Behavior of Learning Rules in Hopfield Neural Network for Odia Script

Ramesh Chandra Sahoo[1]
Research Scholar, Dept. of Computer Science and Applications, Utkal University, Bhubaneswar, India

Sateesh Kumar Pradhan[2]
Dept. of Computer Science & Applications
Utkal University, Bhubaneswar, India

***Abstract*—Automatic character recognition is one of the challenging fields in pattern recognition especially for handwritten Odia characters as many of these characters are similar and rounded in shape. In this paper, a comparative performance analysis of Hopfield neural network for storing and recalling of handwritten and printed Odia characters with three different learning rules such as Hebbian, Pseudo-inverse and Storkey learning rule has been presented. An experimental exploration of these three learning rules in Hopfield network has been performed in two different ways to measure the performance of the network to corrupted patterns. In the first experimental work, an attempt has been proposed to demonstrate the performance of storing and recalling of Odia characters (vowels and consonants) in image form of size 30 X 30 on Hopfield network with different noise percentages. At the same time, the performance of recognition accuracy has been observed by partitioning the dataset into training and a different testing dataset with *k*-fold cross-validation method in the second experimental attempt. The simulation results obtained in this study express the comparative performance of the network for recalling of stored patterns and recognizing a new set of testing patterns with various noise percentages for different learning rules.**

***Keywords*—Hopfield network; Odia script; Hebbian; pseudo-inverse; Storkey; NIT dataset**

I. INTRODUCTION

Odia language is one the oldest and official language of Odisha state in the Indian subcontinent used by more than 50 million people. Modern Odia language consists of 47 different characters out of which 11 are vowels and rest are consonants. Automatic recognition of these Odia characters is particularly difficult because many of these characters are similar looking rounded in shape. Many works have been done so far in the field of OCR (Optical Character Recognition) for various Indian languages but very little research has been done for Odia language. In this study we focused on 11 Odia vowels and 36 basic consonants for storing into and recalling from Hopfield network with different learning rules especially when the patterns gets distorted. We add explicit noise into the input patterns from 10% to 50% while recalling them and compare their results for analysis. The main idea for choosing Hopfield network is that it can be a multiple point attractors for high dimensional space and due to the dynamics of network that guaranteed to convergence to local minima. Hopfield network is an auto associative memory network that reproduces its input pattern as an output even if the input pattern is distorted or partially feed into the network. It a fully connected special type of recurrent network excluding self connection with binary or bipolar inputs. The network returns a stored best matching pattern when an input pattern is presented to the network. The experimental simulation have been carried out in MATLAB2014a environment and the performance parameters such as recalling stored patterns and recognition accuracy of new patterns with noisy version is validate and compared with other techniques in the literature.

II. RELATED WORK

Many articles have been published for various Indian languages like Telgu [1], Tamil [2], Gujarati [3], Kannada [4], Bangla [7] and Gurmukhi [5]. Whereas for Odia character recognition, there are few articles available for study. Development of a novel OCR system for Odia character is difficult and challenging task because most of these characters are identical and roundish shape. In 2002 Chaudhuri et al [6] suggested a system for printed odia script with 96.3% accuracy. Mohanti [8] proposed a system by using Kohonen network to recognize Odia alphabets. Roy et al. [9] proposed a system for Odia handwritten numerals with chain code histogram with a accuracy of 94.8%. Bhowmik et al. [10] suggested a system for Odia handwritten digits with approximately 93% accuracy by using hidden markov as a classifier. Sarangi et al. [11] proposed a classifier by using Hopfield neural network for Odia numerals. Panda et al. [12] proposed a single layer perception for Odia handwritten digits recognition by using gradient and curvature feature extraction methods with an accuracy of 85%. Das et al.[13] presented a hybrid method for Odia digit classification using Kirsch operator. Meher et al.[14] proposed a method for Odia character recognition in the presence of vowel modifiers of complex Odia characters with an accuracy of 91.24%. Kumar et al.[15] proposed an Ant-Miner algorithm for Odia character recognition with an accuracy of 90%. Mishra et al.[16] proposed a fuzzy based algorithm using HMM classifier for handwritten characters with an accuracy of 96.3%. Pujari et al.[17] compares different algorithms proposed so far for Odia character recognition. Chen L –C et al.[18] proposed a modified Hopfield network for character recognition system by using a single error correction-single error detection method to improve the learning in the network. M.B. Sukhaswami et al.[19] proposed a method for Telgu character recognition using Hopfield network and then modified it to Multiple Neural Network Associative Memory (MNNAM) to overcome the storage capacity limitation of Hopfield network. Prateek Malik et al.[20] proposed handwritten character recognition using Hopfield network and wavelet transforms

with various noise percentage. Kalyan S Dash et al. [25], Proposed a hybrid feature based model for Odia numerals with ISI Kolkata database. With hybrid feature and DLQDF classifier they claimed a 98.50% of recognition accuracy and with MQDF classifier they achieved 98.40% of accuracy. Kalyan S Dash et al. [26] presented a review report on various preprocessing, segmentation and feature extraction techniques with several classifiers for Odia character recognition. They also created a standard alphanumeric Odia handwritten character database. The comparisons presented there are being taken into consideration for this work. Indugu Rushiraj et al. [27] Achieved 87.60% accuracy for Odia consonants classification by using shadow, centroid and distance based features with weighted Euclidean distance method for classification. Smruti Rekha Panda and Jogeswar Tripathy [28], proposed a template matching with Unicode mapping for Odia offline character recognition with a accuracy of 97% for basic characters and numerals. Kalyan S Dash et al. [29], proposed a novel feature extraction method called BESAC: binary external symmetry axis constellation with a Boolean matching character recognition technique for Odia and Bangla Numerals and characters. They achieved 99.35% accuracy with ISI Kolkata Odia Numerals, 98.90% for IITBBS Odia numerals, 99.48% for Bangla Numerals and an accuracy of 95.01% for Odia Characters. Subhashree Satapathy et al. [30], proposed a deep neural Autoencoder for dimensionality reduction as well as classification of Odia digits. They achieved a 97.63% of accuracy on 37 training set and different test accuracies obtained as 94.69% on 15 training sets, 97.1% on 20 training sets and 97.4% accuracy on 30 training sets. In Table I, a detail study on Odia characters and numerals recognition is shown with their claimed accuracy.

