Soft Computing Based Texture Classification with MATLAB Tool

Pankaj H. Chandankhede

Abstract:- This paper deals with Implementation of my previous work [1]. Here MATLAB simulation software is use as a platform tool for designing the concept of Texture Classification using Soft Computing Tool as a function of MATLAB. This paper classifies Textures on the basis of two novel approaches of artificial neural network & adaptive neuro-fuzzy inference system. This paper proves that neuro-fuzzy model performed better than the neural network in classification of texture image of three different types.

Index Terms:- Database, Neural Network Toolbox, Training, DCT Features, ANN, ANFIS, FIS Editor.

I. INTRODUCTION

Texture is as image consists of mutually related and interrelated elements. Because of this characteristic of texture it is use in recognition and interpretation. Texture analysis is emerging as important tool in content-based retrieval in large image and video databases. So to solve such queries, texture classification and segmentation schemes are used [4].

Texture classification can be divided into two groups, stationary and non-stationary texture classification. Single texture is present in a stationary texture. While multiple textures in one image are present in non-stationary texture. The different implementations shared a same idea, which they will view different classes as different textures when they are performing the classification process. However, certain parameters of the algorithms must be adjusted to suit the different implementations. Here, 3 different textures are considered.

- Brick Texture
- Metal Texture
- Rural Texture

MATLAB is a high-performance language which integrates computation, visualization, and programming where problems as well as solutions are expressed in mathematical notation. MATLAB consist of wide range of toolboxes. Toolboxes consist of wide range of function that can be used to implement a particular design. MATLAB consist of different toolboxes such as signal processing, fuzzy logic, control systems, wavelets, neural networks. Depending on the design of particular model, proper toolbox is to be selected.

This paper make use of DCT for texture image compression and its DCT coefficients so developed is use to train the two soft-computing models namely neuro computing and neuro fuzzy computing. Depending on this training it is going to classify any unknown sample of texture image taken from database that was previously build for testing purpose.

Manuscript received on April 26, 2012.

Pankaj H. Chandankhede, Electronics & Telecommunication Engineering , G. H. Raisoni College of Engineering, Nagpur, Maharashtra, India, +919561532063, (e-mail: Pankaj.8485@gmail.com).

II. DATABASE

The Texture Database includes approximately 70 images for three different textures that are Brick, Metal and Rural. The database includes 35 images for the testing purpose. These testing images are used to test the training Model for its accuracy in terms of classification.

III. NEURAL NETWORK TOOLBOX

The Neural Network Toolbox consists of tools for designing, implementing, visualizing, and simulating neural networks. For application such as pattern recognition and nonlinear system identification and control, neural networks play a very vital role where analysis is difficult. The Toolbox of Neural Network provides support for many proven network paradigms, as well as graphical user interfaces (GUIs) that enable you to design and manage your networks. The toolbox simplifies the creation of customized functions and networks.

A neural network can learn and therefore can be trained to find solutions, recognize patterns, classify data and forecast future events. The behavior of a neural network is defined by the way its individual computing elements are connected and by the strength of those connections or weights. The training the network is done so that weights are automatically adjusted until it performs the desired task correctly by certain training rule. The GUIs of Neural Network Toolbox make it more user friendly to work with neural network function which are define in Toolbox. Wizard of Neural Network Fitting Tool leads you through the process of fitting data using neural networks. By making use of this fitting tool large and complex data sets can be imported which will quickly create, train & evaluate networks performance. Thus network architecture is easy to visualize and understand.

IV. DISCRETE COSINE TRANSFORM

Firstly a query image is provided and converted into a gray level version. The texture feature vector is obtained from some DCT coefficients. It is computed directly from the DCT coefficients and the spatial localization using sub blocks. Thus, it does not require additional complex computation as well as it overcome some problems such as computational complexity and storage space.

In DCT transform first step is to convert an RGB image into a gray level image. Here block-based DCT transformation is used. Each image is divided into N*N sized sub-blocks where the size of N depends on size of image. The 2D DCT can be written in terms of pixel values f (i, j) for i, j = 0, 1… N-1 and the frequency-domain transform coefficients F (u, v).

$$F(u,v)=$$

$$\frac{1}{\sqrt{2N}}c(u)c(v)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j)*\cos[\frac{(2i+1)u\pi}{2N}].\cos[\frac{(2j+1)v\pi}{2N}]$$

for u, v =0,1,…N-1

where,

$$c(x) = \frac{1}{\sqrt{2N}} \quad \text{for } x= 0$$

$$c(x) = 1 \quad \text{otherwise}$$

The inverse DCT transform is given by…

$$f(u,v)=$$

$$c(u)c(v)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}c(u)c(v)\ F(u)(v)\times\cos[\frac{(2i+1)u\pi}{2N}].\cos[\frac{(2j+1)v\pi}{2N}]$$

for i, j =0,1,…N-1

The job of DCT transforms is to convert a signal from a spatial representation into a frequency representation. Obviously Lower frequencies are more frequent than higher frequencies in a particular image. So, image is transform into its frequency components and gives away a lot of higher frequency coefficients, then the amount of data needed to describe the image can be reduced without sacrificing on image quality.

An image in DCT transformation is represented as a sum of sinusoids of changing magnitudes & frequencies. For a particular image, most of the visually significant information about the image is only concentrated in just a few coefficients & DCT makes the use of this property.So, DCT is often used in image compression applications.

For Compression, the input image is first divided into 8-by-8 blocks and the 2D DCT is computed for each block. This process is followed by quantization, coding using suitable technique and transmitted. To reconstruct the image, receiver decodes the quantized DCT coefficients with the same coding & computes the inverse 2D DCT of each block and then single image is form by blocks . For a particular images, many of the DCT coefficients have values close to zero so these coefficients can be discarded which will not going to affect less in terms of image quality.

Once the quantization process is completed , the 63 AC coefficients are treated separately from the DC coefficient. The DC coefficient represent average value of the 64 image samples.There is usually strong correlation between the DC coefficients & AC coefficients. Lastly all of the quantized coefficients are ordered into the “zig-zag” sequence. This ordering helps to facilitate entropy coding by placing low-frequency coefficients before high-frequency coefficients.

V. NEURO-COMPUTING

Neural networks are inspired by the processing of information in the nervous system. An artificial neuron resembles the same characteristics as that of natural neurons. Such type of network which is collection of neurons is artificial neural networks (ANNs).

VI. NEURO FUZZY SYSTEM

ANN learns by adjusting the interconnections or synaptic weights between layers. One of the popular framework is FIS which is based on the concept of set theory, if-then rules and reasoning. So, it is necessary to combine ANN and FIS. Classification of integration of ANN and FIS is broadly done in three categories namely concurrent model, cooperative model and fully fused model. For solving complex problems, Neuro Fuzzy (NF) computing is a popular framework.

VII. DESIGN AND IMPLEMENTATION

The Texture classification system using neuro computing and neuro-fuzzy computing works according to the above flowchart. Firstly, the Texture database consist of different texture images is created. For this dissertation, three different textures images corresponding to Brick, Metal and Rural is taken. Also, for the testing purpose about 35 different texture images for each class is created.

Initially, images are open from database. These images are converted into grayscale because DCT work on grayscale images. Thus, from each image 324 DCT coefficients are extracted. After that block DCT is applied on each texture image, in order to find its feature. These features are extracted in zig-zag pattern. These features are used to represent each image. Finally, the image is trained using artificial neural network & ANFIS (Adaptive Neuro-Fuzzy Inference System). If the classification is right then it will display the message showing that texture belongs to particular class. Otherwise, images are again train with increasing the number of epoch.

Figure 1 shows the entire flowchart for the implementation of this paper. These steps need to be executed in same sequence for the successful & efficient classification of texture. Figure 1 also shows if the classification is not done properly then again training is required.

VIII. SIMULATION RESULT

This paper makes the use soft computing technique which is use to classify 3 different types of textures. Texture database consist of different textures images of classes (brick, metal and rural). Firstly, DCT is applied on database images, which will give DCT coefficients. These DCT coefficients is use represent the different textures. Each texture image was represented by certain DCT coefficients. Different texture images have been used for testing purpose. Neuro-fuzzy model is capable of determining the architecture automatically that is number of hidden neurons and number of layers of the neural network

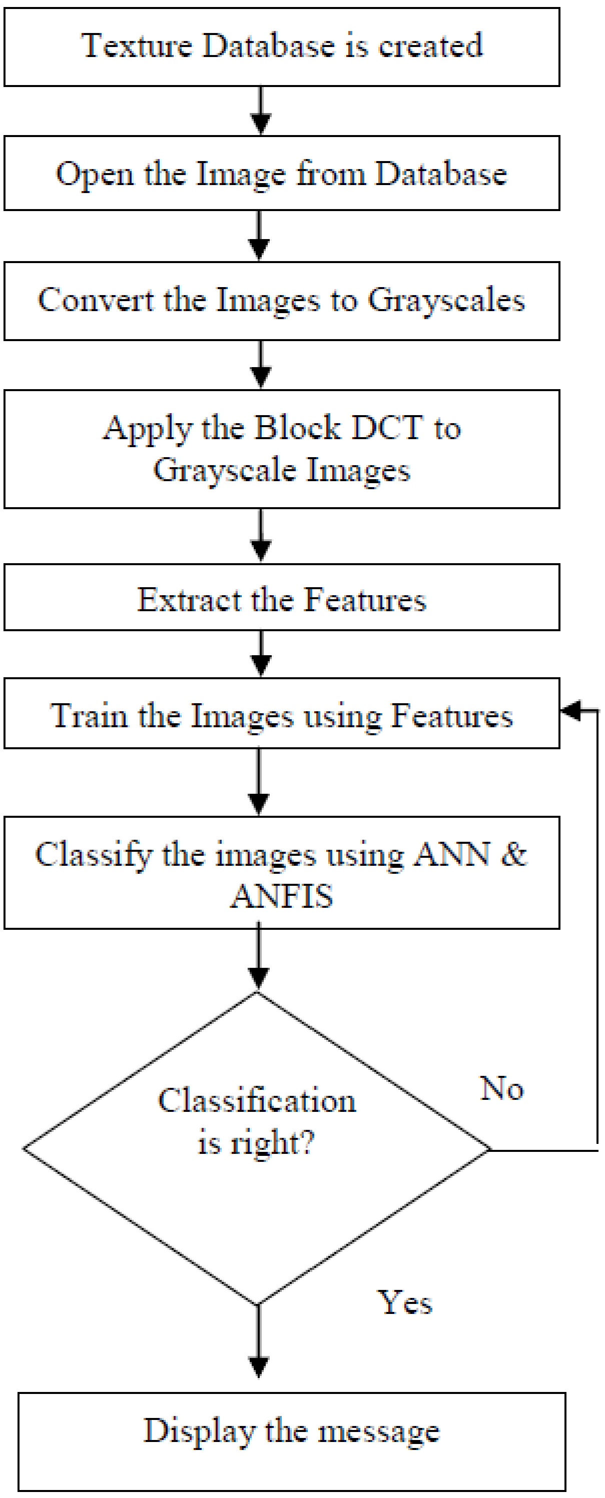

Texture Database is created
Open the Image from Database
Convert the Images to Grayscales
Apply the Block DCT to Grayscale Images
Extract the Features
Train the Images using Features
Classify the images using ANN & ANFIS
Classification is right?
No
Yes
Display the message

Figure 1: Flowchart of Design

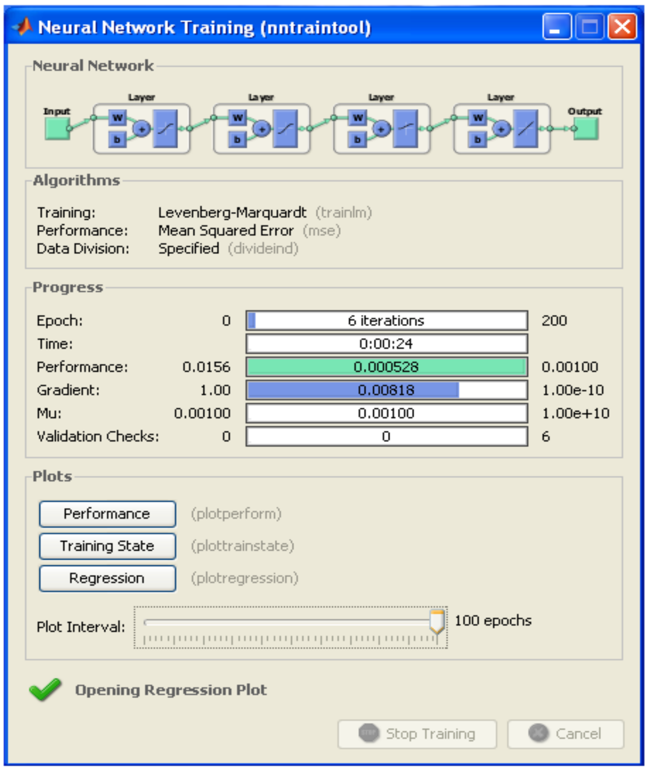
IX. SIMULATION RESULT

This paper makes the use soft computing technique which is use to classify 3 different types of textures. Texture database consist of different textures images of classes (brick, metal and rural). Firstly, DCT is applied on database images, which will give DCT coefficients. These DCT coefficients is use represent the different textures. Each texture image was represented by certain DCT coefficients. Different texture images have been used for testing purpose. Neuro-fuzzy model is capable of determining the architecture automatically that is number of hidden neurons and number of layers of the neural network.

Here, soft computing techniques are implemented in MATLAB software which makes the use of neural network, fuzzy logic and image processing toolbox. In order to make the MATLAB execution process simple, graphical user interface (GUI) is design in MATLAB. Following are the snapshot at various steps while executing the functionality provided in GUI.

Step 1:- Train the image using neural network.

When Training Based on ANN button in GUI is click, then


Neural Network Training (nntraintool)

Neural Network

Algorithms

Training: Levenberg-Marquardt (trainlm)
Performance: Mean Squared Error (mse)
Data Division: Specified (divideind)

Progress

Epoch:	0	6 iterations	200
Time:		0:00:24	
Performance:	0.0156	0.000528	0.00100
Gradient:	1.00	0.00818	1.00e-10
Mu:	0.00100	0.00100	1.00e+10
Validation Checks:	0	0	6

Plots

Performance (plotperform)
Training State (plottrainstate)
Regression (plotregression)

Plot Interval: 100 epochs

Opening Regression Plot

Stop Training Cancel

Fig.2 Training of image using neural network

this gives call to a callback function written in maingui.m file.In callback function, it first give call to generateDataset() function which is used to extract 324 features from each

Figure 2: Training of image using neural network.

image. For this it applies block DCT technique in which input image is converted into 8-by-8 sub images and then DCT is applied to each sub images individually. Then its features are extracted in zig-zag fashion. After that it gives call to load data and TrainNeural function which is used to train the neural network.

nntraintool :- Launches the neural network training GUI in MATLAB.

This function is used to call the GUI in neural network toolbox which makes training GUI visible prior to training. It can be use to check the progress bar of number of Epoch, performance, gradient as well as validation checks. It is also use to observe the plots of performance, training state & regression.

MATLAB consist of train function that trains a network net.

train(net,P,T,Pi,Ai) takes net(Network), P(Network inputs), T(Network targets (default = zeros), Pi(Initial input delay conditions), Ai(Initial layer delay conditions).

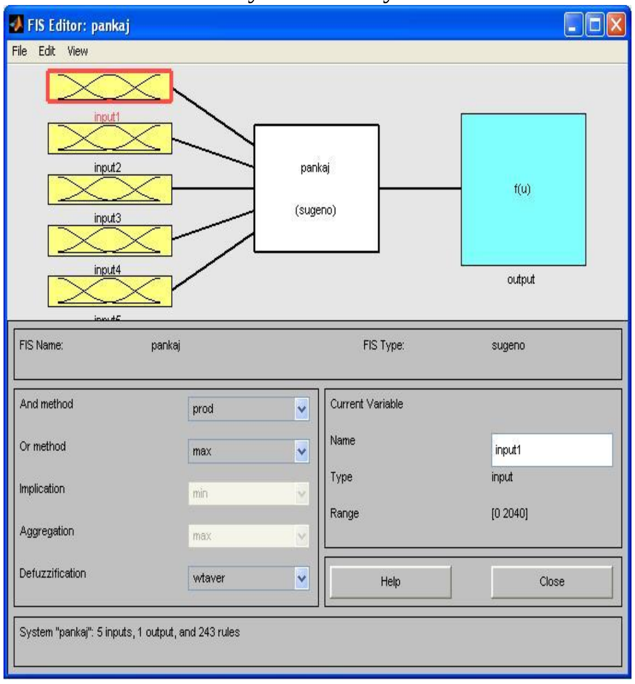
net.trainFcn function is called by train function using values indicated by *net.trainParam*.

Here, in this paper, training occurs until a maximum number of epochs occurs i.e. 200 is reached or the performance goal is met, or any other stopping condition occurs. Although training of neural network take some time but once train it shows result in a more specified form.

Step 2:- Train the image using ANFIS.

When Training Based on ANFIS button in GUI is click, then this gives call to a callback function written in maingui.m file. In callback function, it first gives call to load fismat. While training, load fismat is used for 324 times since, 324 features are extracted from each image. This will load fismat corresponding to features of each image. After that it will train the ANFIS model using the traindata. This model will automatically implement rule and make necessary modification in rule.

Applying Rules & type of membership function using FIS Editor. The FIS Editor is use to display the general information about a fuzzy inference system.



FIS Name: pankaj   FIS Type: sugeno

And method: prod
Or method: max
Implication: min
Aggregation: max
Defuzzification: wtaver

Current Variable
Name: input1
Type: input
Range: [0 2040]

System "pankaj": 5 inputs, 1 output, and 243 rules

Figure 3: FIS editor

Figure 3 shows the five layer ANFIS model in which

Layer 1: is an input layer which transmits the inputs to their corresponding membership functions in the layer 2.

Layer 2: This layer contains the input membership functions implemented as Gaussians.

Layer 3: This layer represents the fuzzy rules.

Layer 4: This layer performs the fuzzy OR, AND or NOT operation between rules that produce the same consequences.

Layer 5: This layer uses Gaussian output membership functions and performs the defuzzyfication operation using a modified Center of Area algorithm.

Finally, the ANFIS Model Structure is shown in Figure 4

This Model gives an idea about input, hidden & output layers with number of neurons used in each layer.

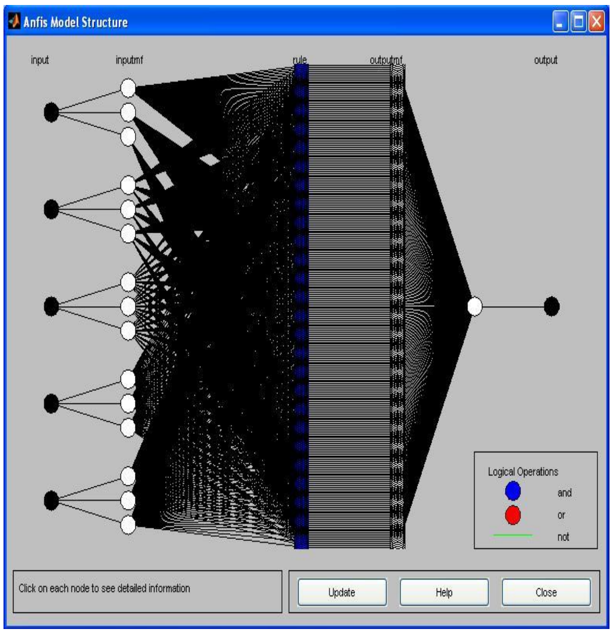


Figure 4: ANFIS Model Structure

Step 3:- Opening a query image from the given Dataset. Firstly the image of Brick Texture is open.

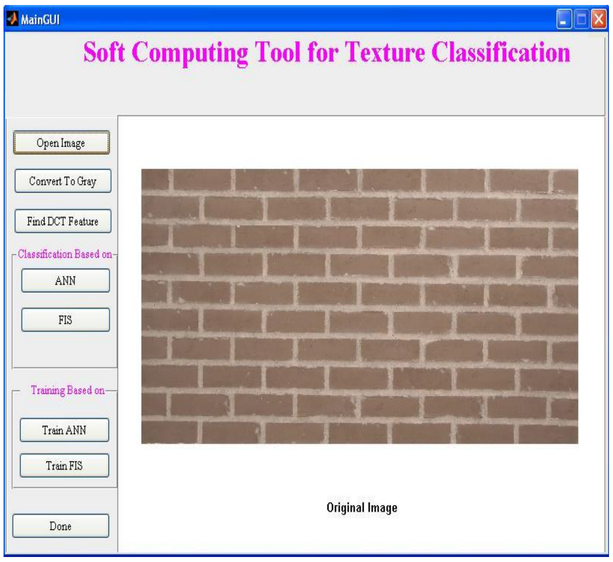


Figure 5: Opening Brick Image from Dataset

When open image button in GUI is click, then this gives call to a callback function written in maingui.m file. In callback function, imgetfile command is used which will open a window asking for image to be open. Give the path of that image file. Also, imshow command is used to display the image in GUI.

Step 4:- Converting an original query image into grayscale. Since, DCT works on grayscale.

When Convert to Gray button in GUI is click, then this gives call to a callback function written in maingui.m file.In callback function, rgb2gray( ) is used which will convert an original color image to gray scale image.

Since, DCT is used in next step which requires gray scale image for its operation. Also, imshow command is used to display the image in GUI.

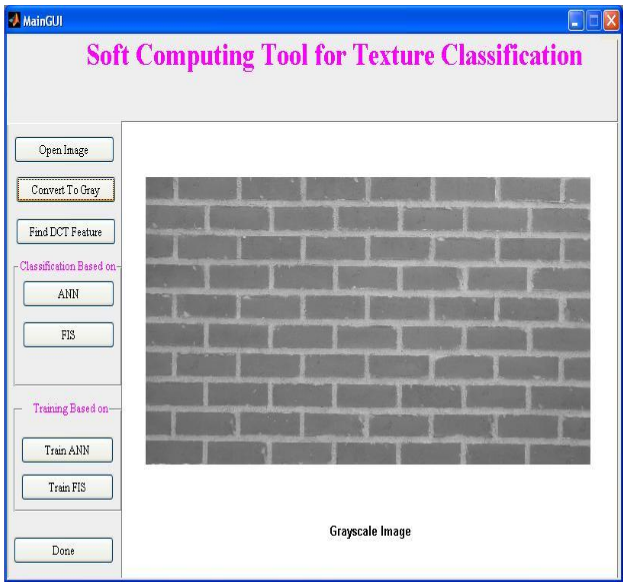


Figure 6: Converting original Brick Image to grayscale

Step 5:- Finding the DCT coefficients of an image.

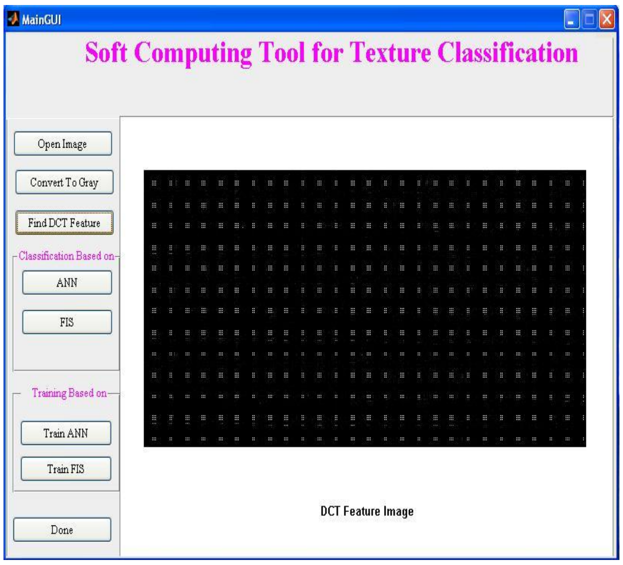


Figure 7: DCT coefficients of grayscale image

When Find DCT Feature button in GUI is click, then this gives call to a callback function written in maingui.m file. In callback function, firstly the input query image is converted into an 8-by-8 sized sub images. Then DCT function is individually applied on that sub images. Also, iterative loop is used to find features from that sub images in a zig-zag fashion. Following code is going illustrate that.

```
for j=1:8:u-8
    for k=1:8:v-8
        d1=dct2(img(j:j+7,k:k+7));
        feature=[feature; d1(1,1) d1(1,2) d1(2,1) d1(3,1)
                d1(2,2) d1(1,3) d1(1,4) d1(2,3) d1(3,2)];
        dctimg(j:j+7,k:k+7)=d1;
    end
end
```

Step 6:- Classify the image using neural network.

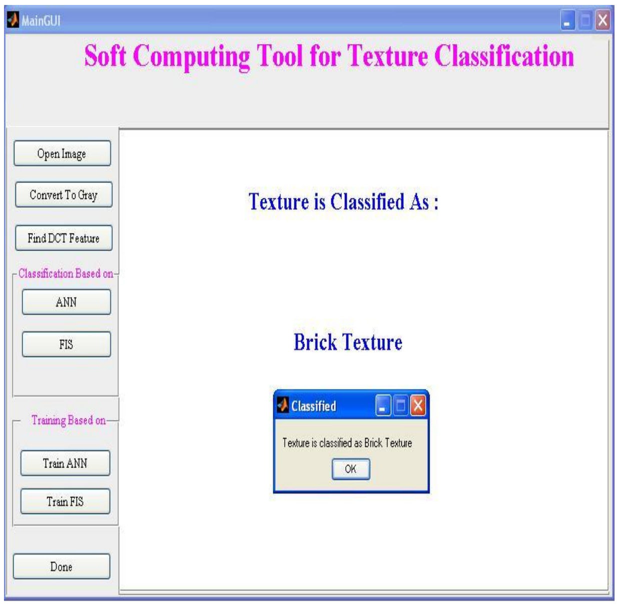


Figure 8:Classification of an image using neural network

When Classification Based on ANN button in GUI is click, then this gives call to a callback function written in maingui.m file. In callback function, it first give call to load neuralnet & load targetname. Depending upon features matching, it will classify an image into any one of the target name.

Step 7:- Classify the image using ANFIS.

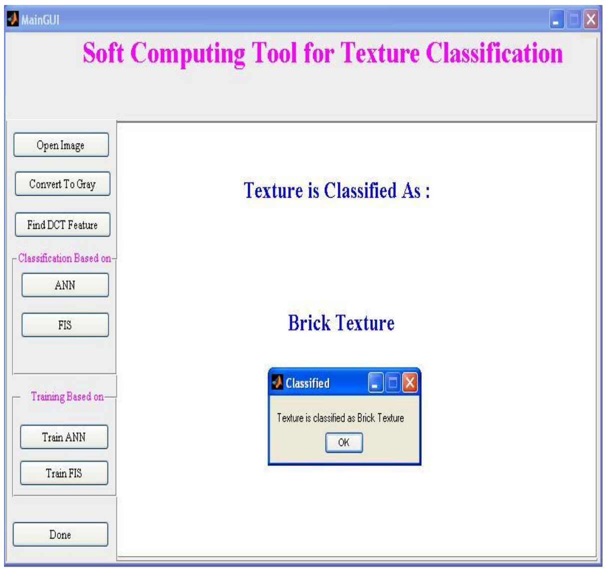


Figure 9: Classification of an image using ANFIS

When Classification Based on ANFIS button in GUI is click, then this gives call to a callback function written in maingui.m file. In callback function, it first give call to load fismat. Depending upon features matching, it will classify an image into any one of the target name.

Here, only brick classification is show using snapshots. Similar process has to be followed for meral & rural images.

X. TEST RESULT

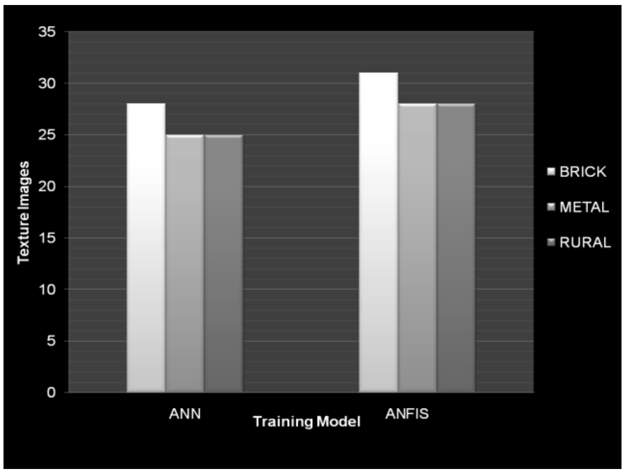


Figure 10: Performance of ANN & ANFIS based on classification efficiency

Test Result shows that for 35 texture image that were used for testing purposes, ANN is used to classify correctly 28 Brick, 25 Metal and 25 Rural texture images. Depending on that its classification efficiency is 74.28 %.Similarly, AFIS is used to classify correctly 31 Brick, 28 Metal and 28 Rural texture images. Depending on that its classification efficiency is 82.85 %.

Table 1:-Reliability of Classification

	ANN	ANFIS
Brick	28	31
Metal	25	28
Rural	25	28

Thus, it is seen from Table 1 that the classification efficiency of FIS is 82.85 % which is considerably greater than the classification efficiency of ANN that is 74.28 %.

XI. CONCLUSIONS

This paper attempted to classify 3 different types of textures using artificial neural networks and Evolving Fuzzy Neural Network (EFuNN).

- To find texture features, DCT coefficients are used.
- DCT coefficients do not require additional complex computation for feature extraction.
- As the high frequency coefficient is less sensitive to human visual systems, first few coefficients of each block is constructed.
- EFuNN is less computationally expensive as compared to neural networks.
- EFuNN adopts a one-pass (one epoch) training technique, which is highly suitable for online learning.
- The proposed prediction models based on soft computing on the other hand are easy to implement.
- EFuNN outperformed the neural network with the best classification (82.85%).
- EFuNN produce desirable mapping functions by training on the given data set.
- The considered connectionist models are very robust & capable of handling noisy.
- Some possible applications applying to our technique are detection of machined surface defects in industry, classification of satellite images in order to landmark the terrain, recovery topography and texture mapping.

REFERENCES

1. Pankaj H. Chandankhede, Parag V. Puranik, and P. R. Bajaj, “Design Approach of Texture Classification using Discrete Cosine Transform with Soft Computing Tool”, IJRTET, Vol. 05, No. 01, Mar 2011.
2. Hee-Jung Bae and Sung-Hwan Jung, “Image Retrieval Using Texture Based on DCT,” Proceedings of ICICS’ 97, Singapore, Vol. 2. 1997, pp. 1065-1068.
3. Salgado, M.F.P., Caminhas, W.M.;Menezes, B.R., “Soft computing approaches in reliability modeling and analysis of repairable systems”, In Proceeding of the Reliability and
4. Maintainability Symposium (RAMS), 2010.
5. Al Nadi, D.A., Mansour, A.M. “Independent Component Analysis (ICA) for texture classification” In Proceedings of 5th International Multi-Conference on the Systems, Signals and Devices, 2008.
6. M. N. Shirazi, H. Noda and N. Takao, “Texture modeling and classification in wavelet feature space”. In Proceedings of the International Conference on Image Processing. Vol. 1, pp. 272-5, 2000.
7. J. R. Smith and S. Chang, “Transform features for texture classification and discrimination in large image databases”. In Proceedings of ICIP-94, vol. 3, pp. 407-11, 1994.
8. G. K. Wallace, “Overview of the JPEG still image compression standard,” SPIE, Vol. 1244, 1990, pp. 220- 233.
9. D. J. Le Gall, “The MPEG video compression algorithm: a review,” SPIE Vol. 1452, 1991, pp. 444- 457.
10. F. Borko, “video and image processing in multimedia systems”, Kluwer Academic publishers, 1995, pp 225-249.
11. T. Hashiyama, T. Furuhashi, and Y. Uchikawa. “A Decision making Model Using a Fuzzy Neural Network”. In Proceedings of the 2nd International Conference on Fuzzy Logic & Neural Networks, Iizuka, Japan, 1992.
12. Sulzberger SM, Tschicholg-Gurman NN, Vestli SJ, “FUN: Optimization of Fuzzy Rule Based System Using Neural Networks”, In Processing of IEEE Conference on Neural Networks, San Francisco, pp 312-316, March 1993.
13. R. Jang. “ANFIS: Adaptive Network-Based Fuzzy Inference System”. IEEE Trans. on Syst., Man, Cybernetics, 23(3):665-685, 1993.
14. Abraham A & Nath B, “Designing Optimal Neuro-Fuzzy Systems for Intelligent Control”, In processing of the Sixth International Conference on Control Automation Robotics
15. Computer Vision, (ICARCV 2000), Singapore, December
16. 2000.
17. R. E. Hamdi, Njah M. and Chtourou M. “Multilayer perceptron training using an evolutionary algorithm”, Int. J. Modelling, Identification and Control, Vol. 5, No. 4, 2008.
18. J. A. Freeman and D. M. Skapura, “Neural networks algorithms, applications, and programming techniques”, Addison-Wesley, Massachusetts, 1992.
19. T. Raden and H. Husoy, “Filtering for texture classification: A comparative study”, IEEE Trans. on PAMI, vol. 21, no. 4, 1999.
20. Juang C. F. and Lin C. T. “An self-constructing neural fuzzy inference network and its applications”, IEEE trans. Fuzzy Syst. vol. 6, no.1, pp. 1231, 1998.
21. Raghu, P.P, Yegnanarayana, B., “Texture classification using a probabilistic neural network and constraint satisfaction model”, In Proceedings of the IEEE International Conference on neural network, 1996.

AUTHORS PROFILE

Pankaj H. Chandankhede, Ph.D. (Persuing), M.Tech(Electronics), B.E.(Electronics & Telecommunication). He is currently working as a Assistant Professor in G.H.Raisoni College of Engineeing, Nagpur, Maharashtra, India.