CSE 574 Introduction to Machine Learning
Project 3: Classification
Report

Jayaram Kuchibhotla
50208766
jayaramk

1. Objective:

The objective of this project is to classify the images of handwritten digits according to the numeric value of the image. Images represent 10 numbers ranging from 0 to 9. These images are present in two data sets MNIST and USPS .In this project, classification is done using 3 approaches 1.Logistic Regression 2.Sinlge Layer Neural Network and 3.Convolutional Neural Network

2. Loading the input data sets:

For the tasks in the project, there are two datasets given, MNIST dataset and US Postal Service data.

<u>MNIST Data Load</u>: This data set consists of the images (represented by 784 i.e. 28*28 values for pixel intensities). Each image consists of numeral ranging from 0 to 9. The label in data set is the actual digit in the image. There are 60000 images and 60000 labels in total. The data set is loaded using the 'pickle' library to divide into 3 partitions training, validation and test datasets.
Size of training data set input matrix = 50000*784
Number of labels in training data set = 50000

Size of validation data set input matrix = 10000*784
Number of labels in validation data set = 10000

Size of test data set input matrix = 10000*784
Number of labels in test data set = 10000

<u>USPS Data:</u> This data set consists of 10 folders of the images (each folder consists of 2000 images, 20000 images in total). Each folder is uniquely named with a digit ranging from 1 to 9, every folder consists of images of one digit only i.e. the name of the folder. The label in data set is the actual digit in the image. The images are loaded from folder using the os.walk() ,os.path() functions along with functions of Python Imaging Library. Each image is reduced into 28*28 image from its actual size so that it is represented by 784 values. These values are loaded into a numpy array and a label is created using the name of the folder itself. This step is repeated to create

a. Input data matrix of Dimension = 19999*784 values
b. Number of labels = 19999

3. **One Hot Encoding**:
Each single digit i.e. the label is represented as a vector of 10 values. If the position of element in the vector is equal to the label value, then the value at that position is 1, otherwise it is 0, in any case there are 9 zeros and one value is one.

4. **Logistic Regression implementation**:
This is implemented using Stochastic Gradient Descent(SGD) running for iterations which is equal to the number of elements in the data set. Learning rate is chosen by trial and error starting with the value 0.001 and the initial weight matrix consists of random values .The dimension of the matrix is 784*10 (784 values represent the pixel intensities while the 10 values represent the number of classes .i.e 10 digits) . In one iteration one row of the input data is considered i.e. 1*784

In Each SGD Iteration following is done

- Calculation of activation vector : The product of the weight vector and the one row input data is calculated and a bias vector is added to the product . Bias vector consists of 10 values ,where each value is 1 .Dimension of the activation vector is 1*10
- The predicted value vector yk is calculated by using numpy.exp() function
  yk (x) = exp (ak) /Summation of(exp (aj) , j ranging from 0 to 9).Dimension of yk is 1*10
- yk vector from every iteration is stored in a matrix for later calculation of accuracy
- Cross Entropy value is calculated
- The difference between yk and tk (output of one hot encoding) is multiplied with the input data vector to get a value which is multiplied with the learning rate .The final product is 784*10 matrix which is Gradient
- Weight values are updated by subtracting the Gradient from the current weight values

Logistic Regression is implemented on the training dataset to get the weights .The weights of training are given as input to the validation dataset and testing dataset to calculate the accuracy

As part of tuning, by trial and error above step is repeated for different combinations of number of epochs and learning rates and the best combination is chosen

T

5. **Single Layer Neural Network(SNN) Implementation :**
Single Layer Neural Network consists of three layers one input layer , one hidden layer and one output layer . Input layer consists of a number of nodes equal to the size of the dataset i.e. rows . Output Layer consists of a number nodes equal to number of classes . In this project it is 10 .The number of nodes in the hidden layer and the learning rate should be decided on trial and error basis . Minimum number of nodes in hidden layer should be a value greater than the number of the features I.e. pixel intensity values which is equal to 784 . Minimum value of learning rate is chosen as 0.001.

