Emotion Detection Using OpenCV for Automatic Facial Recognition

Jonathan Hicks
Earlham College Department of Computer Science
801 National Road West
Richmond, Indiana 47374
jghicks13@earlham.edu

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 emotion detection and will be an extension of current research. In the design of this application, a database is created for each emotion where genuine and feigned emotions are include to allow the application to clearly determine whether or not the emotion of the scanned face is genuine or false[1]. Another database is also created containing negative images, which will be explained in greater detail in section 5.Throughout the creation of these databases, an eye must be kept out for certain potential problems like an obscured face, bad lighting, or non-frontal face[2]. As people, we are able to overlook these problems but it is significantly more difficult for a machine. Once the problems have been accounted for and the databases are ready to go, actual implementation can get underway.

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 research, the decision was made to directly utilize open source code from OpenCV to do the facial recognition aspect. Choosing this path allows for more time for research and focus on the emotion portion of the project. This paper has been broken down into sections based on separate essential components.

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 creates a circle around the person's face to show the user that it did in fact 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 if there is in fact a face there. After the face 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. For example, Curtis Padgett and Garrison Cottrell from the Department of Computer Science at the University of California conducted and experiment to identify human emotion with facial recognition software[1]. 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

Determining whether or not a face is present is perhaps the most important step in the facial recognition process. Mainly because 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.

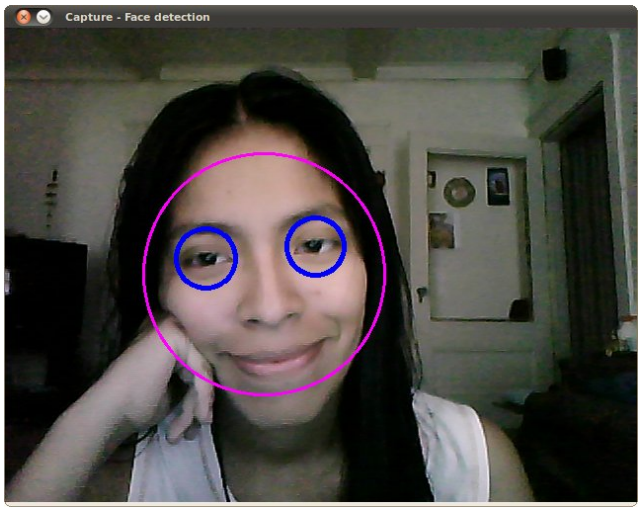

Figure 1: This shows how the classifier displays to the user that it found the found the face. This example also demonstrates the use of a nested-classifier that locates the eyes. [3]

Because this project focuses primarily on facial expressions and emotions, 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. Once the data set is ready, some common problems have to be taken into consideration[2].

One common problem is a partially blocked face. When someone sees a picture of someone and that person's face is partially blocked, that person is able to fill in the missing pieces. Not all systems are capable of filling in those pieces and thus results in poor object detection, or no detection at all. As it stands right now, if the face is blocked in the tiniest bit, the program is not able to detect it. Same goes for if the individual in the current image has their face turned too far to the side or tilted too far back. Another common issue is lighting. Again, this is not as big of an issue for people as it is for machines. When the lighting is too bright, people can squint and still make out that a face is present. This is much more difficult for machines and therefore has to be taken into consideration when creating the positive image database for each emotion. Extreme lighting whether it is natural light or not currently causes the classifier to be less accurate. It is still able to detect a face but the emotion classifier is significantly less accurate. If images in the database contain obscured faces or are lit poorly to where, when creating the classifier, the classifier is unable to extract the necessary facial points from that image. All of these issues could lead to inaccurate results.

2.3 Feature Extraction

There have been many different ways for extracting facial features. Zhang et al. used a multimodal learning technique that focused on learning the representations of texture and landmark modality[4]. 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.

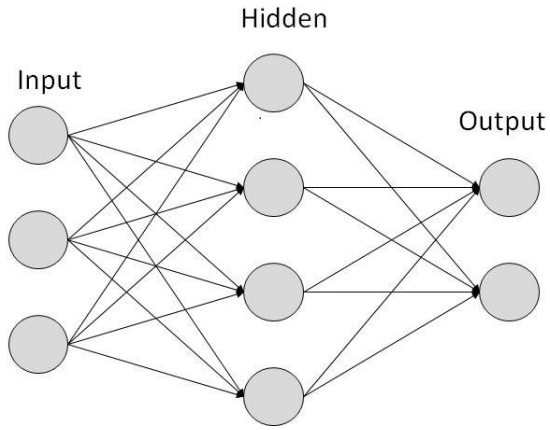

Figure 2: Structure of multimodal algorithm.[5]

As mentioned earlier, 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[6]. 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 predict-

ing 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[6].

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.

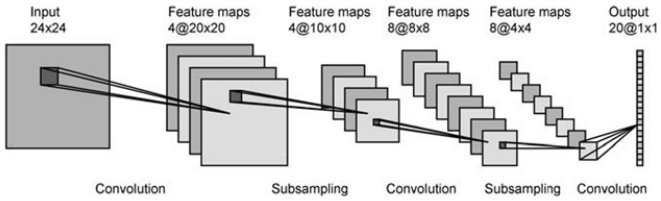

Figure 3: Example of a convolutional network.[4]

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[7]. 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 circle 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. For example,the path to each of the happy images was stored in happy.info which can be seen down below.

s3/1.pgm 1 0 0 92 112
s3/2.pgm 1 0 0 92 112
s3/3.pgm 1 0 0 92 112
s3/4.pgm 1 0 0 92 112
s3/5.pgm 1 0 0 92 112
s3/6.pgm 1 0 0 92 112
s3/7.pgm 1 0 0 92 112
s3/8.pgm 1 0 0 92 112
s3/9.pgm 1 0 0 92 112
s3/10.pgm 1 0 0 92 112
s5/1.pgm 1 0 0 92 112
s5/2.pgm 1 0 0 92 112
s5/3.pgm 1 0 0 92 112

In the path the s represents the subject. For example, s5 is subject five and the 1 0 0 92 112 part of each line simply says that there is one object in the image that is desired with a width of 92 and a height of 112. 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.

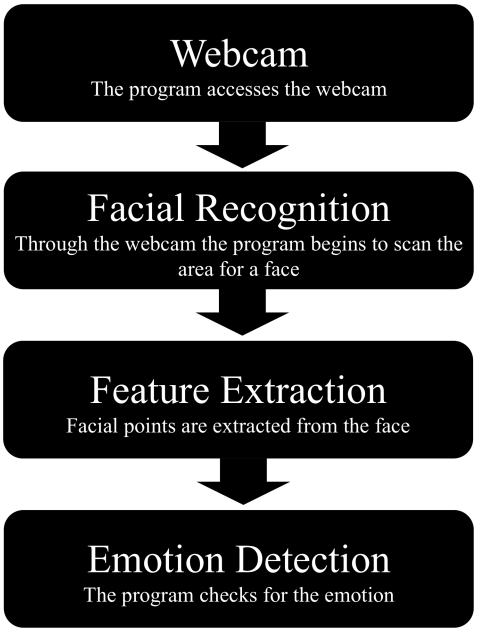

Figure 4: Program Design

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 circle is created around the person's face and the second classifier is called. In this case the nested classifier is the happy 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 happy 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

Over recent years there has been a fair amount of research put into facial feature extraction and feature classification (emotion detection). For example, Zhang et al. taking a multimodal approach for facial expression recognition [4]. In their research they focused primarily on two types of modality, texture and landmark-based. From there, a support vector machine (SVM) is then trained for feature classification; which is an extremely popular choice for classifying features.

SVMs are supervised learning modules (they are given trained data) that contain algorithms used for classification and regression analysis which can then be used to create a hyperplane. An SVM was considered for this project for facial expression recognition, but the general time frame for training the SVM would have taken too long a semester-long project. A simplified version of the hyperplane can be seen below.

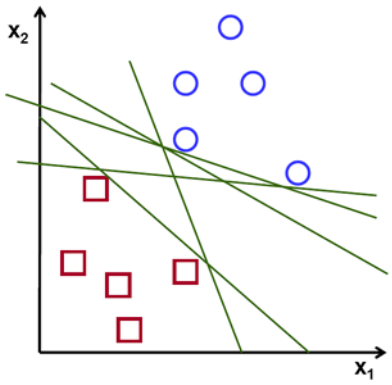

Figure 5: Cartesian Plane with lines and points.[3]

In Figure 3 one can see that there are multiple lines and multiple points. The lines represent multiple solutions to whatever the problem is. In order to determine the best possible solution, you want to choose the line that is the farthest away from all points. The reasoning is that the if a line is too close to a given point, it will not generalize correctly.

Another approach to feature extraction is facial point detection. For this approach, many techniques have been used [6]. A great deal of techniques center around updating facial points iteratively or shape constraints. These techniques can be weak and often times require excellent initializations along with problems such as: extreme facial poses, obscured faces, poor lighting, and the lack of reliability from extracted facial points [6]. Cascaded regression has been proposed as a remedy to some of these issues.

One regression-based approach by Valstar et al. used local patches with random forests to accurately predict facial points. They proposed a method for finding 22 facial points in order to gather the necessary information for automatic facial expression recognition[8]. These facial points include: the corners of the eyes, the corners of the mouth, upper eyelids, and the nostrils. An example can be seen below in figure 4 and the algorithms used for different facial features is explained in the following subsections.

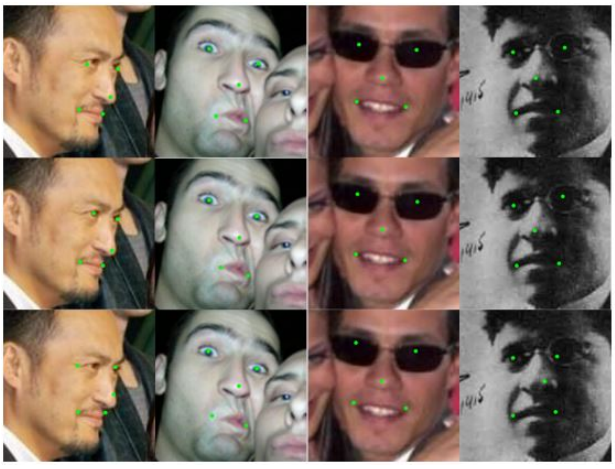

Figure 6: Example of desired facial point detection[6].

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

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

$\mu_{global}(x, y) = (1/M * N)[\sum_{j=0}^{N} \sum_{i=0}^{M} f(i, j)]$

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 $rect(x_I, y_I, h_I, w_I)$ where x_I and y_I are the coordinates of the top left corner, h_I is the height and w_I is the width.

6. RESULTS

Current results show that the program is accurately able to detect when the subject in the video is smiling. It draws a circle (exactly like the circles 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.

7. FUTURE WORK

Going forward more research and more information about facial recognition and emotion detection will be looked in to. Now that the facial recognition software and base classifier has been successfully implemented, the next step is to get the program to be able to detect even more emotions. Currently there exists four total databases: happy, sad, surprised, and angry. However, due to an insufficient amount of images for the other three emotions, the classifier is not completely accurate and does detect those emotions well. Getting those classifiers going well should not be too daunting of a tasks sense that is simply a matter of finding even more reliable images. Ideally each database will have approximately 300 images. The happy database is at 155 as of right now and seeing as how it already works pretty well, doubling the size of the database should increase the accuracy even more.

Increasing the size of each database is not the only project going forward. The next big step is for the program to display via text what the emotion is instead of just generating a circle around the distinguishing trait of a given emotion. This will be more user friendly and will alleviate any uncertainty as to what the circles represent. Another future task is to change the way the classifier is called within the program. As is stands at the moment, the programmer has to go in and change the classifier based on what emotion is desired. The software would ideally be able to scan the image, find the face, and determine what the emotion is without having to pre-select the classifier.

When it is all said and done, the ultimate goal of this project is to be an Android application. An estimated time frame for accomplishing this is going to require about 5-6 additional months of researching and learning Android development. Upon more thought, another direction that this project could be taken is, instead of turning the program into a mobile application, it may make more sense to apply it to something along the lines of the Google Glass idea. This would be a much more challenging task but it could certainly be worth it as the children would be able to just put on these glasses and carry on with their everyday lives. They would be able to carry conversations with people and the program would be displaying that person's emotion live as opposed to the user having to look at images or videos on their phone. The future goals of this project and research aim to make a difference in the world.

8. REFERENCES

[1] C. Padgett and G. Cottrell, "Representing face images for emotion classification," pp. 895 – 900.

[2] A. Samal and P. A. Iyengar, "Automatic recognition and analysis of human faces and facial expressions: a survey," *Pattern Recognition*, vol. 25, no. 1, pp. 65 – 77, 1992.

[3] "Cascade classifier," *Cascade Classifier OpenCV 2.4.13.1 documentation*.

[4] W. Zhang, Y. Zhang, L. Ma, J. Guan, and S. Gong, "Multimodal learning for facial expression recognition," *Pattern Recognition*, vol. 48, no. 10, pp. 3191 – 3202, 2015.

[5] tutorialspoint.com, "Artificial intelligence neural networks."

[6] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[7] C. Busso, Z. Deng, S. Yildirim, M. Bulut, C. M. Lee, A. Kazemzadeh, S. Lee, U. Neumann, and S. Narayanan, "Analysis of emotion recognition using facial expressions, speech and multimodal information," *Proceedings of the 6th international conference on Multimodal interfaces - ICMI '04*, 2004.

[8] M. Valstar, B. Martinez, X. Binefa, and M. Pantic, "Facial point detection using boosted regression and graph models," *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

[9] A. Majumder, L. Behera, and V. K. Subramanian, "Emotion recognition from geometric facial features using self-organizing map," *Pattern Recognition*, vol. 47, no. 3, 2014.