Learning Human Face Detection in Cluttered Scenes

Kah-Kay Sung and Tomaso Poggio
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, U.S.A.

Abstract. This paper presents an example-based learning approach for locating vertical frontal views of human faces in complex scenes. The technique models the distribution of human face patterns by means of a few view-based "face" and "non-face" prototype clusters. A 2-Value metric is proposed for computing distance features between test patterns and the distribution-based face model during classification. We show empirically that the prototypes we choose for our distribution-based model, and the metric we adopt for computing distance feature vectors, are both critical for the success of our system.

1 Introduction

Finding human faces automatically in a cluttered image is a difficult yet important first step to a fully automatic face recognition system. It also has many potential applications ranging from surveillance and census systems to human-computer interfaces. Human face detection is difficult because there can be huge and unpredictable variations in the appearance of face patterns. Because many of these variations are difficult to parameterize, traditional correlation-template pattern matching techniques [1] [2] and geometric model-based object recognition approaches tend to perform inadequately for detecting faces. Some non-parametric approaches, such as *view-based eigen-spaces* [3] and *image invariance* schemes [4], have been recently proposed for detecting face patterns, but so far, they have only been demonstrated on images with little clutter.

1.1 Example-based Learning and Face Detection

In this paper, we formulate the face detection problem as one of learning to recognize face patterns from examples. We use an initial database of about 1000 face mugshots to derive a view-based model for the distribution of face patterns. We then train a multi-layer perceptron net classifier on a sequence of "face" and "non-face" examples, to empirically discover a set of distance feature thresholds that separates "face" patterns from "non-face" patterns. Our learning-based approach has the following distinct advantages over existing techniques:
(1) It depends on the distribution of real face patterns and not on domain specific knowledge to build face models. This immediately eliminates potential modeling errors due to inaccurate or incomplete knowledge.
(2) It derives its operating parameters and thresholds automatically from a large number of annotated input-output examples and not manually from a few trial

Hlaváč, Šára (Eds.): CAIP '95 Proceedings, LNCS 970
© Springer-Verlag Berlin Heidelberg 1995

cases. The thresholds and parameters it arrives at are therefore statistically more reliable because they come from a larger and wider sample of training data.
(3) It can be made arbitrarily robust by increasing the number and variety of its training examples. Both *false positive* and *false negative* detection errors can be easily corrected by further training with the wrongly classified patterns.

2 System Overview and Approach

Our view-based approach searches the image exhaustively over multiple scales for square patches of the human face (henceforth referred to as *canonical face patterns*) whose upper boundary lies just above the eyes and whose lower edge falls just below the mouth (see Figure 1(a)). At each image location and scale, the system classifies the local image pattern as being either "a face" or "not a face", based on a set of local distance feature measurements to the face model.

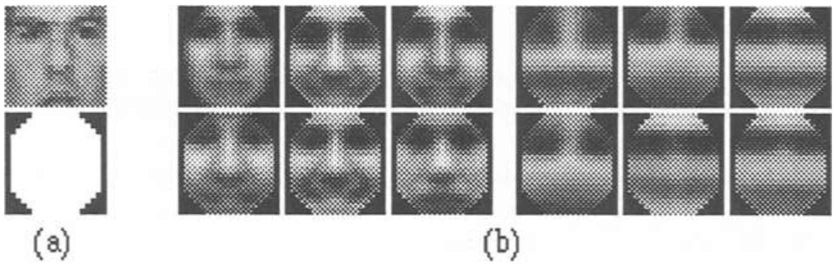

Fig. 1. (a): A "canonical" face pattern and a 19 × 19 mask for eliminating near-boundary pixels of canonical face patterns. **(b):** The 12 prototype patterns for approximating the distribution of face patterns. The 6 patterns on the left are "face" prototypes. The 6 on the right are "non-face" prototypes.

