FloatBoost Learning and Statistical Face Detection

Stan Z. Li, *Senior Member*, *IEEE*, and ZhenQiu Zhang

Abstract—A novel learning procedure, called FloatBoost, is proposed for learning a boosted classifier for achieving the minimum error rate. FloatBoost learning uses a backtrack mechanism after each iteration of AdaBoost learning to minimize the error rate directly, rather than minimizing an exponential function of the margin as in the traditional AdaBoost algorithms. A second contribution of the paper is a novel statistical model for learning best weak classifiers using a stagewise approximation of the posterior probability. These novel techniques lead to a classifier which requires fewer weak classifiers than AdaBoost yet achieves lower error rates in both training and testing, as demonstrated by extensive experiments. Applied to face detection, the FloatBoost learning method, together with a proposed detector pyramid architecture, leads to the first real-time multiview face detection system reported.

Index Terms—Pattern classification, boosting learning, AdaBoost, FloatBoost, feature selection, statistical models, face detection.

1 Introduction

Nonlinear classification of high-dimensional data is a challenging problem. AdaBoost methods, introduced by Freund and Schapire [1], provide a simple yet effective approach for stagewise learning of a nonlinear classification function. A classifier, or classification function, assigns the input a class label, such as +1 or -1. While a good classifier is difficult to obtain at once, AdaBoost learns a sequence of more easily learnable "weak classifiers,"[1] whose performances may be poor but better than random guessing; and boosts (combines) them into a "strong classifier" of higher accuracy.

Originating from the PAC (probably approximately correct) learning theory [2], [3], AdaBoost provably achieves arbitrarily good bounds on its training and generalization errors [1], [4] provided that weak classifiers can perform slightly better than random guessing on every distribution over the training set. It is also shown that such simple weak classifiers, when boosted, can capture complex decision boundaries [5].

Relationships of AdaBoost to functional optimization and statistical estimation have been established recently. It is shown that the AdaBoost learning procedure minimizes an upper error bound which is an exponential function of the margin on the training set [6]. Several gradient boosting algorithms are proposed [7], [8], [9], which provide new insights into AdaBoost learning. A significant advance is made by Friedman et al. [10]. It is shown that the AdaBoost algorithms can be interpreted as stagewise estimation procedures that fit an additive logistical regression model. Both the discrete AdaBoost [1] and the real version [4] optimize an exponential loss function, albeit in different ways. The work [10] links AdaBoost, which was advocated from the machine learning viewpoint, to the statistical theory.

1.1 Boosting Learning

The following two problems associated with AdaBoost motivated us to investigate into a more effective boosting learning algorithm: First, AdaBoost minimizes an exponential (or some other form of) function of the margin over the training set [6]. This is for convenience of theoretical and numerical analysis [10]. However, the ultimate goal in applications of pattern classification is usually to minimize a cost directly (usually linearly) associated with the error rate. A strong classifier learned by AdaBoost is suboptimal for applications in terms of error rate. This problem has been noted, e.g., by [11], but no solutions have been found in the literature.

Second, AdaBoost leaves a challenge of learning weak classifiers to the practitioner's choice. Learning the optimal weak classifier, such as the log posterior ratio given in [4], [10], requires estimation of densities in a feature space. This by itself is a difficult problem, especially when the dimensionality of the space is high. An effective and tractable weak learning algorithm is needed.

In this paper, we propose a novel learning procedure, called FloatBoost (Section 2), to bridge the gap between the goal of conventional boosting learning (maximizing the margin) and that of many applications (minimizing the error rate) by incorporating Floating Search [12] into AdaBoost. The idea of Floating Search is originally proposed for feature selection [12]. An incorporation of the backtrack mechanism from Floating Search into boosting learning allows deletions of weak classifiers that are ineffective in terms of the error rate. Because the deletions in backtrack are performed according to the error rate, an improvement in classification error is guaranteed. This leads to a strong classifier consisting of fewer weak classifiers [13], [14].

[1] A weak classifier can be any simple classification function, for example, a nearest-neighbor classifier, or a thresholded feature value, or a likelihood ratio function, or a posterior ratio function. The face detection system to be described in this paper is based on RealBoost where the posterior ratio type of weak classifiers is used.

