TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

"ADULT IMAGE DETECTION USING SVM"

Bibek Raj Dhakal	Biru Charan Sainju	Suvash Sedhain
062BCT506	062BCT507	062BCT548
bibekraj9@gmail.com	thugliffe@gmail.com	mesuvash@gmail.com

A PROJECT WAS SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULLFILLMENT OF THE REQUIREMENT FOR THE BACHELORS DEGREE IN ELECTRONICS COMMUNICATION / COMPUTER ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING LALITPUR, NEPAL

MARCH,2010

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certify that they have read, and recommended to the Institute of Engineering for acceptance, a project report entitled ”Adult Image Detection Using SVM” submitted by Mr. Bibek Raj Dhakal,Mr. Biru Charan Sainju and Mr. Suvash Sedhain in partial fulfilment of the requirements for the Bachelors degree in Electronics Communication / Computer Engineering.

Supervisor, Dr Shashidhar Ram Joshi

Prof. Dr.

Department Of Electronics and Computer Engineering

Supervisor, Deepen Chapagain

Er.

Department Of Electronics and Computer Engineering

Internal Examiner,

Internal Examiner,

DATE OF APPROVAL:

Copyright

Acknowldgement

First of all, we would like to express our immense gratitude to our supervisors Prof. Dr. Shashidhar Ram Joshi and Mr. Deepen Chapagain for their constant support and motivation during the project. We would also like to thank the Department of Electronics and Computer engineering for providing us the opportunity to explore our interest and ideas in the field of engineering through this project.

We are grateful to Mr. Michael Jones for providing us with the compaq dataset of his images used in MERL Research for our project purpose. We would also like to offer our humble gratitude to all the teachers whose inspiring lectures paved the way and our friends for providing valuable suggestions regarding our project.

Abstract

An Adult image detection system that predicts whether an image is benign or adult based on the model built using SVM as the learning tool and various features of the image to train the system model.Skin color technique, Mpeg-7 Descriptor, Moments , BIC ,etc are among the methodologies used for the feature extraction.

Keywords:

Adult Image, SVM

Contents

List of Tables

List of Figures

List Of Symbols

SCD Scalable Color Descriptor

CLD Color Layout Descriptor

BIC Border/Interior Classifier

SVM Support Vector Machine

1. INTRODUCTION

1.1. Motivation

Our human civilization has been influenced and intoxicated by the web revolution. People of every age use the web for different needs and purposes. Some use it for fun; some use it for their studies and find information while some live on it. Images are essentially part of the modern web and we all agree to the fact that a single picture is worth thousand words. We can find all sort of information on the web and it has been a part of our daily life. However, it also has abundance of images and contents that may be unsuitable for certain age groups. Finding pornographic images posted on social sites and links on study groups is not a new thing in todays world. Pornographic images certainly need to be managed and unavailable to children and men at work.

1.2. Problem Description

Most of the web filters currently in use are based on textual cues or contextual ones and can be easily deceived. There is no proper filtering on the web and any sort of images are found without any proper surveillance. Photo sharing and social networking sites like flickr, facebook etc allows user to upload their pictures and share them. Some people also use these services to post pornographic images which should definitely not be allowed. Rated contents should not be easily accessible without age verification and proper space should be allotted so that only those wanting to find them can find them.

1.3. Objectives of work

Some of the major objectives of our project can be listed as

- To develop a system can learn the patterns of the feature extracted from the images
- To use SVM to predict whether a given image Is pornographic or not based on the leanings from the previous data of the images
- To distinguish adult content from the non adult ones and server as a tool for filtering images

2. BACKGROUND

Current adult image filtering techniques can be classified into three categories: keyword based, blacklist based and content based. Our proposed system is content based i.e. images will be classified on the basis of their content. The classification approach we applied can be broadly classified into two categories- Skin based and non-skin based.

2.1. SVM(Support Vector Machine)

Support Vector Machines are nothing more (or less) than linear learning machines expressed in dual form that map their input vectors to a feature space by the use of kernels and compute the optimal hyperplane there.

The central idea of SVM is the adjustment of a discriminating function so that it optimally uses the separability information of the boundary cases. Given a set of cases which belong to one of two classes, training a linear SVM consists in searching for the hyperplane that leaves the largest possible number of cases of the same class on the same side, while maximizing the distance of either class from the hyperplane. If the training set is linearly separable, then a discriminant hyperplane will satisfy the inequalities:

$$v_i(w.x_i + b) \geq 1 \tag{2.1}$$

where $x_i \in \Re^d$ is a vector of the training set, d being the dimension of the input space, and $y_i \in \{-1, +1\}$ is the corresponding class. Among the separating hyperplanes, the SVM

approach selects the one for which the distance to the closest point is maximal. Since such a distance is $1/||w||$, finding the hyperplane is equivalent to minimizing $||w||^2$ under constraints 2.1. The points closest to the hyperplane are called Support Vectors, and the quantity $2/||w||$ is called the margin (see Figure 2.1); it can be considered a measure of the generalization ability of the SVM: the larger the margin, the better the generalization is expected to be.

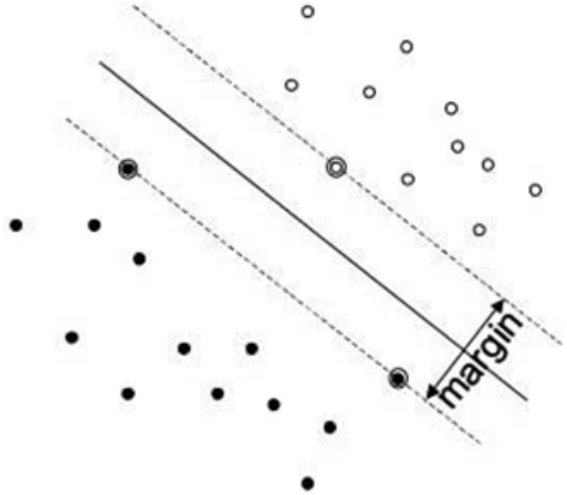


Figure 2.1: Separating Hyperplanes. The Suppor Vectors Are Circled.

When the training set is not linearly separable, the problem is relaxed by introducing a set of slack variables e_i and a penalization for the points that are mis-classified. This leads to the optimization problem,

$$min(||w||^2 + C\sum_{i=1}^{n} e_i), \tag{2.2}$$

under the constraints

$$v_i(w.x_i + b) \geq 1, e_i \geq 0 \tag{2.3}$$

The parameter C determines a trade-off between the error on the training set and the separation of the two classes. The SVM approach can be extended to non-linear decision

surfaces through a non-linear function ϕ which maps the original feature space $\Re$d into a higher dimensional space H. Since the only operation needed on H is the inner product, if we have a kernel function

$$\Re \rightarrow \Re^d \times \Re^d \rightarrow \Re k(x',x'') = \phi(x')\phi(x'') \tag{2.4}$$

ϕ mapping is never explicitly used. The kernel will exists, provided some non stringent conditions are fulfilled. Examples of widely used kernel functions are the polynomial kernel:

$$K(x',x'') = (x'.x'' + 1)^p \tag{2.5}$$

and the gaussian kernel

$$K(x',x'') = \exp(\frac{-||x' - x"||^2}{\sigma}) \tag{2.6}$$

2.1.1. Support Vector Classification

The classication problem can be restricted to consideration of the two-class problem without loss of generality. In this problem the goal is to separate the two classes by a function which is induced from available examples. The goal is to produce a classier that will work well on unseen examples, i.e. it generalizes well. Consider the example in Figure 2.2. Here there are many possible linear classiers that can separate the data, but there is only one that maximizes the margin (maximizes the distance between it and the nearest data point of each class). This linear classifier is termed the optimal separating hyperplane. Intuitively, we would expect this boundary to generalize well as opposed to the other possible boundaries.

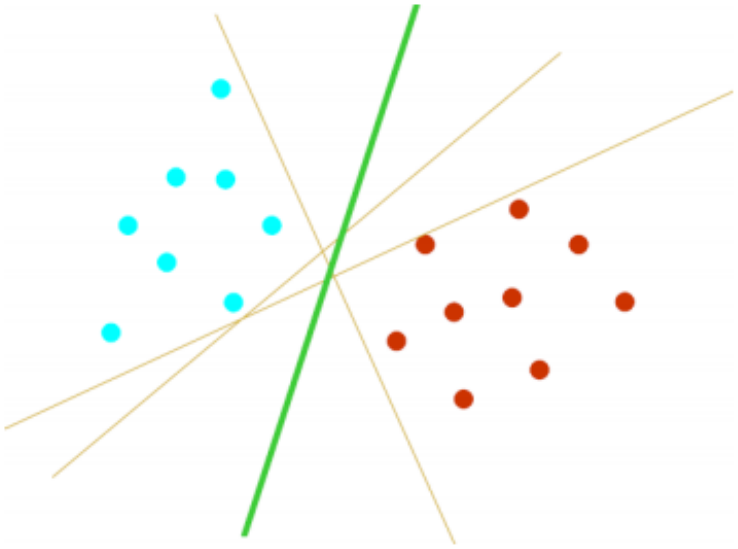


Figure 2.2: Optimal Separating Hyperplane

The Optimal Separating Hyperplane

Consider the problem of separating the set of training vectors belonging to two separate classes,

$$D = \{(x^1, y^1), \ldots (x^l, y^l)\}, x \in \Re^n, y \in \{-1, +1\} \tag{2.7}$$

with a hyperplane,

$$< w, x > + b = 0 \tag{2.8}$$

The set of vectors is said to be optimally separated by the hyperplane if it is separated without error and the distance between the closest vector to the hyperplane is maximal.

2.1.2. Feature Spaces and Kernels

Linear learning machines (such as the hyperplane classier), while being mathematically compelling because of their ease of analysis, have limited computational power and thus limited real-world value . In general, complex real-world applications require a learning machine with much more expressive power. One proposed solution to this problem was to create a network of simple linear classiers (in the form of neural networks) and thus be able to represent nonlinear decision surfaces. However, neural networks have a number of inherent problems, including local minima and many tunable parameters. In addition, it is very complex to analyze a neural network mathematically. Another solution is to map the input vectors into a richer (usually high-dimensional) feature space where they are linearly separable using a nonlinear mapping . In feature space, build a separating hyperplane using a well-understood linear learning machine such as the optimal hyperplane classier (see Figure 2.3). This yields a nonlinear decision surface in input space and is the approach taken by Support Vector Machines.

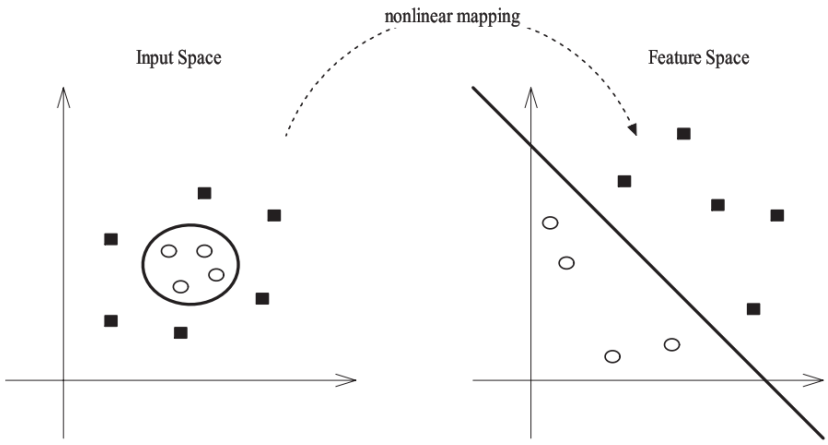


Figure 2.3: A nonlinear mapping from input space to feature space can simplify the classication task.

The optimal hyperplane classier uses only dot products between vectors in input space. In feature space this will translate to $\langle \phi(x).\phi(y) \rangle$. Clearly, this is very computationally expensive, especially if the mapping is to a high-dimensional space. A kernel is a function k(x, y) that given two vectors in input space, returns the dot product of their images in feature

space

$$k(x,y) = \langle \phi(x).\phi(y) \rangle \tag{2.9}$$

Kernel Function

The idea of the kernel function is to enable operations to be performed in the input space rather than the potentially high dimensional feature space. Hence the inner product does not need to be evaluated in the feature space. This provides a way of addressing the curse of dimensionality. However, the computation is still critically dependent upon the number of training patterns and to provide a good data distribution for a high dimensional problem will generally require a large training set. The following theory is based upon Reproducing Kernel Hilbert Spaces (RKHS) [7]. An inner product in feature space has an equivalent kernel in input space,

$$k(x',x'') = \phi(x')\phi(x'') \tag{2.10}$$

provided certain conditions hold. If K is a symmetric positive denite function, which satises Mercers Conditions,

$$k(x',x'') = \sum_{m}^{\infty} (a_m \phi_m(x')\phi_m(x'')), a_m \geq 0 \tag{2.11}$$

$$\iint k(x',x'')g(x)g(x')dxdx' > 0, g \in L_2 \tag{2.12}$$

then the kernel represents a legitimate inner product in feature space. Valid functions that satisfy Mercers conditions are now given, which unless stated are valid for all real x and x'

Gaussian Radial Basis Function Radial basis functions have received signicant attention, most commonly with a Gaussian of the form,

$$K(x,x') = \exp \frac{-||x-x'||^2}{2\sigma^2} \tag{2.13}$$

Classical techniques utilizing radial basis functions employ some method of determining a subset of centres. Typically a method of clustering is rst employed to select a subset of centres. An attractive feature of the SVM is that this selection is implicit, with each support vectors contributing one local Gaussian function, centred at that data point.

2.2. Skin Based Approach

Our human civilization has been influenced and intoxicated by the web revolution. People of every age use the web for different needs and purposes. Some use it for fun; some use it for their studies and find information while some live on it. Images are essentially part of the modern web and we all agree to the fact that a single picture is worth thousand words. We can find all sort of information on the web and it has been a part of our daily life. However, it also has abundance of images and contents that may be unsuitable for certain age groups. Finding pornographic images posted on social sites and links on study groups is not a new thing in today's world. Pornographic images certainly need to be managed and unavailable to children and men at work.

2.2.1. Skin Color Detection Method

The most important feature that provides clues to image content is color. Color is a low level feature, which makes it computationally inexpensive and therefore suitable for real-time object characterization, detection and localization. Generally pornographic images show a lot of skin and thus skin color is a basic feature used in detecting pornographic images. The main goal of skin color detection or classification is to build a decision rule that will discriminate between skin and non-skin pixels. Identifying skin colored pixels involves finding the range of values for which most skin pixels would fall in a given color space

Color Spaces

The purpose of a color space is to facilitate the specification of colors in some standard, generally accepted manner. A color space is a specification of a coordinate system and subspace within a system where each color is represented by a single point. Various color spaces are used for processing digital images. For some purposes, one color space may be more appropriate than others.

The RGB Color Space

The RGB color space originated from CRT display applications. In the RGB space each color appears in its primary spectral component of red, green, and blue. Images represented in the RGB space consist of three component images, one for each primary color. When fed into an RGB monitor, these images combine on the phosphor screen to produce a composite color image .The RGB color space is one of the most widely used color spaces for storing and processing digital image. However, the RGB color space alone is not reliable for identifying skin-colored pixels since it represents not only color but also luminance. Skin luminance may vary within and across persons due to ambient lighting so it is not dependable for segmenting skin and non-skin regions. Chromatic colors are more reliable and these are obtained by eliminating luminance through some form of transformation. The color spaces Normalized RGB, HSV, and YCbCr are transformations commonly used by studies on skin color .

The HSV Color Space

The HSV (Hue, Saturation, Value/Intensity/Luminance) color space describes color with intuitive values, based on the artist's idea of tint, saturation and tone. This was introduced when there was a need to specify color properties numerically. Hue defines the dominant color as described by wavelength, for instance the distinction between red and yellow. Saturation measures the colorfulness of an area in proportion to its brightness such as the distinction between red and pink. Value refers to the color luminance, the distinction between a dark

red and a light red. For skin detection, the value component is discarded to eliminate the undesirable effect of uneven illumination. The transformation is defined by

$$H = \arccos \frac{\frac{1}{2}(R-G)+(R-B)}{\sqrt{(R-G)^2+(R-B)(G-B)}} \tag{2.14}$$

$$S = 1 - \frac{3min(R,G,B)}{R+G+B} \tag{2.15}$$

$$V = \frac{R+G+B}{3} \tag{2.16}$$

Some studies show that HSV is invariant to highlights at white light sources, to matte surfaces, and ambient lighting. However, hue discontinuities and the computation of the luminance component conflict badly with the properties of color vision.

The YCbCr Color Space

YCbCr is a family of color spaces used as a part of the color image pipeline in video and digital photography systems. Y' is the luma component and CB and CR are the blue-difference and red-difference chroma components. Y' (with prime) is distinguished from Y which is luminance, meaning that light intensity is non-linearly encoded using gamma.

Y'CbCr is not an absolute color space, it is a way of encoding RGB information. The actual color displayed depends on the actual RGB colorants used to display the signal. Therefore a value expressed as Y'CbCr is only predictable if standard RGB colorants or an ICC profile are used.The transformation from RGB to YCbCr is defined by

$$Y' = 16 + (65.481R + 128.553G + 24.966B) \tag{2.17}$$

$$Cb = 128 + (-37.797R - 74.203G + 112B) \tag{2.18}$$

$$Cr = 128 + (112R - 93.786G - 18.214B) \tag{2.19}$$

3. IMPLEMENTATION

3.1. Skin Based Approach

We have implement skin based classifier for nudity detection in Images. Skin based implementation composed of

- Skin segmentation algorithm
- Feature selection and SVM training

3.1.1. Skin Segmentation Algorithm

Skin segmentation is widely used in human and face detection applications. Histogram based algorithm are very popular due to its proved efficiency.

Histogram density for skin and non-skin pixels is calculated from the dataset (Compaq Dataset and manually collected images).The color of skin in the visible spectrum depends primarily on the concentration of melanin and hemoglobin. Distribution of the skin colors is affected by the ethnicity an illumination conditions. However, under arbitrary conditions of illumination the variation in skin color will be less constrained. Given a sufficiently large collection of labeled images captured under a wide variety of imaging conditions, we can model the distribution of skin and non-skin colors accurately.

We constructed skin and non-skin histogram models using our classifier training set of

images. Given skin and non-skin histograms we can compute the probability that a given color value belongs to the skin and non-skin classes:

$$P(rbg|skin) = \frac{s[rgb]}{T_s} \tag{3.1}$$

$$P(rbg|\neg skin) = \frac{n[rgb]}{T_n} \tag{3.2}$$

where s[rgb] is the pixel count contained in bin rgb of the skin histogram, n[rgb] is the equivalent count from the non-skin histogram, and T_s and T_n are the total counts contained in the skin and non-skin histograms, respectively.

From the skin and non-histogram models we can construct skin pixel classifier. Skin pixel classifier can be mathematically expressed in terms of likelihood ratio as:

$$\frac{P(rgb|skin)}{P(rgb|\neg skin)} \geq \theta \tag{3.3}$$

Where $0 \leq \theta \geq 1$ is the threshold and can be adjusted to trade-off between correct detections and false positives. We can write as θ a function of the priors and the cost of false positives and false negatives:

$$\theta = \frac{c_p P(\neg skin)}{c_n P(skin)} \tag{3.4}$$

Where c_p and c_n are the application-dependent cost of false positives and false negatives, respectively. One reasonable choice of priors is $P(skin) = T_s/(T_s + T_n)$. Using the likelihood equation each pixel in and image can be classified as skin or non-skin.

We also implemented gaussian mixture model for skin detection. we used pre-computed mixture parameter [1].It was found that gaussian mixture model is computationally expensive and doesn't fit into our performance requirement.

Feature selection and SVM training

We computed feature vector from the output of the skin detector and trained a classifier on these features to determine whether the image is adult or not. We tested for different set of features vector consisting of Percentage of pixels detected as skin, Average probability of the skin pixels, size in pixels of the largest connected component of skin, Number of connected components of skin etc. However, the system didn't meet our desired accuracy.

3.1.2. Gaussian Mixture Model

Mixture model is a probabilistic model for density estimation. A mixture model can be regarded as a type of unsupervised learning. Mixture models consist of collection of component function, usually Gaussian. These component functions are combined to provide a multimodal density. They can be employed to model the skin color distribution in order to perform colour-based skin segmentation. A mixture density function is expressed as the sum of gaussian kernels:

$$P(x) = \sum_{i=0}^{N} w_i \frac{1}{(2\pi)^{\frac{3}{2}} \| \sum_i \|^{\frac{1}{2}}} e^{\frac{1(x-\mu_i)^T \sum_i^- 1(x-\mu_i)}{2}}, \tag{3.5}$$

Where x is an RGB color vector and the contribution of i^{th} Gaussian is determined by a scalar weight w_i , mean vector μ_i and diagonal covariance matrix $\sum_i$.

For modeling skin and non skin classes 16 Gaussian models were used. During this project we used pre-computed [1] Gaussian parameter for skin and non skin models.

Kernel	Mean	CoVariance	Weight
1	(73.53,29.94,17.76)	(765.40,121.44,112.80)	0.0294
2	(249.71,233.94,217.49)	(39.94,154.44,396.05)	0.0331
3	(161.68,116.25,96.95)	(291.03,60.48,162.85)	0.0654
4	(186.07,136.62,114.40)	(274.95,64.60,198.27)	0.0756
5	(189.26,98.37,51.18)	(633.18,222.40,250.69)	0.0554
6	(247.00,152.20,90.84)	(65.23,691.53,609.92)	0.0314
7	(150.10,72.66,37.76)	(408.63,200.77,257.57)	0.0454
8	(206.85,171.09,156.34)	(530.08,155.08,572.79)	0.0469
9	(212.78,152.82,120.04)	(160.57,84.52,243.90)	0.0956
10	(234.87,175.43,138.94)	(163.80,121.57,279.22)	0.0763
11	(151.19,97.74,74.59)	(425.40,73.56,175.11)	0.1100
12	(120.52,77.55,59.82)	(330.45,70.34,151.82)	0.0676
13	(192.20,119.62,82.32)	(152.76,92.14,259.15)	0.0755
14	(214.29,136.08,87.24)	(204.90,140.17,270.19)	0.0500
15	(99.57,54.33,38.06)	(448.13,90.18,151.29)	0.0667
16	(238.88,203.08,176.91)	(178.38,156.27,404.99)	0.0749

Table 3.1: Mixture Of Gaussian Skin Color model

Kernel	Mean	CoVariance	Weight
1	(254.37,254.41,253.82)	(2.77,2.81,5.46)	0.0637
2	(9.39,8.09,8.52)	(46.84,33.59,32.48)	0.0516
3	(96.57,96.95,91.53)	(280.69,156.79,436.58)	0.0864
4	(160.44,162.49,159.06)	(355.98,115.89,591.24)	0.0636
5	(74.98,63.23,46.33)	(414.84,245.95,361.27)	0.0747
6	(121.83,60.88,18.31)	(2502.24,1383.53,237.18)	0.0365
7	(202.18,154.88,91.04)	(957.42,1766.94,1582.52)	0.0349
8	(193.06,201.93,206.55)	(562.88,190.23,447.28)	0.0649
9	(51.88,57.14,61.55)	(344.11,191.77,433.40)	0.0656
10	(30.88,26.84,25.32)	(222.07,118.65,182.41)	0.1189
11	(44.97,85.96,131.95)	(651.32,840.52,963.67)	0.0362
12	(236.02,236.27,230.70)	(225.03,117.29,331.95)	0.0849
13	(207.86,191.20,164.12)	(494.04,237.69,533.52)	0.0368
14	(99.83,148.11,188.17)	(955.88,654.95,916.70)	0.0389
15	(135.06,131.92,123.10)	(350.35,130.30,388.43)	0.0943
16	(135.96,103.89,66.88)	(806.44,642.20,350.36)	0.0477

Table 3.2: Mixture Of Gaussian Non-Skin Color model

3.2. Non Skin Based Approach

3.2.1. MPEG-7

The MPEG-7 standard[2] is an international standard since September 2001 which specifies metadata for describing multimedia content. The interesting part for our project is this part of the standard which defines visual descriptors. These are structures to describe multimedia data. Their exact extraction methods are not standardized. shows an overview of MPEG-7 visual descriptors which are suitable for still images. Since 2001, lots of research has been conducted on making use of these standardized visual descriptors in the field of Computer Vision, especially in Content-Based Image Retrieval Systems

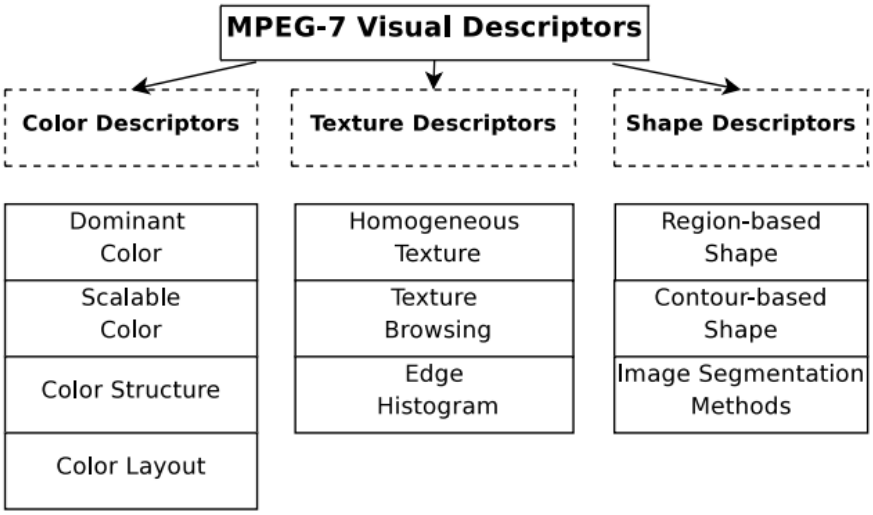


Figure 3.1: MPEG-7 Visual Descriptors For Low-level Features.

