-20-2021 Monday

Lecture#16 # Lecture#21

CNN

UpSampling:

UpSampling layer, to upsample input image size

2x2 filter. is used

1	2
3	4

filter 2x2

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

output

Replication so problem is a replication/redundancy

So we use transposed convolution

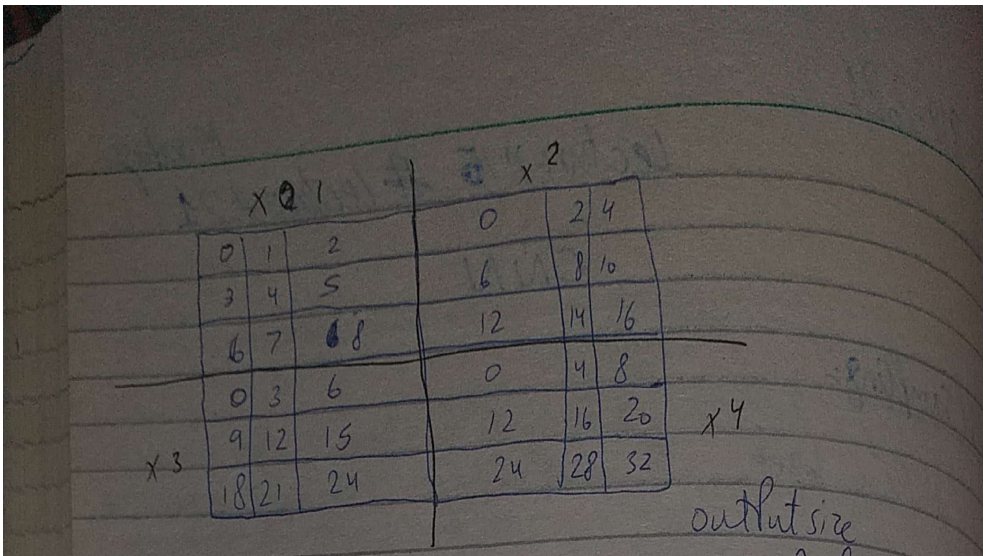
Transposed Convolution / Deconvolution / Inverse Convolution:

1	2
3	4

image 2x2

0	1	2
3	4	5
6	7	6

filter 3x3



output size = 2 × filter size

stride more, output size small.

we can also add Padding in our output

we ~~first~~ add Padding after we get our output from this Deconvolution.

Transfer Learning.

Different architectures for Transfer Learning

Alexnet

VGGnet Vggnet

Googlenet

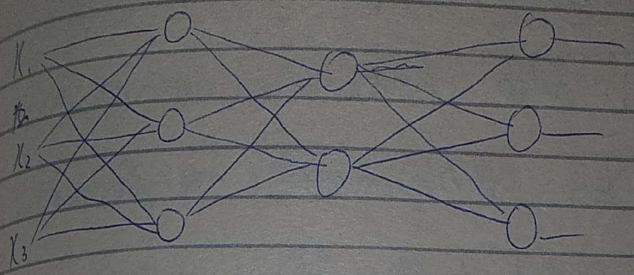
resnet

different no of layers & did different steps in training

imagenet is a library of thousands of pictures & classes (labels)

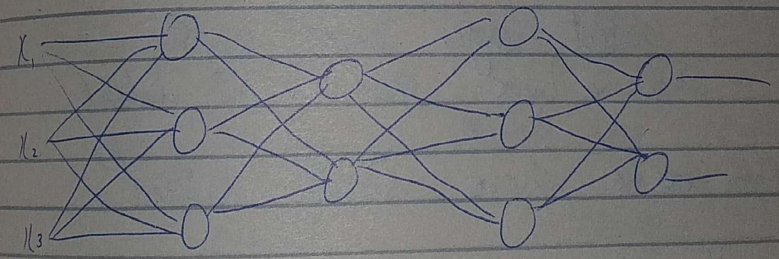
before the resnet model

for example:



resnet for example

So we change it to give 2 labels at the end



add another layer with 2 neurons.

but problem is that we would have probabilities in last layer before our added layers.

So it would change our output.

we could do is remove last neuron i.e 3 & neurons in last layer & we only have 2 neurons Another Problem is input could also be different so we add an input layer.

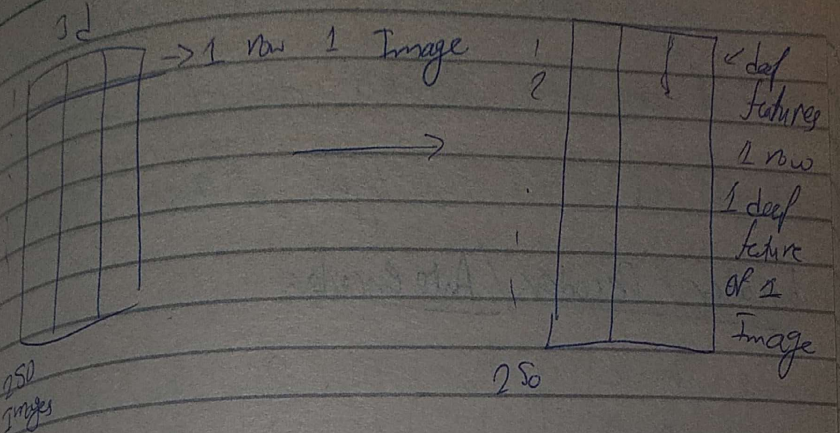
we should also have low learning rate in fine tuning. bcz this model was pretrained. and in our scratch implementation we had high learning rate & we eventually lower it.

Fine tuning:

① Input & output layer our own
when we have good & big dataset.

