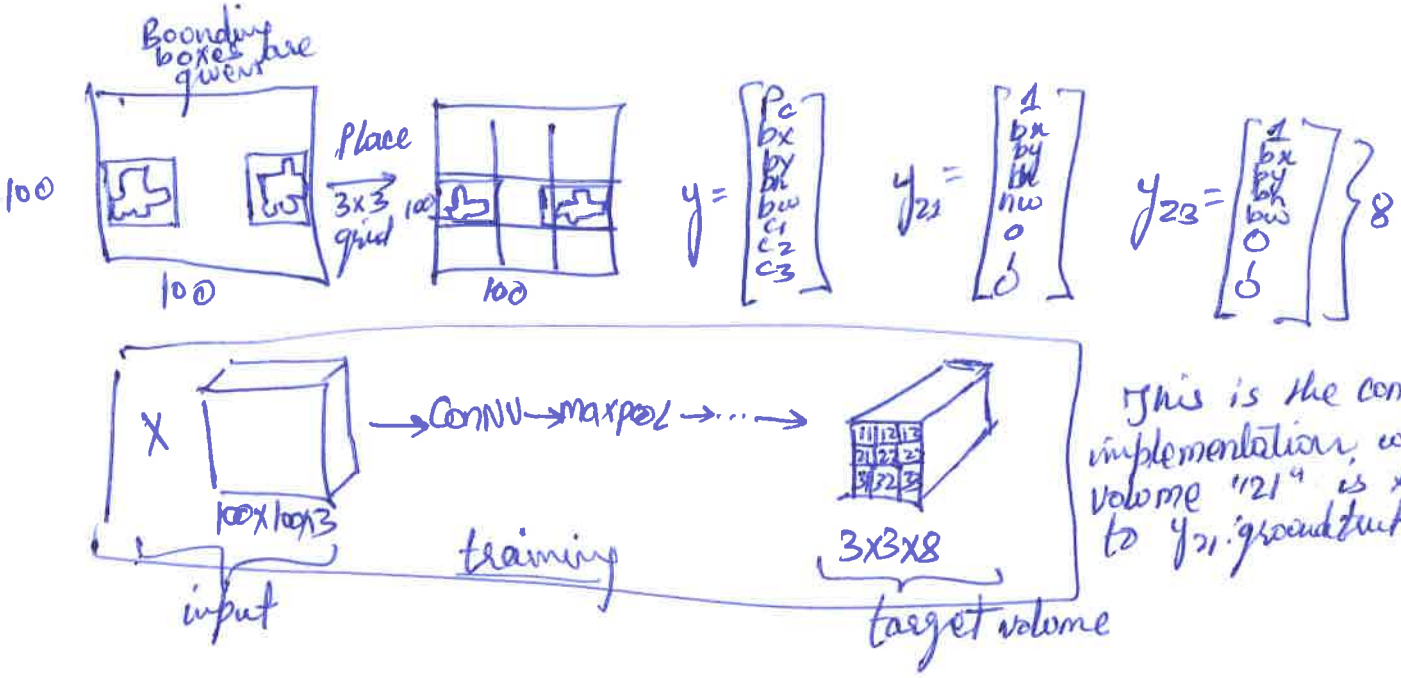
This type of object localization is not that much accurate. Now, we will have a look over Yolo algorithm that can make this accurate.

$$\underset{h_K}{\operatorname{argmax}} \left[w_{em} f(x_K, h_K) + \sum_{j=1}^{N} w_{TRANS} f(h_K, h_j) + \sum_{\substack{i=1 \\ i \neq k}} w f(h_i, h_K) + w_{COMP} f(h_K, y) \right]$$

W3-25

YOLO Algorithm

Redmon et al. 2015, You only look once: Unified real-time object detection (It is hard to read)



W3-26

Evaluating object localization

Intersection over union $IoU = \frac{\text{size of Intersection}}{\text{size of union}}$

"Correct" if $IoU \geq 0.5$ (In general)

* IoU is the overlap of two bounding boxes.

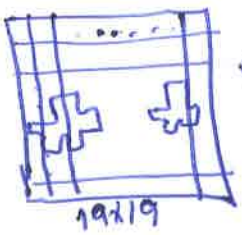
W3-27

Non-Max Suppression

Improves YOLO's accuracy of localizing bounding boxes.

In object detection & localization, we have multiple detections of single object. Non-max suppression give only one location of detection by choosing the bounding box with highest probability.

* Now have a look at non-max suppression algorithm.
It is simple



each output prediction is: $\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \end{bmatrix}$ $c_1 c_2 c_3$ are discarded right now

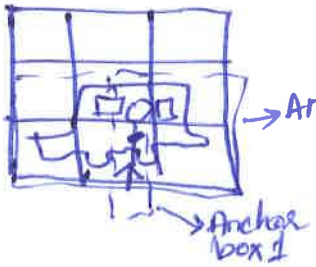
Discard all boxes with $p_c \leq 0.6$

while there are any remaining boxes:

1) Pick the box with the largest p_c. output that as prediction
2) Discard any remaining box with $IoU \geq 0.5$ with the box output in the previous step.

W3-28 Anchor Boxes

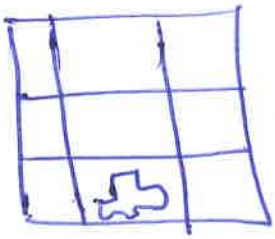
* Anchor Boxes allow us to detect multiple objects in a single grid cell.
(See the slides)



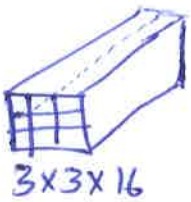
$$y = \begin{bmatrix} p_c \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \} \text{Anchor box 1} \\ \\ \\ \\ \\ \\ \\ \\ \} \text{Anchor Box 2} \end{matrix}$$

W3-29 YOLO Algorithm for object Detection (Redmon et al. 2015)

Training



* we have 2 Anchor boxes
* for each grid cell, $y = \underbrace{2}_{\text{Anchor boxes}} \times \underbrace{8}_{p_c, w_1 \ldots w_4, c_1 c_2 c_3}$
* Thus y for entire image
$y = 3 \times 3 \times 2 \times 8$

Training:  → ConvNet → 

Prediction

See the slides
* Get rid of low probability bounding box
* Apply non-max suppression for each class

W4-2.6 Region Proposal : (R-CNN) (regions)

Girshik et al 2013, Rich feature hierarchies for accurate object detection & semantic segmentation

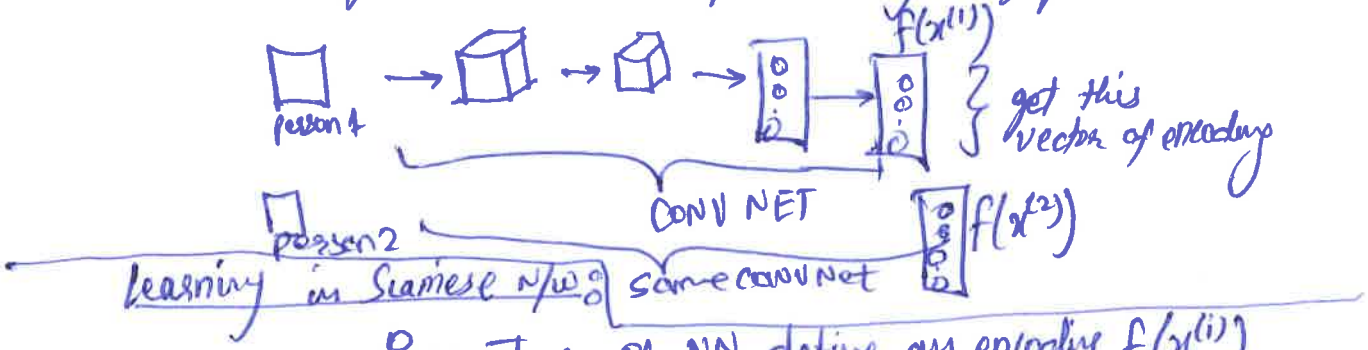
W4-L1

W4-L2

ONE-SHOT learning: learn from one example to recognize the person again.
It can be addressed by learning similarity function.

W4-L3 SIAMESE Network

Taigman et al. 2014. Deepface: closing the gap ...



Parameters of NN define an encoding $f(x^{(i)})$

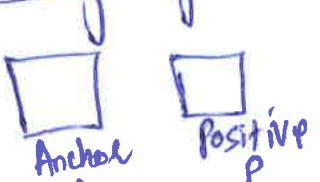
Learn parameters so that:

* If $x^{(i)}, x^{(j)}$ are the same person, $\|f(x^{(i)}) - f(x^{(j)})\|^2$ is small
* If $x^{(i)}, x^{(j)}$ are different persons, $\|f(x^{(i)}) - f(x^{(j)})\|^2$ is large.

