Anisotropic Gaussian Filters for Face Class Modeling

August 31, 2006

Abstract

In this paper we propose a new approach to automatic human face detection which employs anisotropic Gaussian filters as local image descriptors. We then show how the paradigm of classifier combination can be used for building a face detector that outperforms the current state–of–the–art systems, while remaining fast enough for being used in real–time systems. We report a number of results on some reference datasets and we use an unbiased method for comparing the detectors.

1 Introduction

Given a still image, the goal of automatic face detection is to find all the human faces present in the image and to return their support regions (*i.e.* the bounding boxes). This is a key step in any system that relies on face processing (like face recognition or advanced human computer interfaces) and its performance represents a limiting factor for the quality of the whole system [1]. There are a large number of factors that influence the detection, some of them being intrinsic (*e.g.* inter–personal variability, face expression, gender, age and so forth) others being imposed by the environment (*e.g.* illumination, shadowing, camera parameters). Their combined effect makes the task of automatic face detection very challenging and, despite the research effort of the last decades, still unsolved for general cases.

During the years, a full range of methods have been proposed. At one end of the spectrum are the holistic methods, where the whole face is treated as a single object, while at the other end are the feature–based methods, where parts of the faces are identified independently and a final decision is taken by assembling the evidences. Between these two extrema lie other methods that combine global and local information for a better detection. For detailed surveys the reader is referred to [2] and [3].

In this paper we will consider the problem of frontal face detection in gray–scale images, and we will compare our results with some of the most representative methods in the field, which are briefly described hereinafter.

Probably the most significant example of neural networks applied to face detection is provided by the work of Rowley *et.al.*[5]. In the first version of their system, they use a multi–layer perceptron for learning the discriminant function from examples. To emphasize the local dependencies they used three types of receptive fields: an image of 20×20 pixels was divided into 4 10×10 subregions, 16 subregions of size 5×5 and 6 overlapping subregions of size 20×5. Each of these subregions corresponded to one hidden unit (26 in all). To further improve the performance, they have trained various networks that were combined using different schemes (AND, OR, voting and a separate arbitration network). Another approach is to model the manifold of faces by using locally linear subspaces. Sung and Poggio [4] used Principal Component Analysis (PCA) applied to local clusters: they grouped the faces and non–faces into 12 clusters (6 for each class) and then used PCA for constructing the local subspace of each class. Then the decision is taken according to the relative distance of the sample to the mean of both classes. In order to detect faces at any scale and position they use a sliding window which scans a pyramid of images at different scales. The detector proposed by Schneiderman and Kanade [17] also models the probability distribution of the face class, but they employ a naive Bayes classifier. Recently, a real–time face detector has been proposed by Viola and Jones [6] which successfully uses AdaBoost [7] algorithm for learning the discriminant function. The fea-

tures used are simple, Haar–like features, that can be computed with a few additive operations by exploiting a particular form of the image, called integral image. By using features of various sizes and shapes, a large pool of tens of thousands of features is created. The classifier built is a multi–stage (cascaded) classifier, where each stage is trained, using AdaBoost, to reject as many non–faces as possible while retaining (almost) all of the training faces. The classifier corresponding to one stage of the cascade has the form of a linear combination of weak learners, each weak learner using only one feature.

In this paper we introduce the anisotropic Gaussian filters for modeling the local appearance of the faces. As it will be shown, these filters capture better the salient features of the faces, leading to an improved discrimination, while remaining simple to compute. We will also discuss a hybrid system, in which the majority of candidate windows is removed by a system similar to Viola and Jones'[6], but the finer decisions are taken by our system.

The remaining of the paper is structured as follows. Section 2 introduces the new geometrical filters and discusses their ability to model the face patterns. It also gives a brief overview of AdaBoost, a learning algorithm that selects iteratively the best features. Detailed experiments are reported in section 3, where we compare our system with some state–of–the–art detectors. Finally, we draw some conclusions and discuss further improvements in section 4.

2 Boosting anisotropic Gaussian features

We will start with a short overview of the Adaptive Boosting (AdaBoost) algorithm and we will show how it can be used for performing feature selection too. Then we will introduce the anisotropic Gaussian filters used for face modeling.

Let

$$\{(\mathbf{x}_i, y_i), i = 1, \ldots, l\} \subset \mathbb{R}^n \times \{-1, +1\} \qquad (1)$$

be a set of labeled examples generated according to an unknown (but fixed) probability distribution function $P(\mathbf{x}, y)$. The problem of learning may be expressed as an optimization problem in which one wants to find the function f_{α^*} (from a suitably chosen set of functions, indexed here by the parameter α) which minimizes the risk of misclassifying new vectors drawn from the same *pdf* P:

$$\alpha^* = \arg\min_{\alpha} R(\alpha) = \arg\min_{\alpha} \int \mathcal{L}(y, f_\alpha(\mathbf{x}))dP(\mathbf{x}, y), \qquad (2)$$

where R is called *risk* and $\mathcal{L}$ is called *loss functional*. The loss functional penalizes the differences between the true label y and the predicted one $f_\alpha(\mathbf{x})$, and it has specific forms in various learning algorithms.

2.1 AdaBoost

AdaBoost [7] is a learning algorithm which iteratively builds a linear combination of some basic functions (*weak classifiers*) by greedily minimizing the risk based on the exponential loss,

$$L(y, f(\mathbf{x})) = \exp(-yf(\mathbf{x})). \qquad (3)$$

The final decision function has the form

$$f_T(\mathbf{x}) = \text{sign}\left(\beta_0 + \sum_{k=1}^{T} \beta_k h_k(\mathbf{x})\right), \qquad (4)$$

with $h_k : \mathbb{R}^n \to \{\pm 1\}$ being the weak classifiers. Training in the case of AdaBoost comes to finding the weak classifiers and their corresponding weights. For a detailed description of the algorithm the reader is referred to [7]. There are a number of theoretical and practical advantages in using AdaBoost, of importance here being the fact that by suitably chosing the weak classifiers, one may perform a feature selection implicitely when training the classifier. Another important feature of AdaBoost is that it converges towards a large margin classifier with positive impact on its generalization properties. However, in the presence of high levels of noise, AdaBoost, like the majority of classifiers, may overfit the training set.

Finally, note that depending on the application we might prefer to favor one of the classes. In AdaBoost, this can be easily implemented by building an asymmetric version of AdaBoost that encourages the correct classification of the desired examples, and by tunning the final threshold β_0 on an independent set in order to obtain the desired operating point on the ROC curve.

Now let $\mathbf{x} \in \mathbb{R}^n$ be a vector whose components will be denoted by $x_j, j = 1, \ldots, n$. If we let the weak classifiers be

$$h_j(\mathbf{x}) = \begin{cases} 1, & \text{if } \quad p_j x_j < p_j \theta_j \\ -1, & \text{otherwise} \end{cases}, \tag{5}$$

it turns out that AdaBoost will perform a feature selection too. Indeed, the final decision function will be a linear combination depending only on some of the features. This is the particular form of the weak classifiers that will be used for building the face detector and $\mathbf{x}$ will be the vector of all filter responses when applied to one image. For these weak learners (decision stumps), there are two parameters to be tuned: the threshold θ_j (chosen by maximum a posteriori rule) and the parity p_j.

2.2 Anisotropic Gaussian filters

In this section we propose a new set of local filters to be used for constructing the weak classifiers. The filters are made of a combination of a Gaussian in one direction and its first derivative in the orthogonal direction and have been introduced by Peotta et al. in [9] for image compression and signal approximation. The generating function $\phi : \mathbb{R}^2 \to \mathbb{R}$ is given by:

$$\phi(x, y) = x \exp(-|x| - y^2). \tag{6}$$

It efficiently captures contour singularities with a smooth low resolution function in the direction of the contour and it approximates the edge transition in the orthogonal direction with the first derivative of the Gaussian.

In order to generate a collection of local filters, the following transformations can be applied to the generating function:

- Translation by (x_0, y_0): $\mathcal{T}_{x_0,y_0}\phi(x, y) = \phi(x - x_0, y - y_0)$.
- Rotation with θ: $\mathcal{R}_\theta\phi(x, y) = \phi(x\cos\theta - y\sin\theta, x\sin\theta + y\sin\theta)$.
- Bending by r: $\mathcal{B}_r\phi(x, y) = \begin{cases} \phi(r - \sqrt{(x-r)^2 + y^2}, r\arctan(\frac{y}{r-x})) & \text{if } x < r \\ \phi(r - |y|, x - r + r\frac{\pi}{2}) & \text{if } x \geq r \end{cases}$
- Anisotropic scaling by (s_x, s_y): $\mathcal{S}_{s_x,s_y}\phi(x, y) = \phi(\frac{x}{s_x}, \frac{y}{s_y})$.

(a)

(b)

Figure 1: Filters used for modeling the faces: 1(a)Anisotropic Gaussian and 1(b)Haar–like filter.

Figure 2: Some of the first selected base functions.

By combining these four basic transformations, we obtain a large collection of functions $\mathcal{D} = \{\psi_{s_x,s_y,\theta,r,x_0,y_0}(x, y)\} = \{\mathcal{T}_{x_0,y_0}\mathcal{R}_\theta\mathcal{B}_r\mathcal{S}_{s_x,s_y}\phi(x, y)\}$. Figure 1(a) shows some of these functions with various bending and rotating parameters. We define the example $\mathbf{x}_k = (x_{jk})$ as the local responses of an image I_k to the all of the filters from $\mathcal{D}$:

$$x_{jk} = \iint \psi_j(x, y) I_k(x, y)\, dxdy \quad \forall \psi_j \in \mathcal{D}, \tag{7}$$

where the integral is taken over a suitable domain.

Figure 2 shows some functions selected in the first iterations of AdaBoost. It turns out that they are particularly well adapted to capture local contours which are less sensitive to changes of the lighting conditions. In comparison, Haar filters model global contrasts that are more sensitive the the direction of the light source.

2.3 Gaussian vs. Haar–like features

We are interested, first of all, to compare the Gaussian features (GF) with the more commonly used Haar–like features (HF), introduced in [6] (see Figure 1(b) for an example of such features). As we want to gain some insights about the intrinsic discrimination power of the two type of features, we trained two detectors using either the Gaussian filters or the Haar–like features, using the same training sets, and then we compared the two on an independent validation set. The figures 3 and 4 show the classification performace of the two classifiers either in terms of error rates or as Receiver Operating Characteristics curves.

Figure 3: Performance of GF and HF–based detectors

Figure 4: HF vs. GF: ROC curves

It is interesting to note from Figure 3 that while for the first ~ 100 iterations the error rate decreases quickly as we add more features to the model of the both classifiers, it remains practically constant for the HF–based detector. However, the GF model keeps improving, as we add more and more features. This shows that the HFs are not discriminant enough for modeling the finer differences between the two classes. Figure 2 shows the GFs that were selected during the first iterations so those that were deemed the most discriminative. Note also how they adapt to model the most salient features of the faces.

2.4 Cascaded classifiers

We have already noticed that the first features are good enough to produce an acceptable classifier. This observation led to the introduction of a multi–stage decision structure in which at each stage the number of non–faces that are rejected is maximized, constrained by having a low false negative rate. The system has a cascade (or serial) structure, with all the candidates that are not rejected by the i-th being fed into the $i + 1$ stage. Finally, the candidates that are not rejected by the last stage, are labeled as "faces". In our system, each stage was a classifier produced by boosting either the Gaussian or the Haar features.

Such design considerably improved the performances, as the latter stages can be tuned to correctly classify more complex candidates, without being distracted by simpler cases. In addition, the system becomes much faster, because the majority of the candidates is rejected using a few operations.

3 Experiments and results

In this section we report a number of experiments that we performed for assessing the performance of the proposed system and for comparing it with other standard systems. Also, for making our results reproductible, we use the objective scoring function for measuring the performances introduced by Popovici et al. in [15]. According to this evaluation method, there are 3 criteria for measuring the goodness of a detection, accounting for translation, rotation and scaling errors. Each of these criteria is given a score and the final score of a detection is their average. Finally, if a score is larger than a predefined threshold a detection is considered correct.

The detectors we built were following a classical scheme, in which the image was explored at different positions and scales using a sliding window. At each position (in image and scale space) the window is classified as either face or non–face. At the end, multiple detections of the same face must be arbitrated, and we have simply taken the average detection (even if it leads to a slighlty less precise detection).

3.1 Data used and model training

It is our experience that chosing non–square example images improves the performances of the face detector, so we have chosen to use examples of 20×15 pixels in size. The set of positive examples (faces) contained faces from three different publically available datasets: XM2VTS [12], BioID [13], FERET [14] which were cropped and downscaled to 20×15 pixels. After applying a number of transformations (scaling, rotations and translations), in order to incorporate some variability in the training set, the positive set contained $9{,}500$ examples. The set of negative examples (non–faces) was built by bootstrapping from randomly selected images which contained no faces and it contained about $500{,}000$ examples.

Before training we had to choose the parameters of the transformations to be applied to the the Haar–like and Gaussian filters. These tranformations would then condition the size of the set $\mathcal{D}$ of filters and, consequently, the dimensionality of the feature space. In our case, we had $37{,}520$ HFs and $202{,}200$ GFs, respectively.

We have built a four different detectors, either based on a unique type of features or combining the two – in which the first stages were built using HF and the last stages using GF:

- 5 stages of boosted Haar–like features (5HF)
- 12 stages of boosted Haar–like features (12HF)
- 12 stages of boosted Gaussian features (12GF)
- 5 stages of boosted Haar–like features followed by 12 stages of Gaussian filters (5HF+12GF)

3.2 Experiments

The systems were tested on two distinct datasets. The first one, the BANCA database[16], was used to assess the differences between the four systems described above. We have used the English and French subsets, containing $12{,}480$ images in which a single face was present in each of the images. As for any image the exact position of the face (given in terms of eye coordinates) is known, we were able to measure various parameters of the detections, as decribed in [15]. The overall detection rates (equal error rates) are given in Table 1 where we considered as corect detections only those with a score higher than 0.95, which means practically a perfect localization. Clearly,

Table 1: Comparisons of various methods on the BANCA database[16]. Only the detections with a score higher than 0.95 are considered correct.

Detector	% of detections with score > 0.95
5HF	52.08
12HF	86.78
12GF	91.02
5HF+12GF	90.74

the 12GF outperforms all the other detectors. However, it is interesting to note that adding 5 stages of HFs does not significantly change the overall classfication performance (there are only a few faces that are mistakenly rejected), it just considerably speeds up the system, as the great majority of non–faces are rejected during the first stages.

3.3 CMU/MIT Test set

The second round of experiments has been performed on a more challenging dataset commonly used to evaluate performances of face detectors, especially on very low resolution faces. The CMU/MIT Test set [11] was first introduced by Rowley in [5] for testing their system. The first version of this test set contained 23 images with a total of 155 very low resolution faces (it is referred as Dataset 1 in Table 2). The complete set contains 130 images with 507 faces (Dataset 3 in Table 2). However, some of these annotated faces are drawings or sketches and they are counted as false detections in some publications. To address this ambiguity, some papers only consider 123 images with 483 faces (Dataset 2 in Table 2). The three

Table 2: Performance of various detectors on the CMU/MIT set [11] in the 3 configurations. DR: detection rate percentage, FA: number of false alarms.

Methods	Dataset 1		Dataset 2		Dataset 3	
	DR	FA	DR	FA	DR	FA
Rowley [5]	87.1	15	92.5	862	90.5	570
Sung [4]	81.9	13	—	—	—	—
[17]	—	—	93.0	88	94.4	65
Viola [6]	—	—	—	—	91.4	50
12HF	**83.7**	**20**	**88.6**	**95**	**88.3**	**50**
5HF+12GF	**88.3**	**17**	**92.8**	**88**	**91.7**	**50**

versions of the dataset are tested in this paper to avoid any confusion. Table 2 compares our system with the state-of-the-art methods on these datasets. The results in this table have to be taken cautiously as they are affected by many factors: the scanning parameters (scaling factor, window shifting step, etc,...), the technique chosen for merging overlapping windows, the number of training patterns and so forth. In particular, the way the non–face examples are generated has a major impact on the decision functions. Finally some systems used additional post-processing to improve the results. For example, Viola et al. in [6] used a voting strategy between several cascades to reduce the false positive rate. This explains why our system 12HF has slightly lower performances than the implementation in [6]. However, the use of Gaussian filters gives roughly the same results as in [6], without any post-processing. Also in our model we used only 1260 features instead of 6061 as in [6]. One of the best performers, the system of Shneiderman [17], uses several intensity corrections and a complex wavelets–based network and the system is by far the slowest. Adding such pre– and post–processing to our system is beyond the scope of this paper.

4 Conclusions

This paper presents a new face detection system using anisotropic Gaussian filters which lead to high detection rates which may be adapted to perform real–time detection. While the new features are clearly more discriminant than, for example, the features proposed in [6], they are also more computationally demanding. However, by

combining the two approaches, a greatly improved real–time detector can be built. We performed various comparisons and they all concur to say that the new system greatly improves on the sate–of–the–art of the real–time systems.

References

[1] Yann Rodriguez, Fabien Cardinaux, Samy Bengio, and Johnny Mariéthoz, "Estimating the quality of face localization for face verification," in *IEEE International Conference on Image Processing, ICIP*, 2004, vol. 01, pp. 581 – 584.

[2] Erik Hjelmas and Boon Kee Low, "Face detection: a survey," *Computer Vision and Image Understanding*, vol. 83, no. 03, pp. 236–274, 2001.

[3] N. Ahuja M. Yang D. Kriegman, "Detecting faces in images: a survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 01, pp. 34–58, 2002.

[4] Kah Kay Sung and Tomaso Poggio, "Example-based learning for view-based human face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 39–51, 1998.

[5] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade, "Human face detection in visual scenes," in *Advances in Neural Information Processing Systems*, David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, Eds. 1996, vol. 8, pp. 875–881, The MIT Press.

[6] Paul Viola and Michael J. Jones, "Robust real-time face detection," *Int. J. Comput. Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[7] Yoav Freund and Robert E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[8] Vladimir N. Vapnik, *The nature of statistical learning theory*, Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[9] L. Peotta, L. Granai, and P. Vandergheynst, "Very low bit rate image coding using redundant dictionaries," in *Proceedings of the SPIE, Wavelets: Applications in Signal and Image Processing X*. SPIE, November 2003, vol. 5207, pp. 228–239, SPIE.

[10] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky, "Empirical analysis of detection cascades of boosted classifiers for rapid object detection,," in *Lecture Notes in Computer Science, Volume 2781, Sep 2003, Pages 297 - 304*, 2003.

[11] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23–38, 1998.

[12] K Messer, J Matas, J Kittler, J Luettin, and G Maitre, "Xm2vtsdb: The extended m2vts database," in *Second International Conference on Audio and Video-based Biometric Person Authentication*, 1999.

[13] Robert W. Frischholz and Ulrich Dieckmann, "Bioid: A multimodal biometric identification system," *Computer*, vol. 33, no. 2, pp. 64–68, 2000.

[14] P.J. Phillips and al., "The FERET database and evaluation procedure for face-recognition algorithms," in *Image and Vision Computing*, 1998, vol. 16.

[15] V. Popovici, J. Thiran, Y. Rodriguez, and S. Marcel, "On performance evaluation of face detection and localization algorithms," in *Proceedings of the 17th International Conference on Pattern Recognition*, J. Kittler, Ed. August 2004, vol. 1, pp. 313–317, IEEE.

[16] E. Bailly-Bailliere and al., "The banca database and evaluation protocol," in *4th International Conference on Audio- and Video-Based Biometric Person Authentication, Guildford, UK*, Berlin, June 2003, vol. 2688 of *Lecture Notes in Computer Science*, pp. 625–638, Springer-Verlag.

[17] H. Schneiderman and T. Kanade, "A statistical approach to 3d object detection applied to faces and cars," in *International Conference on Computer Vision*, 2000.