Volume 2, Issue 2, February 2012 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Classification System for Handwritten Devnagari Numeral with a Neural Network Approach

Padma R. Bagde*
Dept. of Electronics and Tele-Comm
Sipna's college of engg. and technology,
SGB Amravati university

Krushna D. Chinchkhede*
Dept. of Applied Electronics,
SGB, Amravati University

S.A.Ladhake
Sipna's College of Engineering and Tech
SGB Amravati university

Abstract— **This paper addresses an important and vital problem within the general area of character recognition, namely recognizing Marathi handwritten numerals. Artificial neural network approaches have been recognized as a powerful tool for handwritten numeral recognition. This paper demonstrates the use of single hidden layer MLP NN as a classifier for handwritten Marathi Numerals of Devnagari script. In present study, a MLP NN is designed with Tan sigmoid activation function for hidden and Log sigmoid function for output layer with neurons in hidden layer varied from 16 to 128 in steps of 16, constitutes 8 configurations of MLP NN trained three times each with memory efficient and fast Scaled Conjugate Gradient (SCG) algorithm. An image (64x64) of handwritten digits act as an input to the network, the training is controlled by early stopping criteria so that optimal network is derived. The intended network is analysed on various performances metric such as mse, best linear fit, correlation coefficient and misclassification rate. The scruples analysis of the result on different data partitions such as training, validation and testing provides best network to be further analysed. Further it is shown that the average classification accuracy for the best network is 98.35%, 89.71%, 91.28% and 96.77% on training, validation, testing and overall dataset respectively. On the basis of confusion matrix, results are elaborated with % misclassification for each output class distributed uniformly within dataset of 4465 samples. Network complexity in terms of weights and bias is 492938 connections from input to output.**

Keywords— **Handwritten Numerals recognition, MLP, Scaled Conjugate Gradient (SCG) algorithm, best regression fit, Confusion Matrix, log-sigmoid, tan-sigmoid.**

I. Introduction

Pattern recognition is formally defined as the process whereby received patterns are assigned to one of a prescribed number of classes (categories). The goal of pattern-recognition is to build machines, called classifiers that will automatically assign measurements to classes. A natural way to make class assignment is to define the decision surface. The decision surface is not trivially determined for many real-world problems. The central problem in pattern recognition is to define the shape and placement of the boundary so that the class-assignment errors are minimized. In classification problem, the task is to assign new inputs to one of a number of discrete classes or categories. Here, the functions that we seek to approximate are the probabilities of membership of the different classes expressed as functions of the input variables. In classification, we accept a priori that different input data may be generated by different mechanisms and the goal is to separate the data as well as possible into classes. Here, the input data is assumed to be multi-class, and the purpose is to separate them into classes as accurately as possible. The desired response is a set of arbitrary labels (a different integer is normally assigned to each of the classes), so every element of a class will share the same label. Here, the functions that we seek to approximate are the probabilities of membership of the different classes expressed as functions of the input variables. Class assignments are mutually exclusive, so a classifier needs a nonlinear mechanism such as an all-or-nothing switch. A neural network performs pattern recognition by first undergoing a training session, during which the network is repeatedly presented a set of input patterns along with the category to which each particular pattern belongs. Later, a new pattern is presented to the network that has not been seen before, but which belongs to the same population of patterns used to train the network. The network is able to identify the class of that particular pattern because of the information it has extracted from the training data. Pattern recognition performed by a neural network is statistical in nature, with the patterns being represented by points in a multidimensional decision space. The decision space is divided into regions, each one of which is associated with a class. The decision boundaries are determined by the training process. The construction of theses boundaries is made statistical by the inherent variability that exists within and between classes. It is very difficult to know which training algorithm will be the fastest for a given problem. It depends on many factors, including the complexity of the problem, the number of data points in the training set, the number of weights and biases in the network, the error goal, and whether the network is being used for pattern recognition (discriminate analysis) or function approximation (regression). Character recognition in general and handwritten character recognition in particular, has been an important research area for several decades. So much work done by researcher on it like, To recognizing touching in handwritten numeral strings they utilizing multiagents which performed parallels that introduced two agents have developed to work on the thinned image[1].Handwritten Digit Recognition by Neural Networks with Single-Layer Training is also implemented for handwritten numeral recognition[2].Similarly Modelling the Manifolds of Images of Handwritten Digits model in which The models allow a priori information about the structure of the manifolds to be combined with empirical data[3].Recognizing Handwritten Digits Using Hierarchical Products of Experts, in which main result is that it is possible to achieve discriminative performance comparable with state-of-the-art methods using a system [4]. Representation and Recognition of Handwritten Digits Using Deformable Templates in these two characters are matched by deforming the contour of one to fit the edge strengths of the other, and a dissimilarity measure is derived from the amount of deformation needed, the goodness of fit of the edges, and the interior overlap between the deformed shapes [5].Handwritten Digit Recognition by Adaptive-Subspace Self-Organizing Map (ASSOM) is a recent development in self-organizing map (SOM) computation [6]. Automatic Feature Generation for Handwritten Digit Recognition different evaluation measures orthogonally and information, are used to guide the search for features. The features are used in a back propagation trained neural network [7]. Since the early days of the development of neural networks, the applicability of these systems to automatic character recognition has been extensively studied by various research groups, A two-stage approach for segmentation and recognition of handwritten digit strings collected from mail pieces[8]. New learning method of multi-output Binary neural networks (BNN) is proposed for handwritten digit recognition based on simulated light sensitive model [9]. The performance of character recognition system is largely depending on proper feature extraction and correct classifier selection, for this purpose method proposed like Applications of Neural Network Chips and Automatic Learning in these two new methods for achieving handwritten digit recognition [10]. Slow Feature Analysis (SFA) is an unsupervised algorithm by extracting the slowly varying features from time series and has been used to pattern recognition successfully [11]. Method for calculating first-order derivative based feature saliency information in a trained neural network and its application to handwritten digit recognition [12]. Another different method proposed by researchers and carried out work On a Clustering Method for Handwritten Numeral recognition [13]. During the past half century, significant research efforts have been devoted to character recognition, a rapid feature extraction method is proposed and named as Celled Projection (CP) that compute the projection of each section formed through partitioning an image [14].Method of recognizing handwritten digits by fitting generative models that are built from deformable B-splines with Gaussian "ink generators" spaced along the length of the spline. The splines are adjusted using a novel elastic matching procedure based on the Expectation Maximization (EM) algorithm that maximizes the likelihood of the model generating the data [15] [16].A method of recognizing unconstrained handwritten numerals using a knowledge base is proposed for handwritten digit recognition, some describe system has a generalized recognition scheme based on a knowledge base [17].Researcher proposed classification method. Data classification method based on the tolerant rough set that extends the existing equivalent rough set is proposed and the similarity measure by the distance function of attributes between two objects, to determine the tolerant set among the objects[18][19]. Polygonal approximations generate quite powerful features for classification [20]. Four algorithms developed independently by researcher team This shows that it is possible to reduce the substitution rate to a desired level while maintaining a fairly high recognition rate in the classification of totally unconstrained handwritten ZIP code numerals[21]. Due to the variability of character spacing and the presence of breaking or touching characters, reliable segmentation prior to classification is impossible. In integrated segmentation and recognition of character strings, the underlying classifier is trained to be resistant to non-characters. Evaluate the performance of state-of-the-art pattern classifiers of this kind [22] [23].Pioneering effort for the development of handwritten numeral database of Indian scripts proposed by researcher [24]. In this paper, a design of classifier for classification of handwritten Marathi Numerals has been investigated using MLP neural network trained with Scaled Conjugate Gradient algorithm on the local database. Neural network designed, for varying number of neurons in hidden and output layer. The paper is organized as follows. First the Optimal MLP NN architecture for classification of handwritten Devnagari numeral is discussed. Later, the experimentation with tan sigmoid activation functions in hidden layer and log sigmoid transfer function in output layer is carried out with different trials. The best MLP NN based classifier is derived from various trials for the multi class classification task. Finally the result analysis is carried out on various performance measures such as MSE, R-value (correlation coefficients), and precents classification accuracy for MLP NN with various trials. A 64-bit version of MATLAB® with Neural Network Toolbox running on desktop PC (Intel® Core™2 Duo CPU E8400 @3.00 GHz with 4GB of RAM) is specifically used for obtaining results.

