Chapter 6

Automatic Image Annotation Using Machine Learning

6.1 Introduction

Image searching on internet using natural language is easily possible for the user due to automatic image annotation. Image annotation refers to the tagging of images with appropriate keywords [Setia and Burkhardt, 2006]. The algorithms used in automatic image annotation systems perform four tasks namely segmentation, feature extraction, training annotation system, and annotation of new images. Automatic image annotation can be done using classification based techniques [Guo et al., 2011]. Computer vision has image categorization and understanding as two most challenging task. The image is annotated as per the semantic category by understanding the contents of the image. The information in the digital images at low level can be extracted but relating it to high level semantic concepts is one of difficulty in this process. Image categorization is considered as a supervised learning problem by most of the approaches. These approaches train the system using image dataset that is already tagged, the system learns the categories from this dataset and then new images are assigned keywords as per the categories. The techniques used depends on rely on extracting visual features. These features are representing visual patterns that correspond to a category. Providing suitable sets of visual descriptors has been a topic of active research in the recent years and several approaches have been proposed [Piro et al., 2009].

The automatic image annotation is targeted to assign a few relevant text keywords to an input image that reflects its visual content. As multimedia data of the web is increasing rapidly, at the same time storage capacity of digital devices is not a big issue, due to this fast, efficient and robust image search engine is a critical requirement. Current Internet image search engines ignores image contents though efficiently exploit keyword-based search to retrieve relevant images. Accuracy in

assigning relevant keywords to images as per their contents can improve the indexing and retrieval results by the search engines. This makes the problem of tagging images with appropriate text keywords of immense practical interest. Image annotation is a complicated task for two main causes: The first is the familiar low level to high level semantic gap problem, which points to the fact that it is difficult to extract semantically significant entities using low level image features, e.g. color and texture. The alternative of doing explicit identification of thousands of image objects or classes reliably is currently an unsolved problem. The second difficulty is due to the lack of association between the image regions and keywords in the training data. For each image, one has access to the keywords assigned to the entire image and it is not known which regions of the image correspond to these keywords. [Makadia et al., 2010].

6.2 Machine Learning for Image Annotation

Acquisition of new declarative knowledge, the development of model and cognitive skills through practice, new knowledge organization is to be converted into general, representations those are effective, and the finding new facts and theories through experimentation is a process termed as Learning process[Carbonell et al., 1983].

The goal of machine learning is to construct computer systems that can learn and adapt from their experience [Dietterich, 1999]. Thus, the system becomes competent of the independent acquisition and integration of knowledge is called as machine learning. This learning results in a system that can progress with its own speed or performance of the process. The overall goal of machine learning is to improve efficiency and/or effectiveness of the system.

6.2.1 Artificial Neural Network (ANN)

The processing element at root level in artificial neural network is called neuron. Neuron receives inputs that come from either output generated from other neuron or the environment.

A single layer perceptron can only approximate linear functions of the input and cannot solve problems where the function to be estimated nonlinear. In the same way for nonlinear regression a perceptron cannot be used. The restriction is not applied to the layers between input and output layer called hidden layer in the feedforward design. A multilayer perceptron (MLP) can be used for classification and can

implement nonlinear discriminating function. This can also approximate nonlinear functions of the input for regression [Alpaydin, 2004].

A multilayer perceptron (MLP) is a feedforward artificial neural network design where input data set is map to a set of appropriate output. Every layer of the design is fully connected to the next layer of MLP. It is a directed graph of multiple layers with nodes, where nodes representing neurons in the architecture. Every neuron except input layer neurons are processing element with a nonlinear activation function. Figure 6.1 shows a MLP network with two hidden layers.

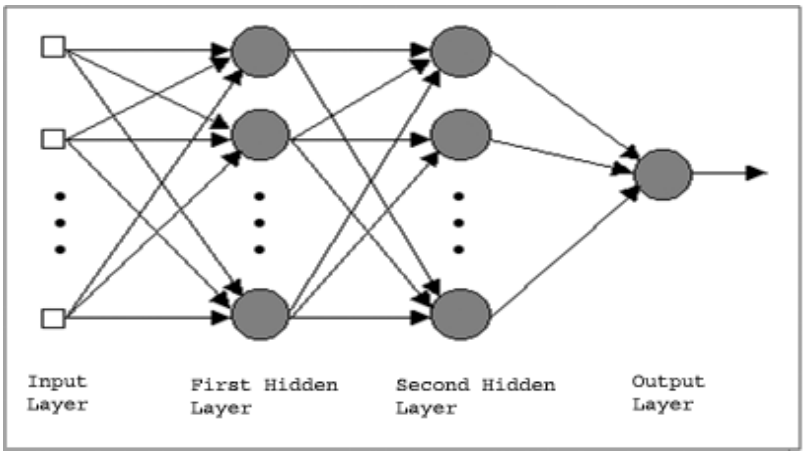

Figure 6.1. A Multilayer Perceptron**.**

A multilayer perceptron deals with nonlinear classification problem because it can form more complex design regions rather than just hyperplanes. The training of network in MLP is done with a supervised learning technique using back propagation learning algorithm [Rumelhart et al., 1986]. The learning in back propagation is carried out to update weights using equation 6.1.

$$W_{ji}(t+1) = W_{ji}(t) + \eta\delta_j O_i \qquad (6.1)$$

Where

δ_j is error gradient
η is learning rate
O_i is input

The error gradient for hidden layer and output layer is calculated as

For output units: $\delta_j = O_j(1 - O_j)(T_j - O_j)$

Where T_j is the desired (target) output activation and O_j is the actual output unit j.

For hidden units: $\delta_j = O_j(1 - O_j)\sum_k \delta_k W_{kj}$

Where $\delta_k$is the error gradient at unit k to which a connection points from hidden unit j.

6.2.2 Decision Trees (DT)

Decision tree is a natural way of presenting a decision-making process, because of simple and easy for anyone to understand [Quinlan, 1986]. A decision tree is a simple, but powerful form of multiple variable analysis and used for attribute values to class mappings. Decision trees provide capabilities to be used in place of multiple linear regressions such as statistical form analysis or in business intelligent systems where multidimensional data analysis is expected [Lomax and Vadera, 2013]. A tree is either a leaf node labeled with a class or a structure consisting of a test node linked to two or more sub trees. A test node computes some outcome based on the attribute values of an instance, where each possible outcome is associated with one of the sub trees. An instance is classified by starting at the root node of the tree. If this node is a test, the outcome for the instance is determined and the process continues using the appropriate sub tree. When a leaf is eventually encountered, its label gives the predicted class of the instance [Quinlan, 1996]. Figure 6.2 shows classification process using decision tree (DT) classifier.

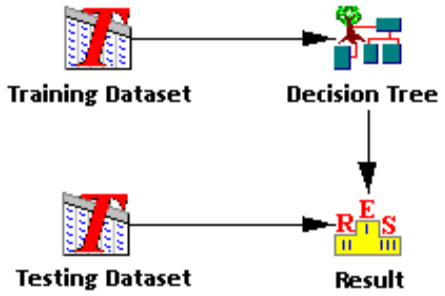

Figure 6.2. Design of Decision Tree (DT) classifier.

6.2.3 Rough Sets(RS)

Rough set theory is for the classification and analysis of imprecise, uncertain or incomplete information and knowledge, and is considered one of the first non-statistical approaches in data analysis [Pawlak, 1982]. The fundamental concept behind rough set theory is the approximation of lower and upper spaces of a set, the approximation of spaces being the formal classification of knowledge regarding the interest domain [Rissino and Lambert-Torres, 2009].

In the discretization of a decision table $\mathcal{A} = (U, A \cup \{d\})$ where Va==(va, wa) is an interval of real's, the objective is to search for a partition P_a of V_a for any $a \epsilon A$. Any partition of V_a is defined by a sequence of the cuts v1<v2<v3<…vk from V_a Hence,

any family of partitions $\{P_a\}_{a\epsilon A}$ can be identified with a set of cuts. The discretization process was targeted to search for a set of cuts satisfying some natural conditions [Komorowski et al., 1999]. Figure 6.3 shows design of a Rough Set (RS) classifier.

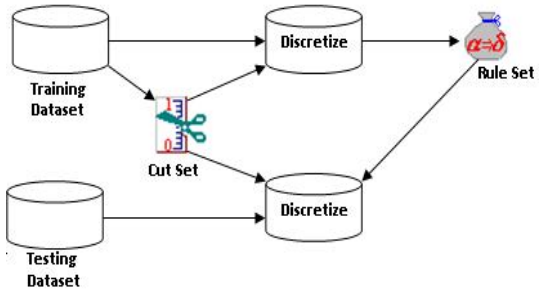

Figure 6.3. Design of Rough Set (RS) classifier.

Let $X \subseteq \mathbb{U}$ be a target set that is represented using attribute subset P; that is, it is to be expressed with an arbitrary set of objects X comprising a single class, and this class (i.e., this subset) is to be expressed using the equivalence class induced by attribute subset P. In general, X cannot be expressed exactly, because the set may include and exclude objects which are indistinguishable on the basis of attributes P.

However, the target set X can be approximated using only the information contained within P by constructing the P-lower and P-upper approximations of X:

$$\underline{P}X = \{x | \{x\}_P \subseteq X\} \quad (6.2)$$

$$\overline{P}X = \{x | \{x\}_P \cap X \neq \emptyset\} \quad (6.3)$$

The tuple $\langle \underline{P}X | \overline{P}X \rangle$composed of the lower and upper approximation is called a rough set; thus, a rough set is composed of two crisp sets, one representing a lower boundary of the target set X, and the other representing an upper boundary of the target set X.

6.2.4 K-Nearest Neighbor (K-NN)

The k-nearest neighbor (K-NN) rule has been one of the well-known and widely used as nonparametric classifiers in pattern classification, introduced by Fix and Hodges. As per the majority of nearest neighbors in the training set an unclassified object is assigned to the class [Fix and Hodges, 1989].

The perception of nearest neighbor classification is, examples are classified based on the class of their nearest neighbors. The technique is referred as k-nearest neighbor (K-NN) because it takes more than one neighbor into account for classification, here k denotes nearest neighbors used in determining the class. Classification is based

directly on the training examples hence it is also called Example-Based Classification or Case-Based Classification.

K-NN classification is composed of two stages; the first is the determination of the neighbors nearest to the test sample and the second is the determination of the class using those neighbors. To find the class simple majority voting or distance weighted voting is used [Gou et al., 2011].

Let $T = \{x_1, x_2, \dots, x_n\}$ be the training set, i.e., a corpus of training vectors or class-labeled points (x_i , c_j). The training vectors $x_i \in R_m$ are vectors in the m-dimensional feature space,$1 \leq i \leq N$, and c_j , j = 1,..., Nc are their corresponding class labels (typically $N > N_c$). Given a query object (x', c') randomly, its unknown class c' is determined as follow:

1) Select the set $T' = \{x_1^{NN}, x_2^{NN}, \dots, x_k^{NN}\}$, the set of k nearest training objects to the selected query object x', arranged in an ascending order in terms of the distance (or dissimilarity) measure between x_i^{NN} and x'. $d(x', x_i^{NN})$ is popularly computed by

Absolute distance

$$d(x', x_i^{NN}) = \sum_{i=1}^{N} |x' - x_i^{NN}| \tag{6.4}$$

Euclidean distance

$$d(x', x_i^{NN}) = \sqrt{\sum_{i=1}^{N} (x' - x_i^{NN})^2} \tag{6.5}$$

2) Assign the class label to the query object x', based on the majority voting class of its nearest neighbors:

$$c' = \arg\max \sum_{x_i^{NN} \epsilon T'} I[c = c_i^{NN}] \tag{6.6}$$

Where c is a class label, c_i^{NN} is the class label for the i^{th} neighbor among k nearest neighbors of the query object. $I[c = c_i^{NN}]$, an indicator function, takes a value of one if the class c_i^{NN} of the neighbor x_i^{NN} is the same as the class c and zero otherwise.

6.3 Shape Tagging Using Machine Learning

Shape classification is the key issue of the automatic image tagging system where learning of the system is the objective with accurate classification. Shape annotation experiments are carried out using machine learning techniques viz. artificial neural networks, decision trees, rough sets and k-nearest neighbor (K-NN).

6.3.1 Artificial Neural Networks (ANN)

Multi-layer perceptron is best suited for classification and hence feed forward neural network is used. Three different architectures are designed during experiments.

6.3.1.1 Single Neural Network Classifier

Network is designed using single neural network classifier has 5 neurons in the input layer and that generates 7 bit output. The layers and the neurons in each hidden layer are finalized by taking various combinations of neurons from 5 to 40 and layers 1 to 3. Figure 6.4 shows the automatic shape tagging process carried out on shape feature vector. The experiments are carried out on 1400 shape images, out of which 1120 images are used for training remaining 280 shape images are used for testing.

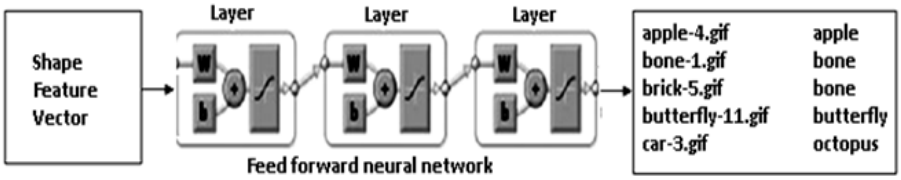

Figure 6.4. Classifier design using ANN.

The input/output mapping of a network is established by adjusting the weights and setting the activation functions of their neurons in hidden and output layers. The activation function used is tan-sigmoid generates output between -1 to 1. Network training function used over here is Levenberg-Marquardt optimization, this function updates weight and bias values according to Levenberg-Marquardt optimization. Back propagation weight/bias learning is carried out using gradient descent with momentum weight and bias learning function. The shape feature vector consists of five features so the five input neurons are required and for 70 output categories 7 bits are essential. If 7 bits are used only 128 categories can be recognized as the shapes are 70 the 7 bits are used for output e.g. apple is represented using [0,0,0,0,0,0,0] and watch is represented using [1,0,0,0,1,1,0].

The input/output mapping for 10 sample categories is shown in following 10 sample records of shape feature vector. This sample feature vector shows first five columns for input while next seven are corresponding to output.

[Input]	[Output]
[28279, 0.159546, 0.92057, 0.701695, 776.5656]	[-1,-1,-1,-1,-1,-1,-1]
[29902, 0.89822, 0.638563, 0.355824, 1374.126]	[-1,-1,-1,-1,-1,-1, 1]
[19100, 0.85, 0.368555, 0.177774, 2530.505]	[-1-1,-1,-1,-1, 1,-1]
[20495, 0.680076, 0.887268, 0.565629, 689.394]	[-1,-1,-1,-1,-1, 1, 1]
[26425, 0.838545, 0.636594, 0.359026, 1030.916]	[-1,-1,-1,-1, 1,-1,-1]

[29232, 0.985921, 0.472284, 0.18652, 1390.972] [-1,-1,-1,-1, 1,-1, 1]
[6102, 0.964691, 0.960038, 0.656482, 401.2376] [-1,-1,-1,-1, 1, 1,-1]
[12017, 0.937438, 0.831166, 0.668577, 571.2447] [-1,-1,-1,-1, 1, 1, 1]
[72401, 0.886965, 0.765104, 0.583066, 2400.522] [-1,-1,-1, 1,-1,-1,-1]
[44137, 0.797699, 0.75178, 0.513627, 1581.148] [-1,-1,-1, 1,-1,-1, 1]

No of Hidden Layers	**No. of Neurons In Hidden Layer**			**Mean Squared Error (MSE)**	Annotation Accuracy **(%)**		
	Ist	IInd	IIIrd		Overall	Training	Testing
1	5	-	-	0.725461	4.43	4.82	2.86
1	15	-	-	0.564409	15.14	15.89	12.14
1	20	-	-	0.684058	16	17.41	10.36
1	40	-	-	0.240241	64.71	70.63	41.07
2	5	5	-	0.657533	4.86	4.82	5
2	5	30	-	0.337191	49.79	54.38	31.43
2	10	25	-	0.846597	13.71	14.73	9.64
2	15	20	-	0.206165	66.64	73.21	40.36
2	20	15	-	0.206381	69.64	75.45	46.43
2	20	40	-	0.064955	81.79	90.98	45
2	30	30	-	0.090823	77.36	85.63	44.29
2	35	40	-	0.04082	84.07	93.84	45
2	40	40	-	0.033862	86	94.91	50.36
3	5	5	5	0.676446	13.07	13.66	10.71
3	5	5	40	0.333504	48.36	53.39	28.21
3	5	10	35	0.147404	73.57	80.18	47.14
3	10	5	30	0.279692	54.64	59.82	33.93
3	10	20	40	0.046758	84.57	92.68	52.14
3	10	35	30	0.03215	87.07	94.73	56.43
3	15	10	25	0.130586	76.43	84.2	45.36
3	15	20	35	0.041359	85.21	93.93	50.36
3	15	25	40	0.04031	86.57	94.29	55.71
3	20	5	30	0.21599	64.71	71.96	35.71
3	20	20	40	0.040819	85.93	94.38	52.14
3	25	25	20	0.046962	84.36	92.86	50.36
3	25	30	30	0.026025	88.07	95.8	57.14
3	30	30	5	0.108662	75.36	83.84	41.43
3	**30**	**30**	**35**	**0.015439**	**89.71**	**97.5**	**58.57**
3	35	25	30	0.049082	84.29	92.14	52.86
3	40	25	15	0.106155	74.57	83.39	39.29

Table 6.1. Sample tagging accuracy by single neural network.

Some sample outputs are listed in Table 6.1. It is observed that with 3 hidden layers with 30, 30 and 35 neurons respectively in 1st, 2nd and 3rd layer gives best tagging accuracy amongst all architectures of network. The performance is measured using Mean Squared Error(MSE). The 3 layered network design 30,30and 35 reduced MSE

to 0.015439. The tagging accuracy is listed in Table 6.2.

Training Accuracy			Testing Accuracy			Overall Accuracy		
Total sample	Correctly classified	Mis-classified	Total sample	Correctly classified	Mis-classified	Total sample	Correctly classified	Mis-classified
1120	1092 (97.5%)	28	280	164 (58.57%)	116	1400	1256 (89.71%)	144

Table 6.2. Tagging performance by single neural network (3 hidden layer 30, 30, 35).

6.3.1.2 Hierarchical Classifier (Domain Expert suggested)

The single network is not able to classify the similar shapes. If the two shapes are similar then a separate network should be deployed to classify only those classes. This gives idea of building a hierarchical design of networks i. e. hierarchical combination of multiple networks can lead in to a better classifier. The hierarchical structure of networks is developed using the similarities in the features of various shapes. The hierarchy is defined by doing schoolwork on the feature vector. The similar shapes have features in ratio are mostly similar and come in a group. Likewise grouping is done to develop a node in level of hierarchy.

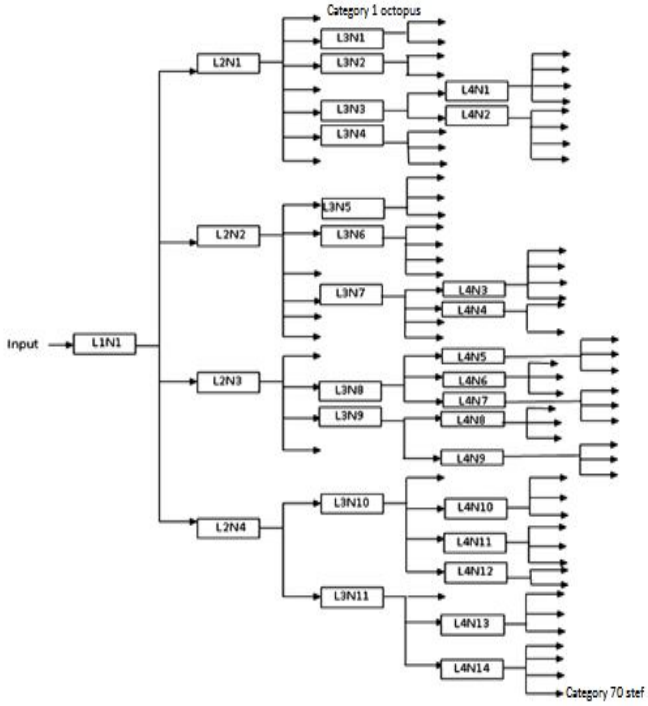

Figure 6.5. Domain Expert suggested Hierarchical design of the classifier.

Figure 6.5 gives hierarchical design of classifier designed. The hierarchy consists of 30 networks in which the output of the leaf networks is used to propagate output. This hierarchical design developed by knowledge base of domain expert (human). The details of each layer with number of hidden layers and processing elements in every

layer are given in table 6.3. Table gives details about the leaf nodes i.e. labels generated by that network.

Layer Number	No of Networks	No of Hidden Layers(HLs) Processing Elements (PEs)		Number of tags generated	Output tags
		Network	HLs-PEs		
1	1	L1N1	2-40,30	0	-
2	4	L2N1	2-5,30	3	Octopus, bottle, device6
		L2N2	1-10	3	device3, comma, pocket
		L2N3	1-35	2	device2, butterfly
		L2N4	2-5,10	0	-
3	11	L3N1	1-5	2	device4,device9
		L3N2	1-5	2	cup, pencil
		L3N3	1-5	0	-
		L3N4	1-10	3	face, hcircle, jar
		L3N5	1-10	3	crown, lmfish, fish,
		L3N6	1-5	4	chopper, frog, guitar, watch
		L3N7	1-10	2	Truck, children
		L3N8	1-20	0	-
		L3N9	1-5	0	-
		L3N10	1-10	1	device7
		L3N11	1-15	1	horseshoe
4	14	L4N1	1-20	4	cellular_phone, flatfish, personal_car, shoe
		L4N2	1-5	4	apple, bell, hat, heart
		L4N3	1-15	4	brick, car, classic, key
		L4N4	1-5	2	misk, teddy
		L4N5	1-25	3	Bat, cattle, tree
		L4N6	1-10	3	device0, device1, device5
		L4N7	1-5	3	camel, elephant, ray
		L4N8	1-5	3	chicken,device8,fountain
		L4N9	1-5	3	bird, spoon, turtle
		L4N10	2-25, 10	3	fork, lizzard, spring
		L4N11	1-10	3	beetle, deer, rat
		L4N12	1-5	2	fly, horse
		L4N13	1-5	3	bone, carriage, hammer
		L4N14	1-25	4	dog glass, sea_snake, stef

Table 6.3. Network details of each hidden layer of domain expert suggested hierarchical design.

6.3.1.3 Hierarchical Classifier (Fuzzy based)

As the experimental results of human suggested cluster for developing hierarchies are once again taken for school work to improve annotation accuracy and it is observed that human suggested hierarchy gives better results than single network but still if fuzzy is used by using clustering the hierarchies can be developed in better way. Fuzzy c-mean clustering algorithm is used to form hierarchies. The classifier is having 5 layers with total 34 neural networks. The groups of shapes are clustered using fuzzy c-mean clustering. The feature vector and number of groups i.e. clusters are input for clustering algorithm. The clustering algorithm creates groups as per

similarity in the input data.

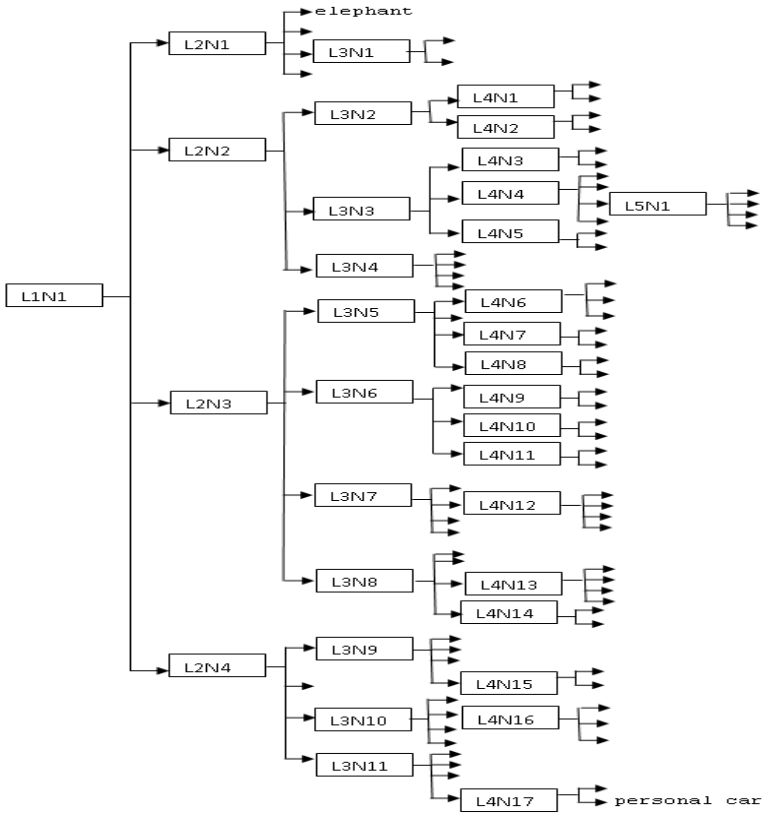

Figure 6.6. Fuzzy clustering based Hierarchical design of the classifier.

The design of classifier is like previous design of hierarchical classifier shown in figure 6.6. Here only number of networks and layers are restructured using fuzzy c-mean algorithm. The layer wise details about every network and the categories tagged are given in Table 6.4. The network at each layer is selected by experimenting single network as pervious experiment by changing hidden layer and number of processing elements. It is observed that with domain expert suggested structure some categories are misclassified due to similarities in some features in some objects of that shape. The results of domain expert suggested classifier are listed in table 6.5.

In the hierarchy first similar categories are classified in one category then in next levels of hierarchy the further classification is done. The features extracted are similar for cellular phone, flat fish, car, shoe, apple, hat, bell, heart, and circle for the first level hence all are classified in a single group. In the next level further it is classified like circle, heart and apple are similar so in a group and other are in other groups. As misclassification is more in some categories lead to modify hierarchy. Fuzzy c-mean clustering is used to form a new hierarchy of networks.

Layer Number	No of Networks	No of Hidden Layers (HLs) Processing Elements (PEs)		Number of tags generated	Output tags
		Network	HLs-PEs		
1	1	L1N1	2- 40, 25	0	-
2	4	L2N1	1-5	3	elephant, heart, device3
		L2N2	2- 30,10	0	-
		L2N3	1-40	0	-
		L2N4	1-15	1	classic
3	11	L3N1	1-10	2	Deer, device9
		L3N2	1-5	0	-
		L3N3	1-15	0	-
		L3N4	1-10	4	Flatfish, glass, hcircle, shoe
		L3N5	1-15	1	carriage
		L3N6	1-10	0	-
		L3N7	1-10	3	Fork, fish, tree
		L3N8	1-20	2	Teddy, hat
		L3N9	1-5	3	device5, butterfly, device8
		L3N10	1-10	3	bat, device7, misk
		L3N11	1-25	3	horse, device6, jar
4	17	L4N1	1-5	2	apple, frog
		L4N2	1-5	2	bone, watch
		L4N3	1-5	2	bird, cup
		L4N4	1-10	3	camel, turtle, lizzard
		L4N5	1-5	2	dog, lmfish
		L4N6	1-5	3	chopper, pencil, sea_snake
		L4N7	1-5	2	Brick, fountain
		L4N8	1-5	2	car, hammer,
		L4N9	1-5	2	spoon, horseshoe,
		L4N10	1-5	2	bottle, children,
		L4N11	1-5	2	stef, truck
		L4N12	1-5	4	Bell, crown, pocket, rat
		L4N13	1-5	4	Beetle, fly, octopus, spring
		L4N14	1-5	2	chicken, key
		L4N15	1-5	2	comma,device4
		L4N16	1-25	3	cattle,device1,device2
		L4N17	1-5	2	device0, ray
5	1	L5N1	1-5	4	cellular_phone, face, guitar, personal_car

Table 6.4. Network details of hidden layers of fuzzy clustering based hierarchical design.

It is observed that with this structure the results are improved and very less objects in some category leads to misclassification. The annotation accuracy of fuzzy clustering based classifier is given in Table 6.5.

HNN Design	Training Accuracy (#1200 Samples)		Testing Accuracy (#280 Samples)		Overall Accuracy (#1400 Samples)	
	Correctly classified samples	Mis-classified sample	Correctly classified Samples	Mis-classified sample	Correctly classified samples	Mis-classified sample
Domain Expert Suggested	977 (87.23%)	143	186 (66.42%)	94	1163 (83.07%)	237
Fuzzy Clustering based	1108 (98.92%)	12	209 (74.64%)	71	1317 (94.07%)	73

Table 6.5. Annotation accuracy of hierarchical classifiers.

6.3.2 Decision Trees (DT)

The 10 folds are created on all 1400 images of MPEG7 shape dataset. Ten folds of 140 images are created out of these 10 folds one is used for testing and remaining nine folds consisting 1260 shapes are used for training. The experiment is carried out 10 times with all folds are tested once and finally the mean is calculated for accuracy measurement.

DT based classifier generates decision tree with total 63 nodes for every fold of training data set. The decision tree for 10^{th} fold is as shown in figure 6.7. Every internal node in this tree corresponds to a decision while leaf node corresponds to a decision of generating tag for query image.

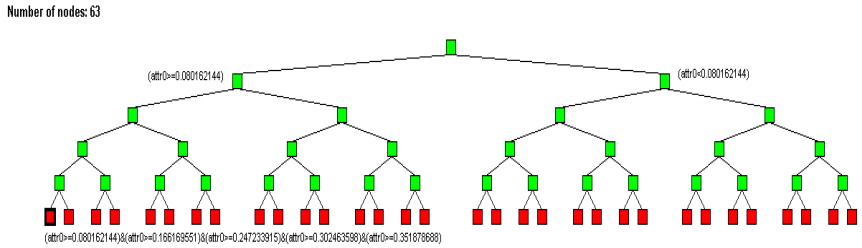

Figure 6.7. Decision Tree for 10^{th} fold.

In the above decision tree the internal nodes corresponds to rules not generating category but subdivides in groups of categories e.g. the first node in fourth level of above tree checks for eccentricity, extent and circularity ratio to generates two sub groups (attr1>=0.083011933)&(attr2<0.16894522)&(attr3>=0.120202322).

The leaf node at last level takes decision about category e.g. for the first leaf rule formed is (attr1>=0.083011933)&(attr1>=0.16894522)&(attr2>=0.250658773)& (attr2>=0.302463598)&(attr3<0.351878688) takes decision whether it is apple or not.

The experiments are fold wise carried out containing 140 test image objects in the respective fold. The classifier could classify maximum 60.71% objects correctly with the fold wise average classification 56.07%. It is observed that DT classifier cannot classify all 140 objects i.e. average coverage of this classifier is 69.65% for all folds. The fold wise annotation accuracy is as shown in table 6.6.

	fold1	**fold2**	**fold3**	**fold4**	**fold5**	**fold6**	**fold7**	**fold8**	**fold9**	**fold10**
Correctly annotated	82	79	85	83	72	80	78	76	77	78
Annotation Accuracy (%)	58.57	56.42	60.71	59.28	51.42	57.14	55.71	54.28	55	55.71
Coverage (%)	73.6	63.6	68.6	69.3	70.7	71.4	67.1	71.4	70	65.7

Table 6.6. Fold wise annotation accuracy of Decision Tree classifier.

