International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 2, Issue 1, ISSN No. 2455-2143, Pages 15-21
Published Online November-December 2016 in IJEAST (http://www.ijeast.com)

CLASSIFICATION OF HANDWRITTEN ANCIENT TAMIL SCRIPTS USING COMPLEX EXTREME LEARNING MACHINE WITH DIFFERENTIAL EVOLUTION TECHNIQUE

Dr. N.Sridevi
Department of Computer Science
Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, Tamil Nadu, India

Dr. P.Subashini
Department of Computer Science
Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, Tamil Nadu, India

***Abstract*— Tamil scripts are basically evolved from the Grantham script around the 7th century Common Era (CE). During 11th century, inscriptions in Tamil scripts came to use in the extreme southern portion of Tamil Nadu. After this, palm leaves and stone inscriptions became the prime media of writing. Therefore, there could be many literature and medicinal notes that has been written on palm leaves and inscriptions. If these inscriptions were digitized, the contents available in them can be used by various categories of people with ease and comfort. Hence the classification of 11th century handwritten ancient Tamil scripts is carried out in this research work. The main objective of this research work is to classify ancient Tamil scripts and to find an optimal solution for the classification. A new method is proposed using Differential Evolution algorithm in the Complex Extreme Learning Machine for classification. The proposed method is tested on 11th century handwritten Tamil scripts. It is observed that the proposed method achieves a high classification rate when compared with other existing methods.**

Keywords— ***Complex Extreme Learning Machine, Differential Evolution, Classification, Handwritten Characters***

I. INTRODUCTION

Translating scanned documents into machine readable form aims to paperless environment which leads to the concept of Optical Character Recognition (OCR). The main idea of an OCR is to identify and analyze a document image by dividing the document into lines and then dividing into words and then into characters. Features are extracted from these characters which are then compared with image patterns to predict the characters. The feature vector obtained from previous phase is assigned a class label and recognized using supervised and unsupervised method [1]. The data set is divided into training set and testing set for each character. In India, still a large number of the people read and write in their native language. Allowing interaction with computers in their native language leads to a better technology penetration. This creates the need for developing handwritten character classification system.
Tamil is one of the 16 major national languages spoken by the South Indian. The writing of Tamil is a combination of alphabetical and syllabic systems. Compared to other Indian language, it has a relatively small number of pure consonants and vowels [2]. The alphabets of Tamil language are very old and it is organized into a systematic way. The alphabet consists of vowels, consonants, composite letters and special letter. Tamil alphabets has 30 basic characters in which 12 are vowels (V) and 18 are consonants (C). It also has 216 composite letters (CV) and one special character (Aydham).

Target of the classification is to reduce the number of possible characters for an unknown character, from the known one [3]. Here the characters are categorized into four groups. They are vowels (V), consonants (C), composite characters (VC) and Aydham. These four classes are taken into consideration for classification of 11th century handwritten ancient Tamil scripts. There are number of classification techniques available [4-6], some of them are Support Vector Machine, K-nearest neighbor, Back Propagation Neural Network (BPNN), Hidden Markov Model, Probabilistic Neural Network (PNN) etc. All this traditional algorithms have their own merits and demerits. The traditional algorithms are far slower than required because slow gradient based learning algorithms are used and all parameters must be tuned iteratively.

Hence in order to overcome these disadvantages of traditional algorithms, this research work mainly concentrates on Extreme Learning Machine (ELM) for classification of 11th century handwritten Tamil scripts. The advantages of ELM over other algorithms are that smallest norm of weights are obtained and least square solution obtained is unique when

International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 2, Issue 1, ISSN No. 2455-2143, Pages 15-21
Published Online November-December 2016 in IJEAST (http://www.ijeast.com)

compared to others. The organization of the paper is as follows: In section 2, feature extraction is explained in brief. Classification using CELM is given briefly in section 3.Section 4 discuss about the proposed method for classification of Tamil handwritten characters. Results and discussion are discussed in section 5 and finally section 6 concludes this paper.

II. FEATURE EXTRACTION

Feature extraction is defined as the process of extraction information from the raw data which is useful for classifying the unknown type into known class. Features are classified into two groups, they are structural features like strokes, end points, etc., and statistical features which are derived from the statistical distribution of points like zoning, moments, etc., [7]. Here statistical feature (Zernike moments) along with regional features are taken for classification of handwritten ancient Tamil scripts. In feature extraction each character is represented as a feature vector, which becomes its identity [8]. Feature vectors are formed using originally extracted features and by combining different feature vectors. This is because use of several types of features still ensures an accurate description of the characters [9].