II. MLP NEURAL NETWORK ARCHITECTURE

A. Design of a MLP NN based Classifier

The multilayer feed-forward neural network can be used for both function fitting and pattern recognition problems. With the addition of a tapped delay line, it can also be used for prediction problems [25] [26].Multiple layers of neurons with nonlinear transfer functions allow the network to learn nonlinear relationships between input and output vectors. The linear output layer is most often used for function fitting (or nonlinear regression) problems. On the other hand, if we want to constrain the outputs of a network (such as between 0 and 1), then the output layer should use a sigmoid transfer function. This is the case when the network is used for pattern recognition problems (in which a decision is being made by the network). The configuration of the MLP NN is determined by the number of hidden layers, number of the neurons in each of the hidden layers, as well as the type of the activation functions used for the neurons. It has been established that an MLP NN that has only one hidden layer, with a sufficient number of neurons, acts as universal approximators of non-linear mappings [27]. Experimentally, it can be verified that the addition of extra hidden layer can enhance the discriminating ability of the NN model. However, it does so at the cost of the added computational complexity. The task of determining the weights from these examples is called training or learning. That is, the weights are estimated from the examples in such a way that the network, according to some metric, models the true relationship as accurately as possible.

B. *Training the MLP NN with Scaled Conjugate Gradient (SCG) Method*

The supervised mode of training process requires a set of examples of proper network behavior, network inputs p and target outputs y .The process of training a neural network involves tuning the values of the weights and biases of the network to optimize network performance. The performance function F for feed forward networks is mean square error (mse), the average squared error between the network outputs a and the target outputs y is defined as follows

$$F = mse = \frac{1}{N}\sum_{i=1}^{N}\left(e_i\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - a_i\right)^2 \qquad (1)$$

There are two different ways in which training can be implemented: incremental mode and batch mode. In incremental mode, the gradient is computed and the weights are updated after each input is applied to the network. In batch mode, all the inputs in the training set are applied to the network before the weights are updated. The batch training is significantly faster and produces smaller errors than incremental training. For training multilayer feed forward networks, any standard numerical optimization algorithm [28] [29] can be used to optimize the performance function, but there are a few key ones that have shown excellent performance for neural network training. These optimization methods use either the gradient of the network performance with respect to the network weights, or the Jacobian of the network errors with respect to the weights. The gradient and the Jacobian are calculated using a technique called the *back propagation* algorithm, which involves performing computations backward through the network. The back propagation computation is derived using the chain rule of calculus. Let us consider the simplest optimization algorithm gradient descent. It updates the network weights and biases in the direction in which the performance function decreases most rapidly, the negative of the gradient. Iteration of this algorithm can be written as