Dominant ColorDescriptor

Dominant Color Descriptor (DCD) provides an effective, compact and intuitive description of the representative colors in an image or region . The descriptor consists of the Color Index (ci), Percentage (pi), Color Variance (vi) and Spatial Coherency (s); the last two parameters

are optional. Then the DCD is defined by:

$$F = (c_i, p_i, v_i), s, i = 1, ..., N \tag{3.6}$$

where N is the number of the colors and . For dominant color extraction, the generalized Lloyd algorithm is used for color clustering. There is one overall Spatial Coherency (SC) value for the whole image and several groups of (c_i, p_i, v_i) for the corresponding dominant colors. The Perceptual colors to represent images based on dominant colors 3.3

Color	Red	Green	Blue
red-brown	0	0	132
green	0	132	0
brown-green	0	132	132
dark blue	132	0	0
purple	132	0	132
blue-green	132	132	0
dark-grey	132	132	132
light-grey	198	198	198
orange-red	0	0	255
vivid-green	0	255	0
yellow	0	255	255
blue	255	0	0
pink-purple	255	0	255
light-blue	255	255	0
white	255	255	255

Table 3.3: Perceptual colors to represent images based on dominant colors

DCD Extraction

The extraction procedure for the dominant color uses the Generalized Lloyd Algorithm (GLA)[3] to cluster the pixel color values.In our system, we quantized the colors into 18 colors as shown in Table 3.3. After defining the colors, for each image implement the following steps:

- Read in the image and create an image array that contains the RGB components of each pixel in the image

- For each pixel in the image do:
  - Search color table for the nearest color by finding the distance between the pixel color I represented as (P_r, P_g, P_b) and the color in the color table C_i represented as (C_{iR}, C_{iG}, C_{iB}) using the distance formula (3):

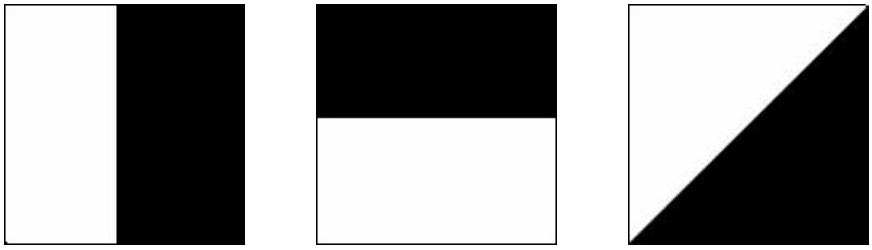
$$C_d = (\sqrt{(P_r - C_{ir})^2 + (P_g - C_{ig})^2 + (P_b - C_{ib})^2}),$$

    i=1,2....18
  - Assign to the pixel the RGB entry in color table for which C_d is the minimum
- Create a frequency table for each assigned color
- Sort the frequency table in descending order MPEG-7 DCD allows at most eight colors to be represented.The highest four frequent colors are then selected with their percentages to create the description of the image.

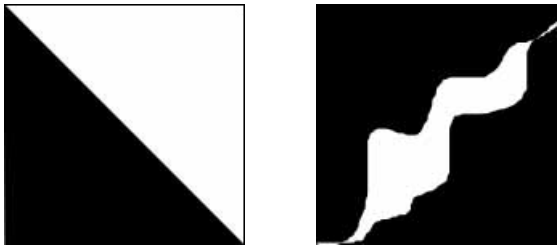
Edge Histogram Descriptor

The EHD represents the spatial distribution of edges in an image. The extraction process of the EHD consists of the following stages:

- The edges in each image-block is categorized into one of the following six types: vertical, horizontal, 45 degree diagonal, 135 degree diagonal, nondirectional edge and no-edge.
- Now a 5-bin edge histogram of each subimage can be obtained. (See 3.2)
- Each bin value is normalized by the total number of image-blocks in the image.
- The normalized bin values are nonlinearly quantized.



(a) Horizontal Edge (b) Vertical Edge (c) 45 degree edge



(d) 135 degree edge (e) non-directional edge

Figure 3.2: Five types of Edges

Color Structure descriptor

This descriptor expresses local color structure in an image using an 8 x 8-structuring element. It counts the number of times a particular color is contained within the structuring element as the structuring element scans the image.[4] Suppose $c_0, c_1, c_2, \ldots c_{M-1}$ denote the M quantized colors. A color structure histogram can then be denoted by $h(m)$,m=0,1,2,...M-1 where the value in each bin represents the number of structuring elements in the image containing one or more pixels with color c_m. The HMMD color space is used in this descriptor.

Component	Subspace	No. of quantisation levels for different no. of histogram bins			
		184	120	64	32
Hue	0	1	1	1	1
	1	8	4	4	4
	2	12	12	6	3
	3	12	12	4	2
	4	24	12	4	2
Sum	0	8	8	8	8
	1	4	4	4	2
	2	4	4	4	4
	3	4	4	4	2
	4	2	4	4	2

Table 3.4: HMMD Color Space Quantization for CSD

The CSD is defined using four color space quantization operating points: 184, 120, 64, and 32 bins. To construct a 184-level quantized color, HMMD color space is quantized nonuniformly as follows. The whole HMMD color space is divided into five subspaces. This sub-space division is performed on the diff parameter . For the respective subspaces, uniform color quantization on the Hue and Sum values results in a 184-level color quantization. The number of quantization levels for each subspace for different number of histogram bins is given in table 3.4.

In order to compute the CSD, an 8x8-structuring element is used. Even though the total number of samples is kept fixed at 64, the spatial extent of the structuring element scales with the image size. The following simple rule determines the spatial extent of the structuring element (equivalently, the sub sampling factor) given the image size:

$$p = max\{0, round(0.5log_2WH - 8)\}$$
$$k = 2^p, E = 8K \tag{3.7}$$

where

W,H image width and height, respectively;

ExE spatial extent of the structuring element;

K sub-sampling factor.

For images smaller than 256x256 pixels, an 8x8 element with no sub-sampling is used. As another example, if the image size is 640x480, then p=1,K=2 and E=16. So, every alternate sample along the rows and columns of a 16x16 -structuring element is then used to compute the histogram.

Each bin of the CSD h(m) represents the number of locations of the structuring element at which a pixel with color c_m falls inside the element. The origin of the structure element is defined by its top-left sample. The locations of the structure element over which the descriptor is accumulated are defined by the grid of pixels of the possibly sub-sampled input image. Structuring element is as shown in fig 3.3

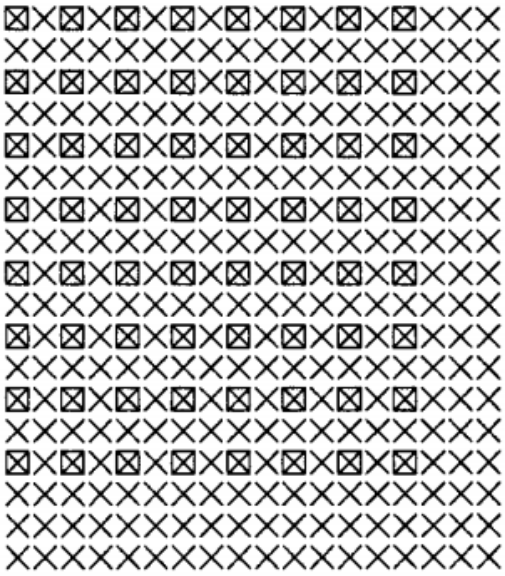


Figure 3.3: Structuring element for images of 640x480

Scalable Color Descriptor

The SCD addresses the interoperability issue by fixing the color space to HSV, with a uniform quantization of the HSV space to 256 bins. The bin values are nonuniformly quantized to a 11-bit value.

This method achieves full interoperability between different resolutions of the color representation, ranging from 16 bits/histogram at the low end to approximately 1000 bits/histogram at the high end.Of course, the accuracy of the feature description is highly dependent on the number of bits used. However, core experiments have shown that good retrieval results are still achievable using only 64 bits, while excellent results can be obtained using medium or full resolution of the descriptor.

The HSV space is uniformly quantized into a total of 256 bins. This includes 16 levels in H, four levels in S, and four levels in V. The histogram values are truncated into a 11-bit integer representation. To achieve a more efficient encoding, the 11-bit integer values are first mapped into a “nonlinear” 4-bit representation, giving higher significance to the small values with higher probability. This 4-bit representation of the 256-bin HSV

Total bits	Hbits	Sbits	Vbits
16	4	2	2
32	8	2	2
64	8	2	4
128	8	4	4
256	16	4	4

Table 3.5: Equivalent Partitioning Of The HSV Color Space

histogram would require 1024 bits/histogram, which is too large a number in the context of many MPEG-7 applications. To lower this number and make the application scalable, the histograms are encoded using a Haar transform. The basic unit of the Haar transform consists of a sum operation and a difference operation [see Fig. 4(a)], which relate to primitive low- and high-pass filters. Summing pairs of adjacent histogram lines is equivalent to the calculation of a histogram with half number of bins.

The high-pass (difference) coefficients of the Haar transform express the information contained in finer-resolution levels (with higher number of bins) of the histogram. Natural image signals usually exhibit high redundancy between adjacent histogram lines. This can be explained by the "impurity" (slight variation) of colors caused by variable illumination and shadowing effects. Hence, it can be expected that the high-pass coefficients expressing differences between adjacent histogram bins usually have only small values. Exploiting this property, it is possible to truncate the high-pass coefficients to integer representation with only a low number of bits.