Clearly, the most critical part of our system is the algorithm for classifying new window patterns as "faces" or "non-faces". The rest of this paper focuses on the approach whose general idea is as follows:

(1) We re-scale each window pattern to 19 × 19 pixels before classification. Matching with a fixed sized window simplifies our algorithm because it allows us to use the same classification procedure for all scales.
(2) In the 19 × 19 dimensional image window vector space, we use a few "face" and "non-face" window pattern prototypes to piece-wise approximate the distribution of canonical face patterns. These pattern prototypes serve as a view-based model for the class of *canonical face patterns.* Each prototype is encoded as a multi-dimensional Gaussian cluster with a centroid and a covariance matrix.
(3) We define a set of image computations to measure the "difference" between a new window pattern and our view-based canonical face model. Each set of image measurements is a vector of directionally dependent distance features between the new window pattern and the stored prototype window patterns in the 19 × 19 pixel image vector space.

(4) We train a *multi-layer perceptron* (MLP) net to classify new window patterns as "faces" or "non-faces" based on their vector of distance measurements to the pattern prototypes.

3 Synthesizing Pattern Prototypes

Our approach uses 6 "face" and 6 "non-face" multi-dimensional Gaussian clusters to piece-wise approximate the distribution of canonical face patterns in the 19×19 pixel image vector space. We choose a piece-wise continuous modeling scheme because we believe face patterns occupy a smoothly varying and continuous manifold in this vector space. Figure 1(b) shows the 12 pattern prototype centers we synthesized for our scheme. The 6 "face" prototypes are synthesized from a database of canonical face window patterns, while the 6 "non-face" prototypes are derived from a similar database of non-face patterns.

3.1 Preprocessing

The first step of synthesizing prototypes is to normalize each sample window pattern in the "face" and "non-face" databases to compensate for certain sources of image variation:

(1) Window resizing: Our scheme re-scales window patterns of different sizes to 19×19 pixels before performing modeling and classification. We choose a 19×19 window size to keep the dimensionality of the vector space manageable small, but also large enough to preserve distinctive features of face patterns.
(2) Masking: We use the 19×19 binary pixel mask in Figure 1(b) to zero-out some near-boundary pixels of each window pattern. For "face" patterns, these masked pixels usually correspond to irrelevant background pixels.
(3) Illumination gradient correction: This operation subtracts a best-fit brightness plane from the unmasked window pixels and helps remove heavy shadows caused by extreme lighting angles.
(4) Histogram equalization: This operation adjusts for several geometry independent sources of window pattern variation, including changes in illumination brightness and differences in camera response curves.

Notice that the same preprocessing steps must also be applied to all new window patterns being classified at runtime.

3.2 Clustering for "Face" Prototypes

We use a database of 4150 normalized *canonical face patterns* to synthesize 6 "face" pattern prototypes. The database contains 1067 real face patterns, obtained from several different image sources. We artificially enlarge the database to 4150 patterns by adding slightly rotated versions of the original face patterns and their mirror images as *virtual examples.*

We use an *elliptical* version of the *k-means* clustering algorithm to compute 6 representative face patterns and their cluster covariance matrices from the enlarged database (see [5] for details). The *elliptical k-means* algorithm fits full covariance Gaussian clusters instead of isotropic Gaussian clusters to the data samples. It approximates the "face" data distribution more closely with the same number of clusters, because locally, the "face" data distribution can be a lot more elongated along certain directions of the image vector space than others.

3.3 Clustering for "Non-Face" Prototypes

There are many naturally occurring "non-face" patterns in the real world that look like faces when viewed in isolation. Figure 2 shows one such example. In order to avoid possible misclassification of these patterns as "faces", we explicitly model their distribution using 6 "non-face" prototypes. These "non-face" prototypes carve out negative regions around the "face" distribution that do not correspond to face patterns.

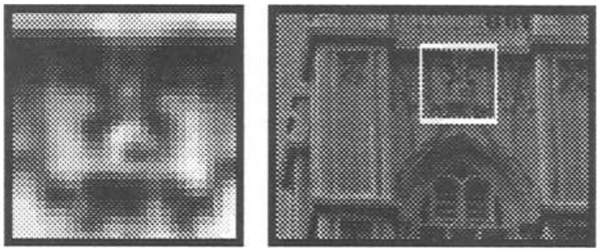

Fig. 2. An example of a naturally occurring "non-face" pattern that resembles a face. **Left:** The pattern viewed in isolation. **Right:** The pattern viewed in the context of its environment.

We use our *elliptical k-means* clustering algorithm to obtain 6 "non-face" prototypes and their cluster covariance matrices from a database of 6189 face-like patterns. Section 5.1 elaborates on how these face-like patterns are found.

4 A 2-Value Distance Metric

Our system classifies a new window pattern based on its vector of distances to the 12 prototype centers. In this section, we define a distribution dependent 2-Value metric for measuring the "distance" between a test pattern and a prototype pattern. We argue in [5] that our 2-Value distance takes into account both the detailed shape of the prototype cluster and the reliability of the shape estimate, and is thus a reasonable and robust measure of "dis-similarity" between a test pattern and the local data distribution near the prototype center.

The first distance value is a directionally dependent *Mahalanobis* distance between the test pattern and the prototype center, in a vector sub-space spanned by the cluster's 75 largest eigenvectors. Let $\mathbf{x}$ be the column vector test pattern, $\boldsymbol{\mu}$ be the prototype pattern, E_{75} be a 75 column matrix whose i^{th} column is a unit vector in the direction of the cluster's i^{th} largest eigenvector, and W_{75} be a diagonal matrix of the corresponding 75 largest eigenvalues. The covariance matrix for the cluster's data distribution in the 75 dimensional sub-space is given by $\Sigma_{75} = (E_{75} W_{75} E_{75}^T)$, and the first distance value is:

$$\mathcal{D}_1(\mathbf{x}, \mu) = \frac{1}{2}(75 \ln 2\pi + \ln |\Sigma_{75}| + (\mathbf{x} - \mu)^{\mathrm{T}} \Sigma_{75}^{-1} (\mathbf{x} - \mu)).$$

The second distance component is a standard Euclidean distance between the test pattern $\mathbf{x}$ and its projection $\mathbf{x_p}$ in the 75 dimensional largest eigenvector sub-space:

$$\mathcal{D}_2(\mathbf{x}, \boldsymbol{\mu}) = ||(\mathbf{x} - \mathbf{x_p})|| = ||(I - E_{75} E_{75}^T)(\mathbf{x} - \boldsymbol{\mu})||.$$

It accounts for pattern differences not captured by the first component. We assume an isotropic sample data distribution in this sub-space of smaller and possibly less accurate eigenvectors, and hence a Euclidean distance measure.

5 The Classifier

We use a *Multi-Layer Perceptron* (MLP) net with one hidden layer to identify "face" window patterns from "non-face" patterns, based on their vector of 2-Value distance features to the 12 prototype centers. The net has 12 pairs of input terminals (for the 12 pairs of distance values), one output unit and 24 hidden units. The net is trained with a standard backpropagation learning algorithm to output a '1' for "face" patterns and a '0' otherwise. Our experiments in the next section show that the number of hidden units and network connectivity structure do not significantly affect the classifier's performance.

5.1 Generating and Selecting Training Examples

Ideally, we would like to train our classifier with as large an example set of "face" and "non-face" distance vectors as possible, in order to attain classification with minimal error. Unfortunately, we must constrain the size of our training database because of limited disk space and training time.

We build a comprehensive but tractable database of "face" and "non-face"" patterns as follows: For "face" patterns, we simply collect all the frontal face views we can find in mugshot databases. Because we do not have access to many large mugshot databases, our sample of "face" patterns remains manageably small.

For "non-face" patterns, the task seems more tricky. In essence, every square non-canonical face window pattern of any size in any image is a valid "non-face" pattern, so the range of possible "non-face" patterns can grow intractably large in principle. To constrain the number of "non-face" examples in our database, we use the following "boot-strap" strategy that incrementally selects "non-face" patterns with high information value:

(1) Start with a small and possibly incomplete set of "non-face" examples in the training database.

(2) Train the MLP net classifier with the current database of examples.

(3) Run the face detector on a sequence of random images. Collect all the "non-face" patterns that the current system wrongly classifies as "faces". Add these "non-face" patterns to the training database as new negative examples.
(4) Return to Step 2.

At the end of each iteration, the "boot-strap" strategy enlarges the current set of "non-face" patterns with new examples that the current system classifies wrongly. These new examples are "useful" because they steer the classifier away from its current mistakes.

6 Results and Performance Analysis

Figure 3 shows some sample face detection results by our system. The system operates on window sizes of 19×19 pixels to 100×100 pixels at width increments of 120%. On a Sparc 10, the algorithm takes about 3 minutes to process a 256×256 pixel image. It usually detects faces within $\pm 5^\circ$ of the vertical.

The system marks each "face" it finds with an appropriately sized dotted box in the output image. Many of the faces in Figure 3 are enclosed by multiple boxes because they have been detected either at a few different scales or at a few slightly offset window positions. The results show that the system (1) does not make many false positive mistakes (none in this case) even for fairly complex scenes, (2) finds faces successfully at very different scales, and (3) detects real faces and hand-drawn faces equally well. The current system does not detect Geordi's face in the top image because his eyes are occluded by an opaque metallic visor, which is atypical of face patterns in our training database.

6.1 Measuring the System's Performance

To quantitatively measure our system's performance, we ran our system on two test databases of new patterns and counted the number of correct detections versus false alarms.

The first test database consists of 301 frontal and near-frontal high quality face images of 71 different people with a fair amount of lighting variation. We use this database to obtain a "best case" detection rate for our system. The second database contains 23 cluttered images with a total of 149 face patterns. The quality of images range from high fidelity CCD pictures to low quantization newspaper scans. We use this database to obtain an "average case" performance measure for our system.

For the first database, our system correctly finds 96.3% of all the face patterns and makes only 3 *false detects.* This result is encouraging because often, one can easily replace poorer sensors with better ones to arrive at this level of performance. For the second database, our system achieves a 79.9% detection rate with 5 *false positives.* The mistakes are mostly either from low quality newspaper scans or hand drawn pictures. We consider this behavior acceptable because the system is merely degrading gracefully with poorer image quality.

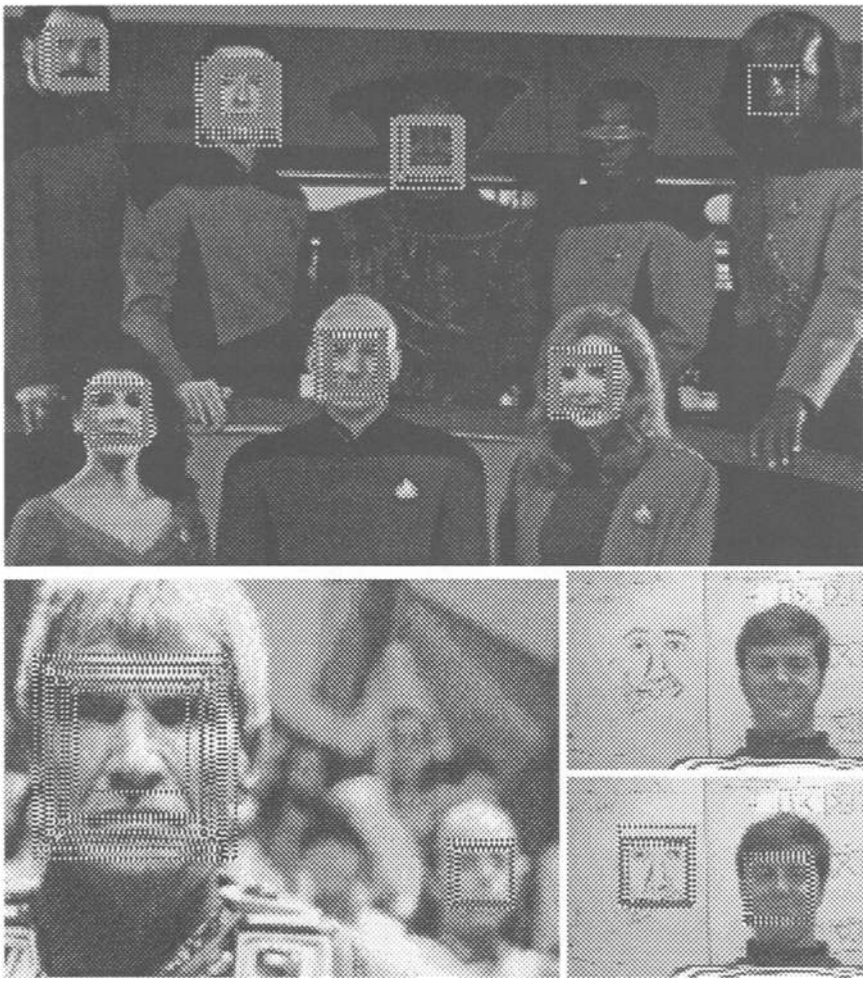

Fig. 3. Some face detection results by our system. See text for details.

6.2 Analyzing the System's Components

We conducted two additional experiments to identify the key components of our face detection algorithm. The first experiment investigates how the classifier's architecture affects our system's overall performance. To do this, we create and test a similar system with a single perceptron unit in place of the original moderately complex *multi-layer perceptron* net as a classifier.

The second experiment compares the performance of our 2-value distance metric with three other distance measures: (1) first component only ($\mathcal{D}_1$) of our 2-value distance metric, (2) the second component only ($\mathcal{D}_2$) of our 2-value distance metric, and (3) the standard Mahalanobis distance ($\mathcal{M}_n$) using all eigenvectors of each cluster. To conduct this experiment, we configure and generate statistics for three new systems, each using one of the three distance measures in place of our 2-Value distance metric. Notice that because the three new distance measures are all single-value measurements, we also have to reduce the number of classifier input terminals from 24 to 12.

Classifier	Distance Metric							
	2-Val		$\mathcal{D}_1$		$\mathcal{D}_2$		$\mathcal{M}_n$	
MLP	96.3%	3	91.6%	21	91.4%	4	84.1%	9
	79.9%	5	85.1%	114	65.1%	5	42.6%	5
Single	96.7%	3	93.3%	15	92.3%	3	93.0%	13
Perceptron	84.6%	13	85.1%	94	68.2%	5	58.6%	11

Table 1. Detection rates versus number of false positives for different classifier architectures and distance metrics. The four numbers for each entry are: **Top Left:** detection rate for first database. **Top Right:** number of false positives for first database. **Bottom Left:** detection rate for second database. **Bottom Right:** number of false positives for second database.

Table 1 summarizes the results of our two experiments. Empirically, the figures suggest that while the classifier's network architecture does not significantly affect the system's performance, our 2-Value distance metric noticeably out-performs the other three distance measures in terms of achieving both high detection rates and few false positive errors simultaneously.

7 Conclusion

We have successfully developed a system for finding unoccluded vertical frontal views of human faces in images. The approach is view based. It models the distribution of face patterns by means of a few prototype clusters, and learns from examples a set of distance parameters for distinguishing between "face" and "non-face" test patterns. We are currently extending the system to detect faces over a wider range of poses instead of just near-frontal views. We stress again, however, that our ultimate goal is to develop a general methodology for taking on feature detection and pattern recognition tasks in multiple domains.

References

1. M. Bichsel. *Strategies of Robust Objects Recognition for Automatic Identification of Human Faces.* PhD thesis, ETH, Zurich, 1991.
2. R. Brunelli and T. Poggio. Face Recognition: Features versus Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, 1993.
3. A. Pentland, B. Moghaddam, and T. Starner. View-based and Modular Eigenspaces for Face Recognition. In *Proc. IEEE CVPR*, pages 84–91, June 1994.
4. P. Sinha. Object Recognition via Image Invariants: A Case Study. In *Investigative Ophthalmology and Visual Science*, vol 35, pages 1735–1740, May 1994.
5. K. Sung and T. Poggio. Example-based Learning for View-based Human Face Detection. AIM–1521, MIT Artificial Intelligence Laboratory, December 1994.