Explore the effect of different hyperparameters while implementing a Simple Fully Connected Neural Network

Introduction

- A neural network is a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experiential knowledge and making it available for use.
- It resembles the brain in two respects:

1. Knowledge is acquired by the network from its environment through a learning process.

2. Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

Advantages of Artificial Neural Network

- 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.

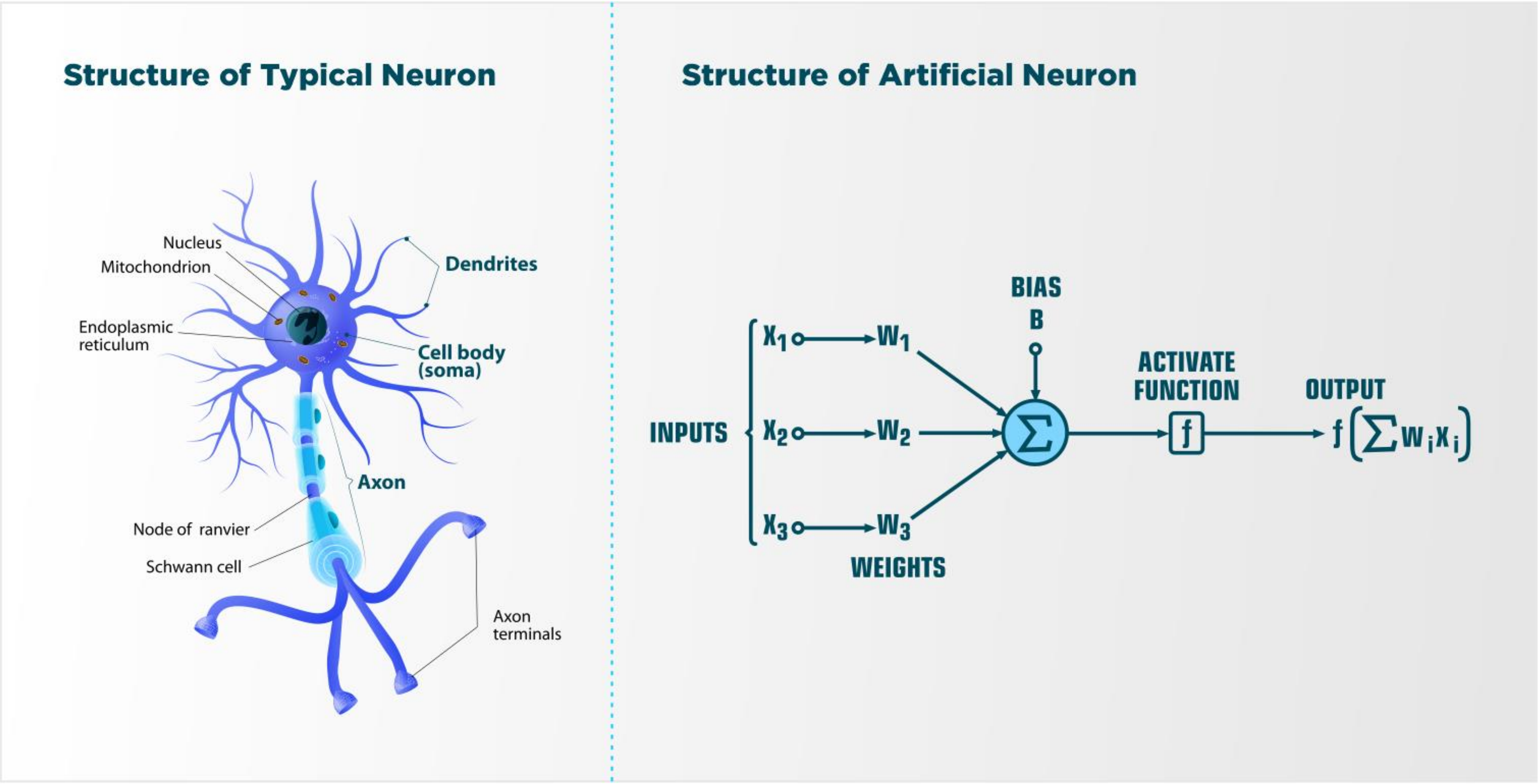
Disadvantages of Artificial Neural Network

- The 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.