$$w_{k+1} = w_k - \alpha_k g_k \qquad (2)$$

Where w_k a vector of current weights and biases, g_k is the current gradient, and α_k is the learning rate. The fastest training algorithm is generally Levenberg-Marquardt algorithm [30].The quasi-Newton method is also quite fast. But both of these methods tend to be less efficient for large networks (with thousands of weights), since they require more memory and more computation time for these cases. Also, Levenberg-Marquardt performs better on function fitting (nonlinear regression) problems than on pattern recognition problems. When training the pattern recognition networks, Scaled Conjugate Gradient (SCG) and Resilient Back propagation (RB) are good choices. Their memory requirements are relatively small, and yet they are much faster than standard gradient descent algorithms. Scaled Conjugate Gradient (SCG) [31] is a supervised learning algorithm for feed forward neural networks, and is a member of the class of conjugate gradient methods. SCG is second order method means that, it makes use of the second derivatives of the goal function, while first-order techniques like standard back propagation only use the first derivatives. A second order technique generally finds a better way to a (local) minimum than a first order technique, but at a higher computational cost. SCG has been shown to be considerably faster than standard back propagation and then other CGMs [32].

C. *Performance measures*

- Mean Square Error (MSE): The process of training a neural network involves tuning the values of the weights and biases of the network to optimize network performance; it measures the network's performance according to the mean of squared errors.
- Regression Analysis: The performance of a trained network can be measured to some extent by the errors on the training, validation, and test sets, but it is often useful to investigate the network response in more detail. One option is to perform a regression analysis between the network response a and the corresponding targets y . In regression analysis, the equation of 'best linear fit' is estimated in least squares sense.
- If there were a perfect fit (outputs exactly equal to targets), the slope would be 1, and the y-intercept would be 0.

- Correlation Coefficient (R-value): The correlation coefficient (R-value) between the network outputs and targets is a measure of how well the variation in the output is explained by the targets. If this number is equal to 1, then there is perfect correlation between targets and outputs. R-value close to 1 indicates a good fit. 18
- Confusion Matrices: A confusion matrix is a simple methodology for displaying the classification results of a network. The confusion matrix is defined by labeling the desired classification on the rows and the predicted classifications on the columns.

III. MLP NN AS A CLASSIFIER

A. Dataset Used in the Experimental Setup

The handwritten numeral dataset is derived from images obtained by scanning the handwritten Numerals. Each individual subject was asked to write a set of 10 numerals on A4 sized paper with blue or black pen 10 to 12 times. Each page digitized into RGB bitmap image using Astra 3400 scanner with the resolution of 400 dpi. Each scanned image is further processed to extract individual Numerals and stored in local database. Thus the dataset formed, consist of 4500 images of Numerals of variable size distributed uniformly among 10 classes, where each numeral (class) is repeated 450 times. Figure1 shows the typical digit set with 10 classes. The raw data (4465 out of 4500 images were selected, one sample of digit “Eight” and 34 samples of digit “Nine” are discarded) is stored as a multidimensional cell array in secondary storage, act as a local database.

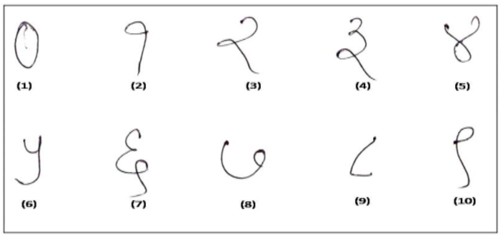


Fig.1 The typical handwritten digit set showing 10 classes.

B. Pre-processing

Morphology is a broad set of image processing operations that process images based on shapes. All 4465 images are converted into binary images; from binary image all connected components (objects) that have fewer than P pixels are removed by morphological operation. Extra columns and rows filled with zeros are removed by automated process. Finally all images are resized to 64x64 pixels each are now ready for further processing Thus for each 64x64 image, we have array of 4096 features (observations). The complete database of 4465 images is represented as matrix of 4096x4465 constitutes an input to the neural network. The target is represented as matrix of 10x4465 constitutes the desired output of neural network. Figure 2 and 3 shows the block diagram of proposed system used during training and testing phase.

C. Dataset Partitions

In order improve the generalization during training of multilayer networks; the general practice is to first divide the data into three subsets. The first subset is the training set, which is used for computing the gradient and updating the network weights and biases. The second subset is the validation set. The error on the validation set is monitored during the training process. The validation error normally decreases during the initial phase of training, as does the training set error. However, when the network begins to over-fit the data, the error on the validation set typically begins to rise.

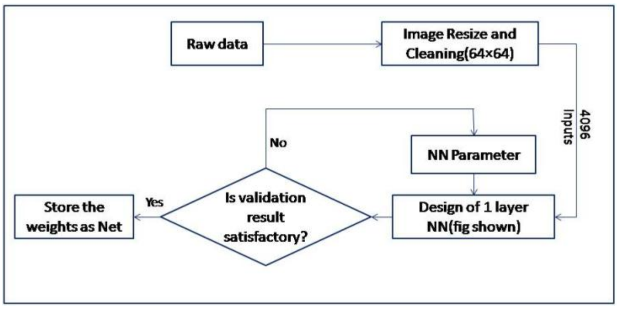


Fig. 2 Training Phase of handwritten digit classifier

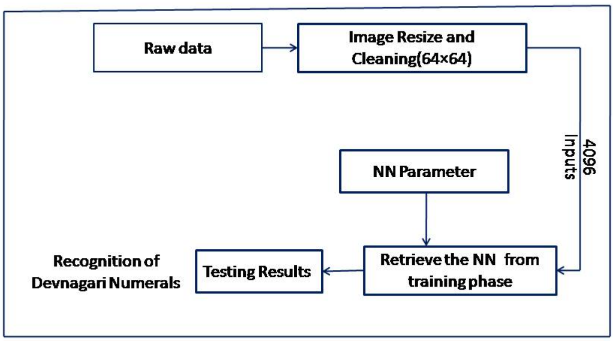


Fig.3 Testing Phase of handwritten digit classifier

The network weights and biases are saved at the minimum of the validation set error. The test set error is not used during training, but it is used to compare different models. It is also useful to plot the test set error during the training process. Multilayer networks can be trained to generalize well within the range of inputs for which they have been trained. However, they do not have the ability to accurately extrapolate beyond this range, so it is important that the training data span the full range of the input space. Hence dataset (4096x4465) is divided randomly into three subsets; Table I shows the data partition scheme for three different trials T1 to T3.

Table I. Data Partition Scheme (Input: Output)

	Traning Set	Validation Set	Testing Set
Trial : T1	4096x3571:10x3571	4096x447:10x447	4096x447:10x447
Trial : T2	4096x3571:10x3572	4096x447:10x447	4096x447:10x447
Trial : T3	4096x3571:10x3573	4096x447:10x447	4096x447:10x447

The training set is 80%, validation and test sets are 10% each randomly selected from 4465 exemplars, so as to increase the robustness of proposed MLP neural network. Table II shows various parameters for MLP NN classifier used in the simulations.

Table II. Various parameters for MLP NN classifier learning with SCG algorithm for Tan-Log sigmoid Transfer Function

Sr. No.	Parameter	Typical Range
1	Number of hidden layer	One
2	Number of neurons in hidden layer	16:16:128
3	Transfer function of hidden layer	Tan-sigmoid transfer function
4	Transfer function in output layer	Log-sigmoid transfer function
5	Maximum Epochs	1000
6	Minimum Gradient	1.00E-06
7	Maximum Validation Checks (max_fail)	101
8	Learning Rule	Scaled conjugate gradient backpropagation
9	Sigma*	5.00E-05
10	Lambda**	5.00E-07

* Determine change in weight for second derivative approximation

** Parameter for regulating the indefiniteness of the Hessian

IV. EXPERIMENTAL DETERMINATION OF NN

A. MLP NN with tan-log sigmoid transfer functions

MLP NN architecture as mentioned in fig. 4 shows use of tan sigmoid neurons in hidden layer and log sigmoid in output layer. As apparent from the network, it has 4096 inputs connected to predefined neurons in hidden layer and 10 neurons in the output layer. Neurons in hidden layer are varied from 16 to 128 in step of 16.

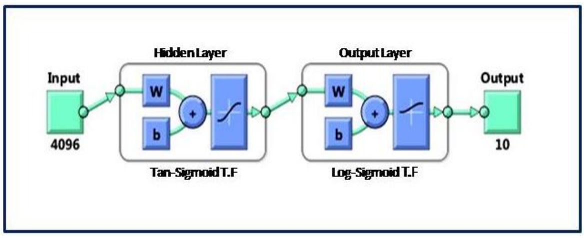


Fig. 4 MLP Neural Network (Tan-sigmoid transfer. function in hidden layer and Log-sigmoid transfer function in output layer).

The error on the validation set is monitored during the training process. When the validation error increases for a specified number of iterations, the training is stopped, and the weights and biases at the minimum of the validation error are saved as a best network. The above mentioned NN network is trained for three different trials with random initialization each time. Figure 4 and Figure 5 depicts the performance measure (mse) and classification accuracy respectively for trial T1, similar results are obtained for trials T2 and T3.

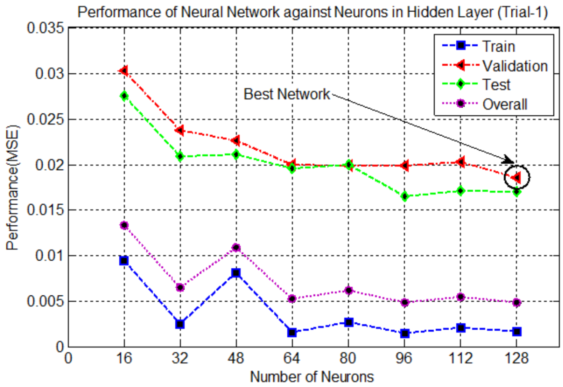


Fig. 5 Performances of NN vs. number of neurons during trial T1.

Table III depicts the performance measure (mse) for the NN on account of various trials T1 to T3 for 1000 epochs. From table III, it is evident that none of the trial last up to 1000 epochs because of early stopping criterion. The performance metric of best network with 128 neurons in hidden layer is shown in figure 6; the best network on account of minimum validation error of 0.018531 is discovered at epoch 158.

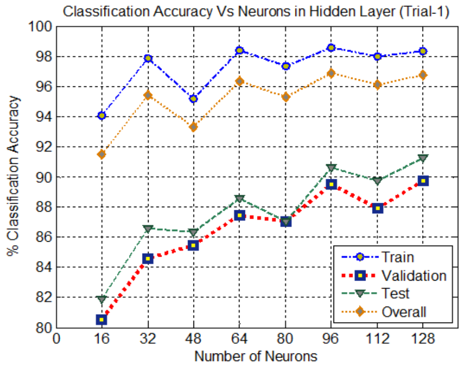


Fig.6 % classification accuracy vs. number of neurons during trial T1.

Training continued further until epoch 232, where it is stopped due to early stopping criterion on validation stop. Figure 8 also shows the graph of gradient versus epoch during the training of best network, which is also the criterion for early stopping.

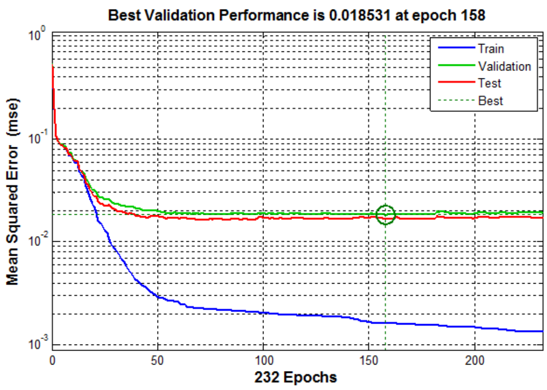


Fig. 7 Performance of network with 128 neurons

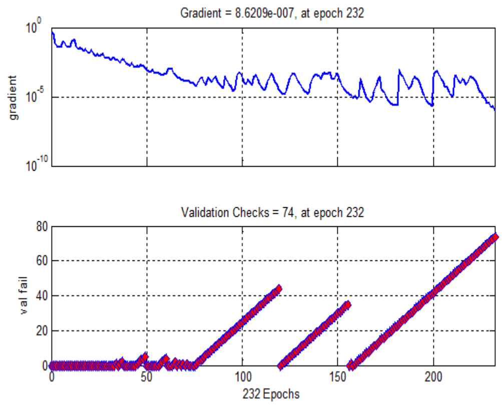
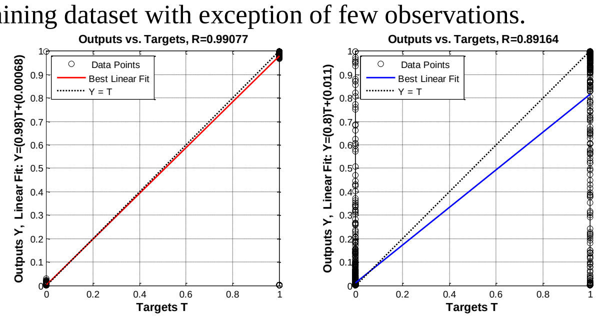


Fig. 8 Training state of network with 128 neurons

TABLE III Overall performance of the classifier for different trials (columns header 1, 2, 3, 4 and 5 represents *slope, y-intercept, r-value, mse and % overall accuracy* on training, validation, testing and overall datasets respectively

TN	NN	Epochs		Train					Validation					Test					Total				
		Best	Total	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
T1	16	64	165	0.834	0.0208	0.949	0.0095	94.09	0.667	0.0351	0.815	0.0303	80.54	0.676	0.0319	0.834	0.0275	81.88	0.801	0.0233	0.925	0.0133	91.51
T2	16	114	215	0.817	0.0160	0.934	0.0118	91.49	0.634	0.0302	0.781	0.0352	77.40	0.649	0.0290	0.790	0.0339	78.97	0.782	0.0187	0.906	0.0163	88.82
T3	16	157	258	0.890	0.0081	0.956	0.0079	93.00	0.705	0.0245	0.817	0.0302	80.09	0.724	0.0234	0.829	0.0284	82.10	0.855	0.0112	0.930	0.0121	90.62
T1	32	69	170	0.951	0.0051	0.986	0.0025	97.84	0.751	0.0194	0.859	0.0237	84.56	0.769	0.0177	0.876	0.0209	86.58	0.913	0.0078	0.963	0.0065	95.39
T2	32	39	140	0.875	0.0143	0.964	0.0067	95.63	0.710	0.0247	0.859	0.0237	84.12	0.744	0.0245	0.878	0.0207	89.04	0.846	0.0164	0.945	0.0098	93.82
T3	32	100	201	0.954	0.0038	0.983	0.0031	97.14	0.750	0.0176	0.859	0.0237	84.79	0.772	0.0189	0.869	0.0221	85.46	0.915	0.0067	0.960	0.0071	94.74
T1	48	52	153	0.855	0.0189	0.956	0.0081	95.16	0.734	0.0293	0.866	0.0226	85.46	0.742	0.0290	0.876	0.0211	86.35	0.832	0.0209	0.939	0.0108	93.30
T2	48	65	166	0.950	0.0041	0.983	0.0030	97.26	0.785	0.0167	0.882	0.0200	88.59	0.790	0.0167	0.887	0.0191	88.59	0.917	0.0066	0.964	0.0063	95.52
T3	48	63	164	0.955	0.0046	0.987	0.0023	97.93	0.768	0.0189	0.872	0.0216	87.02	0.755	0.0170	0.876	0.0211	88.14	0.916	0.0073	0.965	0.0062	95.86
T1	64	141	217	0.977	0.0012	0.991	0.0016	98.40	0.788	0.0125	0.883	0.0199	87.47	0.789	0.0122	0.885	0.0196	88.59	0.940	0.0035	0.970	0.0052	96.33
T2	64	90	191	0.974	0.0015	0.990	0.0018	98.29	0.792	0.0130	0.885	0.0195	88.14	0.814	0.0117	0.898	0.0174	89.49	0.940	0.0037	0.971	0.0051	96.39
T3	64	143	244	0.976	0.0016	0.992	0.0015	98.57	0.788	0.0157	0.878	0.0207	85.91	0.792	0.0129	0.885	0.0195	89.49	0.939	0.0042	0.971	0.0052	96.39
T1	80	79	180	0.962	0.0020	0.985	0.0027	97.34	0.780	0.0139	0.884	0.0198	87.02	0.785	0.0138	0.883	0.0200	87.02	0.926	0.0044	0.965	0.0062	95.27
T2	80	133	234	0.977	0.0008	0.990	0.0017	98.32	0.789	0.0124	0.876	0.0211	87.02	0.805	0.0103	0.897	0.0177	89.26	0.941	0.0029	0.970	0.0053	96.28
T3	80	138	239	0.977	0.0010	0.990	0.0018	98.26	0.806	0.0119	0.892	0.0184	87.70	0.802	0.0107	0.895	0.0179	89.71	0.943	0.0030	0.972	0.0050	96.35
T1	96	171	249	0.983	0.0008	0.992	0.0015	98.57	0.799	0.0114	0.883	0.0199	89.49	0.825	0.0106	0.904	0.0165	90.60	0.948	0.0028	0.973	0.0048	96.86
T2	96	65	166	0.951	0.0036	0.983	0.0031	97.09	0.778	0.0166	0.880	0.0203	87.47	0.788	0.0168	0.884	0.0196	87.70	0.918	0.0062	0.963	0.0065	95.18
T3	96	51	152	0.954	0.0033	0.982	0.0032	97.03	0.790	0.0143	0.889	0.0189	88.37	0.801	0.0136	0.898	0.0174	89.93	0.922	0.0054	0.965	0.0062	95.45
T1	112	123	224	0.972	0.0014	0.988	0.0021	97.96	0.791	0.0131	0.881	0.0203	87.92	0.813	0.0128	0.900	0.0171	89.71	0.938	0.0037	0.970	0.0054	96.13
T2	112	128	229	0.974	0.0013	0.990	0.0018	98.24	0.795	0.0119	0.891	0.0186	88.59	0.803	0.0106	0.899	0.0173	89.71	0.939	0.0033	0.972	0.0050	96.42
T3	112	221	322	0.976	0.0006	0.988	0.0021	97.96	0.810	0.0117	0.884	0.0198	86.58	0.819	0.0116	0.894	0.0181	89.71	0.944	0.0028	0.969	0.0055	95.99
T1	**128**	**158**	**232**	**0.980**	**0.0007**	**0.991**	**0.0017**	**98.35**	**0.805**	**0.0110**	**0.892**	**0.0185**	**89.71**	**0.812**	**0.0094**	**0.901**	**0.0170**	**91.28**	**0.946**	**0.0026**	**0.973**	**0.0049**	**96.77**
T2	128	205	306	0.980	0.0009	0.991	0.0016	98.49	0.812	0.0120	0.890	0.0188	88.59	0.818	0.0112	0.894	0.0181	89.49	0.947	0.0030	0.972	0.0049	96.60
T3	128	206	210	0.989	0.0002	0.995	0.0010	99.05	0.811	0.0100	0.89	0.0188	89.49	0.809	0.0093	0.894	0.0183	91.50	0.953	0.0021	0.975	0.0045	97.33

