Technical Paper

Emotion Detection Using OpenCV for Facial Recognition

Yash Dedania, and Bhoomil Dayani
Department of Information Technology(IT)
CHARUSAT University
Changa,Gujarat,India
20IT023@charusat.edu.in
20IT022@charusat.edu.in

Abstract

Facial recognition is a very useful tool and has been researched extensively in recent years. The applications for facial recognition vary from use in security cameras to emotion detection. Emotion detection in particular is a facet of facial recognition that has great potential in a wide range of fields. In order to tailor the software for emotion detection, a series of steps must be taken. Starting with a database containing positive images (images portraying a specific emotion), a second database must be created that contains negative images (images without a face). Following this, a classifier is trained from the two databases. The classifier is then implemented into the software via a function. This process should ultimately allow for successful emotion detection that can be used in many different applications.

1. Introduction

Currently, there is a vast amount of research focusing on facial recognition and its capabilities. However, there is not as much focus on using facial recognition for emotion detection. I am proposing an application that is capable of determining human emotion, which could have many benefits including interrogations of criminals, as well as an interactive software that helps children with autism detect emotion.

This application will focus entirely on seven types of emotions like anger, fear, disgust, happiness, sadness, and `neutral.`

Facial representation is immensely important for the overall success of the program. Without a solid facial recognition system, the overall application can not function properly or as accurately. Due to the time constraint for this project, the decision was made to directly utilize open-source code from OpenCV to do the facial recognition aspect.

2. DETECTION PROCESS

First, the system must be able to identify whether or not a face is present in the image or video. This was accomplished using a cascade classifier already provided by OpenCV. This classifier works by first finding the face within the image (if one exists) and creating a rectangle around the person's face to show the user that it did locate one.

Once a potential face has been located, the system then has to be able to scan that area and compare it with a database to determine the emotion. After the facial emotion had been confirmed we can then start to look at facial features and use that information to determine the facial expression and emotion of that face. For this particular experiment , I chose to use a haar feature-based cascade classifiers for facial detection. This classifier was trained with a wide variety of positive and negative images and is now used to detect other faces.

2.1 Facial Representation

How the faces in a facial recognition system are represented is crucial to the accuracy of that system. The representations for the faces should be stored in a database. This database is then compared to the image that is scanned by the system to determine if the image is a face. Creating a data set is especially important when trying to determine facial expressions or emotion.

Their original data set included pictures of undergraduate portraying different emotions. However, there were differences with the faces portraying an emotion and the faces feigning that emotion. Those differences resulted in emotion detection being very inaccurate. To correct the issue a validated data set was created by trained actors who were able to accurately portray the emotions. Clearly, being consistent with your data sets is important.

2.2 Facial Detection

It is essentially the beginning of the first step in the process (besides preparing the data set). OpenCV, the software used for facial recognition in this project, uses classifiers to detect objects. These classifiers are trained by the programmer for whatever it is that he or she wants the recognition software to detect.

Figure-1:This shows how the classifier displays to the user that it found the found the face. This example also demonstrates the use of a nested-classifier that locates the eyes.

It was decided that OpenCV be used for facial recognition since the source code and classifier for that is already in place. Training classifiers and databases will be discussed in greater detail in the following section and again in the design section.

The texture modality is a collection of image patches highlighting facial key points. The landmark modality depicts the facial key points in a facial expression sequence. This multimodal learning algorithm is essentially an artificial neural network (ANN) where it takes in numbers and modality as input, stacking these inputs in a hidden layer, and outputting a result. Figure 1 below shows what this multimodal algorithm looks like.

