1. Describe the structure of an artificial neuron. How is it similar to a biological neuron? What are its main components?

Artificial Neurons

Artificial neuron also known as perceptron is the basic unit of the neural network. In simple terms, it is a mathematical function based on a model of biological neurons. It can also be seen as a simple logic gate with binary outputs.

We have heard of the latest advancements in the field of deep learning due to the usage of different neural networks. Most of these achievements are simply astonishing and I find myself amazed after reading every new article on the advancements in this field almost every week. At the most basic level, all such neural networks are made up of artificial neurons that try to mimic the working of biological neurons. I had a curiosity about understanding how these artificial neurons compare to the structure of biological neurons in our brains and if possibly this could lead to a way to improve neural networks further. So if you are curious about this topic too, then let’s embark on a short 5-minute journey to understand this topic in detail…

First, let’s understand how biological neurons work inside our brains…

Biological Neurons

Neurons are the basic functional units of the nervous system, and they generate electrical signals called action potentials, which allows them to quickly transmit information over long distances.  
Almost all the neurons have three basic functions essential for the normal functioning of all the cells in the body.

These are to:  
1. Receive signals (or information) from outside.  
2. Process the incoming signals and determine whether or not the information should be passed along.  
3. Communicate signals to target cells which might be other neurons or muscles or glands.

Now let us understand the basic parts of a neuron to get a deeper insight into how they actually work…

Dendrite

Dendrites are responsible for getting incoming signals from outside

2. Soma

Soma is the cell body responsible for the processing of input signals and deciding whether a neuron should fire an output signal

3. Axon

Axon is responsible for getting processed signals from neuron to relevant cells

4. Synapse

Synapse is the connection between an axon and other neuron dendrites

Working of the parts

The task of receiving the incoming information is done by dendrites, and processing generally takes place in the cell body. Incoming signals can be either excitatory — which means they tend to make the neuron fire (generate an electrical impulse) — or inhibitory — which means that they tend to keep the neuron from firing.

Most neurons receive many input signals throughout their dendritic trees. A single neuron may have more than one set of dendrites and may receive many thousands of input signals. Whether or not a neuron is excited into firing an impulse depends on the sum of all of the excitatory and inhibitory signals it receives. The processing of this information happens in soma which is neuron cell body. If the neuron does end up firing, the nerve impulse, or action potential, is conducted down the axon.

Towards its end, the axon splits up into many branches and develops bulbous swellings known as axon terminals (or nerve terminals). These axon terminals make connections on target cells.

Artificial Neurons

Artificial neuron also known as perceptron is the basic unit of the neural network. In simple terms, it is a mathematical function based on a model of biological neurons. It can also be seen as a simple logic gate with binary outputs. They are sometimes also called perceptrons.

Each artificial neuron has the following main functions:

Takes inputs from the input layer

Weighs them separately and sums them up

Pass this sum through a nonlinear function to produce output.

The perceptron(neuron) consists of 4 parts:

Input values or One input layer  
We pass input values to a neuron using this layer. It might be something as simple as a collection of array values. It is similar to a dendrite in biological neurons.

Weights and Bias  
Weights are a collection of array values which are multiplied to the respective input values. We then take a sum of all these multiplied values which is called a weighted sum. Next, we add a bias value to the weighted sum to get final value for prediction by our neuron.

Activation Function  
Activation Function decides whether or not a neuron is fired. It decides which of the two output values should be generated by the neuron.

Output Layer  
Output layer gives the final output of a neuron which can then be passed to other neurons in the network or taken as the final output value.

2.What are the different types of activation functions popularly used? Explain each of them.

3 Types of Neural Networks Activation Functions

Binary Step Function.

Linear Activation Function.

Sigmoid/Logistic Activation Function.

The derivative of the Sigmoid Activation Function.

Tanh Function (Hyperbolic Tangent)

Gradient of the Tanh Activation Function.

ReLU Activation Function.

The Dying ReLU problem.

Types of Activation Functions

1. Sigmoid Function

In an ANN, the sigmoid function is a non-linear AF used primarily in feedforward neural networks. It is a differentiable real function, defined for real input values, and containing positive derivatives everywhere with a specific degree of smoothness. The sigmoid function appears in the output layer of the deep learning models and is used for predicting probability-based outputs. The sigmoid function is represented as:

[Source](https://arxiv.org/pdf/1811.03378.pdf)

Generally, the derivatives of the sigmoid function are applied to learning algorithms. The graph of the sigmoid function is ‘S’ shaped.

Some of the major drawbacks of the sigmoid function include gradient saturation, slow convergence, sharp damp gradients during backpropagation from within deeper hidden layers to the input layers, and non-zero centered output that causes the gradient updates to propagate in varying directions.

2. Hyperbolic Tangent Function (Tanh)

The hyperbolic tangent function, a.k.a., the tanh function, is another type of AF. It is a smoother, zero-centered function having a range between -1 to 1. As a result, the output of the tanh function is represented by:

[Source](https://arxiv.org/pdf/1811.03378.pdf)

The tanh function is much more extensively used than the sigmoid function since it delivers better training performance for multilayer neural networks. The biggest advantage of the tanh function is that it produces a zero-centered output, thereby supporting the backpropagation process. The tanh function has been mostly used in recurrent neural networks for natural language processing and speech recognition tasks.

However, the tanh function, too, has a limitation – just like the sigmoid function, it cannot solve the vanishing gradient problem. Also, the tanh function can only attain a gradient of 1 when the input value is 0 (x is zero). As a result, the function can produce some dead neurons during the computation process.

3.

* 1. Explain, in details, Rosenblatt’s perceptron model. How can a set of data be classified using a simple perceptron?
  2. Use a simple perceptron with weights *w*0, *w*1, and *w*2 as −1, 2, and 1, respectively, to classify data points (3, 4); (5, 2); (1, −3); (−8, −3); (−3, 0).

a.Explain, in details, Rosenblatt’s perceptron model. How can a set of data be classified using a simple perceptron?

Artificial Neural Networks (ANN) are machine learning models that have been inspired by the brain functioning. Through my next posts I will try to introduce artificial neural networks in a simple high level way, highlighting its capabilities but also showing its limitations. In this first post, I will introduce the simplest neural network, the Rosenblatt Perceptron, a neural network compound of a single artificial neuron. This artificial neuron model is the basis of today’s complex neural networks and was until the mid-eighties state of the art in ANN.

As ANN are inspired by the brain, let’s start describing how the brain works. The brain is a connected network of neurons (approximately 21\*10^9 ) that communicate by means of electric and chemical signals through a process that is known as synapse, in which information from one neuron flows to other neurons. When a neuron is inactive, the electrical difference across the membrane of the neuron (resting potential) is typically around –70 mV. The electrical impulses received from other neurons connected to its axon delivers neurotransmitters that can be both inhibitory or excitatory. If excitatory neurotransmitters increase the membrane voltage and it reaches a certain threshold the cell depolarizes and triggers the action potential that travels from the dendrite to other neurons axons.   Communication among neurons therefore takes place when the action potential arrives to the axon terminal of other presynaptic neuron.

Inspired by the biological principles of a neuron, Franck Rosenblatt developed the concept of the perceptron at Cornell Aeronautical Laboratory in 1957:

A Neuron receives ‘communication messages’ from other neurons in form of electrical impulses of different strength that can be excitatory or inhibitory.

A neuron integrates all the impulses received from other neurons.

If the resulting integration is larger than a certain threshold the neuron ‘fires,’ triggering the action potential that is transmitted to other connected neurons.

Frank Rosenblatt’s perceptron model

Rosenblatt perceptron is a binary single neuron model. The inputs integration is implemented through the addition of the weighted inputs that have fixed weights obtained during the training stage. If the result of this addition is larger than a given threshold θ the neuron fires. When the neuron fires its output is set to 1, otherwise it’s set to 0.

The equation can be re-written as follows including what it’s known as the bias term: .

This model implements the functioning of a single neuron that can solve linear classification problems through very simple learning algorithms. Rosenblatt Perceptrons are considered as the first generation of neural networks (the network is only compound of one neuron ☺ ). This simple single neuron model has the main limitation of not being able to solve non-linear separable problems. In my next post I will describe how this advantage was overcome and what happens when we have a layer of various perceptrons or try different neuron activation functions.

b.Use a simple perceptron with weights *w*0, *w*1, and *w*2 as −1, 2, and 1, respectively, to classify data points (3, 4); (5, 2); (1, −3); (−8, −3); (−3, 0).

This post will discuss the famous Perceptron Learning Algorithm, originally proposed by [Frank Rosenblatt](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.335.3398&rep=rep1&type=pdf) in 1943, later refined and carefully analyzed by [Minsky and Papert](http://science.sciencemag.org/content/165/3895/780) in 1969. This is a follow-up post of my previous posts on the [McCulloch-Pitts neuron](https://towardsdatascience.com/mcculloch-pitts-model-5fdf65ac5dd1) model and the [Perceptron](https://towardsdatascience.com/4d8c70d5cc8d) model.

Citation Note: The concept, the content, and the structure of this article were entirely based on Prof. [Mitesh Khapra](https://www.cse.iitm.ac.in/~miteshk/)’s lectures slides and videos of the [CS7015: Deep Learning](https://www.cse.iitm.ac.in/~miteshk/CS7015.html) course taught at IIT Madras.

Perceptron

You can just go through my previous post on the perceptron model (linked above) but I will assume that you won’t. So here goes, a perceptron is not the Sigmoid neuron we use in ANNs or any deep learning networks today.

The perceptron model is a more general computational model than McCulloch-Pitts neuron. It takes an input, aggregates it (weighted sum) and returns 1 only if the aggregated sum is more than some threshold else returns 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 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 thing I mentioned above). We learn the weights, we get the function. Let's use a perceptron to learn an OR function.

4.Explain the basic structure of a multi-layer perceptron. Explain how it can solve the XOR problem.

The XOr problem is that we need to build a Neural Network (a perceptron in our case) to produce the truth table related to the XOr logical operator. This is a binary classification problem. Hence, supervised learning is a better way to solve it. In this case, we will be using perceptrons.

Solving the XOR problem using MLP

In this blog post, we shall cover the basics of what the XOR problem is, and how we can solve it using MLP.

What is XOR?

Exclusive or is a logical operation that outputs true when the inputs differ.

For the XOR gate, the TRUTH table will be as follows

XOR truth table

XOR is a classification problem, as it renders binary distinct outputs. If we plot the INPUTS vs OUTPUTS for the XOR gate, it would look something like

The graph plots the two inputs corresponding to their output. Visualizing this plot, we can see that it is impossible to separate the different outputs (1 and 0) using a linear equation.

To separate the two outputs using linear equation(s), we would need to draw two separate lines like

Image courtesy: <https://tex.stackexchange.com/>

The above graph perfectly shows why these outputs cannot be separated using a single linear equation. This was a major problem with the initial perceptrons (single layer approach).

What is the XOR problem?

As we have seen above, it is impossible to separate the XOR outputs using just a single linear equation. This is a major problem as during the training of machines, for optimized outputs, the machine is expected to form the mathematical equations on its own.

For a problem resembling the outputs of XOR, it was impossible for the machine to set up an equation for good outputs. This is what led to the birth of the concept of hidden layers which are extensively used in Artificial Neural Networks.

Let’s call the output to be Y, so

Y = A1X1 + A2X2 + A3X3 + …. + B

Here B is the bias, and A1, A2, A3 are the weights. Weights are used to control the signal (strength of the connection) of the connection.

Y can also be called the weighted sum.

The information flow inside a perceptron is a feed-forward type, meaning that the signal flows in a single direction from the input layer to the output layer. All the input layers are independent of each other.

The variation in the weight variables controls the process of conversion of the input values to the output values.

The main limitation of a single-layer architecture (perceptrons) is that it separates the data points using a single line. This has a drawback in a problem similar to the XOR problem, as the data points are linearly inseparable.

How is the XOR problem solved?

The solution to the XOR problem lies in multidimensional analysis. We plug in numerous inputs in various layers of interpretation and processing, to generate the optimum outputs.

The inner layers for deeper processing of the inputs are known as hidden layers. The hidden layers are not dependent on any other layers. This architecture is known as Multilayer Perceptron (MLP).

5.What is artificial neural network (ANN)? Explain some of the salient highlights in the different architectural options for ANN ?

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the brain. ANNs, like people, learn by examples. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process.

An [Artificial Neural Network (ANN)](https://www.geeksforgeeks.org/implementing-ann-training-process-in-python/) is an information processing paradigm that is inspired by the brain. ANNs, like people, learn by examples. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning largely involves adjustments to the synaptic connections that exist between the neurons.

The model of an artificial neural network can be specified by three entities: 

Interconnections

[Activation functions](https://www.geeksforgeeks.org/activation-functions-neural-networks/)

Learning rules

Interconnections:

Interconnection can be defined as the way processing elements (Neuron) in ANN are connected to each other. Hence, the arrangements of these processing elements and geometry of interconnections are very essential in ANN.   
These arrangements always have two layers that are common to all network architectures, the Input layer and output layer where the input layer buffers the input signal, and the output layer generates the output of the network. The third layer is the Hidden layer, in which neurons are neither kept in the input layer nor in the output layer. These neurons are hidden from the people who are interfacing with the system and act as a black box to them. By increasing the hidden layers with neurons, the system’s computational and processing power can be increased but the training phenomena of the system get more complex at the same time.

There exist five basic types of neuron connection architecture :

Single-layer feed-forward network

Multilayer feed-forward network

Single node with its own feedback

Single-layer recurrent network

Multilayer recurrent network

1. Single-layer feed-forward network

In this type of network, we have only two layers input layer and the output layer but the input layer does not count because no computation is performed in this layer. The output layer is formed when different weights are applied to input nodes and the cumulative effect per node is taken. After this, the neurons collectively give the output layer to compute the output signals.

2. Multilayer feed-forward network

This layer also has a hidden layer that is internal to the network and has no direct contact with the external layer. The existence of one or more hidden layers enables the network to be computationally stronger, a feed-forward network because of information flow through the input function, and the intermediate computations used to determine the output Z. There are no feedback connections in which outputs of the model are fed back into itself.

3. Single node with its own feedback

Single Node with own Feedback

When outputs can be directed back as inputs to the same layer or preceding layer nodes, then it results in feedback networks. Recurrent networks are feedback networks with closed loops. The above figure shows a single recurrent network having a single neuron with feedback to itself.

4. Single-layer recurrent network

The above network is a single-layer network with a feedback connection in which the processing element’s output can be directed back to itself or to another processing element or both. A recurrent neural network is a class of artificial neural networks where connections between nodes form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs.

5. Multilayer recurrent network

In this type of network, processing element output can be directed to the processing element in the same layer and in the preceding layer forming a multilayer recurrent network. They perform the same task for every element of a sequence, with the output being dependent on the previous computations. Inputs are not needed at each time step. The main feature of a Recurrent Neural Network is its hidden state, which captures some information about a sequence.

6.Explain the learning process of an ANN. Explain, with example, the challenge in assigning synaptic weights for the interconnection between neurons? How can this challenge be addressed?

Artificial Neural Networks are computational models and inspire by the human brain. Many of the recent advancements have been made in the field of Artificial Intelligence, including Voice Recognition, Image Recognition, Robotics using Artificial Neural Networks. They are the biologically inspired simulations performed on the computer to perform certain specific tasks like -

Clustering

Classification

Pattern Recognition

In general -  It is a biologically inspired network of artificial neurons configured to perform specific tasks. These biological methods of computing are known as the next major advancement in the Computing Industry.

What is a Neural Network?

The term ‘Neural’ has origin from the human (animal) nervous system’s basic functional unit ‘neuron’ or nerve cells present in the brain and other parts of the human (animal) body. A neural network is a group of algorithms that certify the underlying relationship in a set of data similar to the human brain. The neural network helps to change the input so that the network gives the best result without redesigning the output procedure. You can also learn more about [ONNX](https://www.xenonstack.com/blog/onnx/) in this insight.

What are the advantages and Disadvantages of Neural Networks?

The advantages of of Neural Networks are listed below:

A neural network can perform tasks that a linear program can not.

When an element of the neural network fails, its parallel nature can continue without any problem.

A neural network learns and reprogramming is not necessary.

It can be implemented in any application.

It can be performed without any problem.

The disadvantages of of Neural Networks are described below:

The neural network needs training to operate.

The architecture of a neural network is different from the architecture of microprocessors. Therefore, emulation is necessary.

Requires high processing time for large neural networks.

A combination of neurons whose performance vector signifies the creation of real instance parameters of a particular type of an object or it's part.Click to explore about our, [Capsule Networks Benefits](https://www.xenonstack.com/insights/capsule-networks)

What are the parts of Neuron and their Functions?

The typical nerve cell of the human brain comprises of four parts:

Function of Dendrite

It receives signals from other neurons.

Soma (cell body)

It sums all the incoming signals to generate input.

Axon Structure

When the sum reaches a threshold value, the neuron fires, and the signal travels down the axon to the other neurons.

Synapses Working

The point of interconnection of one neuron with other neurons. The amount of signal transmitted depends upon the strength (synaptic weights) of the connections.The connections can be inhibitory (decreasing strength) or excitatory (increasing strength) in nature. So, a neural network, in general, has a connected network of billions of neurons with a trillion of interconnections between them.

What is the difference between Brain and Computer?

What is the difference between Artificial Neural Networks (ANN) VS Biological Neural Networks (BNN)?

|  |  |  |
| --- | --- | --- |
| Characteristics | Artificial Neural Network (ANN) | Biological(Real) Neural Network (BNN) |
| Speed | Faster in processing information. Response time is in nanoseconds. | Slower in processing information. The response time is in milliseconds. |
| Processing | Serial processing. | Massively parallel processing. |
| Size & Complexity | Less size & complexity. It does not perform complex pattern recognition tasks. | A highly complex and dense network of interconnected neurons containing neurons of the order of 1011 with 1015 of interconnections.<strong |
| Storage | Information storage is replaceable means replacing new data with an old one. | A highly complex and dense network of interconnected neurons containing neurons of the order of 1011 with 1015 of interconnections. |
| Fault tolerance | Fault intolerant. Corrupt information cannot retrieve in case of failure of the system. | Information storage is adaptable means new information is added by adjusting the interconnection strengths without destroying old information. |
| Control Mechanism | There is a control unit for controlling computing activities | No specific control mechanism external to the computing task |

Artificial Neural Networks with Biological Neural Network

Neural Networks resemble the human brain in the following two ways -

A neural network acquires knowledge through learning.

A neural network's knowledge is a store within inter-neuron connection strengths known as synaptic weights.

|  |  |
| --- | --- |
| Von Neumann Architecture Based Computing | Ann Based Computing |
| Serial processing - processing instruction and problem rule one at the time (sequential) | Parallel processing - several processors perform simultaneously (multitasking) |
| Function logically with a set of if & else rules - rule-based approach | Function by learning pattern from a given input (image, text or video, etc.) |
| Programmable by higher-level languages such as C, [Java](https://www.xenonstack.com/blog/serverless-architecture/), C++, etc. | ANN is, in essence, the program itself. |
| Requires either big or error-prone parallel processors | Use of application-specific multi-chips. |

Artificial Neural Network (ANN) Vs Biological Neural Network (BNN)

The Biological Neural Network's dendrites are analogous to the weighted inputs based on their synaptic interconnection in the Artificial Neural Network.

The cell body is comparable to the artificial neuron unit in the Artificial Neural Network, comprising summation and threshold unit.

Axon carries output that is analogous to the output unit in the case of an Artificial Neural Network. So, it is model using the working of basic biological neurons.

How does Artificial Neural Network works?

Artificial Neural Networks can be viewed as weighted directed graphs in which artificial neurons are nodes, and directed edges with weights are connections between neuron outputs and neuron inputs.

The Artificial Neural Network receives information from the external world in pattern and image in vector form. These inputs are designated by the notation x(n) for n number of inputs.

Each input is multiplied by its corresponding weights. Weights are the information used by the neural network to solve a problem. Typically weight represents the strength of the interconnection between neurons inside the Neural Network.

The weighted inputs are all summed up inside the computing unit (artificial neuron). In case the weighted sum is zero, bias is added to make the output not- zero or to scale up the system response. Bias has the weight and input always equal to ‘1'.

The sum corresponds to any numerical value ranging from 0 to infinity. To limit the response to arrive at the desired value, the threshold value is set up. For this, the sum is forward through an activation function.

The activation function is set to the transfer function to get the desired output. There are linear as well as the nonlinear activation function.

What are the commonly used activation functions?

Some of the commonly used activation function is - binary, sigmoidal (linear) and tan hyperbolic sigmoidal functions(nonlinear).

Binary - The output has only two values, either 0 and 1. For this, the threshold value is set up. If the net weighted input is greater than 1, the output is assumed as one otherwise zero.

Sigmoidal Hyperbolic - This function has an ‘S’ shaped curve. Here the tan hyperbolic function is used to approximate output from net input. The function is defined as - f (x) = (1/1+ exp(-????x)) where ???? - steepness parameter.

7.Explain, in details, the backpropagation algorithm. What are the limitations of this algorithm?

Backpropagation could be rather sensitive to noisy data and irregularity. The performance of backpropagation relies very heavily on the training data. Backpropagation needs a very large amount of time for training. Backpropagation requires a matrix-based method instead of mini-batch.

Backpropagation

TABLE OF CONTENTS

[What is backpropagation?](https://www.engati.com/glossary/back-propagation#toc-what-is-backpropagation-)

[What is the objective of backpropagation?](https://www.engati.com/glossary/back-propagation#toc-what-is-the-objective-of-backpropagation-)

[How does backpropagation work?](https://www.engati.com/glossary/back-propagation#toc-how-does-backpropagation-work-)

[What does the loss function do?](https://www.engati.com/glossary/back-propagation#toc-what-does-the-loss-function-do-)

[What are the advantages of backpropagation?](https://www.engati.com/glossary/back-propagation#toc-what-are-the-advantages-of-backpropagation-)

[What are the disadvantages of backpropagation?](https://www.engati.com/glossary/back-propagation#toc-what-are-the-disadvantages-of-backpropagation-)

[What are the types of backpropagation networks?](https://www.engati.com/glossary/back-propagation#toc-what-are-the-types-of-backpropagation-networks-)

[Is backpropagation greedy?](https://www.engati.com/glossary/back-propagation#toc-is-backpropagation-greedy-)

[What is bias in backpropagation?](https://www.engati.com/glossary/back-propagation#toc-what-is-bias-in-backpropagation-)

Automate your business at $5/day with Engati

REQUEST A DEMO

What is backpropagation?

Backpropagation is an algorithm that uses gradient descent for supervised learning of artificial neural networks. Short for backward propagation of errors, it calculates the gradient of the error function with regard to the neural network's weights.

The gradient of the final layer of weights is calculated first in backpropagation, and the gradient of the first layer of weights is calculated last. The calculation of e gradient of error essentially occurs backwards through the artificial neural network.

It reuses partial computations of the [gradient](https://www.engati.com/glossary/vanishing-gradient-problem) of a layer while calculating the gradient of the next layer.

Backpropagation was one of the very first techniques that showed that it was possible for artificial neural networks to learn good internal representations.

On inspecting multilayer feedforward networks that were trained with the use of backpropagation, it came to light that multiple nodes learned features that were akin to those designed by human experts.

The algorithm was found to be so efficient that it did not require human experts to discover appropriate features. Because of that, problems that could not be handled by artificial neural networks were now fair game with backpropagation.

Here’s a way for you to get an idea of what backpropagation is like. In the movie Men in Black III, Will Smith’s character has a fight with a villian in which he got shot. But, he had a way of traveling back in time, knowing where the projectile would hit and thus dodging it. This time he got a little further, but got hit by the next projectile, so he went back in time again, and dodged both projectiles. He kept making more progress, getting hit at some point, and going back in time till he reached so close to the villian that he could attack him.

That’s similar to what happens in backpropagation. The backpropagation algorithm goes back from the results obtained and rectifies its errors at every node of the neural network so as to improve the performance of that neural network model.

Source: [Towards Data Science](https://towardsdatascience.com/how-does-back-propagation-in-artificial-neural-networks-work-c7cad873ea7)

What is the objective of backpropagation?

Backpropagation algorithms are essentially the most important part of artificial neural networks. Their primary purpose is to develop a learning algorithm for multilayer feedforward neural networks, empowering the networks to be trained to capture the mapping implicitly.

Its goal is to optimize the weights, thus allowing the neural network to learn how to correctly map arbitrary inputs to outputs.

How does backpropagation work?

