Handwritten Digit Recognition

Shruti Bendale
Person Number: 50289048,
School of Engineering and Applied Sciences,
University at Buffalo, Buffalo, NY, 14214
shrutitu@buffalo.edu

Abstract

The purpose of this project is to compare classify images to recognize the handwritten digits. This task was performed using different classifiers, namely, logistic regression classifier, Neural Networks, Support Vector Machine and Random Forests. The results of these classifiers were compared and analyzed by generating a confusion matrix. The predictions of all these classifiers were combined to design a combined classifier that works on a majority voting system.

1. Introduction

The problem of handwriting recognition is to interpret intelligible handwritten input automatically, which is of great interest in the pattern recognition research community because of its applications to many fields. As one of the fundament problems in designing practical recognition systems, the recognition of handwritten digits is an active research field. Immediate applications of the digit recognition techniques include postal mail sorting, automatic address reading and mail routing, bank check processing, etc.

In this report we train and test a set of classifiers on the MNIST database for pattern analysis in solving the handwritten digit recognition problem. MNIST is a publicly available and widely used dataset consisting of bilevel images of handwritten digits and labels. It is divided into 60000 images for training and 10000 images for testing, where each image is of the size 28 x 28 pixels. We use both the training and the testing samples for the purpose of this project.

The USPS dataset was used to test the models. It contains 20000 images of handwritten digits (200 samples for each digit). We use these samples to test the models generated using the MNIST dataset. The USPS images were resized to fit the size of 28 x 28 pixels.

Four different models (Logistic, NN, SVM and random forests) are applied to compare the performance in terms of both accuracy and speed. Potential improvements including combinations of the classifiers using majority voting are further discussed in this report.

2. Implementation

2.1. Multinomial Logistic Regression:

Multinomial logistic regression is a classification method that generalizes logistic regression to multiclass problems, i.e. with more than two possible discrete outcomes. We have 10 possible discrete outputs for our digit recognition problem.

Softmax function:

The softmax function generates probabilities for each of these output classes and the class with the highest probability is chosen as the predicted class. The equation for the softmax function is given as:

$$softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)}$$ where, $z_i = x_i w_i + bias$

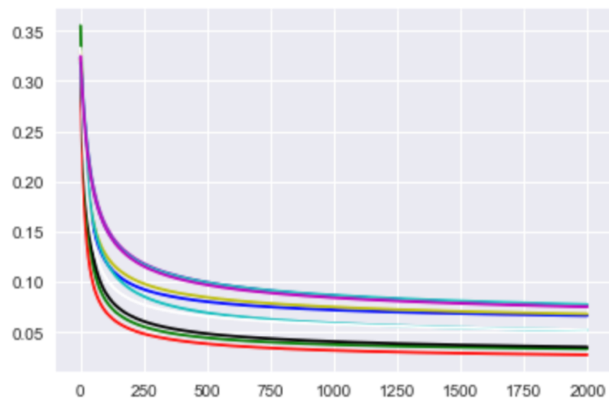
The weights ($w_1, w_2, \ldots, w_i$) represent the learned likelihood of a pixel contributing positively or negatively to the overall image being in a certain class. Thus, the value of the pixel, multiplied with the learned weight, gives a kind of "vote" towards the final result.

Gradient Descent:

The weights are learned through an iterative process called gradient descent and back-propogation whereby error is attributed to specific weights with each prediction. These weights are modified with each iteration to improve the prediction. In this case, we use an error metric called **cross-entropy loss** that can be given by the equation:

$$H_{y'}(y) = -1/m \sum_i y'_i * log(softmax(y_i))$$

The decrease in the cross entropy error for all 10 classes after gradient descent can be seen in the following loss vs number of epochs graph:



Accuracy and Confusion Matrix:

After testing the model performance on the MNIST testing dataset, the accuracy was found to be 91.41% and after testing the performance on the USPS dataset, the accuracy was found to 35.787%. The confusion matrix for the MNIST dataset is given in *figure 2.1* and that for the USPS dataset is given in *figure 2.2*.

```
[[ 956    0   11    4    1   10   13    3    8   11]
 [   0 1105    8    1    4    4    3   15    9    7]
 [   2    2  903   21    5    4    4   22    7    4]
 [   2    3   14  911    1   41    3    7   27   12]
 [   1    0   15    0  912    9   13    8    9   42]
 [   6    3    1   28    0  760   10    0   23    9]
 [   9    4   13    4   10   17  908    0   13    0]
 [   1    1   17   12    2   10    1  936   13   21]
 [   3   17   42   20    8   30    3    3  853    6]
 [   0    0    8    9   39    7    0   34   12  897]]
Overall accuracy: 91.41 %
```

figure 2.1: Confusion Matrix for MNIST

```
[[ 548  176  165   77   42  138  273  159  193   32]
 [   2  345   21    1   77   19   10  217   29  164]
 [ 320  168 1221  140   44  201  392  320  151  154]
 [  61  327  154 1288   59  166  105  460  217  477]
 [ 240  258   55   20  994   40   97   69  121  147]
 [ 215  103  121  301  160 1190  333  155  720  118]
 [  90   32   87   14   42  108  693   26  110   14]
 [  67  395   87   67  160   72   22  348   52  404]
 [ 127  176   63   52  258   42   42  193  340  300]
 [ 330   20   25   40  164   24   33   53   67  190]]
Overall accuracy: 35.78678933946697 %
```

*figure 2.2: Confusion Matrix for USP**S***

2.2. Neural Network:

1.1.1. Deep Neural Network:

We construct a 2-layer deep neural network using tensorflow, where, the input layer has 784 nodes and the output layer has 10 nodes. Each node of the input layer represents a pixel in an image to classify and each node in the output layer represent a distinct target class.

The hyperparameters that were set are as follows:
Hidden layer nodes: 2048
Dropout: 0.2
Activation for hidden layer: relu
Activation for output layer: softmax
Optimizer: adam
Loss: categorical crossentropy

Accuracy and Confusion Matrix:
After testing the model performance on the MNIST testing dataset, the accuracy was found to be 98% and after testing the performance on the USPS dataset, the accuracy was found to 49.52%. The confusion matrix for the MNIST dataset is given in *figure 2.3* and that for the USPS dataset is given in *figure 2.4*.

```
Test loss: 0.09019882881085704
Test accuracy: 0.98
Confusion Matrix:
 [[ 974    1    0    0    1    0    0    1    1    2]
 [    0 1130    2    1    0    0    0    0    2    0]
 [    5    2  999    3    1    0    1    8   13    0]
 [    1    0    6  994    0    2    0    3    2    2]
 [    1    1    4    0  963    0    1    2    3    7]
 [    4    0    0   17    1  863    3    0    3    1]
 [    7    4    0    1    4    6  933    0    3    0]
 [    1    2    5    3    1    0    0 1008    2    6]
 [    5    0    2    3    4    2    1    2  950    5]
 [    3    3    0    4    8    1    0    3    1  986]]
```

figure 2.3: Confusion Matrix for MNIST

```
Test loss: 5.157580578849938
Test accuracy: 0.4952747637292458
Confusion Matrix:
 [[ 702    3  182   35  270  125  153   91  113  326]
 [   43  566  393   92  311   63    7  366  107   52]
 [   73   10 1648   30   20   66   55   47   43    7]
 [   34    9  198 1454    8  194    6   18   66   13]
 [   22   75   48   13 1222   95   29  223  241   32]
 [   68    1  173   57   18 1465   51   18  143    6]
 [  196   19  420   17   49  115 1015   68   33   68]
 [   29  144  312  309   55   23   22  874  221   11]
 [  190   14  175  292  134  227  116  110  710   32]
 [    9   66  109  197  269   30    6  670  395  249]]
```

figure 2.4: Confusion Matrix for USPS

1.1.2. Convolutional Neural Network:

A 2-layer CNN was constructed with the input as (28,28). A CNN does not require flattening of the input as opposed to the DNN. The optimal hypreparameters found after fine tuning are:
Hidden layer nodes: 128
Dropout: 0.2
Activation for hidden layer: relu
Activation for output layer: softmax
Optimizer: adam
Loss: sparse categorical crossentropy