- *S.Z. Li is with Microsoft Research Asia, 3/F Beijing Sigma Center, No. 49 Zhichun Road, Hai Dian District, Beijing 100080, China. E-mail: szli@microsoft.com.*
- *Z.Q. Zhang is with the Beckman Institute 2323, University of Illinois at Urbana-Champaign, 405 N. Mathews Ave., Urbana, IL 61801. E-mail: zzhang6@uiuc.edu.*

Manuscript received 6 June 2003; revised 23 Feb. 2004; accepted 9 Mar. 2004. Recommended for acceptance by V. Govindaraju.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0124-0603.

0162-8828/04/$20.00 © 2004 IEEE Published by the IEEE Computer Society

We also formulate a novel statistical model for learning weak classifiers (Section 3). A stagewise approximation is formulated to estimate the posterior probabilities based on effective features learned from an overcomplete feature set (i.e., in a very high-dimensional feature space). Weak classifiers are then defined as the logarithm of the posterior ratio. This provides a solution to the second problem discussed in the above.

1.2 Learning for Face Detection

The FloatBoost learning algorithm is applied to face detection. The boundary between the face and nonface patterns is highly nonlinear because the face manifold due to variations in facial appearance, lighting, head pose, and expression is highly complex [15], [16]. The learning-based approach has so far been the most effective for constructing face/nonface classifiers. See, e.g., [17], [18], [19], [20] (The reader is referred also to a recent *Handbook of Face Recognition* [21] for subspace/manifold modeling, statistical learning, face detection and recognition, and other aspects of face recognition in theories, algorithms, and applications).

The system of Viola and Jones [22], [23] makes a successful application of AdaBoost to face detection, after earlier work of Tieu and Viola [24] and Schneiderman [25]. There, AdaBoost is adapted to solve the following three fundamental problems in one boosting procedure: 1) learning effective features from a large feature set, 2) constructing weak classifiers each of which is based on one of the selected features, and 3) boosting the weak classifiers into a stronger classifier. Their system is the first real-time frontal-view face detector which runs at about 14 frames per second for a 320×240 image [22]. However, this work, like [17], [18], [19], [20], deals primarily with frontal faces.

In [26], Liu presents a Bayesian Discriminating Features (BDF) method. The input image, its 1D Harr wavelet representation, and its amplitude projections are concatenated into an expanded vector input of 768 dimensions. Assuming that these vectors follow a (single) multivariate normal distribution for face, linear dimension reduction is performed to obtain the PCA modes. The likelihood density is estimated using PCA and its residuals, using the Bayesian techniques presented in [27]. The nonface class is modeled similarly. A classification decision of face-nonface is made based on the two density estimates. The BDF classifier is reported to achieve result which compares favorably against the state-of-the-art face detection algorithms, such as the Schneiderman-Kanade method. It is interesting to note that such good results are achieved with a single Gaussian for face and one for nonface, and the BDF is trained using relatively small data sets—600 FERET face images and nine natural (nonface) images, and the trained classifier generalizes very well to test images. However, more details are needed to understand the underlying mechanism.

The ability to deal with faces of varying head poses (termed "multiview" faces hereafter) is important for many real applications because statistics show that approximately 75 percent of the faces in home photos are nonfrontal [28]. A reasonable treatment for multiview face detection and recognition in the appearance-based framework is the view-based method [29], whereby difficulties in explicit 3D modeling are avoided. Feraud et al. [30] adopt the view-based representation for face detection. Wiskott et al. [31] build elastic bunch graph templates for multiview face detection and recognition. Gong et al. [32] study the trajectories of faces in linear PCA feature spaces as they rotate, and use kernel support vector machines (SVMs) for multipose face detection and pose estimation [33], [34]. Huang et al. [35] use SVMs to estimate facial poses.

In the system of Schneiderman and Kanade [36], multiresolution information is used for different levels of a wavelet transform. The algorithm consists of an array of five face detectors in the view-based framework. Each is constructed using statistics of products of histograms computed from examples of the respective view. It takes 1 minute for a 320×240 image over only four octaves of candidate sizes as reported in [36].[2] While great success has been achieved for frontal-view face detection, much engineering work is needed for real-time multiview face detection.

Here, we present a multiview face detection system (Section 4) as an extension to the work of Schneiderman and Kanade [36] and Viola and Jones [22], [23]. The system applies the FloatBoost algorithm for learning face/nonface classifiers and uses a coarse-to-fine, simple-to-complex architecture called detector-pyramid [13] for efficient computation in the detection of multiview faces. This work leads to the first real-time multiview face detection system in the world. It runs at 200 ms per image of size 320×240 pixels on a Pentium-III CPU of 700 MHz.

Experimental results are presented in Section 5 to demonstrate FloatBoost learning and its use for face detection. Comparisons between FloatBoost and AdaBoost clearly show that FloatBoost yields a stronger classifier which consists of fewer weak classifiers than AdaBoost yet achieves lower error rates. Effectiveness of the detector-pyramid for multiview face detection is also demonstrated.

2 FloatBoost Learning

In this section, we give a brief review of the AdaBoost learning algorithm, in the notion of RealBoost [4], [10], as opposed to the original discrete AdaBoost [1]. Then, we present the FloatBoost learning procedure.

For two class problems, a set of N labeled training examples is given as $(x_1, y_1), \dots, (x_N, y_N)$, where $y_i \in \{+1, -1\}$ is the class label associated with example $x_i \in \mathbb{R}^n$. A stronger classifier is a linear combination of M weak classifiers

$$H_M(x) = \sum_{m=1}^{M} h_m(x). \tag{1}$$

In the real version of AdaBoost [4], [10], the weak classifiers can take a real value, $h_m(x) \in \mathbb{R}$, and have absorbed the coefficients needed in the discrete version ($h_m(x) \in \{-1, +1\}$ in the latter case). The class label for x is obtained as $H(x) = \mathrm{sign}[H_M(x)]$, while the magnitude $|H_M(x)|$ indicates the confidence. Every training example is associated with a weight, which approximates the distribution of the samples. During the learning process, the weights are updated after each iteration in such a way that more emphasis is placed on hard examples which are erroneously classified previously. Such a reweighting process is important for the original

[2] During the revision of this paper, Schneiderman and Kanade [37] reported an improvement in the speed of their system, using an coarse-to-fine search strategy together with various other heuristics (reusing Wavelet Transform coefficients, color preprocessing, etc.). The improved speed is 5 seconds for an image of size 240×256 using a Pentium II at 450MHz.

```
0. (Input)
    (1) Training examples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$,
        where $N = a + b$; of which $a$ examples have $y_i = +1$
        and $b$ examples have $y_i = -1$;
    (2) The maximum number $M_{\max}$ of weak classifiers to be combined;
1. (Initialization)
    $w_i^{(0)} = \frac{1}{2a}$ for those examples with $y_i = +1$ or
    $w_i^{(0)} = \frac{1}{2b}$ for those examples with $y_i = -1$.
    $M = 0$;
2. (Forward Inclusion)
  while $M < M_{\max}$
    (1) $M \leftarrow M + 1$;
    (2) Choose $h_M$ according to Eq.4;
    (3) Update $w_i^{(M)} \leftarrow \exp[-y_i H_M(x_i)]$, and normalize to $\sum_i w_i^{(M)} = 1$;
3. (Output)
    $H(x) = \text{sign}[\sum_{m=1}^{M} h_m(x)]$.
```

Fig. 1. RealBoost algorithm.

AdaBoost. However, it is noted in recent studies [7], [8], [9] that the artificial operation of explicit reweighting is unnecessary because it can be incorporated into a functional optimization procedure of boosting.

An error occurs when $H_M(x) \neq y$, or $yH_M(x) < 0$. The "margin" of an example (x, y) achieved by $H_M(x) \in \mathbb{R}$ on the training set examples is defined as $yH_M(x)$. This can be considered as a measure of the confidence of the prediction made by $H_M(x)$. The upper bound on classification error achieved by H_M can be derived as the following exponential loss function [6]

$$J(H_M) = \sum_i \mathrm{e}^{-y_i H_M(x_i)} = \sum_i \mathrm{e}^{-y_i \sum_{m=1}^{M} h_m(x_i)}. \qquad (2)$$

AdaBoost constructs $h_m(x)$ by stagewise minimization of (2). Given the current $H_{M-1}(x) = \sum_{m=1}^{M-1} h_m(x)$, the best $h_M(x)$ for the new strong classifier $H_M(x) = H_{M-1}(x) + h_M(x)$ is the one which leads to the minimum cost

$$h_M = \arg\min_{h^\dagger} J(H_{M-1}(x) + h^\dagger(x)). \qquad (3)$$

The minimizer is derived as [4], [10]

$$h_M(x) = \frac{1}{2} \log \frac{P(y = +1 | x, w^{(M-1)})}{P(y = -1 | x, w^{(M-1)})}, \qquad (4)$$

where $w^{(M-1)}(x, y) = \exp(-yH_{M-1}(x))$ is the weight after iteration $M - 1$ for the labeled example (x, y) and

$$\begin{aligned} &P\Big(y = +1 | x, w^{(M-1)}\Big) \\ &= \frac{E\big(w(x, y) \cdot 1_{[y=+1]} | x\big)}{E(w(x, y) \| \ x)} = \frac{\sum_{\forall y=+1} w(x, y)}{\sum_{\forall y} w(x, y)}, \end{aligned} \qquad (5)$$

where $E(\cdot)$ stands for the mathematical expectation and $1_{[C]}$ is one if C is true or zero otherwise. $P(y = -1 | x, w^{(M-1)})$ is defined similarly.

The AdaBoost algorithm based on the descriptions from [4], [10] is shown in Fig. 1. There, the reweight formula in Step 2.(3) is equivalent to the multiplicative rule in the original form of AdaBoost [1], [4]. In Section 4, we will present a statistical model for stagewise approximation of $P(y = +1 | x, w^{(M-1)})$.

FloatBoost [13], [14] performs a backtrack after the latest weak classifier h_M is added by AdaBoost. The backtrack deletes, from the set of learned weak classifiers $\{h_m | m = 1, \ldots, M\}$, those h_m which do not help in terms of the error rate, in order to improve on the overall error rate.

The idea of backtrack is originally from Floating Search [12]. It is aimed at dealing with the nonmonotonicity problem (explained below) in sequential feature selection. In the well-known sequential forward selection (SFS) and sequential backward selection (SBS), features are added or deleted one by one to improve the performance, which is step-optimal only. An assumption for such a sequential selection strategy to work well for the entire process is the monotonicity, that is, adding or deleting a feature leads to an improvement in the overall performance. The nonmonotonicity problem is such that adding an additional feature may lead to a drop in the overall performance and there is no way to correct this in later stages.

Several solutions are proposed. The plus-ℓ-minus-r [38], [39] combines SFS and SBS to tackle this problem; there, ℓ features are added or deleted, and then an r step backtrack is performed, with ℓ and r fixed. The Floating Search procedure [12] allows the number of backtracking steps to be controlled instead of being fixed beforehand. Specifically, it adds or deletes $\ell = 1$ feature and then backtracks r steps where r depends on the current situation. This flexibility amends limitations due to the nonmonotonicity problem. Improvement on the quality of selected features is gained with the cost of increased computation due to extended search. The algorithm performs very well in several applications [12],

```
0. (Input)
    (1) Training examples $\mathcal{Z} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$,
        where $N = a + b$; of which $a$ examples have
        $y_i = +1$ and $b$ examples have $y_i = -1$;
    (2) The maximum number $M_{\max}$ of weak classifiers;
    (3) The error rate $\varepsilon(H_M)$, and the acceptance threshold $\varepsilon^*$ .
1. (Initialization)
    (1) $w_i^{(0)} = \frac{1}{2a}$ for those examples with $y_i = +1$ or
        $w_i^{(0)} = \frac{1}{2b}$ for those examples with $y_i = -1$;
    (2) $\varepsilon_m^{\min}$ =max-value (for $m = 1, \ldots, M_{\max}$),
        $M = 0$, $\mathcal{H}_0 = \{\}$.
2. (Forward Inclusion)
    (1) $M \leftarrow M + 1$;
    (2) Choose $h_M$ according to Eq.4;
    (3) Update $w_i^{(M)} \leftarrow \exp[-y_i H_M(x_i)]$, and normalize to $\sum_i w_i^{(M)} = 1$;
    (4) $\mathcal{H}_M = \mathcal{H}_{M-1} \cup \{h_M\}$; If $\varepsilon_M^{\min} > \varepsilon(H_M)$, then $\varepsilon_M^{\min} = \varepsilon(H_M)$;
3. (Conditional Exclusion)
    (1) $h' = \arg\min_{h \in \mathcal{H}_M} \varepsilon(H_M - h)$;
    (2) If $\varepsilon(H_M - h') < \varepsilon_{M-1}^{\min}$, then
        (a) $\mathcal{H}_{M-1} = \mathcal{H}_M - h'$;
            $\varepsilon_{M-1}^{\min} = \varepsilon(H_M - h')$; $M = M - 1$;
        (b) $H_M = \sum_{h \in \mathcal{H}_M} h$;
        (c) goto 3.(1);
    (3) else
        (a) if $M = M_{\max}$ or $J(\mathcal{H}_M) < J^*$, then goto 4;
        (b) $w_i^{(M)} \leftarrow \exp[-y_i H_M(x_i)]$;   goto 2.(1);
4. (Output)
    $H(x) = \text{sign}[\sum_{h(x) \in \mathcal{H}_M} h(x)]$.
```

Fig. 2. FloatBoost Procedure. $\mathcal{H}_M = \{h_1, \ldots, h_M\}$ is the the set of the best M weak classifiers learned so far; $\varepsilon(H_M)$ the error rate achieved by $H_M(x) = \sum_{m=1}^{M} h_m(x)$ (or a weighted sum of missing rate and false alarm rate which is usually the criterion in detection problems); $\varepsilon_m^{\min}$ the minimum error rate achieved so far with an ensemble of m weak classifiers.

[40]. Floating Search is further developed in [41] by allowing more flexibility in the determination of ℓ.

FloatBoost uses backtrack to remove unfavorable weak classifiers from the existing classifiers, in order to achieve a low error rate. The FloatBoost procedure is shown in Fig. 2. In Step 2 (forward inclusion), given already selected weak classifiers, the next best weak classifier is added one at a time, which is the same as in AdaBoost. In Step 3 (conditional exclusion), FloatBoost removes the least significant weak classifier from $\mathcal{H}_M$, subject to the condition that the removal leads to an error rate that is lower than $\varepsilon_{M-1}^{\min}$. These are repeated until no more removals can be done. The procedure terminates when the risk on the training set is below J^* or the maximum number $M_{\max}$ is reached.

With the conditional exclusion incorporated, FloatBoost improves both feature selection and classifier learning. It always results in fewer weak classifiers than AdaBoost to achieve the same error rate ε. Because deletions in backtrack are performed according to the error rate, a lower error rate, and reduced feature set are guaranteed.

3 Learning Weak Classifiers

The optimal weak classifier at stage M is derived as (4). It can be expressed as follows, using $P(y|x, w) = p(x|y, w) P(y)/p(x|w)$:

$$h_M(x) = L_M(x) + T, \tag{6}$$

where

$$L_M(x) = \frac{1}{2} \log \frac{p(x|y = +1, w)}{p(x|y = -1, w)} \tag{7}$$

$$T = \frac{1}{2} \log \frac{P(y = +1)}{P(y = -1)}. \tag{8}$$

The log likelihood ratio (LLR) $L_M(x)$ is learned from the training examples of the two classes. The threshold T is determined by the log ratio of prior probabilities. In practice,

