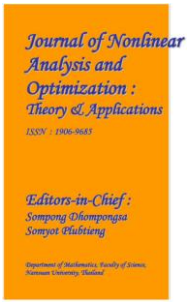
Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 14, Issue. 2, No. 4: 2023
ISSN : **1906-9685**



LEARNING RULE FOR MULTIPLE LAYERED NEURAL NETWORKS WITH NUMERICAL EXAMPLE

Sumathi C B PG and Research Department of Mathematics, Marudhar Kesari Jain College for Women, Tamil Nadu, India
Jothilakshmi R PG and Research Department of Mathematics, Mazharul Uloom College, Tamil Nadu, India
Jayagopi G Department of CSE, Mother Theresa Institute of Engineering and Technology, Andhra Pradesh, India
E – Mail: c.bsumathi@yahoo.in , jothilakshmiphd@gmail.com , jayagopi0075@gmail.com

ABSTRACT
This paper demonstrates the learning rule for layered neural network through supervised data which provides the feature kernel using weighted impulse sum. Also we see how the input data is processed as it passes the layered neural network. And we obtain an important type of convolution neural network for the feature extractor in which the weights are determined through the training process via delta rule. We also encounter the systematic method with numerical example to show that the process of calculation of multiple layer neural networks is finally equivalent to single layer neural network.

Key Words: Neural network, Feature kernel, Learning Rule, Supervised data, Delta rule.

II INTRODUCTION
The neural network started as a single layer neural network and evolved into shallow neural network, followed by the deep neural network. In the layered neural network, the signal enters the input layer, passes through the hidden layer, and leaves through the output layer. During this process, the signal advances layer by layer. In other words, the nodes on one layer receive the signal simultaneously and send the processed signal to the next layer at the same time. The multiple layer neural networks are evolved to overcome the limitations of the single layer neural network. And the calculation process of hidden layer leads to single layer neural network.

2.l Architecture of Neural network
The neural network is the connection of nodes through weights. The information in neural network is stored in the form of weights and bias.

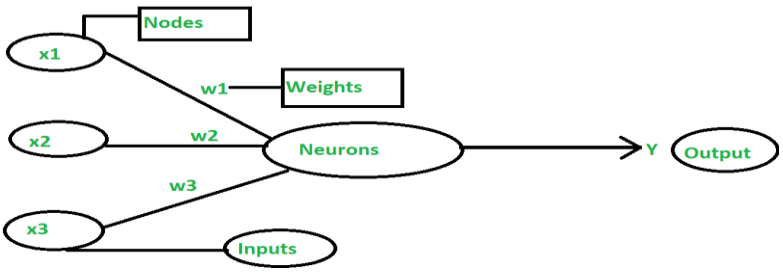


Fig 1: A node that has three inputs

The input signal is multiplied by the weight before it reaches the node. Once the weighted signals are collected at the node, the values are summed. The weight sum of the above figure is

$$v = (w_1 * x_1) + (w_2 * x_2) + (w_3 * x_3) + b \qquad (1)$$

This equation shows that the signal with greater weight has greater effect. The matrix representation of weighted sum equation (1) is as follows

$$v = wx + b \quad (2)$$

Where $w = (w_1, w_2, w_3)$ and $x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$.

As the node enters the weighted sum becomes the activation function which produces the output. The activation function determines the behavior of the nodes. Thus the output form of activation function to the weighted sum is

$$y = \varphi(v) = \varphi(wx + b) \quad (3)$$

2.2 Delta Rule

The Delta rule is a type of numerical method called gradient descent. The gradient descent starts from the initial value and proceed to the solution. The delta rule is the learning rule for single layer neural network.

Delta rule adjust the weight by the following procedure, If an input node contributes to the error of the output node, the weight between the two nodes is adjusted in proportion to the input value x_j and the output error e_i .

