Novel Ceiling Neuronal Model: Artificial Neural Networks

by

Garimella Ramamurthy

Report No: IIIT/TR/2015/-1

Centre for Security, Theory and Algorithms
International Institute of Information Technology
Hyderabad - 500 032, INDIA
September 2015
Novel Ceiling Neuronal Model
Artificial Neural Networks

Sai Dileep Munugoti∗, Anil Rayala†, Rama Murthy Garimella‡
∗Indian Institute of Technology, Guwahati
d.munugoti@iitg.ernet.in
†International Institute of Information Technology
anil.rayala@students.iiit.ac.in, rammurthy@iiit.ac.in

Abstract—In this research paper, a novel neuronal model named “Ceiling Neuron” has been proposed. Learning algorithm and proof of convergence of such a model are discussed. An activation function which mimics the ceiling behavior is proposed for Multi - Layer Neural Networks. The dynamic nature of the proposed activation function is discussed. An associative memory based on Ceiling neuron is proposed and its dynamics are studied. The proposed model is tested on MNIST dataset and results are shown.

Index Terms—Ceiling Neuron, Mutli-Layered Networks, Dynamic Activation Function, Associative Memory

I. INTRODUCTION

Early efforts at emulating the biological neural networks resulted in several models of artificial neuron. One of the interesting models of artificial neuron is the McCulloch - Pitts model. This model utilizes the inputs, \((x_i)_{i=1}^{N}\) and synaptic weights \((w_j)_{j=1}^{N}\) in the following manner

\[ Y = Signum\left\{ \sum_{j=1}^{N} W_j x_j - T \right\} \]

\[ Net = \sum_{j=1}^{N} W_j x_j - T; Y = Sign(Net) = \begin{cases} +1 & \text{if } Net \geq 0 \\ -1 & \text{if } Net < 0 \end{cases} \]

In this model, the so called activation function is the signum function. The net contribution is the weighted sum of inputs (the weights being the synaptic weights) minus the threshold value. As depicted in the above figure, the output of M-P neuron is the net contribution operated on by the signum function. This model of neuron is the basis of single layer perceptron (one of the first Artificial Neural Network models).

Motivated by the fact that biological neuron potentially utilizes multiple threshold values, we propose a novel model of artificial neuron in this research paper. Detailed description of “Ceiling” neuron model is provided in Section - II.

For the ceiling neuron to operate in Multi-Layer feed forward networks, we propose an activation function which has a dynamic nature and mimics the ceiling action in some cases. Several other researchers have proposed activation functions with dynamic nature [4]. The proposed activation function has nice properties which are depicted in Section-V.

This research paper is organized as follows, in Section-II, the ceiling neuron along with its learning algorithm is explained in detail. In section- III, proof of convergence of the proposed learning algorithm is given. In section - IV, an activation function which mimics the ceiling behavior and is continuous is proposed. In section - V, the dynamic nature of the proposed activation function, discussed in section - IV is shown. In section - VI, an associative memory based on ceiling neuron is proposed and its dynamics are discussed. Single hidden layered feed forward neural network with hidden units as ceiling neurons is tested on MNIST data and the results are shown in section - VII. The paper concludes in section- VIII.

II. CEILING NEURON : REACTION TO PERCEPTRON

The ceiling neuron is an artificial neuron very similar to a perceptron. Both have real valued input(s), synaptic weight(s), and threshold(s). The main difference between them is that while a perceptron has a single real valued threshold, the ceiling neuron has multiple real valued thresholds. When presented with an input, the ceiling neuron categorizes the input with respect to each threshold and outputs number of thresholds the given input is ‘above’.

Hence the output of such a neuron is an integer value ranging from 0 to number of thresholds. Consider a ceiling neuron
with 'n' thresholds. If the thresholds of the neuron are \( t_1, t_2 \ldots t_n \), the output of the ceiling neuron is given by

\[
y = \sum_{i=1}^{n} f(W.X - t_i)
\]

Where the function \( f(a) \) outputs '1' if \( a \geq 0 \) and outputs '0' if \( a < 0 \).

A single ceiling neuron divides the region bounded by a hyper plane into \( n+1 \) sections where 'n' is the number of thresholds the neuron has. To implement the same action, one would need 'n' perceptrons.

The proposed idea of Ceiling neuron based artificial networks was first reported in [1].

A. Notational Information

At this stage, it would be helpful to define a standard mathematical notation that will be used throughout this paper and any related papers that may follow. Complying with the commonly used notation, the thresholds will now be talked about in terms of 'biases'. It should be noted that 'bias' is simply the negative of threshold.

A typical ceiling neuron has 'm' inputs, 'm' weights, one associated to each input, 'n' thresholds/biases, and one output. The Synaptic weight vector \( W \) is a column vector comprising of the weights associated to each synapse, i.e. \( W = [w_1, w_2 \ldots w_m]^T \). Similarly, the input vector \( X \) is a column vector consisting of the inputs, i.e. \( X = [x_1, x_2 \ldots x_m]^T \). The thresholds/biases are denoted by \( b_1, b_2 \ldots b_n \). The notation 'step (x)' will be used to imply the step function action, i.e.

\[
\text{step}(x) = 0; x < 0 \\
1; x \geq 0
\]

B. Learning Algorithm for Ceiling Neuron

The process of teaching the neuron/neural network or making the neuron/neural network learn refers to the process of finding the correct weights and biases/thresholds so that the training data are correctly classified with respect to their expected outcomes. This is done by randomly selecting a set of values for the weights and biases and then, updating them according to a learning rule, with respect to every misclassified data point, in multiple iterations, till there are no misclassified data points. This process is same even for the perceptron.

The learning rule is a set of update rules for each weight and bias. For a typical ceiling neuron, let \( W^* \) imply the new/updated weight vector, \( W \) be the current weight vector, and \( X \) be the current data point. Similarly, \( B^* \) denotes the new/updated bias vector and \( B \) means the current bias vector.

The letter \( \eta \) refers to the learning rate. It must be carefully noted that 'd' implies the desired output of the neuron for that particular data point \( X \) and 'o' is the observed output of the neuron for that particular \( X \). On the other hand, 'D' denotes the vector composed of the desired classification value of that particular \( X \) with respect to all thresholds/biases and 'O' refers to the vector composed of observed classification value of that particular \( X \) with respect to all thresholds/biases.

We propose (and later justify) the following learning rule:

For every data point, the values of weights and biases are updated as

\[
W^* = W + \eta(d - o)X \\
B^* = B + \eta(D - O)
\]

Note: By “classification value with respect to \( b_i \)”, we mean to indicate where the data point \( X \) lies with respect to the line \( W.X + b_i = 0 \). If \( X \) lies in the region \( W.X + b_i \geq 0 \), it is classified as ‘1’ and if it lies within \( W.X + b_i < 0 \), it is classified as ‘0’. In short,

“classification value with respect to \( b_i \)’ = \text{step} (W.X + b_i)

It should be observed that the terms ‘d-o’ and ‘D - O’ make sure the updating occurs only for the misclassified data points. They become zero for correctly classified data points. Also observe that, in the training data, the desired neuron outputs are known, but the desired classification value w.r.t every bias is not known. This issue will be elaborated on in the next section.

C. Proof of Convergence For the Learning Algorithm

Now, we prove that the above proposed learning algorithm makes the neuron (network) converge after finite iterations. Because of the similarities between the learning algorithms for Rosenblatt’s perceptron and our ceiling neuron, the proof of convergence also follows similar lines.

As mentioned earlier, the letter ‘D’ refers to the vector of desired classification values and ‘O’ means the vector of observed classification values. These vectors are not apparent, as they are not a part of the learning data. But they can be found with relative ease by taking a zero vector of length ‘n’ (equal to the number of biases) and setting the first ‘d_1’ or ‘o_1’ elements as ones. Note that this is applicable only when the initial biases are in ascending order. Since we will take them as zeros, there is no issue with the above convention.

We will use \( B \) to denote the bias vector. Wherever appropriate, we will use \( H \) to denote the concatenation of the weight and bias vectors i.e. \( H = [W \ B]^T \). By concatenation, we mean the bias vector is appended below the weight vector resulting
in a column vector of length m + n. We assume that W and B are initially zero vectors i.e. \( W_1 = \text{zeros} \) (m, 1) & \( B_1 = \text{zeros} \) (n, 1). Let \( W_{k+1} \) and \( B_{k+1} \) represent the weight vector and the bias vector obtained when updated k times i.e. with respect to k wrongly classified data points. The norm of \( X \) is denoted by \( \|X\| \).

\[
W_{k+1} = W_k + \eta(d_k - o_k)X_k \\
B_{k+1} = B_k + \eta(D_k - O_k)
\]

(1)

i.e.

\[
W_{k+1} = \eta(d_1 - o_1)X_1 + \eta(d_2 - o_2)X_2 + \eta(d_3 - o_3)X_3 + \ldots + \eta(d_k - o_k)X_k
\]

(3)

and

\[
B_{k+1} = B_k + \eta(D_1 - O_1) + \eta(D_2 - O_2) + \eta(D_3 - O_3) + \ldots + \eta(D_k - O_k)
\]

(4)

Since the data, on which the ceiling neuron operates, is assumed to be linearly separable, there exists infinitely many solutions which correctly classify the data. Let \( (W_0, B_0) \) be one such solution.

Let us consider the value of \( H_0^T H_{k+1} \) which is equivalent to \( W_0^T W_{k+1} + B_0^T B_{k+1} \). From (3) and (4), it can be seen that,

\[
W_0^T W_{k+1} + B_0^T B_{k+1} = \eta(d - o)W_0^T X_p + \eta B_0^T (D - O) + \eta(d - o)W_0^T X_2 + \eta B_0^T (D - O) + \ldots + \eta(d - o)W_0^T X_k + \eta B_0^T (D - O)
\]

Let us look at \( p^{th} \) term in the above expression, i.e.

\[
\eta(d - o)W_0^T X_p + \eta B_0^T (D - O).
\]

Since the choice \( (W_0, B_0) \) correctly classifies the data, they correctly classify the point \( X_p \). Hence, it can be easily observed that the value of \( \eta(d - o)W_0^T X_p + \eta B_0^T (D - O) \) is always a positive value. Since it is satisfied for all \( p \), we can say that the minimum value of all the terms of \( W_0^T W_k + B_0^T B_k \) is also positive. Let the minimum value be represented by \( \alpha \). Hence,

\[
W_0^T W_{k+1} + B_0^T B_{k+1} \geq k\alpha
\]

The above equation can be rewritten as

\[
H_0^T H_{k+1} \geq k\alpha
\]

Now we make use of an inequality called Cauchy - Schwarz inequality. Given two vector \( H_0 \) and \( H_{k+1} \), the Cauchy - Schwarz inequality states that

\[
\|H_0\|^2 \|H_{k+1}\|^2 \geq \|H_0^T H_{k+1}\|^2
\]

(5)

Where \( \|\cdot\| \) denotes the Euclidean norm of the vector enclosed within and the inner product \( H_0^T H_{k+1} \) is a scalar quantity. Equation (5) can be written as,

\[
\|H_0\|^2 \|H_{k+1}\|^2 \geq k^2 \alpha^2
\]

Or, equivalently,

\[
\|H_{k+1}\|^2 \geq (k^2 \alpha^2)/\|H_0\|^2
\]

(6)

We now follow another development route,

Consider equation (1) and (2),

\[
W_{s+1} = W_s + \eta(d_s - o_s)X_s \\
B_{s+1} = B_s + \eta(D_s - O_s)
\]

Now let us define a vector \( Q_s \) which is the concatenation of \( (d_s - o_s)X_s \) and \( (D_s - O_s) \), i.e \( Q_s = [(d_s - o_s)X_s \ (D_s - O_s)]^T \). Hence we can combine the equations (1) and (2) as

\[
H_{s+1} = H_s + \eta Q_s
\]

By taking the squared Euclidean norm on both sides, we obtain,

\[
\|H_{s+1}\|^2 = \|H_s\|^2 + \eta^2 \|Q_s\|^2 + 2\eta H_s^T Q_s
\]

(7)

