

PROJECT REPORT – CSE 574

**Handwritten Digit Recognition using Logistic Regression and Neural networks**

**Srinivasan Rengarajan**  
**50097996**

## **OUTLINE:**

Hand written digit recognition is one of the classical problems in Machine learning literature. Simply, put it develops automated machine learning based methods for recognizing handwritten digits and words. This technology was developed about 20-30 years ago and there have been a lot of developments in this particular field of research. One of the important applications of handwritten digit recognition is in United States Postal Service (USPS) where each digit (Pincode) is automatically scanned and the appropriate area code is retrieved. This saves a lot of time and manual labor.

Handwritten digits recognition is an important problem , and not so easy. The main challenge is in extraction of appropriate features so that digits written in any manner are recognized. Some numerals may be improperly used or written in a cursive manner. There are various ways of writing the same number and it is important for the machine learning algorithm to recognize variety in handwriting in order to make a proper and accurate prediction. This involves a significant component of handwriting analysis. The features used here were developed at the Center for Excellence in Document Analysis and recognition (CEDAR) in the University at Buffalo.

## **FEATURES:**

The number of features extracted for this are of three types- Gradient (192), Structural (192) and Concavity (128). These features are extracted based on variability of handwriting across training data extracted from various documents at the United States Postal service (USPS). Each of these documents are scanned and the handwritten numerals are extracted from it. The training corpus was huge in order to cater to different styles, textures of writing. These features are formed into a matrix of size 19978 \* 512. The features vector is compiled by combination of all feature training values given for all digits separately.

A bias value of 1 is added to each column making the size  $19978 * 513$  as the input feature matrix.

### **MACHINE LEARNING algorithm overview:**

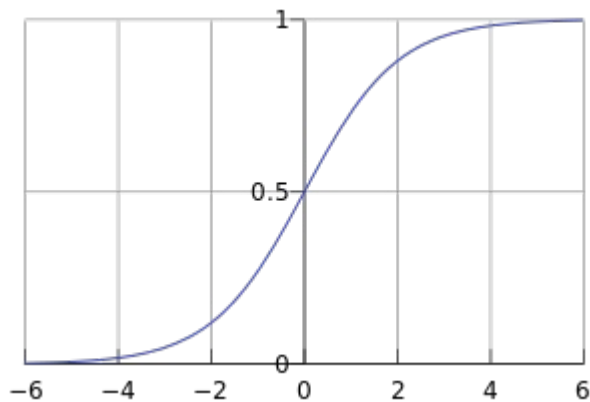
Hand writing digit recognition can be formulated by many machine learning algorithms for instance logistic regression , neural networks, expectation maximization algorithms, mixture models, Gaussian processes etc. Its a classic problem and has been solved using many different algorithms already. The new exciting development in the field is the Extraction of features using deep belief networks (DBN). The field of deep learning is famous because it enables automatic feature extraction. Usually, in the field of machine learning, the features that we extract are based on a specific knowledge of the data, ie, it answers the question of what particular characteristic the particular data has. Deep learning extracts these features on its own. It's powerful and gives extremely good results. This algorithm uses a Markov random field inside the neural network and is found to be an extremely powerful optimizer.

This project report implements two machine learning algorithms namely logistic regression and neural networks and the details of these are given below :

### **LOGISTIC REGRESSION:**

This algorithm uses a soft max function in combination with a gradient descent algorithm in order to iteratively minimize weights in order to obtain a minimum error value. This assumes a  $19978 * 513$  input matrix and also assigns a random weight vector of size  $(513,10)$  where 10 denotes the number of classes here. This is a multi class logistic regression problem where the digits 0-9 are taken as each class. Logistic regression measures the relationship between a categorical dependent variable and one or more independent variables, which are usually (but not necessarily) continuous, by using probability scores as the predicted values of the dependent variable.

Example of logistic sigmoid :



The above function is an example of a logistic sigmoid which maps the input variables to a value which is between -0.5 and +0.5. This is an S shaped curve and usually called the sigmoid or the logistic sigmoid function.

