**UNIT - 5**

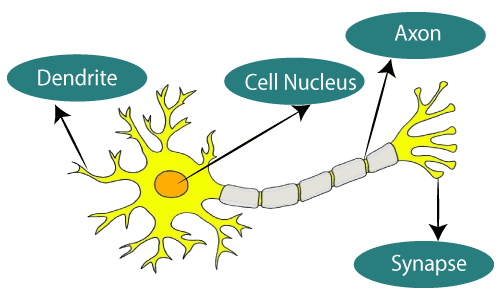
**Neural Network and Deep Learning**

**Artificial Neural Networks:** Introduction Artificial Neural Networks: Introduction, Neural Network representation, Appropriate problems, Perceptrons, Back propagation algorithm.

**Deep Learning**- Introduction, Deep Learning Architectures.

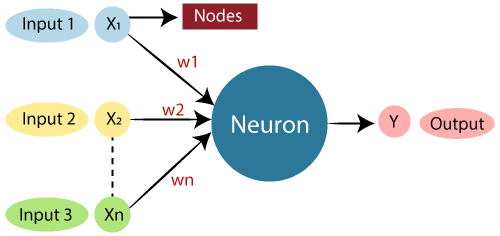
**Introduction**

The term "**Artificial Neural Network**" is derived from Biological neural networks that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes



The given figure illustrates the typical diagram of Biological Neural Network. Pi 3333

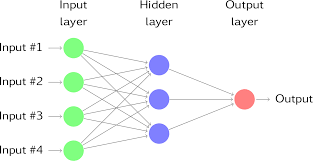
The typical Artificial Neural Network looks something like the given figure.



Dendrites from Biological Neural Network represent inputs in Artificial Neural Networks, cell nucleus represents Nodes, synapse represents Weights, and Axon represents Output.

## Architecture of ANN

Artificial Neural Network primarily consists of three layers:



A neural network consists of three layers. The first layer is the input layer. It contains the input neurons that send information to the hidden layer. The hidden layer performs the computations on input data and transfers the output to the output layer. It includes weight, activation function, cost function.

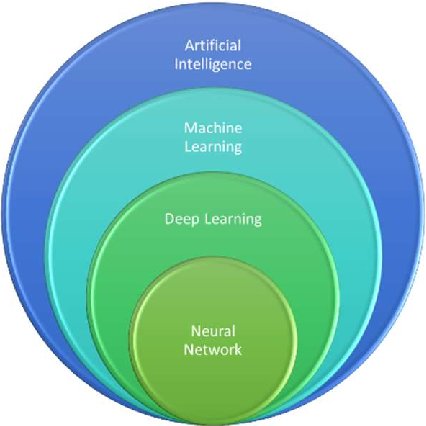
The connection between neurons is known as weight, which is the numerical values. The weight between neurons determines the learning ability of the neural network. During the learning of artificial neural networks, weight between the neuron changes.

## Working of ANN

Firstly, the information is feed into the input layer. Which then transfers it to the hidden layers, and interconnection between these two layers assign weights to each input randomly at the initial point. Then bias is add to each input neuron and after this, the weight sum which is a combination of weights and bias is pass through the activation function. Activation Function has the responsibility of which node to fire for feature extraction and finally output is calculate. Therefore this whole process is known as Forward Propagation. After getting the output model to compare it with the original output and the error is known and finally, weights are updates in backward propagation to reduce the error and this process continues for a certain number of epochs (iteration). Finally, model weights get updates and prediction is done.

## Relation between AI & ANN

In simple terms, machine learning is a subfield of artificial intelligence. Neural networks are a subfield of machine learning. And deep learning algorithms are an advancement on the concept of neural networks.



**Appropriate Problems for ANN**

* training data is noisy, complex sensor data
* also problems where symbolic algos are used (decision tree learning (DTL)) - ANN and DTL produce results of comparable accuracy
* instances are attribute-value pairs, attributes may be highly correlated or independent, values can be any real value
* target function may be discrete-valued, real-valued or a vector
* training examples may contain errors
* long training times are acceptable
* requires fast eval. of learned target func.
* humans do NOT need to understand the learned target func.

## Advantages of Artificial Neural Network

The advantages of the neural network are as follows −

* A neural network can implement tasks that a linear program cannot.
* When an item of the neural network declines, it can continue without some issues by its parallel features.
* A neural network determines and does not require to be reprogrammed.
* It can be executed in any application.

## Disadvantages of Artificial Neural Network

The disadvantages of the neural network are as follows −

* The neural network required training to operate.
* The structure of a neural network is disparate from the structure of microprocessors therefore required to be emulated.
* It needed high processing time for big neural networks.

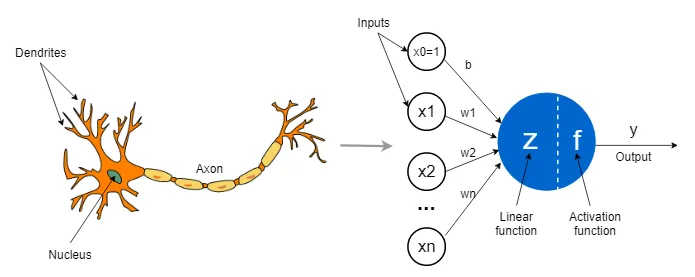
**Artificial Neural Network Applications**

Following are some important ANN Applications –

1. **Speech Recognition:**Speech recognition relies heavily on artificial neural networks (ANNs). Earlier speech recognition models used statistical models such as Hidden Markov Models. With the introduction of deep learning, several forms of neural networks have become the only way to acquire a precise classification.
2. **Handwritten Character Recognition:**ANNs are used to recognize handwritten characters. Handwritten characters can be in the form of letters or digits, and neural networks have been trained to recognize them.
3. **Signature Classification:**We employ artificial neural networks to recognize signatures and categorize them according to the person’s class when developing these authentication systems. Furthermore, neural networks can determine whether or not a signature is genuine.
4. **Medical:**It can be used to detect cancer cells and analyze MRI pictures in order to provide detailed results.

**\*\*\*Perceptron\*\*\***

**Introduction:**

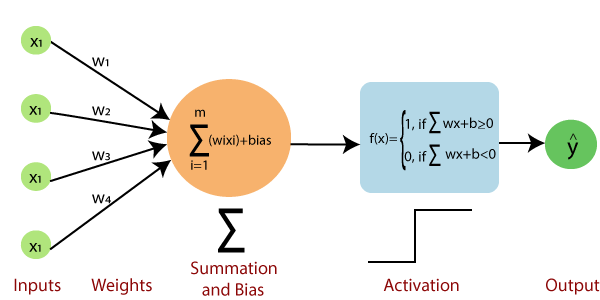


* A biological neuron receives its input signals from other neurons through dendrites (small fibers). Likewise, a perceptron receives its data from other perceptron’s through input neurons that take numbers.
* The connection points between dendrites and biological neurons are called synapses. Likewise, the connections between inputs and perceptrons are called weights. They measure the importance level of each input.
* In a biological neuron, the nucleus produces an output signal based on the signals provided by dendrites. Likewise, the nucleus (colored in blue) in a perceptron performs some calculations based on the input values and produces an output.
* In a biological neuron, the output signal is carried away by the axon. Likewise, the axon in a perceptron is the output value which will be the input for the next perceptrons.

**What is Perceptron?**

Perceptron is Machine Learning algorithm for supervised learning of various binary classification tasks. Further, Perceptron is also understood as an Artificial Neuron or neural network unit that helps to detect certain input data computations in business intelligence.

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.



* ***Input Nodes or Input Layer:***

This is the primary component of Perceptron which accepts the initial data into the system for further processing. Each input node contains a real numerical value.

* ***Wight and Bias:***

Weight parameter represents the strength of the connection between units. This is another most important parameter of Perceptron components. Weight is directly proportional to the strength of the associated input neuron in deciding the output. Further, Bias can be considered as the line of intercept in a linear equation.

* ***Activation Function:***

These are the final and important components that help to determine whether the neuron will fire or not. Activation Function can be considered primarily as a step function.This step function or Activation function plays a vital role in ensuring that output is mapped between required values (0,1) or (-1,1). It is important to note that the weight of input is indicative of the strength of a node. Similarly, an input's bias value gives the ability to shift the activation function curve up or down.

**Why we need Perceptron?**

Frank Rosenblatt (1928 – 1971) was an American psychologist notable in the field of Artificial Intelligence. In 1957 he started something really big. He "invented" a Perceptron program, on an IBM 704 computer at Cornell Aeronautical Laboratory. Scientists had discovered that brain cells (Neurons) receive input from our senses by electrical signals. The Neurons, then again, use electrical signals to store information, and to make decisions based on previous input.Frank had the idea that Perceptrons could simulate brain principles, with the ability to learn and make decisions.

**Definition:**

A neural network link that contains computations to track features and uses Artificial Intelligence in the input data is known as Perceptron. Perceptron is a supervised learning of Binary classifiers. This Algorithm enables neurons to learn and Process Elements in the training set one at a time. Based on the layers, Perceptron models are divided into two types.

**These are as follows:**

1. Single-layer Perceptron Model
2. Multi-layer Perceptron model

* ***Single-layer Perceptron Model:***
* The main objective of the single-layer perceptron model is to analyze the linearly separable objects with binary outcomes.
* If the outcome is same as pre-determined or threshold value, then the performance of this model is stated as satisfied, and weight demand does not change.
* However, this model consists of a few discrepancies triggered when multiple weight inputs values are fed into the model.
* Hence, to find desired output and minimize errors, some changes should be necessary for the weights input.
* "Single-layer perceptron can learn only linearly separable patterns."
* ***Multi-layer Perceptron Model:***
* Like a single-layer perceptron model, a multi-layer perceptron model also has the same model structure but has a greater number of hidden layers.
* The multi-layer perceptron model is also known as the Backpropagation algorithm, which executes in two stages as follows:
* Forward Stage: Activation functions start from the input layer in the forward stage and terminate on the output layer.
* Backward Stage: In the backward stage, weight and bias values are modified as per the model's requirement. In this stage, the error between actual output and demanded originated backward on the output layer and ended on the input layer.

**Advantages of Multi-Layer Perceptron:**

* A multi-layered perceptron model can be used to solve complex non-linear problems.
* It works well with both small and large input data.
* It helps us to obtain quick predictions after the training.
* It helps to obtain the same accuracy ratio with large as well as small data.

**Disadvantages of Multi-Layer Perceptron:**

* + - * + In Multi-layer perceptron, computations are difficult and time-consuming.
* In multi-layer Perceptron, it is difficult to predict how much the dependent variable affects each independent variable.
* The model functioning depends on the quality of the training.

**Algorithm:**

* **Step-1**

In the first step first, multiply all input values with corresponding weight values and then add them to determine the weighted sum. Mathematically, we can calculate the weighted sum as follows:

∑wi\*xi = x1\*w1 + x2\*w2 +…wn\*xn

Add a special term called bias 'b' to this weighted sum to improve the model's performance.

∑wi\*xi + b

* **Step-2**

In the second step, an activation function is applied with the above-mentioned weighted sum, which gives us output either in binary form or a continuous value as follows:

Y = f(∑wi\*xi + b)

* **Step-3**

Check whether the calculated output is equal to the target output .if not we need to update the weights and bias values else no need of updation.

If(y!=t),then

Wi(new)=wi(old)+ αtxi

b(new)=b(old)+ αt

else

Wi(new)=wi(old)

b(new)=b(old)

Note: do not update values

**Example:**

**Problem statement:-** ***Implement AND function on bipolar inputs using Perceptron***

**Solution:**

Bipolar inputs are (1,-1).

**AND function:** If the both inputs are high then only the result will be high. If any one of the input is low then the result will be low.

***Note: Here 1 indicates high input and -1 indicates low input values respectively.***

* **Step 1:**

|  |  |  |
| --- | --- | --- |
| X1 | X2 | Target\_output |
| 1 | 1 | 1 |
| 1 | -1 | -1 |
| -1 | 1 | -1 |
| -1 | -1 | -1 |

Initilize weigths and bais as '0' (For easy calculation). also initilize the learning rate α (Alpha) as 1 which is constant in this algorithm due to here we are processing only one target output at a time.

W1=0 , W2=0 , b(bias)=0 and α =1;

For Test case 1:

|  |  |  |
| --- | --- | --- |
| X1 | X2 | Target\_output(t) |
| 1 | 1 | 1 |

Target output(t)=1;

* **Step 2:**

Calculate output of network using summation and bias formula.

Net input(yin)=∑ wi\*xi + b

here i runs to 0 to n and n indicates the number of input neurons.

Y in=w1x1+w2x2+b

=0(1)+0(1)+0

=0.

* **Step 3:**

After calculating net input apply Activation function.

1. Yin>0

Y= f(Yin)= 0 Yin=0

-1 Yin<0

Y=f(Yin)=f(0)=0

Which is not equal to the target output 1. (0!=1)

So, we need to update the values of weights and bias.

Wi(new)=wi(old)+ αtxi

W1(new)=w1(old)+ αtx1

W1\*=0+(1\*1\*1) (\* is to indicate new)

=0+1

W1\*=1

W2(new)=w2(old)+ αtx2

W2\*=0+(1\*1\*1)

=0+1

W2\*=1

b(new)=b(old)+ αt

b\*=0+(1\*1)

=0+1

b\*=1

w1= αtx1= (1\*1\*1) =1 ( :is to represent delta of w1)

w2= αtx2 = (1\*1\*1) =1

b = αt = (1\*1) =1

now calculate output for second testcase using updated weights and bias values;

W1=1 , W2=1 , b(bias)=1 and α =1;

For Test case 2:

|  |  |  |
| --- | --- | --- |
| X1 | X2 | Target\_output(t) |
| 1 | -1 | -1 |

Target output(t)=-1;

Y in=w1x1+w2x2+b

|  |
| --- |
| 1. Yin>0   Y= f(Yin)= 0 Yin=0  -1 Yin<0 |

=1(1)+1(-1)+1

=1-1+1

=1

Y=f(Yin)=f(1)=1

Which is not equal to

the target output -1. (1 != -1)

So, now Again we need to update the values of weights and bias.

Wi(new)=wi(old)+ αtxi

W1(new)=w1(old)+ αtx1

W1\*=1+(1\*-1\*1)

=1-1

W1\*=0

W2(new)=w2(old)+ αtx2

W2\*=1+(1\*-1\*-1)

=1+1

W2\*=2

b(new)=b(old)+ αt

b\*=1+(1\*-1)

