Convolutional Neural Networks

What are CNNs?

- A special type of ANNs which expects images as inputs
- Like ANNs they too have neurons/activations and weights (estimated during training) and an objective/loss function
- Since the input is images it allows for modifications to the architecture so that
  - There are sparse connections between layers
  - Weights are shared between output neurons in the hidden layer

Applications in Computer Vision & Image Analysis

- Image Recognition
- Object Detection and Localization
- Semantic segmentation
- Medical Image Analysis

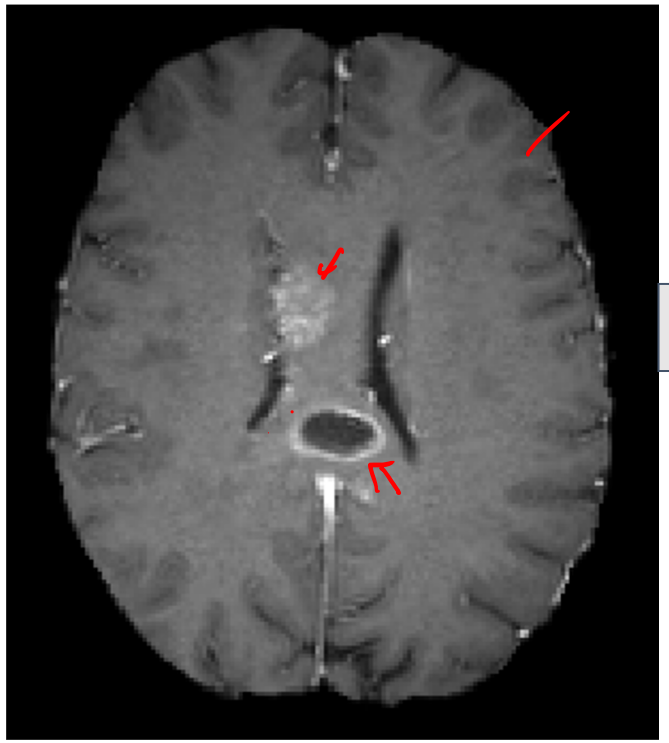
Applications in Image Analysis

Image Analysis
- Object Detection
- Object Recognition
- Instance Segmentation

Herd 91%
Sheep 82%
Pasture 80%
Goats 73%
Livestock 71%
Herding 69%
Herder 62%
Grass 59%
Farm 50%

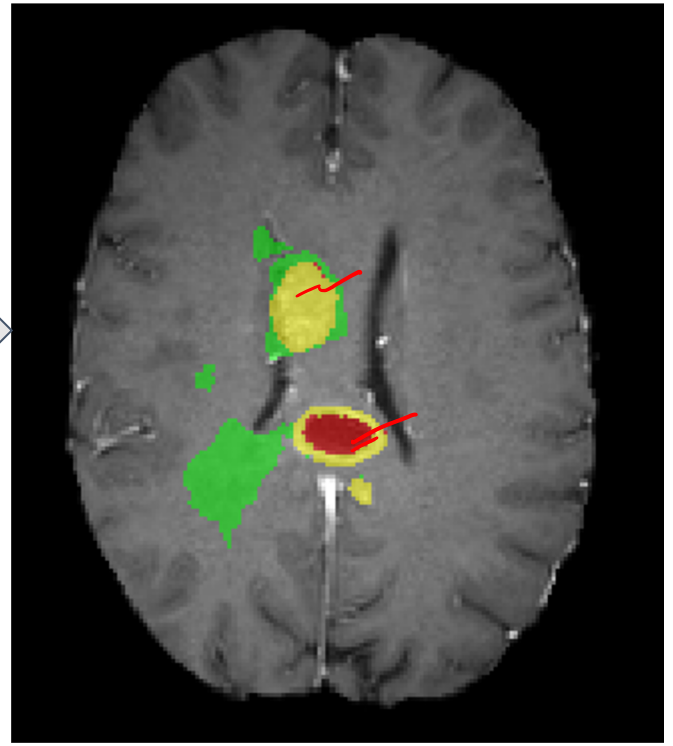
1) https://cloud.google.com/vision/
2) http://silverpond.com.au/object-detector.html