The ANN applications

- Classification, the aim is to predict the class of an input vector
- Pattern matching, the aim is to produce a pattern best associated with a given input vector
- Pattern completion, the aim is to complete the missing parts of a given input vector
- Optimization, the aim is to find the optimal values of parameters in an optimization problem
- Control, an appropriate action is suggested based on given an input vectors
- Function approximation/times series modeling, the aim is to learn the functional relationships between input and desired output vectors;
- Data mining, with the aim of discovering hidden patterns from data (knowledge discovery)

Structure

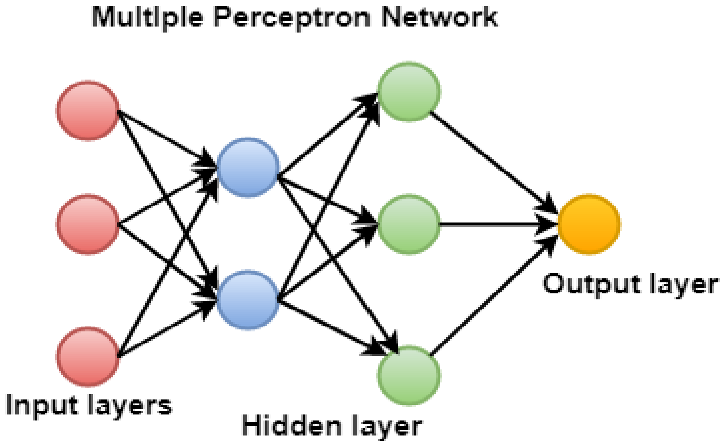

Structure of Typical Neuron
Nucleus
Mitochondrion
Endoplasmic reticulum
Dendrites
Cell body (soma)
Axon
Node of ranvier
Schwann cell
Axon terminals
Structure of Artificial Neuron
INPUTS
X_1
X_2
X_3
W_1
W_2
W_3
WEIGHTS
BIAS
B
Σ
ACTIVATE FUNCTION
f
OUTPUT
$f\left(\sum w_i x_i\right)$

Elements of a Neural Network

Input Layer :- This layer accepts input features. It provides information from the outside world to the network, no computation is performed at this layer, nodes here just pass on the information(features) to the hidden layer.
Hidden Layer :- Nodes of this layer are not exposed to the outer world, they are the part of the abstraction provided by any neural network. Hidden layer performs all sort of computation on the features entered through the input layer and transfer the result to the output layer.
Output Layer :- This layer bring up the information learned by the network to the outer world.