3.2.2. Edge and Moment Approach

Edge Detection

An edge in an image is a contour across which the brightness of the image changes abruptly. In image processing,an edge is often interpreted as one class of singularities. In a function, singularities can be characterized easily as discontinuities where the gradient approaches infinity. However, image data is discrete, so edges in an image often are defined as the local maxima of the gradient.

Edge detection is an important task in image processing. It is a main tool in pattern recognition, image segmentation, and scene analysis. An edge detector is basically a high-pass filter that can be applied to extract the edge points in an image.

Classical Edge Detectors

Many classical edge detectors have been developed over time. They are based on the principle of matching local image segments with specific edge patterns. The edge detection is realized by the convolution with a set of directional derivative masks.The popular edge detection operators are Roberts, Sobel, Prewitt, Frei-Chen, and Laplacian operators.

Sobel Edge Detector

The Sobel operator performs a 2-D spatial gradient measurement on an image and so emphasizes regions of high spatial frequency that correspond to edges. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image.

$$G_x = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$G_y = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these Gx and Gy). These can then be combined together to find the absolute magnitude of the gradient at each point and the ori-

entation of that gradient. The gradient magnitude is given by:

$$|G| = \sqrt{G_x^2 + G_y^2} \tag{3.8}$$

Typically, an approximate magnitude is computed using:

$$|G| = |G_x| + |G_y| \tag{3.9}$$

which is much faster to compute. The angle of orientation of the edge (relative to the pixel grid) giving rise to the spatial gradient is given by:

$$\theta = arctan(\frac{G_y}{G_x}) \tag{3.10}$$

The Sobel operator represents a rather inaccurate approximation of the image gradient, but is still of sufficient quality to be of practical use in many applications. More precisely, it uses intensity values only in a 33 region around each image point to approximate the corresponding image gradient, and it uses only integer values for the coefficients which weight the image intensities to produce the gradient approximation.

Image Moments

Spatial and central moments are important statistical properties of an image.An image moment is a certain particular weighted average (moment) of the image pixels' intensities, or a function of such moments, usually chosen to have some attractive property or interpretation. Image moments are useful to describe objects after segmentation. Simple properties of the image which are found via image moments include area (or total intensity), its centroid, and information about its orientation.

For a 2-D moment of order (p+q) of a digital image f(x,y) is defined as

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y) \tag{3.11}$$

for p,q=0,1,2,....., where the summations are over the values of the spatial coordinates x and y spanning the image. The corresponding *central moment* is defined as

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q f(x,y) \tag{3.12}$$

where

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad and \quad \bar{y} = \frac{m_{01}}{m_{00}}$$

The *normalized central moment* of order (p+q) is defined as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}$$

for p,q=0,1,2,...., where

$$\gamma = \frac{p+q}{2} + 1$$

for p+q=2,3,.....

A set of seven 2-D *moment invariants* that are insensitive to translation,scale change , mirroring and rotation can be derived from these equations. They are

$$\phi_1 = \eta_{20} + \eta_{02} \tag{3.13}$$

$$\phi_2 = (\eta_{20} - eta_{02})^2 + 4\eta_{11}^2 \tag{3.14}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \tag{3.15}$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \tag{3.16}$$

$$\begin{aligned}\phi_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) \\ & [(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\ & (3\eta_{21} - \eta_{03}(\eta_{21} + \eta_{03}) \\ & [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03}^2]\end{aligned} \tag{3.17}$$

$$\begin{aligned}\phi_6 = {} & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12}^2 - (\eta_{21} + \eta_{03})^2] \\ & + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_2 1 + \eta_{03})\end{aligned} \tag{3.18}$$

$$\begin{aligned}\phi_7 = {} & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 \\ & - 3(\eta_{21} + \eta_{03})^2] + (e\eta_{12} = \eta_{30})(\eta_{21} + \eta_{03}) \\ & [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]\end{aligned} \tag{3.19}$$

These sets of equations are commonly referred as Hu set of invariant moments.[5]

3.2.3. BIC

BIC is a pixel classification algorithm that labels pixels into two categories Border and Interior. BIC is based upon Point image processing algorithm.

The BIC approach basically composed of three main components:

- a simple and powerful image analysis algorithm that classifies image pixels as border or interior
- a logarithmic distance to compare histograms
- a compact representation for the visual features extracted from images.

Image Analysis

BIC image analysis algorithm relies on the RGB color-space uniformly quantized in 4X4X4=64 colors. BIC can be implemented using other color-space quantization for eg. YCbCr, HSV etc. After the quantization step, image pixels are classified as border or interior pixels. A pixel is classified as border if it is at the border of the image itself or if at least one of its 4-neighbors (top, bottom, left and right) has a different quantized color. A pixel is classified as interior if its 4-neighbors have the same quantized color. It is important to observe that this classification is mutually exclusive (either a pixel is border or it is interior) and it is based on a inherently binary visual property of the images. We choose 4-neighbors instead of 8-neighbors because, given the simplicity and generality of the problem, the use of 4-neighbors is able to reduce the image analysis complexity without perceptual losses in terms of retrieval effectiveness.

After the image pixels are classified, one color histogram is computed considering only border pixels, and another color histogram is computed considering only interior pixels. In this way, we have the border/interior classification represented for each quantized color.

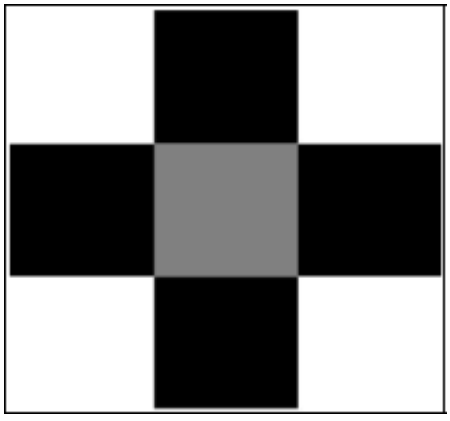


Figure 3.4: Neighbour BIC Classification

Grayed pixel as shown in figure is classified by comparing to its four neighbors (black squares) to classify whether the pixel is interior or Boundary. A pixel is classified as interior if its 4-neighbors have the same quantized color. It is important to observe that this classification is mutually exclusive (either a pixel is border or it is interior). The four neighboring BIC classification can be extended to 8- neighbor scheme. However due to the simplicity

and performance we choose 4-neighbors scheme

Logarithmic Distance function

Image analysis algorithm describes an image by the means of two color histogram with 64 bins. Two histogram can be combined into a single histogram with 64 bins so that vectorial distance function like L1-Norm and L2-Norm can be used to compare the visual features.

Vectorial distances have also well-known limitations. One of such limitations is that a high value in a single histogram bin dominates the distance between two histograms, no matter the relative importance of this single value.To avoid the problem of high values in a single histogram bin dominate the distance between two histograms, the dLog distance function is used. This function uses a logarithm scale reducing 28 times the range of distances between the smallest and the largest histogram bin values, given that in the log-scale a bin is normalized in the interval [0,9].

The dLog function compares histograms in a logarithmic scale, and is defined as:

$$dLog(q,d) = \sum_{i=0}^{M} |f(q[i]) - f(d[i])| \tag{3.20}$$

$$f(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{if } 0 < x \leq 1 \\ |\log_2 x| & \text{,otherwise} \end{cases} \tag{3.21}$$

In the given equation 3.20, q and d are two histograms with M bins each. The value q[i] represents the i^{th} bin of histogram q and d[i] represent the i^{th} bin of histogram d. The histogram bins are normalized between 0 and 255.

The comparison of histograms with the dLog function does not solve the problem of histogram bins with very high values, but diminishes its effects in most of the situations. In a log-scale, the difference between the largest and the smallest distances between histogram bins becomes smaller than in the original scale. In the original scale, the smallest distance

between histogram bins is zero (both images have the same amount of a particular color) and the largest distance is 255 (when the images have just one color and they are different). In our log-scale, the smallest distance is 0 and the largest distance is just 9. The range of distances in the original scale is thus 255=9 = 28 times larger than in the proposed log-scale.

Compact Visual Features Representation

After applying the DLog function we obtain one histogram with 2XQ bins where each bin contains integer values between 0 and 9.Each image is compactly represented by this histogram vector where each position represents a bin. Thus produced compact representation of the images can be directly plugged into SVM as a features vector.

3.3. Final system

After Performing analysis and running tests on different methodologies ,we concluded BIC was among the best. To further enhance the predicting accuracy of the system, we plugged a naive skin based classifier before sending to the BIC process.Images with skin per cent less than 25 were simply returned as non adult. For other images, feature vectors were calculated by the BIC feature extraction process and final output was given on the basis of the prediction of BIC Classifier for the calculated feature vector.The system is as shown in figure **??**.

Figure 3.5: Final System Block Diagram

The above system performed quite better than BIC classifier alone and it also reduced the overload to the BIC classifier.