There are four layers in a backpropagation network. These are the input layer, hidden layer, hidden layer II and final output layer. Every layer works in its own way to take action in order to get the results that we desire and correlate the scenarios to the conditions.

Here’s how backpropagation works:

The input layer receives the input x.

Weights w are used to model the input.

The output is calculated by every hidden layer and data is ready at the output layer

We find the error by observing the difference between the actual output and the desired output.

Go back and adjust the weights in the hidden layer to minimize the error in future runs.

The training phase is done under supervision and the process mentioned above needs to be repeated until the actual output matches the desired output. Once that is done, the model can be used in production.

‍

What does the loss function do?

One or multiple variables will be mapped to real numbers. These represent a price related to those numbers. The [loss function](https://www.engati.com/glossary/loss-function) will calculate the difference between the network output and its probable output.

‍

What are the advantages of backpropagation?

[Backpropagation](https://en.wikipedia.org/wiki/Backpropagation) has numerous advantages, here are some of the most significant ones.

No parameters need to be tuned

The model does not need to learn the features of the function

Backpropagation is a flexible method because prior knowledge of the network is not required.

It is a fast method and is rather easy to implement.

The approach tends to work rather well in most situations.

The user does not need to learn special functions.

‍

What are the disadvantages of backpropagation?

The biggest disadvantages of backpropagation are:

Backpropagation could be rather sensitive to noisy data and irregularity.

The performance of backpropagation relies very heavily on the training data.

Backpropagation needs a very large amount of time for training.

Backpropagation requires a matrix-based method instead of mini-batch.

What are the types of backpropagation networks?

Two kinds of backpropagation networks exist: static and recurrent.

Static backpropagation

In static backpropagation, static outputs are generated by mapping static inputs. Static backpropagation networks have the ability to solve static classification problems like optical character recognition.

Recurrent backpropagation

We conduct recurrent backpropagation until we reach a specific threshold. After that, we calculate the error and propagate it backward.

In static backpropagation, immediate mapping is possible, while in recurrent backpropagation, immediate mapping does not happen.

‍

Is backpropagation greedy?

Backpropagation is significantly faster than other neural network algorithms. You could consider backpropagation to be like an advanced greedy approach. It helps you get the result that you want in a faster manner and has even reduced training times from months to hours. It could pretty much be considered to be the backbone of the neural network.

‍

What is bias in backpropagation?

Most implementations of backpropagation algorithms include an extra class of weights that are called biases. These biases are essentially values that get added to the sums calculated at every node other than input nodes during the feedforward phase.

The negative of a bias could be called a threshold. To keep things simple, biases tend to be visualized as values associated with each node in the intermediate and output layers of a network. In practice, however, they are treated no differently than other weights, with all biases being weights that are associated with vectors leading from a single node whose location is outside of the main network and whose activation is always 1.

8.Describe, in details, the process of adjusting the interconnection weights in a multi-layer neural network.

Artificial Neural Network - Basic Concepts

Neural networks are parallel computing devices, which is basically an attempt to make a computer model of the brain. The main objective is to develop a system to perform various computational tasks faster than the traditional systems. These tasks include pattern recognition and classification, approximation, optimization, and data clustering.

What is Artificial Neural Network?

Artificial Neural Network (ANN) is an efficient computing system whose central theme is borrowed from the analogy of biological neural networks. ANNs are also named as “artificial neural systems,” or “parallel distributed processing systems,” or “connectionist systems.” ANN acquires a large collection of units that are interconnected in some pattern to allow communication between the units. These units, also referred to as nodes or neurons, are simple processors which operate in parallel.

Every neuron is connected with other neuron through a connection link. Each connection link is associated with a weight that has information about the input signal. This is the most useful information for neurons to solve a particular problem because the weight usually excites or inhibits the signal that is being communicated. Each neuron has an internal state, which is called an activation signal. Output signals, which are produced after combining the input signals and activation rule, may be sent to other units.

A Brief History of ANN

The history of ANN can be divided into the following three eras −

ANN during 1940s to 1960s

Some key developments of this era are as follows −

1943 − It has been assumed that the concept of neural network started with the work of physiologist, Warren McCulloch, and mathematician, Walter Pitts, when in 1943 they modeled a simple neural network using electrical circuits in order to describe how neurons in the brain might work.

1949 − Donald Hebb’s book, The Organization of Behavior, put forth the fact that repeated activation of one neuron by another increases its strength each time they are used.

1956 − An associative memory network was introduced by Taylor.

1958 − A learning method for McCulloch and Pitts neuron model named Perceptron was invented by Rosenblatt.

1960 − Bernard Widrow and Marcian Hoff developed models called "ADALINE" and “MADALINE.”

ANN during 1960s to 1980s

Some key developments of this era are as follows −

1961 − Rosenblatt made an unsuccessful attempt but proposed the “backpropagation” scheme for multilayer networks.

1964 − Taylor constructed a winner-take-all circuit with inhibitions among output units.

1969 − Multilayer perceptron (MLP) was invented by Minsky and Papert.

1971 − Kohonen developed Associative memories.

1976 − Stephen Grossberg and Gail Carpenter developed Adaptive resonance theory.

ANN from 1980s till Present

Some key developments of this era are as follows −

1982 − The major development was Hopfield’s Energy approach.

1985 − Boltzmann machine was developed by Ackley, Hinton, and Sejnowski.

1986 − Rumelhart, Hinton, and Williams introduced Generalised Delta Rule.

1988 − Kosko developed Binary Associative Memory (BAM) and also gave the concept of Fuzzy Logic in ANN.

The historical review shows that significant progress has been made in this field. Neural network based chips are emerging and applications to complex problems are being developed. Surely, today is a period of transition for neural network technology.

Biological Neuron

A nerve cell (neuron) is a special biological cell that processes information. According to an estimation, there are huge number of neurons, approximately 1011 with numerous interconnections, approximately 1015.

9.What are the steps in the backpropagation algorithm? Why a multi-layer neural network is required?

What is Artificial Neural Networks?

A neural network is a group of connected I/O units where each connection has a weight associated with its computer programs. It helps you to build predictive models from large databases. This model builds upon the human nervous system. It helps you to conduct image understanding, human learning, computer speech, etc.

What is Backpropagation?

Backpropagation is the essence of neural network training. It is the method of fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and make the model reliable by increasing its generalization.