The regression analysis of '*best linear fit'* is estimated in least squares sense, which provides slope and y-intercept for '*best linear fit'* along with correlation coefficient (R-value) are also depicted in table III for each trial. Figure 9 shows regression line '*best linear fit'* for training, validation, testing and overall data set during trial T1 with 128 neurons in hidden layer. If we see the regression line corresponding to training set indicates that the neural network has been generalized well on training dataset with exception of few observations.



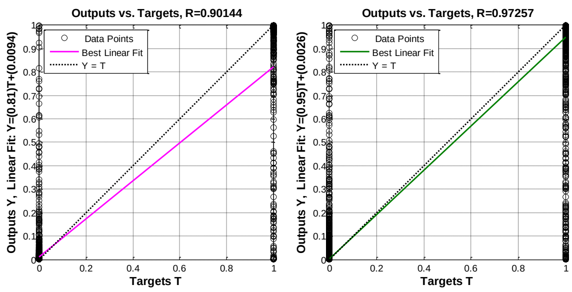


Fig. 9 Regression fit (best linear fit) for (a) training, (b) validation, (c) testing and (d) Total dataset

The slope of best fit is 0.980 along with correlation coefficient (R-value) of 0.991 confirms the optimal training. Similar analysis on validation and test dataset is carried out indicates slope value of 0.805 and 0.812 and correlation coefficient of 0.892 and 0.901, these values are some extent not close to unity indicates the network in-ability to generalize on unseen data. This provides an intuition to modify the configuration of network or reduce the dimensionality of data.

V. Result Analysis and Discussion

As discussed in previous section, the best network is fixed for 128 neurons in hidden layer. The best network is retrieved as a structure from local storage, it's complexity in terms of network weight is 492938 connections from input to output.

$$Network\ complexity = (No.Inputs \times No.Neurons + No.Neurons \times No.Classes) + b \times (No.Neurons + No.Classes)$$

Where b is bias which is treated as single node ($b=1$).
As a classifier the final output is always percentage error on misclassification on individual output classes (digits), the confusion matrix as mentioned in Figure 8 portray the % misclassification rate

	0	1	2	3	4	5	6	7	8	9
0	366	0	0	0	1	2	0	0	0	0
1	0	340	2	10	2	0	3	1	0	0
2	0	0	365	0	1	0	0	1	0	0
3	0	0	0	350	0	0	0	0	0	0
4	0	0	0	0	356	0	0	0	0	0
5	0	0	0	1	4	339	3	2	3	5
6	0	0	0	0	0	0	357	1	0	1
7	0	0	0	0	0	0	0	361	1	0
8	0	0	0	0	0	0	0	0	352	6
9	0	0	2	0	1	0	3	1	2	326