4. PERFORMANCE & ANALYSIS

During the course of our project, we experimented and tested a number of algorithms and methods to compare different techniques regarding the adult image detection.Training and Testing were done on a desktop with following specifications.

- Processor : Intel(R) Core(TM)2 Duo CPU @ 2.8GHz
- Memory : 2GB
- Operating System : Windows Xp/Windows 7
- Python 2.6 , Matlab 2007

For our project purpose,we used a part of images from Compaq Dataset[1] collected by contacting Michael Jones, MERL Research. Due to lack of diverse and monotonous images, we also downloaded and added images to our data set for training and testing purpose.After the images were collected we used our manual labeler to perform manual classification.

We used the labeler for labeling images. For our project, we created data set with following statistics.

Training Dataset Total number of images=1928
No of adult imges = 942
No of Non-adult images = 986

Testing Dataset Total number of images=452

Number of adult images = 202

Number of Non-adult images = 250

Major algorithms that we worked on are as follows.

4.1. Analysis of Skin Based Segmentation

We studied human skin color distribution in RGB and YCbCr color space.Compaq skin data set is used to plot the color distributions. Color distributions were analyzed by plotting each channel of five color-space against each other.plot for RGB and YCbCr color space are shown in figures below.

(a) Input Image (b) Skin Pixels

Figure 4.1: Skin Segmentation

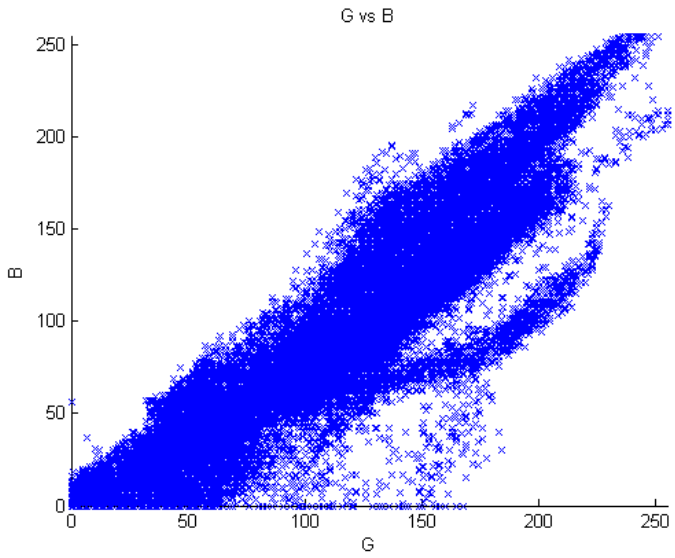


Figure 4.2: G v/s B

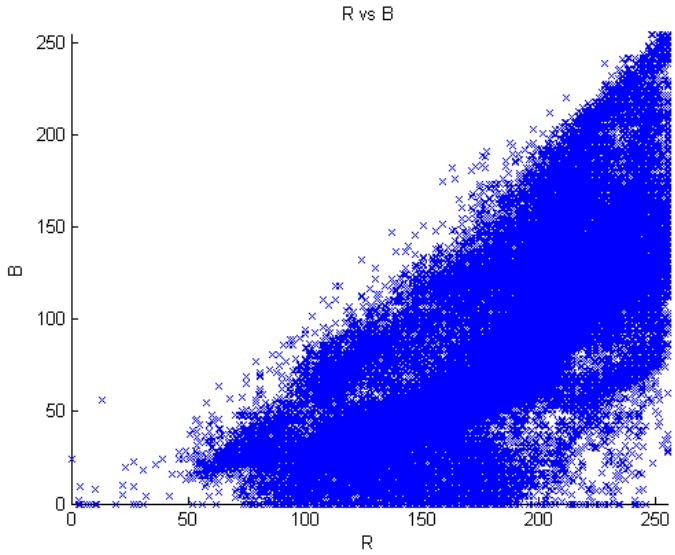


Figure 4.3: R v/s B

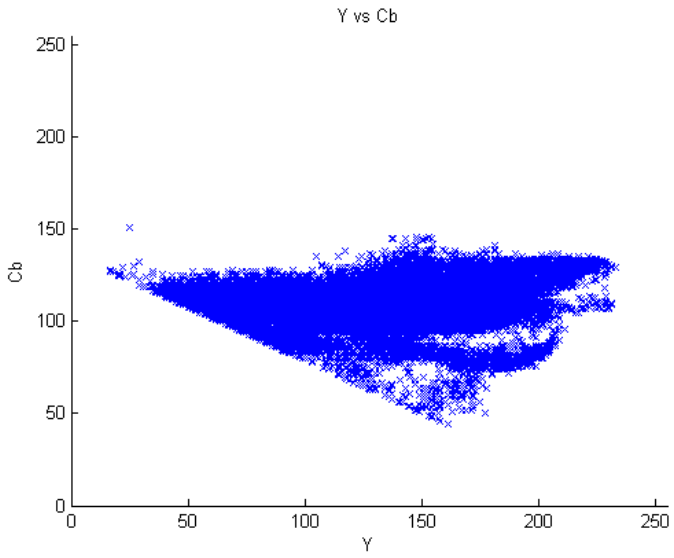


Figure 4.4: Y v/s Cb

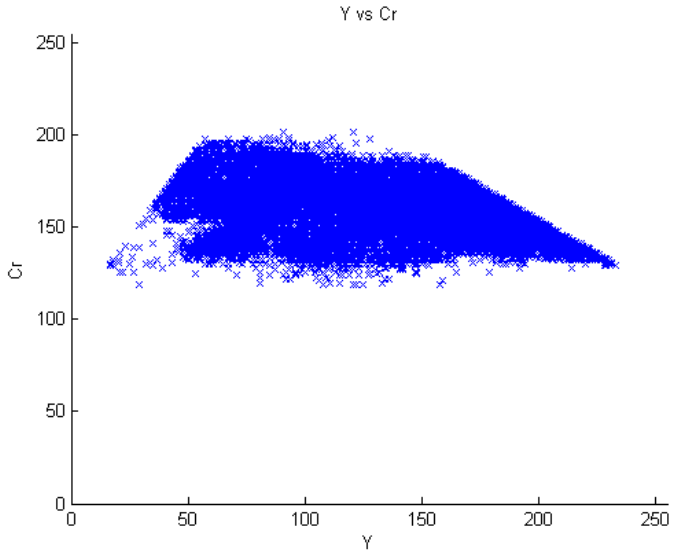


Figure 4.5: Y v/s Cr

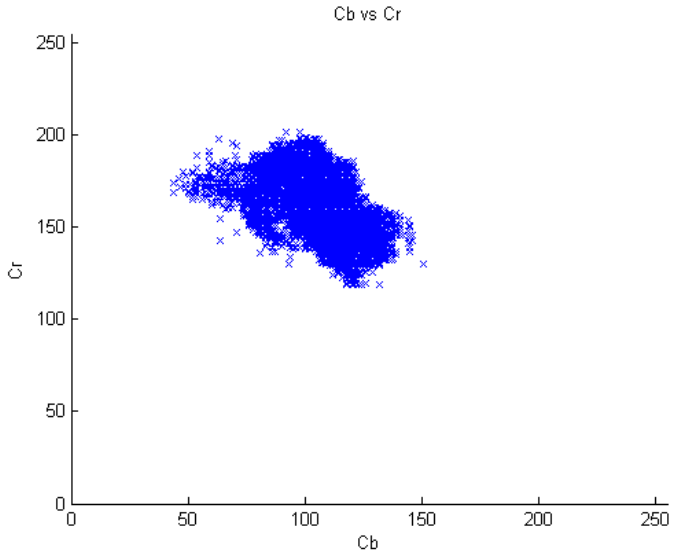


Figure 4.6: Cb v/s Cr

Findings from the plot:

From the skin distribution plot in RGB color-space, we found that skin color is widely spread out in each channel dimension. However the skin distribution plot in YCbCr space shows that skin color is confined in a small portion of Cb and Cr color range. From these findings we defined a range for skin segmentation as:

$$f(x) = \begin{cases} Skin & \text{if } (Cb_{x,y} \in [77,127] \quad \& \quad Cr_{x,y} \in [133,173]) \\ Non-Skin & \text{,otherwise} \end{cases} \tag{4.1}$$

4.2. Analysis of Mpeg-7 Descriptors

MPEG-7, formally named “Multimedia Content Description Interface”, is a standard for describing the multimedia content data that supports some degree of interpretation of the information meaning, which can be passed onto, or accessed by, a device or a computer code. MPEG-7 is not aimed at any one application in particular; rather, the elements that MPEG-7 standardizes support as broad a range of applications as possible.We have used these descriptors as a tool to generate feature vectors for the training purpose and prediction.

4.2.1. Analysis of dominant color Descriptor

Feature Vector Calculation

In this method, we calculated the feature vector of length 8(4 dominant colors + their percentage) as the feature vector and fed it to the training model.We quantized the colors into 18 colors as shown in table 3.3.