#### **EVALUATION ON GIVEN DATASET:**

This classification algorithm is evaluated on the given dataset which contains the input feature space between 0-9. This  $19978 \times 513$  vector is given as input along with a weight vector which is generated at random. This uses a gradient descent algorithm with a learning rate of 0.0001 to iteratively minimize error during training and compute the optimum value of  $w$ . This value of  $w$  is used in future for prediction of incoming handwritten numerals.

This uses a 1 of K coding scheme to compute the target vector. For example, the target vector is assigned a value of 1 and nine zeros if the digit to be recognized is a zero, 0 1 0.....0 if the digit to be found is 1 etc. This constitutes the formation of the target vector. The output is a  $19978 \times 10$  matrix with values for each of the ten classes ( $y_k$ ). The absolute error is computed using the Cross entropy function and the error gradient, the derivative of the sigmoid is used to minimize the weights using the gradient descent algorithm as mentioned above. After the training process, these weights are used to compute the output, test error. Reciprocal rank and other functions.

## RESULTS:

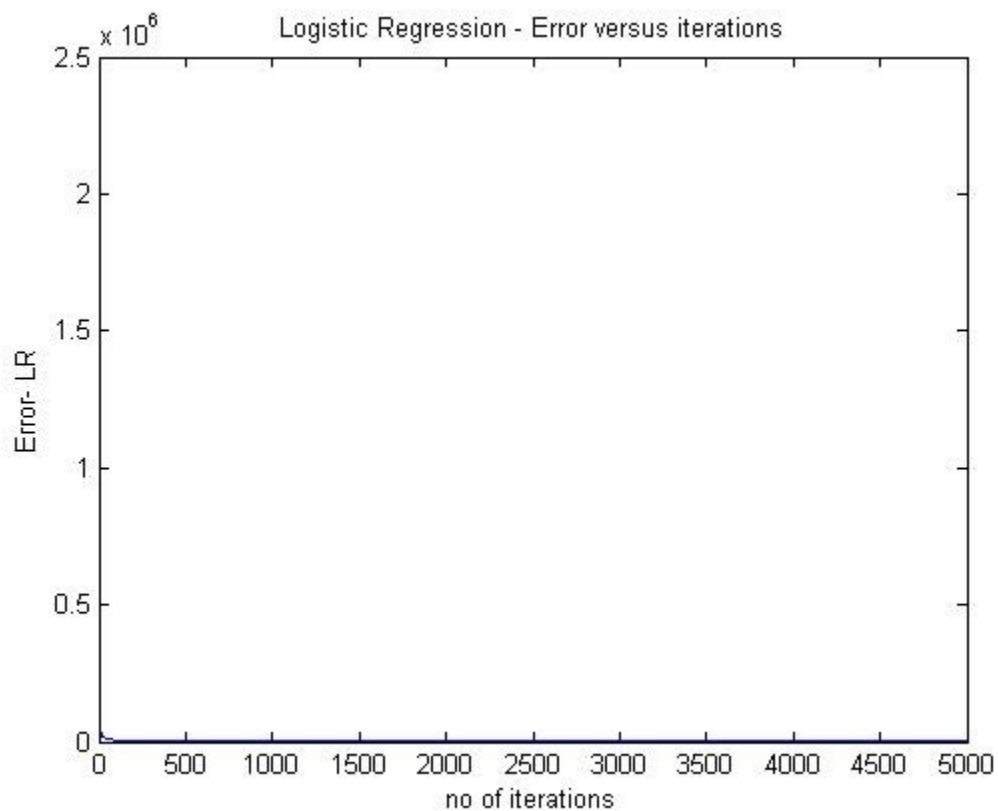
On evaluation in the given dataset, the following parameters have been determined.