Backpropagation in neural network is a short form for “backward propagation of errors.” It is a standard method of training artificial neural networks. This method helps calculate the gradient of a loss function with respect to all the weights in the network.

In this tutorial, you will learn:

[What is Artificial Neural Networks?](https://www.guru99.com/backpropogation-neural-network.html#1)

[What is Backpropagation?](https://www.guru99.com/backpropogation-neural-network.html#2)

[How Backpropagation Algorithm Works](https://www.guru99.com/backpropogation-neural-network.html#3)

[Why We Need Backpropagation?](https://www.guru99.com/backpropogation-neural-network.html#4)

[What is a Feed Forward Network?](https://www.guru99.com/backpropogation-neural-network.html#5)

[Types of Backpropagation Networks](https://www.guru99.com/backpropogation-neural-network.html#6)

[History of Backpropagation](https://www.guru99.com/backpropogation-neural-network.html#7)

[Backpropagation Key Points](https://www.guru99.com/backpropogation-neural-network.html#8)

[Best practice Backpropagation](https://www.guru99.com/backpropogation-neural-network.html#9)

[Disadvantages of using Backpropagation](https://www.guru99.com/backpropogation-neural-network.html#10)

How Backpropagation Algorithm Works

The Back propagation algorithm in neural network computes the gradient of the loss function for a single weight by the chain rule. It efficiently computes one layer at a time, unlike a native direct computation. It computes the gradient, but it does not define how the gradient is used. It generalizes the computation in the delta rule.

Consider the following Back propagation neural network example diagram to understand:

Inputs X, arrive through the preconnected path

Input is modeled using real weights W. The weights are usually randomly selected.

Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.

Calculate the error in the outputs

ErrorB= Actual Output – Desired Output

Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.

Keep repeating the process until the desired output is achieved

Why We Need Backpropagation?

Most prominent advantages of Backpropagation are:

Backpropagation is fast, simple and easy to program

It has no parameters to tune apart from the numbers of input

It is a flexible method as it does not require prior knowledge about the network

It is a standard method that generally works well

It does not need any special mention of the features of the function to be learned.

What is a Feed Forward Network?

A feedforward neural network is an artificial neural network where the nodes never form a cycle. This kind of neural network has an input layer, hidden layers, and an output layer. It is the first and simplest type of artificial neural network.

Types of Backpropagation Networks

Two Types of Backpropagation Networks are:

Static Back-propagation

Recurrent Backpropagation

Static back-propagation:

It is one kind of backpropagation network which produces a mapping of a static input for static output. It is useful to solve static classification issues like optical character recognition.

Recurrent Backpropagation:

Recurrent Back propagation in data mining is fed forward until a fixed value is achieved. After that, the error is computed and propagated backward.

The main difference between both of these methods is: that the mapping is rapid in static back-propagation while it is nonstatic in recurrent backpropagation.

History of Backpropagation

In 1961, the basics concept of continuous backpropagation were derived in the context of control theory by J. Kelly, Henry Arthur, and E. Bryson.

In 1969, Bryson and Ho gave a multi-stage dynamic system optimization method.

In 1974, Werbos stated the possibility of applying this principle in an artificial neural network.

In 1982, Hopfield brought his idea of a neural network.

In 1986, by the effort of David E. Rumelhart, Geoffrey E. Hinton, Ronald J. Williams, backpropagation gained recognition.

In 1993, Wan was the first person to win an international pattern recognition contest with the help of the backpropagation method.

10.Write short notes on:

* + - 1. Artificial neuron
      2. Multi-layer perceptron
      3. Deep learning
      4. Learning rate

1.artificial neuron :

An artificial neuron is a connection point in an [artificial neural network](https://www.techtarget.com/searchenterpriseai/definition/neural-network). Artificial neural networks, like the human body's biological neural network, have a layered architecture and each [network node](https://www.techtarget.com/searchnetworking/definition/node) (connection point) has the capability to process input and forward output to other nodes in the network. In both artificial and biological architectures, the nodes are called neurons and the connections are characterized by synaptic weights, which represent the significance of the connection. As new data is received and processed, the synaptic weights change and this is how learning occurs.

Artificial neurons are modeled after the hierarchical arrangement of neurons in biological sensory systems. In the visual system, for example, light input passes through neurons in successive layers of the retina before being passed to neurons in the thalamus of the brain and then on to neurons in the brain's visual cortex. As the neurons pass signals through an increasing number of layers, the brain progressively extracts more information until it is confident it can identify what the person is seeing. In artificial intelligence, this fine tuning process is known as [deep learning](https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neural-network).

In both artificial and biological networks, when neurons process the input they receive, they decide whether the output should be passed on to the next layer as input. The decision of whether or not to send information on is called [bias](https://www.techtarget.com/whatis/definition/bias) and it’s determined by an activation [function](https://www.techtarget.com/whatis/definition/function) built into the system. For example, an artificial neuron may only pass an output signal on to the next layer if its inputs (which are actually voltages) sum to a value above some particular threshold value. Because activation functions can either be linear or non-linear, neurons will often have a wide range of convergence and divergence. Divergence is the ability for one neuron to communicate with many other neurons in the network and convergence is the ability for one neuron to receive input from many other neurons in the network.

2.multi layer perceptron :

Multi layer perceptron (MLP) is a supplement of feed forward neural network. It consists of three types of layers—the input layer, output layer and hidden layer, as shown in Fig. 3. The input layer receives the input signal to be processed.

Multi layer [perceptron](https://www.sciencedirect.com/topics/engineering/perceptron) (MLP) is a supplement of feed forward [neural network](https://www.sciencedirect.com/topics/chemical-engineering/neural-network). It consists of three types of layers—the input layer, output layer and hidden layer, as shown in Fig. 3. The input layer receives the input signal to be processed. The required task such as prediction and classification is performed by the output layer. An arbitrary number of hidden layers that are placed in between the input and output layer are the true computational engine of the [MLP](https://www.sciencedirect.com/topics/engineering/perceptron). Similar to a [feed forward network](https://www.sciencedirect.com/topics/computer-science/feedforward-network) in a MLP the data flows in the forward direction from input to output layer. The neurons in the MLP are trained with the [back propagation](https://www.sciencedirect.com/topics/computer-science/backpropagation) learning algorithm. MLPs are designed to approximate any continuous function and can solve problems which are not linearly separable. The major use cases of MLP are pattern classification, recognition, prediction and approximation.

3.deep learning :

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost.

4.learning rate :

Learning rate, generally represented by the symbol 'α', shown in equation-4, is a hyper-parameter used to control the rate at which an algorithm updates the parameter estimates or learns the values of the parameters.

In machine learning, we deal with two types of parameters; 1) machine learnable parameters and 2) hyper-parameters. The Machine learnable parameters are the one which the algorithms learn/estimate on their own during the training for a given dataset. In equation-3,  β0, β1 and β2 are the machine learnable parameters. The Hyper-parameters are the one which the machine learning engineers or data scientists will assign specific values to, to control the way the algorithms learn and also to tune the performance of the model. Learning rate, generally represented by the symbol ‘α’, shown in equation-4, is a hyper-parameter used to control the rate at which an algorithm updates the parameter estimates or learns the values of the parameters.

11.Write the difference between:-

* + - 1. Activation function vs threshold function
      2. Step function vs sigmoid function
      3. Single layer vs multi-layer perceptron

1.activation function vs threshold function

An Activation Function decides whether a neuron should be activated or not. This means that it will decide whether the neuron’s input to the network is important or not in the process of prediction using simpler mathematical operations.

The role of the Activation Function is to derive output from a set of input values fed to a node (or a layer).

But—

Let’s take a step back and clarify: What exactly is a node?

Well, if we compare the neural network to our brain, a node is a replica of a neuron that receives a set of input signals—external stimuli.

Threshold Functions Definition 1 A Boolean formula is a Boolean function f : {0, 1} n → {0, 1} which can be computed by Boolean circuits which are trees, i.e. so that every input bit is read exactly once. We will regard Boolean functions as maps 2[n] → {0, 1}, where 2[n] denotes the power set of [n], i.e. the set of all subsets of [n]. Definition 2 A Boolean function is monotone if for all S ⊆ T, f(S) ≤ f(T), i.e. the circuit for f has no negations (we can furthermore assume the circuit is over the de Morgan basis of AND/OR’s). One way to measure the complexity of a Boolean formula is by the size of the corresponding tree circuit. Definition 3 The size of a Boolean formula is the number of gates in the corresponding circuit, including input gates. In this lecture we will consider a special class of monotone functions, threshold functions. Definition 4 The threshold function Thn k : 2[n] → {0, 1} is defined to output 1 if |S| ≥ k and 0 otherwise. For example, Thn 1 (S) is the OR function, Thn n is the AND function, and Thn/2 n is MAJORITY. For a given threshold function, what is the minimum size of a formula computing that threshold function? We deal with the k = 2 case. Here is a naive upper bound: Proposition 5 size(Thn 2 ) ≤ O(n 2 ). Proof Consider the formula which is an AND of (xi ∨xj ) over all i 6= j. This has size n 2 +n+ 1 = O(n 2 ) as desired. We can do slightly better with a recursive construction: Proposition 6 size(Thn 2 ) ≤ O(n log n). Proof Input z ∈ {0, 1} n is concatenation x ◦ y for x, y ∈ {0, 1} n/2 , so Thn 2 (x) = (Thn/2 2 (x) ∧ Thn/2 2 (y)) ∧ (Thn/2 1 (x) ∨ Thn/2 1 (y)). Then the size bound Sn we want is bounded inductively by Sn ≤ 2Sn/2 + O(n), so this gives Sn = O(n log n), specifically Sn = 2ndlog ne − 1. The main theorem that we prove today is the following lower bound that almost matches this upper bound for monotone circuits (monsize is the corresponding formula complexity for monotone formulae): Theorem 7 (K ’64, NRW ’90) monsize(Thn 2 ) ≥ 2dn log ne − 1

2. Step function vs sigmoid function

This article is about a piecewise constant function. For the unit step function, see [Heaviside step function](https://en.wikipedia.org/wiki/Heaviside_step_function).

In mathematics, a [function](https://en.wikipedia.org/wiki/Function_(mathematics)) on the [real numbers](https://en.wikipedia.org/wiki/Real_number) is called a step function if it can be written as a [finite](https://en.wikipedia.org/wiki/Finite_set) [linear combination](https://en.wikipedia.org/wiki/Linear_combination) of [indicator functions](https://en.wikipedia.org/wiki/Indicator_function) of [intervals](https://en.wikipedia.org/wiki/Interval_(mathematics)). Informally speaking, a step function is a [piecewise](https://en.wikipedia.org/wiki/Piecewise) [constant function](https://en.wikipedia.org/wiki/Constant_function) having only finitely many pieces.

Contents

[1Definition and first consequences](https://en.wikipedia.org/wiki/Step_function#Definition_and_first_consequences)

[1.1Variations in the definition](https://en.wikipedia.org/wiki/Step_function#Variations_in_the_definition)

[2Examples](https://en.wikipedia.org/wiki/Step_function#Examples)

[2.1Non-examples](https://en.wikipedia.org/wiki/Step_function#Non-examples)

[3Properties](https://en.wikipedia.org/wiki/Step_function#Properties)

[4See also](https://en.wikipedia.org/wiki/Step_function#See_also)

[5References](https://en.wikipedia.org/wiki/Step_function#References)

Definition and first consequences[[edit](https://en.wikipedia.org/w/index.php?title=Step_function&action=edit&section=1)]

A function {\displaystyle f\colon \mathbb {R} \rightarrow \mathbb {R} } is called a step function if it can be written as[[citation needed](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

{\displaystyle f(x)=\sum \limits \_{i=0}^{n}\alpha \_{i}\chi \_{A\_{i}}(x)}, for all real numbers {\displaystyle x}

where {\displaystyle n\geq 0}, {\displaystyle \alpha \_{i}} are real numbers, {\displaystyle A\_{i}} are intervals, and {\displaystyle \chi \_{A}} is the [indicator function](https://en.wikipedia.org/wiki/Indicator_function) of {\displaystyle A}:

{\displaystyle \chi \_{A}(x)={\begin{cases}1&{\text{if }}x\in A\\0&{\text{if }}x\notin A\\\end{cases}}}

In this definition, the intervals {\displaystyle A\_{i}} can be assumed to have the following two properties:

The intervals are [pairwise disjoint](https://en.wikipedia.org/wiki/Disjoint_set): {\displaystyle A\_{i}\cap A\_{j}=\emptyset } for {\displaystyle i\neq j}

The [union](https://en.wikipedia.org/wiki/Union_(set_theory)) of the intervals is the entire real line: {\displaystyle \bigcup \_{i=0}^{n}A\_{i}=\mathbb {R} .}

Indeed, if that is not the case to start with, a different set of intervals can be picked for which these assumptions hold. For example, the step function

{\displaystyle f=4\chi \_{[-5,1)}+3\chi \_{(0,6)}}

can be written as

{\displaystyle f=0\chi \_{(-\infty ,-5)}+4\chi \_{[-5,0]}+7\chi \_{(0,1)}+3\chi \_{[1,6)}+0\chi \_{[6,\infty )}.}

Variations in the definition[[edit](https://en.wikipedia.org/w/index.php?title=Step_function&action=edit&section=2)]

Sometimes, the intervals are required to be right-open[[1]](https://en.wikipedia.org/wiki/Step_function#cite_note-1) or allowed to be singleton.[[2]](https://en.wikipedia.org/wiki/Step_function#cite_note-2) The condition that the collection of intervals must be finite is often dropped, especially in school mathematics,[[3]](https://en.wikipedia.org/wiki/Step_function#cite_note-3)[[4]](https://en.wikipedia.org/wiki/Step_function#cite_note-4)[[5]](https://en.wikipedia.org/wiki/Step_function#cite_note-5) though it must still be [locally finite](https://en.wikipedia.org/wiki/Locally_finite_collection), resulting in the definition of piecewise constant functions.

A sigmoid function is a [mathematical function](https://en.wikipedia.org/wiki/Mathematical_function) having a characteristic "S"-shaped curve or sigmoid curve.

A common example of a sigmoid function is the [logistic function](https://en.wikipedia.org/wiki/Logistic_function) shown in the first figure and defined by the formula:[[1]](https://en.wikipedia.org/wiki/Sigmoid_function#cite_note-Han-Morag_1995-1)

{\displaystyle S(x)={\frac {1}{1+e^{-x}}}={\frac {e^{x}}{e^{x}+1}}=1-S(-x).}

Other standard sigmoid functions are given in the [Examples section](https://en.wikipedia.org/wiki/Sigmoid_function#Examples). In some fields, most notably in the context of [artificial neural networks](https://en.wikipedia.org/wiki/Artificial_neural_network), the term "sigmoid function" is used as an alias for the logistic function.

Special cases of the sigmoid function include the [Gompertz curve](https://en.wikipedia.org/wiki/Gompertz_curve" \o "Gompertz curve) (used in modeling systems that saturate at large values of x) and the [ogee curve](https://en.wikipedia.org/wiki/Ogee_curve) (used in the [spillway](https://en.wikipedia.org/wiki/Spillway) of some [dams](https://en.wikipedia.org/wiki/Dam)). Sigmoid functions have domain of all [real numbers](https://en.wikipedia.org/wiki/Real_number), with return (response) value commonly [monotonically increasing](https://en.wikipedia.org/wiki/Monotonically_increasing) but could be decreasing. Sigmoid functions most often show a return value (y axis) in the range 0 to 1. Another commonly used range is from −1 to 1.

A wide variety of sigmoid functions including the logistic and [hyperbolic tangent](https://en.wikipedia.org/wiki/Hyperbolic_tangent) functions have been used as the [activation function](https://en.wikipedia.org/wiki/Activation_function) of [artificial neurons](https://en.wikipedia.org/wiki/Artificial_neuron). Sigmoid curves are also common in statistics as [cumulative distribution functions](https://en.wikipedia.org/wiki/Cumulative_distribution_function) (which go from 0 to 1), such as the integrals of the [logistic density](https://en.wikipedia.org/wiki/Logistic_density), the [normal density](https://en.wikipedia.org/wiki/Normal_density), and [Student's t probability density functions](https://en.wikipedia.org/wiki/Student%27s_t-distribution). The logistic sigmoid function is invertible, and its inverse is the [logit](https://en.wikipedia.org/wiki/Logit) function.

3. Single layer vs multi-layer perceptron

A Multi Layer Perceptron (MLP) contains one or more hidden layers (apart from one input and one output layer). While a single layer perceptron can only learn linear functions, a multi layer perceptron can also learn non - linear functions. Figure 4 shows a multi layer perceptron with a single hidden layer

For understanding single layer perceptron, it is important to understand Artificial Neural Networks (ANN). Artificial neural networks is the information processing system the mechanism of which is inspired with the functionality of biological neural circuits. An artificial neural network possesses many processing units connected to each other. Following is the schematic representation of artificial neural network −

The diagram shows that the hidden units communicate with the external layer. While the input and output units communicate only through the hidden layer of the network.

The pattern of connection with nodes, the total number of layers and level of nodes between inputs and outputs with the number of neurons per layer define the architecture of a neural network.

There are two types of architecture. These types focus on the functionality artificial neural networks as follows −

Single Layer Perceptron

Multi-Layer Perceptron

Single Layer Perceptron

Single layer perceptron is the first proposed neural model created. The content of the local memory of the neuron consists of a vector of weights. The computation of a single layer perceptron is performed over the calculation of sum of the input vector each with the value multiplied by corresponding element of vector of the weights. The value which is displayed in the output will be the input of an activation function.

Let us focus on the implementation of single layer perceptron for an image classification problem using TensorFlow. The best example to illustrate the single layer perceptron is through representation of “Logistic Regression”.

Multi-layer Perceptron

Multi-layer perception is also known as MLP. It is fully connected dense layers, which transform any input dimension to the desired dimension. A multi-layer perception is a neural network that has multiple layers. To create a neural network we combine neurons together so that the outputs of some neurons are inputs of other neurons.

A gentle introduction to neural networks and TensorFlow can be found here:

[Neural Networks](https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/)

[Introduction to TensorFlow](https://www.geeksforgeeks.org/introduction-to-tensorflow/)

A multi-layer perceptron has one input layer and for each input, there is one neuron(or node), it has one output layer with a single node for each output and it can have any number of hidden layers and each hidden layer can have any number of nodes. A schematic diagram of a Multi-Layer Perceptron (MLP) is depicted below