Research Article

Intelligent routing between capsules empowered with deep extreme machine learning technique

Naila Samar Naz[1,2] · Muhammad Adnan Khan[3] · Sagheer Abbas[1] · Atifa Ather[4] · Shazia Saqib[3]

Received: 14 November 2019 / Accepted: 9 December 2019 / Published online: 18 December 2019
© Springer Nature Switzerland AG 2019

Abstract
A container is a gathering of neurons whose action vector speaks to the instantiation parameters of a particular kind of substance, for example, an item or an article part. We utilize the length of the action vector to speak to the likelihood that the substance exists and its introduction to speak to the instantiation parameters. Compelling cases at one dimension make expectations, utilizing change networks, for the instantiation parameters of more elevated amount containers. At the point when different forecasts concur, a higher amount of container ends up dynamic. We demonstrate that a discriminatively prepared, multilayer case framework accomplishes best in class execution on Modified National Institute of Standards and Technology (MNIST) and is extensively superior to a deep learning algorithm at perceiving exceedingly covering digits. Deep learning algorithms encouraged by the function and structure of the brain. The deep extreme learning machine (DELM) approach is used to construct a compound that has the least error and highest reliability. All layers are jointly or greedily optimized, depending on the strategy. Deep extreme learning learns all the layers. This paper shows research on the expectation of the MNIST dataset using a DELM. In this article to predict digits better, we have used feedforward and backward propagation deep learning neural networks. When the results were considered, it was observed that deep extreme learning neural network has the highest accuracy rate with 70% of training (42,000 samples), 30% of test and validation (28,000 examples). When comparing the results, it was seen that the intelligent routing between capsules empowered with DELM (IRBC DELM) has the highest precision rate of 97.8%. Simulation results validate the prediction effectiveness of the proposed DELM strategy.

Keywords MNIST · Intelligent routing between capsules · Deep extreme learning · ANN · Feedforward propagation · Backward propagation

1 Introduction

Human vision overlooks insignificant subtleties by utilizing a deliberately decided grouping of obsession focuses on guaranteeing that just a little portion of the optic exhibit is ever prepared at the loftiest goals. Thoughtfulness is a poor manual for seeing the amount of our insight into a scene originates from the grouping of obsessions and the amount we gather from a solitary obsession, yet in this paper, we will expect that a solitary obsession gives us significantly more than only a solitary distinguished item and its properties. It is anticipated that our multilayer visual framework makes a parse tree-like structure on every obsession, and we disregard the issue of how these single-obsession parse trees are composed over numerous obsessions.

Parse trees are, for the most part, developed on the fly by powerfully apportioning memory. Following [1], in any

✉ Muhammad Adnan Khan, madnankhan@lgu.edu.pk | [1]School of Computer Science, NCBA&E, Lahore, Pakistan. [2]Department of Computer Science, The University of Lahore, Lahore, Pakistan. [3]Department of Computer Science, Lahore Garrison University, Lahore, Pakistan. [4]Department of Computer Science, CUI, Lahore, Pakistan.

case, we will expect that, for a solitary obsession, a parse tree is cut out of a fixed multilayer neural system like a figure is cut from a stone. Each layer will be isolated into numerous little gatherings of neurons called "cases" [1, 2], and every hub in the parse tree will compare to a functioning case. Utilizing an iterative steering process, every dynamic case will pick a container in the layer above to be its parent in the tree. For the more elevated amounts of a visual framework, this iterative procedure will take care of the issue of appointing parts to wholes.

The exercises of the neurons inside a functioning case speak to the different properties of a specific substance that is available in the picture. These properties can incorporate various kinds of instantiation parameters, for example, present (position, estimate, and introduction), disfigurement, speed, albedo, tone, surface, and so forth. One exceptionally extraordinary property is the presence of the instantiated substance in the picture. An outstanding method to speak to presence is by utilizing a different strategic unit whose yield is the likelihood that the element exists. In this paper, we investigate an intriguing elective which is to use the general length of the vector of instantiation parameters to speak to the presence of the element and to constrain the introduction of the vector to speak to the properties of component 1. We guarantee that the length of the vector yield if a case cannot surpass one by applying a nonlinearity that leaves the introduction of the vector unaltered, however, downsizes its extent.

The growth of deep learning in the field of artificial intelligence has been astounding in the last decade with about 35,800 research papers being published since 2016 [3]. With this much amount of research, it has been tough to keep up with it for many research organizations and practitioners.

Letter recognition using a neural network is one of the most widely experimented topics in Computer Science. Many research papers acknowledge the vast interest researchers have in letter recognition [4]. Modeling in a neural network also helps researchers to obtain new knowledge about design principles for letter recognition, which is essential for future research [5].

Handwritten digit recognition is an important problem in optical character recognition, and it can be used as a test case for theories of pattern recognition and machine learning algorithms. To promote research on machine learning and pattern recognition, several standard databases have emerged. The handwritten digits are preprocessed, including segmentation and normalization, so that researchers can compare recognition results of their techniques on a common basis as well as reduce the workload [6, 7].

The reasons why we choose to use this MNIST handwritten database are various. Firstly, as mentioned above, it is a standard which is a relatively simple database for fast-testing theories and algorithms. And we want to test neural networks applied to the practical problems in the real world, the handwritten digits in the MNIST database have already been preprocessed including segmentation and normalization so that it could be a good start for us spending minimal efforts on preprocessing and formatting. Besides, there are lots of researchers evaluating their theories and algorithms using MNIST, which means we can compare our results with the results from a rather comprehensive set of literature [8, 9].

The purpose of this paper is to define how deep extreme learning machine is used to resolve the problem of MNIST handwritten digit recognition. Our work is to design a neural network model and then implement it to solve the classification problem. Besides, some extra experiments have been done to test different methods that may consequently influence the performance of our model [10]. We have used a pruning technique for a deep learning model for the MNIST dataset with backpropagation. We have also looked at accuracy, and we have proposed a visualization of how data are getting trained in the final layer with that a comparison is also made with different machine learning algorithms. The final results are also compared with the previous research paper in which the technique of sensitivity was applied with one hidden layer [10, 11]. Backpropagation neural networks are supervised multilayer feedforward neural networks and commonly consist of an input layer, an output layer, and one or several hidden layers.

The rest of the paper is organized as follows. Related work is explained in Sect. 2. The proposed DELM methodology is formulated in Sect. 3. Section 4 presents the simulations and results. Finally, the research work is concluded in Sect. 5.

2 Related work

From the last thirty years, they are using hidden Markov models with Gaussian mixtures as output distributions in speech recognition. They had an emblematical limitation that was ultimately lethal, but these models were easy to learn on small computers. The one-of-n portrayals they use are exponentially wasteful contrasted, state, and intermittent neural system that utilizes dispersed portrayals. To twofold the measure of data that an HMM can recall about the string it has produced up until now, we have to square the quantity of concealed hubs. For an intermittent net, we need to twofold the number of shrouded neurons [12].

Presently those convolutional neural systems have turned into the overwhelming way to deal with article acknowledgment; it bodes well to ask whether there are any exponential wasteful aspects that may prompt their destruction. A decent competitor is trouble that the convolutional layer has in summing up to novel perspectives. The capacity to

manage interpretation is inherent; however, for different components of a relative change we need to pick between reproducing highlight locators on a network that develops exponentially with the number of measurements, and expanding the span of the named preparing set in a likewise exponential manner. Cases [13] stay away from these exponential wasteful aspects by changing over pixel powers.

Baheti et al. [14] showed detached, physically composed digits and they surveyed the execution of different picture sizes. Data tests assembled and digitized at objectives of 150 dpi and performed diverse preprocessing exercises. Pictures are resized in three particular sizes, for instance, 7×5, 14×10 and 16×16. For the portrayal, the gradient descent optimization framework is profound. For the testing purpose 3900 samples composed by hand digits, for testing used 2100 samples and remaining used for planning reason and uncovered affirmation rate of 87.29%, 88.52% and 88.76% on 7×5, 14×10 and 16×16 picture sizes and assumed that with 16×16 picture size best affirmation rate is practiced as more nuances can be gotten if picture measure increases.

In work exhibited in [15] to perceive written by hand numeral, help support vector machine (SVM) approach is proposed. Written by hand tests digitized at 300 dpi and preprocessing steps were connected to expel clamor, skew revision, upgrading picture by morphological activities, measure standardization. All pictures were standardized as 40×40 for which closest neighbor addition procedure is connected, at that point picture is changed over to parallel, and at that point picture is skeletonized to set it up for the next period of acknowledgment. For highlight, the extraction picture is partitioned into different size boxes and dependent on that four capabilities are gotten from which arrangement is done utilizing SVM. Creators have finished up to having 90.55% of normal acknowledgment rate in which "0" is having the most astounding acknowledgment rate with propounding approach.

Analyzed two classifiers to be specific K-nearest neighbor classifier (KNN) and principal component analysis (PCA) to perceive disconnected transcribed Digits, Kamal Moro [16]. Amazon machine image (AMI) display is utilized for highlight extraction and reasoned that KNN is a preferred classifier over PCA as the acknowledgment rate revealed is 90.04% against 84.1% individually. In [17] creators have broadened correlation with principal component analysis (PCA), support vector machine (SVM), K-nearest neighbor (KNN) and Gaussian dissemination work alongside Amazon Machine Image (AMI) as highlight extraction strategy and accomplished acknowledgment rate for distinguishing digits as 84.1%, 92.28%, 90.04% and 87.2% separately with end that SVM is better classifier.

LeCun et al. [18] have accomplished 80.5% of in general execution and in their examination work to perceive transcribed digits utilizing neural system classifiers.

Ciresan et al. [19] have gathered 300 examples of 300dpi with the beginning size of every numeral as 90×90. The creator at that point balanced the difference by CLAHE, for example, differentiate restricted versatile histogram balance calculation thinking about 8×8 tiles and 0.01 as difference upgrade consistent. The limits are then smoothed out by the middle channel of 3×3 neighborhoods. The picture is then recreated to the span of 16×16 pixels utilizing closest neighbor interjection. With the distinction of 2θ each up to 10°, five additional examples for every digit are made in both clockwise and anticlockwise bearings. A gradient descent optimization is utilized for digits order with 278 arrangements of different digits. Out of these 278 sets, 11 sets were made by a standard text style. The achievement rate for standard textual styles as 71.82%, from the 265 sets the writer recorded for written by hand preparing sets as 91.0% and 81.5% were recorded for testing sets as a score.

Capsules make a substantial illustrative suspicion: At every area in the picture, there is at most one example of the kind of element that a case speaks to. This presumption, which was inspired by the perceptual wonder called "swarming" [20], wipes out the coupling issue [21] and enables a case to utilize a conveyed portrayal (its action vector) to encode the instantiation parameters of the substance of that type at a given area. This disseminated portrayal is exponentially more productive than encoding the instantiation parameters by initiating a point on a high-dimensional network, and with the privilege circulated portrayal, containers would then be able to exploit the way that spatial connections can be demonstrated by lattice increases.

3 Methodology

Figure 1 shows the proposed IRBC DELM system model. The proposed system consists of 3 layers named Data Acquisition layer, Preprocessing Layer & Application Layer. In the data acquisition layer, data is collected from the source and through a wireless link it stores in a database that is called raw data. After collecting the raw data preprocessing is performed using the moving average method for prediction of missing values and normalization to mitigating the noise. After preprocessing the Application layer activated, the application layer further divided into 2 sublayers: Prediction & Performance evaluation layer. In the Prediction layer, a deep extreme learning machine is used. And performance evaluation layer evaluates the prediction in terms of accuracy and miss rate.

3.1 Deep extreme learning machine

The deep extreme machine training method is a well-known technique used in various areas. The traditional

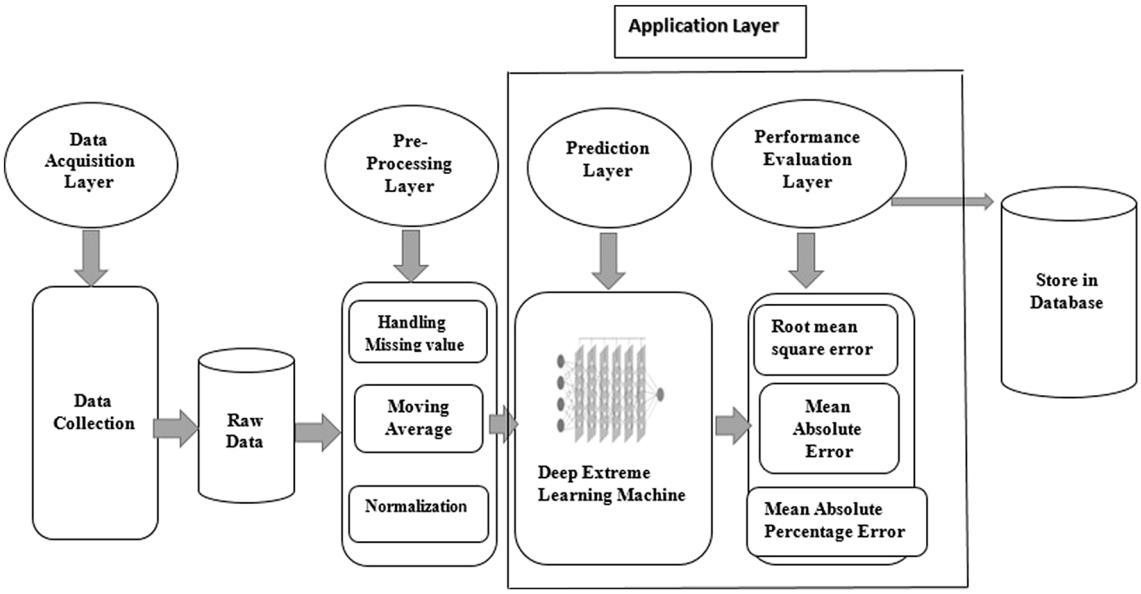

Fig. 1 Detailed diagram for the proposed prediction approach for the IRBC DELM system

ANN algorithms require more samples and slow learning times and can overfit the learning model [22]. The idea of DELM was first specified by Ref. [23]. The DELM is used widely in various areas for classification and regression purposes because DELM learns fast, and it is efficient in the cost of computational complexity. There are three layers included in the DELM model: the input layer, multiple hidden layers, and an output layer. The structural model of a DELM is shown in Fig. 2, where *im* represents input layer nodes, *e* represents hidden layer nodes, and *DP* indicates output layer nodes.

Take a training sample at first $[A, B]=\cdot[a_{ki}, b_{ki}]\cdot(i=1, 2,\ldots, Z)$, and input sample $A=[a_{k1}\ a_{k2}\ a_{k3}\ldots a_{kz}]$ and a targeted matrix $B=[b_{11}\ b_{12}\ b_{13}\ldots b_{1z}]$ were composed of samples from training, the matrices *X* and *Y* can then be represented, respectively, as in Eqs. (1) and (2), where the dimensions *a* and *b* are

