COMPUTER VISION

Basics

COMPUTER VISION

- At the end of the class, you will be familier with basic CNN architecture
- Understand how computer persieve and interpret images
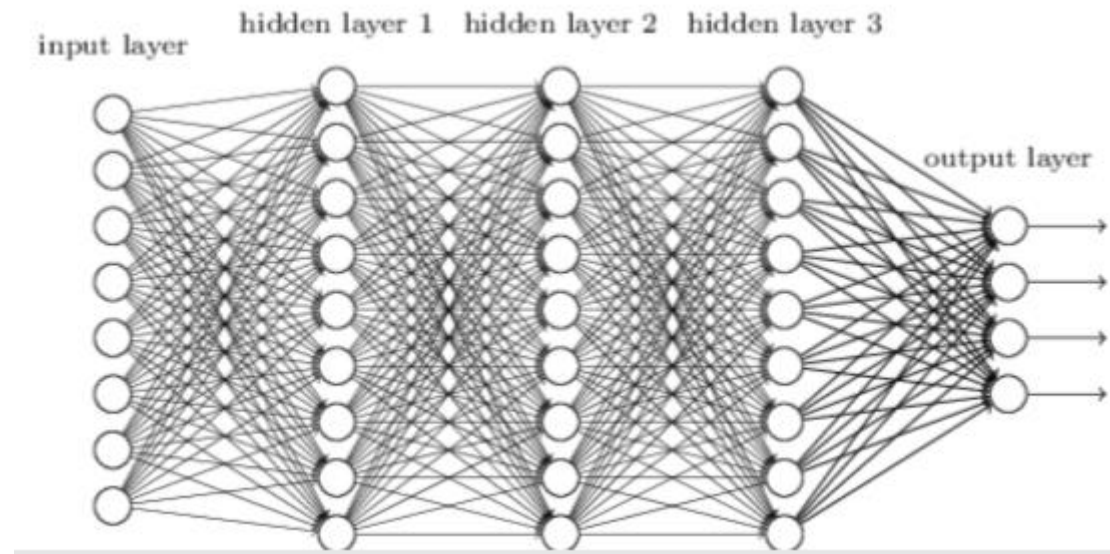- Able to deploy YOLO on your PC

Pre-requisite:

- You are expected to know
- Basic coding algorithm flow
- Linear Algebra
- Comfortable with compiling and executing codes from github

WHAT IS DEEP LEARNING?

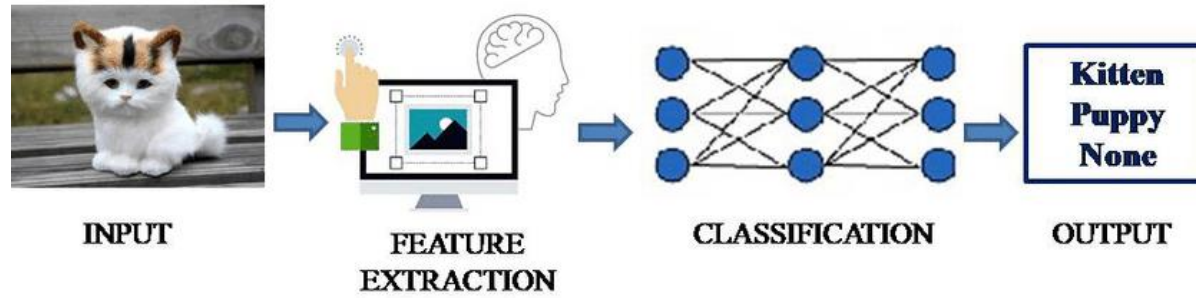Most machine learning methods work well because of human-designed input features or representations – Our job is to find thebestfeatures to send to learning techniques such as SVM.

- Deep learning algorithms learn multiple levels of representations (here: hiddenlayer1,2,3) and an output layer

- From "raw" inputs x (e.g. sound, pixels, characters, or words)

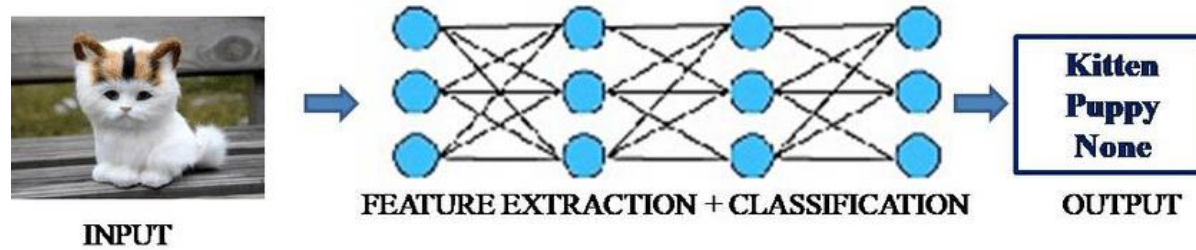- Neural networksare the currently successful method for deep learning

More deeper models results in more accurate representation of features and better classification
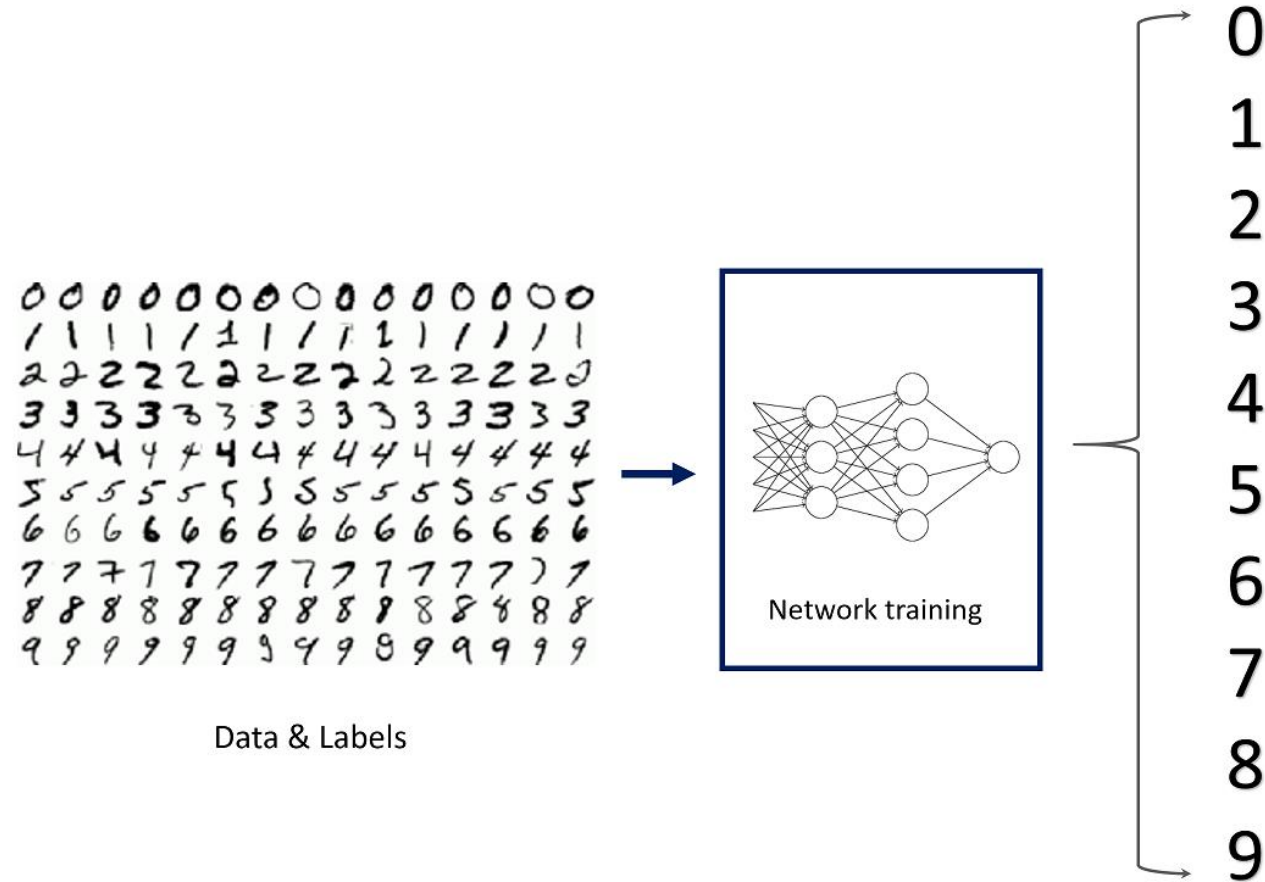
MACHINE LEARNING

INPUT → FEATURE EXTRACTION → CLASSIFICATION → OUTPUT

Kitten
Puppy
None

DEEP LEARNING

INPUT → FEATURE EXTRACTION + CLASSIFICATION → OUTPUT

Kitten
Puppy
None

MNIST – Popular CNN model
a.k.a Hello world of ML

Data & Labels

Network training

0
1
2
3
4
5
6
7
8
9

Lets try to learn the flow involved in building a CNN model capable of finding SWANS

A swan has some unique features. All its features has to be extracted and is used for recognition purposes.
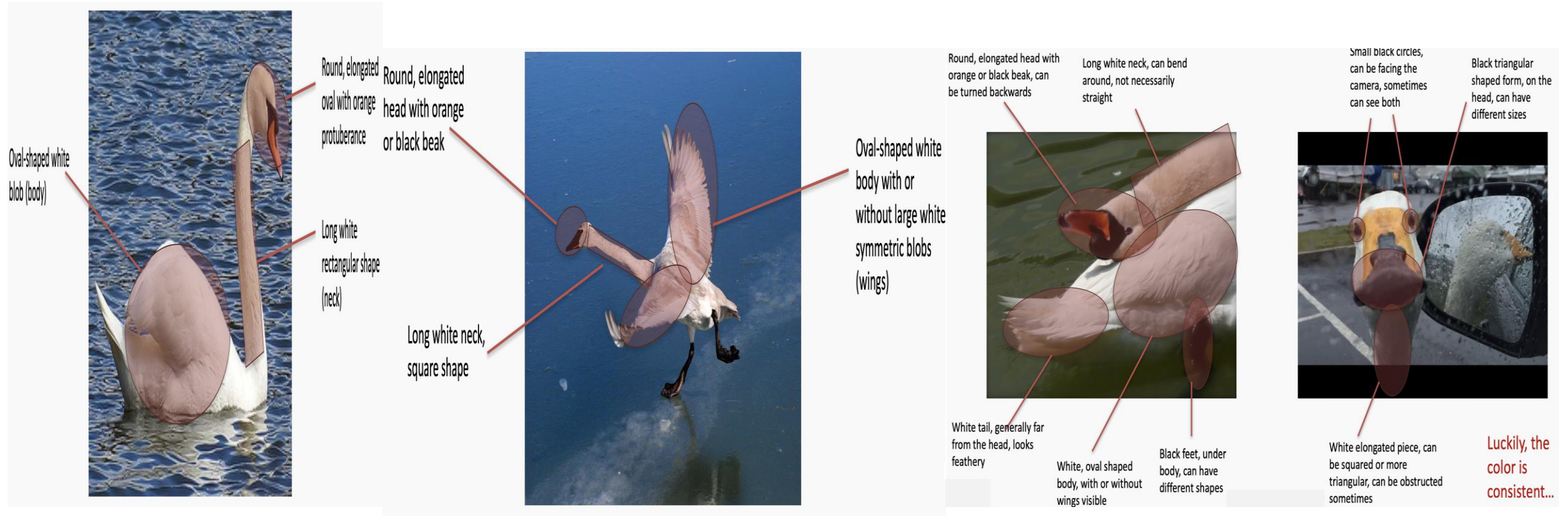
Man in swan tent photographing swans

Worst-case scenarios.

CNN

- The general idea of CNN's is to intelligently adapt to the properties of images:
- Pixel position and neighborhood have semantic meanings
- Elements of interest can appear anywhere in the image

A CNN has

- Convolutional layers
- ReLU layers
- Pooling layers
- a Fully connected layer

Enter the Convolutional Neural Network

- We analyze the influence of nearby pixels by using something called a filter. A filter is exactly what you think it is, in our situation, we take a filter of a size specified by the user (a rule of thumb is 3x3 or 5x5) and we move this across the image from top left to bottom right. For each point on the image, a value is calculated based on the filter using a convolution operation

- A filter could be related to anything, for pictures of humans, one filter could be associated with seeing noses, and our nose filter would give us an indication of how strongly a nose seems to appear in our image, and how many times and in what locations they occur.

CNN LAYERS

- Convolutional layer is the very first layer where we extract features from the images in our datasets

- It decreases the size of the image without losing the relationship between pixels
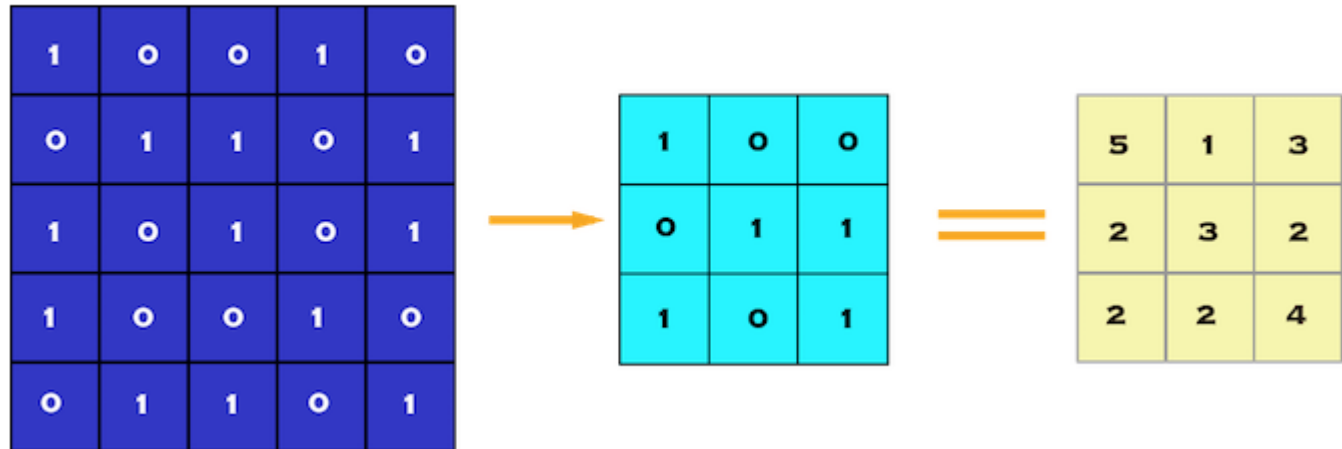
Convolutional layers(Filters)

After the filters have passed over the image, a feature map is generated for each filter. These are then taken through an activation function, which decides whether a certain feature is present at a given location in the image. We can then do a lot of things, such as adding more filtering layers and creating more feature maps, which become more and more abstract as we create a deeper CNN.

Examples of kernel filters for CNN's.

Pooling layers

When constructing CNNs, it is common to insert pooling layers after each convolution layer to reduce the spatial size of the representation to reduce the parameter counts which reduces the computational complexity. In addition, pooling layers also he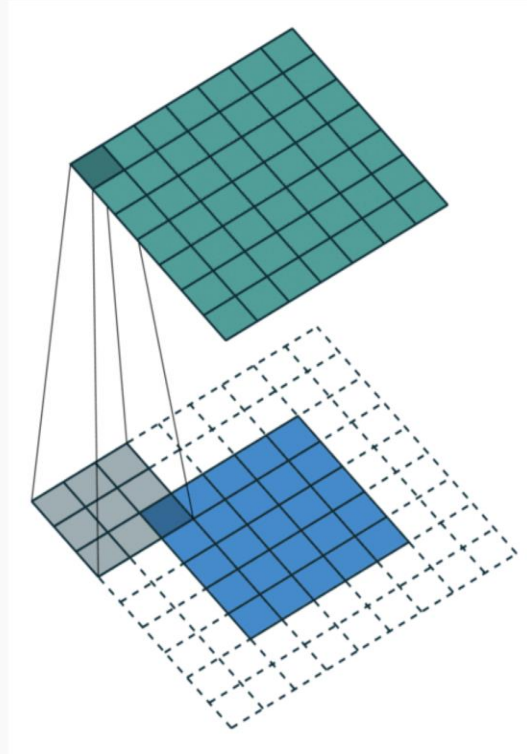lps with the overfitting problem. Basically we select a pooling size to reduce the amount of the parameters by selecting the maximum, average, or sum values inside these pixels. Max Pooling, one of the most common pooling techniques, may be demonstrated as follows:
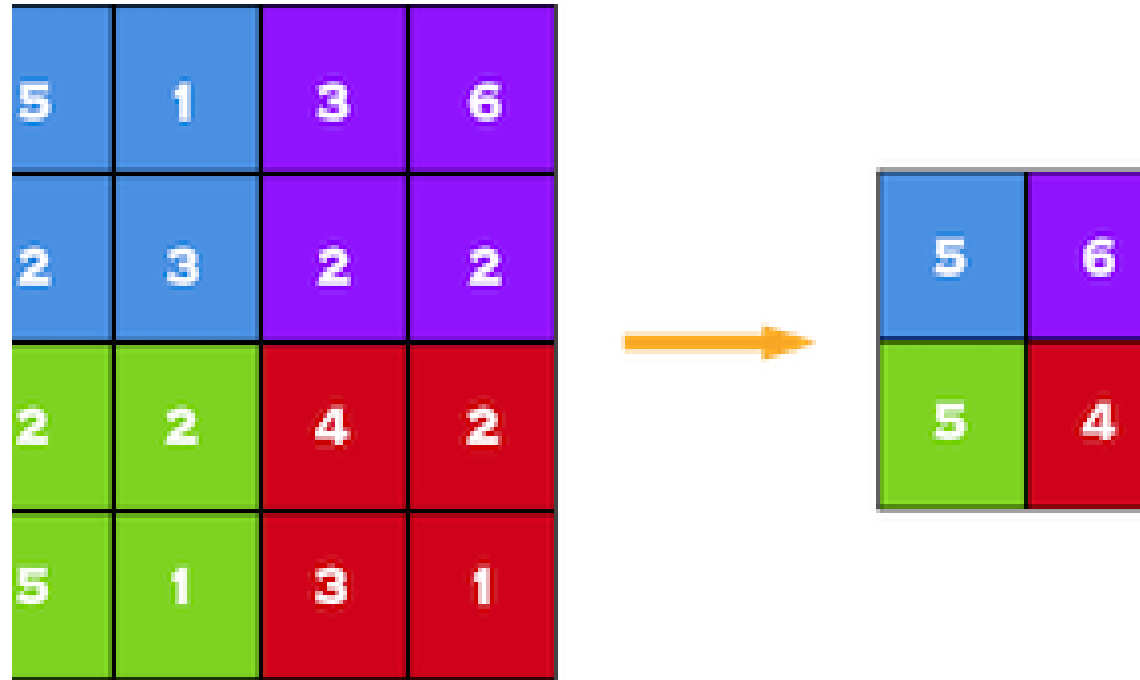
Max Pooling by 2 x 2

Example of how convolutions are applied to images using kernel filters.
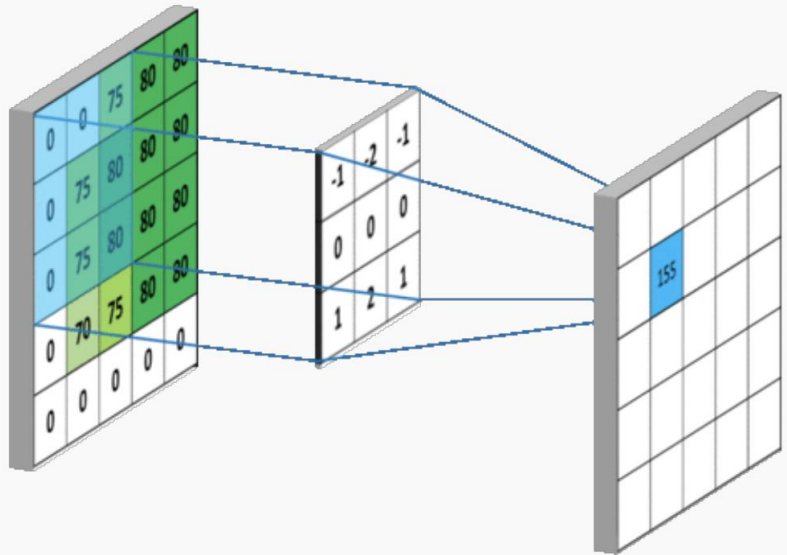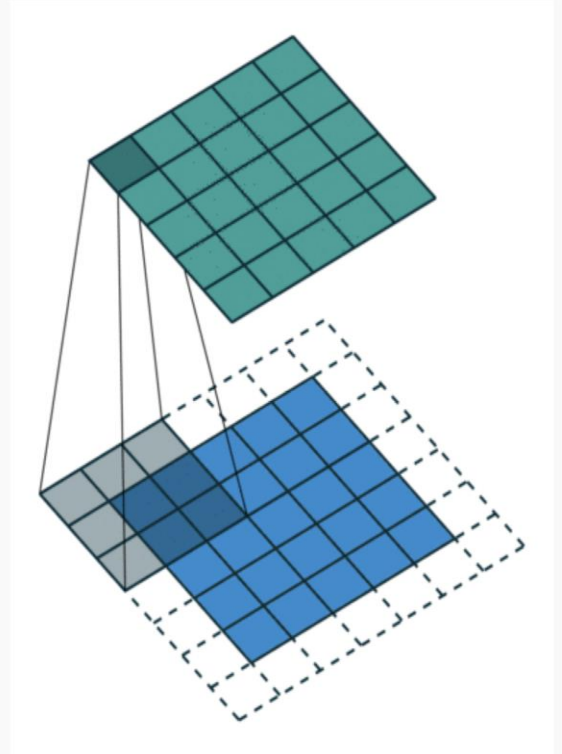

Full padding. Introduces zeros such that all pixels are visited the same amount of times by the filter. Increases size of output.

Same padding. Ensures that the output has the same size as the input.

How do we connect our filters together?
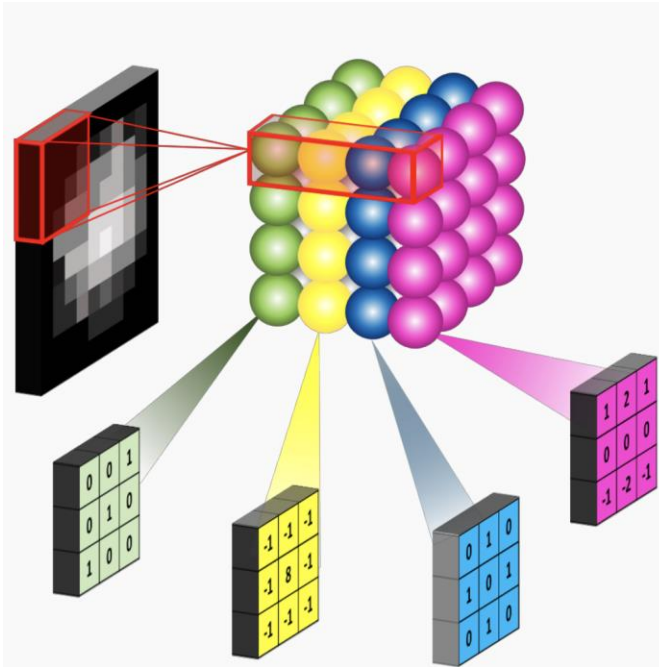
Convolutional layer with four 3x3 filters on a black and white image (just one channel)

Convolutional layer with four 3x3 filters on an RGB image. As you can see, the filters are now cubes, and they are applied on the full depth of the image..

- To be clear, each filter is convolved with the entirety of the 3D input cube but generates a 2D feature map.

- Because we have multiple filters, we end up with a 3D output: one 2D feature map per filter
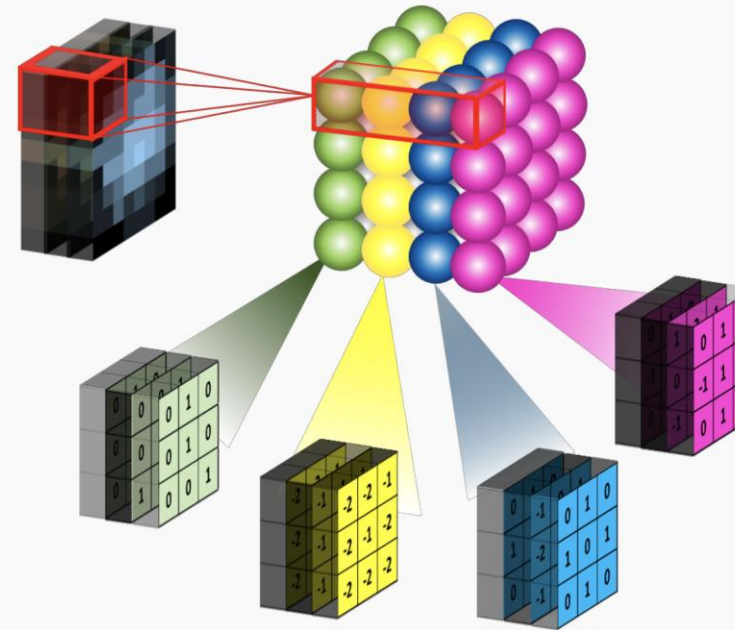
- The feature map dimension can change drastically from one convolutional layer to the next: we can enter a layer with a 32x32x16 input and exit with a 32x32x128 output if that layer has 128 filters.

- Convolving the image with a filter produces a feature map that highlights the presence of a given feature in the image.

Convolutional Layers

Action

- Apply filters to extract features
- Filters are composed of small kernels, learned.
- One bias per filter.
- Apply activation function on every value of feature map

Parameters

- Number of kernels
- Size of kernels (W and H only, D is defined by input cube)
- Activation function
- Stride
- Padding
- Regularization type and value

I/O

- Input: 3D cube, previous set of feature maps
- Output: 3D cube, one 2D map per filter

Pooling Layers

Action

- Reduce dimensionality
- Extract maximum of average of a region
- Sliding window approach

Parameters

- Stride
- Size of window

I/O

- Input: 3D cube, previous set of feature maps
- Output: 3D cube, one 2D map per filte, reduced spatial dimensions

Fully connected Layers

Action

- Aggregate information from final feature maps
- Generate final classification

Parameters

- Number of nodes
- Activation function: usually changes depending on role of layer. If aggregating info, use ReLU. If producing final classification, use Softmax.

I/O

- Input: FLATTENED 3D cube, previous set of feature maps
- Output: 3D cube, one 2D map per filter

Fully connected layers are placed before the classification output of a CNN and are used to flatten the results before classification.
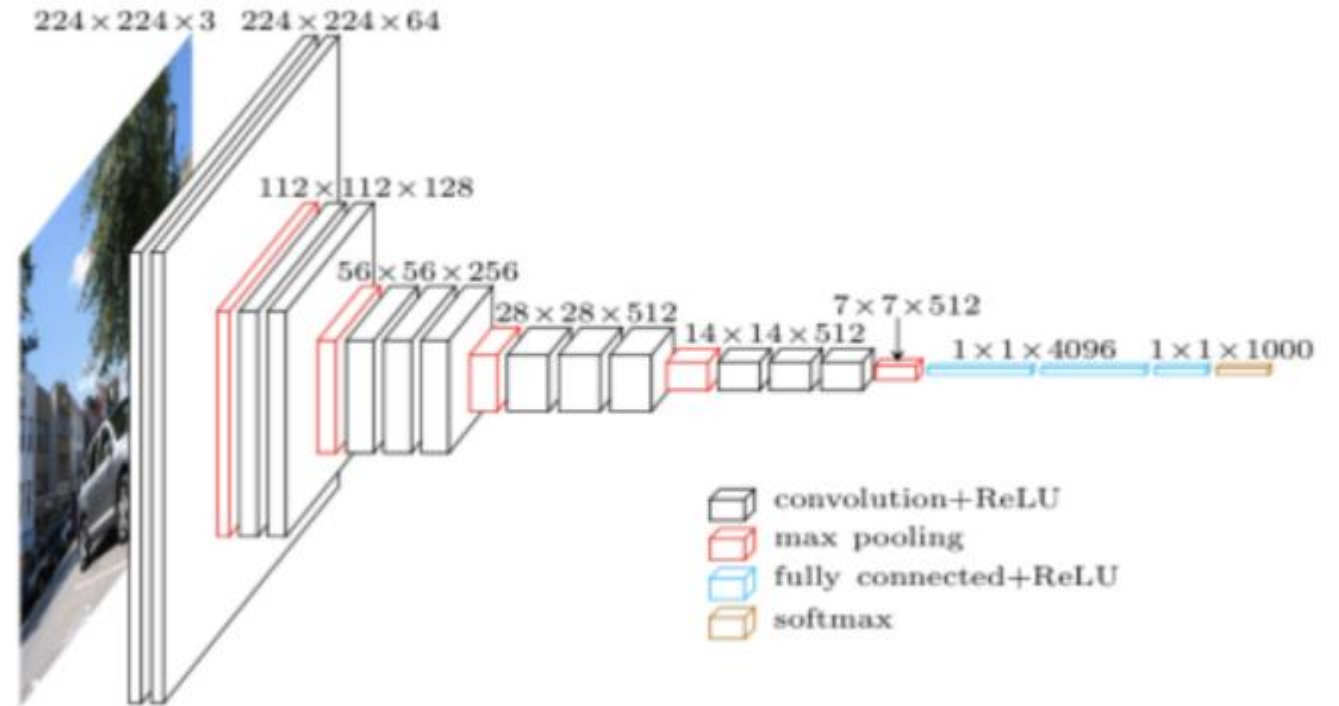

The architecture of a standard CNN.

What do CNN layers learn?

- Each CNN layer learns filters of increasing complexity.
- The first layers learn basic feature detection filters: edges, corners, etc
- The middle layers learn filters that detect parts of objects. For faces, they might learn to respond to eyes, noses, etc
- The last layers have higher representations: they learn to recognize full objects, in different shapes and positions.
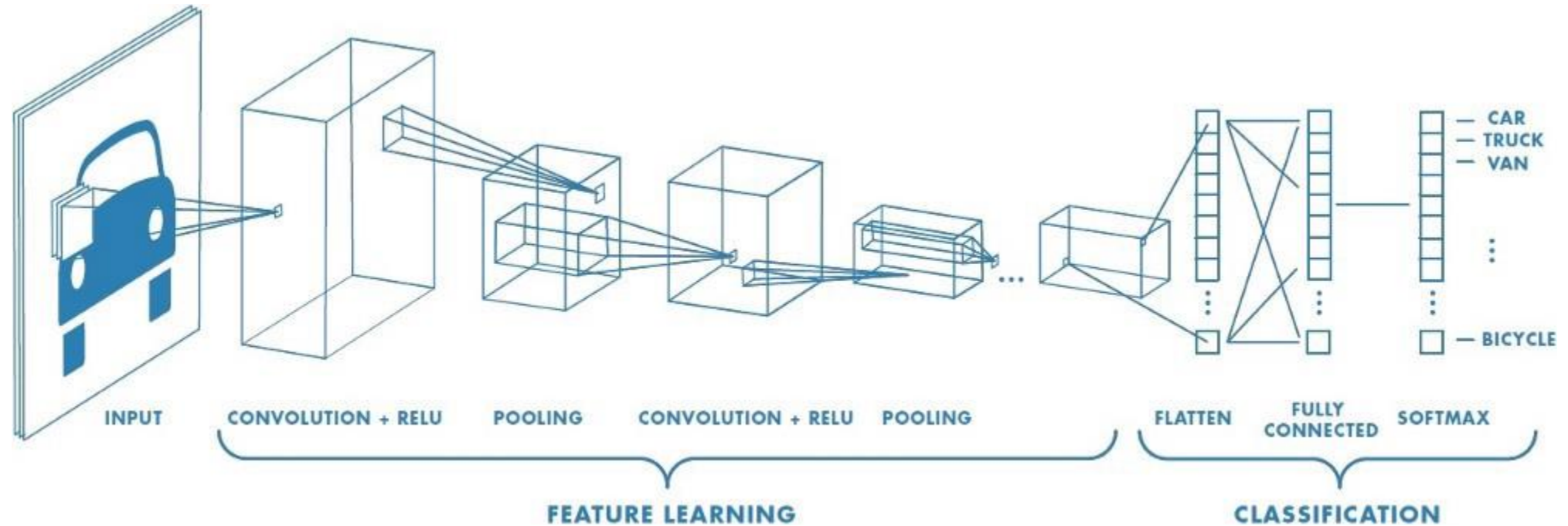
Fully connected layer

- The main purpose of the artificial neural network is to combine our features into more attributes. These will predict the classes with greater accuracy. This combines features and attributes that can predict classes better.
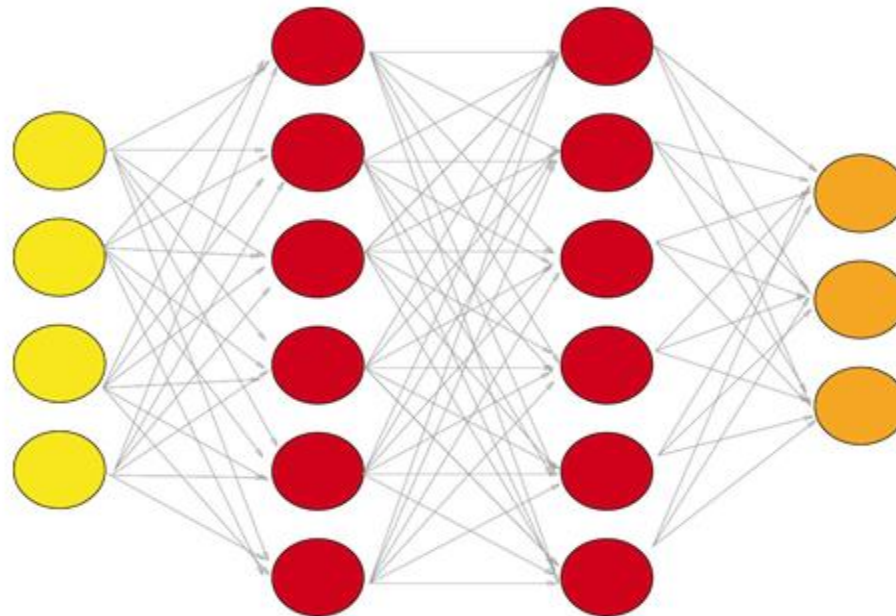
A fully connected layer with two hidden layers

Summary of the CNN Flow:

A CNN

- starts with an input image
- applies many different filters to it to create a feature map
- applies a ReLU function to increase non-linearity
- applies a pooling layer to each feature map
- flattens the pooled images into one long vector.
- inputs the vector into a fully connected artificial neural network.
- processes the features through the network. The final fully connected layer provides the "voting" of the classes that we're after.
- trains through forward propagation and backpropagation for many, many epochs. This repeats until we have a well-defined neural network with trained weights and feature detectors.

YOLO – You Only Look Once

- LeNet-5
- AlexNet
- VGG-16
- Inception-v1
- Inception-v3
- ResNet-50
- Xception
- YOLO-Darket
- Inception-ResNets

Among these, YOLO is found to be the fastest and most efficient Architechture.
YOLO v3 uses a variant of Darknet, which originally has 53 layer network trained on Imagenet. For the task of detection, 53 more layers are stacked onto it, giving us a **106 layer fully convolutional underlying architecture for YOLO v3**.
The Deeper the better!!!

- Darknet is a framework written using C and CUDA which enables easy implementation on various CPU and GPU.

- Yolo is the state-of-the-art object detection algorithm:

- Bounding box prediction • Class prediction • Prediction across scales • Feature extraction • Output Result

How it works?

Yolo initially divides the image into a grid of 13 by 13 cells. And each cells can now predict upto 5 bounding boxes with confident scores. Once the bounding boxes are framed with certain threshold, the image is then applied to the pretrained images at multiple locations and the highest scoring regions are considered for the output.
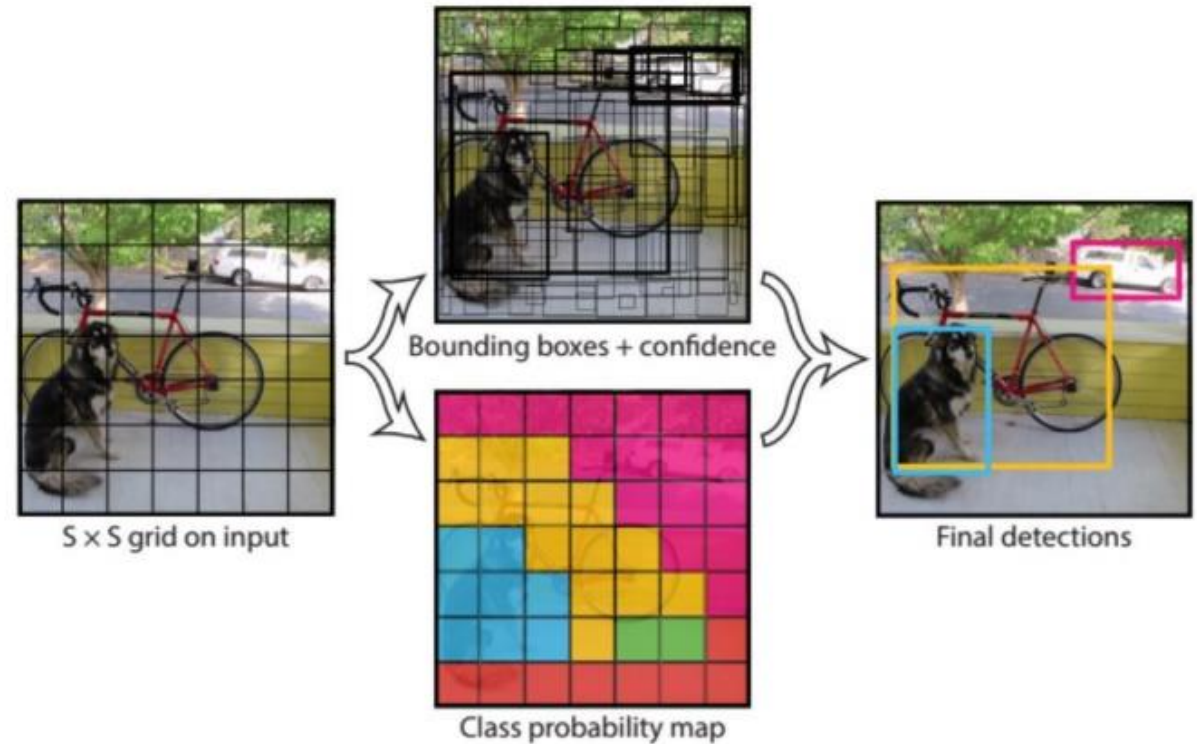

S × S grid on input

Bounding boxes + confidence

Class probability map

Final detections
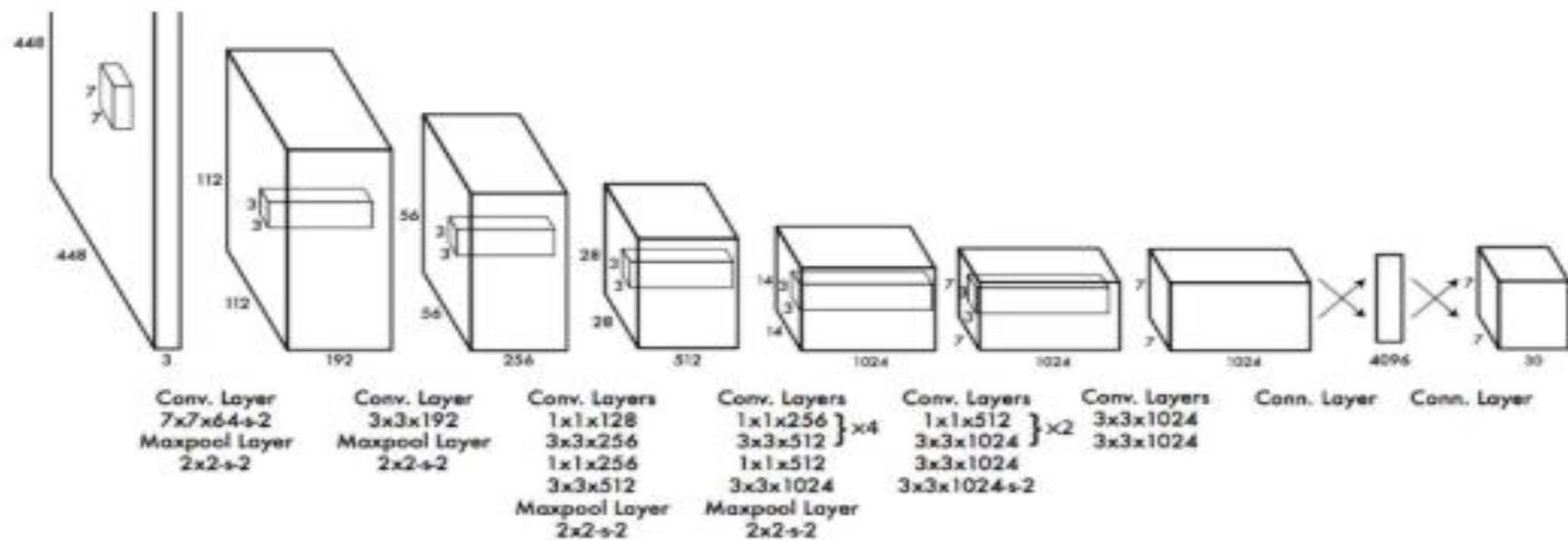
FIG 2: OUTLINE ARCHITECTURE OF YOLO [11]

Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 × 1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224 × 224 input image) and then double the resolution for detection.