Applications in Image Analysis



Preprocessed MR Image of Brain

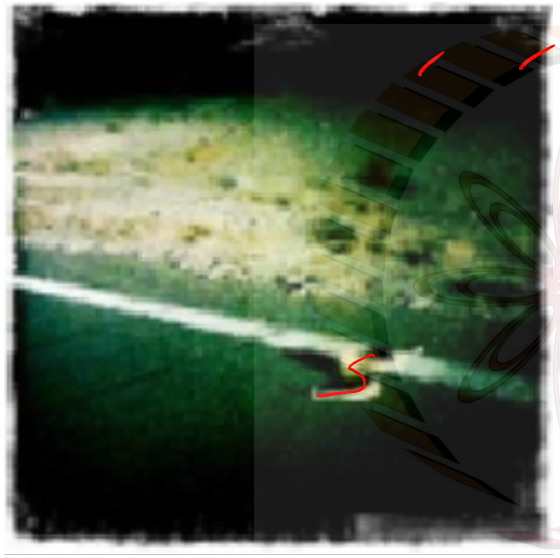
Brain Tumor Segmentation

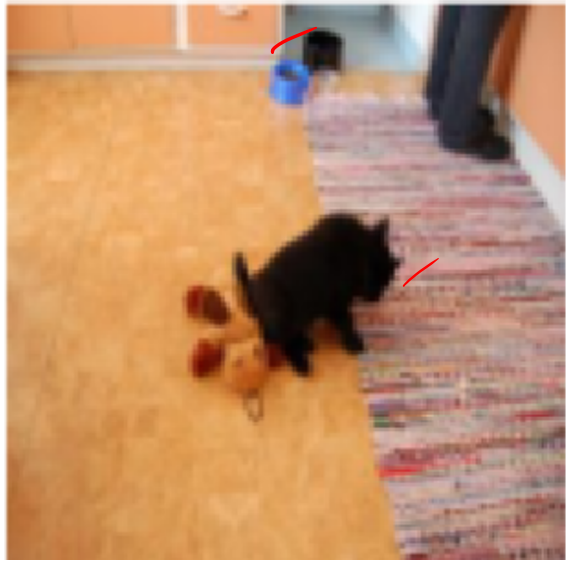


Tumor Segmentation

ImageNet Challenge

1000 classes, a million training images and report top-5 prediction accuracy. The top-5 error is now better than humans- Note humans are not trained! Human error: ~5.1%



rule, ruler	sidewinder	hatchet	schipperke
pencil box, pencil case	maze, labyrinth	vase	schipperke
rubber eraser, rubber	gar, garish	pitcher, ewer	groenendael
ballpoint, ballpoint pen	valley, vale	coffeepot	doormat, welcome mat
pencil sharpener	hammerhead	mask	teddy, teddy bear
carpenter's kit, tool kit	sea snake	cup	jigsaw puzzle

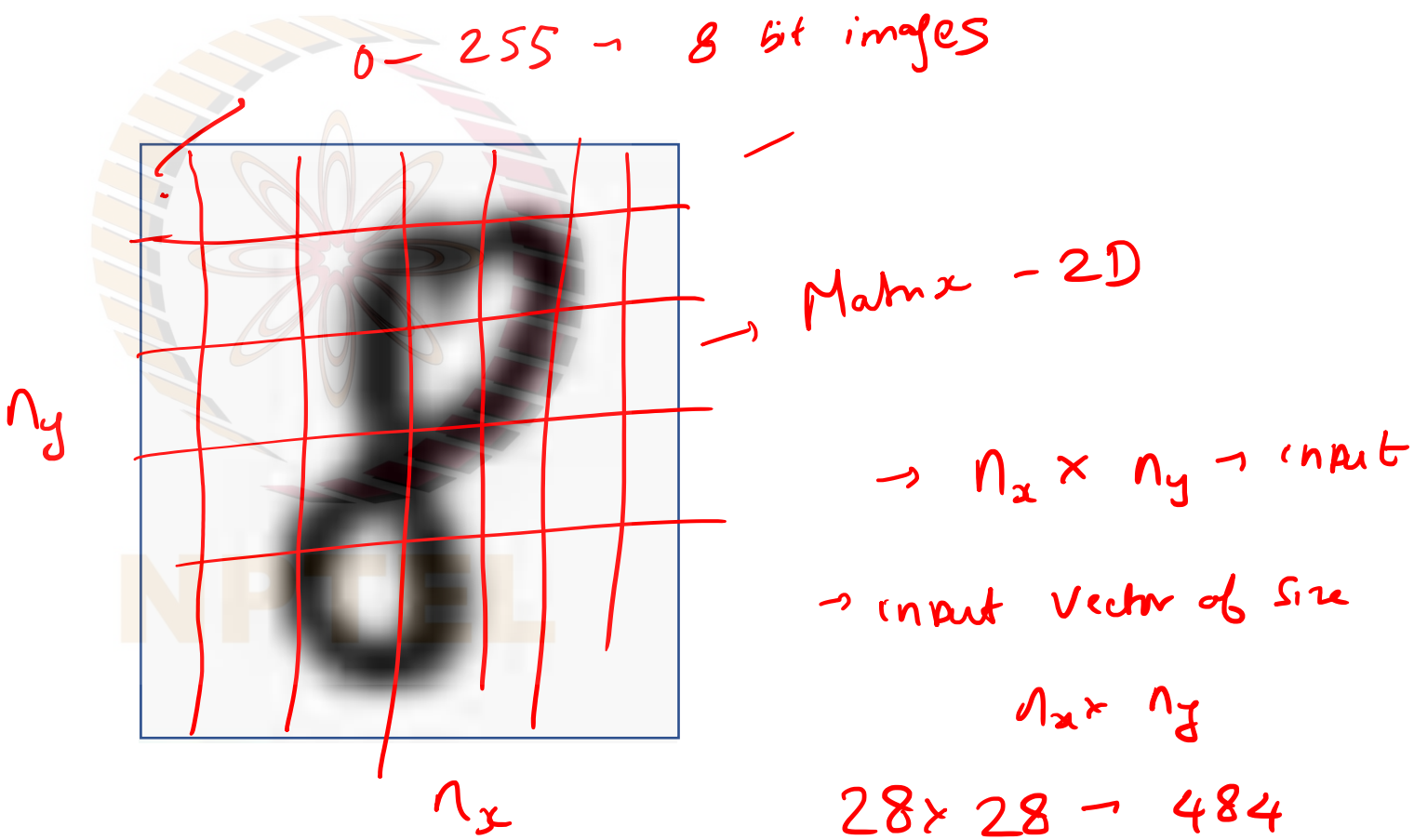
Progress in ImageNet challenge

Model	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	0.79	0.945	22910480	126
VGG16	0.715	0.901	138357544	23
VGG19	0.727	0.91	143667240	26
ResNet50	0.759	0.929	25636712	168
InceptionV3	0.788	0.944	23851784	159
InceptionResNetV2	0.804	0.953	55873736	572
MobileNet	0.665	0.871	4253864	88
DenseNet121	0.745	0.918	8062504	121
DenseNet169	0.759	0.928	14307880	169
DenseNet201	0.77	0.933	20242984	201

The top-1 and top-5 accuracy refers to the model's performance on the ImageNet validation dataset (Source: https://keras.io/applications/)

Image Parametrization

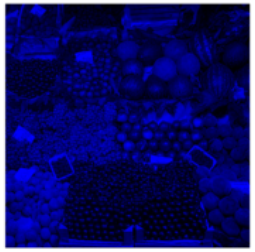
- Gray scale images



https://vimeo.com/127790185

Image Parametrization

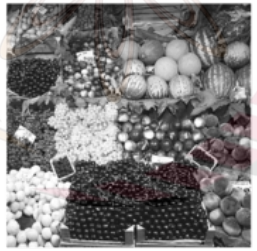
- RGB images

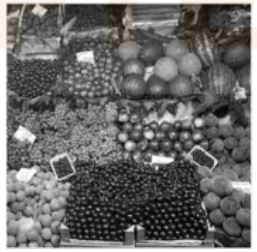


Red

Green

Blue





n_y

n_x

3- channels

$[n_x \times n_y \times 3]$ → input

Why not regular Neural Networks?

- ANNs take a vector of inputs and product as output another hidden layer vector fully connected to the input-
  - For small image sizes the number of weights/parameters to be estimated are not large – but consider a 224x224x3 image - RGB images.
  - A single neuron in the output layer will have 224x224x3 weights coming into it.
  - A 'Volume' image input like RGB images will lead to an explosion in the number of weights – Requires more memory, computations and data

- CNNs exploit the structure of images
  - leading to sparse connections between input and output neurons
  - parameter sharing between output neurons

$30 \times 30 \times 3 \rightarrow$ 2700 → input neurons

$[256 \times 256 \times 3) \rightarrow 10^5$ neurons

neurons

$\rightarrow 10^8 \rightarrow$ weights

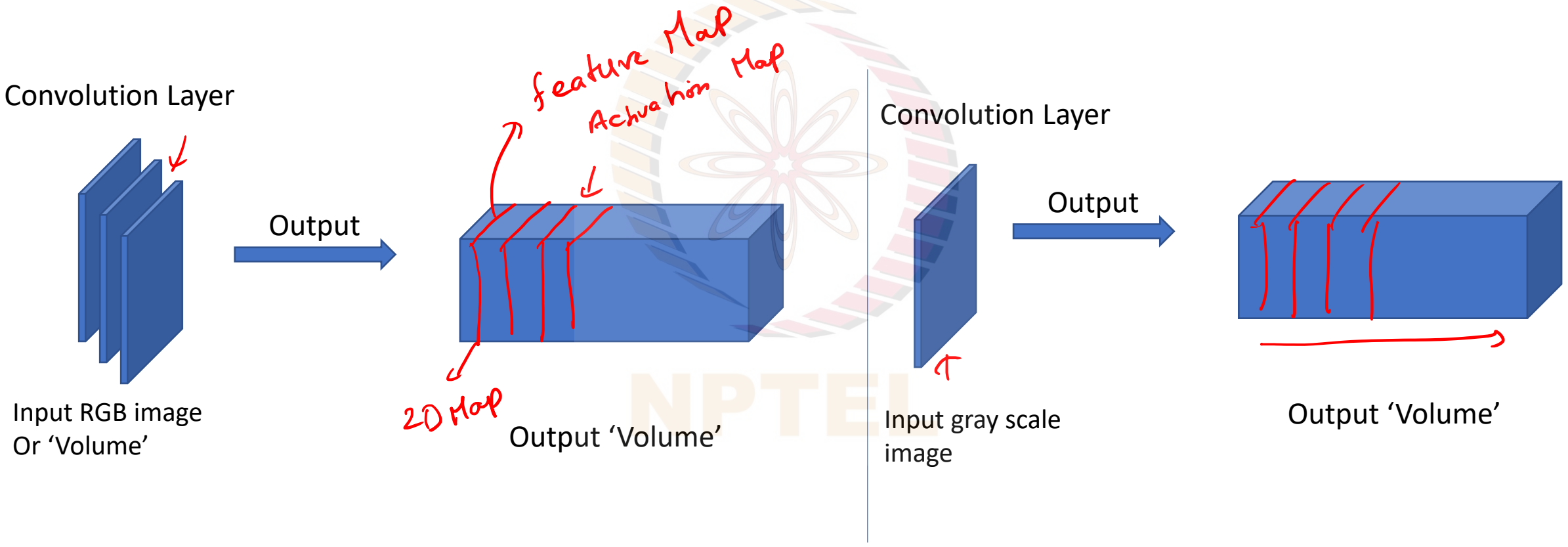
CNN layers and operations

- Like regular ANNs, CNNs stack a sequence of layers followed by an output layer – Classification layer
- Each layer performs a 'Convolution' operation (Conv Layer) or a 'Pooling' operation (Pooling Layer)
- The Conv Layer and Pooling Layers are alternatively stacked – leading to a series of fully connected layers followed by an output layer

