Active Learning in Face Recognition: Using Tracking to Build a Face Model

Robin Hewitt
Hewitt Consulting
San Diego, CA 92150, USA
rhewitt@acm.org

Serge Belongie
University of California, San Diego
La Jolla, CA 92093, USA
sjb@cs.ucsd.edu

Abstract

This paper describes a method by which a computer can autonomously acquire training data for learning to recognize a user's face. The computer, in this method, actively seeks out opportunities to acquire informative face examples. Using the principles of co-training, it combines a face detector trained on a single input image with tracking to extract face examples for learning. Our results show that this method extracts well-localized, diverse face examples from video after being introduced to the user through only one input image. In addition to requiring very little human intervention, a second significant benefit to this method is that it doesn't rely on a statistical classifier trained on a pre-existing face database for face detection. Because it doesn't require pre-training, this method has built-in robustness for situations where the application conditions differ from the conditions under which training data were acquired.

1. Introduction

This paper describes, and demonstrates key elements of, an approach by which a computer can train itself to recognize and distinguish individual faces. In this method, the computer starts with a single image of a person's face. Then, using face recognition and face tracking together, it actively looks for additional informative examples of that face to use as a basis for learning to recognize that user under varied conditions of lighting, background, pose, and expression.

We give results showing that, using this active-learning paradigm, a computer can create a rich and representative dataset for learning to recognize a user's face under a wide range of conditions and expressions. We show also that this is possible starting with only a simple initial face model, based on one input image. The computer, not the user, does the work and manages this learning process.

This "active learning" method is especially well suited to a mobile robot that must recognize specific faces in a wide range of conditions. With traditional methods that are based on statistical classifiers trained in advance, it may be difficult to ensure that these varying conditions are accurately reflected in the training database. With active learning, the robot's face model is built from information gathered under the actual conditions where it is used.

In the presentation below, we first describe the motivation for our proposed method in Section 2 and review how it relates to previous work in Section 3. We describe our methods for building the initial face model from one input image and for extracting new face examples from video in Section 4. Finally, we describe our testing in Section 5 and present results in Section 6.

2. Motivation

Most existing face-recognition systems rely on data prepared ahead of time. The typical approach starts by using a statistical classifier to detect that a face is present. The statistical classifier is normally trained from a pre-existing database of face images. One benefit often claimed for this approach is that a statistical classifier can, in principle, be trained to very high accuracy, given a sufficiently large training database and discriminative features [15], [14]. However, this theoretical accuracy is based on the assumption that the training data are randomly drawn from a population with the same distribution as will be encountered in the actual application. This assumption may not be valid in practice. Statistical classifiers are inherently vulnerable, in that they have no way of adapting or correcting themselves if the conditions actually encountered do not match the training database.

The number of variables that may be relevant in the database distribution is immense: viewing angle, distance, background clutter, lighting spectrum, intensity, angle and diffuseness of lighting, differences between adults and children, differences among ethnic populations, differences between posed photographs and spontaneous expression. With so many variables, there's no assurance that the training database has taken all the relevant variables into account or that their distributions will be the same as will be found in the application context.

This problem may be especially critical for a socially interacting robot. These robots may be used for education, in hospitals, as helpmates for the elderly, or for entertainment. In such applications, the robot may be used in a variety of locations, with different lighting conditions and background clutter. It may view a standing adult's face from below, but interact at eye level with children or patients in wheelchairs. In these and other ways, socially interacting robots will need to recognize individual faces in circumstances that are unconstrained and unpredictable.

In contrast with existing methods, active learning is, by its very nature, adapted to the circumstances in which it is used. It does not rely on a pre-existing database. It does not require the user to pose for a comprehensive set of photographs under varying conditions. It only requires that the subject pose for a single image. From then on, the robot acts on its own initiative, actively seeking opportunities to obtain more images of the subject's face. Based on this expanding data set, it refines its model of the subject's face until it can recognize that face under a wide range of conditions. The robot thus builds its own, customized training database, without laborious assistance from users, in a way that is automatically adapted to its users' conditions.

This active learning method illustrates a broader way of viewing computer vision and the human-computer interface in general, that is, that the computer is an active participant in the human-computer interaction, not only accepting data provided by the user, but taking the initiative to obtain the data that allow it to learn.

3. Related work

Our approach relies on the co-training principles introduced by Blum and Mitchell [4]. Through co-training, two complementary classifiers are able to train each other using a combination of labeled and unlabeled data.

