Digital Signature Verification Using Artificial Neural Networks

Gopichand G, Sailaja G, N. VenkataVinod Kumar, T. Samatha

Abstract: Identification and verification of hard written signature from images is major issue. This is very difficult as even human eye does not have that much visual ability to identify every detail of the in handwritten.Signature changes every time so it is difficult for humans to identify the original and forged ones. By using deep learning which uses the sophisticated is digital configured replica of human brain, we can identify the forgery done in signature with higher accuracy.

Index Terms: deep learning, digital configured replica, forgery, signature

I. INTRODUCTION

The robustness of human brain has always been an enigma and this has caused people to replicate it digitally. The human eye has a great efficiency of recognition due its architecture. This inspiration has led to people constructing artificial neural network and so deep learning.

In this we generally are going to assess the ways a human being would give his signature using some deep learning algorithms and artificial neural networks by which we can train the system accordingly and verify if the signature is real or forged. It would be a great way to authenticate the signatures and verify them accordingly. It would be a better option to verify the signatures using this model rather than visual recognition through human eye which have a high chances of making a mistake.

II. SIGNATURE VERIFICATION

2.1 Offline Signature Verification:

Verification of signatures with features which are already present is called as offline signature verification. The features are very simple and basic and the image scanned through a camera should follow certain methods for verification. Design of these kind of systems is difficult as there will be less features available.

2.2 Nature of Human Signature:

Human signatures are generally generated by the inbuilt functions of the human neuromuscular area which induces rapid movements. This system will largely consist of neurons and muscle and fibers which make us know that the velocity of the hand produces the equation. So signatures for every person are unique. In this model we can assess the person who will give the signature and train our model accordingly.

2.3 Types of Forgeries:

Forgeries of signatures are classified into three types as mentioned below and we will solve and try to prevent all this forgeries in our model.

2.3.1 Random forgery:

A signature which is forged and it maybe the genuine signature of other person**.**

1.3.2 Casual Forgery :

A signature forgery in which the one who is doing the forgery will know the name of the victim

1.3.3 Skilled Forgery:

As the name suggests a person who is skilled professional is forging signatures is involved in forging the signatures.

III. NEURAL NETWORK OUTLINE:

A system which does computing and is combines with basic, and highly coincidental processing elements which use the data to get a highly relevant and faster response from the inputs taken. Artificial neural network models are a subpart of the machine learning models which are motivated by the functioning of the brain. Neural networks generally work like the neurons of the brain and the connected neurons will work in a network process to collect and process the data for providing the necessary output.There will be an input layer to the system which consists of all the patterns in which the system should process and also the necessary inputs and it communicates with the hidden layer as shown in the below figure and the hidden layers use the patterns and inputs by the input layer and are used to find out a relevant function for the task to be performed and then they communicate with the output layers to display the final output.

Feedforward mechanism:

This mechanism does not form circles like many artificial neural networks. This mechanism goes in a single way from the input to the hidden layers to the output and do not form any loops or circles in the process.

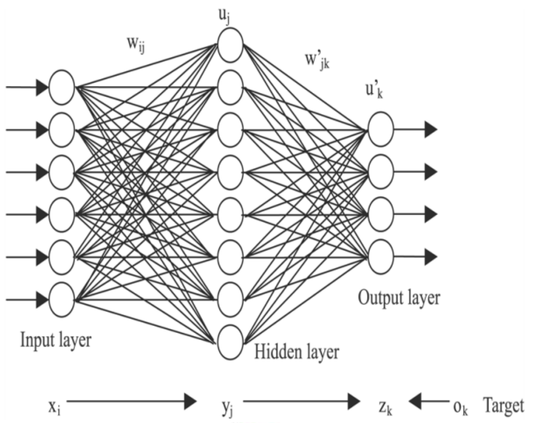
Revised Manuscript Received on Janurary 25, 2019.

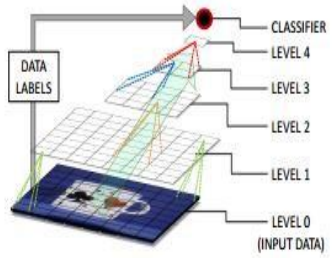
Gopichand G,Assistant Professor, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore,India

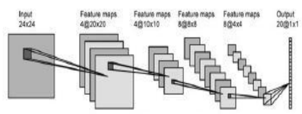
Sailaja G, Assistant Professor, Computer Science and Engineering, SV Engineering College for Women, Tirupati, India.

N. VenkataVinod Kumar, Assistant Professor, Computer Science and Engineering, Annamacharya Institute of Technology & Sciences, Tirupati, India

T. Samatha, Assistant Professor, Computer Science and Engineering, Annamacharya Institute of Technology & Sciences, Tirupati, India.







IV. PROPOSED METHODOLOGY :

In our proposed method we will construct a neural network by optimizing some existing neural networks and it will have a use the data structure tree along with nodes similar to human eye which has neurons and it used for recognition of patterns. There are several steps involved in our method and it goes as following.

V. IMPLEMENTATION

5.1 Pre-processing:

In image processing application, pre-processing is required to remove discrepancies, from the input image. Signatures are changed to greyscale, using following equation as:

Grey color = (0.114*Blue)+ (0.299*Red) + (0.5876*Green)

The important factor in preprocessing stage is to build standard signature which is prepared for extraction of features. The pre-possessing stage includes:

5.1.1 Image scaling:

Let H = input image height & W= input image width. The image can be fit to 100*100 pixels by applying the equations:

Xnew = (Xold * 100)/H;

Xnew is calculated X coordinate and Xold aris the original X coordinate

Ynew = (Ynew * 100)/W;

Ynew is calculated Y coordinate and Yold aris the original Y coordinate

With the above equations,the image is scaled to a uniform 100*100 pixels image.

5.1.2 Normalization of signature:

It is possible that signature can be fractured due to imperfections in image scanning and capturing. It is also possible that the dimensions of signature can vary from person to person and even the same person can sometimes have different sizes based on the mood and environmental factors. So a process is required to overcome the size variation problem and achieve a standard signature size for all signatures.

We also need to preserve the characteristic ratio between height and width of a signature.

After performing the normalization process, all the signatures will have the similar dimension. Normalization process is done based on the below equations

Ynew = [(Yold-Ymin)/ (Ymax-Ymin)]*M

Xnew = [(Xold-Xmin)/ (Xmax-Xmin)]*M

Xold and Yold are coordinates for original input signature,

Xnew and Ynew are coordinates for normalized signature,

M = Width or height meant to the normalized signature

5.1.3 Thinning:

It is possible that the signature is written on different pen and the thickness thus varies from one pen to another. The purpose of thinning is to eliminate thickness differences in signature by making all of them one pixel thick.Thinning is used to enhance the object's global properties and to transform the input image into a compact form.

5.2 Feature Extraction:

This method is used for extracting the necessary and essential features from the input image. A feature vector is created from the features extracted. Each signature has a unique feature vector. These features are extracted as follows

The feature extraction module uses moment invariants to extract texture features of the image using central moment and derived invariant moment. The central moment, μ, with respect to the centroid, and the normalized central moment, are calculated as:

$$m_{pq} = \sum_x \sum_y (x - x')^p (y - y')^q a_{xy}$$

$$\eta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{\lambda}}$$

$$\text{where, } \lambda = \frac{(p+q)}{2} + 1, (p+q) > 2$$

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Central Moments	Derived Invariant Moments
$\mu_{00}=m_{00}$	$I_1=\eta_{20}+\eta_{02}$
$\mu_{10}=0$	$I_2=(\eta_{20}-\eta_{02})^2+4\eta_{11}^2$
$\mu_{01}=0$	$I_3=(\eta_{30}-3\eta_{12})^2+(3\eta_{21}-\eta_{03})^2$
$\mu_{20}=m_{20}-x' m_{00}$	$I_4=(\eta_{30}+\eta_{12})^2+(\eta_{21}-\eta_{03})^2$
$\mu_{02}=m_{02}-y' m_{01}$	$I_5=(\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})^2((\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2)+(3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03})(3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2)$
$\mu_{11}=m_{11}-y' m_{10}$	$I_6=(\eta_{20}-\eta_{02})((\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2)+4\eta_{11}(\eta_{30}+\eta_{21})(\eta_{21}+\eta_{03})$
$\mu_{30}=m_{30}-3x' m_{20}+2x' m_{10}$	$I_7=(3\eta_{12}-\eta_{30})(\eta_{30}+\eta_{12})((3\eta_{30}\eta_{12})-3(\eta_{21}+\eta_{03})^2)+(3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03})((3\eta_{30}\eta_{12})^2-(\eta_{21}+\eta_{03})^2)$

