Boosted Detection of Objects and Attributes

Abstract

We present a new framework for detection of object and attributes in images based on boosted combination of primitive classifiers. The framework directly minimizes the detection error by learning a set of simple, computationally efficient threshold-based detectors. We apply this framework to segmentation of human skin and detection of faces in images. We show that despite its simplicity the method performs on par with more complex traditional models in detection accuracy while outperforming them in scalability. This can be especially beneficial in applications of the framework to real-time detection tasks.

1. Introduction

Statistical models for detection of objects and their attributes in images have been popular for a number of years. Different models of varying degrees of complexity have been proposed and utilized in tasks such as edge and color detection [1, 2, 3, 4, 5], and face detection and recognition[6, 7, 8, 9], to name just a few. A common thread of these approaches is that their improved performance usually came at the cost of high computational and algorithmic complexity. Similarly, computationally simpler techniques often required large amounts of data to compensate for the weaknesses of the model. Examples of these trends may be seen in, for instance, face detectors of Rowley et al. [8] and skin models of Jones and Rehg [5].

Recent developments in the theory of machine learning and applied statistics have shown that simple combinations of weak models (learners, classifiers, regressors) may have a surprisingly effective combined performance. Techniques such as *bagging* and *boosting*[10] have proved both practically and theoretically that particular choices of combining simple statistical learners may significantly reduce not only the training error but also the more elusive generalization error.

In this paper we show how one such technique can be utilized for the task of object detection. We first describe a general theory of boosting, motivated by a number of its appealing theoretical properties. We then propose a *boosted* use of simple thresholding classifiers that leads to computationally simple yet highly accurate detectors of objects and their attributes in images. Finally, we demonstrate the utility of our approach by constructing simple boosted detectors for the problems of color-based image segmentation and face detection.

2. Boosted Combination of Classifiers

Boosting (c.f. [10]) is a learning technique which has caught attention in the statistical and machine learning communities. The Boosting algorithm works by sequentially training a series of weak learners which are then combined into a single strong classifier. Each weak learner (classifier) attempts to minimize classification error on a particular distribution of the training data. Freud and Schapire [10] proved that the boosted combination of classifiers not only minimizes the empirical classification error ("error on the training set") but also leads to a minimized bound on the true Bayesian error ("error on the test set.") Various classifiers of different complexity such as naive Bayes, decision trees and Bayesian networks, have been used in place of the weak learner and have all lead to significantly improved (*boosted*) final classification performance in a number of empirical studies.

Consider a supervised binary classification problem with training data given by $S = \{(x_1, y_1), ..., (x_m, y_m)\}$, where x_i is the feature vector and y_i is its label, $y_i \in \{-1, +1\}$. The goal of the learning algorithm is to solve for a classifier $h : X \rightarrow Y$ that minimizes the generalization error

$$\epsilon_{gen} = \Pr_{D}[h(x) \neq y],$$

where D is the distribution of the data (x, y). However, direct minimization of ϵ_{gen} is impossible and one attempts to minimize the empirical error

$$\epsilon_{emp} = \Pr_{\hat{D}}[h(x) \neq y] = \sum_i L_{0-1}(x_i, y_i)$$

where $\hat{D}$ now denotes the empirical distribution on the training set and $L_{0-1}(x, y)$ is the $0-1$ classification loss defined as

$$L_{0-1} = \begin{cases} 0 & , yh(x) \geq 0 \\ 1 & , yh(x) < 0 \end{cases}$$

Nevertheless, minimization of the empirical error remains a difficult task. Schapire and Freud [10] instead suggest minimization of an upper bound on the classification $0-1$ loss

$$L_{0-1}(x, y) \leq L_{ab} = \exp(-yh(x))$$

This task becomes feasible and can be accomplished using the *Adaboost* algorithm:

ADABOOST

1. Given $S = \{(x_1, y_1), ..., (x_N, y_N)\}$; $y_i \in \{-1, +1\}$ initialize

$$D_1(i) = 1/N.$$

2. For $k = 1, ..., K$

   (a) Train classifier h_k using data S and distribution D_k.

   (b) Choose

$$\alpha_k = \frac{1}{2} \ln \left(\frac{1 - \epsilon_k}{\epsilon_k} \right)$$

   where $\epsilon_k = Pr_{D_k}[h_k(x_i) \neq y_i]$.

   (c) Update

$$D_{k+1}(i) \propto D_k(i) \exp(-\alpha_k y_i h_k(x_i)).$$

3. The final hypothesis is

$$H(x) = sign \left(\sum_{k=1}^{K} \alpha_k h_k(x) \right).$$

Adaboost algorithm starts by assigning equal weight to all training examples. In each iteration, the weight of the misclassified samples is increased and the weight of correctly classified samples is decreased. In other words, each successive classifier focuses on samples not learned by the previous models. The weighting factor on data at step k depends on the expected error ϵ_{k-1} made by the previous classifier. After K iterations the final classifier is constructed as a linear combination of individual K models. The weight given to each classifier in the combination is proportional to the empirical error the classifier makes on the training set.

As mentioned before, the Adaboost algorithm can be shown to minimize a bound on the empirical error. If the weights (α_k) are chosen in the way described above, the training error is bounded by

$$\prod_k \left[2\sqrt{\epsilon_k(1 - \epsilon_k)} \right] \quad (1)$$

One can note that, if each of the individual weak classifiers performs at least slightly better than the random, the training error decreases exponentially. Schapire et al. [10] have also shown that the generalization error of the final hypothesis ϵ_{gen} is bounded by

$$\epsilon_{gen} \leq \Pr_{\hat{D}}(y \sum \alpha_k h_k(x) \leq \theta) + O\left(\sqrt{\frac{d}{N\theta^2}} \right)$$

for any positive θ and d (the VC-dimension of h.) $y \sum \alpha_k h_k(x)$ can be viewed as the *margin* of point (x, y) and Adaboost indeed maximizes the margin of the combined classifier. Empirically it has been shown that Adaboost is able to generalize well for any reasonable K.

We next show how boosting can be used to construct highly accurate yet simple detectors of objects and their attributes in images. We employ simple thresholding functions as the weak classifiers h_k. Training of each weak classifier now reduces to optimal selection of its threshold. We demonstrate the utility of the boosting approach on two common computer vision tasks: color-based image segmentation and face detection.

2.1 Boosted Skin Color Segmentation

Segmentation of images into skin color regions and non-skin color regions is a common practical problem in computer vision. Skin segmentation is often the first step in solving more complex vision problems. For example, one can use skin color-based image segmentation to localize different body parts and then use more complex algorithms in just those portion of the images to do improved tracking, detection of hands and faces. However, because most of this task relies on the performance of the basic color based segmentation technique, it becomes imperative that the method adopted is accurate. At the same time, it is also necessary to have a computationally efficient technique as one cannot afford a high overhead from the basic segmentation.

A number of skin detection techniques have been suggested in literature (c.f. [1, 2, 3, 4, 5]). Among the most successful are learning-based statistical techniques. In general the techniques can be divided into two categories: parametric, such as Gaussians mixtures, and non-parametric, such as histograms. Gaussian mixture methods [1, 2] work well for smaller amounts of training data (provided they are sufficient to learn the parameters) but do not show good generalization. Moreover, complex Gaussian models require overhead in floating computations which can make them impractical for implementation on, for instance, hand held devices. Histogram-based techniques [5] exhibit very good performance, but they require large amounts of training data and fine bin size.

Boosting provides an alternative to these methods by reducing both the memory and the computational requirements. By optimally combining a set of simple skin color

classifiers, boosted skin detector optimizes skin detection rate on the training set of images and, at the same time, generalizes well to unseen data.

In our implementation, boosted skin detector classifies each image pixel as either skin or non-skin. The detector consists of a number of weak, threshold-based detectors:

$$h_k \in \left\{ t(\mathbf{x}|\theta, c) = sign(x^{(c)} - \theta) \right\}.$$

Here, $\mathbf{x}$ denotes the image pixel, θ is the decision threshold, and c selects the color component of the pixel, for instance $c \in \{R, G, B\}$. Therefore, each threshold detector takes one color component of the pixel as its input, compares it against threshold θ and labels it as the skin or non-skin. Optimal choice of θ, or learning of weak classifiers in the boosting framework, corresponds to the threshold θ^* and component c^* that yield the lowest misclassification rate on the training set of pixels.

