Convolution neural network

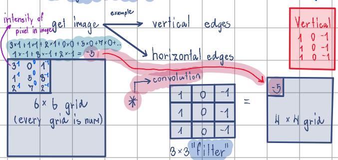
1. Computer vision

You sent image to computer. made, then figure out what is that. You don't need to figure out. Just locate it will do it by yourself.

convolution operation → using large image but in small size (small image (bad model), big image (overfitting model))

↳ using filters scan big image (numbers) to create Feature map (new images create by filter that scan)

2. Edge detection → use for simplified image



best number to use for filter?

Sobel filter			Scharr filter			edge detector		
1	0	-1	3	0	-3	1	0	-1
2	0	-2	10	0	-10	1	0	-1
1	0	-1	3	0	-3	1	0	-1

3. Padding

Control output size (→ same picture not same size)

→ adding layers of zero or other value outside actual data

original image	convolution! *	filter	=	result
n × n		f × f		(n-f+1)×(n-f+1)
6 × 6		3 × 3	=	4 × 4

1) Valid convolution (no padding) e.g. |6×6| * |filter 3×3| = 4 × 4

2) Same convolution (Pad then output size = input size)

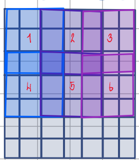
e.g. output ⌊n + 2p − f + 1⌋ × ⌊n + 2p − f + 1⌋ (original image, padding, filter; ↳ usually odd number)

you want output = original

$n + 2p - f + 1 = n$ → more pixel you need to add more

$p = \frac{f-1}{2}$ size padding

4. Strided convolution → step jump in convolution



e.g. filter 3×3 with stride = 2

image n×n 7×7	*	stride = 2, 3 × 3 filter	=	output 3 × 3

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

e.g. $\frac{7+0-3}{2} + 1 = \frac{7-3}{2} + 1 = \frac{4}{2} + 1 = 3$ #

5. Technical note on "cross-correlation" vs "convolution"

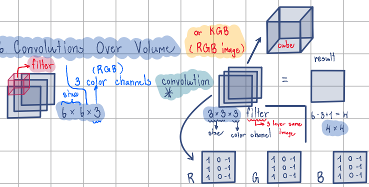
1	2	3
4	5	6
7	8	9

not flip

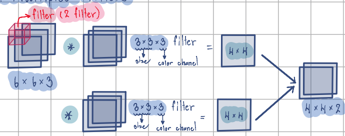
9	8	7
6	5	4
3	2	1

flip ↕ and ↔

6. Convolutions Over Volume

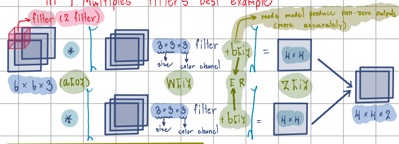


7. Multiples filters



8. One layer of a convolution network

↳ in 7. Multiples filters best example



$Z^{[1]} = W^{[1]} * a^{[0]} + b^{[1]}$ (bias) ← convolution operation

$a^{[1]} = g(Z^{[1]})$ ← add non-linear activation, the network can learn more complex

activation function

e.g. ReLu, sigmoid, tanh

call "ReLU" "Rectified Linear Unit"

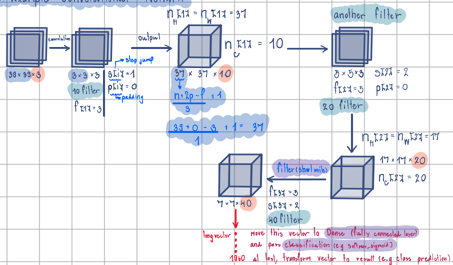
transform result that value < 0 to 0

Question

If you have 10 filters that are 3×3×3 in one layer of a neural network, how many parameters does that layer have?

click to know answer

9. Example Convolutional Network



convolution
39×39×3
3×3×3
10 filter
$f^{[1]} = 3$
$s^{[1]} = 1$
$p^{[1]} = 0$
→ step jump
↳ padding
output
$n_H^{[1]} = n_W^{[1]} = 37$
$n_C^{[1]} = 10$
37 × 37 × 10
$\frac{n+2p-f}{s} + 1$
$\frac{39+0-3}{1} + 1 = 37$
another filter
3×5×3
$f^{[2]} = 5$
$s^{[2]} = 2$
$p^{[2]} = 0$
20 filter
$n_H^{[2]} = n_W^{[2]} = 17$
17 × 17 × 20
$n_C^{[2]} = 20$
filter (short write)
$f^{[3]} = 3$
$s^{[3]} = 2$
40 filter
7 × 7 × 40
long vector
1960 at last, transform vector to result (e.g. class prediction)
move this vector to Dense (fully connected layer) and pass classification (e.g. softmax, sigmoid)