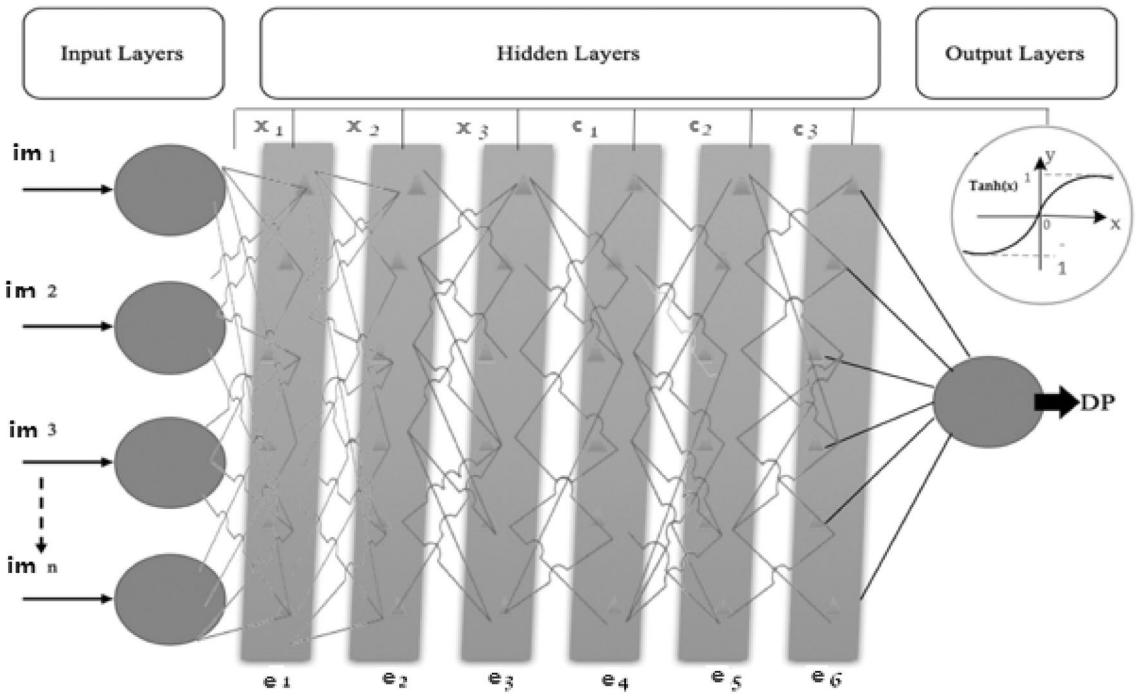

Fig. 2 Structural diagram of deep extreme learning machine

the input matrix and output matrix. The ELM will then adjust weights arbitrarily among input and the hidden layers where v_{k1} is the input weight kth node and hidden layer node of *l*th as shown in Eq. (3).

$$\mathrm{A} = \begin{bmatrix} a11 & a12 \cdots & a1o \\ a21 & a22 \ldots & a2o \\ a31 & a32 \ldots & a3o \\ \vdots & \vdots & \vdots \\ a_{p1} & a_{p2} & a_{po} \end{bmatrix} \tag{1}$$

$$\mathrm{B} = \begin{bmatrix} b11 & b12 \cdots & b1o \\ b21 & b22 \ldots & b2o \\ b31 & b32 \ldots & b3o \\ \vdots & \vdots & \vdots \\ b_{p1} & b_{p2} & b_{po} \end{bmatrix} \tag{2}$$

$$\mathrm{C} = \begin{bmatrix} c11 & c12 \cdots & c1o \\ c21 & c22 \ldots & c2o \\ c31 & c32 \ldots & c3o \\ \vdots & \vdots & \vdots \\ c_{p1} & c_{p2} & c_{po} \end{bmatrix} \tag{3}$$

$$D = \begin{bmatrix} d11 & d12 \cdots & d1o \\ d21 & d22 \ldots & d2o \\ d31 & d32 \ldots & d3o \\ \vdots & \vdots & \vdots \\ d_{p1} & d_{p2} & d_{po} \end{bmatrix} \tag{4}$$

Next, the ELM has randomly selected the biases of the hidden layer nodes, as in Eq. (5). Extreme learning machine also preferred an *h*(*A*) function that was the network activation function. The resulting matrix is shown in Eq. (6). Equation (7) represents the column vector of the resulting matrix *T*.

$$B = \left[b1, b2, b3 \ldots bp\right]T \tag{5}$$

$$E = \left[e1, e2, e3 \ldots ef\right]raf \tag{6}$$

$$\mathrm{e1} = \begin{bmatrix} e_{1j} \\ e_{2j} \\ T_{3j} \\ \vdots \\ T_{rj} \end{bmatrix} = \begin{bmatrix} \sum_{l=1}^{a} O_{k1} h\left(c_k a_l + b_k\right) \\ \sum_{l=1}^{a} O_{k2} h\left(c_k a_l + b_k\right) \\ \sum_{l=1}^{a} O_{k3} h\left(c_k a_l + b_k\right) \\ \vdots \\ \sum_{l=1}^{a} O_{kr} h\left(c_k a_l + b_k\right) \end{bmatrix} \quad (l = 1, 2, 3, \ldots, Q) \tag{7}$$

Then we can obtain Eq. (8) about Eqs. (6) And (7). The outcome of the hidden layer is *P*, and transposition of O as O′ and values of weight matrix b are calculated in Eq. (9) with the least square method.

$$Pd = O' \tag{8}$$

$$d = P^{+}O' \tag{9}$$

The *d* regularization term was used to increase the network's overall stability [24]. Deep learning emerges, and nowadays, it is a very famous subject for researchers. A system with a minimum of four layers with inputs/outputs fulfilling the needs of a deep learning system. The neurons of every layer are trained in a deep neural network on a diverse set of parameters with the result of the previous layer. This allows extensive datasets to be processed by the deep learning networks. Deep learning captured numerous researchers' attention because it is very effective in solving real-world issues. DELM is used in our proposed work to capture both ELM advantages and deep learning. DELM's model comprised of a single input layer with four neurons, six hidden layers, with ten neurons, each hidden layer, and Fig. 2 shows one layer of output with one neuron. The test and error method for selecting the number of nodes from hidden layers was used because of the lack of any special mechanism for specifying hidden layer neurons. The second hidden layer output is achievable as:

$$P_1 = Od^{+} \tag{10}$$

Where d^{+} is the general inverse of matrix *d*. Thus, in the hidden layer two values can be easily attained through Eq. (9).

$$h\left(C_1 P + B_1\right) = P_1 \tag{11}$$

In Eq. (11), the parameters V_1, *M*, Y_1, and M1 represent the first two hidden layers' weight matrix, preference of the first neurons in the hidden level, the first hidden layer assessed output, and the second hidden layer estimated output, respectively.

$$C_{PE} = h^{-1}\left(P_1\right)P_E^{+} \tag{12}$$

P_E^{+} is the inverse of P_E, and to compute Eq. (3), the active function *h*(*A*) is used. Hence, the required outcome of the second hidden layer is revised as follows by indicating the correct *h*(*A*) activation function:

$$\begin{aligned} &P_2 = h\left(C_{PE}P_E\right) \text{ where } C_{PE}P_E = Nete_2 \\ &P_2 = h\left(Nete_2\right) \end{aligned} \tag{13}$$