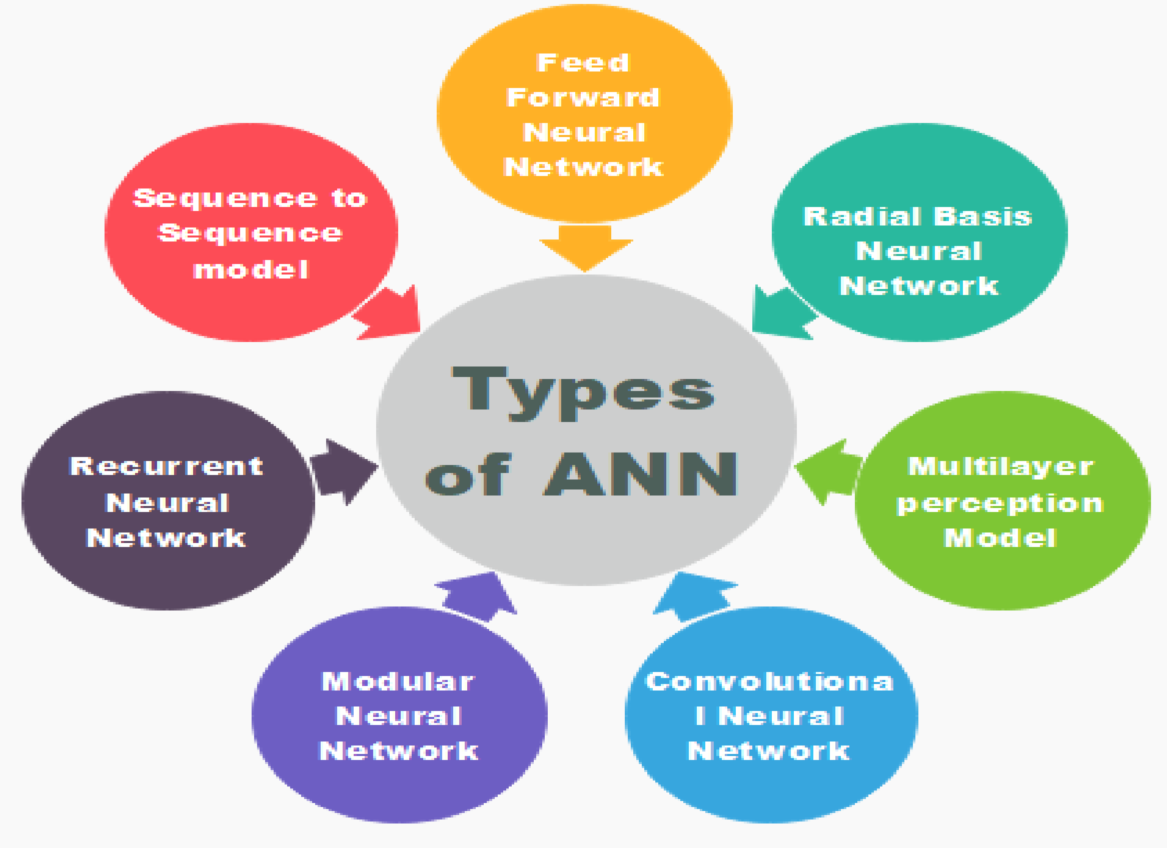
The neural network is made up many **perceptron**s.

Perceptron is a single layer neural network. It is a binary classifier and part of supervised learning. A simple model of the biological neuron in an artificial neural network is known as the perceptron.

