EVA4

Assignment-1

Channels & kernels.

Kernel: A unit. In image processing a kernel, a convolution matrix or mask is a small matrix. It is used for blurring, sharpening, embossing, edge detection & more. This is accomplished by performing convolution b/w a kernel & an image.

In CNN dict.: kernel = filters = feature detector.

Channels: Channels refers to a certain component that defines pixel values in a digital image.

Ex: A color image is an aggregate of three channels: RGB (red, green & blue). The color data of an image is stored in these arrays of values, known as channels.

How to choose b/w smaller & larger ~~pixel~~ filter size?

How to choose b/w smaller & larger ~~pixel~~ filter size?

- Smaller filter looks at very few pixels at a time hence it has small receptive field. Whereas large filter will look a lot of pixel hence having large receptive field.

- When we work on/with smaller filters we focus on every minute details & capture smaller complex features from image, whereas when we work with large filters we tends to search for generic features which will give us basic components.

- After capturing smaller/minute features from image we can make use of them later in the processing. We lose this benefit with large filters as they focus on generics not specific features.

Why 3x3 & not any other filter like 5x5 or 7x7?

- Less filter less computation, big filter more computation.
- It learns large complex features easily, whereas large filters learn simple features.
- Output layers will be less when we use 3x3 filters as compared to 5x5 or bigger filters.

~~***~~

- Also, since there will be less number of output layers when using 3x3 filters & therefore less memory will be required to store them as compared to 5x5 or bigger filters.

Note

199x199 on convolution with 3x3 → 197x197.

Lecture-2 Notes

Note If we convolve a 3×3 kernel on a 5×5 image, the output channel we would create will have a resolution of 3×3. This is true only in the case when:

① : we are not using any padding (we are not adding additional 0's on the boundaries of our ~~output~~ input image, changing it's resolution, say from 5×5 to 7×7) &

② we are not using stride of more than 1.
(in general in our fig. whenever the kernel moves, it just skips 1 pixel. If it were to skip 2 pixels, that would be called a stride of 2)

Note Max Pooling.

How many layers would we need to move 400×400 image to 1×1?

As we saw, we need to add around **200 layers**, as each convolution reduce the size of image/channel by 2.

So, **400/2 = 200** gives the no. of layers to add.

Now, these are an insanely large no. of layers. We can do much better than this.

Max pooling

4	3	8	5
9	1	3	6
6	3	5	3
2	5	2	5

Input (4 x 4)

Filter size = 2 x 2.
Stride = 2.

→ Max pooling →

9	8
6	5

Output (2 x 2)

How many kernels are required?

We would need a set of edges & gradients to be detected to be able to represent the whole image.

Through experiments, we have learned that we should use around 32 or 64 kernels in first layer, increasing the number of kernels slowly.

390
155 Max pooling

Should we around 32 or 64 Kernels in first layer, increasing the number of kernels slowly.

390
1ss maxpooling

Let's assume we add 32 kernels in I st layer,
64 kernels in II nd layer &
128 " in III rd layer & so on.

Our Network would look sth like this:

400 x 400	(3x3) x 32	398 x 398 x 32.
398 x 398	(3x3) x 64	396 x 396 x 64

....

One need to observe here that the input to second layer is not 398 x 398 but 398 x 398 x 32. as we added 32 kernels.

Note: Each kernel would create its own channel.

3x3 is misleading!

Let's represent it properly.

as $3 \times 3 \times 1$

Total no. of kernels.

Input	Kernel	Output
$400 \times 400 \times 1$	$(3 \times 3 \times 1) \times 32$ (1 channel.)	$398 \times 398 \times 32$
$398 \times 398 \times 32$	$(3 \times 3 \times 32) \times 64$	$396 \times 396 \times 64$

32 channel signifies ?? Are they RGB ??

....

Notice that our kernels in 2nd layer has 32 channels.

Note:

① Our kernel must have an equal number of channels as the input channel.
because each channel in the kernel will look only at 1 channel.
(say channel # 23)
(channel no. 23 in input)

② If we have ∞ number of channels in the input, our kernels must have ∞ channels.
This has nothing to do with output channels.

③ Output channels are equal to number of kernels we use.

Let's represent it properly.

of $3 \times 3 \times 1$

Input	Kernel (Total no. of kernels ↑)	Output
$400 \times 400 \times 1$	$(3 \times 3 \times 1) \times 32$ (+ Channel.)	$398 \times 398 \times 32$
$398 \times 398 \times 32$	$(3 \times 3 \times 32) \times 64$	$396 \times 396 \times 64$

32 channel signifies ?? Are they RGB ??

....

Notice that our kernels in 2nd layer has 32 channels.

Note:

① Our kernel must have an equal number of channels as the input channel.
because each channel in the kernel will look only at 1 channel. (say channel # 23)
(channel no. 23 in input)

② If we have ∞ number of channels in the input, our kernels must have ∞ channels.
This has nothing to do with output channels.

