Vidya Pratishthan's

Kamalnayan Bajaj Institute of Engineering and Technology

Vidyanagari, Baramati, Pune 413 133.

INDEX

Department of :- Artificial Intelligence And Data Science CLASS T.E

SR. NO.	NAME OF EXPERIMENT	Expt. Conducted on	Expt. Checked on	PAGE NO.	SIGN	REMARK
1.	Write Python Program to plot activation functions.	7/2/23	13/2/23	1		
2.	Generate AND-NOT function using M-P model.	10/2/23	13/2/23	5		
3.	Write Python Program using perceptron to recognize even/odd number.	14/2/23	27/2/23	11		
4.	With suitable diagram demonstrate perceptron decision region.	28/2/23	20/4/23	14		

CERTIFICATE

This is to certify that Mr./Miss Devrat Yashraj Deepak of Class T.E (A.I.D.S) Roll No. 237091 has Satisfactorily Completed the term work of the subject Software laboratory 2 for 6th Semester of 2022-2023.

Date : 12/05

Staff Member / In-charge

H.O.D.

PRINCIPAL

Vidya Pratishthan's

Kamalnayan Bajaj Institute of Engineering and Technology

Vidyanagari, Baramati, Pune 413 133

INDEX

Department of :- Artificial Intelligence And Data Science CLASS T.E

SR. NO.	NAME OF EXPERIMENT	Expt. Conducted on	Expt. Checked on	PAGE NO.	SIGN	REMARK
5.	Bidirectional Associative Memory with 4 pair vectors.	10/3/23	20/4/23	17		
6.	ANN processing forward & backward propagation.	17/3/23	18/5/23	25		
7.	Neural Network for multiclass classification.	28/3/23	18/5/23	28		
8.	Python program for ART-1 Network.	21/4/23	18/5/23	31		
9.	Implement CNN object detection algorithm.	9/5/23		34		
10.	Image classification through tensorflow, try to improve hyperparameter tunning.	13/5/23	18/5/23	37		
11.	MNIST Hand-Written character detecting using Pytorch, Keras, tensorflow	13/5/23	18/5/23	40		

CERTIFICATE

This is to certify that Mr./Miss Devrat Yashraj Deepak of Class T.E (A.I.D.S) Roll No. 223703 has Satisfactorily Completed the term work of the subject, Software laboratory-2 for 6th Semester of 2022-2023

Date : / /

Staff Member

In-charge

H.O.D.

PRINCIPAL

Assignment No. 1

6/2/2023

Problem Statement :

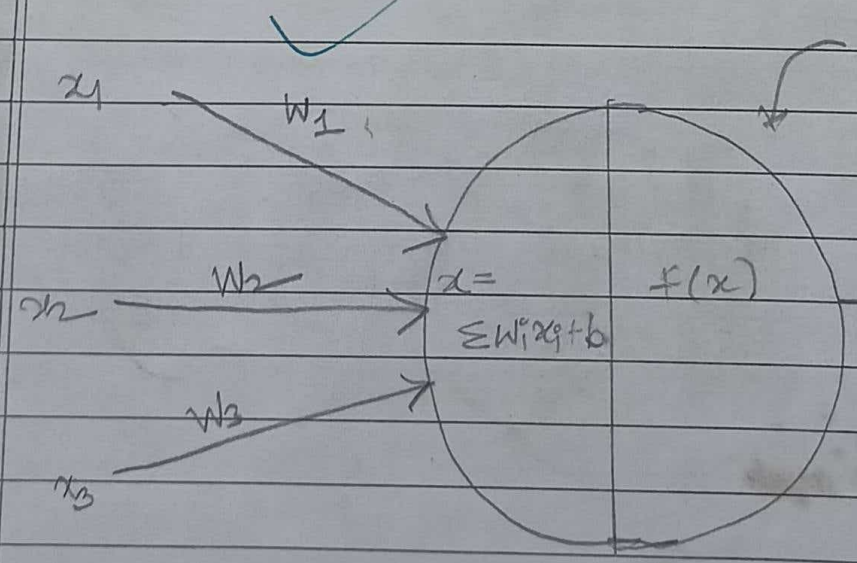
Write a program to plot a few functions that are being used in neural networks.

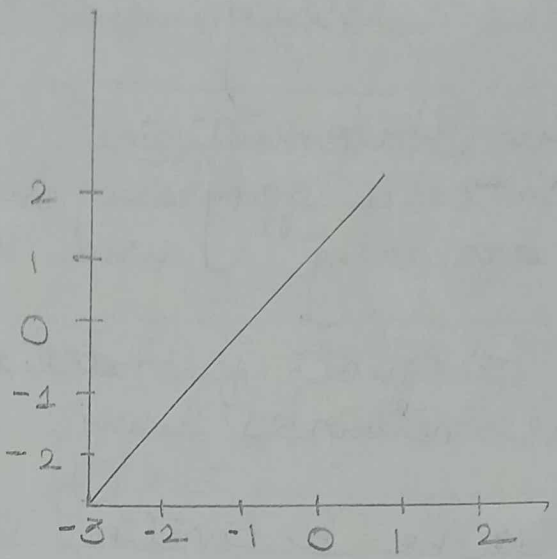
Aim : To plot few activation functions used in neural networks using Python.

Theory : ① The activation function in the neural networks are applied to the output of a neuron (or layers of neurons), which modifies outputs.

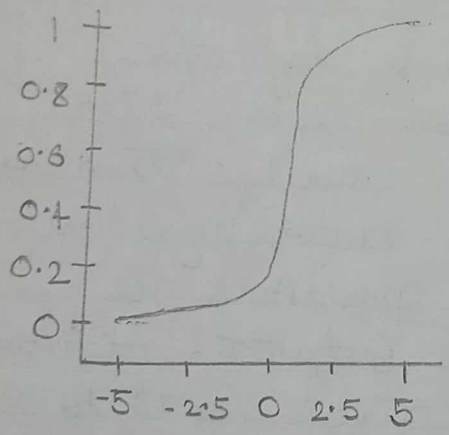
② We use activation functions because if the activation function is non-linear, it allows for neural networks with usually two or more hidden layers to map non-linear functions.

③ The main purpose of activation functions is to produce non-linearity into the outputs. Activation Function will decide how much to fire 1 or 0 to the output or keep as it is.

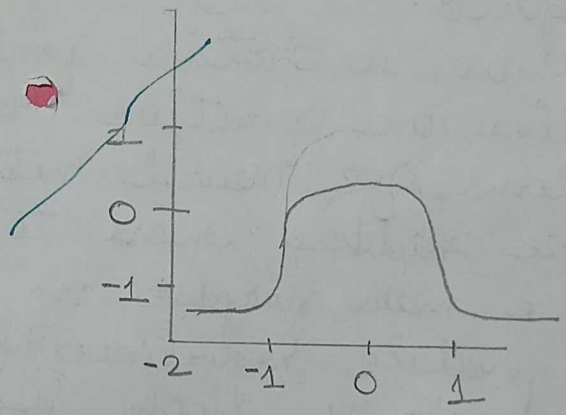




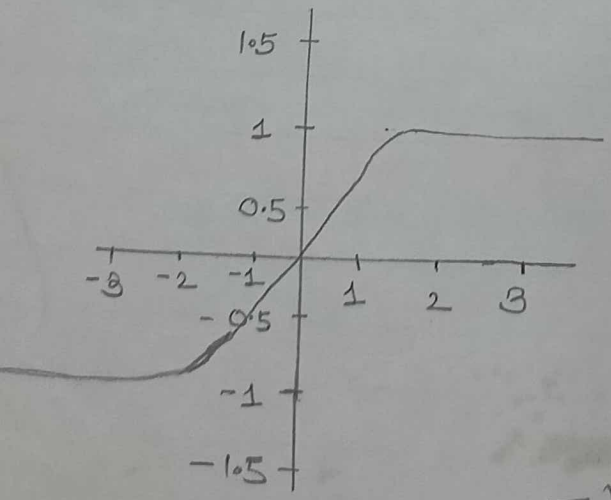
Linear Function



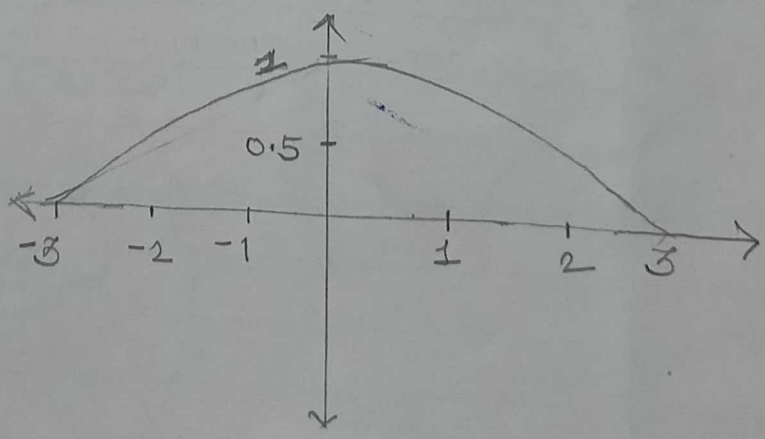
Sigmoid



its Derivative



Tanh



its Derivative

There are various types of Neural Networks and has activation functions in it:

① Binary Step function:

i. It has certain threshold value that decides whether neuron should be activated or not.

ii. When the input value of threshold is greater than it then the neuron is activated otherwise its is deactivated.

Mathematical Representation:

$$f(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

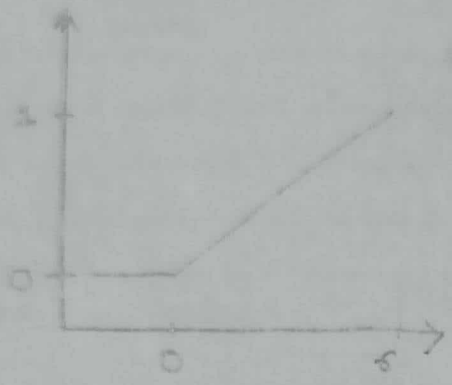
② Linear activation function:

i. It is also known as "No activation function"

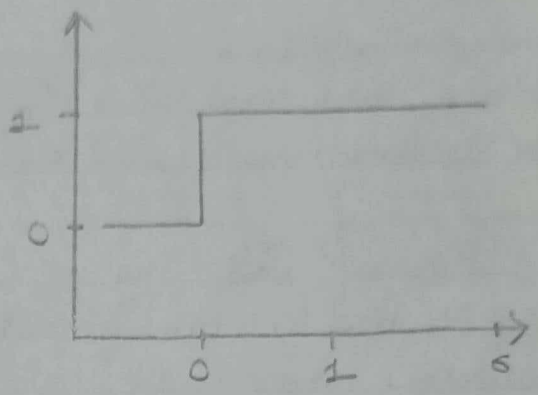
ii. The function displays the weighted sum of the inputs, simply splits the value it was given.

Mathematical Representation:

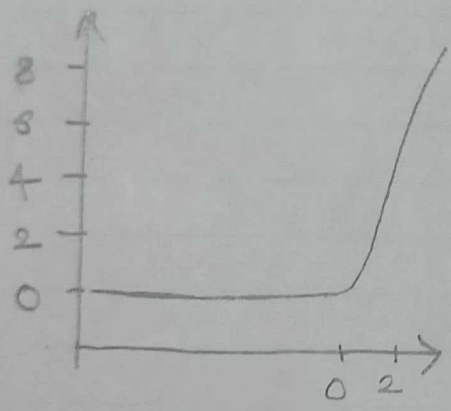
$$f(x) = x$$



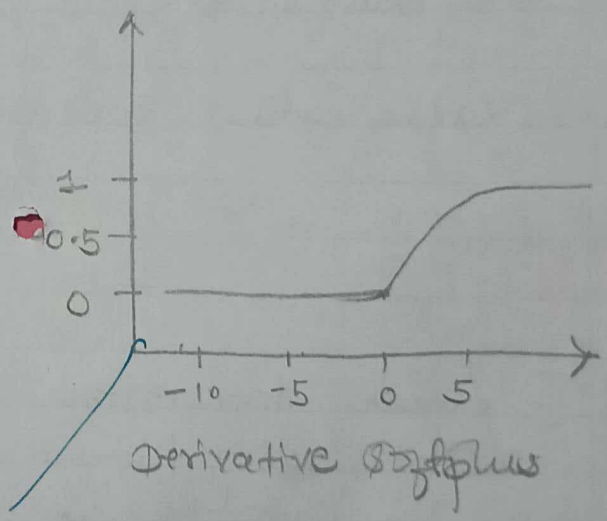
ReLU



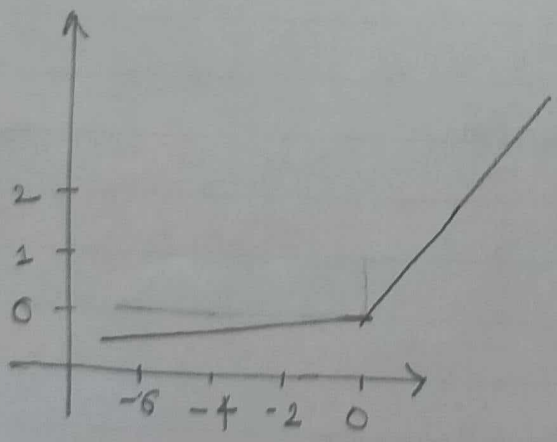
Derivative of ReLU



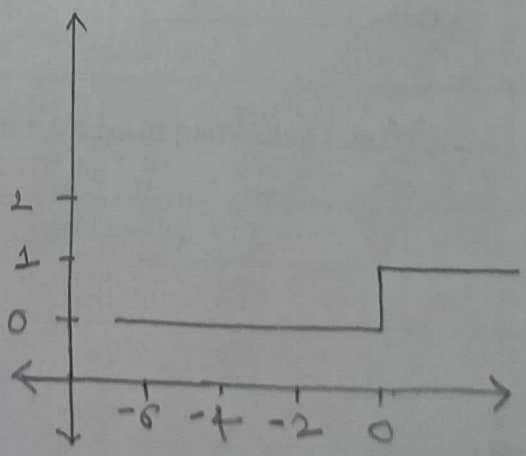
Softplus



Derivative Softplus



Leaky ReLU



Derivative of Leaky ReLU

③ Non Linear Activation Functions:
This types of activation produces non-linearity in the outputs.

#1 Sigmoid Activation Function:
- It is a function plotted as 'S' Shaped curve.

- Equation:

$$Y = \frac{1}{1+e^{-x}}$$

$$f'(x) = f(x)\,(1-f(x))$$

- Range & Nature: 0 to 1
It is non-linear and very steep
Small change in x would bring large change in y.

#2 Tanh Function:
- This activation function works more better than Sigmoid function.

- It's actually mathematically shifted version of Sigmoid function.

$$f(x) = Y = \frac{2}{1+e^{-2x}} - 1$$

$$f'(x) = \frac{1-(e^{x}-e^{-x})^{2}}{(e^{x}+e^{-x})^{2}}$$

- Nature & Range: -1 to 1, non-linear.

#3 RELU Function:

- It stands for Rectified Linear Unit.
- Implemented in hidden layers of Neural Network.

- Equation: $A(x) = \max(0, x)$
  it gives output x if x is positive 0 otherwise.

- Nature & Range: Non linear, 0 to infinity

#4. Softmax Function:

- Softmax function is also a type of sigmoid function, but useful when we are trying to handle multi-class classification problems.

- Nature: Non-linear

-

- Used in output layer of the classifier where we are actually trying to attain the probabilities to define the class of each input.

- Equation: $\frac{\exp(z_i)}{\sum_j \exp(z_j)}$

#5 Leaky Relu function:

- Leaky Relu is improved version of Relu function to solve Dying Relu problem as it has small positive slope in the negative area.

Mathematical Representation:

$$f(x) = \max(0.1x, x)$$

Derivative of leaky Relu:

$$f'(x) = \begin{cases} 0 & \text{if } x \geq 0 \\ 0.01 & \text{if } x < 0 \end{cases}$$

#6 Parametric Relu function:

- It is a function of another version aims to solve the problem of gradients becoming zero for the left half of the axis.

- This function provides the slope of the negative part of the function as an argument by performing backpropagation the most appropriate value of a learnt.

Mathematical Representation:

$$f(x) = \max(ax, x)$$

where as 'a' in slope.

#7 Exponential Linear Unit (ELU) function:

- ELU is variant of RELU that modifies the slope of negative part of the function.

- ELU uses, log curve to define negative values

unlike leaky Relu & PRelu function with straight line.

Mathematical Representation:

$$f(x) : \begin{cases} x & \text{for } x > 0 \\ \alpha(e^x - 1) & \text{for } x < 0 \end{cases}$$

Derivative of Exponential Linear Unit function:

$$f'(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ f(x) + \alpha & \text{if } x < 0 \end{cases}$$

#8. Swish function:

- It is self-gated activation function developed by research at google.

$$f(x) = x * \text{sigmoid}(x)$$

Conclusion: Activation Functions are used in neural networks. The non-linear activation function have higher gradients which allow a given model learn and execute difficult task.

VPKBIET

Assignment No. 2

10/2/2023

Aim: Generate ANDNOT Function using McCulloch-Pitts neural net by Python program.

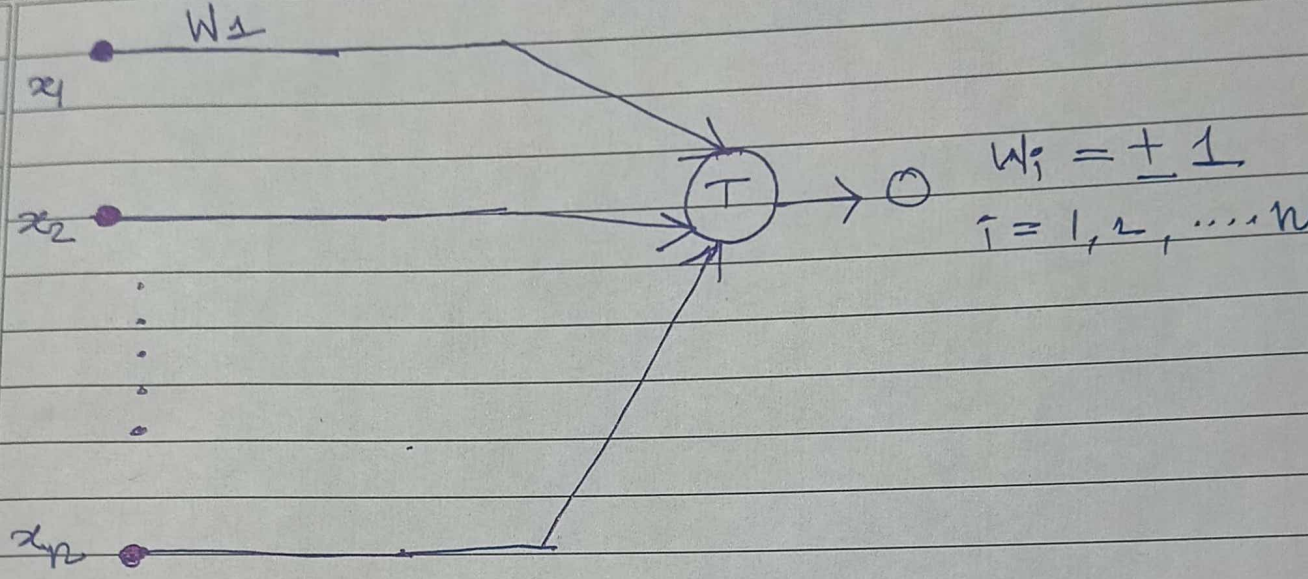
Objective: Design a MP model for logic functions e.g. ANDNOT.

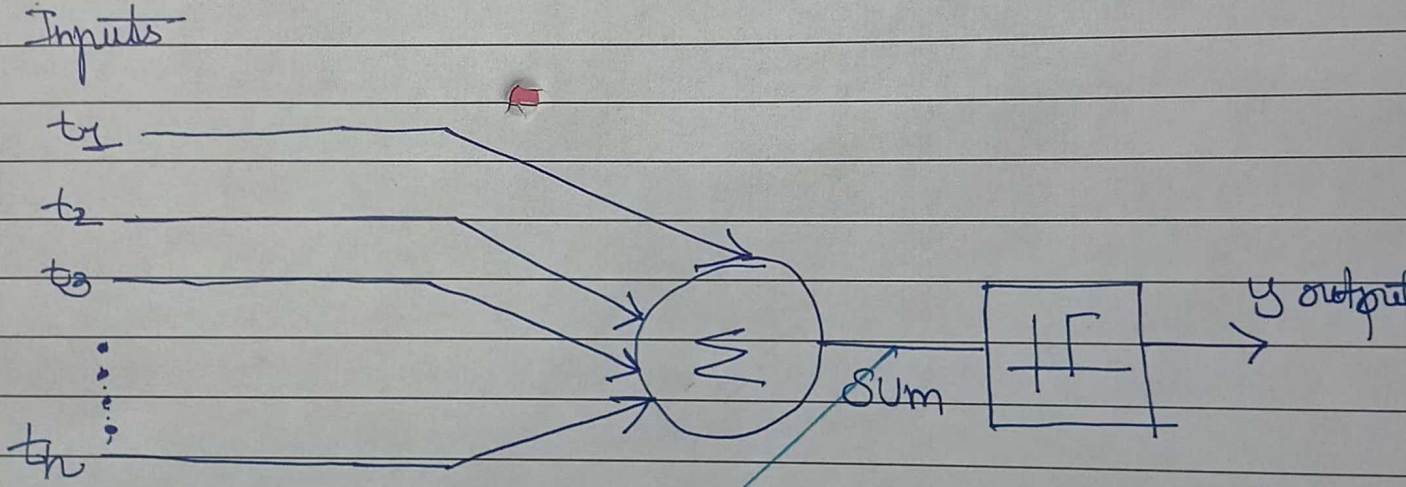
Theory: McCulloch-Pitts neural model:

i. The first definations of a synthetic neuron model based on the highly simplified considerations of the biological model described in the preceding section was formulated by McCulloch & Pitts (194

ii. The inputs x_i for $i = 1, 2, \ldots, n$ are 0 or 1 depending on the absence or presence of the input at instant k. The neuron's output signal is denoted as O_j. The firing rule for this model is defined as follows:

$$O^{k+1} = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} W_i x_i^k \geq T \\ 0 & \text{if } \sum_{i=1}^{n} W_i x_i^k < T \end{cases}$$





Activation Value : X $= \left(\sum_{i=0}^{n} W_i x_i \right)$

ANDNOT Function :

x_1	x_2	y
1	1	0
1	0	1
0	1	0
0	0	0

ANN with two inputs neurons & single output neuron can operate as an AND NOT logic function if we choose weights.

$W_1 = 1$, $W_2 = -1$ & threshold $\theta = 1$

Y_{in} is a activation value

$x_1 = 1$, $x_2 = 1$

$$Y_{in} = (W_1 \cdot x_1) + (W_2 \cdot x_2) = (1x1) + (-1)(1)$$
$$= 0 \quad , \quad y_{in} < \theta ,$$
$$= S_0 \; Y = \theta$$

$x_1 = 1$, $x_2 = 0$

$Y_{in} = (1x1) + (0x-1) = 1$, $y_{in} = \theta$ $S_0 \cdot Y = 1$

$x_1 = 0$, $x_2 = 1$

$Y_{in} = (0x1) + (1x-1) = -1$, $Y_{in} < \theta$, $S_0, Y = 0$

$x_1 = 0$, $x_2 = 0$

$Y_{in} = (0x1) + (-1) = 0$ $Y_{in} < \theta$, $S_0, Y = 0$

S_0 , $Y = [0\ 1\ 0\ 0]$

Expected Outputs:

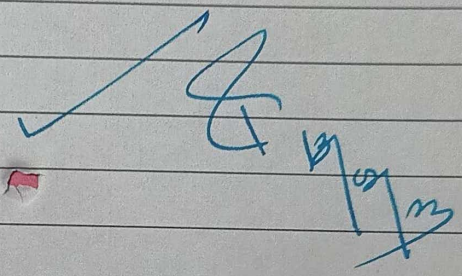
Weights of neuron :-

$W_1 = 1$, $W_2 = -1$, Threshold = 1

With Outputs of Neuron :-

0 1 0 0

Conclusion: In this assignments we learn about different gates with successful generating AND NOT function using McCulloch-Pitts neural network by a Python Program.



VPKBIET

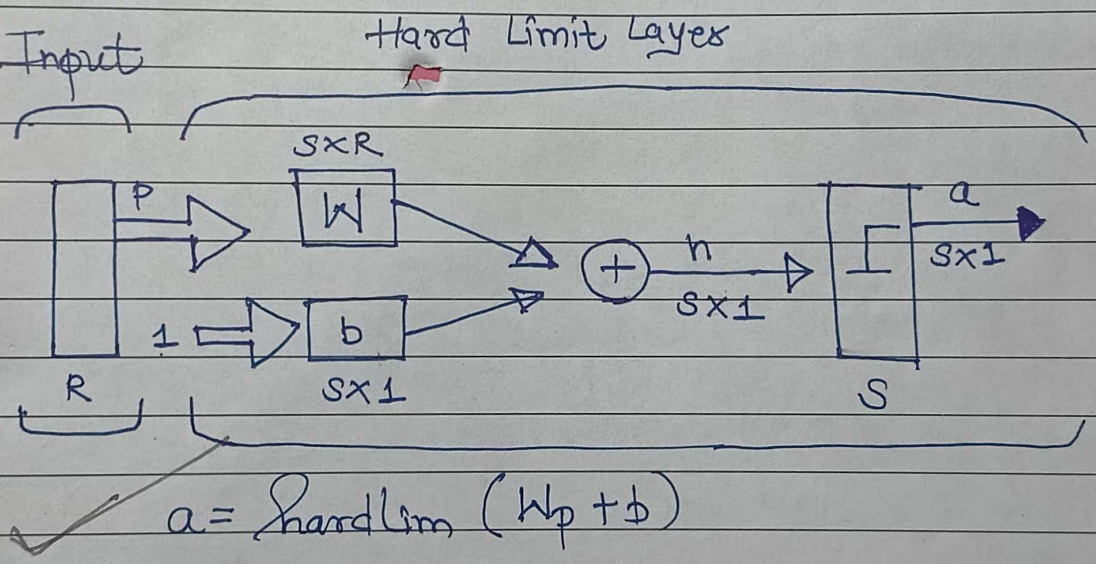
Assignment No: 3

Problem Statement: Write a Python Program using Perceptron Neural Network to recognize even and odd numbers. Given numbers are in ASCII from 0 to 9.

Aim: To Study a neural network that recognize even and odd numbers based on its binary format.

Theory:

Output of a network: $a = hardlim(Wp + b)$



Neural Network is similar to logistic regression (perceptron) but with more layers and hidden layers.

Working of Perceptron Neural Network:

1. It is capable of recognizing the even / odd numbers and convert them to 8-bit binary representation.

ii. The program takes input of number and then the preceptron neural network is then used to classify based on the last bit of binary representation.

iii. The program has 2 hidden layer with 10 and 8 nodes respectively.

iv. Sigmoid Function is used as the activation function and the backpropagation algorithm is used to train the network.

v. The program first preparas the training data, which consists of the decimal numbers and binary classification of even or odd. The network is trained on this data using train function.

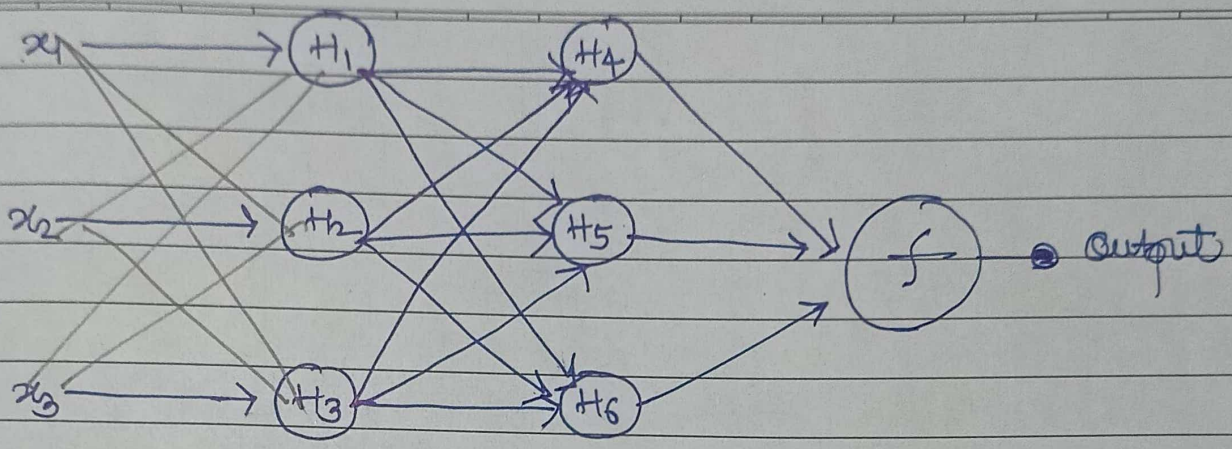
Multi-Layer Neural Network:

① It contains one or more hidden layer (apart from one input and one output layer)

② Multiple-Neuron perceptron can classify inputs into many categories.

③ Sigmoid activation function: Real values inputs and squashed to the range between 0 & 1.

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$



Algorithm:

① Multiply all inputs values with corresponding weights and add them to determine weighted sum.

$$\sum W_i \cdot X_i = x_1 \cdot W_1 + x_2 \cdot W_2 + x_3 \cdot W_3 \ldots\ldots x_n W_n$$

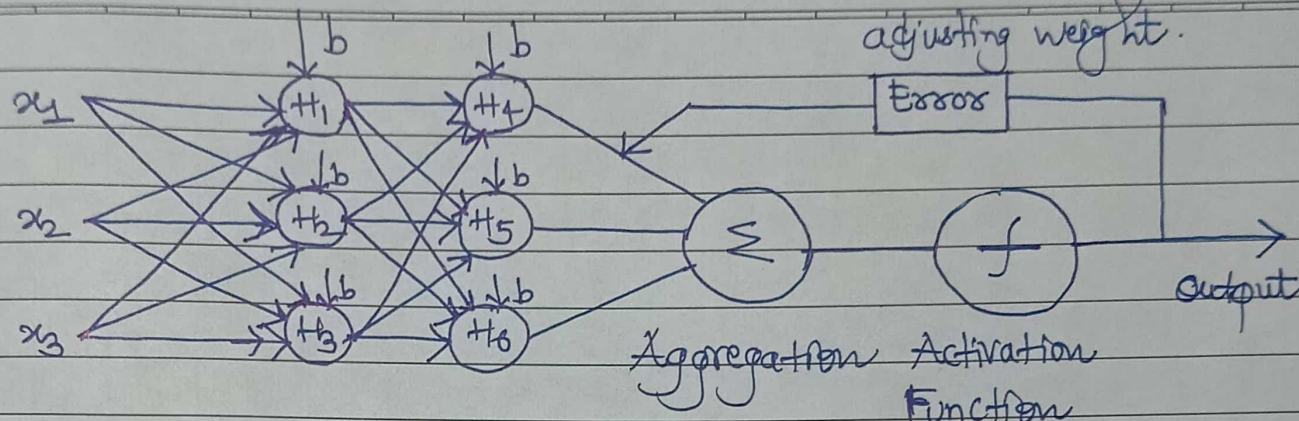
② Add some bias b to improve the model performance

$$\sum W_i \cdot X_i + b$$

③ Activation Function is applied with above mentioned sum

$$Y = f\left(\sum W_i \cdot X_i + b\right)$$

④ The output is in the form of binary or continuous values.



b = bias

f = Activation Function

Σ = Aggregation

x_1, x_2, x_3 = Inputs

Conclusion: The Perceptron Neural Network is a powerful tool for classification tasks and has capability to classify the even & odd number with two hidden layers.

VPKBIET

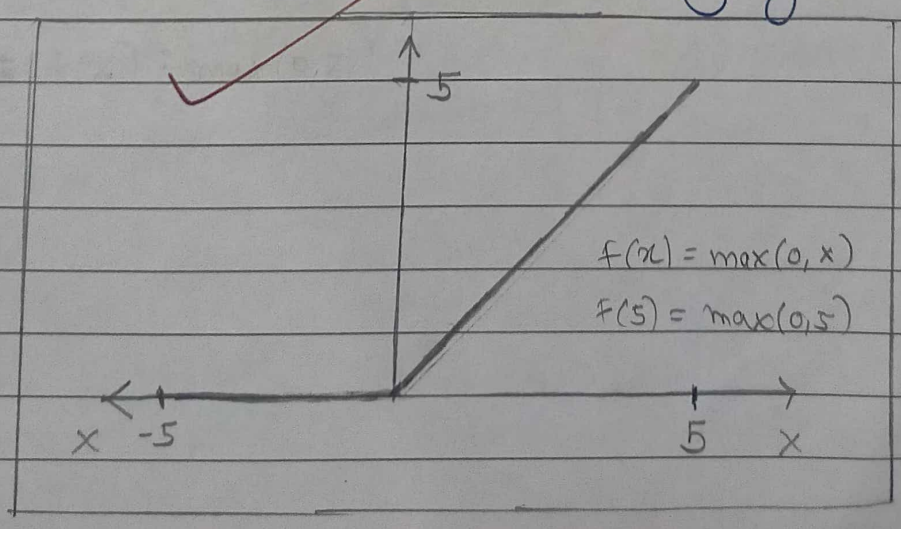
Assignment No. 4

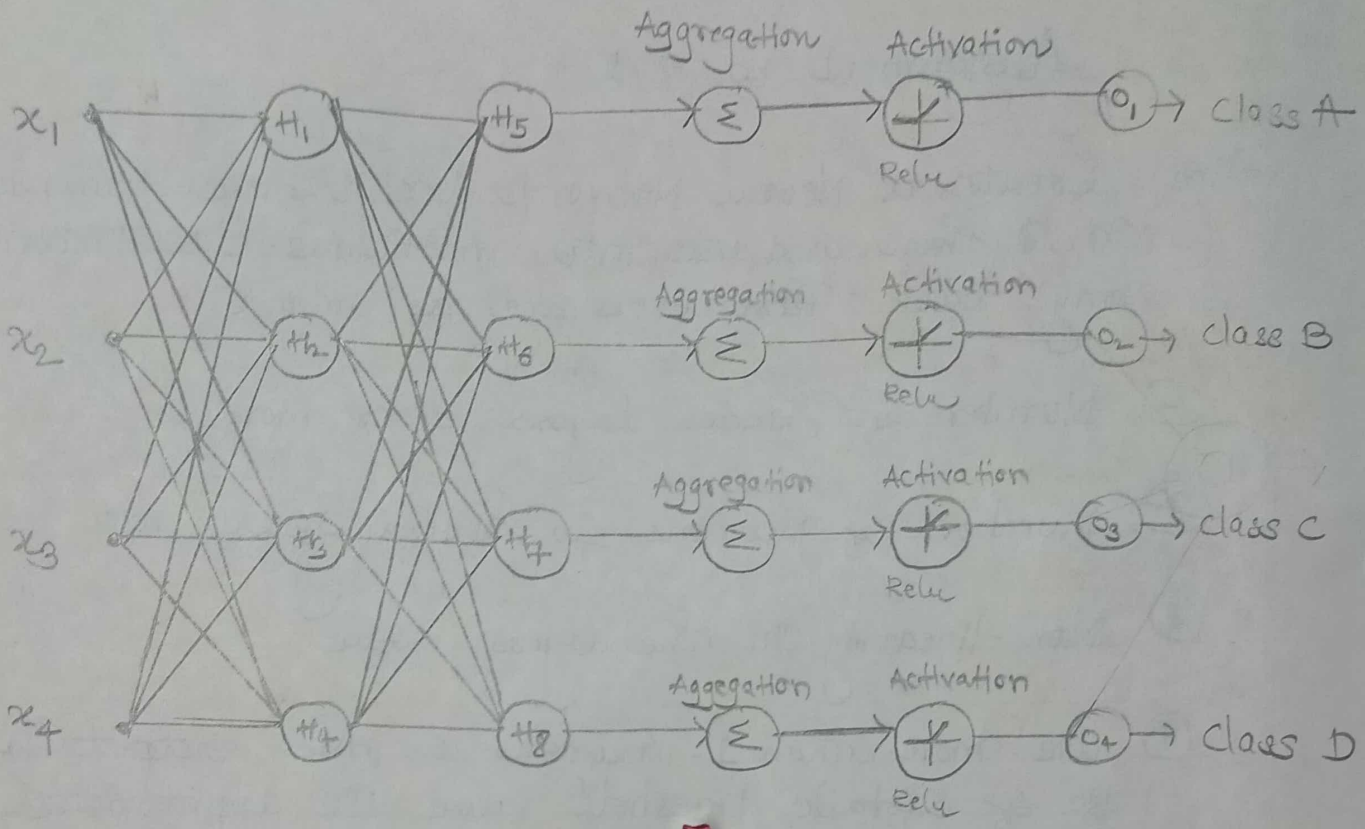
Aim: Create a Neural Network Architecture from Scratch in Python and use into multiclass classification on any data. Parameters are as follows:

1. Number of hidden layers: 1 or more
2. Number of Neurons in hidden layers: 100
3. Non-linearity in the layers: Relu
4. Use more than 1 neurons in the output layer use of suitable threshold value use appropriate optimization algorithm.

Theory: Neural Network is one of the most popular machine learning algorithm. Neural Networks have high efficiency. Neural Network can take any number of inputs and has n hidden layers and has output layer of activated value.

Activation function we are going to use is Relu:





Multiclass Classification

VPKBIET

Use of Multiclass Classification:

i. It is type of supervised learning task in which the goal is to predict the class label of the input sample for possible class labels.

ii. For example, an image classification, the goal may be to predict whether an image is dog, cat, bird. Classification goal is to predict input belongs to specific class.

How multiclass classification works?

1. Inputs vectors are given to neural network.
2. The data flows from one layer to another layer in forward direction.
3. After doing summation and applying activation function as Relu in network we get 4 classes of outputs.
4. If we need to backpropogate error we need to use gradients descents algorithms to modify weights and biases if required.

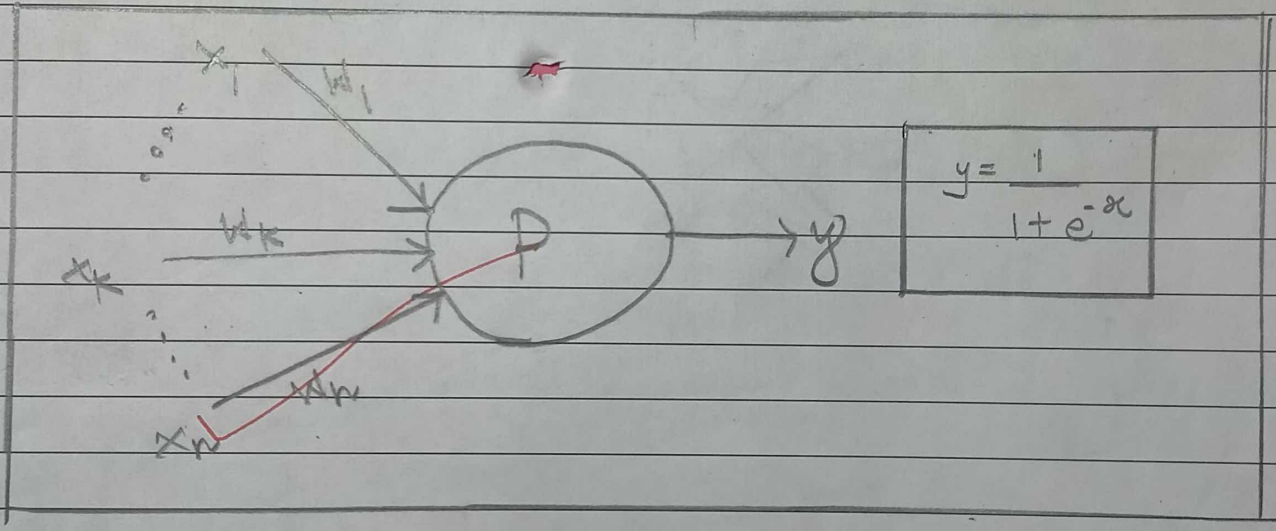
Conclusion: We studied use of Relu function for classifying the outputs in 4 categories i.e working of multiclass classification.

VPKBIET

Assignment No. 5

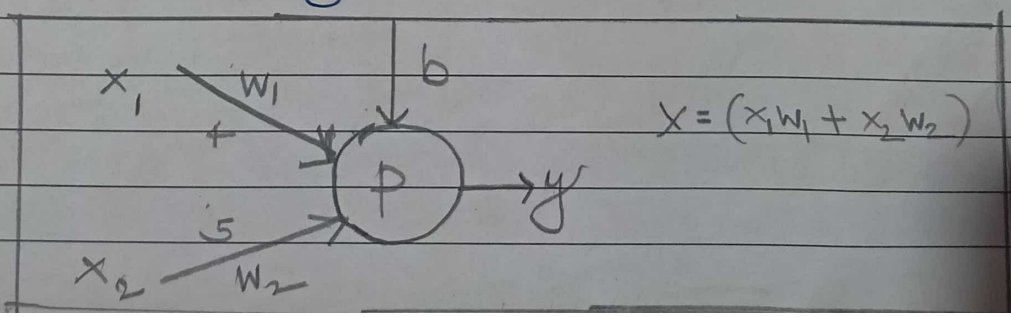
Aim: With a suitable example demonstrate the perceptron learning law with its decision regions using python. Give the output in graphical form.

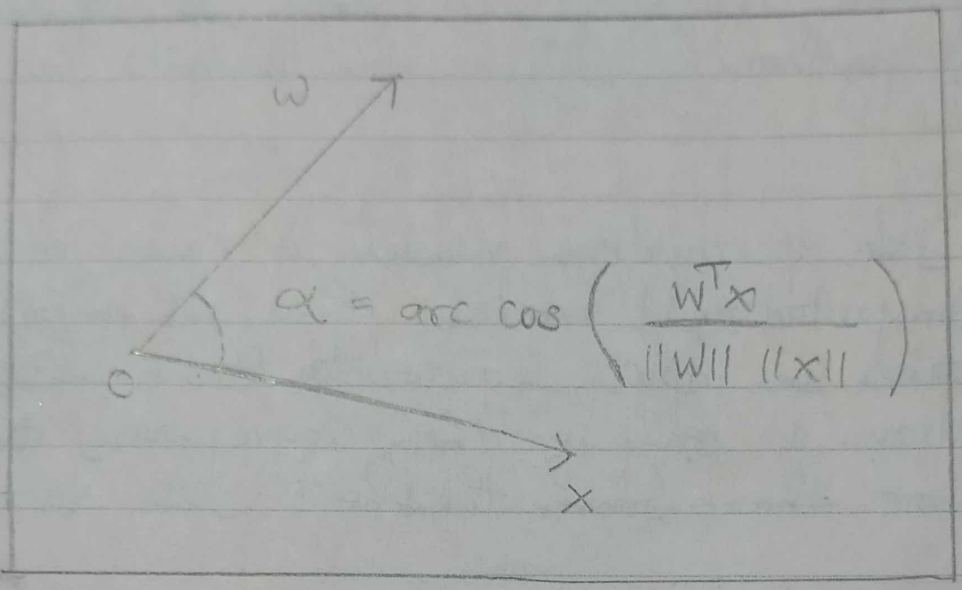
Theory: The perceptron model is more general computational model than MP model. It takes an input, aggregates it (weighted sum) returns 1 only if the aggregated sum is more than some threshold else returns 0.



$$X = x_1w_1 + x_2w_2 + x_3w_3 + \cdots x_kw_k + \cdots + x_nw_n$$

$$f(X) = y$$





A more accepted convention,

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

$$= 0 \quad \text{if} \sum_{i=0}^{n} w_i * x_i < 0$$

where, $x_0 = 1$ and $w_2 = -\theta$

Computing the dot product:

$$W = [w_0, w_1, w_2, \ldots w_n]$$

$$x = [x_1, x_2, x_3, x_4 \ldots x_n]$$

$$W \cdot X = W^T X = \sum_{i=0}^{n} w_i x_i$$

Perceptron learning Algorithm: Our goal is to finds the W vector that can properly classify positive inputs and negative inputs in our data.

Setting threshold value (Th):

$y < Th \rightarrow 0$

$y > Th \rightarrow 1$

if $y < Th$

$W = W + X$

$y > Th$

$W = W - X$

VPKBIET

Angle between Two Vectors:

The same old dot products can be computed differently. eg only you know the angle between vectors and their individual magnitude.

$$W^T x = \|w\| \|x\| \cos\alpha$$

$$\therefore \quad \cos\alpha = \frac{W^T \cdot x}{\|w\| \cdot \|x\|}$$

If the angle is 90° value of α is 0. And the vectors are perpendicular to each other.

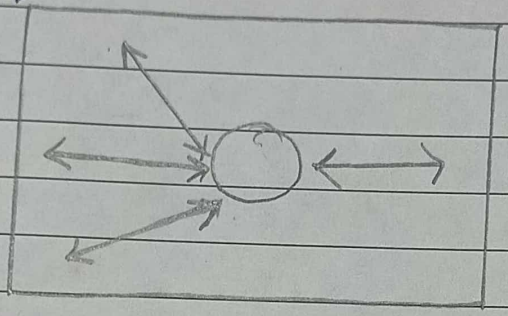
Conclusion: Hence, in this assignment we studied the concepts of perceptron and perceptron learning algorithm we implemented the perceptron learning algorithm on OR Gate and plot the graph of the same.

13/05

Assignment No. 5

Title : Write a Python program Bidirectional Associate Memory for 4 input vectors.

Theory :



$$y_i = x_i W_{ij}$$

$$x_i = W_{ij} y_i^T$$

Bidirectional Associative Memory

(1) Bidirectional Associative Memory is a type of neural network used in associative memory.

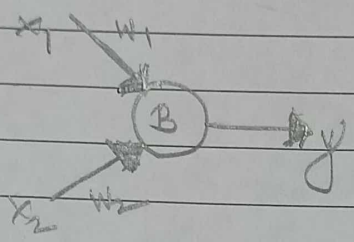
(2) When input pattern is given to network it can retrieve input pattern again based on output pattern.

(3) Algorithm : i Give input vector to input layer and set some target vectors.

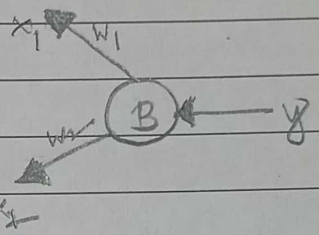
ii. Define the BAM function that takes inputs and weights. Compare aggregated and computed values with target vectors.

iii. Similarly all the vectors are matched and repeatedly inverse operation is done to get input vectors to get.

VPKBIET



Inputs operation



Reverse operation

Conclusion: BAM which is Bidirectional Associative Memory is a type of neural network which stores some value in memory for short time and then the output vectors can again retrieve the input vectors by following opposite direction.

CS 2014

VPKBIET

Assignment No: 7

Aim: Implement Artificial Neural Network training process in Python by using forward propagation & back propagation.

Theory: The training of ANN involves use of various backpropagation algorithm in which we modify the weights & biases so that error/loss gets minimized.

Forward Propogation: It is the process of calculating the outputs of the neural network from given inputs patterns. Vectors are fed to input layer then they are passed to hidden layer and then linearly ahead, then input follows a non-linear activation function. Then output of each layer is then passed as input to the next layer until the final output is obtained.

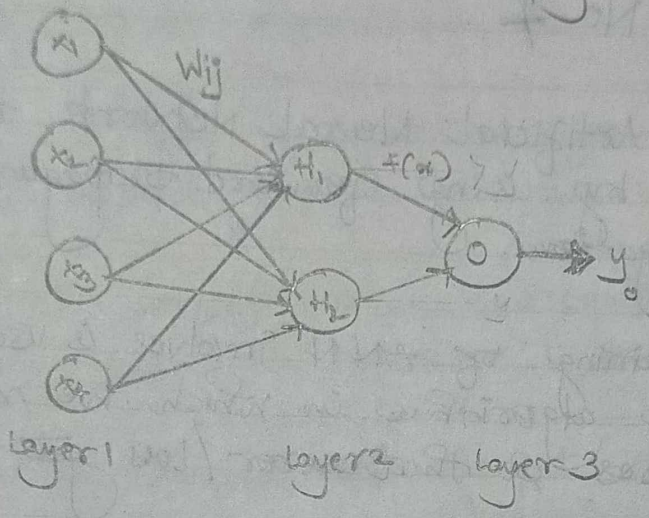
Backpropogation: In this process error are minimized by adjusting weights and biases. We use gradient descent algorithm to minimize errors and train the neural network.

Weights and biases are updated on the following formulas:

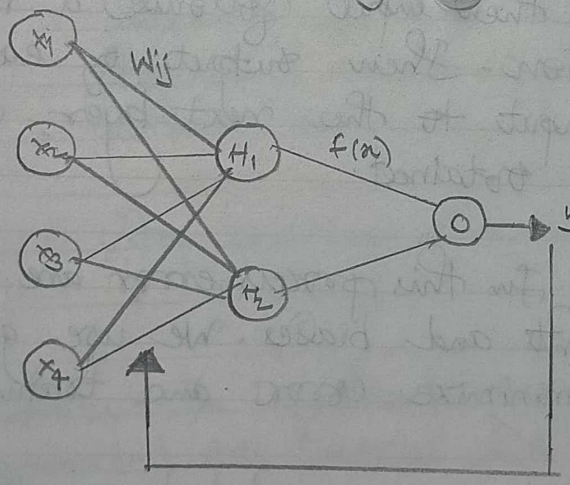
$$W_N = W_0 - \alpha \frac{\partial E}{\partial W_0}$$

$$B_N = B_0 - \alpha \frac{\partial B}{\partial B_0}$$

Foorward Propagation



Backpropagation



VPKBIET

Backpropagation Algorithm:

i. Take input vectors from inputs and feed it to first layer.

ii. Compute some aggregation and apply some activation function and get outputs.

iii. Compare with target vectors.

iv. minimize the errors with loss function

$$E = (O - y_T)^2$$

v. Adjusts the weights and biases accordingly. Stop when epochs cycle show minimum or desired error.

vi. Stop training the network.

Conclusion: 1. In forward propagation input vector are forwarded to layer 1 and then performing aggregation it passed to layer 2 and output is calculated.

2. In Backpropagation input is feed and aggregation is performed it is passed layer by layer and then we obtain some outputs. Outputs we got is compared with target vectors and likewise error is minimized.

VPKBIET

Assignment No. 8

Title: Write a python program to illuminate ART-1 Neural Network.

Theory: i. It is a type of self organizing neural network architectures that clusters the pattern space and produce archetypal weight vector templates.

ii. ART-1 has two layer networks that discovers pattern cluster templates in arbitrarily Boolean pattern sets.

iii. There are two layers inputs and cluster layer.

(1) Inputs Unit (F1 layer): It further has following two portions:

- F1 a layer Inputs portion: No processing in this portion rather than having input vectors only.

- F1 b layer Interface portion: Combines signal from the inputs portion with that of F_2 layer.

(2) Cluster Units (F_2 layer): This is competitive layer. The unit having largest net inputs is selected to learn the inputs pattern.