CNN Layers & Operations

- CNNs take as input an image ‘Volume’ – For e.g. RGB volume or sub-volume of a 3D medical image- and each layer outputs another 3D volume following a convolution or pooling
- Contrast to ANN where the output from each layer is another vector of Neurons

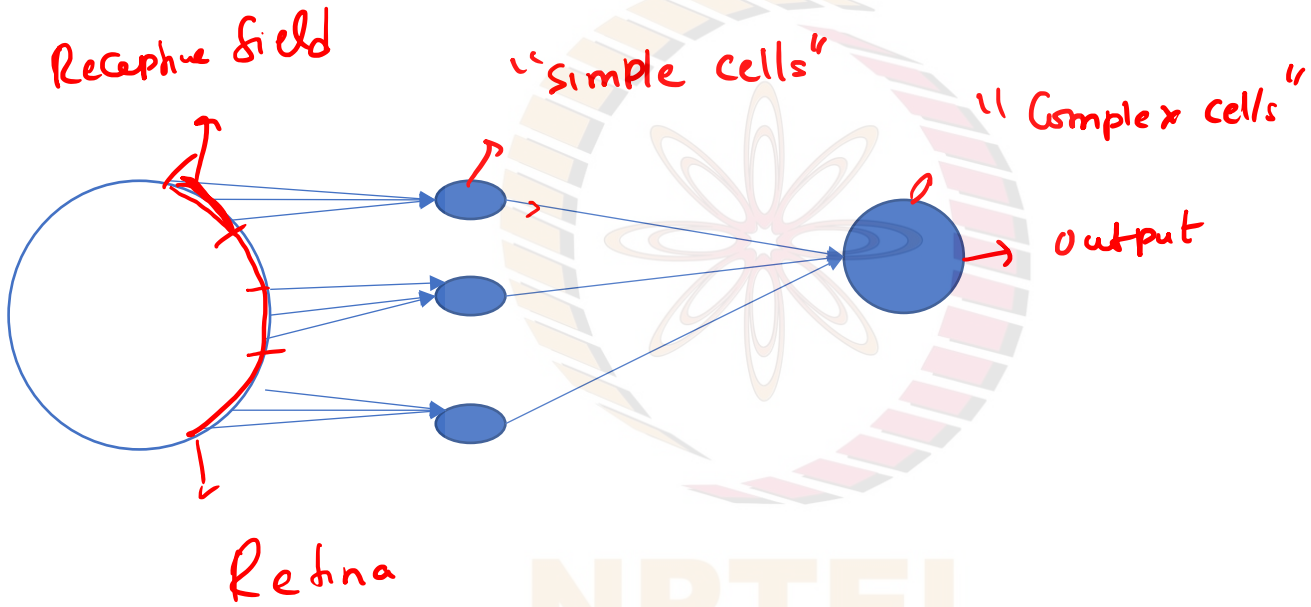
CNN layers


Convolution Layer
Output
Input RGB image
Or 'Volume'
feature Map
Activation Map
20 Map
Output 'Volume'
Convolution Layer
Output
Input gray scale
image
Output 'Volume'

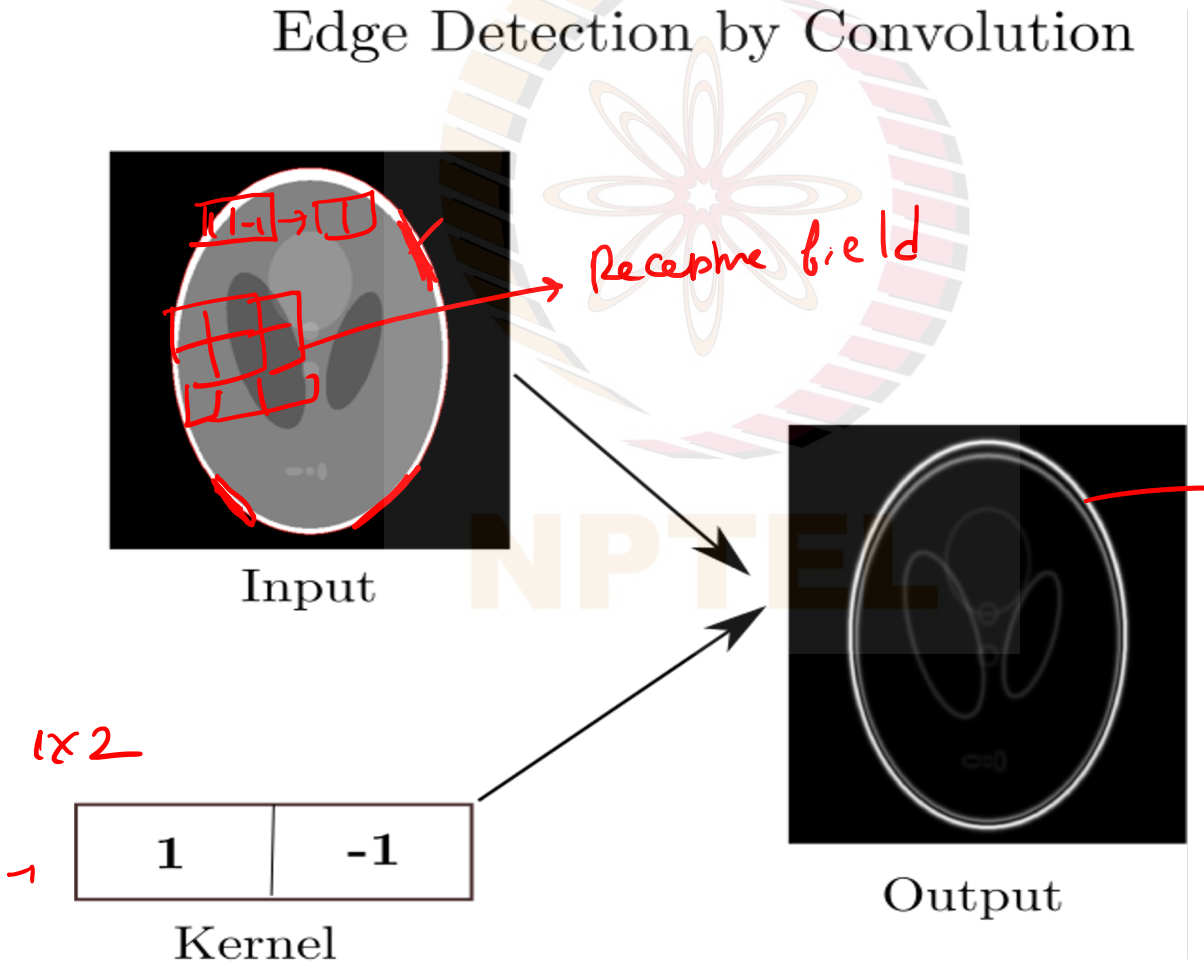
Why Convolutions? Biological Inspiration