III. Odia Character Set and Chanlleges

Odia language is the mother tongue of the Indian state of Odisha and the script originally came from Brahmi script. The way of writing Odia language is unique and different from other regional languages in India. Modern Odia script consists of 11 vowels, 36 consonants and ten digits. Apart from these basic Odia character set, the Odia ligatures may be formed either merging a vowel diacritic with the consonant or by clustering two or more consonants. There are nearly 116 composite characters also known as Juktakhyaras. A printed version of Odia characters and numerals are shown in Fig. 1, 2 and 3 for vowels, consonants and numerals respectively. Most of the Odia characters are roundish in shape and very similar to each other. Recognition of Odia characters are more challenging pattern recognition task as i) many roundish and similar shape, ii) there is no reference line as in other Indian regional languages like Hindi and Bangla, iii) unavailability of more number of standard databases for both printed and handwritten Odia characters. Few samples of Odia characters with almost similar in shape are shown in Fig. 4.

IV. Hopfield Neural Network

Hopfield network [21] is merely the best known auto-associator neural network that acts as content addressable memory. It is a fully connected network with symmetric weight where no neuron is connected to itself. The neurons of this Hopfield network are updated asynchronously and in parallel and this type of networks guaranteed to converge a closest learnt pattern. The weight matrix of N neurons Hopfield network is given by a $N \times N$ matrix W with symmetric weights and the diagonal is set to zero. The state vector S at any point of time of the Hopfield network is given as:

$$S = [S_1, S_2, \dots, S_N] \tag{1}$$

Where S_i is the current output state of unit i and the local net input to i^{th} unit is given by

$$h_i = \sum_{j \neq i} w_{ij} S_j, \tag{2}$$

Where w_{ij} is the weight of connection j to i unit and the next state of the of a unit is given by

$$S' = \begin{cases} +1 & if\ h_i \geq \theta \\ -1 & if\ h_i < \theta' \end{cases} \tag{3}$$

Where θ is the threshold at unit i.

V. Learning Rules

Learning rules in Hopfield network is basically finds the set of connection weights which allow the network to produce desired response when a pattern is submitted to the network. To compare the performance of Hopfield network with Odia vowel characters, three learning rules are used in this study. These learning rules are presented in the following section.

A. Hebbian Learning Rule [22]

Hebbian rule states that when one neuron stimulates another neuron from two connected neurons in the network for firing then the connection weight between these two cells is strengthened otherwise it is weakened. In other words we can say when a weight contributes to firing a neuron, the weight is increased. The Hebbian rule for updating the weights is presented below:

- Initialize all weights to zero

$$w_{ij} = 0, for\ i = \{1,2, \dots, n\ ; j = 1,2, \dots, m\}$$

- For each input training(I)-target output(O) vector pair {I:O}, set the following activation as

$$x_i = I_i, for\ i = 1,2,..n\ \&\ y_j = O_j, for\ j = 1,2,..m$$

- Adjust the weight for $\{i = 1,2,..n\ \&\ j = 1,2,..m\}$ as follows

$$w_{ij}^{new} = w_{ij}^{old} + x_i y_j \tag{4}$$

- Now set the activation of output unit as follows

$$y_o = \begin{cases} +1 & if\ h > 0 \\ 0 & if\ h = 0 \\ -1 & if\ h < 0 \end{cases} \tag{5}$$

$$where\ h = \sum_i x_i w_{ij}$$

To store(train) K patterns $\{y_1, y_2, \dots, y_k\}$ with a N nodes Hopfield network performs the following equation by Hebbian learning rule.

For training the two associated patterns in each pair are the same which means $x_k = y_k$ with both have same dimension i.e. $m = n$. Then the weight matrix is obtained for K patterns as the sum of their outer products as:

$$W_{n\times n} = \frac{1}{N}\sum_{k=1}^{K} Y_k X_k^T = \frac{1}{N}\sum_{k=1}^{K} Y_k Y_k^T \quad (6)$$

And in the matrix form it is as follows:

$$W_{n\times n} = \frac{1}{N}\sum_{k=1}^{K} \begin{bmatrix} y_1^k \\ y_2^k \\ \vdots \\ y_N^k \end{bmatrix} [y_1^k, y_2^k \cdots y_N^k]$$

And the weight between connecting node i to j is

$$w_{ij} = \frac{1}{N}\sum_{k=1}^{K} y_i^k\, y_j^k = w_{ji} \text{ and } w_{ii} = 0\ \forall\ i \quad (7)$$

Hebbian learning rule is incremental and local which means the previous matrix can reused when we add a new pattern to the memory that to be learn.

B. Pseudo-Inverse Learning Rule [23]

Pseudo-inverse learning rule uses the pseudoinverse of the pattern matrix while Hebbian learning rule uses the pattern correlation pattern matrix. When pattern vectors are not orthogonal this learning method is more efficient and it is neither local nor incremental which means a new pattern cannot incrementally added to the network and update does not depend on either side of the connection as it calculate the inverse of the weight matrix. The procedure to apply pseudo-inverse learning rule to obtain the weight matrix of Hopfield network is explained as follows.

- Let input patterns X of vectors $\{x_1, x_2, \dots, x_n\}$ and target pattern Y of vector $\{y_1, y_2, \dots, y_n\}$ then we can write:

$$\mathrm{WX} = \mathrm{Y} \quad (8)$$

$where\ W\ is\ the\ weight\ matrix$

- If the matrix X has an inverse, then we can rewrite:

$$\mathrm{W} = \mathrm{XY}^{-1} \quad (9)$$

- Again if X is not a square matrix then no exact inverse will exist, but it has been shown that it will minimize to $\mathrm{F(W)} = \sum_{\mathrm{i=1}}^{\mathrm{n}} ||\mathrm{y_i} - \mathrm{Wx_i}||^2$ by using the pseudo-inverse matrix and is as follows:

$$\mathrm{W} = \mathrm{YX}^{+} \quad (10)$$

Where X^{+} is the Moore Penrose pseudoinverse, then the pseudoinverse of real matrix X is the unique matrix that satisfies:

$$\mathrm{XX^{+}X} = \mathrm{X}, \quad \mathrm{X^{+}XX^{+}} = \mathrm{X^{+}},$$
$$\mathrm{X^{+}X} = (\mathrm{X^{+}X})^{\mathrm{T}} \text{ and } \mathrm{XX^{+}} = (\mathrm{XX^{+}})^{\mathrm{T}}$$

- So the pseudo-inverse weight matrix can be calculated as :

$$\mathrm{W_{pinv}} = \mathrm{W}^{\mathrm{T}} * (\mathrm{W} * \mathrm{W}^{\mathrm{T}})^{-1} \quad (11)$$

$where\ W^T\ is\ the\ transpose\ of\ weight\ matrix$

- Hence the Pseudo-inverse learning rule is given by:

- Hence the Pseudo-inverse learning rule is given by:

$$w_{ij}^l = \frac{1}{N}\sum_{l=1}^{K}\sum_{m=1}^{K} y_i^l (Q^{-1}) y_j^m \quad (12)$$

$where\ Q = \frac{1}{N}\sum_{i=1}^{n} y_k^l y_k^m$

Pseudo-inverse rule works well with linearly independent patterns and stores up to N patterns in an N unit network. This rule finds a set of orthogonal vectors and calculate output weight matrix by pseudoinverse solution.

C. Storkey Learning Rule [24]

Amos Storkey in 1997 proposed a new learning rule named as Storkey learning rule is also both local and incremental and provides greater storage capacity ($\frac{N}{2*sqrt(lnN)}$) than Hebbian learning rule ($\frac{N}{2lnN}$). It is local because it only considers neurons at either side and incremental as new pattern can be added to the network without knowledge of old pattern that have been also used for training. The learning rule mathematically defined as:

$$w_{ij}^0 = 0\ for\ all\ i,j \quad (13)$$

And

$$w_{ij}^p = w_{ij}^{p-1} + \frac{1}{N}\left(y_i^p y_j^p - \frac{1}{n} y_i^p h_{ji}^p - \frac{1}{n} h_{ij}^p y_j^p \right) \quad (14)$$

Where w_{ij}^p is the weight between i and j after p^{th} pattern has been learnt and y^p is the new learning pattern.

$$h_{ij}^p = \sum_{k=1\ k\neq i,j}^{n} w_{ik}^{p-1} y_k^p\ is\ a\ form\ of\ local\ field.$$

VI. Experimental Setup and Simulation

A. Dataset

The dataset used for this paper is a combination of printed and handwritten Odia vowel characters which are collected from NIT, Rourkela database for handwritten characters and some downloaded printed vowel characters. The NIT database consists of 47 folders of different Odia characters written by 160 individuals in a specific format with A4 sheet. Each folder contains 320 handwritten samples of dimension 81×81. In this paper we consider all images of vowels contains in the first 11 folders of downloaded database. For printed Odia vowel characters we downloaded 20 different samples of each character type, so a total of 3740 samples with 340 samples per each class are collected. For consonants we consider 100 images from each of 36 consonants folders with a total of 3600 patterns for training and testing in our first implementation and divide these patterns with *k*-fold cross validation method for second part of implementation.

B. Experimental Setup

The experimental environment used in this paper is Window8.1 as OS with 16GB memory and Intel *i*7 runs in MATLAB2014a. Two experimental scenarios are presented in this paper. First experiment is to recall all the patterns with

different noise percentages from the network. These noises are explicitly introduced with five variations such as 10%, 20%, 30%, 40% and 50% while recall the stored patterns. Two sample images of Odia vowel characters are shown in Fig. 6 with different noise percentages from 10 to 50 percentage of noise .In our second experiment; we partitioned the data with k-fold cross validation method for different training and testing dataset and then recognize test patterns with various noise percentages. To reduce the processing time of the network, image sizes are resize to 30×30.

C. Implementation and Discussion

Before training the network with three mentioned learning rules, the dataset is divided into training and testing and pre-processed. Then we applied histogram equalization, binarization and finally resize them to 30×30 to speed up the training process. For the first implementation we trained the full dataset and tested all patterns for recalling whereas for the second implementation the full dataset is shuffled and then partitioned in training-testing dataset with $k\,(=5) - fold\ cross\ validation$ method. Both the experiments were implemented for three mentioned learning rules with various noise percentages. The training (storage) algorithm is discussed in algorithm-1 for three learning rules to get the weight matrix. The final weight matrix obtained from the *algorithm-1* is used to predict the class labels of test data for each learning rule and the recalling algorithm is given in *algorithm-2*. To predict a test class label hamming distance metric has been considered in this study that finds the minimum hamming distance difference between the test pattern and one of the trained patterns.

Algorithm-1: Hopfield_Training ()

Input:

Input training Dataset *X*

Learning Rule *rule*

Output:

Weight Matrix *W*

1. Shuffle dataset *X*
2. Split *X* into training and testing dataset (*Full dataset for training and testing in first experiment and k-fold data partition for second experiment)*
3. Get the training dataset *train*
4. Initialize the weight matrix to zero $W_{ij} = 0, \forall\ i, j$
5. Repeat following steps for all the training patterns
6. **if** *rule* is "Hebbian" then
7. perform matrix multiplication of first pattern P_1 with its transpose and obtained the new weight matrix by equation (6). Finally get the weight matrix of all patterns by equation (7).
8. **else if** *rule* is "Pseudo-Inverse" then
9. Obtained the weight matrix by equation (12)
10. **else if** *rule* is "Storkey" then
11. Obtained the weight matrix by equation (14)
12. **end if**
13. **end** step-5
14. Assign the diagonal of the final weight matrix to zero and normalize the weight matrix by dividing with N.

Algorithm-2: Hopfield_Recall/Recognize ()

Input:

Testing dataset *test*

Weight Matrix *W* (*As per Learning rule)*

Activation Function Y_0

Output:

Output Pattern Y_i

Model Accuracy *In %*

1. Load the weight matrix *W (As per Learning Rule)*
2. Set *correct_pat=0*
3. Set *total_test=0*
4. **for** all test patterns in *test* **do**
5. Assign a test pattern to variable $t=test_i$, *for i=1 to all test patterns*
6. **repeat**
7. equation (2)
8. equation (3)
9. **until** network is stable
10. Assign the output of step-9 to *ypredict*
11. Obtained the minimum *Hamming_distance of ypredict* with all the trained patterns and assign to Y_t
12. Assign Y_{true} = actual label of test pattern *t*
13. **if** $Y_t = Y_{true}$ then
14. *correct_pat=correct_pat + 1*
15. **end if**
16. *total_test=total_test + 1*
17. **end for**
18. *Accuracy=correct_pat / total_test*
19. **return** *Accuracy*

D. Implementation-I

In this paper, the first experiment is performed to demonstrate the recall efficiency of the full dataset (3740 Odia vowel patterns) in Hopfield network with three different learning rules such as, Hebbian learning, pseudo-inverse learning and Storkey learning for various noise percentages and a comparative result is shown in Table II. Similarly for consonants, 100 samples of each class (i.e. 3600 patterns) are considered to store into Hopfield network by using three mentioned learning rules and the recalling efficiency with various noise percentages are shown in Table VI. It has been observed that the recalling efficiency of vowels are more than consonants with different noise percentages as many of these consonants are similar rounded shapes in their upper part.

E. Implementation-II

Second experiment is presented to demonstrate the performance of recognition accuracy by splitting the dataset in train and test data with *k*-fold cross validation method for three different learning rules. In *k*-fold cross validation, the dataset is partitioned into *k* randomly chosen equal (or nearly equally) sized subsets (or folds). One fold is used to validate (testing) and other *k-1* folds are used for training in the network. This process of training and testing is repeated for k times such that each subset (fold) of data is used for testing exactly once. In this paper we portioned Odia vowels dataset into 5-fold (*k=5*) with 748 patterns are in each fold and 720 patterns in each fold for Odia consonants. The *k*-fold cross validation with *k=5* is shown in Fig. 5 and for each learning

rules we performed validation for five times named model-1, model-2, model-3, model-4 and model-5 so that each segment of data must be tested once. The recognition accuracy of each model and the average accuracy of each learning rules for Odia vowels are presented in Table III, Table IV and Table V with different noise percentages respectively whereas Table VII, Table VIII and Table IX presented here demonstrates the recognition rate for Odia consonants with Hebbian, Pseudo-inverse and Storkey learning rules respectively. Fig. 7, 8 and 9 are presented here to demonstrate the comparison of different models of k-fold validation method for Odia vowels with Hebbian, Pseudo-inverse and Storkey learning rules respectively whereas Fig. 10 demonstrates the comparison graph of all three learning rules. At the same time Fig. 11, Fig. 12 and Fig. 13 presents the comparison for Odia consonants with three learning rules respectively and Fig. 14 represents the comparison graph of three learning rules for Odia consonants.

TABLE. I. COMPARISON OF DIFFERENT PROPOSED MODELS FOR ODIA CHARACTER RECOGNITION

References	Method Used	Claimed Accuracy(In%)
Chaudhuri et al.	Feature tree classifier, Run-number based Matching	96.30
Debananda Padhi	GA with ANN	94.00
Pradeepta Sarangi et al	Hopfield Network	95.40
Pradeepta K. Sarangi	ANN	85.30
K. Roy et al.	ANN	97.69
Mamata Nayak et al.	Tesseract OCR	100
Bhagirath Kumar et al.	AMA	92.42- 97.87
Rasmi Ranjan Das et al.	BPNN and SVM	83.33(BPNN)
		93.40(SVM)
Manoj K. Mahto et al.	Quadrant Mean method	93.20
Peeta Basa Pati et al.	NN, KNN	82.33(NN)
		72.27(kNN)
T.K.Mishra et al.	BPNN with DTC & DWT	92(DTC)
		82.70(DWT)
S.D.Meher et al.	BPNN	91.24
U.Pal et al.	Quadratic Classifier	94.60
B. Majhi et al.	ANN with Gradient, Curvature features	99.30(Grad.)
		95.66(curv.)
T. K. Bhowmik et al.	HMM	90.50
K.Roy et al.	NN and Quadratic	94.81
N. Tripathy et al.	Normalization and thinning Free automatic scheme	97.74
Kalyan S Dash et al.	Hybrid Features with DLQDF & MQDF	98.50 98.40
Indugu Rushiraj et. al.	Shadow, centroid and distance based features	87.60
S. R. Panda et. al.	Template matching with Unicode mapping	97.00
Kalyan S Dash et.al.	BESAC features with Boolean matching	99.35(ISI) 98.9% (IITBBS) 99.48(ISI)
S. Satapathy et. al.	Deep neural autoencode	97.63

TABLE. II. RECALL EFFICIENCY (IN %) OF HOPFIELD NETWORK WITH THREE LEARNING RULES FOR ODIA VOWELS

Learning Rules	Recall accuracy (In %) with Different noise percentages					
	0%	10%	20%	30%	40%	50%
Hebbian	100	100	100	100	98.56	94.22
Pseudoinverse	100	100	100	100	99.50	97.50
Storkey	100	100	100	100	100	99.12

TABLE. III. RECOGNITION ACCURACY (IN %) FOR ODIA VOWELS IN HOPFIELD NETWORK WITH HEBBIAN LEARNING RULE

Model	Recognition accuracy (In %) with Different noise percentages					
	0%	10%	20%	30%	40%	50%
Model-1	91.20	91.20	90.75	88.56	88.00	86.86
Model-2	93.75	93.50	93.02	92.77	91.90	90.99
Model-3	96.66	96.66	96.20	95.83	94.77	94.20
Model-4	95.82	95.80	95.00	94.87	93.99	92.56
Model-5	91.60	91.60	91.33	90.88	90.00	89.66
Average	**93.80**	**93.75**	**93.26**	**92.58**	**91.73**	**90.85**

TABLE. IV. RECOGNITION ACCURACY (IN %) FOR ODIA VOWELS IN HOPFIELD NETWORK WITH PSEUDO-INVERSE LEARNING RULE

Model	Recognition accuracy (In %) with Different noise percentages					
	0%	10%	20%	30%	40%	50%
Model-1	92.20	91.90	91.75	90.56	89.00	88.06
Model-2	94.75	94.50	94.02	93.67	92.83	92.00
Model-3	97.87	97.66	97.20	96.33	95.87	95.20
Model-4	95.89	95.89	95.20	95.07	94.90	93.63
Model-5	92.66	92.33	92.33	92.00	91.45	91.06
Average	**94.67**	**94.45**	**94.10**	**93.52**	**92.81**	**91.99**

TABLE. V. RECOGNITION ACCURACY (IN %) FOR ODIA VOWELS IN HOPFIELD NETWORK WITH STORKEY LEARNING RULE

Model	Recognition accuracy (In %) with Different noise percentages					
	0%	10%	20%	30%	40%	50%
Model-1	92.50	92.50	92.25	91.63	91.00	89.66
Model-2	95.75	95.75	95.02	94.67	94.03	93.22
Model-3	98.50	98.50	97.90	97.33	96.77	96.00
Model-4	96.22	96.22	95.92	95.35	94.89	94.03
Model-5	92.99	92.99	92.63	92.23	92.00	91.86
Average	**95.20**	**95.20**	**94.74**	**94.24**	**93.73**	**92.95**

TABLE. VI. RECALL EFFICIENCY (IN %) OF HOPFIELD NETWORK WITH THREE LEARNING RULES FOR ODIA CONSONANTS

Learning Rules	Recall accuracy (In %) with Different noise percentages					
	0%	10%	20%	30%	40%	50%
Hebbian	100	100	100	97.33	95.56	92.22
Pseudoinverse	100	100	100	98.00	96.50	94.50
Storkey	100	100	100	99.00	98.02	96.22

TABLE. VII. RECOGNITION ACCURACY (IN %) FOR ODIA CONSONANTS IN HOPFIELD NETWORK WITH HEBBIAN LEARNING RULE

Model	Recognition accuracy (In %) with Different noise percentages					
	0%	*10%*	*20%*	*30%*	*40%*	*50%*
Model-1	90.20	90.20	89.75	88.66	87.00	85.86
Model-2	92.85	92.50	92.02	90.77	89.20	87.99
Model-3	96.22	95.89	95.20	94.83	94.00	93.20
Model-4	94.82	94.10	93.60	92.87	91.09	90.06
Model-5	92.60	91.60	91.33	90.22	89.30	88.66
Average	**93.35**	**92.85**	**92.38**	**91.47**	**90.11**	**89.15**

TABLE. VIII. RECOGNITION ACCURACY (IN %) FOR ODIA CONSONANTS IN HOPFIELD NETWORK WITH PSEUDO-INVERSE LEARNING RULE

Model	Recognition accuracy (In %) with Different noise percentages					
	0%	*10%*	*20%*	*30%*	*40%*	*50%*
Model-1	91.00	90.90	90.01	88.77	88.00	86.10
Model-2	93.22	92.89	92.00	91.02	89.89	88.66
Model-3	97.00	96.66	95.83	94.90	94.00	93.83
Model-4	95.89	95.66	95.00	94.33	93.87	92.02
Model-5	93.00	92.88	91.66	91.00	90.10	89.20
Average	**94.02**	**93.79**	**92.90**	**92.00**	**91.17**	**89.96**

TABLE. IX. RECOGNITION ACCURACY (IN %) FOR ODIA CONSONANTS IN HOPFIELD NETWORK WITH STORKEY LEARNING RULE

Model	Recognition accuracy (In %) with Different noise percentages					
	0%	*10%*	*20%*	*30%*	*40%*	*50%*
Model-1	92.06	91.02	90.63	89.20	88.83	87.50
Model-2	93.66	93.66	92.83	91.02	90.87	90.00
Model-3	98.00	98.00	97.02	96.33	95.20	94.00
Model-4	96.00	95.83	95.00	94.66	93.88	92.33
Model-5	94.00	93.22	92.83	91.33	90.67	89.83
Average	**94.74**	**94.34**	**93.66**	**92.50**	**91.89**	**90.73**

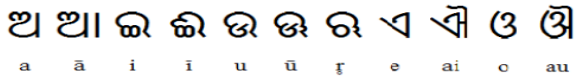

Fig. 1. Odia Vowels with English Transliteration.

Fig. 2. Odia Consonants with English Transliteration.

Fig. 3. Odia Numerals with English Numbers.

Fig. 4. Few Similar Odia Characters.

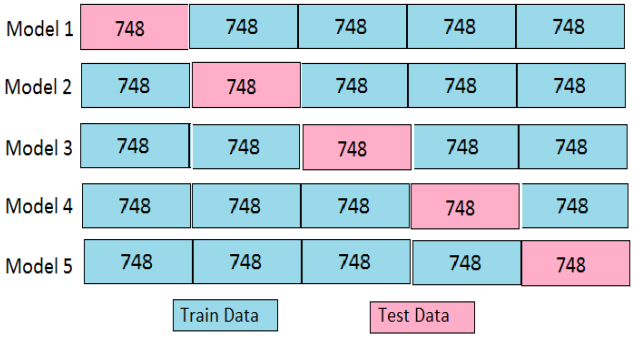

Fig. 5. K-Fold Cross Validation Data Partition for Training and Testing Data with k=5.

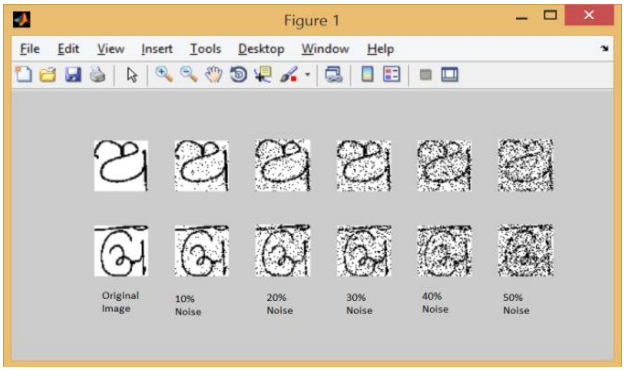

Fig. 6. Sample Images of Two Odia Characters with 10% to 50% Noise.

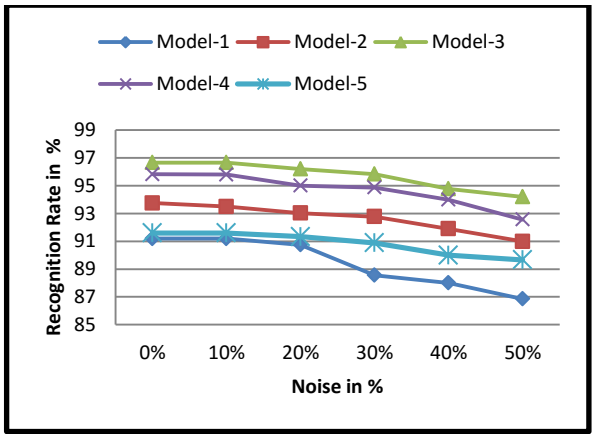

Fig. 7. Recognition Rate of Odia Vowels by K-Fold Cross Validation Model with Hebbian Learning.

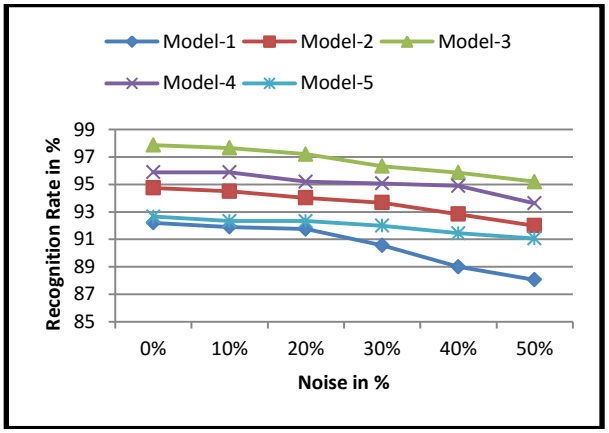

Fig. 8. Recognition Rate of Odia Vowels by K-Fold Cross Validation Model with Pseudo-Inverse Learning.

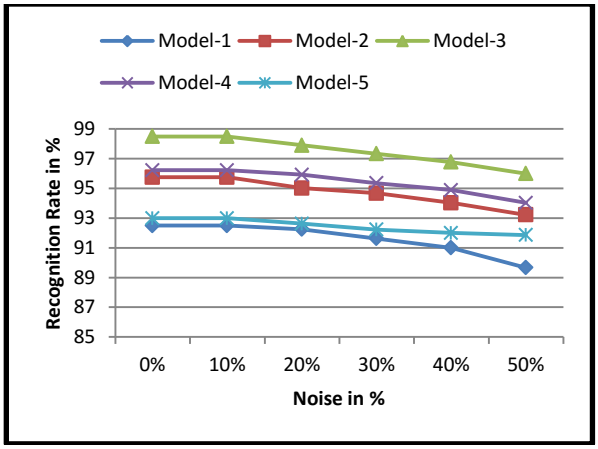

Fig. 9. Recognition Rate of Odia Vowels by K-Fold Cross Validation Model with Storkey Learning.

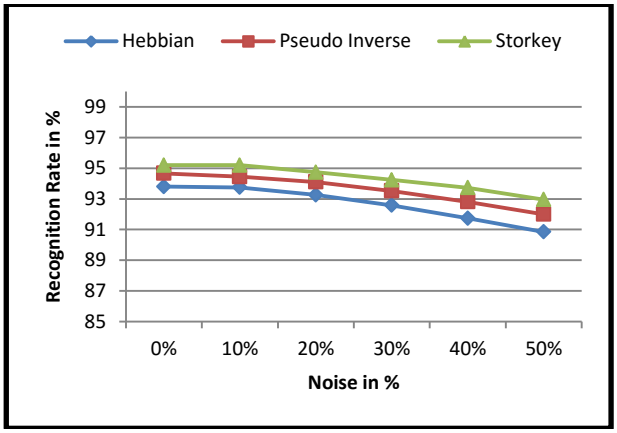

Fig. 10. Comparison Line Graph of Three Learning Rules with Hopfield.

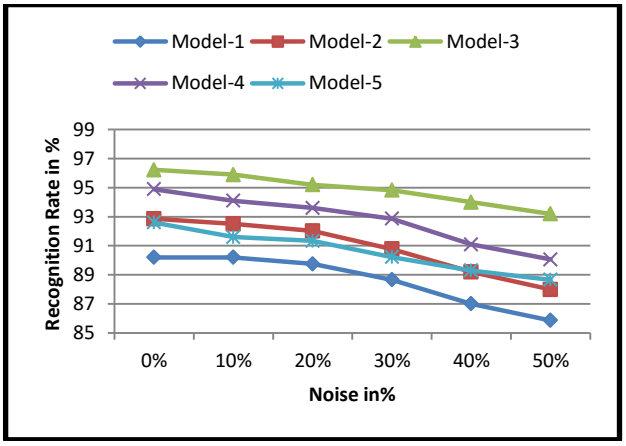

Fig. 11. Recognition Rate (Odia Consonants) of 5-Fold Validation Model for Hebbian Learning Rule.

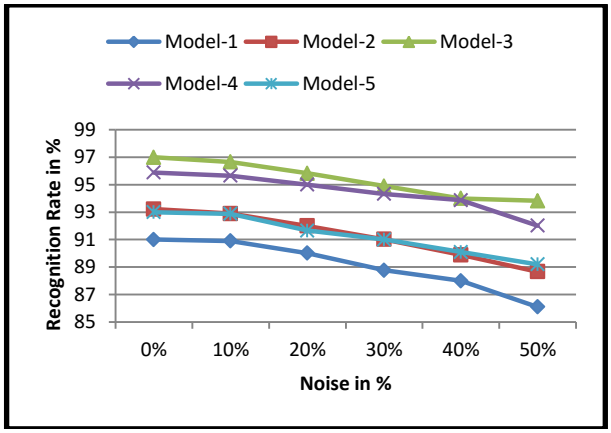

Fig. 12. Recognition Rate (Odia Consonants) of 5-Fold Validation Model for Pseudo-Inverse Learning Rule.

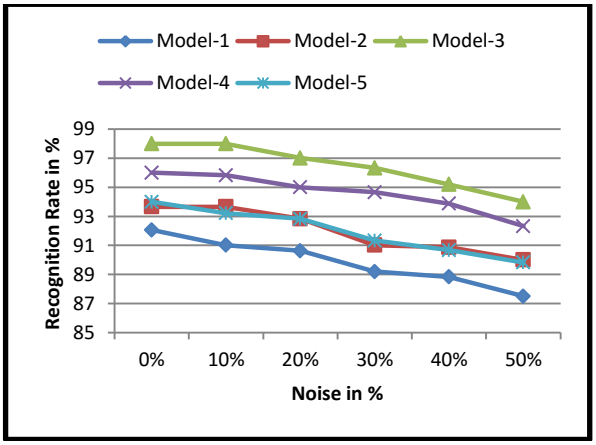

Fig. 13. Recognition Rate (Odia Consonants) of 5-Fold Validation Model for Storkey Learning Rule.

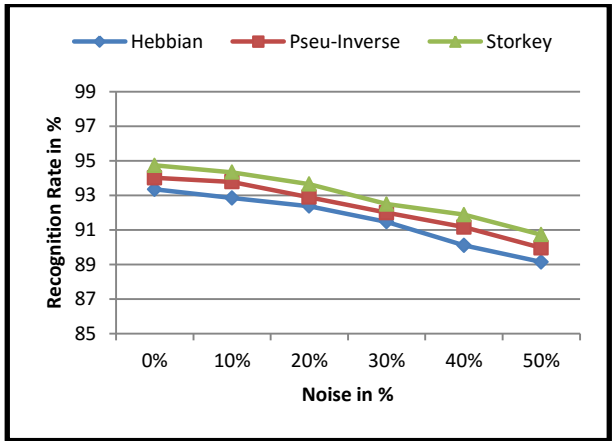

Fig. 14. Comparison Line Graph of Three Learning Rules in Hopfield Network for Odia Consonants.

VII. Conclusion

The present study demonstrated the comparison of three learning rules such as Hebbian, Pseudo-inverse and Storkey rule in Hopfield neural network for recalling and recognizing Odia handwritten and printed basic characters (vowel & consonants) in image form. Our study mainly focused on the recalling performance (stored patterns) and recognizing efficiency (new patterns) for distorted Odia characters. It is hard to compare with other models as very little work has been done so far and no such standardized test set available for Odia character recognition, hence the results we obtained are compared with claimed outputs of different methods in literature survey. The simulation results shows a better accuracy of 95.20% with Storkey learning rule than Hebbian (93.80%) and Pseudo-inverse (94.67%) learning rules without noise for Odia vowels and in case of Odia consonants Storkey rule (94.74%) also out performed than other rules. In this simulation work for recognising a pattern, we used a machine learning validation method named *k*-fold cross validation to partitioning the dataset in five sub parts and in every iteration one sub part is for testing and other four sub parts are used for training by which each pattern must be tested with various noise percentages. It has been seen that Storkey learning rule provides better accuracy in terms of recognising new test sets with noise as compared to other two learning rules.

References

[1] S. Rajasekaran and B. Deekshatulu, “Recognition of printed telugu characters”, Computer Graphics and Image Processing, vol. 6, no. 4, pp. 335–360, 1977.

[2] G. Siromoney, R. Chandrasekaran, and M. Chandrasekaran, “Computer recognition of printed tamil characters”, Pattern Recognition, vol. 10, no. 4, pp. 243–247, 1978.

[3] M. Maloo and K. Kale, “Support vector machine based gujarati numeral recognition”, International Journal on Computer Science and Engineering, vol. 3, no. 7, pp. 2595–2600, 2011.

[4] B. Dhandra, G. Mukarambi, and M. Hangarge, “Zone based features for handwritten and printed mixed kannada digits recognition”, in IJCA Proceedings on International Conference on VLSI, Communications and Instrumentation (ICVCI), no. 7, pp. 5–8, 2011.