III. COMPLEX EXTREME LEARNING MACHINE

A new learning algorithm for single hidden layer feedforward neural network called the extreme learning machine is proposed by Guang-Bin Huang et.al [10]. Unlike traditional algorithms which may face difficulties in manually tuning parameters such as learning rate, learning epochs and local minima, ELM avoid such difficulties and provide good solutions. ELM algorithm is extended from real domain to the complex domain and here fully complex activation functions are used. Similar to ELM, the input weights and hidden layer biases of CELM are randomly chosen based on some continuous distribution probability and the output weights are calculated analytically. The learning speed of CELM is much faster and it also avoids local minima.

Given a series of training samples (z_i, y_i), where i = 1, 2 ... N, $z_i \in C^n$ *and* $y_i \in C^m$, the outputs of the single hidden layer feed forward network with complex activation function for these N training data is given by

$$\sum_{k=1}^{\bar{N}} \beta_k g_c(w_k.z_i + b_k) = o_i, \quad i = 1,\ldots,N \tag{1}$$

where $w_k \in C^n$ is the complex input weight vector connecting the input layer neurons to the hidden neuron, $\beta_k = [\beta_{k1}, \beta_{k2}, \ldots \beta_{km}]^T \in C^m$ is the complex output weight vector connecting the hidden neuron and the output neurons and $b_k \in C$ is the complex bias of the k[th] hidden neuron. g_c is a fully complex activation function. The above equation can be written as

$$H\beta = O \tag{2}$$

and the number of hidden neurons is usually less than the number N of training samples

$$H(w_1,\ldots,w_{\bar{N}}, z_1,\ldots,z_{\bar{N}}, b_i,\ldots b_{\bar{N}}) \tag{3}$$

$$= \begin{bmatrix} g_c(w_1.z_1 + b_1) & \ldots & g_c(w_{\bar{N}}.z_1 + b_{\bar{N}}) \\ \vdots & \ldots & \vdots \\ g_c(w_1.z_N + b_1) & \cdots & g_c(w_{\bar{N}}.z_N + b_{\bar{N}}) \end{bmatrix}_{N \times \bar{N}}$$

(4)

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\bar{N}}^T \end{bmatrix}_{\bar{N} \times m}, O = \begin{bmatrix} o_1 \\ \vdots \\ o_N^T \end{bmatrix}_{N \times m} \text{ and } Y = \begin{bmatrix} y_1^T \\ \vdots \\ y_N^T \end{bmatrix}_{N \times m} \tag{5}$$

Here, the complex matrix H is called the hidden layer output matrix. For fixed input weights and hidden layer biases, least squares solution of the linear system with minimum norm of output weight can be obtained [10]. The resulting least square solution is given by

$$\hat{\beta} = H^{\dagger} Y \tag{6}$$

where $H^{\dagger}$ is the Moore-Penrose generalized inverse of complex matrix H.

The following algorithm describes how CELM is used in classification of 11[th] century handwritten Tamil scripts.

A. Algorithm for classification

Input: Training data, Testing data and number of hidden neurons

Output: Training and testing accuracy

Step 1: Training data and testing data i.e. features of the characters are loaded

Step 2: From training and testing data sets, the class labels are extracted and it is saved as Target vector.

Step 3: Complex random numbers are generated for the input weight of size (number of Hidden neurons X number of input neurons).

Step 4: Bias of hidden neurons are randomly generated from the complex numbers.

Step 5: Hidden layer output matrix H is calculated using the "asinh" activation function for the training data.

International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 2, Issue 1, ISSN No. 2455-2143, Pages 15-21
Published Online November-December 2016 in IJEAST (http://www.ijeast.com)

Step 6: Moore Penrose inverse matrix $H^{\dagger}$ is calculated
Step 7: Output weight is calculated using Eq 6.12.
Step 8: To find the actual output of the training data, the output weight is multiplied with $H^{\dagger}$.
Step 9: Repeat steps 5 to 8 to calculate the output of testing input.
Step 10: Classification accuracy for training and testing data is calculated.
This algorithm works with complex activation function which are infinitely differentiable. Here, inverse hyperbolic function known as "arcsinh" is used as activation function. The main drawback of CELM is that since the input weights and biases are randomly generated, more number of hidden neurons is required in order to classify the handwritten Tamil scripts. So to overcome this drawback, CELM is optimized using Differential Evolution.

IV. PROPOSED METHOD

Complex Extreme Learning Machine (CELM) just randomly chooses the input weights and hidden biases, hence much of the learning time traditionally spent in tuning these parameters are saved. As the output weights are computed based on the prefixed input weights and hidden biases, there may exists a set of non-optimal or unnecessary input weights and hidden biases. However, CELM may need higher number of hidden neurons due to the random determination of the input weights and hidden biases [11]. Therefore, in order to calculate an optimal input weights and hidden biases, Differential Evolution (DE) algorithm is used.

CELM not only learns much faster than the traditional gradient-based learning algorithms but also avoids many difficulties such as stopping criteria, learning rate, learning epochs and local minima. It is found that the extreme learning machine generally require more number of hidden neurons than the traditional algorithms. Evolutionary algorithms (EAs) are widely used as global searching method for optimization. Therefore, the hybrids of EA with analytical methods provide promising results for network training. Here a new novel method is proposed by combining CELM with Differential Evolution (DE).

A. Differential Evolution

Differential Evolution (DE) has the ability and efficiency to locate global optimum over other EAs [12]. DE is a parallel direct search method. DE's basic strategy can be described as Given a set of parameter vectors

$$x_{i,G} \; i = 1,2,\cdots NP \qquad (7)$$

as a population for each generation G. DE generates new parameter vectors by adding the weighted difference between two population vectors to a third vector. This operation is called as "mutation" [13].

Mutation: For each target vector $x_{i,G}, i = 1,2,3,\cdots, NP$, a mutant vector is generated according to

$$v_{i,G+1} = x_{r1,G} + F.(x_{r2,G} - x_{r3,G}) \qquad (8)$$

with random and mutually different indexes $r_1, r_2, r_3 \in \{1,2,\cdots, NP\}$ and F is a real and constant factor $\in [0,2]$ which controls the amplification of the differential variation

$$(x_{r2,G} - x_{r3,G}) \qquad (9)$$

The mutated vector's parameters are mixed with the parameters of another predetermined vector, the target vector to form a trial vector. This mixing is referred to as "crossover".
Crossover: In order to increase the diversity of the parameter vectors, crossover is introduced. The trial vector

$$u_{i,G+1} = (u_{1i,G+1,} u_{2i,G+1,} \cdots, u_{Di,G+1}) \qquad (10)$$

is formed, where

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & if\ (randb(j) \le CR)\ or\ \ j = rnbr(i) \\ x_{ji,G} & if\ (randb(j) > CR)\ and\ \ j \ne rnbr(i) \end{cases}$$
$$j = 1,2,\cdots D \qquad (11)$$

$randb(j)$ is the jth evaluation of a random number generator. CR is the crossover constant, which has to be determined by the user. $randb(i)$ is a randomly chosen index. To decide whether or not the trial vector should become a member of generation G +1, the trail vector is compared to the target vector. This is known as "Selection".
Selection: If vector $u_{i,G+1}$ is better than $x_{i,G}$, then $x_{i,G+1}$ is set to $u_{i,G+1}$ otherwise the old value of $x_{i,G}$ is retained to $x_{i,G+1}$.

B. Algorithm for classification of 11th century handwritten scripts using DE-CELM

Input: Training data, Testing data and number of hidden neurons
Output: Training and testing accuracy
Step 1: Training and testing data files are loaded.
Step 2: Class labels from training and testing data sets are extracted and stored in target vectors.
Step 3: In order to calculate the input weights and biases for the CELM, the constants of Differential Evolution such as

International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 2, Issue 1, ISSN No. 2455-2143, Pages 15-21
Published Online November-December 2016 in IJEAST (http://www.ijeast.com)

Cross over, step size F, Number of population size are initialized to 0.5, 0.8, and 200 respectively.

Step 4: Number of population, best population member, number of function evaluations is initialized.

Step 5: To evaluate the best member after initialization, step 6 to step 10 are followed.

Step 6: Output weight and biases are calculated analytical using Moore Penrose Inverse matrix for the first member in the population.

Step 7: Step 6 is repeated for other members in the population.

Step 8: Each member is compared against one another, in order to find the population which is filled with best members.

Step 9: DE minimization is carried out.

Step 10: Vectors which are allowed to enter the new population are selected.

Step 11: From the new population, input weight and bias of hidden neurons are calculated.

Step 12: Hidden layer output matrix H is calculated using the "asinh" activation function for the training data.

Step 13: Moore Penrose inverse matrix $H^{\dagger}$ is calculated

Step 14: Output weight is calculated using $\hat{\beta} = H^{\dagger} Y$.

Step 15: To find the actual output of the training data, the output weight is multiplied with $H^{\dagger}$.

Step 16: Repeat steps 11 to 15 are repeated to calculate the output of testing input.

Step 17: Classification accuracy for training and testing data is calculated.

V. RESULTS AND DISCUSSION

In order to measure the performance of the classifiers, a sample of 3000 characters are taken from the book "Tirukkural in Ancient Scripts" written by Gift Siromoney, Govindaraju .S and Chandrasekaran .M, published in the year 1980 [14], which is the digitized form of stone inscription. Specification of feature vectors and the percentage of training and testing data used for classification of 11th century handwritten ancient Tamil scripts is shown in Table 1.To measure the performance of the optimized complex extreme learning machine, it is compared with complex extreme learning machine and extreme learning machine.

Table 1. Specification of Feature Vectors

Feature Vector	# Attributes	Training data in %	Testing data in %
Zernike moment (FV1)	7	50	50
		75	25
Regional Features(FV2)	6	50	50
		75	25
Zernike + Regional Features (FV3)	13	50	50
		75	25

The classifiers are trained using the feature vectors FV1, FV2 and FV3 respectively. The training and testing data are given in the form of text file, in which the features are given in row and column format. Each row contains information regarding to one particular character. First column contains the expected output labels for the classification of 11th century handwritten ancient Tamil scripts and rest of the columns contains features of the particular character. Figures 1, 2, 3 and 4, show the classification results obtained by training the ELM, CELM and DE-CELM classifiers using feature vector FV1.

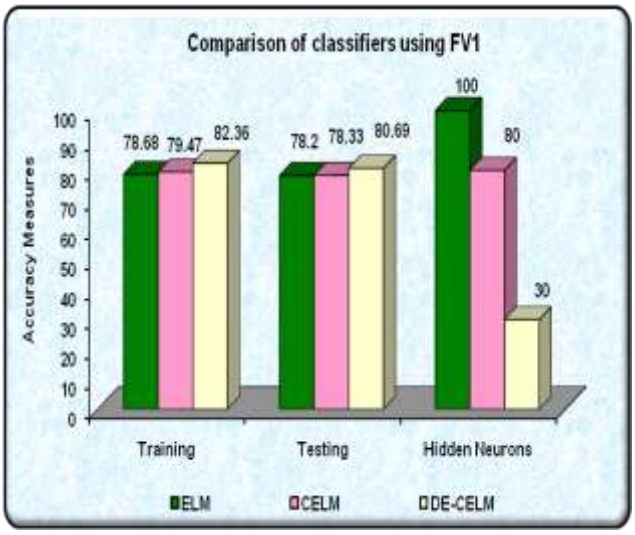

Fig 1: Comparison of Classifiers based on Accuracy by using FV1 for training set of 50%

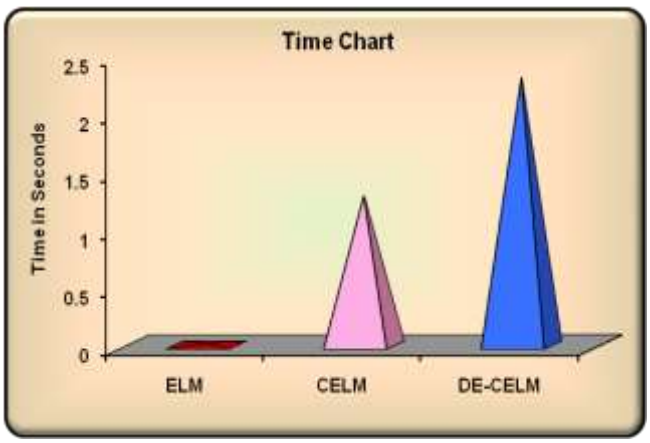

Fig 2: Comparison of classifiers based on Time by using training set of 50% for FV1