In this work we assume discrete-valued color components, hence a fixed set of possible thresholds, $1, .., Q$, suffices. For instance, Q can be 255 or some other integer. For a C component color space ($C = 3$ for RGB, for instance) the space of all possible threshold functions has Q^C elements. However, since our method is treating components independently of each other, the effective space reduces to CQ. The boosted color detector algorithm is now simply the ADABOOST algorithm of Section 2 with the following training step:

SKINBOOST

2(a) Select a threshold function among CQ possible functions which minimizes[1] the classification error on the training set, $h_k = t(\theta_k^*, c_k^*)$, such that

$$(\theta_k^*, c_k^*) = \arg\min_{\theta, c} \sum_i D_k(i) \left[t(\mathbf{x}_i|\theta, c) \neq y_i \right],$$

where y_i is 'skin' or 'non-skin'.

The upper limit on the number of weak classifiers K is approached when the training error falls to zero or when each subsequent weak detector fails to yield further improvement ($\epsilon_k = 0.5$).

[1] It can be easily shown that the error of this classifier is less than 50%, provided we also allow negating the output of the threshold function. This means that one effectively needs to double the number of weak detectors to $2 \times Q * C$, corresponding to $t(x)$ and $-t(x)$.

Boosted skin detector $H(x)$ obtained in this fashion relies on the superposition of weighted thresholding functions. An example of the complex threshold functions is shown in Figure 1. Note that the final boosted detector is a

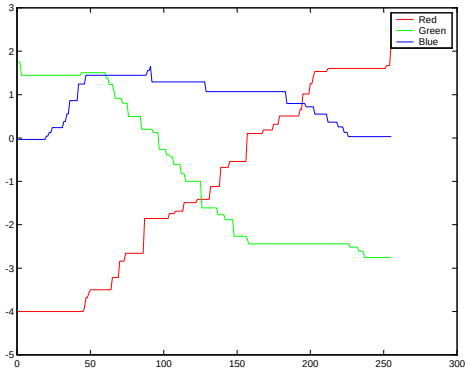


Figure 1: Boosted color detector. Shown are linear combinaitons of weak detectors (thresholding functions) assigned to R, G, and B channels, respectively. Final detector $H(x)$ is a weighted and then thresholded combination of the three (R,G,B) functions. Horizontal axis corresponds to intensities in the three channels.

thresholded linear combination of the three functions. This superposition, in turn, results in *adaptive* partitioning of color space into regions assigned to one of the two classes, skin and non-skin.

Even though the classifiers at each step of this algorithm are trivial and the optimization/learning can be accomplished by a simple exhaustive search, the combined performance of the boosted model is very good. Moreover, detection process simply involves evaluation of N threshold functions which can be efficiently implemented using C lookup tables. In the next section we present results of skin color segmentation using our classification technique. We also include a comparison with both the Gaussian mixtures and the histogram models in two different color space.

2.2 Boosted Face Detection

Face detection is another important basic task often encountered in computer vision. A number of applications have been developed that require recognition of human faces. Even though face detection is a difficult task and various methods that have been used are often complex, we show how a simple boosted algorithm can be used to accurately

and efficiently solve this detection task.

A number of models, primarily statistical, have explored various ways of modeling faces [6, 7, 8, 9]. Unlike the task of color-based segmentation which imposes little if any spatial constraints on the model, detection of faces also relies on spatial constraints. To extend the proposed boosting framework to detection of faces we consider a set of simple weak classifiers of the following form:

$$h_k \in \left\{ t(\mathbf{X}|\theta, l) = sign(X^{(l)} - \theta) \right\}.$$

$\mathbf{X}$ now denotes a vectorized image of gray-scale pixel values and $X^{(l)}$ is its $l-th$ component (pixel). Analogous to the color detection task, each threshold detector takes one image pixel as its input, compares it against threshold θ and labels it as the "face" or "non-face". Learning of the weak classifiers now corresponds to the threshold θ^* and pixel l^* that yield the lowest rate of error on the training set of images. The rest of the face detection algorithm assumes the same general form of ADABOOST where the weak classifier learning step becomes

FACEBOOST

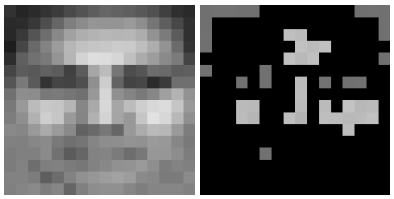
2(a) Select a threshold function which minimizes the classification error on the training set, $h_k = t(\theta_k^*, l_k^*)$, such that

$$(\theta_k^*, l_k^*) = \arg\min_{\theta,l} \sum_i D_k(i) \left[t(\mathbf{X}_i|\theta, l) \neq y_i \right],$$

where y_i is 'face' or 'non-face'.

Even though it seems trivial for the task of face detection, experimental results in the next section indicate that its performance is better than that of some complex classifiers such as support vector machines.

Boosted face detection algorithm selects an optimal subset of image pixels $\{l_k^*\}$ and thresholds $\{\theta_k^*\}$ corresponding to those pixels that best discriminate faces from non-faces, while minimizing the detection error. Figure 2(a), shows an example of an average face obtained from training data. Figure 2(b) shows a typical "face" image sampled from the function learned using boosting. Each non-white location corresponds to a pixel selected by the boosting algorithms. Intensity of each selected pixel is in turn determined by a linear combination of the thresholding functions. Similar to the case of the color classifier, the selected pixel may have one or more such thresholding functions that specify ranges of intensities associated with faces and non-faces. The example images indicates that the pixels selected by the algorithm are the ones which correspond to the important features of the face (eyes, mouth, nose and hair). This follows our intuition and agrees with the average face shown in Figure 2(a).



(a) Average face (b) "Boosted face"

Figure 2: Points on the "boosted face" align with most descriptive features on the average face. "Boosted face" was obtained by sampling from the learned boosted threshold model.

It is once again important to stress that the boosted face detector selects only a *subset* of all image pixels. This clearly reduces the computational complexity of detection because only a small number of pixels needs to be examined. Moreover, computation of the threshold function can be implemented using a fast lookup table.

In the next section we present details of the detection experiments. We compare performance of the boosting method with standard methods like SVM and nearest neighbor classifier.

3. Experiments

We conducted two series of experiments to evaluate the power of our boosted detector models. The first set of experiments involved segmentation of images into skin and non-skin color regions. The second set of experiments dealt with the face detection.

3.1. Skin Color Segmentation

The algorithm from Section 2.1 was used for skin color segmentation. We have compared its performance to two standard skin color models based on histograms and Gaussian mixtures.

We selected a boosted skin detector with 16 threshold levels in each color channel. Comparable histogram and Gaussian mixture models had 16 uniform bins in each channel and three mixture components, respectively. Finally, we considered three typical color spaces: RGB, normalized RGB, and HSV. All models were trained and evaluated on the same dataset of 1200 images. Ground truth labeling of all image pixels in the data set was done manually.

Method	Err	Det	False Pos
Boosting	11.52	89.58	12.63
Mixture of Gaussians	11.49	89.28	12.26
Histogram	12.45	91.37	16.27

Table 1: Percentage error, detection rate, and false positives (per pixel) in RGB space.

Figure 3.1 shows an example of the empirical error behavior on the training and the test sets during training. As expected, the test error continued to decrease even when the training error became stationary, indicating good generalization performance of the model.

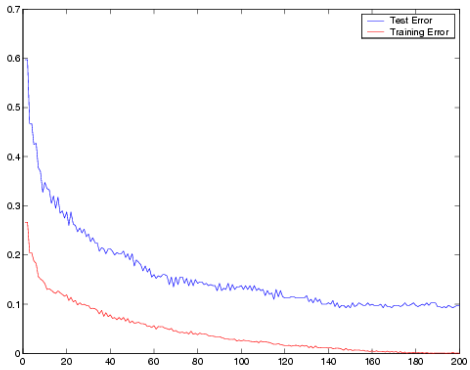


Figure 3: Error of classification on the training and test sets as a function of the number of weak detectors. Note that the test error continues to decrease even when the training error reaches a plateau, as the number of weak detectors increases.

Skin segmentation results are summarized in Tables 3.1 through 3.1. We observe that in each case the boosted detector compares favorably in error rate and false positive rate to both the histogram method and the mixture of Gaussian. This suggests that a simple boosted detector may be the desired choice among the three models given its low computational complexity and memory requirements.

Finally, we applied our boosted skin detector to a number of arbitrary color images from the web. Figure 4(b) illustrates very good skin segmentation results obtained on one of those images.

Method	Err	Det	False Pos
Boosting	10.95	90.77	12.69
Mixture of Gaussians	12.61	94.15	19.37
Histogram	12.45	91.81	16.71

Table 2: Percentage error, detection rate, and false positives (per pixel) in normalized RGB space.

Method	Err	Det	False Pos
Boosting	9.51	93.20	12.27
Mixture of Gaussians	10.96	87.52	9.44
Histogram	12.71	91.40	16.82

Table 3: Percentage error, detection rate, and false positives (per pixel) in HSV space.

3.2. Face Detection Experiment

We conducted a set of preliminary experiments to evaluate the boosted face detector. Our training set contained 1400 frontal face images cropped out of images downloaded from the web. Each image was rescaled to a 16x16 window and normalized for intensity. We randomly chose another 1400 images which did not contain any faces. Part of the data was used for training and the rest was employed for testing (five fold crossvalidation was done to obtain consistency in the performance measure.)

Table 3.2 outlines classification performance for four different classifiers: boosted face detector, support vector machine (SVM) with a linear kernel function, SVM with an RBF kernel, and a nearest neighbor classifier. It is evident that the boosted detector outperforms two of the classifiers and performs on par with the complex RBF SVM in the number of false positives. Although nearest neighbor classifier gave the best detection performance it also had a very high false positive rate. The error variance in all cases was less that 1%, which indicates significance of our results.

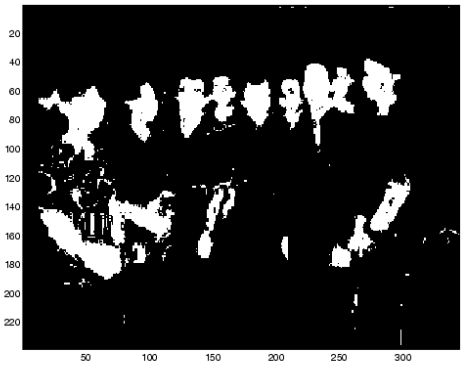
Figures 5(a) and 5(b) depict results of applying the boosted face detector on two arbitrary web images selected to contain multiple faces.

Again, the complexity of the boosted classifier is less than that of the most other classifiers. Note also that the

Method	Detection	False Positives
Boosting	95.13	9.87
SVM (linear)	85.33	33.71
SVM (RBF)	93.0	10.25
Nearest Neighbor	96.61	35.04

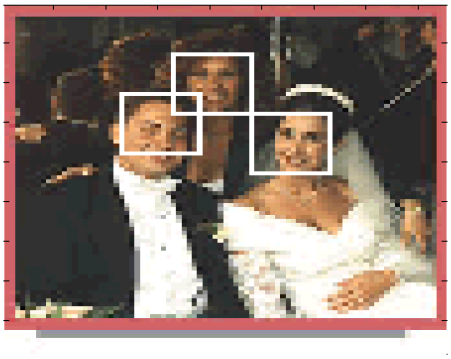
Table 4: Percentage error in terms of percentage of faces correctly detected and the percentage of faces misclassified.

(a) Original

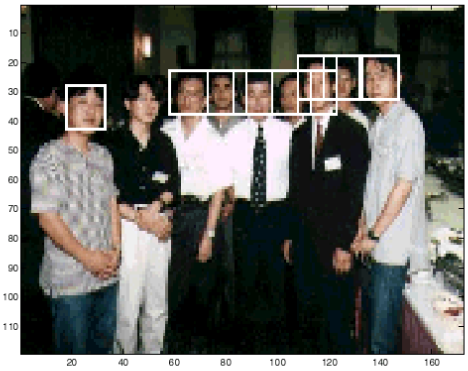


(b) Skin color mask

Figure 4: Detection of skin color in an arbitrary web image using the boosted skin detector.



(a)



(b)

Figure 5: Detection of faces in arbitrary web images using the boosted face detector.

boosted classifier facilitates efficient implementation of the multiscale search because the evaluation is limited to a *subset* of image pixels.

4. Discussion

We have shown in the previous sections that optimal combination of simple classifiers, determined by boosting, even in a basic state space (e.g., RGB or normalized intensities) can lead to detectors whose performance is comparable to that of some complex classifiers. One reason for this may lie in the complexity of the state space partitions obtained in this fashion. Our framework allows multiple simple classifiers to form complex partition of the state space, adapted to the statistics of the training set but, also, with some guarantees with respect to the test set. These guarantees indeed seem to hold well for the set of experiments we conducted—for instance, performance of both skin and face detectors remains high over relatively adverse lighting conditions present in arbitrary web images. The use of more complex state spaces ("features") may slightly improve detection performance in terms of (generalized) detection error, but it comes at the additional cost in computational complexity. This may hinder applicability of these detectors in low-resource applications (e.g., handheld devices.)

The inherent performance score of the detector is the symmetric total classification error, based on the L_{0-1} symmetric loss. As such, the score equally penalizes all types of errors (false and true positives and negatives). We have empirically shown that, in spite of that, our detectors exhibit a good false positive performance, indicating their optimality and good generalization properties. Unfortunately, basic formulation of the boosting theory does not allow one to immediately utilize non-symmetric loss functions and, thus, exhibit control over individual error types. This is one reason for the lack of ROC evaluation, commonly seen in other detection studies.

5. Previous Work

Boosting theory has begun to find its application in computer vision in recent years. For instance, Viola et al. [11] and Pavlovic et al. [12] have both used boosting to combine classifiers of different complexity. Viola's work focused on image search in large databases. They used complex filters and boosting of a single threshold classifier to generate discriminative features. On the other hand, in our framework we allow multiple thresholding functions to generate more complex partitions of the simpler state space. Pavlovic et al. used dynamic Bayesian networks for event detection in video and employed boosting to improve this detector's performance. Unlike ours, this approaches attempted to utilize complex base classifiers that were not always guaranteed to perform better than 50%, a constraint necessary for boosting.

6. Conclusions and Future Work

We have presented a statistical framework for detection of objects and attributes in images based on boosting of weak classifiers [10]. By selecting a simple thresholded function as the weak classifier we have demonstrated how to obtain optimal detectors of skin color and faces. These simple yet highly efficient classifiers guarantee minimization of the training error as well as good generalization performance. Our preliminary results on two representative sets of web images suggest that detection performance of the boosted detectors is on par with that of the more complex traditional detection methods such as histograms, Gaussian mixtures, and support vector machines. However, our method is of lower complexity and trivial to implement and execute. As such, the boosted detection method may prove appropriate for applications in real-time and low-resource computing devices, such as handhelds.

We plan to conduct further comparative tests of the boosted detectors against other state-of-the-art models. Furthermore, we will extend the study to include comparison with the weak detectors that operate on more complex features, such summary features on image neighborhoods.

References

[1] T. S. Jebara and A. Pentland, "Parameterized structure from motion for 3d adaptive feedback tracking of faces," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 144–150, 1997.

[2] J. Yang, W. Lu, and A. Waibel, "Skin-color modeling and adaptation," in *Proc. ACCV*, pp. 687–694, 1998.

[3] D. A. Forsyth, M. Fleck, and C. Bregler, "Finding naked people," in *Proc. European Conference on Computer Vision*, pp. 593–602, 1996.

[4] J. Z. Wang, J. Li, G. Wiederhold, and O. Firschein, "System for screening objectionable images using daubechies' wavelets and color histograms," in *Proc. IDMS*, 1997.

[5] M. J. Jones and J. Rehg, "Statistical color models with application to skin detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 274–280, 1999.

[6] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 586–591, 1991.

[7] K. Sung and T. Poggio, "Example-based learning for view-based human face detection'," a.i. memo 1521, MIT, December 1994.

[8] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 203–208, 1996.

[9] T. Rikert, M. Jones, and P. Viola, "A cluster-based statistical model for object detection," in *ICCV*, 1999.

[10] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Machine Learning: Proceedings of the thriteenth international conference*, pp. 148–156, 1996.

[11] K. Tieu and P. Viola, "Boosting image retrieval," in *CVPR00*, pp. I:228–235, 2000.

[12] V. Pavlovic, A. Garg, J. Rehg, and T. Huang, "Multimodal speaker detection using error feedback dynamic bayesian networks," in *CVPR00*, pp. II:34–41, 2000.