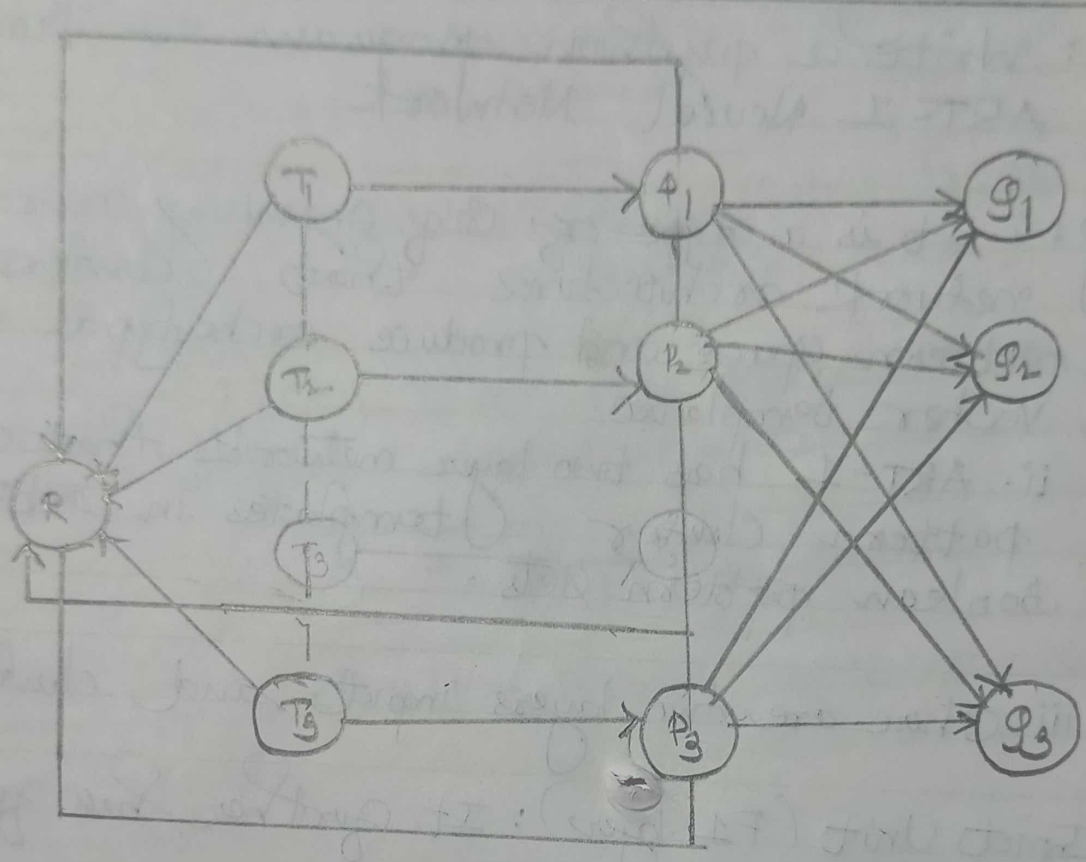


Fig. ART-1

VPKBIET

Parameter Used:

n = number of components in the input vector.

m = maximum number of clusters that can be formed.

b_{ij} = weights from f_1^b to f_2 layer, i.e. bottom up weights.

t_{ij} = weights from f_2 to f_1^b layer; i.e top down weights.

ρ = vigilance parameter.

$||x||$ = Norm of vector x.

Algorithm:

1. Initialize learning rate, vigilance parameters and the weights as follows

   $\alpha > 1$ and $0 < \rho \leq 1$

   $0 < b_{ij}(0) < \frac{\alpha}{\alpha - 1 + n}$ and $t_{ij}(0) = 1$

2. Continue Step 0-9, when the Stopping Condition is true.

(3) Continue Step 4-6 for every training input.

(4) Set activations of all, F_1 a and F_2 units as follows:

$f_1 = 0$ and $F_1 a$ = input vectors.

(5) Input signal from $F_1 a$, $F_1 b$ layer must be sent like

$s_i = x_i$

(6) For every inhibited F_2 node

$y_i = \sum b_{ij} x_i$ the condition is $y_j \neq -1$.

(7) Perform Step 8-10 when reset is true.

(8) Find J for $y_j > y_j$ for all nodes j

(9) Again calculate activation on F_2 b as follows

$x_i = s_i t_{ji}$

(10) Now after calculating the norm of vector X and vector s, we need to check the reset condition as follows.

If $\|x\| / \|s\| <$ vigilance parameter s.
then inhibit node J and go to Step 7.

Else, If $\|x\| \|s\| \geq$ vigilance parameters β. then proceed further.

(11) Weights updating for node J can be done as follows

$$b_{ij}(new) = \frac{\alpha x_i}{\alpha - 1 + \|x\|}$$

$$t_{ij}(new) = x_i$$

(12) The stopping condition for algorithm must be checked and it may be as follows.

- Do not have any change in weight.
- Reset is not performed for units.
- Maximum number of epochs reached.

Conclusion: ART1 neural network is for clustering only binary vectors and has processing speed.

VPKBIET

Assignment No. 9

Title: Write a Python program to implement CNN object detection. Discuss numerous performance evaluation metrics for evaluating the object detecting algorithms performance.

Theory: Object detection is the task of identifying and localizing objects of interests within an image or video. The task of object detection can be acheived using various methods, including deep learning based approaches such as convolutional Neural Network the goal is to train a CNN model to predict bounding, boxes and class labels for objects within image.

Algorithm for object detection:

1. Data Collection → collect image dataset.
2. Data Pre-processing → Normalization and Resizing of images.
3. CNN model selection → select pre trained CNN model.
4. Predict the accuracy of CNN and define the loss.

VPKBIET

Performance Evaluation Metrics:

(1) Mean Average Precision (MAP): It measures the average precision of the model at various recall levels.

(2) Intersection over Union (IOU): A measures of the overlap between the predicted bounding box and ground truth bounding box.

(3) Precision: Percentage of correctly detected objects out of all detected objects.

(4) Recall: Percentage of correctly detected objects out of all ground truth objects.

(5) F1-Score: Harmonic mean of precision & recall.

(6) False Negative Rate (FNR): Percentage of positive samples that are incorrectly classified are negative.

Conclusion: We have learnt to implement CNN for objects detection and also discussed about performance evaluation metrics for object detection.

Assignment No. 10.

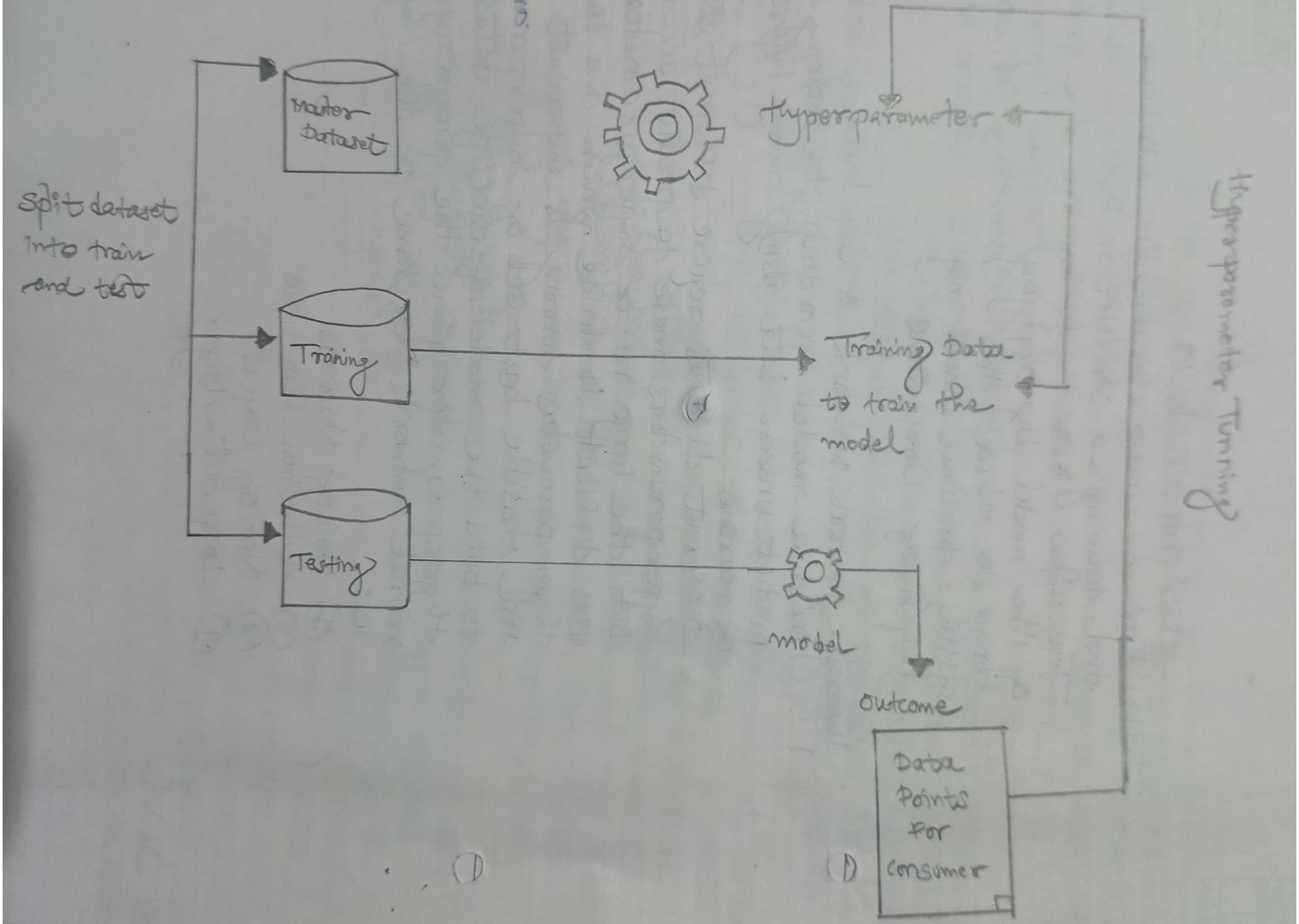
Title: For an image classification challenge, create and training a conVNet in Python using Tensorflow. Also try to improve the performance of the model by applying various hyper parameters tuning to reduce the overfitting or under fitting problem that might occur. Maintain graph of comparisons.

