ALGONQUIN COLLEGE

CST8506 ADVANCED MACHINE LEARNING

Week 3

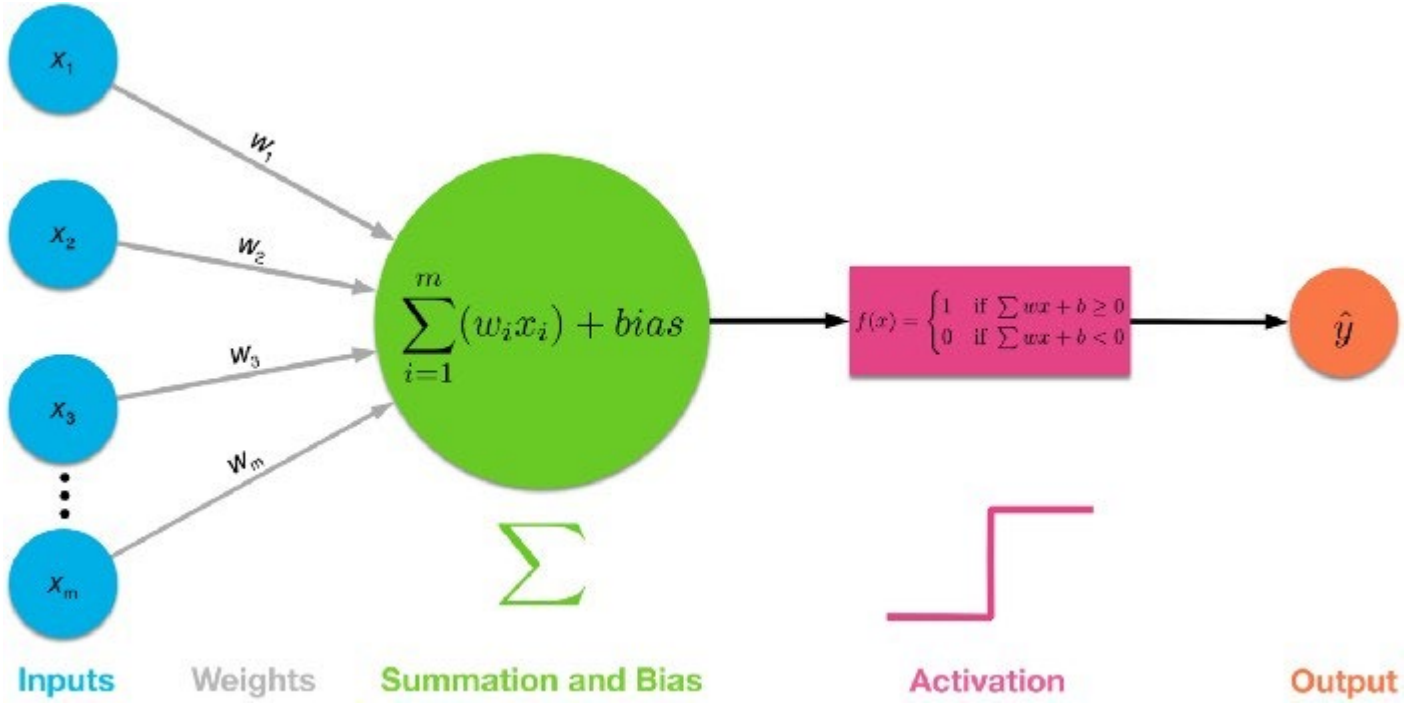
Neural Networks

Professor: Dr. Anu Thomas
Email: thomasa@algonquincollege.com
Office: T315

Agenda

- Recap on
  - ANN
  - Perceptron, MLP
- CNN

Single-layer Perceptron Network



Taken from: https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f

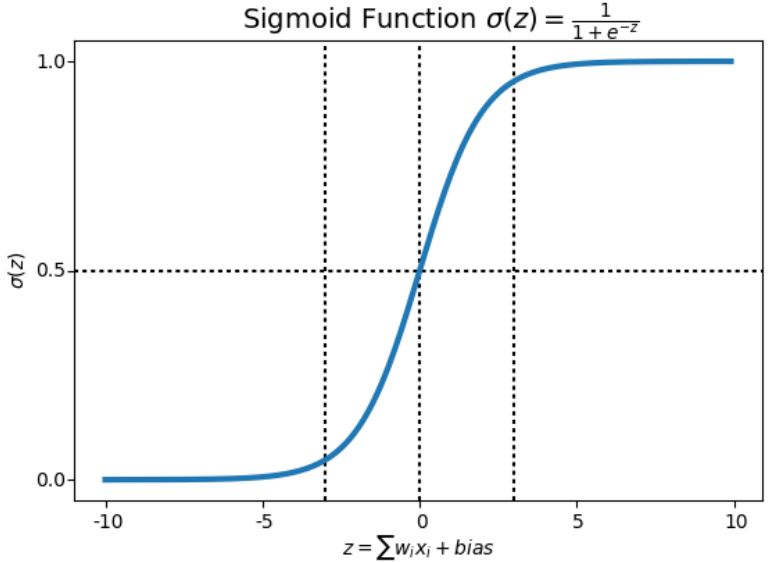
Activation Functions

Sometimes, activation function is as simple as:

$$Output = \begin{cases} 1 \; if \sum w_i x_i + b \; \geq 0 \\ 0 \; if \sum w_i x_i + b \; < 0 \end{cases}$$

Most commonly used Activation function is the Sigmoid function:

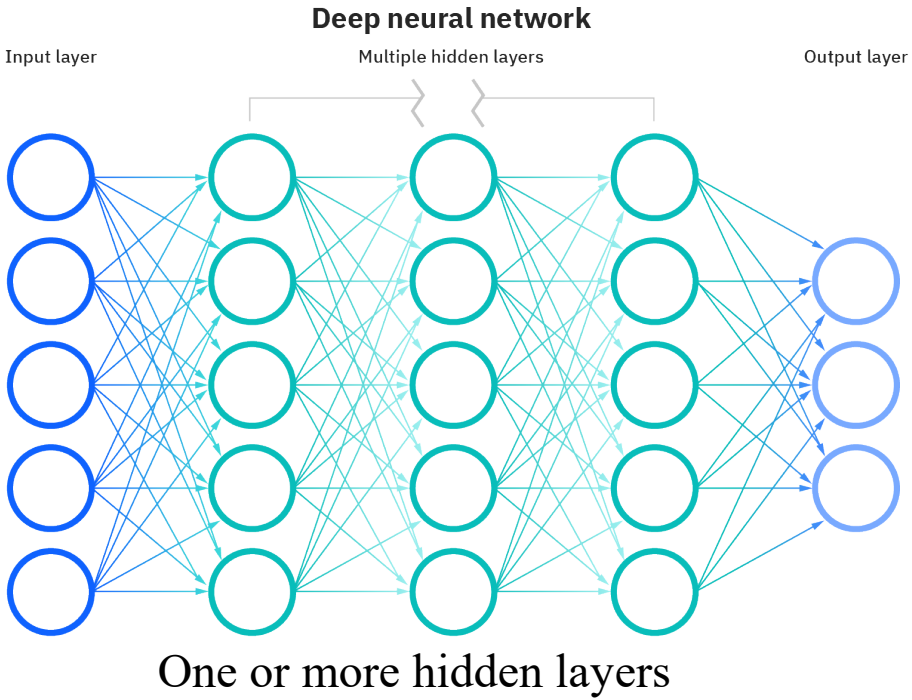
$$\sigma(z) = \frac{1}{1+ e^{-z}}$$



Activation Functions

- Sigmoid
  - $f(x) = \frac{1}{1+ e^{-x}}$
  - Maps values between 0 and 1, often used in binary classification
- Tanh
  - $f(x) = \frac{e^{x}-e^{-x}}{e^{x}+ e^{-x}}$
  - Maps values between -1 and 1, often used in hidden layers
- ReLU (Rectified Linear Unit)
  - $f(x) = \max(0, x)$
  - Only outputs positive values
- Softmax
  - Used in multi-class classification, it outputs probabilities summing up to 1

Multi-layer Perceptron



One or more hidden layers

Picture taken from https://www.ibm.com/cloud/learn/neural-networks

Some of the parameters

- hidden_layer_sizes: specify the number of layers and the number of nodes in each layer.
  `hidden_layer_sizes=(5,3)`
  Means we have 2 hidden layers, first one with 5 nodes and second one with 3 nodes.
- activation: activation function

Check https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html to see all parameters

Types of Neural Networks

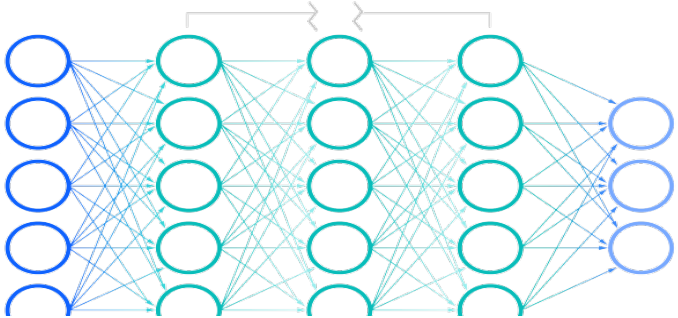
- Perceptron
  - single neuron
  - Single layer perceptron can only learn linearly separable problems
- Multi-layer Perceptron
  - Input layer, one or more hidden layers, output layer
  - Feed-forward NN
    - Data flows only in one direction, without any feedback loops or recurrent connections.
  - Uses back propagation for training
    - Repeatedly adjust the weights to minimize the difference between actual output and the desired output
- Convolutional NN
  - Utilized for computer vision etc.
- Recurrent NN
  - For text processing

Convolutional Neural Networks

Why we need it?