Accuracy and Confusion Matrix:
After testing the model performance on the MNIST testing dataset, the accuracy was found to be 98.52% and after testing the performance on the USPS dataset, the accuracy was found to 48.76%. The confusion matrix for the MNIST dataset is given in *figure 2.5* and that for the USPS dataset is given in *figure 2.6*.

```
Test loss: 0.0560671895022675
Test accuracy: 0.9852
Confusion Matrix:
 [[ 960    0    2    0    0    1   12    1    2    2]
 [    0 1126    4    0    0    0    2    3    0    0]
 [    0    0 1021    0    0    0    2    7    2    0]
 [    0    0    0  999    0    5    0    3    3    0]
 [    0    1    4    0  965    0    2    0    1    9]
 [    1    0    1    5    0  881    2    1    1    0]
 [    3    2    0    0    1    2  949    0    1    0]
 [    0    1    7    0    0    0    0 1018    0    2]
 [    2    0    3    5    1    5    0    4  950    4]
 [    1    3    0    6    5    6    1    4    0  983]]
```

figure 2.5: Confusion Matrix for MNIST

```
Test loss: 4.800172557170179
Test accuracy: 0.4875743787099948
Confusion Matrix:
 [[ 387    1  253   55  226   45  173   23   47  790]
 [   36  467  184   56  541   89   77  300  190   60]
 [    5    6 1737   34   22   77   42   32   37    7]
 [    0    2  148 1175    5  552   14   36   51   17]
 [    1   13  123   59 1101   28    9  195  428   43]
 [    3    1   52  210    7 1636    6   17   24   44]
 [   20    6  458   17   60  154 1243    3   25   14]
 [    3   19  275  524   21   45   24  926  139   24]
 [   15    2  131  528   16  375   27   64  664  178]
 [    2    0  162  616   59   20    6  216  504  415]]
```

figure 2.6: Confusion Matrix for USPS

2.3. Support Vector Machine:

SVM generates a hyperplane that acts as a separator between the classes. If the classes are not linearly seperable, the points are projected into higher dimensions in order to calculate a hyperplane. This is called as the kernel trick. 'rbf' kernel is a radial basis function given by the following formula:

$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right)$ where $||x- x'||^2$ is the squared Euclidean distance between two data points x and x′.

An SVC classifier using an RBF kernel has two parameters: gamma and C.

- gamma is the decision region. When gamma is low, the 'curve' of the decision boundary is very low and thus the decision region is very broad and vice versa.
- C is a parameter of the SVC learner and is the penalty for misclassifying a data point. Higher the value of C, higher the penalty.

SVM was implemented using the *'svm.SVC(kernel=kernel, C=c,gamma=gamma)'* function in sklearn.metrics library. The best parameters found after tuning are:
kernel='rbf', C=2, gamma=0.001

Accuracy and Confusion Matrix:
After testing the model performance on the MNIST testing dataset, the accuracy was found to be 95% and after testing the performance on the USPS dataset, the accuracy was found to 41%. The confusion matrix for the MNIST dataset is given in *figure 2.7* and that for the USPS dataset is given in *figure 2.8*.

```
Confusion matrix:
[[ 967    0    1    0    0    5    4    1    2    0]
 [   0 1122    2    3    0    1    3    1    3    0]
 [   7    1  969    8   10    2   10   10   13    2]
 [   1    1   18  947    1   16    1    9   12    4]
 [   1    1    7    0  936    0    7    2    2   26]
 [   7    4    5   32    6  809   13    1   10    5]
 [   9    3    4    1    5    8  927    0    1    0]
 [   1   12   23    6    7    1    0  960    3   15]
 [   4    5    7   15    8   24   10    7  892    2]
 [   8    6    1   14   29    4    1   14    5  927]]
```

figure 2.7: Confusion Matrix for MNIST

```
Confusion matrix:
[[ 580    2  424   22  265  253   68   52    6  328]
 [ 102  413  311  154  264  173   43  500   23   17]
 [ 126   16 1416   67   38  192   60   55   19   10]
 [  69    5  199 1128    8  479    5   64   30   13]
 [  15   53   95   15 1150  251   25  220   74  102]
 [ 104   18  281  116   20 1337   61   39   19    5]
 [ 178    6  521   31   89  381  749   10    6   29]
 [  44  208  433  301   57  403   15  460   56   23]
 [  73   22  220  203   81 1000   95   40  242   24]
 [  18  150  237  293  188  158    6  510  212  228]]
```

figure 2.8: Confusion Matrix for USPS

2.4. Random Forests:

Random forest builds a forest of multiple decision trees and merges them together to get a more accurate and stable prediction. The forest it builds, is an ensemble of Decision Trees, most of the time trained with the "bagging" method. The general idea of the bagging method is that a combination of learning models increases the overall result.
It was implemented using the '*RandomForestClassifier()*' function in sklearn. RandomForestClassifier library.

Accuracy and Confusion Matrix:
After testing the model performance on the MNIST testing dataset, the accuracy was found to be 94.46% and after testing the performance on the USPS dataset, the accuracy was found to 32%. The confusion matrix for the MNIST dataset is given in *figure 2.9* and that for the USPS dataset is given in *figure 2.10*.

```
random forest accuracy:  0.9446
Confusion matrix:
[[ 966    0    1    0    0    5    3    1    3    1]
 [   0 1118    3    3    0    2    5    0    3    1]
 [  10    2  990   11    1    0    4    7    7    0]
 [   2    3   19  940    1   16    2    8   13    6]
 [   1    4    8    1  924    0    6    1    6   31]
 [   8    3    1   38    7  817    5    2    8    3]
 [  16    5    7    0    9    5  913    0    3    0]
 [   2    6   21    5    3    2    0  977    2   10]
 [   9    2   12   12   10   17    6    6  890   10]
 [  10    6    5   13   37    7    0   13    7  911]]
```

figure 2.9: Confusion Matrix for MNIST

```
Confusion matrix:
[[608  36 273  94 331 170 119 153  11 205]
 [ 87 642 187  96  90 177  37 640  23  21]
 [204 132 939 130  52 274  66 163  26  13]
 [109  89 262 962  41 350  28 106  26  27]
 [ 43 260 139  88 789 153  61 352  31  84]
 [231 101 213 238  72 918  62 114  18  33]
 [291  92 333  73 147 366 545 117  14  22]
 [ 75 407 480 212  41 139  46 550  24  26]
 [105 129 282 276 151 685 102 110 110  50]
 [ 67 274 367 309 203 165  30 409  71 105]]
```

figure 2.10: Confusion Matrix for USPS

2.5. Combined Classifier (Majority voting):

For each sample, we compare the predicted values of all classifiers. The value that was predicted the most number of times is the predicted value of the combined classifier.
For example, For a sample x,
Logistic model predicts 2, CNN predicts 7, DNN predicts 7, SVM predicts 1, Random forests predicts 9
The final prediction of the combined model will be 7 (as it was predicted by 2 out of 5 models)

Accuracy and Confusion Matrix:
After testing the model performance on the MNIST testing dataset, the accuracy was found to be 97.53% and after testing the performance on the USPS dataset, the accuracy was found to 44%. The confusion matrix for the MNIST dataset is given in *figure 2.11* and that for the USPS dataset is given in *figure 2.12*.

```
Combined accuracy using majority voting:  0.9753
Confusion matrix:
[[ 978    0    5    1    0    1    1    0    2    0]
 [   0 1128    0    1    0    2    2    2    3    0]
 [   6    0  995    5    0    1    3    1    6    1]
 [   1    1    7  994    0    6    2    2    3    0]
 [   1    1    3    0  964    0    3    1    2   11]
 [   4    0    2   10    0  857    6    1    5    3]
 [   7    2    2    1    2    1  941    0    3    0]
 [   3    4    8    2    3    1    0 1009    3   11]
 [   3    0    5    6    4    4    4    3  928    3]
 [   5    5    1   10   11    2    0    4    6  959]]
```

figure 2.11: Confusion Matrix for MNIST

```
Combined accuracy for USPS using majority voting:  0.44
Confusion matrix:
[[ 639    2  342   43  285  158   65   51   50  365]
 [ 121  512  201  161  284  109   25  507   65   15]
 [ 118   11 1501   73   37  111   54   56   31    7]
 [  57    1  141 1395   12  296    4   48   38    8]
 [  20   80   48   25 1205  157   18  192  177   78]
 [  99   14  188  110   21 1445   53   36   26    8]
 [ 217   12  398   45   85  301  877   13   20   32]
 [  69  195  346  384   43  157   16  645  130   15]
 [ 133   21  163  257   92  755   86   64  383   46]
 [  20  148  166  383  163  105    6  500  290  219]]
```

figure 2.12: Confusion Matrix for USPS

3. Inferences

A comprehensive summary of the testing accuracies for all the six classifiers is shown in *table 3.1* and *figure 3.*

Accuracy	**Logistic**	**DNN**	**CNN**	**SVM**	**Random Forests**	**Ensemble Classifier**
Testing MNIST	91.41%	98.52%	98%	95%	94.46%	97.53%
Testing USPS	35.79%	48.75%	49.52%	41%	32%	44%

Table 3.1: Testing accuracies of all the classifiers

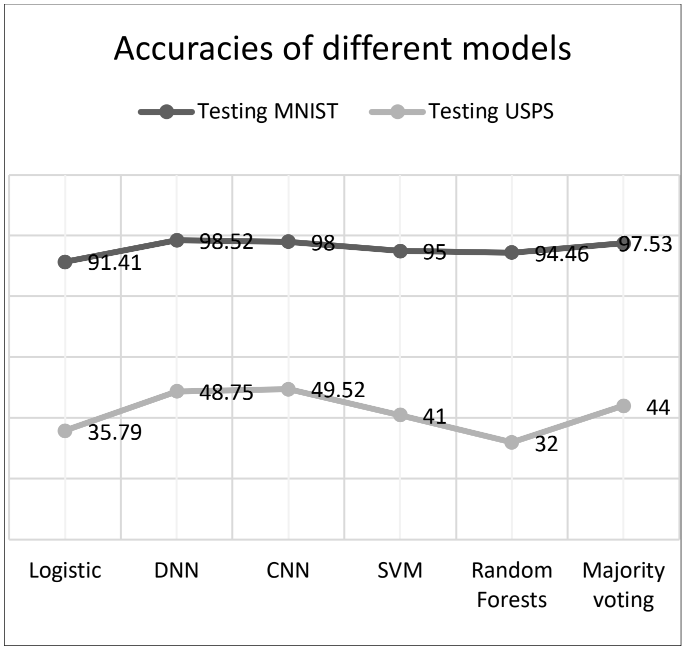


Figure 3

After an overall estimation and comparison of the performance of the above-mentioned classifiers, we can make the following observations:

- The neural network models give the most accuracy, followed by SVM, Random Forests and Logistic Regression.
- SVM took the most time for training the model, followed by random forests, neural networks and logistic regression.
- Random forests have less number of hyperparameters to tune. It gives the best performance on the default configuration of sklearn. The main limitation of Random Forest is that a large number of trees can make the algorithm too slow.
- Logistic regression, Random forests can be used for both classification and regression problems.
- Neural Network is a very convenient and scalable method of generating machine learning models.

The "**No Free Lunch**" theorem states that there is no one model that works best for every problem. The assumptions of a great model for one problem may not hold for another problem, so it is common in machine learning to try multiple models and find one that works best for a particular problem. The CNN model gave the best accuracy for USPS dataset but for the MNIST dataset, DNN gave the best accuracy. This proves the 'No Free Lunch' theorem.