There are 2 weight matrices and 2 bias vectors

Weight matrix Wji represents weight values between the input layer and hidden layer.
Dimension is 784*M   M = number of nodes in the hidden layer

Weight matrix Wkj represents weight values between the hidden layer and output layer
Dimension is M*K   M = number of nodes in the hidden layer, K= number of classes i.e. 10

bj vector , dimension is 1*M   M =1000
bk vector, dimension is 1*K   K = 10

SNN is implemented using SGD. The number of iterations is equal to the number of elements in the data set . In one iteration one row of the input data is considered with dimension 1*784.

In Each SGD Iteration following is done.

- wji is multiplied with input data and added with the bj vector . The resultant value has a dimension of 1*M .This value is passed into logistic sigmoid to get another vector zj which also has a dimension of 1*M
- Calculation of ak value (activation value): zj value is multiplied with wkj value and bias bk is added to get a 1*K vector
- The predicted value vector yk is calculated by using numpy.exp() function
  yk (x) = exp (ak) /Summation of(exp (aj) , j ranging from 0 to 9).Dimension of yk is 1*K
  Each yk row matrix is calculated for later computation of accuracy
- Back propagation :
  -The difference between yk and tk gives deltak with dimension 1*K
  - The derivative of h(ak) is calculated as h(ak)(1 – h(ak)) as logistic regression is considered. wkj and delta are multiplied to give a value say p2.The derivative and p2 are multiplied to give deltaj vector which has a dimension of 1*M
  - The gradient ‘G1’ is obtained by taking product of the input data and deltaj which is then multiplied with scalar ‘eta’(learning rate).The dimension of G1 is 784*M
  -The gradient ‘G2’ is obtained by taking product of deltak and zj matrix which is a then multiplied with scalar ‘eta’(learning rate). The dimension of G2 is M*K
  - The gradient ’G3’ is obtained by taking product of learning rate and deltaj .Dimension of G3 is 1*M
  - The gradient ’G4’ is obtained by taking product of learning rate and deltak. Dimension is 1*K
  - The difference between Wji and G1 gives newer Wij value
  - The difference between Wkj and G2 gives newer Wkj value
  - The difference between bj and G3 gives newer bj value
  - The difference between bk and G4 gives newer bk value

The predicted value yk is used to recalculate the weights by approaching in the reverse direction .Hence the name back propagation algorithm

SNN is implemented on the training dataset to get the weights .The weights of training are given as input to the validation dataset and testing dataset to calculate the accuracy

As part of tuning, by trial and error above step is repeated for different combinations of

Number of nodes in hidden layer, number of epochs and learning rates and the best combination is chosen

6. **Convolutional Layer Neural Network(CNN)**

CNN is implemented using Tensorflow library.

CNN is trained using the MNIST data.
Accuracy is calculated using MNIST test dataset
Additionally accuracy is also calculated using USPS dataset

7. **Calculation of Accuracy values**: The yk (prediction value row vectors) which are calculated in every iteration are collected into a single matrix ykcollectmat[] this matrix is converted into a matrix of 0s and 1s, the maximum value in row is assigned 1 and others are assigned to zero in that row . The position of the maximum is noted .This position and the digit in the data label are compared .If they are equal, success count is incremented

Accuracy = SuccessCount/SizeofDataSet* 100

8 **Discussion of Results:**

Logistic Regression Results:

Accuracy for Logistic Regression:

Logistic Regression							
S.No	Hyper Paramter Tuning			Accuracy			
	Epoch	Learning rate		Training Data Set	Validation data set	Testing Data Set	USPS Data Set
1	1	0.0001		87.024	91.62	91.17	10
2	1	0.0002		87.222	91.49	91.27	10
3	1	0.0055		87.024	91.54	91.46	10
4	1	0.0078		86.882	91.34	91.27	10
5	1	0.0099		87.352	91.75	91.69	10

Entropy for Logistic Regression:

Logistic Regression							
S.No	Hyper Paramter Tuning			Entropy			
	Epoch	Learning rate		Training Data Set	Validation data set	Testing Data Set	USPS Data Set
1	1	0.0001		0.44195	0.174098	0.00018	2.30258
2	1	0.0002		0.26283	0.15669	0.000415	2.30258
3	1	0.0055		0.64194	0.50966	0.0004	2.30258
4	1	0.0078		0.81015	0.62172	0.00116	2.30258
5	1	0.0099		0.32884	0.22382	0.001887	2.30258

Final Values for Logistic Regression:

Learning Rate eta = 0.0099
Number of Epochs = 1
Accuracy for Training Data Set = 87.352
Accuracy for Validation Data Set =91.75
Accuracy for Testing Data Set = 91.69
Accuracy for USPS Data Set = 10 %

For the MNIST test data set, it can be observed that 91% of predictions for classifications are correct while 9% are incorrect which shows that model is trained well

For the USPS data set, it can be observed that only 10% of predictions for classifications are correct while 90% are incorrect which shows that model trained on MNIST data is not effective for classifying the USPS data

Single layer neural network Results:

Accuracy for Neural Network

Single Layer Neural Network						
Hyper Paramter Tuning			Accuracy			
Epoch	Learning rate	M	Training Data Set	Validation data set	Testing Data Set	USPS Data Set
1	0.0001	785	85.4	91.24	91.47	30.27
1	0.0002	785	86.357	91.85	92.43	30.29
1	0.0055	900	87.462	92.23	91.5	30.21
1	0.0099	900	87.17	92.6	92.29	30.34
1	0.0099	1000	89.526	92.71	92.32	30.36

Entropy for Neural Network

Single Layer Neural Network						
Hyper Paramter Tuning			Entropy			
Epoch	Learning rate	M	Training Data Set	Validation data set	Testing Data Set	USPS Data Set
1	0.0001	785	0.24421	0.12927	0.006075	0.00168
1	0.0002	785	0.25444	0.13424	0.005082	0.00168
1	0.0055	900	0.28417	0.1561	0.00812	0.00168
1	0.0099	900	0.23213	0.12021	0.0077	0.00168
1	0.0099	1000	0.26512	0.117229	0.007772	0.00168

Final Values for Neural Network:

Learning Rate eta = 0.0099
Number of Epochs = 1
Accuracy for Training Data Set = 89.526
Accuracy for Validation Data Set =92.71
Accuracy for Testing Data Set = 92.32
Accuracy for USPS Data Set = 30.36

For the MNIST test data set, it can be observed that 92% of predictions for classifications are correct while 8% are incorrect which shows that model is trained well

For the USPS data set, it can be observed that only 30% of predictions for classifications are correct while 70% are incorrect which shows that model trained on MNIST data is not effective for classifying the USPS data

CNN Results:
Final values for CNN

Last observed training accuracy for MNIST data =100 %
Test accuracy for MNIST data = 99.18%
Test accuracy for USPS data = 10.00%

For the MNIST test data set, it can be observed that 99% of predictions for classifications are correct while 1% are incorrect which shows that model is trained well

For the USPS data set, it can be observed that only 10% of predictions for classifications are correct while 90% are incorrect which shows that model trained on MNIST data is not effective for classifying the USPS data

Conclusion:
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(MNIST) may not hold for another problem(USPS)

Classsification Model	MNIST data accuracy	USPS data accuracy
Logistic Regression	91.69%	10%
Single Layer Neural Network	92.32%	30.36%
Convolutional Neural Network	99.18%	10.00%

As the accuracy for USPS dataset is quite less compared to accuracy for the MNIST data, where all the three models are trained using MNIST data ,'No Free Lunch' Theorem is supported