=1-1

b\*=0

w1= αtx1= (1\*-1\*1) =-1

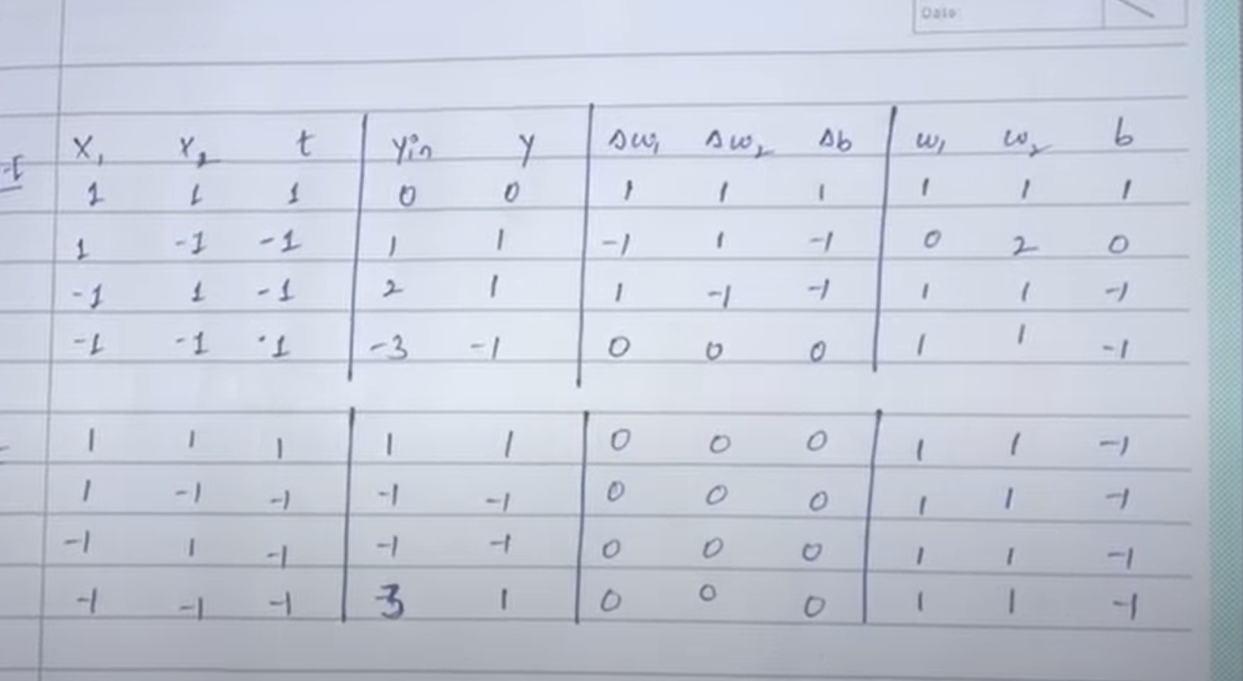
w2= αtx2 = (1\*-1\*-1) =1

b = αt = (1\*-1) =-1

Repeat the same process to calculate output for third and fourth testcase using updated weights and bias values;

***Note: No need of Updating weights and bias values if y=t which means the calculated output after applying activation function f(yin) is equal to the target output.***

Now do the same calculation for all 4\_testcases in 2nd Iteration.

**You can check your answers using this below table!** 

Characteristics of Perceptron

The perceptron model has the following characteristics.

Perceptron is a machine learning algorithm for supervised learning of binary classifiers.

In Perceptron, the weight coefficient is automatically learned.

Initially, weights are multiplied with input features, and the decision is made whether the neuron is fired or not.

The activation function applies a step rule to check whether the weight function is greater than zero.

The linear decision boundary is drawn, enabling the distinction between the two linearly separable classes +1 and -1.

If the added sum of all input values is more than the threshold value, it must have an output signal; otherwise, no output will be shown.

Limitations of Perceptron Model

A perceptron model has limitations as follows:

The output of a perceptron can only be a binary number (0 or 1) due to the hard limit transfer function.

Perceptron can only be used to classify the linearly separable sets of input vectors. If input vectors are non-linear, it is not easy to classify them properly.

**Backpropagation:**

Backpropagation is an algorithm that backpropagates the errors from the output nodes to the input nodes. Therefore, it is simply referred to as the backward propagation of errors.

* Backpropagation is a widely used algorithm for training feedforward neural networks.
* It computes the gradient of the loss function with respect to the network weights.

The backpropagation algorithm works by computing the gradient of the loss function with respect to each weight via the chain rule, computing the gradient layer by layer, and iterating backward from the last layer to avoid redundant computation of intermediate terms in the chain rule.

**Features of Backpropagation:**

1. it is the [gradient descent](https://www.geeksforgeeks.org/gradient-descent-algorithm-and-its-variants/) method as used in the case of simple perceptron network with the differentiable unit.
2. it is different from other networks in respect to the process by which the weights are calculated during the learning period of the network.
3. training is done in the three stages :
   * + the [feed-forward](https://www.geeksforgeeks.org/multilayer-feed-forward-neural-network-in-data-mining/) of input training pattern
     + the calculation and backpropagation of the error
     + updation of the weight

**Types of Backpropagation**

There are two types of backpropagation networks.

* **Static backpropagation:**Static backpropagation is a network designed to map static inputs for static outputs. These types of networks are capable of solving static classification problems such as OCR (Optical Character Recognition).
* **Recurrent backpropagation:** Recursive backpropagation is another network used for fixed-point learning. Activation in recurrent backpropagation is feed-forward until a fixed value is reached. Static backpropagation provides an instant mapping, while recurrent backpropagation does not provide an instant mapping.

### 

### Architecture:

### 

### Working of Backpropagation:

Neural networks use supervised learning to generate output vectors from input vectors that the network operates on. It Compares generated output to the desired output and generates an error report if the result does not match the generated output vector. Then it adjusts the weights according to the bug report to get your desired output.

**Backpropagation Algorithm:**

**Step 1:** Inputs X, arrive through the preconnected path.

**Step 2:** The input is modeled using true weights W. Weights are usually chosen randomly.

**Step 3:**Calculate the output of each neuron from the input layer to the hidden layer to the output layer.

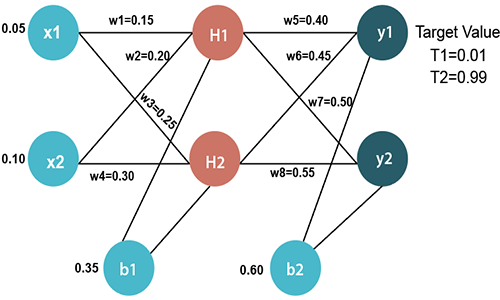
**Step 4:** Calculate the error in the outputs

Backpropagation Error= Actual Output – Desired Output

**Step 5:** From the output layer, go back to the hidden layer to adjust the weights to reduce the error.

**Step 6:** Repeat the process until the desired output is achieved.

**Example:**

****

### Input values:

X1=0.05  
X2=0.10

### Initial weight:

### W1=0.15 w5=0.40 W2=0.20     w6=0.45 W3=0.2 w7=0.50 W4=0.30     w8=0.55

### Bias Values:

b1=0.35     b2=0.60

### Target Values:

T1=0.01  
T2=0.99

we first calculate the values of H1 and H2 by a forward pass.

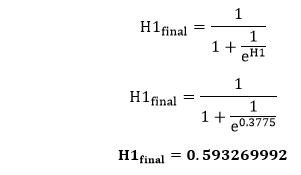
### Forward Pass

To find the value of H1 we first multiply the input value from the weights as

H1=x1×w1+x2×w2+b1

H1=0.05×0.15+0.10×0.20+0.35  
 **H1=0.3775**

To calculate the final result of H1, we performed the sigmoid function as

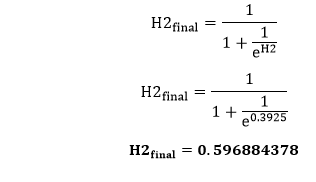


We will calculate the value of H2 in the same way as H1

H2=x1×w3+x2×w4+b1  
H2=0.05×0.25+0.10×0.30+0.35

**H2=0.3925**

To calculate the final result of H1, we performed the sigmoid function as

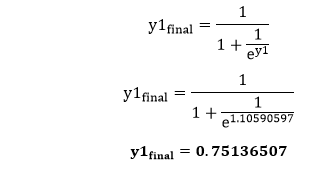


Now, we calculate the values of y1 and y2 in the same way as we calculate the H1 and H2.

 y1=H1×w5+H2×w6+b2

y1=0.593269992×0.40+0.596884378×0.45+0.60  
**y1=1.10590597**

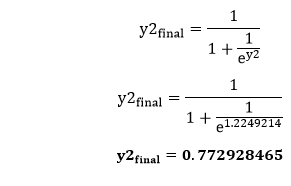
To calculate the final result of y1 we performed the sigmoid function as



We will calculate the value of y2 in the same way as y1

 y2=H1×w7+H2×w8+b2                              
 y2=0.593269992×0.50+0.596884378×0.55+0.60  
  **y2=1.2249214**

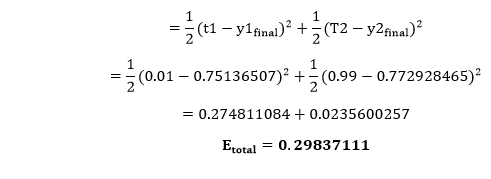
To calculate the final result of y1, we performed the sigmoid function as

  
Our target values are 0.01 and 0.99. Our y1 and y2 value is not matched with our target values T1 and T2.

Now, we will find the **total error**, which is simply the difference between the outputs from the target outputs. The total error is calculated as

Backpropagation Process in Deep Neural Network

So, the total error is

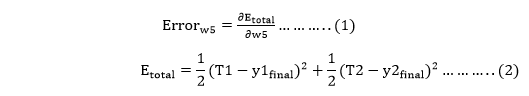


Now, we will backpropagate this error to update the weights using a backward pass.

To update the weight, we calculate the error correspond to each weight with the help of a total error. The error on weight w is calculated by differentiating total error with respect to w.

Backpropagation Process in Deep Neural Network

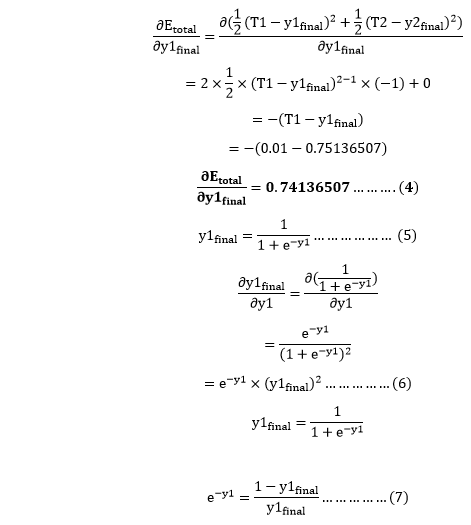
We perform backward process so first consider the last weight w5 as



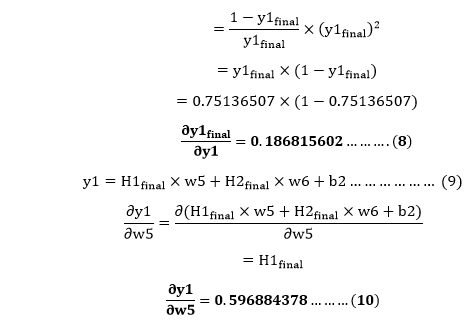
From equation two, it is clear that we cannot partially differentiate it with respect to w5 because there is no any w5. We split equation one into multiple terms so that we can easily differentiate it with respect to w5 as

Backpropagation Process in Deep Neural Network

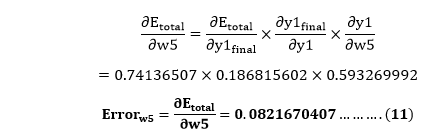
Now, we calculate each term one by one to differentiate Etotal with respect to w5 as



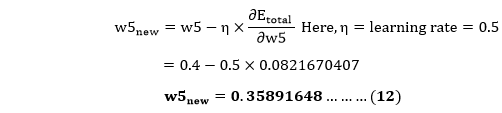
Putting the value of e-y in equation (5)



So, we put the values of Backpropagation Process in Deep Neural Network in equation no (3) to find the final result.



Now, we will calculate the updated weight w5new with the help of the following formula



In the same way, we calculate w6new,w7new, and w8new and this will give us the following values

**w5new=0.35891648**

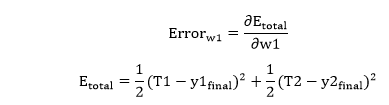
**w6new=408666186**

**w7new=0.511301270**  
**w8new=0.561370121**

Backward pass at Hidden layer

Now, we will backpropagate to our hidden layer and update the weight w1, w2, w3, and w4 as we have done with w5, w6, w7, and w8 weights.

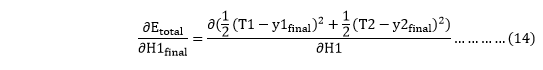
We will calculate the error at w1 as



From equation (2), it is clear that we cannot partially differentiate it with respect to w1 because there is no any w1. We split equation (1) into multiple terms so that we can easily differentiate it with respect to w1 as

Backpropagation Process in Deep Neural Network

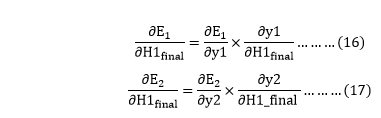
Now, we calculate each term one by one to differentiate Etotal with respect to w1 as



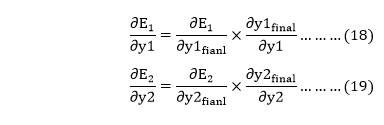
We again split this because there is no any H1final term in Etoatal as

Backpropagation Process in Deep Neural Network

Backpropagation Process in Deep Neural Network will again split because in E1 and E2 there is no H1 term. Splitting is done as

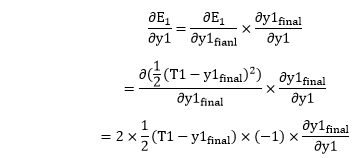


We again Split bothBackpropagation Process in Deep Neural Network because there is no any y1 and y2 term in E1 and E2. We split it as

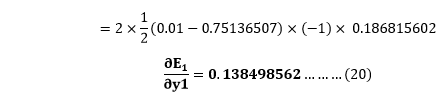


Now, we find the value of Backpropagation Process in Deep Neural Network by putting values in equation (18) and (19) as

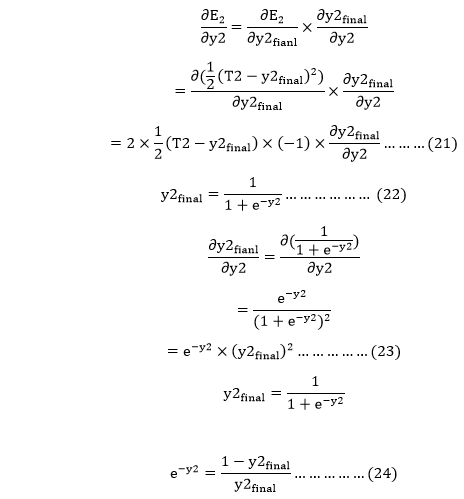
From equation (18)



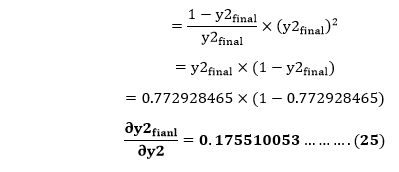
From equation (8)



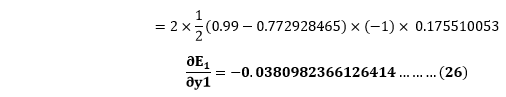
From equation (19)



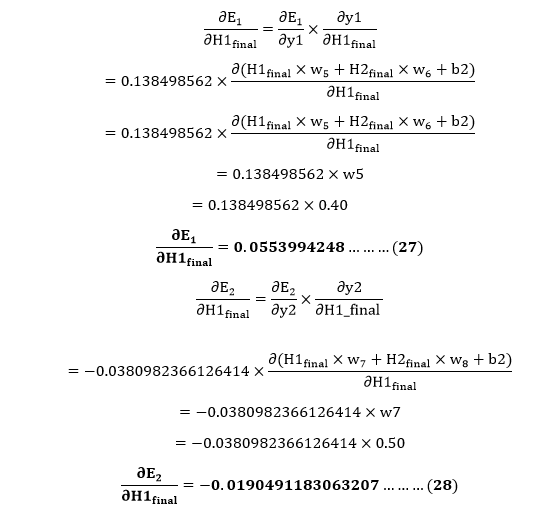
Putting the value of e-y2 in equation (23)



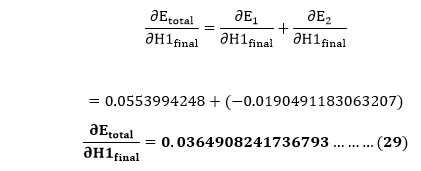
From equation (21)



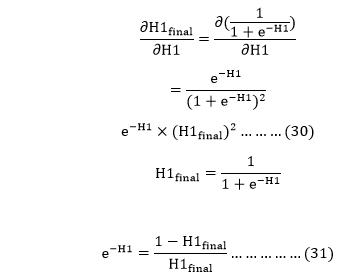
Now from equation (16) and (17)



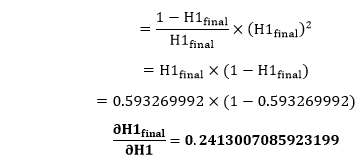
Put the value of Backpropagation Process in Deep Neural Network in equation (15) as



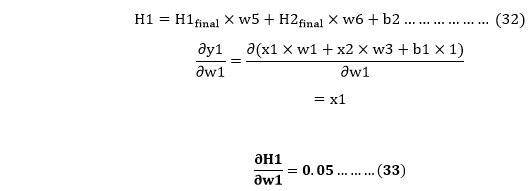
We haveBackpropagation Process in Deep Neural Networkwe need to figure outBackpropagation Process in Deep Neural Networkas



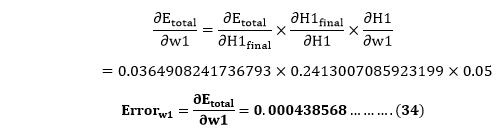
Putting the value of e-H1 in equation (30)



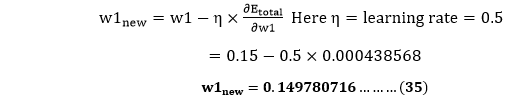
We calculate the partial derivative of the total net input to H1 with respect to w1 the same as we did for the output neuron:



So, we put the values of Backpropagation Process in Deep Neural Network in equation (13) to find the final result.



