**Report about Momentum Neural Network**

**Neural network momentum:**

Neural network momentum is an ordinary technique that increase the performance and improves the speed of training and accuracy. The process of training a neural network is finding values for the error at each node, biases, and weights by using some input values, the calculated outputs values are near to match the target or the correct values.

**Advantage of neural network momentum:**

1. It increase the speed of learning and it gives a huge boost.
2. It improve the accuracy.

**Disadvantage of neural network momentum:**

1. Choosing inappropriate learning rate and momentum factor will lead to decrease the speed of learning.

**Why we use learning rate and momentum factor:**

The learning rate and momentum factor give assist to control the convergence speed and the system stability. For example, if the value of the learning rate is very small (learning rate value from 0 to 1), then the algorithm will take very long time to converge. But the momentum factor (momentum factor value from 0 to 1) allows the increment of learning rate**.**

**How momentum neural network works:**

The process of training a neural network is finding values for the error at each node, biases, and weights by using a set if a given input values, the calculated outputs values are near to match the target or the correct values. In details, we have two phase, forward and backward phase. So in the forward phase, we start with those inputs and initial weights to calculate net for each node (net = Σn­i=0 Wi Xi) then we calculate the s(net) for each node (s(net) = ) till we calculate the net and s(net) of the output layer. Then we calculate the error squared for output layer (E2 = (Desired - Output)2), if the error squared less than or equal the MSE(Mean Squared Error usually very small ex. MSE = 0.3) then go to the next sample and do the same previous steps for it, else we move on to the backward phase. In the backward phase we calculate error at each node (error at output layer = Output\*(1 - Output)\*(Desired - Output) and error at other nodes except input layer = Output\*(1 - Output)\*(Σ W\*error) summation of weights multiplied by errors of the nodes linked with the current node) then we calculate the new weights(Updated Weights) of the neural network for all nodes (Wnew = (momentum factor \* Wold) + (Learning rate(η) \* Input that link with the weight \* the error of the node that the arrow pointing for)). After calculating error at each node and updating all the weights we move to the next sample and start all over again from the forward phase. Then after training some samples by the previous steps, we can now test some samples to check if they classified or unclassified by computing the average of the outputs of the output layer then check the result value with the given desired values. If the calculated outputs values are near to match the desired, then it is classified, else it is not classified.