5.3 Neural network training:

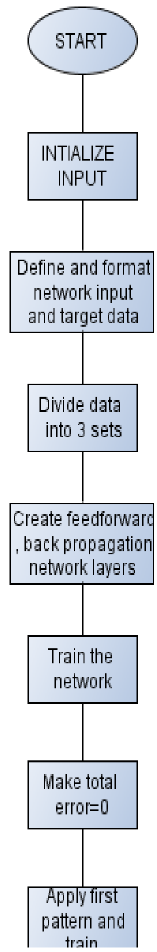
The features extracted are fed to natural network as inputs. Before that the networks are trained with data sets.Each neural network has a corresponding user to it. So a user has two neural networks one with feedforward mechanism and the other with feedback mechanism. The user's features are given as input to both the neural networks and the output is recorded.

VI. FEEDFORWARDMECHANISM ALGORITHM:

```
get data set
define activate(weights, inputs) set as activation set as weights[-1] for i in range 0 to len(weights)-1 activation + set as weights[i] * inputs[i]
    return activation
define transfer(activation) set as
    return 1.0 / (1.0 + e^-activation)
define forward propagation(network, row):
 inputs set as row for layer in network:
        for neuron in layer:
  activation set as activate(neuron['weights'], inputs) neuron['output'] set as transfer(activation) temp inputs. Add(neuron['output'])
inputs set as temp inputs return inputs
define train network(network, train, learning rate, number of epochs, number of output neurons) set as
for epoch in range(number of epochs):
   for row in train:
            outputs set as forward propagation(network, row)
            expected set as [0 for i in range(number of output neurons)]
  expected[row[-1]]setas1 backward_propagate_error(network,expected) update_weights(network, row, learning rate)
define initialize network(number of input neurons, number of hidden neurons, numberof output neurons)
set as network set as list()
    hidden_layer set as [{'weights':[random() for i in range 0 to number of input neurons +1]}
 for i in range 0 to number of hidden neurons]     network.add(hidden_layer)
     output_layer set as [{'weights':[random() for i in range 0 to number of hidden neurons +1]}
 for i in range 0 to number of output neurons]
     network.add(output_layer)
    return network
define predict(network, row) set as
    outputs set as forward propagation(network, row)  return outputs. Index(max(outputs))
define backpropagation(train, test, learning rate, number of epoches, number of hidden neurons):   number of input neurons set as len(train[0]) - 1
    number of output neurons set as len( ([row[-1] for row in train]))
    network set as initialize_network(number of input neurons, number of hidden neurons, number of output neurons)
train_network(network, train, learning rate, number of epoches, number of output neurons) predicts set as list() for row in test:
predict set as predict(network, row)
        predicts.add(predict) return(predicts)
number of folds set as 6 learning rate set as 0.2 number of epoches set as 1500
number of hidden neurons set as 10

    set assign folds set as cross_validation_split of data

    for every fold in folds
        set assign training set as list of folds
        remove fold from training
        training set as sum of training
        for row in fold
            copy of row set as list(row)
            add copy of row to test data
            l set ast element of copy of row set as None
        predicted set as algorithm(training , test data , *args)
        actual set as l set ast element of row for all row in field
         correct set as 0
    for i in range 0 to length of actual
        if actual[i] set as set as predicted[i] then
            correct + set as 1
     accuracy set as correct / length of actual * 100.0
    add accuracy to scores
    return scores
print the scores
print mean accuracy set as sum of scores/length of scores
```



VII.OUTPUT EXPLANATION:

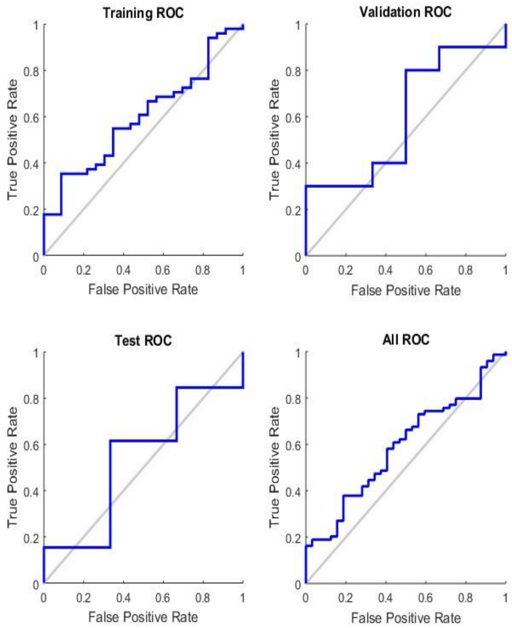
The output for this model will be 1 if the signature is real and will be 0 if the image is forged.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

