FACE DETECTION USING COARSE-TO-FINE SUPPORT VECTOR CLASSIFIERS

Hichem Sahbi
Imedia Research Group
INRIA, BP 105, 78153
le Chesnay ,France
Hichem.Sahbi@inria.fr

Donald Geman
Johns Hopkins University
3400 N. Charles Street
Baltimore, MD 21218
geman@cis.jhu.edu

Nozha Boujemaa
Imedia Research Group
INRIA, BP 105, 78153
le Chesnay, France
Nozha.Boujemaa@inria.fr

ABSTRACT

We describe a new face detection algorithm based on a hierarchy of support vector classifiers (SVMs) designed for efficient computation. The hierarchy serves as a platform for a coarse-to-fine search for faces: most of the image is quickly rejected as "background" and the processing naturally concentrates on regions containing faces and face-like structures. The hierarchy is tree-structured: In proceeding from the root to the leaves, the SVMs gradually increase in complexity (measured by the number of support vectors) and discrimination (measured by the false alarm rate), but decrease in the level of invariance. Reduced complexity is achieved by clustering support vectors and shifting the decision boundary in order to satisfy a "conservation hypothesis" that preserves positive responses from the original set of support vectors. The computation is organized as a depth-first search and cancel strategy. The gain in efficiency is enormous.

1. INTRODUCTION

Face recognition is becoming a key ingredient, and challenge, for many applications, such as authentication and database indexing. These applications require very rapid face detection (and perhaps accurate pose estimation in order to extract the face) due to time limitations and the large amount of data. Several methods for face detection are discussed in the literature, including artificial neural networks [1], support vector machines [2], Bayesian inference [3], deformable templates [4], graph-matching [5] and skin color learning [6]. In this paper, we present a hierarchical face detection algorithm based on SVM classifiers. As in [7], the goal is to quickly reject background subimages and focus the processing on faces and face-like structures. Using the same "pose decomposition" as in [7], we obtain an accurate face detector based on a tree-structured network of SVM classifiers which is also fast due to using only very crude SVMs at the beginning followed by a steady increase in complexity (measured by the number of support vectors).

2. POSE DECOMPOSITION

We denote by $\theta = (p, \phi, s)$ the pose (position, tilt and scale) of a face. The set of poses is recursively subdivided, resulting in a nested family of partitions; Λ denotes a generic cell. Building the hierarchical detector necessitates training an SVM classifier for each pose set Λ, whose cost increases as the size of Λ decreases. In addition, the SVM dedicated to Λ is, in principle, invariant to Λ in the sense of responding positively for face presentations with pose in Λ. (cf. Fig.1.(B)). The definition of the sets Λ is similar to that in [7] and the training data for Λ is produced by randomly generating a certain number of faces images at various appearances from Λ starting from an initial dataset.

The face position p is taken as the midpoint between the eyes, the scale s as the distance between the eyes and the tilt ϕ is relative to the axis perpendicular to the segment joining the eyes. A scene is processed by visiting *non-overlapping* 16×16 blocks and processing the surrounding image data to detect all faces whose position falls in the block and whose scale s lies in the interval $[10, 20]$; the range of tilts is $[-20, +20]$. Faces at scales $[20, 160]$ are detected by repeated down-sampling of the original image, once for scales $[20, 40]$, twice for $[40, 80]$ and thrice for $[80, 160]$.

3. FEATURES, LEARNING AND SVMS

Throughout this section our objective is to build a SVM for a given pose cell Λ, based on the related training examples. The full set of pose cells is described in §4. We refer to the hierarchical family of classifier where each node contains an SVM classifier f as the **f-network**.

3.1. Features and learning

The basic training for SVMs [8] involves finding an linear hyper-plane which optimizes generalization capability, i.e., performance on unseen examples. We are given l observations $x_i \in R^n$ with associated labels $y_i, i = 1, ..., l$. The

representation x_i of a face is the 8×8 array of low frequency coefficients of the Daubechies wavelet transform for a 64×64 subimage; the subimage contains the face in the sense that the position p falls in the 16×16 block centered in the subimage. The label y_i is positive if the reference window contains a face, strictly negative otherwise. Our objective is to train a mapping $x \mapsto y = f_\Lambda(x, \alpha)$ for a vector of parameters α, where $f_\Lambda(x, \alpha) = \sum_{i=1}^{N_S} \alpha_i . y_i . K(x, x_i) + b$.

