Classification of image using a genetic general neural decision tree

Vijay Pal Dhaka and Manoj Kumar Sharma*

Jaipur National University,
Jagatpura Jaipur Rajasthan,
Jaipur – 302017, India
Email: vijaypal.dhaka@gmail.com
Email: manoj186@yahoo.co.in
*Corresponding author

Abstract: The decision trees are the techniques of machine learning, classification and recognition of images. The neural trees provide a solution for combining both decision tree and neural network. The neural tree node containing a neural network is called neural node. In the proposed study a genetic algorithm-based decision tree is formed to extract required number of branches for decision nodes in decision tree. In continuation of this, a genetic algorithm-based neural network is formed to extract required number of output nodes and hidden nodes in the neural network. Finally, by combination of the general decision tree and general neural tree, a general neural decision tree has been formed to design a decision node for each internal node in general neural decision tree. In the experimental phase general neural decision tree overrun the performance of decision tree and neural tree for classification of an image.

Keywords: neural decision trees; genetic algorithm; fuzzy; neural node; applied pattern.

Reference to this paper should be made as follows: Dhaka, V.P. and Sharma, M.K. (2015) 'Classification of image using a genetic general neural decision tree', *Int. J. Applied Pattern Recognition*, Vol. 2, No. 1, pp.76–95.

Biographical notes: Vijay Pal Dhaka is a young and dynamic technocrat with 13 years of intensive experience in industry and academia. He received his MTech and PhD in Computer Science from Dr. B.R. Ambedkar University, Agra. He has more than 36 publications in international journals and paper presentations in 27 conferences/seminars; he always strives to achieve academic excellence. He has been awarded by employers with 'Employee of the Quarter Award', 'Mentor of the Year Award' and with letters of appreciations for his commitment, advocacy and mentorship. He has organised several conferences, seminars and workshops.

Manoj Kumar Sharma is a young and dynamic technocrat with ten years of intensive experience in industry and academia. He received his MTech and PhD in Computer Science from Jaipur National University, Jaipur. He has more than 20 publications in international journals and paper presentations in 15 conferences/seminars; he always strives to achieve academic excellence. He has organised several conferences, seminars and workshops.

1 Introduction

Decision trees are the techniques of machine learning, recognition, classification systems. Heumann (2011) demonstrates the use of the new Worldview-2 sensor, object-based image analysis (OBIA) and support vector machine (SVM). Naeem et al. (2013) applied decision tree to classify the frontal facial images basis of their gender. Sharma et al. (2013) developed a decision tree classification algorithm using open source support for classification of remotely sensed satellite data. Pedrycz and Sosnowski (2000) preceived the data as a collection of information granules.

Witold and Zenon (2005) proposed the C-fuzzy decision tree (CFDT) derived from the well-known fuzzy C-means (FCM) algorithm. The CFDT is based on multivariable decision trees. Shukla and Tiwari (2012) proposed the fuzzy decision tree based on the genetic algorithm (GA). In GCFDT, the centre of each cluster is a real number, which is directly encoded in the bit string of GA. The length of bit strings is fixed since the number of clusters generated by GA is fixed. Yang (2010) proposed the length of bit strings is set to the size of the training dataset in the GA. In the DTs, the method selects a node, t, to split at a time, and then the clustering algorithm is applied to classify the training samples contained in t to generate many clusters. Figure 1(a) shows decision node, t, in DT. Figure 1(b) shows neural node t, in NT. In the neural node t, the neural network consists of r output nodes in the output layer. Utgoff (1998) proposed the perceptron tree. Sirat and Nadal (1990) presented a binary structure of NT for solving two-class problems. Foresti and Micheloni (2002) proposed a generalised neural tree, in which the learning rule is calculated by minimising a cost function which represents a measure of the overall classification error of the tree. Guo and Gelfand (1992) proposed a neural tree with multilayer perceptron (MLP) at the internal nodes. This model allows to divide the input space with arbitrary hypersurfaces. Foresti (2004) proposed the adaptive high order neural tree (AHNT). Nodes are high-order perceptrons whose order depends on the complexity of the training set reaching that node. Maji (2008) proposed the MLP is used to design the neural tree, namely NNtree. Micheloni et al. (2012) proposed the balanced neural tree (BNT) to reduce tree size and improve classification with respect to classical neural tree.

Figure 1 Decision nodes and neural nodes, (a) decision tree t (b) neural tree t

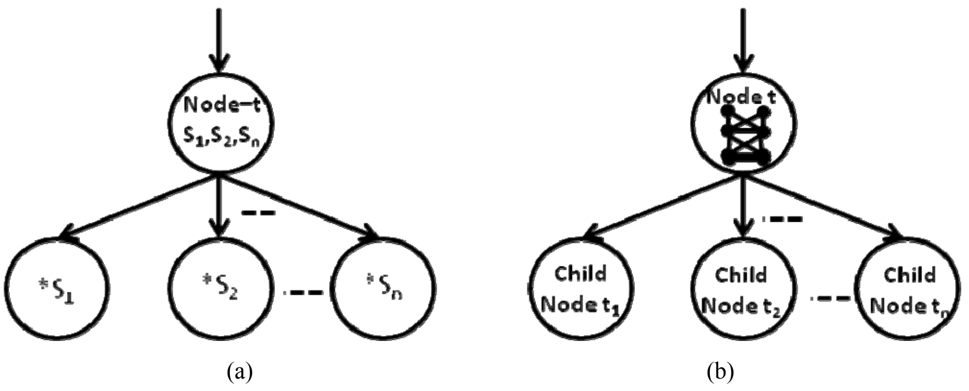
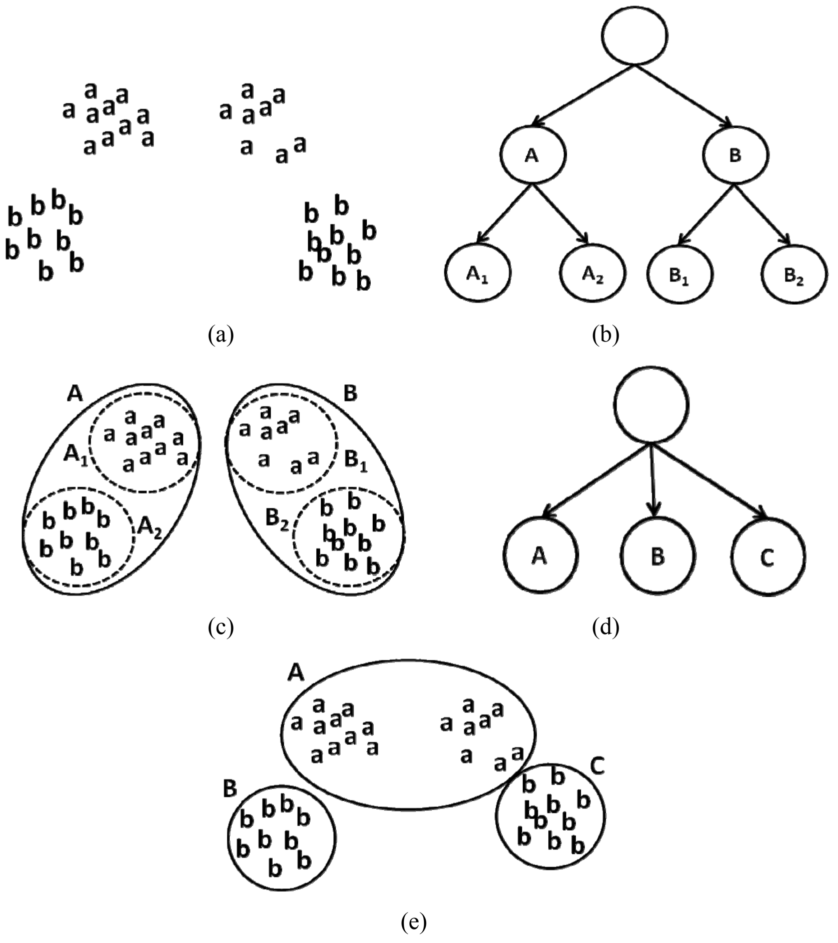


Figure 2 An example to illustrate that the proposed GDT outperforms the fixed-branch DTs, (a) original dataset (b) binary decision tree generated by the FCM (c) clustering result obtained by the FCM (d) our proposed GDT (e) clustering result obtained by the proposed GADNT

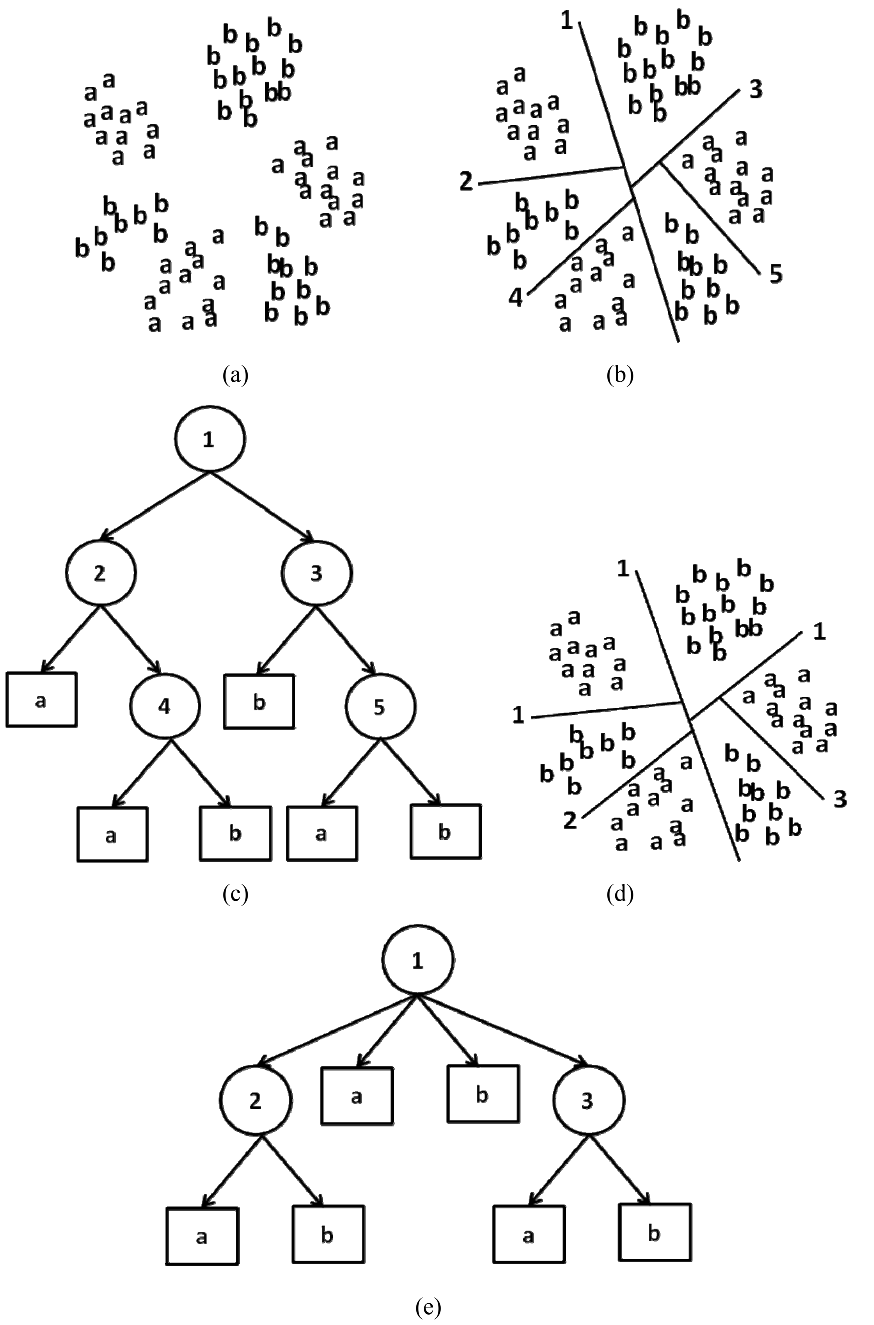


2 Methodology

The proposed method consists of three stages: GA-based decision tree, GA-based node tree and finally combines both decision and node tree in genetic decision node tree. In all of these three stages, GA decision node is designed first to search the appropriate number of branches of all decision nodes. Figure 2(a) shows the fundamental dataset consists of two classes labelled by the symbols ‘a’ and ‘b’. In Figure 2(b), the FCM algorithm selects two samples labelled ‘b’ as the initial seeds to generate two clusters, A and B. Then, each of these two clusters continues to be divided into another two clusters, respectively. Figure 2(c) shows the corresponding binary decision tree. Figure 2(d) shows the clustering result by the proposed GADNT, and the corresponding GDT is generated as shown in Figure 2(e). Clearly, the FCM algorithm generates the redundant clusters, and then the computing complexity of the tree in Figure 2(c) is greater than that of the tree in

Figure 2(e). Thus, the performance of GADNT is enhanced and the GADNT outperforms the GA proposed (Yang, 2010).

Figure 3 An example to illustrate that the proposed GNT and other NTs, (a) original dataset (b) partition of the dataset by the traditional methods (c) the binary NTs (d) partition of dataset by our proposed GANNT (e) our proposed GNT

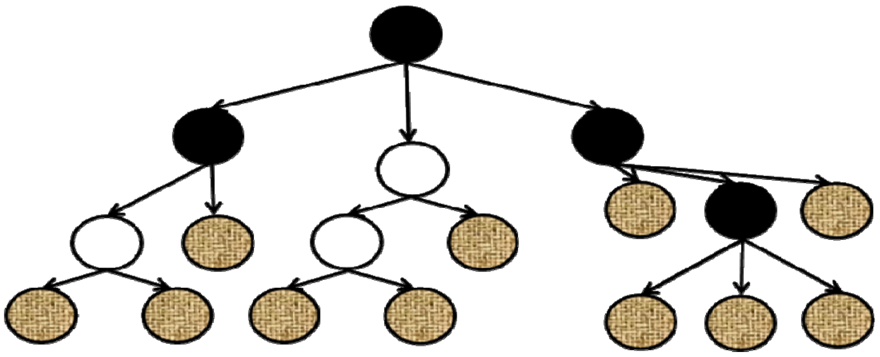


The GA-based neural network can automatically search the number of hidden and output nodes according to the computing complexity and classification error rate of the general neural tree. Figure 3(a) shows the original dataset that consists of two classes. Figure 3(c) shows the partition of the dataset when the traditional NT is designed. The corresponding NT is shown in Figure 3(b) and produced five neural networks five internal nodes in NT. If an input sample is classified, it needs to be calculated with an average of 3.2 neural networks. Figure 3(e) shows the partition of the dataset by our proposed GANNT and the corresponding GNT is shown in Figure 3(d). The input sample needs to be calculated with an average of two neural networks.

