Texture Classification Based on DCT and Soft Computing

Golam Sorwar, Ajith Abraham and Laurence S Dooley

School of Computing and Information Technology
Monash University (Gippsland Campus)
Churchill, Victoria 3842, Australia
Email:{Golam.Sorwar,Ajith.Abraham, Laurence.Dooley}@infotech.monash.edu.au

Abstract: Classification of texture pattern is one of the most important problems in pattern recognition. In this paper, we present a classification method based on the Discrete Cosine Transform (DCT) coefficients of texture image. As the DCT works on gray level image, the color scheme of each image is transformed into gray levels. For classifying the images with DCT we used two popular soft computing techniques namely neurocomputing and neuro-fuzzy computing. We used a feedforward neural network trained using the backpropagation learning and an evolving fuzzy neural network to classify the textures. The soft computing models were trained using 80% of the texture data and remaining was used for testing and validation purposes. A performance comparison was made among the soft computing models for the texture classification problem. We also analyzed the effects of prolonged training of neural networks. It is observed that the proposed neuro-fuzzy model performed better than neural network.

Keywords: Texture classification, DCT, Neurocomputing, Neuro-Fuzzy Computing.

1. Introduction

Texture as a primitive visual cue has been studied for a long time. Various techniques have been developed for texture segmentation, texture classification and texture synthesis. Although texture analysis has a long history, its applications to real image data have been limited to-date. An important and emerging application where texture analysis can make a significant contribution is the area of content-based retrieval in large image and video databases. Using texture as a visual feature, one can query a database to retrieve similar patterns. Texture Classification and segmentation schemes are very important in answering such queries.

Most exiting approaches to texture feature extraction use statistical methods. For the analysis of a texture image, it requires large storage space and a lot of computation time to calculate the matrix of features such SGLDM (Spatial Gray Level Dependence Matrix), NGLDM (Neighboring Gray Level Dependence Matrix) [1] etc. In spite of the large size of each matrix, a set of their scalar feature calculated from the matrix is not efficient to represent the characteristics of image content.

In general, neighboring pixels within an image tend to be highly correlated. As such, it is desired to use an invertible transform to concentrate randomness into fewer, decorrelated parameters. The Discrete Cosine Transform (DCT) has been shown to be near optimal for a large class of images in energy concentration and decorrelating. It has been adopted the JPEG and MPEG coding standard [2][3]. The DCT decomposes the signal into underlying spatial frequencies, which then allow further processing techniques to reduce the precision of the DCT coefficients consistent with the Human Visual System (HVS) model. Such the DCT coefficients of an image tend themselves as a new feature, which have the ability to represent the regularity, complexity and some texture features of an image [4], it can be directly applied to image data in the compressed domain. This may be a way to solve the large storage space problem and the computational complexity of the existing methods.

Soft computing was first proposed by Zadeh [11] to construct new generation computationally intelligent hybrid systems consisting of neural networks, fuzzy inference system, approximate reasoning and derivative free optimization techniques. It is well known that the intelligent systems, which can provide human like expertise such as domain knowledge, uncertain reasoning, and adaptation to a noisy and time varying environment, are important in tackling practical computing problems. In contrast with conventional artificial intelligence techniques which only deal with precision, certainty and rigor the guiding principle of hybrid systems is to exploit the tolerance for imprecision, uncertainty, low solution cost, robustness, partial truth to achieve tractability, and better rapport with reality.

In our research, we used an artificial neural network trained using backpropagation algorithm and an evolving fuzzy neural network (neuro-fuzzy system) [9] for classifying the texture data. The soft computing models were evaluated based on their classification efficiency of the different texture data sets. We also evaluated the performance of the neural network by increasing the training epochs. In section 2, we present DCT transform followed by texture feature extraction in section 3. In Section 4 and 5 we present some basic theoretical aspects of neural networks and neuro-fuzzy systems followed by experimentation set up in section 6. Some discussions and conclusions are provided towards the end.

2. Block DCT-based Transform

Most existing approach to texture feature extraction use statistical methods. For the analysis of a texture image, it requires large storage space and a lot of computational time to calculate the matrix of features. For solving the problems, some researchers proposed to use DCT [4] for texture representation. The block diagram of the proposed one is shown in Figure 1.