Example – Iris vs Rose

All Images were taken from google search



Output:
Iris/Rose

Machine Learning vs Deep Learning - Example

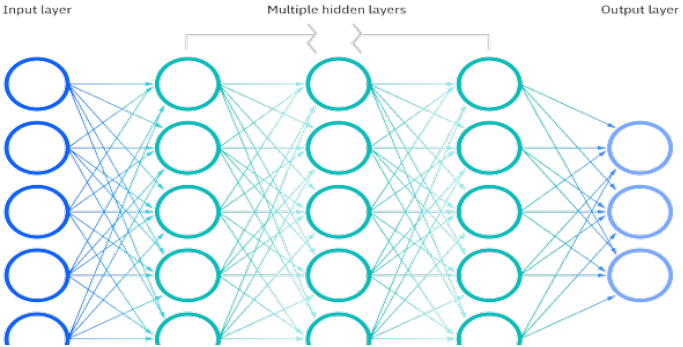
Machine Learning model

Feature Extraction

Classification

Output: Cat/ Not Cat

Deep Learning model



Output: Cat/ Not Cat

Feature extraction and classification

Image Classification

High number of inputs (for example, if the picture is 1600 X 1200, then 1600 *1200 * 3 = 5,760,000 inputs – 5 million inputs)

If we have 1000 nodes in the first layer, then we will have 5 million * 1000 weights, which is 5 billion weights

High computational and memory requirements

Can we use MLP for this problem? **NO**

Solution: Convolutional Neural Network

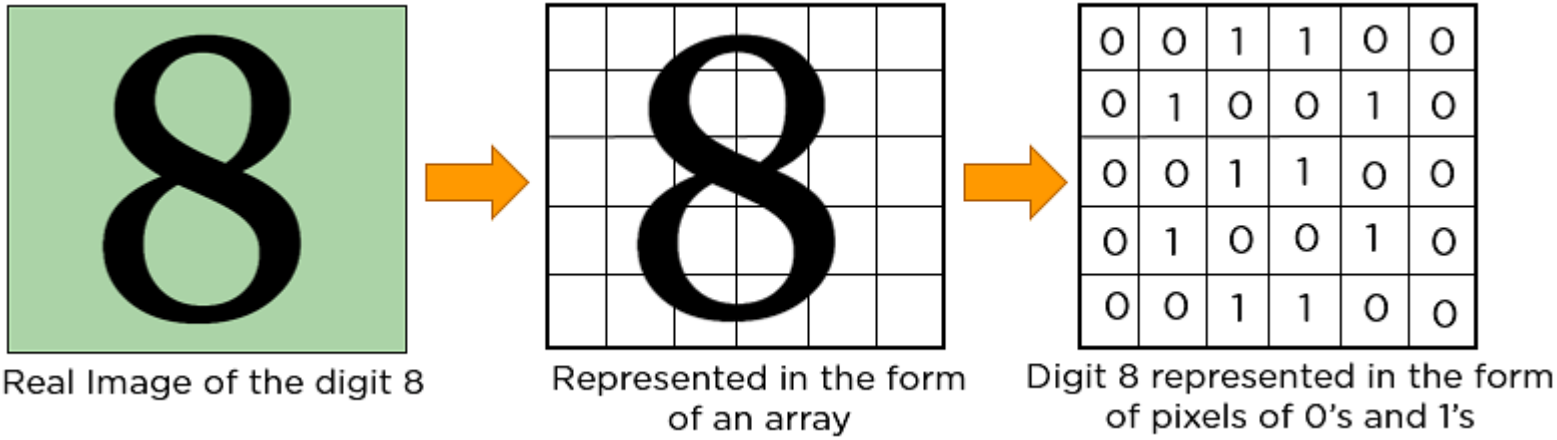
Pic taken from: http://wallpaperstone.blogspot.com/2012/03/meow-meow-cat-wallpapers.html

Sample Problems to Solve in Vision

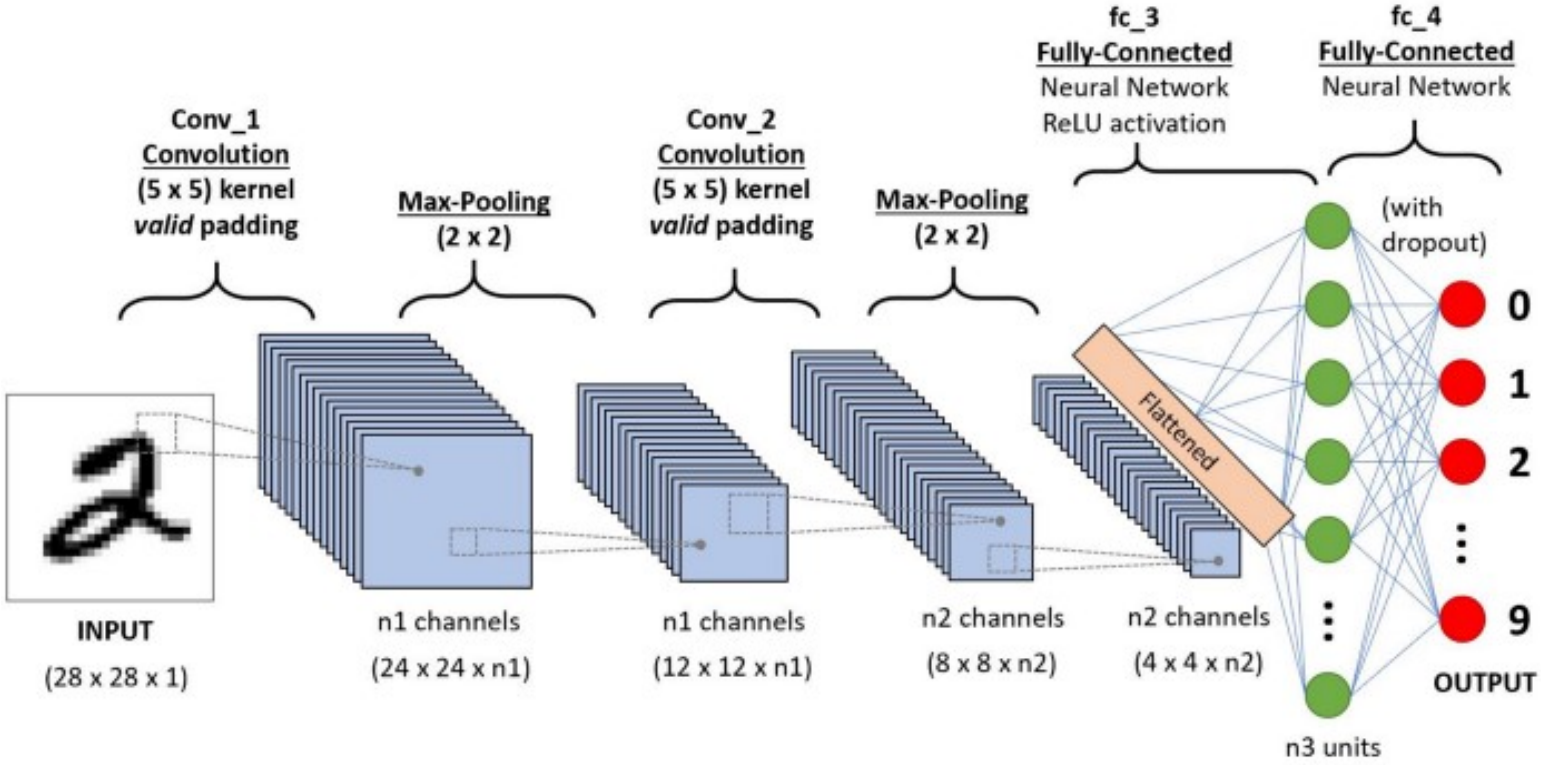
- Image classification
  - take an input picture, classify it as a cat or not
- Object Detection
  - For self-driving cars, find objects around

Convolutional Neural Network

- Feed forward NN
- Generally used to analyze images
- Done by processing images in the form of arrays of pixel values



Convolutional Neural Network



Taken from: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

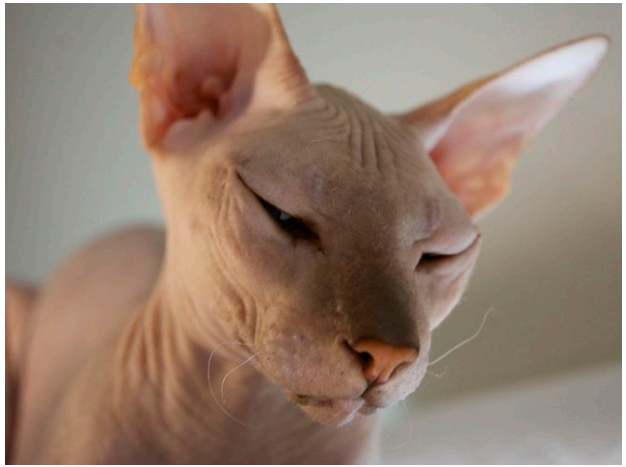
Convolutional Neural Network

Objective: reduce the images into a form that is easier to process, without losing critical features that helps in prediction

Terminologies

- Convolution
- Filter
- Padding (Valid or Same)
- Stride
- Pooling (Max-pooling or average-pooling)

Edge Detection



Cat – Original Picture

Basic edge filters applied to the greyscale image of cat

Basic edge filters applied to the RGB image of cat

Picture taken from: https://austingwalters.com/edge-detection-in-computer-vision/

Edge Detection

Based on the variation in light intensity at different parts of the image, we should be able to find:

- Vertical edges
- Horizontal edges
- Edges at different angles

These edges give us important information!

Convolution

Convolve the image matrix with a filter, which is another matrix.

0 [1]	3 [0]	4 [-1]	4	3	0
0 [1]	4 [0]	5 [-1]	5	3	0
0 [1]	0 [0]	2 [-1]	2	0	0
0	3	5	5	4	0
1	4	5	5	4	1
2	4	4	5	5	0

6 x 6 image

*

Convolution operator

1	0	-1
1	0	-1
1	0	-1

filter

=

-11			

Output image
Reduced to 4 x 4

Convolution

0	3^{1}	4^{0}	4^{-1}	3	0
0	4^{1}	5^{0}	5^{-1}	3	0
0	0^{1}	2^{0}	2^{-1}	0	0
0	3	5	5	4	0
1	4	5	5	4	1
2	4	4	5	5	0

*

1	0	-1
1	0	-1
1	0	-1

Convolution operator

=

-11	-4		

Edge Detection Filters

1	0	-1
1	0	-1
1	0	-1

Vertical

1	1	1
0	0	0
-1	-1	-1

Horizontal

1	0	-1
2	0	-2
1	0	-1

Sobel

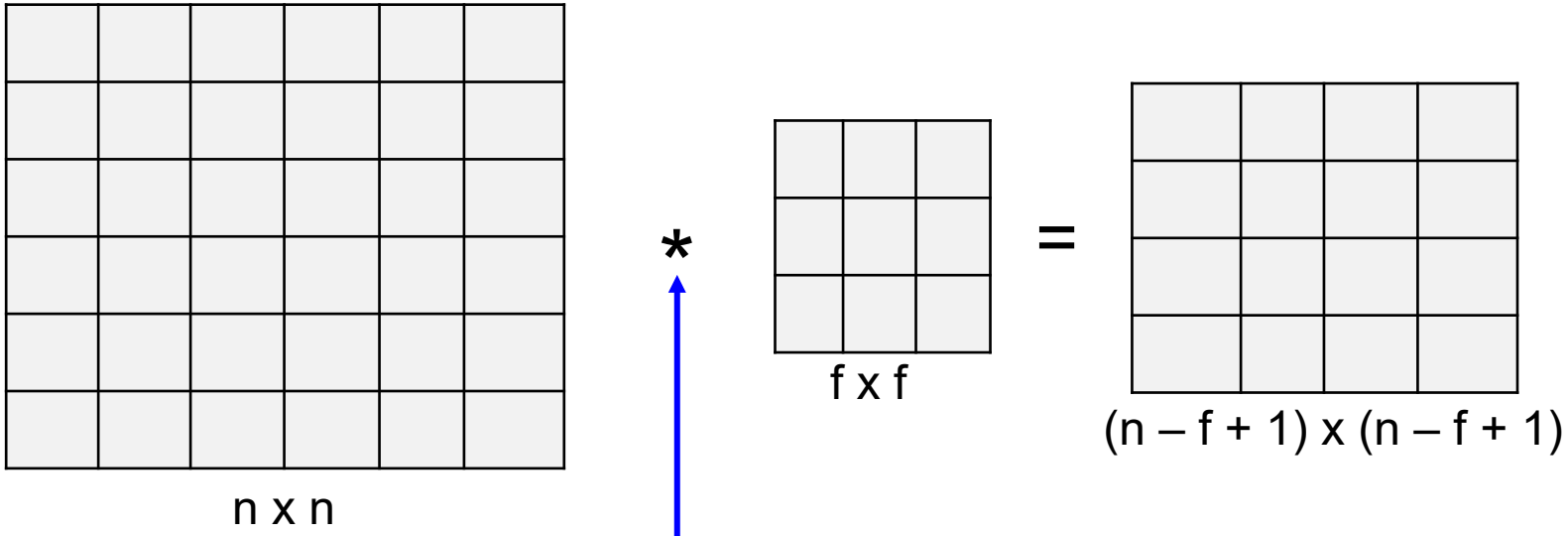
3	0	-3
10	0	-10
3	0	-3

Scharr

Can we have our own filters?

Yes, we can consider this filter as a parameter and learn them!

Output of Convolution with filters

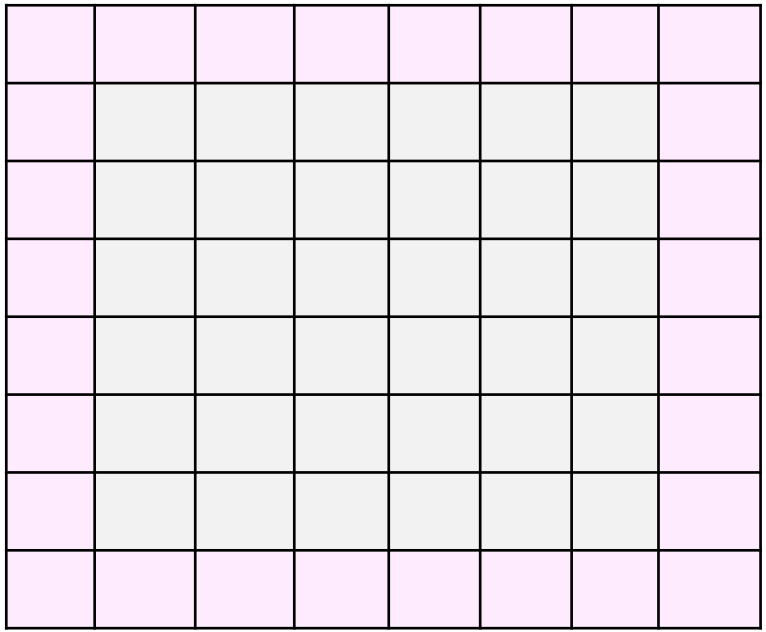


f is conventionally odd number

Padding

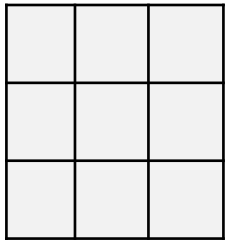
- Refers to the number of pixels added to an image
- Padding is added to the frame of the image to give more space for the filter to cover the image. Once padding is added, previous end pixels will be part of multiple 3x3 matrices.
- This is a process to make sure the size of the output is not less than that of the input (if we have a 6x6 image convoluted with a 3x3 filter, output will be 4x4. If we add one layer of padding around the picture, output for a 6x6 image will be another 6x6 image).

