A study of the generalization and performance of no-propagation algorithm compared to back-propagation in single hidden layer Feed Forward Neural networks

Ahmad Hassan, Rana ElNagar

German University In Cairo, Egypt

Abstract

No-propagation algorithm is a relatively new algorithm proposed by widrow et.al as an iterative method of learning in feed forward neural network. In this term paper further investigate the performance and results of the no-propagation algorithm when applied to MNIST dataset compared to back-propagation. We found that no-propagation performed poorly compared to back-propagation. Adding more neurons to the hidden layer driving the output layer did improve the performance of No-propagation.

Keywords: Neural Networks, No-propagation, Back-propagation

1. Introduction

Back-propagation has been the most used algorithm for training feed forward neural networks due to it's robustness and solidity offering successful implementations with good results. The method tunes the weights using gradient decent to minimize the mean square error. The connection between all layers are tuned by the algorithm making it's convergence slow due to having a large computational complexity. Widrow etal [1] presented an alternative method of training a feed forward neural network having the spirit of the extreme learning machine where only the weights connecting the output layer to the hidden layer preceding naming the algorithm no-propagation. In their paper, the algorithm was evaluated on a classification of dataset of chinese characters investigating the effect of addition of noise to the training set. Widrow et.al [1] found that the performance of both algorithms were quite

similar when evaluating the algorithm on noisy datasets even though the back propagation performed better. In section I we introduce back propagation. In section II we introduce No-propagation algorithm and the concept of linear independence which is the key to no-propagation . In section III we evaluate both algorithms on MNIST dataset benchmarking them against each other. In section IV we present the results of the benchmarking.

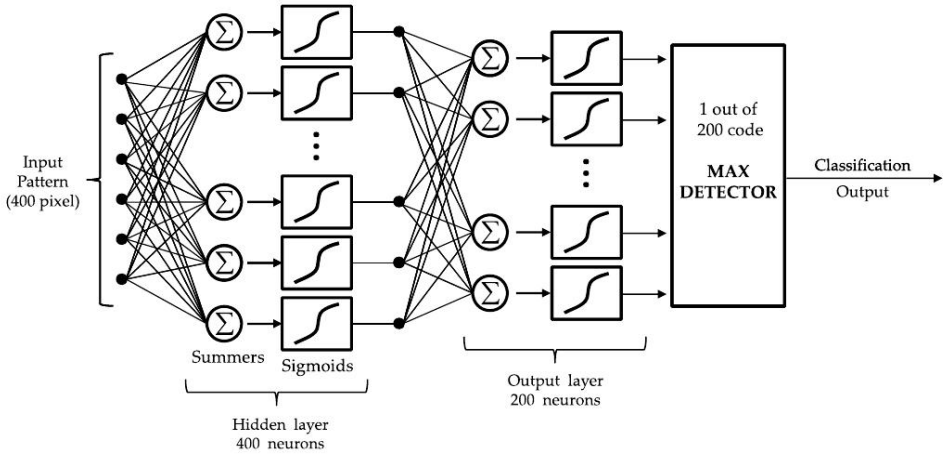

Figure 1: Fig. 1 A 3 layer neural network for classification

2. Backpropagation

The backpropagation algorithm uses steepest decent approaches to tune the weights by adjusting them in the direction of the opposite gradient where

$$\hat{\nabla}_k = \frac{\partial \epsilon_k^2}{\partial \mathbf{W}_k} = \left\{ \begin{array}{c} \frac{\partial \epsilon_k^2}{\partial w_{1k}} \\ \vdots \\ \frac{\partial \epsilon_k^2}{\partial w_{nk}} \end{array} \right\}.$$

ϵ_k^2 is the sum of the squares of error in each of the output neurons. The derivation will not be done here a very simple derivation id provided in [2]. With the presentation of each input vector during training, an instantaneous gradient is obtained of the mean square error with respect to each of the weights (the synaptic weights). Iteratively changing the weights in the direction of the negative gradient leads to an approximate minimum mean square error

solution. The training patterns are presented repeatedly to bring the mean square error, averaged over the set of training patterns, down to minimum. The Back-Propagation algorithm is known to find optimal solutions to the weight settings that are in fact local optima A pseudo code explaining the algorithm :

3. No-prop algorithm

The no propagation algorithm is an iterative approach presented as an alternative to the back-propagation algorithm borrowing the spirit of the extreme learning machine, in the No-prop algorithm weights of all hidden layer neurons are randomly set and fixed and the only learn-able weights are of the last layer of the neural network. The least mean square capacity of a network is defined as the number of distinct patterns that can be presented to the network during training with zero mean square error. The no-prop algorithm relies on the concept of linear Independence. Referring to the network in fig.1 if the number of hidden neurons is equal to or greater than the capacity (the network is said to be trained at capacity), each training pattern undergoes random non-linear transformation then it is possible to perfectly realize the response pattern of the network because they are distinct and linearly independent.

