Block truncation coding with color clumps: A novel feature extraction technique for content based image classification

SUDEEP THEPADE¹, RIK DAS^{2,*} and SAURAV GHOSH³

¹Department of Information Technology, Pimpri Chinchwad College of Engineering, Pune 411044, India

²Department of Information Technology, Xavier Institute of Social Service, Ranchi, Jharkhand 834001, India

³A.K. Choudhury School of Information Technology, University of Calcutta, Kolkata 700 009, India

e-mail: sudeepthepade@gmail.com; rikdas78@gmail.com; sauravghoshcu@gmail.com

MS received 15 June 2014; revised 5 January 2016; accepted 21 March 2016

Abstract. The paper has explored principle of block truncation coding (BTC) as a means to perform feature extraction for content based image classification. A variation of block truncation coding, named BTC with color clumps has been implemented in this work to generate feature vectors. Classification performance with the proposed technique of feature extraction has been compared to existing techniques. Two widely used public dataset named Wang dataset and Caltech dataset have been used for analyses and comparisons of classification performances based on four different metrics. The study has established BTC with color clumps as an effective alternative for feature extraction compared to existing methods. The experiments were carried out in RGB color space. Two different categories of classifiers viz. K Nearest Neighbor (KNN) Classifier and RIDOR Classifier were used to measure the classification performances. A paired *t* test was conducted to establish the statistical significance of the findings. Evaluation of classifier algorithms were done in receiver operating characteristic (ROC) space.

Keywords. Classification; retrieval; color clumps; threshold; RGB; *t* test.

1. Introduction

Massive amount of image data generated everyday has been stored and maintained in digital form by the image databases. These huge datasets may be explored to find the correlations among them and to employ such correlations to create limited image categories inside the image database [1]. A crisp digest of the image content can be provided by generation of image groups to be used for effective means of image database management [2]. Thus, designing an efficient algorithm for extraction of feature vectors from the images has been the fundamental necessity for image classification. A new BTC based feature extraction algorithm named BTC with Color Clumps for content based image classification has been proposed in this work. This paper has compared the classification performances of existing feature extraction techniques with respect to the proposed one in two different classifier environments. A paired *t* test was conducted to establish the statistical significance of the results. The quantitative comparison has established the superiority of the proposed technique.

- The objective of the paper is to introduce a novel feature extraction technique.

- The novel technique has considerably decreased the feature vector size and has made it independent of image dimension.
- The proposed method has reduced the average time of feature extraction from each image in the dataset.
- The classification results have statistical significance of improved performance.

2. Related work

Primary step for classification involved feature extraction from image data out of the assorted mix of images. Threshold selection has been considered as an efficient tool for feature extraction from images after binarization. Selection of threshold has been imperative to differentiate the background of an image from its foreground. A number of factors including ambient illumination, variance of gray levels within the object and the background, inadequate contrast, etc. can influence threshold selection process [3–5]. At the outset, threshold selection can be carried out with three different techniques namely, mean threshold selection, local threshold selection and global threshold selection. Image classification has been executed earlier by considering mean threshold selection and using block

*For correspondence

truncation coding method as a tool for feature extraction for six different color spaces [6–8]. A recent approach has considered bit plane slicing for extraction of features using mean threshold selection for binarization of the images [6–8]. Another approach has considered ternary block truncation coding to extract the feature vector from the images [9]. A robust approach deploying multilevel mean threshold selection for block truncation coding has been used to categorize the images for six different color spaces [6–8]. Various image databases were classified using local discriminant models and global integration [10]. Some of the feature vector extraction approaches included averaging and histogram techniques [11, 12]. Vector quantization [11–13] and gradient were used to compute the texture. Another technique has explored binary BTC with even and odd images for feature extraction [14]. Local threshold selection techniques have been implemented by considering mean and standard deviation for threshold computation [15–21]. A feature extraction process has divided the image into 4×4 blocks and has extracted features of dimension 6 for each block [22]. It has characterized the local features of the images using wavelets. Global threshold selection has been constructively used for binarization of images with bimodal histogram [23, 24]. Modified Niblack's local threshold selection for significant bit planes has been effective for extraction of feature vectors but has higher time consumption for execution due to selection of significant bit planes [25]. Genetic algorithm was implemented to extract color and texture based features by 3D color histogram and Gabor filters [26]. Local descriptors of color and texture has been used for feature extraction from the Color moments and moments on Gabor filter by dividing the image into non-overlapping blocks [27]. Point features of interest were extracted from images for image identification [28]. Combination of color layout descriptor and Gabor texture descriptor for feature extraction has been used for retrieval purpose [29]. Color, texture and spatial structure descriptors were considered for feature extraction [30]. Features from images were extracted by using wavelet packets and Eigen values of Gabor filters [31]. Exploring the intra-class and inter-class feature extraction from images has improved image recognition results [32]. Fusion based techniques of feature extraction by Color Difference Histogram and Angular Radial Transform were also proposed [33]. A novel technique of feature extraction has been proposed by the authors based on color

clumps with Block Truncation Coding. The technique has outclassed the existing techniques of content based image identification and has shown statistical significance in contribution for better content based recognition.

3. Block truncation coding (BTC)

Block Truncation coding has been a simple compression algorithm devised in the year 1979 at the early stage of image processing. It segments the image into $n \times n$ (typically 4×4) non-overlapping blocks. Initially it was designed for the grayscale images and later extended for color images. In this algorithm the blocks were coded one at a time. The restructured block comprised of new values calculated from the mean and standard deviation for each block. The value of mean and standard deviation remains same as of the original block [34, 35]. This work has considered each color component as a block to implement the concept of block truncation coding. An example of block truncation coding considering R, G and B color component as blocks for RGB color space has been shown in figure 1.

4. Proposed technique (BTC with color clumps)

The proposed method has primarily implemented the concept of block truncation coding (BTC) by extracting each color component Red (R), Green (G) and Blue (B) as blocks from each of the images. Further, color clumps have been calculated for feature extraction from images in Wang database and the process was carried out in RGB color space. The color clumps have approximated the distribution of gray values in each color component and have created disjoint subsets of gray values for each color component. Color clump method logically divided the dimension of the intensity values in each color component R, G and B into different number of partitions at each stage of feature vector extraction. The mean threshold for each color clump was calculated and each gray value was compared with the threshold. The values higher than the threshold were clustered in the upper intensity group and the values lower than the threshold were clustered for the lower intensity groups. The mean of the two clusters thus formed was considered as the mean upper intensity and mean lower intensity for the corresponding clump. Thus, the number of feature vectors was twice the number of the color clumps formed and was independent of the size of the image. The proposed approach has shown maximum precision and recall values with 16 clumps. Hence, the feature vector size was 32 for each color component and 96 on the whole for three color components, irrespective of the dimension of the image. The partitions considered were in multiples of 2 as shown in figure 2(b). In this approach the partitions were ranged from 2 to 32. Thus,

Figure 1. Red, green and blue components as blocks.

Figure 2. (a) Formation of feature vectors from 2 clumps. (b) Distribution of color clumps and feature vectors (fv) for each color component in BTC with Color Clumps.

the number of stages of feature vector extraction was 5. The first stage through the fifth stage was named as BTC with 2 color clumps, BTC with 4 color clumps, BTC with 8 color clumps, BTC with 16 color clumps and BTC with 32 color clumps respectively. Another algorithm was

proposed for feature vector extraction which has divided the test images into blocks of 4×4 [22]. It has extracted localized features of training images using wavelets. The dimension of feature vectors for each block was considered as 6. Hence for an image of dimension 256×256 ,

the total number of $4 * 4$ blocks considered was 4096. The dimension of feature vector for each image was $4096 * 6 = 24,756$. The huge size of feature vector of 24,756 has increased computational overhead and complexity. It has slow training process for the classifier and has lengthened the process of image identification. The number of blocks was dependent on image dimension. Therefore, the number of feature vectors was also variable. In this paper, the proposed approach has a fixed feature vector size of 96 independent of the image dimension. The feature vector size was much smaller compared to the existing method [22] discussed above and have resulted in fast recognition of test images. The process of feature extraction has been demonstrated in Algorithm 1.

Algorithm 1

Begin

1. Input Image $x(i, j)$ of size $m * n$
2. Extract each Color Component Red ($r(i, j)$), Green ($g(i, j)$) and Blue ($b(i, j)$) from Image $x(i, j)$
3. Subsequently divide the intensity values logically into N ($N=2, 4, 8, 16, 32$) partitions with minimum intensity value $Int_{min} = 0$ and maximum intensity value $Int_{max} = 255$

for $Max_Part = 0$ to N

for $i=1:m$

for $j=1:n$

if ($ColorComponent(i, j) \geq Max_Part * \left(\frac{Int_{max} + 1}{N}\right)$ && $ColorComponent(i, j) \leq Max_Part * \left(\frac{Int_{max} + 1}{N}\right) + \left(\left(\frac{Int_{max} + 1}{N}\right) - 1\right)$)

$ColorClump_{ColorComponent}(i, j) = ColorComponent(i, j)$

4. The threshold value T_x for each color component R, G and B is calculated for each color clump N

$$T_{ColorClump_{ColorComponent}} = \text{mean}(\text{nonzeros}(ColorClump_{ColorComponent}(i, j)))$$

/* $ColorComponent$ = Red, Green and Blue */

5. Total number of generated threshold values for N color clumps are $N * T_x$
6. Calculate the upper and lower mean intensity for corresponding color clumps by computing the mean of intensity values of pixel higher than the threshold value and lower than threshold value respectively for the corresponding color clump
7. Repeat 4–6 for feature extraction from all the N partitions for $N \geq 2$ and compare classification performances with $(N-1)$ partitions for $N \geq 2$.
8. Stop when classification performance of current N partitions is inferior compared to $N-1$ partitions.

End

The maximum number of feature vectors extracted is equivalent to twice the number of partitions for each color component.

Algorithm 1 has been numerically illustrated with an example in figure 2(a) for generating two clumps.

5. Existing feature extraction techniques

Feature extraction for content based image classification has been done previously with different techniques. All the techniques were performed in RGB color space and the existing techniques have been given in Algorithm 2 through 9.

5.1 Multilevel BTC

Four levels of BTC based feature extraction technique have been carried out [6–8]. The information extracted in the form of features was based on pixel distribution in various color levels for different color spaces present in the given image. Algorithm 2 describes the process of feature extraction for the different BTC levels.

Algorithm 2

Begin

1. Input an image I with three different color components denoted as $cc1$, $cc2$ and $cc3$ respectively of size $m * n$ each.

/* where $\{(cc1), (cc2), (cc3)\} = \{R, G, B\}$ for RGB color space */

2. Calculate the threshold value T_x for each color component R, G and B

$$T_x = (1/m * n) * \sum_{i=1}^m \sum_{j=1}^n x(i, j)$$

/* $x=R, G$ and B */

3. Compute binary image maps for the given image

$$Bitmap_x(i, j) = \begin{cases} 1, \text{if } x(i, j) \geq T_x \\ 0, \text{if } x(i, j) < T_x \end{cases}$$

4. Generate six image features (two image features X_{upmean} and $X_{lowmean}$ for each color component) for binary BTC for the given image

$$X_{upmean} = \frac{1}{\sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i, j)} * \sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i, j) * X(i, j)$$

$$X_{lowmean} = \frac{1}{m * n - \sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i, j)} * \sum_{i=1}^m \sum_{j=1}^n (1 - Bitmap_x(i, j)) * X(i, j)$$

5. The six feature vectors thus obtained in level 1 BTC acts as six thresholds for level 2
6. Repeat 4 to compute X_{upmean} and $X_{lowmean}$ for each of six thresholds which will calculate twelve elements of feature vectors in level 2 BTC
7. Repeat 4 to compute twenty four and forty eight elements of feature vector respectively for level 3 BTC and level 4 BTC as calculated in 6.

End

5.2 Binary BTC with bit plane slicing

Binary BTC technique of threshold computation has been followed by this approach to obtain the threshold value for computing the upper mean and lower mean for each color

component. Bit plane slicing [6–8] considered a given image to be sliced in different planes initiating from the least significant bit (LSB) to the most significant bit (MSB). An eight bit binary vector has denoted the intensity value of each pixel. The value of each bit can either be 0 or 1. Each bit plane has been represented by a binary matrix. Figure 4 has shown the original image along with the image slices. Significant bit planes carried rich information contents while the insignificant bit planes contained noise. Algorithm 3 and figures 3, 4, 5 has been for binary BTC with bit plane slicing.