Initialize the weight at adequate value. Take the input from the training data and enters into the neural network. Calculates the error of outputy_i, from the correct output d_i , to the input.

$$e_i = d_i - y_i \quad (4)$$

Calculate the weight according to the delta rule

$$\Delta w_{ij} = \alpha e_i x_j \quad (5)$$

Adjust the weight as follows

$$w_{ij} \leftarrow w_{ij} + \alpha e_i x_j \quad (6)$$

Where

x_j – The output from the input node j

e_i- The error of the output node i

w_{ij}- The weight between the output node i and input node j

α- learning rate ($0 < \alpha \leq 1$)

Perform the steps for all the training data until the error reaches an acceptable level. The learning rate α determines how much the weight is to be changed per time. If this value is too high, the output wanders around the solution and fails to converge. If it is too slow the calculation reaches the solution too slowly. For an arbitrary activation function the delta rule is expressed as

$$w_{ij} \leftarrow w_{ij} + \alpha \delta_i x_j \quad (7)$$

And $$\delta_i = \varphi'(v_i)\, e_i \quad (8)$$

We also derive the delta rule with sigmoid function, which is used as activation function. The Sigmoid function is defined as

$$\varphi(x) = \frac{1}{1+e^{-x}} \quad (9)$$

We get on differentiating

$$\varphi'(x) = \varphi(x)(1 - \varphi(x)) \quad (10)$$

Substituting in (8), we arrive

$$\delta_i = \varphi(v_i)(1 - \varphi(v_i))e_i \quad (11)$$

Again from (7), we have

$$w_{ij} \leftarrow w_{ij} + \alpha\varphi(v_i)(1 - \varphi(v_i))e_i x_j \quad (12)$$

Thus the weight is determined in proportion to the output node error e_i and the input node value x_j

III NUMERICAL EXAMPLE

Consider the neural network with single hidden layer

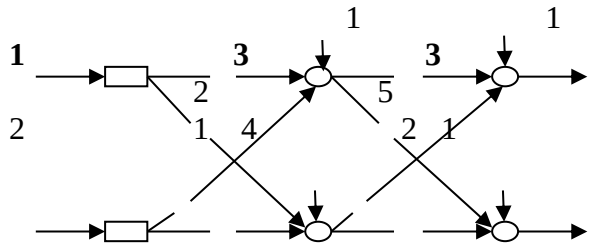


Fig 2 : A Neural network with a single hidden layer

The first node of the hidden layer calculates the output as:

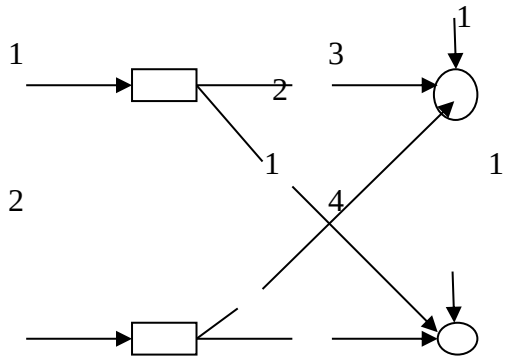


Fig 3 : Output from the hidden layer

Weighted Sum: $\nu = (3 \text{ x } 1) + (1 \text{ x} 2) + 1 = 6$

Output: $y = \varphi(v) = \nu = 6$

In a similar manner, the second node of the hidden layer calculates the output as:

Weighted Sum: $\nu = (2\text{x } 1) + (4\text{x}2) + 1 = 11$

Output : $y = \varphi(v) = \nu = 11$

The weighted sum calculation can be combined in a matrix equation as follows

$$\nu = \begin{bmatrix} 3 \text{ x}1 + 1\text{x } 2 + 1 \\ 2 \text{ x } 1 + 4 \text{ x}2 + 1 \end{bmatrix} = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 11 \end{bmatrix}$$

The weight of the first node of the hidden layer lay in the row, and the weight of the second node are in the second row. This result can be generalized as stated in equation 2

Now let us determine the output layer

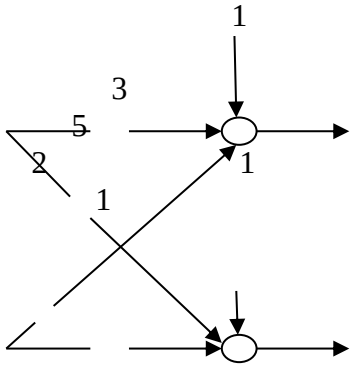


Fig 4 : Output from the output layer

Let us use equation 2 to calculate the output

Weighted Sum $\nu = \begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 11 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 41 \\ 42 \end{bmatrix}$

Output $y = \varphi(v) = \begin{bmatrix} 41 \\ 42 \end{bmatrix}$

Also Substituting the equation of weighted sum of the hidden layer into the equation of weighted sum of the output layer yields the following

$$\nu = \begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 11 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix} \left(\begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right) + \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix} \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 13 & 11 \\ 17 & 9 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 6 \\ 7 \end{bmatrix}$$

This matrix equation shows that this example neural network is equivalent to a single layer neural network as shown in the following figure

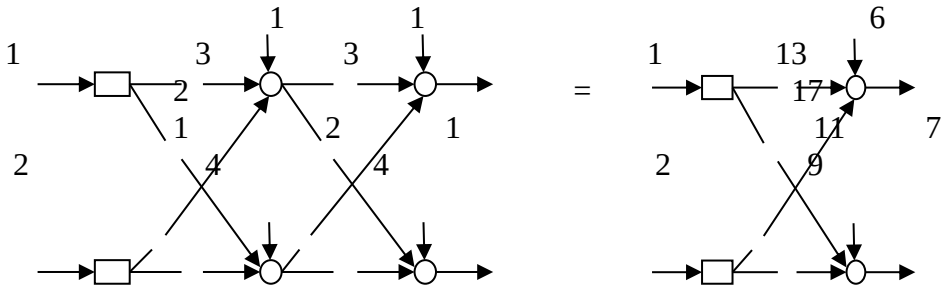


Fig 5 : Neural network is equivalent to single layer neural network

IV CONCLUSION

The delta rule is an iterative method that gradually reaches the solution and shows that the network should be trained repeatedly with the training data until we reach the solution. Also we clearly demonstrated the learning rule for layered neural network through supervised data using weighted impulse sum on repeated process equivalent to the single layer neural network.

References:

[1] Abbas YA, Najmaddin AS and Gulnar WS 2009 A Modified Conjugate Gradient Formula for Back Propagation Neural Network Algorithm Journal of Computer Sciences **5** 849-856

[2] Breiman, L. Bagging predictors. Machine learning. 1996, 24.2: 123-140.

[3] Chapelle, O., and Zien, A. Semi-supervised classification by low density separation. AISTATS, 2005, (pp. 5764).

[4] Erhan, D., Bengio, Y., Courville, A., Manzagol, P. A., Vincent, P., and Bengio, S. Why does unsupervised pre-training help deep learning?. The Journal of Machine Learning Research, 2010, 11: 625-660

.[5] Moller MF 1990 A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning Neural Networks **6** 525-533

[6] Phil Kim. MATLAB Deep Learning", "Chapter 6 Convolution Neural Network", Springer Science LLC, 2017

[7] Ranzato, M., and S zummer, M. Semi-supervised learning of compact document with deep networks. In Proceedings of the 25th international conference on Machine learning. ACM, 2008. p. 792- 799.

[8] Weston, J., Ratle, F., and Collobert, R. Deep learning via semi-supervised embedding.In Proceedings of the 25th international conference on Machine learning. ACM, 2008. p. 1168- 1175.

[9] XiangwenWang, Xianghong, Neural Network, Supervised learning in spiking neural networks: A review of algorithms and evaluations, Volume 125, May 2020, Pages 258-280