"AN APPLICATION APPROACH FOR "FRANK ROSENBLATT" NEURAL NETWORK PERCEPTION"

Praveen Kr. Malik, Ramesh Kumar, Shekhar Pundir*
RGEC, Meerut
SCRIET, Meerut*
malikbareilly@gmail.com

Abstract: This paper describes a robust, accurate, efficient, low-resource, medium-data, application-based method for using Frank Rosenblatt perception method for Neural Network. Frank Rosenblatt [1962] & Minsky & Papert [1988] developed large class of artificial neural network called perceptron. This perceptron learning rule uses an iterative method for weight adjustment that is more power full then any other previous one (Like Hebb Net OR Mc Culloch Pitts etc). In most of the experiment it has been kept in mind that how the adjustment of weights should be done according to the threshold output function & Mc Culloch Pitts model. But in most of the cases other condition like learning rate, Data type (Binary OR Bipolar) & other conditions are not considered. Here we have done some experiment on the learning rate, Data type & other conditions which show that if we consider these also, then the output of the perceptron is more optimized.

Keywords: - Rosenblatt Perception, Learning rate, Weights, input & output.

INTRODUCTION TO NEURAL NET

Development of the Artificial Neural Network (ANN's) is about 50 years old. Artificial neural network are the result of academic investigation that uses mathematical formulation to model nervous system operation. A neural network is used to learn patterns & relationship in data. Neural network do not require explicit coding of the problems. In fact they require raw data to be processed. ANN's are the signal processing system whose basic concept has been taken from the concept of the human brain with the basic element Neuron.

Elements of human neurons are: -

1. Dendrite's
2. Synapses
3. Axon
4. Soma (Cell body)

Basic functioning of biological neural network (human neuron) can be discussed with [1]

One important parameter of neural network is to learn the neuron. There are three major tech. to be used to learn the neuron.

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Training

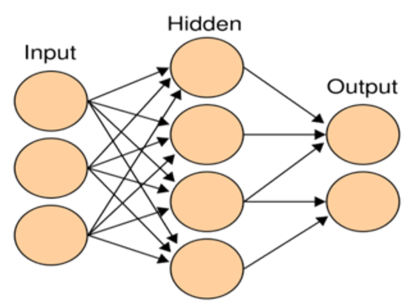
Learning methods can be discussed with [2], [3]

To take the decision of the output there are many activation function. These activation functions can be used according to our requirements. Major activation functions are: -

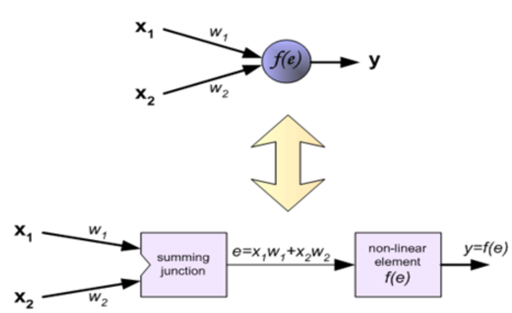
1. Identity Function
2. Binary Step Function
3. Signum Function
4. Signoidal Function
5. Tan Hyperbolic Function

Activation functions can be discussed with [4], [5]

Basic model of neural network is as shown: -



All the inputs coming to the neuron are multiplied with their synapse weights & added together as shown: -



This output is compared with the threshold set by the activation function & the decision is taken accordingly either 0 or 1 OR -1 or +1.

FRANK ROSENBLATT MODELING

Fundamentally Rosenblatt perceptron consist of a single neuron with adjustable weights & bias. Rosenblatt also found that if the pattern used to train the perceptron are drawn from two linearly separable classes, the perceptron algo converges and position the decision surface in the form of a hyper plane between the two classes. [6], [7]

This can be better explain with a example: - Let us take the two neuron at the input side & one neuron at the output side. Let the input are X1 & X2. Desired output is Y & computed output is Y1. The activation function for the output is unit step i.e. either 0 or 1. (i.e. either Y=0 or Y=1).

As output according to the Rosenblatt is given by

Y1 = X0*W0 + X1*W1 + X2*W2. (1)

(Initially X0 input is treated as 1 (i.e. Bias) & W0 is the weight for the bias)

If this output if equal to the desired output i.e. the weights (W1 & W2) chosen are correct. Else we have to modify the weights. There are two possibilities if Y1is not equal to Y.

If Y=1 & Y1=0

Wi(new) = Wi(old) + ηX(i) (2)

If Y=0 & Y1=1

Wi(new) = Wi(old) - ηX(i)(3)

Where η is the learning rate. Again these calculated Wi(new) & Wi(old) are treated as new weights & calculated the output. If the output is same as desired that means weights has been adjusted. Otherwise the procedure will be repeated. Till Y1=Y.

In this procedure it has been considered that the bias is 1, W(0) is fixed & the learning rate is also fixed (Mostly 1 or less then 1). In this paper, I have done some experiment & found that is we change the value of learning rate η or bias in some defined fashion then result will be very fruitful.

Algorithm for Rosenblatt perceptron [8]: -

1. Read the X(1), X(2)
2. Read the Bias X(0)
3. Read desired output Y
4. Read the initial weights W(0) & W(1)
5. Read the W(0)
6. Compute the output
   Y1 = X0*W0 + X1*W1 + X2*W2
7. If Y1==Y, end
8. If Y=1 & Y1=0
   Wi(new) = Wi(old) + ηX(i)
9. If Y=0 & Y1=1
   Wi(new) = Wi(old) - ηX(i)

Goto step 6.
end;

I have written a program in MATLAB to demonstrate the problem & analyze the result.

MATLAB Perception for Rosenblett: -

```
i=1;     j=1;
k=1;    l=1;
W1(1)=1;
W2(1)=0;
x0=input ('Enter Value of X0= ');
w0=input ('Enter Value of W0= ');
n=input ('Enter Value of rate of learning n = ');
x1=input ('Enter Value of First input  x1= ');
x2=input ('Enter Value of second input x2= ');
y=input ('Enter desired output Y= ');
x=1;
while x
x=0;
y1=w0*x0+x1*w1(i)+x2*w2(j);

if y1<=0
y1=0;
else
y1=1;
end

if y1==y
x=0;
end

if y==1 & y1==0
w1(k)=w1(i)+n*x1;
w2(l)=w2(j)+n*x2;
end

if y==0 & y1==1
w1(k)=w1(i)-n*x1;
w2(l)=w2(j)-n*x2;
end

k=i+1;      l=j+1;
end

disp(' ');
disp('Perception for Given Data : - ');
```

```
disp('Final Weight W1 & W2 will be : -');
disp(w1(i));
disp(w2(j));
```

As explained in the program first the value of x0 & w0 will be entered. Then the learning rate is entered followed by the value of input (x0 & x1) & the desired output (Either +1 or 0). Then according to the formula Y1, it is being calculated that the desired is same as of input or not. If yes, that means the weights are correct. Otherwise we have to adjust the weight according to the equation 2 & 3 and then the algorithm will calculate the new weight & put in equation 1 again & find the value of Y1. if Y1 is same as desired then it is OK. Otherwise repeat the same procedure until we get y1 = desired y.

In this paper, I have done some experiments on Rosenblatt perception & find the effect of the learning rate & the input data on the learning perception. I have also find a relation in between the calculated weights of the learning perception.

During the experiment it is being assumed that it is a two neuron network. Each having single input. Network is also single layer. As per the Rosenblatt learning perception, it also being confirmed that the output is also either +1 or 0. ie. If the output (Y1) is more then 0, Y = +1 & if the output (Y1) is less than 0, Y = 0.

Experiment 1: -

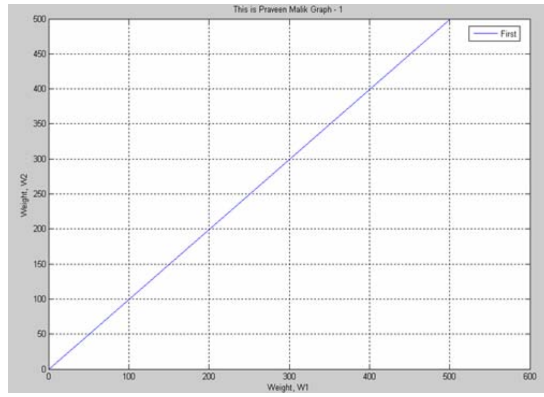
In the first experiment I have taken as x0 = 1, w0 = -2, x1 = 0.5, x2 =0.5 & Y = +1. I have changed the value of learning rate η. As per the experiment output is: -

And if we draw a graph in between the calculated

Table 1.
(If we increase learning rate while Y desired output is +1)

X0	W0	η	X1	X2	Y	W1	W2
1	-2	**0.00**	0.5	0.5	**+1**	1.000	0.000
1	-2	**0.25**	0.5	0.5	**+1**	1.125	0.125
1	-2	**0.50**	0.5	0.5	**+1**	1.250	0.250
1	-2	**0.75**	0.5	0.5	**+1**	1.375	0.375
1	-2	**1.00**	0.5	0.5	**+1**	1.500	0.500
1	-2	**2.00**	0.5	0.5	**+1**	2.000	1.000
1	-2	**3.00**	0.5	0.5	**+1**	2.500	1.500
1	-2	**4.00**	0.5	0.5	**+1**	3.000	2.000
1	-2	**5.00**	0.5	0.5	**+1**	3.500	2.500
1	-2	**10.00**	0.5	0.5	**+1**	6.000	5.000
1	-2	**100.00**	0.5	0.5	**+1**	51.000	50.000
1	-2	**1000.00**	0.5	0.5	**+1**	501.000	500.000

weights i.e. w1 &w2.



This graph shows that the relations between the new calculated weights are linear. i.e. if we change the learning rate η, on the same data x1& x2, it will give a proportional changes of the value on both the weight. Increase in the learning rate will not make an abrupt effect on any of the weight.

Experiment 2: -

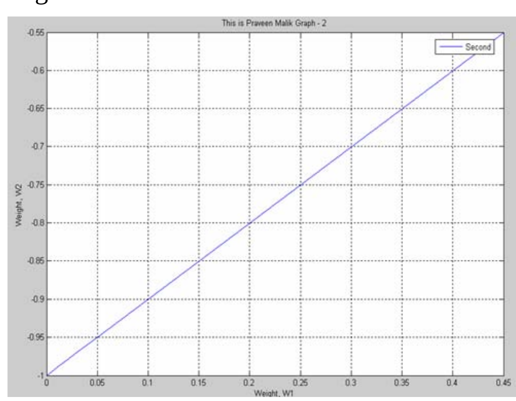
In the second experiment, I have taken the same value of x0 = 1, w0 = -2, x1 = 0.5, x2 =0.5 but Y = 0. I have changed the value of learning rate η. As per the experiment output is : -

And if we draw a graph in between the calculated

Table 2
(If we increase learning rate while Y desired output is 0)

X0	W0	η	X1	X2	Y	W1	W2
1	-2	**0.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**0.25**	0.5	0.5	**0**	1.000	0.000
1	-2	**0.50**	0.5	0.5	**0**	1.000	0.000
1	-2	**0.75**	0.5	0.5	**0**	1.000	0.000
1	-2	**1.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**2.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**3.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**4.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**5.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**10.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**100.00**	0.5	0.5	**0**	1.000	0.000
1	-2	**1000.00**	0.5	0.5	**0**	1.000	0.000

weights i.e. w1& w2.



Again it has been noted that the output is linear as far as the value of w1 & w2 are concern. But one point is there which can be also be noticed that, If we keep the same input data but the output is zero. i.e. Y=0. the weights are constant. i.e Rosenblatt perception is very optimized & linear if the output is assumed to be 0. It is very important to note that there is no effect of the learning rate. Or we can also say that if the output is zero, there is no need for the learning rate. It is similar to the concept the if any person don't want to do something there will be no effect of the own learning rate.

Experiment 3: -

I have also done my experiment on some random data, which also reflect that the changes in the weights are

Table 3. Random Value: -

X0	W0	η	X1	X2	Y	W1	W2	
2	-5	0.5	2	2	+1	2	1	
2	-5	0.5	2	2	0	1	0	
-2	5	0.5	2	2	+1	2	1	
-2	5	0.5	2	2	0	1	0	
1	1	0.5	1	1	+1	1	0	(true)
1	1	0.85	1	1	+1	1	0	(true)

linear.

