Component Based SVM Facial Recognition

Alex Bullard, William Martin, and Joe Redmon

May 9, 2010

1 Introduction

Facial recognition is a popular field of image analysis that has seen a lot of attention in recent years due to its many biometric and commercial applications. Uses range from security and identification for government bodies to human computer interaction for consumer purposes.[5] In particular, face recognition offers a potential breakthrough in access control and enforcing security restrictions. For this type of application, the user base is quite small and the captured images have constrained illumination and angle. Thus, access control facial recognition has seen high commercial success along with competing technologies, such as retina or fingerprint recognition. Past examples include IBM's FaceIt [8] controlled screen saver, the FaceVACS-Entry [1] security system that combines with a card terminal, and the FaceKey [3] biometric control system that incorporates fingerprint recognition. Given this strong precedence of facial recognition applications, we hope to present our own realization of a facial recognition program.

There are a number of different approaches to performing face recognition, which have varying levels of success. Some of the better-known algorithms utilize eigenfaces [9] or active appearance models [2] to identity an image. However, eigenface approaches suffer from requiring extremely constrained frontal face images and potentially large amounts of training data to deal with high variability. Active appearance models are more promising for a noisy environment but require computationally expensive models. Our attempt at facial recognition follows the popular use of machine learning to determine the differences between images. By utilizing support vector machines to create models we are able to create a complicated description of what features determine an individual.

2 Support Vector Machines

Support vector machines (SVMs) give us a supervised learning method for classifying both facial features and individuals based upon these. After a decent amount of training, an SVM can predict whether an input falls into one of two categories. This is done by first constructing a hyperplane in a high dimensional space and then mapping input to points in this space. The input is determined to be in one category or the other by measuring its distance from the hyperplane. The strength of the category correspondence is then given by the magnitude of this distance. SVMs have the advantage of forming a convex function so there are no local minima that could be potential pitfalls. In addition, there are fewer parameters to tune when using SVMs in comparison to other AI techniques.

3 Algorithm Implementation

Our algorithm was based heavily upon two papers that describe a facial recognition approach and a method to describe image features. The first paper by Heisele, Serre, and Poggio [4] presents a component-based procedure for facial recognition using SVMs to learn the images. Our realization of this algorithm is a slightly simplified approach that still adheres to the overall structure of the process. However, Heisele's paper does not describe in the detail the basic feature vector used to describe a block of pixels so we adopted an approach given by Jiang [6]. We also used Thorsten Joachims' wonderful implementation of a classifier SVM (*SVM light* [7]) rather than creating our own.

Our implementation in general goes as follows:

1. Capture a grayscale frontal image of a face and scale it down to 100 pixels in width while maintaining its aspect ratio.

2. Incrementally run a 30 pixel wide square window over the image and record a feature vector describing each block of pixels. The feature vector is created by applying a histogram of gradients (HoG) approach to all four quadrants of the image. Each histogram has nine bins that represent different ranges of gradient directions. Every pixel is allowed to place a vote weighted by its gradient magnitude into the appropriate bin. After normalizing the histograms, we place the 36 resulting frequencies into a feature vector.

3. Run the list of feature vectors through five different trained linear feature classifier SVMs (right eye, left eye, nose, right mouth, left mouth).

4. Take the maximum classifier output for each feature within its expected area of the image. The corresponding window will then be used to describe the appropriate feature in the image.

5. Place the five values found in step four into another feature vector and run it through a trained polynomial person classifier SVM for each person in the database.

6. The person classifier SVM that gave the highest output is then considered to be the identified individual.

In addition to this algorithm, we also created a graphical user interface (Fig. 1) in order to automate the process for a user. It was implemented using the Objective-C Cocoa Framework in order to utilize the iSight API for Apple webcams. The interface allows one to capture a cropped photo, reveal where features were detected and finally identify the individual as one of the trained people.

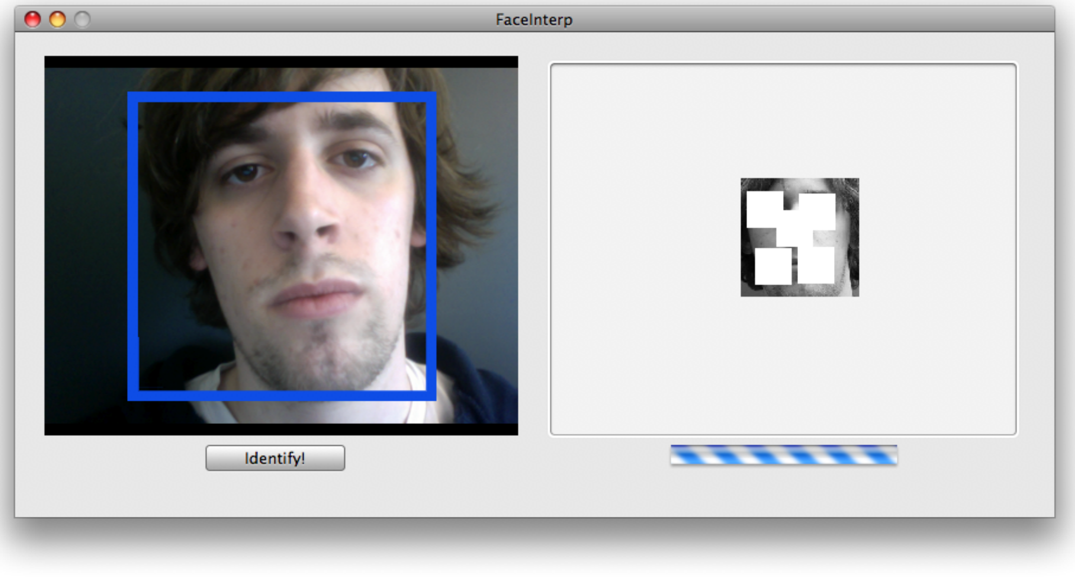

Figure 1: Screenshot of our interface for facial recognition.

4 Data collection

Several papers on facial recognition gathered data by first constructing 3D models of individuals and then capturing perfect training images from numerous different angles and illuminations. This resulted in thousands of both positive and negative training examples from which a model could learn. However, we did not have the luxury of rendering 3D models. Rather, we took a more basic approach and captured approximately one minute of video of each participant while they made slight facial expressions. Since the head did not move throughout the video we were able to extract thousands of images of faces. We first used the open source program FFmpeg to extract 25 frames per second from the captured video and then used the UNIX convert utility to create batch cropped images of individual features (Fig. 2). These feature images were used as positive training examples while negative examples were gathered from various other parts of the face that did not describe a particular feature. We should note that while the feature SVM trained within a matter of seconds, the person classifier SVM took several hours to fully train due to the massive amount of pictures.

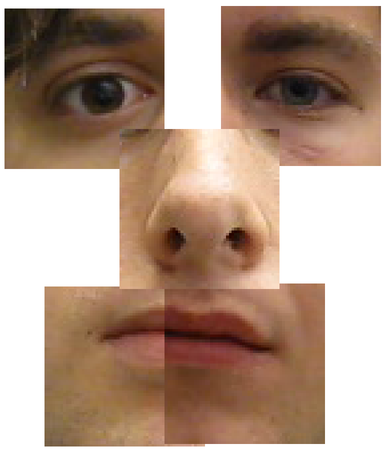

Figure 2: Examples of the images used to train individual features.

5 Results

We tested our program with a series of frontal face images of the three trained individuals. Out of 30 test images, our algorithm correctly identified 23 of these giving us 23 percent recognition errors. We note that the implementation had tendencies to correctly identify some features more often than other. Similarly, some of the individuals were significantly easier for the SVMs to identify. Earlier tests with less complete models gave us highly erroneous results, which indicated that large amounts of training were necessary for this algorithm. The SVM models that were finally tested did however give great results due to the thousands of trained images used.

6 Conclusion

Our realization of facial recognition was quite successful given that it is our first attempt at such an algorithm. The primary challenge we encountered however was collecting data and training our SVMs. In future iterations of this work we suggest using 3D rendered models of faces in order to obtain many more illuminations and angles. In addition, it would be helpful to use several more features in order to better distinguish individuals. Nevertheless, our implementation was still able to identify the correct individual with remarkable accuracy.

References

[1] Cognitec. Cognitec home page. `http://www.cognitec-systems.de/index.html`.

[2] G. Edwards, T. Cootes, and C. Taylor. Face recognition using active appearance models. *Lecture Notes in Computer Science*, (1407):581–595, 1998.

[3] FaceKey. Facekey home page. `http://www.facekey.com/`.

[4] B. Heisele, T. Serre, and T. Poggio. A component-based framework for face detection and identification. *International Journal of Computer Vision*, 74(2):167–181, 2006.

[5] T. Huang, Z. Xiong, and Z. Zhang. Face recognition applications. In *Handbook of Face Recognition*, pages 371–390. Springer, New York, 2005.

[6] W. Jiang. Human feature extraction in vs image using hog algorithm. University of Science & Technology of China research project.

[7] T. Joachims. Svm light. `http://svmlight.joachims.org/`.

[8] L-1 Identity Solutions. Faceit sdk. `http://www.identix.com/pages/101-faceit-sdk`.

[9] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 1991.