In the final stage, general neural decision tree is proposed, which is designed by combination of general decision tree and general neural tree. The general neural decision tree consists of decision nodes and neural nodes.

Figure 4 shows an example to illustrate the general neural decision tree. In the general neural decision tree, designing of a decision node is referred due to less computational complexity of decision node.

Figure 4 An example of GNDT (see online version for colours)\



Notes: Black circle: neural node, white circle: decision node, grey circle: leaf node

3 Design of the GADNT

The GADNT is proposed to design the DNs in GDT. Let T be the general decision tree with t leaf node that has m_t training samples, $b_z = (b_z^1, b_z^2, \ldots b_z^f) \in R^f$ for $1 \le z \le m_t$, in T. The GADNT is a genetic clustering algorithm to search the clustering result in the dataset.

At first a threshold ϕ, is defined to organise the number of child nodes of node t and the number of child nodes of node t should be less than or equal to the threshold ϕ. Then, a population of N strings is randomly generated and the length of each string is smaller than or equal to the threshold ϕ. Let R_x denote that the xth string in the population, and let the length of string R_x be l_x, $1 \le l_x \le \phi$. Then, the string R_x can be represented as $R_x = (r_1, r_2, \ldots r_{l_x})$, where $1 \le r_j \le m_t$ and $1 \le j \le l_x$. Each r_j is randomly generated in string R_x, and represents the corresponding training sample b_{r_j} that is regarded as a seed to generate a cluster. Each cluster can be designed as a child node of t in string R_x. Thus, string R_x generates l_x child nodes for node t. The method for generating a clustering from the seeds of string $R_x = (r_1, r_2, ...r_{l_x})$ is described as follows.

The string R_x includes two sets of input samples, $Q_1 = \{b_{r_j} \mid r_j \in R_x\}$ and $Q_2 = \{b_{r_i} \mid r_i \notin R_x\}$. Then, these l_x training samples, $b_{r_1}, b_{r_2}, ...b_{r_{l_x}}$, in Q_1 are used as the seeds to generate a set of clusters, $\text{CLUSTER} = \{C_1, C_2, ...C_{l_x}\}$. Initially, each cluster C_j contains only one input sample, $b_{r_j} \in Q_1$, and then the centre S_j of cluster C_j is set to b_{r_j}. Then, the training samples in Q_2 are considered one at a time and the membership between each training sample and the centres S_j, for $1 \le j \le l_x$, are calculated. The probability, $P(C_j, b_z)$, is used to measure the membership between the training sample, b_z, in Q_2 and cluster C_j in CLUSTER.

If $P(C_j, b_z) \ge P(C_k, b_z)$ for $1 \le k \le l_x$, the training sample, b_z, is classified to C_j. After all of the training samples in Q_2 have been classified, the centre, S_j, is updated and set to the average of training samples contained in C_j, for $1 \le j \le l_x$. The fitness of string R_x is defined as

$$\lambda_{GDT}(t_i) = \frac{\Delta R}{\Delta V} = \frac{R(T) - R(T')}{V(T) - V(T')} \tag{1}$$

$$\text{Fitness}(R_x) = \lambda_{GDT}(t) \tag{2}$$

where $\lambda_{GDT}(t)$ is the slope of the classification error rate and computing complexity between T and T'. If $\lambda_{GDT}(t)$ is large, the fitness of string R_x tends to be large.

The time complexity of GADNT it takes $O(Nm^2)$ time in the worst case and time complexity for G generations is $O(GNm^2)$.

5 Design of the GANNT

The GANNT is proposed to design the neural nodes in GNT. Let T be the GNT consists of t leaf node with m_t training samples, $b_z = (b_z^1, b_z^2, ...b_z^f) \in R^f$ for $1 \le z \le m_t$, in T. In the initialisation step of GANNT, a population of N strings is randomly generated and each string represent the different number of hidden and output nodes due to variant in string. The GANNT, having two thresholds, ϕ_1 and ϕ_2, to control the number of output nodes in the output layer. Suppose the size of the range $[\phi_1, \phi_2]$ is M, where $M = \phi_2 - \phi_1 + 1$. Then, these N strings are partitioned into M subsets, S_i for $1 \le i \le M$, and the strings in subset S_i are set to generate the same number of output nodes, O_i. Thus, $O_i = \phi_1 + (i - 1)$ for $1 \le i \le M$. Let R_x^y denote that the xth string in the subset, S_y, where $1 \le y \le M$, $O_y = \phi_1 + (y - 1)$ and $1 + \frac{N}{M}(y-1) \le x \le \frac{N}{M}y$. Let each input sample, $b_z = (b_z^1, b_z^2, ..., b_z^f) \in R^f$, contain f elements. The string, R_x^y, represents that the neural network has f input nodes, O_y output nods and h_x hidden nodes. The values of θ_j for $1 \le j \le h_x$ denote the activation thresholds. The values of $w_{i,j}$, for $1 \le i \le f$ and $1 \le j \le h_x$, indicate the weights between the input and hidden layer, and the values of $w_{j,k}$, for

$1 \le j \le h_x$ and $1 \le k \le O_y$, indicate the weights between the hidden and output layer. Then, the string, $R_x^y \in S_y$, is presented as follows.

$$R_x^y = \begin{pmatrix} h\_node_x(1), h\_node_x(2),. \\ ., h\_node_x(h_x) \end{pmatrix}$$

where

$$h\_node_x(j) = \left(\theta_j, w_{11}, ..., w_{fj}, v_{j1}, ..., v_{jO_y}\right)$$

for $1 \le j \le h_x$.

Let $R_1^1 \in S_1$ represent the neural network with two hidden nodes ($h_1 = 2$).