- In 1962 Hubel and Wiesel did a series of studies measuring activations in the neuronal cells of cat visual cortex.
- Simple Cells- Respond to edges, lines etc in the receptive field of the cells- linear response
- Complex Cells- Take as input, activations from the simple cells and produces a rectified output- invariant to translation

Biological Inspiration



Why Convolutions?



Convolutions

- Every output neuron is connected a small neighbourhood in the input through a weight matrix – Filter or kernel.
- We can define multiple kernels for every conv layer each giving rise to an output.
- Each filter is moved around the input giving rise to one 2D output. The outputs corresponding to each filter are stacked giving rise to an output volume.

Convolution

1	0	2	2	1
0	2	1	1	3
7	0	1	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

Convolution

1	0	2	2	1
0	2	1	1	3
7	1	0	1	2
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

13		

Convolution

1	0 1	2 1	2 0	1
0	2 2	1 1	1 1	3
7	0 1	1 0	2 2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	

Convolution

1	0	2	2	1
0	2	1	1	3
7	0	1	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	13

Convolution

1	0	2	2	1
0 1	2 1	1 0	1	3
7 2	0 1	1 1	2	1
5 1	1 0	3 2	2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	13
28		

Convolution

1	0	2	2	1
0	2 1	1 1	1 0	3
7	0 2	1 1	2 1	1
5	1 1	3 0	2 2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	13
28	11	

Convolution

1	0	2	2	1
0	2	1	1	3
7	0	1	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	13
28	11	14

Convolution

1	0	2	2	1
0	2	1	1	3
7 1	0 1	1 0	2	1
5 2	1 1	3 1	2	2
2 1	3 0	6 2	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	13
28	11	14
35		

Convolution

1	0	2	2	1
0	2	1	1	3
7	0 1	1 1	2 0	1
5	1 2	3 1	2 1	2
2	3 1	6 0	1 2	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	13
28	11	14
35	13	

Convolution

1	0	2	2	1
0	2	1	1	3
7	0	1	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13	12	13
28	11	14
35	13	29

K feature Maps

$n_x \times n_y \cdot f_x \cdot f_y$

$n_x - f_x + 1$

$n_y - f_y + 1$

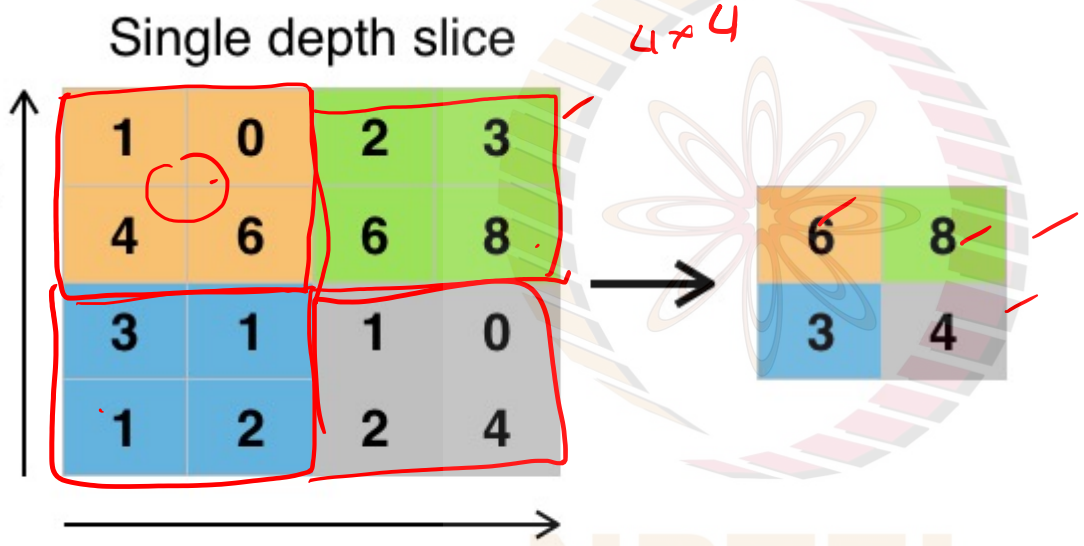
Pooling

- Provides translational invariance by subsampling
- Reduces size of the feature maps
- Average Pooling and Max Pooling are commonly used

256 x 256 → 32 x 32

factor 8 reduction

Max Pooling



4×4

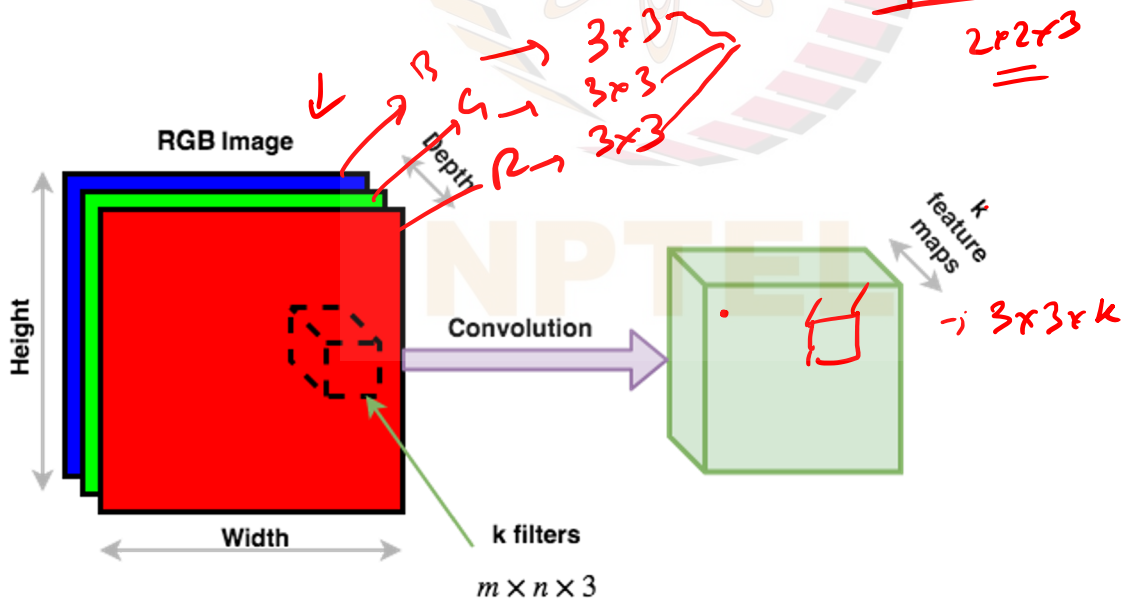
2×2, Stride-2

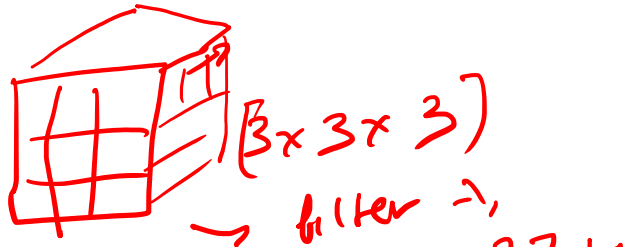
https://upload.wikimedia.org/wikipedia/commons/e/e9/Max_pooling.png

Volume Convolutions

Every layer of a ConvNet has the same API:

- Takes a 3D volume of numbers
- Outputs a 3D volume of number





Size of Output Volume

- Size of the Output Volume or Feature map depend on
  - Size of Input Feature Map
  - Kernel size
  - Zero padding
  - Stride

Padded Convolution

2	1	1
0	1	2
1	3	2

Image [3X3]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

(×1

Padded Convolution

0	0	0	0	0
0	2	1	1	0
0	0	1	2	0
0	1	3	2	0
0	0	0	0	0

P=1

Image [3X3]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

Padded Convolution

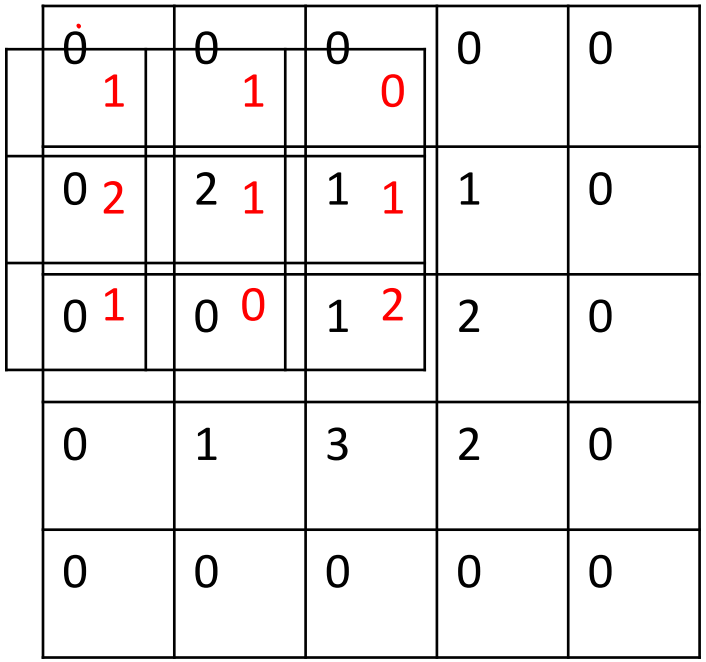


Image [3X3]

*

Convolution

3x3 Kernel

=

5		

Padded Convolution

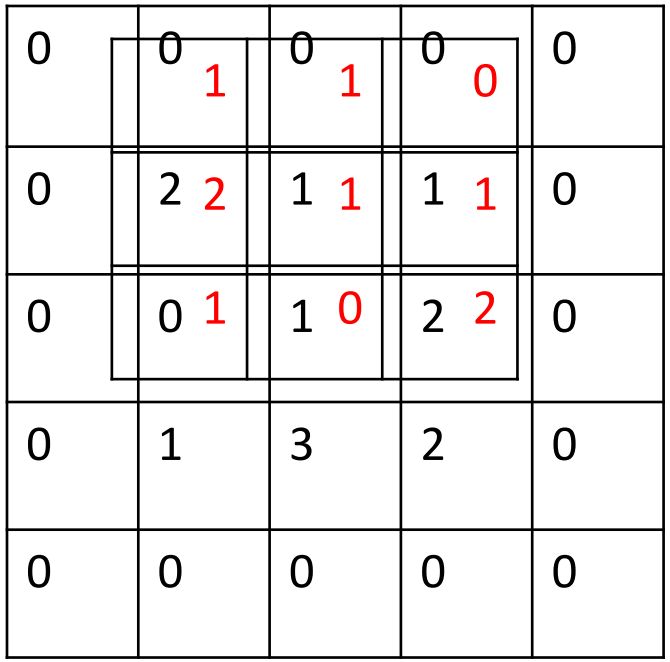
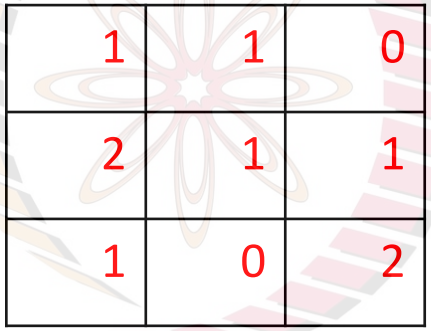


Image [3X3]

*



3x3 Kernel

=

5	10	

Convolution

Padded Convolution

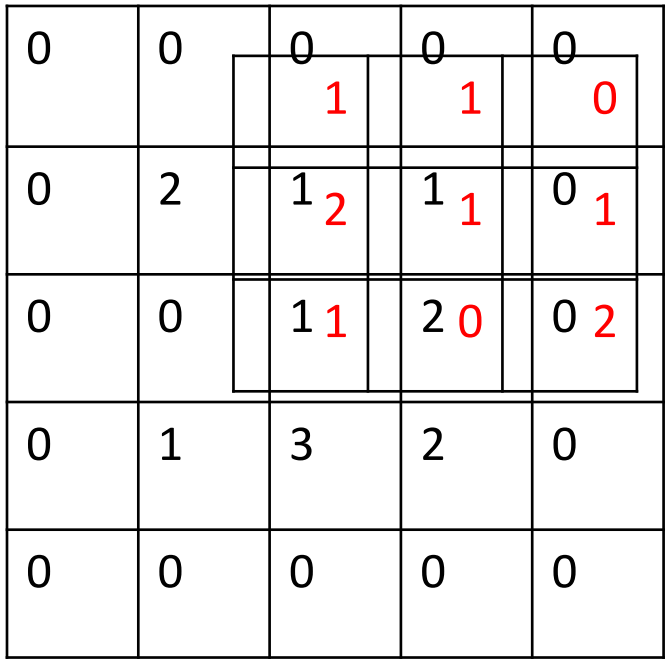
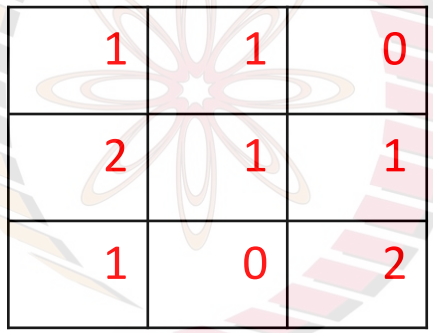


Image [3X3]

*

Convolution



3x3 Kernel

=

5	10	4

Padded Convolution

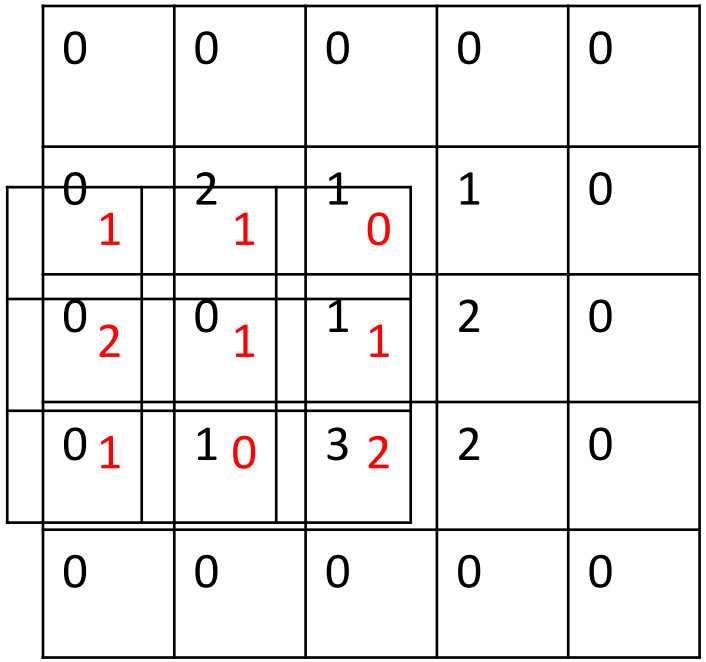


Image [3X3]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

5	10	4
9		

Padded Convolution

0	0	0	0	0
0	2	1	1	0
0	0	1	2	0
0	1	3	2	0
0	0	0	0	0

Image [3X3]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

5	10	4
9	11	

Padded Convolution

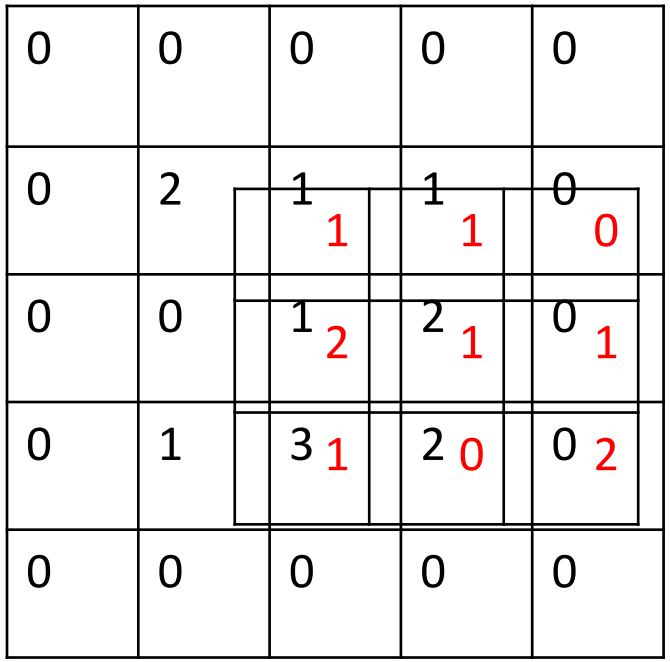
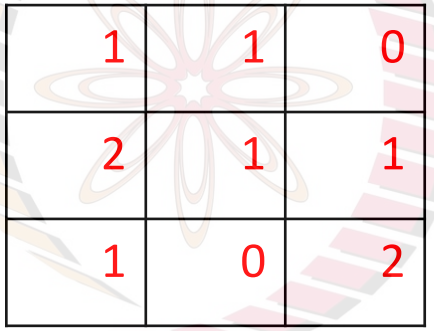


Image [3X3]

*

Convolution



3x3 Kernel

=

5	10	4
9	11	9

Padded Convolution

0	0	0	0	0
0	2	1	1	0
0	0	1	2	0
0	1	3	2	0
0	0	0	0	0

Image [3X3]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

5	10	4
9	11	9
4		

Padded Convolution

0	0	0	0	0
0	2	1	1	0
0	0	1	2	0
0	1	3	2	0
0	0	0	0	0

Image [3X3]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

5	10	4
9	11	9
4	8	

Padded Convolution

0	0	0	0	0
0	2	1	1	0
0	0	1	2	0
0	1	3	2	0
0	0	0	0	0

Image [3X3]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

f -> odd

=

5	10	4
9	11	9
4	8	11

$n - f + 1 + 2P$

$3 - 3 + 1 + 2 \cdot 1 = 3$

$\rightarrow 2P = f - 1$

$P = \left(\frac{f-1}{2}\right)$

Strided Convolutions

Strided Convolution – Stride = 1

1	0	2	2	1
0	2	1	1	3
7	0	1	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

Strided Convolution

1 1	0 1	2 0	2	1
0 2	2 1	1 1	1	3
7 1	0 0	1 2	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

13		

Strided Convolution

1	0	2	2	1
0	2	1	1	3
7	0	1	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13		13

Strided Convolution

1	0	2	2	1
0	2	1	1	3
7 1	0 1	1 0	2	1
5 2	1 1	3 1	2	2
2 1	3 0	6 2	1	5

Image [5X5]

*

Convolution

1	1	0
2	1	1
1	0	2

3x3 Kernel

=

13		13
35		

Strided Convolution

1	0	2	2	1
0	2	1	1	3
7	0	1	2	1
5	1	3	2	2
2	3	6	1	5

Image [5X5]

*

1	1	0
2	1	1
1	0	2

3x3 Kernel

Convolution

=

13		13
35		29

$w^T x \rightarrow$ non-linearity

Relu

$$\frac{n - f + 2p}{s} + 1$$

$$\frac{5 - 3 + 0}{2} + 1 \rightarrow 2 \frac{1}{2}$$