② when we have small dataset
we will remove last layer & use output of previous layer before last layer & use these outputs as features
use these features as in training
as input for a small neural network of 1 or 2 layers.

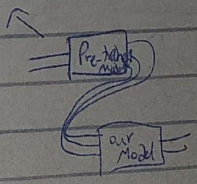
use resnet by default for Transfer learning.



we will reduce dimensionality & we have extracted features from image & discarded the original Image.

we didn't use small dataset for a big architecture to prevent overfitting on this dataset

In this case, we use 2 models. one our resnet, one our own model.



when to use Resnet?

use it when dataset is similar or for same task.

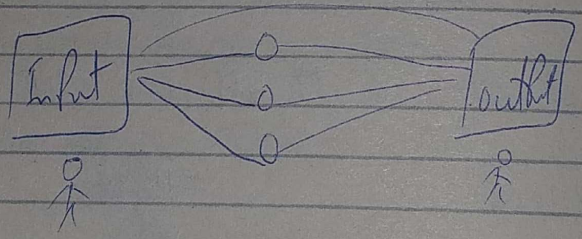
CNN usually learn first few layers ← high level features edges, circles, lines vertical, horizontal lines

then after that the layers learn specific part of the Google dataset

Then layers learn dataset specific features.
So if we have different dataset, then we should use first few layers of these Models.

Encoder / Decoder / Auto encoder

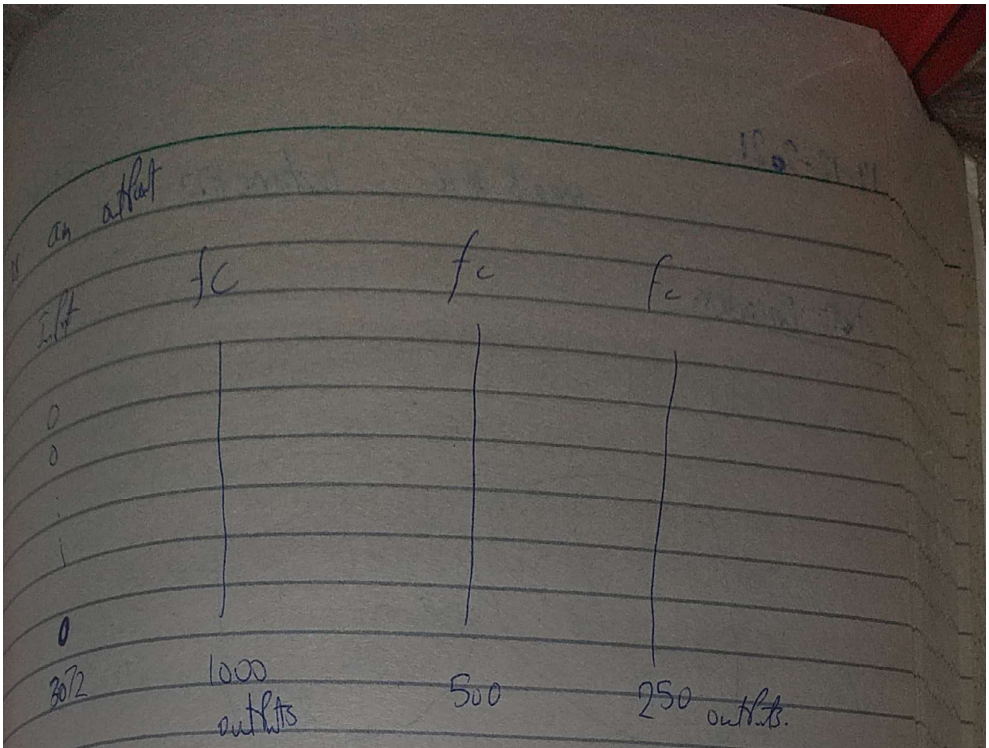
a	b	c	D_0	D_1	D_2	D_3	D_4 D_5 D_6 D_7
0	0	0	1	0	0	0	0 0 0 0
0	0	1	0	1	0	0	0 0 0 0
0	1	0	0	0	1	0	0 0 0 0
0	1	1	0	0	0	1	0 0 0 0
1	0	0	0	0	0	0	1 0 0 0



Unsupervised learning

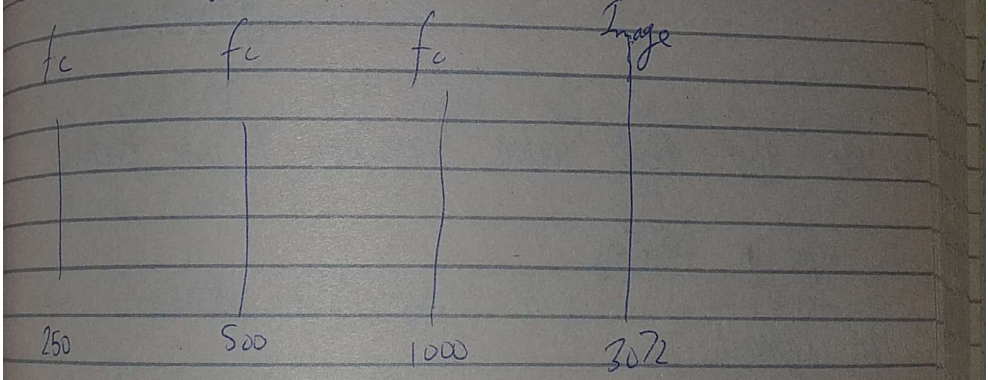
No labels.

Input images, then net try to generate same output



Convert big image to small image

After 250 layer we do



Input is same output

We could use this in transfer learning, the 250 outputs as features of original image. This is autoencoder for feature extraction. last layer called bottleneck