International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:3, No:4, 2009

Bi-lingual Handwritten Character and Numeral Recognition using Multi-Dimensional Recurrent Neural Networks (MDRNN)

Kandarpa Kumar Sarma, *Member, IEEE,*

Abstract—The key to the continued success of ANN depends, considerably, on the use of hybrid structures implemented on cooperative frame-works. Hybrid architectures provide the ability to the ANN to validate heterogeneous learning paradigms. This work describes the implementation of a set of Distributed and Hybrid ANN models for Character Recognition applied to Anglo-Assamese scripts. The objective is to describe the effectiveness of Hybrid ANN setups as innovative means of neural learning for an application like multi-lingual handwritten character and numeral recognition.

Keywords—Assamese, Feature, Recurrent.

I. Introduction

Applications of Artificial Neural Network (ANN) with huge data sets have always been known for the long periods of time taken for training. A hardware dependence is observed in these cases. It, at times, has been regarded as one of the drawbacks of ANNs for which the search for alternatives and replacements has driven studies and experiments in several application areas including pattern recognition. This work attempts to explore the capacity of the ANN configured in a co-operative heterogeneous architecture and the input variation - tackling capability for Anglo-Assamese handwritten character and numeral recognition. The basic blocks involved are Multi Layer Perceptron (MLP) and the Recurrent Neural Network (RNN) both associated with supervised learning involving the back-propagation (BP) algorithm. Cooperative heterogeneous architectures not only help to raise the computational power but also provide better discrimination capability to a complex pattern recognition application like the one attempted in this work. Cooperative architectures are also known as committee machines [1] [2]. These can be hybrid combination of different ANN types working in concert and trained in a cooperative environment configured to attain a common goal of achieving greater success rates at the cost of higher computation loads. Also, such architectures help to extend the range of parallelism associated with the ANN and considerably reduce the time required for training. Simultaneously, with less time required for training, such set-ups also allow to investigate the effect of large volumes of data on the learning ability of ANNs. Several works [3] to [11] have explored the hardware dependence of ANN for applications involving complex patterns and huge data volumes. Hybrid architectures have also received attention with similar considerations.

K. K. Sarma is with the Department of Electronics and Communication Technology, Gauhati University, Guwahati, Assam, PIN-781014, India. e-mail: (kandarpaks@gmail.com).

This work describes the implementation of a multi-dimensional RNN and a MLP- RNN hybrid ANN structure for developing models of character recognition applied to Anglo-Assamese scripts and numerals. The first stage of the work is related to the feature extraction process of characters and numerals of Assamese- an important language in the North Eastern part of India. The second stage of the work is related to the implementation of multiple ANNs in cooperative format for handling Assamese scripts and numerals. The third part of the system deals with Anglo- Assamese scripts and keeps track of performance of ANNs while tackling such mixed inputs implemented in cooperative systems for ascertaining performance levels while handling mixed, time and size varying inputs. A detailed description is also included regarding the experiments performed and the results derived. Finally, a brief account concludes the description and provides likely directions the work can take in the future.

II. Characteristics of the language-specific samples

The samples included in the work contains printed and handwritten characters and numerals of Assamese and English written by 10 different persons on five different days and digitized using a flatbed scanner at 150 dpi.

A. Assamese Characters:

Assamese is an important language in the North Eastern part of India. Many Indian languages including Assamese and Bengali have the same origin-Brahmi, hence there are certain similarities in the general shape and appearance; but due to their dissimilarities with respect to two characters and some of the compound characters, there is a necessity to develop a separate character recognition system for Assamese scripts where the attempts can be independent. Moreover, very few examples are known which are aimed at the development of an Optical Character Recognition (OCR) system exclusively for Assamese scripts. Some of the primary features of Assamese scripts maybe depicted as below:

- Assamese script is formed by 11 vowels, 41 consonants, over 10 modifiers and over 300 compound characters. The use of upper and lower case letters like in English is not there in Assamese.
- There is a use of head line (called matra in Assamese) in certain characters including consonants and vowels. It helps in segmentation of the characters easily but as many

of the segmented characters with their head lines missing appear similar, it makes classification difficult.

- A typical group of handwritten Assamese words maybe classified into three zones as shown in Figure 1:
  1) **Upper Zone:** Area above the head line. It is characterized by the presence of extensions of the modifiers.
  2) **Middle Zone:** Area where the main body of the character is present.
  3) **Lower Zone:** Area where some of the modifiers exist.

  Handwritten scripts can contain huge amount of variations due to differences in writing styles, orientation etc (Figure 2). Writer variation results in modification and alteration of shape, size, inclination and distribution of characters. Certain writing styles make characters touch each other. These are some of the variations observed.

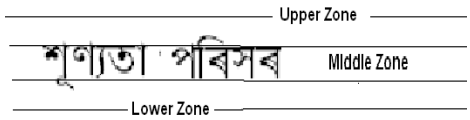

Fig. 1. Assamese words with three different zones

Fig. 2. Assamese and English samples

B. Assamese Numerals:

Assamese numerals are similar to that used by Bengali language. Isolated numerals in Assamese are characterized by the presence of zero-like, multiple zero-like, compound zero-like and straight line containing shapes, curvatures, compound shapes with curvature and lines etc (Figure 2). Zeros of both Assamese and English languages are similar. Similarity also exists between Assamese numeral 4 and English 8 (Figure 2). Handwritten numerals can contain huge amount of variations due to differences in writing styles, orientation etc. Writer variation results in modification and alteration of shape, size, inclination and distribution of numerals. Certain writing styles make numerals touch each other. Similarity can also be generated due to writer induced variations in case of the numerals 2 of both languages and Assamese 7 and English 9 (Figure 2). An ANN is well suited for such applications because of its ability to learn adaptively.

III. Feature Extraction

The feature extraction is carried out separately for the characters (Figure 2) and numerals. This is due to the size variations. The character set is over hundred while the number of numerals is twenty.

A. Feature extraction of characters:

The feature vector represents a unique set of data providing relevant information regarding shape, size, morphology etc of characters. The length of the feature vectors is more than 178 due to the assumption that each character should have at least two features [[12] [13] [14] [15]]. The feature extraction process is aimed at making the process robust enough to deal with two sizes, 32x32 and 64x64 having two fonts including italic characters. The features considered include geometrical, statistical, morphological, tomographic and hybrid ones for the entire training set so as to train the MLPs individually for all the configured forms. The hybrid feature set is the proposed feature vector which is a selective mixture of geometrical, statistical, morphological, tomographic and hybrid types so that it provides better invariance to shape, size and inclination.

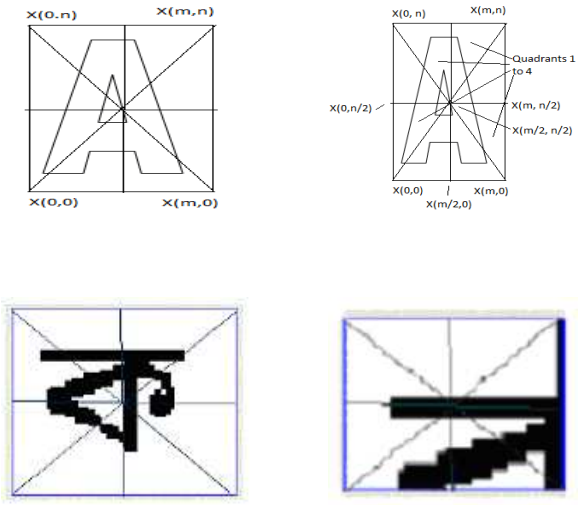

Fig. 3. Geometrical feature extraction

1) **Geometrical Features:** These features are associated with the shape, profile and physical structure and the geometrical attributes of an image. Among the geometrical

features, the following are considered- Horizontal and Vertical Projection, Euclidian Distance Measurements from the center to different corners of the character at varied inclinations of measurements (Figure 3), Distance vectors etc. Horizontal and vertical projection of pixels yield the following features. For a input character X(m, n), horizontal projection features are given as

$$H[n] = \sum_{m} X(m, n) \tag{1}$$

Vertical projection features are similarly given as

$$V[n] = \sum_{n} X(m, n) \tag{2}$$

For the entire character, Euclidean distance features can be extracted as:

$$F[k] = Euclidean\|X(\frac{m}{2}, \frac{n}{2}), X(i, j)\| \tag{3}$$

for i= $0 : \frac{m}{2} : m$ and j= $0 : \frac{n}{2} : n$;
Similarly, if X(m, n) is subdivided into four quadrants as $\sum_{i=1}^{4} X_i(p_i, q_i)$, for each of these quadrants

$$F_{ij} = Euclidean\|\sum_{i=1}^{4} X_i(p_i, q_i), X(p_j, q_j)\| \tag{4}$$

such that $X(p_j, q_j)$ lay in the edges of each quadrant at 45^0 angles with respect to $\sum_{i=1}^{4} X_i(p_i, q_i)$.

2) **Statistical Features:** These maybe used for qualitative description of object boundaries by using statistical moments like mean, variance and other higher moments. Mean provides average gray-level value, variance is linked to contrast, the third moment is a measure of the skewness and the fourth order moment provides information on relative flatness of an image. Similarly, higher order moments provide additional texture measures and entropy associated with an image. For statistical features the following are considered: Singular Value Decomposition (SVD), Principal Component Analysis (PCA), Total mass and up to 7^{th} order moments.
3) **Morphological Features:** The morphological features are useful for extracting image components useful in representation and description of region shape such as boundaries, skeletons and the convex hull. The morphological features considered in the work are extracted using edge detection, morphological gradient, skeletonisation, alternating sequential filtering (ASF), morphological distance transform, thinning and thickening, morphological regional statistics, dilation and erosion and morphological opening and closing.
4) **Tomographic Projection Features:** Radon transform of each of the characters are used as a feature set. The radon transform produces projections of the image intensity along a radial line oriented at a specific angle with the projection computed at certain number of points. Mean values of radon transforms taken over a 180 degree rotation calculated at 4 different points generated 320 values.
5) **Hybrid Features:**In addition to the above, a novel set of hybrid features has been used in the training. A multi-feature set comprising a judicious mixture of various features have been formulated [[12] [13] [14]]. The feature vector considered includes a mixture of selected features of geometrical, statistical, morphological and tomographic projections totaling up to 284 so as to make it more robust.

The hybrid feature set is able to provide-

- Unique samples linked with shape, profile and physical structure of an image. This is possible because of the use of certain geometrical features.
- Quantitative description of shape of image boundaries using statistical moments like mean, variance etc which are linked to average gray level distribution, contrast, skewness, relative flatness, texture measures, entropy etc.
- Image information associated with boundaries, shape, size, connectivity etc through the use of certain morphological features.
- Projection of image intensities along a radial line oriented over a 360o direction taken at certain number of points.

In this manner the hybrid feature vector represents a near complete description of the profile of a character. The feature set incorporates all the components through which it is better placed to deal with multi font variations. Also, Assamese like several other Indian languages uses scripts that demonstrate subtle variations in the morphological profile between individual characters. These variations can be noted proficiently by the morphological features than the other types. Since the statistical features provide quantitative description of the shape of the character image boundaries, this contribution aids the hybrid feature set in tackling noise mixed inputs. This way statistical feature received greater importance and made the second highest contribution in forming the hybrid feature set. Also statistical components have considerable amount of principal components which have reduced correlatedness among the components of the feature set and have minimized the effects of dimensionality. The next significant contribution has been made by the geometrical features which represents information related to the physical profile of a character. Assamese scripts show lots of geometrical variations due to the presence of corners, cusps, horizontal and vertical lines. These geometrical profiles at times show certain similarities but have minor variations as well which differentiate a character from the other. The geometrical feature set is well suited to extract these variations in the physical profile of the characters. Finally, tomographic projections are required to obtain multi angle views of image intensity variations which provide a robust set of features at selected points.

B. Assamese Numerals:

The modified features are derived from the original hybrid feature set of length 284 by extracting only those values that have a difference between 20 to 24 percent between adjacent samples. By following this criterion, feature vector length of 50 is obtained [15].

IV. MDFRRTRNN Architecture for Handwritten Character and Numeral Recognition

The primary motivation behind the formulation of the hybrid RNN-MLP architecture is two folded. First, it is related to the ability of the RNN to deal with writer induced variations observed in hand-written character recognition and the sample length variation seen in numeral inputs in mixed sample sets used in this work as part of the combined alphabet and numeral recognition system in two languages. Second, is the MLPs ability to perform the role of a class mapper [2]. Character set inputs are normalized to 32x32 and 64x64 size while numeral sets are 16 x 16 and 32 x 32 forms. In a mixed sample set, this variation makes an MLP based recognizer a bit less optimal as has been observed in this work. Hence,

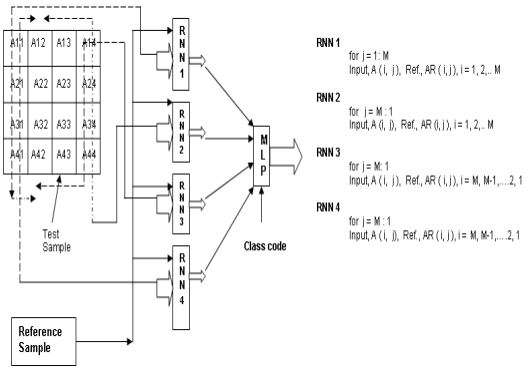

Fig. 4. Multi-Dimensional Fully Recurrent Real Time RNN (MDFRRTRNN) architecture suitable for hand written character recognition

a structure coined multi-dimensional fully recurrent real time RNN (MDFRRTRNN) is formulated as shown in Figure 4. This structure uses four numbers of RNNs and one MLP to generate the output. Each of the RNNs is reconfigurable as per the input size. There is one hidden layer which is 1.5 times longer than the input. The output layer has the same length as that of the reference pattern is but changes as per the requirement. Log-sigmoid activations are used in the input and output layers while tan-sigmoid activation functions are placed in the hidden layer. Two training methods namely Levenberg-Marquardt (LM) BP and adaptive learning rate BP with a momentum term are used. The second method turns out to be the most suitable one in terms of time and memory required and performance generated.

The first RNN block labelled RNN1 receives a real-time input of a M x M sample by running a scan from left top to bottom and towards right. It is fed by a similar sample from the reference pattern to carry out the training. The second RNN block (RNN 2) takes a real time scanned sample from top right to bottom and then left. The complete process is depicted as in Figure 4 and the related training and reference inputs indicated. The output of each of the RNN blocks is fed to an MLP which compares them to the class code given to it as reference. This learning by both the RNNs and the MLP continues till the desired global MSE goal is attained. Initially each of the RNNs and the MLP is provided individual MSE convergence and classification parameters which are related to the global limits upon which the success-rate of the structure depends. The training of the RNN blocks continues simultaneously with individual parameters after which the training of the MLP proceeds. For characters and numerals if the sizes are different the input layers and the corresponding structure of the RNN blocks changes while the MLP retains the initially defined form and acts as a class mapper using the outputs of the real-time RNN blocks.

Let $u_k(n)$ be a set of inputs and $x_k(n)$ be the state vectors for k number of RNNs at time n. Let

$$x_k(n+1) = \sum_k [\sum_{i=1}^{M} x_{ki}(n) w_{ki}(n) + \sum_{j=1}^{N} x_{kj}(n) w_{kj}(n)] \quad (5)$$

such that

$$y_k(n+1) = \sum_k \sum_i \varphi(x_k(n+1)) \quad (6)$$

where k=1, 2, 3, 4.

If $d_k(n)$ is the set of desired outputs, the error vector set can be obtained as

$$e_k(n) = d_k(n) - y_k(n) \quad (7)$$

such that

$$\xi_k(n) = \frac{1}{2} e_k^T(n) e_k(n) \quad (8)$$

When $\xi_k(n)$ is minimized using Real Time Recurrent Learning (RTRL) algorithm such that

$$\nabla_W \xi_k(n) = \Sigma_k \frac{\delta \xi_k(n)}{\delta w_k} \quad (9)$$

$y_k(n)$ form inputs to the MLP in Figure 4. The RNNs are trained by following the considerations described in Section IV-A. The learning of the mapper MLP is carried out by following the steps described in Section IV-B.

A. RNN Training:

Two basic algorithm for RNNs which depends on these modes are summarized below-

1) **Back - Propagation Through Time:** The back-propagation-through-time (BPTT) algorithm for training of a recurrent network is an extension of the standard back-propagation algorithm [2]. It may be derived by unfolding the temporal of the network into a layered feedforward network, the topology of which grows by one layer at every time step. Let N denote a recurrent network required to learn a temporal task, starting from time n_0 all the way up to time n. Let N^* denote the feedforward network that results from unfolding the temporal operation of the recurrent network N. The unfolded network N^* is related to the original network N as follows:
   - For each time step in the interval (n_0, n), the network N^* has a layer of containing K neurons,

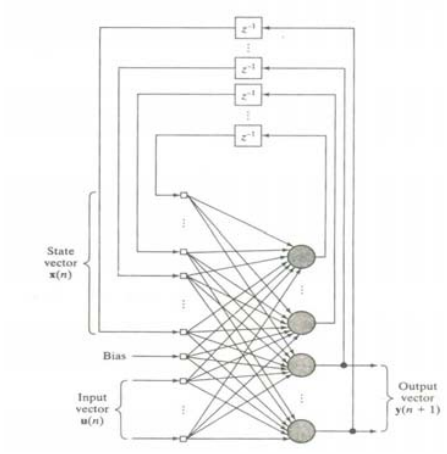

Fig. 5. Recurrent Network for formulation of the RTRL algorithm

where K is the number of neurons contained in the network N.

- In every layer of the network N^* there is a copy of each neuron in the network N.
- For each time step $l \in [n_0, n]$,the synaptic connection from neuron*i* in layer *l* to neuron j in layer l+1 of the network N^* is a copy of the synaptic connection from neuron *i* to neuron *j* in the network N.

2) **Real-Time Recurrent Learning:** Another learning method referred to as *Real-Time Recurrent Learning*(RTRL), derives its name from the fact that adjustments are made to the synaptic weights of a fully connected recurrent network in areal time, that is, while the network continues to perform its signal processing function [2]. Figure 5 shows the the layout of such a RNN. It consists of q neurons with m external inputs. The network has two distinct layers: a concatenated input-feedback layer and a processing layer of computation nodes. Correspondingly, the synaptic connections of the network are made up of feed-forward and feedback connection. The state-space description of the network is defined by the following expression:

$$x(n+1) = \varphi(W_a x(n) + W_b u(n)) \tag{10}$$

$$y(n) = Cx(n) \tag{11}$$

whereW_a is a q-by-q matrix, W_b is a q-by-$(m+1)$ matrix, C is a p-by-q matrix. The eq 12 is reproduced here in the following expanded form

$$x(n+1) = \begin{pmatrix} \varphi(w_1^T \xi(n)) \\ \vdots \\ \vdots \\ \varphi(w_j^T \xi(n)) \\ \vdots \\ \vdots \\ \varphi(w_q^T \xi(n)) \end{pmatrix} \tag{12}$$

where it is assumed that all the neurons have a common activation function $\varphi(.)$. The $(q+m+1)-by-1$ vector $w_j$is the synaptic weight vector of neuron j in the RNN.

$$w_j = \begin{pmatrix} w_{a,j} \\ w_{b,j} \end{pmatrix} j = 1, 2, ..., q \tag{13}$$

where $w_{a,j}$ and $w_{b,j}$ are the j^{th} columns of the transposed weight matrices $(w_a^T$ and $(w_b^T$, respectively. The (q+m+1)-by-1 vector $\xi(n)$ is defined by

$$\xi(n) = \begin{pmatrix} x(n) \\ u(n) \end{pmatrix} \tag{14}$$

where $x(n)$ is the the q-by-1 state vector and $u(n)$ is the $(m+1)$ -by-1 input vector. The first element of $u(n)$ is $+1$ and, in a corresponding way, the first element of $w_{b,j}$ is equal to the bias b_j applied to neuron j. The RTRL algorithm can be summarized as follows-

Parameters:

m = dimensionality of input space
q = dimensionality of state space
p = dimensionality of output space
w_j = synaptic weight vector of neuron

Initialization:

- Set the synaptic weights of the algorithm to small values selected from a uniform distribution.
- Set the initial value of the state vector $x(0) = 0$
- Set $\Lambda_j(0) = 0 for j = 1, 2, ...q.$

Computations: Compute for $n = 0, 1, 2.....,$

$$\Lambda_j(n+1) = \Phi(n)[W_a(n)\Lambda_j(n) + U_j(n)]$$
$$e(n) = d(n) - Cx(n)$$
$$\Delta w_j(n) = \eta C \Lambda_j(n) e(n)$$ The definitions of $x(n), \Lambda_j(n), U_j(n) and \Phi(n)$ are given in eqs 12, eqs 15, eqs 16 and eqs 17respectively. Introducing three new matrices $\Lambda_j(n), U_j(n), \Phi(n)$ as given below:

- $\Lambda_j(n)$ is a $q-by-(q+m+1)$ matrix defined as the partial derivative of the state vector $x(n)$ with respect to the weight vector w_j;

$$\Lambda_j(n) = \frac{\delta x(n)}{\delta w_j}, \qquad j = 1, 2....., q \tag{15}$$

- $U_j(n)$ is a $q-by-(q+m+1)$ matrix hose rows are all zero, except for the j^{th} row that is equal to the transpose of vector $\xi(n)$:

$$U_j(n) = \begin{pmatrix} 0 \\ \xi_n^T \\ 0 \end{pmatrix} \longleftarrow j^{th}, \quad j = 1, 2....., q \tag{16}$$

- $\Phi(n)$ is a q-by-q diagonal matrix whose k^{th} diagonal elements is the partial derivative of the activation function with respect to its argument, evaluated at $w_j^T \xi(n)$:

$$\Phi(n) = diag\ (\varphi'(w_1^T \xi(n)), ..\varphi'(w_j^T \xi(n)), .., \varphi'(w_q^T \xi(n))) \tag{17}$$

B. MLP Training:

The MLP is trained using (error) Back Propagation (BP) depending upon which the connecting weights between the layers are updated. This adaptive updating of the MLP is continued till the performance goal is met [2]. One cycle through the complete training set forms one epoch. The above is repeated till MSE meets the performance criteria. While repeating the above the number of epoch elapsed is counted. A few methods used for MLP training includes:

- Gradient Descent(GDBP)
- Gradient Descent with Momentum BP(GDMBP)
- Gradient Descent with Adaptive Learning Rate BP(GDALRBP) and
- Gradient Descent with Adaptive Learning Rate and Momentum BP(GDALMBP).

V. Combined Bi-Lingual Character - Numeral Recognition using MLP-RNN Hybrid Architecture

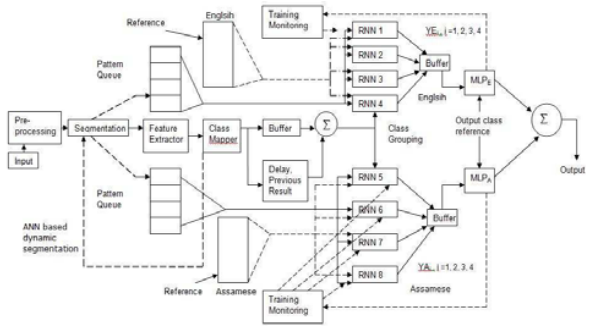

Fig. 6. Bi-Lingual Character - Numeral Recognition using MDFRRTRNN Architecture

The complete set-up is depicted in Figure 6. The system receives combined input which after pre-processing is passed through segmentation block. For hand-written inputs, the block uses a dynamic ANN-based method [16]. Next, the segmented samples are passed through the feature extraction block which as per the considerations described in Section II generates the feature set for each of the characters and numerals separately for use with the MLP-based class mapper. This classifier just places the inputs in two clusters in terms of the two languages namely English and Assamese. This stage slots the process into two sections which tries to evoke a cooperative nature in this ANN based recognition. Its significance lay in the fact that eight and nine in English are similar in appearance to four and seven in Assamese respectively for which a lot of miss hits were recorded in [13] and [14]. The use of the delay in conjunction with the buffer allows a language specific referencing while dealing with the problematic inputs (eight and nine in English and four and seven in Assamese) and placing them into respective class slots. It prevents false recognitions as seen in [13] and [14].

Training parameters of each of the RNN clusters are governed by the considerations described in Section 3.5.1. A training monitoring block is incorporated to work in coordination with the class-mapper MLPs (MLP_E and MLP_A) because training requirements vary with input. A numeral requires less epochs while a character needs more training session. Similar variations are observed with special characters as well. Hence, the training is adaptive and continues as per the demand.

As soon as the better of the training time or MSE criteria of the RNN clusters are met, the output of each of the RNN blocks are fed individually to the class-mapper MLPs which receives reference class codes to generate the output. The RNN clusters require a queue to hold the input patterns which are generated by certain directional scan as depicted in Figure 4.

Let a training set of m input - output pairs be $(x^1, t_1), (x^2, t_2), ...(x^m, t_m)$ be given and N networks are trained using this set of data. For simplicity, let for n-dimensional input there be a single output. Let for network functions f_i for a number of networks represented by indices $i = 1, 2, ...N$, the cooperative or committee network formed generates as output given as

$$f = \frac{1}{N}\sum_{i=1}^{N} f_i \tag{18}$$

The rationale behind the use of the averaging in the output of the cooperative or committee network as given by eq. 18 is the fact that if one of the constituent networks in the ensemble is biased to some part of the input samples, the ensemble average can scale down the prediction error considerably [1]. A quadratic error function can be computed from each of the error vectors e_i using the ensemble function f as

$$Q = \sum_{i=1}^{m}[t_i - \frac{1}{N}\sum_{i=1}^{N} f_i]^2 \tag{19}$$

Using matrix notation, the quadratic error can be expressed as

$$Q = |\frac{1}{N}(1, 1, ...)E|^2 = \frac{1}{N^2}(1, 1, ...1)EE^T(1, 1, ...1)^T \tag{20}$$

EE^T is the correlation matrix representing the error residuals. If each function approximation produces uncorrelated error vectors, the matrix EE^T is diagonal and the i^{th} diagonal element Q_i is the sum of quadratic deviations for each functional approximation, i.e $Q_i = \|e^i\|^2$. Thus,

$$Q = \frac{1}{N}(\frac{1}{N}(Q_1 + Q_2 + ... + Q_N)) \tag{21}$$

It implies that the total quadratic error of the ensemble is less by a factor $\frac{1}{N}$ than the average of the quadratic errors of the total computed approximations. This holds only if N is not very large. If the quadratic errors are not uncorrelated, i.e if EE^T is not symmetric, a weighted combination of N functions f_i can be approximated as

$$f = \sum_{i=1}^{N} w_i f_i \tag{22}$$

The weights w_i must be computed in such a way as to minimize the expected quadratic deviation of the function f for

the given training set. With the constraint with the constraint $w_1 + + w_N = 1$, eq. 20 transforms to

$$Q = \frac{1}{N^2}(w_1, w_2, ...w_N)EE^T(w_1, w_2, ...w_N)^T \quad (23)$$

Differentiating the above eq. 23 w. r. t $w_1, , w_N$, and using a Lagrangian multiplier λ for the constraint $w_1 + + w_N = 1$, the above functional modifies to

$$Q' = \frac{1}{N^2}wEE^T + \lambda(1, 1, ...1)w^T \quad (24)$$

$$= \frac{1}{N^2}wEE^T + \lambda 1 w^T \quad (25)$$

where 1 is a row vector with all its N components equal to 1. Setting the partial derivative of Q' with respect to w to zero this leads to

$$\frac{1}{N^2}wEE^T + \lambda 1 = 0 \quad (26)$$

With simplification,

$$\lambda = \frac{1}{N^2 1(EE^T)^{-1}1^T} \quad (27)$$

The optimal weight set can be calculated as

$$w = \frac{1(EE^T)^{-1}}{1(EE^T)^{-1}1^T} \quad (28)$$

assuming that the denominator does not vanish. This method, however, is dependent on the constraint that EE^T is not ill-conditioned.

VI. Experimental Results and Discussion

During training, the number of epochs elapsed and the MSE attained are recorded for each of the ANN blocks as well as the corresponding global values. For the dynamic segmentation stage and class mapper, the input has a size of 142 x 284 where 142 is the number of classes and 284 is the feature vector length. These include 89 Assamese and 26 each of capital and small case English letters and special characters. This set next expands to about 474 classes which include a few compound characters used in Assamese. Experiments were carried out by taking a sample size of 500 for both printed and handwritten characters. The next set of inputs consists of 20 classes of numerals of Assamese and English each having a feature vector size of 50. In case of the combined character and numeral recognition system, the feature vector is not used. It is replaced by the real-time scanned samples of the patterns used as depicted in Figure 4. A sample size of 500 for both printed and handwritten inputs have been considered for the experiment. Noise mixed input for both the cases of characters and numerals have been included to test the versatility of the models developed. The results are discussed separately as below:

1) *MDFRRTRNN Architecture* For a character input of 32 x 32 the RNN blocks with one hidden layer with tan-sigmoid activation and log-sigmoid activation at the input and output layers are trained so that the desired performance levels can be generated. The set-up is depicted in Figure 4. The RNN blocks are trained with RTRL-algorithm using variable learning rate and a momentum term which requires atleast 1.5 times more time than an MLP but reaches an MSE convergence value more than four times less while generating a success rate superior by 5 to 6 $\%$ consistently for a sample size of 500. This composite block requires, on average, 87.09 to 133.5 minutes to train on a stand alone system with a sample size of 100 characters each of 32 x 32 and 64 x 64 sizes written by five different persons. Success rate is around 96$\%$ consistently for hand-written forms while it is 100$\%$ for printed inputs. For numerals this time is around 75.1 to 111.2 minutes for 16 x 16 and 32 x 32 size variations. This value of success rate shows that the RNNs in a cluster in combination with heterogeneous blocks work better while receiving real-time raster inputs of the handwritten samples though the training sessions are much longer compared to a MLP.
2) *Combined Bi-Lingual Character - Numeral Recognition* This composite block requires, on average, 185 to 305 minutes to train on a stand alone system with a sample size of 100 characters each of 32 x 32 and 64 x 64 sizes written by five different persons. It is over two times compared to the time required by a single MLP-RNN Hybrid MDFRRTRNN block. Success rate is around 97$\%$ with regularity for hand-written forms which approaches 100$\%$ for printed cases. The cases of local minima are nearly non-existent. Out of about 75 training runs with all the inputs, only in two cases, the learning curves of one or two RNNs suffered a slowing down and the convergence oscillated around some MSE value. The training time for numerals with 16 x 16 and 32 x 32 size variations is about 22.5$\%$ less on an average after ten runs for each sample with a training set of 40 and testing set of 60. The success rates are marginally better.

Thus, heterogeneous clusters with objective-specific job distribution generate better accuracy of recognition in case of complex patterns like bi-lingual handwritten characters and numerals.

VII. Conclusion

The accuracy generated by a hybrid architecture formed by heterogeneous blocks shows superior success rates but needs more time to learn the input patterns. The hybrid architecture proposed in this work is better equipped to deal with a complex pattern classification task involving printed and handwritten characters and numerals in two different languages. The co-operative learning that forms the basis of working of the combined character - numeral recognizer tackles minor to sudden variations in input samples which are fed to the system captured with attributes of size / length and time varying nature. The hybrid configuration with heterogeneous ANN blocks is found to be well suited for such applications when modeled to work in a cooperative learning environment.
The success rate is aided by several factors. These are a class-mapper block which places the bi-lingual inputs into two definite language specific clusters. The performance of this block is boosted by a hybrid feature set which captures the relevant

details of the input samples to help the class-mapper MLP lay strict discrimination boundaries with minimum errors. This stage is also assisted by an ANN-driven dynamic segmentation block which is designed specially to deal with handwritten inputs.

In case of numeral recognition, earlier experiments carried out as part of the work showed miss hits between a few Assamese and English numbers due to similarity in appearance. This problem is tacked further by adopting a pattern matching block which keeps track of the previous language specific input handled and passing over the decision thus derived to the class-mapper to help make better discrimination and minimize faults due to miss hits.

The speed of execution, however, is an issue which can be resolved by using specialized hardware and device level parallelism.

REFERENCES

[1] R. Rojas, *Neural Networks A Systematic Introduction*, Springer, Berlin, 1996.
[2] S. Haykin, *Neural Networks- A Comprehensive Foundation*, 2^{nd} Ed., Pearson Education, New Delhi, 2003.
[3] J. L. McClelland and D. E. Rumelhart, "Distributed Memory and the Representation of General and Specific Information," Journal of Experimental Psychology: General, 114, pp. 159-188 1985.
[4] D. E. Rummelhart, G. E. Hinton and R. J. Williams, "Learning Representations by Back-Propagation Errors," *Nature*, 323, pp. 533-536 1986.
[5] T. Alvager, D. P. Beach and D. Herrmann, "A Hardware Implementation of Artificial Neural Networks," *Advanced Neural Computing Research Center Annual Report*, Aspen Technology Indiana State University, Terre Haute, IN, USA, 2001.
[6] M. Misra, "Implementation of Neural Networks on Paralel Architectures", PhD Thesis Presented to the Faculty of the Graduate School University of Southern California, December, 1992.
[7] U. A. Muler, A. Gunzinger and W. Guggenbuh, "Fast Neural Net Simulation with a DSP Processor Array", Electronics Laboratory, Swis Federal Institute of Technology, February, 1993.
[8] N. B. Serebdzija, "Simulating Artificial Neural Networks on Parallel Architectures", *Computer, Vol. 29, No. 3, pp. 56-63, 1996.*
[9] T. Schoenauer, A. Jahnke, U. Roth and H. Klar, "Digital Neurohardware: Principles and Perspectives", *Neuronal Networks in Applications - NN'98 - Magdeburg*, pp. 101-106, 1998
[10] N. Sundararajan and P. Saratchandran , "Parallel Architectures for Artificial Neural Networks ", *Journal of the Institute of Electrical and Electronics Engineers Inc.*, 1998.
[11] T. Heirs, "A High Performance Multiprocessor DSP System", Masters Thesis, Computer Science and Artificial *Intelligence Laboratory*, Massachusetts Institute of Technology, May, 2001
[12] K. K. Sarma, P. K. Bora and C. Mahanta, "Innovative Feature Set for Multi Layered Perceptron (MLP) Based Assamese Character Recognition", *Proceedings of 2nd Indian International Conference on Artificial Intelligence (IICAI-2005), pp. 473- 491, 2005.*
[13] K. K. Sarma, "Novel Feature Set for Neural Character Recognition " , *Proc. of the 5th International Symposium on Robotics and Automation, San Miguel Regla Hidalgo, Mexico , pp. 409-414, August, 2006.*
[14] K. K. Sarma, "MLP-based Assamese Character and Numeral Recognition using an Innovative Hybrid Feature Set", *Proc. Of the Proceedings of 3rd Indian International Conference on Artificial Intelligence (IICAI-2007), Pune, India , pp. 585 -600, December, 2007.*
[15] K. K. Sarma, "Modified Hybrid Feature Set for Assamese Character and Anglo-Assamese Hand Written Numeral Recognition", *International Journal of Imaging, Volume 1, pp. 13 -28, Autumn 2008.*
[16] K. Bhattacharjya and K. K. Sarma, "ANN-based Innovative Segmentation Method for Handwritten text in Assamese", *IJCSI International Journal of Computer Science Issues, Vol. V, No. IJCSI-2009-10-20, October, 2009.*

Kandarpa Kumar Sarma , presently with the Department of Electronics and Communication Technology, Gauhati University, Assam, India, completed MSc in Electronics from Gauhati University in 1997 and MTech in Digital Signal Processing from IIT Guwahati, Guwahati, India in 2005 where he further continued his research work. His areas of interest include Applications of ANNs and Neuro-Computing, Document Image Analysis, 3-G Mobile Communication and Smart Antenna.