A Multi Agent Approach for Texture Based Classification and Retrieval (MATBCR) using Binary Decision Tree

R. Baskaran **M. Deivamani** **A. Kannan**

Department of Computer science and Engineering,
Anna University, Chennai – 600 025, INDIA
baaski@cs.annauniv.edu, m_deivamani@yahoo.com, kannan@annauniv.edu

Abstract: *Texture Analysis has been used in a range of studies for recognizing synthetic and natural textures. We propose a simple, novel and yet effective method for classifying and retrieving images based on texture descriptor. In our system, the information needed for classifying the different types of textures are extracted from the Gabor features, Co-occurrence matrices and Law's Method. For Feature pre-selection the contextual merit algorithm is used along with the decision tree. For Multi-class classification, SVM with Binary Decision Tree is used. Canberra distance metrics is used for similarity computation. A Multi-Agent system consists of a group of distributed Texture agents that organize their knowledge, goals and plans. In addition, it supports relevance feedback. Our MATBCR model results have been compared with other Texture Based Retrieval System and better prediction accuracy has been observed.*

Keywords: *Texture Analysis, Statistical Classification, Support Vector Machines, Multi – Agents, Binary Decision Tree, Optimization and Relevance Feedback.*

Received: *September 06, 2004* | ***Revised:*** *January 15, 2005* | ***Accepted:*** *February 10, 2005*

1. Introduction

In recent years, very large collections of images and videos have grown rapidly. In parallel with this growth, content-based retrieval and querying the indexed collections are required to access visual information. In order to index and answer queries that the users pose to seek features derived from visual content of the images. These features usually contain information's such as Color, Texture, Shape and Spatial relations to represent and index the image. Image texture has emerged as an important visual primitive to search and browse through large collections of similar looking patterns. Several researchers have explored use of texture for content-based retrieval.

When designing an image classification and retrieval system, the following issues should be addressed.

- How to extract the features of an image resourcefully.
- How to classify the images
- How to compare two images, i.e. the definition of the image similarity measure.
- How to make it interactive, i.e. the strategy of relevance feedback.

Texture methods are categorized as: statistical, geometrical, structural, model-based and signal processing features. However, no one computational method works well for all textures, unlike the human visual system. Combining one or more techniques can improve the retrieval performance, as judged by human users. For this issue, the main texture features are currently derived from Gabor filters, Co-occurrence matrices and Law's method. It is well known that correlated and irrelevant features may degrade the performance of the system. So, doing the feature pre-selection using the contextual merit along with the decision tree optimizes these features. In the feature pre-selection the relevant features that are required to classify the image have been chosen. More number of relevant features may help us in retrieving more similar images from the database.

To solve the second issue, the Support Vector Machines with Binary Decision Tree is most useful in classification problems. Binary Decision Tree: this method, also called *tennis tournament*, builds the binary tree where each node represents SVM. Each class is regarded as player, and in each match the system classifies the texture feature according to the decision of the SVM trained on the pair of classes. The process continues until the class is identified.

To solve the third issue, the similarity / distance measures are used. Experimental results have proved that the performance can be improved significantly by using Canberra metric as compared to traditional approaches, such as Euclidean, Mahalanobis, Manhattan.

To solve the fourth issue, the relevance feedback is used.

In this paper, we propose the framework to address the above four issues. For the feature extraction, we adopt Gabor Filter bank, Co-occurrence matrices and Law’s method, which has an excellent performance in extracting texture features of the image. The feature pre-selection is performed using the contextual merit algorithm along with decision tree. To classify the images, SVM with Binary Decision tree is used. Performance of the different similarity metrics, such as Euclidean, Canberra, Mahalanobis, Bray-Curtis and Manhattan are experimented. To make the system more interactive the relevance feedback is used.

The organization of the paper is as follows: Section (2) describes the Texture features and feature pre-selection. The Classification process using SVM with Binary Decision Tree is presented in Section (3). Section (4) describes the Similarity measures. The proposed System Architecture is presented in Section (5). In Section (6**)** we provide Experimental results that evaluate all aspects of the framework. Section (7) concludes with a discussion of the future work.

1.1. Texture

Texture is an important property of many types of images. Image Texture has a number of perceived qualities, which play an important role in describing texture. Laws [7] identified the following properties as playing an important role in describing texture: uniformity, density, coarseness, roughness, regularity, linearity, directionality, frequency, and phase.

Texture Analysis has been used in a range of studies for recognizing synthetic and natural textures. Three different types of texture analysis are considered: segmentation, classification and synthesis. Texture Segmentation deals with detecting the texture boundaries in an image. Texture Classification deals with the recognition of image regions using texture properties. Each region in an image is assigned a texture class. Texture methods are categorized as: statistical, geometrical, structural, model-based and signal processing features [8]. Weszka et al. [2] compared the Fourier spectrum; Second order Grey-level statistics, Co-occurrence statistics and Gray level run length statistics and found the co-occurrence were the best. Manjunath and Ma [1] have shown that image retrieval using Gabor features outperforms efficiently. Ojala et al. [11] compared a range of texture methods using nearest neighbor classifiers including Grey-level difference method, Law's measures, Center-Symmetric covariance measures and local binary patterns applying them to Brodatz images. The best performance was achieved for the gray level difference method. Law's measures are criticized for not being rotationally invariant, for which reason other methods performed better.

1.2 Multi-Agent

Agents are defined as software or hardware entities that perform some set of tasks on behalf of users with some degree of autonomy. Agents are being advocated as a next generation model for engineering - complex, distributed systems [9]. The complexity of many problems can drive a developer to consider the use of multiple agents in developing a solution. But, while increasing the number of agents might simplify the work each individual agent must accomplish, there is a corresponding increase in the complexity of the division of labor and the communication between agents. This, in turn, ensures that there is a smooth growth in agent-oriented approach in the Image Retrieval System. The performance of Agent based image retrieval is based on three key terms for addressing a problem. They are *Decomposition, Abstraction and Organization*. The objective of agent-based approach gives one a clear way to decompose the problems into agents. The agents should possess the adaptive behavior, which doesn’t need to specify all interactions in advance.

2. Texture Features

Texture is one of the most important features used in an image. In this section, we describe three groups of widely used texture features:

- Features based on co-occurrence matrix.
- Features based on 2D Gabor functions and
- Features based on Law's method.

2.1 Texture features based on co-occurrence matrix

In statistical approaches, the textures are described by statistical measures. One commonly applied and referenced method is the co-occurrence method, introduced by Haralick [7]. In the co-occurrence method, the relative frequencies of gray level pairs of pixels at certain relative displacements are computed and stored in a matrix, the co-occurrence matrix P. For G gray levels in the image, P will be of size G x G. If G is large, the number of pixel pairs contributing to each element, p_{ij}, in P will be low and the statistical significance poor.

On the other hand, if the number of gray levels is low, much of the texture information may be lost in the image quantization. Since the task is supervised segmentation, overlapping sub images will be used. As suggested by other researchers, the nearest neighbor pairs at orientations 0, 45, 90, and 135 degree are used in the project. Haralick [7] suggests 14 features describing the two-dimensional probability density function pij. Four features that commonly used are selected. There are the Angular Second Moment (ASM), Contrast (CON), Correlation (COR), and Entropy (ENT). They are given in the table 1.

Table 1. Some Texture Features Extracted From Gray Level Co-Occurrence Matrices.

Texture Feature	Formula
ASM	$\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} p_{ij}^2$
CON	$\sum_{n=0}^{G-1} N^2 \left\{ \sum_{\lvert i-j \rvert = n} p_{ij} \right\}$
COR	$\frac{1}{\sigma_x \sigma_y} \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} ij p_{ij} - \mu_x \mu_y$
ENT	$\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} p_{ij} \log p_{ij}$

where μ, σ are the means and the standard deviations corresponding to the distributions.

$$p_i^{(x)} = \sum_{j=0}^{G-1} p_{ij} \qquad (1)$$

$$p_j^{(y)} = \sum_{i=0}^{G-1} p_{ij} \qquad (2)$$

We extract each of these four features at each of the four different orientations, thus the feature vectors are 16-dimensional.

2.2 Texture features based on Gabor functions

Texture features are extracted using Gabor filter bank as in [1]. A 2D Gabor function g(*x, y*) and its Fourier transform G(u, v) can be written as:

$$g(x, y) = \left(\frac{1}{2\pi\sigma_x\sigma_y}\right)\exp\left[-\frac{1}{2}\left[\frac{x2}{\sigma 2x}+\frac{y2}{\sigma 2y}\right]+2\pi jWx\right] \qquad (3)$$

and

$$G(u, v) = \exp\left\{-\frac{1}{2}\left[\frac{(u-W)2}{\sigma_u 2}+\frac{v2}{\sigma_v 2}\right]\right\} \qquad (4)$$

Where, $\sigma_u = 1/2\ \pi\sigma_x$ and $\sigma_v = 1/2\ \pi\sigma_y$. Gabor function form a complete but non-orthogonal basis set. Expanding a signal using this basis provides a localized frequency description. A class of self – similar functions, referred to as Gabor Wavelet is discussed below.

Let *g (x, y)* be the mother Gabor wavelet, then this self similar filter dictionary can be obtained by appropriate dilation and rotations of one of *g(x, y)* through the generating function:

$$g_{mn}(x, y) = a^{-m} G(x, y), \quad a>1,\ m, n = \text{integer}$$

$$x' = a^{-m}(x \cos\theta + y \sin\theta)$$

and $$y' = a^{-m}(-x \sin\theta + y \cos\theta), \qquad (5)$$

where, $\theta = n\pi/K$ and K is total number of orientations. The Scale factor a^{-m} in (5) is meant to ensure that the energy is independent of m. In multi-resolution decomposition number of scales is equal to number of stages. Texture features are calculated with Gabor filter bank with four scales and six orientations. Applying these Gabor filters to an image results in 24 filtered images. The mean and standard deviation of each filtered images are calculated and taken as a feature vector.

$$\bar{f} = [\mu_{00}, \mu_{01}, ..., \mu_{35}, \sigma_{00}, \sigma_{01}, ..., \sigma_{35}] \quad (6)$$

where, the subscript represents the scale(0...,3) and orientations (0,.,5). The feature vector dimension is 48.

2.3 Law's texture features

Laws [7] observed that certain gradient operators such as Laplacian and Sobel operators accentuated the underlying microstructure of texture within an image. This was the basis for a feature extraction scheme based on series of pixel impulse response arrays obtained from combinations of 1-D vectors shown in Figure 1. Each 1-D array is associated with an underlying microstructure and labeled using an acronym accordingly. The arrays are convolved with other arrays in a combinatorial manner to generate a total of 25 masks, typically labeled as *L5L5* for the mask resulting from the convolution of the two L5 arrays.

Level L5 =	[1 4 6 4 1]
Edge E5 =	[-1 -2 0 2 1]
Spot S5 =	[-1 0 2 0 -1]
Wave W5 =	[-1 2 0 -2 1]
Ripple R5 =	[1 -4 6 -4 1]

Fig.1. Five 1-D arrays identified by Laws [7].

These masks are subsequently convolved with a texture field to accentuate its microstructure giving an image from which the energy of the microstructure arrays is measured together with other statistics. We compute 5 amplitude features mean, standard deviation, skewness, kurtosis and energy measurements. Since there are 25 different convolutions, altogether we obtain a total of 125 features.

2.4 Feature Pre-selection

Selecting the right set of features for classification is one of the important processes in the classification. Decision tree induction algorithm is employed for an automatic feature selection [3], [10]. While some other statistical classification algorithms require the feature subset to be in a preprocessing phase. The number of features used for classification can be reduced which is important for image interpretation tasks since feature calculation is a time consuming process.

Very often we don't know a-priori what the relevant features are for a classification task. For feature subset selection the contextual merit [2] is used. This algorithm employs a merit function based upon weighted distances between features. Assign a weightage to the features based upon how well they discriminate instances that are close to each other in the Euclidean space. The final sets of features are used for all further analysis.

3. Classification Using SVM With Binary Decision Tree

The basic SVM scheme works only for binary classification and in order to deal with a multi-class problem, a generalized scheme is introduced. In our System the SVM One Vs One Binary Decision Tree is used. This scheme is most efficient in terms of time required for training and classification.

A binary decision tree is a rooted, directed tree with two types of vertices: terminal vertices and non-terminal vertices.
Each non-terminal vertex *v* is labeled by a variable *var (v),* and has two successors:

- *Low (v)* corresponding to the case where *var (v)* is assigned 0 and

- *High (v)* corresponding to the case where *var (v)* is assigned 1.

Terminal vertices *v* has no children and is labeled by *value (v)* $\in \{0, 1\}$.

A binary decision tree for the two-bit comparator, given by the formula

$f(a1,a2,b1,b2) = (a1 \leftrightarrow b1) \wedge (a2 \leftrightarrow b2)$, is show in the below Figure 2.

We can decide if a truth assignment satisfies the formula as follows:

- Traverse the tree from the root to a terminal vertex.
- If variable *v* is assigned 0, the next vertex on the path will be *low (v).*

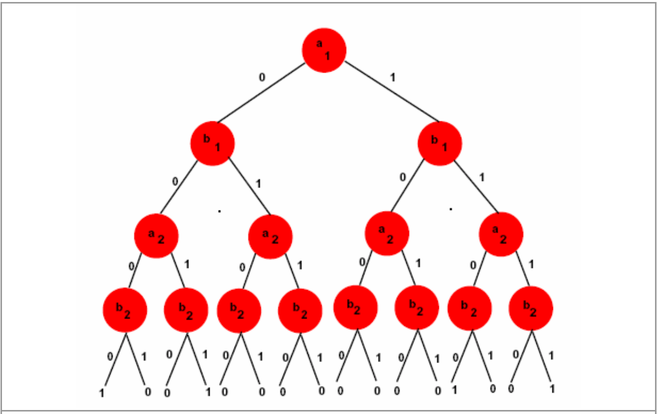

Figure 2. General Binary Decision Tree.

- If variable *v* is assigned 1, the next vertex on the path will be *high (v).*
- The value that labels the terminal vertex will be the value of the function for this assignment.

In the comparator example, the assignment

a1 ← 1, a2 ← 0, b1←1, b2← 1
leads to a leaf vertex labeled 0, so the formula is false for this assignment.

The Support Vector Machines with Binary Decision Tree takes advantage of both the efficient computation of the tree architecture and the high classification accuracy of SVM's. Although *(N-1)* SVM's need to be trained for an *N*-class problem, it only requires testing log_2N SVM's for the classification decision. In practice the overlap in the tree-structured decision regions may be allowed to reduce performance degradation with slightly more SVM's.