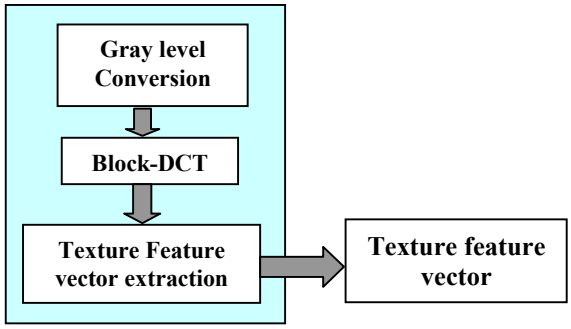


Figure 1. The block diagram of the texture feature extraction method.

For the DCT transform, we convert an RGB image into gray level image. For spatial localization, we then use the block-based DCT transformation. Each image is divided into $N*N$ sized sub-blocks. The two dimensional DCT can be written in terms of pixel values $f(i, j)$ for $i, j=0,1,\ldots,N-1$ and the frequency-domain transform coefficients $F(u, v)$:

$$F(u,v) = \frac{1}{\sqrt{2N}} C(u)C(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j)$$

$$\times \cos\left[\frac{(2i+1)u\pi}{2N}\right] \cdot \cos\left[\frac{(2j+1)v\pi}{2N}\right] \quad (1)$$

for $u,v = 0,1,\ldots,N-1$

where

$$c(x) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for x=0} \\ 1 & \text{otherwise} \end{cases}$$

The inverse DCT transform is given by

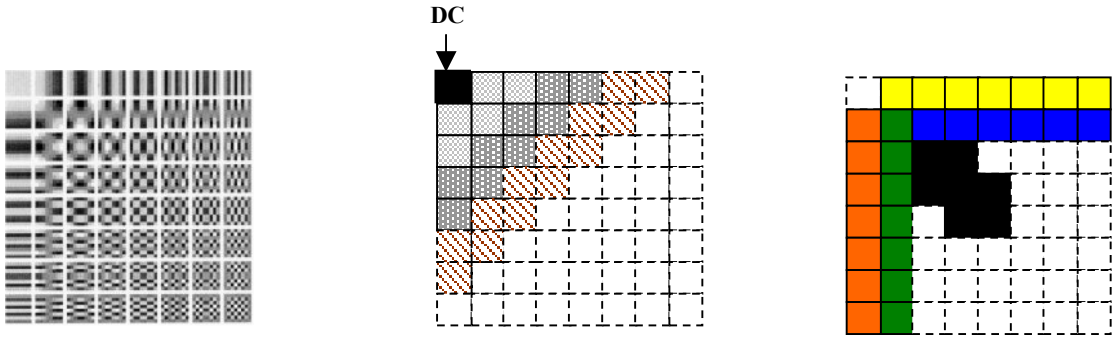
$$f(i,j) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} c(u)c(v)F(u,v)$$

$$\times \cos\left[\frac{(2i+1)u\pi}{2N}\right].\cos\left[\frac{(2j+1)v\pi}{2N}\right] \quad (2)$$

for i,j =0.1,…, N-1.

3. Texture Feature Vector Extraction

For efficient texture feature extraction, we use some DCT coefficients in compressed domain as the feature vectors. Each sub block contains one DC coefficients and other AC coefficients as Figure 2.



(a) **(b)** **(c)**

Figure 2. **(a)** DCT Basis Pattern information **(b)** Vector element from frequency components **(c)** Vector elements from directional

Since it is well known that the HVS is less sensitive to errors in high frequency coefficients than it is for lower frequencies component of DCT, we extract the feature set of 9 vector components in which first one is DC coefficient of each sub-block which represents the average energy or intensity of the block and other 8 AC coefficients which represent some different pattern of image variation and directional information of the texture; for example, the coefficients of the most upper region and those of the most left region in a DCT transform domain represent some vertical and horizontal edge information, respectively in Figure 2(c).

4. Artificial Neural Network

Neural networks are computer algorithms inspired by the way information is processed in the nervous system [5]. An important difference between neural networks and other AI techniques is their ability to learn. The network "learns" by adjusting the interconnections (called weights) between layers. When the network is adequately trained, it is able to generalize relevant output for a set of input data. A valuable property of neural networks is that of generalisation, whereby a trained neural network is able to provide a correct matching in the form of output data for a set of previously unseen input data. Learning typically occurs by example through training, where the training algorithm iteratively adjusts the connection weights (synapses). Backpropagation (BP) is one of the most famous training algorithms for multilayer perceptrons. BP is a gradient descent technique to minimize the error E for a particular training pattern. For adjusting the weight (w_{ij}) from the i-th input unit to the j-th output, in the batched mode variant the descent is based on the gradient ∇E ($\frac{\delta E}{\delta w_{ij}}$) for the total training set:

$$\Delta w_{ij}(n) = -\varepsilon * \frac{\delta E}{\delta w_{ij}} + \alpha * \Delta w_{ij}(n-1) \quad (3)$$

The gradient gives the direction of error E. The parameters ε and α are the learning rate and momentum respectively [6].

5. Neuro-Fuzzy Systems

We define a neuro-fuzzy system [7] as a combination of ANN and Fuzzy Inference System (FIS) in such a way that neural network learning algorithms are used to determine the parameters of FIS [8]. An even more important aspect is that the system should always be interpretable in terms of fuzzy *if-then* rules, because it is based on the fuzzy system reflecting vague knowledge. We used Evolving Fuzzy Neural Network [9] implementing a Mamdani type FIS [10] and all nodes are

created during learning. EFuNN have a five-layer structure as shown in Figure 4. The input layer represents input variables. The second layer of nodes represents fuzzy quantification of each input variable space. Each input variable is represented here by a group of spatially arranged neurons to represent a fuzzy quantization of this variable. Different membership functions can be attached to these neurons (triangular, Gaussian, etc.). The nodes representing membership functions can be modified during learning. New neurons can evolve in this layer if, for a given input vector, the corresponding variable value does not belong to any of the existing MF to a degree greater than a membership threshold.

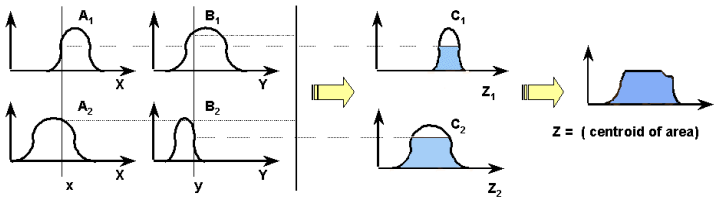


Figure 3. Mamdani fuzzy inference system

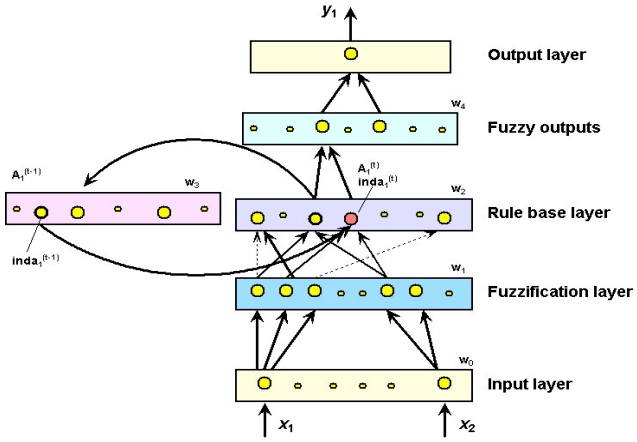


Figure 4. Architecture of EFuNN

The third layer contains rule nodes that evolve through hybrid supervised/unsupervised learning. The rule nodes represent prototypes of input-output data associations, graphically represented as an association of hyper-spheres from the fuzzy input and fuzzy output spaces. Each rule node *r* is defined by two vectors of connection weights – $W_1(r)$ and $W_2(r)$, the latter being adjusted through supervised learning based on the output error, and the former being adjusted through unsupervised learning based on similarity measure within a local area of the input problem space. The fourth layer of neurons represents fuzzy quantification for the output variables. The fifth layer represents the real values for the output variables. EFuNN evolving algorithm used in our experimentation was adapted from [9].

6. Experimentation Setup and Results

Training and Testing Data

In our research, we attempted to classify 3 different types of textures using soft computing techniques. We used DCT coefficients to represent the different textures. Each texture image was represented by 327 DCT coefficients. Our texture database consisted of 240 different textures and we manually classified into three different classes (*brick*, *metal* and *rural*). 192 texture datasets were used for training the soft computing models and remaining 48 texture datasets were used for testing purposes.

While the proposed neuro-fuzzy model was capable of determining the architecture automatically, we had to do some initial experiments to determine the architecture (number of hidden neurons and number of layers) of the neural network. After a trial and error approach we found that the neural network was giving good generalization performance when we had 2 hidden layers with 90 neurons each. In the following sections we report the details of our experimentations with neural networks and neuro-fuzzy models.

EFuNN training

We used 4 membership functions for each input variable and the following evolving parameters: sensitivity threshold *Sthr*=0.99, error threshold *Errthr*=0.001. EFuNN training has created 162 rule nodes as shown in Figure 5. Empirical results are reported in Table 1 and 2.