Successful visual co-training has been previously demonstrated in [12]. Our approach differs from [12] in that we use a tracker instead of a second classifier to identify informative training examples. By using a classifier and a tracker together, we take advantage of the temporal continuity of video sequences to validate both tracking and classification against one another while generating additional training examples. More significantly, our approach does not rely on pre-trained statistical classifiers to bootstrap the learning process. We start with only one input image.

Abramson and Freund, in [1], also build on the co-training paradigm introduced in [4] by pairing a trained classifier with a human. While this method reduces the amount of human labor, it still requires a great deal of human processing. Our approach requires very little human effort.

In [9], [10], and [11], Lee et al. combine recognition and tracking for face recognition tasks. Their approach differs from ours in requiring that a set of training images be acquired as the first step. The training images are used to learn a visual appearance manifold for each individual and the transition probabilities within that manifold. A tracker is used to semi-automate preparing the dataset, but human intervention was still required to make sure each video frame contained a properly positioned face region ([9]) and to manually assign face images to pose clusters ([10]). Our method, in contrast, uses an automated process in which a low-error detection at the start and end of a tracking sequence validates correct alignment for the tracked face region.

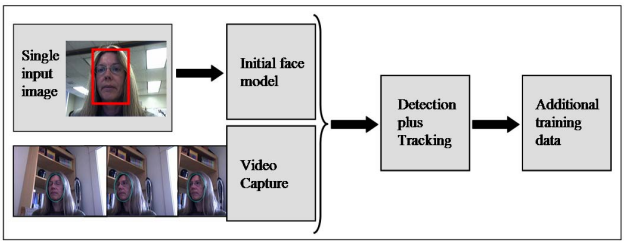

Figure 1. Block diagram of the process for generating additional positive examples of a user's face.

Viola and Jones [15] demonstrated an efficient face detector based on brightness differences between adjacent rectangular regions. Balas and Sinha [2] extended this feature type by lifting the requirement that light and dark rectangles be adjacent. They point out that any two rectangular regions can be paired. Balas and Sinha use these features, which they call dipoles, as descriptors for general object recognition.

In our one-shot method for learning an initial face model, we use these dipole features in a different way. Because each rectangle of a dipole can be made smaller than the bright or dark region it captures, these features can represent large-scale, high-contrast facial characteristics with soft localization. Since they're not constrained to be adjacent, reliably high-contrast region pairs can be combined to create high-level appearance filters that efficiently remove all but a small fraction of the image from the search space. This filtering step of our one-shot recognition method is similar to the method Ferencz et al. use in [7].

4. Methods

4.1. Overview of the approach

Figure 1 shows an overview of the process by which a robot can autonomously acquire a diverse set of informative face examples for an individual user. These positive examples can be used as input to a machine-learning algorithm.

This process begins with a single input image. The input image is used to create an initial face model for the user.

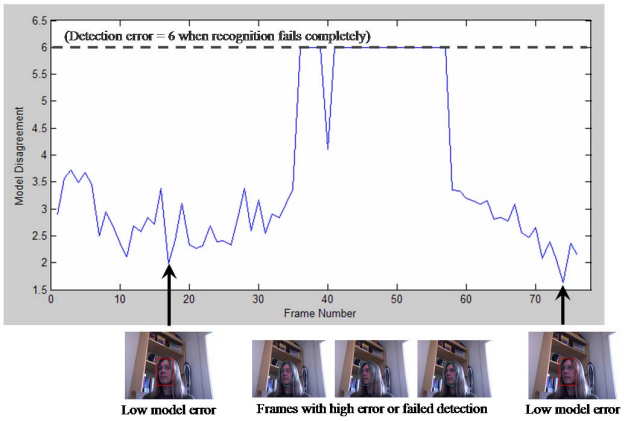

Figure 2. To acquire additional, informative face images, we look for sequences of frames with high model error or failed detection that are bracketed by two frames with low model error. We run the tracker over this sequence, starting at one of the low-error end frames and stopping at the other.

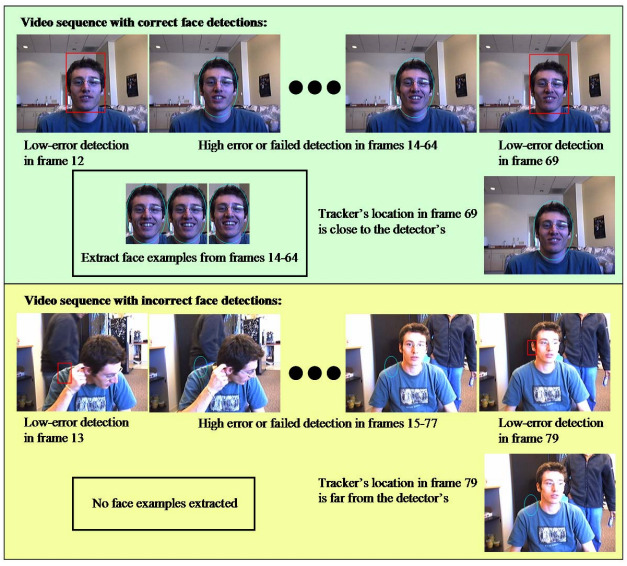

Figure 3. At the end of the sequence, the tracker's final location is compared to the location of the low-error detection in this frame. If these two image locations correspond closely, intervening frames that have low tracking error are extracted as face examples for learning.

This initial face model doesn't require security-level accuracy. It only needs to keep the user interested and engaged with the robot. As the user interacts, the robot searches its video input for high-confidence matches to the initial face model. When this search process finds a good match (one with sufficiently low error), the robot begins storing the incoming video stream. During these user interactions, the robot simply gathers video data. Processing occurs later, offline.

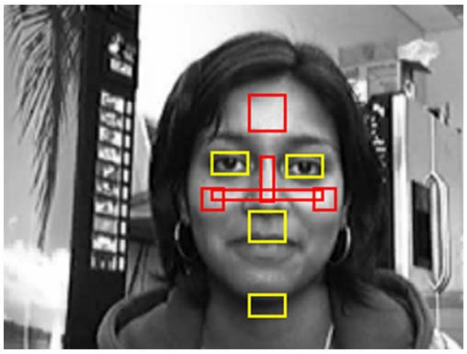

Figure 4. Face regions selected as light and dark rectangles for generating the large-scale features in an initial face model. Dark rectangles are outlined in yellow, light rectangles in red.

Offline processing uses the principles of co-training to find and extract informative face examples for each user. The goal of co-training is to build up a useful set of positive examples for one classifier with examples detected by a different, complementary classifier. For co-training to work, each classifier must, with high confidence, find positive examples that were either close to or outside the other classifier's decision boundary. Our method replaces one of the classifiers with a tracker. As in two-classifier co-training, if the tracker locates the face correctly and with high confidence in video frames where the initial face model either failed or matched with high error, it gives us positive examples that are specifically useful for improving the initial face model.

In the following sections, we describe the initial face model and how it's used for detection. We then give details of our method for combining tracking and detection to extract informative, positive face examples for learning.

4.2. Creating the initial face model

The initial face model is created by one-shot learning. A user presents a single, frontal-view face example as a video-frame capture and draws a bounding rectangle to indicate its location. Users are encouraged to include enough margin to capture areas of the head that are likely to be distinctive. For example, profile lines at the top and sides of the head can be distinctive features, even against a cluttered background [6]. Currently, users are also asked to mark the locations of eyes and nose tip. Since these locations need not be precise center points, we expect to be able to fully automate this step.

Our initial face model uses two types of features: large-scale features and small-scale features.

The large-scale features, described in [2], consist of paired light and dark rectangular regions. The regions within a pair need not be adjacent. Each region is rep-

resented by its average pixel brightness, and the average-brightness difference between paired regions used as a descriptor.

Some face knowledge is used to select good regions for large-scale features. The forehead (when visible), nose, and cheeks are generally brighter than the eyes, the shadowed area under the nose, and the area under the chin. The location for each of these light and dark rectangles is determined by examining an appropriate subregion of the user's face input (based on eye and nose-tip locations) for its brightest or darkest area. Once these light and dark rectangles (shown in Figure 4) are located, they're combined into large-scale features.

The selected features are organized into a filtering cascade. Each cascade level consists of three features. Two of these are a bilaterally-symmetric pair (for example, two eye-to-cheek features), and the third is a vertical feature (for example, forehead-to-nose-shadow). Section 4.3 describes how this filtering cascade is applied during search.

Our small-scale features are located at interest points within the input face image. Various interest-point operators could be used. We chose the Difference of Gaussians blob detector, described in [13], and the Harris Corner detector, described in [8]. Our descriptors for these small-scale features are similar to the SIFT descriptors presented in [13] or the HOG descriptors presented in [6]. They consist of a 4×4 cell array of partially overlapping subregions. Each subregion is represented as a weighted histogram of eight gradient directions.

4.3. Searching with the initial face model

To search a video frame for a match to an initial face model, potential face regions at multiple scales and at each image location are evaluated with the large-scale-feature cascade. A dipole feature is considered present if the difference in average brightness between the light and dark regions is half the observed level in the input image. During search, any face region that passes this test for two of the three dipole regions at that cascade level proceeds to the next level. This threshold setting (half the brightness difference observed in the input image) was selected based on ROC-curve analysis of a dataset consisting of forward-facing face images for six users and 50 cluttered background images. At and below this fraction of the original brightness difference, there were no false negatives from the cascade.