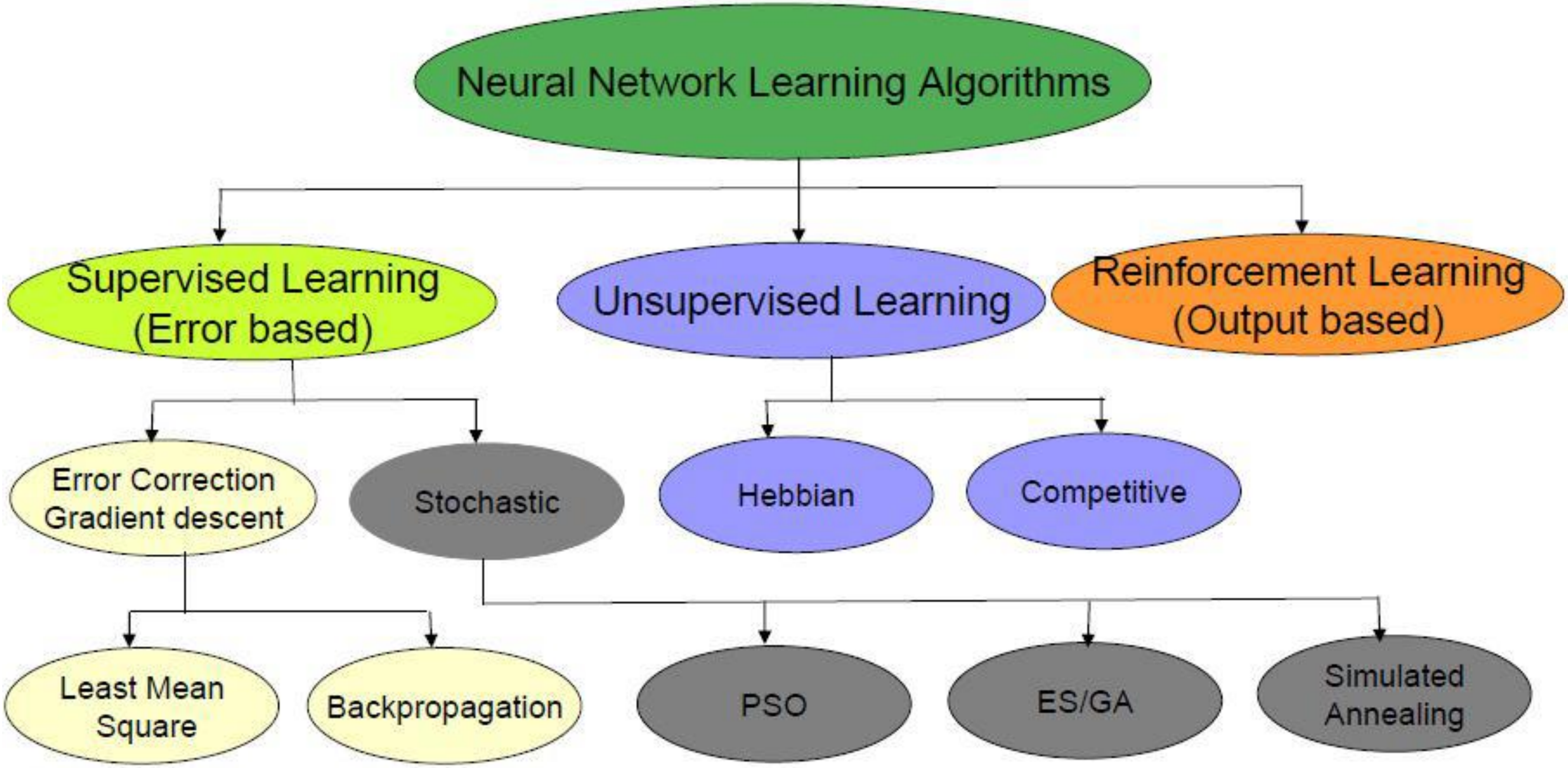


Classification of ANN based on architecture

Neural Network works the same as the human nervous system functions. There are several types of neural network. These networks implementation are based on the set of parameter and mathematical operation that are required for determining the output.


Feed Forward Neural Network
Sequence to Sequence model
Radial Basis Neural Network
Types of ANN
Recurrent Neural Network
Multilayer perception Model
Modular Neural Network
Convolutional Neural Network

Classification of Neural Networks


Neural Network Learning Algorithms
Supervised Learning (Error based)
Unsupervised Learning
Reinforcement Learning (Output based)
Error Correction Gradient descent
Stochastic
Hebbian
Competitive
Least Mean Square
Backpropagation
PSO
ES/GA
Simulated Annealing

Hyperparameters

- **Hyperparameters** are the **variables which determines the network structure**(Eg: Number of Hidden Units) and the **variables which determine how the network is trained**(Eg: Learning Rate).

Learning Rate

- **Learning Rate:** This is a hyperparameter that specifies how fast a neural network updates its gradient parameters.
- The learning rate simply specifies the rate at which a neural network learns – that is how much of a step it takes descending along the cost function.
- If the learning rate is too small, the model will converge or descend slowly and this can be computationally expensive. However, if the learning rate is too large, the model may take gigantic descents and miss the global minimum.
- A good practice is to use a decaying learning rate – that is one that starts out large and changes to a small value overtime. A longer training time is usually advised to prevent high bias and low variance – underfitting.

Number of iterations for training

- **Number of iterations for training:** This refers to the number of times your model performs forward and backward propagations, as well as updates its weight and bias parameters.
- One practice to have a good model that is not overfitted is to use early stop.
- This is a practice where training is stopped where the cost starts increasing steadily instead of decreasing.
- The downside is that you are trying to perform two orthogonal processes simultaneously – optimize cost function and prevent overfitting.

Number of Hidden Layers

- **Number of Hidden Layers:** This refers to the number of layers in your neural network that are not the input and output layers.
- If we use a small number of layers, it could result in underfitting. If we use a large number of hidden layers, there could be an overfitting problem.
- A good rule of thumb is to use a sizeable number of hidden layers with regularization, as that will help prevent overfitting.

Number of Neurons in each layer

- **Number of Neurons in each layer:** This refers to the number of nodes or units in each (hidden) layer of your neural network.
- As with the number of hidden layers, a small number of neurons could cause underfitting, while a large number of hidden layers could cause overfitting.
- Often times, using the same number of units in each hidden layer works well.
- So does an increasing or decreasing unit size across each hidden layers. Also, using a first hidden layer larger than the number of input features is advised.
- Furthermore, unsupervised learning usually perform better with larger layers and units when compared supervised learning.

Activation Functions

- **Activation Functions:** This refers to the functions that learn non-linear features in every layer of a neural network.
- If you are familiar with a simple neural network, you would know ever node has two major operations – a linear function and a non-linear function obtained with the activation function.
- This activation functions helps the network to learn the non-linear properties of the data set.
- The Rectified Linear Unit is usually the most popular choice for hidden layers.
- Depending on the type of output you need from your neural network, you can select an appropriate activation function for it.

Weight Initialization

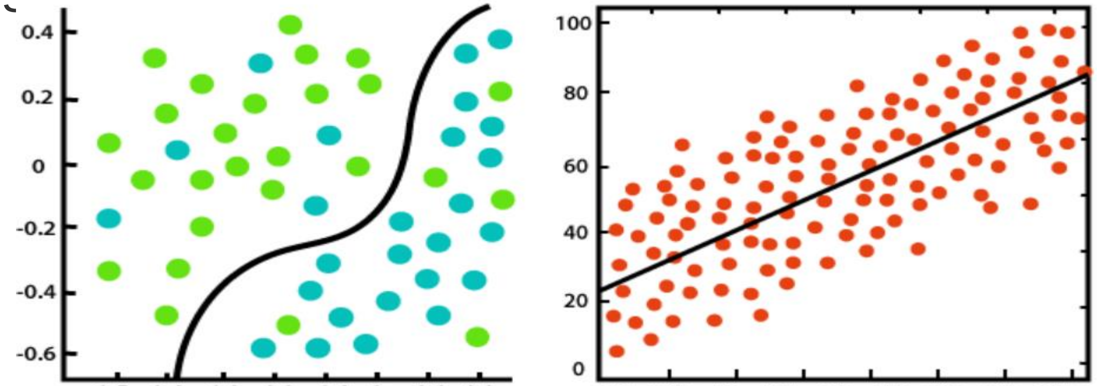
- **Weight Initialization:** The initialization of weights can have a great impact on the performance of the neural network.
- The weights can affect how quickly or if at all the local minimum is found by the network training algorithm.
- This problem is often known as the vanishing or exploding gradient problem. Usually, random numbers from a normal or Gaussian distribution are advised, with a very small standard deviation (about 0.1).
- Some activation functions also perform better with the addition of a variance term – e.g. the He Initialization for the ReLu activation function.

Regularization Type and Parameters

- **Regularization Type and Parameters:** Notice that the specification of many of the hyperparameters could often cause overfitting or underfitting.
- Regularization is a practice in machine learning that is used to curb overfitting.
- It involves smoothening out the decision boundary of algorithms with a regularization term.
- There are different types of regularization. For neural networks, the popular ones are L1, L2 and Dropout regularizations.

Problem type

- Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labeled datasets. But the difference between both is how they are used for different machine learning problems.
- The main difference between Regression and Classification algorithms that Regression algorithms are used to **predict the continuous** values such as price, salary, age, etc. and Classification algorithms are used to **predict/Classify the discrete values** s or Not Spam, etc.



Classification Regression

Problem

- https://playground.tensorflow.org

Try different Hyperparameters using this Url

References

- Basic Hyperparameter Tuning for Neural Networks – Tech-Quantum
- Regression vs Classification in Machine Learning - Javatpoint