6.3.3 Rough Sets (RS)

Using other shape based features the tagging accuracy of DT classifier can be improved, but from rough set theory it is learn that approximation of upper and lower bound by making crisp set can improve annotation accuracy. Therefore a novel RS based classifier is proposed for shape annotation. For RS based classifier more than 400 decision rules are generated for every fold of training data set. These rules are used to classify 140 test image objects in the respective fold. For the 10th fold it is observed that 515 rules are generated the sample rules are listed below:

Rule 1
attr0="(0.0176904,0.0256749)")&(attr1="(0.926158976,0.959491008)")&(attr2="(-Inf,0.628161024)") &(attr5="(0.142508,0.244596992)") =>(attr6=carriage[18])
Rule 2
attr0="(0.0566444,0.0734632)")&(attr1="(0.545681984,0.691417024)")&(attr5="(0.329627008,0.478323008)")=>(attr6=teddy[18])
Rule 3
attr0="(-Inf,0.0176904)")&(attr1="(0.959491008,Inf)")&
(attr2="(0.756294016,0.847793024)")&(attr5="(0.329627008,0.478323008)")=>
(attr6= children[17])
Rule 4
(attr0="(0.258416,Inf)")&(attr3="(0.784579968,Inf)")=>(attr6=device3[17])
(attr1="(0.146943008,0.545681984)")&(attr5="(0.478323008,Inf)")&(attr0="(0.0734632,0.107135)")=>(attr6=apple[15])
Rule 5
(attr1="(0.959491008,Inf)")&(attr2="(-Inf,0.628161024)")&(attr3="(-Inf,0.277504992)")&(attr5="(0.142508,0.244596992)")=>(attr6=bone[15])
.
.
.
The rules stated above are using five features represented by five attributes in table and sixth attribute is shape category. The features area, eccentricity, extent, perimeter and circularity ratio are represented using attr1, attr2, attr3, attr4 and attr5 respectively and the category is represented as attr6.

In rule 1 if the value of attr0 is between 0.0176904 to 0.0256749 and attr1 is between 0.926158976 to 0.959491008 and attr2 has maximum value 0.628161024 and attr5 is between 0.142508 to 0.244596992 then attr6 is carriage means the shape has assigned category carriage. The rules are generated with combinations of values at attributes hence for each category rules generated are in different number in each rule at the end the rule number for that category is given e.g. for rule 1 it is actually 18th rule for

carriage. The category wise distribution of rules is shown in Figure 6.8. The classifier could classify maximum 75.71% objects correctly with the fold wise average classification 61.78%. It is observed that average coverage of RS classifier is 64.65%. Table 6.7 gives annotation accuracy for each fold with the percentage of classified shapes i.e. coverage.

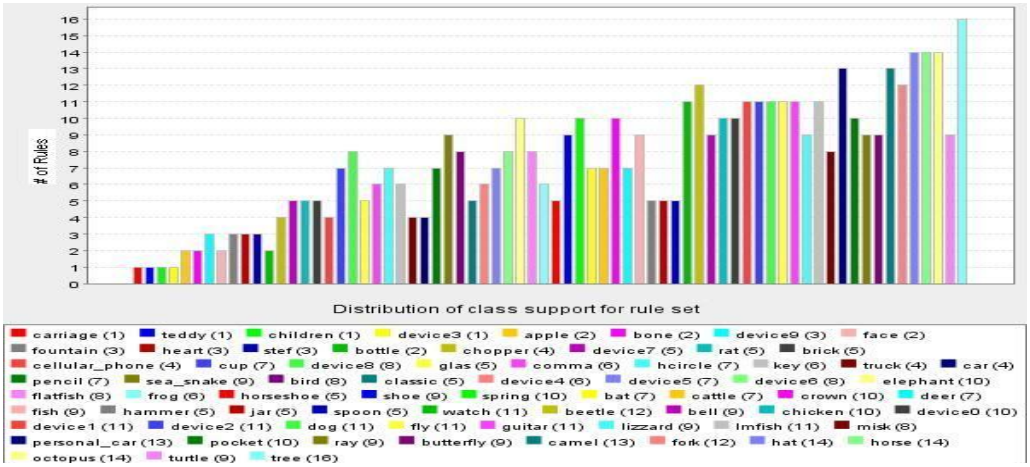

Figure 6.8. Category wise rule distribution.

	fold1	fold2	fold3	fold4	fold5	fold6	fold7	fold8	fold9	fold10
Correctly annotated	100	84	95	81	87	83	86	86	96	106
Annotation Accuracy (%)	71.42	60	67.85	57.85	62.14	59.28	61.42	61.42	68.57	75.71
Coverage (%)	66.4	64.3	65	62.9	60.7	61.4	65	62.1	70.7	72.9

Table 6.7. Fold wise annotation performance of Rough Set classifier.

6.3.4 K-Nearest Neighbor (K-NN)

The shape annotation is carried out on test data containing five columns for input features and seventy bits for output. In the sample data given in records ten samples are taken for 10 categories. In output one bit is 1 because the decision is made in one against all type of method. e.g. category apple is represented by [1,0,0,0,0,…,0,0,] where one 1 bit is followed by 69 zeros and for category watch is represented by [0,0,0,0,0,…,0,1,].

[Input]	[Output]
[28279, 0.159546, 0.92057, 0.701695, 776.5656]	[1,0,0,0,0,0,0, … ,0]
[29902, 0.89822, 0.638563, 0.355824, 1374.126]	[0,1,0,0,0,0,0, … ,0]
[19100, 0.85, 0.368555, 0.177774, 2530.505]	[0,0,1,0,0,0,0, … ,0]
[20495, 0.680076, 0.887268, 0.565629, 689.394]	[0,0,0,1,0,0,0, … ,0]
[26425, 0.838545, 0.636594, 0.359026, 1030.916]	[0,0,0,0,1,0,0, … ,0]

[29232, 0.985921, 0.472284, 0.18652, 1390.972]	[0,0,0,0,0,1,0, … ,0]
[10014, 0.873863, 0.669251, 0.436911, 710.056]	[0,0, … 0,1,0, … ,0]
[4270, 0.868872, 0.842708, 0.657328, 340.708]	[0,0, … 0,0,0,1,0,0]
[24289, 0.8346, 0.79856, 0.553936, 830.725]	[0,0, … 0,0,0,0,1,0]
[19603,0.994046,0.706058,0.524677,1067.487]	[0,0, … 0,0,0,0,0,1]

K-NN classifier take decision for all 140 test object images in every fold. Out of these 140 image maximum 120 objects are correctly classified for annotation. The details of fold wise experimental details are stated in table 6.8.

	fold1	**fold2**	**fold3**	**fold4**	**fold5**	**fold6**	**fold7**	**fold8**	**fold9**	**fold10**
Correctly annotated	112	105	120	104	107	113	115	110	118	118
Annotation Accuracy (%)	80	75	85.71	74.28	76.42	80.71	82.14	78.57	84.28	84.28
Coverage (%)	100									

Table 6.8. Fold wise tagging accuracy of k-NN classifier.

Using K-NN coverage is achieved 100% because the technique works nonlinearly and tries to find nearest class that may result in misclassification. Table 6.9 shows performance analysis of overall classification (correct annotation) of DT, RS and K-NN. Thus annotation using DT is 56.07%. Rule extraction using rough sets improves the performance to 61.78% and K-NN given 80.35% output. Here the classifiers have stability as demonstrated by standard deviation (SD)

The term ‘coverage’ is used to represent the number of classified (either correct or wrong) objects. It is observed that the coverage using decision tree and rough sets is not 100%. With rough sets, even though cut supported discretized feature vector based rules could not adequate enough to cover all unseen objects.

Classifier	**Performance (%)**			**SD**
	Min	**Max**	**Mean**	
Decision Tree	51.42	60.71	56.07	0.02
Rough Set / Rule Extraction	57.85	75.71	61.78	0.05
K- Nearest Neighbor	74.28	85.71	80.35	0.02

Table 6.9. Analysis of DT, RS and K-NN classifiers annotation performance.

6.3.5 Overall Analysis of Shape annotation

The annotation accuracy analysis shows K-NN performance is best compare to decision tree, rough sets and all variants of ANN based classifiers. Table 6.10 shows cross tabular analysis of overall classification (correct annotation) of all classifiers. The annotation accuracy using ANN with single network is 58.57%. ANN in hierarchy gives 66.42% with domain expert suggested hierarchy which is replaced by

fuzzy clustering improves annotation results to 74.64%. Using K-NN based classifier the annotation accuracy is improved up to 80.35%.

Classifier		Annotation Accuracy (%)
K- Nearest Neighbor (10 fold)		**80.35**
Decision Tree (10 fold)		56.07
Rough Set Rule Extraction(10 fold)		61.78
Artificial Neural Network (applied on 280 objects)	Single (3 hidden layer 30, 30, 35)	58.57
	Domain expert suggested Hierarchical	66.42
	Fuzzy clustering based Hierarchical	74.64

Table 6.10. Overall analysis of shape annotation.

6.4 Image Tagging Using Machine Learning

Corel dataset is used for automatic image categorization and annotation experimentations. Out of 1000 images, 500 images are used for training purpose whereas remaining 500 images are used for testing. The experiment is extended using Caltech 256 image dataset with 15,30,45,60 images for training and rest for testing.

Two types of experiments are carried out on color images. In first ANN, DT, RS, K-NN classifiers are tried out on every full image without considering segmented regions. The images are representing color and texture features. Color features are extracted globally and locally given Global Color Feature (GCF) vector of size 3 and Local Color Feature (LCF) vector of size 48. Gabor filter is used to extract texture given Texture Feature (TXF) vector of size 60. In combination of local and global color with texture gives two vectors Global Color Texture Feature (GCTXF) vector of size 63 and Local Color Texture Feature (LCTXF) vector of size 108. In the second type of experiment segmented images with regions are used to extract features and MIL based K-NN is used as a classifier for annotation.

6.4.1 Artificial Neural Networks (ANN)

MLP is experimented with five feature vectors with different network architectures. Network design is tested with different combinations of hidden layers and number of neurons to determine the best tagging performance. Network is designed using single neural network classifier that generates output 4 bit since categories are 10. The size input features is 3, 48, 60, 63 and 108 for global color, local color, texture, global

color with texture and local color with texture respectively. For each feature a input neuron is required hence number of neurons required are equal to size of input feature vector. In output layer since categories are ten, 4 bits are sufficient for 10 categories. For global color feature input/output mapping is given by the 10 records from feature vector representing each category. The first three values are input values while next four are output bit representing a category e. g [-1,-1,-1,-1] represents Africa.

[Input]	[Output]
[0.373459326,-0.205529728, -0.300245098]	[-1,-1,-1,-1]
[-0.359490551, 0.156838757, 0.229779412]	[-1,-1,-1, 1]
[0.302793755,-0.154147296,-0.260416667]	[-1,-1, 1,-1]
[0.120583402,-0.168827991, 3.31E-02]	[-1,-1, 1,1]
[6.29E-02, 0.179593834,-0.318627451]	[-1, 1,-1,-1]
[7.33E-02, 5.24E-02,-0.392156863]	[-1, 1,-1,1]
[0.867091208,-0.484462931,-0.689338235]	[-1, 1,1,-1]
[0.334428924,-0.278933203,-0.340073529]	[-1, 1, 1, 1]
[0.333607231, 6.61E-02,-0.582107843]	[1,-1,-1,-1]
[2.86E-02,-0.163934426, 0.161151961]	[1,-1,-1, 1]

Activation function used is tan sigmoid and learning is done using back propagation algorithm. The layers and the neurons in each layer are finalized by taking various combinations of neurons from 1 to 30 and hidden layers 1 to 2. The sample ANN output of Local Color and Texture Features (LCTXF) experiments are shown in Table 6.11.

No of Hidden Layers	No of PEs in 1st HL	No of PEs in 2st HL	MSE	Overall Accuracy (%)	Training Accuracy (%)	Testing Accuracy (%)
1	1	-	0.589063	2.1	2.6	1.6
1	2	-	0.454157	4.7	5.6	3.8
1	6	-	0.112674	69.9	87.8	52
1	16	-	0.070078	72.1	94	50.2
1	26	-	0.071001	74	93.4	54.6
1	30	-	0.03967	75.6	96.6	54.6
2	1	1	0.580929	2	2.6	1.4
2	1	2	0.530485	10	9.8	10.2
2	1	4	0.546878	16.4	18	14.8
2	1	16	0.816593	8.5	8.2	8.8
2	1	28	0.886939	12	13.4	10.6
2	1	30	0.648705	14.4	17	11.8
2	2	1	0.525536	0.5	0.6	0.4
2	2	2	0.52984	1.9	2.6	1.2
2	2	4	0.403573	32.4	39	25.8
2	4	8	0.235879	57.2	69.8	44.6
2	4	10	0.191697	60.7	80	41.4

No of Hidden Layers	No of PEs in 1st HL	No of PEs in 2st HL	MSE	Overall Accuracy (%)	Training Accuracy (%)	Testing Accuracy (%)
2	4	30	0.100685	68.7	91	46.4
2	6	1	0.507512	15.4	18	12.8
2	6	28	0.06572	73.4	94	52.8
2	8	28	0.061002	71.1	94.4	47.8
2	10	16	0.239854	57	76.8	37.2
2	12	6	0.058514	75.4	95	55.8
2	12	8	0.058455	75.1	95.4	54.8
2	14	8	0.066052	74.5	93	56
2	14	10	0.066147	73.7	94.6	52.8
2	16	8	0.061498	74.1	94.8	53.4
2	**16**	**10**	**0.050114**	**76.5**	**95.2**	**57.8**
2	16	12	0.046896	74.8	95.4	54.2
2	20	14	0.035981	76.2	97.2	55.2
2	20	22	0.031949	76.7	97.4	56
2	22	26	0.02556	76.3	97.8	54.8
2	22	28	0.032928	75.2	96.8	53.6
2	30	14	0.246291	59.8	76	43.6
2	30	16	0.038892	74.5	97	52

Table 6.11. Sample tagging Accuracy by single neural network.

The tagging accuracy and layer details with all five vectors are given in table 6.12.

Features	Artificial Neural Network			
	No of Hidden Layers	No of PEs	MSE	Output (%)
Global Color Features (GCF)	2	(30,24)	0.558511	21.6
Texture Features (TXF)	1	30	0.168169	29.4
Global Color and Texture Features (GCTXF)	2	(12,10)	0.13765	43.6
Local Color Features (LCF)	2	(14,16)	0.060098	55.2
Local Color and Texture Features (LCTXF)	2	(16,10)	0.050114	57.8

Table 6.12. Analysis of Categorization and Tagging Performance of ANN.

	Africa	Beaches	Building	Bus	Dinosaur	Elephant	Flower	Food	Horses	Mountain
Africa	19	1	3	3	2	5	1	12	2	2
Beaches	3	16	6	8	0	3	0	1	4	9
Building	9	3	11	13	0	4	0	6	1	3
Bus	3	4	4	35	0	0	0	3	0	1
Dinosaur	0	0	0	0	47	1	1	0	1	0
Elephant	4	8	3	4	0	22	0	4	2	3
Flower	1	0	1	1	0	1	45	1	0	0
Food	5	0	2	5	0	9	4	22	1	2
Horses	5	5	0	0	0	0	0	0	37	3
Mountain	1	8	0	4	0	1	0	1	0	35

Table 6.13. Confusion matrix generated by ANN classifier using LCTX features.

The all five feature vectors by applying classifier generates five confusion matrices. The table 6.13 gives confusion matrix using LCTXF feature vector. From these matrices it is observed that using LCTX the results are improved for dinosaur and flowers categories. For categories horses and mountain the accuracy is improved cause of addition of texture features in color features.

6.4.2 Decision Trees (DT)

Decision tree divides data set into pieces not larger than a predefined size. After decomposition the fragment represented as terminal nodes or leafs in decision tree. In order to perform data decomposition, i.e. to construct a decision tree that can be further used as a classifier first is to make decomposition. The parameters to be set when starting decomposition algorithm are Maximal size of leaf, Discretization in leafs, Shortening ratio. After decomposition the test table is classified using a decision tree generated. There has to exist at least one decomposition tree object, which means that it has to be calculated in prior to classification. Figure 6.9 shows screen for setting parameters for making decomposition.

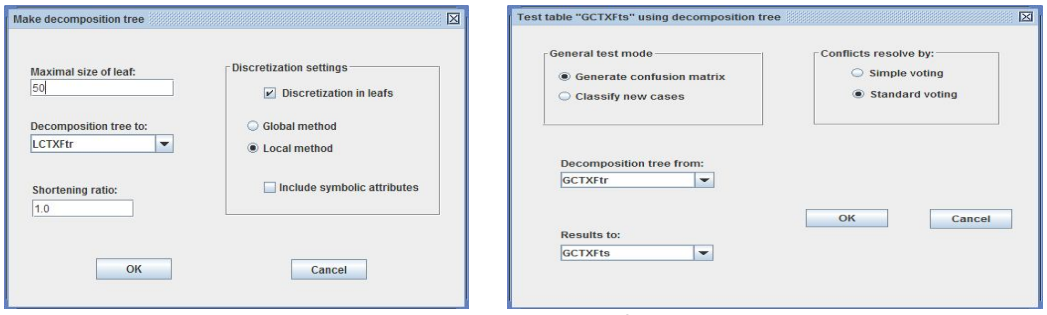

Figure 6.9. Making decision tree and classification using decision tree.

The dialog window and the available classification options are as in figure 6.9. The main option selection is type of voting (simple or standard). The test mode is selected either for calculating confusion matrix or to drive last column of data that is name of class for test tuple. Experiments are carried out using all five combinations of feature vectors. First the decision trees are generated and then test data is classified using decision tree. Table 6.14 gives labeling accuracy of decision trees.

Features	Correctly Classified	Accuracy (%)	Coverage (%)
Global Color (GCF)	111	22.2	67.2
Local Color (LCF)	156	31.2	72.2
Texture(TXF)	102	20.4	65
Global Color Texture (GCTXF)	133	26.6	67.6
Local Texture Color(LCTXF)	167	33.4	64.2

Table 6.14. Analysis of Categorization and Tagging Performance of DT.

Accuracy of tagging using texture features is least compare to other four combinations of features. Global color features (slope of RGB histogram) individually gives low accuracy in tagging but it is improved as combined with texture. Color features extracted locally (grid based) gives features of patches in the image gives better results compare to global color and texture. Combining local colors with texture gives best results. Annotation accuracy is improved up to 33.4%. The best results are obtained using local color in combination with texture (LCTX). Table 6.15 gives confusion matrix for LCTX features.

	Africa	Beaches	Building	Bus	Dinosaur	Elephant	Flower	Food	Horses	Mountain
Africa	11	2	1	3	2	6	3	6	1	0
Beaches	2	8	5	2	0	0	1	1	3	6
Building	7	2	6	6	1	1	0	1	1	3
Bus	2	3	4	6	2	1	0	2	1	4
Dinosaur	1	0	0	3	40	1	0	0	0	0
Elephant	5	1	4	2	1	15	0	1	2	2
Flower	0	1	3	2	0	0	22	7	0	1
Food	5	1	2	2	0	4	7	12	1	1
Horses	0	0	1	0	0	8	0	2	18	0
Mountain	2	7	0	2	1	1	0	0	1	13

Table 6.15. Confusion matrix for LCTX using decision tree classifier.

The confusion matrices generated using five types of features shows that in five categories Africa, Bus, Dinosaur, Elephant, Horses local color along with texture outperforms other four types. Global color features along with texture for Building, Flower, categories gives better results since images from these categories have high color strength for full image. Figure 6.10 shows graph demonstrating category wise tagging performance with respect to features.

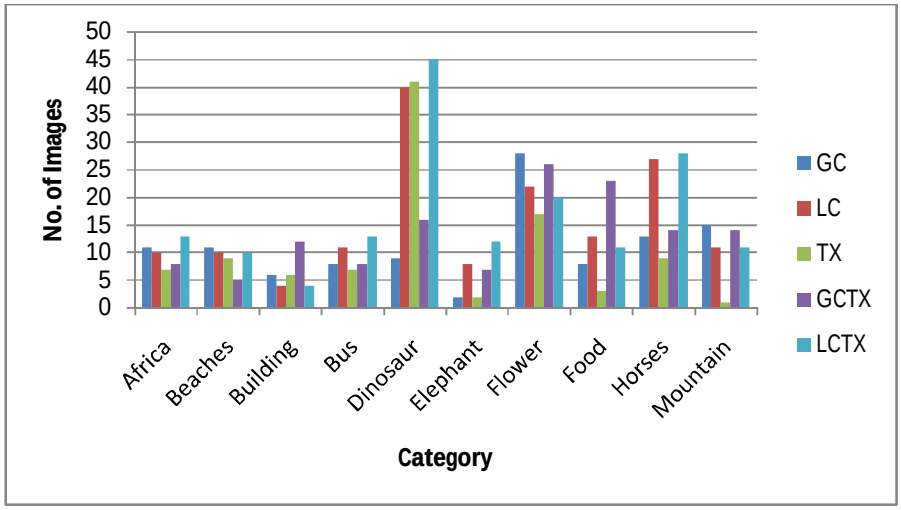

Figure 6.10. Category wise tagging performance of Decision tree classifier.

6.4.3 Rough Sets (RS)

Rough set works with the help of rules generated from the dataset. Decision rules make it possible to classify objects, i.e. assign the value of decision attribute. Before generating the rule the data must be descretized by using cut set. The cut set is generated on train data by selecting method of cut generation. Once a cut set generated train and test tables can be descretized. Figure 6.11 shows cut set generation and descretization steps.

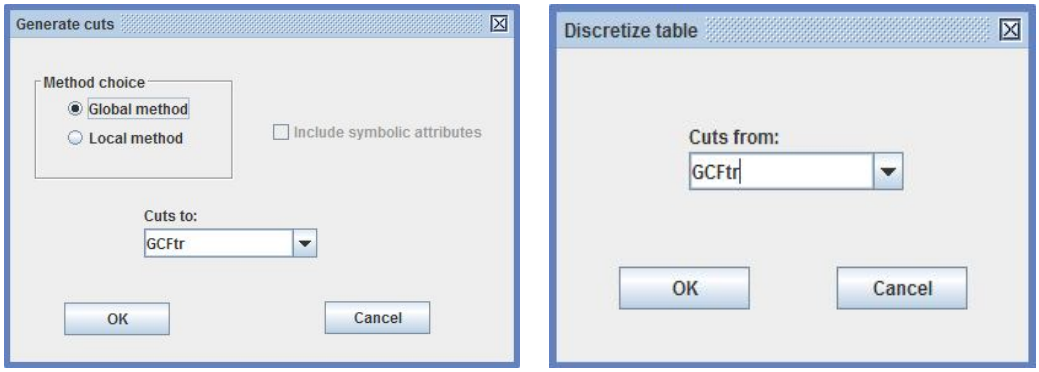

Figure 6.11. Descritization of data using cut set generation.

The descretized data is in range to generate rules. The rules generation and classification using these rules is carried out using Rough Set Exploration Systems (RSES). The rule generation is carried out using four algorithms viz. exhaustive, genetic, covering, LEM2 algorithms. These algorithms are experimented to classify the test data. It is observed that genetic algorithm gives better results compare to other algorithms. For genetic algorithms speed is to selected in range Low, Medium or High. Cover parameter is to be set only for LEM2 algorithm. The dialog for selecting algorithm and setting parameters is as shown below.

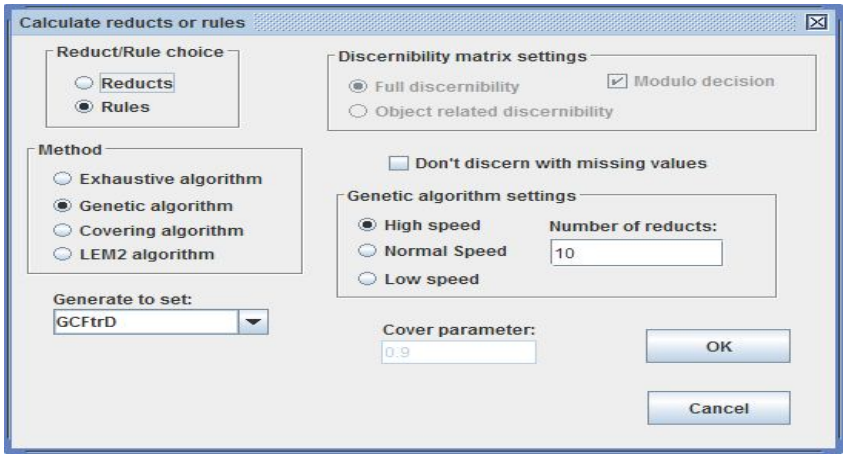

Figure 6.12. Rule generation for rough set classifier.

The statistical information is shown in Figure 6.13 for rules generated for global color features.

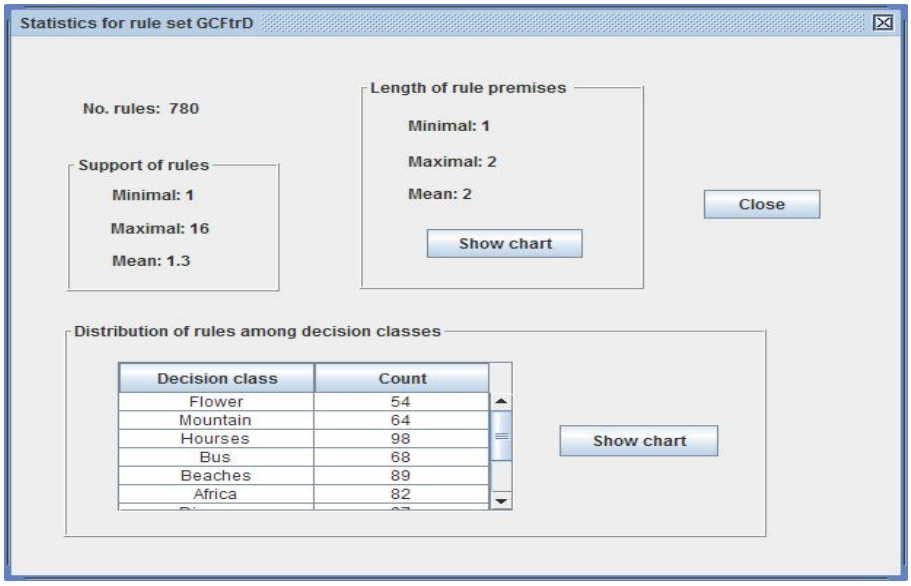

Figure 6.13. Statistical information for rules generated.

All five feature combinations are experimented using rules generated. Table 6.16 gives accuracy of labeling using rough set classifier for annotation. It is observed that the coverage is improved 100% compare to decision trees. Only for global features using rough sets it is 79.4% i.e. for 20.6% image objects are not classified to any category.

Features	Correctly Classified	Tagging Accuracy (%)	Coverage (%)
Global Color(GCF)	131	26.5	79.4
Local Color(LCF)	270	54	100
Texture(TXF)	151	30.2	100
Global Color Texture (GCTXF)	242	48.4	100
Local Texture Color(LCTXF)	286	57.2	100

Table 6.16. Analysis of Tagging Performance of Rough Set classifier.

The accuracy of global features is low compare to other four combinations of features. Though global color features (slope of RGB histogram) individually gives low accuracy in tagging but it is improved as combined with texture. Color features extracted locally (grid based) gives features of regions in the image gives better results compare to global color and texture. The results are improved by combining local colors with texture up to 57.8%. The results are improved using local color in combination with texture (LCTX) for six categories i.e. Beaches, Dinosaur, Elephant, Flower, Food and Mountain. Table 6.17 gives confusion matrix for LCTX features.

	Africa	Beaches	Building	Bus	Dinosaur	Elephant	Flower	Food	Horses	Mountain
Africa	15	2	2	4	3	4	2	14	4	0
Beaches	1	12	12	4	2	2	0	2	9	6
Building	1	1	9	13	5	4	4	5	4	4
Bus	5	4	10	19	2	1	1	3	2	3
Dinosaur	0	0	0	0	50	0	0	0	0	0
Elephant	4	1	2	5	1	27	0	2	3	5
Flower	0	0	1	1	0	0	44	3	1	0
Food	2	0	0	5	5	2	5	29	2	0
Horses	0	1	1	0	0	2	0	0	46	0
Mountain	0	3	2	3	2	4	1	0	0	35

Table 6.17. Confusion matrix for LCTX using Rough Set classifier.

Considering all confusion matrices generated using five types of features it is observed that in category Dinosaur 100% performance is achieved by local color, global color and texture as well as local color and texture. Flower images are classified with respect local color feature as its performance is same after adding texture features. While Texture along with global color classifies images from Bus and building better that other features. Figure 6.14 shows graph demonstrating category wise tagging performance with respect to features.

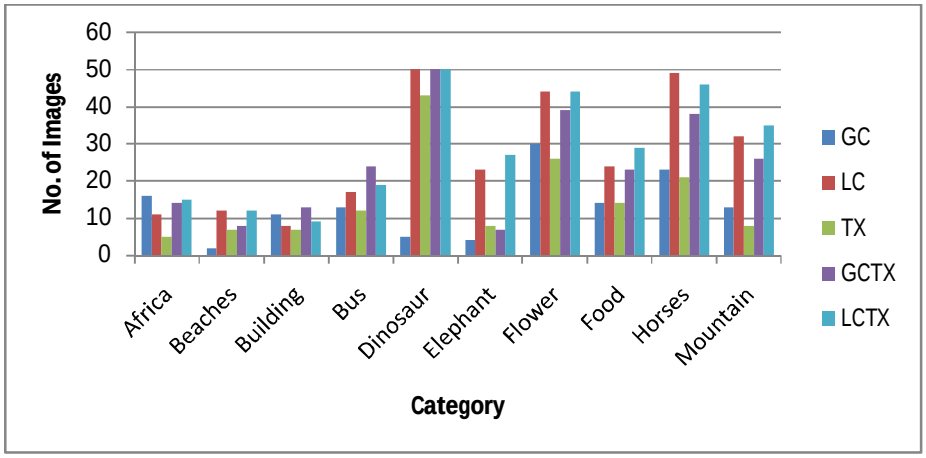

Figure 6.14. Category wise tagging performance of rough set classifier.

6.4.4 K-Nearest Neighbor (K-NN)

The k-nearest neighbor (K-NN) classification method does not require a separate training step. It is an instance-based classification technique. During experiments both absolute and euclidian distances are calculated and majority voting algorithm is used to classify. The experiments are carried out by gradually increasing k from 1 to 100. The value of k suggesting number of neighbors is finalized by analyzing the accuracy of tagging in above experiments. Image tagging for test data is done on final k

selected to calculate confusion matrix. Following table gives annotation accuracy for all five feature vectors at each value of k.

The test data contains columns for input features and ten bits for output. In the sample data given in records below the size of input is three as it represents global color feature vector. Ten samples are taken for 10 categories with record for each category. In output one bit is 1 cause the decision is made in one against all type of method. e.g. [1,0,0,0,0,0,0,0,0,0] corresponds to Africa while [0,1,0,0,0,0,0,0,0,0] corresponds to beaches.

[Input]	[Output]
[0.300739523,0.10472229,-0.606617647]	[1,0,0,0,0,0,0,0,0,0]
[-0.162284306,-2.69E-03,0.246017157]	[0,1,0,0,0,0,0,0,0,0]
[0.481922761,-0.139466601,-0.545343137]	[0,0,1,0,0,0,0,0,0,0]
[0.292317173,-0.122339124,-0.281862745]	[0,0,0,1,0,0,0,0,0,0]
[8.69E-02,0.112551994,-0.269607843]	[0,0,0,0,1,0,0,0,0,0]
[0.397493837,-0.139466601,-0.416666667]	[0,0,0,0,0,1,0,0,0,0]
[0.922555464,-0.472229019,-0.784313725]	[0,0,0,0,0,0,1,0,0,0]
[0.387017256,-0.134573036,-0.410539216]	[0,0,0,0,0,0,0,1,0,0]
[0.223089565,0.160019574,-0.533088235]	[0,0,0,0,0,0,0,0,1,0]
[-0.248562038,-0.141913384,0.546875]	[0,0,0,0,0,0,0,0,0,1]

	Annotation Accuracy(%)						Annotation Accuracy(%)				
K	**GCF**	**LCF**	**TXF**	**GCTXF**	**LCTXF**	**K**	**GCF**	**LCF**	**TXF**	**GCTXF**	**LCTXF**
1	32.6	**58.8**	28.6	44.4	**62**	51	39.6	53	31.8	48	56
2	27.4	54.8	27.4	42.4	57.6	52	39.4	52.8	31.4	48.2	56.2
3	31.6	57.8	30.4	45	61.2	53	39.6	53.4	31.6	48.6	56.4
4	33.8	57.2	28.8	47.4	61.6	54	39.6	53.6	31.6	48	56.8
5	32.8	57.8	28.2	48	62	55	39.2	53.2	31.2	47.8	56.6
6	35.8	57.6	25.8	50	60.6	56	39.6	53.4	31.2	47.6	56.6
7	36.4	56.6	27.8	50.2	61.2	57	39.8	53	31	47.2	56.6
8	37	56.2	29.4	50.8	60.4	58	39.4	53.4	31.2	47.2	55.8
9	37.4	57.6	28.6	51.4	60.8	59	39.6	53.2	31.4	47.4	56.4
10	35.8	56	28.4	51.6	60.4	60	39.4	53.4	31.6	47.2	55.8
11	38.2	57	28.2	51	59.6	61	39.6	53.6	31.2	47.8	55.6
12	36.6	56.2	29.4	51.6	58.6	62	39.6	53.6	31.8	47.2	56
13	35.8	56.2	28.8	51.4	58.6	63	40.2	53.6	32.4	47.6	55.4
14	36.2	55.8	28.4	50.8	58.6	64	40.4	52.8	31.6	47.2	56
15	36.4	55.4	29.8	**52.2**	58.2	65	40	52.4	32.2	47.8	56
16	37.4	55.8	30.8	51.4	58.6	66	40	52.4	32.6	47.8	55.4
17	37.8	55.8	29.8	50.4	59	67	39.6	52.2	32	48.2	55
18	38	55.4	29.8	49.8	58.4	68	39.4	52.4	32.4	47.8	54.8
19	37.4	53.8	30.2	51.2	58.6	69	39.6	52.8	32.2	47.8	55.4
20	37.2	54.6	30.2	51.2	58	70	39.8	53	32.2	47.8	54.6
21	37.6	55	32.6	50	58.4	71	39.8	53.6	33	48	54.6
22	37.6	55.2	31.8	50.4	58.2	72	39.8	53.8	33.2	47.2	54.2
23	37.4	56	32.2	50.8	57.6	73	39	53.2	33.2	46.4	54.6
24	37.8	55.8	32.2	50	57.6	74	38.4	52.6	32.6	46.4	55
25	38.8	54.8	32.4	49.6	57.8	75	39	52	33.2	45.6	56
26	39.8	54.8	32	49.4	57.4	**76**	38.2	52.2	**33.6**	46	54.8
27	39	54	30.6	49.6	57.8	77	38.6	52	32.6	47	54.4

K	Annotation Accuracy(%)					K	Annotation Accuracy(%)				
	GCF	LCF	TXF	GCTXF	LCTXF		GCF	LCF	TXF	GCTXF	LCTXF
28	39.4	53.4	30.4	49.2	57.8	78	38.4	52.2	32.2	47.4	55
29	39.2	54.2	31.4	50	58.2	79	39.4	51.6	32.6	46.6	55
30	40.4	53.8	31.8	48.2	57.4	80	39	51.4	32.4	47.2	54.8
31	42	54	30.8	50.2	58	81	39	51.2	32.4	46.8	54.8
32	**42.6**	53.6	31.8	50.8	57.6	82	38.8	50.6	32.6	47.4	54.8
33	42	54	31.6	50.8	57.6	83	39.4	50.2	31.8	46.6	54.8
34	42	54.8	32.4	51.2	58.2	84	39	50.2	31.6	46.6	54.4
35	40.8	55.6	33.2	51.2	58	85	38	50.8	32.2	46.4	54.2
36	40.4	55.4	32.2	51	58.4	86	37.6	50.6	32	46.2	55.4
37	40.6	54.6	33	50.4	58.2	87	38.2	51	31.6	46.4	54.8
38	40.8	54.4	32.6	50.4	57.2	88	38.4	50.6	32	46.4	54.4
39	40.4	54.8	32.6	49.6	57	89	38.2	51.2	32.2	46.4	53.8
40	40.2	54.8	33.2	48.8	56.6	90	38.4	51.4	32.4	46.4	54.6
41	39.8	54.4	33.4	49.6	56	91	37.8	52	32.2	47	55
42	39.6	54.2	33	49	57	92	37.6	51.6	32.2	47	54.2
43	39	54.4	33	49	56.8	93	37.6	51.4	31.8	47.4	54
44	39	54.4	32.4	49	56.2	94	37	51.6	31.6	47.4	54.6
45	39	53.6	32.2	48.8	56.2	95	36.6	51	31.8	47.2	54.2
46	39.4	53.8	31.8	48.4	56	96	36.8	51	31.8	47.6	55
47	39.6	53.8	30.8	48.2	56.4	97	36.8	51	31.4	47.4	54.8
48	39.6	53.6	30.4	48.4	57.6	98	36.6	50.4	31.4	47.2	54.6
49	39.6	54.4	31	47.8	56.6	99	36.4	50.8	31.6	47.2	54.4
50	40	53.8	31.4	48	56.4	100	36.4	51	31.2	46.8	54.6

Table 6.18. Annotation accuracy of K-NN for all feature vectors with increasing K.

Tagging performance table above is summarized in table 6.19 with best accuracy in each feature category along with number of neighbors and number of correctly annotated images. The accuracy of texture features is low compare to other four combinations of features. Though global color features (slope of RGB histogram) individually gives 42.6% accuracy but it is improved as combined with texture up to 52.2%. Color features extracted locally (grid based) gives features of regions in the image gives better results compare to global color and texture. The results shows 58.8% correct annotation for local color features and are improved by combining local colors with texture up to 62%.

Features	Correctly Classified	Nearest Neighbors	Tagging Accuracy (%)
Global Color(GCF)	213	32	42.6
Local Color(LCF)	294	76	58.8
Texture(TXF)	168	15	33.6
Global Color Texture (GCTXF)	261	1	52.2
Local Texture Color(LCTXF)	310	1	62

Table 6.19. Analysis of Tagging Performance of K-NN classifier.

Africa, Bus and Mountain categories shows improved results with global color features. The results obtained using local color in combination with texture (LCTX)

for categories Dinosaur, Elephant, Flower, Food, Horses are improved compare to all experiments. Confusion matrix for LCTX features is given in table 6.20. Considering all confusion matrices generated using five types of features it is observed that category Dinosaur gives 98% accuracy of tagging using local color, global color and texture as well as local color and texture. Images from building category are classified with respect local color feature but most other categories are classed to buildings due to similarity in features especially with beach images. Texture along with global color classifies images from beach category better that other features. Figure 6.15 shows graph demonstrating category wise labeling accuracy with respect to features.

	Africa	Beaches	Building	Bus	Dinosaur	Elephant	Flower	Food	Horses	Mountain
Africa	21	1	12	3	2	7	0	3	1	0
Beaches	5	8	22	3	1	1	0	0	5	5
Building	1	2	35	7	1	1	0	3	0	0
Bus	5	0	21	23	0	0	0	1	0	0
Dinosaur	0	0	1	0	49	0	0	0	0	0
Elephant	2	2	9	2	0	32	0	2	1	0
Flower	0	0	4	2	0	0	41	3	0	0
Food	5	0	10	2	1	3	1	28	0	0
Horses	0	0	1	0	0	1	0	0	48	0
Mountain	0	3	12	7	1	1	0	1	0	25

Table 6.20. Confusion matrix for LCTX using K-NN classifier.

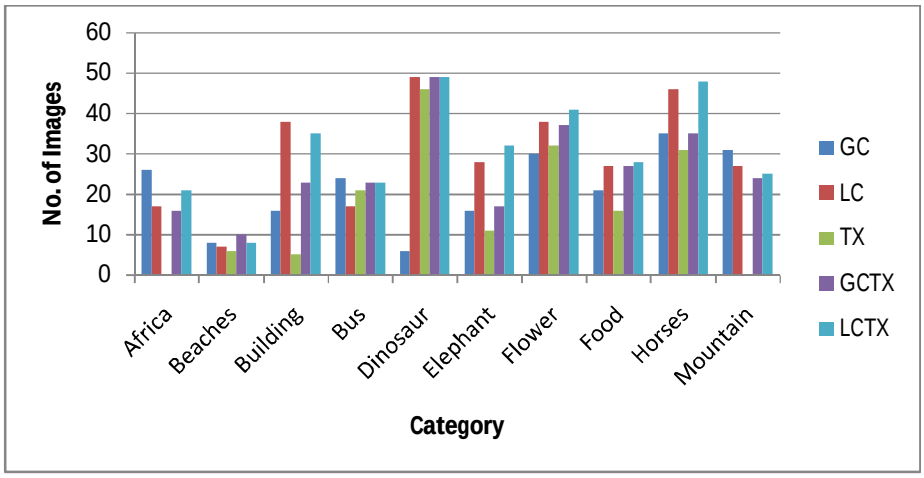

Figure 6.15. Category wise tagging performance of K-NN classifier.

6.4.5 Multiple Instance Learning based K-Nearest Neighbor (MIL-KNN)

Supervised learning for the training data with incomplete knowledge about lables is modified for classification purpose. This changed type of learning is known as multiple instance learning (MIL). In supervised learning, a discrete or real-valued label is assigned to every training instance. In comparison, in MIL the bags of instances are with labels[Dietterich et al., 1997].

The k-nearest neighbor rule, also called the majority voting k-nearest neighbor, is one of the oldest and simplest non-parametric techniques in the pattern classification literature. Classification using multiple instance learning can be done using K-NN by changing to the regular K-NN algorithm.

K−NN classification has two stages; the first is the determination of the nearest neighbors and the second is the determination of the class using those neighbors by simple majority voting or by distance weighted voting. For the determination of the nearest neighbors the distance between two bags is computed by using Hausdorff distance. By definition, two sets A and B are Hausdorff distanced if and only if, every point of A is within distance d of at least one point of B, and every point of B is within distance d of at least one point of A[Edgar, 2008].

Formally, given two sets of points

$A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$,

The Hausdorff distance is defined as:

$$H(A,B) = \max\{h_{\max}(A,B), h_{\min}(A,B)\} \quad (6.7)$$

Where maximal distance is calculated for $h(A,B)$ as

$$h_{\max}(A,B) = \max_{a \in A} \min_{b \in B} \| a - b \|$$

Where minimal distance is calculated for $h(A,B)$ as

$$h_{\min}(A,B) = \min_{a \in A} \min_{b \in B} \| a - b \|$$

In second step the class label is assigned to the new query object x', based on the majority voting class of its nearest neighbors:

$$c' = \arg\max \sum_{x_i^{NN} \epsilon T'} I\left[c = c_i^{NN}\right] \quad (6.8)$$

where c is a class label, c_i^{NN} is the class label for the i-th neighbor among k nearest neighbors of the query object. $I\left[c = c_i^{NN}\right]$, an indicator function, takes a value one if the class c_i^{NN} of the neighbor x_i^{NN} is the same as the class c and zero otherwise.

The three feature vectors color, texture and color along with texture are multidimensional vectors. The experiments are carried out by gradually increasing k from 1 to 25 for both minimum and maximum Hausdorff distance on all the three feature vectors. The neighbors are decided by analysis of results obtained to finalize k

for the experiments to tag unlabeled images. Following table gives annotation accuracy for all three feature vectors at each value of k.

K	Annotation Accutacy(%)					
	Color Features(CF)		Texture Features(TXF)		Color and Texture Features(CTXF)	
	MAX HD	MIN HD	MAX HD	MIN HD	MAX HD	MIN HD
1	**57.8**	36.6	32.4	22	59.8	22.4
2	52.2	33.2	33.2	21.6	57.2	20.6
3	55.2	37.8	33.2	24.8	59	23
4	56	38.2	33.6	27.6	59.6	24
5	57	38.6	33.6	24	60.4	27
6	56.2	38.8	33.8	23.8	60	27.8
7	56.6	39.6	35	22.6	60.6	29.2
8	56.4	38.6	35	23.2	59.6	28.2
9	57.2	40	36.2	23.2	59.4	29
10	55.6	**40.4**	37	22.2	60.2	28.6
11	55.4	41	36	23.4	60	29.2
12	54.2	40.4	36.4	24.6	59.8	28.6
13	52.8	39.4	36.4	25.4	**60.8**	29.8
14	53.2	38.4	37	25.8	60	30.6
15	53	38.8	38	26.2	59.2	30.4
16	53.4	38.8	39	25.6	60.2	30.2
17	53.8	40.4	38.2	27.2	58.4	29.2
18	55.4	40.4	37.2	**27.8**	57.6	30
19	53.6	40	37.6	27.2	57.4	**31.4**
20	53	40.2	38.4	27.8	57	30.6
21	53	39	**39.2**	26.6	57	30.4
22	53	38.8	39	27.2	56.4	30.6
23	53.6	38	38.8	26.8	55.8	31
24	54	39.6	38.2	25.8	56.8	30.8
25	52.8	39	39.2	26	55.8	30.6

Table 6.21. Annotation accuracy of MIL based K-NN for all feature vectors.

Tagging performance above is summarized in table 6.22 with best accuracy in each feature category along with number of neighbors and number of correctly annotated images.

Feature Extraction	K-NN					
	MaxHD			MinHD		
	Correctly Classified	Nearest Neighbors	Output (%)	Correctly Classified	Nearest Neighbors	Output (%)
Color Features (CF)	289	1	57.8	202	10	40.4
Texture Features (TXF)	195	21	39.2	139	18	27.8
Color and Texture Features(CTXF)	304	13	60.8	157	19	31.4

Table 6.22. Analysis of Tagging Performance of MIL based K-NN classifier.

All three combinations of features are experimented with maximal and minimal Hausdorff distance. For K-NN classifier at k=13 and features vector is CTXF the accuracy obtained is 60.8%. Improvement in accuracy is achieved in categories dinosaur and mountain because the texture features extracted using gabor wavelet gives magnitude of energy of the regions which is similar for these categories.

Bus, Food, Horses and Flower categories shows good results using color features. The results are improved using color in combination with texture (CTX) for categories Africa, Beaches, Building, Dinosaur , Elephant and Mountain. Confusion matrix for CTX features is given in table 6.23.

	Africa	beach	buildings	buses	dinosaurs	elephants	flowers	horses	mountains	food
Africa	30	1	5	5	6	1	0	0	2	0
beach	1	27	13	1	0	1	0	0	0	7
buildings	9	4	27	2	1	2	3	0	0	2
buses	4	1	1	40	3	0	0	1	0	0
dinosaurs	0	0	0	0	50	0	0	0	0	0
elephants	5	11	8	0	3	22	0	0	0	1
flowers	6	0	1	4	0	0	28	6	5	0
horses	10	0	0	7	6	0	5	20	1	1
mountains	1	1	0	0	0	0	0	2	46	0
food	2	14	12	4	0	3	0	0	1	14

Table 6.23. Confusion matrix for CTX features using MIL based K-NN classifier.

From all five confusion matrices generated using five feature combinations; it is observed that category Dinosaur gives 100% tagging accuracy using color and texture. Category dinosaur is containing single object images so easy to classify. While food contains many mix objects misleading classification. Africa is also a category containing mix types of textures and colors and due to it objects of other categories are misclassified as Africa.

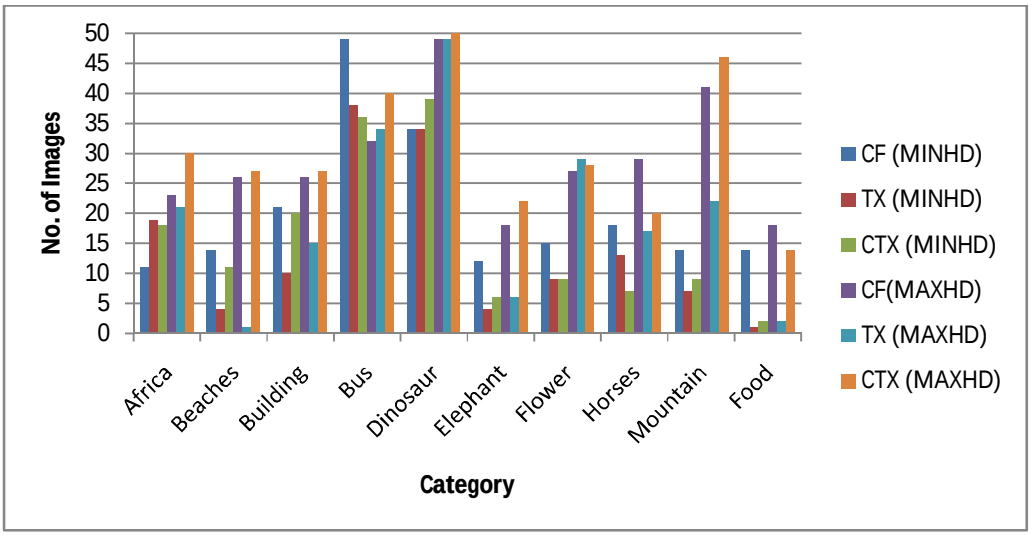

Figure 6.16. Category wise tagging accuracy of MIL based K-NN classifier.

The above experiment is continued on Caltech 256 image dataset. As per the literature studied the training is done on 15, 30, 45 and 60 images from each category and rest dataset is used for testing. The annotation accuracy at no of training samples 60 on color and texture features improved up to 38.44%.

Training Samples	15				30			
Distance	maxHD		minHD		maxHD		minHD	
	NNs	Accuracy(%)	NNs	Accuracy (%)	NNs	Accuracy (%)	NNs	Accuracy (%)
Color Features	1	17.22	1	12.45	2	19.44	2	13.66
Texture Features	2	19.64	3	15.57	12	27.68	11	16.27
Color and Texture Features	1	28.3	2	19.45	13	32.12	12	23.15
Training Samples	**45**				**60**			
Distance	**maxHD**		**minHD**		**maxHD**		**minHD**	
	NNs	**Accuracy (%)**	**NNs**	**Accuracy (%)**	**NNs**	**Accuracy (%)**	**NNs**	**Accuracy (%)**
Color Features	1	21.78	1	16.54	1	24.74	2	17.55
Texture Features	11	26.44	12	19.65	11	28.66	12	20.18
Color and Texture Features	14	35.45	15	24.12	14	**38.44**	13	26.32

Table 6.24. Caltech 256 Tagging Accuracy of MIL based K-NN classifier.

6.5 Conclusion

The shape images are classified since the objects are having difference in eccentricity, circularity ratio and extent in shape contour. The features selected in final feature vector are proportion of the shape properties. The representation of shape in feature form is rotation, translation and scaling invariant so machine learning techniques can classify and annotate. The shape boundaries of categories, viz. bottle, cellular phone, fork, octopus, rat, watch, exhibits sufficiently different features so that can discriminate a category from other.

Using single feedforword neural networks the tagging accuracy measured is 58.57%. If the contour is similar, the features extracted are similar though the shapes are different and hence lead misclassification of the shapes. The category e.g. device0, device1, device2, device3 are similar in contour so features representing ratio of circularity, rectangularity are similar and hence neural network misclassify these objects. In order to improve the initial tagging performance of ANN classifier we transformed the network to fuzzy clustering supported hierarchical network architecture. The experimental data shows that using single neural network the accuracy was improved by introducing hierarchical classifier up to 66.42%. Since domain expert suggested hierarchy cannot form a better hierarchy Fuzzy c-mean clustering is used and the results are improved up to 74.64%. In the hierarchy the similar categories are classified in a big category first then in next levels of hierarchy the further classification is done. The features extracted are similar for cellular phone, flat fish, car, shoe, apple, hat, bell, heart, and circle for the first level hence all are classified in a single group. In the next level further it is classified like circle, heart and apple are similar so in a group and other are in other groups.

With decision trees the decision making at every node is done in binary, rough set uses boundary approximation and then with respect to upper and lower bound the decision is made this improved the tagging accuracy compare to decision tree. The accuracy of tagging can be further improved with selection of more effective shape features. K-NN is an instance-based classification technique. Comparing the results of the shape annotation performance of classifiers it is observed that K-NN classifier outperforms all the classifiers with accuracy 80.35%.

During the experiments for annotating color images, it is observed that local color along with texture out performs other feature combinations. It is experimented that with K-NN classification accuracy is improved. The results using K-NN show that single object images e.g. dinosaur classify up to 98% but in case of beach images features are similar to building so misclassify beach to buildings.

Decision trees are not able to discriminate since from the raw features only if-else based binary tree is generated and used to classify. Comparatively rough sets give better results since during rule formation approximation boundaries plays the role. With category dinosaur rough sets gives 100% results and for categories flowers and horses results are improved. Experiments using MIL based learning; it is observed that L, u, v color strength in the image regions along with texture out performs other feature combinations with K-NN classifier. The results show that single object images with features strongly discriminating e.g. dinosaur classify up to 100%. Improvement is achieved in case of mountain category since the luminance level of these images is high and it refers in color features so as to discriminate from other images. The problem with beach images to misclassify in buildings is resolved using MIL based learning since regions are separated from images.

* * *