Now, we will calculate the updated weight w1new with the help of the following formula



In the same way, we calculate w2new,w3new, and w4 and this will give us the following values

**w1new=0.149780716**                        
 **w2new=0.19956143**                         
 **w3new=0.24975114**                         
 **w4new=0.29950229**

We have updated all the weights. We found the error 0.298371109 on the network when we fed forward the 0.05 and 0.1 inputs. In the first round of Backpropagation, the total error is down to 0.291027924. After repeating this process 10,000, the total error is down to 0.0000351085. At this point, the outputs neurons generate 0.159121960 and 0.984065734 i.e., nearby our target value when we feed forward the 0.05 and 0.1.

**Advantages:**

* It is simple, fast, and easy to program.
* Only numbers of the input are tuned, not any other parameter.
* It is Flexible and efficient.
* No need for users to learn any special functions.

**Disadvantages:**

* It is sensitive to noisy data and irregularities. Noisy data can lead to inaccurate results.
* Performance is highly dependent on input data.
* Spending too much time training.
* The matrix-based approach is preferred over a mini-batch.

### Applications of Backpropagation:

The applications are

* The neural network is trained to enunciate each letter of a word and a sentence
* It is used in the field of [speech recognition](https://www.elprocus.com/voice-recognition-modules-working-procedure-applications/)
* It is used in the field of character and face recognition

Deep Learning.

## What is Deep Learning?

Deep learning can be considered as a subset of [machine learning](https://www.simplilearn.com/tutorials/machine-learning-tutorial/what-is-machine-learning). It is a field that is based on learning and improving on its own by examining computer algorithms. While machine learning uses simpler concepts, deep learning works with artificial neural networks, which are designed to imitate how humans think and learn. Until recently, [neural networks](https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-neural-network) were limited by computing power and thus were limited in complexity. However, advancements in [Big Data analytics](https://www.simplilearn.com/what-is-big-data-analytics-article) have permitted larger, sophisticated neural networks, allowing computers to observe, learn, and react to complex situations faster than humans.

## **Example of Deep Learning at Work**

Let’s say the goal is to have a neural network recognize photos that contain a dog. All dogs don’t look exactly alike – consider a Rottweiler and a Poodle, for instance. Furthermore, photos show dogs at different angles and with varying amounts of light and shadow. So, a training set of images must be compiled, including many examples of dog faces which any person would label as “dog,” and pictures of objects that aren’t dogs, labeled (as one might expect), “not dog.” The images, fed into the neural network, are converted into data. These data move through the network, and various nodes assign weights to different elements. The final output layer compiles the seemingly disconnected information – furry, has a snout, has four legs, etc. – and delivers the output: dog.

Now, this answer received from the neural network will be compared to the human-generated label. If there is a match, then the output is confirmed. If not, the neural network notes the error and adjusts the weightings. The neural network tries to improve its dog-recognition skills by repeatedly adjusting its weights over and over again. This training technique is called supervised learning, which occurs even when the neural networks are not explicitly told what "makes" a dog. They must recognize patterns in data over time and learn on their own.

### Advantages of Deep Learning:

Deep learning has several advantages over traditional machine learning methods, some of the main ones include:

1. **Automatic feature learning:**Deep learning algorithms can automatically learn features from the data, which means that they don’t require the features to be hand-engineered. This is particularly useful for tasks where the features are difficult to define, such as image recognition.
2. **Handling large and complex data:** Deep learning algorithms can handle large and complex datasets that would be difficult for traditional machine learning algorithms to process. This makes it a useful tool for extracting insights from big data.
3. **Improved performance:**Deep learning algorithms have been shown to achieve state-of-the-art performance on a wide range of problems, including image and speech recognition, natural language processing, and computer vision.
4. **Handling non-linear relationships:**Deep learning can uncover non-linear relationships in data that would be difficult to detect through traditional methods.
5. **Handling structured and unstructured data:** Deep learning algorithms can handle both structured and unstructured data such as images, text, and audio.
6. **Predictive modeling:**Deep learning can be used to make predictions about future events or trends, which can help organizations plan for the future and make strategic decisions.

### Disadvantages of Deep Learning:

While deep learning has many advantages, there are also some disadvantages to consider:

1. **High computational cost:**Training deep learning models requires significant computational resources, including powerful GPUs and large amounts of memory. This can be costly and time-consuming.
2. **Overfitting:**Overfitting occurs when a model is trained too well on the training data and performs poorly on new, unseen data. This is a common problem in deep learning, especially with large neural networks, and can be caused by a lack of data, a complex model, or a lack of regularization.
3. **Lack of interpretability:** Deep learning models, especially those with many layers, can be complex and difficult to interpret. This can make it difficult to understand how the model is making predictions and to identify any errors or biases in the model.
4. **Dependence on data quality:** Deep learning algorithms rely on the quality of the data they are trained on. If the data is noisy, incomplete, or biased, the model’s performance will be negatively affected.
5. **Data privacy and security concerns:** As deep learning models often rely on large amounts of data, there are concerns about data privacy and security. Misuse of data by malicious actors can lead to serious consequences like identity theft, financial loss and invasion of privacy.
6. **Lack of domain expertise:** Deep learning requires a good understanding of the domain and the problem you are trying to solve. If the domain expertise is lacking, it can be difficult to formulate the problem and select the appropriate algorithm.

**Applications of Deep Learning**

### 1. Healthcare

The healthcare sector has long been one of the prominent adopters of modern technology to overhaul itself. As such, it is not surprising to see Deep Learning finding uses in interpreting medical data for

* the diagnosis, prognosis & treatment of diseases
* drug prescription
* analysing MRIs, CT scans, ECG, X-Rays, etc., to detect and notify about medical anomalies
* personalising treatment
* monitoring the health of patients and more

One notable application of deep learning is found in the [diagnosis and treatment of cancer](https://genomemedicine.biomedcentral.com/articles/10.1186/s13073-021-00968-x).

Medical professionals use a CNN or Convolutional Neural Network, a Deep learning method, to grade different types of cancer cells. They expose high-res histopathological images to deep CNN models after magnifying them 20X or 40X. The deep CNN models then demarcate various cellular features within the sample and detect carcinogenic elements.

### 2. Personalized Marketing

Personalized marketing is a concept that has seen much action in the recent few years. Marketers are now aiming their advertising campaigns at the pain points of individual consumers, offering them exactly what they need. And Deep Learning is playing a significant role in this.

Today, consumers are generating a lot of data thanks to their engagement with social media platforms, IoT devices, web browsers, wearables and the ilk. However, most of the data generated from these sources are disparate (text, audio, video, location data, etc.).

To cope with this, businesses use customisable Deep Learning models to interpret data from different sources and distil them to extract valuable customer insights. They then use this information to predict consumer behaviour and target their marketing efforts more efficiently.

So now you understand how those online shopping sites know what products to recommend to you.

### 3. Financial Fraud Detection

Virtually no sector is exempt from the evil called “fraudulent transactions” or “financial fraud”. However, it is the financial corporations (banks, insurance firms, etc.) that have to bear the brunt of this menace the most. Not a day goes by when criminals attack financial institutions. There are a plethora of ways to usurp financial resources from them.

Thus, for these organizations, detecting and predicting financial fraud is critical, to say the least. And this is where Deep Learning comes into the picture.

Financial organizations are now using the concept of anomaly detection to flag inappropriate transactions. They employ deep learning algorithms, such as logistic regression (credit card fraud detection is a prime use case), decision trees, random forest, etc., to analyze the patterns common to valid transactions. Then, these models are put into action to flag financial transactions that seem potentially fraudulent.

Some examples of fraud detection being deterred by Deep Learning include:

* identity theft
* insurance fraud
* investment fraud
* fund misappropriation

### 4. Natural Language Processing

NLP or Natural Language Processing is another prominent area where Deep Learning is showing promising results.

Natural Language Processing, as the name suggests, is all about enabling machines to analyze and understand human language. The premise sounds simple, right? Well, the thing is, human language is punishingly complex for machines to interpret. It is not just the alphabet and words but also the context, the accents, the handwriting and whatnot that discourage machines from processing or generating human language accurately.

Deep Learning-based NLP is doing away with many of the issues related to understanding human language by training machines ([Autoencoders](https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798) and [Distributed Representation](https://www.sciencedirect.com/topics/computer-science/distributed-representation)) to produce appropriate responses to linguistic inputs.

One such example is the personal assistants we use on our smartphones. These applications come embedded with Deep Learning imbued NLP models to understand human speech and return appropriate output. It is, thus, no wonder why Siri and Alexa sound so much like how people talk in real life.

Another use case of Deep Learning-based NLP is how websites written in one human language automatically get translated to the user-specified language.

### 5. Autonomous Vehicles

The concept of building automated or self-governing vehicles goes back 45 years when the Tsukuba Mechanical Engineering Laboratory unveiled the world’s first semi-automatic car. The car, a technological marvel then, carried a pair of cameras and an analogue computer to steer itself on a specially designed street.

However, it wasn’t until 1989 when ALVINN (Autonomous Land Vehicle in a Neural Network), a modified military ambulance, used neural networks to navigate by itself on roads.

Since then, deep learning and autonomous vehicles have enjoyed a strong bond, with the former enhancing the latter’s performance exponentially.

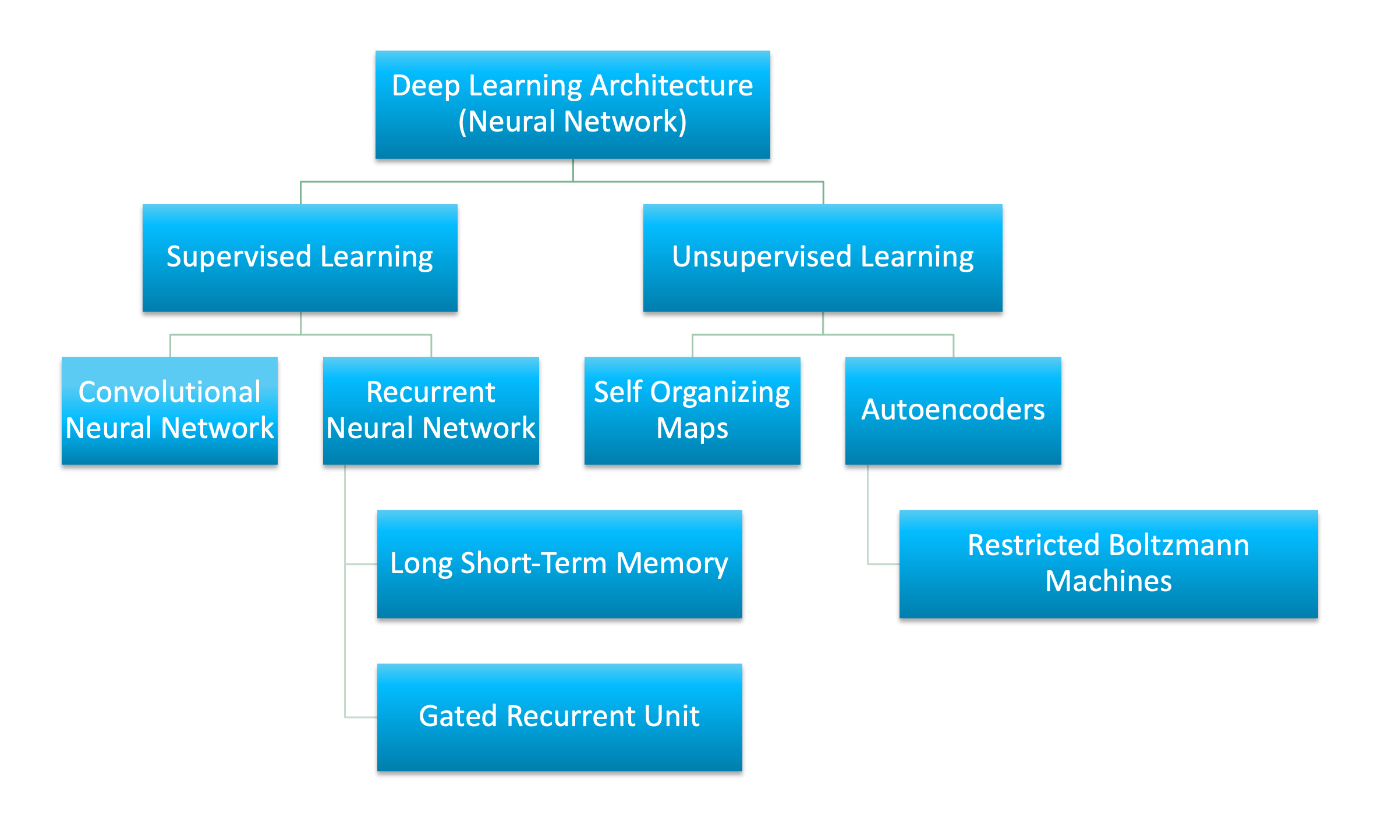
Autonomous vehicles use cameras, sensors – LiDARs, RADARs, motion sensors –  and external information such as geo-mapping to perceive their environment and collect relevant data. They use this equipment both individually and in tandem for documenting the data.

**DEEP LEARNING ARCHITECTURES**

Connectionist architectures have existed for more than 70 years, but new architectures and graphical processing units (GPUs) brought them to the forefront of artificial intelligence. Deep learning isn't a single approach but rather a class of algorithms and topologies that you can apply to a broad spectrum of problems.

While deep learning is certainly not new, it is experiencing explosive growth because of the intersection of deeply layered neural networks and the use of GPUs to accelerate their execution. Big data has also fed this growth. Because deep learning relies on training neural networks with example data and rewarding them based on their success, the more data, the better to build these deep learning structures.

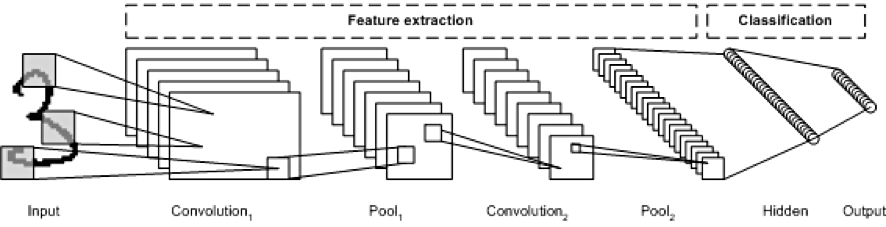
The number of architectures and algorithms that are used in deep learning is wide and varied. This section explores six of the deep learning architectures spanning the past 20 years. Notably, long short-term memory (LSTM) and convolutional neural networks (CNNs) are two of the oldest approaches in this list but also two of the most used in various applications.



Artificial neural network (ANN) is the underlying architecture behind deep learning. Based on ANN, several variations of the algorithms have been invented.

Convolutional neural networks

* The architecture is particularly useful in image-processing applications.
* It is used for pattern detection and it is what makes it useful for image processing.
* It also has multiple layers 1.input layer 2.convolutional layer 3.output layer \
* In each convolutional layer we should specify the number of filters a layer should have.
* Filters are what detect patterns



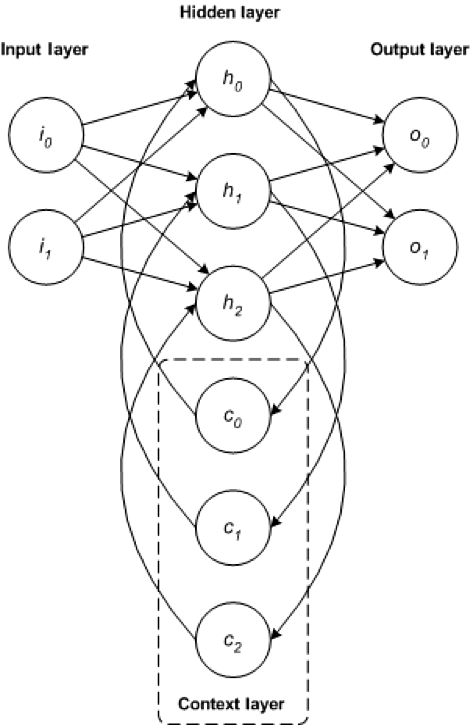
### Recurrent neural networks

The RNN is one of the foundational network architectures from which other deep learning architectures are built.

When we take feed forward neural network it only concentrates on current state Hence there is no memory,so they cannot handle sequential data.

In RNN we have memory so we can consider previous state also.

The primary difference between a typical multilayer network and a recurrent network is that rather than completely feed-forward connections, a recurrent network might have connections that feed back into prior layers (or into the same layer). This feedback allows RNNs to maintain memory of past inputs and model problems in time.



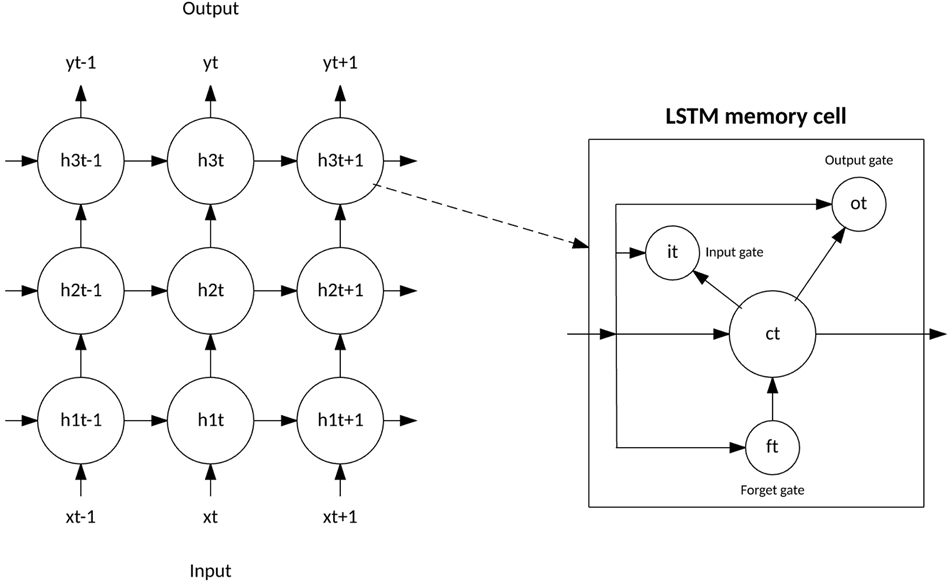
# Long Short-Term Memory Networks (LSTMs)

The LSTM departed from typical neuron-based neural network architectures and instead introduced the concept of a memory cell.

The memory cell can retain its value for a short or long time as a function of its inputs, which allows the cell to remember what's important and not just its last computed value.

The LSTM memory cell contains three gates that control how information flows into or out of the cell.

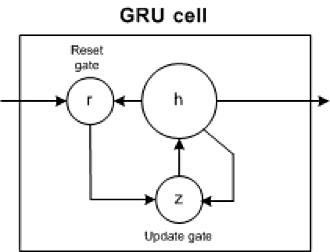
* The **input gate** controls when new information can flow into the memory.
* The **forget gate** controls when an existing piece of information is forgotten, allowing the cell to remember new data.
* The **output gate** controls when the information that is contained in the cell is used in the output from the cell.



# GRU (Gated Recurrent Units)

#### GRU is a modified light-weighted version of lstm

In GRU it combines both long and short term memory into its hidden state.



It has two gates:

1.Update Gate:How much of past memory to retain.

2.Reset Gate:How much of past memory to forget.

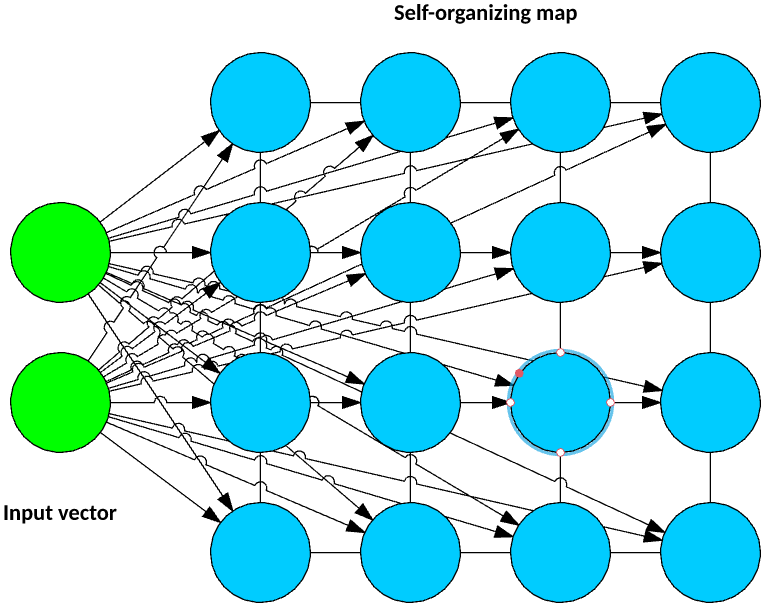
## Unsupervised deep learning

Unsupervised learning refers to the problem space wherein there is no target label within the data that is used for training.

This section discusses three unsupervised deep learning architectures: self-organized maps, autoencoders, and restricted boltzmann machines. We also discuss how deep belief networks and deep stacking networks are built based on the underlying unsupervised architecture.

### **Self-organized maps**

Self-organized map (SOM) was invented by Dr. Teuvo Kohonen in 1982 and was popularly known as the Kohonen map. SOM is an unsupervised neural network that creates clusters of the input data set by reducing the dimensionality of the input. SOMs vary from the traditional artificial neural network in quite a few ways.



The first significant variation is that weights serve as a characteristic of the node. After the inputs are normalized, a random input is first chosen. Random weights close to zero are initialized to each feature of the input record. These weights now represent the input node. Several combinations of these random weights represent variations of the input node. The euclidean distance between each of these output nodes with the input node is calculated. The node with the least distance is declared as the most accurate representation of the input and is marked as the best matching unit or BMU. With these BMUs as center points, other units are similarly calculated and assigned to the cluster that it is the distance from. Radius of points around BMU weights are updated based on proximity. Radius is shrunk.