10. Type of layer in a convolution network

- Convolution (CONV)
- Pooling (POOL)
- Fully connected (FC)

reduce the size of the inputs

11 Pooling layer : Max pooling

input 4×4 → output 2×2

Hyperparameter

f = 2, S = 2

1	3	2	4
2	9	1	1
1	5	2	3
3	6	1	2

9	2
6	3

take average values instead of max values

12 Pooling layer : Average pooling

1	3	2	1
2	9	1	1
1	5	2	3
3	6	1	2

f = 2, S = 2

3.75	1.25
3.75	2

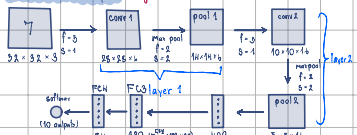
no parameter ⇒ just use hyperparameters that you set

Summary of pooling

Hyperparameters:

- f : filter size
- S : stride
- usually not using p: padding

13. CNN example e.g. LeNet - 5



common pattern → conv → pool → conv → pool → FC → FC → softmax

14 Why convolutions

e.g.

32×32×3 = 3072 → 28×28×6 = 4704 ≈ 14M

1) Parameter sharing ⇒ A feature detector (known as a vertical edge detector)

2) Sparsity of connections ⇒ In each layer, each output value depends only on a small number of inputs

Putting it together

Training set $(x^{(1)}, y^{(1)}) \dots (x^{(m)}, y^{(m)})$

Cost $J = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$

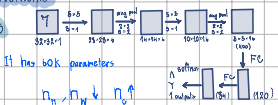
use gradient descent to optimize/reduce J

15. Why look at case studies

- Classic networks ⇒ LeNet - 5, AlexNet, VGG
- ResNet (152 layer)
- Inception

16. Classic Networks

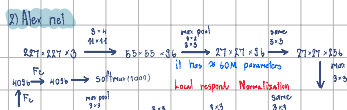
1) LeNet - 5



It has 60K parameters

$n_H, n_W \downarrow \quad n_C \uparrow$

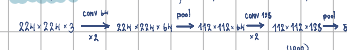
2) Alex net



It has ≈ 60M parameters

3) VGG - 16

conv = 3×3 filters, S = 1, Max-Pool = 2×2, S = 2

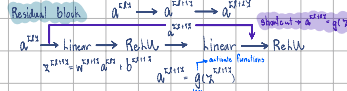


It has ≈ 138M parameters

17 Resnets (Residual Networks)

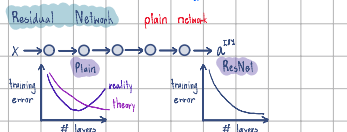
use for very deep NN

Residual block

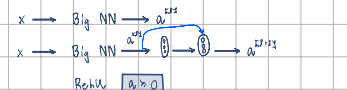


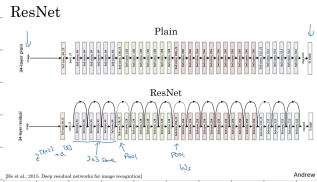
Residual Network

plain network

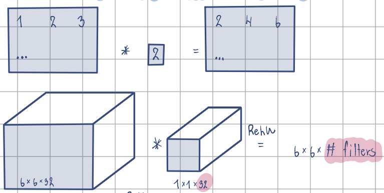


18 Why ResNets work



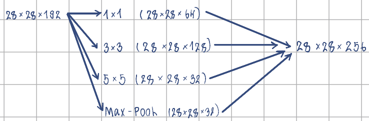


19. Network in Network and 1×1 convolutions



e.g. 28×28×192 → (ReLU, CONV 1×1×192, 32 filters or 192) → 28×28×32 (192)

20. Inception Network Motivation



The problem of computation cost

28×28×192 → (CONV 5×5, 32 filters) → 28×28×32

(28×28×32) × (5×5×192) = 120 million (expensive and hard to train)

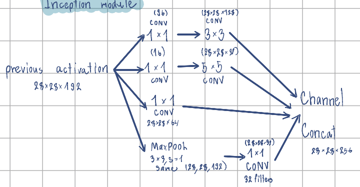
just using 1×1 convolution

28×28×192 → (CONV (16 filters) 1×1×192) → 28×28×16 → (CONV 5×5×16 (32 filters)) → 28×28×32