3.2. The f-network vs. the g-network

SVM classifiers have proven to have a good generalization capacity and be easily trained. They do, however, have the disadvantage of a prohibitive online cost (evaluation of the decision function), at least with respect to many other classifiers. Consequently, evaluating the f-network on each pattern x is very costly [1]. Various proposals have been made to reduce the complexity of the form of an SVM decision boundary. Burges et al [9] introduced the "reduced set" technique which generates a set of support vectors and associated weights in an optimization framework, but more complex than the original one since the minimization is carried out in a space of dimension $d \times N_R$, where d is the dimensionality of the data and N_R is the required number of support vectors ("the simplification"). Our approach is to learn, for each pose cell Λ, a simplified SVM decision function g_Λ which depends on the training set for Λ and whose complexity depends on the level (depth) $\mathcal{L}(\Lambda)$ in the hierarchy. We refer to this simplified hierarchy of SVMs as the **g-network**. Since we use a coarse-to-fine strategy for evaluation of the network, the decision functions at the upper levels of the g-network must be rapidly computed and yet respond negatively - and hence "cancel" the evaluation of all finer SVMs in the associated subtree - in the majority of cases. This is possible since only a small percentage of the subimages visited actually have faces.

3.2.1. *Building the g-network*

Again, our objective is to build a simple classifier g_Λ for separating two classes: subimages with and without a face *with pose in* Λ. Let N_f and N_g denote, respectively, the number of support vectors for f_Λ and g_Λ. Since we will choose $N_g << N_f$, g_Λ will be evaluated more efficiently than f_Λ. Let $\Sigma_{f_\Lambda} = \{x_1, ..., x_{N_f}\}$ be the set of support vectors obtained after standard training using samples of faces with poses in Λ. In what follows, we aim to create a new, smaller set of support vectors $\Sigma_{g_\Lambda} = \{z_1, ..., z_{N_g}\}$, called the reduced set. The simplified SVM for Λ is then of the form $g_\Lambda(x, \gamma) = \sum_{k=1}^{N_g} \gamma_k . y_k . K(x, z_k) + b'$.

[1] Not all the SVMs in the network are in general evaluated; there is a coarse-to-fine process which starts with the SVM at the top of the hierarchy (network) and then performs a depth-first search; usually very few of SVMs are evaluated. See §4 and [7].

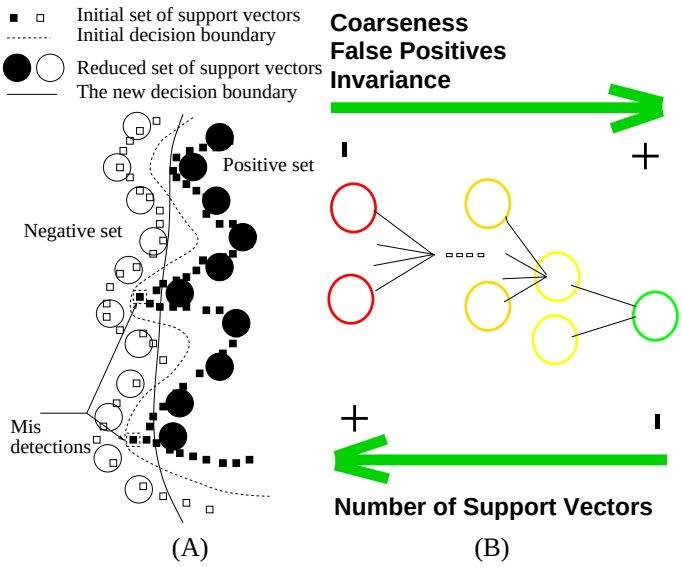

Fig. 1. (A) Simplification of the decision function leads to missed detections and false alarms. (B) The hierarchical description of the g-network in terms of cost, invariance and false alarms.

We use the clustering algorithm in [10] to first cluster the original set of support vectors in terms of spatial proximity. The candidates for support vectors in the reduced set are the cluster centers. Let C be the number of centers after clustering the set Σ_{f_Λ} and let $\mathcal{L}$ be the depth of Λ in the hierarchy, with $\mathcal{L} = 0$ for the root cell and $\mathcal{L}$ equal to the maximum (tree) depth for a leaf cell, say n. Then we take $C = N_f / 2^{n-\mathcal{L}}$ as an upper bound on the size of Σ_{g_Λ}. Thus the root SVM in the g-network has at most $N_f / 2^n$ support vectors *whereas the SVMs at the leaves of the g-network are the same as the SVMs at the leaves of the f-network*, i.e, there is no simplification for the finest pose cells. In addition, for non-leaf cells deep in the hierarchy the modeling capacity is little changed, i.e., degrades slowly with the reduction in the number of support vectors.

3.2.2. *The conservation hypothesis*

One important difference between g_Λ and f_Λ, particularly near the root, is that without further modifications, g_Λ is in general no longer an invariant for faces with poses in Λ (cf. Fig.1.(A)). The conservation hypothesis we impose is that, for any given pattern x, if $f_\Lambda(x, \alpha)$ is positive then $g_\Lambda(x, \gamma)$ must also be positive. Notice that positivity of f_Λ for "large" Λ by no means signals a face as there is a non-trivial false alarm rate. On the other hand, we can assume that $f_\Lambda(x, \alpha) \geq 0$ for nearly all face patterns x. Let $\{train\}_\Lambda = \{train\}_\Lambda{}^+ \cup \{train\}_\Lambda{}^-$ and (resp. $\{test\}_\Lambda$) be the training (resp. testing) set for poses in Λ. We intro-

duce the following measures of efficiency for g_Λ:

$$\rho_\Lambda = \frac{\#\{x \in \{train\}_\Lambda \;/\; f_\Lambda(x,\alpha) \geq 0 \;\&\; g_\Lambda(x,\gamma) \leq 0\}}{\#\{x \in \{train\}_\Lambda \;/\; f_\Lambda(x,\alpha) \geq 0\}}$$

$$\beta_\Lambda = \frac{\#\{x \in \{test\}_\Lambda \;/\; f_\Lambda(x,\alpha) \in [0,1] \;\&\; g_\Lambda(x,\gamma) \leq 0\}}{\#\{x \in \{test\}_\Lambda \;/\; f_\Lambda(x,\alpha) \in [0,1]\}}$$

$$\eta_\Lambda = \frac{\#\{x \in \{train\}_\Lambda^- \;/\; g_\Lambda(x,\gamma) \leq 0\}}{\#\{x \in \{train\}_\Lambda^-\}}$$

We regard the fractions ρ_Λ and β_Λ as the empirical training and testing risks, respectively, relative to positive examples. A null empirical training risk guarantees that all the face training examples are classified as positive using g_Λ. The empirical testing risk is a measure of error in terms of the fraction of test face examples which are mis-classified by g_Λ among those belonging to the observation region between positive support vectors and the decision boundary under f_Λ. Finally, the factor η_Λ is an empirical measure of "background rejection efficiency" (statistical power) of g_Λ.

3.2.3. Bias variation

We want each g_Λ to satisfy the conservation hypothesis *and* have as high a background rejection efficiency η_Λ as possible. We introduce a *bias variation* technique to enforce the conservation hypothesis, at least empirically (i.e., $\rho_\Lambda = 0$), by subtracting (if necessary) a bias σ_Λ from g_Λ in order to force the classifier to respond positively to all the face training examples (belonging to the Λ) at the expense of a reduction in background rejection efficiency.

Let S_Λ^- be the set of negative support vectors for f_Λ. We consider three possible choices $\sigma_{1\Lambda}, \sigma_{2\Lambda}, \sigma_{3\Lambda}$ for σ_Λ. In each case, we define $\sigma_{j\Lambda} = 0$ if there are no misclassified face training examples. Otherwise they are given by:

$$\sigma_{1\Lambda} = min\,\{g_\Lambda(x_i) \;/\; x_i \in \{train\}_\Lambda^+\} \tag{1}$$

$$\sigma_{2\Lambda} = min\,\{g_\Lambda(x_i) \;/\; x_i \in S_\Lambda^- \cup \{train\}_\Lambda^+\} \tag{2}$$

$$\sigma_{3\Lambda} = median\,\{g_\Lambda(x_i) \;/\; x_i \in S_\Lambda^- \cup \{train\}_\Lambda^+\} \tag{3}$$

Clearly $\sigma_{1\Lambda}$ is the smallest negative bias in $\{train\}_\Lambda^+$, and represents the negative of the maximum distance from the decision boundary for g_Λ to a (misclassified) positive training example. Thus, $g_{1\Lambda}(x) = g_\Lambda(x) - \sigma_{1\Lambda}$ will classify as positive all examples in ${\{train\}_\Lambda}^+$. In this case, the background rejection efficiency is high. However, since we have shifted the decision boundary by the minimum amount necessary to guarantee the conservation hypothesis (null ρ_Λ), the empirical testing risk β_Λ is relatively high, which means $g_{1\Lambda}$ is likely to miss some unseen faces examples, mainly those which reside in the margin space. Using the bias $\sigma_{2\Lambda}$ preserves a null empirical testing risk but significantly decreases η_Λ, which results in an inefficient classifier $g_{2\Lambda}(x) = g_\Lambda(x) - \sigma_{2\Lambda}$ in terms of background rejection. The bias $\sigma_{3\Lambda}$ offers a good balance between η_Λ and β_Λ with $\rho_\Lambda = 0$.

3.2.4. The coarse-to-fine search strategy

Given a pattern x, an encoded subimage using the Daubechies wavelet transform, we apply a depth-first search (and cancel) strategy as described in [7] in order to generate the final answer, face or background, of the network. The global classifier, i.e., the g-network, declares a face if and only if there is at least one complete "chain" of positive responses from the root SVM to a leaf SVM. The search terminates upon finding a positive chain and the pattern x is classified as a face with pose given by an average from the leaf cell. Equivalently, the network responds negatively if an only if there is a "null covering" of the hierarchy in the sense of a collection of negative responses whose corresponding cells cover all poses; the search is terminated upon finding such a null covering. Thus, for example, if $g_\Lambda \leq 0$ for the root cell Λ, the search is terminated as there cannot be a chain of ones.

4. EXPERIMENTS

In this section, we provide an evaluation of the coarse-to-fine g-network classifier in terms of background rejection efficiency, detection rate and processing time. We collected 100 images from the CMU database and 100 images from the online CMU web demo. Experiments involve scenes with frontal views of faces, and we use for training the Olivetti database containing 400 faces of 40 individuals which are used to synthesize 2000 faces initially for each cell detector. We note that this training set is at least an order of magnitude smaller than those typically used for training SVMs and neural networks.

4.1. Generalization

By the conservation hypothesis we know there is a chain of positive responses for both networks for every positive training example. We cannot of course guarantee that every (unseen) positive example is classified as a face by either the f-network or the g-network, or that positive classification by the f-network implies the same by the g-network. *However*, in our experience, the false negative error rate of each SVM in the f-network is extremely small, which means there will be a chain of ones for nearly every face. Moreover, in such a case, we nearly always observe a chain of positive responses in the g-network as well. In particular, the false negative rate of the g-network is quite reasonable (see 4.2). These tradeoffs will be explored in more detail elsewhere.

4.2. Background rejection and face detection efficiency

We estimate the background rejection efficiency $BRE_\mathcal{L}$ for each level $\mathcal{L}$ by the fraction of the visited patterns which

Table 1. Background rejection efficiency ($BRE_{\mathcal{L}}$) and face detection rates ($FDR_{\mathcal{L}}$) of the SVM network for different levels $\mathcal{L}$ in the hierarchy.

Level $\mathcal{L}$ in the hierarchy	$BRE_{\mathcal{L}}$	$FDR_{\mathcal{L}}$	Mean number of SVs
0 (Coarsest cell)	77 %	100 %	19
1 (Translation split)	82 %	100 %	39
2 (Translation split)	86 %	100 %	79
3 (Translation split)	88 %	100 %	158
4 (Rotation split)	91 %	100 %	321
5 (Scale, finest split)	96 %	87.6 %	622

were rejected by the g-network (cf. Table.1). We also estimated the face detection rate $FDR_{\mathcal{L}}$ by the fraction of face patterns which were classified as positive by at least one detector belonging to the level $\mathcal{L}$. In what follows, $\mathcal{L}$ ranges from 0 (the root) to the depth of the tree. It can be seen from the previous table that missed face rates are 0% at every level in the hierarchy except at the finest one.

In our experiments, 77% of the traversed patterns from natural images were rejected by the coarsest SVM in the g-network, which has only 19 support vectors; this is 30 times faster than computing the corresponding f-classifier. Recall that the two networks coincide at the leaf cells.

Fig. 2. Detected and localized faces.

The coarse-to-fine classifier detects 87.6% of the faces in our testing set, which is roughly similar to the performance of other methods, perhaps somewhat worse. The false alarm rate per visited pattern (45×10^{-5}) is quite low, and corresponds on average to 2 or 3 false alarms per image. The g-network is far more efficient than the f-network as it requires 8(s) whereas the f-network requires 120(s) to process an image of dimension 450×350 using a 450 Mhz PentiumII. This is faster than [1] and [2] and slower than [7]. Other measures of efficiency should also be considered, such as the training complexity; one must solve a constrained quadratic programming problem in a dimension equal to the cardinality of the training set. Many buffering techniques, such as those in [2], can be applied to increase the training set for a particular pose set. Obviously, there are many tradeoffs to consider in terms of the training cost, the number of support vectors and error rates.

5. CONCLUSION AND FUTURE WORK

We presented a new "coarse-to-fine" face detection algorithm based on SVM classifiers. Many techniques in the literature perform face detection using a pose-dedicated classifier which is applied to all possible locations, scales and rotations. This exhaustive search is very expensive and inefficient. In contrast, the coarse-to-fine exploration of face presentations was performed by building increasingly complex SVM decision functions for increasingly more constrained sets of poses. Clustering and "bias variation" techniques were introduced in order to build optimized decision boundaries for each cell for a given number of support vectors. Bias variation allows one to impose a "conservation hypothesis" about the empirical risk relative to positive examples and at the same time to reject as many background samples as possible. We are currently investigating various tradeoffs, such as those between background rejection efficiency and the number of support vectors, among various training parameters, and between the performance of the g-network and f-networks. We are also investigating extending this coarse-to-fine framework to include other aspects of the presentation of a face (3D pose, partial occlusion,etc).

6. REFERENCES

[1] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23–38, 1998.

[2] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," *In IEEE Int Conference on Computer Vision and Pattern Recognition*, pp. 130–136, 1997.

[3] T.F.Cootes and C.J.Taylor, "Constrained active appearance models," *In International Conference on Computer Vision*, pp. 748–754, 2001.

[4] J. Miao, B. Yin, K. Wang, L. Shen, and X. Chen, "A hierarchical multiscale and multiangle system for human face detection in complex background using gravity center template," *In Pattern Recognition*, vol. 32, no. 7, pp. 1237–1248, 1999.

[5] T. Leung, M.C. Burl, and P. Perona, "Finding faces in cluttered scenes using random labelled graph matching," *In International Conference on Computer Vision*, pp. 637–644, 1995.

[6] R.L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *In IEEE International Conference on Image Processing*, pp. 1046–1049, 2001.

[7] F. Fleuret and D. Geman, "Coarse-to-fine visual selection," *In Int Journal of Computer Vision*, vol. 41, no. 2, pp. 85–107, 2001.

[8] V. N. Vapnik, "The nature of statistical learning theory," *Springer Verlag*, 1995.

[9] C. Burges and B. Scholkopf, "Improving the accuracy and speed of support vector machines," *Neural Information Processing Systems, Cambridge. MIT Press*, pp. 375–381, 1997.

[10] Rajesh N. Dave, "Characterization and detection of noise in clustering," *In Pattern Recognition*, vol. 12, no. 11, pp. 657–664, 1991.