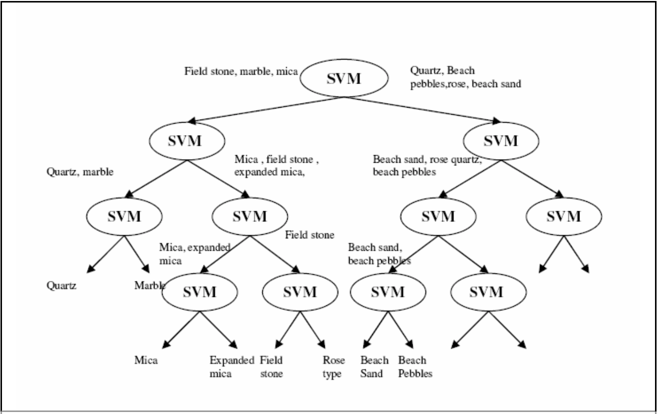
SVM
Field stone, marble, mica
Quartz, Beach pebbles,rose, beach sand
Quartz, marble
Mica , field stone , expanded mica,
Beach sand, rose quartz, beach pebbles
Field stone
Mica, expanded mica
Beach sand, beach pebbles
Quartz
Marble
Mica
Expanded mica
Field stone
Rose type
Beach Sand
Beach Pebbles

Figure 3. SVM with Binary Decision Tree

As shown in Figure 3, the SVM-BDT (Support Vector Machines with Binary Decision Tree) solves an *N*-class pattern recognition problem with a hierarchical binary tree, of which each node makes binary decisions with an SVM. The hierarchy of binary decision subtasks should be carefully designed before the training of each SVM classifier.

To reduce the performance degradation due to improper grouping we also allow overlaps between the two groups. Since the misclassification tends to occur with confusing samples on the decision boundary, the overlap actually provides a chance to correct those confusing samples and results in better performance.

3.1 SVM Classifier

3.1.1 Statistical Learning theory

The Support Vector Machines (SVM), the solution of the universal feed forward networks, pioneered by Vapnik [12], is known as the excellent tool for classification and regression problems of good generalization performance. In distinction to the classical neural networks the formulation of learning problem leads the quadratic programming with linear constraints.

3.1.2 Hypothesis Proposed by Vapnik and Chervonenkis

In the following we consider two-class classification and take the cost function to be the 0/1 loss function, i.e.,

$$C(fs(x), y) = \begin{cases} 1, iffs(x) \neq y \\ 0, otherwise \end{cases} \qquad (7)$$

so that the risk is the error rate. A principled way to minimize true error is to upper bound in probability the true error and minimizes the upper bound. This is the approach of statistical learning theory that leads to the formulation of the SVM.

Theorem *(Vapnik and Chervonenkis): Let H be an hypothesis space having VC dimension d. For any probability distribution P(x, y) on X* x{-1,+1}, *with probability 1-δ over random training sets S, any hypothesis f∈ H that makes k errors on S has error no more than*

$$\left[err_p(f_s) \leq \frac{k}{l} + \frac{2}{l}(d \log \frac{2el}{d} + \log \frac{4}{\partial}) \right] \qquad (8)$$

provided *d* ≤1 (i.e. the true error is less than the empirical error plus a measure of the capacity of the hypothesis space). This leads to the idea of *structural risk minimization*. That is the empirical risk is minimized for a sequence of hypothesis spaces and the final hypothesis is chosen as that which minimizes the bound given in equation (8).

4.Similarity Measures

The distance metric can be termed as similarity measure, which is the key-component in Content Based Image Retrieval. In this section we address the common method used for similarity measures. In conventional texture image retrieval, the Euclidean or Mahalanobis distances between the images in the database and the query image are calculated and used for ranking. The query image is more similar to the database images, if the distance is smaller. If x and y are 2D feature vectors of database image and query image respectively. Then the distance metrics are defined as:

The Euclidean distance is

$$d_E(x, y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2} \tag{9}$$

The Mahalanobis metric is

$$d_{Mah}(x, y) = \sqrt{(x-y)' Cov(x) - 1(x-y)} \tag{10}$$

The Canberra and Bray – Curtis distance metrics are given in equation (11) and (12) respectively [3].

$$d_c(x, y) = \sum_{i=1}^{d} \frac{|x_i - y_i|}{|x_i| + |y_i|} \tag{11}$$

$$d_{BC}(x, y) = \sum_{i=1}^{d} \frac{|xi - yi|}{xi + yi} \tag{12}$$

The Manhattan distance metric is

$$d_M(x, y) = \sum_{i=1}^{d} |x_i - yi| \tag{13}$$

5. System Architecture of MATBCR

The Figure 4 depicts the architecture of the MATBCR System. The System MATBCR Provides the Storage and Retrieval of images based Texture descriptors. The Image Retrieval occurs through two phases.

1. Testing Phase
2. Training Phase

5.1 Testing Phase

In this phase, the Query Image is segmented and then the feature vectors are extracted using the Gabor filters. The classification is done based on the SVM with Binary Decision Tree.

5.2 Training Phase

In this Phase, the Image Set is segmented, extracted as in the Phase 1. In this Phase the SVM Based Learning (training) was performed. Both the feature vectors of the query image and the feature vectors of the image training set are compared for similarity. If there are similar images in the database then the similar images are retrieved according to their similarity. The Relevance Feedback was also used to achieve effective system.

5.3 Multi-Agent System

The multi-agent approach is modular and decentralized; it is a natural way of modeling problems frequently occurring in diverse domains. As a result, this approach can often enhance overall system performance, specifically along the dimensions of computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility and reuse. Multi-agent systems can be extremely powerful, but they neglect a powerful resource – the user. To create the best of all possible systems, we need to consider the strengths and weaknesses of human and machine reasoning, designing an approach that takes advantage of the strengths of each while minimizing the impact of their respective weaknesses. Figure 5 shows some of the applications of agents. They are particularly useful when working remotely from the server, and for processing data that can be presented in a convenient form.

Different types of agents used in our system. They are:

1.Texture Modernizing Agent: This agent is used to monitor any updates to the database. Similarly monitors the processes that are related to mean, mean feature and image database.

2.Texture Quality Agent: It gathers the feature weights of the image. It also monitors and assigns the weights to the image feature during the feature pre-selection.

3.Texture Similarity Agent: It monitors the similarity computation and retrieves the similar image as output. It also monitors the distance measures and manages the table. When there is more than one similar image in the table it assigns the weightage to the image according to their relevancy.

4.Texture Supervision Agent: It monitors the over-all development in the system, such as feature extraction, optimization, learning, classification, similarity computation and retrieval process.

5.Texture Catalog Agent: It gathers all the information that requires for cataloging the database. It indexes the images in the database. Each time it indexes the database when there is update and removal of an image.

6.Texture Agent**:** It extracts the texture features represented by the vectors. Also it organizes the process during the feature extraction and manages the extracted features than to update it properly to the database.

5.4 Texture Image Database

We have used the Brodatz [10] set of textures as our working set. While the Brodatz set represents only a subset of texture of interest it is still useful for performance evaluation.

Size of each texture is 512x512. Each 512x512 images are divided into sixteen 128x128 non-overlapping sub-images.

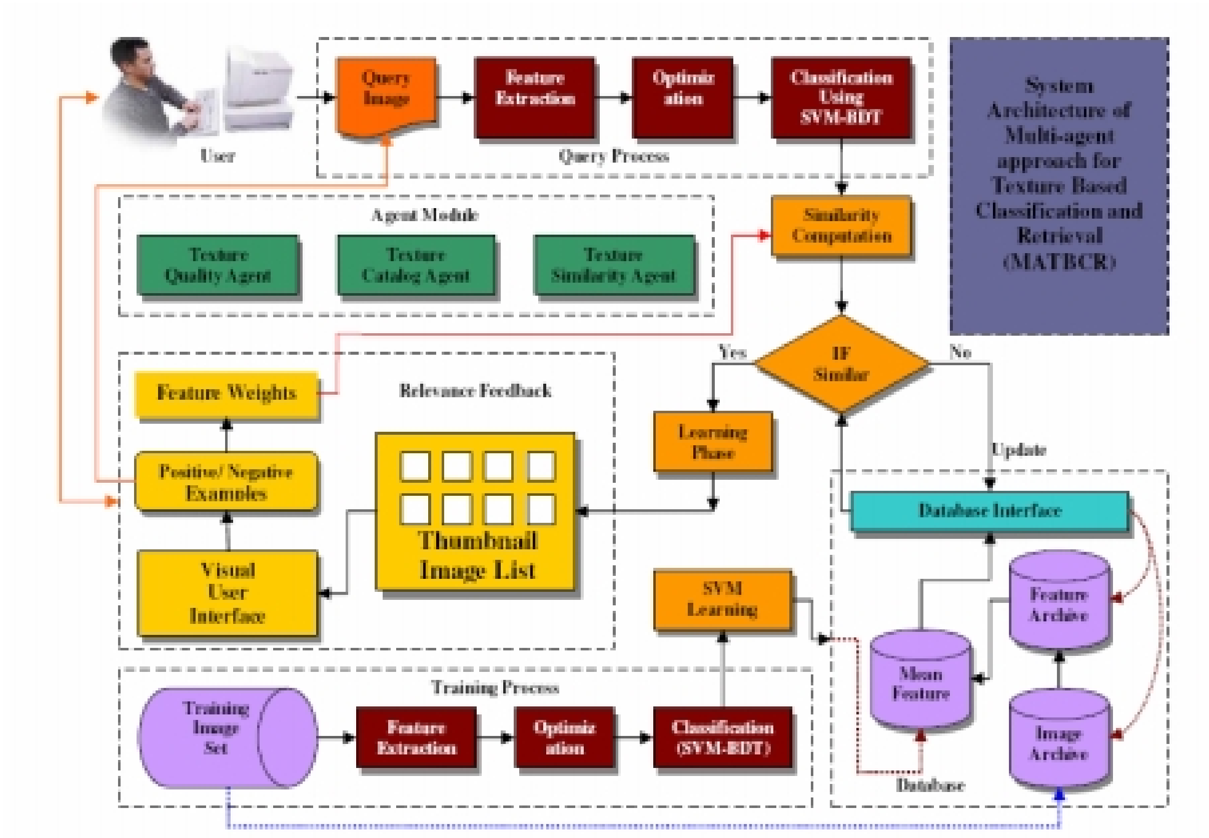

Figure 4. System Architecture of Multi-agent approach for Texture Based Classification and Retrieval using Binary Decision Tree

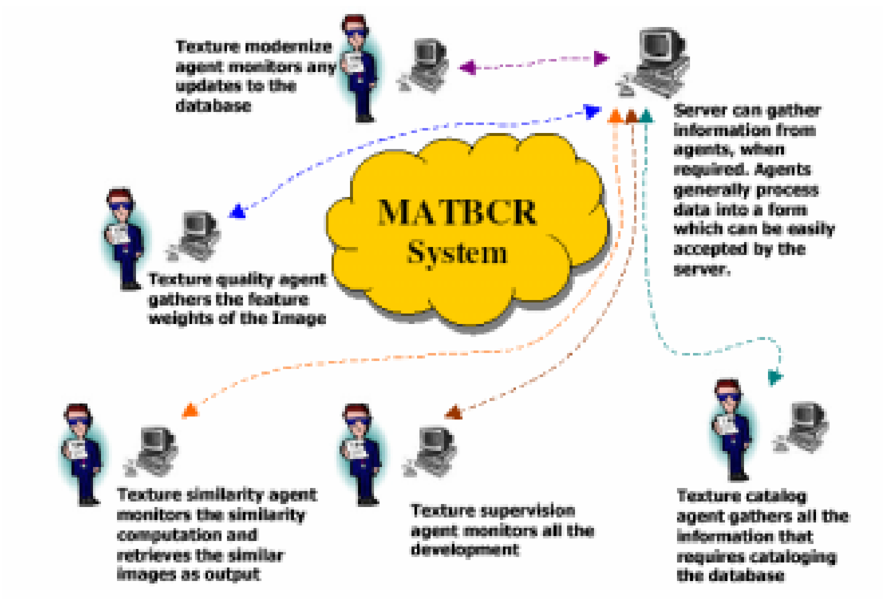
Texture modernize agent monitors any updates to the database
Server can gather information from agents, when required. Agents generally process data into a form which can be easily accepted by the server.
MATBCR System
Texture quality agent gathers the feature weights of the Image
Texture similarity agent monitors the similarity computation and retrieves the similar images as output
Texture supervision agent monitors all the development
Texture catalog agent gathers all the information that requires cataloging the database

Figure 5. Agents interact with MATBCR System

6. Experiment Results

Running the experiment with different metric on same set of images we are able to find which distance metric gives the best similarity retrieval of image. The Canberra metrics and Bray-Curtis distance metrics performed exceptionally well than other distance metrics. The fact is that in this metrics equation given in 11 and 12, the numerator signifies the difference and the denominator normalizes the difference. Thus distance values will never exceed one, being equal to one whenever either of the attributes is zero.
The GUI Interface of MATBCR System is shown in Figure 6, which shows the list of images present in the selected directory and includes the appropriate similarity computation metric.

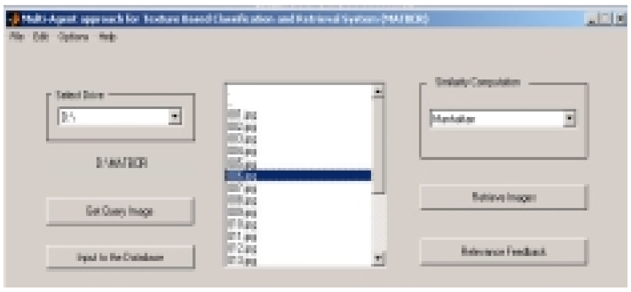

Figure 6. GUI Interface of MATBCR System

Figure 7 shows the query image selected from the directory for which similar images need to be found. For the given query image, top five retrieved results are displayed as the output and is shown in Figure 8.