Theory: Neural Network is deep learning technic to build a model according to training data to predict unseen data using many layers consisting of neurons.

Different datasets require different sets of hyperparameters make to predict accurately. But the large no. of hyperparameters make it difficult to decide which one to choose. Hyperparameters tunning is important to find the possible best set of hyperparameters to build the model of specific dataset.

Hyperparameter given the performance of a neural network. Some of the hyperparameters as:

1. No. of Neurons.
2. Activation Functions.
3. No. of layers.
4. Drop outs layer.


Master Dataset
Split dataset into train and test
Training
Testing
Training Data to train the model
hyperparameter
model
outcome
Data Points for consumer
Hyperparameter Tuning

VPKBIET

① No. of Neurons: The no. of neurons governs with the complexity of the neural network structure. The task with a more complex level to predict needs more neurons.

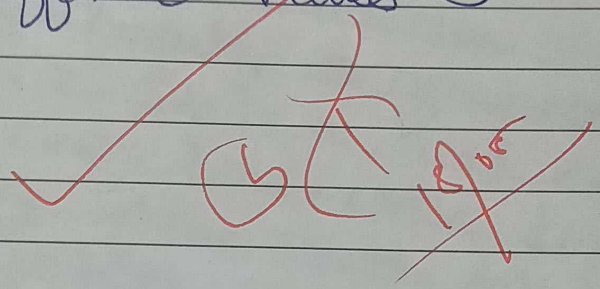
② Activation Functions: The input values moving from a layer to another layer keeps changing according to the activation functions. The activation functions decides how to compute the input values of layer into the output values.

③ No. of layers: Layers in the neural network also determine the result of the prediction model. A smaller number of layers is enough for a simpler problem, but a larger number of layers needed to build the model for a more complicated problem. The number of layers can be tuned using iterative loops.

④ Dropout Layer: The dropout layer, as the name suggests a certain number of neurons in a layer the rate of how much percentage of neurons in a layers. The rate of how much percentage of neurons to drop is set in dropout rate.

As seen above, the every parameter used in the important values that we set of affects the outputs result. Hyperparameters tunning, tunes these parameters so as to get the optimal values of these parameters & therefore get an optimal values of these parameters.

Conclusions: Hyper parameters turned are the dataset specific. The turning of hyperparameters is based and a dataset & its result. If we change the dataset, the hyperparameter may have a different values.



VPKBIET

Assignment No. 11

Title : MNIST Handwritten Character detection using Pytorch keras & Tensorflow.

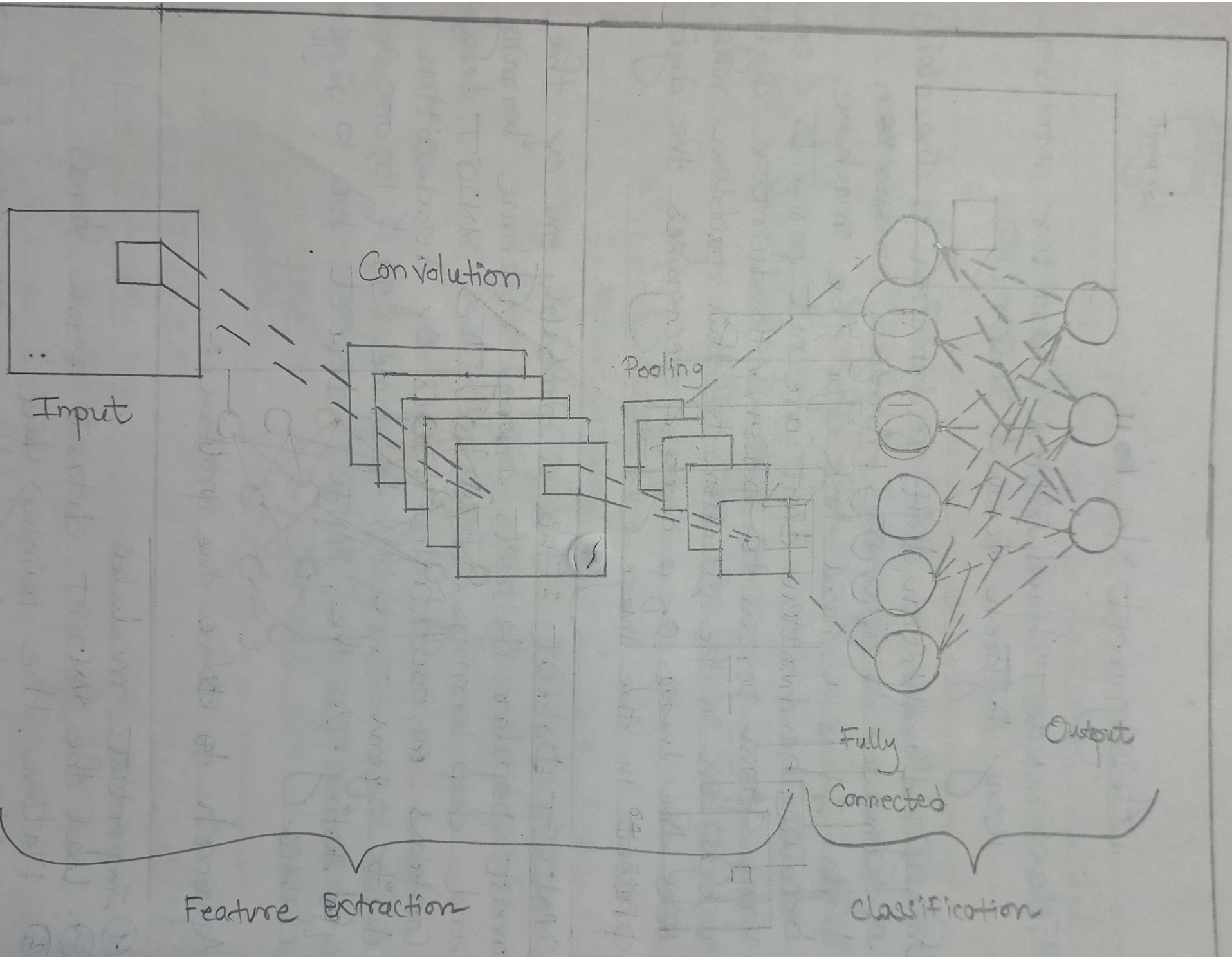
Theory : Handwritten digit recognition is the ability of computer to recognize human handwritten digits. It is a hard task for the machine because handwritten digit are not perfects & can vary from person to person. Handwritten digit recognition is the solution to this problem which uses the image of a digit & recognizes the digit present in the image.

MNIST Dataset : This is probably one of the most popular datasets among machine learning and deep learning enthusiasts. The MNIST dataset contains 60,000 traing images of handwritten digits from zero (0) to nine (9) & 10,000 images for testing. So the, MNIST dataset has 10 different classes.

Approach to solve the problem :-

1. Important modules.
2. Load the MNIST dataset from keras.
3. Flatten the training data :

We need to convert the 2-Dimensional input data into a single-dimensional format of feeding into a single-dimensional format for feeding into the model. This is the acheived by a


Convolution
Pooling
Input
Fully
Connected
Output
Feature Extraction
Classification

process from called flattening. In this process, the 28 x 28 grid image is converted into a single-dimensional array of 784 (28 x 28).

4. Create a simple naval network in keras
Create a simple single layer naval network & compute the accuracy.

5. Evaluate test accuracy:
Using the confusion Matrix

6. Add a hidden layer:
After adding hidden layer again evaluate the test accuracy. The accuracy changes with different activation, no. of neurons & no. of layers as well.

Then supply an individual image & test whether the model predict it accurately or not. In such a way, we can implement the handwritten character recognition.

Conclusion: The handwritten character recognition can be implemented in keras as well as CNN to get maximum accuracy & detect character accurately.