③ Output channels are equal to Numer of kernels we use.

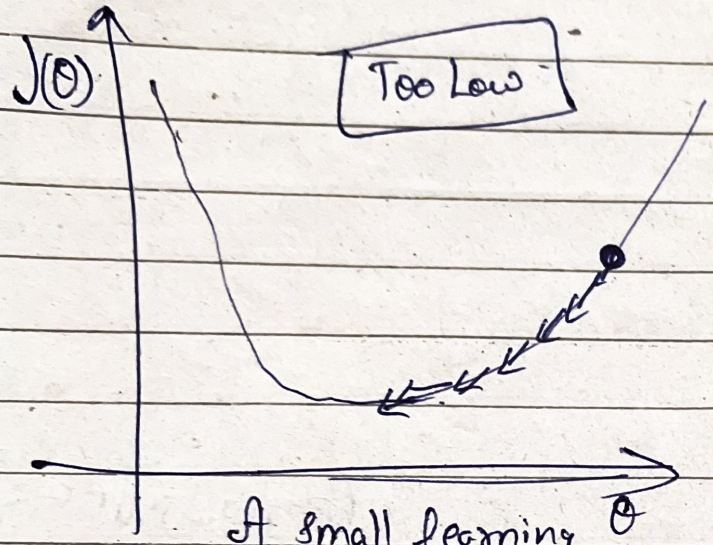
Receptive Field.

The receptive field is defined as the region in the input space that a particular CNN's feature is looking at. (i.e. affected by).

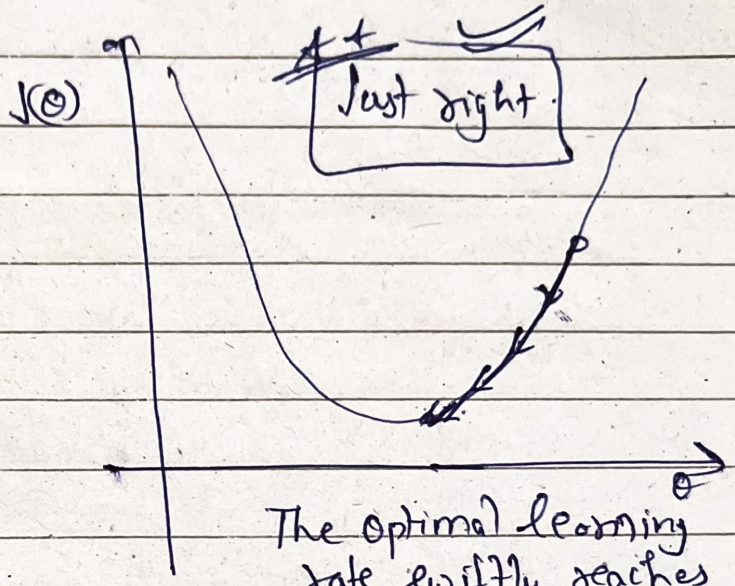
A RF of a feature can be described by its: ~~center~~.

1. center location
2. size.

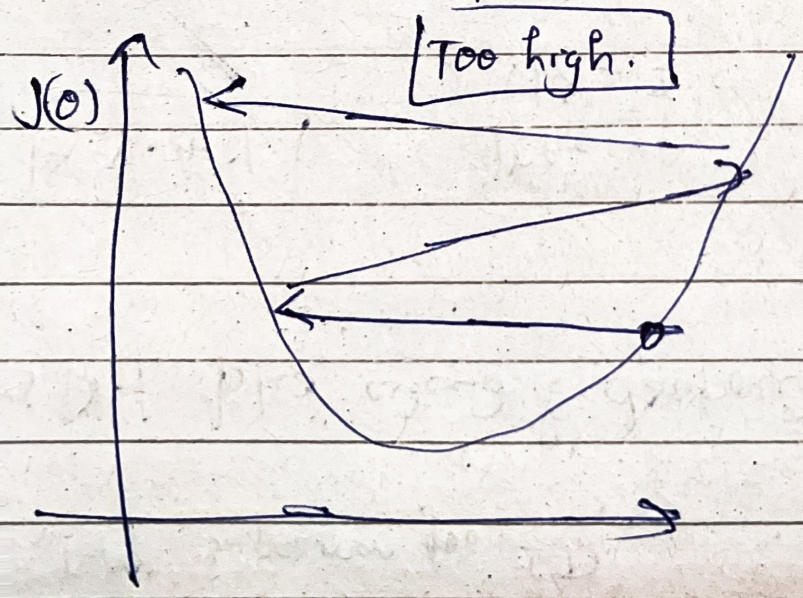
Learning Rate.



A small learning rate requires many updates before reaching minimum point.



The optimal learning rate swiftly reaches the minimum point.



Too large of a learning rate causes drastic updates which leads to divergent behaviour.

```
optimizer = optim.SGD(model.parameters(), lr=0.01,
```