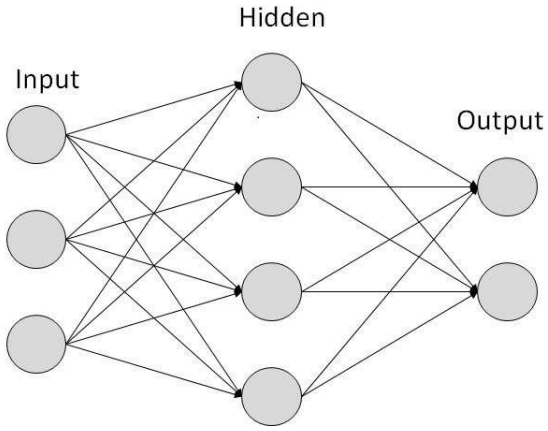


Figure - 2: Structure of multimodal algorithm

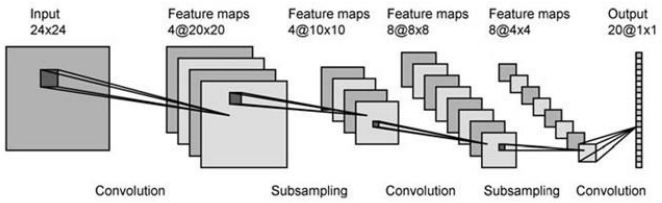
The main source for feature extraction and face detection used for this project is the cascade classifier. The way the classifier works is it first has to be trained. In order to train the classifier, the programmer must have a few hundred samples of the object he or she wishes to detect. These samples, also called positive images, are all different variations of the same thing. This allows for

more variety and provides more room for error. These positive images are then paired with negative images (images that are not wanted for detection). The reason behind the negative and positive images allows the classifier to be trained accurately and to not detect things other than what is desired.

2.4 Facial Points

The most common method for emotion detection is facial point detection. Facial point detection focuses on specific places on the face such as the corner of the eye, corners of the mouth, tip of the nose, beginning and end of the eyebrows, and there are many different models used in facial point detection. Facial point detection has been extensively studied in recent years. There are two general approaches to facial point detection. The first is to classify search windows and the second is to directly predict the positions of the key points. For classifying search windows, a component detector is trained for each facial point and the window location is based on a local regression. By directly predicting the positions of the facial points, scanning is eliminated and thus, this is more efficient. Sun et al. propose a cascaded regression for facial point detection that include three levels of convolutional networks.

Convolutional networks are neural networks (CNNs) that are similar to artificial neural networks (ANNs) in that they have trainable weights and these weights take in some input, as shown in Figure 2. The convolutional neural network, however, takes in the entire face as input and uses texture context information to extract features in deeper structures of the face. The first level of their network makes accurate predictions rather than making a rough estimate. The other two levels are for refining the earlier estimations of the facial points. These two levels are more shallow because they focus on smaller regions of the facial points whereas the first layer is scanning the entire face. At all three levels, multiple convolutional networks are combined to improve accuracy and reliability.



3. WHAT IS EMOTION?

Emotion is something that has been extensively studied and researched for many years up to now and has been found to be hard to classify. When most people think emotion, they think of the emotions themselves. In psychology there are six accepted archetypal emotions: happiness, anger, sadness, disgust, surprise, and fear. While these emotions are in fact true, not many people stop to think about what causes them or how we as people are able to recognize and determine these emotions. More often than not, we have the ability to look someone over and determine their emotion just from their facial expressions.

3.1 Emotion Detection

The key to accurately depicting emotion from images is to first extract the necessary facial features which are then either applied to different action units (AUs) or directly to the classifiers. Facial motion plays a key role in expressing certain emoitons. The muscles within the face are altered to intentionally portray certain emotions, which human beings are able to recognize, even if the expression is very subtle[7]. One method for emotion detection is to first find the geometrics of certain facial points. This allows for accurately determining the general shape and location of each facial feature. For this project though, by creating a database for each emotion separately, I was able to feed them into a command-line function that ultimately allowed me to train accurate classifiers.

These classifiers can then be called in a function within the source code provided by OpenCV to detect the emotion specified. For example in one test using the Happy.xml classifier I created, the software first scans the face for key points that are similar to that of the classifier and then displays a rectangle around the mouth indicating that it detected a smile (often associated with being happy). The following subsections break down each facial feature and use different methods and thresholding algorithms to determine the location of each facial point for that feature.