Padding - Example



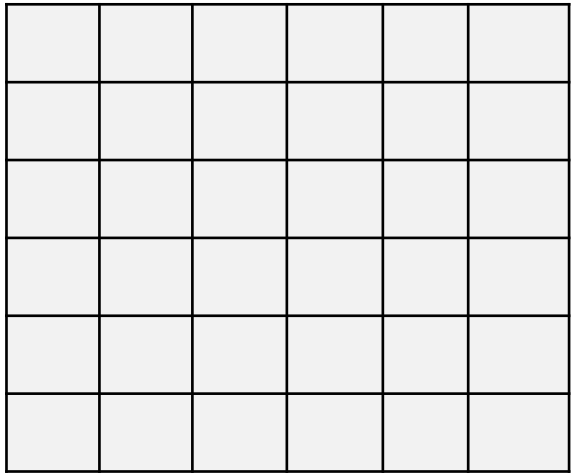
8 x 8

*



3 x 3

=



6 x 6

Output: $n + 2p - f + 1 \quad \times \quad n + 2p - f + 1$

Valid vs Same Padding

- **Valid**: No padding
  - n x n image filtered with f x f filter gives $(n - f + 1) \times (n - f + 1)$ image
- **Same**: add padding such that output size should be the same as input size
  - $n + 2p - f + 1 = n$
  - $p = \frac{f-1}{2}$

Strided Convolution

4^{2}	3^{3}	5^{3}	7	6	5	3
7^{1}	7^{0}	8^{1}	9	8	7	6
4^{-1}	3^{0}	7^{2}	8	6	4	3
6	7	8	5	6	2	3
2	3	4	7	5	4	1
0	2	3	1	5	7	2
0	0	2	4	3	1	0

*

2	3	3
1	0	1
-1	0	2

=

57		

Strided Convolution

4	3	5^{2}	7^{3}	6^{32}	5^{3}	3^{3}
7	7	8^{1}	9^{0}	8^{11}	7^{0}	6^{1}
4	3	7^{-1}	8^{0}	6^{21}	4^{0}	3^{2}
6	7	8	5	6	2	3
2	3	4	7	5	4	1
0	2	3	1	5	7	2
0	0	2	4	3	1	0

*

2	3	3
1	0	1
-1	0	2

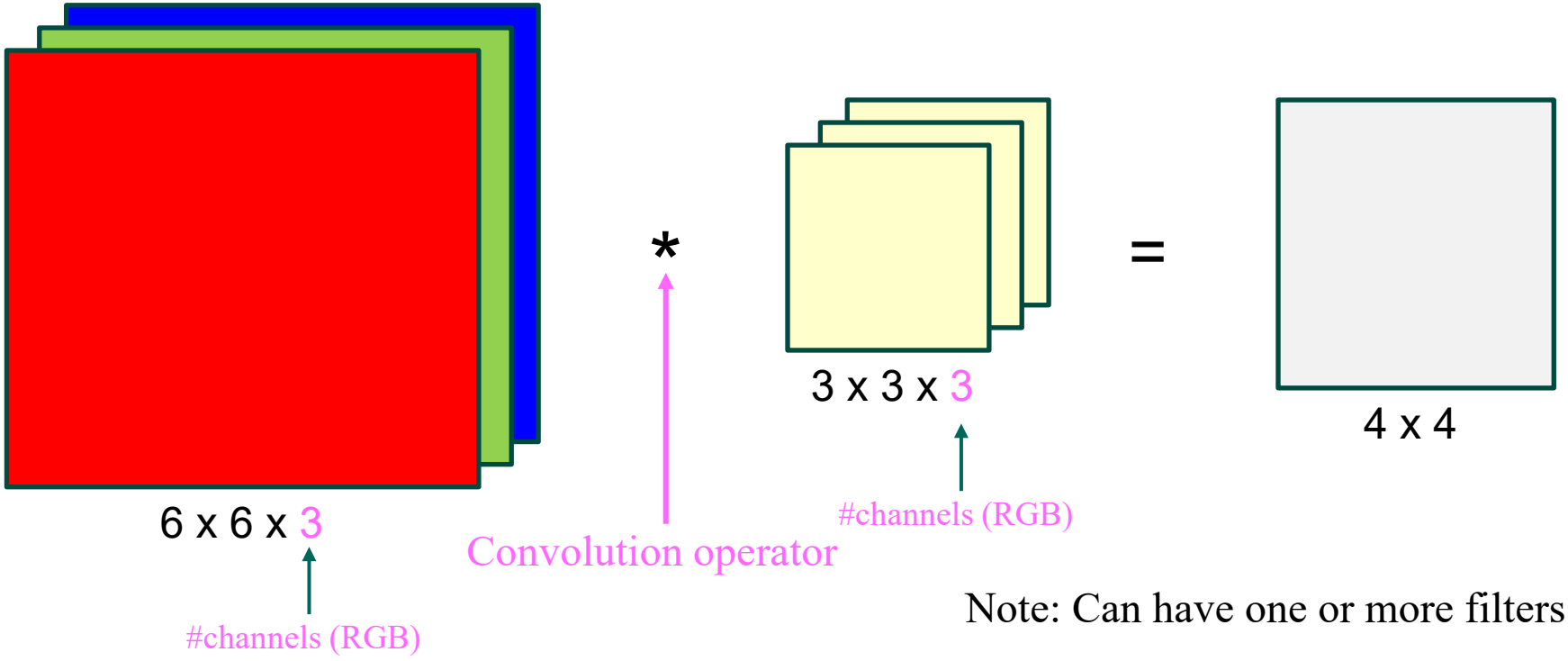
=

57	70	50

n x n image convolve with an *f x f* filter with a padding *p* and stride *s*, output will be

