Back Propagation Neural Network

Dr. K.Vijayarekha
Associate Dean
School of Electrical and Electronics Engineering
SASTRA University, Thanjavur-613 401
# Table of Contents

1. Back Propagation Neural Network ....................................................... 3
  
  1.1. Architecture .................................................................................. 3
  1.2. Algorithm ..................................................................................... 3
  1.3. Activation Function ...................................................................... 4
  1.4. Training Algorithm ...................................................................... 4
1. Back Propagation Neural Network

Backpropagation is a training method used for a multi layer neural network. It is also called the generalized delta rule. It is a gradient descent method which minimizes the total squared error of the output computed by the net.

Any neural network is expected to respond correctly to the input patterns that are used for training which is termed as memorization and it should respond reasonably to input that is similar to but not the same as the samples used for training which is called generalization.

The training of a neural network by back propagation takes place in three stages

1. Feedforward of the input pattern
2. Calculation and Back propagation of the associated error
3. Adjustments of the weights

After the neural network is trained, the neural network has to compute the feedforward phase only. Even if the training is slow, the trained net can produce its output immediately.

1.1. Architecture

A multi layer neural network with one layer of hidden unitss is shown in the figure.

The output units and the hidden units can have biases. These bias terms are like weights on connections from units whose output is always 1. During feedforward the signals flow in the forward direction i.e. from input unit to hidden unit and finally to the output unit. During back propagation phase of learning, the signals flow in the reverse direction.

1.2. Algorithm

The training involves three stages

1. Feedforward of the input training pattern
2. Back propagation of the associated error
3. Adjustments of the weights.

During feedforward, each input unit (Xi) receives an input signal and sends this signal to each of the hidden units Z1, Z2, …Zn. Each hidden unit computes its activation and sends its signal to each output unit. Each output unit computes its activation to compute the output or the response of the neural net for the given input pattern.
During training, each output unit compares its computed activation $y_k$, with its target value $t_k$ to determine the associated error for the particular pattern. Based on this error the factor $\partial_k$ for all $m$ values are computed. This computed $\partial_k$ is used to propagate the error at the output unit $Y_k$ back to all units in the hidden layer. At a later stage it is also used for updation of weights between the output and the hidden layer. In the same way $\partial_j$ for all $p$ values are computed for each hidden unit $Z_j$. The values of $\partial_j$ are not sent back to the input units but are used to update the weights between the hidden layer and the input layer.

Once all the $\partial$ facts are known, the weights for all layers are changed simultaneously. The adjustment to all weights $w_{jk}$ is based on the factor $\partial_k$ and the activation $z_j$ of the hidden unit $Z_j$. The change in weight to the connection between the input layer and the hidden layer is based on $\partial_j$ and the activation $x_i$ of the input unit.

### 1.3. Activation Function

An activation function for a back propagation net should have important characteristics. It should be continuous, differentiable and monotonically non-decreasing. For computational efficiency, it is better if the derivative is easy to calculate. For the commonly used activation function, the derivative can be expressed in terms of the value of the function itself. The function is expected to saturate asymptotically.

The commonly used activation function is the binary sigmoidal function.

### 1.4. Training Algorithm

The activation function used for a back propagation neural network can be either a bipolar sigmoid or a binary sigmoid. The form of data plays an important role in choosing the type of the activation function. Because of the relationship between the value of the function and its derivative, additional evaluations of exponential functions are not required to be computed.

Algorithm

Step 0: Initialize weights

Step 1: While stopping condition is false, do steps 2 to 9

- **Step 2:** For each training pair, do steps 3 - 8

  **Feed forward**

  - **Step 3:** Input unit receives input signal and propagates it to all units in the hidden layer
  - **Step 4:** Each hidden unit sums its weighted input signals
  - **Step 5:** Each output unit sums its weighted input signals and applied its activation function to compute its output signal.
**Backpropagation**

Step 6: Each output unit receives a target pattern corresponding to the input training pattern, computes its error information term

\[ \delta_k = (t_k - y_k) f'(y_{ink}) \]

Calculates its bias correction term

\[ \Delta W_{ok} = \alpha \delta_k \]

And sends \( \delta_k \) to units in the layer below

Step 7: Each hidden unit sums its delta inputs

\[ \delta_{in_j} = \sum_{k=1}^{m} \delta_k w_{jk} \]

Multiplies by the derivative of its activation function to calculate its error information term

\[ \delta_j = \delta_{in_j} f'(z_{in_j}) \]

Calculates its weight correction term

\[ \Delta v_{ij} = \alpha \delta_j x_i \]

And calculates its bias correction term

\[ \Delta v_{oj} = \alpha \delta_j \]

*Update weights and biases*

Step 8: Each output unit updates its bias and weights

\[ W_{jk}(\text{new}) = W_{jk}(\text{old}) + \Delta w_{jk} \]

Each hidden unit updates its bias and weights

\[ V_{ij}(\text{new}) = V_{ij}(\text{old}) + \Delta v_{ij} \]

Step 9: Test stopping condition