Algorithm 3
Begin

1. Input an image I with three different color components denoted as $cc1$, $cc2$ and $cc3$ respectively of size $m*n$ each.
/* where $\{(cc1), (cc2), (cc3)\} = \{(R), (G), (B)\}$ for RGB color space */
2. Calculate the threshold value T_x for each color component R,G and B

$$T_x = (1/m*n) * \sum_{i=1}^m \sum_{j=1}^n x(i,j)$$

/* $x = R, G$ and B */

3. Compute binary image maps for the given image

$$Bitmap_x(i,j) = \begin{cases} 1, \text{if } x(i,j) \geq T_x \\ 0, \text{if } x(i,j) < T_x \end{cases}$$

4. Generate image features for binary BTC for the given image

$$X_{upmean} = \frac{1}{\sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i,j)} * \sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i,j) * X(i,j)$$

$$X_{lomean} = \frac{1}{m*n - \sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i,j)} * \sum_{i=1}^m \sum_{j=1}^n (1 - Bitmap_x(i,j)) * X(i,j)$$

5. Generate threshold value for the given image with the higher bit planes starting from bitplane 5 to bitplane 8.

$$BP_{avg}(bp5=1, \text{or } bp6=1, \text{or } bp7=1, \text{or } bp8=1) = (1/m*n) * \sum_{i=1}^m \sum_{j=1}^n x(i,j)$$

6. Compute image maps for bit plane slicing

$$Bpbitmap_x(i,j) = \begin{cases} 1, \text{if } x(i,j) \geq T_x \\ 0, \text{if } x(i,j) < T_x \end{cases}$$

7. Compute the feature vectors $xbitplane_{hi}$ and $xbitplane_{lo}$ for bit planes

$$X_{bpupmean} = \frac{1}{\sum_{i=1}^m \sum_{j=1}^n BpBitMap_x(i,j)} * \sum_{i=1}^m \sum_{j=1}^n BpBitMap_x(i,j) * X(i,j)$$

$$X_{bplomean} = \frac{1}{m*n - \sum_{i=1}^m \sum_{j=1}^n BpBitMap_x(i,j)} * \sum_{i=1}^m \sum_{j=1}^n (1 - BpBitMap_x(i,j)) * X(i,j)$$

End

5.3 Ternary BTC

Individual color threshold intervals were computed for each color component in RGB color space. Feature vector for ternary intensity values was compared with the threshold intervals to calculate the feature vectors for ternary BTC [9]. Information necessary for classification in the image can be represented by the intensity values higher than threshold, in between the two different thresholds and lower than the threshold. Intensity segmentation has been done for better representation of the given image. Algorithm 4 describes the technique of Ternary BTC.

Algorithm 4
Begin

1. Input an image I with three different color components denoted as $cc1$, $cc2$ and $cc3$ respectively of size $m*n$ each.

/* where $\{(cc1), (cc2), (cc3)\} = \{(R), (G), (B)\}$ for RGB color space */

2. Calculate the threshold value T_x for each color component R,G and B */

$$T_x = (1/m*n) * \sum_{i=1}^m \sum_{j=1}^n x(i,j)$$

/* $x = R, G$ and B */

3. Generate one overall luminance threshold T from T_x

$$T = \frac{T_x}{3}$$

/* $x = R, G$ and B */

4. Calculate individual color threshold for each color component. T_{shxl} is the lower threshold and T_{shxh} is the higher threshold

$$T_{shxl} = T_x - \lfloor T_x - T \rfloor, \quad T_{shxh} = T_x + \lceil T_x - T \rceil$$

/* $x = R, G$ and B */

5. Compute ternary image maps

$$T_{mx}(i,j) = \begin{cases} 1 & \text{iff } x(i,j) > T_{shx} \\ 0 & \text{iff } T_{shxl} \leq x(i,j) \leq T_{shxh} \\ -1 & \text{iff } x(i,j) < T_{shxl} \end{cases}$$

/* $x = R, G$ and B */

6. Generate image features tUx , tMx and tLx for ternary BTC.

$$tUx = \frac{1}{\sum_{i=1}^m \sum_{j=1}^n T_{mx}(i,j) \text{ iff } T_{mx}(i,j) = 1} * \sum_{i=1}^m \sum_{j=1}^n x(i,j) \text{ iff } T_{mx}(i,j) = 1$$

$$tMx = \frac{1}{\sum_{i=1}^m \sum_{j=1}^n T_{mx}(i,j) \text{ iff } T_{mx}(i,j) = 0} * \sum_{i=1}^m \sum_{j=1}^n x(i,j) \text{ iff } T_{mx}(i,j) = 0$$

$$tLx = \frac{1}{\sum_{i=1}^m \sum_{j=1}^n T_{mx}(i,j) \text{ iff } T_{mx}(i,j) = -1} * \sum_{i=1}^m \sum_{j=1}^n x(i,j) \text{ iff } T_{mx}(i,j) = -1$$

/* $x = R, G$ and B */

End

5.4 Binary BTC using even and odd image

The given image has undergone horizontal flip across X and Y axis. The Even Image has been formed by associating the flipped image with the Original Image and the Odd Image has been created by subtracting the flipped image from the Original Image. Feature vector extracted from Original Image was associated with Even Image and Odd Image [14] and has been considered for classification purpose. The same images with flipped versions were used in order to explore the option of extracting more information from the same given data. The process has been shown in figure 6 and the algorithm for calculating the feature vector has been given in Algorithm 5.

Algorithm 5

Begin

1. Input an image I with three different color components denoted as cc1, cc2 and cc3 respectively of dimension $m*n$ each.

/* where {(cc1), (cc2), (cc3)} = {(R), (G), (B)} for RGB color space */

2. Create the horizontal flip of I and name it I' across X and Y axis of dimension $m*n$.
3. Create the even Image I_{even} and the odd image I_{odd} of dimension $m*n$

$$I_{even} = \frac{(I + I')}{2}$$

$$I_{odd} = \frac{(I - I')}{2}$$

4. Calculate the threshold value T_x for each color component R,G and B for the given even image I_{even} and odd image I_{odd}

$$T_x = (1/m*n) * \sum_{i=1}^m \sum_{j=1}^n x(i,j)$$

/* x=R, G and B */

5. Compute binary image maps even image I_{even} and odd image I_{odd}

$$Bitmap_x(i,j) = \begin{cases} 1, \text{if } x(i,j) \geq T_x \\ 0, \text{if } x(i,j) < T_x \end{cases}$$

6. Generate image features for binary BTC even image I_{even} and odd image I_{odd}

$$Xupmean = \frac{1}{\sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i,j)} * \sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i,j) * X(i,j)$$

$$Xlomean = \frac{1}{m*n - \sum_{i=1}^m \sum_{j=1}^n Bitmap_x(i,j)} * \sum_{i=1}^m \sum_{j=1}^n (1 - Bitmap_x(i,j)) * X(i,j)$$

7. Concatenate the feature vector of I and I_{even}
8. Concatenate the feature vector of I and I_{odd}
9. Concatenate the feature vector of I and I_{even} and I_{odd}

End

5.5 Feature extraction using Niblack threshold selection

Niblack's method has been used for binarization of images from different categories as shown in figure 7 [17, 18]. Threshold for each pixel was calculated by the method for each color component by sliding a rectangular window over the component. The threshold was computed based on the local mean $m(i,j)$ and standard deviation $\sigma(i,j)$ with a window size of (25×25) . The threshold was given as $T(i,j) = m(i,j) + k.\sigma(i,j)$. Here, k was a constant having value of 0.6 in the proposed method. The algorithm has been given in Algorithm 6.

Algorithm 6

Begin

1. Input an image I with three different color components R, G and B respectively of size $m*n$ each.
2. Calculate the local threshold value T_x for each pixel in each color component R,G and B using Niblack's Method.

$$T_x(i,j) = m_x(i,j) + k.\sigma_x(i,j)$$

where $k=0.6$

/*x = R, G and B */

3. Compute binary image maps for each pixel for the given image.

$$Bitmap_x(i,j) = \begin{cases} 1, \text{if } x(i,j) > T_x \\ 0, \text{if } x(i,j) \leq T_x \end{cases}$$

/*x = R, G and B */

4. Generate image features for the given image for each color component.

5.6 Feature extraction using Sauvola's threshold selection

Sauvola's local threshold selection for image binarization was an improvement over Niblack's methodology as in Algorithm 7 [19, 20]. Sauvola's method has computed the mean and the standard deviation of the grey levels. The brightness of an image was calculated by the mean as images with higher mean were visibly brighter than images with lower mean. Contrast of the image considered for threshold calculation by computing the standard deviation. Binarization with Sauvola's threshold has been shown in figure 8.

Algorithm 7**Begin**

1. Input an image I with three different color components R, G and B respectively of size $m*n$ each.
2. Calculate the local threshold value T_x for each pixel in each color component R, G and B using Niblack's Method.

$$T_x(i, j) = m_x(i, j) * [1 + k(\frac{\sigma_x(i, j)}{R} - 1)]$$

 $k=0.5$ and $R=128$

/*x = R, G and B */

3. Compute binary image maps for each pixel for the given image.

$$Bitmap_x(i, j) = \begin{cases} 1, & \text{if } x(i, j) > T_x \\ 0, & \text{if } x(i, j) < T_x \end{cases}$$

/*x = R, G and B */

4. Generate image features for the given image for each color component.

5.7 Feature extraction using Bernsen's threshold selection

Bernsen's threshold selection method has a local window size of $w = 31$. The calculation of threshold was carried out by considering the mean of the minimum and the maximum gray value within the local window specified for each color component as in Algorithm 8 and figure 9. The difference of the maximum and the minimum gray values was taken as the contrast [15, 21]. A contrast value lower than contrast threshold k has designated the pixel inside the window as 0 or 1 based on the class description of the

window. Thus the algorithm was dependent on the value of k and the size of window.

Algorithm 8**Begin**

1. Input an image I with three different color components R, G and B respectively of size $m*n$ each.
2. Calculate the local threshold value T_x for each pixel in each color component R, G and B using Bernse's Method.
3. Compute binary image maps for each pixel for the given image.

$$Bitmap_x(i, j) = \begin{cases} 1, & \text{if } x(i, j) > T_x \\ 0, & \text{if } x(i, j) < T_x \end{cases}$$

/*x = R, G and B */

4. Generate image features for the given image for each color component.

5.8 Feature extraction using Otsu's threshold selection

Otsu's method has calculated the optimum threshold to separate the two classes of pixels for each color component in such a way so that their combined intra-class variance is minimal [23, 24]. Comprehensive investigation has been carried out for the threshold that minimizes the intra-class variance represented by the weighted sum of variances of the two classes of pixels for each of the three color components. The algorithm has been given in Algorithm 9 and the binarization technique has been shown in figure 10.

Figure 3. Original image. **Figure 5.** Amalgamated image of Bit plane 5, 6, 7 and 8. **Figure 4.** Images of different Bit planes.

Figure 6. Formation of even and odd images.

Figure 7. Binarized image for feature extraction using Niblack's threshold.

Algorithm 9
Begin

1. Input an image I with three different color components R, G and B respectively of size $m \times n$ each.
2. Calculate the local threshold value T_x for each pixel in each color component R, G and B using Otsu's Method.
3. Compute binary image maps for each pixel for the given image.

$$Bitmap_x(i, j) = \begin{cases} 1, & \text{if } .x(i, j) > T_x \\ 0, & \text{if } .x(i, j) < T_x \end{cases}$$

*/*x = R, G and B */*

4. Generate image features for the given image for each color component.

6. Classifiers

Classification process with various BTC based feature extraction techniques have been compared using two different classifiers as follows:

6.1 *K-nearest neighbor (KNN) classifier (distance based classifier)*

It has been termed as instance based classifier and was operated according to similarity measure between two instances. Mean Square Error (MSE) method was followed to compare the feature vectors of the images in the database. The nearest neighbor in the instance space was

Figure 8. Binarized image for feature extraction using Sauvola's threshold.

Figure 9. Binarized image for feature extraction using Bernsen's threshold.

located for classification and then the unknown instance was designated with the same class of the identified nearest neighbor [36]. The expression has been given in the following equation.

$$MSE = \frac{1}{MN} \sum_{y=1}^M \sum_{x=1}^N [I(x, y) - I'(x, y)]. \quad (1)$$

I and I' being the two images used for comparison using the MSE method.

6.2 *RIDOR classifier (rule based classifier)*

A set of if-then rules was implemented by RIDOR classifier to perform classification. Each record in the database was covered by exactly one rule. Mutually exclusive rules and exhaustive rules were responsible for this technique. Classification starts with an empty rule which was followed by

Figure 10. Binarized image for feature extraction using Otsu's threshold.

growing one rule [37]. Removal of training records covered by this rule had taken place and the previous steps were repeated until the stopping criteria were met. The exceptions having lowest error rate had been generated for the default rule by Ripple-Down Rule (RIDOR) learner followed by generation of “best” exception for each exception. A tree like expansion of exception was carried out with its leaf having the only default rule without exception.

7. Experimental process

A subset of Corel stock photo database known as Wang dataset [22] has been considered for evaluation of the BTC based techniques. It has been considered as a widely used public dataset. Consideration of the images from the dataset was done without compromising in size and quality for the experimental purpose. Another dataset, namely, Caltech dataset, comprising of 8127 images divided into 100 categories was also considered for the experimental purpose. The datasets were divided into 10 categories. Each category comprised of 100 images. Figures 11 and 12 have shown samples of the original databases. Ten-fold cross validation scheme was implemented to evaluate the classification performances for different feature vector extraction techniques. The process has been called n fold cross validation [38]. The value of n being an integer and was considered as 10 in this work. The entire dataset was divided into 10 subsets. One subset was considered as the testing set and the rest nine subsets were considered to be training set. The method was repeated for 10 trials and the performance of the classifiers were evaluated by combining the 10 results thus obtained after evaluating the 10 folds.

Classification of query images as a precursor of content based image retrieval was also examined with the proposed technique of feature extraction. A given query for retrieval was classified to its nearest matching category with the help of KNN Classifier. Once the query was classified to a specific category, the retrieval process has searched for related images only within the category of interest. This has drastically reduced the searching space and has proficiently improved retrieval results. A paired t test was conducted to

Figure 11. Sample images of different categories from Wang database.

Figure 12. Sample images of different categories from Caltech database.

establish the statistical significance of the image identification results [39].

7.1 Performance measure

Assessment of performance for various feature extraction techniques discussed in the previous sections for content based image classification has been conducted using different evaluation parameters. Initial comparison was carried out to find out the optimum value of partition N considered for feature extraction with proposed technique by comparing the classification results for two parameters viz., misclassification rate (MR) and F1score. Once the value of partition N with minimum misclassification rate (MR) and maximum F1 score was established among the different values of

Table 1. Misclassification rate (MR) and the F1 score comparison for different values of color clumps subset (N) while classifying with KNN classifier.

RGB color space	Color clumps $N = 2$	Color clumps $N = 2$	Color clumps $N = 4$	Color clumps $N = 4$	Color clumps $N = 8$	Color clumps $N = 8$	Color clumps $N = 16$	Color clumps $N = 16$	Color clumps $N = 32$	Color clumps $N = 32$
Categories	MR	F1 score	MR	F1 score	MR	F1 score	MR	F1 score	MR	F1 score
Tribals	0.08	0.65	0.08	0.69	0.07	0.73	0.06	0.75	0.12	0.42
Sea beach	0.08	0.60	0.08	0.61	0.08	0.56	0.08	0.59	0.16	0.40
Gothic structure	0.13	0.42	0.14	0.46	0.14	0.44	0.13	0.48	0.17	0.13
Bus	0.10	0.54	0.07	0.68	0.08	0.66	0.05	0.78	0.12	0.45
Dinosaur	0.00	1.00	0.00	1.00	0.00	0.99	0.00	1.00	0.00	1.00
Elephant	0.06	0.72	0.05	0.76	0.05	0.79	0.07	0.70	0.09	0.58
Roses	0.02	0.91	0.01	0.94	0.02	0.91	0.02	0.93	0.03	0.85
Horses	0.02	0.91	0.02	0.92	0.01	0.96	0.02	0.90	0.04	0.81
Mountains	0.13	0.37	0.10	0.47	0.10	0.51	0.11	0.43	0.15	0.38
Average	0.07	0.681	0.06	0.725	0.060	0.727	0.059	0.728	0.099	0.56

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
72	1	9	7	0	8	0	1	2	Tribals
8	58	10	0	0	5	0	1	18	Sea beach
11	7	43	16	0	4	3	2	14	Gothic structure
13	0	22	51	0	1	1	0	12	Bus
0	0	0	0	100	0	0	0	0	Dinosaur
6	2	5	1	1	75	0	3	7	Elephant
4	4	0	1	0	2	88	1	0	Roses
2	2	1	0	0	2	0	92	1	Horses
4	18	17	13	0	10	1	2	35	Mountains

Figure 13. Confusion matrix for $N = 2$ while classifying with KNN classifier.

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
79	1	10	3	0	5	0	1	1	Tribals
9	58	13	1	0	3	0	0	16	Sea beach
18	8	52	8	0	7	1	1	5	Gothic structure
8	2	20	63	0	1	0	0	6	Bus
0	0	0	0	100	0	0	0	0	Dinosaur
6	1	9	0	1	77	0	4	2	Elephant
5	2	2	0	0	1	89	1	0	Roses
1	2	1	0	0	1	0	95	0	Horses
3	15	20	11	0	7	0	4	40	Mountains

Figure 14. Confusion matrix for $N = 4$ while classifying with KNN classifier.

N considered to create the color clumps, the proposed methodology was compared with the existing techniques in terms of precision, recall along with misclassification rate (MR) and F1 score for content based image classification [24]. The MR and F1 score along with few other parameters viz., true positive (TP) rate/recall, false positive (FP) rate,

true negative (TN) rate, false negative (FN) rate and precision have standard definitions as follows:

Misclassification rate (MR) This has been the error rate of the classifier that indicates the proportion of instances that have been wrongly classified and is given as in the following equation

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
82	0	10	3	0	3	0	1	1	Tribals
10	46	21	3	0	9	0	0	11	Sea beach
13	2	51	15	0	3	6	0	10	Gothic structure
3	3	19	66	0	1	1	0	7	Bus
1	0	0	0	99	0	0	0	0	Dinosaur
6	1	6	1	0	79	0	4	3	Elephant
4	1	3	1	0	1	90	0	0	Roses
0	1	0	0	0	1	0	98	0	Horses
7	11	21	11	0	4	0	1	45	Mountains

Figure 15. Confusion matrix for $N = 8$ while classifying with KNN classifier.

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
83	0	8	1	0	3	0	4	1	Tribals
6	54	12	3	0	9	0	1	15	Sea beach
16	6	52	9	0	7	2	0	8	Gothic structure
0	4	13	76	0	1	0	0	6	Bus
0	0	0	0	100	0	0	0	0	Dinosaur
8	2	9	0	1	72	0	5	3	Elephant
1	2	2	1	0	1	91	2	0	Roses
0	2	0	0	0	2	0	95	1	Horses
6	13	21	6	0	11	3	3	37	Mountains

Figure 16. Confusion matrix for $N = 16$ while classifying with KNN classifier.

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
38	5	13	16	1	12	3	4	8	Tribals
2	47	12	4	0	10	0	2	23	Sea beach
12	31	12	12	0	4	0	3	26	Gothic structure
8	15	15	45	0	2	2	0	13	Bus
0	0	0	0	100	0	0	0	0	Dinosaur
9	13	8	2	0	57	0	5	6	Elephant
6	0	2	11	0	0	78	2	1	Roses
6	1	1	0	0	6	1	81	4	Horses
0	24	15	10	0	7	0	2	42	Mountains

Figure 17. Confusion matrix for $N = 32$ while classifying with KNN classifier.

$$MR = \frac{FP + FN}{TP + TN + FP + FN}. \quad (2)$$

F1-score precision and recall (TP rate) can be combined to produce a metric known as F1 score which is given as in the following equation

$$F1Score = \frac{2 * Precision * Recall}{Precision + Recall}. \quad (3)$$

True positive (TP) rate/recall This metric signifies the probability of a classifier to produce true positive result and is being given as in the following equation

$$TP Rate/Recall = \frac{TP}{TP + FN}. \quad (4)$$

False positive (FP) rate It has been defined as the probability that a classifier produces erroneous results as positive results for negative instances and is given as in the following equation.

$$FP Rate = \frac{FP}{FP + TN}. \quad (5)$$

True negative (TN) rate This metric has been defined as the probability of a classifier produces results as negative

Table 2. Misclassification rate (MR) and the F1 score comparison for different values of color clumps subset (N) while classifying with RIDOR classifier.

RGB color space	Color clumps $N = 2$	Color clumps $N = 2$	Color clumps $N = 4$	Color clumps $N = 4$	Color clumps $N = 8$	Color clumps $N = 8$	Color clumps $N = 16$	Color clumps $N = 16$	Color clumps $N = 32$	Color clumps $N = 32$
Categories	F1 score		F1 score		F1 score		F1 score		F1 score	
	MR		MR		MR		MR		MR	
Tribals	0.09	0.59	0.10	0.57	0.10	0.59	0.08	0.66	0.15	0.29
Sea beach	0.10	0.50	0.10	0.49	0.10	0.53	0.12	0.45	0.15	0.26
Gothic structure	0.16	0.27	0.14	0.28	0.13	0.38	0.13	0.41	0.15	0.15
Bus	0.11	0.53	0.10	0.60	0.08	0.68	0.05	0.76	0.13	0.38
Dinosaur	0.00	1.00	0.00	0.99	0.00	0.98	0.00	0.99	0.00	0.99
Elephant	0.09	0.58	0.09	0.60	0.09	0.59	0.09	0.57	0.12	0.48
Roses	0.03	0.84	0.03	0.89	0.03	0.86	0.03	0.85	0.05	0.80
Horses	0.05	0.79	0.05	0.78	0.05	0.76	0.05	0.78	0.14	0.54
Mountains	0.13	0.43	0.14	0.38	0.11	0.51	0.13	0.43	0.21	0.03
Average	0.09	0.615	0.08	0.618	0.08	0.652	0.08	0.655	0.12	0.44

Figure 18. Category wise comparison of classification in terms of misclassification rate (MR) for KNN classifier. **Figure 19.** Comparison of average values of misclassification rate (MR) for KNN classifier.

results for negative instances and is given as in the following equation.

$$TN = \frac{TN}{FP + FN} \quad (6)$$

Figure 20. Category wise comparison of classification in terms of F1 score for KNN classifier. **Figure 21.** Comparison of average values of F1 score for KNN classifier.

False negative (FN) rate This has been the probability of a classifier produces erroneous results as negative results for positive instances as in the following equation

$$FNRate = \frac{FN}{TP + FN} \quad (7)$$

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
61	1	11	15	0	5	0	4	3	Tribals
4	47	13	2	0	9	3	6	16	Sea beach
9	12	27	20	0	8	5	3	16	Gothic structure
8	3	18	56	0	2	1	0	12	Bus
0	0	0	0	100	0	0	0	0	Dinosaur
13	9	7	1	0	57	0	4	9	Elephant
1	0	3	6	0	0	81	8	1	Roses
4	2	3	1	0	5	2	83	0	Horses
6	13	16	10	0	10	1	1	43	Mountains

Figure 22. Confusion matrix for $N = 2$ while classifying with RIDOR classifier.

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
60	2	4	11	0	9	2	5	7	Tribals
8	43	7	4	0	14	2	5	17	Sea beach
13	8	24	23	0	7	2	5	18	Gothic structure
7	0	13	66	0	1	5	2	6	Bus
0	0	0	0	99	1	0	0	0	Dinosaur
6	7	7	2	0	63	0	8	7	Elephant
2	0	4	2	0	0	89	3	0	Roses
8	4	0	0	0	2	0	82	4	Horses
8	13	15	12	0	14	1	0	37	Mountains

Figure 23. Confusion matrix for $N = 4$ while classifying with RIDOR classifier.

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
64	3	9	6	1	6	0	5	6	Tribals
7	49	15	5	0	7	1	8	8	Sea beach
9	12	36	10	0	11	4	3	15	Gothic structure
4	0	9	73	0	3	1	1	9	Bus
1	0	1	0	97	1	0	0	0	Dinosaur
15	3	5	3	0	59	1	4	10	Elephant
4	1	1	6	0	1	81	6	0	Roses
5	6	1	2	0	7	0	79	0	Horses
7	11	14	10	0	6	1	1	50	Mountains

Figure 24. Confusion matrix for $N = 8$ while classifying with RIDOR classifier.

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
72	5	6	5	0	4	2	2	4	Tribals
8	43	16	2	0	11	4	3	13	Sea beach
14	13	42	5	0	6	3	1	16	Gothic structure
2	4	8	77	0	1	1	0	7	Bus
0	0	0	0	99	1	0	0	0	Dinosaur
11	7	9	0	1	55	0	5	12	Elephant
2	2	0	0	1	1	83	11	0	Roses
2	5	5	2	0	4	1	78	3	Horses
6	11	18	11	0	10	1	1	42	Mountains

Figure 25. Confusion matrix for $N = 16$ while classifying with RIDOR classifier.

cat1	cat2	cat3	cat4	cat5	cat6	cat7	cat8	cat9	Classified as
27	12	10	14	0	18	4	7	8	Tribals
7	24	13	8	0	12	0	6	30	Sea beach
6	20	12	16	0	10	4	8	24	Gothic structure
15	15	6	35	0	5	8	0	16	Bus
0	0	0	0	99	1	0	0	0	Dinosaur
17	8	10	2	0	50	0	4	9	Elephant
3	1	4	7	0	0	81	3	1	Roses
7	2	4	1	0	6	3	74	3	Horses
7	2	4	1	0	6	3	74	3	Mountains

Figure 26. Confusion matrix for $N = 32$ while classifying with RIDOR classifier.

Figure 27. Category wise comparison of classification in terms of misclassification rate (MR) for RIDOR classifier.

Figure 28. Comparison of average values of misclassification rate (MR) for RIDOR classifier.

Precision This has been the probability that an object is classified correctly as per the actual value and is given as in the following equation

$$\text{Precision} = \frac{TP}{TP + FP}. \quad (8)$$

8. Results and discussion

The experimentation has been conducted with Intel core i5 processor having 4 GB RAM and Matlab 7.11.0(R2010b). Color Clumps method of feature extraction has divided the intensity values of the given images in Wang dataset in N subsets where $N = 2, 4, 8, 16$ and 32 . Classification with

Figure 29. Category wise comparison of classification in terms of F1 score for RIDOR classifier.

Figure 30. Comparison of average values of F1 score for RIDOR classifier.

Figure 31. Comparison of average precision and average recall values for classification with KNN classifier with Caltech dataset.

Figure 32. Comparison of average precision and average recall values for classification with RIDOR classifier with Caltech dataset.

different values of N in RGB color space has been shown in the results. Henceforth, classification of query was performed for content based image retrieval. The result of retrieval was compared with state-of-the art techniques. All the evaluations with the proposed technique were statistically compared to the existing techniques by means of paired t test [39]. The comparison has revealed significant improvement in results with proposed technique of feature extraction.

8.1 Classification with KNN classifier for color clumps method of feature extraction

Primarily, the misclassification rate (MR) and the F1 score were compared to find out the efficiency of classification with each given values of N as shown in table 1. The misclassification rate (MR) decreased till $N = 16$ and the F1 score increased for the same value of N . The minimum misclassification rate (MR) was observed for $N = 16$ which was 0.059 and the corresponding highest F1 score was 0.728. This indicated better and increased performance of classification up to $N = 16$ subsets of the intensity values as in table 1. But the F1 score declined to 0.56 for $N = 32$ and the misclassification rate increased to 0.099 for classification results in table 1. This indicated that the optimum subset for feature extraction that can be created for intensity values in multiples of 2 was restricted till 16 to boost the classification performance with KNN classifier of images in Wang dataset for color clumps method of feature extraction. Category wise comparison of classification in terms of misclassification rate (MR) and F1 score for each value of N used to extract features with color clumps method has been shown in figures 18 and 20 respectively. Comparison of average values of misclassification rate (MR) and F1 score for each partition N used for feature extraction with color clumps has been shown in figures 19 and 21 respectively. The confusion matrices generated during classification after feature extraction with each value of N has been given in Figures 13, 14, 15, 16, 17.

8.2 Classification with RIDOR classifier for color clumps method of feature extraction

Feature extraction with color clumps method was done with same values of N ranging from 2 to 32 with an interval of multiples of 2 for classification with RIDOR classifier. The value of F1 score in table 2 has increased to 0.655 till $N = 16$ and decreases to 0.44 with $N = 32$. So misclassification rate (MR) was reduced to 0.08 for $N = 16$ starting from $N = 2$ for classification with RIDOR classifier and increased with $N = 32$ as in table 2. Thus it can be concluded that $N = 16$ has shown analogous behavior for classification with RIDOR classifier and KNN classifier and can be considered as the optimal number of subsets for feature extraction even in case of RIDOR classifier. Category wise comparison of classification in terms of misclassification rate (MR) and F1 score for each value of N used to extract features with color clumps method has been shown in figures 27 and 29. Comparison of average values of misclassification rate (MR) and F1 score for each partition N used for feature extraction with color clumps has been shown in figures 28 and 30 respectively. The confusion matrices generated during classification after feature extraction with each value of N has been given in figures 22, 23, 24, 25, 26.

Henceforth, the proposed technique, namely, BTC with color clumps has been tested with Caltech dataset comprising of 8,127 images. The precision and recall values of classification with KNN and RIDOR classifiers have been given in figures 31 and 32 for different color clump values.

BTC with color clumps method of feature extraction for $N = 16$ has shown best performance for content based image classification for both the datasets as seen in tables 1 and 2 as well as in figures 31 and 32. The results in tables 1 and 2 have shown the highest F1 score and the minimum misclassification rate (MR) for $N = 16$ in both the classifier environments of KNN and RIDOR. The second dataset, namely, Caltech dataset comprised of almost eight times image data compared to the first dataset, namely, Wang dataset. Nevertheless, the trend for performance rise and performance decline based on number of partitions of color clumps remains same in both the cases. The number of segmentations has increased with the number of clumps. The segmented regions carry the information of both foreground and background from which the color clumps based feature extraction has been carried out. However, the foreground and background information of the image remains optimal till $N = 16$ clumps. But, as the number of clumps increases beyond that, the segmented regions were unable to represent the foreground and the background of the images favorably. Hence, the robustness of the feature vector decreases which has resulted in decreased classification results. Therefore, $N = 16$ was considered to be the optimal number of clumps for feature vector extraction with the proposed technique. It was also observed that KNN classifier has portrayed higher F1 score of 0.727 compared

Figure 33. Comparison of average precision, average recall, average misclassification rate and average F1 score of existing techniques w.r.t proposed technique of feature extraction for classification done with KNN Classifier in RGB color space.

Table 3. *t* test for comparison of precision results for classification.

Comparison	t-calc	p value	Significance
Feature extraction with multilevel BTC [6]	2.6265	0.0303	Significant
Feature extraction with ternary BTC [14]	3.518	0.0079	Significant
Feature extraction with binary BTC with Bit Plane slicing [6–8]	2.799	0.0232	Significant
Feature extraction with binary BTC using original + even image [14]	2.9328	0.0189	Significant
Feature extraction with binary BTC using original + odd image [14]	3.4357	0.0089	Significant
Feature extraction with binary BTC using original + even image + odd image [14]	2.4385	0.0407	Significant
Feature extraction with Niblack's threshold selection [17]	3.8687	0.0048	Significant
Feature extraction with Sauvola's threshold selection [19]	2.3519	0.0465	Significant
Feature extraction with Bernsen's threshold selection [21]	2.6583	0.0289	Significant
Feature extraction with Otsu's threshold selection [24]	3.6749	0.0063	Significant

to the F1 score of 0.655 of RIDOR classifier for Wang dataset. Hence, the precision and recall along with F1 score and misclassification rate (MR) for classification with KNN

Classifier with the proposed technique of feature extraction was further compared to that of state-of-the-art techniques as in figure 33.

Table 4. Comparison of precision results for retrieval.

Categories	Proposed	Walia and Pal [33]	Jalab [29]	Hiremath and Pujari [27]	Rahimi and Moghaddam [32]	Li and Wang [22]
Tribals	60	54	32.3	48	41.8	48
Sea beach	100	51	61.2	34	44.6	32
Gothic structure	40	38	39.2	36	49.2	35
Bus	60	46	39.5	61	30.5	36
Dinosaur	100	100	99.6	95	55.4	95
Elephant	80	63	55.7	48	69.9	38
Roses	80	90	89.3	61	61.1	42
Horses	100	93	65.2	74	57.2	72
Mountains	60	48	56.8	42	43.7	35
Food	60	39	44.1	50	43.5	38
Average precision	74	62.2	58.29	54.9	49.69	47.1

Table 5. *t* test for comparison of precision results for retrieval.

Comparison	t-calc	<i>p</i> value	Significance
Walia and Pal [33]	2.3508	0.0432	Significant
Jalab [29]	3.057	0.0136	Significant
Hiremath and Pujari [27]	3.1172	0.0124	Significant
Rahimi and Moghaddam [32]	4.0369	0.0029	Significant
Li and Wang [22]	4.4703	0.0016	Significant

8.3 Performance comparison of classification with different techniques of feature extraction

Proposed feature extraction technique has been compared to existing state-of-the-art techniques used to extract features from the images.

The parameters for comparison were precision, recall, misclassification rate (MR) and F1 score. The comparisons have been shown in figure 32.

The comparison in figure 33 has shown that classification with proposed technique of feature extraction has outperformed all the existing techniques.

A paired *t* test was conducted in table 3 to determine that whether the difference in precision values for classification were generated from a population with zero mean.

$$H_0 : \mu_d = 0 \text{ vs. } H_1 : \mu_d < 0.$$

The value of *t*-calc was computed by taking the difference between two sample means considered for each comparison in table 3. The strength of evidence against the null hypothesis was measured by the *p* value.

The *p* values calculated have indicated significant difference in precision results for classification with the proposed technique of feature extraction and the null hypothesis was rejected.

Further, the classification technique with proposed method of feature extraction was used as a precursor of retrieval. The query for retrieval was initially classified with the help of KNN classifier and was then forwarded to search for nearest matches only within the class of interest.

A total of 50 images were randomly chosen from 10 different categories of which five images were selected from each category. The classification of the query images was carried out by implementing Canberra distance measure as in Eq. (9) and each of the query images was fired one at a time. The ranking of the retrieved images was carried out by using Canberra distance and the top 20 retrieved images were selected. The precision value for retrieval was compared to state-of-the-art techniques in table 4 in which the proposed method has surpassed all the existing techniques.

$$\text{Canberra distance} = \sum_{i=1}^n \frac{|Q_i - D_i|}{|Q_i| + |D_i|} \quad (9)$$

where Q_i is the query image and D_i is the database image.

A paired *t* test was conducted with respect to the retrieval technique with proposed method of feature extraction to verify whether the differences in precision values corresponding to the existing retrieval techniques were generated from a population with zero mean:

$$H_0 : \mu_d = 0 \text{ vs. } H_1 : \mu_d < 0.$$

The *p* values in table 5 have shown significance after the *t* test was conducted. Hence the null hypothesis was rejected and the statistical significance of improved retrieval after query classification with the proposed technique of feature extraction was established.

Finally, the average time for feature extraction from each image in the Wang dataset with the proposed technique was compared to the existing techniques of feature extraction as shown in figure 34.

The comparison in figure 34 has shown that feature extraction with ternary BTC has consumed the least average time followed by the proposed technique. Hence, the proposed method has taken the minimum average time for feature extraction compared to rest of the existing techniques except feature extraction with ternary BTC. However, classification results with the proposed technique of feature extraction have outclassed all the existing techniques.

Figure 34. Comparison of average computation time for feature extraction from each image.

Figure 35. Classifier comparison in ROC space.

Thus, the novel technique was effectively introduced in Algorithm 1 with a small feature vector size of 96 on the whole for all three color components. The feature vector size was not dependent on the dimension of the image and the average time for feature extraction was also not as much of the state-of-the-art techniques. Evaluation of the improved classification results with the proposed technique of feature extraction has been verified with t test for statistical significance. Thus BTC with color clumps was established as an efficient and better performing feature extraction technique over state-of-the-art methods of feature extraction.

9. Comparison of classifiers in ROC space

Two different discrete classifiers viz., KNN and RIDOR classifier were used to compare the different parameters of classifications for the proposed technique. Classifier performances can be well visualized and compared with the help of ROC graph [38]. A single ROC point in the ROC

space is resulted from the only confusion matrix generated by these types of classifiers to produce only a single class decision on each instance. The x -axis of this 2D graph is the FP Rate whereas the y axis is the TP Rate. A point in the graph denotes the performance of the classifier. The point given as (0,1) in the graph signifies the ideal classifier. The classifier point located nearer to (0,1) has been considered to be a better classifier w.r.t the other one. A ROC point sliding across the diagonal line $y = x$ would indicate random classifier. Figure 35 shows the comparison of KNN and RIDOR classifier in ROC space. It has been observed that KNN classifier is nearer to (0,1) with co-ordinate point (0.033,0.73) compared to RIDOR classifier (0.043,0.66). Thus KNN classifier was considered to have better performance compared to RIDOR classifier. The weighted average of TP Rate and the FP Rate for classification with feature extraction by the proposed method of all the categories of images were considered for each classifier for this comparison. Lower TP rate termed RIDOR classifier as “conservative” as it has made positive classifications only with strong evidences compared to KNN Classifier. On the other hand, KNN Classifier has been observed to be “liberal” for higher TP Rate as it has made positive classifications with weak evidences compared to RIDOR Classifier.

10. Conclusion

The paper presents a novel technique for BTC based feature extraction to perform content based image classification. The proposed approach was based on logical partitioning of the image data in multiples of 2 and has resulted in better classification performances compared to existing techniques. The significance of the proposed methodology was due to the computation of non-overlapping threshold on

- [27] Hiremath P S and Pujari J 2007 Content based image retrieval using color, texture and shape features. In: *International Conference on Advanced Computing and Communications*, pp. 780–784
- [28] Banerjee M, Kundu M K and Maji P 2009 Content-based image retrieval using visually significant point features. *Fuzzy Sets Syst.* 160: 3323–3341
- [29] Jalab H A 2011 Image retrieval system based on color layout descriptor and Gabor filters. In: *IEEE Conference on Open Systems (ICOS)*, pp. 32–36
- [30] Shen G L and Wu X J 2013 Content based image retrieval by combining color texture and CENTRIST. In: *IEEE International Workshop on Signal Processing*, vol. 1, pp. 1–4
- [31] Irtaza A, Jaffar M A, Aleisa E and Choi T S 2013 Embedding neural networks for semantic association in content based image retrieval. *Multimedia Tool Appl.* 1–21
- [32] Rahimi M and Moghaddam M E 2013 A content based image retrieval system based on color ton distributed descriptors. *Signal Image Video Process.* 1–14
- [33] Walia E and Pal A 2014 Fusion framework for effective color image retrieval. *J. Vis. Commun. Image R.* 25: 1335–1348
- [34] Guo J-M and Lin C-Y 2010 Parallel and element-reduced error-diffused block truncation coding. *IEEE Trans. Commun.* 58: 1905–1908
- [35] Guo J-M and Wu M-F 2009 Improved block truncation coding based on the void-and-cluster dithering approach. *IEEE Trans. Image Process.* 18: 211–213
- [36] Han J, Kamber M and Pei J 2011 Classification: Advanced methods. In: *Data mining concepts and techniques*. 3rd ed. Morgan Kaufmann Publishers, Elsevier, pp. 423–425.
- [37] Kotsiantis S B 2007 Supervised machine learning: A review of classification techniques. *Informatica* 31: 249–268
- [38] Sridhar S 2011 *Image features representation and description digital image processing*. Oxford University Press, New Delhi, India, pp. 483–486
- [39] Yildiz O T, Aslan and Alpaydin E 2011 Multivariate statistical tests for comparing classification algorithms. *Lecture Notes in Computer Science*, vol. 6683. Springer Berlin Heidelberg, pp. 1–15