Updating the weighted matrix *d* among the second layer and third layer as per Eq. (14), P_2^{+} is the inverse of P_2. The estimated layer three results are thus shown as Eq. (15).

$$d_{new} = P_2^+ O \tag{14}$$

$$\mathrm{P} = Od^+new \tag{15}$$

$N\mu^+new$ is the inverse of the weight matrix μ_{new}. The DELM then sets the matrix $V_{ME1} = [Y_2, V_2]$. Equations (8) and (9) allow the third layer output to be achieved.

$$P_3 = h^{-1}\left(P_2 C_2 + B_2\right) = h\left(Nete_3\right) \tag{16}$$

$$C_{PE1} = \left(d^{-1}\left(P_3\right)P_{E1}^+\right) \tag{17}$$

The M_2 indicates the required outcome and the hidden layer 2 in Eq. (16), the weight is represented among the third hidden layer and third hidden layer by C_2, and the B_2 has three neurons of the hidden layer. P_{E1}^+ is the inverse of $E1$, and $g^{-1}(A)$ indicates the opposite of g(A)? In Eq. (18), the logistical sigmoid function was assumed. Equation (19) is calculated as the third hidden level output.

$$h(a) = \frac{1}{1 + e^{-a}}$$
$$P_3 = h\left(C_{PE1} P_{E1}\right) \text{ where } C_{PE1} P_{E1} = Nete_3 \tag{18}$$

$$P_3 = h\left(Nete_3\right) \tag{19}$$

In the end, the resulting weighted matrix is calculated as in Eq. (20) concerning the hidden level 3 and the final layer result. Equation (21) represents the likely outcome of hidden layer 3.

$$d_{new} = P_4^T\left(\frac{1}{\lambda} + P_4^T P_4\right)^{-1} O \tag{20}$$

$$P_4 = Od_{new}^+ \tag{21}$$

Nd_{new}^+ is the transposed of the weight matrix μ_{new}. The DELM afterward creates the matrix $C_{PE2} = [B_3, C_3]$. Equations (13) and (22) can be used to achieve the output of the fourth layer.

$$P_4 = h^{-1}\left(P_3 C_3 + B_3\right) = h\left(Nete4_1\right) \tag{22}$$

$$C_{PE2} = d^{-1}\left(\left(P_4\right)P_{E2}^+\right) \tag{23}$$

The desired output is represented in Eq. (9) by P_3, C_3 is the weight between the third and fourth hidden layers, and the bias of the neurons on the third hidden layer is B_3. P_{E1}^+ is the inverse of P_{E1}, and h−1(x) is the opposite of the h(x) activation feature. The sigmoidal logistic function is assumed. In Eq. (24), following is the measuring of 3rd and 4th hidden layer values:

$$P_4 = h\left(Nete4_2\right) \tag{24}$$

Finally, in Eq. (25), the output weight matrix from the fourth to the output layers is calculated. Equation (26) indicates the assessed outcome of the fifth layer. Equation (27) shows the required output of the DELM system.

$$d_{new} = P_5^T\left(\frac{1}{\lambda} + P_5^T P_5\right)^{-1} O \tag{25}$$

$$P_5 = Od_{new}^+ \tag{26}$$

$$f(a) = P_5 \beta_{new} \tag{27}$$

We have examined the procedure of calculating the four hidden layers of the DELM system. The theory of cycle was used to show the DELM computing procedure. Equations (16, 20) can be recalculated to record the parameters of every hidden layer and ultimately the last result of the DELM network. If hidden layers are increased, the same process can be reused and performed in the same way.

$$op = \frac{1}{1 + e^{-Netej}} \text{ where } j = 1, 2, 3 \ldots r \tag{28}$$

3.2 Backward propagation

The backpropagation algorithm incorporates weight initialization, feedforward, back-error propagation, and weight, and bias update is subject to distinctive developments. An activation function like $f(x) = \text{Sigmoid}(x)$ exists on each neuron in the hidden layer. The sigmoid input function and the hidden layer of the DELM can be composed in this way:

$$E = \frac{1}{2}\sum_j \left(\tau_j - op_j\right)^2 \tag{29}$$

τ_j = Desired output; $\mathrm{op_j}$ = calculated output

Equation (29) shows the backpropagation of error, which can be calculated by the sum of the square of the desired output from the calculated output divided by 2.

To reduce the overall error, the adjustment of weight is required. Equation (31) shows the rate of change in weight for the output layer.

$$\Delta \mathrm{P}_{ij}^{I=6} \propto -\frac{\partial \mathrm{E}}{\partial \mathrm{P}^{\mathrm{I}=6}} \quad \begin{matrix} \text{where } i = 1, 2, \ldots 10(Neurons) \\ \mathrm{j} = \mathrm{o/p\ Layers} \end{matrix} \tag{30}$$

$$\Delta \mathrm{P}_{ij}^{I=6} = -\mathrm{const}\frac{\partial \mathrm{E}}{\partial \mathrm{P}^{\mathrm{I}=6}} \tag{31}$$

Writing Eq. (31) by using the chain rule method.

$$\Delta P_{ij}^{l=6} = -\text{const}\frac{\partial E}{\partial op_j^l} \times \frac{\partial op_j^l}{\partial NeteP_j^l} \times \frac{\partial NeteP_j^l}{\partial P_{i,j}^l} \tag{32}$$

The value of change weight can be achieved after substituting the values in Eq. (32) as shown in Eq. (33).

$$\Delta P_{ij}^{l=6} = \text{const}(\tau j - op_j) \times \left(op_j^l\left(1 - op_j^l\right) \times op_j^l\right)$$

From op to P_6.

$$\Delta P_{ij}^{l=6} = \text{const}\xi_j op_j^l \tag{33}$$

The calculation to determine appropriate weight change to the hidden weight is shown in the following procedure. It is more complex because the weighted connection can lead to errors at all nodes.

