NAME: Sanjana Sandeep
SID: 861245086
DATE: 12/07/2016
COURSE: CS 229

1 Introduction

In this experiment, a Neural Network for the handwriting recognition problem is presented. The feed forward neural network is improved with parameter tuning and finally the effect of bagging on neural networks is explored.

2 Approach to the Problem

The type of problem presented here falls into the pattern recognition class. The Feed-forward backpropagation Neural Network is chosen, as it can fit any type of function including nonlinear ones. And since the goal is to classify and not to interpret, neural networks is a good choice. For the implementation, the Neural Network pattern recognition toolbox function- *nprtool* is used.
The training function used is *Scaled Conjugate Gradient*. The choice is based on the fact that it improves the speed of convergence, adapts the learning rate and also works well with *early stopping*. Now, since the error is back-propagated, the transfer function must be differentiable, smooth, monotonic and bounded. The *tan-sigmoid* transfer function is used, since it is faster than other differentiable transfer functions. The loss function is the cross-entropy loss function.
In most situations, the performance improvement with a second hidden layer is very small. One hidden layer should suffice.(This consensus was reached, by comparing the effective performance in both cases, resulting in a very small difference). The data is divided into the training set- 75%, the test set - 10% and the validation set- 15 %.This is done for the purpose of evaluating the classifier, the final classifier is trained with the entire dataset.
To aid against overfitting, *early stopping* is used, with 6 validation checks. In addition to early stopping, weight decay is used to keep the weights small. Also, random restarts is implemented since the learning is through gradient descent (avoids getting stuck in local minima).
Now, the objective is to tune the free parameters of the neural network.

3 Initial Results and Analysis

The parameters- weight decay (λ) and the number of hidden units (n_h) are varied, to get different results.The neural network for each combination is trained with random restarts(5 times).

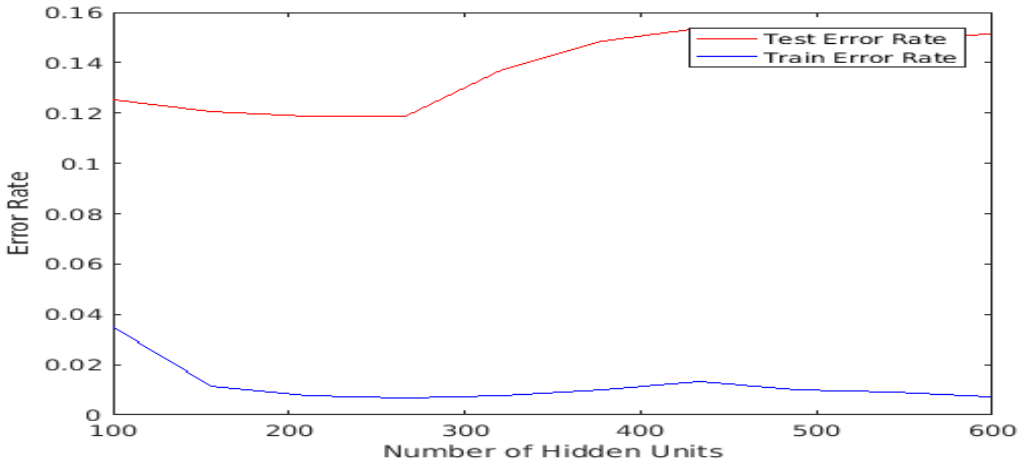

Fig 1. Illustrates the effect of increasing the number of hidden units on the error rate. The plot corresponds to the best values from the random restarts.

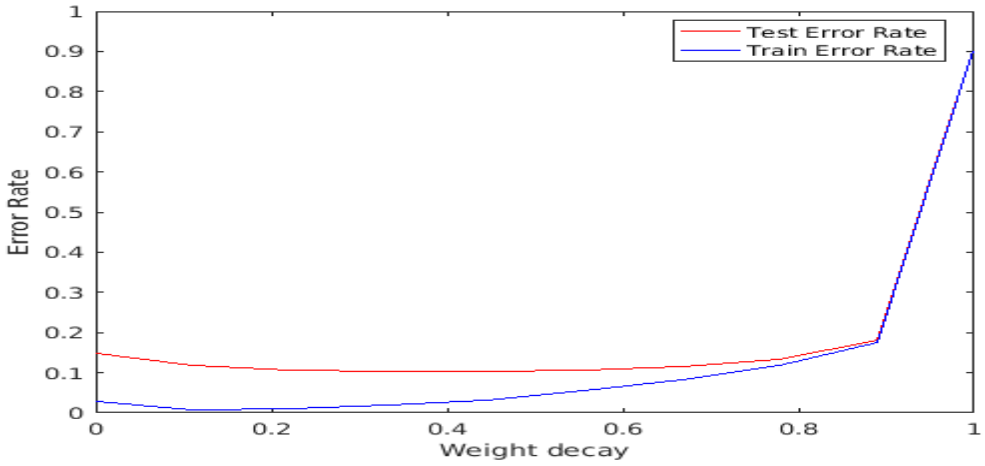

Fig 2. Illustrates the effect of varying the weight decay values on the error rate. The plot corresponds to the best values from the random restarts.

Comparing the different Neural Network architectures, it is noticed that the best training accuracy is achieved when $\lambda \approx 0.1$ and $n_h \approx 250$.
The neural network that is obtained from this set of parameters has the following values of test and training accuracies for 5 random restarts:

Training Accuracy:	0.9747	0.9930	0.9921	0.9928	0.9585
Test Accuracy:	0.8508	0.8795	0.8805	0.8789	0.8505

Therefore, the best training accuracy is 99.3%. Hence, it is highly powerful. However, the best test accuracy achieved is only 88.05%. The wide gap between the test and the training accuracies, show that the network doesn't generalize well on new data, and runs into the problem of overfitting. In addition, since there is not much of a variance between the 5 different experiments, it can be inferred that there are not many local minima that are encountered.

4 Description of Improvement

In order to reduce the the variance and improve the test accuracy, an ensemble neural net is proposed. Feature engineering could also reduce the variance, but since the point of neural networks is to automatically engineer the features, this idea is dropped. Bagging is chosen as opposed to boosting, since the neural network is already powerful (high training accuracy) and boosting is designed for a weak learner. Multiple neural networks are trained on various structures of training data to reduce model variance.

5 Final Results and Analysis

To test whether this hypothesis actually holds good, there is a need to get the most optimal number of classifiers in bagging.

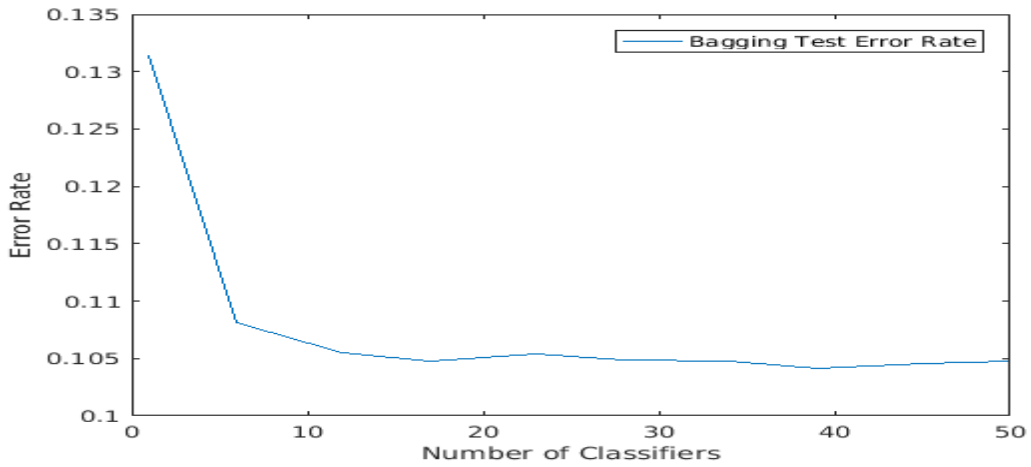

Fig 3. Plot of test error comparisons for the bagged neural network.

From the figure, it can be noticed that the single best neural network, gives a test accuracy of 86.86%. The test accuracy increases rapidly, by averaging multiple neural nets, and remains constant after adding 10-15 classifiers.

The best test accuracy achieved is 89.53%. Also, increasing the number of classifiers beyond 10 might not be helpful, since it increases the execution time, without any drastic improvement in the test accuracy. Therefore, it is ideal to fix the number of classifiers to 10. There is an improvement in the test accuracy of 3% by bagging neural-nets. Therefore, bagging does enhance the performance of the classifier by reducing the variance. After having analyzed the results, the final classifier is then trained on the whole data-set, to achieve a better accuracy.

Bottom line- the bagging technique works well with not just decision tree classifiers but also neural networks.

References

[1] Ruqing Chen and Jinshou Yu. Conference: Natural Computation, 2007. ICNC 2007.

[2] Zhuo,Zheng. "Boosting and Bagging of Neural Networks with Applications to Financial Time Series."

[3] MathWorks- Neural Net Pattern Recognition and Classification app.