It can be seen that \( H_s^T Q_s \) is always less than zero. We therefore write (6) as

\[
\|H_{s+1}\|^2 \leq \|H_s\|^2 + \eta^2 \|Q_s\|^2
\]

Or equivalently,

\[
\|H_{s+1}\|^2 - \|H_s\|^2 \leq \eta^2 \|Q_s\|^2
\]

The above equation can also be written as

\[
\|H_{k+1}\|^2 - \|H_0\|^2 \leq \eta^2 \sum \|Q_s\|^2 \text{ for } s = 1, 2 \ldots k
\]

Since we have assumed that \( W_0 \) and \( B_0 \) are both zero vectors it implies that \( \|H_0\| = 0 \) since \( H_0 = [W_0 \ B_0]^T \). Therefore the above equation simplifies as

\[
\|H_{k+1}\|^2 \leq \eta^2 \sum \|Q_s\|^2 \text{ for } s = 0 \ldots k.
\]

Let \( \beta \) be the maximum of all \( \|Q_s\| \)

\[
\|H_{k+1}\|^2 \leq \eta^2 \beta
\]

(8)

From equations (6) and (8), we can deduce that

\[
(k^2 \alpha^2)/\|H_0\|^2 \leq \|H_{k+1}\|^2 \leq \eta^2 \beta
\]

As we can see, the lower bound of \( \|H_{k+1}\|^2 \) is increasing with \( k^2 \) while the upper bound is increasing with \( k \) for every iteration. Since the values \( \beta, \alpha^2 \) and \( \|H_0\|^2 \) are all positive we can state that the value of \( \|Q_s\| \) cannot be greater than some \( k_{max} \) where \( k_{max} \) can be given as

\[
k_{max} = \eta^2 \beta/\|H_0\|^2 / \alpha^2
\]

We thus proved that for a fixed \( \eta \), for all \( k \) and initial weight and bias vectors as zeroes, the ceiling neuron converges after finite number of iterations.
III. MULTI-LAYER FEED FORWARD NETWORKS

Every neuron has what is called an activation function. It is a rule based on which the neuron decides its output for a particular set of inputs. Hence, any neuron is completely described and comprehensively modelled by its activation function. A single neuron has great utility but the true potential of the neuron is realized when coupled with other neurons to form a multi-layered neural network. But for a neuron to be compatible with its role in a multi-layered neural network, its activation function has to be differentiable and, by extension, continuous. This is needed for the back-propagation algorithm to work. But the activation function used thus far i.e. the ceiling function is neither continuous nor differentiable. Hence, we need an alternate activation function which closely mimics the ideal activation function (the ceiling function) without losing continuity or differentiability.

Let \( \text{sigm}(x) \) denote the sigmoid function (bounded between 0 and 1). Inspired by the original sigmoid function, we propose the following activation function:

\[
y = \text{sigm}(W^T x + b_1) \ast (1 + [\text{sigm}(W^T x + b_2) \ast (1 + [\text{sigm}(W^T x + b_3) \ast (1 + \ldots)])])
\]

It can be observed that the proposed activation function approximates the ideal ceiling function extremely well i.e. for all practical intents and purposes.

IV. DYNAMIC ACTIVATION FUNCTION

The ceiling neuron’s Multi-layer activation function is a composite of several sigmoids, the spacing between which is determined by the biases. It must be noted that there is no constraint on the bias values. Keeping this in mind, we arrive to an interesting conclusion. As the bias values are individually varied, the activation function assumes starkly different shapes. This has interesting implications, considering the fact that the complexity of the activation function determines the difficulty in fitting the data and the number of neurons needed to do so.

The biases in a neural network are appropriately adjusted by the back-propagation algorithm to correctly classify the given data. Since each neuron has multiple biases, each controlled by the back-propagation algorithm, this neural network has more degrees of freedom than the conventional one. Hence, the ceiling neuron has the intelligence to choose the most suitable activation function for a given data, so as to optimize the number of neurons required to classify the data. Lower number of neurons means lower training time and less tangled networks.

To prove the versatility of the ceiling neuron, we plot the activation functions for various combinations of bias distances and showcase the activation function’s shapeshifting ability.