International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 2, Issue 1, ISSN No. 2455-2143, Pages 15-21
Published Online November-December 2016 in IJEAST (http://www.ijeast.com)

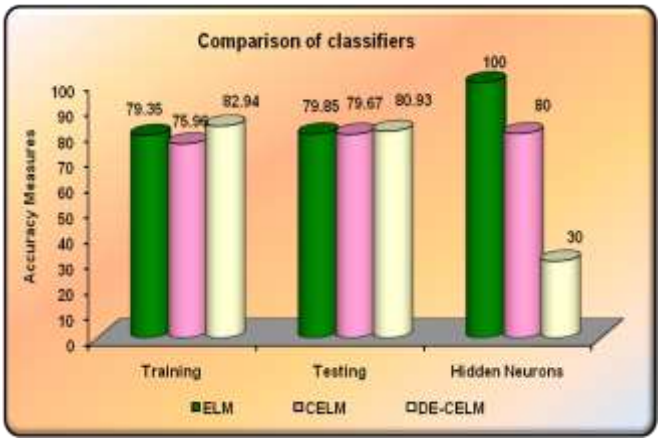

Fig 3: Comparison of Classifiers based on Accuracy by using FV1 for training set of 75%

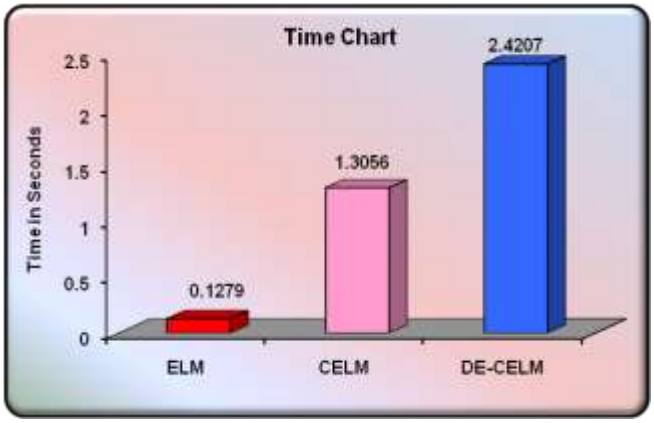

Fig 4: Comparison of classifiers based on Time by using training set of 75% for FV1

By comparing the above figures, it is observed that when the percentage of training data is increased from 50% to 75%, the training accuracy of the classifiers increases by maximum of 0.67% and testing accuracy by 1.65% but the time taken for classification has also increased with number of hidden neurons as 100, 80, and 30 respectively. Even if there is an increase in percentage of training set, the hidden neurons remains the same for the classifiers. Hence, in order to increase the classification accuracy with minimum time taken, the classifiers are trained using feature vector FV2 and their experimental results are shown in Figures 5, 6, 7 and 8.

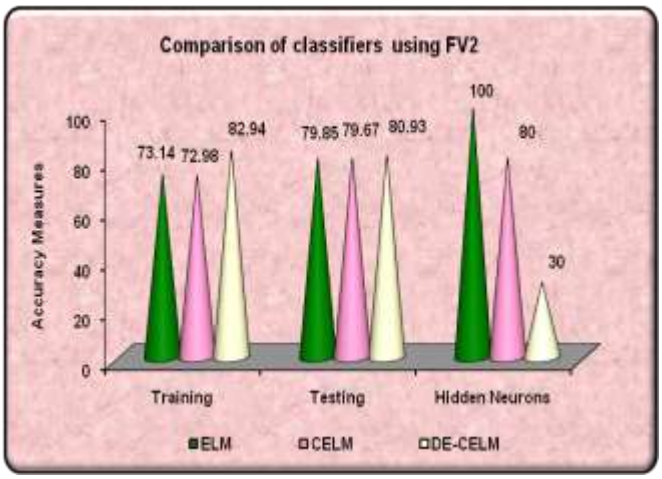

Fig 5: Comparison of Classifiers based on Accuracy by using FV2 for training set of 50%

