1. What is ANN?

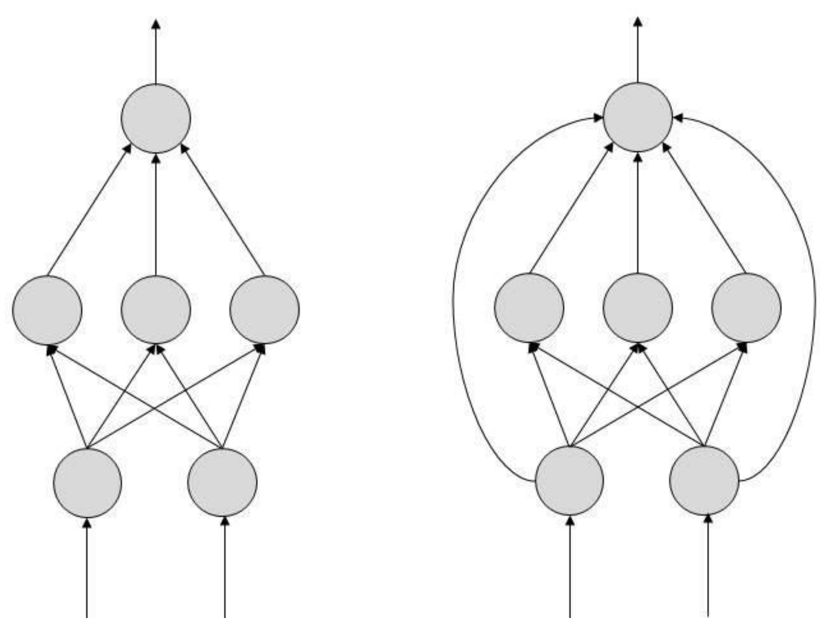
   An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output.

   ANNs are considered nonlinear statistical data modeling tools where the complex relationships between inputs and outputs are modeled or patterns are found.

   ANN is also known as a neural network.

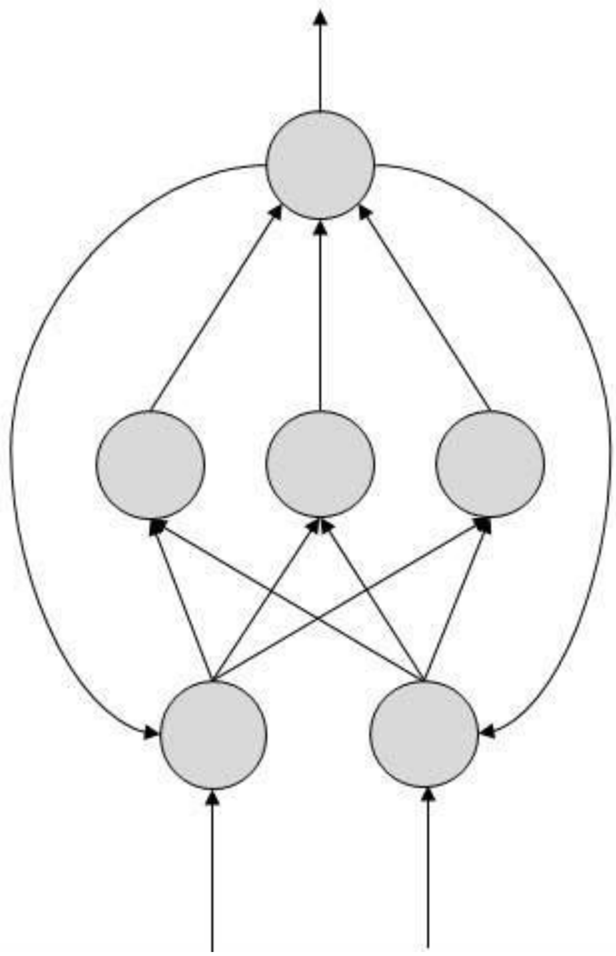
2. Types of ANN?
   a. Feed Forward ANN

      The information flow is unidirectional. A unit sends information to other unit from which it does not receive any information. There are no feedback loops. They are used in pattern generation/recognition/classification. They have fixed inputs and outputs.



