Module 1

Learning Factors

The backpropagation algorithm was originally introduced in the 1970s, but its importance wasn't fully appreciated until a famous paper in 1986 by David Rumelhart, Geoffrey Hinton, and Ronald Williams. The paper describes several neural networks where backpropagation works far faster than earlier approaches to learning, making it possible to use neural nets to solve problems which had previously been insoluble. Today, the backpropagation algorithm is the workhorse of learning in neural networks.
Although Backpropagation is the widely used and most successful algorithm for the training of a neural network of all time, there are several factors which affect the Error-Backpropagation training algorithms. These factors are as follows.

1. Initial Weights

Weight initialization of the neural network to be trained contribute to the final solution. Initially before training the network weights of the network assigned to small random uniform values. If all weights start out with equal weight values then there is a major chance of bad solution if required final weights are unequal. Similarly, if initial random uniform weights are not uniform (between 0 to 1) then there are chances of stuck in global minima and also the step towards global minima are very small as our learning rate is small. Learning of network is very slow if initial weights fall far from global minima of the error plot as an error of network is a function of weights of the network. To get faster learning of network initial weights of neural network should fall nearer to global minima of the error plot. Many empirical studies of the algorithm point out that continuing training beyond a certain low error plateau results in the undesirable drift of weights and this causes error to increase and mapping function implemented by network goes on decreasing. To address this issue training should be restarted with newly initialized random uniform weights.

2. Cumulative weight adjustment vs Incremental Updating

The Error backpropagation learning technique is based on a single pattern error detection in which it requires small adjustment of weights which follow each training pattern in a training data. This technique of adjusting of weights at every step when pattern applied to a network is called as Incremental Updating. The Error backpropagation or Gradient Descent technique also implements the gradient descent minimization of the overall error function computed over a complete cycle of patterns, provided that learning constant sufficiently small. This scheme is known as cumulative weight adjustment and error for this technique is calculated using the following expression.

$$E = \frac{1}{2} * \sum_{p=1}^{P} \sum_{k=1}^{K} (d_{pk} - O_{pk})^2$$

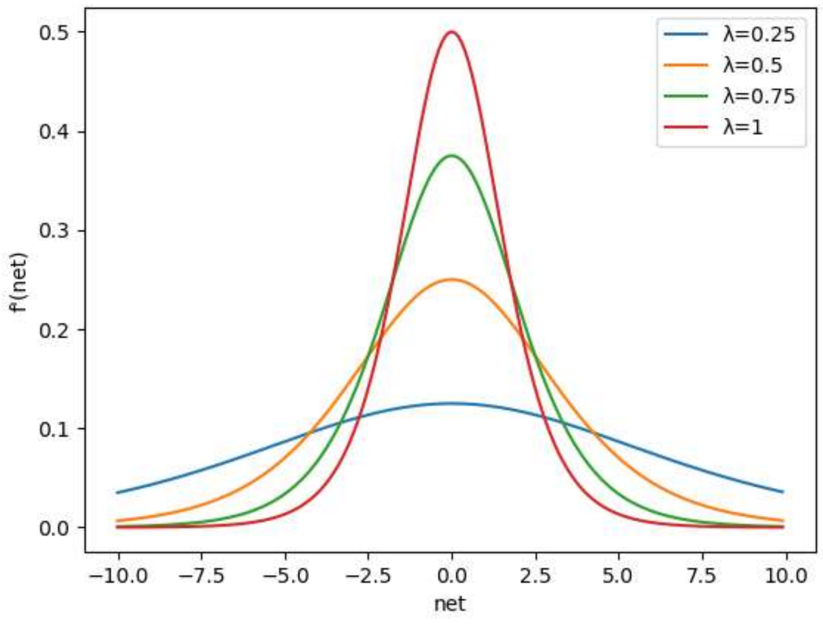
Although both these techniques can bring satisfactory solutions, attention should be paid to the fact that training works best under random conditions. For incremental updating, patterns should be chosen randomly of different classes from training set so that the network should not overfit for same class patterns.

3. The steepness of the activation function λ

Gradient descent learning algorithm uses the continuous type of activation function, the most common used activation function is the sigmoid function (unipolar and bipolar). This sigmoid function is characterised by a factor called steepness factor λ. the derivative of activation function serves as multiple factors in building error signal term of a term of the neuron. Both, choice and shape of activation function affect the speed of network learning. The derivative of activation function(unipolar sigmoid) is given by,

$f'(net) = 2\times\lambda\times exp(-net)/[1+exp(-net)]^2$

The following figure shows the slope function of the activation function and illustrates how the stiffness λ affect the learning of the network.



derivative of activation function for different λ values for the fixed learning constant all adjustments of weights are in proportion to the steepness coefficient λ. When we use the large value of λthen we get a similar result when we use large learning constant η.

4. Learning Constant η.

The effectiveness and convergence of Error backpropagation are based on the value of learning constant η. The amount by which weights of network updated is directly proportional to the learning factor η and hence it plays the important role in error signal term of a neuron.

$$\Delta W = -\eta * C * f'(net)$$

where C is error signal term of neuron and $f'(net)$ is the derivative of activation function.

When we use a larger value of η, our network takes wider steps to reach global minima of error plot. Due to a larger value of η, there is a chance of missing global minima if error plot yields shorter global minima. Similarly if we use a smaller value of η, our network takes shorter steps to reach global minima of error plot but in this case, there is a chance of stuck in local minima of error plot. To overcome these situation weights of a network are newly initialized with small random values and retrain the network.

5. Momentum method

Momentum method deals with the convergence of gradient descent learning algorithm. Its purpose is to accelerate the convergence of learning. This method supplements the current

Module 1

weight adjustments with the fraction of most recent weight adjustment. This is usually done with the following formulae,

$$\Delta W(t) = -\eta \times \nabla E(t) + \alpha * \Delta W(t-1)$$

where t and t-1 represent a current and previous step and **α** is user selective momentum constant (should be positive). The second term on the right side of the equation represents the momentum term. For total N steps using the momentum method, the weight change can be expressed as

$$\Delta W(t) = -\eta \times \sum \alpha \hat{} n * \nabla E(t-1)$$

Typically **α** is chosen between 0.1 to 0.8. By adding the momentum term the weight adjustment is enhanced by the fraction of adjustment of weights at the previous step. This approach leads to the global minima of the error curve with a much greater step