Experiment 4: -

Main beauty of this paper is fourth experiment. In this experiment, I have taken some small & equal values of the input data & initial weights. Collected data for this experiment is a follows: -

Here, if I consider x0 = 1, w0 = 1, x1 = 1, x2 = 1,

Table 4:

For a particular case: -

X0	W0	η	X1	X2	Y	W1	W2	
1	1	0.5	1	1	0	0.50	-0.50	(false)
1	1	1.00	1	1	0	0	-1	(true)

i.e. the learning rate should be in between 0.50 & 1.00

X0	W0	η	X1	X2	Y	W1	W2	
1	1	0.55	1	1	0	0.45	-0.55	(false)
1	1	0.75	1	1	0	0.25	-0.75	(false)
1	1	0.85	1	1	0	0.15	-0.85	(false)
1	1	0.90	1	1	0	0.10	-0.90	(false)
1	1	0.95	1	1	0	0.05	-0.95	(false)
1	1	0.96	1	1	0	0.04	-0.96	(false)
1	1	0.97	1	1	0	0.03	-0.97	(false)
1	1	0.98	1	1	0	0.02	-0.98	(false)
1	1	0.99	1	1	0	0.01	-0.99	(false)
1	1	1.00	1	1	0	0	-1	(true)

learning rate = 0.05 & Y=0 (desired). Then as per the perception final weights are not true. It gives w1 = 0.50 & w2 = -0.50. While on the other hand, if I increase the learning rate from 0.5 to 1.00, output is true i.e it changes the weights from 0.50 & -0.50 to 0 & -1, which gives the correct value of Y1 as per the eq. (1). i.e. it can learn if the rate of learning of 1 or more then 1. I also increase the value learning rate from 0.50 to 1.00 gradually. But I found that the perception does not learn till 1.00. On 0.99 learning rate output is **false** while on 1.00 learning rate output is **true**.

RESULT

As we know that that learning rate should be in between 0 & 1 for any learning perception. But here, I would like to make some changes according to my experiments. Let us consider an example, that if any teacher is taking the class & a student is listening his/her lecture, then the knowledge he/she may acquire will depend on his/her learning rate. i.e. If his/her learning rate is 0 he cannot understand the lecture (Even he cannot listen the lecture also). But if student pay his full attention towards the lecture then there is no possibility that he/she cannot understand the lecture. As in the human being we can not pay the attention more then of our learning rate 1 (i.e. 100 % attention). But in the neural network it is possible that we can give the learning rate either 1 or more then1 also. And we have proved it with our last example also, that if the learning rate is small then it might possible that system cannot learn over the small data. While on the other hand if we increase the learning rate then system can learn true over the same data.

CONCLUSION

1. It has been noticed in the previous four experiments that the output relation between the final weights of the Rosenblatt perception is linear .i.e. one weight should not try to put more load on the other one, if the desired output is considered to be +1. While on the hand both weights are constant & give a linear relation between them if the desired output is 0.
2. It has also been shown that the learning rate for generally small value of input should be 1 or more then 1.

REFERENCE

1. J A Anderson & E Rosenfeld, "Neuro computing: Foundation of research", Cambridge, MA:MIT Press 1989
2. P. Rujan. A fast method for calculating the perceptron with maximal stability. J. Phyicss.13:277-290, 1993.
3. L. Villalobos and F. L. Merat. Learning capabiUty assessment and feature space optimization for higher-order neural networks. IEEE Trans. Neural Networks 6:267-272, 1995.

4. S. I. Gass. Linear Programming: Methods and Applications. McGraw-Hill, New York, 1985.
5. K. G. Murty. Linear Programming. John Wiley, New York, 1983.
6. F. Rosenblatt. Principles ofNeurodynamics. Spartan, New York, 1962.
7. F. Rosenblatt. The perceptron: "A probabilistic model for information storage and organization in the brain. Psychol. Rev. 65:386-408, 1958.
8. F. Rosenblatt. "Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms".Spartan Books, Washington, DC, 1961.