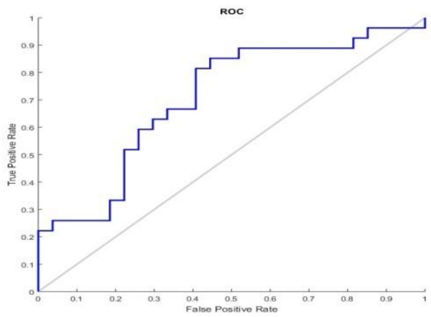
VIII. RESULT AND PERFORMANCE:

8.1 ROC GRAPH:

8.1.1 ROC GRAPH FROM TRAINING



8.1.2ROC PLOT AFTER TESTING



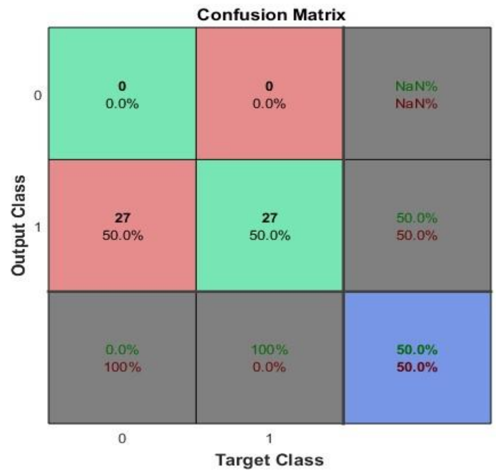
8.2 INFERNECE FROM THE GRAPH

As we can see that the graph we can intervene that there is high possibility in getting accurate value.

8.3 CONFUSION MATRIX:

8.3.1 Confusion matrix –training:

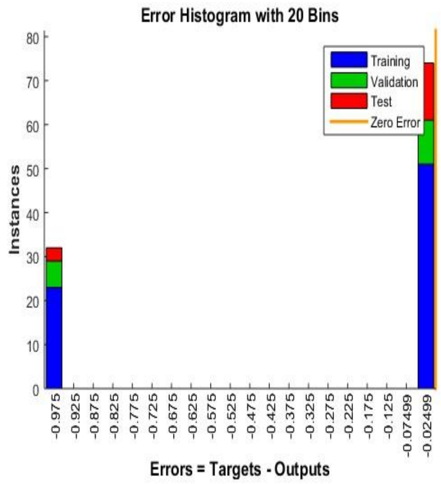
8.3.2 Confusion Matrix from testing:

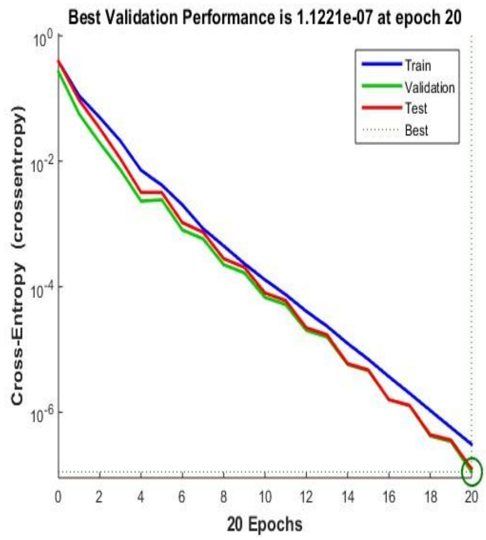


8.3.3 Interpretation:

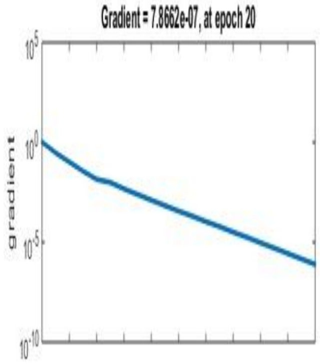
As we can observe the in testing matrix the possibility is distributed 50% equally is sign that the neural network works efficiently

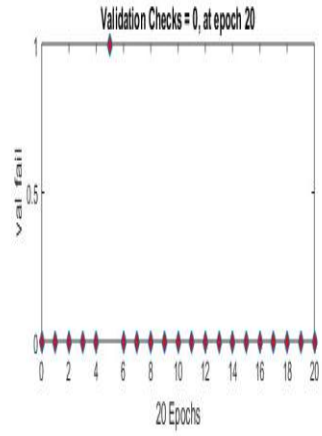
IX. OTHER GRAPHS:



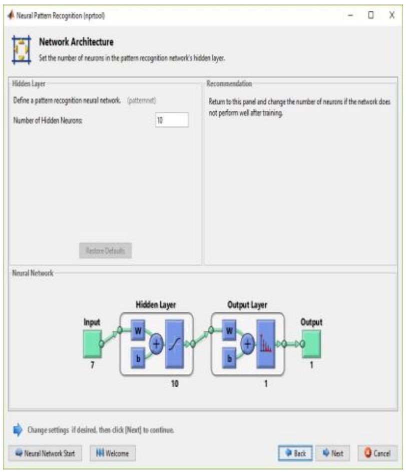


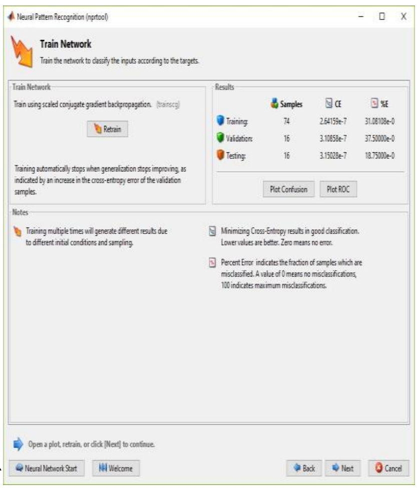
Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

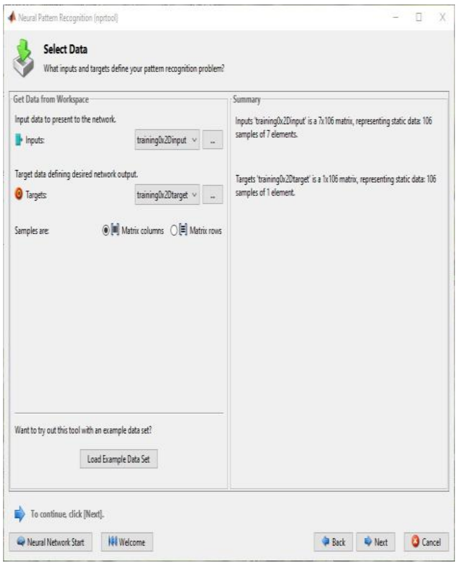


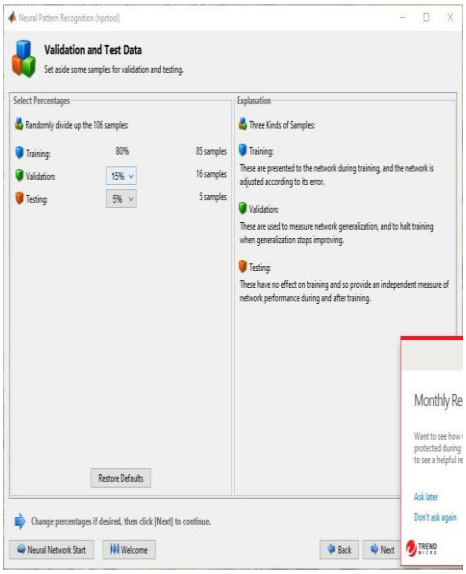


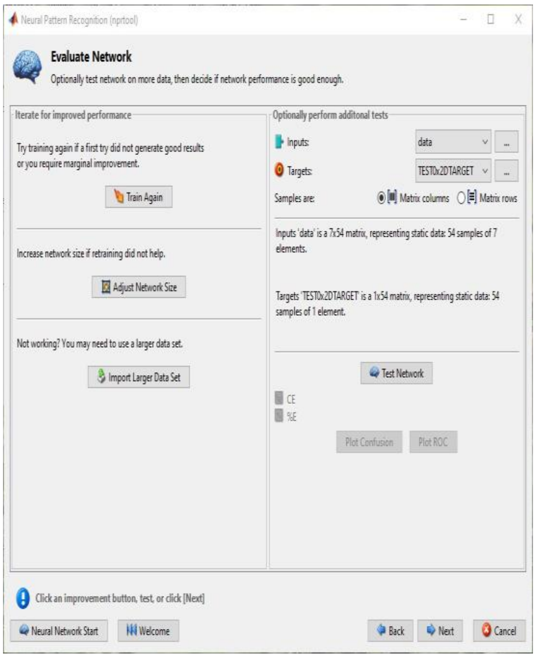
X. WORK PLACE IMAGES:



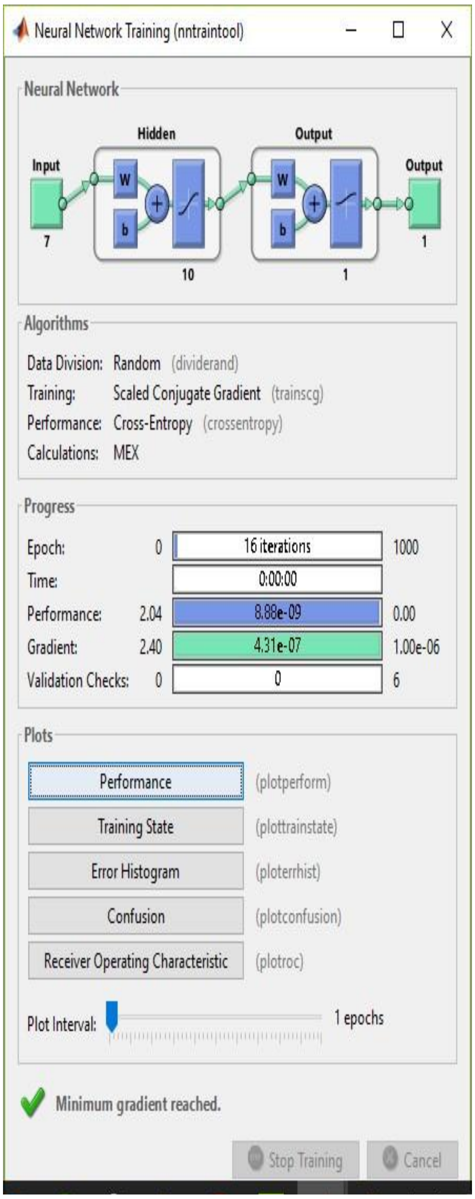








Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication



REFERENCES

1. S. Yin, A. Jin, Y. Han, and B. Yan, "Image-based handwritten signature verification using hybrid methods of discrete Radon transform , principal component analysis and probabilistic neural network," Appl. Soft Comput. J., vol. 40, pp. 274–282, 2016.
2. K. Wrobel, R. Doroz, P. Porwik, J. Naruniec, and M. Kowalski, "Engineering Applications of Artificial Intelligence Using a Probabilistic Neural Network for lip-based biometric verification," Eng. Appl. Artif. Intell., vol. 64, no. January, pp. 112–127, 2017.
3. D. Suryani, E. Irwansyah, R. Chindra, D. Suryani, E. Irwansyah, and R. Chindra, "ScienceDirect O fflfflineine Signature Signature Recognition Recognition and and Verification Verification System System using usingfficient Fuzzy Kohonen Clustering Network (EFKCN) Algorithm E fficient Fuzzy Kohonen Clustering Network (EFKCN) Algorithm," ProcediaComput. Sci., vol. 116, pp. 621–628, 2017.
4. Y. Serdouk, H. Nemmour, and Y. Chibani, "New off-line Handwritten Signature Verification method based on Artificial Immune Recognition System," Expert Syst. Appl., vol. 51, pp. 186–194, 2016.
5. Y. Serdouk, H. Nemmour, and Y. Chibani, "Handwritten signature verification using the quad-tree histogram of templates and a Support Vector-based artificial immune classification ☆," Image Vis. Comput., vol. 66, pp. 26–35, 2017.
6. P. Porwik, R. Doroz, and T. Orczyk, "Signatures veri fi cation based on PNN classi fi eroptimised by PSO algorithm," vol. 60, pp. 998–1014, 2016.
7. S. Kumar, D. Prosad, and P. Pratim, "Fast recognition and verification of 3D air signatures using convex hulls," Expert Syst. Appl., vol. 100, pp. 106–119, 2018.
8. N. Khera and S. A. Khan, "Microelectronics Reliability Prognostics of aluminum electrolytic capacitors using arti fi cial neural network approach," Microelectron. Reliab., vol. 81, no. October 2017, pp. 328–336, 2018.
9. A. Fallah, M. Jamaati, and A. Soleamani, "A new online signature verification system based on combining Mellintransform , MFCC and neural network," Digit. Signal Process., vol. 21, no. 2, pp. 404–416, 2011.
10. D. Dabrowski, "Condition monitoring of planetary gearbox by hardware implementation of artificial neural networks," Measurement, vol. 91, pp. 295–308, 2016.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication