*International Journal of Latest Trends in Engineering and Technology*
*Special Issue SACAIM 2016, pp. 567-571*
*e-ISSN:2278-621X*

# COMPARATIVE STUDY ON ARTIFICIAL NEURAL NETWORKS ADALINE, MADALINE AND HEBB'S

Osborne D'Silva[1], Shanoob K.V[2] and Mithun Dsouza[3]

Abstract: In many real-world applications, we want our computers to perform complex pattern recognition problems, such as the one just termed. Since our systems are obviously not suited to this type of problem, we therefore borrow features from the physiology of the brain as the basis for our new processing models. Hence, the technology is known as artificial neural systems (ANS) technology, or simply neural networks. Various algorithms are used for the analysis of artificial neural networks. In this paper we are investigating different algorithms used in artificial neural networks and the applications of ANS.

Keywords: Artificial Intelligence, Machine Learning, Algorithms, Neural Computing, Pattern Recognition

## I. INTRODUCTION

Artificial neuron networks are the computational model based on the structure and functions of biological neural networks. These crude electronic models basically learns from the experiences andit is very good at a wide variety of problems, most of which involve finding trends in large quantities of data. These biologically methods of computing are assumed to be the next most importantimprovement in the computing industry. There are many algorithms used for analysing the artificial network. This paper describes and examines the various algorithms used in the artificial neural network.

## II. PROBLEM STATEMENT

- The perceptrons were not capable of implementing certain elementary functions like XOR
- Evaluates Performance of learning algorithm
- A comparative study of learning techniques as the Adaline,Madaline and Hebbs.
- We will conclude on an algorithm that is better for learning techniques.

## III. PROPOSED ALGORITHM

### *A) THE ADALINE (Adaptive Linear Neuron or Adaptive Linear Element)*

The Adaline is a single layer neural network with multiple nodes where each node accepts multiple inputs and generates one output. It consists of a weight, a bias and a summation function. In the learning phase the weights are adjusted according to the weighted sum of the inputs (the net).

[1] *Aloysius Institute of Management & Information Technology(AIMIT), St.Aloysius College(Autonomous) Mangaluru, Karnataka - 575022*
[2] *Aloysius Institute of Management & Information Technology(AIMIT), St.Aloysius College(Autonomous) Mangaluru, Karnataka - 575022*
[3] *Aloysius Institute of Management & Information Technology(AIMIT), St.Aloysius College(Autonomous) Mangaluru, Karnataka - 575022*

1)The Adaine Algorithm:

Steps:

1. Weights and bias are set to some random fraction values but not to zero. Set the learning rate α.
2. Perform steps 3 to 7 till the stopping condition is false
3. Perform steps 4 to 6 for each bipolar training pair(s:t)
4. The input layer containing the input units is applied with activation function $x_i=s_i$
5. Calculate the net input to output of the network

$Y_{in} = b + \sum x_i w_i$

where i=1 to n and n is the number of input neurons at input layer.

Apply the activation function

$Y=f(y_{in}) =$ 1 if $y_{in} > \theta$

0 if $y_{in} = \theta$

-1 if $y_{in} < \theta$

6. Update the weights and bias

$w_i(new)=w_i(old)+ \alpha\ (t-y_{in})\ x_i$

$b_i(new)=b_i(old)+ \alpha\ (t-y_{in})$

7. Train the network until there is no weight change. This is the stopping condition for the network. If it is not met then start again from step number 2.

*B) THE MADALINE(Multiple Adaptive Linear Neuron )*

It is a three-layer (input, hidden, output), fully connected, feed-forward artificial neural network architecture for classification that uses ADALINEunits in its hidden and output layers, i.e. its activation function is the sign function. More than one adaline forms the resultant network.

X1 and x2 are the two input neurons connected with the neurons in the hidden layer z1 and z2 the result of this network represented in the form of observed output(y).z1 and z2 represent output at the hidden layer connected with the bias b1 and b2.similarlly bas b3 is connected with y.The network is combination of static and dynamic approach. The second part of network is static thus by reducing its weighted bias less by 50%.

1)The Madaine Algorithm:

Steps:

1. Weights and bias are set to some random fraction values but not to zero. Set the learning rate α.
2. Let $v_1 = v_2 = b_3= 0.5$
3. Perform steps 3 to 9 till the stopping condition is false
4. Perform steps 4 to 8 for each bipolar training pair(s:t)
5. The input layer containing the input units is applied with activation function $x_i=s_i$
6. Calculate the net input to output at hidden layer of the network

$Z_{in1}= b_1 + x_1 w_{11} + x_2w_{21}$

$Z_{in2}= b_2 + x_1 w_{12} + x_2w_{22}$

Apply the activation function

$Z_1= f(z_{in1}) =$ 1 if $z_{in1} > \theta$

$-1$ if $z_{in1} <= \theta$

$Z_2 = f(z_{in2}) =$ $1$ if $z_{in2} > \theta$

$-1$ if $z_{in2} <= \theta$

7. Calculate the actual output

$Y_{in} = b_3 + z_1 v_1 + z_2 v_2$

where n is the number of input neurons at input layer.

Apply the activation function

$Y = f(y_{in}) =$ $1$ if $y_{in} > \theta$

$-1$ if $y_{in} <= \theta$

8. Find the error and update the weights and bias

If t=y, then

$w_{ij}(new)=w_{ij}(old)$

$b_j(new)=b_j(old)$

else if t != y, then

if(t=1) then,

$w_{ij}(new)=w_{ij}(old)+ \alpha (1-z_{ini}) x_i$

$b_i(new)=b_i(old)+ \alpha (1-z_{ini})$

else if(t = -1) then,

$w_{ij}(new)=w_{ij}(old)+ \alpha (-1-z_{ini}) x_i$

$b_i(new)=b_i(old)+ \alpha (-1-z_{ini})$

9. Train the network until there is no weight change. This is the stopping condition for the network. If it is not met then start again from step number 2.

*C) The Hebb's Network*

The Hebb rule determines the change in the weight connection from ui to uj by Dwij = r * ai * aj, where r is the learning rate and ai, aj represent the activations of ui and uj respectively. Thus, if both ui and uj are activated the weight of the connection from ui to uj should be increased.

1)The Hebb's Algorithm:

Steps:

1. Initialize the weights. Initially they have to be set to zero; $w_i$=0, bi=0, n is the number of input patterns.
2. Steps 3 to 5 have to be performed for each training vector and target output pair(s:t)
3. Input unit activations are set $x_i=s_i$
4. Output unit activations are set to y=t
5. Adjust the weights.

$w_i(new)=w_i(old)+x_i.y$

6. Update the bias

$b_i(new)=b_i(old)+y$

## IV.EXPERIMENTAL RESULTS

### Adaline Network

Enter the value for x1 & x2

1 1
1 -1
-1 1-1 -1
The result of weights is

| x1 | x2 | y | t | dw1 | dw2 | db | w1 | w2 | b | (t-y)^2 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0.600 | -1.6 | -0.32 | -0.32 | -0.32 | -0.12 | -0.12 | -0.12 | 2.560 |
| 1 | -1 | -0.12 | 1.12 | 0.224 | -0.22 | 0.224 | 0.103 | -0.34 | 0.103 | 1.254 |
| -1 | 1 | -0.34 | -0.65 | 0.131 | -0.13 | -0.13 | 0.235 | -0.47 | -0.02 | 0.430 |
| -1 | -1 | 0.212 | -1.21 | 0.242 | 0.242 | -0.24 | 0.477 | -0.23 | -0.26 | 1.470 |

The sum of (t-yin)^2 is5.715

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -0.02 | -0.97 | -0.19 | -0.19 | -0.19 | 0.282 | -0.42 | -0.46 | 0.951 |
| 1 | -1 | 0.245 | 0.754 | 0.150 | -0.15 | 0.150 | 0.433 | -0.57 | -0.31 | 0.569 |
| -1 | 1 | -1.32 | 0.326 | -0.06 | 0.065 | 0.065 | 0.368 | -0.51 | -0.24 | 0.106 |
| -1 | -1 | -0.10 | -0.89 | 0.179 | 0.179 | -0.17 | 0.547 | -0.33 | -0.42 | 0.803 |

The sum of (t-yin)^2 is8.145

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -0.21 | -0.78 | -0.15 | -0.15 | -0.15 | 0.390 | -0.49 | -0.58 | 0.617 |
| 1 | -1 | 0.296 | 0.703 | 0.140 | -0.14 | 0.140 | 0.531 | -0.63 | -0.44 | 0.494 |
| -1 | 1 | -1.60 | 0.607 | -0.12 | 0.121 | 0.121 | 0.409 | -0.51 | -0.32 | 0.369 |
| -1 | -1 | -0.22 | -0.77 | 0.155 | 0.155 | -0.15 | 0.565 | -0.35 | -0.47 | 0.604 |

The sum of (t-yin)^2 is10.23

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -0.26 | -0.73 | -0.14 | -0.14 | -0.14 | 0.418 | -0.50 | -0.62 | 0.535 |
| 1 | -1 | 0.295 | 0.704 | 0.140 | -0.14 | 0.140 | 0.559 | -0.64 | -0.48 | 0.496 |
| -1 | 1 | -1.68 | 0.685 | -0.13 | 0.137 | 0.137 | 0.422 | -0.50 | -0.34 | 0.470 |
| -1 | -1 | -0.26 | -0.73 | 0.147 | 0.147 | -0.14 | 0.569 | -0.35 | -0.49 | 0.541 |

The sum of (t-yin)^2 is12.27

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -0.28 | -0.71 | -0.14 | -0.14 | -0.14 | 0.426 | -0.50 | -0.63 | 0.515 |
| 1 | -1 | 0.290 | 0.709 | 0.141 | -0.14 | 0.141 | 0.568 | -0.64 | -0.49 | 0.503 |
| -1 | 1 | -1.70 | 0.707 | -0.14 | 0.141 | 0.141 | 0.426 | -0.50 | -0.35 | 0.499 |
| -1 | -1 | -0.27 | -0.72 | 0.144 | 0.144 | -0.14 | 0.570 | -0.35 | -0.49 | 0.520 |

The sum of (t-yin)^2 is14.31

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -0.28 | -0.71 | -0.14 | -0.14 | -0.14 | 0.427 | -0.50 | -0.64 | 0.510 |
| 1 | -1 | 0.287 | 0.712 | 0.142 | -0.14 | 0.142 | 0.570 | -0.64 | -0.49 | 0.507 |
| -1 | 1 | -1.71 | 0.712 | -0.14 | 0.142 | 0.142 | 0.427 | -0.50 | -0.35 | 0.507 |
| -1 | -1 | -0.28 | -0.71 | 0.143 | 0.143 | -0.14 | 0.571 | -0.35 | -0.49 | 0.513 |

The sum of (t-yin)^2 is16.35
2.430
2.085
2.044
2.039
2.039

**Madaline Network**

Enter value for x1 and x2

| | |
|---|---|
| 1 | 1 |
| 1 | -1 |
| -1 | 1 |
| -1 | -1 |

| x1 | x2 | t | Zin1 | Zin2 | Z1 | Z2 | Yin | Y | W11 | W21 | b1 | W12 | W22 | b2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -1 | 0.55 | 0.45 | 1 | 1 | 1.5 | 1 | -0.725 | -0.524 | -0.475 | -0.675 | -0.524 | -0.575 |
| 1 | -1 | 1 | -0.675 | -0.725 | -1 | -1 | -0.5 | -1 | 0.1125 | -1.387 | 0.3625 | 0.1624 | -1.387 | 0.2875 |
| -1 | 1 | 1 | -1.137 | -1.262 | -1 | -1 | -0.5 | -1 | -0.956 | -0.256 | 1.4312 | -0.906 | -0.256 | 1.4187 |
| -1 | -1 | -1 | 2.6437 | 2.5812 | 1 | 1 | 1.5 | 1 | 0.8656 | 1.5343 | -0.390 | 0.9156 | 1.5343 | -0.371 |

**Hebb's Network**

**ANDNOT**

Enter value for x1 and x2

| | |
|---|---|
| 1 | 1 |
| 1 | -1 |
| -1 | 1 |
| -1 | -1 |

The final output

| x1 | x2 | y | dw1 | dw2 | db | w1 | w2 | b |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | -1 | 1 | 1 | -1 | 1 | 0 | -2 | 0 |
| -1 | 1 | -1 | 1 | -1 | -1 | 1 | -3 | -1 |
| -1 | -1 | -1 | 1 | 1 | -1 | 2 | -2 | -2 |

## V. CONCLUSION

The Essential requirement of Artificial Neural Networks is to test the response of the neurons. The experimental results shows that MADALINE has good features regarding the response of the neurons because it is a combination of multiple ADALINE networks.

## REFERENCES


[1] Bumptrees for Efficient Function, Constraint, and Classification by Stephen M. Omohundro, International Computer Science Institute http://nips.djvuzone.org/djvu/nips03/0693.djvu

[2] GA-RBF A Self-Optimising RBF Network by Ben Burdsall and Christophe Giraud-Carrier http://citeseer.nj.nec.com/71534.html

[3] Improving Classification Performance in the Bumptree Network by optimising topology with a Genetic Algorithm by Bryn V Williams, Richard T. J. Bostock, David Bounds, Alan Harget http://citeseer.nj.nec.com/williams94improving.html

[4] Evolving Fuzzy prototypes for efficient Data Clustering by Ben Burdsall and Christophe Giraud-Carrier http://citeseer.nj.nec.com/burdsall97evolving.html

[5] Neural Networks by Christos Stergiou and Dimitrios Siganos http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html