From P_6 to P_1 or P_n, where l = 5,4,3,2,1

$$\Delta P_{i,c}^{l} \propto -\left[\sum_j \frac{\partial E}{\partial op_j^l} \times \frac{\partial op_j^l}{\partial NeteP_j^l} \times \frac{\partial P_j^l}{\partial op_c^l}\right] \times \frac{\partial op_c^l}{\partial NeteP_c^l} \times \frac{\partial NeteP_c^l}{\partial P_{i,c}^l}$$

$$\Delta P_{i,c}^{l} = -\boldsymbol{E}\left[\sum_j \frac{\partial E}{\partial op_j^l} \times \frac{\partial op_j^l}{\partial NeteP_j^l} \times \frac{\partial P_c^l}{\partial op_c^l}\right] \times \frac{\partial op_c^l}{\partial NeteP_c^l} \times \frac{\partial NeteP_c^l}{\partial M_{i,c}^l}$$

$$\Delta P_{i,c}^{l} = \boldsymbol{E}\left[\sum_j \left(\tau j - op_j^l\right) \times op_c^l\left(1 - op_j^l\right) \times \left(P_{c,j}\right)\right] \times op_j^l\left(1 - op_j^l\right) \times \alpha_{i,c}$$

$$\Delta P_{i,c}^{l} = \boldsymbol{E}\left[\sum_j \left(\tau j - op_j^l\right) \times op_c^l\left(1 - op_j^l\right) \times \left(P_{c,j}\right)\right] \times op_j^l\left(1 - op_j^l\right) \times \alpha_{i,c}$$

$$\Delta P_{i,c}^{l} = \boldsymbol{E}\left[\sum_j \left(\tau j - op_j^l\right) \times op_c^l\left(1 - op_j^l\right) \times \left(P_{c,j}\right)\right] \times op_c^l\left(1 - op_c^l\right) \times \alpha_{i,c}$$

$$\Delta P_{i,c}^{l} = \boldsymbol{E}\left[\sum_j \xi_j(M_{c,j}^l)\right] \times op_c^l\left(1 - op_j^l\right) \times \alpha_{i,c}$$

$$\Delta P_{i,c}^{l} = \boldsymbol{E}\xi_c\alpha_{i,c}$$

where

$$\xi_c = \left[\sum_j \xi_j(P_{c,j}^l)\right] \times op_c^l\left(1 - op_c^l\right)$$

The process of upgrading the weight and bias among the output and the hidden layer is shown in Eq. (34).

$$\boldsymbol{P}_{ij}^{l=6}(\boldsymbol{t}) = \boldsymbol{P}_{ij}^{l=6}(\boldsymbol{t}) + \lambda\Delta\boldsymbol{P}_{ij}^{l=6} \tag{34}$$

Equation (9) shows how to update the weight and bias among the input and the hidden layer.

$$\boldsymbol{P}_{i,c}^{l}(\boldsymbol{t}) = \boldsymbol{P}_{i,c}^{l}(\boldsymbol{t}+1) + \lambda\Delta\boldsymbol{P}_{ij}^{l} \tag{35}$$

4 Simulations and results

In the proposed article, the deep extreme learning algorithm has been applied to data [10]. In this article, IRBC DELM was used to train and fit 24,000 sets of data. This data arbitrarily divides into 60% of training (42,000 samples); 40% of data are used for validation and testing (28,000 samples). Data were previously processed to remove data abnormalities and free the data from error. IRBC DELM has attempted to discover the finest configuration model for MNIST prediction in different hidden layers, hidden neurons and combinations of activation functions [25]. Therefore, we have tried the same number of neurons and different types of active functions in hidden layers. In this work, we used the proposed IRBC DELM for prediction to properly test the effectiveness of this algorithm. To measure the performance of this DELM algorithm together with the counterpart algorithms, we used different statistical measures.

$$\text{Misrate} = {}^{O_1}/_{T_0} + {}^{O_0}/_{T_1} \tag{36}$$

$$\text{Accuracy} = \frac{\left({}^{O_0}/_{T_0} + {}^{O_1}/_{T_1}\right)}{T_0 + T_1} \tag{37}$$

For the occupancy data set [23], the DELM approach was used, and the results obtained can be seen in Tables 1 and 2. As can be seen from Table 1, we take 60% of data (42,000 samples) for training from the dataset [10]. We

Table 1 Training accuracy of the proposed IRBC DELM system with varying hidden layers during the prediction of digits in training

N = 42,000 (no. of samples)	Result (output) (O_Z, O_{NZ})	
	$O_{Z\ (Zero)}$	$O_{NZ\ (Not\text{-}Zero)}$
Input		
$Z_{=4132\ (Zero)}$	4050	82
$NZ_{=37,868\ (Not\text{-}Zero)}$	61	37,806

Table 2 Testing accuracy of the proposed IRBC DELM system with varying hidden layers during the prediction of digits in testing

N = 28,000 (no. of samples)	Result (output) (O_O, O_{NO})	
	$O_{O\ (One)}$	$O_{NO\ (Not\text{-}One)}$
Input		
$O_{=4650\ (One)}$	4613	37
$NO_{=23,350\ (Not\text{-}One)}$	49	23,301

have two expected outputs Zero and Not-Zero. Zero results demonstrated that the digit has predicted, and Not-Zero result demonstrated that digit has not predicted. In Table 1, it is shown that on 42,000 data samples, we have expected the output of 4132 Zero-digit samples and 37,868 Not-Zero samples. After applying training on 42,000 data samples, we get the result of 4050 samples zero output and 37,806 samples of nonzero output. After comparing with expected output and outcome that got after applying the proposed approach, it can be shown in Table 1 that the result of our proposed approach during training is 97% accurate, and the miss rate is 3%. In this proposed method, we get 4050 zero-digit samples output while the expected output is 4132 zero-digit samples and 37,806 not-zero samples, while the predicted output is 37,868 positive samples.

As can be seen from Table 2, we take 40% of data (28,000 samples) for testing and validation from the dataset [10]. In Table 2, it is shown that on 28,000 data samples, we have expected the output of 4650 one-digit samples and 23,350 not-one samples. After applying training on 28,000 data samples, we get the result of 4613 samples of digit one output and 23,301 samples of not-one output. After comparing with expected output and result that got after applying the proposed approach, it can be shown in Table 2 that the result of our proposed approach during testing and validation is 90.2% accurate, and the miss rate is 9.8%. In this proposed method, we get 4613 digit one sample output while the expected output is 4650 digit one samples and 23,301 not-one samples while the expected output is 23,350 not-one samples.

It is observed that the individual performance in training of the proposed method (DELM) of digit 0 was 98%

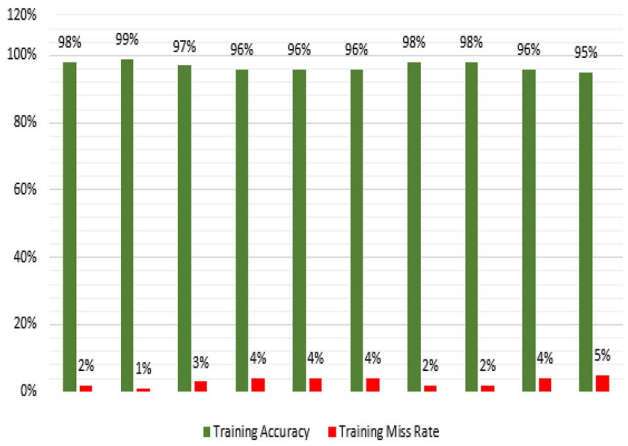

Fig. 3 Individual performance comparison of the proposed IRBC DELM system (0–9 digits) with varying hidden layers during the prediction of MNIST dataset in testing and training

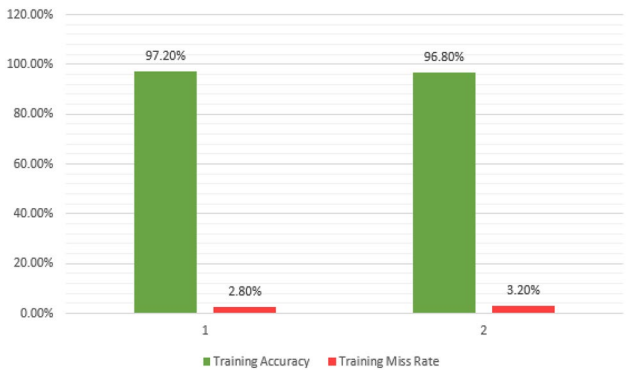

Fig. 4 Overall performance comparison of the proposed IRBC DELM system (0–9 digits) with varying hidden layers during the prediction of MNIST dataset in testing and training

accurate as shown in Fig. 3, while the miss rate of training is 2%. Digit 1 was 99% accurate, as shown in Fig. 3, while the miss rate of training is 1%. Digit 2 was 97% accurate, as shown in Fig. 3, while the miss rate of training is 3% and so on.

It is observed that the overall performance in training of the proposed method (DELM) was 97.2% accurate, as shown in Fig. 4, while the miss rate of training is 2.8%. But in testing and validation, the overall performance of the proposed method (DELM) was 96.8% accurate, as shown in Fig. 4, while the miss rate of training is 3.2%. It is shown in Fig. 4 that in training phase results accuracy increases with the minimum miss rate as compared to the testing and validation phase.

According to Table 3, DELM utilized the MNIST data set [25] consisting of 70,000 data samples; the mean square error in each round is increasing. Table 3 shows that during training the mean square error of DELM methodology

Table 3 Proposed IRBC DELM system performance in terms of MSE, accuracy and miss rate

	MSE	Accuracy (%)	Miss Rate (%)
Training	$7.23 * 10^{-3.5}$	97.2	2.8
Testing	$6.56 * 10^{-3}$	96.8	3.2

Table 4 MSE comparison of the proposed IRBC DELM system with the state of the art [10]

	Accuracy (%)	MSE (%)
In CapsNet [10]	79	21
CNN **(**traditional convolutional neural network**)** [10]	66	24
DELM (proposed)	96.8	3.2

is $7.23 * 10^{-3.5}$ and during testing the mean square error is $6.56 * 10^{-3}$, respectively. It is shown in Table 3 that during training, accuracy of the proposed DELM approach is 97.2% and miss rate is 2.8%. During testing, accuracy of the proposed DELM approach is 96.8% and miss rate is 3.2%.

According to Table 4, CNN [10] utilized the MNIST data set [24] consisting of 70,000 data samples. In the CapsNet [10] approach, there is an accuracy rate of 79%, and miss rate is 21%. Table 4 shows that during training, the accuracy is 66%, and the miss rate is 24% of CNN [10] methodology. It also observed from Table 4, the accuracy of the proposed DELM approach with the same number of instances is 96.8% and miss rate is 3.2%.