W4-L4 Triplet loss

* How to define objective function for Siamese N/W.
* Schroff et al. 2015, FaceNet: A unified embedding for face recognition and clustering

Learning objective



* We are dealing with three images, anchor positive & negative. That's why it is called triplet loss

we want:

$$\|f(A) - f(P)\|^2 + \underbrace{\alpha}_{\text{margin}} \leq \|f(A) - f(N)\|^2$$

$$\Rightarrow \|f(A) - f(P)\|^2 - \|f(A) - f(N)\|^2 + \underbrace{\alpha}_{\text{NN margin}} \leq 0$$

Loss function (Triplet loss function for Siamese n/w)

Given three image A(anchor), P(positive), N(negative)
triplets

$$d(A,P,N) = \max\left(\|f(A)-f(P)\|^2 - \|f(A)-f(N)\|^2 + \alpha, 0\right)$$

over all training set:

$$J = \sum_{i=1}^{m} d\left(A^{(i)}, P^{(i)}, N^{(i)}\right)$$

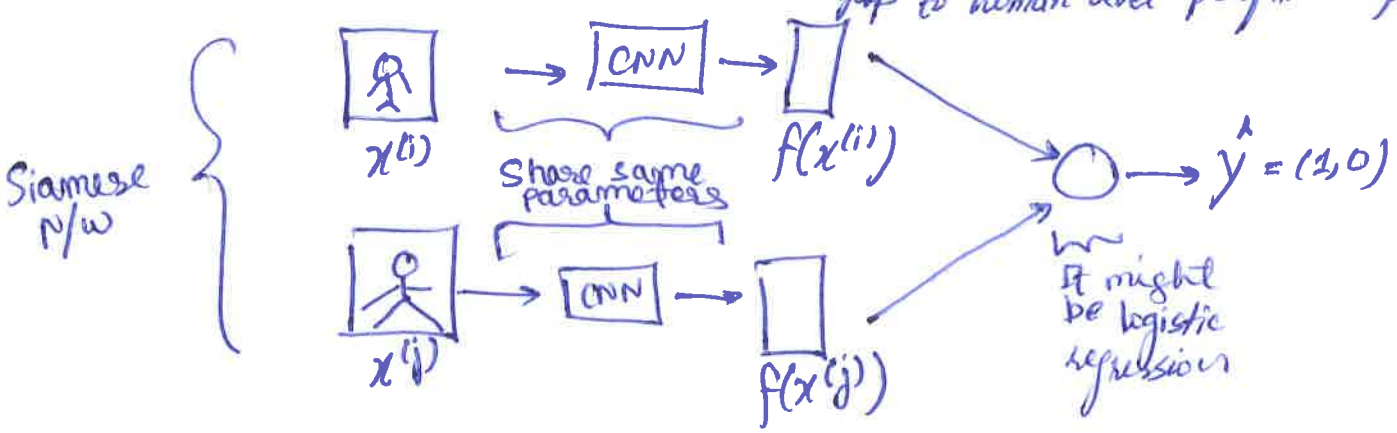
For training set
Choose triplets that are "hard" to train on.

$$d(A,P) + \alpha \leq d(A,N)$$

such that $d(A,P) \approx d(A,N)$

W1-L5 Face Verification & Binary Classification

(Taigman et al. 2014 DeepFace closing the gap to human level performance)



Out

$$= \sum_{k=1}^{128} \left| f(x^{(i)})_k - f(x^{(j)})_k \right|$$

If $f(x^{(i)}) \in \mathbb{R}^{128}$

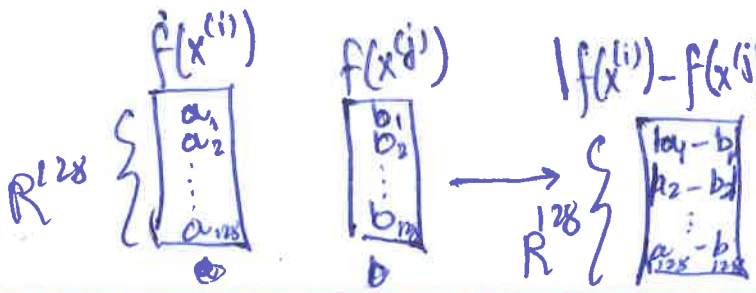
then

$$\hat{y} = \sigma\left(\sum_{k=1}^{128} w_i \left| f(x^{(i)})_k - f(x^{(j)})_k \right| + b\right)$$

(Sigmoid)

It could be Chi-Squared similarity function as well

$$\frac{\left(f(x^{(i)})_k - f(x^{(j)})_k\right)^2}{f(x^{(i)})_k + f(x^{(j)})_k}$$



W5-26

NEURAL STYLE TRANSFER

W5-27

Visualizing CNN

Very interesting: Zeiler & Fergus 2013, Visualizing and understanding CNN

W5-28

Neural STYLE transfer cost function

Gatys et al. 2015. A neural algorithm of artistic style.

$$J(G) = \alpha J_{CONTENT}(C, G) + \beta J_{STYLE}(S, G)$$

W5-29 Content Cost function

W5-210 Style Cost function

W5-211 1D & 3D Convolution