	0	1	2	3	4	5	6	7	8	9
0	35	0	0	0	0	0	0	0	0	0
1	0	35	4	0	0	1	3	0	0	0
2	0	0	38	2	0	0	1	1	0	0
3	0	1	1	44	0	0	1	0	0	0
4	0	0	2	0	47	0	1	2	0	0
5	0	1	0	0	3	46	4	0	0	0
6	0	0	0	1	2	1	38	2	0	0
7	0	1	0	0	0	0	1	38	0	0
8	0	0	0	0	0	0	0	0	41	5
9	0	0	0	2	0	0	1	0	2	39

	0	1	2	3	4	5	6	7	8	9
0	46	0	0	0	0	0	0	0	0	0
1	0	38	3	3	0	2	0	0	0	3
2	0	0	39	1	0	0	0	1	0	0
3	0	0	2	47	2	0	1	1	0	0
4	0	0	1	0	40	0	0	0	1	0
5	0	0	1	1	2	32	1	1	0	1
6	0	0	0	0	0	1	44	2	0	0
7	0	0	0	0	0	0	1	46	1	0
8	0	0	0	0	0	0	1	0	43	1
9	0	0	0	0	0	0	3	1	0	33

	0	1	2	3	4	5	6	7	8	9
0	447	0	0	0	1	2	0	0	0	0
1	0	413	9	13	2	3	6	1	0	3
2	0	0	442	3	1	0	1	3	0	0
3	0	1	3	441	2	0	2	1	0	0
4	0	0	3	0	443	0	1	2	1	0
5	0	1	1	2	9	417	8	3	3	6
6	0	0	0	1	2	2	439	5	0	1
7	0	1	0	0	0	0	2	445	2	0
8	0	0	0	0	0	0	1	0	436	12
9	0	0	2	2	1	0	7	2	4	398

Fig. 10 Confusion Matrix for (a) training, (b) validation, (c) testing and (d) Total dataset

Finally, the consolidate result of misclassification is calculated from confusion matrix, Table IV given below shows % misclassification among the individual classes.

TABLE IV. Overall performance of the classifier with 128 neurons

Class ID	Training Set				Validation Set				Testing Set				Total Data Set			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
0	369	366	3	0.81	35	35	0	0	46	46	0	0.00	450	447	3	0.67
1	358	340	18	5.03	43	35	8	18.6	49	38	11	22.45	450	413	37	8.22
2	367	365	2	0.54	42	38	4	9.524	41	39	2	4.88	450	442	8	1.78
3	350	350	0	0.00	47	44	3	6.383	53	47	6	11.32	450	441	9	2.00
4	356	356	0	0.00	52	47	5	9.615	42	40	2	4.76	450	443	7	1.56
5	357	339	18	5.04	54	46	8	14.81	39	32	7	17.95	450	417	33	7.33
6	359	357	2	0.56	44	38	6	13.64	47	44	3	6.38	450	439	11	2.44
7	362	361	1	0.28	40	38	2	5	48	46	2	4.17	450	445	5	1.11
8	358	352	6	1.68	46	41	5	10.87	45	43	2	4.44	449	436	13	2.90
9	335	326	9	2.69	44	39	5	11.36	37	33	4	10.81	416	398	18	4.33
Total	3571	3512	59	1.65	447	401	46	10.29	447	408	39	8.72	4465	4321	144	3.23

A : Total No. of instances B:Total No. of instances classified as a designated class
C: Total No. of instances misclassified D: % misclassification rate

Table IV indicates misclassification rate of 22.45% for digit "1" in test set.

VI. Conclusion

As a classifier, the best one hidden layer MLP NN with Tan-log activation function for hidden and output layer investigated gives an impression to perform reasonably. When it is evaluated on the training instances, it works as an almost good classifier with error rate of 1.65% on training. Here, the regression fit is found to be 0.991. On validation dataset, the error rate was significantly higher that is 10.29%, the regression fit is found to be 0.892. On test dataset, the error rate was that is 8.72%, the regression (best linear) fit is found to be 0.901. Thus it results in overall average classification accuracy of 96.77%. Overall single hidden layer MLP NN with Tan-log activation function demonstrate its acceptable discriminating ability in separating data possibly into 10 classes at the cost of increase computational complexity of 492938 connections from input to output. This provides an intuition to modify the configuration of network or reduce the dimensionality of data.

References

[1] Reda Alhajj and Ashraf Elnagar, "Multiagents to Separating Handwritten Connected Digits" *IEEE transactions on systems, man, and cybernetics—part a: systems and humans,* Vol. 35, No. 5, september 2005

[2] Stefan Knerr, L Con Personnaz, and G Crard Dreyfus, "Handwritten Digit Recognition by Neural Networks with Single-Layer Training" *IEEE transactions on neural networks*, Vol. **3,** No.6 November 1992.

[3] Geoffrey E. Hinton, Peter Dayan, and Michael Revow , "Modeling the Manifolds of Images of Handwritten Digits" *IEEE transactions on neural networks,* Vol. 8, No. 1. january 1997

[4] Guy Mayraz and Geoffrey E. Hinton, "Recognizing Handwritten Digits Using Hierarchical Products of Experts" *IEEE transactions on pattern analysis and machine intelligence*, Vol. 24, No. 2, February 2002

[5] Anil K. Jain, Douglas Zongker, "Representation and Recognition of Handwritten Digits Using Deformable Templates" *IEEE transactions on pattern analysis and machine intelligence,* Vol. 19, No. 12, December 1997

[6] Bailing Zhang, Minyue Fu, Hong Yan, and Marwan A. Jabri, "Handwritten Digit Recognition by Adaptive-Subspace Self-Organizing Map (ASSOM)" *IEEE transactions on neural networks*, Vol. 10, No. 4, july 1999

[7] Paul D. Gader, ,and Mohamed Ali Khabou, , "Automatic Feature Generation for Handwritten Digit Recognition'' *IEEE transactions on pattern analysis and machine intelligence,* vol 18, no 12, december1996

[8] Sungwon Park, Yunho Yang, and Gyeonghwan Kim, Seon-Hwa Jeong, "A two-stage approach for segmentation and recognition of handwritten digit strings collected from mail pieces" *Proceedings of the 17th International Conference on Pattern Recognition* (ICPR'2004)

[9] Jung H. Kim1, Byungwoon Ham1, Jui K. Chenl and Sung-Kwon Park2, "The Learning of Multi-output Binary Neural

Networks for Handwritten Digit Recognition" *Proceedings of 1993 International Joint Conference on Neural Networks*

[10] Y.Le Cun, L. D. Jackel,B. Boser,J. S. Denker,H. P. Graf,I. Guyon,D. Henderson,R. E. Howard,W. Hub bard, "Handwritten Digit Recognition: Applications of Neural Network Chips and Automatic Learning" November 1989 - *IEEE Communications Magazine,*

[11] Yaping Huang, Jiali Zhao, Mei Tian, Qi Zou and Siwei Luo "Slow feature discriminant analysis and its application on handwritten digit recognition" *Proceedings of International Joint Conference on Neural Networks,* Atlanta, Georgia, USA, June 14-19, 2009

[12] A.D. Parkins and A.K. Nandi, "Method for calculating first-order derivative based feature saliency information in a trained neural network and its application to handwritten digit recognition" *IEE Proc.-Vis. Image Signal Process*., Vol. 152, No. 2, April 2005

[13] Xu Ye, Zhang Wei, "On a Clustering Method for Handwritten Digit Recognition" Third International Conference on Intelligent Networks and Intelligent Systems 2010

[14] M. Zahid Hossain,M. Ashraful Amin, Hong Yan, "Rapid feature extraction for bangle handwritten digit recognition" Proceedings of the 2011 *International Conference on Machine Learning and Cybernetics,* Guilin, 10-13 July, 2011

[15] Michael Revow, Christopher K.I. Williams, and Geoffrey E. Hinton "Using generative model for handwritten digit recognition" *IEEE transactions on pattern analysis and machine intelligence*, Vol. 18, No. 6, june 1996

[16] Zhongkang Lu, Zheru Chi, Wan-Chi Siu "Extraction and Optimization of B-Spline PBD Templates for Recognition of Connected Handwritten Digit Strings" *IEEE transactions on pattern analysis and machine intelligence*, ,Vol. 24, No. 1, january2002

[17] Tuan .A.Mai and Ching Y.Suen "A Generalized Knowledge-Based System for the Recognition of Unconstrained Handwritten Numerals" *IEEE transactions on systems, man, and cybernetics*, Vol. 20, No. 4, July/August 1990.

[18] Daijin Kim and Sung-Yang Bang "A Handwritten Numeral Character Classification Using Tolerant Rough Set" *IEEE transactions on pattern analysis and machine intelligence*, Vol. 22, No. 9, September 2000.

[19] Y. S. Huang and C. Y. Suen, "A Method of Combining Multiple Experts for the Recognition of Unconstrained Handwritten Numerals" *IEEE transactions on pattern analysis and machine intelligence*, Vol. 17, No. 1, January 1995

[20] Theodosios Pavlidis and Farhat Ali, "Computer Recognition of Handwritten Numerals Polygonal Approximations" by -5, No. 6, November 1975

[21] Ching Y. Suen, Christine Nadal, Raymond Legault, Tuan A. Mai, and Louisa Lam, "Computer Recognition of Unconstrained Handwritten Numerals" *proceedings of the IEEE*. Vol. 80. no 7, July 1992

[22] Cheng-Lin Liu, Hiroshi Sako, Hiromichi Fujisawa, "Effects of Classifier Structures and Training Regimes on Integrated Segmentation and Recognition of Handwritten Numeral Strings" *IEEE transactions on pattern analysis and machine intelligence*, Vol. 26, No. 11, November 2004

[23] David J. Quarmby and John Rastall, "Experiments on Handwritten Numeral Classification" *IEEE transactions on systems, man, and cybernetics*, Vol. SMC- 1, No. 4, October 1971

[24] Ujjwal Bhattacharya and B.B. Chaudhuri, "Handwritten Numeral Databases of Indian Scripts and Multistage Recognition of Mixed Numerals" *IEEE transactions on pattern analysis and machine intelligence*, Vol. 31, No. 3, March 2009

[25] Duda, R. O., and Hart, P. E., "Pattern Classification and Scene Analysis," New York: John Wiley (1973)

[26] Haykin S., Neural Networks: A Comprehensive Foundation, McMillan, New York (1994).

[27] Hornik K. M. Stinchcombe M., and White H. "Multilayer Feedforward Networks Are Universal Approximators," Neural Networks, 2(5), pp. 359-66 (1989).

[28] Fletcher R., Practical Methods of Optimization, 2nd edition, New York: John Wiley (1987).

[29] Powell M. J. D., "Radial basis functions for multivariable interpolation: A review," IMA Conference on Algorithms for the Approximation of Functions and Data, pp. 143-167, RMCS, Shrivenham, England (1985).

[30] M. Hagan, H. Demuth, and M, Beale, Neural Network Design, Thomson Learning, India, (2003).

[31] Martin Fodslette Møller: A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks 6(4): 525-533 (1993)

[32] R. Gonzalez, and R. Woods, Digital Image Processing, Second edition, Pearson Education, India, (2006).