3.1. Linear Independence

Let there be three input vectors $X_1 \to Y_1, X_2 \to Y_2, X_3 \to Y_3$

$$X_1 = \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{n1} \end{bmatrix} X_2 = \begin{bmatrix} x_{12} \\ x_{22} \\ \vdots \\ x_{n2} \end{bmatrix} X_3 = \begin{bmatrix} x_{13} \\ x_{23} \\ \vdots \\ x_{n3} \end{bmatrix} \tag{1}$$

$$Y_1 = F \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{n1} \end{bmatrix} Y_2 = F \begin{bmatrix} x_{12} \\ x_{22} \\ \vdots \\ x_{n2} \end{bmatrix} y_3 = F \begin{bmatrix} x_{13} \\ x_{23} \\ \vdots \\ x_{n3} \end{bmatrix} \tag{2}$$

Y_1, Y_2 and Y_3 would be linearly independent if there exists no α and β to

satisfy the equation :

$$F \begin{bmatrix} x_{13} \\ x_{23} \\ \vdots \\ x_{n3} \end{bmatrix} = \alpha F \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{n1} \end{bmatrix} + \beta F \begin{bmatrix} x_{12} \\ x_{22} \\ \vdots \\ x_{n2} \end{bmatrix} \tag{3}$$

It was shown [1] experimentally that the linear independence depends on the number of weights of each of the output layer neurons which is controlled by the number of neurons in the hidden layer driving it.

Figure 2: Fig. 2 Different version of a digit

4. Evaluation of the algorithms

In this section we present an experiment of applying both methods of training a single hidden layer neural network for the purpose of classifying hand written digits of the MNIST dataset and bench-marking them against each other and investigating the effect of increasing the number of neurons of the hidden layer on the classification results.

The 3 layer feed forward neural network shown in fig.1 was used. The MNIST dataset consists of 60000 training examples and 10000 test examples of handwritten digits in the form of size normalized gray-scale images. Fig 2 shows different examples of the same character. Each image in the dataset had 28 x 28 pixels. The input vector consisted of 784 components corresponding to a component for every pixel in an image. The network was

trained with all the training patterns and was evaluated on all the test patterns. The classification error and the training mean square error was noted. The process was done twice using 784 neurons in the hidden layer and by increasing the number of neurons to 1500.

	Back prop	**No prop**
Classification error	391	1624

Table 1: 748 neurons in the hidden layer

	Back prop	**No prop**
Classification error	270	723

Table 2: 1500 neurons in the hidden layer

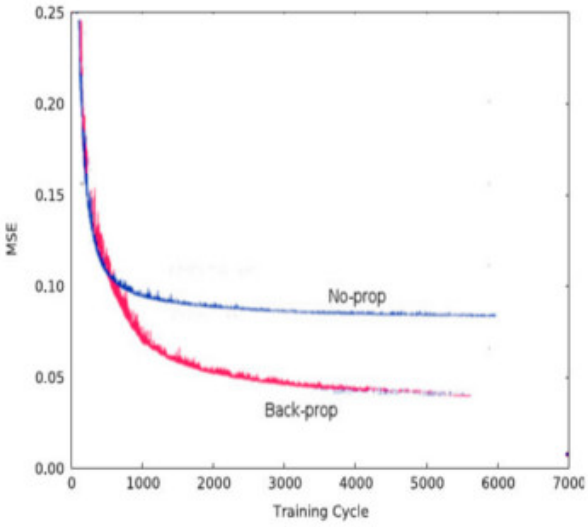

Figure 3 : Training with 784 hidden neurons

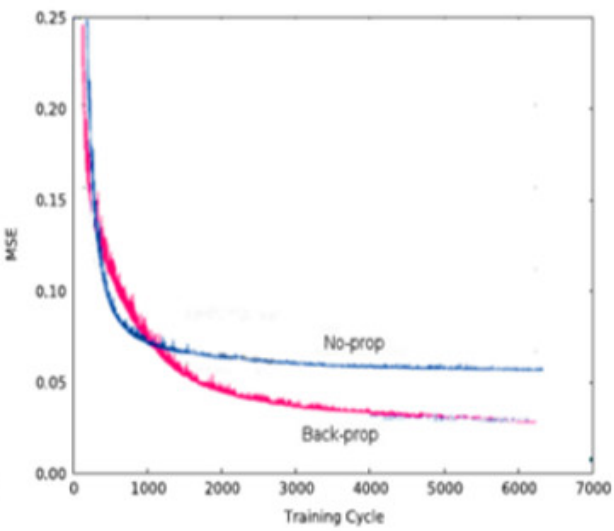

Figure 4 : Training with 1500 hidden neurons

Table 1 shows the results of the classification using 784 neurons in the hidden layer. It shows a huge difference between no-prop and back-prop the performance of the back-prop was unrivaled . Increasing the number of hidden neurons in the second experiment to 1500 benefited both of the algorithms performance however it benefited the no-prop more but the results

still show that they are not close to each other. The back-prop outperformed the no-prop.

The time of convergence for the no-prop was significantly faster than the back-prop . Fig 3 shows the change in mean square error during training cycles.

5. Conclusion

The no-prop algorithm does not give a comparable performance to the back-propagation algorithm and the classification error obtained was significantly larger. The no-prop algorithm however improves with the addition of extra neurons to the hidden layer which is in most cases unfeasible or when training at capacity which is similar to brute force and defeats the purpose.

[1] B. Widrow, A. Greenblatt, Y. Kim, D. Park, The no-prop algorithm: A new learning algorithm for multilayer neural networks, Neural Networks 37 (2013) 182–188.

[2] B. Widrow, M. A. Lehr, Backpropagation and its applications, INNS Summer Workshop on Neural Network Computing for the Electric Power Industry (1992) 21–29.