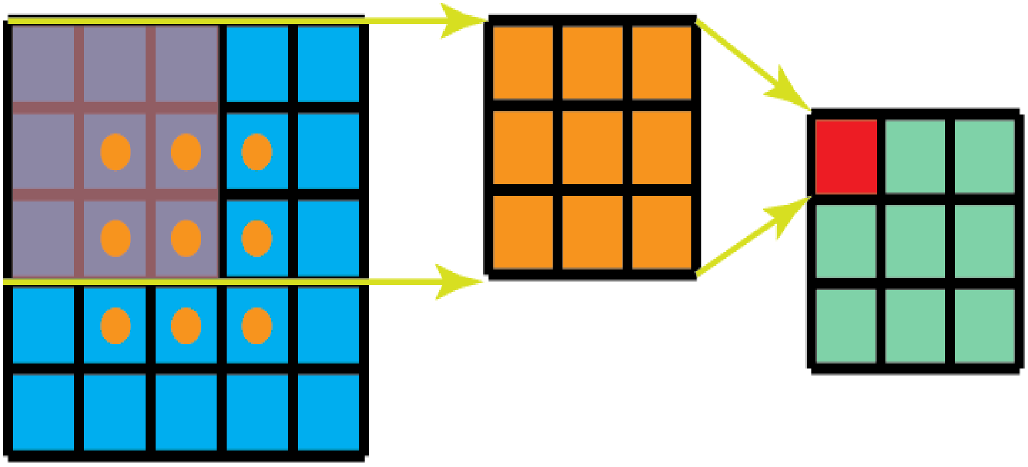
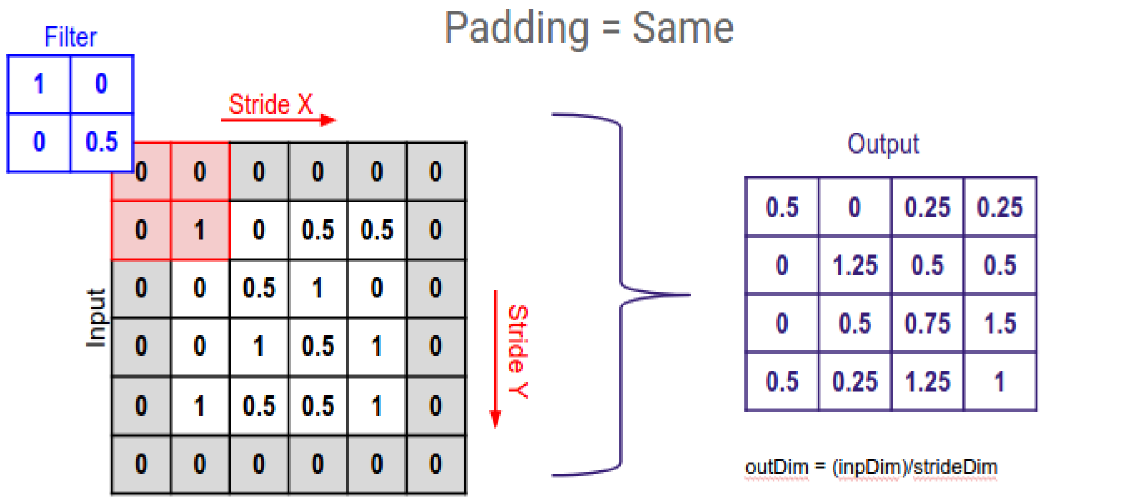
Convolutional Neural Networks

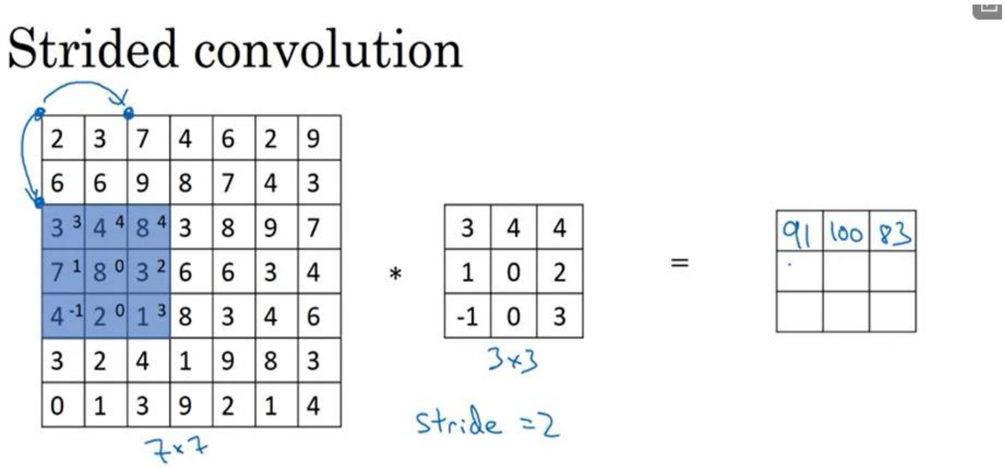
Week 1:

1. Convolution – ft.nn.conv2d



2. Padding - it's an additional layer that we can add to the border of an image.For an example see the figure below there one more layer added to the 4*4 image and now it has converted in to 5*5 image

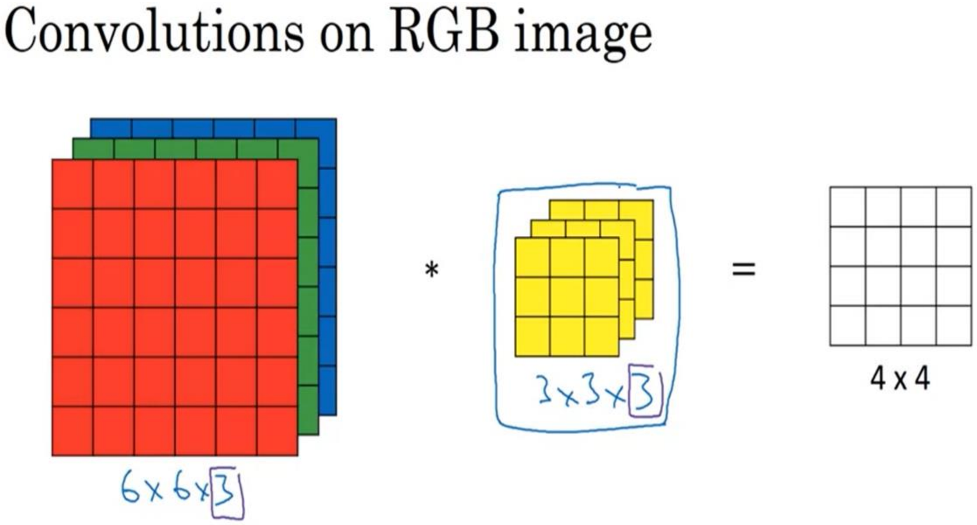




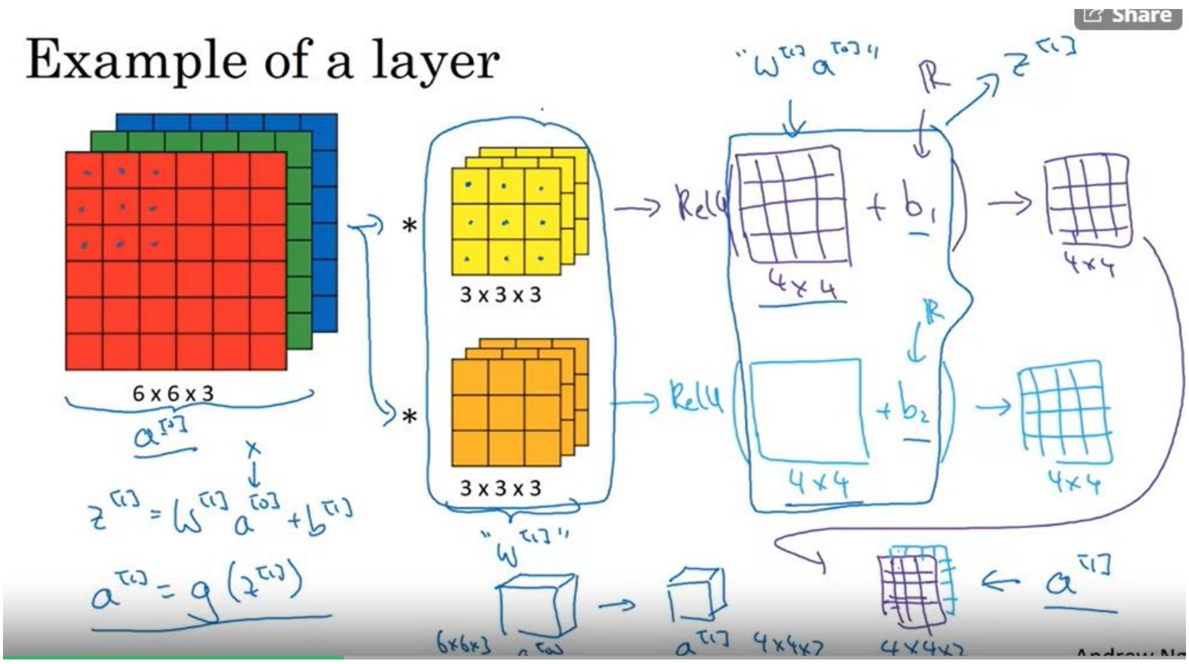
3. Strided:

a. size = (n+2p-f)/s+1 x (n+2p-f)/s+1
b. If size is not and integer then use math.floor

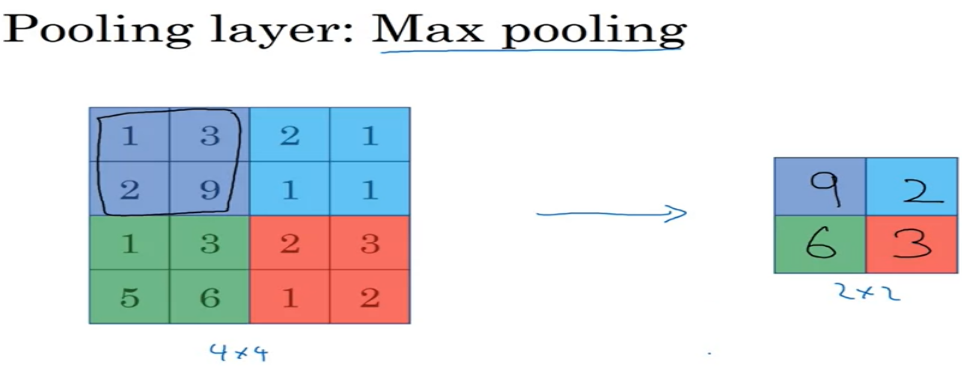
4. Convolution on RGB image:

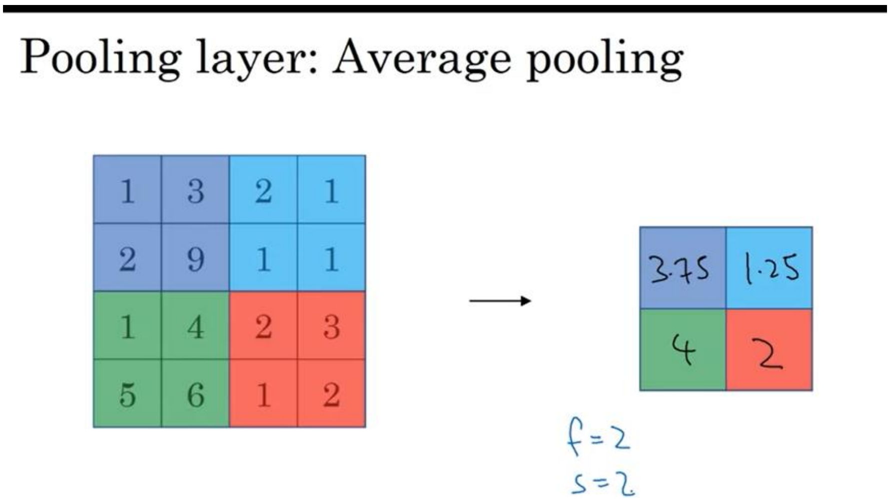


5. Example of conv net:



6. Pooling layer- Other than convolutional layers, ConvNets often also use pooling layers to reduce the size of the representation, to speed the computation, as well as make some of the features that detects a bit more robust.





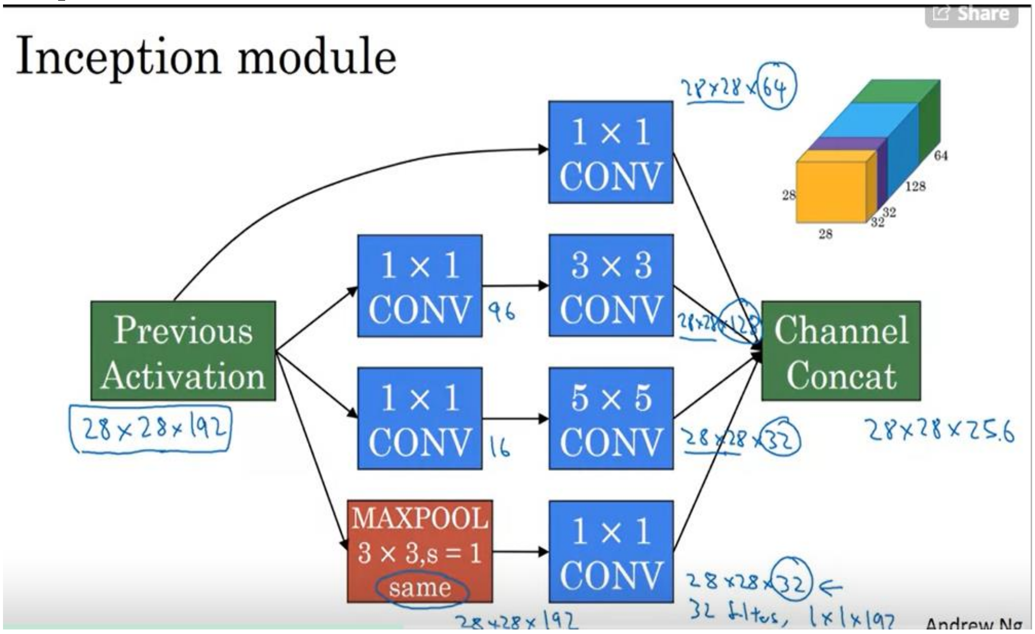
7.
8. Parameter sharing – a feature detector, that's useful in one part of the image is probably useful in another part of the image
9. Sparsity of connections – in each layer, each output value depends only on a small number of inputs

Week 2

10. LeNet-5:
    a. In a first step use 5x5 filter with a 1 stride
    b. Then applies average pooling
    c. Then apply Convolutions 5x5x16
    d. Then applies average pooling
    e. Then FullyConnected values
    f. Then another FC
    g. Then softmax or sigmoid
11. AlexNet:
    a. Input starts with 227x227x3 images
    b. Then apply 11x11x96 ConvolutionNet filter with strider = 4
    c. Then apply max pooling 3x3
    d. Then perfoms convolution with padding 5x5x256
    e. Then maxpool 3x3
    f. Then convolution net 3x3x384 with padding(same) (3 times)
    g. Then maxpool 3x3
    h. Then Fullyconnected layer (3 times)
    i. Then apply softmax
12. VGG – 16
    a. Start with 224x224x3
    b. Apply convolution with padding 3x3x64 (times 2)
    c. Then uses pooling layer
    d. Apply conv layers with padding 3x3x128 (time2)
    e. Apply pool
    f. Apply conv with padding 3x3x256 (times3)
    g. Apply pooling
    h. Apply conv with padding 3x3x512 (times3)
    i. Apply pooling
    j. Apply conv with padding 3x3x512 (times 3)
    k. Apply pooling layer

l. Fc (times 2)
m. Apply softmax

13. Inception Network:



Week 3

14. Object detection:
    a. First create a label training set, also you can cut and crop out anything else except the object
    b. Then you can create a convnet and inputs an image
    c. If you have a test image like this what you do is you start by picking a certain window size, shown down there and then you would input into this convnet a small rectangular region.
15. Implementation of convolutional sliding image: You can implement the entire image and colvolutionally make all the predictions at the same time.
16. Intersection over union computes the size of intersection devided by size of unioun. If IoU >= 0.5 'correct'
17. Non-max suppression algorithm:
    a. Discard all boxes with Pc <= 0.6(chance that it is an object)
    b. While there are any remaining boxes:
        i. Pick the box with the largest Pc and output that as a prediction
        ii. Discard any box with IoU >= 0.5 with the box output in a previous step
18. Anchor box algorithm:
    a. Each object in training image is assigned to grid cell that contains that objects midpoint
    b. With two anchor boxes: each object in training image is assigned to grid cell that contains objects midpoint and anchor box for the grid cell with highest IoU
19. Bounding box prediction:
    a. For each of grid you specify your label Y

Week 4:

20. One shot learning – learning from one example to recognize object again
21. Learning similarity function – d(img1, img2) = degree of difference between images

22. To train on triplet loss, you need to take your training set and map it to a lot of triples. So, here is our triple with an anchor and a positive, both for the same person and the negative of a different person. Here's another one where the anchor and positive are of the same person but the anchor and negative are of different persons and so on. And what you do having defined this training sets of anchor positive and negative triples is use gradient descent to try to minimize the cost function J \
23. Style cost function

Style cost function

$$J^{[l]}_{style}(S,G) = \frac{1}{\left(2n_H^{[l]}n_W^{[l]}n_C^{[l]}\right)^2}\sum_{k}\sum_{k'}(G^{[l](S)}_{kk'} - G^{[l](G)}_{kk'})$$

24. Neural Style Transfer is an algorithm that given a content image C and a style image S can generate an artistic image
25. It uses representations (hidden layer activations) based on a pretrained ConvNet.
26. The content cost function is computed using one hidden layer's activations.
27. The style cost function for one layer is computed using the Gram matrix of that layer's activations. The overall style cost function is obtained using several hidden layers.
28. Optimizing the total cost function results in synthesizing new images