[28×28×16×192 = 2.4M] + [28×28×32×5×5×16 = 10M] = 12.4M

21. Inception Network (put a lot of these modules together)

Inception module



Inception network ⇒ a lot of Inception moduel put together

22. Transfer learning

e.g. 3 classes 1) Tigger 2) Misty 3) Neither

download implementation NN, change softmax to match your data.

just train softmax for outputs or freeze some layers and train some layers

$x \rightarrow \dots \rightarrow$ softmax $\rightarrow \hat{Y}$

23. Data Augmentation ⇒ improve performance computer visions

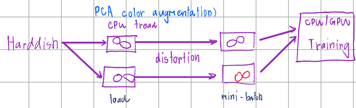
Common augmentation method

1) Mirroring
2) Random Cropping
3) Ratation
4) Shearing
5) local warping...

⋮

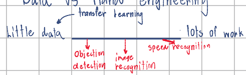
6) Color shifting ⇒ increase or decrease some color chanel

Advanced: PCA, ML-class.org, AlexNet paper



24. State of computer vision

Data vs hand engineering



Two source of knowledge

- Labeled data (X, Y)
- Hand engineer feather, network architecture, other components

tips

- Essembling ⇒ train several networks independently and average their outputs
- Multi-crop at test time ⇒ run classifier on multiples versions of test imagest results.

use open source code implementation

use pretained model and fine-tune on your dataset

Object Detection

25 Object Localization

Image classification ⇒ input picture then output result

Classification with localization ⇒ putting bouding box around result

Detection ⇒ multiples object

Classification with localization

image → CONV Net → (more output unit) softmax
→ bounding box
b_x, b_y, b_h, b_w

Define the target label y

need to output b_x, b_y, b_h, b_w, class label

y = [Pc, bx, by, bh, bw, C1, C2, C3]

Pc: is there an object (probability)

car 1, dog 2, neither 0

$\mathcal{L}(\hat{y}, y) = (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + \dots + (\hat{y}_8 - y_8)^2$ if $y_1 = 1$

$(\hat{y}_1 - y_1)^2$ if $y_1 = 0$

26. Landmark Detection

detect eye

key position

$l_{1x}, l_{1y}, l_{2x}, l_{2y}, \dots, l_{64x}, l_{64y}$

output unit

$(64 \times 2) + 1 = 129$

27. Object Detection

example training set = x (image), y (output label)

image → ConvNet → y

Sliding windows detection

change size of windows

from small to big (every iteration)

disadvantage ⇒ computation cost (use more resource) (too slow)

28. Convolutional implementation of sliding windows

→ better performance than sliding windows detection

Turning FC layer into convolutional layer

e.g. 14×14×3 →(5×5) 10×10×16 →(Max Pool 2×2) 5×5×16 →(FC) 400 →(FC) 400 → softmax (4) $\hat{y}$

14×14×3 →(5×5) 10×10×16 →(Max Pool 2×2) 5×5×16 →((400) 5×5×16 FC) 1×1×400 →(FC (400) 1×1) 1×1×400 →(1×1) 1×1×4

Convolutional implementation of sliding windows

14×14×3 →(5×5) 10×10×16 →(Max Pool 2×2) 5×5×16 →(FC 5×5) 1×1×400 →(FC 1×1) 1×1×400 →(FC 1×1) 1×1×4

16×16×3 →(5×5) 12×12×16 →(Max Pool 2×2) 6×6×16 →(FC 5×5) 2×2×400 →(FC 1×1) 2×2×400 →(FC 1×1) 2×2×4

29. Intersection Over Union (IoU) (How to tell obj Algorithm is working well)

Evaluating object localization

calculate intersection part area

= size of (intersection) / size of (union)

"correct" if IoU ≥ 0.5

30. Nonmax Suppression (make sure algorithm detect only one)

e.g. Pc (probability of detection)

0.8, 0.7 → left 0.8

seperate image to 19×19

Each output prediction is [Pc, bx, by, bh, bw] (multiple classes e.g. C1 car, C2 people, C3 more)

Discard all boxes with $P_c \leq 0.6$

any remaining boxes (output as prediction)

1. pick the box with largest Pc
2. Discard with box that IoU ≥ 0.5

31. Anchor Boxes

Overlapping objects

Anchor box 1 Anchor box 2

y = [Pc, bx, …, C3 (Anchor box 1), Pc, bx, …, C3 (Anchor box 2)]

2×8 → Pc, bx, by, bw, bh, C1, C2, C3

output ⇒ 3×3×16

32. Yolo Algorithm

Training

y = [Pc, bx, by, bh, bw, C1, C2, C3] (5 + 3 classes)

y ⇒ 3×3×2×8 (3 by 3 grid, anchors, 5 + # classes)

100×100×3 → ConvNet → 3×3×16

Making prediction

P_c ⇒ number of classes (1 car, 2 cat, 0 dog, 3 people)

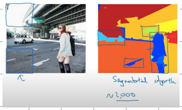
$C_{1 \dots}$ ⇒ check classes

Outputting the non-max supressed output

- for each grid call, get 2 predicted bounding boxes
- get rid of low probability predictions
- for each class use non-max suppression to generate final prediction

33. Region Proposals (R-CNN)

→ pick just few regions (windows) that make sense to run (instead running sliding windows)



Segmentation Algorithm

Fast R-CNN → use convolution implementation of sliding windows to classify all the proposed regions

Faster R-CNN → use convolution network

34. What is face recognition

Face Verification	Face recognition
- Input image, name/ID	- Has a database of K persons (need accuracy > 99%)
- Output whether the input images is that of the claimed person	- Get an input image
	- Output ID if the image in any of the K persons

35. One-Shot Learning

→ learning from one example to recognize

input_image_person → CNN → softmax ○ (4 or 5 outputs) → small example training data sets

Learning a similarity person

d(img1, img2) = degree of difference between images

If d(img1, img2) ≤ τ "same"

> τ "different"

36. Siamese Network

image $x^{(1)}$ → ▢ → ▢ → ▢ → ▢ → ▢ (128) $f(x^{(1)})$ parameters — encoding of $x^{(1)}$

image $x^{(2)}$ → get different $f(x^{(2)})$

comparison

$$d(x^{(1)}, x^{(2)}) = \| f(x^{(1)}) - f(x^{(2)}) \|^2$$

small → same | large → different

37. Triplet loss

Learning Objective

Anchor (A) | Positive (P) — quite look like

Anchor (A) | Negative (N) — different

$\| f(A) - f(P) \|^2 \leq \| f(A) - f(N) \|^2$ (margin)

$\to d(A,P)$ $\quad \to d(A,N)$

add α to make sure the NN won't get output = 0 on → +α margin

$$\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha \leq 0$$

Loss function — 10 picture for 1 persons better accuracy

given 3 images A, P, N

$$\mathcal{L}(A,P,N) = \max(\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha, 0)$$

(can be ≤ 0 or > 0)

cost function

$$J = \sum_{i=1}^{m} \mathcal{L}(A^{(i)}, P^{(i)}, N^{(i)})$$

Choosing the triplets A, P, N

- During training, if A, P, N are chosen randomly

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

- Choose triplets that're hard to train on. $d(A,P) \approx d(A,N)$

38. Face Verification

Learning the similarity function

new $x^{(i)}$ → ... → ▢ $f(x^{(i)})$ (can precompute) → ○ → $\hat{y}$ — Outputs (1 or 0) similar or not similar

database $x^{(j)}$ → ... → ▢ $f(x^{(j)})$

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

(sigmoid; 128 feature)

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

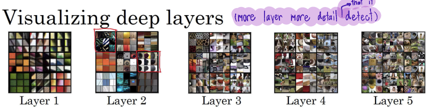
39. What is neural style transfer?

original images	
Content (C)	Style (S)

original images + style — Generated image (G)

40. What are deep CNs learning? (ConvNets)

- pick a unit in layer 1. (looking for edge, particular shade of color (simple features)) Find the nine images patches that maximize the unit's activation (Repeat for other unit)



(more layer more detail that it detect)

41. Cost function (for build neuro style transfer)

Content (C) + Style (S) ⇒ Generated image (G)

cost function; check how similar

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

find the generated image G

1. Initiate G randomly G: 100 × 100 × 3 (→ RGB)
2. Use gradient descent to minimize J(G) (small change style picture)

$$G = G - \frac{d}{dG}(J(G))$$

42. Content Cost Function

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

- say using hidden layer l to compute cost (→ not too small or too big)
- use pre-trained ConvNet (e.g. vgg network)
- Let $a^{[l](C)}$ and $a^{[l](G)}$ be the activation of layer l
- If $a^{[l](C)}$ and $a^{[l](G)}$ are similar then they will have same content

$$J_{content}(C, G) = || a^{[l](C)} - a^{[l](G)} ||^2$$

43. Style Cost Function