Classification

Raghavendra Prakash Nayak

***Abstract*—this project is to implement and evaluate classification algorithms. The classification task is to recognize a 28 ⇥ 28 grayscale handwritten digit image and identify it as a digit among 0, 1, 2, ... , 9. This task will be performed using Logistic Regression, Neural Network and Convolution Neural Networks.**

I. Introduction

Hand written digit recognition is formulated by many machine learning algorithms like logistic regression, neural networks, Gaussian processes and so on. 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, i.e., the Deep learning extracts these features on its own. The machine learning algorithms named logistic regression and neural networks are implemented

II. The Dataset

For this project, I have used the Mixed National Institute of Standards and Technology (MNIST) dataset is a large database of handwritten digits that is commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning. It was created by "re-mixing" the samples from NIST's original datasets.

The database contains 60,000 training images and 10,000 testing images. Half of the training set and half of the test set were taken from NIST's training dataset, while the other half of the training set and the other half of the test set were taken from NIST's testing dataset

III. The Method

The images and the labels were extracted from the dataset. These were processed into a MATLAB matrix that contains the feature vectors and a MATLAB vector that contains tables. The MNIST dataset is originally partitioned into a training set and a testing set. The training is done on the training set of 60,000 images and the testing is conducted on the test set of 10,000 images.

A. Logistic Regression:

Logistic regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution. Thus, it treats the same set of problems as regression using similar techniques, with the latter using a cumulative normal distribution curve instead.

This algorithm uses a gradient descent algorithm in order to determine the globally optimal weights for which the corresponding Error value is minimal. This method processes a (60000,785) input matrix (including bias) and also considers a random weight vector of size (785,10) 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.

This classification algorithm is evaluated on the given dataset which contains the input feature space between 0-9. This 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 to iteratively minimize error during training and compute the optimum value of weight 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 to be zero and so on. This constitutes the formation of the target vector. The output is a matrix with values for each of the ten classes. The absolute error is computed using the Cross entropy function and the Gradient of Error function, the derivative of the sigmoid is used to minimize the weights using the gradient descent approach. These weights are used to compute the accuracy and error rate with respect to the testing data.

The general form of equation is given by:

I am using a 1-of-K coding scheme t =[t1…..tk].for our multi-class classification task. Our multiclass logistic regression model could be represented in the form:

$$p(\mathcal{C}_k|\mathbf{x}) = y_k(\mathbf{x}) = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

Where the activation ak is given by ak = wk + bk. The cross entropy error function for multiclass classification problem seeing a training sample x would be;

$$E(\mathbf{x}) = -\sum_{k=1}^{K} t_k \ln y_k$$

Where yk = yk (x). The gradient of the error function would be,

$$\nabla_{\mathbf{w}_j} E(\mathbf{x}) = (y_j - t_j)\,\mathbf{x}$$

We then use the stochastic gradient descent which uses the first order derivative to update;

$$\mathbf{w}_j^{t+1} = \mathbf{w}_j^t - \eta \nabla_{\mathbf{w}_j} E(\mathbf{x})$$

to find the optimum of the error function and find the solution for wj.

B. *Neural Network*

In machine learning and cognitive science, artificial neural networks (ANNs) are a family of models inspired by biological neural networks (the central nervous systems of animals, in particular the brain) and are used to estimate or approximate functions that can depend on a large number of inputs and are generally unknown.

This classifier contains input, hidden and output layers. Weights are assigned for the transition from the input to hidden and hidden to output layer. These are denoted by W1 and W2.

The activation function from the input to the hidden layer and from the hidden layer to output is given by the “Sigmoid function”. The Sigmoid function is applied on the product of the input matrix and the weights W1 assigned at random. Also, the input bias of 1 is added to the input matrix. The error and other parameters computed indicate the number of neurons in the hidden layer.

The weights W1 and W2 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 W2. The output comes with an activation function and all the other processes associated with training the neural network.