[5] D. V. Sharma, G. S. Lehal, and P. Kathuria, “Digit extraction and recognition from machine printed Gurmukhi documents”, in Proceedings of the International Workshop on Multilingual OCR. ACM, no. 12, pp. 1–8, 2009.

[6] B. Chaudhuri, U. Pal, and M. Mitra, “Automatic recognition of printed oriya script”, Sadhana, vol. 27, no. 1, pp. 23–34, 2002.

[7] U. Pal and B. Chaudhuri, “Automatic recognition of unconstrained off-line Bangla handwritten numerals”, in Advances in Multimodal Interfaces—ICMI 2000. Springer, pp. 371–378, 2000.

[8] S. Mohanti, “ Pattern recognition in alphabets of Oriya language using Kohonen neural network”, Int. J. Pattern Recogn. Artif. Intell. Pp.1007–1015, 12 (1998).

[9] K. Roy, T. Pal, U. Pal, and F. Kimura, “Oriya handwritten numeral recognition system” , in Eighth International Conference on Document Analysis and Recognition. IEEE, pp. 770–774, 2005.

[10] T. K. Bhowmik, S. K. Parui, U. Bhattacharya, and B. Shaw, “An hmm based recognition scheme for handwritten oriya numerals”, in 9th International Conference on Information Technology. IEEE, pp. 105–110, 2006.

[11] P. K. Sarangi, A. K. Sahoo, and P. Ahmed, “Recognition of isolated handwritten oriya numerals using hopfield neural network”, International Journal of Computer Applications, vol. 40, no. 8, pp. 36–42, 2012.

[12] J. Panda, M. Panda, A. Samal, and N. Das, “Odia handwritten digit recognition using single layer perceptron”, International Journal Of Electronics And Communication Engineering & Technology, vol. 5, no. 4, pp. 80–88, 2014.

[13] K. S. Dash, N. Puhan, and G. Panda, “A hybrid feature and discriminant classifier for high accuracy handwritten odia numeral recognition”, in IEEE Region 10 Symposium, IEEE, pp. 531–535, 2014.

[14] S. Meher and D. Basa, “An intelligent scanner with handwritten odia character recognition capability” , in Fifth International Conference on Sensing Technology (ICST). IEEE, pp. 53–59, 2011.

[15] B. Kumar, N. Kumar, C. Palai, P. K. Jena, and S. Chattopadhyay, “Optical character recognition using ant miner algorithm: a case study on Oriya character recognition”, International Journal of Computer Applications, vol. 61, no. 3, pp. 17–22, 2013.

[16] T. K. Mishra, B. Majhi, P. K. Sa, and S. Panda, “Model based odia numeral recognition using fuzzy aggregated features”, Frontiers of Computer Science, vol. 8, no. 6, pp. 916–922, 2014.

[17] P. Pujari and B. Majhi, “A comparative study of classifiers on recognition of offline handwritten odia numerals” , in International Conference on Electrical, Electronics, Signals, Communication and Optimization (EESCO). IEEE, pp. 1–5, 2015.

[18] Chen, L., Fan, J. and Chen, Y. , “A High Speed Modified Hopfield Neural Network and A Design of Character Recognition System”, IEEE Chung-Yung Christian University, CH3031-2/91/0000-0308, pp.308-314, 1991.

[19] M.B. Sukhaswami, P. Seetharamulu and Arun K. Pujari “Recognition of Telugu Characters using Neural Networks”, International Journal of Neural Systems VOL. 06, NO. 03.

[20] Prateek Malik ; Rinku Dixit “Handwritten Character Recognition Using Wavelet Transform and Hopfield Network” , International Conference on Machine Intelligence and Research Advancement, IEEE, 2013.

[21] Hopfield, John J. "Neural networks and physical systems with emergent collective computational abilities.” Proceedings of the national academy of sciences, pp.2554-2558, 79.8 (1982).

[22] Hebb D “The Organization of Behavior. A Neuropsychological Theory”. Wiley, New York, NY, 1949.

[23] Personnaz, L., I. Guyon, and G. Dreyfus, “Collective Computational Properties of Neural Networks: New Learning Mechanisms”, Physical Review A, 1986. 34(5): pp. 4217-4228.

[24] Storkey, Amos. "Increasing the capacity of a Hopfield network without sacrificing functionality", Artificial Neural Networks – ICANN'97, pp.451-456, 1997.

[25] Kalyan S Dash, N. B. Puhan, Ganapati Panda, “A hybrid feature and discriminant classifier for high accuracy handwritten Odia numeral recognition”, 2014 IEEE Region 10 Symposium, pp. 531-535, 2014.

[26] Kalyan S Dash, N. B. Puhan, G. Panda, “Odia character recognition: a directional review”, Artificial Intelligence Review, Springer, Vol. 48(4), pp. 473-497, 2016.

[27] Indugu Rushiraj, Souvik Kundu and Baidyanath Ray, "Handwritten Character Recognition of Odia Script", International conference on Signal Processing, Communication, Power and Embedded System (SCOPES), pp. 764-767, 2016.

[28] Smruti Rekha Panda and Jogeswar Tripathy, " Odia Offline Typewritten Character Recognition using Template Matchingwith Unicode Mapping", International Symposiwn on Advanced Computing and Communication (ISACC), 2015.

[29] Kalyan S Dash , N.B. Puhan and Ganapati Panda, " BESAC: Binary External Symmetry Axis Constellation for unconstrained handwritten character recognition", Pattern Recognition Letters, Elsevier, Vol. 83(3), pp. 413-422, 2016.

[30] Subhashree Satapathy et. al., "Printed Odia Numerals Recognition using Stacked Autoencoder", 8th International Congress of Information and Communication Technology, ICICT 2019, Procedia, Computer Science, pp.790-797, 2019.