Approximating Number of Hidden layer neurons in Multiple Hidden Layer BPNN Architecture

Saurabh Karsoliya
Student, Department of Computer Science & Engineering,
Maulana Azad National Institute of Technology, Bhopal, India.

***Abstract*—Hidden layers plays a vital role in the performance of Back Propagation Neural Network especially in the case where problems related to arbitrary decision boundary to arbitrary accuracy with rational activation functions are encountered. Also, multiple hidden layer can approximate any smooth mapping to any accuracy. The process of deciding the number of hidden layers and number of neurons in each hidden layer is still confusing. In this paper, an survey is made in order to resolved the problem of number of neurons in each hidden layer and the number of hidden layers required.**

***Index Terms*— Neural Network, BPNN, Hidden Layer, Neurons,**

I. Introduction

Neural Network is a vast domain of technology where one can implement “human brain decision making power” into computer programs on the basis of error and approximation. Also, lot of research and development has been made in the field of Artificial Intelligence with the help of Neural Network, Fuzzy Logic and Genetic Algorithm. Usually Neural Network comprises of three types of layer viz. Input Layer, Hidden Layer and Output layer. Initially in the development phase hidden layer was not used in the problems having linearly separable domain. But when any function that contains a continuous mapping from one finite space to another, one has to make use of single hidden layer. The complexity increase when problems related to arbitrary decision boundary to arbitrary accuracy with rational activation functions are evaluated by the neural network in which two hidden layers are used. So deciding the number of hidden layers and number of neurons in each hidden layer is a challenging issue while considering any complex problem. A Neural Network contain following layers:

A. Input Layer:

The Input layer is a layer which communicates with the external environment that presents a pattern to the neural network. Once a pattern is presented to the input layer, the output layer will produce another pattern. In essence, this is all the neural network does. The input layer should represent the condition for which we are training the neural network. Every input neuron should represent some independent variable that has an influence over the output of the neural network [4].

B. Output Layer:

The output layer of the neural network is what actually presents a pattern to the external environment. The pattern presented by the output layer can be directly traced back to the input layer. The number of output neurons should be directly related to the type of work that the neural network is to perform. To determine the number of neurons to use in output layer, first consider the intended use of the neural network. If the neural network is to be used to classify items into groups, then it is often preferable to have one output neuron for each group that input items are to be assigned into. If the neural network is to perform noise reduction on a signal, then it is likely that the number of input neurons will match the number of output neurons. In this sort of neural network, the patterns to leave the neural network in the same format as they entered [4].

C. Hidden Layer:

The hidden layer is the collection of neurons which has activation function applied on it as well as provide an intermediate layer between the input layer and the output layer. Many researches has been made in evaluating the number of neurons in the hidden layer but still none was accurate.

Another question arises that how many hidden layers has to be used when dealing with complex problem. If the data is linearly separable than there is no need to use hidden as the activation function can be implemented to input layer which can solve the problem. But in case of problems which deals with arbitrary decision boundary to arbitrary accuracy with rational activation functions then one has to use two or three hidden layer. Also, the number of neurons that should be kept in each hidden layer need to be calculated. If the number of neurons are less as compared to the complexity of the problem data then “Underfitting” may occur. Underfitting occurs when there are too few neurons in the hidden layers to adequately detect the signals in a complicated data set. If unnecessary more neurons are present in the network then “Overfitting” may occur. Several methods are used till now which do not provide the exact formula for calculating the number of hidden layer as well as number of neurons in each hidden layer. In the next section, different techniques/assumptions are elaborated in the literature survey for calculating the number of hidden layer and the number of neurons in each hidden layer.

II. Previous Work

Many researcher put their best effort in analyzing the solution to the problem that how many neurons are kept in hidden layer in order to get the best result, but unfortunately no body succeed in finding the optimal formula for calculating the number of neurons that should be kept in the hidden layer so that the neural network training time can be reduced and also accuracy in determining target output can be increased. Basically when dealing with the number of neurons in the input layer, one has to analyze about the data which is trained. For example, while dealing with handwritten numeral recognition using neural network for pin code recognition [5], the box size in which the pin number is written is taken under consideration for determining the number of neurons in the input layer. If the box size is 10x15 then 150 neurons must be taken as a input layer, so that every pixel contribute its value to the input layer individually. When the output layer is considered, then it depends on the chosen model configuration. For example, in this paper we are concentrating on the recognition of pin number in any scripting language, so 16 neurons are taken in output layer so that it can be mapped to Unicode which has 16 bit for identification. If the neural network is a regressor, then the output layer has a single node.

Now the number of neurons in the intermediate layer is taken under consideration which is the soul purpose of this paper. Before that one fact should be kept in mind that if the data is linearly separable then there is not at all any use of hidden layer. Linear Activation function can be directly implemented on the input and output layer. One hidden layer will be used when any function that contains a continuous mapping from one finite space to another. Two hidden layer can represent an arbitrary decision boundary to arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy [14].

When Multilayer Back Propagation Neural Network is under consideration then the number of hidden layers and approximation of neurons in each hidden layer need to be calculated. Since every neural network architecture is based on the training data samples (i.e. application specific), so in this paper, training data is taken as handwritten pin code samples of different scripting language and two hidden layers are used in order get more accuracy in recognition. Unnecessary increasing hidden layer may causes increase in the complexity of network. This is because in the weight balancing stage, weights between the first hidden layer and second hidden layer are also considered for weight updating which depends upon the error gap between the actual and target output.

III. Determining Hidden Layers and Hidden Nodes

Firstly, number of hidden nodes approximation for the whole neural network is important and afterwards the decision of selecting the number of layers as well as proportion of neurons between the first and second hidden layer is required. Usually some rule-of-thumb methods are used for determining the number of neurons in the hidden nodes.

- The number of hidden layer neurons are 2/3 (or 70% to 90%) of the size of the input layer. If this is insufficient then number of output layer neurons can be added later on.[1]
- The number of hidden layer neurons should be less than twice of the number of neurons in input layer. [2]
- The size of the hidden layer neurons is between the input layer size and the output layer size.[6]

But the above three myths are not considered to be true always because not only the input layer and the output layer decides the size of the hidden layer neurons but also the complexity of the activation function applied on the neurons, the neural network architecture, the training algorithm and most important the training samples database on which the neural network is designed to execute.

Multiple hidden layers are used in the applications where accuracy is the criteria and no limit for the training time is mentioned. Even the drawback of using multiple hidden layers in the neural network is that they are more prone to fall in bad local minima [9].
Also, experimental results in this paper shows that the number of neurons in first and second hidden layer should be kept nearly equal so that the network can be trained easily. Usually samples similar to data which contain discontinuities, such as a saw tooth waveform requires multiple hidden layers.

An “structured trial and error” method is used by maximum developer for creating a neural network’s layer approximation.
As usual, input layer neurons are estimated depending upon the data samples which is to be trained. For example, if 15x10 pixel block of data is taken as sample then 150 neurons are taken as input layer. Similarly output layer neurons can be calculated depending upon the number of types of output. In this approach, 16 neurons are taken in output layer because Unicode scheme is used. For hidden layer, initially few random numbers are neurons are taken and samples are allowed to train on it. If the network fails to converge after a reasonable period, restart training up to five times, so that it is assured that it has not fallen into a local minima. If the network still fails to converge then add few more neurons in the layer and allow it to train. Still no better results found then it is necessary to add second hidden layer. Note that first and second hidden layer must contain equal number of neurons in order to train the network easily. In order to get best approximation of the hidden layer neurons, equal amount of number of neurons in both hidden layers can be reduced and again training is done so that one can check whether the network converges to the same solution even after reducing the number of hidden layer neurons.

Yinyin Liu, Janusz A. Starzyk, Zhen Zhu [9] in their approach to optimize the number of neurons in the hidden layer using benchmark datasets and estimation of the signal-to-noise-ratio figure. The method utilizes a quantitative criterion based on the SNRF to detect overfitting automatically using the training error only, and it does not require a separate validation or testing set. This method reduces the overfitting problem in the network.

Rivals I. & Personnaz L.[7] used techniques based on least squares estimation and statistical tests for estimating the number of neurons in the hidden layer. Two approaches were used phase wise , first the bottom-up phase in which initially the number of neurons are increased up to an extent till the network becomes ill-conditioned. Afterwards in the next phase i.e. the top-down approach the neural network has to go through statistical Fisher tests. The second phase usually reduces the complexity of the neural network.

F. Fnaiech in [8] make an attempt to prune the hidden nodes of the feed forward architecture by initially creating a non linear activation function of hidden nodes as Taylor's expansion and then NARX (nonlinear auto regressive with exogenous input) model is used. Afterwards nonlinear order selection algorithm proposed by Kortmann-Unbehauen most relevant signal of the NARX are selected and finally BPNN algorithm is used in order to prune the hidden nodes.

Stuti Asthana and Rakesh K Bhujade in [5] made an approach to use two hidden layer for recognizing multiscript number recognition using artificial neural network on postcard, keeping accuracy as a chief criteria. More than 95% accuracy was obtained due to the use of two hidden layer. Moreover equal number of neurons are used in both layers so as to get best accuracy. This work has been tested on five different popular Indian scripts namely Hindi, Urdu, Tamil ,English and Telugu and satisfactory results were obtained under ideal condition.

The decision method for selecting number of neurons to getting optimal solution using two phase method is describe in [3] by Kazuhiro Shin-ike . In two phase method the termination condition is same as the trial and error method but in new approach dataset is divided into four groups in which two groups of data are used in first phase to train the network and one group of remaining data set is used in second phase to train the network and last group of data set is used predict the output values of the trained network and this experiment is repeated for different number of neurons to get the minimum number of error terms for selecting the number of neurons in the hidden layer. Using this approach correctness rate is achieve 87.5 percent.

IV. WHY NOT FOUR HIDDEN LAYERS.

As previously discussed in this paper that one or two hidden layer are sufficient to solve any non linear complex problem. Also, if accuracy is the major and most needed criteria for designing the network then one can adopt the solution of third hidden layer, but this will increase overall complexity of the neural network and the total training time will be increased. There is no need to use four hidden layer in neural network architecture. As mentioned in [1] that the number of hidden layer neurons are 2/3 (or 70% to 90%) of the size of the input layer and in [2] that the number of hidden layer neurons should be less than twice of the number of neurons in input layer, then

2/3 neurons of input layer = first hidden layer
2/3 neurons of input layer = second hidden layer
2/3 neurons of input layer = third hidden layer
Total neurons in hidden layers=3x2/3= 2 (Twice of Input)

So, if rule of thumb of [1] and [2] are considered then this is proved theoretically that use of fourth hidden layer may led to problem in neural network training. Many researchers suggest that least hidden layer must be used in order to save the training time and complexity.

V. PROPOSED BPNN ALGORITHM FOR THREE HIDDEN LAYER

The MLP learning algorithm for three hidden layer is outlined below.

1. Initialize the network, with all weights set to random numbers between –1 and +1.
2. Present the first training pattern, and obtain the output.
3. Compare the network output with the target output.
4. Propagate the error backwards.

(a) Correct the output layer of weights using the following formula.

$$\mathbf{W_{h3o} = W_{h3o} + (\eta\delta_o O_{h3})}$$

where $\mathbf{W_{h3o}}$ is the weight connecting hidden unit $\mathbf{h_3}$ with output unit **O**, $\boldsymbol{\eta}$ is the learning rate, $\mathbf{O_{h3}}$ is the output at hidden unit $\mathbf{h_3}$. $\boldsymbol{\delta_o}$ is error given by the following.

$$\boldsymbol{\delta_o = O_o(1- O_o)(T_o - O_o)}$$

(b) Correct the input weights using the following formula.

$$\mathbf{W_{h2h3} = W_{h2h3} + (\eta\, \delta_{h3} O_{h2})}$$

where $\mathbf{W_{h2h3}}$ is the weight connecting second and third hidden layers with Oh2 is the input at node of the third hidden layer, η is the learning rate. $\boldsymbol{\delta_{h3}}$ is calculated as follows.

$$\boldsymbol{\delta_{h3} = O_{h3}(1- O_{h3})\Box(\delta_o W_{h3o})}$$

(c) Correct the input weights using the following formula.

$$\mathbf{W_{h1h2} = W_{h1h2} + (\eta\, \delta_{h2} O_{h1})}$$

where $\mathbf{W_{h1h2}}$ is the weight connecting first two hidden layers with , Oh1 is the input at node of the second hidden layer, η is the learning rate. $\boldsymbol{\delta_{h2}}$ is calculated as follows.

$$\boldsymbol{\delta_{h2} = O_{h2}(1- O_{h2})\Box(\delta_o W_{h2o})}$$

(d) Correct the input weights using the following formula.

$$\mathbf{W_{ih1} = W_{ih1} + (\eta\, \delta_{h1} O_i)}$$

where $\mathbf{W_{ih1}}$ is the weight connecting node i of the input layer with node of the first hidden layer, $\mathbf{O_i}$ is the input at node of the hidden layer, $\boldsymbol{\eta}$ is the learning rate. $\boldsymbol{\delta h_1}$ is calculated as follows.

$$\boldsymbol{\delta_{h1} = O_{h1}(1- O_{h1})\Box(\delta_o W_{h1h2})}$$

5. Calculate the error, by taking the average difference between the target and the output vector.
6. Repeat from 2 for each pattern in the training set to

complete one epoch.

7. Repeat from step 2 for a specified number of epochs, or until the error ceases to change.

VI. CONCLUSION

Practically, it is very difficult to determine a good network topology just from the number of inputs and outputs. It depends critically on the number of training examples and the complexity of the classification trying to learn. There are problems with one input and one output that require millions of hidden units, and problems with a million inputs and a million outputs that require only one hidden unit, or none at all. However, this is true fact that by taking suitable number of hidden layers and the number of neurons in each hidden layer, better results in less training time can be obtained. Many researchers develop approach to estimate the number of neurons and hidden layers requirement for a neural network but the approximation also get dependable on the type of the database samples for which the network is designed.

By increasing the number of hidden layers up to three layer, accuracy can be achieved up to great extent but complexity of the neural network and training time is increased many folds. If accuracy is the main criteria for designing a neural network then hidden layers can be increased. Also, unnecessary increment in the neurons or layer will led to overfitting problem. So it is quiet essential that before designing the neural network, training database samples must be analyzed so that approximation of number of neurons and hidden layers can be guessed properly.

REFERENCES

[1] Boger, Z., and Guterman, H., 1997, "Knowledge extraction from artificial neural network models," *IEEE Systems, Man, and Cybernetics Conference*, Orlando, FL, USA

[2] Berry, M.J.A., and Linoff, G. 1997, *Data Mining Techniques*, NY: John Wiley & Sons

[3] Kazuhiro Shin-ike, “A Two Phase Method for Determining the Number of Neurons in the Hidden Layer of a 3-Layer Neural Network”, SICE Annual Conference 2010, August 18-21, 20 1 0, The Grand Hotel, Taipei, Taiwan.

[4] http://www.heatonresearch.com/node/707

[5] Stuti Asthana and Rakesh K Bhujade, “Handwritten Multiscript Pin Code Recognition System having Multiple hidden layers using Back Propagation Neural Network”, International Journal of Electronics Communication and Computer Engineering, ISSN(Online): 2249 - 071X, Volume 2, Issue 1, Sep 2011

[6] Blum, A., 1992, Neural Networks in C++, NY: Wiley.

[7] Rivals I. & Personnaz L. “A statistical procedure for determining the optimal number of hidden neurons of a neuralmodel”, Second International Symposium on Neural Computation (NC’2000), Berlin, May 23-26 2000.

[8] Fnaiech, F., Fnaiech, N., Najim, M., 2001 “A new feedforward neural network hidden layer neuron pruning algorithm ”IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01).

[9] Yinyin Liu, Janusz A. Starzyk, Zhen Zhu, “Optimizing Number Of Hidden Neurons in Neural Networks”, http://www.ohio.edu/people/starzykj/network/Research/Papers/ Recent%20conferences../Hidden%20Neurons%20AIA2007_549 -204.pdf

[10] Jacques de Villiers and Etienne Barnard, 1992, “Backpropagation Neural Nets with One and Two Hidden Layers”, IEEE Transactions on Neural Networks, Vol. 4, No. 1.