30-11-2021 Wednesday

Week #14 Lecture #18

Convolutional Neural Networks (CNN's)

We run a filter throughout an Image and each time we get a result. At the end we get a filtered image.

filter could be considered as weights.

After element wise multiplication, & Also doing Sum of all these elements. N

Also sum all values (weights) in filter (n) & divide Sum by Sum of filter & replace central element in image part with this divided result. (weighted) $\left(\frac{N}{n}\right)$

This is called image smoothening / Noise reduction

Usually we copy original image & apply filters on copied image & result is that image

After applying filter & to 1 part, then we go to next image part which is after skipping 1 column/Pixel to the right

Filter.

① To smooth Image / Noise reduction.

1	1	1
1	1	1
1	1	1

② To remove horizontal Lines / Detect vertical Lines.

-1	0	1
-1	0	1
-1	0	1

③ To remove Vertical Lines / Detect horizontal Lines.

-1	-1	-1
0	0	0
1	1	1

In General:-

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Result filtered image is called Features Map

Example.

22	22	22	22	22
22	22	22	22	22
50	50	50	50	22
22	50	50	50	22
22	22	22	22	22

Image

-1	-1	-1
0	0	0
1	1	1

filter

Applying filter.

$22 \times -1 + 22 \times -1 + 22 \times -1$
$+ 22 \times 0 + 22 \times 0 + 22 \times 0$
$+ 22 \times 1 + 50 \times 1 + 50 \times 1$

all 22 = 2
all 50 = 5

$= \boxed{6}$ Activation

6		

← features Map

Apply filter to anent part

$22 \times -1 + 22 \times -1 + 22 \times -1$
$+ 22 \times 0 + 22 \times 0 + 22 \times 0$
$+ 50 \times 1 + 50 \times 1 + 50 \times 1$

$= 9$

6	9	

Apply filter again

$22 \times -1 + 22 \times -1 + 22 \times -1$
$+ 22 \times 0 + 22 \times 0 + 22 \times 0$
$+ 50 \times 1 + 50 \times 1 + 22 \times 1$

$= 6$

6	9	6

Apply filter again Now shifted down

$22 \times -1 + 22 \times -1 + 22 \times -1$
$+ 2 \times 0 + 50 \times 0 + 50 \times 0$
$+ 22 \times 1 + 50 \times 1 + 50 \times 1$

$= 6$

6	9	6
6		

After applying a filter to all image we get

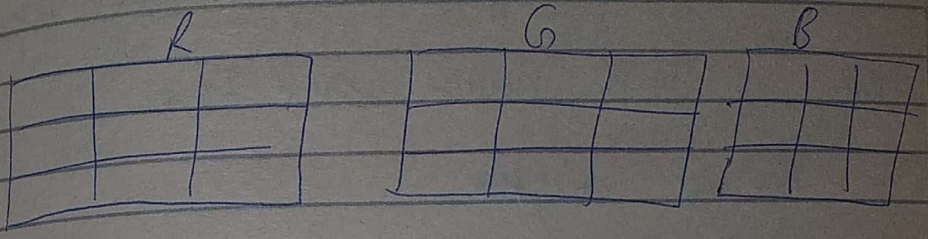
6	9	6
6	9	6
-6	-9	-6

features Map Result.

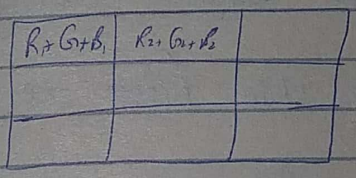
In case of RGB, we have filter size as

3 × 3 × 3
width × height × channels

Now we apply 3 filters on each channel



Now, we have feature Map simply 3×3



2D feature Map

After applying filter then we add all values for all channels
Activation values and place in features Map.

Advantages:

① In CIFAR, dataset we had 3072 features

x_1
x_2
.
.
x_{3072}

and if a single neuron attached to all inputs we had 3072 weights, if we had 200 neurons then we had 3072×200 weights

Now, if an image had $200 \times 200 \times 3$ size, then a single neuron would have

120,000 weights

In all of this, a neuron is looking at whole image, now if we use filter as weights for a single neuron. Now no. of weights are drastically reduced.

(2) In previous case, we were looking at a weight for a single pixel. Now, a weight is looking at multiple pixels.

(3) Optimization is increased.

How to get size of our feature Map?

First we have an image (1D)

e.g

12	10	15	5

image

And have filter | -1 | 1 | → detects vertical lines

Apply filter

-1
1

→ detect horizontal lines

$12 \times -1 + 10 \times 1 = -2$

$10 \times -1 + 15 \times 1 = 5$

$15 \times -1 + 5 \times 1 = -10$

feature Map | -2 | 5 | -10 |

In general, we have odd numbered filter size.

size of feature map:

$$R = S - F + 1$$

↙ Apply this formula for width & height separately.

S = size of image

F = size of filter.

e.g $S = 4$

$F = 2 \Rightarrow R = 4 - 2 + 1 \Rightarrow R = 3$

2D Image feature Map size:

$S = 5_h \times 7_w$

$F = 3_h \times 3_w$

for width:

$W_{out} = 7 - 3 + 1$

$\boxed{W = 5}$

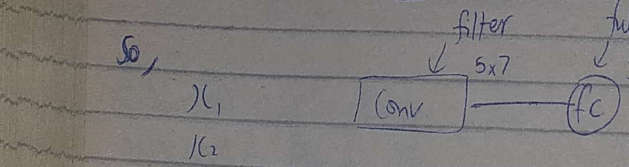
for height:

$h_{out} = 5 - 3 + 1$

$\boxed{h = 3}$

Resultant Image size would be

$\boxed{3 \times 5}$



So,

x_1

x_2

x_{307}

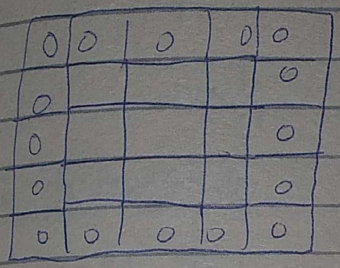
we initalize convolutional layers in front of NN to reduce dimensions & then at the end we have

fully connected layer.

We want our featur Map to be same size as original image size, we do Zero Padding

Zero Padding:

Pad Zero's to original image.



1 Zero Padding.

So that our image don't get down sampled.

as we have added 1 to left & 1 to right in with & size formula Becomes.

$h = I_h - F_h + 1 + 2P$ $\quad P = 1$ for above.

$W = I_w - F_w + 1 + 2P$ $\quad I_h$ = height of Image, I_w = width of Image

what to Add Padding: if filter size change.

as filter size becomes big, our feature Map gets small

we want formula st we get feature map size as original image size

Zero Padding depends on Filter size.

$P = (F-1)/2$ # For both height & width.

or $P = \left\lfloor \frac{F}{2} \right\rfloor$

Example:

Image size = 13x13

Filter size = 5x5

Find Padding:

$$P = \frac{5-1}{2}$$

$$P = \frac{4}{2}$$

$$P = 2$$

$$h = 13 - 5 + 1 + 2(2)$$

$$= 13$$

$W = 13 - 5 + 1 + 2(2) = 13$.

if image is 13×12

filter 5×5 $\quad P = 2$

$h = 13 - 5 + 1 + 2(2) = 13$

$W = 12 - 5 + 1 + 2(2) = 12$

So, image size doesn't affect formulas.

filter size is usually square with same height & width.

Jump in filter/stride:

Before it was 1 pixel jump to right or below

Now this changes our features map.

① Computation is decreased

② Dimensions are decreased further.

③ Some pixels are not looked at. So weights are only looking at some pixels.

Now formula becomes

$$W = \frac{(I_W - F_W + 2P)}{S} + 1 \qquad h = \frac{(I_h - F_h + 2P)}{S} + 1$$

S = stride

Multiple filters in one Convolution layer:

one conv layer may have multiple filter.

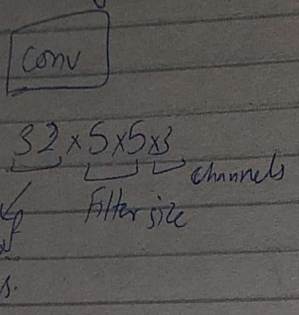


Image shapes to be resized to same shape.