Handwriting Recognition

Final Project

CS 229: MACHINE LEARNING

December 6, 2016

Authored by: Nikhil Kamthe (861245635)

Introduction

Handwriting recognition is a common problem encountered in automating day to day tasks like identifying postal addresses on envelops, number plate for vehicles etc. A complete handwriting recognition system handles formatting, performs correct segmentation, entails character recognition etc. This project aims to solve the problem of character recognition by building a classifier which can use the scanned pixel data of an image to identify the character.

Approach to the Problem

Neural Networks and SVM are two of the most widely used classifiers for Handwriting recognition [1][2]. Although evidence suggests that SVM performs better at solving this problem [3], I have selected Neural Network for this project to demonstrate how it can be tuned to give better results. The initial approach uses the default configuration provided by Matlab viz. Neural Network Toolbox with some basic parameter tuning. Following section explains how the optimal value for these parameters was selected.

Initial Results and Analysis

Although Neural Networks are good at understanding behavior of several systems, tuning it is not trivial. As an initial approach, following parameters were tuned to check their effect on the accuracy.

Training and Transfer Function

Performance of following training functions were compared to decide which one to choose for the given dataset viz. Scaled Conjugate Gradient (SCG), Conjugate Gradient with Powell/Beale Restarts (CGB), Resilient Backpropagation (NRP), Polak-Ribiére Conjugate Gradient (CGP), Fletcher-Powell Conjugate Gradient (CGF), Variable Learning Rate Gradient Descent (GDX), One Step Secant (OSS). Looking at the error rate (*Figure 1*) and time taken for training (*Figure 2*), Scaled Conjugate Gradient (SCG) was selected as the training algorithm. Also, using Hyperbolic tangent sigmoid as transfer function gave max accuracy.

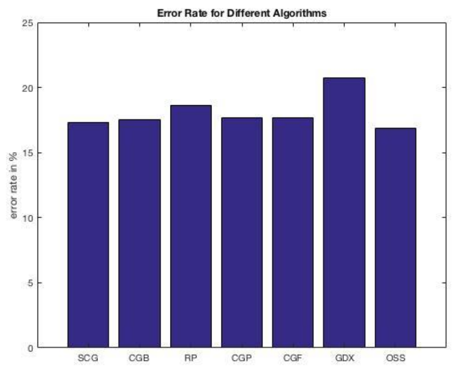

Figure 1

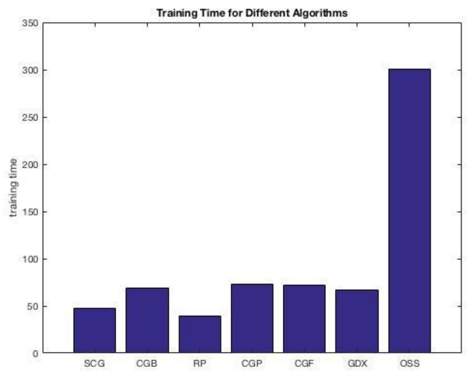

Figure 2

Number of Hidden Layers and Units

Though there are certain general methods to select the number of hidden layers and hidden units in each of these layers, these parameters generally differ for different problems. In this project, various experiments were conducted to find the optimal number each of these for the given dataset; by changing the value of these parameters and comparing the error values.

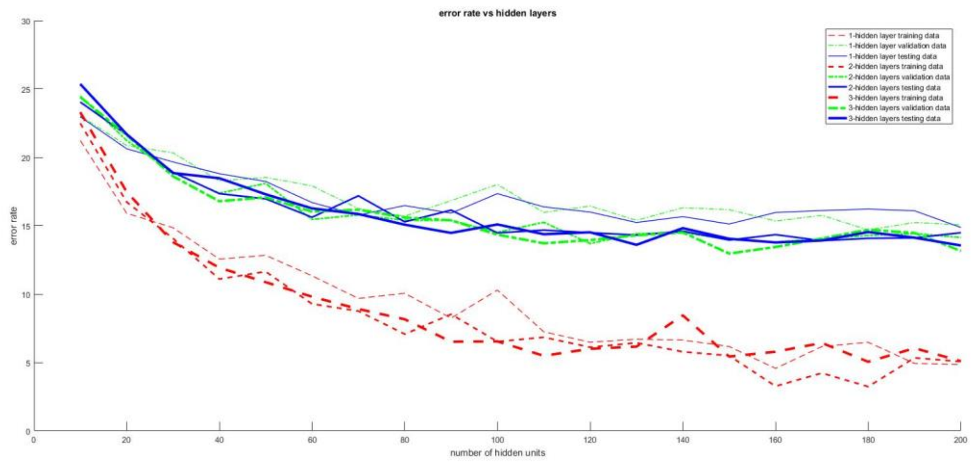

Figure 3

For the given dataset, increasing the number of hidden units in a layer increased the accuracy of the classifier but increasing the number of hidden layers did not improve the accuracy much (refer *Figure 3*). Hence a Neural Network with 1 hidden layer and 150 hidden units was chosen for further analysis.

Regularization Parameter

One of the biggest problem of Neural Networks is overfitting. As the number of iterations increases, the network learns the training data too well to fit even the outliers. One of the methods of avoiding this and improving network generalization is early stopping [4], which is automatically provided along with the Neural Network Toolbox. Another one is changing the value of regularization parameter. Experiments were conducted to select an optimal value of this regularization parameter for the given dataset. *Figure 4* shows the results of these experiments. Using these observations, 0.1 was chosen as the value of the regularization parameter

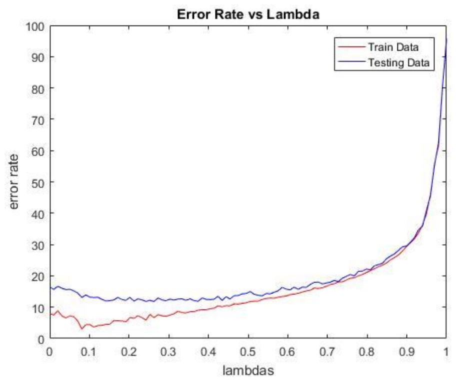

Figure 4

After tuning these parameters, the accuracy of the classifier increased to 87% from 76%.

Description of Improvements

Even after using early stopping for improving network generalization, the difference between the training error and testing error was significant (refer *Figure 5*). It shows that the classifier still had high variance. To improve the variance of the resulting classifier, a feature selection algorithm was run on the dataset to reduce the number of features. The algorithm was designed to combine both forward and backward selection methods. During each iteration, a best feature was added using forward selection and worst among the selected features is removed using backward selection. As a second improvement, Bagging was used to improve the variance of our classifier.

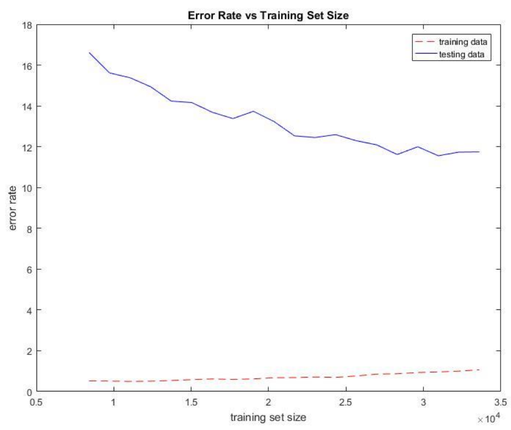

Figure 5

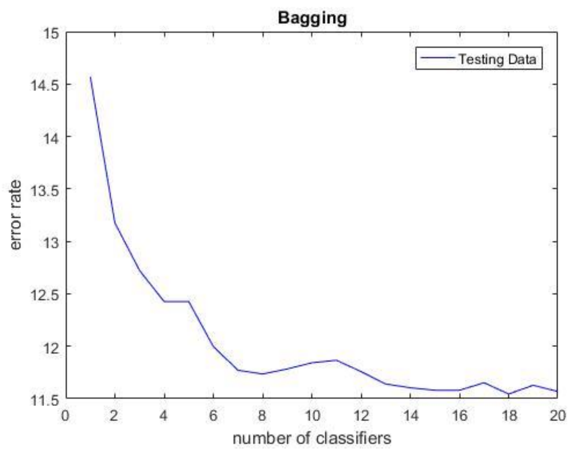

Figure 6

Final Results and Analysis

Unfortunately, the features returned by the feature selection algorithm did not improve the accuracy (it is slightly less than before). One of the reasons for this might be the fact that the algorithm is greedy in nature and doesn't always give the highest accuracy (global maxima), but can give a reasonable accuracy with reduced number of features (which reduces both training and testing time). Features returned by a simple forward selection did not improve the accuracy either, for similar reason. Another explanation could be the fact that Neural Network doesn't really need feature selection and performs feature engineering on its own unlike classifiers like SVM. Further analysis needs to be done to confirm this inference.

Bagging on the other hand was successful in reducing the variance; which is apparent from the fact that testing error was reduced. *Figure 6* shows how changing the number of classifiers (in bagging) affected the accuracy of our classifier. After analyzing these results, a bag of 15 Neural Networks were selected for bagging. Bagging improved the accuracy of classifier by about 2% (from 87% to 89%).

References

[1] R Plamondon, S.N Srihari, On-line and off-line handwritten recognition: a comprehensive survey IEEE Trans. Pattern Anal. Mach. Intell., 22 (2000), pp. 62–84

[2] Pal U., Chaudhuri B.B. (2004): Indian script character recognition: a survey. Pattern Recognit. 37, 1887–1899

[3] Chayaporn Kaensar: A Comparative Study on Handwriting Digit Recognition Classifier Using Neural Network, Support Vector Machine and K-Nearest Neighbor, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013

[4] Matlab: Neural Network Toolbox Documentation

[5] Hagan, M.T., H.B. Demuth, and M.H. Beale, Neural Network Design, Boston, MA: PWS Publishing, 1996