Experimental Results

We obtained a cross validation accuracy of 58.6283 per cent and a testing accuuracy of 70.354 per cent on the testing data set. The ROC obtained is as shown in fig4.7

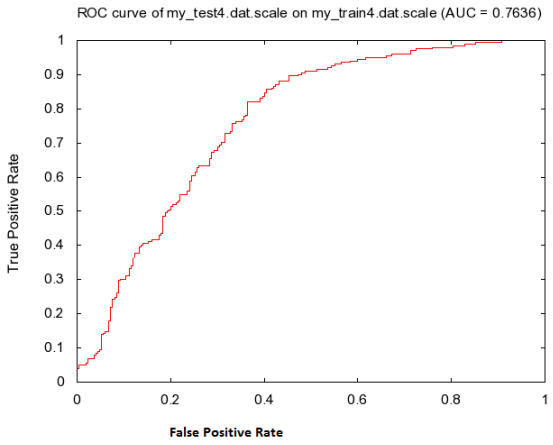


Figure 4.7: ROC for the DCD

4.2.2. Analysis of Scalable Color Descriptor & EHD

Feature vector Extraction

In this technique, a total of 69 features (64 bins scalable color histogram descriptor + 5 edge descriptor) were used in the feature vector. Feature vectors were calculated[4], and fed to the

SVM for learning purpose.

Experimental Result

We obtained a cross validation accuracy of 84.7922 per cent and a testing accuracy of 78.3186 per cent on the training data set.The ROC obtained is as shown in figure4.8

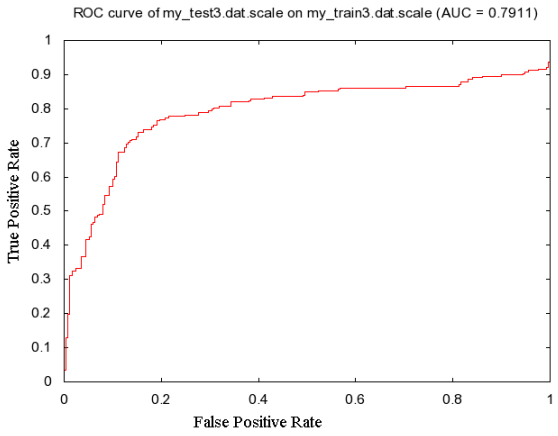


Figure 4.8: ROC for the SCD+EHD

4.2.3. Analysis of Color Layout Descriptor

Experimental Results

4.3. Analysis of Edge and Moments

Feature Vector Calculation

In this technique, a total of 28 features (21 moments up to order five + 7 moment invariants) were used in the feature vector. The input image was first converted to an equivalent edge map using the sobel edge filter and then feature vectors were calculated. we compute the normalized central moments up to order ve and the translation,rotation and scale invariant based

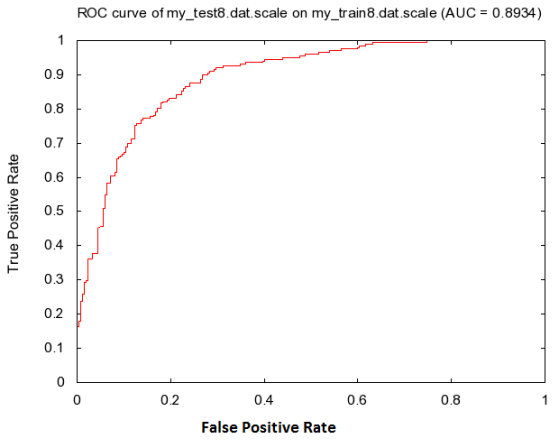


Figure 4.9: ROC for the CLD

on the grayscale edge image using the definitions.Feature vector containing these 21+7=28 moments is computed and stored in the training database.[6]

Experimental Results

We obtained a cross validation accuracy of 71.586 per cent and a testing accuracy of 61.3097 per cent on the training data set.The ROC obtained is as shown in figure4.10

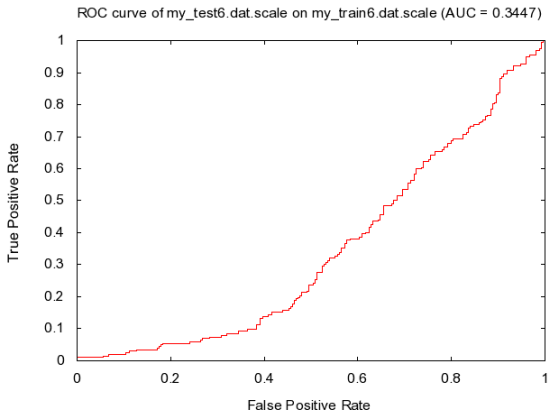


Figure 4.10: ROC for the Moment

4.4. Analysis of BIC

Feature vector extraction:

The output format of BIC, a 2 Q bin histogram, is easily adapted to serve as input to the SVM classifier. We calculated the feature vector of length 128(BIC histogram) and 129(BIC histogram and percent skin area). We constructed SVM model using both 128 and 129 features vector scheme.

(a) Input Image

(b) Interior Pixels

(c) Boundary Pixels

Figure 4.11: BIC pixel classification

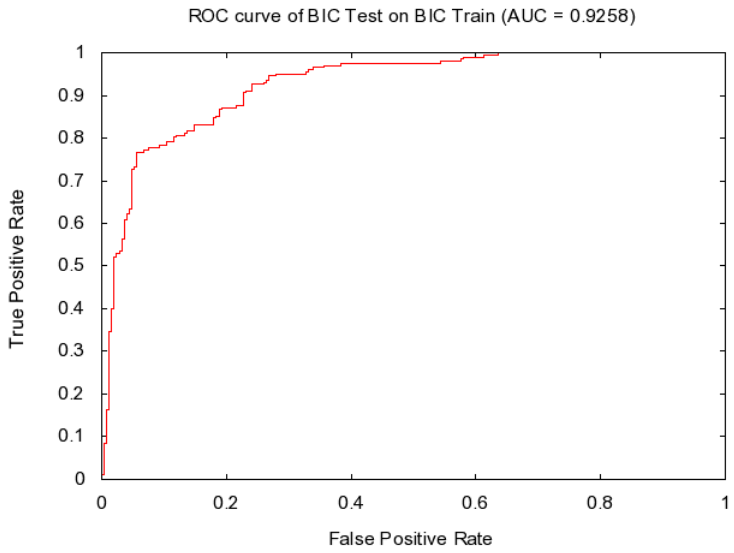


Figure 4.12: ROC for the BIC

5. CONCLUSIONS AND FURTHER WORK

5.1. Conclusions

During the course of the project, we tried and tested different prevalent approaches to Adult image detection and tried to optimize them for the best results. We tried the naive skin based approach, Mpeg-7 descriptor based approach and BIC approach for the adult image detection.These all methods had different time complexity and different accuracies.From the results obtained from our project, we concluded that BIC combined with skin color method was the most suitable and gave the best results.

5.2. Future Work

At present, We have the efficiency of our system around the early 80 per cent mark. We would love to add some more features that stands out and push the efficiency towards the 90 per cent mark.More research needs to be done regarding the adult image classification problem and it would be more efficient if we could plug an organ detecting to the system.

A. Matlab Codes

Matlab Code For Calculating Skin Percent

```
function percent=calculateSkinPercent(RGBimage)
   YCbCrimage=rgb2ycbcr(RGBimage);

   min_Cb = 77;
   max_Cb = 127;
   min_Cr = 133;
   max_Cr = 173;

   % get a desirable binary image with the acquired range
   imag_row=size(YCbCrimage,1);
   imag_col=size(YCbCrimage,2);

   binImage=zeros(imag_row,imag_col);

   Cb=zeros(imag_row,imag_col);
   Cr=zeros(imag_row,imag_col);

   Cb(find((YCbCrimage(:,:,2) >= min_Cb) & (YCbCrimage(:,:,2) <= max_Cb)))=1;
   Cr(find((YCbCrimage(:,:,3) >= min_Cr) & (YCbCrimage(:,:,3) <= max_Cr)))=1;
   binImage= (Cb.*Cr);
   skin_pixels = size(nonzeros(binImage));
   skin_pixels_count = skin_pixels(1);
```

```
    percent = (skin_pixels_count)/(imag_row*imag_col);
```

Matlab Code For Genarating Dlog Histogram

```
 function [ dlogHistogram ] = generateDlogHistogram( histogramCounts )
%GENERATEDLOGHISTOGRAM Summary of this function goes here
%   Detailed explanation goes here
    dlogHistogram = zeros(size(histogramCounts));
    for i= 1 : length(histogramCounts)
        value = histogramCounts(i);
        if(value == 0)
            dlogHistogram(i) = 0;
        elseif(value ==1)
            dlogHistogram(i) = 1;
        else
            dlogHistogram(i) = round(log2(value)+1);
       end
    end
```