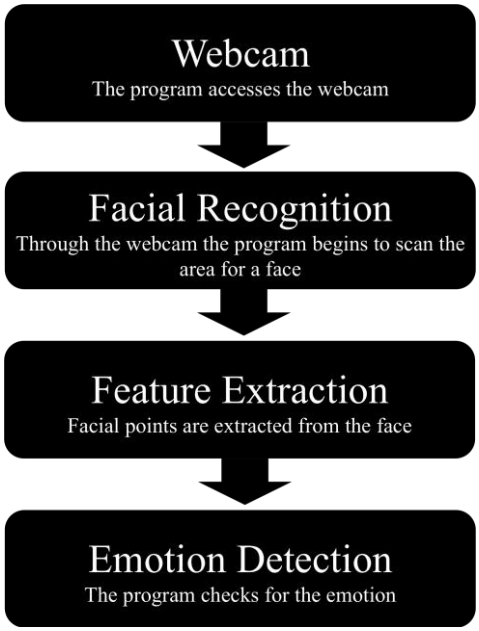
4. DESIGN

For this project, the structure and design were relatively straightforward and simple. The first step was to create a database for each emotion that I wanted the software to detect. To accomplish this task I did some research and found an excellent starting database of images. This database contained forty subjects who were each photographed ten times and in each of their photographs they were portraying a different emotion. I determined that the best way to go about creating my own database for each emotion was to go through all of the subjects and label the emotions they were portraying. Once all the emotions were classified, the paths to each image for a given emotion was stored in a file.

The happy database currently has the most samples in it at approximately 155 images thus making it the most accurate of the four classifiers that I have created.

From that point, the only thing left to do was to train the classifiers for the emotions. This was accomplished by calling ”opencv-createsamples” from the command line. This function takes the

happy.info file that I created and turns it into happy.vec which has now resized all of the images in the happy.info file. The last step was to train the classifier. As previously mentioned, the classifier is trained by being fed positive and negative images as input with the positive images in this case being the happy subjects. This was done my calling ”opencv-traincascade” from the command line. This function takes the happy.vec file and the file I created containing the negative images, and trains the classifier to detect all of the features associated with the happy subjects. By giving the taking both the positive database and the negative database as input, the function is able to train the classifier to accurately distinguish what the user wants to be detected, and what the user does not want to be detected.



In terms of the actual experimentation, the program first starts off by first accessing the laptop camera. The program then accesses the first cascade classifier that is called within it. This initial classifier is the facial detection classifier already provided by OpenCV. When it is called, it begins scanning the image for a face and when it is found a blue rectangle is created around the person’s face and the second classifier is called. In this case the nested classifier is the seven emotion classifier that I created. This classifier is performing the exact same way as the face detect classifier except for the fact that it is scanning for facial points and features that it associated with the seven emotions subjects in the database. When it finds the features and points, blue circles are generated around them. This is just one way in which emotion detection can be achieved as there are many other methods out there as well.

5. RELATED WORK

5.1 Eyes

The eyes are an extremely important facial feature as they help align the face and allow for a good reference point to other facial features. To detect the eyes one method was to use a thresholding algorithm that takes in input from every pixel from the eye region[9]. This algorithm is represented as

$$T_{local}(x,y) = \mu_{global}(x,y) + k * \sigma_{local}(x,y)$$

$$\mu_{global}(x,y) = (1/M * N)[\sum_{N}^{P} \sum_{M}^{P} f(i,j)]$$

where $T_{local}(x,y)$ is the threshold value of the pixel at (x,y) and $\sigma_{local}(x,y)$ is the standard deviation. μ_{local} is the local mean and μ_{global} is the global mean and $M * N$ is the size of the eye.

5.2 Eyebrows

The eyebrows are just as important if not more important in determining emotion as the eyes are. This is because the eyebrows are a key point in facial expressions. If the eyebrows are raised, the individual is likely to be surprised or happy as opposed to when the eyebrows are sort of squinted which is often associated with concern or anger. The location of the eyebrow is first found using facial geometry. Once located, the facial points are stored in a vector. The positioning of these facial points will aid in determining the emotion.

5.3 Nose

As the nose lies between the eyes and represents the approximate center of the face, the rough estimation of the nose should be simple. The thresholding algorithm from earlier can now be applied to the nose as well as a contouring method to find the two nostrils.

5.4 Lips

Majumder et al. provide a step-by-step algorithm for estimating the location of the lips. The first step is to find the center of the eyes and the height of the nose[9]. From there they form a rectangle around the projected region of the mouth by using the function $rect(x_l,y_l,h_l,w_l)$ where x_l and y_l are the coordinates of the top left corner, h_l is the height and w_l is the width.

6. RESULTS

Current results show that the program is not much accurately able to detect when the subject in the video is smiling. It draws a rectangle (exactly like the rectangle around the face and eyes in Figure 1) around the mouth when the person smiles. This shows that the classifier was trained fairly well, but some issues such as poor lighting and tilted or turned heads have given the program trouble. The program had a particularly difficult time staying consistent when demonstrated in a very bright room. It also proved to be a little inconsistent when the user turned their had too far to one side or if they had their had tilted too high or too low. Despite these minor issues though it is great to see the program being able to detect smiles and other facial points related to happiness.