   b. Recurrent ANN

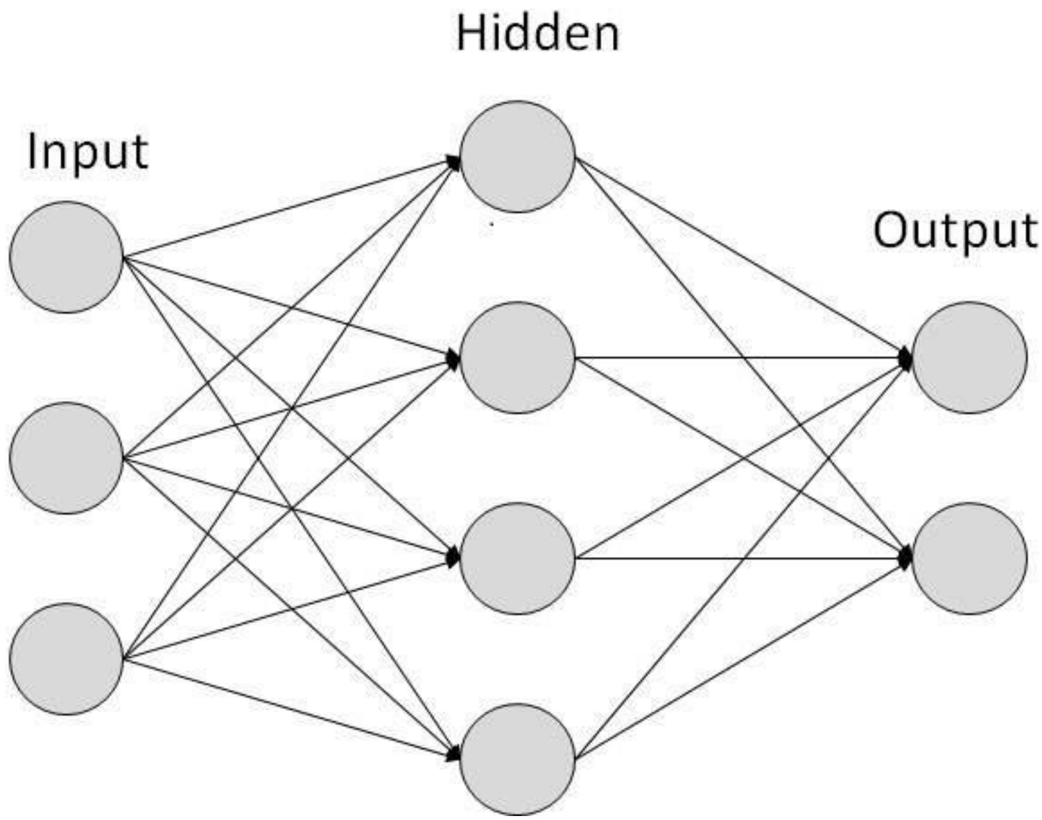
A **recurrent neural network** (**RNN**) is a class of artificial neural network where connections between units form a directed cycle. This creates an internal state of the network which allows it to exhibit dynamic temporal behavior.
Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs.



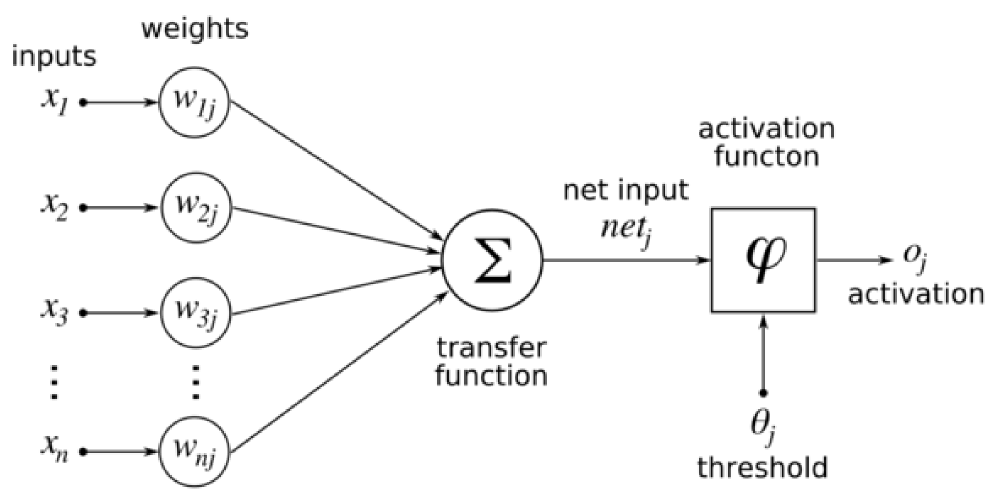
3. Basic structures of ANN.

ANNs are composed of multiple nodes, which imitate biological neurons of human brain. The neurons are connected by links and they interact with each other. The nodes can take input data and perform simple operations on the data. The result of these operations is passed to other neurons. The output at each node is called its **activation** or **node value.**

Each link is associated with **weight.** ANNs are capable of learning, which takes place by altering weight values.



4. How ANN works?



5. Training Rules of ANN
    a. Perceptron training rule

1.initialize the weights (either to zero or to a small random value)

2.pick a learning rate m (this is a number between 0 and 1)

3.Until stopping condition is satisfied (e.g. weights don't change):

For each training pattern (x, t):

compute output activation y = f(w x)

If y = t, don't change weights

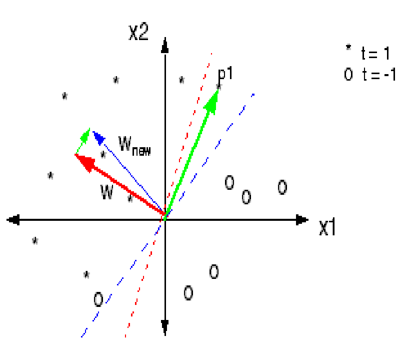
If y != t, update the weights:

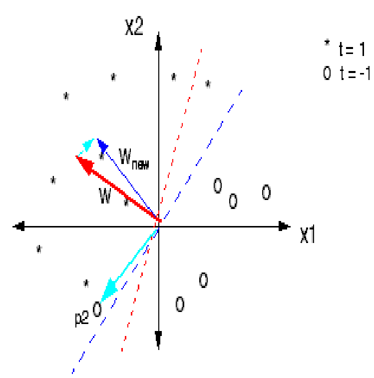
w(new) = w(old) + 2 m t x

or

w(new) = w(old) + m (t - y) x, for all t

Consider wht happens below when the training pattern p1 or p2 is chosen. Before updating the weight W, we note that both p1 and p2 are incorrectly classified (red dashed line is decision boundary). Suppose we choose p1 to update the weights as in picture below on the left. P1 has target value t=1, so that the weight is moved a small amount in the direction of p1. Suppose we choose p2 to update the weights. P2 has target value t=-1 so the weight is moved a small amount in the direction of -p2. In either case, the new boundary (blue dashed line) is better than before.





6. Application of ANN
   a. Data Mining
   b. Prediction
   c. Image Processing
   d. Deep Learning

- **Please search in internet for details**