Matlab Code For Genarating optimised Histogram

```
 function[hist_count] = generateHistogramOptimized(img,BicMask,totalColor,nbins)
%GENERATEHISTOGRAMOPTIMIZED Summary of this function goes here
%   Detailed explanation goes here
    quantizedImage = uint8(floor(double(img)./totalColor));

    %extract RGB components of quantizedImage
    red = quantizedImage(:,:,1);
    green = quantizedImage(:,:,2);
    blue = quantizedImage(:,:,3);

    imageDimension = size(img);
    numRows = imageDimension(1);
    numColumns = imageDimension(2);

    interiorPointsCount = length(nonzeros(BicMask));
    boundaryPointsCount = (numRows*numColumns) - interiorPointsCount;
    boundary_data = zeros(1,boundaryPointsCount);
    interior_data = zeros(1,interiorPointsCount);
    interior_counter = 1;
    boundary_counter = 1;

    for row = 1:numRows
        for column = 1:numColumns
            value = red(row,column)*16+green(row,column)*4+blue(row,column)*1;
            if(BicMask(row,column)~=0)
                %interior_data = [interior_data value];
                interior_data(interior_counter) = value;
                interior_counter = interior_counter+1;
            else
                boundary_data(boundary_counter) = value;
```

```
                boundary_counter = boundary_counter + 1;
                %boundary_data = [boundary_data value];
            end
        end
    end

    boundary_data = boundary_data(1:boundary_counter-1);
    interior_data = interior_data(1:interior_counter-1);
    data = [interior_data boundary_data];
    hist_count = hist(double(data),nbins);
```

Matlab Code For Genarating Feature List

```
 function features = generateFeatureList( img )
%GENERATEFEATURELIST generates the feature vectors (histogram)
%Input:
%     @img : image matrix of input image
%Output:
%     @data : matrix containing the features(historgram) of the images in
%     the sourcedir
     [ interior,interior_mask,boundary,boundary_mask ] = bicClassifier(img,64);
     [ hist_count ] = generateHistogramOptimized( img,interior_mask,64,128);
     [ norm_data ] = normalizeData( hist_count, 255 );
     [ features ] = generateDlogHistogram( norm_data );
```

Matlab Code For BIC Classifier

```
function[interior,interior_mask,boundary,boundary_mask]=bicCla-
                                                    ssifier(img,totalColor)
%BICHISTOGRAM Summary of this function goes here
%   Detailed explanation goes here

    %totalColor = floor(256/colorPerChannel);
    %quantize the image to specified bits per channel
    quantizedImage = floor(double(img)./totalColor);

    %extract RGB components of quantizedImage
    red = quantizedImage(:,:,1);
    green = quantizedImage(:,:,2);
    blue = quantizedImage(:,:,3);

    imageDimension = size(img);
    %interior = 255*(ones(imageDimension,'uint8'));
    %boundary = 255*(ones(imageDimension,'uint8'));

    interior = 255*(zeros(imageDimension,'uint8'));
    interior_mask = zeros(imageDimension(1:2));
    boundary = 255*(zeros(imageDimension,'uint8'));
    boundary_mask = zeros(imageDimension(1:2));

    numRows = imageDimension(1);
    numColumns = imageDimension(2);

    %initialize the boundary for given image
    boundary_mask(1,:)=1;
    boundary_mask(:,1)=1;
    boundary_mask(numRows,:)=1;
```

```
boundary_mask(:,numColumns)=1;

for row = 2:numRows-1
    for column = 2:numColumns-1
        left_red_color = red(row,column-1);
        left_green_color = green(row,column-1);
        left_blue_color = blue(row,column-1);
        if(red(row-1,column)==left_red_color && red(row,column+1) ==
        left_red_color && red(row+1,column) == left_red_color &&
        green(row-1,column) ==left_green_color && green(row,column+1) ==
        left_green_color && green(row+1,column) == left_green_color
        && blue(row-1,column) == left_blue_color && blue(row,column+1)
        == left_blue_color && blue(row+1,column) == left_blue_color)
            interior(row,column,:) = img(row,column,:);
            interior_mask(row,column) = 1;
        else
            boundary(row,column,:) = img(row,column,:);
            boundary_mask(row,column)=1;
        end
    end
end
```

B. List Of Figures

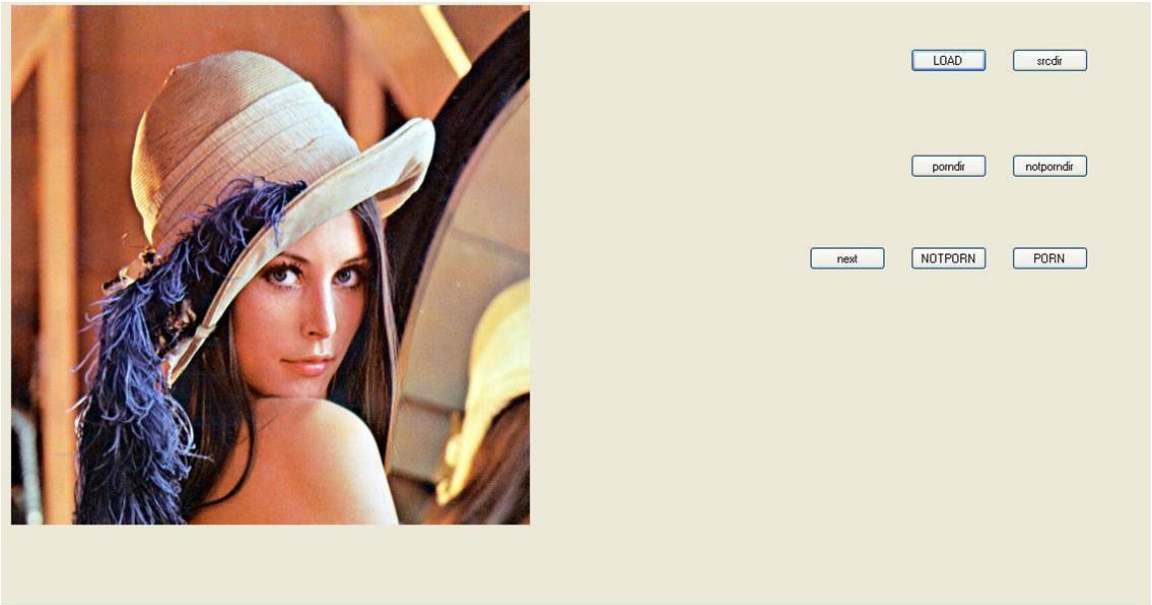


Figure B.1: Manual Labeler for Manual Classification

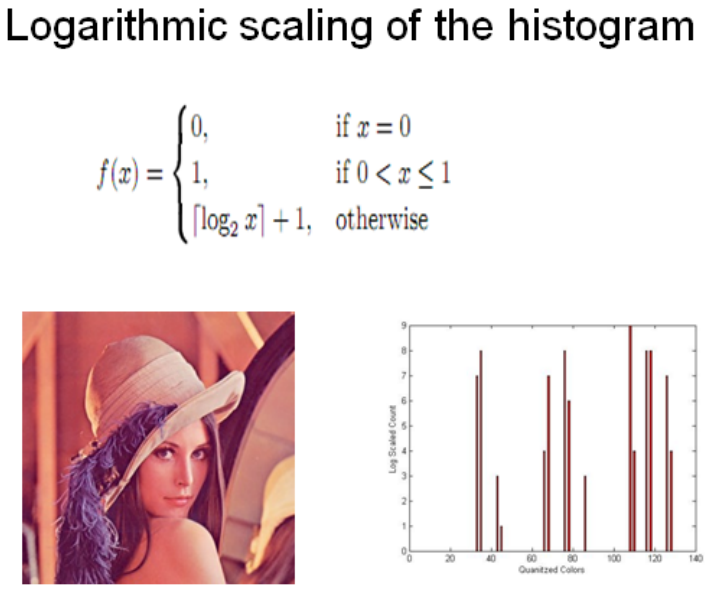


Figure B.2: BIC Procedure

Figure B.3: User Interface for Classification

Bibliography

[1] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," *International Journal of Computer Vision*, vol. 46, pp. 81–96, January 2002.

[2] J. M. M. Sanchez, "Mpeg-7: Overview of mpeg-7 description tools, part 2.," *IEEE MultiMedia*, vol. 9, no. 3, pp. 83–93, 2002.

[3] A. Gersho and R. Grey, *Vector Quantization and Signal Compression.* Massachusetts: Dover, ninth dover printing, tenth gpo printing ed., 1991.

[4] B. S. Manjunath, J.-R. Ohm, V. V. Vasudevan, and A. Yamada, "Color and texture descriptors," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 11, no. 6, pp. 703–715, 2001.

[5] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. IT-8, pp. 179–187, February 1962.

[6] J. Z. Wang, G. Wiederhold, and O. Firschein, "System for screening objectionable images using daubechies' wavelets and color histograms," in *IDMS '97: Proceedings of the 4th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services*, (London, UK), pp. 20–30, Springer-Verlag, 1997.

[7] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, 1950.