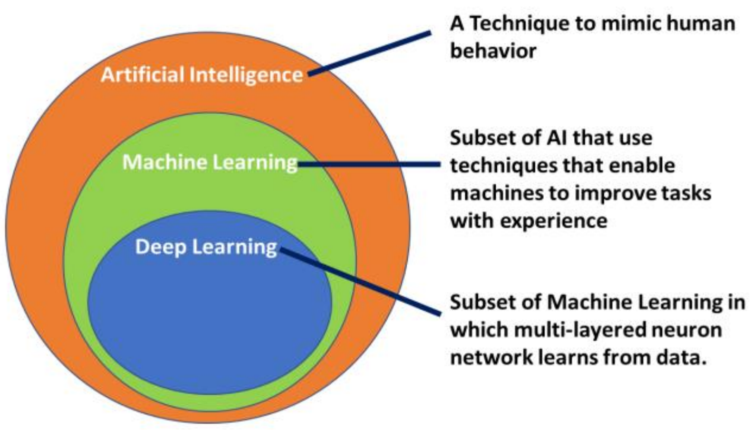
Introduction: Why deep learning, Various paradigms of learning problems, Perspectives, and Issues in the deep learning framework.
Feed-forward neural network: Biological Neurons and Biological Neural Networks, Perceptron learning, activation functions, Artificial Neural Networks, Learning XOR problem, and multi-layer perceptron.

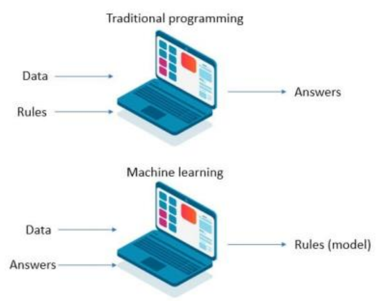
AI Vs ML Vs DL


Artificial Intelligence
A Technique to mimic human behavior
Machine Learning
Subset of AI that use techniques that enable machines to improve tasks with experience
Deep Learning
Subset of Machine Learning in which multi-layered neuron network learns from data.

Artificial Intelligence:

Artificial Intelligence is basically the mechanism to incorporate human intelligence into machines through a set of rules (algorithm). AI is a combination of two words: "Artificial" meaning something made by humans or non-natural things and "Intelligence" meaning the ability to understand or think accordingly. Another definition could be that **"AI is basically the study of training your machine (computers) to mimic a human brain and it's thinking capabilities".**

Machine Learning:

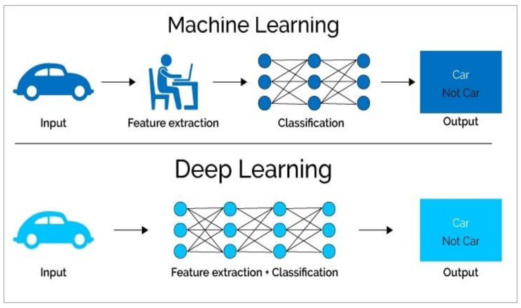


Machine Learning is basically the study/process which provides the system(computer) to learn automatically on its own through experiences it had and improve accordingly without being explicitly programmed. **ML is an application or subset of AI.** ML focuses on the development of programs so that it can access data to use it for themselves. The entire process makes observations on data to identify the possible patterns being formed and make better future decisions as per the examples provided to them. **The major aim of ML is to allow the systems to learn by themselves through the experience without any kind of human intervention or assistance.**

Deep Learning:

Deep Learning is basically a sub-part of the broader family of Machine Learning which makes use of **Neural Networks**(similar to the neurons working in our brain) to mimic human brain-like behavior. DL algorithms focus on **information processing patterns** mechanism to possibly identify the patterns just like our human brain does and classifies the information accordingly. DL works on larger sets of data when compared to ML and **prediction mechanism is self-administered by machines**.

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to "learn" from large amounts of data.



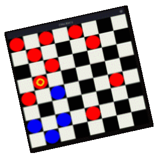
What is Well Posed Learning Problem

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E. To have a well-defined learning problem, three features needs to be identified:

1. The class of tasks (T)
2. The measure of performance to be improved (P)
3. The source of experience (E)

Examples:

- A checkers learning problem:
  - Task T: playing checkers.
  - Performance measure P: percent of games won against opponents.
  - Training experience E: playing practice games against itself.

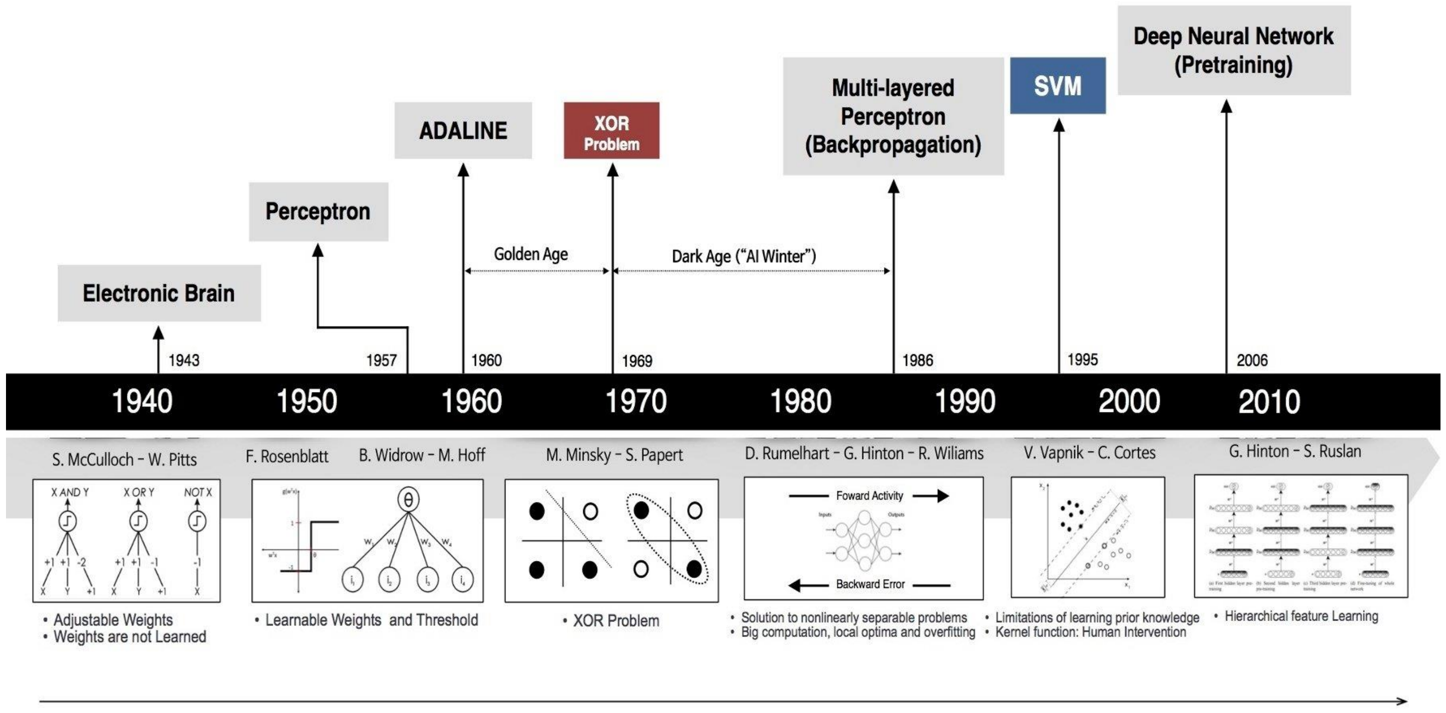


- A Self driving learning problem:
  - Task T: driving on public routes using vision sensors
  - Performance measure P: average distance travelled before an error (as judged by human overseer)
  - Training experience E: a sequence of images and steering commands recorded while observing a human driver

INTRODUCTION: (DEEP LEARNING)

History of Deep Learning Over the Years

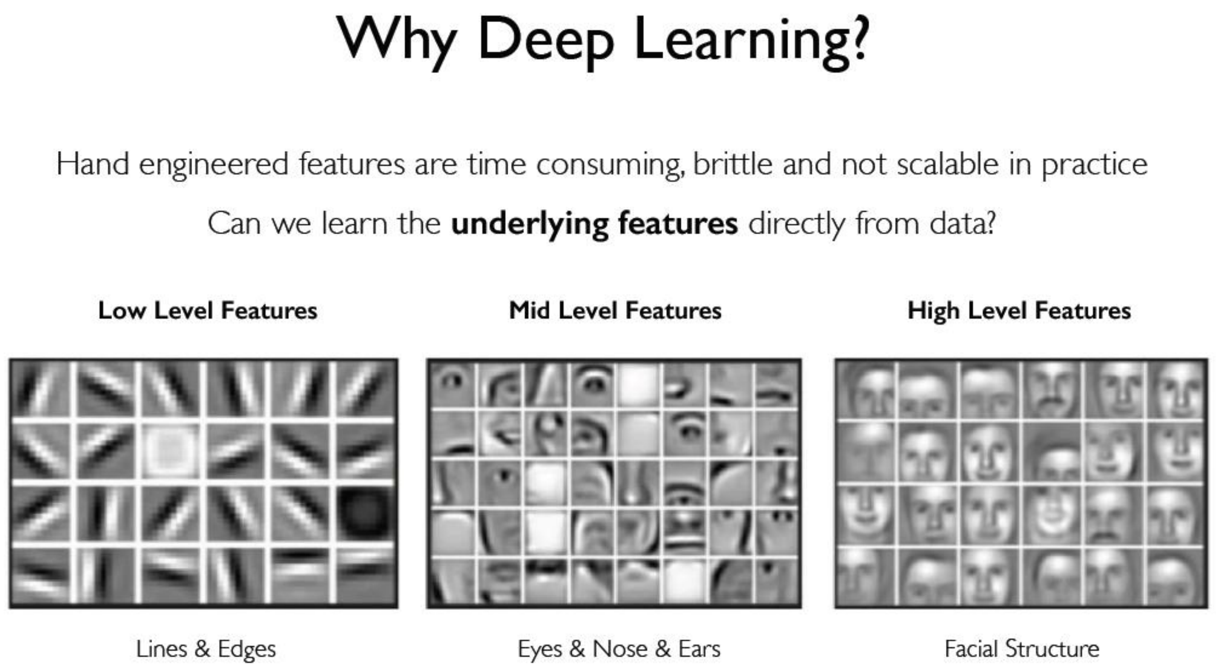
The history of deep learning dates back to 1943 when Warren McCulloch and Walter Pitts created a computer model based on the neural networks of the human brain. They used a combination of mathematics and algorithms they called threshold logic to mimic the thought process. Since then, deep learning has evolved steadily, over the years with two significant breaks in its development. The development of the basics of a continuous Back Propagation Model is credited to Henry J. Kelley in 1960. Stuart Dreyfus came up with a simpler version based only on the chain rule in 1962. The concept of back propagation existed in the early 1960s but only became useful until 1985.


Electronic Brain
Perceptron
ADALINE
XOR Problem
Multi-layered Perceptron (Backpropagation)
SVM
Deep Neural Network (Pretraining)
Golden Age
Dark Age ("AI Winter")
1943
1957
1960
1969
1986
1995
2006
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
F. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
D. Rumelhart - G. Hinton - R. Wiliams
V. Vapnik - C. Cortes
G. Hinton - S. Ruslan
X AND Y
X OR Y
NOT X
Foward Activity
Backward Error
• Adjustable Weights
• Weights are not Learned
• Learnable Weights and Threshold
• XOR Problem
• Solution to nonlinearly separable problems
• Big computation, local optima and overfitting
• Limitations of learning prior knowledge
• Kernel function: Human Intervention
• Hierarchical feature Learning

Brief history of neural network

Importance of Deep Learning

- Machine learning works only with sets of structured and semi-structured data, while deep learning works with **both structured and unstructured data.**
- Deep learning algorithms can perform **complex operations efficiently**, while machine learning algorithms cannot
- Machine learning algorithms use labeled sample data to extract patterns, while deep learning accepts **large volumes of data as input** and analyzes the input data to extract features out of an object
- The performance of machine learning algorithms decreases as the number of data increases; so to maintain the **performance of the model**, we need a deep learning



Why Deep Learning?

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

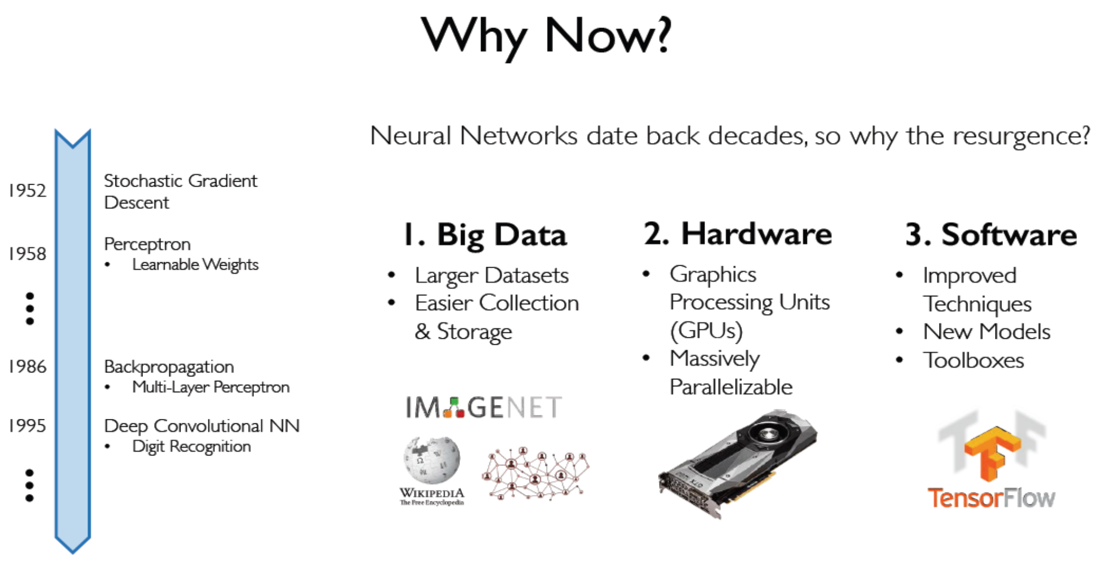
Can we learn the **underlying features** directly from data?

- The advantage of deep learning over machine learning is the redundancy of the so-called **feature extraction**.
- In traditional machine learning methods including decision trees, SVM, naïve Bayes classifier and logistic regression. These algorithms are also called flat algorithms. “Flat” here refers to the fact these algorithms cannot normally be applied directly to the raw data (such as .csv, images, text, etc.). We need a preprocessing step called feature extraction.
- The result of feature extraction is a representation of the given raw data that these classic machine learning algorithms can use to perform a task. For example, we can now classify the data into several categories or classes. Feature extraction is usually quite complex and requires detailed knowledge of the problem domain. This preprocessing layer must be adapted, tested and refined over several iterations for optimal result.

Let’s look at a concrete example. If you want to use a machine learning model to perform face recognition, we humans first need to identify the unique features of a image (eye, ear, nose, etc.), then extract the feature and give it to the algorithm as input data. In this way, the algorithm would perform face recognition. That is, in machine learning, a programmer must intervene directly in the action for the model to come to a conclusion.

In the case of a deep learning model, the feature extraction step is completely unnecessary. The model would initially recognize these unique characteristics like different arcs, lines along with the direction and orientation of lines. With the help of these low level features, the model can identify the edges of the nose, ear etc., then finally with the help of previous features the model can make correct predictions without human intervention.

In fact, refraining from extracting the characteristics of data applies to every other task you’ll ever do with neural networks. Simply give the raw data to the neural network and the model will do the rest.



Why Now?

Neural Networks date back decades, so why the resurgence?

1952 Stochastic Gradient Descent

1958 Perceptron
- Learnable Weights

1986 Backpropagation
- Multi-Layer Perceptron

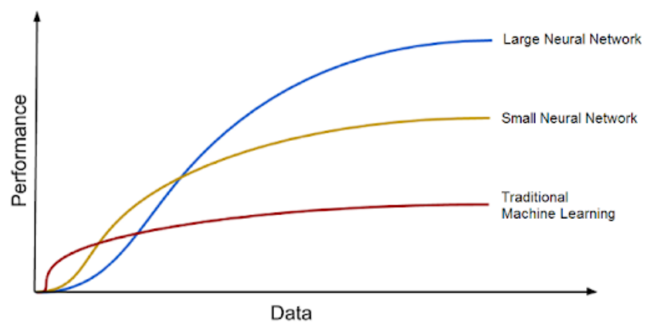
1995 Deep Convolutional NN
- Digit Recognition

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

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

3. Software
- Improved Techniques
- New Models
- Toolboxes

- As Google Fellow Jeff Dean explains, the rapid evolution in big data technologies over the past decade has positioned us well to now imbue machines with near human-level understanding.
- As per Andrew Ng, the chief scientist of China's major search engine Baidu and one of the leaders of the Google Brain Project, **"The analogy to deep learning is that the rocket engine is the deep learning models and the fuel is the huge amounts of data we can feed to these algorithms."**
- We now have a pretty good handle on how to store and then perform computation on large data sets. GPU has become a integral part now to execute any Deep Learning algorithm.
- Deep learning models tend to increase their accuracy with the increasing amount of training data, whereas traditional machine learning models such as SVM and naive Bayes classifier stop improving after a saturation point.



When to use Deep Learning or not over others?

1. Deep Learning out perform other techniques if the data size is large. But with small data size, traditional Machine Learning algorithms are preferable.
2. Deep Learning techniques need to have high end infrastructure to train in reasonable time.
3. When there is lack of domain understanding for feature introspection, Deep Learning techniques outshines others as you have to worry less about feature engineering.
4. Deep Learning really shines when it comes to complex problems such as image classification, natural language processing, and speech recognition.

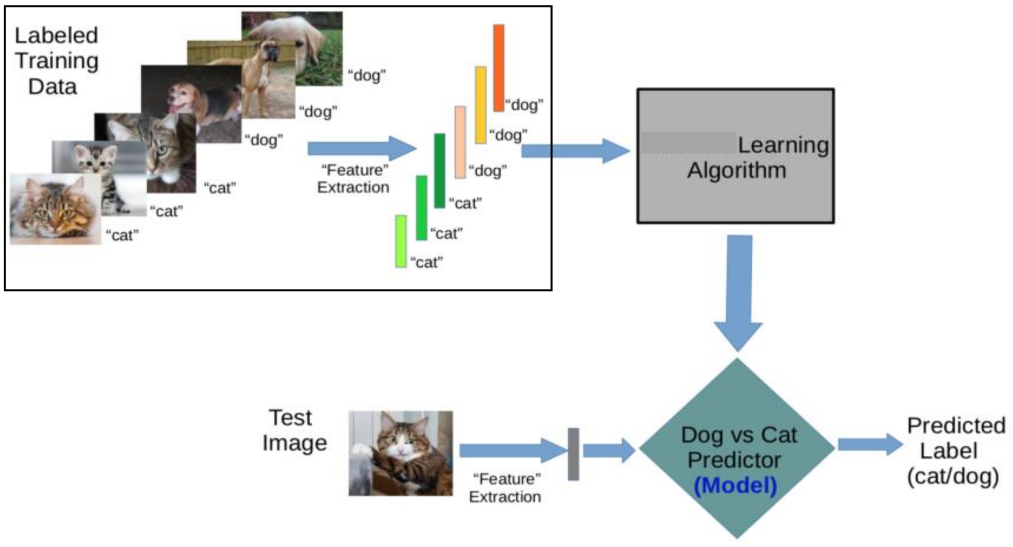
Various paradigms of learning process:

- Deep Supervised learning
- Deep Unsupervised learning
- Deep Semi-supervised learning
- Deep Reinforcement learning

Deep Supervised learning:

- Deep supervised learning is a type of machine learning technique where a neural network is trained to learn patterns and relationships between input features and output labels using a labeled dataset.
- The main advantage of this technique is the ability to collect data or generate a data output from the prior knowledge. However, the disadvantage of this technique is that decision boundary might be overstrained when training set doesn't own samples that should be in a class.

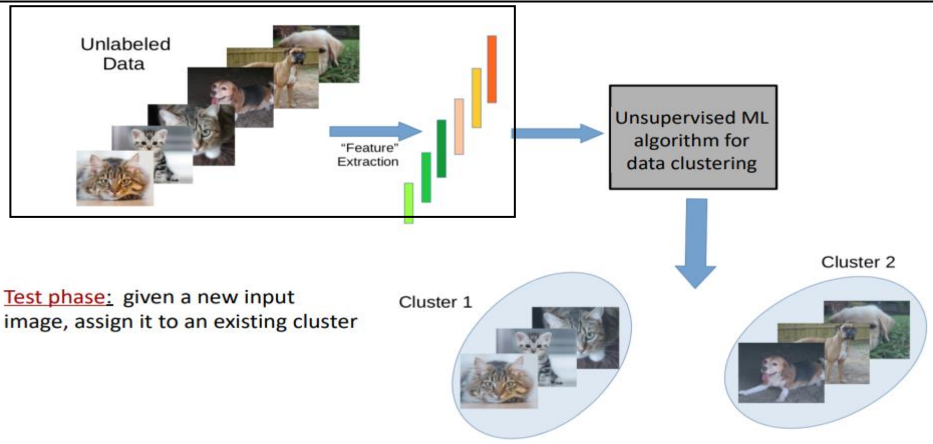
- For Deep Learning, there are several supervised learning techniques, such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and deep neural networks (DNNs). In addition, the RNN category includes gated recurrent units (GRUs) and long short-term memory (LSTM) approaches.



- Deep supervised learning has been successfully applied to a wide range of applications, including computer vision, speech recognition, natural language processing, and many others. It has led to significant advances in these fields, and has become a powerful tool for developing intelligent systems that can automate complex tasks.
- Overall, this technique is simpler than other techniques in the way of learning with high performance.
- ***Advantages***
  - High accuracy
  - End-to-end learning
  - Generalization
  - Scalability
- ***Disadvantages***
  - Dependence on labeled data
  - Overfitting
  - Limitations in certain domains
  - Computationally expensive
  - Vulnerability to adversarial attacks

Deep-unsupervised learning:

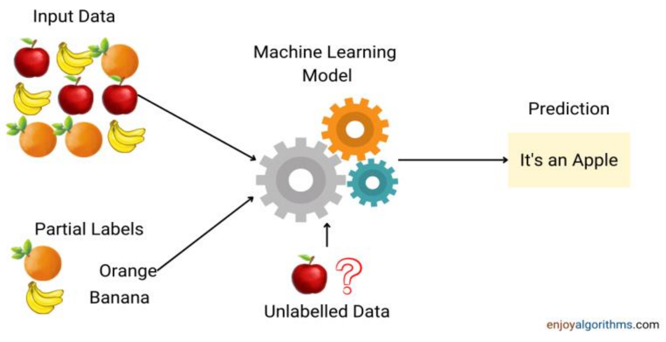
- Deep unsupervised learning is a type of machine learning technique where a neural network is trained to learn patterns and relationships in unlabeled data without the need for explicit supervision or labels. This means that the neural network is trained on a set of input data with no corresponding output labels, and the goal is to learn a representation of the data that captures the underlying structure and patterns.
- The most common types of deep unsupervised learning are generative models, such as autoencoders, generative adversarial networks (GANs), and variational autoencoders (VAEs). These models learn to generate new data that is similar to the input data, while also capturing the underlying structure and patterns.
- Moreover, RNNs, which include GRUs and LSTM approaches, have also been employed for unsupervised learning in a wide range of applications.
- Deep unsupervised learning is used in a wide range of applications, including market segmentation, anomaly detection, image compression, Audio processing, language translation, sentiment analysis, and text classification and many others.
- Overall, unsupervised learning is a powerful and widely-used approach to machine learning that provides a way to explore and understand the structure and patterns in the data without being told what the outputs should be.
- The main disadvantages of unsupervised learning are unable to provide accurate information concerning data sorting and computationally complex.


Unlabeled Data
"Feature" Extraction
Unsupervised ML algorithm for data clustering
Test phase: given a new input image, assign it to an existing cluster
Cluster 1
Cluster 2

- ***Advantages***
  - Ability to learn from unstructured and unlabeled data
  - Scalability
  - Generalization
  - Feature learning
- ***Disadvantages***
  - Lack of interpretability
  - Computational complexity
  - Difficulty in selecting appropriate model architectures
  - Difficulty in fine-tuning for downstream tasks
  - Lack of explicit supervision

Deep Semi- Supervised learning:

- In semi-supervised learning, the model is trained on a combination of labeled and unlabeled data. The goal is to leverage the information present in the labeled data to improve the model's ability to generalize and make predictions, while also leveraging the information present in the unlabeled data to learn more about the underlying structure of the data.
- This approach can be particularly useful when the amount of labeled data available is limited, but there is a large amount of unlabeled data available.
- Semi-supervised learning algorithms can be classified into two main categories: ***self-training and co-training***.
- In self-training, the model is first trained on the labeled data, and then used to make predictions on the unlabeled data. The samples from the unlabeled data that are confidently predicted are then added to the labeled data, and the process is repeated until a stopping criterion is met.
- In co-training, two separate models are trained on different views of the same data, and they are used to annotate the unlabeled data, which is then used to update both models.
- Deep Semi-supervised learning can be useful in a variety of applications, including image classification, speech recognition, and natural language processing, Robotics, Anomaly detection among others.

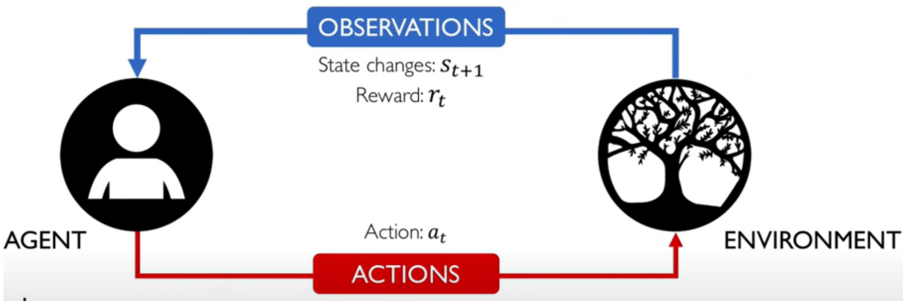


- In this approach, a small portion of the data is labeled, and the remaining data is unlabeled. The labeled data is used to guide the learning process, while the unlabeled data is used to learn a more general representation of the data.

- ***Advantages***
  - Efficient use of labeled data
  - Better generalization
  - Cost-effective
  - Handling imbalanced datasets
  - Better performance on complex and high-dimensional data
- ***Disadvantages***
  - Requires careful tuning
  - Limited applicability
  - Unlabeled data may not always be helpful
  - Potential overfitting

Deep Reinforcement learning:

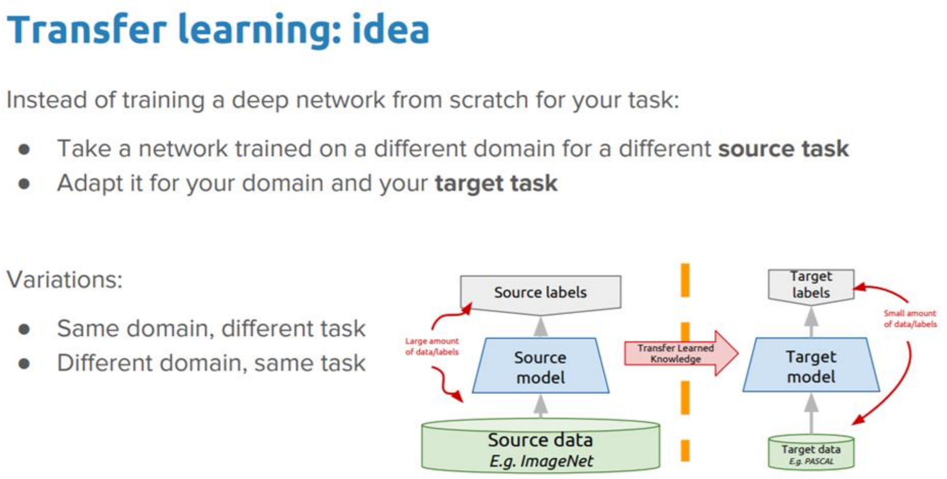
- Deep reinforcement learning is a machine learning technique that combines deep neural networks with reinforcement learning, a type of learning that involves an agent learning to take actions in an environment to maximize a reward signal.



- In deep reinforcement learning, a deep neural network is used to represent the policy or value function of the agent. The neural network takes as input the state of the environment, and outputs the action to take. The neural network is trained using reinforcement learning algorithms, such as Q-learning or policy gradient methods, to optimize the reward obtained by the agent.
- Reinforcement learning algorithms can be divided into two main categories: **model-based and model-free.** Model-based algorithms use a model of the environment to plan and make decisions, while model-free algorithms do not require a model and instead learn from experience.
- Reinforcement learning has been applied to a wide range of problems, including robotics, autonomous systems, game AI, and decision-making systems. Some well-known examples of successful applications of reinforcement learning include AlphaGo, the computer program that defeated the world champion at the game of Go, and OpenAI's Dactyl, a robot that can solve the Rubik's cube using reinforcement learning.
- Overall, reinforcement learning is a powerful tool for decision making and control, and it has the potential to revolutionize the way that machines learn and interact with their environment.
- ***Advantages***
  - Flexibility
  - Real-time adaptation
  - Autonomous learning
  - Handling uncertainty
  - Generalization
  - Ability to learn from sparse rewards
- ***Disadvantages***
  - Sample inefficiency
  - Requires careful tuning
  - Risk of instability
  - Black box nature
  - Difficulty in defining reward functions

Transfer learning:

- Transfer learning is a technique in deep learning that allows a pre-trained neural network to be fine-tuned for a new task, typically with much fewer training examples than would be required if training from scratch. This is possible because pre-trained networks have already learned a rich set of features from the large amounts of data they were trained on, which can be useful for a wide range of tasks.
- In transfer learning, the pre-trained network is used as a feature extractor, and the final layers, which are task-specific, are trained to perform the new task
- For example, a pre-trained image classification network can be fine-tuned to perform object detection, semantic segmentation, or even to generate captions for images.
- The main benefit of transfer learning is that it allows you to leverage the knowledge learned from a large dataset to perform well on a new task with limited data. This can save a significant amount of time and computational resources compared to training a network from scratch.
- It's important to note that not all pre-trained networks are suitable for all tasks, and it's crucial to choose the right pre-trained network and fine-tune it appropriately for the specific task at hand.
- *Advantages*
  - Reduced training time
  - Better performance with limited data
  - Reusable models
  - Improved generalization
  - Easier to implement
- *Disadvantages*
  - Task-specific limitations
  - Overreliance on pre-training
  - Lack of interpretability
  - High computational cost
  - Dataset bias



Transfer learning: idea

Instead of training a deep network from scratch for your task:

- Take a network trained on a different domain for a different **source task**
- Adapt it for your domain and your **target task**

Variations:

- Same domain, different task
- Different domain, same task

Deep Learning framework

- *Deep learning (DL) frameworks offer building blocks for designing, training, and validating deep neural networks through a high-level programming interface.*
- A deep learning framework is a software library that provides a set of tools and algorithms for building and training artificial neural networks.
- Deep learning frameworks make it easier to develop deep learning models by providing pre-built components and tools for tasks such as data preprocessing, model definition, and model training.
- This allows developers to focus on the creative aspects of deep learning, such as architecture design and feature engineering, rather than the low-level details of implementing neural network algorithms.
- Some popular deep learning frameworks include **TensorFlow, PyTorch, Keras, Caffe, and Theano.**

Perspectives with deep learning framework

There are several perspectives to consider when evaluating deep learning frameworks:

1. **Ease of use:** Some deep learning frameworks are designed to be more user-friendly and easier to use, which can make them a good choice for beginners or those with limited experience in machine learning.
2. **Performance:** Performance is an important consideration, especially for large-scale or time-sensitive tasks. Some deep learning frameworks are optimized for speed and efficiency, and can be used to train models more quickly or with fewer resources.
3. **Flexibility:** Some deep learning frameworks are more flexible and allow for more customization, while others are more limited in terms of what can be achieved.
4. **Community support:** A strong and active community can be valuable for getting help with problems and for staying up-to-date with the latest developments in deep learning.
5. **Integration with other tools:** If you are using other tools or technologies in your project, it may be important to choose a deep learning framework that integrates well with those tools.
6. **Cost:** Some deep learning frameworks are open source and free to use, while others may have licensing fees or other costs associated with their use.

Issues with deep learning framework

Despite their popularity and success in various applications, deep learning frameworks also have some limitations and issues that need to be considered:

1. **Complexity:** Deep learning models can be complex and difficult to understand, especially for those with limited experience in machine learning. The high number of parameters and non-linear relationships can make it challenging to interpret the results and fine-tune the models.
2. **Resource requirements:** Training deep learning models can be computationally intensive, requiring large amounts of memory and computational power. This can make it challenging to train models on personal computers or embedded devices with limited resources.
3. **Data bias:** Deep learning models are only as good as the data they are trained on. If the training data contains bias or inaccuracies, this can result in biased or inaccurate predictions.
4. **Overfitting:** Overfitting occurs when a deep learning model learns the training data too well and is unable to generalize to new, unseen data. This can result in poor performance on the test data.
5. **Lack of transparency:** The internal workings of deep learning models can be difficult to interpret, which can make it challenging to understand the reasoning behind certain predictions. This lack of transparency can also make it difficult to identify and correct errors in the model.
6. **Model selection:** Choosing the right deep learning model for a particular task can be challenging, as there are many different types of models to choose from, each with its own strengths and weaknesses.

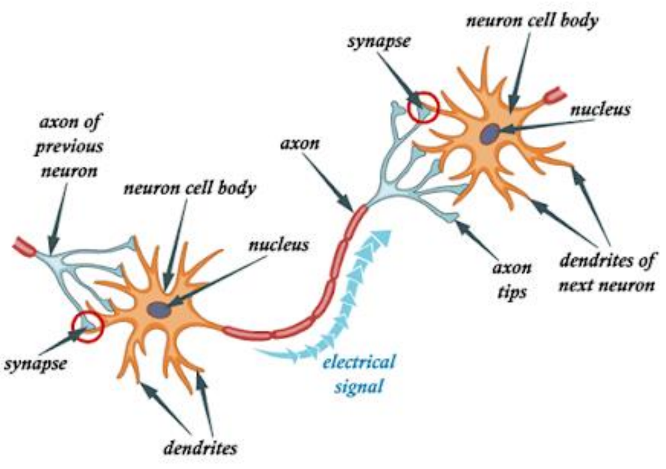
These are some of the main issues with deep learning frameworks, and ongoing research is aimed at addressing these limitations and making deep learning more accessible and interpretable.

How Do Deep Learning Neural Networks Work?

BIOLOGICAL NEURON AND BIOLOGICAL NEURAL NETWORKS

- Motivation behind neural network is human brain.
- Human brain is called as the best processor even though it works slower than other computers. Many researchers thought to make a machine that would work in the prospective of the human brain. Human brain contains billion of neurons which are connected to many other neurons to form a network so that if it sees any image, it recognizes the image and processes the output.

In short, a biological neural network consists of numerous neurons.



- A typical biological neuron consists of a cell body, dendrites and an axon.
- Dendrites are thin structures that emerge from the cell body.
- An axon is a cellular extension that emerges from this cell body.
- Most neurons receive signals through the dendrites and send out signals along the axon.
- At the majority of synapses, signals cross from the axon of one neuron to the dendrite of another.
- All neurons are electrically excitable due to the maintenance of voltage gradients in their membranes. If the voltage changes by a large enough amount over a short interval, the neuron generates an electrochemical pulse called an action potential. This potential travels rapidly along the axon and activates synaptic connections.
- In the similar manner, it was thought to make artificial interconnected neurons like biological neurons making up an Artificial Neural Network (ANN).
- Each biological neuron is capable of taking a number of inputs and produce output. Neurons in human brain are capable of making very complex decisions, so this means they run many parallel processes for a particular task. One motivation for ANN is that to work for a particular task identification through many parallel processes.

ARTIFICIAL NEURAL NETWORKS

Unit or Neuron

An artificial neuron is a mathematical function based on a model of biological neurons, where each neuron takes inputs, weighs them separately, sums them up and passes this sum through a nonlinear function to produce output.

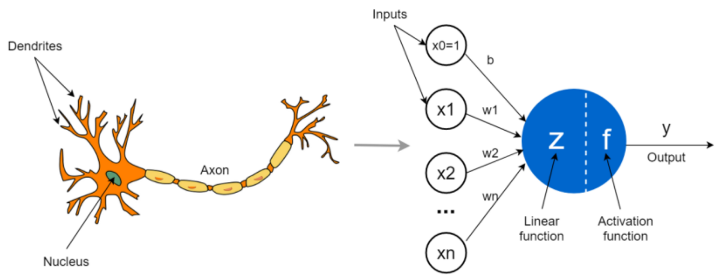
- The building block of a neural network is a single computational unit.
- A unit takes a set of real valued numbers as input, performs some computation on them, and produces an output.
- A neural unit takes a weighted sum of its inputs, with one additional term in the sum called a bias term.
- Given a set of inputs x1...xn, a unit has a set of corresponding weights w1...wn and a bias b, so the weighted sum z can be represented as:

$$z = b + \sum_i w_i x_i$$

- It is more convenient to express this weighted sum using vector notation.

z = w.x + b here z is a real valued number.

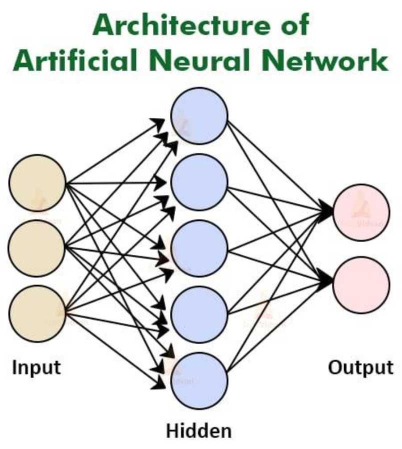
- The artificial neuron has the following characteristics:
  - A neuron is a mathematical function modeled on the working of biological neurons
  - It is an elementary unit in an artificial neural network
  - One or more inputs are separately weighted
  - Inputs are summed and passed through a nonlinear function to produce output
  - Every neuron holds an internal state called activation signal
  - Each connection link carries information about the input signal
  - Every neuron is connected to another neuron via connection link



Biological Neuron	Artificial Neuron
Dendrites	Input
Cell Nucleus(Soma)	Node
Axon	Output
Synapse	Interconnections

Artificial Neural Network:

- A modern neural network is a network of small computing units, each of which takes a vector of input values and produces a single output value.
- Artificial Neural Networks (ANNs) are a type of machine learning model inspired by the structure and function of the human brain.
- The basic building block of an ANN is an artificial neuron, which receives one or more inputs and produces an output based on a set of weights and a bias term.
- An ANN typically consists of an input layer, one or more hidden layers, and an output layer. The input layer receives the input data, which is then passed through the hidden layers to the output layer. Each hidden layer consists of multiple neurons, and the number of neurons and layers can vary depending on the complexity of the task
- The weights and bias terms are learned through a process called training, where the model adjusts its parameters to minimize the difference between its predicted outputs and the true outputs of a set of training data.
- Neural networks share much of the same mathematics as logistic



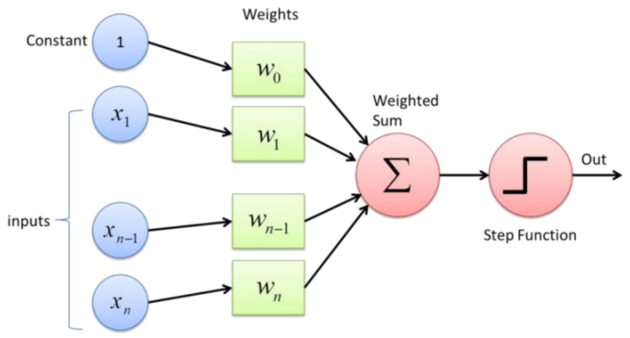
regression. But neural networks are a more powerful classifier than logistic regression.

- **Architecture:**
  - It is composed of interconnected artificial neurons arranged in layers, and it is capable of learning from examples and making predictions on new data.
  - A neural network consists of three layers.
    - Input layer
    - Hidden layer
    - Output layer
  - The weight between neurons determines the learning ability of the neural network.
  - The number of hidden layers, the number of neurons in each layer, and the activation function used can all be adjusted to optimize the performance of the network for a given problem.

ANN	BNN
They are slow in processing information.	Processing speed is fast as compared to Biological Neural Network.
Allocation for storage to a new process is easy	Allocation for Storage to a new process is strictly irreplaceable
The process can operate in massive parallel operations.	Processes operate in sequential mode.
Information is distributed into the network throughout into sub-nodes, even if it gets corrupted it can be retrieved.	If any information gets corrupted in the memory it cannot be retrieved.
There is no control unit to monitor the information being processed into the network.	The activities are continuously monitored by a control unit.

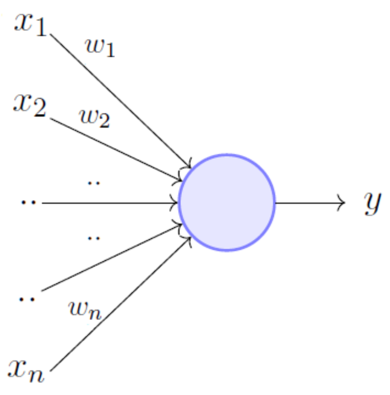
Perceptron Learning

- Perceptron is also understood as an Artificial Neuron or neural network unit that helps to detect certain input data computations in business intelligence.
- Perceptron was introduced by Frank Rosenblatt in 1957.
- The perceptron model is a more general computational model.
- Perceptron model is also treated as one of the best and simplest types of Artificial Neural networks.
- However, it is a supervised learning algorithm of binary classifiers. Hence, we can consider it as a single-layer neural network with four main parameters, i.e., **input values, weights and Bias, net sum, and an activation function.**
- It takes an input, aggregates it (weighted sum) and returns 1 only if the aggregated sum is more than some threshold else returns 0.
- Perceptron Learning Rule states that the algorithm would automatically learn the optimal weight coefficients. The input features are then multiplied with these weights to determine if a neuron fires or not.



$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

- A step function is **is used by perceptron.** The output of step function is a certain value, A_1, if the input sum is above a certain threshold and A_0 if the input sum is below a certain threshold.

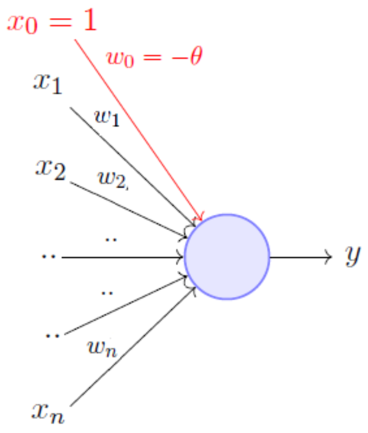


$$y = 1 \quad if \sum_{i=1}^{n} w_i * x_i \geq \theta$$
$$= 0 \quad if \sum_{i=1}^{n} w_i * x_i < \theta$$

Rewriting the above,

$$y = 1 \quad if \sum_{i=1}^{n} w_i * x_i - \theta \geq 0$$
$$= 0 \quad if \sum_{i=1}^{n} w_i * x_i - \theta < 0$$

- Rewriting the threshold as shown above and making it a constant input with a variable weight, we would end up with something like the following:



A more accepted convention,

$$y = 1 \quad if \sum_{i=0}^{n} w_i * x_i \geq 0$$
$$= 0 \quad if \sum_{i=0}^{n} w_i * x_i < 0$$
$$where, \quad x_0 = 1 \quad and \quad w_0 = -\theta$$

- A single perceptron can only be used to implement **linearly separable** functions. It takes both real and boolean inputs and associates a set of **weights** to them, along with a **bias** (the threshold mentioned above). We learn the weights, we get the function.
- The Perceptron algorithm learns the weights for the input signals in order to draw a linear decision boundary.
- Let's use a perceptron to learn an OR function.

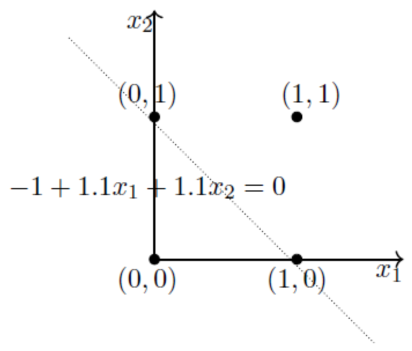
OR function using a perceptron:

x_1	x_2	OR	
0	0	0	$w_0 + \sum_{i=1}^{2} w_i x_i < 0$
1	0	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
0	1	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
1	1	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$

$$w_0 + w_1 \cdot 0 + w_2 \cdot 0 < 0 \implies w_0 < 0$$
$$w_0 + w_1 \cdot 0 + w_2 \cdot 1 \geq 0 \implies w_2 > -w_0$$
$$w_0 + w_1 \cdot 1 + w_2 \cdot 0 \geq 0 \implies w_1 > -w_0$$
$$w_0 + w_1 \cdot 1 + w_2 \cdot 1 \geq 0 \implies w_1 + w_2 > -w_0$$

One possible solution is

$$w_0 = -1, \; w_1 = 1.1, \; w_2 = 1.1$$



Limitation of a Perceptron model:

1. The output of a perceptron can only be a binary number (0 or 1) due to the hard-edge transfer function.
2. It can only be used to classify the linearly separable sets of input vectors. If the input vectors are non-linear, it is not easy to classify them correctly.

Activation Functions:

- **Activation function** : z is a linear function of x. Instead of using z as the output, neural units apply a non-linear function f to z. The output of this function is the activation value (a) for the unit.
- The activation value is the final output of the unit, and we generally refer it as y.

y = a = f(z)

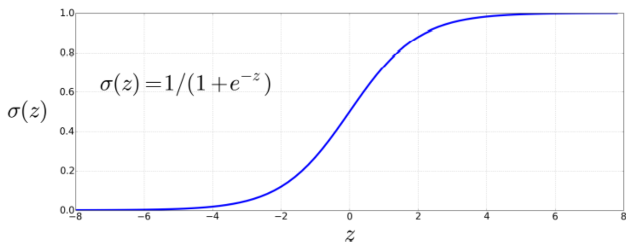
- The activation function is a non-linear transformation that we do over the input before sending it to the next layer of neurons or finalizing it as output.
- Activation functions are used to determine the output of a neuron and are critical to the performance of a neural network.
- An Activation Function decides whether a neuron should be activated or not.
- The Activation Functions can be basically divided into 2 types-
  - Linear Activation Function
  - Non-linear Activation Functions
- There are three popular non-linear functions.
  - the sigmoid,
  - the tanh,
  - the ReLU.

Sigmoid :

- It is always convenient to start with the sigmoid function.

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

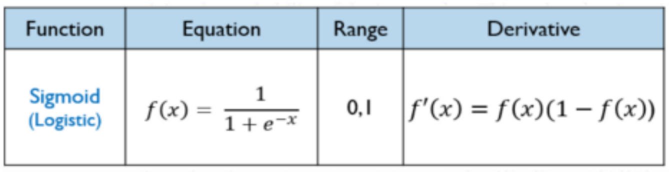
- The sigmoid shown in the below figure has a number of advantages;
  - it maps the output into the range [0,1], which is useful in squashing outliers toward 0 or 1.
  - It's differentiable.



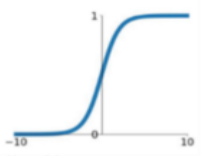
- the output of the neural unit after sigmoid substitution is

$$y = \sigma(\mathbf{w}\cdot\mathbf{x}+b) = \frac{1}{1+\exp(-(\mathbf{w}\cdot\mathbf{x}+b))}$$

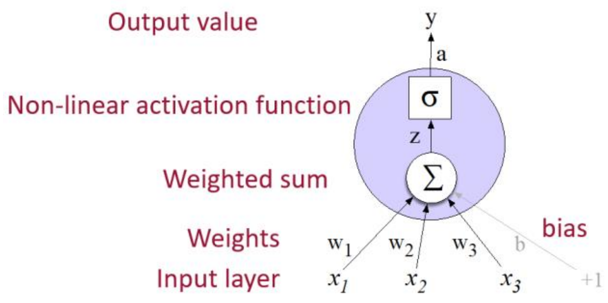
- It is commonly used for models where we have to predict the probability as an output. Since probability of anything exists only between the range of 0 and 1, sigmoid is the right choice because of its range.
- The function is differentiable and provides a smooth gradient, i.e., preventing jumps in output values. This is represented by an S-shape of the sigmoid activation function.



Function	Equation	Range	Derivative
Sigmoid (Logistic)	$f(x) = \frac{1}{1+e^{-x}}$	0,1	$f'(x) = f(x)(1-f(x))$



- The final schematic of a basic neural unit is shown in the below figure.



- In this example the unit takes 3 input values x_1, x_2, and x_3, and computes a weighted sum, multiplying each value by a weight (w_1, w_2, and w_3, respectively), adds them to a bias term b, and then passes the resulting sum through a sigmoid function to result in a number between 0 and 1.

Example:

- Let's suppose we have a unit with the following weight vector and bias:

$$w = [0.2,0.3,0.9]$$
$$b = 0.5$$

What would this unit do with the following input vector?

$$x = [0.5,0.6,0.1]$$

The resulting output y would be:

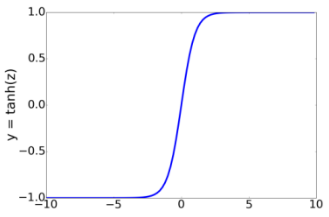
$$y = \sigma(\mathbf{w}\cdot\mathbf{x}+b) = \frac{1}{1+e^{-(\mathbf{w}\cdot\mathbf{x}+b)}}$$

$$= \frac{1}{1+e^{-(.5*.2+.6*.3+.1*.9+.5)}} = \frac{1}{1+e^{-0.87}} = .70$$

**Note : Calculate output y using *tanh* and *ReLU* also.

tanh:

- tanh is similar to sigmoid and almost always better than sigmoid.
- Its value ranges from -1 to +1.
- tanh is shown in below figure and it is calculated as.

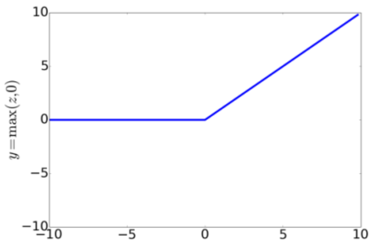


$$y = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

- The output of the tanh activation function is Zero centered; hence we can easily map the output values as strongly negative, neutral, or strongly positive.
- Usually used in hidden layers of a neural network as its values lie between -1 to 1; therefore, the mean for the hidden layer comes out to be 0 or very close to it. It helps in centering the data and makes learning for the next layer much easier.
- Derivative of Tanh function suffers "Vanishing gradient and Exploding gradient problem".

ReLU:

- ReLU stands for rectified linear unit.
- It is the simplest activation function, and perhaps the most commonly used.
- It's just the same as z when z is positive, and 0 otherwise.
- ReLU is shown in below figure and it is calculated using.

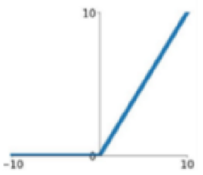


$$y = max(z, 0)$$

- Since only a certain number of neurons are activated, the ReLU function is far more computationally efficient when compared to the sigmoid and tanh functions.
- ReLU accelerates the convergence of gradient descent towards the global minimum of the loss function due to

its linear, non-saturating property.

Function	Equation	Range	Derivative
ReLu (Rectified Linear Unit)	$f(x) = \begin{cases} 0 \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases}$	$0, +\infty$	$f'(x) = \begin{cases} 0 \; for \; x < 0 \\ 1 \; for \; x \geq 0 \end{cases}$



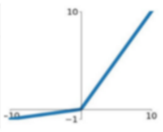
Leaky ReLU:

- ***Leaky ReLU function is nothing but an improved version of the ReLU function with introduction of "constant slope"***
- The standard ReLU function returns zero for any input value less than zero, which can cause "dying ReLU" problem where a neuron becomes permanently inactive and stops learning during training. Leaky ReLU helps to solve this issue by introducing a small slope (typically 0.01) for negative values, which allows for a small gradient and ensures that the neuron can continue to learn.
- The mathematical formula for Leaky ReLU is:

  f(x) = max(ax, x)

  where x is the input to the function, and a is the slope for negative values. When a=0, Leaky ReLU becomes the standard ReLU function.

Function	Equation	Range	Derivative
Leaky ReLu	$f(x) = \begin{cases} \alpha x \; for \; x < 0 \\ x \; \; for \; x \geq 0 \end{cases}$	$-\infty, +\infty$	$f'(x) = \begin{cases} \alpha \; for \; x < 0 \\ 1 \; for \; x \geq 0 \end{cases}$



α =0.01

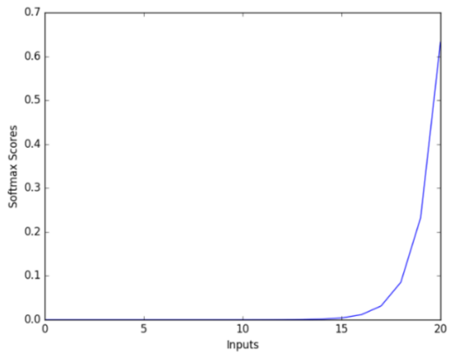
Softmax:

- Softmax is an activation function that scales numbers/logits into probabilities.
- **"Softmax function returns the probability for a datapoint belonging to each individual class."** (The softmax function takes a vector of real numbers as input and returns a probability distribution over the input values)
- It is most commonly used as an activation function for the last layer of the neural network in the case of multi-class classification.
- The softmax function transforms the input values to a range between 0 and 1, which can be interpreted as probabilities. The output values represent the likelihood of each class, and the class with the highest probability is chosen as the final prediction.
- The softmax function works by exponentiating the input values and then normalizing the resulting vector so that the values sum to 1. The mathematical formula for the softmax function is:

Formula

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

σ	= softmax
$\vec{z}$	= input vector
e^{z_i}	= standard exponential function for input vector
K	= number of classes in the multi-class classifier
e^{z_j}	= standard exponential function for output vector



Name	Equation	Derivatives	Range
Softmax	$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}}$ for $i = 1, \ldots, J$	$\frac{\partial f_i(\vec{x})}{\partial x_j} = f_i(\vec{x})(\delta_{ij} - f_j(\vec{x}))$[5]	$(0, 1)$

Why do you need non-linear activation functions?

- If we removed the activation function from our algorithm that can be called linear activation function.
- Linear activation function will output linear activations
  - Whatever hidden layers you add, the activation will be always linear like logistic regression (So its useless in a lot of complex problems)

- You might use linear activation function in one place - in the output layer if the output is real numbers (regression problem). But even in this case if the output value is non-negative you could use RELU instead.

The XOR Problem

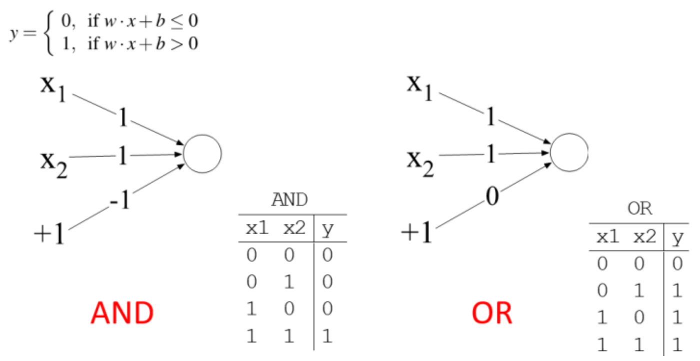
- **perceptron** : perceptron is a very simple neural unit that has a binary output and does not have a non-linear activation function. The output y of a perceptron is 0 or 1, and is computed as follows.

$$y = \begin{cases} 0, & \text{if } \mathbf{w}\cdot\mathbf{x}+b \le 0 \\ 1, & \text{if } \mathbf{w}\cdot\mathbf{x}+b > 0 \end{cases}$$

- Consider the task of computing elementary logical functions of two inputs, like AND, OR, and XOR. The truth tables for those functions are shown below.

AND			OR			XOR		
x1	x2	y	x1	x2	y	x1	x2	y
0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	1	1
1	0	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	0

- It's very easy to build a perceptron that can compute the logical AND and OR functions of its binary inputs;


$y = \begin{cases} 0, & \text{if } w\cdot x+b \le 0 \\ 1, & \text{if } w\cdot x+b > 0 \end{cases}$
x_1
1
x_2
1
-1
+1
AND
x_1
1
x_2
1
0
+1
OR

AND

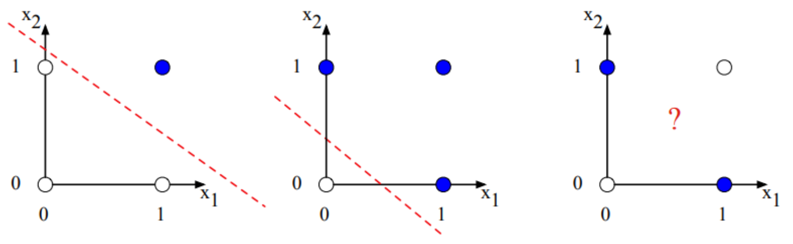
x1	x2	y
0	0	0
0	1	0
1	0	0
1	1	1

OR

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

- But it's not possible to build a perceptron to compute logical XOR!

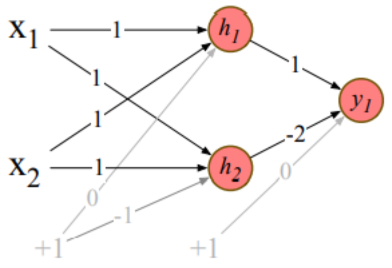
- The reason for this is "perceptron is a linear classifier". For a two-dimensional input x_1 and x_2, the perception equation, $w_1x_1 + w_2x_2 + b = 0$ is the equation of a line.
- This line acts as a decision boundary in two-dimensional space in which the output 0 is assigned to all inputs lying on one side of the line, and the output 1 to all input points lying on the other side of the line.
- Following figure shows the possible logical inputs (00, 01, 10, and 11) and the line drawn by one possible set of parameters for an AND and an OR classifier.



a) x_1 AND x_2 b) x_1 OR x_2 c) x_1 XOR x_2

- Notice that there is simply no way to draw a line that separates the positive cases of XOR (01 and 10) from the negative cases (00 and 11). We say that XOR is not a linearly separable function.

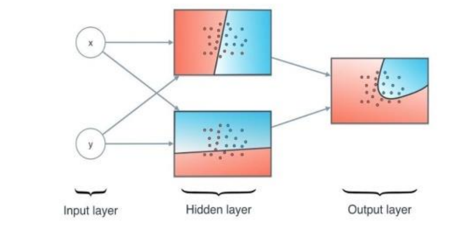
The solution : neural network



- XOR can't be calculated by a single perceptron
- XOR can be calculated by a layered network of units.

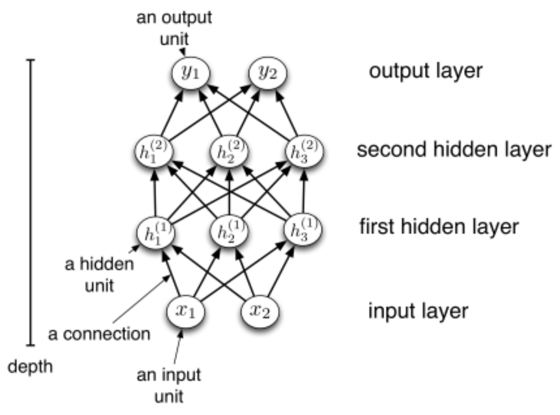
- The below network shows a figure with the input being processed by two layers of neural units. The middle layer (called *h*) has two units, and the output layer (called *y*) has one unit. A set of weights and biases are shown for each ReLU that correctly computes the XOR function.
- Consider input x = [0, 0]. If we multiply each input value by the appropriate weight, sum, and then add the bias b, we get the vector [0 -1], apply the ReLU and we get [0 0], and the final value is 0.
- The solution to the XOR problem requires a network of units with nonlinear activation functions.

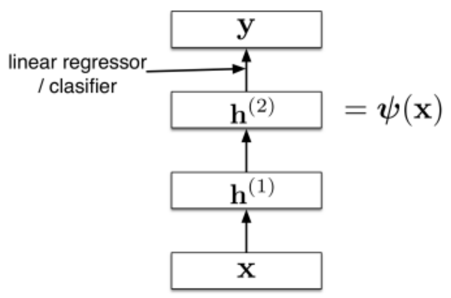
Multilayer perceptron



Combining Neurons allows Neural Networks to learn more complex, Nonlinear Decision Boundaries

- It is a neural network where the mapping between inputs and output is non-linear.
- A Multilayer perceptron has input and output layers, and one or more hidden layers with many neurons stacked together.
- Multilayer Perceptron falls under the category of feedforward algorithms, because inputs are combined with the initial weights in a weighted sum and subjected to the activation function, just like in the Perceptron. But the difference is that each linear combination is propagated to the next layer.
- Every unit in one layer is connected to every unit in the next layer; we say that the network is fully connected. The first layer is the input layer, and its units take the values of the input features. The last layer is the output layer, and it has one unit for each value the network outputs. All the layers in between these are known as hidden layers.





Let's write out the MLP computations mathematically.

The input units as x_j and the activation of the output unit as y. The units in the 'l' hidden layer will be denoted $h_i^{(l)}$. Our network is fully connected, so each unit receives connections from all the units in the previous layer. This means each unit has its own bias, and there's a weight for every pair of units in two consecutive layers. Therefore, the network's computations can be written out as

$$h_i^{(1)} = \phi^{(1)}\left(\sum_j w_{ij}^{(1)} x_j + b_i^{(1)}\right)$$

$$h_i^{(2)} = \phi^{(2)}\left(\sum_j w_{ij}^{(2)} h_j^{(1)} + b_i^{(2)}\right)$$

$$y_i = \phi^{(3)}\left(\sum_j w_{ij}^{(3)} h_j^{(2)} + b_i^{(3)}\right)$$

Since there is a weight for every pair of units in two consecutive layers, we represent each layer's weights with a weight matrix $W^{(l)}$. Each layer also has a bias vector $b^{(l)}$. The above computations are therefore written in vectorized form as:

$$\mathbf{h}^{(1)} = \phi^{(1)}\left(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\right)$$
$$\mathbf{h}^{(2)} = \phi^{(2)}\left(\mathbf{W}^{(2)}\mathbf{h}^{(1)} + \mathbf{b}^{(2)}\right)$$
$$\mathbf{y} = \phi^{(3)}\left(\mathbf{W}^{(3)}\mathbf{h}^{(2)} + \mathbf{b}^{(3)}\right)$$

But it has more to it.

If the algorithm only computed the weighted sums in each neuron, propagated results to the output layer, and stopped there, it wouldn't be able to learn the weights that minimize the cost function. If the algorithm only computed one iteration, there would be no actual learning.

Question Bank

1. What is deep learning? Why deep learning is more popular and why now?
2. Explain various paradigms of deep learning problems.
3. What is deep learning framework. Explain the Perspectives, and Issues in the deep learning framework.
4. Explain the concept of a Perceptron with a neat diagram.
5. What is Artificial Neural Network and its architecture?
6. List and explain the various activation functions used in modeling of artificial neuron. Also explain their suitability with respect to applications.
7. Explain the structure of biological neuron in detail. What is the motivation for Neural network.
8. Discuss the XOR problem and suggest a solution using Neural Network.
9. Explain Multi-layer perceptron in detail.
10. Explain about perceptron learning with the implementation of OR function.
11. Summarize the mathematical model of neural unit
12. Explain the following non-linear functions by covering mathematical intuition, plots and derivates
    a. Sigmoid
    b. ReLU
    c. Tanh
13. Explain different applications of deep learning
14. Consider a unit with the following input vector, weight vector, and bias and compute the output by applying sigmoid, relu and tanh activation functions. a. w = [0.2,0.3,0.9] b. b = 0.5 c. x = [0.5,0.6,0.1]
15. Differentiate between deep supervised and unsupervised learning in detail with examples.