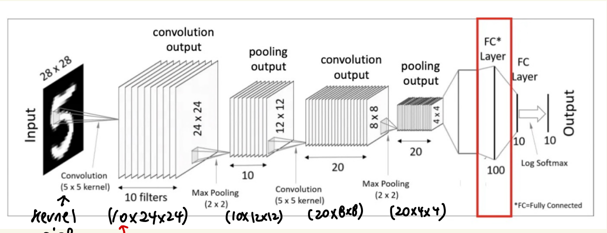
CNN Structure



kernel size.   (10x24x24)   (10x12x12)   (20x8x8)   (20x4x4)

"channels"

*RGB has 3 Channels ex) 3x64x64

*Greyscale has 1 channel. ex) 1x28x28

* Padding:

1	1	1
0	1	1
0	0	1

→

0	0	0	0	0
0	1	1	1	0
0	0	1	1	0
0	0	0	1	0
0	0	0	0	0

Why?

if input data is $\mathbb{R}^{5\times5}$, and apply kernel 3x3,

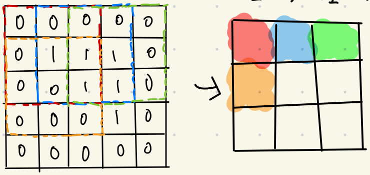
$\mathbb{R}^{5\times5} \to \mathbb{R}^{3\times3}$

After padding to data, $\mathbb{R}^{5\times5} \to \mathbb{R}^{7\times7}$, and when we apply 5x5 kernel, $\mathbb{R}^{7\times7} \to \mathbb{R}^{5\times5}$.

It prevents loosing information

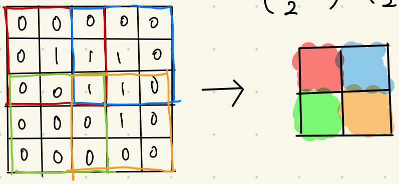
* Stride: How far the filter moves in every step along

Default: 1 $\left(\frac{5-3}{1}+1\right)\times\left(\frac{5-3}{1}+1\right)=3\times3$



0	0	0	0	0
0	1	1	1	0
0	0	1	1	0
0	0	0	1	0
0	0	0	0	0

Stride = 2 $\left(\frac{5-3}{2}+1\right)\times\left(\frac{5-3}{2}+1\right)=2\times2$



0	0	0	0	0
0	1	1	1	0
0	0	1	1	0
0	0	0	1	0
0	0	0	0	0

*Dimension: $\left(\frac{d_1-k_1}{s}\right)+1 \times \left(\frac{d_2-k_2}{s}\right)+1$

* Convolutional layer: Convolution + (Normalize) + Activation

Optional

* Pooling: We can subsample the pixels to make image smaller with fewer parameters to characterize the image

e.g.: Max Pooling

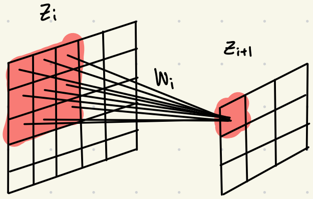
12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

2x2 Max Pooling →

20	30
112	37

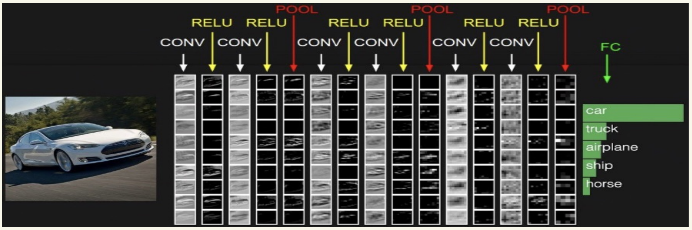
Code example w/ PyTorch :

```
nn.Sequential(
    nn.Conv2d(input_channel, output_channel, kernel_size, stride, padding, bias=False),),
    nn.BatchNorm2d(output_size),
    nn.LeakyReLU(0.2),
)
```



☆ Convolution is a linear operator

Hence, we need to follow Convolution with a non linearity (e.g. ReLU) to get nonlinear functions


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse