Online Bengali Handwritten Numerals Recognition Using Deep Autoencoders

Arghya Pal*, B. K. Khonglah†, S. Mandal†, Himakshi Choudhury†,
S. R. M. Prasanna †, H. L. Rufiner‡ and Vineeth N Balasubramanian*
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad, Telangana 502285 *
Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati, Guwahati-781039 †
Facultad de Ingeniera y Ciencias Hdricas - Universidad Nacional del Litoral - CONICET,
Ciudad Universitaria, Paraje El Pozo, S3000 Santa Fe, Argentina ‡
E-mail: {cs15resch11001, vineethnb}@iith.ac.in, {banriskhem, prasanna}@iitg.ernet.in, lrufiner@gmail.com

***Abstract*—This work describes the development of online handwritten isolated Bengali numerals using Deep Autoencoder (DA) based on Multilayer perceptron (MLP) [1]. Autoencoders capture the class specific information and the deep version uses many hidden layers and a final classification layer to accomplish this. DA based on MLP uses the MLP training approach for its training. Different configurations of the DA are examined to find the best DA classifier. Then an optimization technique have been adopted to reduce the overall weight space of the DA based on MLP that in turn makes it suitable for a real time application. The performance of the DA based system is compared with systems constructed using Hidden Markov Model (HMM) and Support Vector Machine (SVM). The confusion matrices of DA, HMM and SVM are analyzed in order to make a hybrid numeral recognizer system. It is found that hybrid system gives better performance than each of the individual systems, where the average recognition performances of DA, HMM and SVM systems are 97.74%, 97.5% and 98.14%, respectively and hybrid system gives a performance of 99.18%.**

I. Introduction

Autoencoders are machine learning tools having same output vector dimension as input vector dimension aiming to reconstruct input with minimum reconstruction error. If the number of hidden layers are more than one, the autoencoder is considered to be Deep. Autoencoder network based on Restricted Boltzmann Machine (RBM) proposed in [2] is used to form a Deep Network. This work is motivated by [1], where Deep Autoencoder (DA) based on Multilayer perceptrons (MLP), trained through the back propagation algorithm has been explored for the task of speech emotion recognition. Good initial weights of DA were obtained using a method of pre-training and was shown to give promising results. It is here worthy to mention that though the functional form of RBM and autoencoder are quite similar, their training procedure and interpretation are quit different. But both of them can be extended to form a deep network to perform several classification and regression tasks. In this work we have explored the use of DA (from now onwards we use DA to mean Deep Autoencoder based on MLP) for the task of online handwritten Bengali numeral recognition. This work finds its place in various domain like online form filling applications, census data collection, banking operations, number dialing systems and many more.

978-1-5090-2361-5/16/$31.00 © 2016 IEEE

Some of the works that recognizes online handwritten characters from different perspectives include HMM based handwriting recognition systems for Bangla [3], Tamil [4], Telugu [5], Assamese [6], SVM based system for Tamil [7] and Devanagari [8]. There are reported claims on handwritten Bengali characters using MLPs [9], [10], but those works mainly focus on offline recognition, which is not compatible with the present study of online recognition.

A central goal of this work is to classify handwritten online handwritten Bengali numerals by using DA based on MLP. By adding a final classification layer DA was converted into a Deep Classifier (DC). The final classification layer worked mainly based on "winner takes all" philosophy. Best configuration of DC was chosen as per the validation set performance. The present work is different from [1], in two ways. First, we have used an approximation of standard backpropagation known as "Marquard Bacpropagation Algoritm" (MBP) [11] to incorporate a higher guarantee to converge. For the rest of the paper, backpropagation will imply MBP and not the simple backpropagation. Secondly, inspired by the work in [12], we further applied an optimization technique to the weight space of the deep network to reduce its size and yields a network with good recognition time making it feasible for a real-time application. The average recognition rate of the DA was then compared with HMM and SVM and class-wise performances of the DA, HMM and SVM systems were analyzed, to build a hybrid system using the algorithm *combination*, so as to further improve the recognition rate. A similar kind of work can be found in [13], where the combination of HMM-SVM had been done using test set performance. But the main difference is that our work has taken validation set performance to combine those state-of-art systems.

The paper is organized as follows: Section II describes data collection and feature extraction method. Online Bengali hand-

Hindu-Arabic Numerals	০	১	২	৩	৪	৫	৬	৭	৮	৯
Arabic-Indic Numerals	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Bengali / Assamese Numerals	০	১	২	৩	৪	৫	৬	৭	৮	৯

Fig. 1. Last row represents basic Bengali Numerals (one-to-one relation with Indo-Arabic numerals)

written recognition system using DA classifier is described in Section III. Section IV explores a technique to reduce the DA network. Section V explores the use of other state-of-art systems such as HMM and SVMs for online Bengali handwritten recognition. Hybrid system is described in Section VI. The summary and conclusion is given in Section VII.

II. Data Collection and Feature Extraction

Having a one-to-one relation with Indo-Arabic numeral set, Bengali numeral consists of ten characters (Fig. 1). In the present study, we considered recognition of isolated Bengali numeral mentioned in Fig. 1. Raw data was collected from 200 native Bengali writers using an open source toolkit provided by C-DAC Pune on the Lenovo ThinkPad PC in different sessions, without putting any constraint to the writers. A mixture of frequent writers, non-frequent writers, writers form different age groups, professions was collected to ensure robustness of the system for a real world situation. At each session each writer had given ten samples of each numeric character and that made a total count of 30000 raw samples per class. From that collected raw data 20000 samples of each numeric characters had been reserved for training and 5000 samples of each numeric characters had been reserved for both testing and validation to make a clear segregation between train set, test set and validation dataset. Due to the large variations in shape, size, writing style and position in unconstrained collected raw data (i.e. temporally ordered sequence of pen coordinates (x, y)), some preprocessing techniques like normalization, smoothing, removal of duplicate points and re-sampling [13] have been done to find fixed length feature vector. In general, preprocessing was employed to mitigate the intraclass variations and to provoke inter class variations. During the preprocessing (i.e. normalization, smoothing, removal of duplicate points and re-sampling), the first and last points of each raw data were unchanged as they contain vital information [13]. So, preprocessing for each numeral gave 60 equidistant points x_p, y_p where p = 1 to 60, comprising of pen-down, pen-up and other 58 equidistant meaningful points. Preprocessed coordinates x_p, y_p along with first derivatives (i.e. $x_p^{'}$ and $y_p^{'}$) and second derivatives (i.e. $x_p^{''}$ and $y_p^{''}$) of each preprocessed point made a feature vector of dimension 360. So, input to each state-of-art system in this present study is actually a vector with 360 dimension, i.e. 6 (x_p, y_p, $x_p^{'}$, $y_p^{'}$, $x_p^{''}$ and $y_p^{''}$) multiplied with 60 points each. This set of parameters has already been studied [13] and have reported a major overall information gain to discriminate numeric characters.

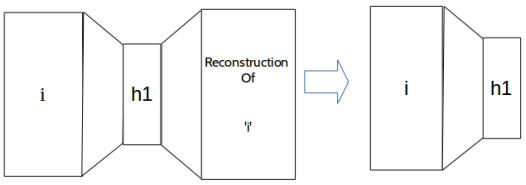

Fig. 2. Autoencoder with input, first hidden layer and reconstruction of input; i. e. $i + h_1+$ reconstruction of i

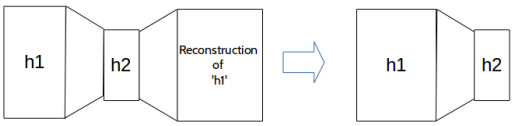

Fig. 3. Autoencoder with first hidden layer, second hidden layer and reconstruction of first hidden layer; i. e.$h_1 + h_2+$reconstruction of h_1

III. Online Numeral Recognition using Deep Autoencoders (DA)

Deep Autoencoder (DA) network is feed forward and fully connected network consisting of an input layer i, two hidden layers h_1 and h_2 and one output layer o. Six feature set x_p, y_p, $x_p^{'}$, $y_p^{'}$, $x_p^{''}$ and $y_p^{''}$, each having sixty equidistant points acquired from feature extraction set is unrolled to form a 360 dimensional feature vector which is the input to DA. Making the input and output class fixed, the number of units of each hidden layer was modified in order to find the best classifier.