The cascade's output is a list of pixels, with each pixel associated with an image region at one search scale. The number of surviving pixels from a typical 320×240 image ranges from a few hundred to a few dozen for each scale. Since the cascade features use soft localization, the surviving image pixels form compact clusters. These are combined using region growing, further reducing the total number of potential face regions. By the next stage of the search process, the total number of search regions has been reduced from about $700,000$ (scales$\times$pixels) to about five or ten.

Algorithm for extracting face examples from stored video

1. Search for initial face model in each video frame. Plot model error v. frame number.

2. Find frame series with low model error. Run tracker over these frames. Set the tracking-quality threshold for high-confidence face tracking.

3. Find frame series with high model error bracketed by two frames with low model error.

4. Track face from start to end of frame series located in Step 3. Validate that the tracker's final location corresponds with the detector's location in the final frame of the series.

5. Extract frames that are within the tracker's high-confidence threshold.

Figure 5. Algorithm steps for finding and extracting informative face examples from stored video.

Once the image has been filtered against the large-scale features, the remaining candidate positions are evaluated using small-scale descriptors. The metric by which candidate regions are evaluated for model agreement requires explanation. Initially, the standard deviation for descriptor similarity between a feature in the model and that feature's appearance in the wild is unknown. Without more knowledge about appearance variability, the absolute value of a descriptor's similarity metric may be a poor guide to the quality of the match. Instead, each feature is assigned the (x, y) location corresponding to its best match within a local search area. Disagreement with the model is then computed as the sum of displacements between where each feature occurred in the model and where its best match was found. Intuitively, this error metric is a measure of cumulative local strain placed upon the model as each feature location is pulled and stretched to align with its best match in the image region.

4.4. Finding informative face examples

Finding and extracting informative face examples from stored video sequences occurs offline. Figure 5 shows the steps for doing that.

The first processing step is to search each frame in the video sequence using the initial face model described above. Each frame in which the user's face is detected is scored by its model error, as described above, and the location and scale of that frame's best match to the input image is recorded. Frames in which no match was found are flagged and assigned a high error. The result of this step is a plot of model error versus frame number, as shown in Figure 2.

The second step is to find sets of consecutive frames that have low model error and run the tracker over these frame sets. This step validates the tracker for use with this video sequence and determines the tracker's high-confidence tracking-quality threshold. The tracking location in each frame is compared to the detection location, and if they are close, tracking is considered correct. If the tracker is far from the detected face location, tracking in this frame is flagged as having failed. The results of this step are used to plot a cumulative density function for tracking quality when tracking has failed on these frames. A high-confidence tracking-quality threshold is selected based on the CDF. We selected this threshold as the 10% value for tracking failure.

In the third processing step, we locate sets of consecutive frames with high model error or failed detection that are bracketed by two frames with low model error (see Figure 2). Next, each candidate frame series is evaluated by tracking the face from beginning to end of the series. If the tracker's final location in each direction corresponds well with the location of the best face detection from the previous step, the frame series is retained for further processing. If, however, the tracker's final location is far from the location of this frame's low-error detection, no face examples are extracted from the intervening frames. This evaluation step is illustrated in the top portion of Figure 3.

The requirement that only high-error, intervening frames between two low-error detections be used for extracting new face examples helps prevent a stable background element that gives rise to a false positive from contaminating the dataset.

The requirement that the tracker's final location agree well with the location of the terminal low-error detection in a frame series helps validate both tracking and detection. The bottom portion of Figure 3 illustrates the reason for this validation step. The detector used in this series was crippled by removing several of its small-scale features to increase its vulnerability to false-positive errors. Ephemeral matches from wrinkled fabric in frame 12 and from the user's ear in frame 79 created two low-error detections that bracket a series of high-error detections in frames 15-77. Since the tracker's terminal location doesn't coincide well with the second low-error detection, however, false-positives from the intervening frames are prevented from entering the dataset for the user's face.

In the frame sets that are retained, the tracking-quality threshold determined in Step 2 serves as the basis for deciding whether to extract a new face example from each intervening frame. Since its purpose in this context is to serve as a co-training partner, the tracker's internal quality metric should be complementary to the face detector's. We used color-histogram-based trackers similar to the trackers described in [3] and [5]. For each video sequence, we selected the tracker to use with that sequence and set its parameter values dynamically so as to maximize location correspondence in Step 2.

5. Testing

To test the ability of this method to identify and extract new face examples in a complex environment, we captured short webcam videos of six subjects. The videos were taken in indoor environments with visual distractions that included passersby and/or significant background clutter. Subjects were asked to move to different locations between videos to introduce appearance changes due to differences in lighting direction.

