Contents

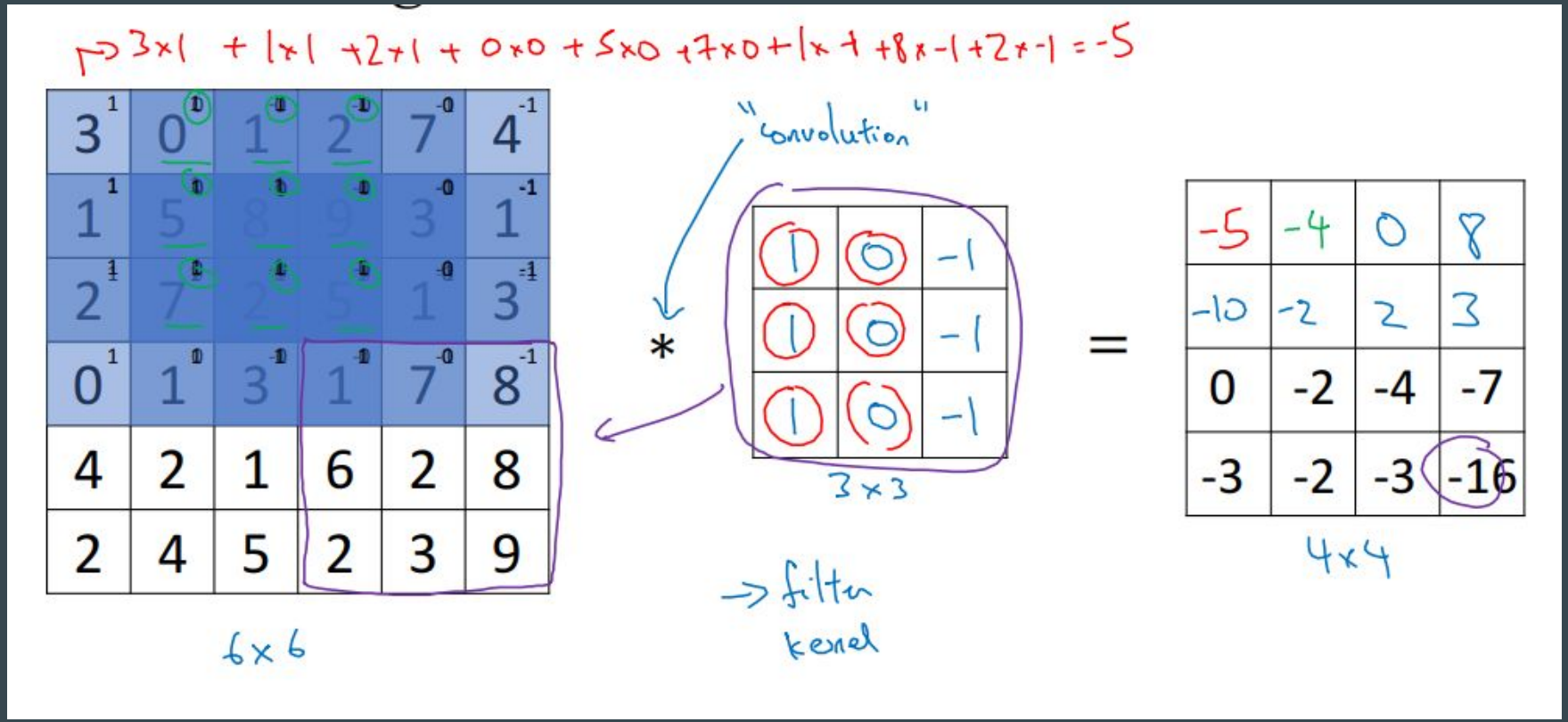
Representation of 2D images

- These 2D images are represented in matrix.
- dimensions=(n_h,n_w,n_c)
- n_h=height of image
- n_w=width of image
- n_c=number of channels
- Single channel image: n_c=1