The general form of the equation is given as:
I am using a neural network with one hidden layer. Suppose the input layers are denoted by xi and the output is yk. The feed forward propagation is as follows:

$$z_j = h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + b_j^{(1)}\right)$$

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + b_k^{(2)}$$

$$y_k = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

Where zj is the activation of the hidden layer h and h(.) is the activation function for the hidden layer. Here I have used logistic sigmoid for the activation function.

Cross entropy function is used:

$$E(\mathbf{x}) = -\sum_{k=1}^{K} t_k \ln y_k$$

where yk = yk (x). The back propagation is done as follows:

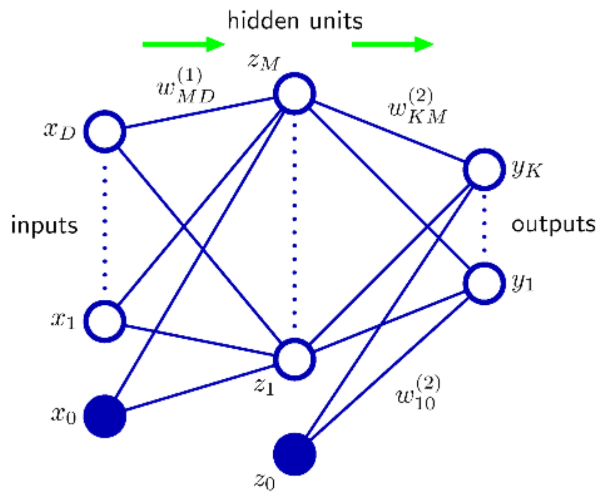


Figure: Network Diagram for the two-layer neural network

$$\delta_k = y_k - t_k$$

$$\delta_j = h'(z_j) \sum_{k=1}^{K} w_{kj} \delta_k$$

the gradient of the error function would be:

$$\frac{\partial E}{\partial w_{ji}^{(1)}} = \delta_j x_i, \qquad \frac{\partial E}{\partial w_{kj}^{(2)}} = \delta_k z_j$$

Having the gradients, the stochastic gradient is used to train the neural network;

$$\mathbf{w}^{t+1} = \mathbf{w}^{t} - \eta \nabla_{\mathbf{w}} E(\mathbf{x})$$

where w is all the parameters of the neural network.

C. Convolutional Neural Network:

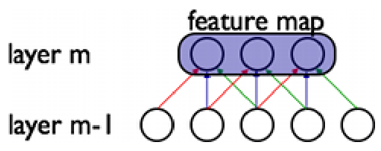
Convolutional neural network is a type of feed- forward artificial neural network where the individual neurons are tiled in such a way that they respond to overlapping regions in the visual field. Convolutional networks were inspired by biological processes and are variations of multilayer perceptrons designed to use minimal amounts of pre-processing. They have wide application in image and video recognition.

When used for image recognition, convolutional neural networks (CNNs) consist of multiple layers of small neuron collections which look at small portions of the input image, called receptive fields. The results of these collections are then tiled so that they overlap to obtain a better representation of the original image; this is repeated for every such layer. Because of this, they are able to tolerate translation of the input image. Convolutional networks may include local or global pooling layers, which combine the outputs of neuron clusters.

A feature map is obtained by repeated application of a function across sub-regions of the entire image, in other words, by *convolution*of the input image with a linear filter, adding a bias term and then applying a non-linear function. If we denote the k-th feature map at a given layer as h^k, whose filters are determined by the weights W^k and bias b_k, then the feature map h^k is obtained as follows (for $tanh$ non-linearities):

$$h^k_{ij} = \tanh((W^k * x)_{ij} + b_k).$$

In addition, in CNNs, each filter h_i is replicated across the entire visual field. These replicated units share the same parameterization (weight vector and bias) and form a *feature map*.



In the above figure, we show 3 hidden units belonging to the same feature map. Weights of the same color are shared—constrained to be identical. Gradient descent can still be used to learn such shared parameters, with only a small change to the original algorithm. The gradient of a shared weight is simply the sum of the gradients of the parameters being shared. Replicating units in this way allows for features to be detected *regardless of their position in the visual field.* Additionally, weight sharing increases learning efficiency by greatly reducing the number of free parameters being learnt. The constraints on the model enable CNNs to achieve better generalization on vision problems.

