CONVOLUTIONAL NEURAL NETWORKS :

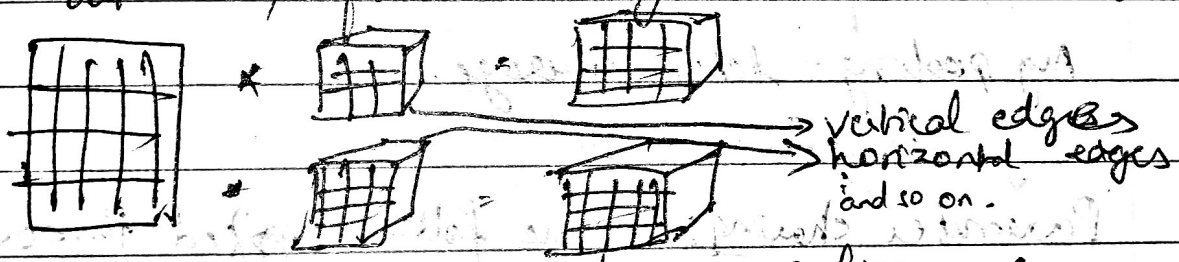
n x n image ; f x f filter ; padding = p ; stride = s

$$a = \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor \Rightarrow a \times a$$

Convolution is better called technically as cross-correlation.

Convolutions over volumes :

* use 3D filters corresponding to channels.
* No. of channels in filters = # channels in images.
* Cout - specifies how many such channels are used.



$$\frac{n - f + 2p}{s} + 1 \times \frac{n - f + 2p}{s} + 1 \times C_{out}$$

ONE LAYER OF CNN:

No. of params is irrespective of the size of the i/p image.

For ex: 3 x 3 filters of count 10 has

$3 \times 3 = [9 + 1 (\text{bias})] \times 10 = 280$ params.

Notations: $f^{[l]}$ = filter size | $p^{[l]}$ = padding | $s^{[l]}$ = stride | $n_c^{[l]}$ = # filters

Input: $n_H^{[l-1]} \times n_W^{[l-1]} \times n_c^{[l-1]}$

Output: $n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

$$n_{H/W}^{[l]} = \left\lfloor \frac{n_{H/W}^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$$

Activations:
$a^{[l]} \rightarrow n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

with batches $\rightarrow m \times a^{[l]}$

bias: $n_c^{[l]} - (1, 1, 1, n_c^{[l]})$

Weights:
$f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$

CONVNET EXAMPLE:
- Same Conv.
- Valid convolution

POOLING:

Max pooling intuition:
large numbers - a feature is detected
all features detected is given by M.P. If no feature low number is maintained.

* applied for each channel. So # channels will be same $n_{c,n} = n_{c(n+1)}$

Avg pooling: takes average.

Why CONVOLUTIONS?

Parameter sharing: → same filter applied throughout the image.

Sparsity of connections: o_p only depends on a particular filter of inputs.

Translation invariance: A pic of cat shifted a couple of pixels is still a cat.

DEEP CONVOLUTIONAL MODELS: Case Studies.

Architectures:

> LeNet-5 [60k params]

> AlexNet ⇒ 111th to LeNet but bigger ⇒ ReLU

[60M] ⇒ Multiple GPU's ⇒ Local Response Normalization (LRN) – normalizing a val across channels to avoid high activations.

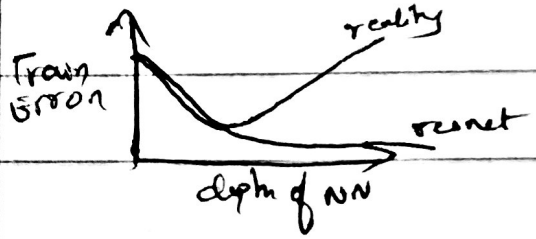
> VGG-16 net:

[138 M]

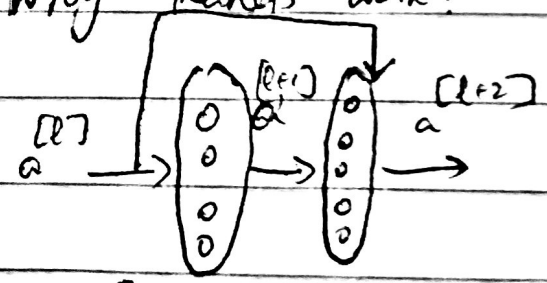
Resnets:

makes use of skip connection.

In reality, train error ↑ with deeper networks.



Why Resnets work?



$w^{[l+2]} \simeq 0$, after regularization

$$a^{[l+2]} = g\left(w^{[l+2]} a^{[l+1]} + b^{[l+2]} + w_s a^{[l]}\right)$$

↓ if diff dimension.

1 x 1 convolutions:
- shrink the # channels, thus ↓ volume.

Inception Network (Motivation)
- uses diff filter sizes & max pool layers but maintains image dims.
1x1 convolution - bottleneck (reduces computation cost significantly - does not affect performance)

Inception network:
- concatenation of inception blocks.
side branches: Take hidden layers & make a prediction

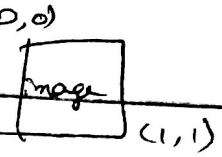
Transfer learning:
- freeze the bottom layers of the trained n/w by setting freezed & trainableParams = 0.
- ↑ more examples & ↓ # layers freeze at bottom & ↑ layers trained at top.
- Precompute one layer's activation for an image X & get a feature rep & train softmax on it.

Data Augmentation:
i) Mirroring - horizontal flip.
ii) Random cropping - should be reasonably large subsets.

3) Rotation / Shearing / Local warping etc..
4) Color shifting : R+20, G-20, B+20 ; PCA/B dec learn from a probability distribution.
5) PCA color augmentation ; If img. X is more R, G but less B, PCA CA reduces more in RG but less in B. (Refer AlexNet paper ↑)
* Have multiple CPU threads for loading & augmenting images.
> Hyperparams – how much color shift, warp factor etc..?

State of Computer Vision:
2 secs of knowledge:
>labelled data > hand-engineering (hacks)
* Do well on benchmark datasets : Tips:
i) Ensembling (Train NN's and avg. their ops)
ii) Multi-crop at test time - 10 crop of test image & average it.
iii) QUIZ ←

OBJECT DETECTION:



- classification with localization
  - train n/w to output b_x, b_y - midpoints of bboxes & b_h, b_w & class labels.

$$y = [p_c \; b_x \; b_y \; b_h \; b_w \; c_1 \; c_2 \; c_3]^T$$

(p_c ↓ 0/1 - no object / not?)

obj detection, loss $L(\hat{y}, y) = \begin{cases} (\hat{y}_1 - y_1)^2 + \cdots (\hat{y}_8 - y_8)^2 & \text{if } y_1 = 1 \\ (\hat{y}_1 - y_1)^2 & \text{if } y_1 = 0 \end{cases}$

c_1, c_2, c_3 - softmax o/p

b_x, b_y, b_h, b_w - squared error loss.

p_c - logistic regression loss

Landmark detection:

use consistent order of landmark points to get coordinates of landmarks.

$$y = \begin{bmatrix} p_c \\ l_{1x} \\ l_{1y} \\ \vdots \\ l_{nx} \\ l_{ny} \end{bmatrix} \begin{matrix} \rightarrow \text{human face or no?} \\ \} \; n \end{matrix}$$

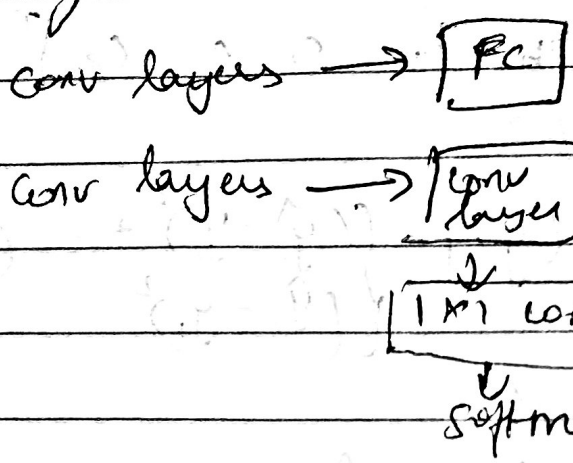
Object detection:

Sliding windows detection:

- start with small windows & slide across original image to get cropped images, that are fed into ConvNets.

CONVOLUTION IMPLEMENTATION OF SLIDING WINDOW:

- Turning FC into convolutional layers.



- Instead of using sliding windows, run the image across convolutions & in the final output, we get n×n×4 instead of 1×1×4, we can then check the n×n o/ps to check if object is detected.

Problem: bounding box dims. are not accurate.

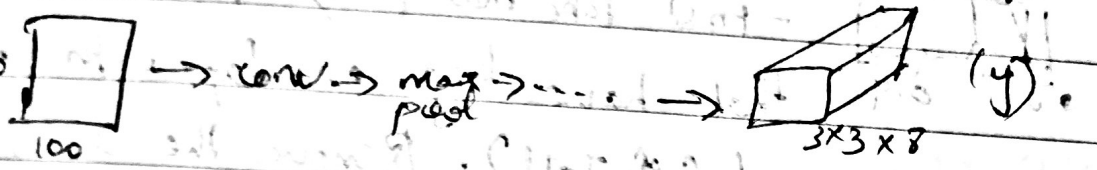
BOUNDING BOX PREDS:-

YOLO - You Only Look Once

* Split images as grid cells.
* Perform localization & classification for each grid.
* One $y = [\ldots]$ for each grid cell.
* An object belongs to the grid cell in which the object's midpoint falls.

If # n grid cells - O/p :- n x n x 8
(3x3x8) on video.

NN :-



Limitations:

- Restricted to 1 object per grid
- In practice, more objects do not occur in fewer grids

Efficiency : convolution implementation looks at the image only once. 9 grids in parallel.

BBoxes: b_x, b_y, ~~b_h, b_w~~ -> are b/w 0 and 1, as they are relative to the grid cell.

b_h, b_w -> >1 as one can have a bigger car.

INTERSECTION OVER UNION : (IOU)

- Evaluate ODA.
- Size of Intersection / Size of union => correct if IoU ≥ 0.5
- ↑ IoU, ↑ accuracy. (also measures similarity of 2 bboxes).

NON-MAX SUPPRESSION :

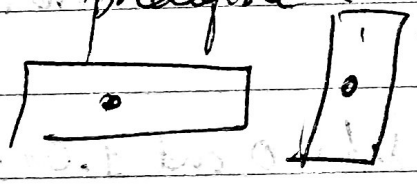
> Multiple detections of same objects. (ISSUE)

> In a 19x19 grid, multiple grids may detect an object, but the midpoints belongs to only one.

- First take max (0.9), all other rectangles that have high overlap with max are suppressed (of IoU). Remove the rects that are suppressed. Finds max & supperses others (non-max sup.)

ANCHOR BOXES:

→ If there are >1 object in grids, predefine shapes as anchor boxes.



→ y-outputs are concatenated.

→

YOLO ALGORITHM:

→ puts all the above readings together.

REGION PROPOSALS:

→ RCNNs, picks regions to run CNN's on.

R-CNN: propose regions. classify proposed regions one at a time. O/p label + bbox.