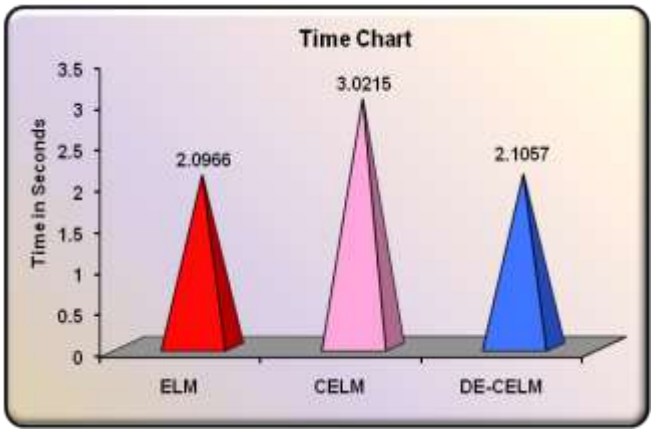

Fig 6: Comparison of classifiers based on Time by using training set of 50% for FV2

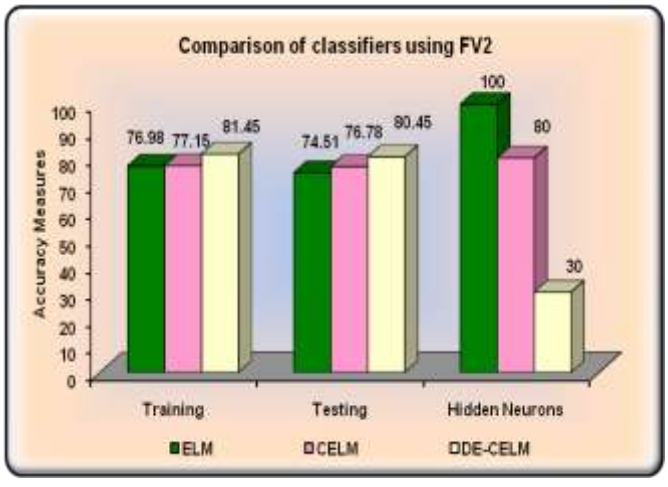

Fig 7: Comparison of Classifiers based on Accuracy by using FV2 for training set of 75%

International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 2, Issue 1, ISSN No. 2455-2143, Pages 15-21
Published Online November-December 2016 in IJEAST (http://www.ijeast.com)

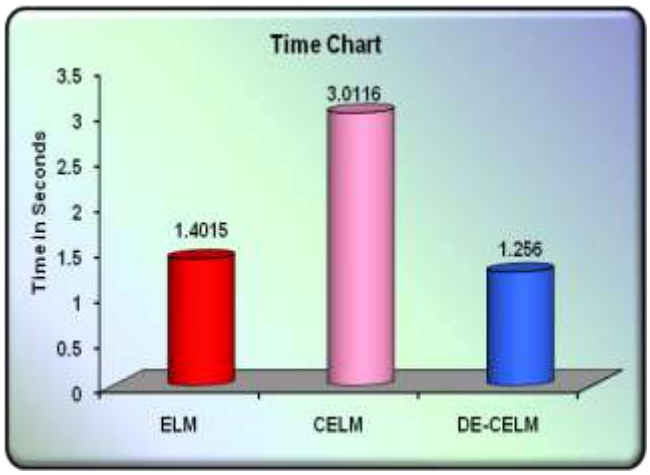

Fig 8: Comparison of classifiers based on Time by using training set of 75% for FV2

From figures 5 and 6, it has been observed that by increasing the training data from 50 to 75%, the classification accuracy of the classifiers increases on an average of 2% with decrease in time taken for classification which is represented using figures 7 and 8 respectively. Hence, to further increase the accuracy of the classifiers, each classifier is trained using the feature vector FV3 and its results are shown graphically using figures 9, 10, 11 and 12 respectively.

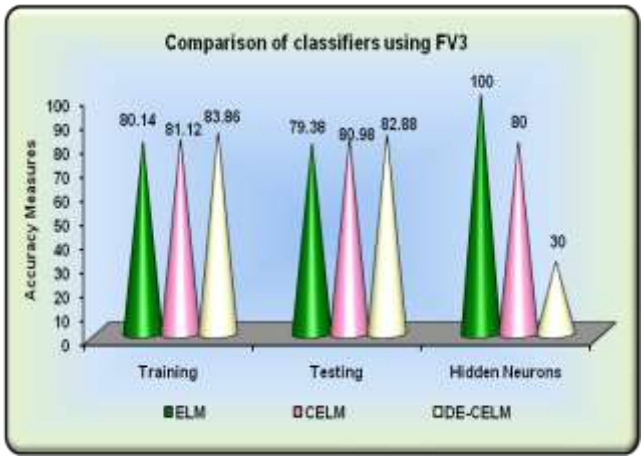

Fig 9: Comparison of Classifiers based on Accuracy by using FV3 for training set of 50%

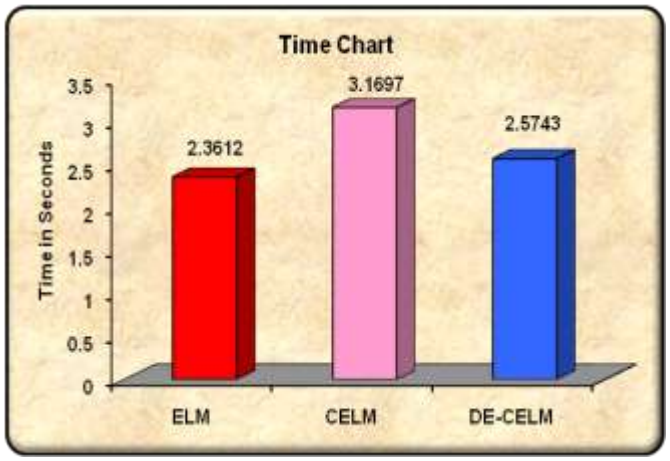

Fig 10: Comparison of classifiers based on Time by using training set of 50% for FV3

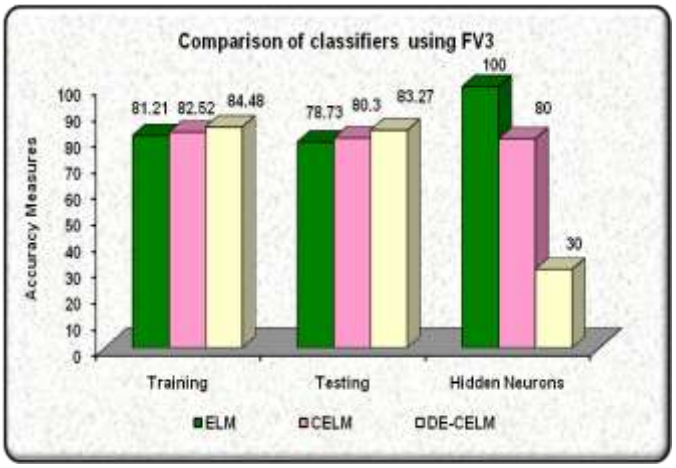

Fig 11: Comparison of Classifiers based on Accuracy by using FV3 for training set of 75%

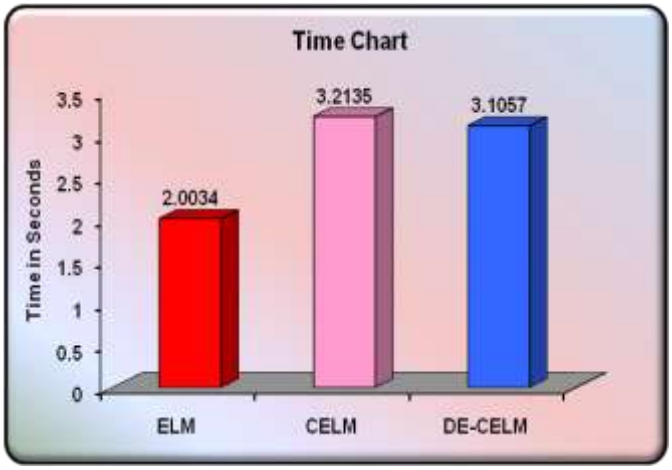

Fig 12: Comparison of classifiers based on Time for training set of 75% for FV3

When the training data of feature vector FV3 is increased, there is a minimum hike in the accuracy of the classifiers with an increase in the time taken for classification. This is because, the number of attributes in the feature vector is more when compared to vectors FV1 and FV2.

The experimental results point up that the DE-CELM minimizes the required number of hidden neurons by optimizing the input weights and biases. From the experimental results, it has been observed that the DE-CELM provides a good classification accuracy of 83.27% when compared to ELM and CELM with minimum number of hidden neurons.

VI. CONCLUSION

Optical Character Recognition is becoming the essential part of document analysis and used in many applications like postal information processing, script recognition, language identification and so on. Many techniques have been used to recognize handwritten Tamil characters but they all use traditional algorithms. The main aim of this research work is to find an optimal solution for classification of handwritten ancient Tamil scripts using Extreme Learning Machines.

ELM takes more number of hidden neurons in order to classify the characters. So to decrease the number of hidden neurons taken, Complex Extreme Learning Machine (CELM) is used. When compared with ELM, CELM gives highest classification accuracy with less number of hidden neurons but the training time taken by CELM is more. Hence, to decrease the time taken and to increase the classification accuracy with less number of hidden neurons, optimized complex extreme learning machine is proposed. Here, Differential evolution (DE) is used in CELM in order to optimize the input weights and hidden biases. Experimental results show that the DE-CELM, gives a highest rate of 83.27% with minimum number of hidden neurons and decrease in training time when compared to CELM, which is the main objective of this research work.

VII. REFERENCE

[1] Ratnashil N Khobragade , Dr. Nitin A. Koli, Mahendra S Makesar, “Analysis Of Methods For Recognition Of Devnagari Script” , International Journal Of Pure And Applied Research In Engineering And Technologyv(IJPRET),Vol.2,No.8, pp.no: 27-38,2014

[2] Indra Gandhi .R and Iyakutti .K, “An Attempt to Recognize Handwritten Tamil Character using Kohonen SOM”, International Journal of Advanced Networking and Applications, Vol.1.No.3, pp.no: 188 – 192, 2009.

[3] Sengottaiyan .N and Sureshkumar .C,“Handwritten South Indian Language Recognition using Artificial Neural Network”, International Journal of Advanced Research in Technology, Vol.1, No.1, pp.no: 87 – 90,2011.

[4] Venkatesh J and Sureshkumar C, “ Tamil Handwritten Character Recognition using Kohonon’s Self Organizing Map”, International Journal of Computer Science and Network Security, Vol.9,No.12,pp.no:156-161,2009.

[5] Jagadeesh Kannan R et al,”Off-Line Cursive Handwritten Tamil Character Recognition”, International Conference on Security Technology, IEEE Computer Society,pp.no:159-164,2008.

[6] Rajakumar S and Subbiah Bharathi V, “ 7th Century Ancient Tamil Character Recognition from Stone Inscriptions”, Indian Journal of Computer Science and Engineering, Vol.3,No.5,pp.no:673-677,2012.

[7] Heutte L et al, “A structural/statistical features based vector for handwritten character recognition”, Pattern Recognition Letters 19, pp.no: 629 – 641, 1998.

[8] Sridevi .N and Subashini .P, “Moment Based Feature Extraction for Classification of Handwritten Ancient Tamil Scripts”, International Journal of Emerging trends in Engineering and Development, Vol.7, No.2, pp.no: 106-115, 2012.

[9] Ramteke .R.J and Mehrotra .S.C, “Feature Extraction Based on Moment Invariants for Handwriting Recognition”, IEEE Conference on Cybernetics and Intelligent Systems, pp.no: 1- 6, 2006.

[10] Guang-Bin Huang, Qin-Yu Zhu and Chee-Kheong Siew, “Extreme Learning Machine: Theory and applications”, Neurocomputing 70, pp.no: 489 – 501, 2006.

[11] Qin-Yu Zhu, Qin .A.K., Suganthan .P.N. and Guang-Bin Huang,“Rapid and brief communication Evolutionary extreme learning machine”, Pattern Recognition 38, pp.no:1759 – 1763,2005.

[12] Rainer Storn and Kenneth Price, ”Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces”, Journal of Global Optimization 11, pp.no: 341 – 359,1997.

[13] Xiaohui Yuan, Yanbin Yuan, Cheng Wang,, “An Novel Neural Network Training Based on Hybrid DE and BP”,The International Federation for Information Processing,Vol.187,pp.no: 477 – 481.

[14] Gift Siromoney, Govindaraju .S and Chandrasekaran .S, “Tirukkural in Ancient Scripts”, 1980.