

# Discovering Deep Learning

Date : 3rd June 2020 | Speaker : Ayon Roy | Event : Webinar by Tech Table

Visit - [AYONROY.ML](http://AYONROY.ML)

# Hello Buddy!

I am **Ayon Roy**

**B.Tech CSE ( 2017-2021 )**

Data Science Intern @ **Lulu International Exchange**, Abu Dhabi  
( **World's Leading Financial Services Company** )

Brought **Kaggle Days Meetup** Community in India for the 1st time

**If you haven't heard about me yet, you might have been living under the rocks. Wake up !!**

# Agenda ( 3-6-2020 )

- AI, ML, DL in simple words
- What is Deep Learning ( DL ) ?
- Where DL is used ?
- Key components of DL
- DL Libraries
- Why DL is growing ?
- Challenges in DL
- Resources to learn DL

**LET'S GET STARTED!**





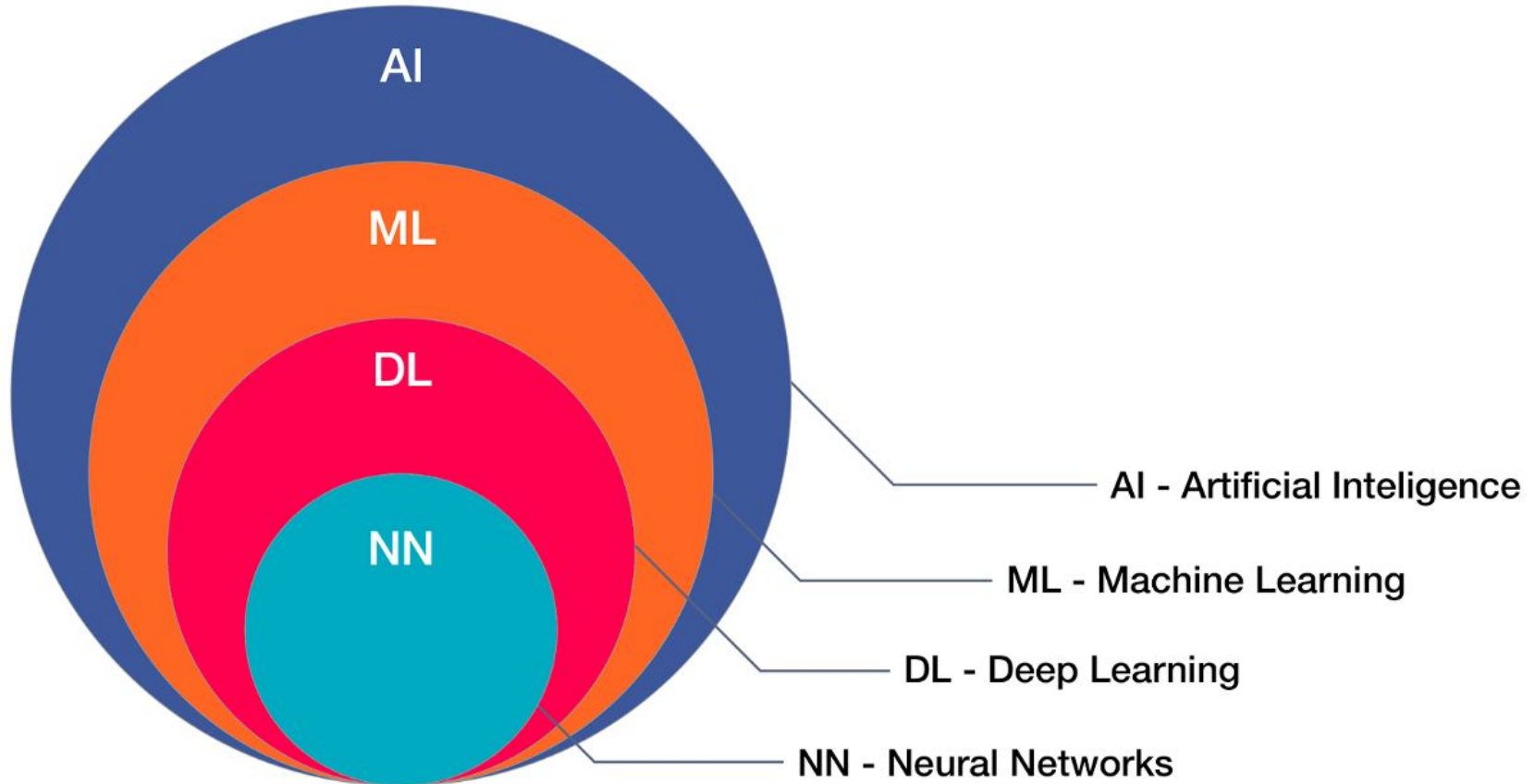
# AI, ML, DL in simple words

**Artificial Intelligence** – Human Intelligence Exhibited by Machines

**Machine Learning** – An Approach to Achieve Artificial Intelligence

**Deep Learning** – A Technique for Implementing Machine Learning

# Graphical Representation of AI, ML, DL



# In words of NVIDIA

The easiest way to think of their relationship is to visualize them as concentric circles with AI – the idea that came first – the largest, then machine learning – which blossomed later, and finally deep learning – which is driving today's AI explosion – fitting inside both.

<https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/>



What is **Deep Learning** ?

It is the fastest growing segment of Artificial Intelligence.

It uses many **layered neural networks** to learn levels of representation & abstraction that makes sense of data.

A better view for **ML & DL**

**Classic ML algos** solved many problems that rule-based programs struggled with, but they are poor at dealing with soft data such as images, video, sound files, and unstructured text. We would have to do a lot of feature engineering etc. in these processes.

After that, the engineers use ML on top of the extracted features. Creating such an AI model **takes years**.

**DL algos** solve the same problem using deep neural networks in comparatively less time.

**Where Deep Learning is  
used ?**

**Computer Vision:** To make sense of the content of images and video

**Voice and Speech Recognition:** When you utter a command to your Amazon Echo smart speaker or your Google Assistant, deep-learning algorithms convert your voice to text commands.

**Natural Language Processing (NLP) and Generation (NLG):** To extract the meaning of unstructured text, which has been a historical pain point for classic software.

# Key components of Deep Learning

# Neural Networks



## What are Neural Networks ?

Computer System inspired by biological networks of neurons that learn progressively i.e which improves performance to do tasks; by considering examples generally without task specific programming.

## Why Neural Networks ?

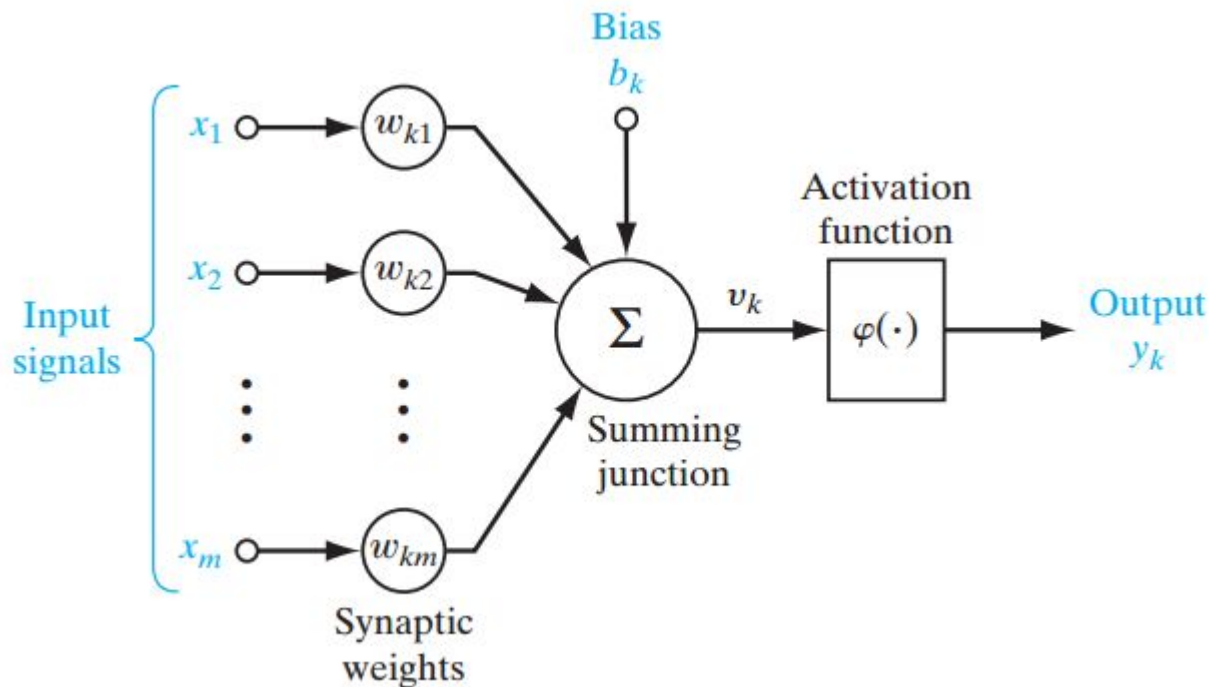
Neural Networks learn by example. So now computers can do things what we don't exactly know how to do.

# Perceptron

The elementary entity of a neural network.

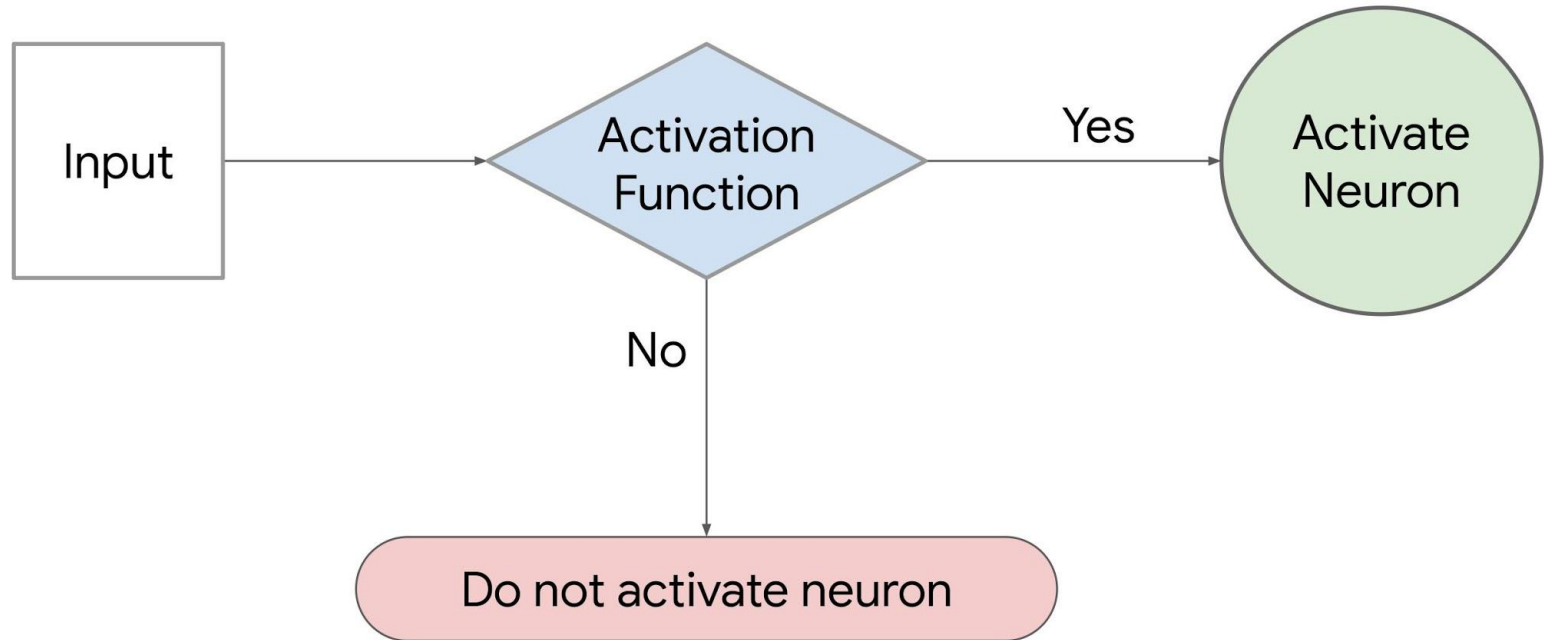
The most basic form of neural network which can also learn

# Single Perceptron

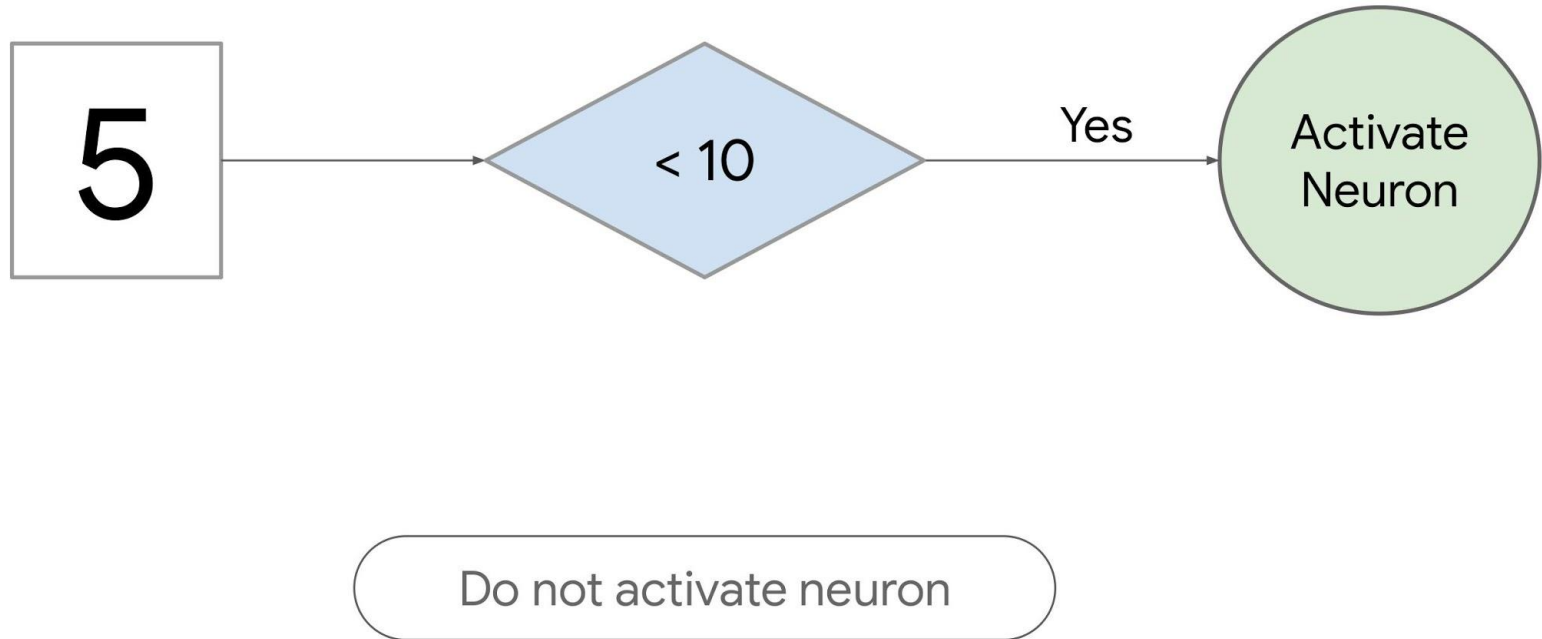


Many such Perceptrons combine together to form a Neural Network

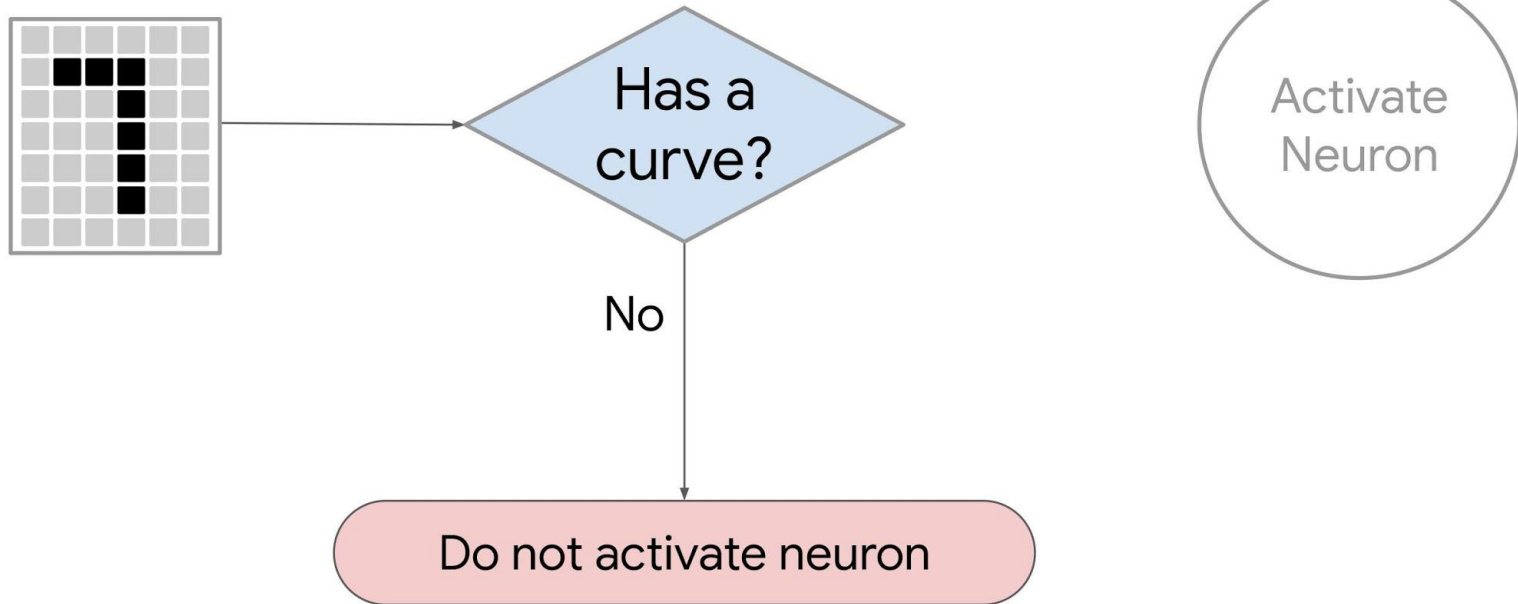
## Perception of Artificial Neuron



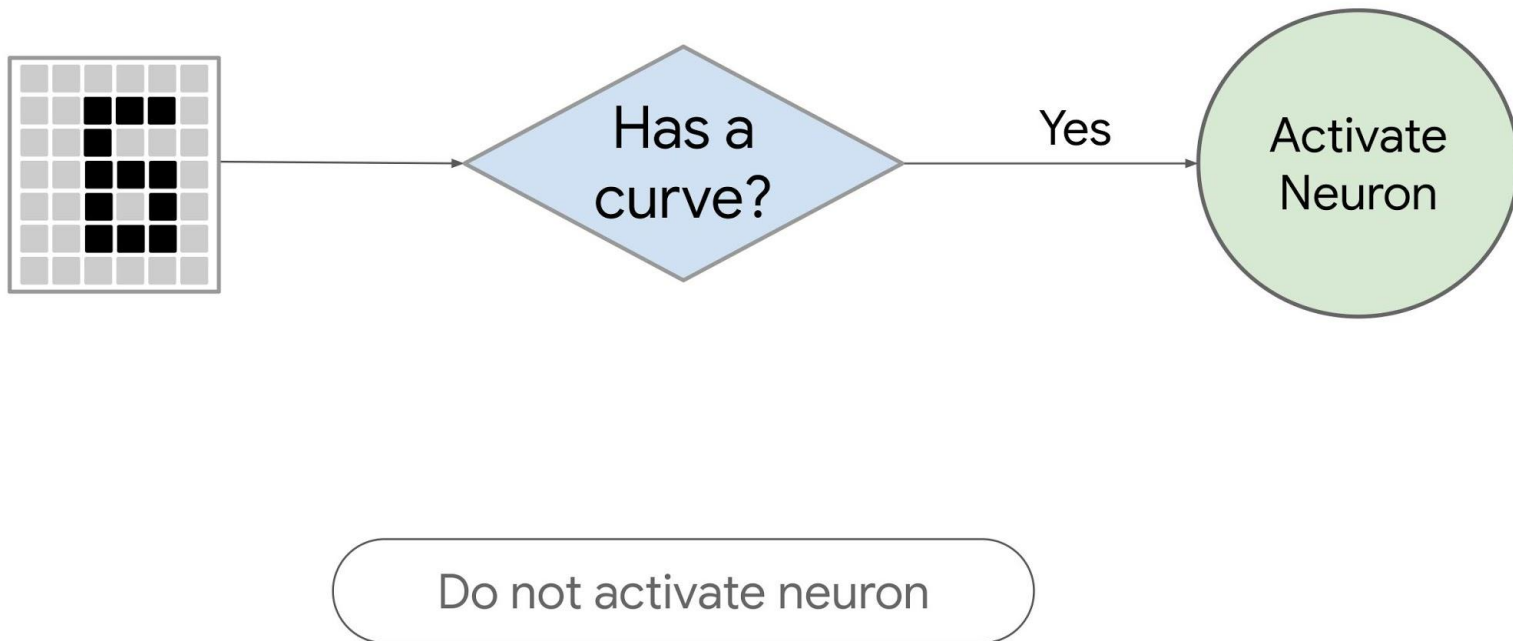
## Perception of Artificial Neuron



# Perception of Artificial Neuron



# Perception of Artificial Neuron



# Basic Terminologies

One **epoch** = one forward pass and one backward pass of *all* the training examples

**Batch size** = the number of training examples in one forward/backward pass. The higher the batch size, the more memory space you'll need.

No. of **iterations** = number of passes, each pass using [batch size] number of examples. To be clear, one pass = one forward pass + one backward pass (we do not count the forward pass and backward pass as two different passes).

## Example

**if you have 1000 training examples, and your batch size is 500, then it will take 2 iterations to complete 1 epoch.**



## Loss Function

Method of evaluating how well your algorithm models your dataset. If the predictions are totally off, it will output a higher number. If they're pretty good, it'll output a lower number. As you change pieces of your algorithm to try and improve your model, your loss function will tell you if you're getting anywhere.

## Optimizers

During the training process, we change the parameters (weights) of our model to minimize the loss function, and make our predictions better. But how to do that? Optimizers tie together the loss function and model parameters to shape and mold the model into its most accurate possible form.

**The loss function is the guide to the terrain, telling the optimizer when it's moving in the right or wrong direction.**

**Learning Rate :** The learning rate is a parameter that determines how much an updating step influences the current value of the weights. When training a neural network, if the learning rate is :

- **Too small** a learning rate and your neural network may not learn at all
- **Too large** a learning rate and you may overshoot areas of low loss (or even overfit from the start of training)

**Momentum :** It is a value between 0 and 1 that increases the size of the steps taken towards the minimum by trying to jump from a local minima.

- If the momentum term is large then the learning rate should be kept smaller.
- A large value of momentum also means that the convergence will happen fast.
- But if both the momentum and learning rate are kept at large values, then you might skip the minimum with a huge step.
- A small value of momentum cannot reliably avoid local minima, and can also slow down the training of the system.

A right value of momentum can be either learned by hit and trial or through cross-validation.

**Flatten** : The process of converting all the resultant 2 dimensional arrays into a single long continuous linear vector. This means you can combine all the found local features of the previous convolutional layers. Each feature map channel in the output of a CNN layer is a "flattened" 2D array created by adding the results of multiple 2D kernels (one for each channel in the input layer).

**Batch Normalization** : A technique for improving the performance and stability of artificial neural networks. It is a technique to provide any layer in a neural network with inputs that are zero mean/unit variance. Debates are there that it does not reduce internal covariate shift, but rather smooths the objective function to improve the performance and it also helps in length-direction decoupling, and thereby accelerates neural networks.

**Decay** : An additional term in the weight update rule that causes the weights to exponentially decay to zero, if no other update is scheduled.

**Dense Layer** : The most basic neural network architecture in deep learning containing the densely connected or fully-connected layers. In this layer, all the inputs and outputs are connected to all the neurons in each layer.

**Dropout** refers to not considering neurons during particular forward or backward pass during the training phase of certain set of neurons which is chosen at random. It is done “to prevent over-fitting”. A fully connected layer occupies most of the parameters, and hence, neurons develop co-dependency amongst each other during training which curbs the individual power of each neuron leading to over-fitting of training data.

## Hyperparameter Tuning

Tuning the hyperparameters effectively can lead to a massive improvement in the overall performance.

Following are a few common hyperparameters we frequently work with in a deep neural network:

- Learning rate –  $\alpha$
- Momentum –  $\beta$
- Adam's hyperparameter –  $\beta_1, \beta_2, \epsilon$
- Number of hidden layers
- Number of hidden units for different layers
- Learning rate decay
- Mini-batch size

Learning rate usually proves to be the most important among the above. This is followed by the number of hidden units, momentum, mini-batch size, the number of hidden layers, and then the learning rate decay.

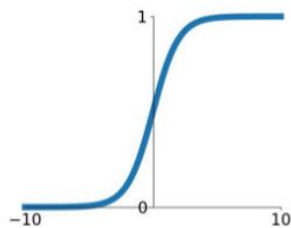
**Hyperparameter tuning relies more on experimental results than theory, and thus the best method to determine the optimal settings is to try many different combinations evaluate the performance of each model.**

**Visit - [AYONROY.ML](https://ayonroy.ml)**

# Activation Functions

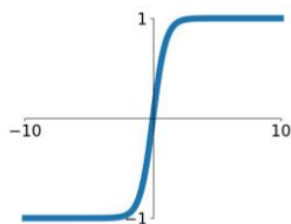
## Sigmoid

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



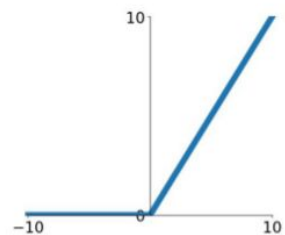
## tanh

$$\tanh(x)$$



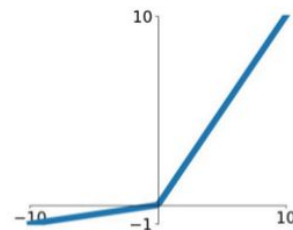
## ReLU

$$\max(0, x)$$



## Leaky ReLU

$$\max(0.1x, x)$$

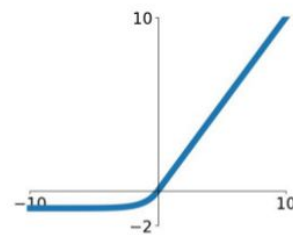


## Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

## ELU

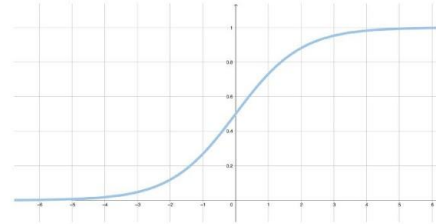
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



# Common Activation Functions

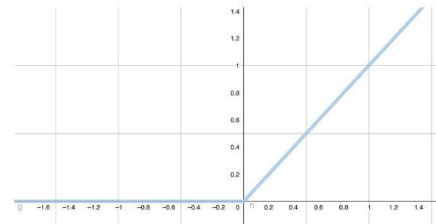
**Sigmoid** activation function converts the weighted sum to a value between **0** and **1**.

$$F(x) = \frac{1}{1 + e^{-x}}$$



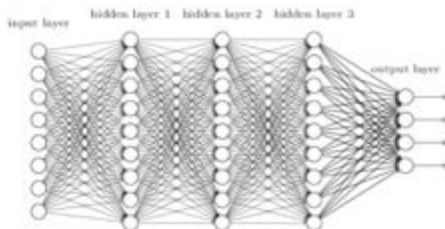
**ReLU** (Rectified Linear Unit) activation function often works a little better than a smooth function like the sigmoid, while also being significantly easier to compute.

$$F(x) = \max(0, x)$$



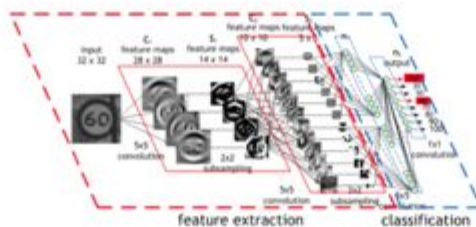
providing lift for  
classification and  
forecasting models

## Deep Neural Networks



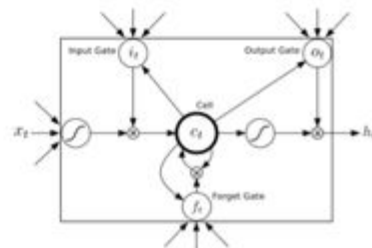
feature extraction  
and classification of  
images

## Convolutional Neural Networks



for sequence of events,  
language models, time  
series, etc.

## Recurrent Neural Networks





# Some Common Deep Learning Libraries

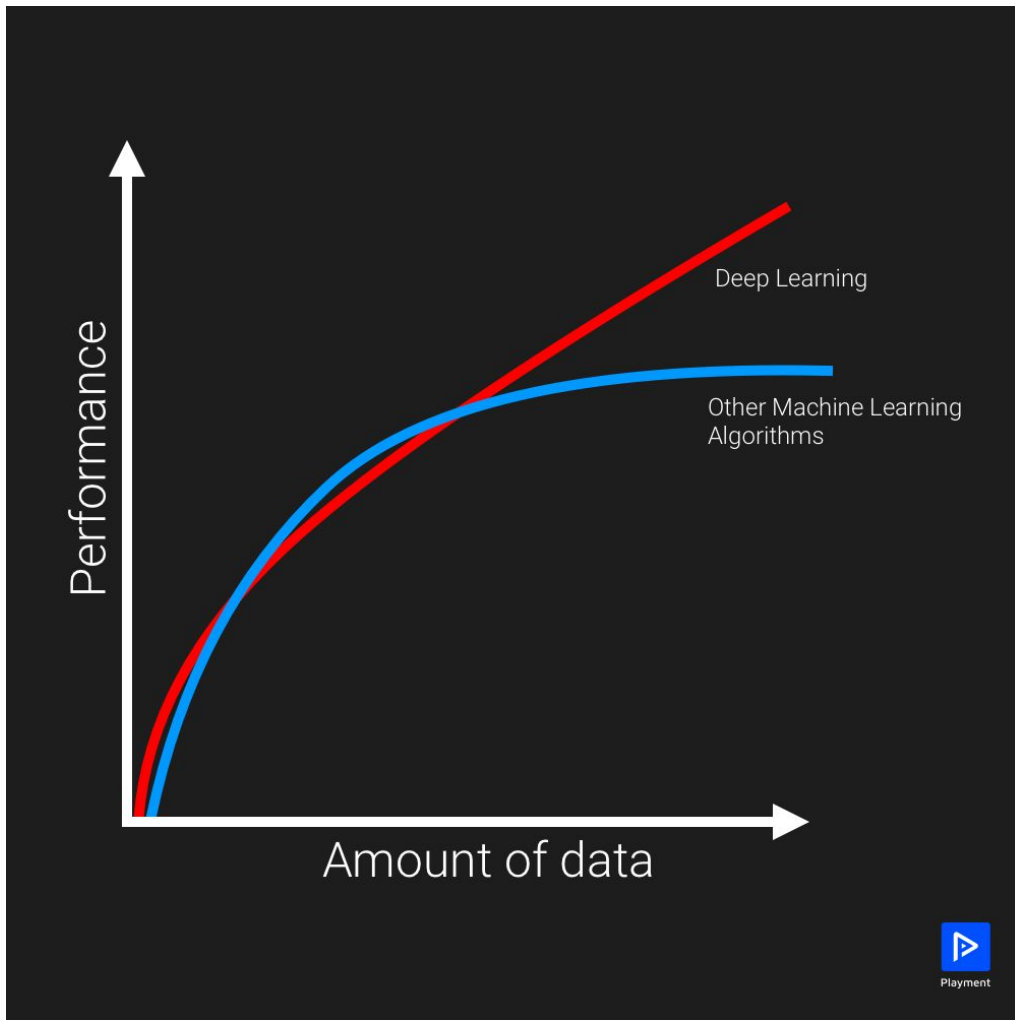
- Theano
- Tensorflow
- Keras
- PyTorch

**And Many more**



Why AI, ML, DL is growing  
now ?

1. The sharp decrease in costs associated with data storage and processing.
2. The advent of the Internet economy and the explosion in mobile apps.
3. The abundance of open-source tools.
4. The development of a wealth of innovative ML and DL algorithms.
5. Availability of GPUs etc.



Playment

Visit - [AYONROY.ML](http://AYONROY.ML)

# Challenges in Deep Learning

**Data dependency:** For many problems, there's not enough quality training data to create deep learning models.

**Lack of generalization:** Deep-learning algorithms are good at performing focused tasks but poor at generalizing their knowledge. Unlike humans, a deep learning model trained to play StarCraft won't be able to play a similar game: say, WarCraft.

**Explainability:** Neural networks develop their behavior in extremely complicated ways—even their creators struggle to understand their actions. Lack of interpretability makes it extremely difficult to troubleshoot errors and fix mistakes in deep-learning algorithms.

**Algorithmic bias:** Deep-learning algorithms are as good as the data they're trained on. The problem is that training data often contains hidden or evident biases, and the algorithms **inherit these biases**. For instance, a facial-recognition algorithm trained mostly on pictures of white people will perform less accurately on non-white people.



# Resources to learn Deep Learning

<https://www.deeplearning.ai/>

Deep Learning courses on <https://www.fast.ai/>

<https://www.slideshare.net/TessFerrandez/notes-from-course-era-deep-learning-courses-by-andrew-ng>




<https://github.com/mbadry1/DeepLearning.ai-Summary>

# INTRO TO DEEP LEARNING

INPUT LAYER: AREA, # ROOMS, LOCATION, WEALTH  
HIDDEN LAYER: 200, 100, 50  
OUTPUT LAYER: PRICE

## SUPERVISED LEARNING

INPUT: X	OUTPUT: Y	NN TYPE
HOME FEATURES	PRICE	STANDARD NN
AD+USER INFO	WILL CLICK ON AD (0/1)	
IMAGE	OBJECT (1...1000)	CONV. NN (CNN)
AUDIO	TEXT TRANSCRIPT	RECURRENT NN (RNN)
ENGLISH	CHINESE	
IMAGE/RADAR	POS OF OTHER CARS	CUSTOM/HYBRID

STRUCTURED:  UNSTRUCTURED:   "THE QUICK BROWN FOX"   
 HUMANS ARE GOOD AT THIS

### WHY NOW?

LOTS OF DATA, HARDWARE OPTIMIZED ALGOS

PERFORM. vs AMT. OF DATA (LABELED)

- LARGE NN
- MED NN
- SMALL NN
- CLASSIC ML


ONE OF THE BIG BREAKTHROUGHS HAS BEEN MOVING FROM SIGMOID TO RELU FOR FASTER GRADIENT DESCENT

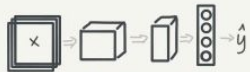
SIGMOID vs RELU

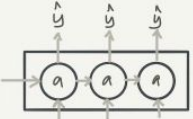
IDEA → CODE → EXPERIM. → IDEA

FASTER COMPUTATION IS IMPORTANT TO SPEED UP THE ITERATIVE PROCESS

### NETWORK ARCHITECTURES

STANDARD NN: 

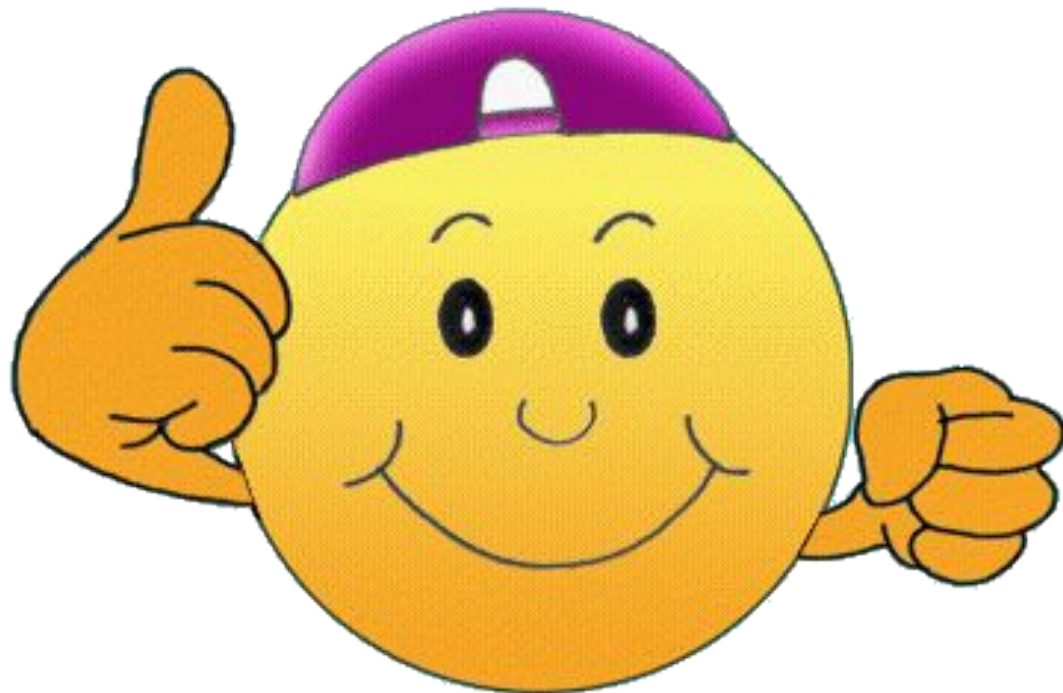
CONVOLUTIONAL NN: 

RECURRENT NN: 

© TessFerrandez

Visit - [AYONROY.ML](http://AYONROY.ML)

**GO FOR IT !**



**GOOD LUCK !**

Let me answer your Questions now.

Finally, it's your time to speak.



# Danke Scheon

Questions ? Any Feedbacks ? Did you like the talk?  
Tell me about it.

If you think I can help you,  
connect with me via

**Email** : [ayonroy2000@gmail.com](mailto:ayonroy2000@gmail.com)

**LinkedIn / Github / Telegram Username** : [ayonroy2000](#)

**Website** : <https://AYONROY.ML/>