A. Sigmoid function: Single bias

Consider the simple case of a single bias. It can be easily observed from the general form of the proposed activation function that, for \( N = 1 \), the activation function imitates the standard sigmoid function.

\[
y = \text{log}(1.4 + e^{(W.X + b - 0.3203)})
\]

By taking a limiting case where the number of biases tends to infinity and each bias is separated by 1 from the previous one,

\[
N \rightarrow \infty; \ b_n = b_{n-1} - 1
\]

The proposed activation function, under the above conditions, becomes

\[
y = \text{sigm}(W^T x - 1) \ast (1 + [\text{sigm}(W^T x - 2) \ast (1 + [\text{sigm}(W^T x - 3) \ast (1 + \ldots)])])
\]

It can be empirically shown that the above equation closely resembles the graphical behavior of the softplus function (albeit slightly modified, as shown below) and approaches the below equation with negligible error.

\[
y = \log(1 + e^{(W.X + b - 0.3203)})
\]
C. Equidistant biases with distance 8

Similar to the previous case, the number of biases tends to infinity and the distance between biases is taken to be 8. The resulting activation function is plotted below.

D. Biases with unequal distances

Considering the case of non-equidistant biases, the following plot is obtained.

V. ASSOCIATIVE MEMORY BASED ON CEILING NEURON

Several researchers have proposed several forms of associative memories like [6],[7]. The most famous of them is a Hopfield Neural network [5]. It is a form of recurrent neural network which is a non-linear dynamical system based on weighted, undirected graph. In a Hopfield Network, each node of the graph represents an artificial neuron which assumes a binary value \{+1 or -1\} and the edge weights correspond to synaptic weights. It acts as an associative memory and has many practical applications.

So, we build a network similar to the Hopfield network based on our proposed ceiling neuron. If the network consists of ‘n’ neurons, then the state space becomes an asymmetric bounded lattice. If the function ‘f’ proposed in our model is changed to a sign function, then the state space becomes a symmetric bounded lattice. The advantage of such a network is that it has huge state space compared to conventional Hopfield network and it leads to increment in storage capacity of the network if properly trained.

A. Notational Information for Associative Memory

Let the number of neurons in the network be “n”. Now W(n x n) denotes the synaptic weight matrix containing elements of the form \( w_{ij} \) (i\(^{\text{th}}\) row and j\(^{\text{th}}\) column) where \( w_{ij} \) is the weight between i\(^{\text{th}}\) neuron and j\(^{\text{th}}\) neuron. Let us suppose that every neuron has “p” biases. Let \( T(px1) \) denote the threshold matrix in which i\(^{\text{th}}\) column denotes the biases of the i\(^{\text{th}}\) neuron in increasing order. \( S_i(px1) \) denotes a column vector corresponding to state of i\(^{\text{th}}\) neuron. It is constructed uniquely from state of i\(^{\text{th}}\) neuron in such a way all elements in it are \{1,-1\} and sum of the elements should be state of ith neuron and number of changes in the column vector if we see from top to bottom should be at most 1.

Ex: If the state is 3 and number of biases is 5 then column vector is [ 1 1 1 -1 -1].

Theorem 1: Let a network be denoted by Weight matrix ‘W’ and threshold matrix ‘T’, then the following hold true:

- If the network is operating in a serial mode and W is a symmetric matrix, then the network always converges to a stable state.
- If the network is operating in parallel mode and W is a symmetric matrix, then network will always converge to stable state or to a cycle of length 2.

B. Proof of Convergence for the Associative Memory

Before proving the convergence, we need to define a suitable energy function based on the network. If we can prove that the energy after every update is non increasing (i.e. change in energy \( \leq 0 \)), then we can say that the energy of the network converges to a stable state.
The Energy Function:

\[ E = -1 \times (\sum_{i,j} S_i(t)^T W_{i,j} S_j(t) - S_i(t)^T T_i - S_j(t)^T T_j) \]

Where \( T_i \) denotes \( i \)th column in the threshold matrix \( T \) and \( W_{i,j} \) denotes a matrix with all equal elements which are \( w_{i,j} \) (weight between \( i \)th and \( j \)th neuron) and \( S_i \) and \( S_j \) are column vectors corresponding to states of \( i \)th and \( j \)th neurons.

Now let us take a serial update case in which state of \( i \)th neuron is updated, then change in energy is given by

\[ E(S_i(t+1)) - E(S_i(t)) = -1 \times \{ (S_i(t+1) - S_i(t))^T \times \left( \sum_j W_{i,j} S_j(t) - T_i \right) + \sum_j (S_j(t)^T W_{j,i} (S_i(t+1) - S_i(t))) - ((S_i(t+1) - S_i(t))^T T_i) \} \]

We know that \( S_i(t+1) - S_i(t) \) and \( \sum_j W_{i,j} S_j(t) - T_i \) have same sign (element wise) because,

\[ S_i(t+1) = \text{sign}(\sum_j W_{i,j} S_j(t) - T_i) \]

Hence the first part in \( E(S_i(t+1)) - E(S_i(t)) \), eq (9) is always negative or zero and if we can show that second part is equal to first part then we proved that \( E(S_i(t+1)) - E(S_i(t)) \) is always negative or zero.

We know that \( S_i(t+1) - S_i(t) \) and \( \sum_j W_{i,j} S_j(t) - T_i \) have same sign (element wise) because \( W_{i,j} \) is a symmetric matrix.

So, \( \sum_j (S_j(t)^T W_{j,i} (S_i(t+1) - S_i(t))) = (\sum_j W_{i,j} S_j(t)) (S_i(t+1) - S_i(t))^T \) because \( W_{i,j} \) is a symmetric matrix.

Hence,

\[ E(S_i(t+1)) - E(S_i(t)) = -2 * (S_i(t+1) - S_i(t))^T \times \left( \sum_j W_{i,j} S_j(t) - T_i \right) \]

and we already showed that

\[ (S_i(t+1) - S_i(t))^T \times \left( \sum_j W_{i,j} S_j(t) - T_i \right) \]

is positive.

Hence, change in energy is always negative or zero. Since the energy is a lower bounded function and since it always decreases or remains constant, we can say that the energy of the system converges.

Convergence of network: If the energy of the network is converged, there are two possibilities

- \( S_i(t+1) - S_i(t) = 0 \) and change is energy is zero
- All the elements of \( S_i(t+1) - S_i(t) \) might not be zero but the corresponding elements in \( \left( \sum_j W_{i,j} S_j(t) - T_i \right) \) might be zero hence making change in energy zero. This kind of change is only possible when one or many elements of \( S_i \) changes from -1 to +1.

Hence, once the energy network is converged then it is clear from the above facts that network attains a stable state after finite intervals.

We can prove parallel update case by converting it into serial update as shown in [2]. Hence our above stated theorem is proved.

Finally, we proved that we can construct an associative memory based on our proposed neuron model because of its convergent dynamics. The advantage of our model is that here state is very large compared to the conventional model which leads to more storage capacity. Efficient learning mechanisms will be studied later.

VI. Numerical Experiments

The proposed ceiling neuron based Neural Network with a single hidden layer having 300 hidden neurons has been trained and tested on MNIST database. The output layer has 10 sigmoidal neurons. In this experimentation, we have used stochastic gradient descent (SGD) with a learning rate of 0.1. This neural network has correctly classified 97.89% of MNIST test set after 90 epochs. This is a substantial increase from the similar experimentation in [3] which showed that a single hidden layer sigmoidal neural network with 300 hidden units produced an accuracy of 95.3%.

VII. Conclusion

In this research paper, a novel neural network named “Ceiling Neuron” is proposed and learning algorithm for the same is given. It is shown that a single ceiling neuron using the proposed learning algorithm divides the region bounded by a hyper plane into \( n+1 \) sections. An activation function which mimics the ceiling nature is proposed to be operated in Multi-Layered networks. The dynamic nature of such an activation function is discussed. An associative memory based on the ceiling neuron is proposed and its dynamics are studied. A single hidden layered neural network with activation function as proposed in section-IV is tested on MNIST data and results are presented.
REFERENCES


