EMOTION CLASSIFICATION IN HUMAN FACES

Computer Science 490: Senior Project

Chris Riederer

Adviser: Drew McDermott

Yale University, Spring 2010

Abstract:

In this project, the researcher attempts to construct a system that will be able to categorize images of faces based on the face’s emotional state. In particular, an effort was made to classify a face as happy or unhappy.

The motivations for such a project are many. A system that can successfully classify faces based on emotional content could be used for a variety of tasks. For example, a program might be used to improve Human Computer Interaction, to label images, to serve as a psychological diagnostic tool, or even to improve a computer’s ability to play poker. Furthermore, there is currently a rift in the psychology community over whether or not “universal expressions” exist. If a system could be designed that could classify faces under these expressions, it could help to support a particular theory.

The researcher decided to use images of faces for a variety of reasons. As mentioned previously, various psychology papers have reported “universal expressions” that people recognize across different cultures. The existence of expressions that any person can recognize seems to support an argument that a computer could do the same. Furthermore, focusing on one domain of emotion communication allowed for a simpler model, and one that perhaps could be applied more robustly in uncertain environments.

To accomplish this goal, a set of training images from the Yale Face Database were used. This Database consists of test subjects photographed with different expressions and in varying lighting conditions. A series of “feature detectors”, designed to quantify specific characteristic of the face, were developed. These feature detectors were run on several different images for each test subject and then averaged, giving an average vector for each subject. The feature detectors were then run on the happy faces, giving a happy vector. Using both the neutral and happy vectors as inputs, a neural network was trained to try and differentiate between the two for each face.

Although many of the feature detectors were accurate and effective, the resulting network was unfortunately rather inaccurate. A discussion of reasons for this and future improvements are provided.
Motivation

The concept of software that can recognize emotion is certainly not a new one. However, due to recent changes in technology and its use, the possibilities for such a system have grown and shifted.

As computing becomes more ubiquitous, with operating systems running on phones and cameras, the way humans interact with computers will necessarily change. A keyboard and mouse is no longer a viable option in many scenarios, and more natural interfaces will need to be developed. Programs that can ascertain the emotional state of the user will be certainly be useful on devices of the future. Additionally, smaller, cheaper hardware components such as video cameras will allow for computing to become a more integral part of the workplace. For example, systems have already been developed and installed in semi-trailer trucks to detect and awaken sleeping drivers (Mohanty).

The applications of an emotion-recognition system do not necessarily have to be commercial. In the field of psychology, many experimental results require knowledge of the emotional state of the test subject. An automated system for emotion detection could help remove any bias or sampling error, whether it is on the part of the subject or experimenter, or created through poor experimental design. Beyond experimentation, diagnosis is an area where noisy information can lead to a high degree of uncertainty. The field of psychology is also currently undecided as to the status of “universal faces”. One camp of psychologists cites research that claims that certain configurations of the face are recognized as the same emotions in different cultures across the world (Ekman). Researcher James A. Russell and others say that such studies are flawed, and that culture is much more influential in mapping an expression to an emotion. A computational system that could recognize emotions in people of varying cultures would lend credence to the former hypothesis.

Originally, we were interested in developing a system that could detect deception through image processing. Such a system could be useful in a variety of fields. For example, it might classify smiles as real (known in psychology literature as a “Duchenne” smile) or faked. The program could result in a fast and non-intrusive polygraph, an enhanced poker-playing computer, or even detect high levels of anxiety or nervousness at airport security checkpoints.

With so many important potential applications, the world is certainly ready for a system that can at the very least begin to guess at what emotions a human may be feeling.

Background:

Human beings display their emotions in multiple ways and through specific behaviors. Examples abound: speech changes in pitch or speed, gestures becomes more exaggerated, body temperatures rise, can shift, and the heartbeat’s pace fluctuates. Because many of these behaviors are well-defined and easily detectable, it seems that a computational system of emotion detection should be possible.