IV. RESULT

Choosing Hyper-parameters:

Learning Rate η:

As we take steps towards convergence we do so in iterations. η helps measuring those steps. If we take big steps we can overshoot our target and if our steps are too small it takes a long time for convergence.

Selecting iterations and Eeta:

At first we vary N and keep λ fixed. Then we can observe the following trend. If we increase the value of N then the error also increase and becomes constant. If we decrease N, then also the error keeps increasing. All of the parameters were hardcoded and tuned and then the most appropriate values were selected in the program.

The outcome of the learning models are reported in terms of accuracy and misclassification error.

Logistic Regression:

Eeta = 0.001

Number of iterations = 200

Following are the results:

Accuracy - 92.40%

Misclassification rate on training set of 60,000 images – 6.465%

Misclassification rate on test set of 10,000 images – 7.6%

Neural Network:

Eeta = 0.072

Number of iterations = 200

Following are the results:

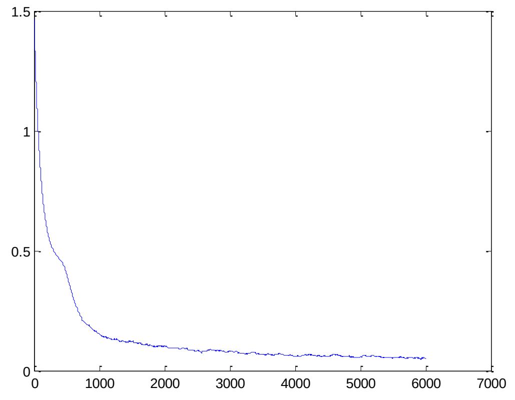
Accuracy - 92.93%

Misclassification rate on training set of 60,000 images – 4.85%

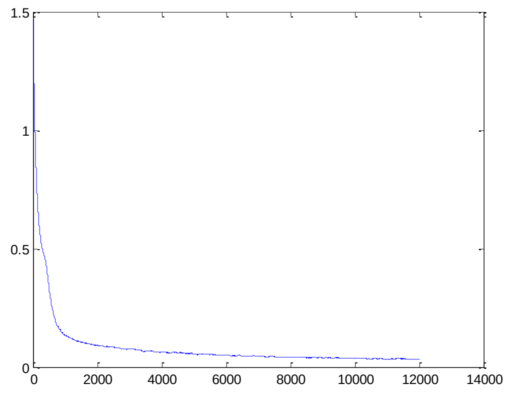
Misclassification rate on test set of 10,000 images – 7.07%

Convolution Neural Network:

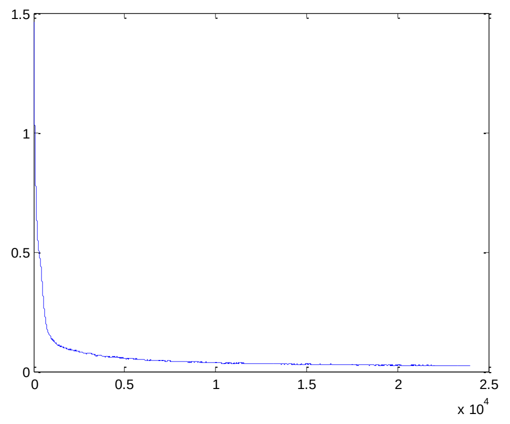
Changing the parameters such as epoch from 1 to 100 and other parameters such as batch size and alpha reduce the misclassification error.



With epoch as 1, the error = 11.1%



With epoch as 10, the error = 2.73%



With epoch as 20, the error = 1.78%

V. REFERENCES

1. MNIST Database- https://en.wikipedia.org/wiki/MNIST_database
2. Convolutional Neural Networks – https://en.wikipedia.org/wiki/Convolutional_neural_network http://deeplearning.net/tutorial/lenet.html
3. Project3- Description – CSE574 Introduction to Machine Learning.
4. Artificial Neural Networks- https://en.wikipedia.org/wiki/Artificial_neural_network
5. Logistic Regression- https://en.wikipedia.org/wiki/Logistic_regression