A Fast and Accurate Face Detector based on Neural Networks

Raphaël Féraud and Olivier Bernier
and Jean-Emmanuel Viallet and Michel Collobert
France-Télécom R & D

October 17, 2000

Abstract

Detecting faces in images with complex backgrounds is a difficult task. Our approach, which obtains state of the art results, is based on a new neural network model: the Constrained Generative Model (CGM). Generative, since the goal of the learning process is to evaluate the probability that the model has generated the input data, and constrained since some counter-examples are used to increase the quality of the estimation performed by the model. To detect side view faces and to decrease the number of false alarms, a conditional mixture of networks is used. To decrease the computational time cost, a fast search algorithm is proposed. The level of performance reached, in terms of detection accuracy and processing time, allows to apply this detector to a real world application: the indexation of images and videos.

Keywords : combination of models, face detection, generative modesl, machine learning, neural networks, projection.

1 Introduction

To detect a face in an image means to find its position in the image plane (x,y) and its size or scale (z). Two broad classes of algorithms can perform this task.

An image of a face can be considered as a set of features such as eyes, mouth, nose with constrained positions and size within an oval: an explicit model can be used. One of the simplest and fastest method to realize the feature extraction is the projection of the image or the edge image on the vertical axis to find the eyes or the mouth, and on the horizontal axis to

locate the nose [22, 7, 19]. Several other methods are currently used to perform the feature extraction: Gabor filter [25], oval detection [31, 24]... A similarity measurement between features is then used for face recognition or face detection task: Mahalanobis distance [7], cross-correlation [2, 7, 5], graph matching [25], elastic matching of features [40], decision tree [19], neural network [7], belief network [8]...

Considering that an image of face is a particular event in the set all the possible images, extracted windows of the image can be analyzed to determine if these windows contain faces or parts of background. A probabilistic or statistic model can be used to analyze the pixels intensity of each sub-window (extracted window of the image). This model can be built with different methods: neural networks [6, 38, 12, 33, 20, 36, 29, 30, 13, 14], principal components analysis [35, 11, 15, 17, 18, 26], Kullback distance and maximum likelihood method [10], Support Vector Machines [27, 28]...

For face detection, the advantage of explicit models is usually the speed of the features extraction algorithm and the similarity measurement task in comparison to the methods directly based on the analysis of pixels intensity. For explicit models, since the features have to be detected, the range (minimum size of detected faces) and the robustness to partial occlusion of faces are generally lower than for those based on probabilistic models. As a consequence, the performances of probabilistic models based on direct sub-windows evaluation are usually better.

Our approach first implements simple processes, based on standard image processing and then more sophisticated processes based on statistical analysis. In section 2, the different components of the face detector are described: a motion filter, a color filter, a pre-network filter and a large neural network filter based on a new model of neural network. A combination of neural networks is used, to extend the face detection ability in orientation. In section 3, a fast search algorithm for face detection is presented. It speeds up the detection process by a factor 25. After analyzing and comparing the performances of our face detector with previously reported face detectors in section 4, section 5 describes a real application: indexation of face images for the web crawler of France Telecom, *VoiLa*.

2 The face detector

Our purpose is to classify a sub-window x, of size 15x20 pixels extracted from an image, as a face ($x \in \mathcal{V}$) or as a non-face ($x \in \mathcal{N}$). In this section, we describe the different components of the face detector which consists of four filters. These filters, from the simplest, fastest and less accurate to the most complex, slowest and most accurate, are the following:

- a motion filter typically rejects 90 % of the hypothesis (location and scale of possible face) in the case of video sequences,
- a skin color filter typically discards 60 % of the hypothesis in the case of color images,
- a multi-layer perceptron, called pre-network, filters 93 % of the remaining hypothesis,
- a modular system, based on a combination of a new neural network model called Constrained Generative Model (CGM), processes the 0.04 % remaining hypothesis.

The architecture of the face detector is hierarchical: at each stage a percentage of the hypothesis is excluded (figure 2). The advantage of this architecture is first to reduce the computational time cost since the first filters are faster. Second, assuming that filters are independent, the false alarm rate can be improved. Indeed, estimations of the false alarm rates are 0.1 for the motion filter, 0.4 for the color filter, 0.01 for the multi-layer perceptron and 10^{-7} for the modular system. If the filters were independent, the final false alarm rate could reach 10^{-10} ! The drawback of this architecture is the risk of reducing the detection rate. The first three filters must reach a very high detection rate to circumvent this problem.

2.1 Hypothesis elimination

Assuming that a face moves most of the time (speaking, breathing, eye blinking), the motion filter is activated in video sequences. It consists of a simple thresholded difference of images. Depending on the threshold and on the video sequence, our experiments on automatic framing [9] show that it typically excludes 90 % of the hypothesis.

Figure 1: Result of the color filter on a color image. White pixels correspond to skin color

When color information is available, a color filter, made up of a table of pixels, collected manually on a large collection of face images [9], is applied. A binary image is obtained (figure 1). The sub-windows, which

contain a small number of skin pixels, are considered as background sub-windows. The others, corresponding approximately to 40 % of the total number of sub-windows (depending on the image), are evaluated by the following filter: the neural network pre-filter.

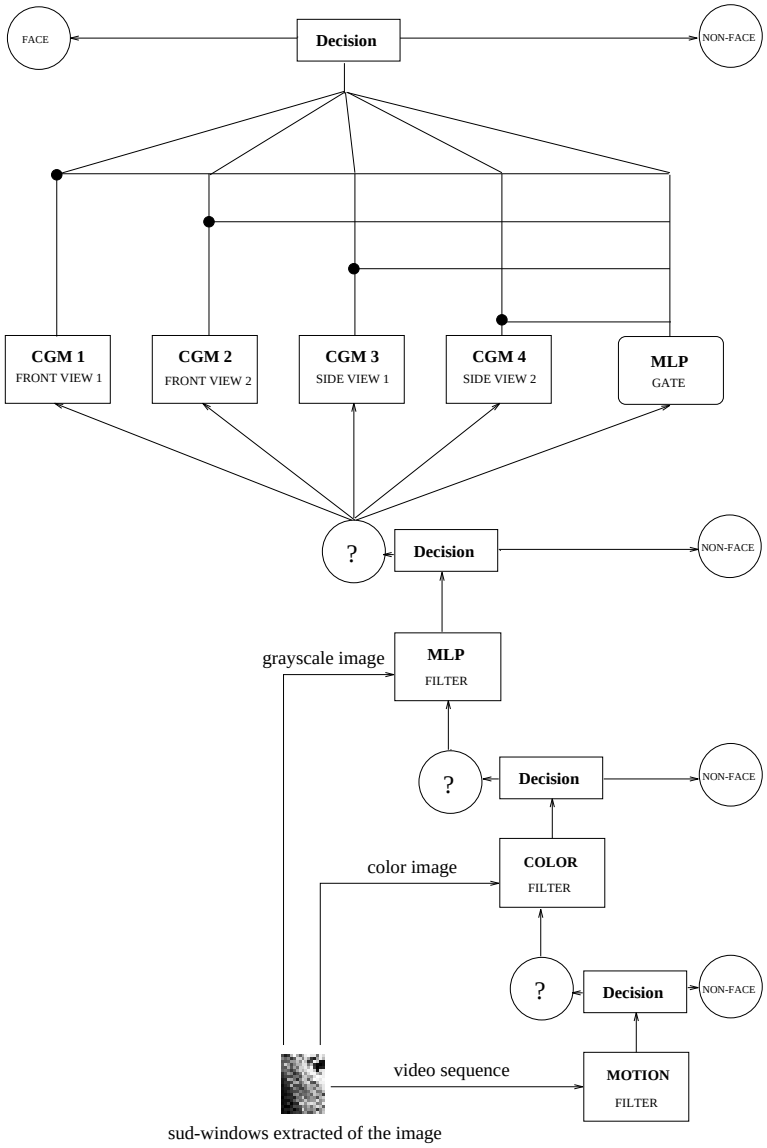


Figure 2: The face detector is composed by four stages. The last filter is the only one which is able to decide if the analyzed sub-windows is a face. (MLP: Multi-Layer Perceptron, CGM: Constrained Generative Model)

The pre-network is a single multi-layer perceptron (MLP) [4, 30, 36]. It has 300 inputs, corresponding to the size of the extracted sub-windows, 20 hidden neurons, and one output (face/non-face), for a total of 6041 weights. The pre-network is trained using standard back-propagation. The face training set is composed by 8000 front view and side view faces.

Figure 3: On the left examples of enhanced and smoothed front view faces $[0^o, 20^o]$. On the right examples of enhanced and smoothed turned faces $[20^o, 40^o]$.

Examples in figure 3 represent centered faces in 15 by 20 pixels sub-windows. Approximately $50,000$ specific non-face examples (15 by 20 pixels sub-windows, which do not correspond to faces) were collected using an iterative algorithm described later. The sub-windows are enhanced by a histogram equalization, and smoothed. Then, they are normalized by subtraction of the average face. The obtained pre-network is a relatively small and fast network with a very high detection rate (above 99 %) but also with a high false alarm rate (up to 1 %). This network, unusable alone because of its poor false alarm rate, is used as a filter which discards more than 93 % of the hypothesis.

2.2 The Constrained Generative Model

Two types of statistical model can be applied to face detection: discriminant models and generative models. Since collecting a representative set of non-face examples is impossible, our approach to face detection is to use a generative model. The Principal Component Analysis [32] (PCA) technique produces axes where the variance of the set of faces is maximum without taking into account the set of non-faces. This analysis can be used as a generative model to detect faces in an image [35]. The likelihood of the observed data x is then the product of two terms [26] based on two distances :

1. a distance to the principal subspace, based on the reconstruction error between an input sub-windows and its projection on the principal subspace, to discard non-face example which are projected on face examples,
2. a distance to a cluster in the principal subspace to delimit the cluster containing the set of faces.

The underlying assumption needed is that a linear subspace fitting the set of faces exists. If this assumption is not verified this model overestimates the set of faces (figure 4). The authors propose to use a mixture of linear subspaces to fit the manifold [26]. Another approach is to use a non-linear auto-associative neural network. An auto-associative network, using one hidden layer and linear activation functions performs a PCA [1]. Using three hidden layers of non-linear neurons, an auto-associative neural network is able to perform a non-linear dimensionality reduction [23].

However, owing to local minima, the obtained solution can be close to the principal components analysis.

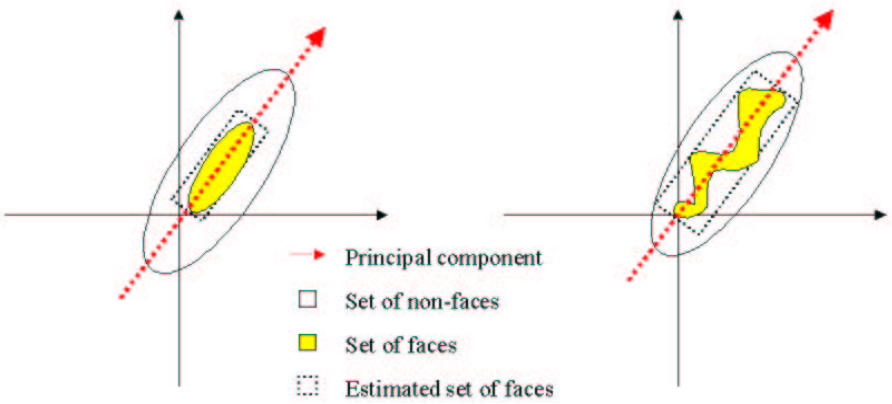


Figure 4: On the left, the set of faces can be fitted by a line. The estimation of the set of faces is accurate. On the right the shape of the set of faces is non-linear, the model overestimates the set of faces.

As in the previous case, our approach is to model the distance to the set of faces to evaluate the probability of an input sub-window to be a face. This distance is based on a projection $\mathcal{P}$ of a point x of the input space $\mathcal{E}$ on the set of face $\mathcal{V}$. We define this projection as :

$$\mathcal{P}(x) = \arg\min_{y \in \mathcal{V}}(d(x, y))$$

Where d is the Euclidean distance, $\mathcal{N}$ is the set of non-faces, and $\mathcal{E} = \mathcal{V} \cup \mathcal{N}$ is the set of all possible windows, with $\mathcal{V} \cap \mathcal{N} = \emptyset$.

As we have a sample of $\mathcal{V}$, we approximate the projection $\mathcal{P}$ of x on $\mathcal{V}$ as :

$$\mathcal{P}_{knn}(x) = \frac{1}{k}\sum_{i=1}^{k} v_i$$

Where $v_1, v_2, \ldots, v_n$ are the k nearest neighbors in the training set of faces of v, the nearest face of x. The number of nearest neighbors, k, needed to approximate the nearest face example of x, decreases as the density of the sample grows. The distance between an input vector x and the set of faces is approximated by :

$$\mathcal{D}(x, \mathcal{V}) \sim \|\mathcal{P}_{knn}(x) - x\|$$

Using a threshold, this distance allows to classify an input vector as a face or as a non-face. The accuracy of this approximation grows with the number of examples. However, the number of floating operations grows linearly with the number of examples. As a consequence, the computational time needed to evaluate the distance can be important.

To improve the previous algorithm, we propose to approximate the projection on set, $\mathcal{P}(x)$, using a neural network. The goal of the learning process is to evaluate the projection of an input example on the set. The output layer has the same size than the input layer. The neurons of the output layer corresponds to the coordinates in the input space $\mathcal{E}$ of the projected input (figure 5). To achieve this goal, we minimize the following cost function :

$$C_W = \sum_i (\mathcal{P}_W(x_i) - \mathcal{P}_{NN}(x))^2$$

Where W is the vector of weights of the neural network. $\mathcal{P}_{NN}$ is an approximation of the projection $\mathcal{P}$ defined before :

- if $x \in \mathcal{V}$, then $\mathcal{P}_{NN}(x) = x$,
- if $x \notin \mathcal{V}$: $\mathcal{P}_{NN}(x) = \mathcal{P}_{knn}(x)$

To classify an input sub-window x, the distance to the set is computed using the projection :

- $\mathcal{D}(x, \mathcal{V}) \sim \|x - \mathcal{P}_W(x)\|$, where $\mathcal{P}_W(x) = \hat{\mathcal{P}}_{NN}(x)$ is the reconstructed sub-window by the neural network,
- let $x \in \mathcal{E}$, then $x \in \mathcal{V}$ if and only if $\|x - \mathcal{P}_W(x)\| \leq \tau$, with $\tau \in \mathbb{R}$, where τ is a threshold used to adjust the sensitivity of the model.

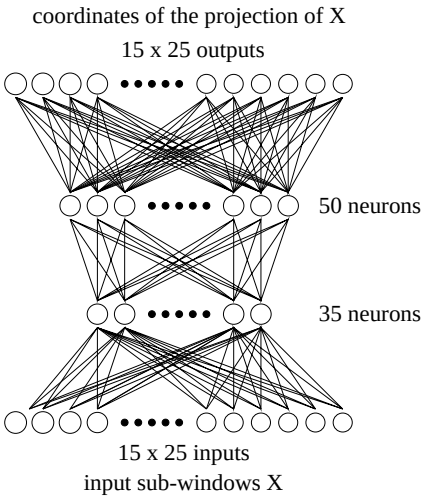


Figure 5: The use of three layers of weights allows to evaluate the distance between an input image and the set of face image: $\mathcal{D}(x_i, \mathcal{V}) \sim \|\mathcal{P}_W(x_i) - x_i\|$. The first and last layers both consist of 300 neurons, corresponding to the image size 15x20. The first hidden layer has 35 neurons and the second hidden layer has 50 neurons

Notice that the approximation of $\mathcal{P}$ by $\mathcal{P}_{NN}$ outperforms the one obtained by $\mathcal{P}_{knn}$, since face examples are reconstructed as themselves. As a

consequence, if the neural network generalizes the learned projection, its estimation ($\hat{\mathcal{P}}_{NN}$) of the projection should be better than the one obtained by $\mathcal{P}_{knn}$. Moreover, when testing, the computational time does not grow with the number of face examples. It depends of a fixed number weights, corresponding to the architecture of the neural network.

The drawback of this approach is that it needs non-face examples to model the projection. As we assume that the true dimension of the set of faces is lower than the input space (the size of input sub-windows), we can use a non-linear dimension reduction to reduce the number of non-face examples needed. To obtain a non-linear model with a multilayer perceptron, one hidden layer of non-linear neurons has to be used [39]. However, as we want to obtain a non-linear dimension reduction and a non-linear relation between the sub-manifold (the compression layer) and the projection layer (the output layer), we need an additional hidden layer (figure 5).

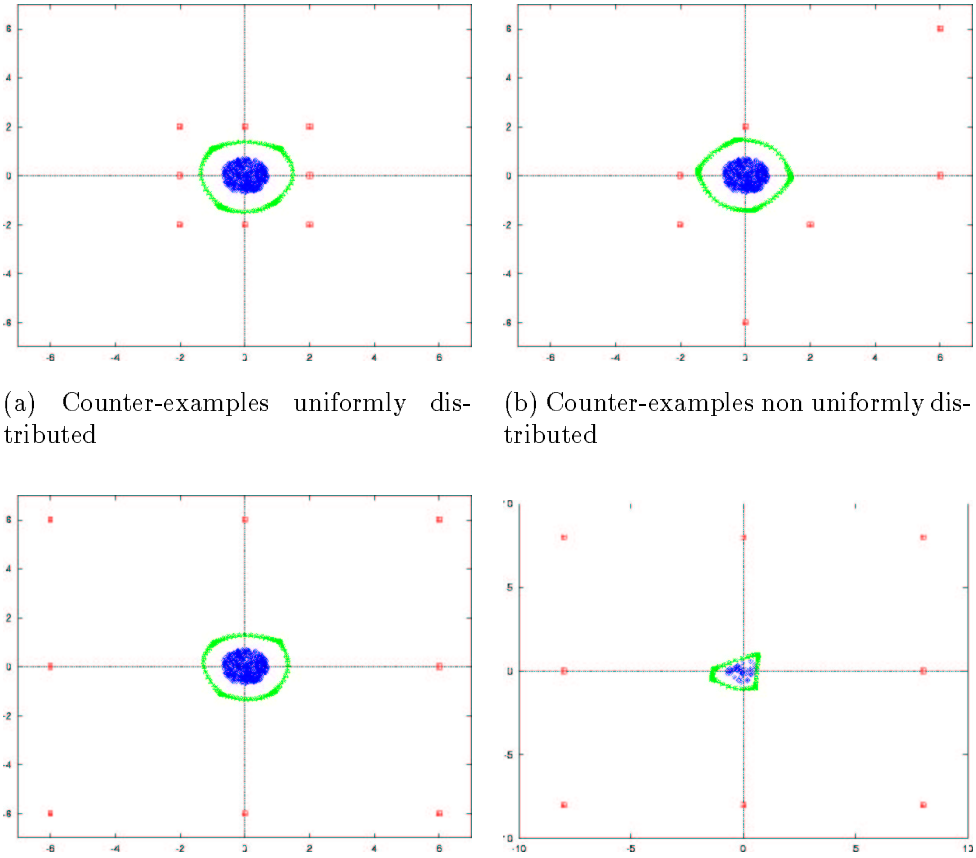
In the case of standard non-linear dimensionality reduction, the reconstruction error is related to the position of a point from the principal sub-manifold in the input space. Nevertheless, a point can be near to the principal sub-manifold ($\mathcal{V}'$) and far from the set of faces ($\mathcal{V}$). With the algorithm proposed, the reconstruction error is related to the distance between a point and the set of faces. As a consequence, if we assume that the learning process is consistent [37], our algorithm is able to evaluate the probability that a point belongs to the set of faces.

Let y be the binary random variable, $y = 1$ corresponding to a face example and $y = 0$ to a non-face example, we express this probability as:

$$P(y = 1|x) = e^{-\frac{(x-\hat{x})^2}{\sigma^2}} \text{ , where } \sigma \text{ depends on the threshold } \tau$$

We noticed on figure 6 that using a few number of counter-examples, the Constrained Generative Model (CGM) can perform an accurate estimation of the set of examples. However, the two counter examples used were not chosen randomly. They belonged to the principal plane of the set of examples. Here, we detail the algorithm we use to collect such counter-examples. The non-face database B_{nf}, corresponding to the face database B_f, is collected by an iterative algorithm similar to the one used in [33] or in [29]:

1. $B_{nf} = \emptyset$, $t = 0$, $F_\tau^0 = \infty$
2. the neural network is trained with $B_f + B_{nf}$,
3. the threshold τ^t is chosen such that the detection rate D_τ^t, on a validation set composed of face sub-windows, is equal to a target detection rate D_τ^*,
4. the false alarm rate of the model, F_τ^t, is then evaluated on a validation set of background images,



(a) Counter-examples uniformly distributed

(b) Counter-examples non uniformly distributed

(c) Counter-examples far from the set

(d) non dense set of examples

Figure 6: The elements of the class $\mathcal{V}$ (the points at the center of figures) belong to a disk. Different sets of counter-examples (the isolated points) are used to build the CGM. The model has the following architecture : 2 input neurons, two hidden layers of two neurons and two ouput neurons. The obtained boundaries correspond to points x where $\mathcal{D}(x, \mathcal{V}) = 1$. The quality of the density estimation depends on the density of the set of examples, more than on the proximity and the distribution of counter-examples.

5. the face detection system is tested on a training set of background images,
6. a maximum of 100 sub-images x_i are collected from the training set of background images with $\mathcal{D}\ (x_i, \mathcal{V}) \leq \tau^t$,
7. $B_{nf} = B_{nf} + \{x_0, \ldots, x_n\}$, $t = t + 1$,
8. while $F_\tau^{t-1} > F_\tau^t$ go back to step 2.

Since the non-face set ($\mathcal{N}$) is too large, it is not possible to prove that this algorithm converges in a finite time. Nevertheless, in only 8 iterations,

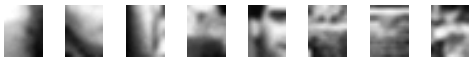


Figure 7: Left to right: the counter-examples successively collected by the algorithm are increasingly similar to real faces (iteration 1 to 8).

collected counter-examples are close to the set of faces (figure 7). In our experiments, we set the target detection rate on a set of sub-windows to 75%. The effective detection rate, the detection rate on a set of images, can be higher owing to the strong correlation between extracted windows from an image.

We use a similar boosting approach to collect the set of examples :

1. $B_f = B_0$, $t = 0$, $D_t = 0$
2. the model is build using using the previous algorithm to collect counter-examples and to evaluate the thresold τ^t,
3. the model is tested on the set of images containing faces I_t,
4. the faces $\{y_0, \ldots, y_n\}$, which are not detected, are manually cropped,
5. $B_f = B_f + \{y_0, \ldots, y_n\}$, $t = t + 1$,
6. while the detection rate $D_t < D_t^*$ go back to step 2.

To evaluate the sensibility of our algorithm to the density of the set of examples and to the distribution of the counter-examples, we use an illustrative and simple problem: to determine if a point belongs or not to a disk, in a two dimensional space (figure 6). Each sample of the true set (the disk) is drawn from an uniform distribution and some counter-examples are chosen to observe the behavior of our model under different initial conditions.

The most important issue (figure 6a, 6c), is that, contrary to discriminant algorithm such as MLP or Support Vector Machines, boundary points of the two classes are not needed in the training set to determine the boundary between the two distributions. Moreover, in figure 6b, we notice that an uniformly distributed set of counter-examples is not needed.

On the last test (figure 6d), we reduce the number of positive examples. Here the approximation of the projection $\mathcal{P}$ by $\mathcal{P}_{NN}$ is not accurate. The model underestimates the disk. To obtain an accurate estimation of the distance between a point and the set of examples, a dense training set of examples is needed.

There are several applications, such as face detection or word spotting, where the goal is to isolate a small cluster with unknown shape in a large space. Our algorithm is well suited to the case of dense clusters.

2.3 Combination of CGMs

In order to reduce the false alarm rate and to extend the face detection ability in orientation, three architectures, combining several CGMs, have been tested [14]: an ensemble, a conditional mixture and a conditional ensemble.

The use of ensemble of networks to reduce the false alarm rate was shown in [29]. The output f of the ensemble is the mean of the outputs of each estimator f_i:

$$f(x) = \frac{1}{N} \sum_i f_i(x)$$

Where N is the number of estimators and $f_i(x) = P_i(y = 1|x)$ the output of the CGM i.

As y is a binary variable, we have $f_i(x) = E_i[y|x]$ and $f(x) \sim E[y|x]$. If all the CGM estimators are identically and independently distributed, then, the variance of the generalization error of the ensemble is divided by a factor N [16].

The second combination model proposed is the conditional mixture. It uses several CGM models and a gate network as in the case of mixture of experts [21] (figure 2). A random variable θ is used to partition the training set, for example in two subsets:

1. $\mathcal{E}_1$, the set of front view faces and the corresponding counter-examples ($\theta = 1$),
2. $\mathcal{E}_2$, the set of side view faces and the corresponding counter-examples ($\theta = 2$),

Each module evaluates the probability of an extracted sub-window of the image to be a face, knowing the value of the random variable θ. Supposing that the partition ($\theta = 1, \theta = 2$) can be generalized to every input, including the non-face sub-windows, the gate network learns the partition. The output of the gate network for the CGM j is:

$$f_W(x, j) = \hat{P}(\theta = j|x)$$

Where W are the weights of the gate network. Then, the output of the modular system is:

$$\hat{P}(y = 1|x) = \sum_{j=1}^{N} (f_W(x, j) \hat{P}(y = 1|x, \theta = j)$$

Where the value of the random variable $y = 1$ corresponds to a face sub-window, N is the number of estimator, and $\hat{P}(y = 1|x, \theta = j)$ is the output of the CGM j. The cost function used during the training phase of the gate network is:

$$C_W = \sum_{x_i \in \mathcal{E}} [\sum_{j=1}^{N} (f_W(x_i, j) \hat{P}(y = 1 | x_i, \theta = j) - y_i]^2$$

This system is quite different from a mixture of experts introduced in [21]: each module is trained separately on a subset of the training set and then the gating network learns to combine the outputs. Since prior knowledge is used to part the training set, and since each module is trained separately, the capacity [37] of this system is less than for the more general case: the mixture of experts.

The last architecture described, the conditional ensemble, is trained on the face example as the conditional mixture and on the non-face example as the ensemble (the target of the gate network is the mean output).

For example, if two estimators are used, four sets are defined:

- $\mathcal{F}$ is the front view face set,
- $\mathcal{P}$ is the turned face set, with $\mathcal{F} \cap \mathcal{P} = \emptyset$,
- $\mathcal{V} = \mathcal{F} \cup \mathcal{P}$ is the face set,
- $\mathcal{N}$ is the non-face set, with $\mathcal{V} \cap \mathcal{N} = \emptyset$,

Our goal is to evaluate $P(x \in V | x)$. Each estimator computes respectively:

- $P(x \in F | x \in \mathcal{F} \cup \mathcal{N}, x)$ $(CGM1(x))$,
- $P(x \in P | x \in \mathcal{P} \cup \mathcal{N}, x)$ $(CGM2(x))$,

Using the Bayes theorem (see [14] for the demonstration), we have:

$$P(x \in \mathcal{V} | x) = P(x \in \mathcal{N} | x)[CGM1(x) + CGM2(x)] \quad (1)$$
$$+ P(x \in \mathcal{P} | x) CGM2(x) + P(x \in \mathcal{F} | x) CGM1(x) \quad (2)$$

Then, we can deduce the behavior of the conditional ensemble:

- in $\mathcal{N}$, if the output of the gate network is 0.5, and as in the case of ensembles, the conditional ensemble reduces the variance of the error (first term of the right side of the equation (1)),
- in $\mathcal{V}$, as in the case of the conditional mixture, the conditional ensemble permits to combine two different tasks (second term of the right side of the equation (2)): detection of turned faces and detection of front view faces.

The gate network $f_W(x)$ is trained to calculate the probability that the tested image is a face $(P(x \in \mathcal{V} | x))$, using the following cost function:

$$C_W = \sum_{x_i \in \mathcal{V}} ([f_W(x_i) CGM1(x) + (1 - f_W(x_i))] CGM2(x) - y_i)^2 + \sum_{x_i \in \mathcal{N}} (f_W(x_i) - 0.5)^2$$

3 The search algorithm

In this part, we focus on a way to reduce the computational time cost of the face detection process. The detector locates faces in a sub-window of fixed size, 15x20 pixels. To detect faces at different scales, a sub-sampling of the original image is performed. The exhaustive search leads to evaluate a very large number of sub-windows: all the sub-windows in all the sub-sampled images have to be tested. The goal of the first two filters (motion and color filter) is to eliminate hypothesis, using a very small amount of processing time. Nevertheless, for gray scale images, these filters cannot be used. The only remaining filter is the pre-network filter, which consists of $6,041$ weights: for each extracted sub-windows, $6,041$ multiplications and $6,041$ additions must be made (the modular system (figure 2) is made up of $140,741$ weights).

To reduce this computational time cost, a simple multi-layer perceptron can be used [30, 36], such as our pre-network. It determines the possible location of faces, and then a larger network is used to achieve precise location. Another approach, developed by Ben-Yacoub [3], is to calculate the Fourier transform of the image and of the neural network filter, and then to process the image in the Fourier space.

This interesting approach is not adapted to a local normalization of the image such as the histogram equalization we use. To reduce the computational time cost of the face detection process, our approach is to reduce the number of sub-windows analyzed.

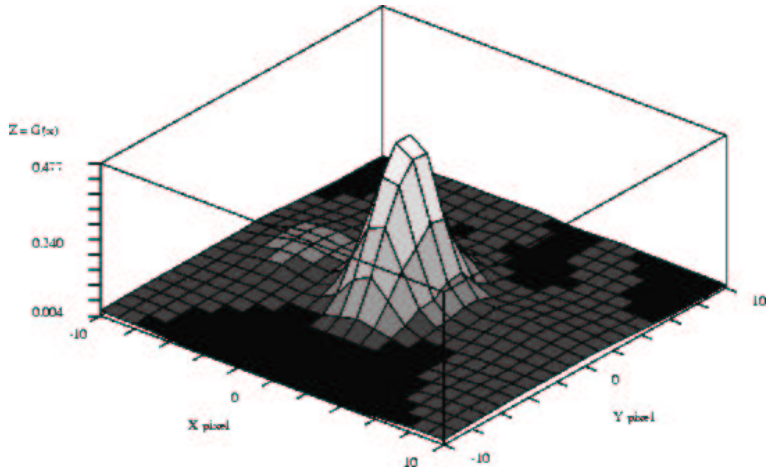


Figure 8: On the Z axis, the mean output of the modular system, over all the detected faces sub-windows of the CMU test set 1. The (X,Y) plane is the image plane. The origin corresponds to a face sub-window. The farther a sub-window is from the origin, the lower the output of the modular system ($G(x)$).

Our face detector is very selective: its mean output on background sub-windows is low in comparison to its mean output on face sub-windows. Moreover, around a face sub-window, the output of our face detector is a

monotonous and growing function (figure 8). These properties leads us to use the following algorithm to speed up the face detection process:

1. at each scale, each intersection point of a regular grid, corresponding to some pixels uniformly distributed in the image (figure 9), is tested by the detector (motion filter, color filter, neural network filter and modular system),
2. a local exhaustive search is performed around the points where $G(x)$, the output of the last module, is greater than a first threshold,
3. at each scale, the sub-windows, corresponding to the points of the local exhaustive search where $G(x)$ is greater than a second threshold, are stored in a set $\mathcal{V}_c$,
4. an overlapping elimination or summation (depending of the overlapping surface), between the different positions and scales of the sub-windows of $\mathcal{V}_c$, is performed to locate the faces.

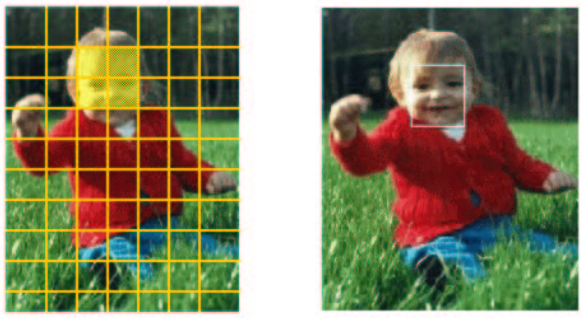


Figure 9: First, each intersection point of the grid is tested. Second, a exhaustive search is performed around the points of intersection, where the output of the system is high. In this illustrative example, 54 points of intersection of the grid are tested. Only one corresponds to a high output of the detector. The exhaustive search is performed inside the the dashed rectangle.

For example, in the color image of figure 9, the exhaustive search of a face, of size within the range [15x20,150x200] pixels, needs $500,000$ tests. The use of our fast search algorithm reduces the number of tests to $25,000$. $18,600$ hypothesis are discarded by the color filter. $5,800$ of the $6,400$ remaining sub-windows are eliminated by the pre-network filter and then the modular system evaluates only 600 sub-windows. The processing time is 0.3 second on a 333 MHz DEC Alpha.

4 Experimental results

In the first part of this section, a comparison between different models and combination of models is shown, using the exhaustive search. In the

second part, we analyze the influence of the search algorithm on the detection rate, false alarm rate and processing time. We describe our final face detector, and compare it to other systems. Our face database contains 8000 various face examples. This database is divided into four subsets of equal size, corresponding to different views: [0^o, 20^o] , [20^o, 40^o], [40^o, 60^o], [60^o, 80^o]. Each subset of face examples is collected using the algorithm described in section 2. Using these subsets, five CGMs are constructed: the first four (CGM1, CGM2, CGM3, CGM4) corresponds to each orientation range. The last one, CGM5, uses the whole face database. 75% of each face subset is used for the training and the 25% remaining faces allow to select the model. During the learning process, non-face examples are collected on a set of 100 background images. To select the model, a second set of 100 background images is used to evaluate the false alarm rate. Five sets of counter-examples are collected from the training set of background images, for each CGM, using the algorithm described in section 2. Each obtained set of extracted sub-windows contains approximately $2,000$ counter-examples. According to the experiments of section 2, the number of counter-examples needed by our model is very small in comparison to the number of counter-examples used by a discriminant multi-layer perceptron (for example, $50,000$ for the pre-network).

The size of the training windows is 15x20 pixels. The windows are enhanced by histogram equalization to obtain a relative independence to lighting conditions, smoothed to remove the noise and normalized by the average face, evaluated on the training face set.

4.1 Comparison of models

In this section we compare the different combination models, described previously, to choose the best one for our final detector. For this comparison, the ensemble of CGMs consists of three CGMs: a front view face detector (CGM1), a side view face detector (CGM3) and a general face detector (CGM5). The conditional mixture and the conditional ensemble use the same estimators (CGM1 and CGM3). The same architecture is used for the gate network. It has 300 inputs, corresponding to the size 15x20 pixels, 25 hidden neurons and one output.

To achieve comparison between models, two tests are performed. The first one allows to evaluate the limits in orientation of the face detectors. The *Sussex face database*, containing ten faces with ten orientations between 0 degree and 90 degrees, is used (table 1). Although, the general face detector (CGM5) uses the same learning face database than the different combinations of CGMs, it has a smaller orientation range than the conditional mixtures of CGMs, and the conditional ensemble of CGMs. The performances, on turned faces, of the ensemble of CGMs are low. The different models are trained on different face databases part according to the orientation criteria. Thus, the ensemble underlying assumption is not

Table 1 : **Results on Sussex face database**

orientation (degree)	CGM1	CGM3	CGM5	Ensemble	Conditional ensemble	Conditional mixture
0	100.0 %	100.0 %	100.0 %	100.0 %	100.0 %	100.0 %
10	62.5 %	100.0 %	87.5 %	100.0 %	100.0 %	100.0 %
20	50.0 %	100.0 %	87.5 %	87.5 %	100.0 %	100.0 %
30	12.5 %	100.0 %	62.5 %	62.5 %	100.0 %	100.0 %
40	0.0 %	100.0 %	50.0 %	12.5 %	62.5.0 %	87.5 %
50	0.0 %	75.0 %	0.0 %	0.0 %	37.5 %	62.5 %
60	0.0 %	37.5 %	0.0 %	0.0 %	0.0 %	37.5 %
70	0.0 %	37.5 %	0.0 %	0.0 %	0.0 %	25.0 %

verified: the estimators are not identically distributed. This test shows that the combination, by a gate neural network of different CGMs, trained on different training set, allows to extend the detection ability to both front view and turned faces. The conditional mixture of CGMs obtains results in term of orientation and false alarm rate close to the best CGMs used to construct the mixture (see table 1 and table 2).

Table 2 : **Results on the CMU test set A**
GM: the model trained without counter-examples, CGM1: front view face detector, CGM3: turned face detector, CGM5: general face detector, SWN: shared weight network,
Ensemble (CGM1,CGM3,CGM5), Conditional ensemble (CGM1,CGM3,gate), Conditional mixture (CGM1,CGM3,gate).

Model	Detection rate	False alarms rate	False alarms
GM	84 %	$1000\ 10^{-6}$	≈ 20000
CGM1	77 %	$5.43\ 10^{-6}$	47
CGM3	85 %	$6.3\ 10^{-6}$	212
CGM5	85 %	$1.36\ 10^{-6}$	46
one SWN (Rowley95)	84 %	$8.13\ 10^{-6}$	179
Ensemble	74 %	$0.71\ 10^{-6}$	24
Conditional ensemble	82 %	$0.77\ 10^{-6}$	26
Conditional mixture	87 %	$1.15\ 10^{-6}$	39

The second test allows to evaluate the false alarms rate. We use the test set A of the CMU, containing 42 images of various quality. First, these results show that the model, trained without counter-examples (GM), overestimates the distribution of faces and its false alarm rate is too large to use it as a face detector. Secondly, the estimation of the probability distribution of face images performed by one CGM (CGM5) is more precise than the one obtained by [29] with one SWN (see table 2). Since the results

of the conditional ensemble of CGMs and the conditional mixture of CGMs are close on this test, the detection rate versus the number of false alarms is plotted (figure 10), for different thresholds. The conditional mixture of CGMs curve is above the one for the conditional ensemble of CGMs. Since the conditional mixture obtains better results on the two tests, we chose this combination model for our final face detector.

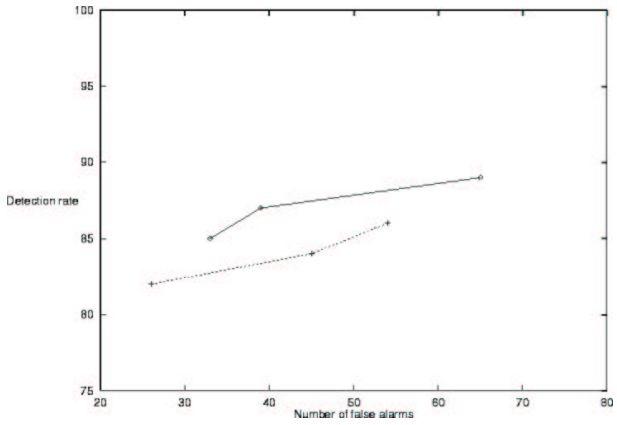


Figure 10: Detection rate versus number of false alarms on the CMU test set A. In dashed line conditional ensemble and in solid line conditional mixture.

4.2 Results of the face detector

The best performances are obtained by the conditional mixture of CGMs Nevertheless, the false alarm rate is still high (table 2), and the detection rate of side view faces is low (table 1). To solve this problem, four estimators are used (CGM1, CGM2, CGM3, CGM4) and then combined using the conditional mixture. The gate network has 300 inputs, 100 hidden neurons and one output. 6000 face images are used for the training and 2000 face images to select the model. A set of 5000 non-face examples is collected by an iterative algorithm on a set of 100 background images. A set 80 images containing faces on complex background allows to select the model. We compare our system with the best results published so far [30] on the test 1 of the CMU. It consists of 130 gray scale images, containing 507 faces, most of them front view faces.

To evaluate the detection ability in orientation, we use a larger test set than the *Sussex face database*. Our test set is composed of 30 individuals per orientation. The number of views is 10 (one per 10 degrees), 17 individuals are males and 13 are females.

In section 2, we noticed that our architecture is hierarchical (figure 2). Then, if the pre-network has a false alarm rate on the order of one percent and the modular system has a false alarm rate around 5.10^{-8}, and the estimators are independent, the expected value of the false alarm rate is 10^{-9}. The result (table 3) shows that the estimators are not independent,

Figure 11: The face detector search faces of size between 15x20 pixels and 300x400 pixels on the CMU test set. The number of tested hypothesis by the modular system is (left to right) 2781, 704 and 132. When enhancing image containing text by an histogram equalization, a face can appear in an extracted sub-window. Without body information, the face detector cannot eliminate it. The rough drawing faces are mostly not detected by the face detector. Side view faces are detected up to 90.

since the false alarm rate of the algorithm 1 and the algorithm 2 are close (around 10^{-8}). Nevertheless, the detection rate of these algorithms are close. Moreover, if the number of tests is the same, the computational time cost of one test is reduced by a factor 23 (corresponding to $140,000$ versus $6,000$ weights) using the pre-network filter.

Our fast search algorithms (number 3 and 4 in table 3) is based on the assumption, that the farther a sub-window is from a face sub-window, the lower the output of the face detector. Since the detection rate of the fast search algorithms is close to the one obtained by the exhaustive search (algorithm 2 in table 3) this assumption is verified. The false alarm rate of the fast search algorithm is higher than for the exhaustive search, together with a lower number of false alarms. This is not a contradiction, since the fast search algorithm focuses on the part of the images where the output of the face detector is high.

The test set 1 of the CMU contains a significant (26) number of rough

Table 3 : **Results on the test 1 of the CMU**
Algorithm 1: exhaustive search without the pre-network
Algorithm 2: exhaustive search
Algorithm 3: fast search (grid 3-3)
Algorithm 4: very fast search (grid 5-5)

Model	Detection rate	Missed faces	False alarm rate	False alarms
[Rowley,1998]	84 %	83	$1.2\ 10^{-7}$	10
[Rowley,1998] fast	77 %	117	$9.6\ 10^{-8}$	8
Algorithm 1	86 %	73	$9.7\ 10^{-8}$	8
Algorithm 2	84 %	80	$4.85\ 10^{-8}$	4
Algorithm 3	83 %	88	$1.6\ 10^{-7}$	3
Algorithm 4	81 %	95	$1.4\ 10^{-7}$	1

Table 4 : **Results on the test CNET set**
Algorithm 1: exhaustive search
Algorithm 2: fast search (grid 3-3)
Algorithm 3: very fast search (grid 5-5)

orientation (degree)	Algorithm 1	Algorithm 2	Algorithm 3
0	99.0 %	99.0 %	99.0 %
10	97.0 %	97.0 %	97.0 %
20	95.0 %	95.0 %	95.0 %
30	99.0 %	97.0 %	95.0 %
40	96.0 %	96.0 %	95.0 %
50	92.0 %	91.0 %	90.0 %
60	89.0 %	88.0 %	87.0 %
70	66.0 %	64.0 %	62.0 %
80	45.0 %	44.0 %	42.0 %
90	23.0 %	23.0 %	22.0 %

drawing faces or non-human faces, which mostly are not detected by our detector (figure 11). Nevertheless, our face detector has a detection rate equivalent to the one proposed by Rowley [30] (86% versus 84%) with a an equivalent number of false alarms (8 versus 10, see table 3). The fast version of the search algorithm has a higher detection rate (81% versus 77%) and a lower number of false alarms (1 versus 8) than the fast version of the CMU (see table 3). These results were obtained using only $7,109,000$ tests versus $83,000,000$ for the fast version of the CMU (a factor 11). Moreover, our face detector is able to detect side view faces (table 4).

The detection rate of all algorithms (exhaustive search, fast search and

very fast search) is around 90% up to 60 degrees. Finally, the performances of our face detector in terms of detection rate, false alarm rate and computational time cost is sufficient to apply it to real world applications, such as images and videos indexation or automatic framing.

5 Indexation of face images

Currently, most of the indexation engines on the Web are based on textual information. Information in a web page consists both of text and images. Therefore, the result of an image search, using a textual indexation engine, can be very noisy. In this section, we propose an image indexation engine, based on our face detector, in order to collect Web images containing faces [34]. The proposed service allows to sort easily images of faces. Moreover, access providers could store at low cost the face information: a cropped frame, containing the face can be stored instead of the whole image (figure 12).

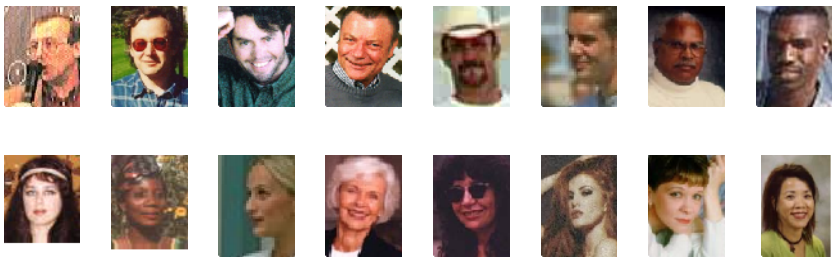


Figure 12: The extracted frames, after detection, contain the relevant information for indexation of face images, with a low storage cost.

Knowing the location (x, y, z) and number of faces, the image can be indexed with the following labels: portrait or group picture, image containing a face, and background image. Merging this information and the textual information, the functionalities proposed by our system are the following:

- automatic extraction of the frame containing faces to present the search results,
- search of a particular face image: *image of John Coltrane*,
- search of a portrait: *portrait of Bill Clinton*,
- search of a group photo: *picture of Beatles*.

The difficulty of this problem is to process the amount of information contained in the Web pages. The answers of the search engine must be, on the one hand, as non-noisy as possible, and on the other hand, as numerous

as possible. As a consequence, the false alarm rate must be very low to obtain non-noisy answers. Since the amount of information (in this case face images) is very important on the Web, there are two ways to collect many image of faces: fast search and high detection rate.

The very fast search algorithm (see section 3) is used in the search engine. To evaluate its performances, a large test set was collected on the Web. It contains 13,182 images of various size [108x108,1024x1024]. Most of these images are color images, but some of them are gray-scale. 3,468 images are background images, and to ease the evaluation of the results, the 9,714 others are selected so that the images contain only one face. For most of them the background is complex. There are 6,004 faces of male, and 3,710 of female. The variability of facial expressions, of orientations (in and out of the plane of the image) and of backgrounds is very high. The face detector search faces of size between 15x20 pixels and 300x400 pixels. To evaluate the influence of the use of the color information on the false alarm rate, detection rate, and on the average processing time, this test is made with and without the color filter.

Table 6 : **Results on the large test set**
Algorithm 1: exhaustive search without the color filter
Algorithm 2: very fast search without the color filter
Algorithm 3: very fast search with the color filter
Computer: DEC Alpha 333 MHz

Model	average processing time/image	detection rate	false alarms	tests of the color filter	tests of the MLP	tests of the mixture
Algorithm 1	34.3s	80.1 %	151		$1,669\ 10^6$	$117\ 10^6$
Algorithm 2	2.9s	76 %	99		$143\ 10^6$	$9.4\ 10^6$
Algorithm 3	1.27s	74.7 %	46	$143\ 10^6$	$62\ 10^6$	$5.3\ 10^6$

Due to the important variability of this test set, the detection rate of the face detector is lower than the one observed on the CMU test set (76% versus 81% for the very fast version and 80.1 % versus 86% for the exhaustive search). The use of the color filter reduces the detection rate (approximately by 1%). Nevertheless, the key points for this application are the false alarm rate to reduce the number of noisy answers, and the average processing time per image. The use of the color filter is beneficial for both key points and gives a very fast and accurate face detector: the average processing time is approximately 1s and the false alarm rate is on the order of 1 per 300 full images.

6 Conclusion

The new neural network model proposed, the Constrained Generative Model, performs an accurate estimation of the face set, using a small set of counter-examples. As we noticed in section 2, the requirement of this model is essentially a dense set of faces. The drawback of this algorithm is the size of the model. It is overcome by the use of several pre-filters and a fast search algorithm. The obtained face detector is one of the most accurate of the published face detectors: it detects side view faces as well as front view faces, its false alarm rate is on the order of 5.10^{-8} and using the fast search algorithm proposed, the number of indexed images could be raised to $100,000$ per day (the remaining bottleneck is the retrieving time of an image on the Web). To improve the detection rate and the false alarm rate, more estimators can be used without significant increase of the processing time, since the modular system processes only 0.4% of the extracted sub-windows (without the motion filter).

References

[1] P. Baldi and K. Hornik. Neural networks and principal components analysis: Learning from examples without local minima. *Neural Networks*, 2:53–58, 1989.

[2] R. J. Baron. Mechanisms of human facial recognition. *International Journal of Man Machine Studies*, 15:137–178, 1981.

[3] S. Ben-Yacoub. Fast object detection using mlp and fft. Technical report, IDIAP, Martigny, Switzerland, 1997.

[4] O. Bernier, M. Collobert, R. Féraud, V. Lemaire, J.E. Viallet, and D. Collobert. Multrak : a system for automatic multiperson localization and tracking in real-time. In *ICIP*, Chicago, USA, 1998.

[5] D. Beymer. Face recognition under varying pose. Technical report, M.I.T, 1993.

[6] H. Bouattour, F. Fogelman-Soulié, and E. Viennet. Solving the human face recognition task using neural nets. *Artificial Neural Networks*, 2:1595–1598, 1992.

[7] R. Brunelli and T. Poggio. Face recognition: Features versus templates. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 15, pages 1042–1052, October 1993.

[8] K. Choong Yow and R. Cipolla. Detection of human faces under scales, orientation and viewpoint variations. In *International conference on automatic face and gesture recognition*, pages 295–300, 1996.

[9] M. Collobert, R. Féraud, G. Le Tourneur, O. Bernier, J.E Viallet, Y. Mahieux, and D. Collobert. Listen: a system for locating and

tracking individual speaker. In *Second International Conference On Automatic Face and Gesture Recognition*, 1996.

[10] A. J. Colmenarez and T. S. Huang. Face detection and recognition. *Springer-Verlag, NATO ASI Series*, pages 174–185, 1998.

[11] G. W. Cottrell and P. Muro. Principal components analysis of images via back propagation. In *SPIE: Visual Communication and Image Processing*, volume 101, pages 1070–1077, 1988.

[12] P. Duchnowski, M. Hunke, D. Büsching, U. Meier, and A. Waibel. Toward movement-invariant automatic lip-reading and speech recognition. In *ICASSP-95*, 1995.

[13] R. Féraud. Face recognition: From theory to applications. *Springer-Verlag, NATO ASI Series*, 163:424–432, 1998.

[14] R. Féraud and O. Bernier. Ensemble and modular approaches for face detection: a comparison. In *Neural Information Processing System*, volume 10, december 1997.

[15] M. K. Fleming and G. W. Cottrell. Categorization of faces using unsupervised feature extraction. In *IJCNN*, volume 2, pages 65–70, 1990.

[16] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias-variance dilemma. *Neural Computation*, 4:1–58, 1992.

[17] B. A. Golomb, D. T. Lawrence, and T. J. Sejnowski. Sexnet: A neural network identifies sex from human faces. *Neural Information Processing Systems 3*, pages 572–577, 1991.

[18] R. Hoogenboom and M. Lew. Face detection using local maxima. In *International conference on automatic face and gesture recognition*, pages 334–339, 1996.

[19] J. Huang, S. Gutta, and H. Wechsler. Detection of human faces using decision trees. In *International conference on automatic face and gesture recognition*, pages 248–252, 1996.

[20] H. M. Hunke. Locating and tracking of human faces with neural network. Technical Report CS-94-155, CMU, 1994.

[21] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptative mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

[22] T. Kanade. Picture processing by computer complex and recognition of human faces. Technical report, Kyoto University, Departement of computer science, 1973.

[23] M. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37:233–243, 1991.

[24] D. Maio and D. Maltoni. Fast face location in complex backgrounds. *Springer-Verlag, NATO ASI Series*, 1998.

[25] B. S. Manjunath, R. Chellappa, and C. von der Malsburg. A feature based approach to face recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 373–378, 1992.

[26] B. Moghaddam and Pentland A. Probabilistic visual learning for object detection. In *The 5th International Conference on Computer Vision*, Cambridge MA, June 1995.

[27] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *CVPR*, 1997.

[28] C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *ICCV*, 1998.

[29] H. Rowley, S. Baluja, and T. Kanade. Human face detection in visual scenes. In *Neural Information Processing Systems 8*, 1995.

[30] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *PAMI*, 1998.

[31] R. Séguier. Human faces detection and tracking in video sequences. In *7th Portugese on Pattern Recognition*, 1995.

[32] Springer-Verlag, editor. *Principal Component Analysis*. New York, 1986.

[33] K. Sung and T. Poggio. Example-based learning for view-based human face detection. Technical report, M.I.T, 1994.

[34] M. J. Swain, C. Frankel, and A. Vassilis. Weebseer: An image search engine for the world wide web. In *CVPR*, 1997.

[35] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[36] R. Vaillant, C. Monrocq, and Y. Le Cun. Original approach for the localisation of objects in images. In *IEE Proc-Vis Image Signal Process*, volume 141, pages 572–577, August 1994.

[37] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York Heidelberg Berlin, 1995.

[38] E. Viennet and F. Fogelman-Soulié. Scene segmentation using multiresolution analysis and mlp. *Artificial Neural Networks*, 2:1599–1602, 1992.

[39] H. White and K. Hornik. Mutilayer feedforward networks are universal approximators. *Neural Network*, 2:359–366, 1989.

[40] A. L. Yuille, D. S. Cohen, and P. W. Hallinan. Feature extraction from faces using deformable templates. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 104–109, 1989.