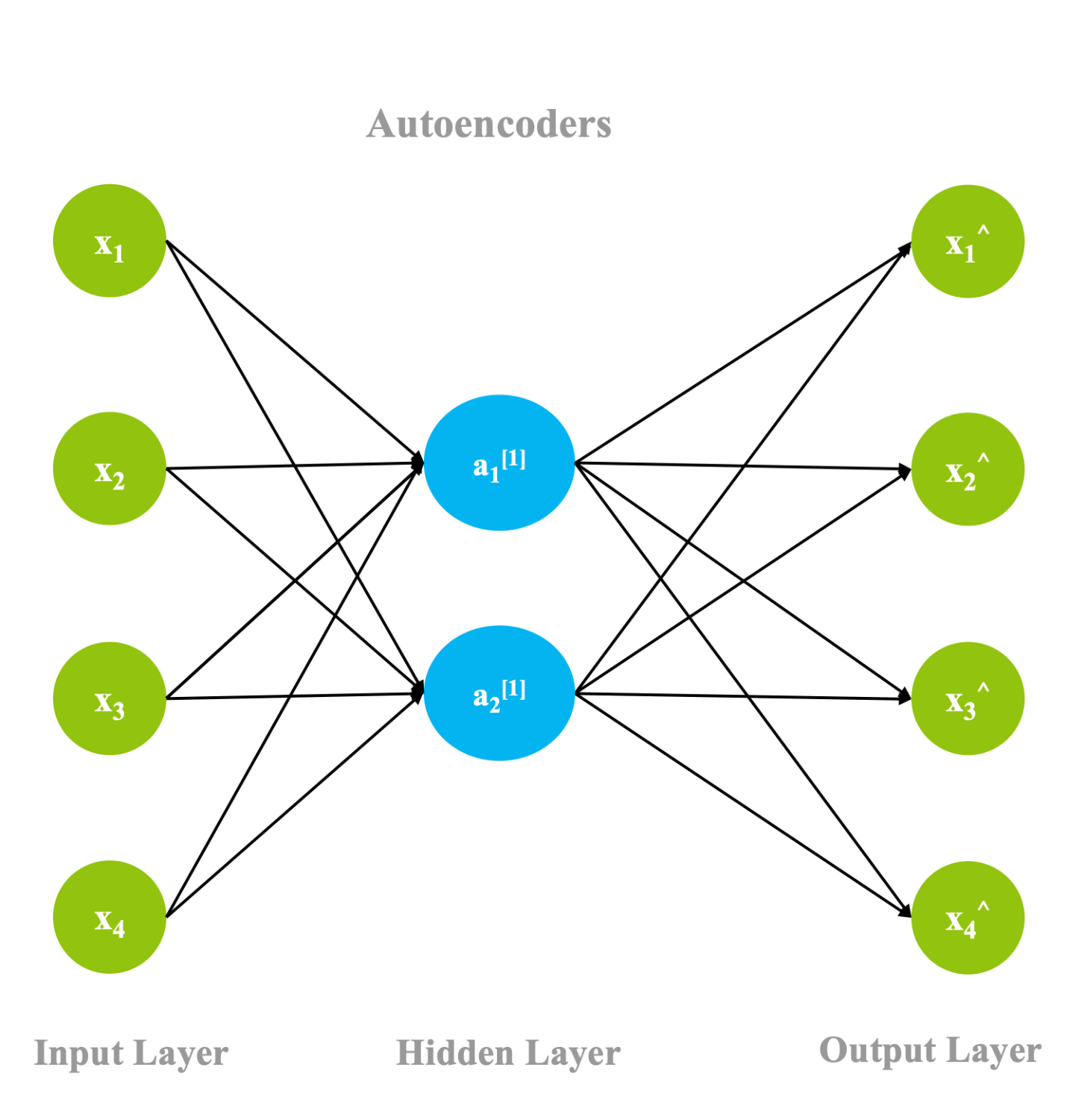
Next, in an SOM, no activation function is applied, and because there are no target labels to compare against there is no concept of calculating error and back propogation.

Example applications: Dimensionality reduction, clustering high-dimensional inputs to 2-dimensional output, radiant grade result, and cluster visualization

### **Autoencoders**

Though the history of when autoencoders were invented is hazy, the first known usage of autoencoders was found to be by LeCun in 1987. This variant of an ANN is composed of 3 layers: input, hidden, and output layers.

First, the input layer is encoded into the hidden layer using an appropriate encoding function. The number of nodes in the hidden layer is much less than the number of nodes in the input layer. This hidden layer contains the compressed representation of the original input. The output layer aims to reconstruct the input layer by using a decoder function.



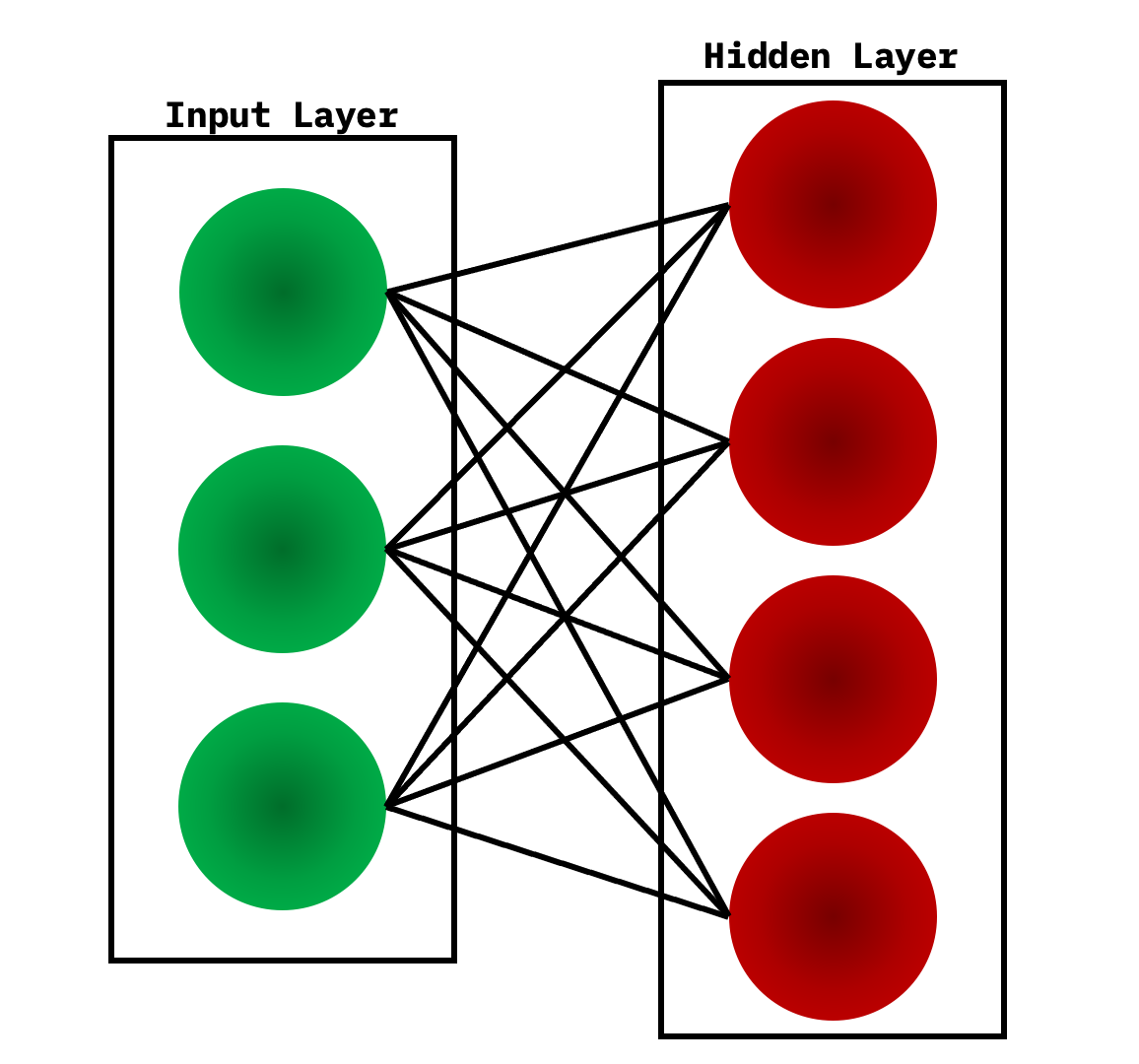
During the training phase, the difference between the input and the output layer is calculated using an error function, and the weights are adjusted to minimize the error. Unlike traditional unsupervised learning techniques, where there is no data to compare the outputs against, autoencoders learn continuosly using backward propagation. For this reason, autoencoders are classified as self supervised algorithms.

Example applications: Dimensionality reduction, data interpolation, and data compression/decompression

### **Restricted Boltzmann Machines**

Though RBMs became popular much later, they were originally invented by Paul Smolensky in 1986 and was known as a Harmonium.

An RBM is a 2-layered neural network. The layers are input and hidden layers. As shown in the following figure, in RBMs every node in a hidden layer is connected to every node in a visible layer. In a traditional Boltzmann Machine, nodes within the input and hidden layer are also connected. Due to computational complexity, nodes within a layer are not connected in a Restricted Boltzmann Machine.



During the training phase, RBMs calculate the probabilty distribution of the training set using a stochastic approach. When the training begins, each neuron gets activated at random. Also, the model contains respective hidden and visible bias. While the hidden bias is used in the forward pass to build the activation, the visible bias helps in reconstructing the input.

Because in an RBM the reconstructed input is always different from the original input, they are also known as generative models.

Also, because of the built-in randomness, the same predictions result in different outputs. In fact, this is the most significant difference from an autoencoder, which is a deterministic model.