Different from the greedy layer wise training algorithm in [14], a strategy has been taken by [1] which finds good initial weights by learning one layer of feature at a time known as pre-training the DA. To cut a long story short, main steps proposed in [1] are as follows:

- Step 1: To initialize weights corresponding to links between the input i and first hidden layer h_1(i.e. $i+h_1$), an autoencoder with $i + h_1+$ reconstruction of i had been used. Different values of hidden layer was tested (See Table II) in order to find best autoencoder which gave minimum Sum of Squared Errors (SSE) for the validation dataset.
- Step 2: $i + h_1$ (i.e. coder part) has been taken from step 1 using those networks which gave a minimum SSE for train dataset (see Fig. 2). Train and validation set was passed to generate output.
- Step 3: Output of the previous step was the input to the next autoencoder, i.e. h_1+h_2+ reconstruction of h_1. This $h_1 + h_2+$ reconstruction of h_1, autoencoder initialized link weights between first hidden layer h_1 and second hidden layer h_2.
- Step 4: From different configurations (Table II), best autoencoder has been selected using the same criterion described previously. By removing the output layer of $h_1 + h_2+$ reconstruction of h_1 network, $h_1 + h_2$ (i.e. coder part) was obtained (see Fig 3).
- Step 5: Construction of DA with input i, first hidden

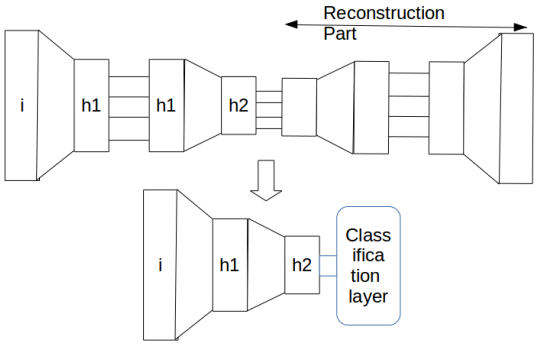

Fig. 4. Construction of Deep autoencoder and Deep classifier

layer h_1, second hidden layer h_2, using networks $i + h_1$ and $h_1 + h_2$, has two fold. First, $i + h_1$ and $h_1 + h_2$ were combined by making one-to-one link (one-to-one links are not supposed to being modified during training) between units h_1 of $i + h_1$ and $h_1 + h_2$ which results the construction of network $i + h_1 + h_2$. The network $h_2^{'} + h_1^{'} + i^{'}$ is created by inverting network $i + h_1 + h_2$ leaving weights unmodified. At last DA was constructed by merging $i + h_1 + h_2$ and $h_2^{'} + h_1^{'} + i^{'}$ where h_2 and $h_2^{'}$ junction was treated as one unit h_2. The best DA, i.e. $i + h_1 + h_2$, is obtained by eliminating the decoder part. Train and validation set was passed to generate output, which on the other hand, was the input to the next layer which consists of initializing output layer weights. The output layer weights of the classifier were obtained by taking a copy of h_2 (which act as input layer) and output layer consisting of 10 classes corresponding to the 10 basic characters of Bengali numerals. Input patterns were the output from network $i + h_1 + h_2$ and the desired output is as per the labeled data provided in database. The classification rule applied in the output layer was "winner takes all" [1]. After a final "fine-tuning", the best classifier is selected (see Fig 4).

Backpropagation algorithm free parameters consist of Learning Rate (LR) , Momentum (M), Flat Spot Elimination (FSE) and Tolerance (T). Two combinations [1] of LR, M, FSE and T have been taken namely Parameters A (PA) and Parameters B (PB) for the experiment described in Table I and Table II.

TABLE I
TRAINING PARAMETERS FOR STEP 1,3,5(REF. [1])

	Step 1				Step 3				Step 4			
	LR	M	FSE	T	LR	M	FSE	T	LR	M	FSE	T
PA	0.08	0.1	0.1	0.1	0.08	0.1	0.1	0.7	0.07	0.08	0.1	0.1
PB	0.08	0.08	0.07	0.07	0.08	0.1	0.07	0.07	0.05	0.05	0.07	0.07

A series of experiment with seven different network structures and two combinations of backpropagation algorithm parameters, leading to fourteen configurations was conducted to evaluate the performance of DA. Removing the restriction (i.e. the number of units in a layer should be less than in the previous one) which stated in [1], different configurations (Table II) were tested and it was evident that for handwritten Bengali numerals, network layout $360 + 140 + 80 + 10$ gave best performance compared to the other configurations, where 360 is the input dimension, 140 is the number of units in first hidden layer h_1, 80 is the number of units in second hidden layer h_2 and 10 denotes output class as Bengali numerals has ten basic characters. So, configurations shows in Table II are in the form $i + h_1 + h_2 + o$. From the Table II, it is clear that using PB the results are scattered between 94.935% to 98.54% having a standard deviation 0.94. But, using PA the results are scattered between 90.042% to 95.62%, giving a standard deviation of 1.6756. So, it is clear that PB gives good result than PA. The performance of the model for the validation set (consisting of 5000 samples) is shown in the form of confusion matrix and is depicted in Table III.

TABLE II
AVERAGE RECOGNITION ACCURACIES OF DA WITH DIFFERENT CONFIGURATIONS

Configurations	PA (in %)	PB (in %)
360+200+130+10	94.87	94.935
360+210+140+10	92.561	95.3125
360+200+170+10	93.32	95.7375
360+240+170+10	95.62	95.5867
360+240+200+10	94.164	96.975
360+140+70+10	90.042	96.73
360+140+80+10	93.43	**98.54**

TABLE III
CONFUSION MATRIX OF DA

	0	1	2	3	4	5	6	7	8	9
0	**98.8**		1		0.2					
1		**95.4**						1		3.6
2			**99.2**							0.8
3	1			**98.8**			0.2			
4	0.6		0.2		**99.2**					
5						**99.6**	0.4			
6						0.4	**99.6**			
7	1		0.2	3				**95.8**		
8		1			1		2		**96**	
9		3		1		1				**95**

IV. COMPRESSING DEEP AUTOENCODER (DA) TO MAKE IT SUITABLE FOR A REAL TIME APPLICATION

DA gives reasonably good results for problems that need high level of abstraction such as digit recognition, language processing, object recognition, fraud detection etc. But as the problems become complex, DA needs several layers with lots of parameters that sometime suffers from over fitting and makes it infeasible for a real-time application. To make the DA feasible for a real time application we used the concept popularly known as "Optimal Brain Damage (OBD)" [12], that helped us to make a network having a better generalization with good learning and classification capacity by selectively deleting unimportant weights from DA. Unimportant weights means those weights in the network whose expunction will give less / negligible effect on the training error. Expunction of unimportant weights was done by thresholding "saliency",

TABLE IV
CONFUSION MATRIX OF HMM

	0	1	2	3	4	5	6	7	8	9
0	**98.2**	0.8		0.4						0.6
1	0.2	**99.8**						1		3.6
2			**93.4**		5.2	1.4				0.8
3				**98.8**						1.2
4					**99**	1				
5			0.4		9	**90.6**				
6	0.4				1		**98.6**			
7				1.2				**98.8**		
8					0.4				**99.6**	
9				1		0.8				**98.2**

where "saliency" explicitly depends on second derivative of each parameter [12]. So, second derivative of each parameter was computed (say, h_{kk}) after training of DA and using those h_{kk} we had computed "saliencies" for each parameter. It is defined as

$$saliencies = \frac{h_{kk} * w_{ij}^2}{2} \quad (1)$$

Here w_{ij} is the connection from unit j and unit i for all (i, j). Parameters with "saliencies" below a threshold can be effectively deleted and hence gives a reduced version of the network with less parameter, improved learning and classification ability. Using this technique $360 + 140 + 80 + 10$ get reduced to $360{+}20{+}12{+}10$ with 98.04% recognition accuracy at validation dataset and recognition time is reduced to 80% with respect to the original network without optimization.

V. ONLINE NUMERAL RECOGNITION USING HMM AND SVM

Using the same set of training, testing and validation data used to model DA, other state-of-art methods like HMM and SVM are explored. These models are then compared with the DA in terms of the class-wise performance and statistical significance.

HMM is a stochastic process belief to follow Markov process with hidden states. A HMM is a tuple (S, Σ, π, a, b); S = set of states $\{s_1, s_2, ..., s_N\}$; Σ = set of output symbols $\{w_1, w_2, ..., w_M\}$; π = initial state distribution $\pi_i = P(q_1 = i)$; a = state transition probability, $a_{ij} = P(q_{t+1} = s_j | q_t = s_i)$, $1 \leq i, j \leq N$ for a given i, j; b is emission probability, for a give j, k $b_{jk} = P(O_t = v_k | s_t = j), k = 1, 2, ..., M$, where O_t is the observation and s_t is the active state at time t. In our experiment we have taken left to right HMM without state skipping. The best result is obtained using 7 states and 20 Gaussian mixtures. Average recognition rate of HMM in validation dataset is 97.5%. Confusion matrix for this system have shown in Table IV. It shows that there is a confusion between numeral class 5 and 6 as their writing style is almost same. Similarly, for class 1 and 9 , 2 and 4 there is also a confusion since they have similar curvatures.

SVM with radial basis function as kernel had been used to classify isolated handwritten Bengali numerals. All experiments using SVM in this work had been carried out using the LibSVM tool [15]. It is found that SVM had given best average recognition accuracy of 98.14% using c = 8 and $gamma$ = 0.125 in validation dataset. Class-wise performance is given in the last row of Table V. The confusion between numerals 5 and 6 and also between 1 and 9 is reduced by the SVM classifier as SVM uses the philosophy called margin-of-separation different from temporally ordered sequence of data used by HMM. Average recognition rate using SVM is 98.14%. Class-wise recognition accuracy is shown in Table V.

TABLE V
CLASS WISE PERFORMANCE OF EACH SYSTEMS

	0	1	2	3	4	5	6	7	8	9
HMM	98.2	**99.8**	93.4	98.8	99	90.6	98.6	**98.8**	**99.6**	**98.2**
DA	98.8	95.4	**99.2**	98.8	**99.2**	**99.6**	**99.6**	95.8	96	95
SVM	**99**	98.8	97.4	**99**	97.6	98.6	98.8	97.4	97.8	97

VI. HYBRID NUMERAL RECOGNIZER USING HMM, DA AND SVM

All the state-of-art systems DA, SVM and HMM had been studied exclusively from the standpoint of their basic philosophy. It had been observed that each of those state-of-art systems had have their own pros and cons in terms of recognition and that leads to the immediate idea of integrating them to form a combined recognizer. It can be observed from Table V that the average recognition rate of SVM is slightly greater than that of the DA and HMM. It can be observed that for validation dataset numerals 1, 7, 8 and 9 HMM gives the best performance while for numerals 0 and 3 SVM gives the best performance. For numerals 2, 4, 5 and 6 the DA classifier performs the best among the three classifiers in validation dataset. The class-wise performance/scores were then normalized and class wise performance at validation set were analyzed to construct a hybrid model. Inspired by *Comb-2* in [13], an algorithm named as *Combination* was developed to make a hybrid system using DA, HMM and SVM for Bengali numerals taking DA as baseline classifier. Different from *Comb-2*, *Combination* will consider Validation set performance rather than testing set performance described in *Comb-2*. The basic idea of algorithm *Combination* is as follows:

1) The class wise performance and confusion matrices of HMM, SVM and DA systems with validation dataset were compared by considering DA as the baseline classifier. For a given class C_i (where, C_i could be any of the 10 classes, i.e. i = class 1 (i.e. character 0), to, class 10 (i.e. character 9)) performance of DA, HMM and SVM were analyzed and any improvement of score produced by HMM or SVM for that class with respect to DA performance was recorded. For example, consider any arbitrary class C_i and say performance of DA, HMM and SVM are C_i^d,C_i^h and C_i^s respectively. Now we had taken maximum score among C_i^d,C_i^h and C_i^s for that class C_i. Following this, it was observed that HMM gave the best performance for numerals 1, 7, 8 and 9 and for numerals 0, 3 SVM gave the

English Numerals	0	1	2	3	4	5	6	7	8	9
Numerals	০	১	২	৩	৪	৫	৬	৭	৮	৯
Accuracy	99 SVM	99.8 HMM	99.2 DAM LP	99 SVM	99.2 DAM LP	99.6 DAM LP	99.6 DAM LP	98.8 HMM	99.6 HMM	98.2 HMM

Fig. 5. Class wise recognition considering best recognition performance of three classifiers HMM, DA and SVM.

TABLE VI
CONFUSION MATRIX OF HYBRID SYSTEM

	0	1	2	3	4	5	6	7	8	9
0	99.4	0.6	-	-	-	-	-	-	-	-
1	-	100	-	-	-	-	-	-	-	-
2	-	2	97.2	-	-	-	-	-	0.8	-
3	-	-	-	99.8	0.2	-	-	-	-	-
4	-	-	-	-	99.8	0.2	-	-	-	-
5	-	-	-	-	-	98	2	-	-	-
6	-	-	-	-	-	-	100	-	-	-
7	1.2	-	-	-	-	-	-	98.8	-	-
8	-	-	-	-	-	-	-	-	100	-
9	-	1.2	-	-	-	-	-	-	-	98.8

best result. For numerals 2, 4, 5 and 6, DA gives best result in validation dataset. Fig. 5 shows class wise the best performances given by three systems HMM, DA and SVM. Using this knowledge results were further combined to develop a hybrid numeral recognizer.

2) It was observed that the average performances of the three systems were comparable as we had used same data partitions for training, testing and validation for modeling the three methods HMM, DA, SVM, i.e. 97.5% , 97.74% and 98.14% respectively.

3) If a validation sample had been predicted by HMM classifier as one among numerals 1, 7, 8 and 9, then output from HMM was being considered as the final output or, if the predicted output was one of the numerals 2, 4, 5 and 6 the output from DA had been taken as the final output and if the predicted output was one of numerals 0, 3 the output from SVM had been taken as the final output.

4) If any confusion is found at the time of predicting the test sample, DA output for that test sample can be considered to break the confusion and hence DA acts as baseline classifier.

The average recognition rate of the hybrid system is 99.18% which is much better than each of the individual systems. Confusion between numerals 5 and 6 which arises in HMM and confusion between 1 and 9 which arises in DA systems are significantly reduced in the hybrid system. The confusion matrix is shown in Table VI.

To ensure the significance of hybrid system with HMM, DA and SVM, we developed four other hybrid systems which consists of HMM-SVM (HMM as baseline classifier), HMM-DA (DA as baseline classifier), SVM-DA (DA as baseline classifier) and SVM-DA (SVM as baseline classifier). The testing procedure is followed with same combination algorithm and same database (i.e. same train, test and validation set) described at Section III. Average recognition accuracies are listed in Table VII.

TABLE VII
AVERAGE RECOGNITION RATES OF HYBRID SYSTEMS

Hybrid System	Overall Performance(%)	Baseline classifier
HMM-SVM	98.2	HMM
HMM-DA	**98.66**	**DA**
DA-SVM	98.4	DA
DA-SVM	98.46	SVM
HMM-DA-SVM	**99.18**	**DA**

It is observed from Table VII that the hybrid systems which have been constructed by combining DA with any other state-of-art system i.e. HMM or SVM, gives better performances than the hybrid system which is constructed by using HMM and SVM. This shows the significance of DA in the hybrid system comprising of HMM-DA-SVM combination.

VII. SUMMARY, CONCLUSION AND FUTURE WORK

In this paper we have explored the use of DA for online Bengali numeral recognition. Recognition accuracy at testing obtained with compressed DA $360 + 40 + 80 + 10$ is 98.04%. Recognition result with this configuration is fairly good (i.e. 97.74%) and gives a near performance with respect to other state-of-arts HMM and SVMs. Then following the combination algorithm a hybrid numeral recognizer is made, which gives a very good performance (i.e. 99.18%) than all individuals. To ensure the significance of DA in the hybrid system with HMM-DA-SVM, a test has been performed following combination algorithm and database (i.e. same train, test and validation set) with four other hybrid systems.

The DA could be explored further to make a denoising autoencoders by adding small amounts of noise in each stage which could improve robustness of the final classifier. In the same way, the so called "drop out" method that randomly eliminates some inputs but maintains the "pressure" of reconstruction at the output of each autoencoder could be studied. Other interesting issue to discuss for future work could be the analysis of the internal representations learned by DA.

REFERENCES

[1] Cibau, Neri E., Enrique M. Albornoz, and Hugo L. Rufiner. "Speech Emotion Recognition using a Deep Auto Encoder." *Anales de la XV Reunion de Procesamiento de la Informacion y Control* (2013): 16-20.

[2] Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the Dimensionality of Data with Neural Networks." *Science* 313.5786 (2006): 504-507.

[3] S.K. Parui, K. Guin, U. Bhattacharya, and B.B. Chaudhuri, "Bangla Handwritten Character Recognition using HMM, in Proc. *19th Int. Conf. on Pattern Recognition (ICPR)*, pp. 1-4, Tampa FL, 2008

[4] Ramakrishnan, A. G., and K. Bhargava Urala. "Global and local features for recognition of online handwritten numerals and Tamil characters." Proceedings of the *4th International Workshop on Multilingual* OCR. ACM, 2013.

[5] V.J. Babu,L. Prasanth, R.R. Prasanth, R.R. Sharma, G.V.P. Rao and A. Bharath, "HMM-based Online Handwriting Recognition System for Telugu Symbols," Proc. *9th Int. Conf. on Document Analysis and Recognition (ICDAR)*, Curitaba, Brazil, 2007, pp. 63-67.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
Arghya Pal, B. K. Khonglah, S. Mandal, Himakshi Choudhury, S. R. M. Prasanna, H. L. Rufiner & Vineeth N. Balasubramanian; "Online Bengali Handwritten Numerals Recognition Using Deep Autoencoders"
Proc. of the 22nd National Conference on Communications (NCC 2016), mar, 2016.

[6] G.S. Reddy, B. Sarma, R.K. Naik, S.R.M. Prasanna and C. Mahanta, "Assamese Online Handwritten Digit Recognition System using Hidden Markov Models," Workshop on *Document Analysis and Recognition*, 2012

[7] A. Arora and A.M. Namboodiri, "Hybrid Model for Recognition of Online Handwriting in Indian Scripts," Proc. of *Int. Conf. on Frontiers in Handwriting Recognition* , pp. 433-438, Kolkata, 2010.

[8] H. Swethalakshmi, A. Jayaraman, V.S. Chakravarthy and C. Chandra Sekhar, "Online Handwritten Character Recognition of Devanagari and Telugu Characters using Support Vector Machines," *International Workshop on Frontiers in Handwriting Recognition*, 2006.

[9] U. Bhattacharya, M. Shridhar, and S.K. Parui, "On Recognition of Handwritten Bangla Characters," *ICVGIP* 2006, LNCS 4338, pp. 817828, 2006.

[10] T.K. Bhowmik, U. Bhattacharya, S.K. Parui, "Recognition of Bangla Handwritten Characters Using an MLP Classifier Based on Stroke Features," *11th International Conference, ICONIP*, Calcutta, India, November 22-25, 2004.

[11] Werbos, P. J, "Training Feed Forward Networks with the Marquardt Algorithm," 1994, *IEEE Transactions on Neural Networks*, vol. 5, no. 6, November 1994

[12] Yann Le Cun and John S. Denker and Sara A. Solla, "Optimal Brain Damage",Advances in Neural Information Processing Systems,1990, 598–605.

[13] Sarma, Bandita, et al. "Handwritten Assamese Numeral Recognizer using HMM & SVM Classifiers." *Communications (NCC), 2013 National Conference* on. IEEE, 2013.

[14] G.E. Hinton, S. Osindero, Y. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*,vol. 18, no. 7, pp. 1527-1554, 2006.

[15] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM : A Library for Support Vector Machines," *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm"

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
Arghya Pal, B. K. Khonglah, S. Mandal, Himakshi Choudhury, S. R. M. Prasanna, H. L. Rufiner & Vineeth N. Balasubramanian; "Online Bengali Handwritten Numerals Recognition Using Deep Autoencoders"
Proc. of the 22nd National Conference on Communications (NCC 2016), mar, 2016.