Convolution

- We convolve an image using a kernel /filter.
- Convolve means
  - Place filter over a region of image
  - Multiply corresponding elements of image and filter.
  - And add it.
  - This value is O_{ij}.
- Parameters
  - Stride
  - Padding
  - Filter Size
  - Number of filters

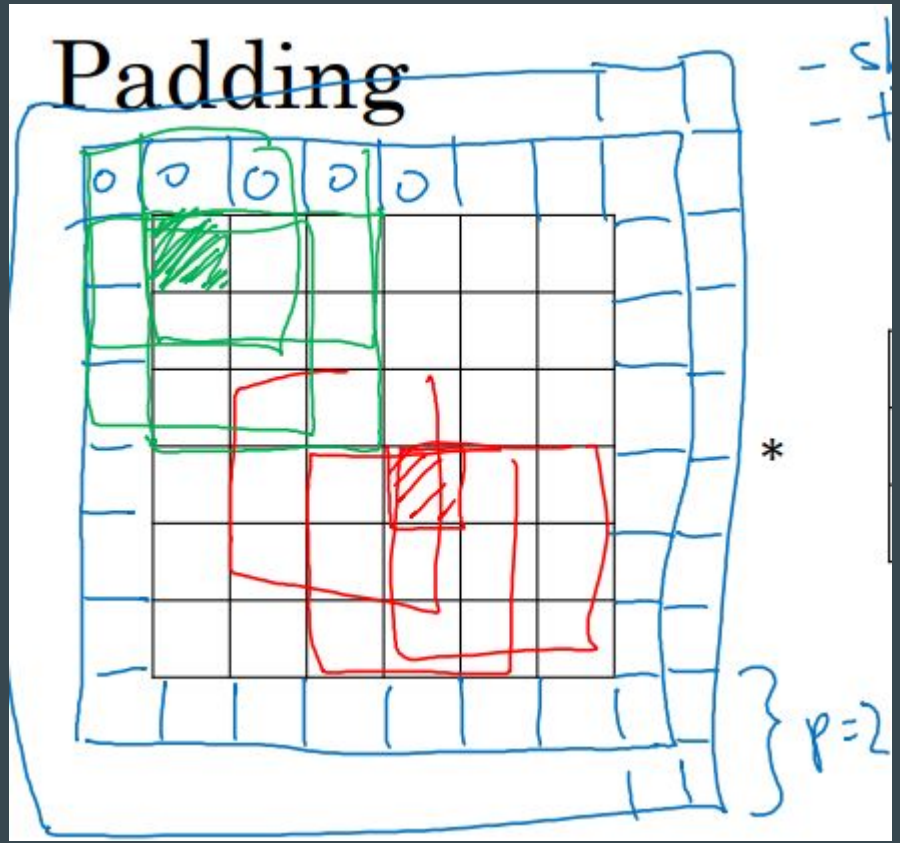
Convolution



Basic Convolution

- $n^{l}_{h/w} = (n_{h/w} - f) + 1$
- In the given example
  - $n_{h/w} = 6$
  - $f = 3$
  - $n^{l} = (6-3)+1 = 4$

Padding

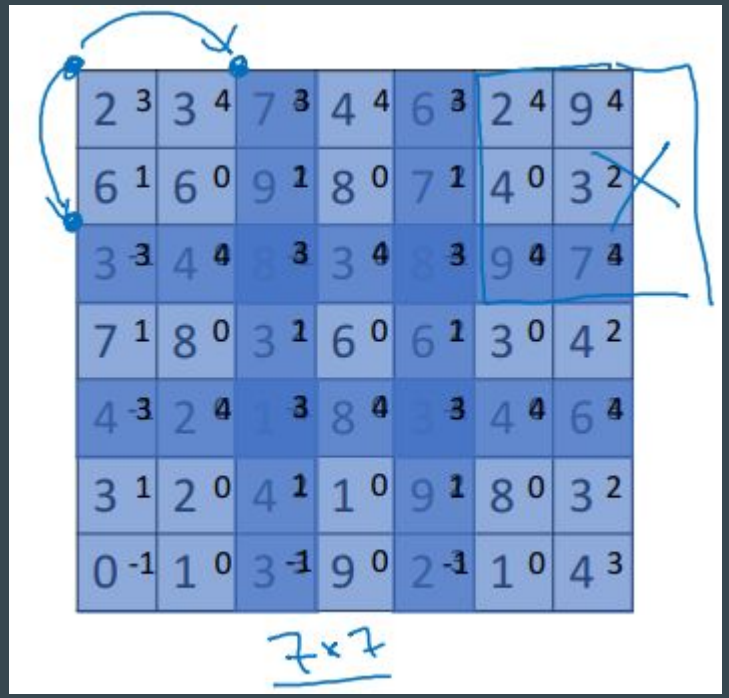


- Add extra boundary cells to the image
- For 1 layer of padding 2 cells are added to each row and column.
- We do to make input and output dimension to be same
- And a boundary pixels get involved in convolution less number of times compared to center pixels.

Padding

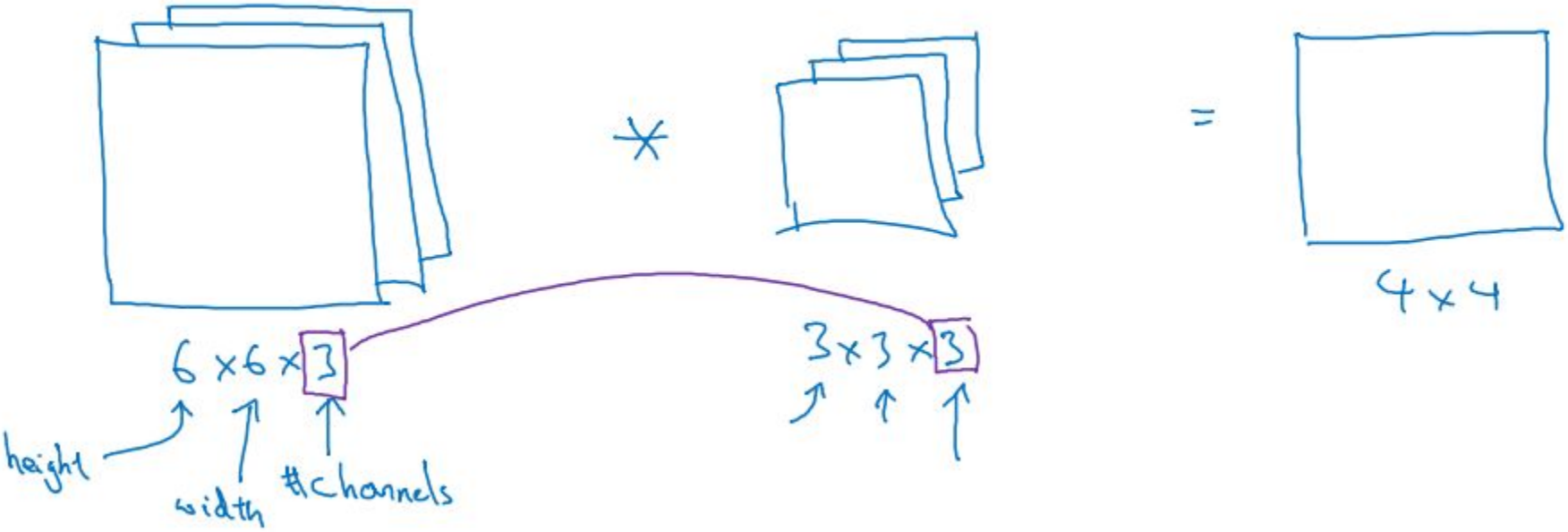
- Valid Convolution: No padding
- Same Convolution: Padding is done to make input size equals to output size.
  - $n+2p-f+1=n$
  - $p=(f-1)/2$

Stride



- We skip some possibilities of convolution.
- $n^{l}_{h/w}=(n_{h/w}+2p-f)/s+1$
- Example:
- stride=2
- Filter size=3
- 7*7---------->(7-3)/2+1=(4*4)

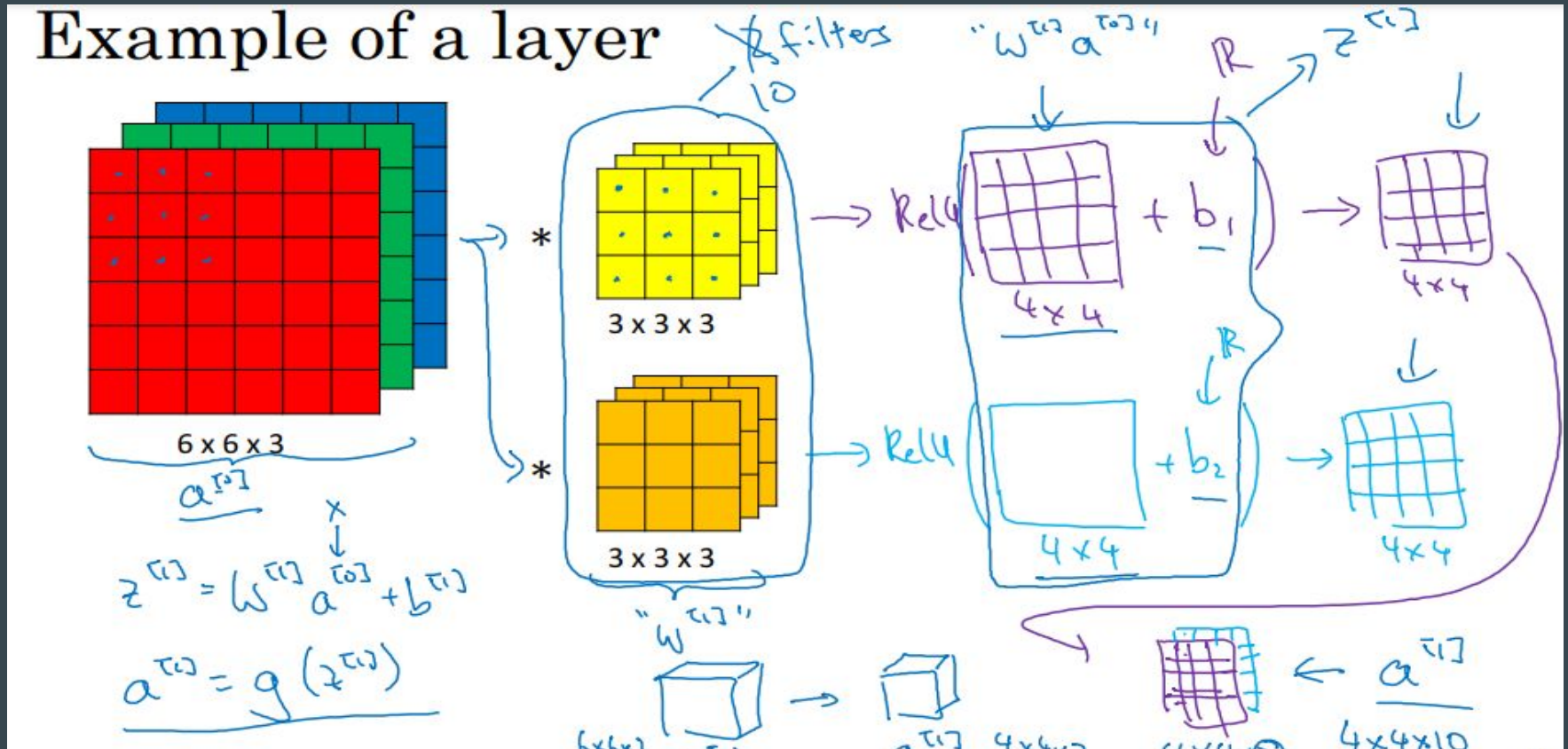
Multi Channel CNN



Multi Channel CNN

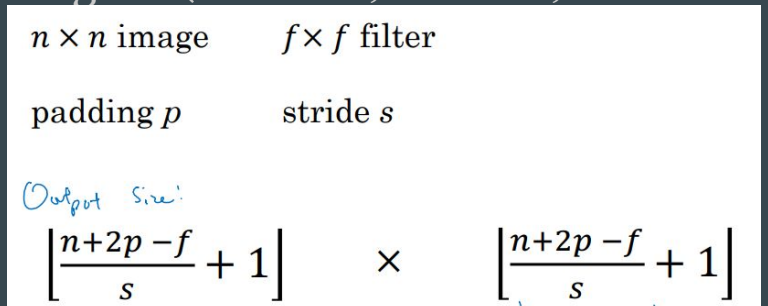
- Number of Filter channels must be equal to Number of Input channels.
- Single Filter produce a 2D feature map.
- O_{ij}=over a region(conv($img_1 k_1$)+conv(img_2,k_2)+con(img_3,k_3)).
- Multiple Filters produce multiple feature maps.
- Each feature map is added with it's bias and applied non linearity function.
- And stack them and it's another conv layer.
- $n^{'}$=(n+2p-f)/s+1
- Dimensions of output of convolution: ($n^{'}_h$,$n^{'}_w$,n_c).
- n_c=Number of filters in that layer.

Multi Channel CNN



Summary of Notations

- Filter Size : $F^{[l]}$
- Padding : $P^{[l]}$
- Stride : $S^{[l]}$
- Number of Filters: $n_c^{[l]}$
- Input : $(n_h^{[l-1]}, n_w^{[l-1]}, n_c^{[l-1]})$
- Output : $(n_h^{[l]}, n_w^{[l]}, n_c^{[l]})$
- Each Filter : $(F^{[l]}, F^{[l]}, n_c^{[l-1]})$
- Weights: (filtersize,filtersize,number of channels,Number of filters)
-

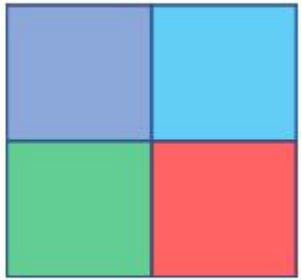


Pooling Layer

- Commonly used max pooling.
- Used to
  - Reduce the size of representation.
  - Detect features effectively.
- This nothing but a filter over a region and taking value as max in that region.
- Hyperparameters of max pooling:
  - Filter size
  - Stride
  - Padding
- No Parameters to learn.
- Commonly used Hyperparams (f,s) : (2,2),(3,2)