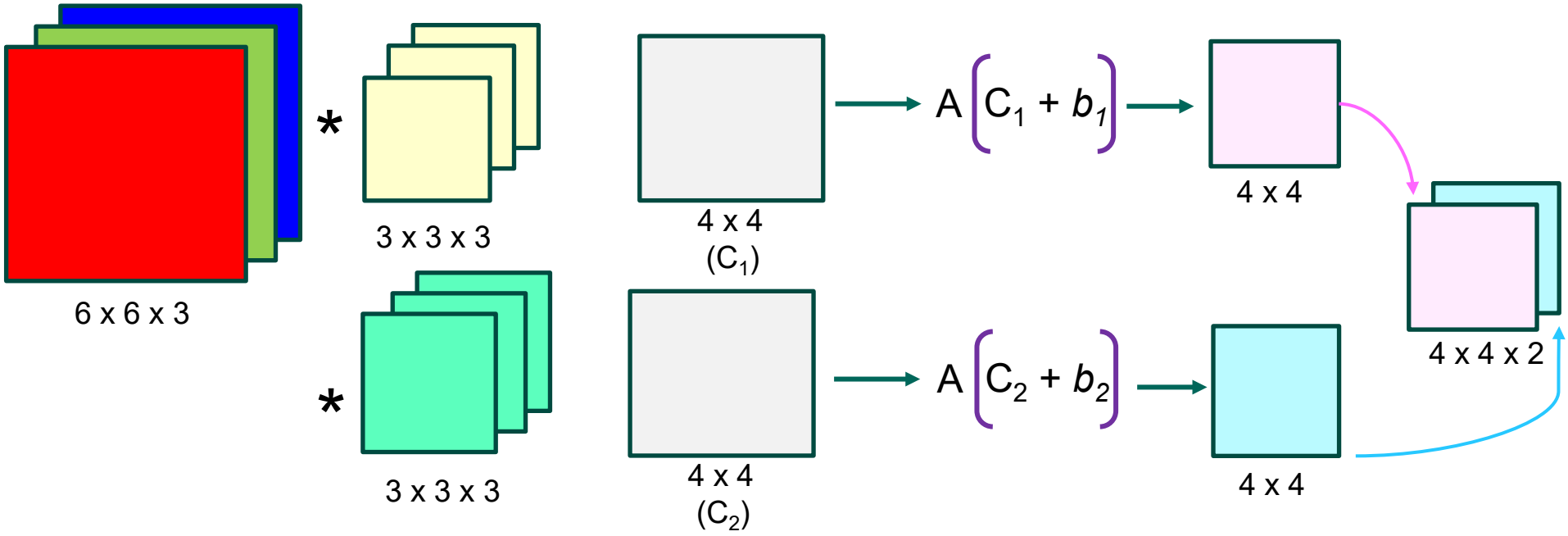
$$\left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor$$

Convolutions on RGB Images



Note: Can have one or more filters

One Layer of a CNN



In this example, we have 2 filters. We can have more!

Example

24 x 24 x 3

Filter: 3 x 3
Stride: 1
Padding: 0
#filters: 10

22 x 22 x 10

Filter: 5 x 5
Stride: 2
Padding: 0
#filters: 20

9 x 9 x 20

Filter: 5 x 5
Stride: 2
Padding: 0
#filters: 40

3 x 3 x 40

flattened

MLP

Output
Cat (Y/N)

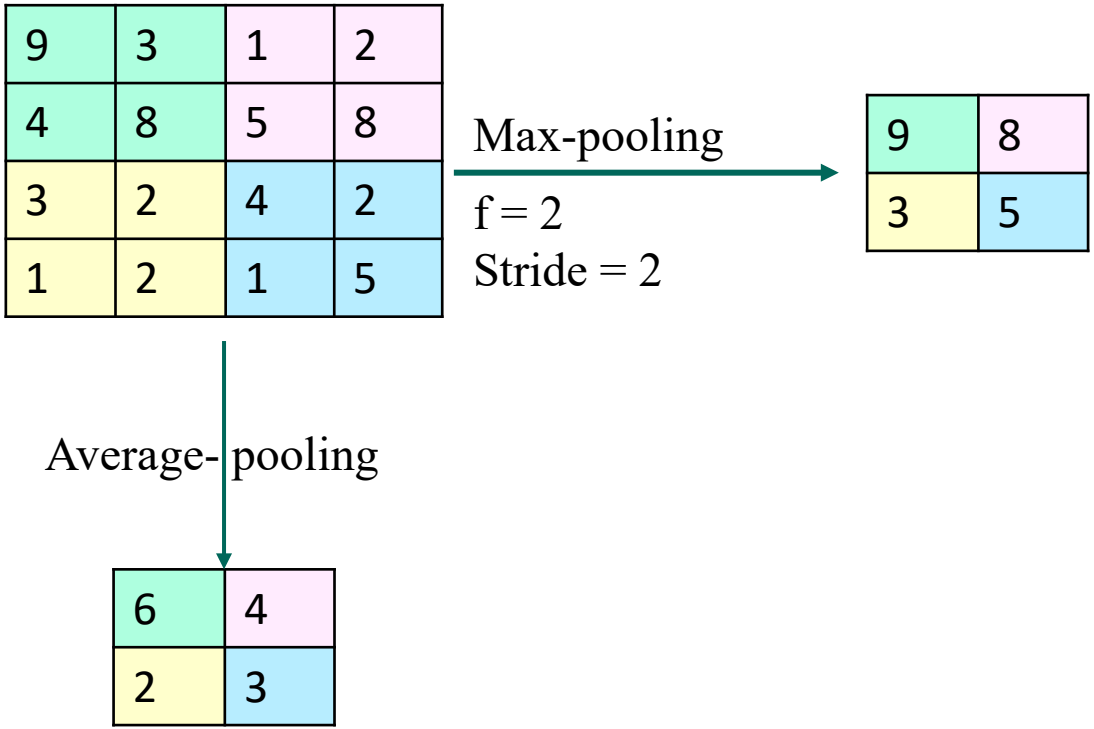
24 * 24 * 3 = 1728
reduced to
3 * 3 * 40 = 360

Pic taken from https://www.pexels.com/photo/grey-and-white-short-fur-cat-104827/

Types of Layers in a CNN

- Convolution
- Pooling
- Fully Connected

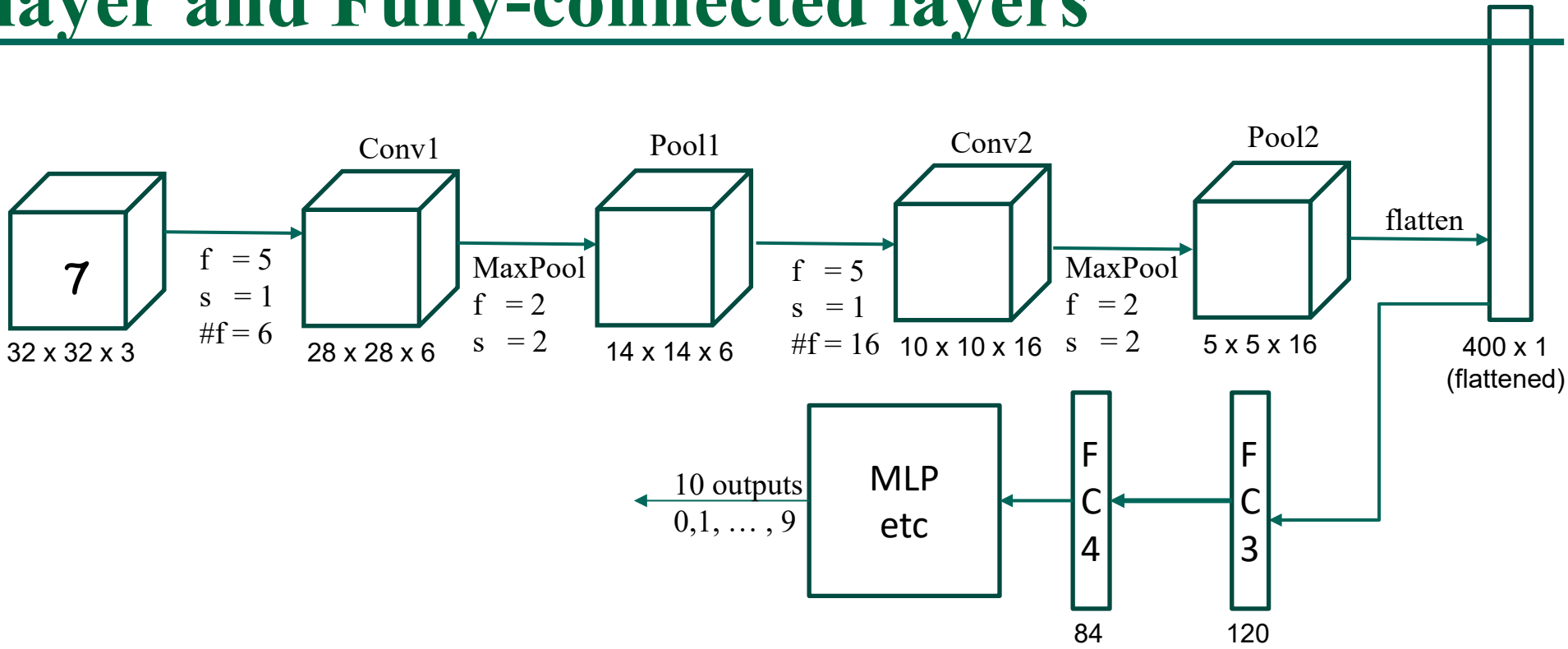
Pooling Layers



Benefits of Pooling

- Reduces dimensions and computation
- Reduces overfitting as there are less parameters
- Makes the model tolerant towards small variations and distortions
- Filters all important features and filters out noise

Example with Convolution layer, Pooling layer and Fully-connected layers



Convolution Operation - Summary

- Objective is to extract high-level features like edges from the input image
- Can have multiple convolution layers
- Conventionally, first ConvLayer captures low level features like edges, color, gradient orientation etc.
- Reduce the size without losing relevant information
- Once it is reduced, the output matrix will be flattened and feed it to some classifiers like MLP

References

- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
- https://medium.com/swlh/convolutional-neural-networks-22764af1c42a
- https://austingwalters.com/edge-detection-in-computer-vision/
- https://www.geeksforgeeks.org/machine-learning/activation-functions-neural-networks/