Since there are multiple physiological characteristics that give information about someone’s emotions, why not use more than one? There are actually a variety of reasons.
First, collecting data from more than one behavior would require a different method of processing information for each behavior, adding to the complexity and size of any such program and possibly requiring more hardware components. Furthermore, some integration between the multiple sources of data would have to be supported, requiring a higher-level theory. Finally, many of the physiological tests necessary to detect might be invasive, painful, or cumbersome. A survey of the field of emotion recognition in 2002 suggested that these and other difficulties have caused many previous researchers to focus on using just one physiological indicator of emotion (Pantic). Thus, a system that uses only one behavior to recognize emotions might be faster, simpler, less obtrusive, and cheaper, as well as supported by a larger amount of scientific literature.

The face seems a particularly rich source of information about emotions, and as such it would be suitable to base our emotion classifying program around it. Evidence from psychology experiments seem to suggest that the face dominates a human’s estimation of another human’s emotion. Two widely cited papers by Mehrabian state that 55% of whether a subject is liked on first impression is based on facial expression. Psychology researcher Paul Ekman has found evidence to support that cross-cultural emotional classification can occur based solely on images of faces. Because of these findings, the face was used as our physiological marker of emotion.

An overall emotion classification system was too ambitious a goal for the purposes of this research. Instead, a program to detect just one particular emotion was developed. The emotion of “happiness” was selected for a variety of reasons. Happiness is one the six universal emotions discussed above. It typically has easily detectable physiological signs in the face, such as a smile and smaller eyes. Previous work has been done on smile detection which would be useful for this case. Finally, as the researcher working with a large number of images, I preferred to look at smiling faces than angry or disgusted ones. The overall ambition, however, was to create a system that could be easily applied to other emotions in one particular subject.

To attack the problem of happiness detection using images of faces, I decided to follow an approach backed up by previous papers (Pantic). The overall problem of emotion classification has many difficult components, such as finding a face in an image, finding various parts of the face for analysis, getting meaningful data from any or all of those parts, and using that data to come to a result about the face. Like many researchers before me, I decided to assume that the problem of finding a face had already been solved. Thus, a database of face images was used.

An attempt was made to pull information from specific features of each face, features such as mouth curvature or face width. These features will certainly change as a subject moves from one emotional state to another. These features could then serve as training data for an emotional classification system.

Many of these features were found based on regularities in the dimension of faces and the use of edge detectors. In particular, the gradients of pixel intensities, Canny edge detection,
and Sobel edge detection were utilized. These are fairly common techniques and many sources of information are readily available to the interested reader.

Artificial Neural Networks are used often in tasks of classification. Furthermore, Yale’s resources provided an easy to use neural network toolbox. For these reasons, the decision was made to use a neural network as means of classification. An attempt was made to give information to the network about both what the normal face looked like and what a smiling face looked like. Humans are able to recognize features instantly from a face and imagine what the face would look like in a different mood. Additionally, humans are typically able to see someone’s face move from a neutral to smiling pose. Thus, the neural network should be given information relating to both the original face and the smiling face. Therefore, an effort was made to combine the information derived from the face of a subject, both in a smiling and non-smiling pose.

Methods:
The overall task of training the network has a variety of steps. First, the decision was made to use Matlab and face databases. After this, each image had to be processed to find just the face. This is discussed in the section Preprocessing. Beyond this, Feature Extraction is discussed in its own section, showing how particular features were quantified. Finally, training and use of the artificial neural network is discussed.

Use of Matlab

Matlab is a computing environment and programming language used for many technical tasks. It has many large libraries of functions relevant to the task of image processing, as well as an active online support community. For these reasons, and because of its availability on many Yale computers, the project was accomplished in this language.

Use of Face Databases

As previously mentioned, the overall task of recognizing emotion has many subproblems. Developing my own system to correctly identify and rotate faces was beyond the scope of this paper, and I instead chose to use a database of images of faces. For my project I needed a database of not only faces, but the same face in multiple emotional poses. I found two open and free databases that fit these needs, the Yale Face Database and JAFFE.

The Yale Face Database, available online http://cvc.yale.edu/projects/yalefaces/yalefaces.html, provided a free, easy to use source of standardized testing data. The database has 15 subjects in a variety of poses and lighting conditions. Each subject in the database had multiple emotional expressions, allowing for the option to test varying emotions or classify in different ways.

The Japanese Female Facial Expression Database, or JAFFE, is a similar database consisting of 10 subjects with 7 poses. Each pose is one of the universally recognized emotions or a “neutral” pose. There are multiple versions of each pose for each subject. JAFFE also has a website, whose URL is http://www.kasrl.org/jaffe.html.
Both databases were created in the late 1990s, and hence have lower image quality than what some may use today.

**Preprocessing**

Although the images are fairly standard in both databases, they were not of just the faces but rather the faces and some amount of background. It was necessary to cut out extraneous features of the faces, such as hair, neckline, or background, before proceeding. This process is described briefly below.

As some of the images sometimes contained a black edge of pixels, the outermost layer of pixels was removed. Some of the features later on utilized a threshold based on the pixel intensities of the overall image. This threshold was then computed and saved, so that it could be utilized with specific smaller areas of the image. The next task was to find just the face. We gradually narrowed in on this task in a few steps.

First, after converting the image to binary, the largest region passing a certain threshold of darkness corresponded to the face, as the background was white. Getting the coordinates of the bounding box of this region, we were then able to find a smaller region in the original image that contained the face.

A further narrowing of the face was attempted, based on algorithms from two papers (Peng, Ryu). The first involved calculating the columns with the most readily apparent vertical edges, using a Sobel edge detector. Sobel edge detection allows for detection of edges in a certain orientation. This proved to not always work completely, as the edge of the head, as opposed to the edge of the face, was often found. A more effective, but still less than perfect, method was to calculate the gradient image of the picture. The columns with the largest intensity tended to correlate more strongly with the edges of the face in this case.

Segmentation was often made more effective by running one of the previous methods twice. The first time would often remove the background up to the edge of the head, and the second time would remove the image up to the face. The size of this resulting image provided an estimation for the size of the face.

**Feature Extraction**

After appropriately preprocessing the image, we were now ready to find features of the face. A feature is any mapping from an image to some number, represented by a MATLAB float type. The motivation behind this was to develop a method of converting an observation about a face into a data type that could be used in a standard neural network.

Features were chosen based on a combination of background research, practicality, and intuition. A paper by Choi, Peaker, and Wong provided many helpful guidelines of features to select, some of which I used. Since all images in the Yale Face Database and JAFFE are grayscale, I couldn’t use any color information. Furthermore, humans are able to analyze the emotions of grayscale images. Some of the features I utilized looked at specific areas of the face, whereas others made calculations across the entire face. As humans aren’t necessarily
the best at picking out which component of a vector is the most important in a data set, I included features that didn’t necessarily appeal to intuition but might be useful none-the-less.

Peng et al provided a useful method of roughly finding the eyes. Simply calculating the row with the highest sum in the gradient image provided a useful metric for where the eyes would be. After finding this row, a number of rows above and below were taken based on the estimated size of the face, which was in turn based on the size of the image after preprocessing.

From this eye row, the center of the face could be surmised, by our usual method of summing values for each column or row. In the swath of the image containing the eyes, the area of largest white space is often the one in the very middle of the face. Thus, finding the sums of the intensity of each column, the center of the face was detected.

An attempt was made to find the exact location of the eyes through template matching. Using templates from the Peng et al. paper, an eye image was fitted over all locations in the estimated area of the eye row. Error was calculated based on difference of pixel intensities squared. I attempted altering the error function, the size of the template, and the actual template used. Although this method has proven effective in various papers, I did not find it accurate enough to use.

After finding the eyes and the center of the face, it was easy to find the forehead. I simply took a piece of the image centered on the center of the face that was above the eyes. The forehead can be useful in distinguishing fear from happiness, as wrinkles tend to appear in distress in this area. A measurement of the amount of edge pixels can then be taken to see how many wrinkles might be in the area.

To find the mouth of a face, I first attempted to repeat the process used above based on the gradient. Achieving unsatisfactory results, eventually I settled on a method from Ryu and Oh. A Sobel edge detector was first applied to the face, with the orientation set to horizontal. The resulting image is binary, with horizontal edges set to 1 (white) and all other points set to 0 (black). For the lower half of the image, the row with the most pixels set to 1 is most likely the upper or lower lip of the mouth. Based on the size of the image, we can then select an area around this that will include the entire mouth.

Once we have found the mouth, we can find the curvature. Applying the Canny detector, which tends to have less noise than other edge detection algorithms, we obtained another binary image of just the mouth. To get rid of noisy points, we applied an alteration to the threshold used in the Canny detector. Canny uses two thresholds. The high threshold is what a pixel value needs to be above for it to be considering a starting edge. The low threshold is then used to continue these edges. After getting too many noisy points using the automatic threshold calculated by Matlab, the high threshold was raised and the low threshold lowered. Additionally, search was focused around the center rows and center columns to find the mouth. This was to avoid noisy pixels that may have ended up in our image of the mouth (bits of the noise, edge of the face, goatee, etc. It was then easy to find the left, right, and bottom points.
From these three points we could do an approximation of the angle of the mouth, giving us a feature. A measurement for the curvature of the top lip and bottom lip were taken.

Other features such as the overall “verticality” or “horizontality” of certain parts of the image were used. To get these values, a Sobel edge detector with orientation vertical or horizontal was applied to the face. As mentioned before, this set the value of coordinates whose pixels correspond to edges equal to 1, and 0 otherwise. The percentage of 1s in the resulting binary matrix then served as our feature’s value. The intuition behind this is that someone with a change in the number of edge pixels in their face is more likely to have part of their face become wrinkled. Wrinkling might correspond to facial deformations due to smiling, yawning, etc. Verticality and horizontality was an easy measurement to apply to each component of the face, and so I included these measurements for the mouth and eyes as well.

Training

The two databases were used to train and test separate networks, due to differences in how the feature detectors worked on each. In this section, the network produced by the JAFFE database is discussed. A near identical method was used on the Yale Face Database.

For each subject of the database, a set of vectors was obtained by running feature detection on their faces. Each subject in the JAFFE database did the same emotional pose three times. The results of each emotional pose were averaged. This was completed for multiple emotions. The training data was then the concatenation of the vector for the neutral pose and the vector for another pose. This was to retain the information of what the normal face looked like. Negative training data consisted of the neutral pose and a non-happy pose, whereas positive data was of the neutral pose and a happy pose. After training, the networks were utilized on random faces from the database, always a neutral face and a different emotional state, be it happy or something else.

Results:

Unfortunately, the project proved ambitious in scope and was not successful in being able to differentiate different emotional states with a high degree of accuracy. Despite this, many valuable feature-detectors were developed. Hopefully, this code can be utilized in the future, as I plan to make it available open source on Matlab’s Mathworks website.

A page of images from the feature detectors is attached.

Conclusion:

Although the original task was not accomplished, there are a variety of possible conclusions that can be drawn from this paper. That the model failed suggests that at least one of its components was faulty. Thus, we can examine each to see the way it possibly could have gone wrong, and what each failure suggests.

Feature detection was an immensely complicated and difficult task. During research, a variety of papers were discovered that suggested straightforward methods that worked with high
accuracy. As might be expected, these methods were really only accurate on their own particular set of training data. The feature detectors used never worked with as high accuracy as in their original papers, and sometimes failed completely. As the researcher, I often found myself making small tweaks to try to get the feature detectors working for every face. This could suggest that image processing literature might be better served by testing their programs on different image databases than those they originally wrote their papers using. This could provide an insight into how accurate their methods might be across all faces.

Although feature-based processing seems intuitively to be a fine notion, in practice the time and difficulty in finding features seems to be a sizable setback. Mathematical methods such as eigenfaces and PCA have no bias from the researcher’s intuition, and may be a more reliable method in the future.
Images

Subject 14

Forhead and Eyes, Center Column Shown

Mouth

Detection of top of mouth, angle
References


Resources

Japanese Female Facial Expression Database (JAFFE)  
http://www.kasrl.org/jaffe.html

Yale Face Database  
http://cvc.yale.edu/projects/yalefaces/yalefaces.html