**ACCURACY - 97.6%**

**ERROR RATE- TRAINING = 4%**

**ERROR RATE- TESTING = 2.4%**

**RECIPROCAL RANK = 1**

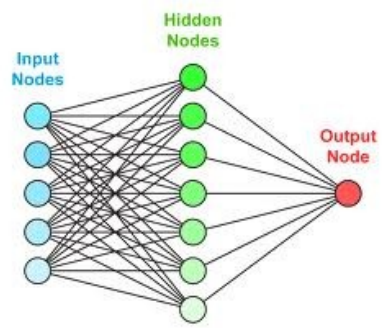


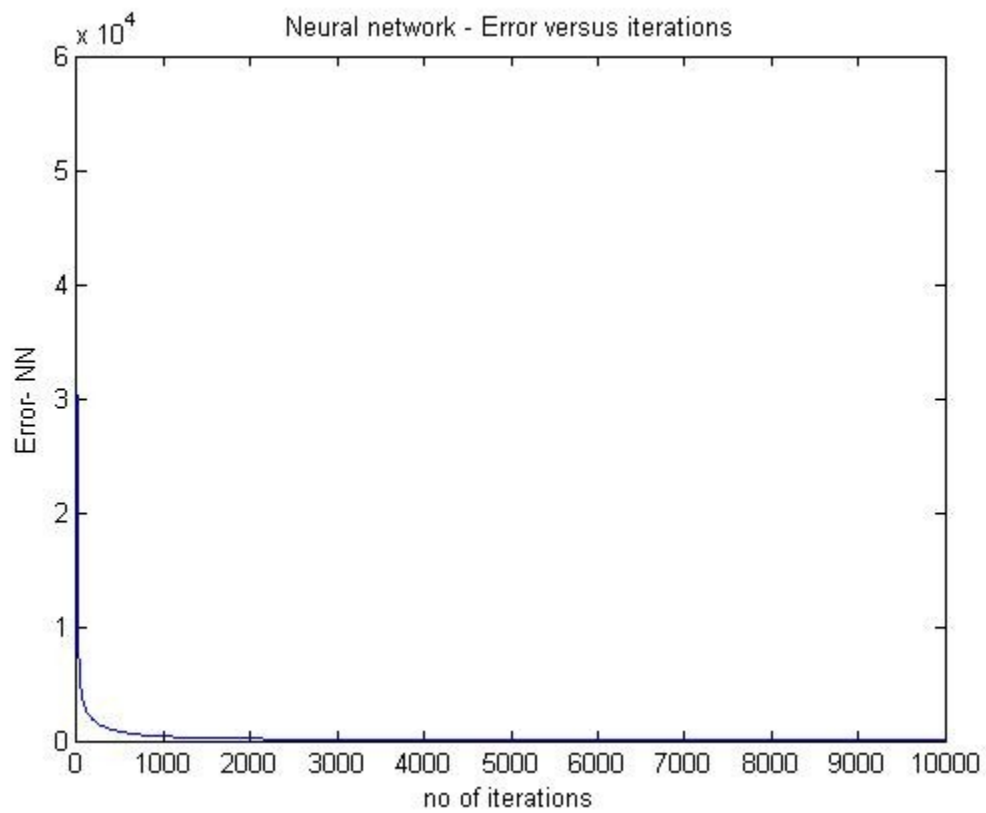
## 2. NEURAL NETWORKS:

The second classifier used here is the neural network. This is often compared with the impulse response

to a synapse by a neuron in a biological framework. This contains input, hidden and output layers as shown in the diagram. Weights are assigned for the transition from the input to hidden and hidden to output layer. These are denoted by  $w_1$  and  $w_2$ .

The activation function from the input to the hidden layer is given by the tanh function and the function from the hidden to output layer is given by the softmax function. The tanh function is applied on the product of the input matrix and the weights  $w_1$  assigned at random. As in the case of logistic regression, the input bias of 1 is added to the input matrix. Here, the error and other parameters computed are a function of the number of hidden neurons, or the number of neurons in the hidden layer denoted by the letter,  $M$ . In our case, we assign  $M$  arbitrarily to the value 20. The weights  $w_1$  and  $w_2$  are iteratively minimized using the gradient descent algorithm as in case of logistic regression and the output is calculated using the weight from the hidden to output layer  $w_2$ . The output is not the direct multiplication of the testing data with the weights, but comes with an activation function, and all the other processes associated with training the neural network. The network is trained with  $M=20$  and run for 10,000 iterations in order to minimize error value





## **RESULTS:**

- 1. Accuracy - 96.8%**
- 2. Error rate-train**
- 3. Error rate testing – ~3%**
- 4. Reciprocal rank= 1**

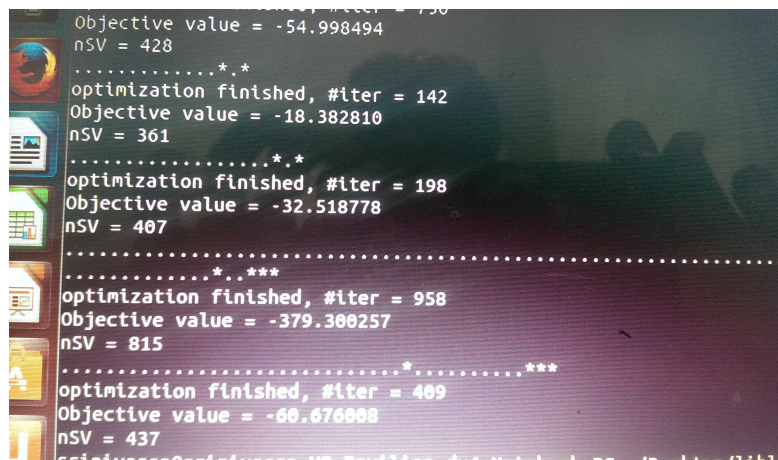
## **COMPARISON WITH SVM:**

The results of the 2 algorithms- logistic regression and neural network are evaluated with a standard



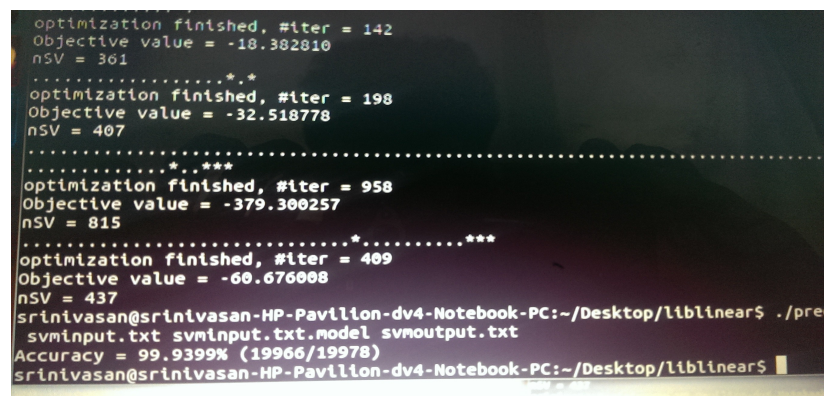
package – LIBSVM. The train parameters for the model are

1. ./train -M 0 svminput.txt - This corresponds to the input matrix (train) in LIBSVM format. This generates an svminput.txt.model file which defines the parameters of the model.
2. The figure shown below shows the snapshot of training. NSv = 437, with 489 iterations.



```
Objective value = -54.998494
nSV = 428
.....*.*
optimization finished, #iter = 142
Objective value = -18.382810
nSV = 361
.....*.*
optimization finished, #iter = 198
Objective value = -32.518778
nSV = 407
.....*.*
optimization finished, #iter = 958
Objective value = -379.300257
nSV = 815
.....*.*
optimization finished, #iter = 409
Objective value = -60.676008
nSV = 437
```

2. The testing involves the input as the test file in svm format, the model file and the file to which the output is written, svmoutput.txt.



```
optimization finished, #iter = 142
Objective value = -18.382810
nSV = 361
.....*.*
optimization finished, #iter = 198
Objective value = -32.518778
nSV = 407
.....*.*
optimization finished, #iter = 958
Objective value = -379.300257
nSV = 815
.....*.*
optimization finished, #iter = 409
Objective value = -60.676008
nSV = 437
srinivasan@srinivasan-HP-Pavilion-dv4-Notebook-PC:~/Desktop/Liblinear$ ./pre
svminput.txt svminput.txt.model svmoutput.txt
Accuracy = 99.9399% (19966/19978)
srinivasan@srinivasan-HP-Pavilion-dv4-Notebook-PC:~/Desktop/Liblinear$
```

The accuracy of the model trained with SVM - 99.936%

## COMPARATIVE ANALYSIS:

The results are compared as : SVM> Neural network> Logistic regression. SVM is a powerful

classifier and gives a high accuracy. The NN model will give a high accuracy when the number of hidden layers are increased. Various types of neural networks like Convolutional and Deep belief networks (DBN) are used to increase performance measures.

This report hence summarizes the hand written digit recognition problem with LR and NN classifiers.