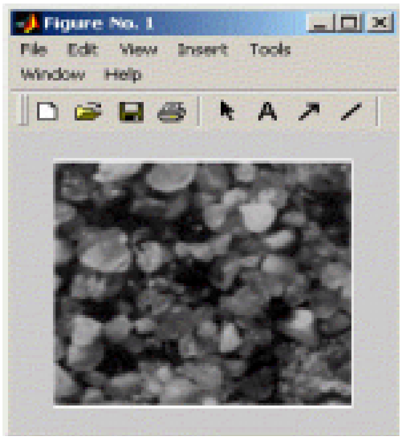

Figure 7. Query Image

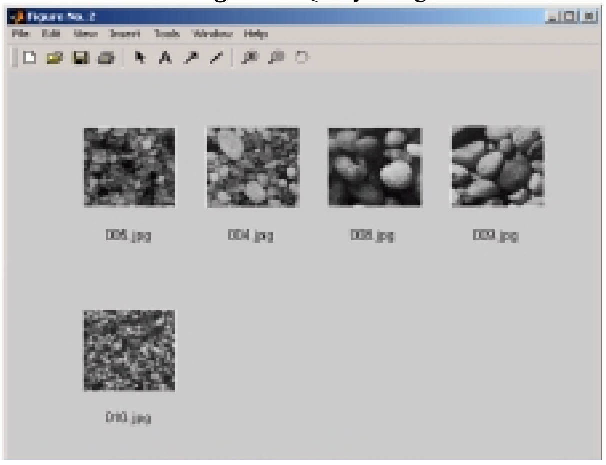

Figure 8. Top Five Relevant Images Retrieved by the MATBCR System using Manhattan Distance metric

Our content-based image retrieval system is first evaluated in terms of retrieval effectiveness. In order to evaluate effectiveness of retrieval systems, two well-known metrics, *precision* and *recall* [6], are used:

$$\text{Precision} = \frac{\text{No. of retrieved images that are relevant}}{\text{No. of retrieved images}} \quad (14)$$

$$\text{Recall} = \frac{\text{No. of retrieved images that are relevant}}{\text{Total number of relevant images}} \quad (15)$$

Based on the Recall and Precision definition we draw the graph with the Total Number of retrievals and is shown in Figure 9 and Figure 10 respectively.

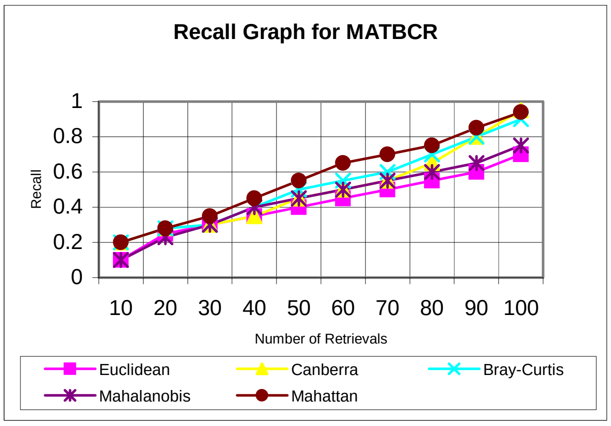

Figure 9. Recall Graph

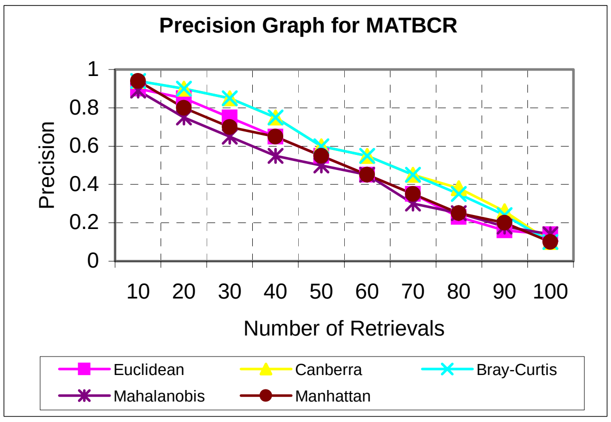

Figure 10. Precision Graph

The Percentage of retrieval of the MATBCR system done before optimization and after optimization is given in Figure 11 and Figure 12.

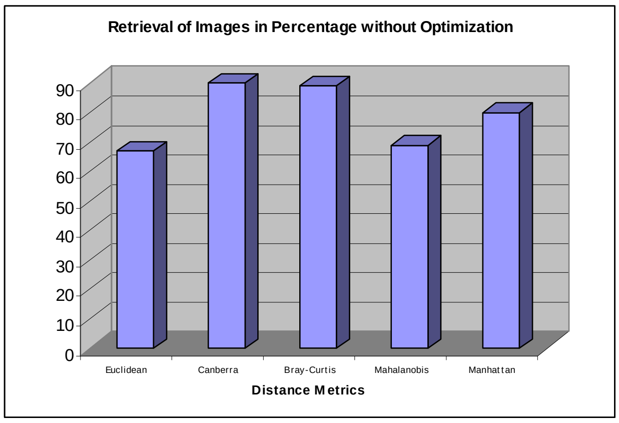

Figure 11. Retrieval Performance of the MATBCR system before Optimization

The Retrieval performance of the MATBCR system without optimization is shown in Figure 11. The result shows that the retrieval performance is at the maximum 89% for Canberra and Bray-Curtis and it reaches 94% incase of optimization. Hence optimization improves the retrieval effectiveness and it is shown in Figure 12.

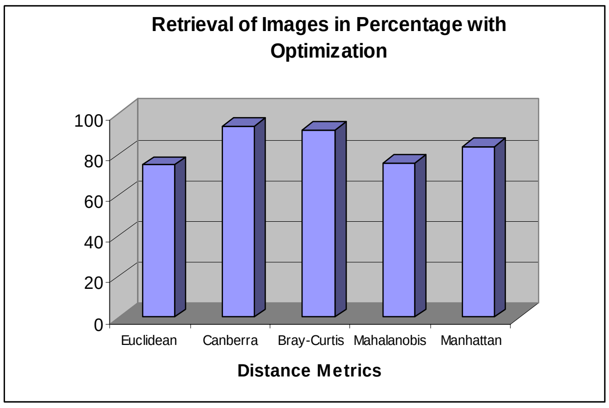

Figure 12. Retrieval Performance of the MATBCR system after Optimization

Table 2 shows the retrieval type of a MATBCR System for the different feature extraction methods and system takes almost 3.67 seconds to retrieve. The Retrieval performance in terms of time when compared to the number of images is shown in Figure 13.

Table 2. Retrieval Time evaluated using the MATBCR system

Feature Extraction Method	**Retrieval Time (seconds)**
Euclidean	3.56
Canberra	3.33
Bray-Curtis	3.35
Mahalanobis	3.67
Manhattan	3.34

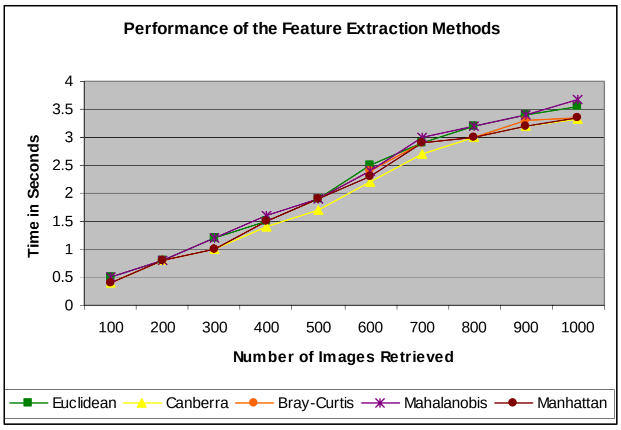

Figure 13. Performance of Feature Extraction Methods.

7. Conclusion

We have presented a detailed architecture of Multi Agent approach for Texture - Based Classification and Retrieval System (MATBCR). We have compared the retrieval efficiency of different distance metrics. The Retrieval efficiency of the Canberra and Bray-Curtis are good when compared to other distance metrics. The overall System Retrieval efficiency is 94%.

8. References

[1] B.S.Manjunath and W.Y. Ma Texture features for browsing and retrieval of large image data, IEEE Transaction on Pattern Analysis and Machine Intelligence, Volume: 18(8), pages 837-842, August 1996.

[2] Hong, S.J, Use of Contextual information for feature ranking and discretization, IEEE Transaction on Discovery and Data Engineering, Volume: 9, issue: 5, pages 718-730, September 1997.

[3] Jacobsen. C, Zscherpel . U, and Perner P. A Comparison between Neural Networks and Decision Trees, Pattern Recognition, Springer Verlag, pages 144-158, 1999.

[4] J.S.Weszka, C.R. Dyer and A.Rosenfeld, A Comparative study of texture measures for terrain classification, IEEE Transaction on System, Man and Cybernetics, Volume: 6, pages 269-285, 1976.

[5] Manesh kokare, B.N. Chatterji and P.K.Biswas Comparison of similarity metrics for texture image retrieval International conference on image processing, 2003.

[6] R.M Haralick, K.Shanmugam and I.Dinskin, Textural features for image classification, IEEE Transaction on System, Man and Cybernetics Volume: SMC-3 pages 610-621, 1973.

[7] Sungmoon Cheong, Sang Hoon oh, Soo-Young Lee, Support Vector Machines with Binary Tree Architecture for Multi-class classification, Neural Information processing- Letters and Reviews Volume: 2, No: 3, 2004.

[8] K.S. Jones. Information Retrieval Experiment, Butterworth and co. 1981.

[9] K.L. Laws, Textured Image Segmentation, Ph.D thesis, University of Southern California, 1980.

[10] Koller. D and Sahami. M, Toward Optimal Feature Selection, Proceedings of the Thirteenth International conference on Machine Learning, pages 284-292, 1996.

[11] M. Tuceyran and A.K.Jain, Texture Analysis, Hand Book of Pattern Recognition and computer vision, Chapter 2, World Scientific & Co, Singapore, pages 235-276, 1993.

[12] N.R Jennigs and Wooldridge, Agent-oriented Software Engineering, Handbook of Agent Technology, AAAI/MIT Press-2000

[13] P.Brodatz, Textures: A Photographic album for artists and designers, New York: Dover, 1966.

[14] T. Ojala, M.Petikainen, A Comparative study of texture measures with classification based on feature distributions, Pattern Recognition, Volume: 29, No: 1, pages 51-59, 1996.

[15] Vapnik, Statistical Learning Theory, John Wiley & Sons, 1998.