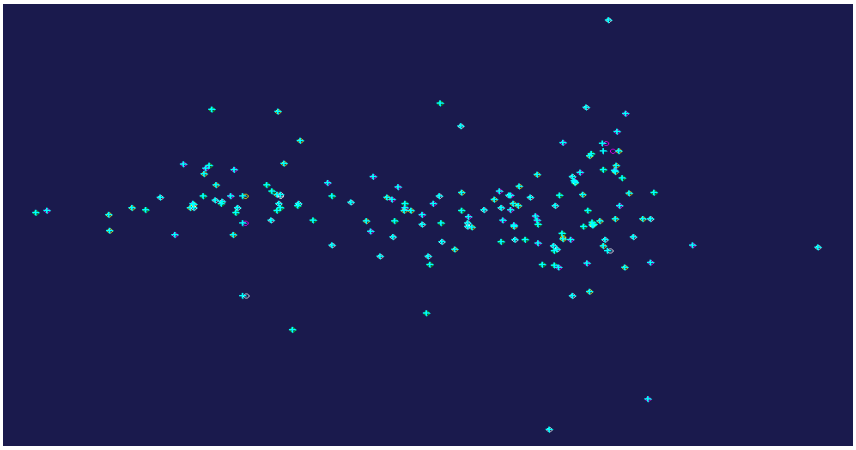


Figure 5. Learned rule nodes by EFuNN learning

ANN training

We used a neural network trained using backpropagation algorithm. We used 1 input layer, 2 hidden layers and an output layer [327-90-90-3]. Input layer consists of 327 neurons corresponding to the input variables. The first and second hidden layer consists of 90 neurons each with tanh-sigmoidal activation function. The initial learning rate and momentum were set as 0.05 and 0.1 respectively. Training errors (RMSE) and performance achieved after 5000, 15,000, 20,000 and 40,000 epochs are reported in Table 1 and 2. Approximate computational load in Giga Flops is reported in Table 1 and is graphically depicted in Figure 6.

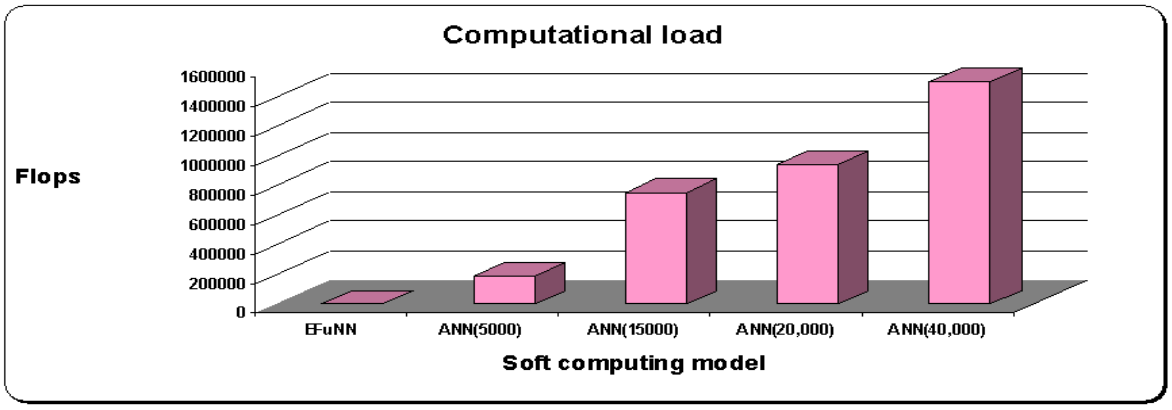


Figure 6. Computational load of soft computing models

Test results

Table 2 summarizes the comparative performance of EFuNN and ANN. The best classification was obtained using EFuNN (88%) and was 85.4% for neural networks.

Table 1. Training performance of Soft computing models

	EFuNN	ANN (5000 ep)	ANN (15000 ep)	ANN (20000 ep)	ANN (40,000 ep)
RMSE (Training)	$0.2e^{-003}$	$3.9e^{-003}$	$1.4e^{-004}$	$9.4e^{-005}$	$8.5e^{-005}$
Giga Flops	152	187860	751100	939000	1502200

Table 2. Test results and performance comparison of texture classification

		EFuNN	ANN (5000 ep)	ANN (15000 ep)	ANN (20000 ep)	ANN (40,000 ep)
Brick **(16 Nos)**	A_1	15	14	14	12	12
	A_2	1	2	2	4	4
Metal **(16 Nos)**	B_1	16	13	14	15	15
	B_2	0	3	2	1	1
Rural **(16 Nos)**	C_1	11	12	11	14	
	C_2	5	4	5	2	2
Total (48 Nos)	$X = (A_1+B_1+C_1)$	42	39	39	41	41
	$Y = (A_2+B_2+C_2)$	6	9	9	7	7
***Reliability of classification**		88 %	81.25 %	81.25 %	85.4 %	85.4 %

*Reliability = $\left(\frac{X}{48}\right) \times 100$

7. Conclusions

In this paper, we attempted to classify 3 different types of textures using artificial neural networks and Evolving Fuzzy Neural Network (EFuNN). For texture feature we consider the DCT coefficient, which does not require additional complex computation for feature extraction. As the high frequency coefficient is less sensitive to human visual systems, we constructed a feature matrix considering the first few coefficients of each block. EFuNN outperformed the neural network with the best reliability of classification (88%). As depicted in Table 1, EFuNN is less computational expensive while compared to neural networks. EFuNN adopts a one-pass (one epoch) training technique, which is highly suitable for online learning. Hence online training can incorporate further knowledge very easily. We also studied the generalization performance of BP training when the training epochs were increased from 5000 epochs to 40,000 epochs. When the number of epochs were increased, it was interesting to note the continuous reduction of the training error (RMSE) but the generalization error (classification accuracy) however tends to settle after 20,000 epochs. Compared to ANN, an important advantage of neuro-fuzzy model is its reasoning ability (*if-then* rules) of any particular state [12].

The proposed prediction models based on soft computing on the other hand are easy to implement and produces desirable mapping function by training on the given data set. Moreover, the considered connectionist models are very robust, capable of handling the noisy and approximate data and might be more reliable in worst situations. Choosing suitable parameters for the soft computing models is more or less a trial and error approach. Optimal results will depend on the selection of parameters. Selection of optimal parameters may be formulated as a evolutionary search to make SC models fully adaptable and optimal according to the requirement.

References

[1] Furht Borko, Video and Image processing in multimedia Systems, Kluwer Academic publishers, 1995, pp 225-249.

[2] Wallace G K, " Overview of the JPEG still Image Compression standard," SPIE, Vol. 1244, 1990, pp. 220-233.

[3] Le Gall D J, " The MPEG Video Compression Algorithm: A review," SPIE Vol. 1452, 1991, pp. 444-457.

[4] Sang-Mi Lee, Hee_Jung Bae, and Sung-Hwan Jung, "Efficient Content-Based Image Retrieval Methods Using Color and Texture," ETRI Journal, Vol. 20, pp. 272-283, 1998.

[5] Zurada J M, Introduction to Artificial Neural Systems, PWS Pub Co, 1992.

[6] Abraham A and Nath B, Optimal Design of Neural Nets Using Hybrid Algorithms, In proceedings of 6th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2000), pp. 510-520, 2000.

[7] Abraham A and Nath B, Designing Optimal Neuro-Fuzzy Systems for Intelligent Control, The Sixth International Conference on Control, Automation, Robotics and Vision, (ICARCV 2000), December 2000.

[8] Cherkassky V, Fuzzy Inference Systems: A Critical Review, Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications, Kayak O, Zadeh LA et al (Eds.), Springer, pp.177-197, 1998.

[9] Kasabov N, Evolving Fuzzy Neural Networks - Algorithms, Applications and Biological Motivation, in Yamakawa T and Matsumoto G (Eds), Methodologies for the Conception, Design and Application of Soft Computing, World Scientific, pp. 271-274, 1998.

[10] Mamdani E H and Assilian S, An experiment in Linguistic Synthesis with a Fuzzy Logic Controller, International Journal of Man-Machine Studies, Vol. 7, No.1, pp. 1-13, 1975.

[11] Zadeh LA, Roles of Soft Computing and Fuzzy Logic in the Conception, Design and Deployment of Information/Intelligent Systems, Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications, O Kaynak, LA Zadeh, B Turksen, IJ Rudas (Eds.), pp1-9, 1998.

[12] Kasabov, N. and Woodford B. Rule Insertion and Rule Extraction from Evolving Fuzzy Neural Networks: Algorithms and Applications for Building Adaptive, Intelligent Expert Systems, I*n* Proceedings of the FUZZ-IEEE'99 International Conference on Fuzzy Systems*,* Seoul, Korea, 1999.