Then,

$$R_1^1 = \left(h\_node_1(1), h\_node_1(2)\right)$$

where

$$h\_node_1(1) = \left(\theta_1, w_{11}, w_{21}, v_{11}, v_{12}, v_{13}\right),$$
$$h\_node_1(2) = \left(\theta_2, w_{12}, w_{22}, v_{21}, v_{22}, v_{23}\right)$$

In the reproduction the fitness of the string depends on both the classification error rate and the computing complexity of the GNT. The fitness function of string R_x^y is defined as

$$\lambda_{GNT}(t_i) = \frac{\Delta R}{\Delta V} = \frac{R(T) - R(T')}{V(T') - V(T)} \tag{3}$$

$$\text{Fitness}\left(R_x^y\right) = \lambda_{GNT}(t) \tag{4}$$

where $\lambda_{GDT}(t)$ is the slope of the classification error rate and computing complexity between T and T'. The reproduction probability, $Pr(R_x^y)$, of the string R_x^y is defined as

$$Pr\left(R_x^y\right) = \frac{\text{Fitness}\left(R_x^y\right)}{\sum_{y=1}^{M} \sum_{R_x^y \in S_y} \text{Fitness}\left(R_x^y\right)} \tag{5}$$

If a string has a large reproduction probability, it has a high chance of being selected as the string in the next generation. In the crossover phase, the crossover operator is applied to the subsets, S_y for $1 \le y \le M$, respectively. If the crossover operator is applied to subset S_y, then a pair of strings, $R_l^y \in S_y$, is selected to do crossover operator in subset Sy. The crossover operator with two random integers (a = 2 and b = 3) is applied to the pair of strings, $\hat{R}_1^1$ and $\hat{R}_2^1$, are generated in the next generation. The pair of strings, R_1^1 and R_2^1, are shown as follows.

$$R_1^1 = \left(h\_node_1(1), \overset{a=2}{|} h\_node_1(2) \right)$$

where

$$h\_node_1(1) = (\theta_1, w_{11}, w_{21}, v_{11}, v_{12}, v_{13}),$$
$$h\_node_1(2) = (\theta_2, w_{12}, w_{22}, v_{21}, v_{22}, v_{23})$$
$$R_2^1 = \left(h\_node_2(1), h\_node_2(2), \overset{b=3}{|} h\_node_2(3) \right)$$

where

$$h\_node_2(1) = (\theta_1, w_{11}, w_{21}, v_{11}, v_{12}, v_{13}),$$
$$h\_node_2(2) = (\theta_2, w_{12}, w_{22}, v_{21}, v_{22}, v_{23})$$
$$h\_node_3(2) = (\theta_3, w_{13}, w_{23}, v_{31}, v_{32}, v_{33})$$

After the crossover operator is applied to R_1^1 and R_2^1, two new strings, $\hat{R}_1^1$ and $\hat{R}_2^1$, are generated as follows.

$$\hat{R}_1^1 = (h\_node_2(1), h\_node_2(2), h\_node_1(2))$$
$$\hat{R}_2^1 = (h\_node_2(1), h\_node_2(3))$$

5 Design of the GNDT

The GNDT is a tree-structured classifier that contains two kinds of internal nodes, the neural and decision nodes. That is, each internal node can be designed as a decision node or a neural node. The leaf nodes in the GNDT are only used to position the output space. Like both the GNT and GDT, the growing method of GNDT selects a leaf node to split at a time to grow the GNDT. Then, the computing complexity of the leaf node t_i, $V(t_i)$, is defined as

$$CD(t_j) = q_j \tag{6}$$

$$V(t_i) = \sum_{t_j \in L(t_i)} \rho(t_j) \tag{7}$$

where $\rho(t_j)$ is

$$V(T) = \sum_{t_i \in H} P(t_i) V(t_i) \tag{8}$$

where $CD(t_j)$ is defined as equation (6) and $CN(t_j)$ is defined as

$$CN(t_j) = fh_j + h_j r_j \tag{9}$$

Let $P(t_i)$ as follows.

$$\rho(t_j) = \begin{cases} CD(t_j), \text{ if } t_j \text{ is a decision node} \\ CN(t_j), \text{ if } t_j \text{ is a neural node} \end{cases} \tag{10}$$

probability on the training samples in the node t_i. Finally, the computing complexity of T, V(T), is defined as

$$V(T) = \sum_{t_i \in H} P(t_i) V(t_i) \tag{11}$$

The classification error rate of the GNDT is determined by the training samples contained in the leaf nodes. The classification error rate of T, R(T), is defined as that of both the GNT and GDT. The R(T) is defined as

$$R(T) = \sum_{t_i \in H} P(t_i) R(t_i) \tag{12}$$

If node t_i is regarded as a decision node, the GADNT is applied to node t_i. Let the tree T' indicate the tree T after node t_i is split into r_i child nodes: $t_{i,1}$, $t_{i,2}$, …, $t_{i,ri}$, by the GADNT. Then, the slope of the classification error rate and computing complexity between that of T and T' is:

$$\lambda_{GNDT}(t_i, T') = \frac{\Delta R}{\Delta V} = \frac{R(T) - R(T')}{V(T') - V(T)} \tag{13}$$

where R(T) is defined as equation (12) and V(T) is defined as

$$V(T) = \sum_{t_i \in H} P(t_i) V(t_i) \tag{14}$$

If node t_i is regarded as a neural node, the GANNT is applied to node t_i. Let the tree T'' indicate the tree T after node t_i is split into O_i child nodes, $t_{i,1}, t_{i,2}, ..., t_{i,o_i}$. Then, we also obtain $\lambda_{GNDT}(t_{i,}T'')$ as follows.

$$\lambda_{GNDT}(t_i, T'') = \frac{\Delta R}{\Delta V} = \frac{R(T) - R(T'')}{V(T'') - V(T)} \tag{15}$$

where R(T) is defined as equation (12) and V(T) is defined as equation (8). The node with the largest value of λ_{GNDT} is preferred when the GNDT is designed. Then, the value of $\hat{\lambda}_{GNDT}(t_i)$ is set as

$$\hat{\lambda}_{GNDT}(t_i) = \max\left\{\lambda_{GNDT}(t_{i,}T'), \lambda_{GNDT}(t_{i,}T'')\right\} \tag{16}$$

If $\lambda_{GNDT}(t_{i,}T') < \lambda_{GNDT}(t_{i,}T'')$, node t_i is designed as a neural node; otherwise node t_i is designed as a decision node.

Let the set H consist of the leaf nodes in T. After each leaf node in H is determined to be designed as a decision node or a neural node, we can obtain $\hat{\lambda}_{GNDT}(t_i)$.

$$k = \arg\max_{t_i \in H} \hat{\lambda}_{GNDT}(t_i) \tag{17}$$

Algorithm 1 Design_GNDT

```
Input: The storage limitation $\delta$, training dataset.
Output: The GNDT.
Step 1  Root node of tree T contain all of the training samples, H denote the set of leaf nodes
        in T. Initially, the set of H contains the root node. Set SPACE = 1.
Step 2  While SPACE < $\delta$
        Step 2.1  For each leaf node $t_i$ in H and $R(t_i) > 0$
                  Step 2.1.1  The GADNT is applied to design the node $t_i$ as a decision
                              node and generates ri child nodes of $t_i$. Let the tree $T'$
                              indicate the tree T after node $t_i$ is split into $r_i$ child nodes.
                              Calculate the value of $\lambda_{GNDT}(t_{i,}T')$ as equation (13).
                  Step 2.1.2  The GANNT is applied to design the node $t_i$ as a neural
                              node and generates $O_i$ child nodes of $t_i$. Let the tree $T''$
                              indicate the tree T after node $t_i$ is split into $O_i$ child nodes.
                              Calculate the value of $\lambda_{GNDT}(t_{i,}T'')$ as equation (15).
                  Step 2.1.3  Calculate as equation (16). If $\lambda_{GNDT}(t_{i,}T') < \lambda_{GNDT}(t_{i,}T'')$,
                              the node $t_i$ is designed as a neural node; otherwise, the
                              node $t_i$ is designed as a decision node.
        Step 2.2  Calculate $k = \arg\max_{t_i \in H} \lambda_{GDNT}(t_i)$ as equation (17). Let the node $t_k$ be
                  divided into w child nodes. The growing method selects the node $t_k$ to
                  split, and then the new decision tree $\hat{T}$ is generated. Delete the node $t_k$ in
                  the set of H and add these w new nodes in $\hat{T}$ to the set of H.
        Step 2.3  Set $T = \hat{T}$ and SPACE = SPACE + w.
Step 3  Output the GNDT, T
```

The testing algorithm of the GNDT is described as follows.

Algorithm 2 Test_GNDT

```
Input: The input sample X.
Output: The class of X.
Step 1  Let t* be the root node of the GNT.
Step 2  While t* is not a leaf node
        Step 2.1  If the node t* is a neural node, the sample X is the input to the neural
                  network, NET(t*). Let the sample X be classified to the k Output in the
                  NET(t*). Then, the input sample X towards the corresponding child node $t_k$
                  of the node t* in the GNDT. Goto Step 2.3.
        Step 2.2  If the node t* is a decision node, the sample X is compared with those
                  classification vectors contained in the node t*. Let the sample X and the
                  classification vector $S_k$ have a minimum Euclidean distance. Then, the
                  input sample X towards the corresponding child node $t_k$ of the node t* in
                  the GNDT.
        Step 2.3  Set $t^* = t_k$
Step 3  Output the representative class of node t*.
```

6 Experiments

6.1 Datasets

Three image datasets: COREL, satellite and PASCAL VOC2007 are used to test the performance of the proposed method and state-of-the-art methods (Le Borgne et al., 2007). We used 20 images per category for a total of 220 images for learning, while 1,817 images were used for testing. Each image size is 256 × 384 or 384 × 256. In the experiments, each image was divided into blocks of 16 × 16 pixels. Each block was then transformed by a Haar wavelet transform (Gonzalez and Woods, 1992) to obtain four subbands. The mean values (mv) and standard deviations (sd) of the four subbands were calculated as follows.

$$\mathrm{mv} = \frac{1}{N^2}\sum_{i,j=1}^{N} v(i,j) \tag{18}$$

$$\mathrm{sd} = \sqrt{\frac{1}{N^2}\sum_{i,j=1}^{N}\left[v(i,j) - \mathrm{mv}\right]^2} \tag{19}$$

where N denotes the size of the subband, which is set to 8 in this experiment, and v(i, j) denotes the wavelet coefficient at location (i, j) in the subband.

The satellite image dataset was taken from the UCI machine learning data repository (Asuncion and Newman, 2007). The satellite image dataset consists of 6,435 images belonging to six classes of images (red soil, cotton crop, grey soil, damp grey soil, soil with vegetation stubble, very damp grey soil), and the dataset is divided into training and test sets, with 4,435 examples in the training set and 2,000 in the test set. The database consists of the multi-spectral values of pixels in 3 × 3 neighbourhoods in a satellite image. The number of attributes is 36 (four spectral bands × nine pixels in a neighbourhood). Each pixel is an eight-bit binary word, with 0 corresponding to black and 255 to white.

The PASCAL VOC2007 is one of the most challenging datasets for image classification. The dataset has been split into 50% for training and 50% for testing. The distribution of images and objects by class are approximately equal across the training and test sets. In total there are 9,963 images in the PASCAL VOC2007 dataset.

6.2 Performance of the GDT over GADNT

The users set the value of the storage limitation, δ and five storage limitations are used to confirm the efficiency of the GDT for each dataset. Tables 1 to 3 list the performance of the GDT with the storage constraints. In the GDT, the GADNT, is proposed to generate the child nodes of each decision node. The parameters that are used in the GADNT are a population of 300, a crossover rate of 80% and a mutation rate of 5%. Five hundred generations are run and the best solution is retained. Also, the number of child nodes generated by the GADNT is less than, or equal to, the threshold ϕ that is given by the users. Thus, the length of bit strings is less than, or equal to, the threshold ϕ in the GADNT. From Tables 1 to 3, we get conclusions.

1 From the field 'AVE_G', we find that a small value of ϕ indicates that the solution space required for searching by the GADNT is small, and the GADNT can then quickly converge. Notably, the set of solutions generated by using a smaller value of ϕ is a subset of the solution space by using a larger value of ϕ. Therefore, for the example in Table 1, we can conclude that the GDT designed by the GADNT using $\phi = 15$ is better than that generated by the GADNT using $\phi = 5$.

2 For example in Table 1, from the field 'AVE_N', the GADNT produces similar GDTs, such as $\phi = 15$ and $\phi = 30$.

In Table 1, the GADNT using $\phi = 15$ is sufficient to find a good GDT. Similarly, the values, $\phi = 6$ in Table 2 and $\phi = 15$ in Table 3 are sufficient to design the good GDTs.

6.3 *Performance of the GNT over GANNT*

In the experiment, the storage of the neural tree is simply defined as the number of neural nodes in the tree. Five storage thresholds are applied to the GANNT to design the GNTs. In Tables 4 to 6, the 'AVE_H' denotes the average number of hidden nodes. The users can set the number of output nodes in a range, $[\phi_1, \phi_2]$, the value of ϕ_1 is set to two because two classes are the minimum number of classes needed for distinguishing. The parameters that are used in the GANNT are a population of 300, a crossover rate of 80% and a mutation rate of 5%. In Tables 4 to 6, the values of δ and ϕ_2 are set the same as those in Tables 1 to 3, respectively.

Table 1 The design of GDT by using the GADNT on the COREL dataset

Training phase					*Testing phase*	
δ	ϕ	*AVE_N*	*DEP*	*AVE_G*	*AVE_C*	*CER*
5	5	4.3	6	132	12.43	10.48
	15	6.6	4	254	14.52	10.44
	30	6.6	4	346	14.52	10.44
10	5	4.5	9	126	18.45	8.61
	15	6.5	6	261	21.54	8.45
	30	6.5	6	372	21.54	8.44
15	5	4.4	14	130	22.36	7.98
	15	6.6	9	259	34.65	7.25
	30	6.6	9	378	24.65	7.23
20	5	4.5	18	134	26.46	7.10
	15	6.4	12	261	27.54	6.33
	30	6.4	12	381	27.53	6.31
∞	5	4.8	23	131	28.65	3.85
	15	6.5	16	267	29.73	3.10
	30	6.5	16	412	29.72	3.04

Table 2 The design of GDT by using the GADNT on the satellite image dataset

Training phase					*Testing phase*	
δ	ϕ	*AVE_N*	*DEP*	*AVE_G*	*AVE_C*	*CER*
5	6	3.6	3	121	6.26	13.9
	9	3.6	3	195	6.26	13.9
	12	3.6	3	265	6.26	13.9
10	6	3.8	4	128	8.94	10.75
	9	3.8	4	189	8.96	10.70
	12	3.8	4	255	8.95	10.70
15	6	3.8	5	133	14.15	8.42
	9	3.8	5	186	14.13	8.40
	12	3.8	5	259	14.14	8.41
20	6	3.7	6	126	18.23	6.73
	9	3.7	6	179	18.22	6.72
	12	3.7	6	261	18.23	6.72
∞	6	3.6	12	132	22.56	1.35
	9	3.6	12	172	22.55	1.38
	12	3.6	12	254	22.57	1.36

Table 3 The design of GDT by using the GADNT on the PASCAL VOC2007 dataset

Training phase					*Testing phase*	
δ	ϕ	*AVE_N*	*DEP*	*AVE_G*	*AVE_C*	*CER*
5	5	3.9	4	156	6.31	35.54
	15	6.1	3	213	11.25	35.42
	30	6.1	3	354	11.25	35.42
10	5	3.8	6	168	9.21	34.45
	15	6.2	4	223	15.32	34.12
	30	6.2	4	366	15.35	34.13
15	5	3.8	9	173	14.32	33.13
	15	6.2	6	225	18.65	32.71
	30	6.2	6	356	18.69	32.72
20	5	3.9	12	163	17.55	32.10
	15	6.1	8	221	21.32	31.65
	30	6.1	8	361	21.38	31.69
∞	5	3.8	18	171	22.34	29.10
	15	6.1	14	264	27.31	27.75
	30	6.1	14	374	27.40	27.79

Table 4 Design of GNT using the GANNT on the COREL dataset

Training phase					*Testing phase*	
δ	*[ϕ_1, ϕ_2]*	*AVE_H*	*AVE_O*	*DEP*	*AVE_C*	*CRE*
5	[2, 5]	2.4	3.5	4	6.35	9.51
	[2, 15]	2.3	5.5	3	6.45	9.41
	[2, 30]	2.4	5.5	3	6.45	9.41
10	[2, 5]	2.3	3.9	5	9.23	7.85
	[2, 15]	2.3	5.2	4	12.62	7.32
	[2, 30]	2.4	5.2	4	11.62	7.32
15	[2, 5]	2.3	3.9	6	15.85	7.30
	[2, 15]	2.4	5.2	5	16.82	6.15
	[2, 30]	2.4	5.2	5	16.82	6.15
20	[2, 5]	2.4	3.9	7	17.20	6.55
	[2, 15]	2.4	5.2	6	18.25	4.60
	[2, 30]	2.4	5.2	6	18.25	4.60
∞	[2, 5]	2.3	3.9	9	21.25	3.62
	[2, 15]	2.4	5.2	8	22.43	2.95
	[2, 30]	2.3	5.2	8	22.43	2.95

Although the average number of child nodes in a neural node is smaller than that in a decision node, the computing complexity of a neural node is still greater than that of a decision node. Similarly, the same conclusions can also be obtained from Tables 2 to 6.

Table 5 Design of GNT using the GANNT on the satellite dataset

Training phase					*Testing phase*	
δ	*[ϕ_1, ϕ_2]*	*AVE_H*	*AVE_O*	*DEP*	*AVE_C*	*CRE*
5	[2, 6]	2.2	3.4	3	5.46	13.42
	[2, 9]	2.4	3.4	3	5.45	13.42
	[2, 12]	2.4	3.4	3	5.45	13.42
10	[2, 6]	2.3	3.4	4	9.63	8.75
	[2, 9]	2.4	3.5	4	8.62	8.78
	[2, 12]	2.4	3.4	4	8.63	9.84
15	[2, 6]	2.4	3.4	6	15.25	5.54
	[2, 9]	2.5	3.5	6	13.32	5.56
	[2, 12]	2.4	3.5	6	13.32	5.55
20	[2, 6]	2.4	3.5	8	19.30	3.42
	[2, 9]	2.5	3.5	8	17.35	5.41
	[2, 12]	2.5	3.4	8	17.32	3.42
∞	[2, 6]	2.4	3.4	10	23.34	1.19
	[2, 9]	2.4	3.4	10	21.33	1.20
	[2, 12]	2.4	3.4	10	21.35	1.19

Table 6 Design of GNT using GANNT on the VOC2007 dataset

Training phase					*Testing phase*	
δ	$[\phi_1, \phi_2]$	*AVE_H*	*AVE_O*	*DEP*	*AVE_C*	*CRE*
5	[2, 5]	2.6	3.5	4	6.42	35.42
	[2, 15]	2.6	5.6	3	6.57	35.32
	[2, 30]	2.6	5.6	3	6.57	35.32
10	[2, 5]	2.7	3.6	5	11.23	33.95
	[2, 15]	2.6	5.6	4	11.32	33.53
	[2, 30]	2.7	5.5	4	11.35	33.53
15	[2, 5]	2.7	3.5	6	15.70	33.58
	[2, 15]	2.6	5.5	5	15.95	32.40
	[2, 30]	2.6	5.6	5	15.98	32.42
20	[2, 5]	2.6	3.5	8	17.45	32.80
	[2, 15]	2.6	5.6	6	17.34	31.23
	[2, 30]	2.7	5.6	6	17.46	31.25
∞	[2, 5]	2.6	3.6	12	23.50	28.22
	[2, 15]	2.6	5.6	10	23.42	27.34
	[2, 30]	2.7	5.7	10	23.45	27.34

6.4 *Performance of the GNDT*

The GNDT is the combination of decision nodes and neural nodes. From Tables 7 to 9, we observe that if the value of δ is small, the value of 'NUM_NN' is larger than the value of 'NUM_DN'. This indicates that the nodes in the low height of the GNDT are preferred to be designed as the neural nodes; then the classification error rate of the GNDT can be rapidly reduced. If the value of δ is increased, the value of 'NUM_NN' is smaller than the value of 'NUM_DN'. This indicates that the nodes in the large height of the GNDT are preferred to be designed as the decision nodes; then the computing complexity of the GNDT can be reduced.

Table 7 The performance of the GNDT on the COREL dataset

Training phase ($\phi = 15$, $\phi_1 = 2$, $\phi_2 = 15$)				*Testing phase*	
δ	*NUM_NN*	*NUM_DN*	*DEP*	*AVE_C*	*CER*
5	4	1	3	2.76	9.83
10	7	3	4	3.45	7.41
15	8	7	5	4.32	6.12
20	9	11	6	5.12	4.91
∞	9	16	10	6.14	2.92

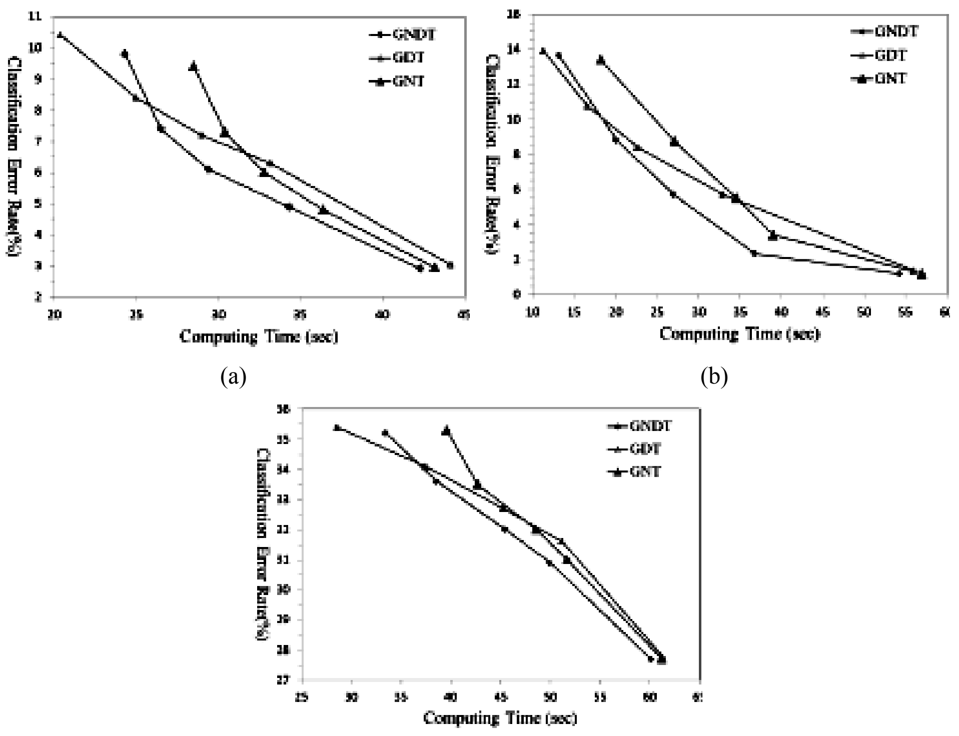
Table 8 The performance of the GNDT on the satellite dataset

Training phase ($\phi = 6$, $\phi_1 = 2$, $\phi_2 = 6$)				*Testing phase*	
δ	*NUM_NN*	*NUM_DN*	*DEP*	*AVE_C*	*CER*
5	5	0	4	3.12	13.65
10	8	2	5	4.18	8.8
15	10	5	6	5.32	5.7
20	10	10	8	6.27	2.35
∞	10	15	10	7.15	1.19

Table 9 The performance of the GNDT on the VOC2007 dataset

Training phase ($\phi = 15$, $\phi_1 = 2$, $\phi_2 = 15$)				*Testing phase*	
δ	*NUM_NN*	*NUM_DN*	*DEP*	*AVE_C*	*CER*
5	5	0	3	2.34	35.2
10	9	1	5	4.14	33.6
15	10	5	7	6.32	32
20	11	9	9	7.46	30.9
∞	12	13	12	8.63	27.68

Figure 5 Comparison of the performance between GNDT, GDT and GNT, (a) COREL dataset (b) satellite image dataset (c) PASCAL VOC2007 dataset



In Figure 5, the GNDT is compared to the GDT and GNT. The classification error rate of GDT is less than that of GNT while the computing time is decreased.

6.5 Comparison with state-of-the-art methods

6.5.1 COREL dataset

In Table 10, the overall classification accuracy of the MPEG-7 descriptors is at most 47.7% when edge histogram is merged with scalable colour. The accuracy of the method proposed in Le Borgne et al. (2007) is 54.0% with the ICA luminance global signature (Gica) and 57.6% with the ICA luminance local signature (Lica). When the colour information is added, the results reach 55.6% for the global signature and 63.8% for the local signature and proposed methods also outperform the MPEG-7.

Table 10 Accuracy rate of different methods on the COREL dataset

Class name		*Cities*	*Indoor*	*Fire works*	*Cars*	*Egypt*	*Flowers*
Class size		200	541	100	200	100	400
Proposed GADNT		72.8	63.6	97.4	62.5	48.4	81.3
Proposed GANNT		72.8	63.5	97.3	62.5	48.3	81.2
Proposed GNDT		72.4	63.2	97.4	62.3	48.3	81.0
MPEG-7	EH	33.9	43.8	48.8	45.0	13.8	41.8
	CL	43.3	25.7	85.0	27.2	31.3	24.5
	SC	19.4	15.4	73.8	26.1	52.8	30.0
	HT	25.0	21.1	72.5	22.8	10.0	30.5
	EH+CL	41.7	39.0	96.3	49.4	26.3	39.5
	EH+SC	24.8	48.0	96.3	49.4	26.9	39.5
	HT+CL	39.7	24.2	73.8	33.9	15.0	33.2
	HT+SC	32.8	21.5	73.8	24.4	8.8	32.9
	EH+CL+SC	51.7	43.4	95.0	52.2	31.3	50.3
Methods in Le Borgne et al. (2007)	Gica	70.0	41.1	47.5	44.4	25.0	82.9
	Lica	46.1	51.1	91.3	50.0	25.0	81.6
	Gica+ colour	47.8	55.3	56.3	50.0	36.3	70.5
	Lica+ colour	69.4	66.6	88.8	56.7	33.8	73.2

6.5.2 Satellite image dataset

The satellite image dataset was taken from the UCI machine learning data repository. Table 11 lists the comparison of the proposed methods with other methods on the satellite image dataset. The proposed GNDT outperforms the other methods, including the BNT.

Table 11 Accuracy rates of different methods on the satellite image dataset

Methods	*Accuracy %*						
	RS	*CC*	*GS*	*DGS*	*SVS*	*VDGS*	*AVE*
Proposed GADNT	91.65	91.85	91.85	92.54	91.65	89.21	91.46
Proposed GANNT	91.50	91.80	91.85	92.40	91.65	89.21	91.40
Proposed GNDT	90.50	91.25	91.52	92.25	91.18	89.05	90.79
MLP (Tanwani et al., 2009)	85.24	84.56	86.54	86.28	84.54	84.22	86.20
Bayesian (Cheeseman and Stutz, 1996)	85.24	84.56	86.54	86.28	84.54	84.22	85.40
NNTree (Maji, 2008)	85.24	84.56	86.54	86.28	84.54	84.22	87.10
BNT (Micheloni et al., 2012)	85.24	84.56	86.54	86.28	84.54	84.22	85.25
C4.5 (Dai and Ji, 2014)	85.24	84.56	86.54	86.28	84.54	84.22	85.20
C5.0 (Gong et al., 2009)	85.24	84.56	86.54	86.28	84.54	84.22	85.23

Notes: RS: red soil, CC:cotton crop, GS: grey soil, DGS: damp grey soil, SVS: soil with vegetation stubble, VDGS: very damp grey soil

6.5.3 PASCAL VOC2007 dataset

In 2014, the classification results on the PASCAL VOC2007 dataset were announced in Fernando et al. (2014). Table 12 lists the comparison results of the proposed methods with other methods on the PASCAL VOC2007 dataset. The accuracy of the ‘aero’, ‘car’, ‘table’, ‘plant’ and ‘train’ pictures by using the GNDT, are slightly lower than those by using the FLH+BOW+FV method.

Table 12 Accuracy rates of different methods on the PASCAL VOC2007 dataset

Methods	*Accuracy %*						
	Aero	*Bicyc*	*Bird*	*Boat*	*Bottle*	*Bus*	*Car*
Proposed GADNT	79.3	79.3	56.1	76.6	75.9	75.8	87.3
Proposed GANNT	79.1	79.3	56.1	76.6	75.9	75.8	87.2
Proposed GNDT	78.9	78.4	55.9	76.5	75.8	75.6	87.2
Hard voting (Csurka et al., 2004)	62.6	51.7	38.1	57.5	23.8	50.8	67.7
Soft voting (van Gemert et al., 2008)	68.5	58.2	40.6	60.8	26.7	60.6	72.2
Salient coding (Huang et al., 2011)	71.3	64.2	45.5	67.4	29.8	63.9	78.2
Fisher vector (Perronnin et al., 2010)	78.8	67.4	51.9	70.9	30.8	72.2	79.9
FLH+BOW+FV (Fernando et al., 2014)	80	78	55.9	76.2	75.5	75.6	88.1
C4.5 (Dai and Ji, 2014)	62.3	52.1	43.2	53.5	26.3	46.3	66.8
C5.0 (Gong et al., 2009)	62.3	52.2	43.3	53.5	26.4	46.3	66.8

7 Conclusions

In this study, three classifiers namely GADNT, GANNT and GNDT, with the storage constraints are proposed for image classification. The proposed GADNT is able to design the proper number of child nodes of each decision node in the GDT according to the classification error rate and computing complexity of GDT. In GNT, the GANNT is proposed to search for the proper number of hidden and output nodes in the neural network according to the classification error rate and computing complexity of GNT. Finally, the GNDT, the combination of GDT and GNT, is proposed. The GNDT is designed by two kinds of nodes, the decision and neural nodes. The growing method is proposed to determine whether each internal node should be designed as a decision node or a neural node in the GNDT according to the classification error rate and computing complexity of the GNDT. In our experiments, the GNDT outperforms GDT, GNT and other state-of-the-art methods.

References

Asuncion, A. and Newman, D.J. (2007) 'UCI machine learning repository' [online] http://www.ics.uci.edu/~mlearn/MLRepository.html.

Cheeseman, P. and Stutz, J. (1996) 'Bayesian classification (autoclass): theory and results', in *Advances in Knowledge Discovery and Data Mining*, pp.153–180.

Csurka, G., Bray, C., Dance, C. and Fan, L. (2004) 'Visual categorization with bags of keypoints', *ECCV*.

Dai, W. and Ji, W. (2014) 'A map reduce implementation of C4.5 decision tree algorithm', *International Journal of Database Theory and Application*, Vol. 7, No. 1, pp.49–60.

Fernando, B., Fromont, E. and Tuytelaars, T. (2014) 'Mining mid-level features for image classification', *International Journal of Computer Vision*, Vol. 108, No. 3, pp.186–203.

Foresti, G.L. (2004) 'An adaptive high-order neural tree for pattern recognition', *IEEE Trans. Systems, Man, Cybernetics – Part B: Cybernetics*, Vol. 34, No. 2, pp.988–996.

Foresti, G.L. and Micheloni, C. (2002) 'Generalized neural trees for pattern classification', *IEEE Trans. Neural Networks*, November, Vol. 13, No. 6, pp.1540–1547.

Gong, G., Lv, J. and Huang, Y. (2009) 'The method of effectively improve the C5.0 algorithm', *Computer Engineering and Design*, Vol. 30, No. 22, pp.5197–5199.

Gonzalez, R.C. and Woods, R.E. (1992) *Digital Image Processing*, Addison-Wesley, Boston, MA.

Guo, H. and Gelfand, S.B. (1992) 'Classification trees with neural networks feature extraction', *IEEE Trans. Neural Networks*, May, Vol. 3, No. 6, pp.923–933.

Heumann, B.W. (2011) 'An object-based classification of mangroves using a hybrid decision tree – support vector machine approach', *Remote Sens.,* Vol. 3, No. 11, pp.2440–2460.

Huang, B.Y., Kaiqi, H., Yinan, Y. and Tieniu, T. (2011) 'Salient coding for image classification', *Computer Vision and Pattern Recognition, IEEE Conference on CVPR*, June, pp.1753–1760.

Le Borgne, H., Guérin-Dugué, A. and O'Connor, N.E. (2007) 'Learning midlevel image features for natural scene and texture classification', *IEEE Transactions on Circuits and Systems for Video Technology*, March, Vol. 17, No. 3, pp.286–297.

Maji, P. (2008) 'Efficient design of neural network tree using a single splitting criterion', *Nerocomputing*, Vol. 1, No. 71, pp.787–800.

Micheloni, C., Rani, A., Kumarb, S. and Foresti, G.L. (2012) 'A balanced neural tree for pattern classification', *Neural Networks*, Vol. 27, No. 4, pp.81–90.

Naeem, M., Khan, A., Qureshi, S.A. and Riaz, N. (2013) 'Gender classification with decision trees', *International Journal of Signal Processing, Image Processing and Pattern Recognition*, February, Vol. 6, No. 1, pp.165–176.

Pedrycz, W. and Sosnowski, Z.A. (2000) 'Designing decision trees with the use of fuzzy granulation', *IEEE Trans. Syst. Man Cybern., Part A. Syst. Humans*, Vol. 30, No. 2, pp.151–159.

Perronnin, F., Sanchez, J. and Mensink, T. (2010) 'Improving the fisher kernel for large-scale image classification', *European Conference on Computer Vision*, pp.143–156.

Sharma, R., Ghosh, A. and Joshi, P.K. (2013) 'Decision tree approach for classification of remotely sensed satellite data using open source support', *J. Earth Syst. Sci.*, October, Vol. 122, No. 5, pp.1237–1247.

Shukla, S.K. and Tiwari, M.K. (2012) 'GA guided cluster based fuzzy decision tree for reactive ion etching modeling: a data mining approach', *IEEE Trans. on Semiconductor Manufacturing*, Vol. 25, No. 1, pp.45–56.

Sirat, J.A. and Nadal, J.P. (1990) 'Neural tree: a new tool for classification', *Network*, Vol. 1, No. 4, pp.423–438.

Tanwani, A., Afridi, J., Shafiq, M. and Farooq, M. (2009) 'Guidelines to select machine learning scheme for classification of biomedical datasets', in *LNCS*, Vol. 5483, pp.128–139.

Utgoff, P.E. (1998) 'Perceptron tree: a case study in hybrid concept representation', in *Proc. VII Nat. Conf. Artificial Intelligence*, pp.601–605.

van Gemert, J., Geusebroek, J., Veenman, C. and Smeulders, A. (2008) 'Kernel codebooks for scene categorization', *ECCV*.

Witold, P. and Zenon, A.S. (2005) 'C-fuzzy decision trees', *IEEE Trans. Systems, Man, Cybernetics, C, Applications and Reviews*, Vol. 35, No. 4, pp.498–511.

Yang, S.B. (2010) 'Fuzzy variable-branch decision tree', *Journal of Electronic Imaging*, Vol. 19, No. 4, pp.043001–043012.