**FaceZen:**

**Java Source Code for Face Recognisation**

//face regonisation from images

package com.learn;  
 import org.opencv.core.\*;  
 import org.opencv.imgcodecs.Imgcodecs;  
 import org.opencv.imgproc.Imgproc;  
 import org.opencv.objdetect.CascadeClassifier;  
 import org.opencv.objdetect.Objdetect;  
 import org.opencv.core.Mat;  
 import org.opencv.core.MatOfRect;  
 import org.opencv.core.Core;  
 import org.opencv.core.Rect;  
 import org.opencv.core.Scalar;  
 import org.opencv.core.Size;  
 import static org.opencv.imgcodecs.Imgcodecs.\*;  
public class FaceDetector{  
 public static void main(String[] args){  
 System.*loadLibrary*(Core.*NATIVE\_LIBRARY\_NAME*);  
 Mat image = *imread*("images/jeevan.jpg");  
 // create method for detect and save  
 *detectAndSave*(image);  
 }  
 private static void detectAndSave(Mat image) {  
 //create some objects  
 MatOfRect faces = new MatOfRect();  
 //convert to gray scale  
 Mat grayFrame = new Mat();  
 Imgproc.*cvtColor*(image, grayFrame, Imgproc.*COLOR\_BGR2GRAY*);  
 //improve contrast for better results  
 Imgproc.*equalizeHist*(grayFrame, grayFrame);  
 int height = grayFrame.height();  
 int absoluteFaceSize = 0;  
 if(Math.*round*(height \* 0.2f) > 0){  
 absoluteFaceSize = Math.*round*(height \* 0.2f);  
 }  
 //Detect faces  
 CascadeClassifier faceCascade = new CascadeClassifier();  
 //load trained data file  
 faceCascade.load("data/haarcascade\_frontalface\_alt2.xml");  
 faceCascade.detectMultiScale(grayFrame, faces, 1.1, 2, 0|Objdetect.*CASCADE\_SCALE\_IMAGE*,  
 new Size(absoluteFaceSize,absoluteFaceSize) , new Size());  
 //write to file  
 Rect[] faceArray = faces.toArray();  
 for(int i = 0; i< faceArray.length; i++){  
 //draw rect  
 Imgproc.*rectangle*(image, faceArray[i], new Scalar(0,0,255), 3);  
 }  
 Imgcodecs.*imwrite*("images/jeevanO.jpg", image);  
 System.*out*.println("write success" +faceArray.length);  
 }  
}

//Input:

//output:



//live face recognisation

package com.learn;  
  
import org.opencv.core.\*;  
import org.opencv.highgui.HighGui;  
import org.opencv.imgproc.Imgproc;  
import org.opencv.objdetect.CascadeClassifier;  
import org.opencv.objdetect.Objdetect;  
import org.opencv.videoio.VideoCapture;  
  
public class face2 {  
 public static void main(String[] args) {  
 System.*loadLibrary*(Core.*NATIVE\_LIBRARY\_NAME*);  
  
 // Open the default camera (index 0)  
 VideoCapture capture = new VideoCapture(0);  
  
 // Check if camera is opened successfully  
 if (!capture.isOpened()) {  
 System.*out*.println("Failed to open the camera.");  
 return;  
 }  
  
 Mat frame = new Mat();  
  
 while (true) {  
 // Read the current frame from the camera  
 capture.read(frame);  
  
 // Create a method for detecting and saving faces  
 *detectAndSave*(frame);  
  
 // Display the resulting frame  
 HighGui.*imshow*("Face Detection", frame);  
  
 // Check for 'Esc' key press to exit the loop  
 if (HighGui.*waitKey*(1) == 27)  
 break;  
 }  
  
 // Release the camera and destroy the window  
 capture.release();  
 HighGui.*destroyAllWindows*();  
 }  
  
 private static void detectAndSave(Mat image) {  
 // Create some objects  
 MatOfRect faces = new MatOfRect();  
  
 // Convert to grayscale  
 Mat grayFrame = new Mat();  
 Imgproc.*cvtColor*(image, grayFrame, Imgproc.*COLOR\_BGR2GRAY*);  
  
 // Improve contrast for better results  
 Imgproc.*equalizeHist*(grayFrame, grayFrame);  
  
 int height = grayFrame.height();  
 int absoluteFaceSize = 0;  
  
 if (Math.*round*(height \* 0.2f) > 0) {  
 absoluteFaceSize = Math.*round*(height \* 0.2f);  
 }  
  
 // Detect faces  
 CascadeClassifier faceCascade = new CascadeClassifier();  
 // Load trained data file  
 faceCascade.load("data/haarcascade\_frontalface\_alt2.xml");  
 faceCascade.detectMultiScale(  
 grayFrame, faces, 1.1, 2, 0 | Objdetect.*CASCADE\_SCALE\_IMAGE*,  
 new Size(absoluteFaceSize, absoluteFaceSize), new Size()  
 );  
  
 // Draw rectangles on the detected faces  
 Rect[] faceArray = faces.toArray();  
  
 for (Rect faceRect : faceArray) {  
 Imgproc.*rectangle*(image, faceRect, new Scalar(0, 0, 255), 3);  
 }  
  
 System.*out*.println("Detected faces: " + faceArray.length);  
 }  
}

//Output: