Predicting handwritten Digits from the United States Postal Service (USPS)

Introduction

In this report, multilayer neural network is used to predict the digit type using 265 input features corresponding to the grid representation of the images. The file contains data recording handwritten digits from the United States Postal Service (USPS). More in details, the data consists of grayscale (16x16) grid representations of image scans of the digits "0" through "9" (10 digits). The data is divided into train and test data, containing respectively 7291 and 2007 observations.

For the data training and test, 256 input features are stored in vectors:

- x_train = the input data for training
- x_test = the input data for test
- y_train = the target class labels for training
- y_test – the target class labels for test

Here is the process detail of training, validation and performance assessment of the data in order:
Part A - Start with 2 hidden layers to predict the type of digit, and add the regularization
Part B - Start a new model by adding 1 extra hidden layer, and add the regularization

Here are the steps of data preparation:

1. **Standardizing the input data**
   When the file is loaded, the input data such x and x_test are large matrices, so we need to normalize them. Meanwhile, the target class data (y and y-test) are encoded with keras function: to_categorical.

2. **Plot the data**

```
# plot few example digits
par(mfrow=c(1,3),mar=rep(1.5,4))
plot_digit(111,x_train)
plot_digit(222,x_train)
plot_digit(333,x_train)
```

By plotting the data, we can see the nature of the data.

In this part, I created an initial neural network with two hidden layers using ReLU activation. ReLU activation is known better than any other activation because it does not activate all the neurons at the same time, leads to relatively efficient computation. For output activation, Softmax function is often described as a combination of multiple sigmoids which widely used for binary classification problems.

In the first layer there are 256 neurons and following by the second layer which is 128 then the third layer is 64. The number of units in the output layer should be 10, each for ten available digits (0-9). Here is the set of estimated error:

Hyperparameter	Training*	Test**	Training Reg	Test Reg
Part A - 2 hidden layers				
0.0005	0.001645863	0.063278496	**0.003017426**	**0.062282026**
0.0010	0.001645863	0.063278496	0.005349040	0.061783731
0.0025	0.001645863	0.063278496	0.011383891	0.061783731
0.0040	0.001645863	0.063278496	0.016870141	0.064275026
0.0050	0.001645863	0.063278496	0.018104494	0.064275026
0.0090	0.001645863	0.063278496	0.028254032	0.072247148
0.0100	0.001645863	0.063278496	0.030448496	0.074240148
Part B - 3 hidden layers				
0.0005	0.000411451	0.063278496	0.000685751	0.059292495

** and ** didn't use any hyperparameter*

Discussion

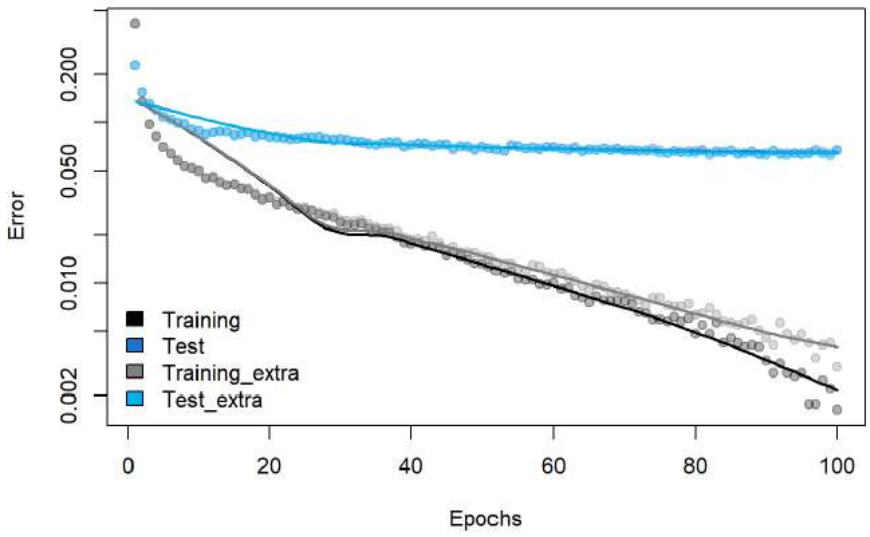
The best neural network model is the one with general characteristic and the lower estimated of test error. Another finding is more layers we have in a model, there is a big risk of overfitting, and regularization is expected to be one of ways to tackle this issue.

First, the difference between 2 layers and 3 layers is the training error, it is getting very low for 3 layers model training data (0.000274301) because the data is more augmented and the 'representative' for each layer tend to fit all the data points which will be **overfitting.** In this case the training accuracy is very high, but validation accuracy will be relatively close to the one with 2 layers. That is not a significant improvement. So, the potential predictive performance improvement obtained by adding one extra hidden layer to the network architecture is not relevant for this data.

Second, based on trial and errors, I changed the hyperparameter of the L2 regularization for 2 layers neural network (see green and purple columns for the error estimations). L2 regularization is known explicitly removes the effect of collinearity by modifying the covariance matrix. And regularizes allow to apply penalties on layer parameters or layer activity during optimization. These penalties are incorporated in the loss function that the network optimizes.

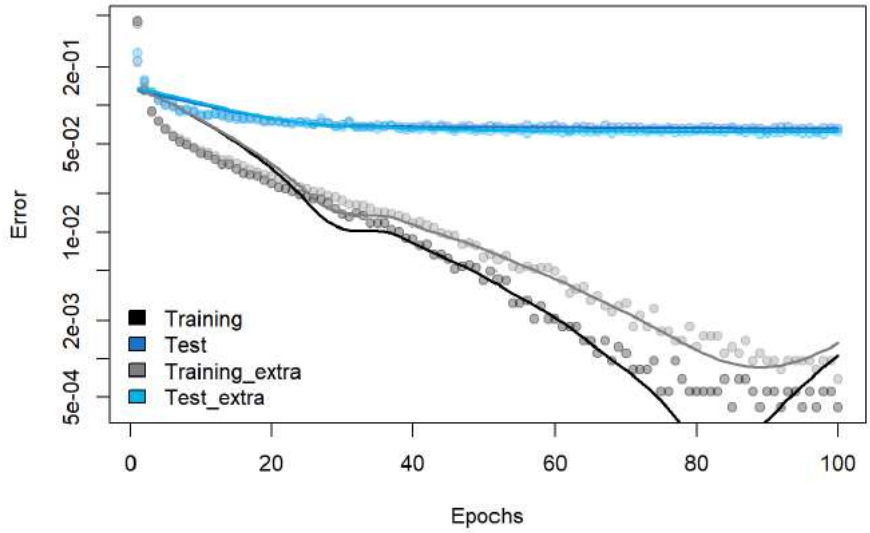
The use of one approach for regularization shown that the regularization on 2 layers is not worth it, because the test error of the regularized model is not significantly lower than the test error of the unregularized one even with the smaller magnitude of hyperparameter. Furthermore, maybe we need to train the regularization with larger number of epochs. We can compare part A and part B results in the next page.

Below is the plot of 2 layers model **without regularization** and **with regularization**



.

And the plot of 3 layers model **without regularization** and **with regularization**



In my opinion, with these conditions we will need a larger number of epochs and more tuning to decide the regularization. Otherwise, it can be considered as wise decision to choose two layers neural network without any regularization in this case since so far, the simplest model is good enough to support the data prediction.

Two layers model with 256 neurons, 128 neurons and 10 output fit general characteristic and has low errors. More number of epochs can be considered to see the improvement of predictive performance.

The code is adapted from:

Machine Learning and AI – Practical Lab 5 (with additional modification).