In the case of CapsNet and CNN [10], with a single hidden layer when the neuron count increases, the performance of the system is also increased, as shown in Table 4. Proposed IRBC DELM based solution also compared with CNN [11] based solution in the case of an increase in a number of hidden layers. This means that the performance of the system is increased by the increased number of neurons but not much as in the proposed IRBC DELM system. It demonstrates that increases in the number of neurons are not sufficient, but for a better result, hidden layers should also be increased to improve the performance of the system, as shown in Table 4. It is observed that the proposed IRBC DELM system gives attractive results in the cost of computational complexity.

5 Conclusion

Modeling the prediction of digits is a challenging task in humans. We have proposed a model of digits prediction for humans to improve prediction accuracy. The proposed IRBC method is a new expert system based on an artificial neural system with deep extreme learning machine (DELM) high potential points to digits. For any new test as recommended by experts, the proposed model of digit prediction can be extended. In this work, a deep extreme learning approach with feedforward and backpropagation neural networks algorithm was used for the digits prediction to the dataset that was obtained from the Kaggle [23]. Various numbers of the hidden layer neurons were defined, and diverse activation features were used for the ideal arrangement of different DELM parameters to obtain an optimized DELM structure. For measuring the performance of such machine learning algorithms, we used various statistical measures. These measuring figures show that the execution of proposed IRBC DELM in comparison with other algorithms is far better. Compared to past approaches, our proposed DELM technique produces attractive results. The proposed technique has 97.8% accurate which is more accurate as compared to previously published techniques. It is observed that the proposed IRBC DELM system gives attractive results in the cost of computational complexity. We have confidence in initial results and currently find different substitutes and collect data in the direction outlined above to expand this work.

Compliance with ethical standards

Conflicts of interest The authors declare that there are no conflicts of interest regarding the publication of this article.

References

1. Kumar Patel D (2012) Handwritten character recognition using multiresolution technique and euclidean distance metric. J Signal Inf Process 3(2):208–214
2. Sheth R, Chauhan NC, Goyani MM, Mehta KA (2011) Handwritten character recognition system using chain code and correlation coefficient. Communication 31–36
3. Pal A, Singh D (2010) Handwritten english character recognition using neural networks 1(2):5587–5591
4. Singh P (2011) Radial basis function for handwritten devanagari numeral recognition. Int J 2(5):126–129
5. Yamada H, Nakano Y (1996) Cursive handwritten word recognition using multiple segmentation determined by contour analysis. IEICE Trans Inform Syst E79-D:464–470
6. Blumenstein M, Yu X, Verma B (2007) An investigation of the modified direction feature for cursive character recognition. Pattern Recognit 40:376–388
7. Kimura F, Kayahara N, Miyake Y, Shridhar M (1997) Machine and human recognition of segmented characters from handwritten words. In: Proceedings of the fourth international conference on document analysis and recognition, vol 2. IEEE, pp 866–869. https://ieeexplore.ieee.org/abstract/document/620635/
8. Cruz RMO, Cavalcanti GDC, Ren TI (2010) Handwritten digit recognition using multiple feature extraction techniques and

classifier ensemble. In: 17th international conference on systems, signals, and image processing, pp 215–218
9. Lee LL, Gomes NR (1997) Disconnected handwritten numeral image recognition. In: Proceedings of 4th ICDAR, pp 467–470
10. Sabour S, Frosst N, Hinton GE (2017) Dynamic routing between capsules. In: Advances in neural information processing systems, pp 3856–3866, 2017
11. Kumar VV, Srikrishna A, Babu BR (2010) Classification and recognition of handwritten digits by using mathematical morphology. Sadhana 35:419–426
12. Vasant AR (2012) Performance evaluation of different image sizes for recognizing offline handwritten gujarati digits using neural network approach. In: 2012 international conference on communication systems and network technologies, pp 271–274
13. Maloo M, Kale KV (2011) Support vector machine based Gujarati numeral recognition. Int J Comput Sci Eng 3(7):2595–2600
14. Baheti MJ, Kale KV, Jadhav ME (2011) Comparison of classifiers for Gujarati numeral recognition. Int. J. Mach. Intell. 3(3):93–96
15. Baheti MJ (2012) Gujarati numeral recognition: affine invariant moments approach. Soft Comput 12:140–146
16. Kamal Moro MF (2013) Gujarati handwritten numeral optical character through neural network and skeletonization. J Syst Comput 12(1):40–43
17. Desai AA (2010) Gujarati handwritten numeral optical character reorganization through neural network. Pattern Recognit 43(7):2582–2589
18. LeCun Y, Cortes C, Burges JC (n.d.) The mnist database of handwritten digits. http://yann.lecun.com
19. Ciresan DC, Meier U, Gambardella LM, Schmidhuber J (2011) Convolutional neural network committees for handwritten character classification. In: 2011 international conference on document analysis and recognition (ICDAR). IEEE, pp 1135–1139
20. Deng L (2012) The MNIST database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Process Mag 29(6):141–142
21. Gedeon TD, Harris D (1992) Progressive image compression. In: International joint conference on neural networks (IJCNN), vol 4. IEEE, pp 403–407
22. Cheng J, Duan Z, Xiong Y (2015) QAPSO-BP algorithm and its application in vibration fault diagnosis for hydroelectric generating unit. Shock Vib. 34:177–181
23. Huang G-B, Wang DH, Lan Y (2011) Extreme learning machines: a survey. Int J Mach Learn Cybern 2:107–122
24. Wei J, Liu H, Yan G, Sun F (2017) Robotic grasping recognition using multi-modal deep extreme learning machine. Multidimens Syst Signal Process 28:817–833
25. https://www.kaggle.com/ngbolin/mnist-dataset-digit-recognizer/data

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.