At the start of each session, each subject was asked to assume a neutral expression and face into the camera for an initial input image. Subjects were then encouraged to engage in normal conversation and interact with either the robot or with the other people present while being filmed.

The resulting videos were then processed to extract additional face examples.

6. Results

Figure 6 shows the initial face input and some of the new face examples extracted from the video sequences using this method.

The number of face samples obtained for each subject per 500-frame video sequence varied from none to 12. In no cases was a non-face area extracted by this method.

Localization is generally good in the extracted samples. The main difficulty in acquiring consistent face samples was scale selection. This difficulty appears to have been due to the lack of robust scale adaptation in the tracker implementations we used.

Acknowledgments

The authors thank the UCSD students who generously volunteered their time as test subjects. We also thank Bill Avrin for his insightful suggestions and improvements to the text.

Serge Belongie was supported under National Science Foundation CAREER Grant 0448615 and the Alfred P. Sloan Research Fellowship.

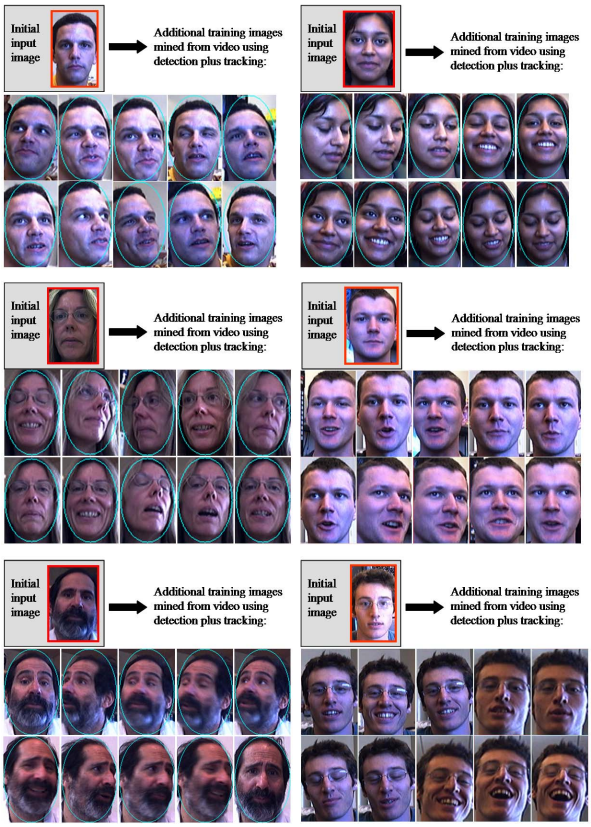

Figure 6. Examples of additional training examples extracted by this method.

References

[1] Y. Abramson and Y. Freund. Active learning for visual object recognition. http://www.cs.ucsd.edu/~yfreund/papers/sevillePaper.pdf. 2

[2] B. Balas and P. Sinha. Receptive field structures for recognition. Technical report, Computer Science and Artificial Intelligence Laboratory, MIT, 2005. MIT-CSAIL-TR-2005-015 (AIM-2005-006, CBCL-246). 2, 3

[3] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998. 5

[4] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory*. Morgan Kaufmann, 1998. 2

[5] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 142–149, 2000. 5

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005. 3, 4

[7] A. Ferencz, E. Learned-Miller, and J. Malik. Building a classification cascade for visual identification from one example. In *International Conference on Computer Vision (ICCV)*, pages 286–293, 2005. 2

[8] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 147–151, 1988. 4

[9] K. C. Lee, J. Ho, M. H. Yang, and D. Kriegman. Video-based face recognition using probabilistic appearance manifolds. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 313–320, 2003. 2

[10] K. C. Lee, J. Ho, M. H. Yang, and D. Kriegman. Online learning of probabilistic appearance manifolds for video-based recognition and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 852–859, 2005. 2

[11] K. C. Lee and D. Kriegman. Visual tracking and recognition using probabilistic appearance manifolds. *Computer Vision and Image Understanding (CVIU)*, 99(3):303–331, 2005. 2

[12] A. Levin, P. Viola, and Y. Freund. Unsupervised improvement of visual detectors using co-training. In *International Conference on Computer Vision (ICCV)*, pages 626–633, 2003. 2

[13] D. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV)*, volume 2, pages 1150–1157, 1999. 4

[14] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998. 1

[15] P. Viola and M. Jones. Robust real-time object detection. In *Proceedings IEEE International Workshop on Statistical and Computational Theories of Vision*, 2001. 1, 2