Pooling

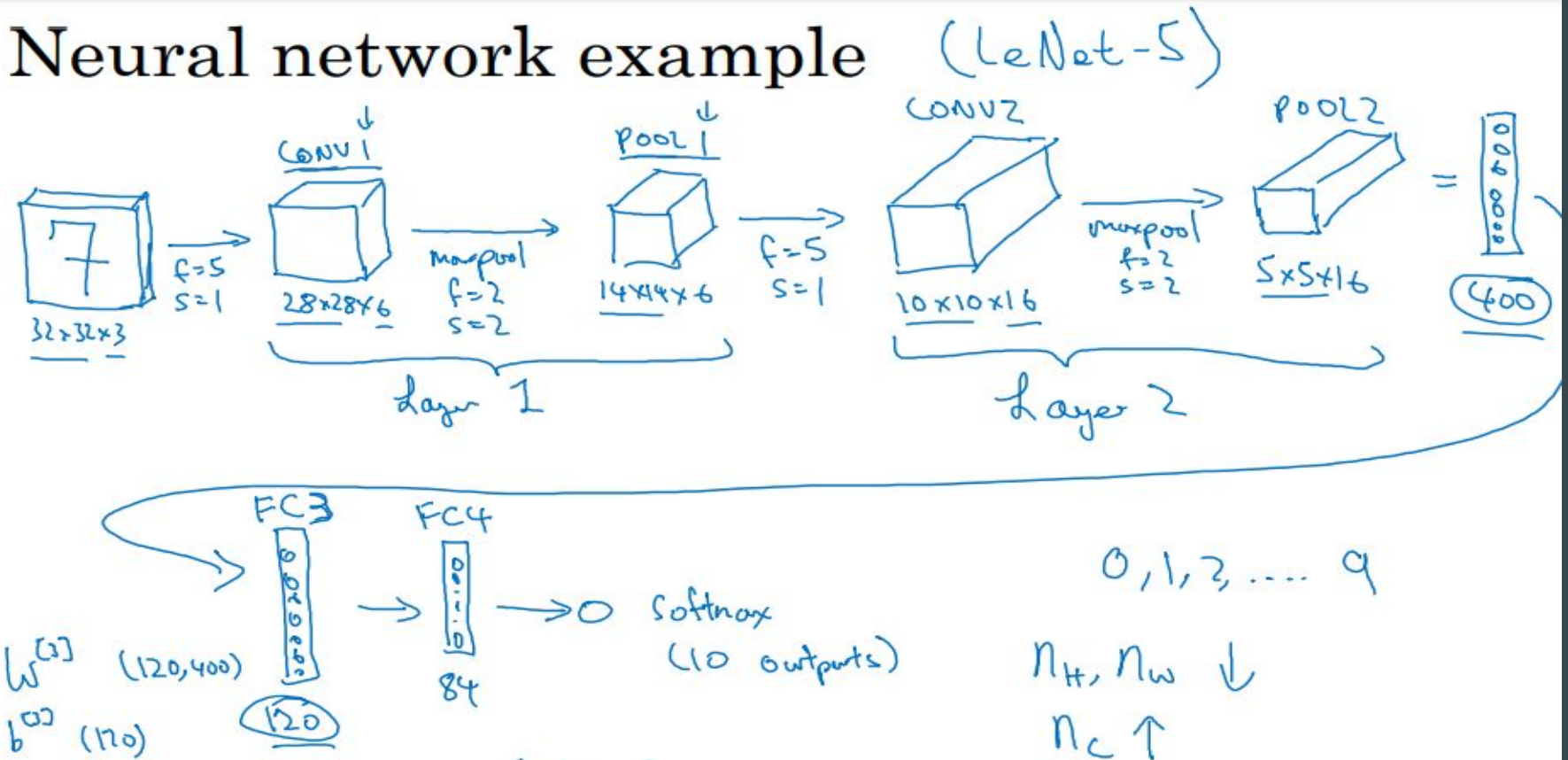
1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2



Advantages of CNN over Traditional NN

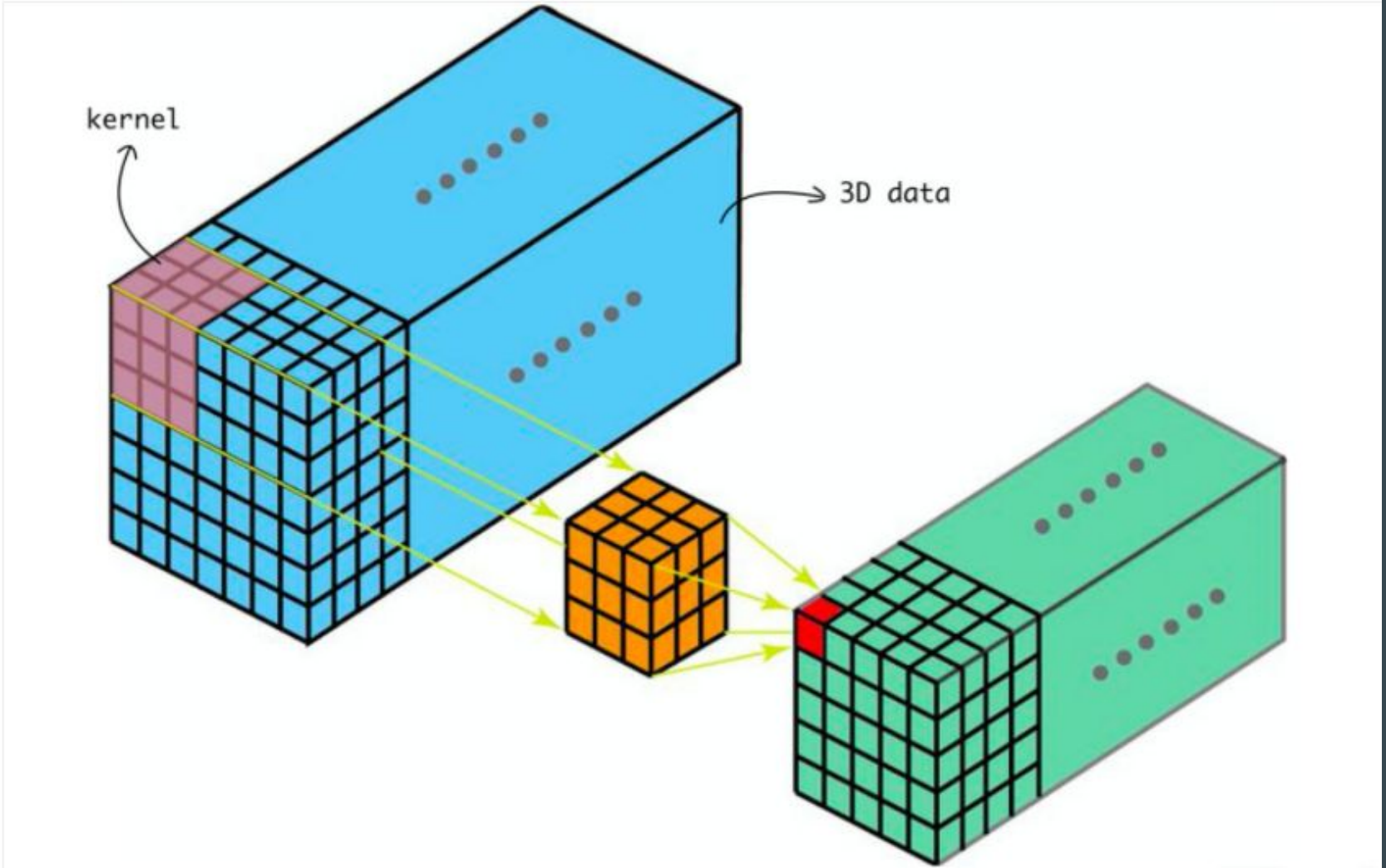
- Parameter Sharing: same parameters are used to for all regions
- Sparsity of Connections : each output node depend only on few nodes of input layer instead of depending on all nodes.

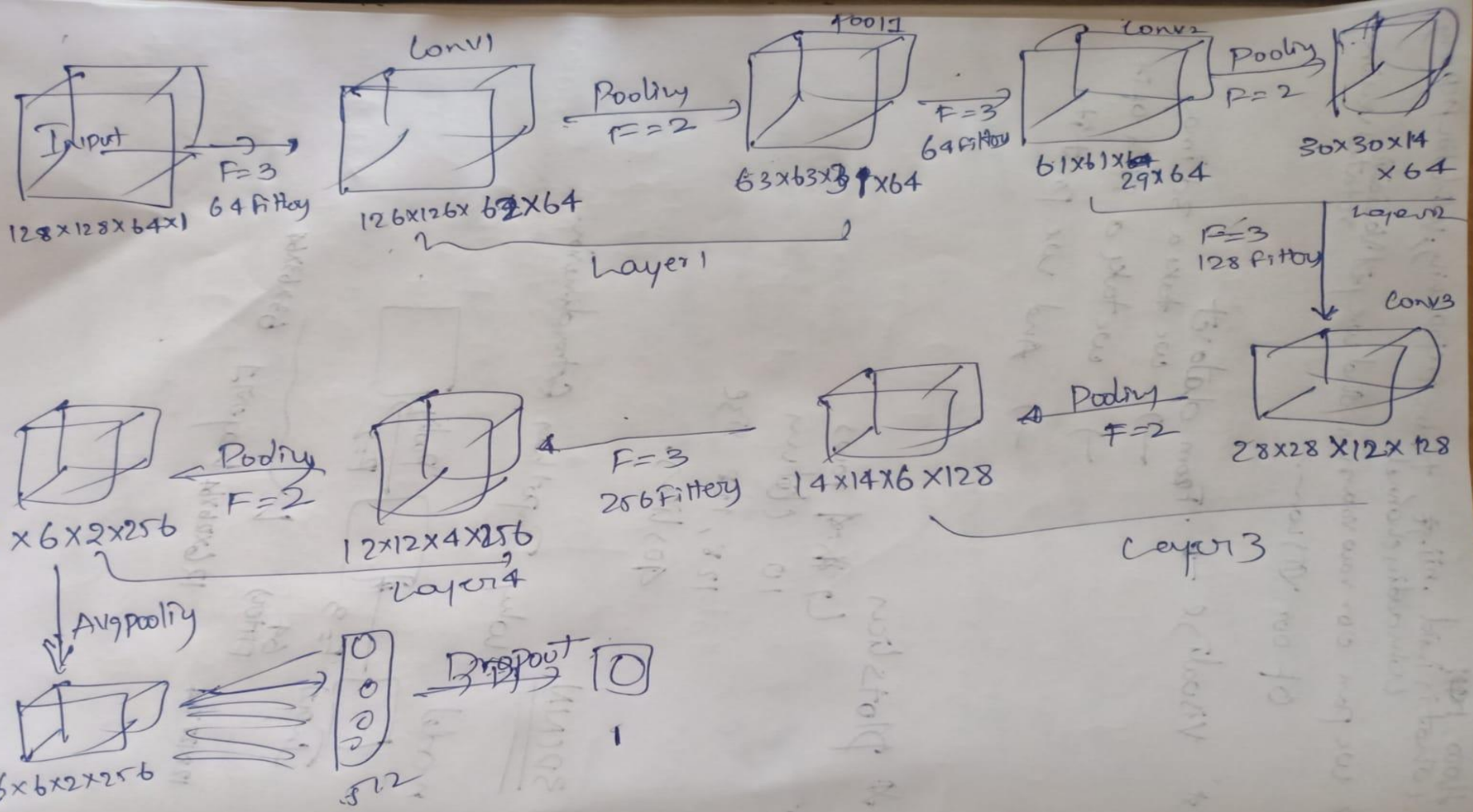
Example Model



3D CNN (Single Channel)

- Each channel consists of 3D data.
- Input Image : (height,width,depth,# of in_channels)
- Kernel size : (F,F,depth',# of in_channels)
- Weights size : (F,F,depth',# of in_channels,# of filters)
- Output size : (height',width',depth',# of filters)
- Mostly height and width are same but depth may not be.
- Same for kernel also
- Convolution Process:
  - Left to right
  - Top to Bottom
  - Go in
- Output of Each filter is 3D data.


kernel
3D data


Input
128x128x64x1
F=3
64 Filters
Conv1
126x126x 62 X64
Pooling
F=2
Pooling
63x63x 31 X64
F=3
64 Filters
Conv2
61x61x64
29x64
Pooling
F=2
30x30x14
x64
Layer1
Layer2
F=3
128 Filters
Conv3
28x28 X12X 128
Pooling
F=2
14x14x6 X128
F=3
256 Filters
12x12x4X256
Pooling
F=2
6x6x2x256
Layer4
Layer3
AvgPooling
6x6x2x256
512
Dropout
1