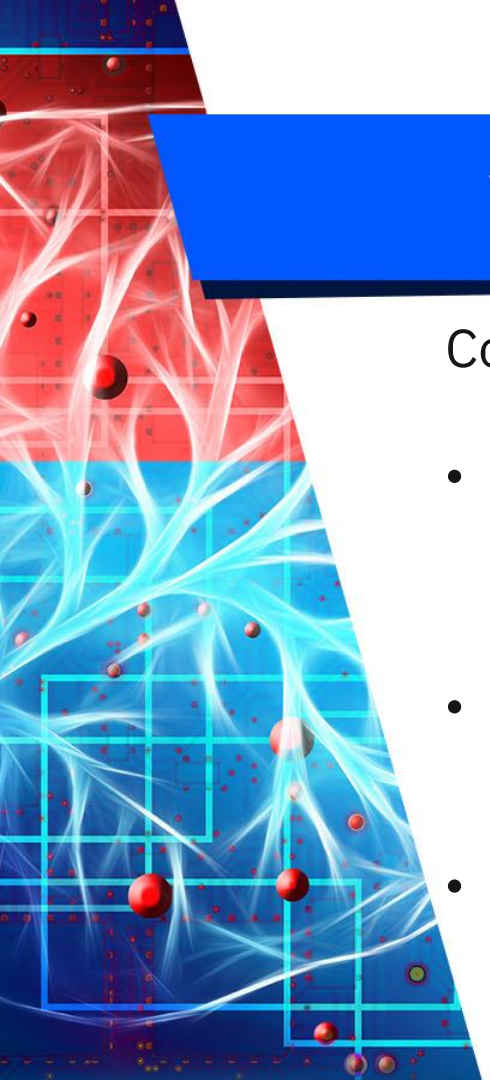
Deep Learning for Computer Vision

By
Mona M. Soliman
SRGE Member
Faculty of computers and Artificial Intelligence

Agenda

- Computer Vision
- Machine Learning
- Deep Learning
- Computer Vision Applications



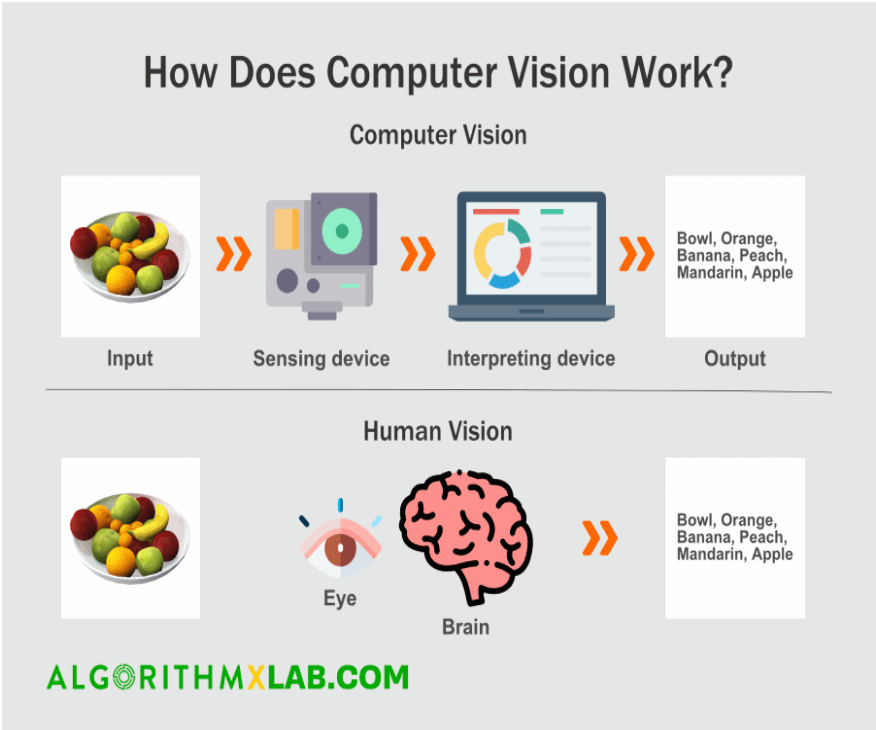
What is Computer Vision?

Computer vision is a field of artificial intelligence (AI) that :

- Enables computers and systems to derive meaningful information from digital images, videos and other visual inputs.
- Take actions or make recommendations based on that information.
- AI enables computers to think, computer vision enables them to see, observe and understand.

Computer Vision Vrs. Human Vision

- Computer vision works much the same as human vision, except humans have a head start.
- Human sight has the advantage of lifetimes of context to train how to tell objects apart, how far away they are, whether they are moving and whether there is something wrong in an image.
- Computer vision trains machines to perform these functions, but it has to do it in much less time with cameras, data and algorithms rather than retinas, optic nerves and a visual cortex.



Computer Vision Applications

Applications of Computer Vision

Visual Navigation

Augmented Reality

Optical Character Recognition

Panorama Stitching

3D Reconstruction

Games

3D Object Recognition

Surveillance

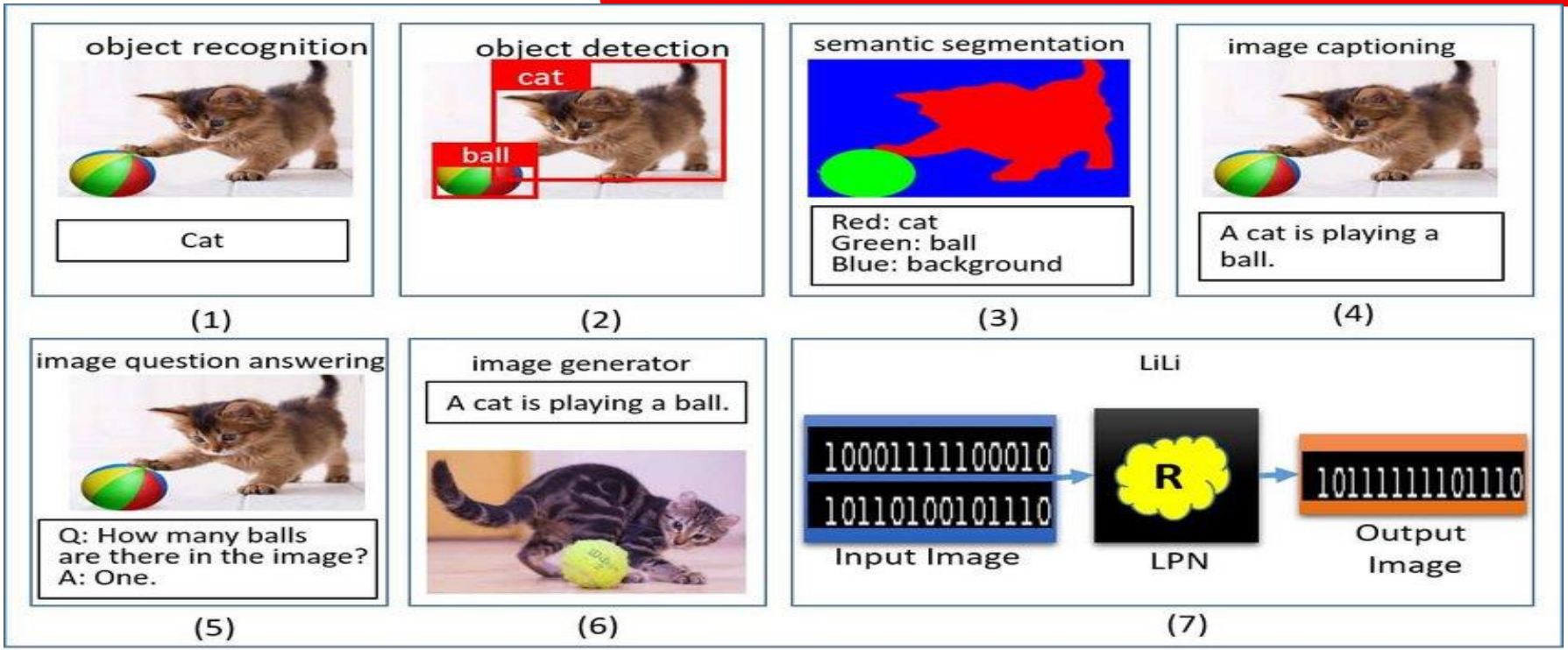
Segmentation

Human Computer Interfaces

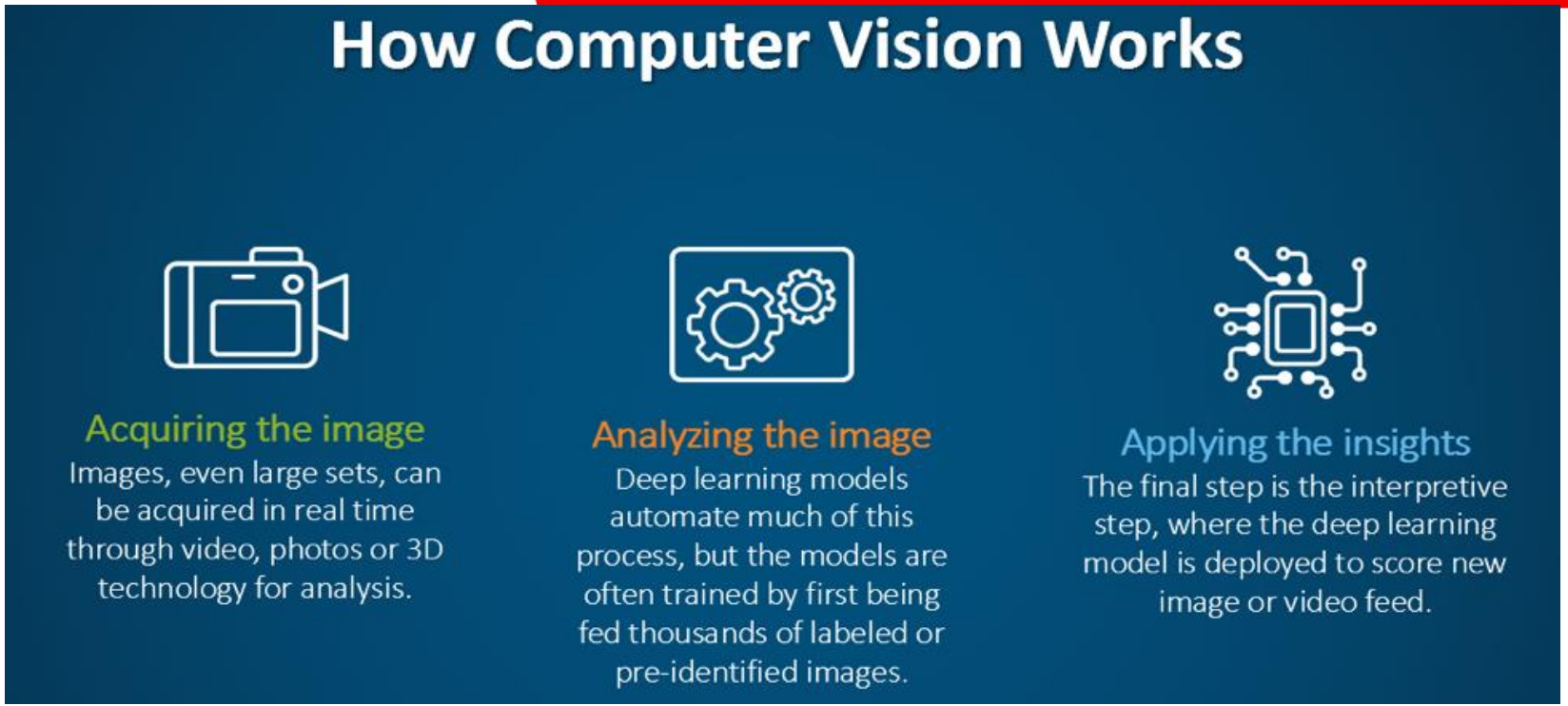
Inspection

Automation

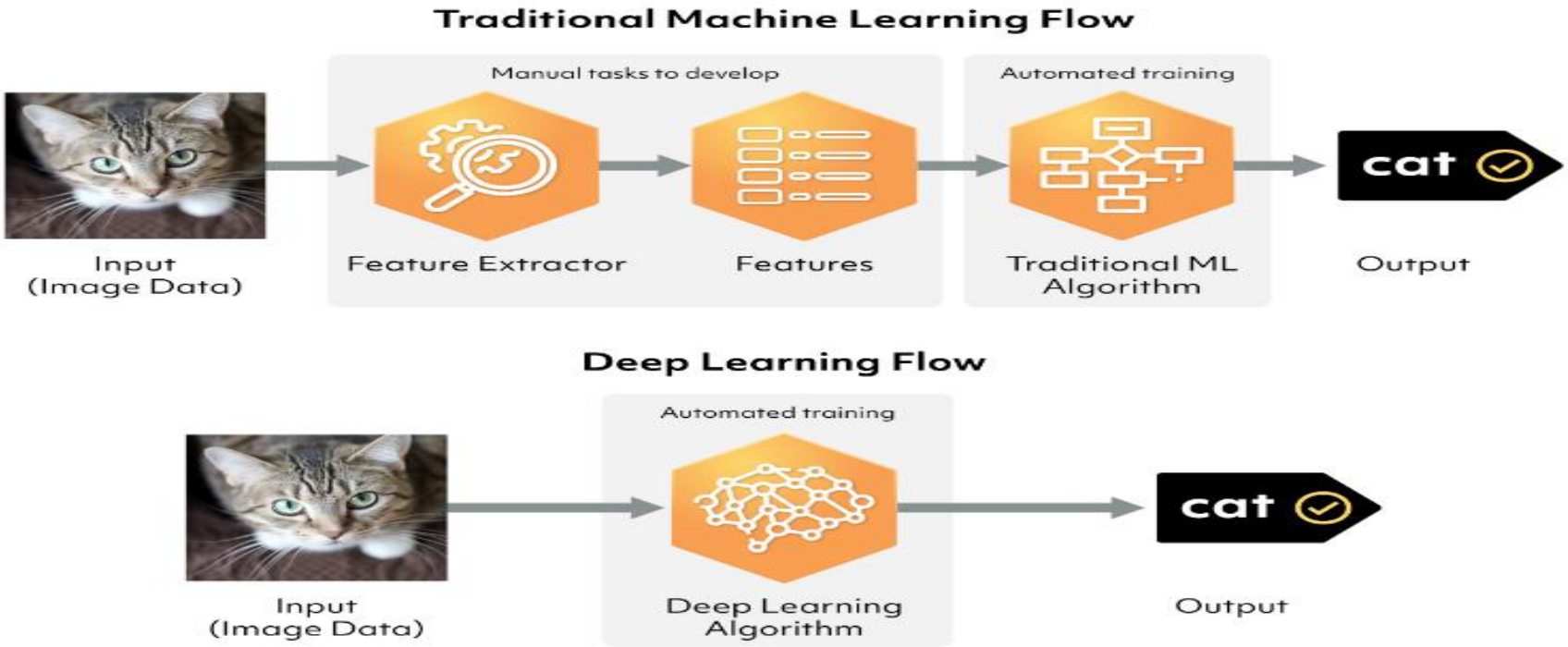
Computer Vision Examples



How Computer Vision Works

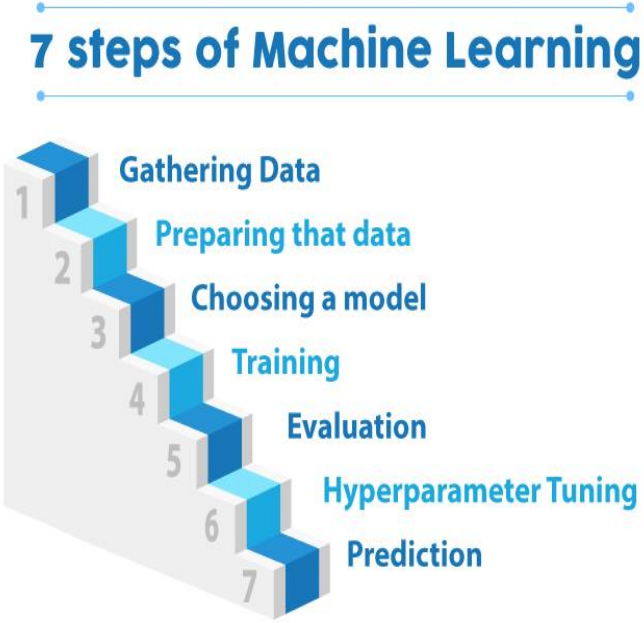


Machine Learning Vrs Deep Learning



Deep learning workflow for computer vision.

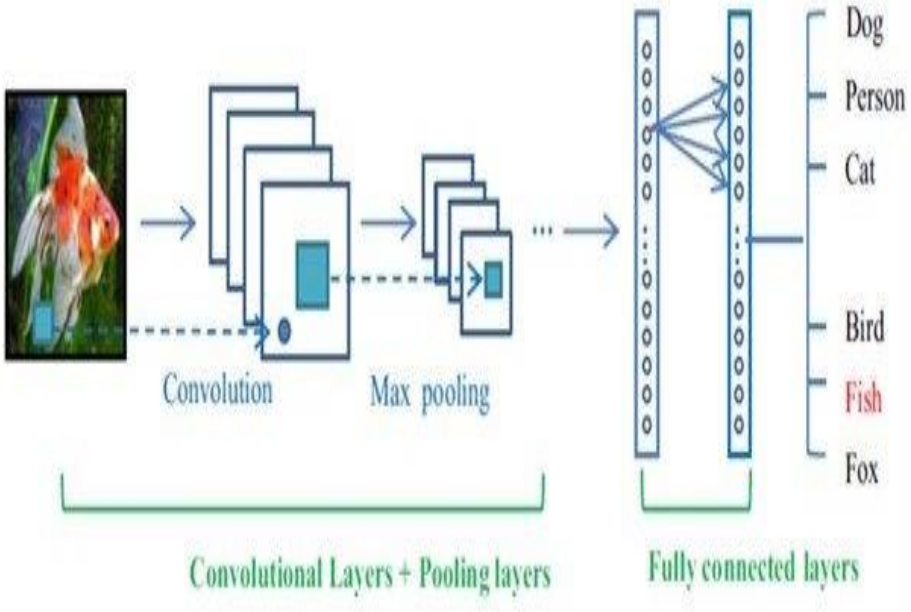
How ML works?



Training	Test Model
Load data	Load test data
Extract features	Extract features
Train model	Predict using model
Evaluate	Result

How DL works?

- Deep learning is a class of machine learning algorithms in the form of a neural network that uses a cascade of layers (tiers) of processing units to extract features from data and make predictive guesses about new data

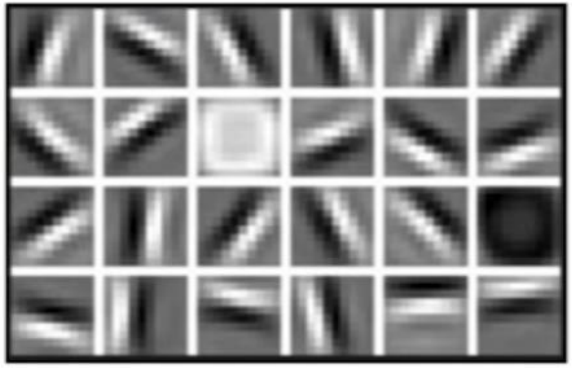


Why Deep learning?

Hand engineered features are time consuming, brittle, and not scalable in practice

Can we learn the **underlying features** directly from data?

Low Level Features



Lines & Edges

Mid Level Features

Eyes & Nose & Ears

High Level Features

Facial Structure

Why Deep Learning Now?

1952 Stochastic Gradient Descent

1958 Perceptron
- Learnable Weights

1986 Backpropagation
- Multi-Layer Perceptron

1995 Deep Convolutional NN
- Digit Recognition

Neural Networks date back decades, so why the resurgence?

1. Big Data
- Larger Datasets
- Easier Collection & Storage

IMAGENET

WIKIPEDIA The Free Encyclopedia

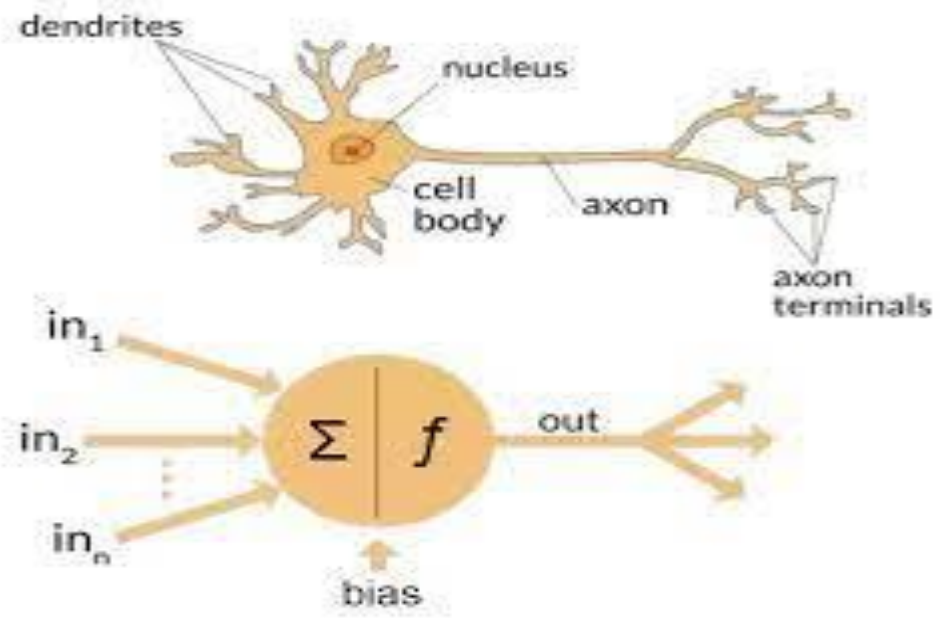
2. Hardware
- Graphics Processing Units (GPUs)
- Massively Parallelizable

3. Software
- Improved Techniques
- New Models
- Toolboxes

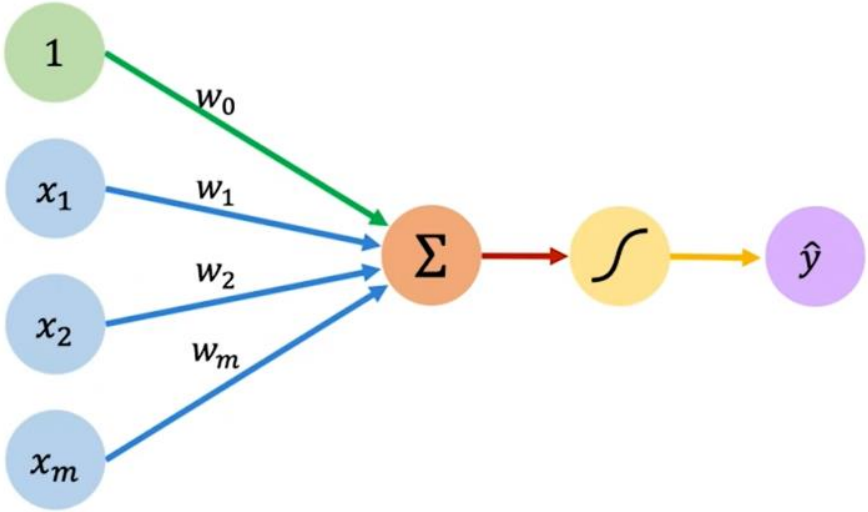
TensorFlow

The structure building block of DL

- What is an Artificial Neural Network? Collection of connected units called artificial Organized in layers of signaling cascades
- Each neuron transmits a signal to another neuron
- Neurons may have a weight that varies as learning proceeds, which can increase or decrease the strength of the signal that it sends downstream



The perceptron



Output

Linear combination of inputs

$$\hat{y} = g\left(w_0 + \sum_{i=1}^{m} x_i \, w_i \right)$$

Non-linear activation function

Bias

$$\hat{y} = g\left(w_0 + \boldsymbol{X}^T \boldsymbol{W} \right)$$

where: $\boldsymbol{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$ and $\boldsymbol{W} = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}$

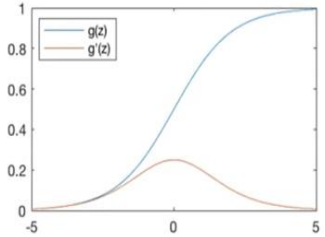
Activation Function

$$\hat{y} = g(w_0 + X^T W)$$

where: $X = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$ and $W = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}$

Common Activation Functions

Sigmoid Function

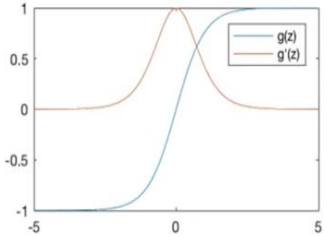


$$g(z) = \frac{1}{1 + e^{-z}}$$

$$g'(z) = g(z)(1 - g(z))$$

`tf.math.sigmoid(z)`

Hyperbolic Tangent

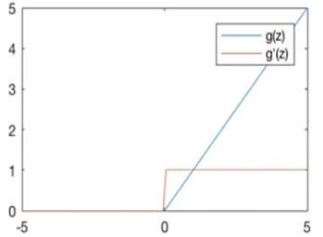


$$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

$$g'(z) = 1 - g(z)^2$$

`tf.math.tanh(z)`

Rectified Linear Unit (ReLU)



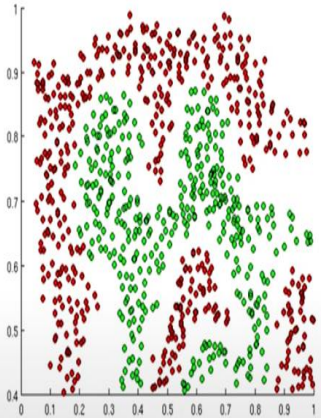
$$g(z) = \max(0, z)$$

$$g'(z) = \begin{cases} 1, & z > 0 \\ 0, & \text{otherwise} \end{cases}$$

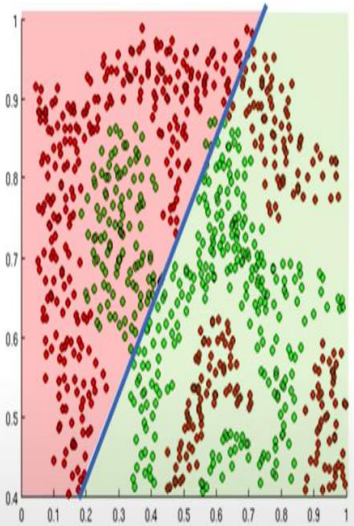
`tf.nn.relu(z)`

Activation Function Cont.

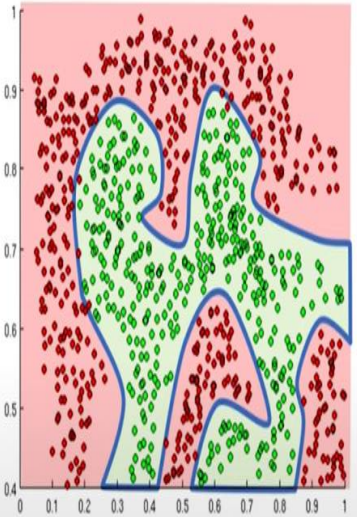
The purpose of activation functions is to ***introduce non-linearities*** *into the network*



What if we wanted to build a neural network to distinguish green vs red points?



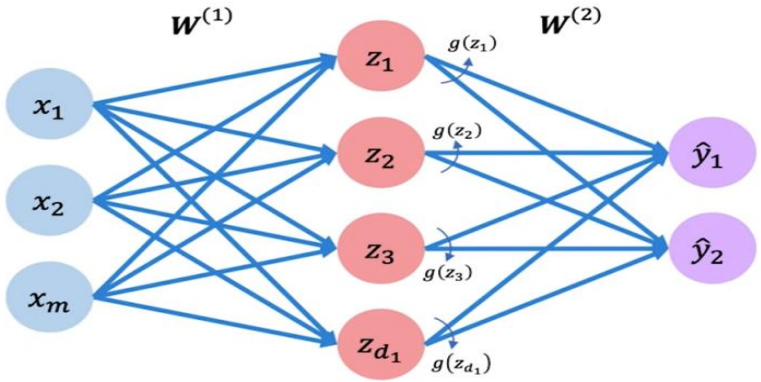
Linear activation functions produce linear decisions no matter the network size



Non-linearities allow us to approximate arbitrarily complex functions

Single Layer NN

Single Layer Neural Network

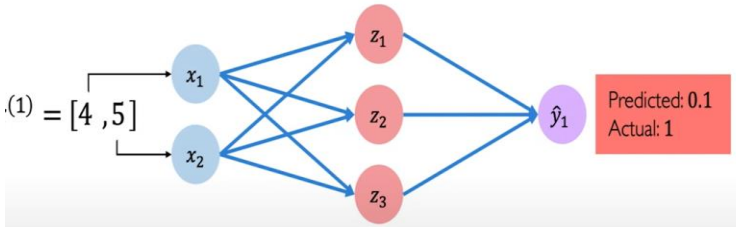


$$z_i = w_{0,i}^{(1)} + \sum_{j=1}^{m} x_j \, w_{j,i}^{(1)} \qquad \hat{y}_i = g\left(w_{0,i}^{(2)} + \sum_{j=1}^{d_1} g(z_j) \, w_{j,i}^{(2)}\right)$$

Loss Function

Quantifying Loss

*The **loss** of our network measures the cost incurred from incorrect predictions*

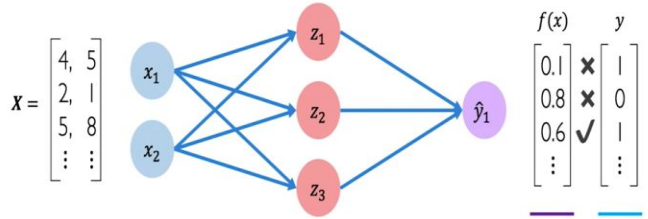


$$\mathcal{L}\left(f\left(x^{(i)}; \boldsymbol{W}\right), y^{(i)}\right)$$

Predicted Actual

Empirical Loss

*The **empirical loss** measures the total loss over our entire dataset*



Also known as:
- Objective function
- Cost function
- Empirical Risk

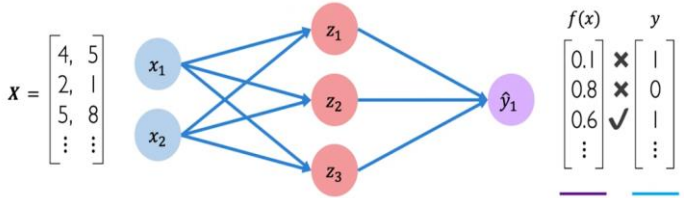
$$J(\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}\left(f\left(x^{(i)}; \boldsymbol{W}\right), y^{(i)}\right)$$

Predicted Actual

Types of Loss Functions

Binary Cross Entropy Loss

Cross entropy loss *can be used with models that output a probability between 0 and 1*

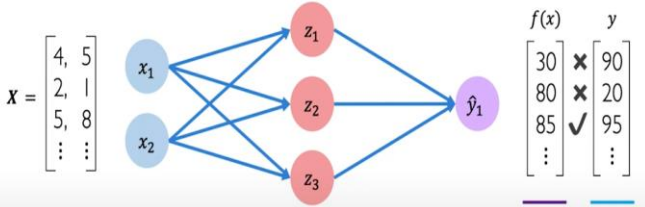


$$J(\boldsymbol{W}) = -\frac{1}{n}\sum_{i=1}^{n} y^{(i)} \log\left(f(x^{(i)};\boldsymbol{W})\right) + (1-y^{(i)}) \log\left(1-f(x^{(i)};\boldsymbol{W})\right)$$

Actual Predicted Actual Predicted

Mean Squared Error Loss

Mean squared error loss *can be used with regression models that output continuous real numbers*



$$J(\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} \left(y^{(i)} - f(x^{(i)};\boldsymbol{W})\right)^2$$

Actual Predicted

Training Neural Network

Loss Optimization

We want to find the network weights that ***achieve the lowest loss***

$$W^* = \underset{W}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\left(f\left(x^{(i)}; W\right), y^{(i)}\right)$$

$$W^* = \underset{W}{\operatorname{argmin}} J(W)$$

Remember:

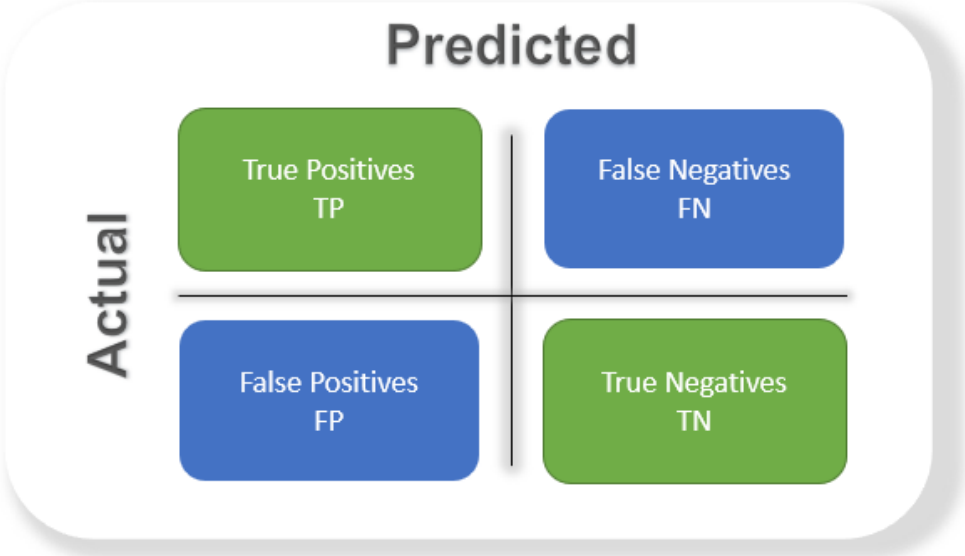
$$W = \{W^{(0)}, W^{(1)}, \dots\}$$

Gradient Descent

Algorithm

1. Initialize weights randomly $\sim \mathcal{N}(0, \sigma^2)$
2. Loop until convergence:
3. Compute gradient, $\frac{\partial J(W)}{\partial W}$
4. Update weights, $W \leftarrow W - \eta \frac{\partial J(W)}{\partial W}$
5. Return weights

Model Evaluation



Metric	Formula
True positive rate, recall	$\frac{TP}{TP+FN}$
False positive rate	$\frac{FP}{FP+TN}$
Precision	$\frac{TP}{TP+FP}$
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
F-measure	$\frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$

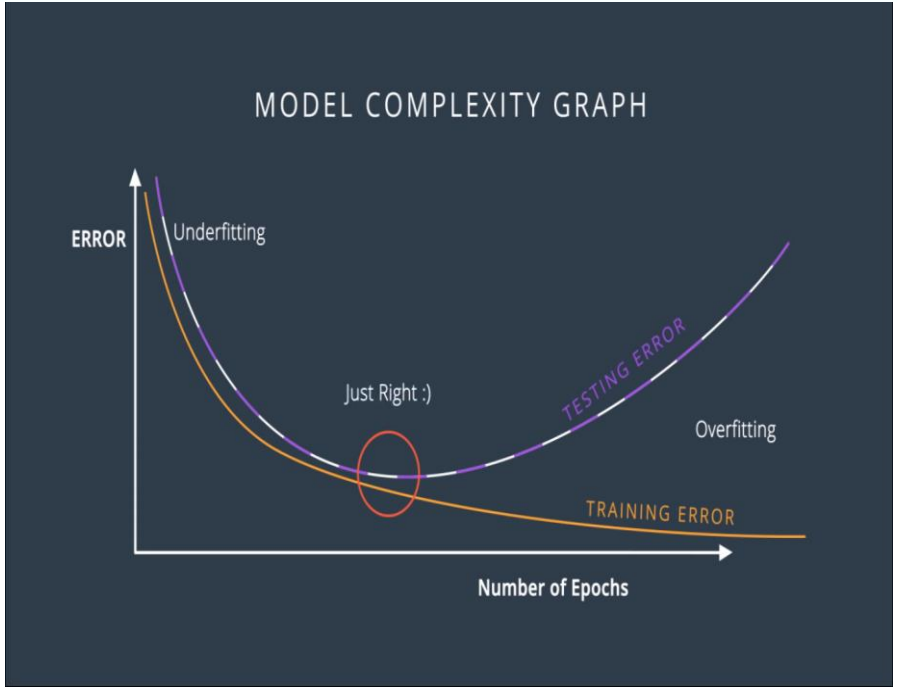
Model Performance

Reducing Overfitting

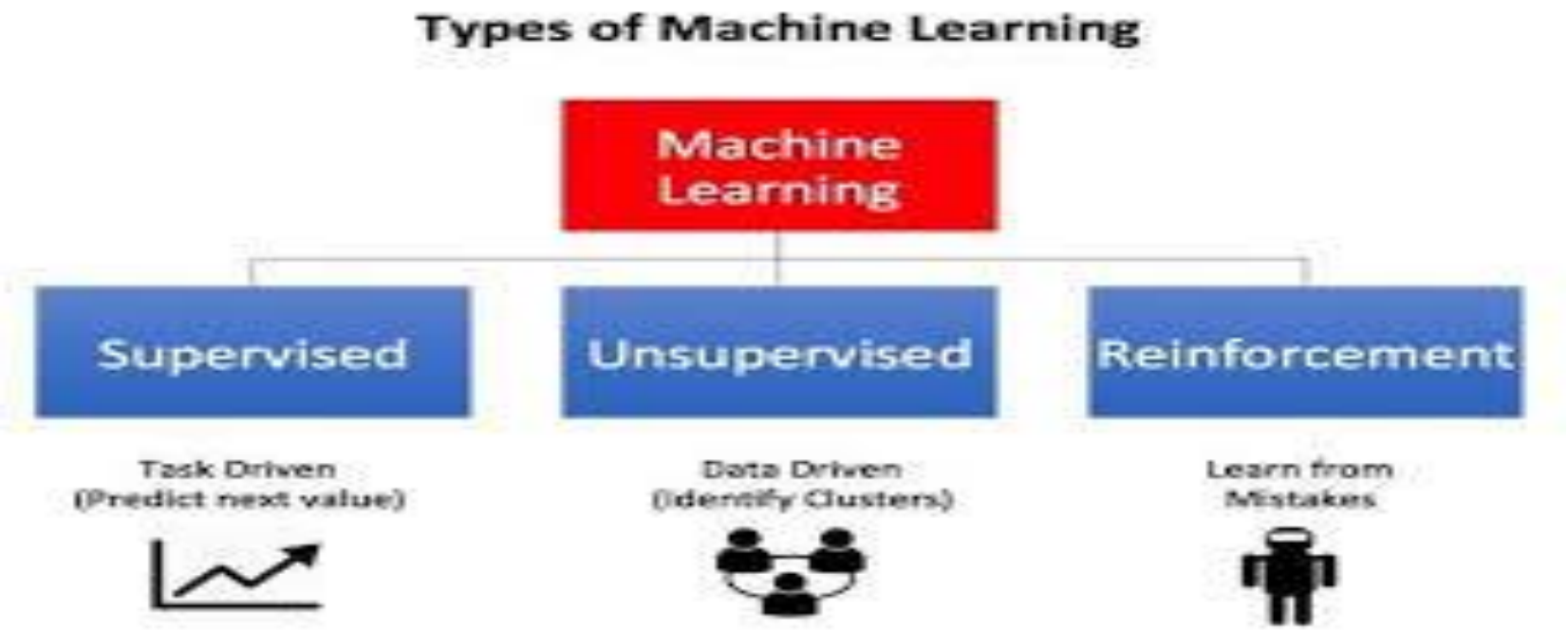
- Increase Training Data
- Reduce Model Complexity
- Early stopping during Training Phase
- L1 and L2 regularization
- Dropouts for Neural Network

Reducing Underfitting

- Increase Training Data
- Increase complexity of Model
- Increase no. of features
- Remove Noise from data
- Increase no. of training epochs

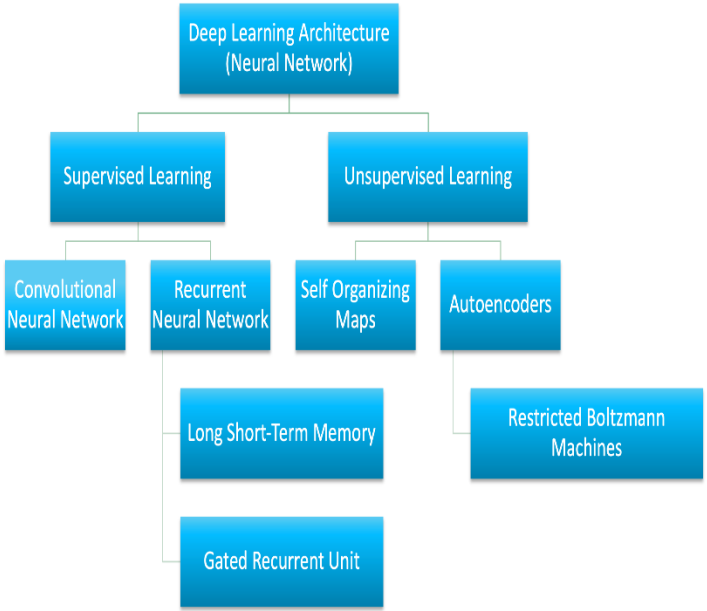


Types of Learning



Types of DL neural network

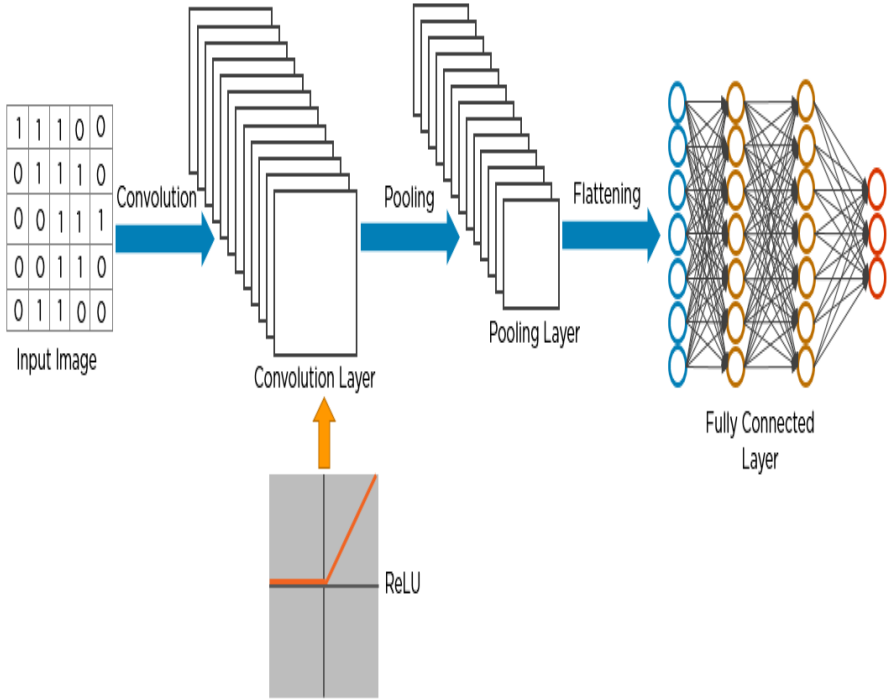
Supervised Learning	Convential Neural Network (CNN)	AlexNet,2012
		ZFNet 2013
		VGGNet 2014
		GoogLeNet 2015
		ResNet 2015
		Inception-ResNet 2017,
		Xception 2017
		PolyNet 2017.
Unsupervised Learning	Auto-Encoder	Sparse Autoencoders ,2013
		Denoising Autoencoders
		Contractive Autoencoders
		Variational Autoencoders
	Recurrent neural networks	Stacked RNN
		LSTM
	Deep Generative Network	Deep Belief Network (DBN)
		Deep Boltzmann Machine
		Generative Adversarial Networks
Reinforcement Learning	Q-Learning	



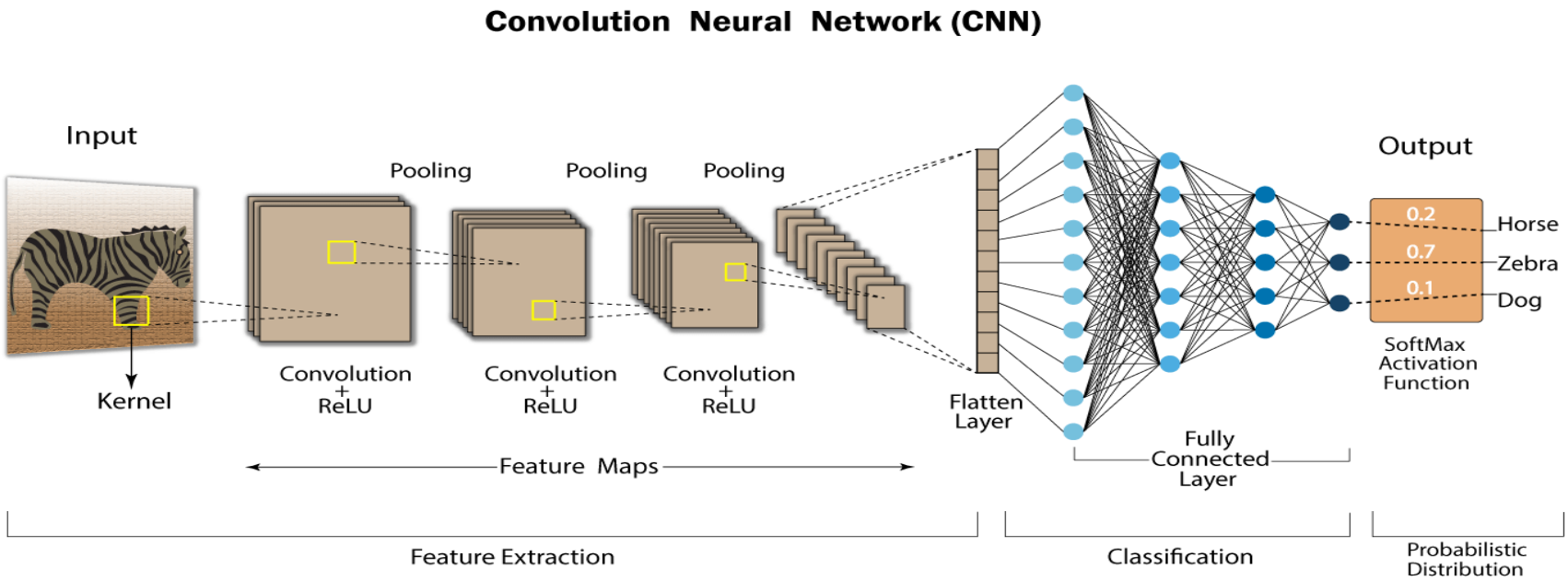
Convolutional Neural Network

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs.
They have three main types of layers, which are:

- *Convolutional layer*
- *Pooling layer*
- *Fully-connected (FC) layer*

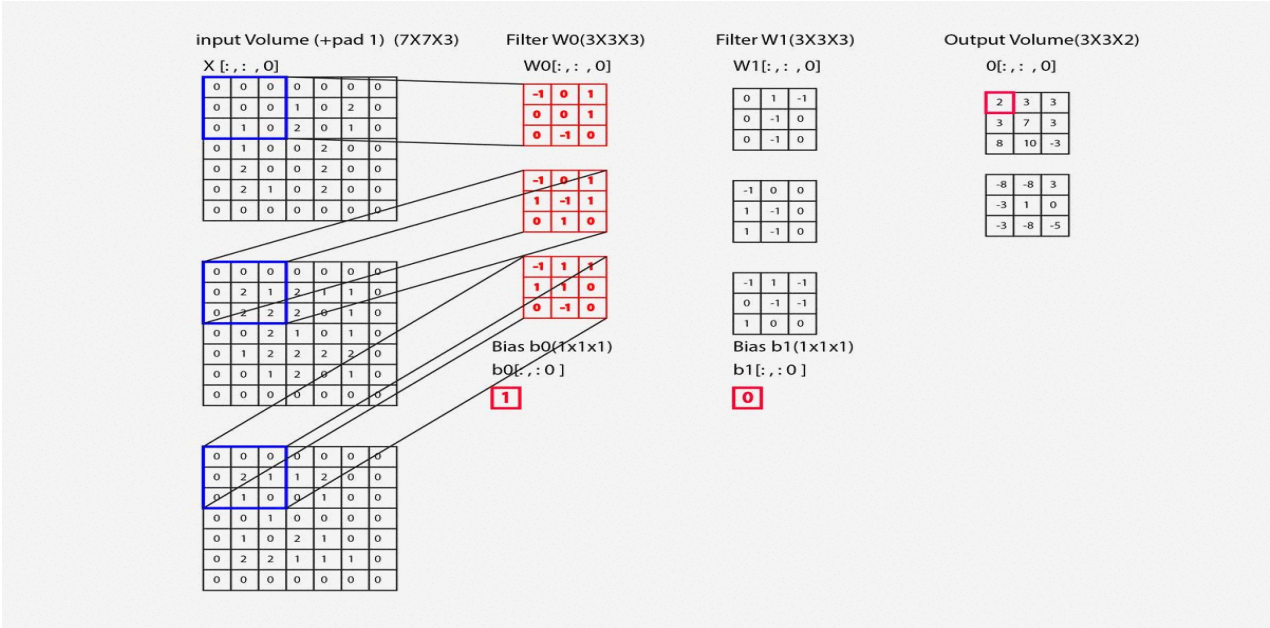


Convolution Neural Network



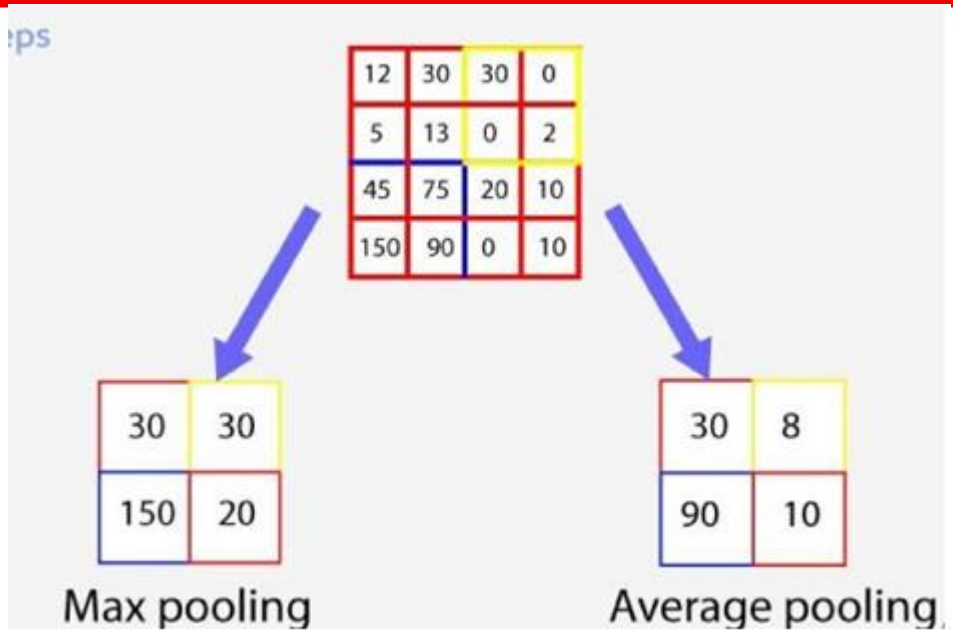
Convolution Layer

The convolutional layer's main objective is to extract features from images and learn all the features of the image which would help in object detection techniques.



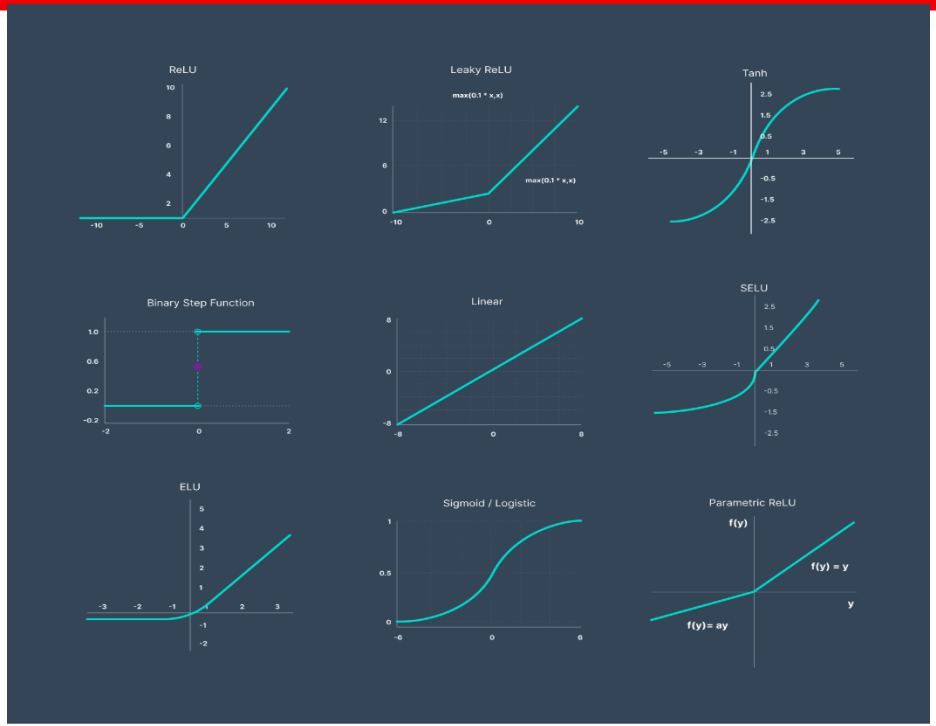
Pooling Layer

- After convolution, we perform pooling to reduce the number of parameters and computations.
  There are different types of pooling operations, the most common ones are max pooling and average pooling.

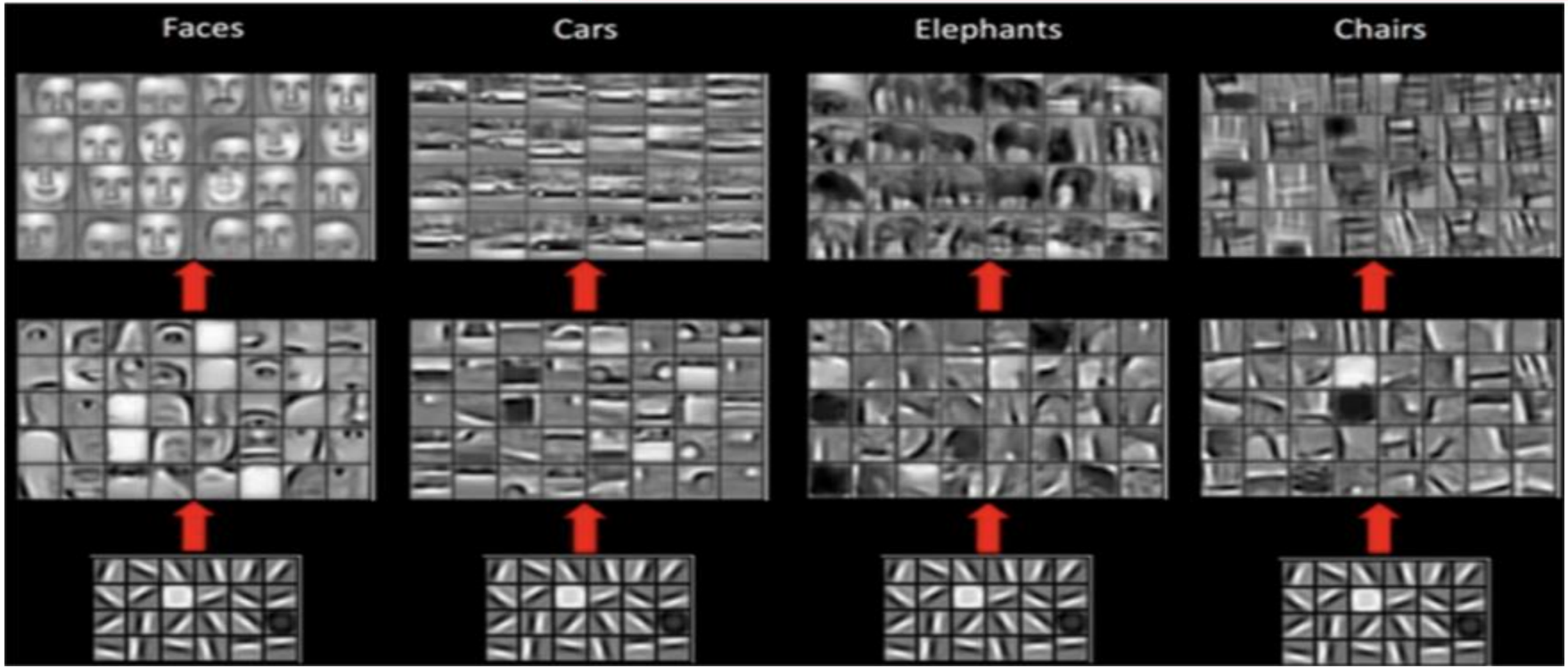


Activation Function

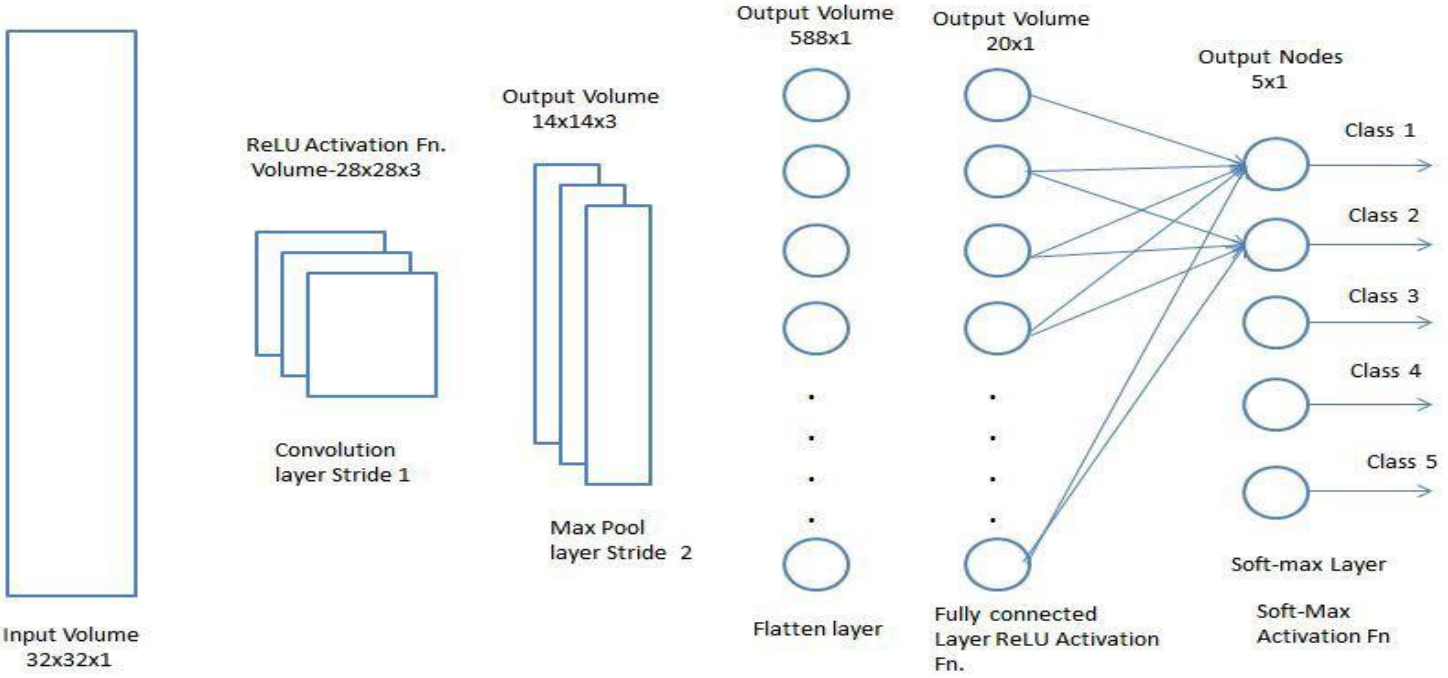
- It helps in making the decision about which information should fire forward and which not by making decisions at the end of any network.



Convolutional layers: Feature Extraction



Fully Connected layer



Case Study -1: Action Loclization

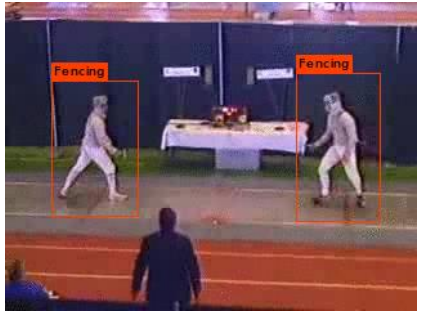
Real-time Action localization aims to localizing an action and predicting its class label in online streaming video

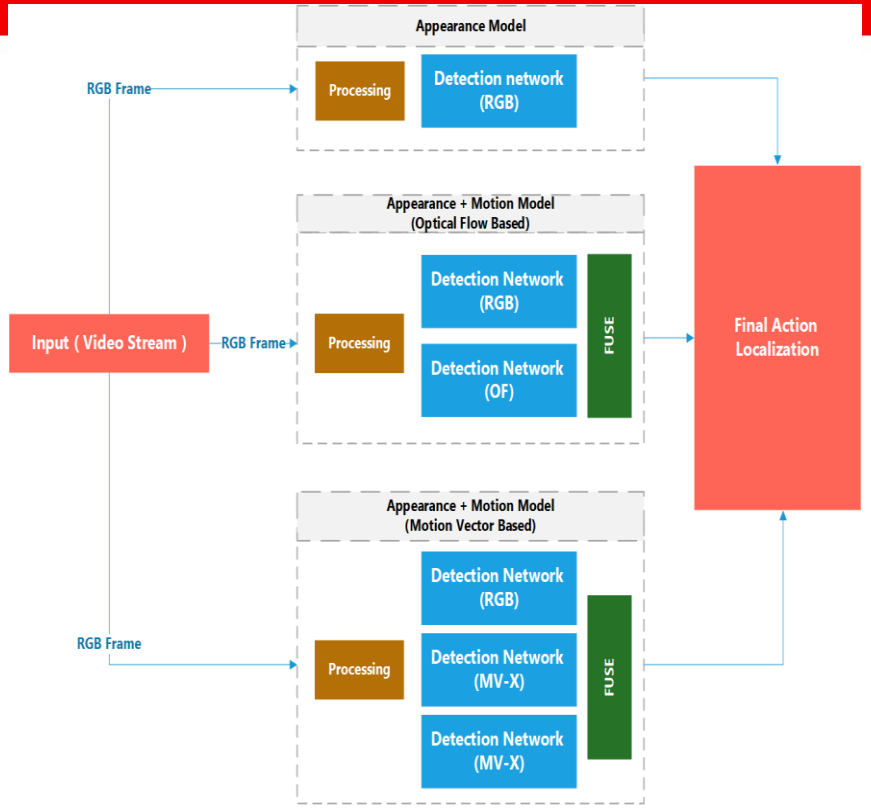
Video Class is Basketball

+

Bounding Box on the action

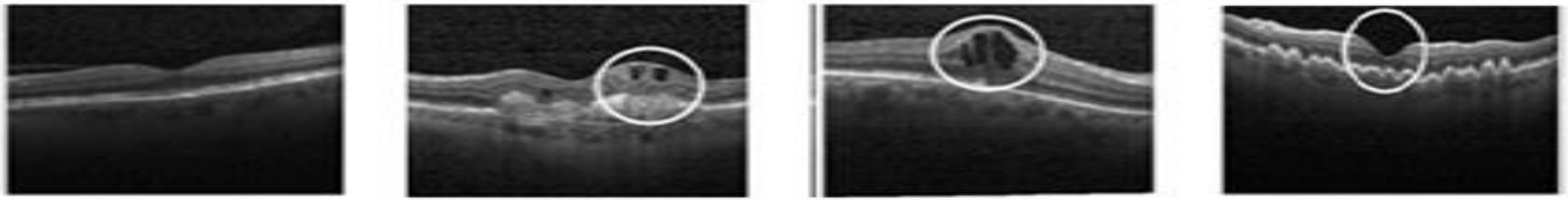
Action localization Model and results





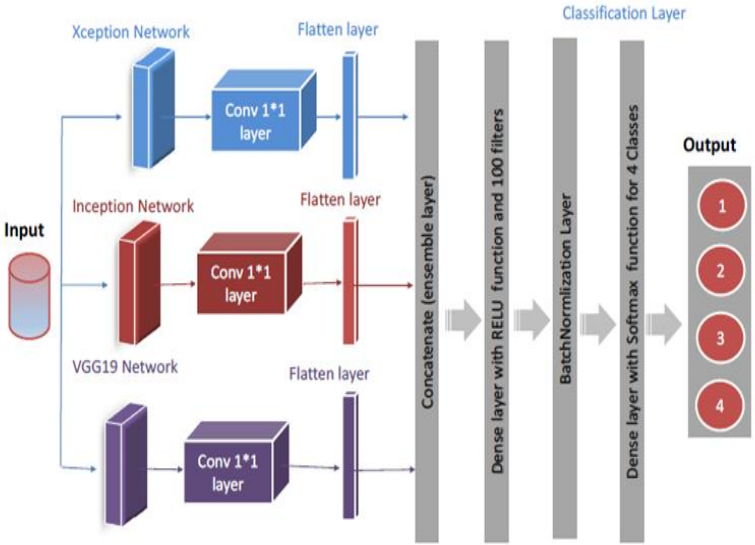
Case Study 2: Retinal Diseases Diagnosis

- Retinal diseases are the most common cause of early-age loss of eyesight. Most of them trigger retinal visual symptoms.

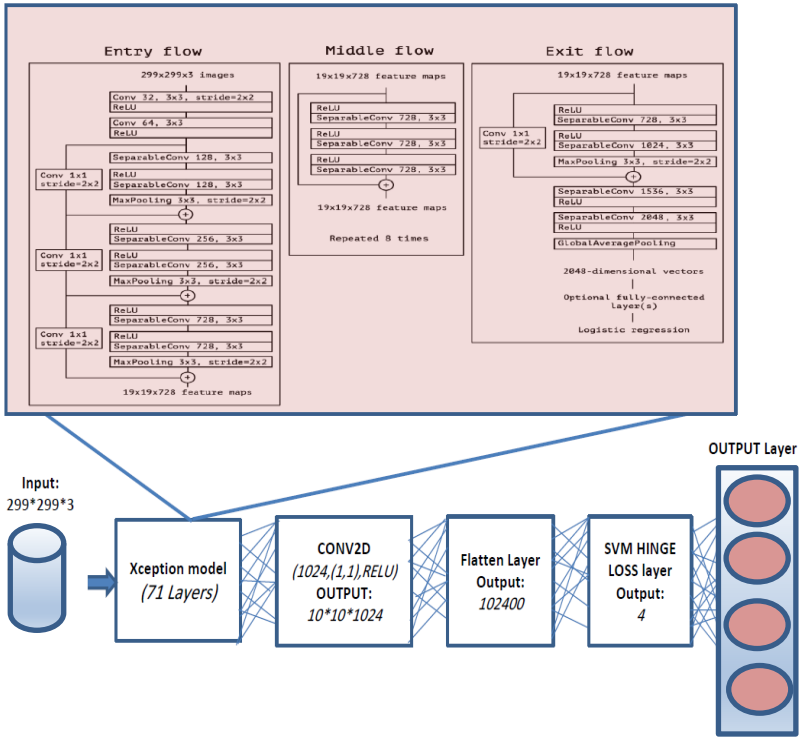


Representative (OCT) images with circle indicated to the lesion sites.

Retinal dieses Model



The overall architecture of end-to-end ensemble model



Retinal Diseases Results

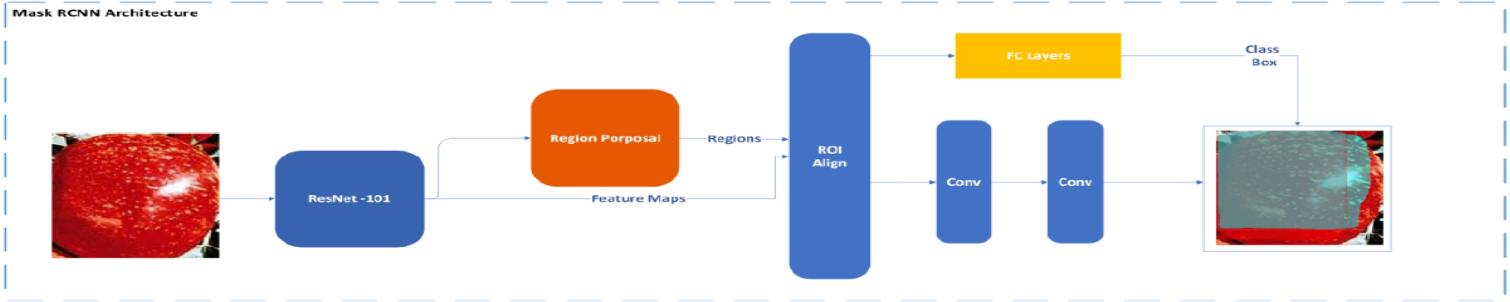
Model	Dataset	Accuracy %
Inception V3 based model with Hinge loss	OCT Images	93
Xception based model with Hinge loss	OCT Images	98
TL " Xception " with softmax	OCT Images	95.8
TL " Inception-V3 " with softmax	OCT Images	91.9
TL " VGG19 " with softmax	OCT Images	93.0
Average Ensemble Learning	OCT Images	94.5
Proposed Ensemble Learning	**OCT Images**	**99.9**

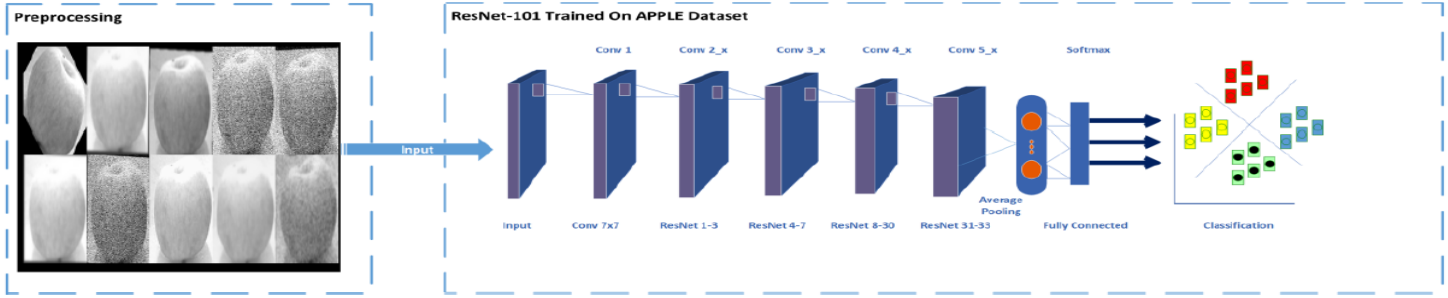
Case Study 3: Food Sorting

- **Deep Learning for Apple Diseases:**

Diseases and pests cause huge economic loss to the apple industry every year. The identification of various apple diseases is challenging for the farmers as the symptoms produced by different diseases may be very similar, and may be present simultaneously. This paper is an attempt to provide the timely and accurate detection and identification of apple diseases.

Apple Diseases Classification Model





Apple Disease classification Results

Rot	Scab	Blotch

Thank You

mona.solyman@fci-cu.edu.eg