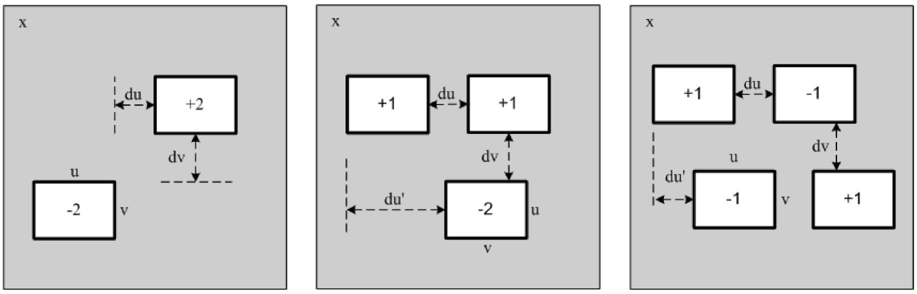


Fig. 3. Three types of simple Haar wavelet like features z'_k defined on a face example x. Each block in x consists of $u \times v$ pixels. du, dv, du', dv' are the distances between blocks. The number $\pm 1, 2$ inside a block is the weight for the pixels in the block. A feature takes a value calculated by weighted sum of the pixel values over the blocks in the window.

T can be adjusted to balance between the detection and false alarm rates (i.e., to choose a point on the ROC curve).

Learning optimal weak classifiers requires modeling the LLR of (7). Estimating the likelihood for high-dimensional data x is a nontrivial task. In this work, we derive the likelihood $p(x|y, w^{(M-1)})$ based on an overcomplete scalar feature set $\mathcal{Z} = \{z'_1, \ldots, z'_K\}$ and make use of the stagewise characteristics of boosting learning to approximate the likelihood based on effective features learned stagewise. More specifically, we approximate $p(x|y, w^{(M-1)})$ by $p(z_1, \ldots, z_{M-1}, z'|y, w^{(M-1)})$, where $z_m (m = 1, \ldots, M-1)$ are the features that have already been selected from $\mathcal{Z}$ by the previous stages, and z' is the feature to be selected. The set $\mathcal{Z}$ of candidate features and a method for constructing weak classifiers based on these features are described in the following.

A scalar feature $z'_k : x \rightarrow \mathbf{R}$ is a transform from the n-dimensional (400D if a face example x is of size 20×20) data space to the real line. For multiview face detection, three basic types of scalar features are used, as shown in Fig. 3. For each face example of size 20×20, there are hundreds of thousands of different z'_k for admissible $u, v, du, dv, du'dv'$ values, so $\mathcal{Z}$ is an overcomplete feature set for the intrinsically low-dimensional face pattern x. These block difference features have scalar values which are extensions to the steerable filters or Haar wavelets used in [42], [22]. These features can be computed very efficiently [43] from the summed-area table [44] or integral image [22]. Recently, Lienhart and Maydt proposed an extended set of features for dealing with in-plane rotations [45].

In this work, an optimal weak classifier (6) is associated with a single scalar feature; to construct the best new weak classifier is to choose the best corresponding feature. The feature selection or weak classifier construction is based on stagewise approximation of $p(x|y, w^{(M-1)})$.

We can approximate $p(x|y, w^{(M-1)})$ by using the conditional distributions of $z_1(x)$, $z_2(x)$, ..., $z_{M-1}(x)$ (the selected features) and $z'(x)$ (one to be selected):

$$\begin{aligned} p(x|y, w^{(M-1)}) &\approx p(z_1, z_2, \ldots, z_{M-1}, z'|y, w^{(M-1)}) \\ &= p(z_1|y, w^{(M-1)})\, p(z_2|y, z_1, w^{(M-1)}) \cdots \qquad (9) \\ &\quad p(z_{M-1}|y, z_1, \ldots, z_{M-2}, w^{(M-1)}) \\ &\quad p(z'|y, z_1, \ldots, z_{M-1}, w^{(M-1)}). \qquad (10) \end{aligned}$$

When $\mathcal{Z}$ is an overcomplete basis set and the z_m's learned by AdaBoost are weakly dependent, the above stagewise approximation becomes increasingly accurate as M grows. Note that the following holds

$$p(z'|y, z_1, \ldots, z_{m-1}|w^{(m-1)}) = p(z'|y, w^{(m-1)}) \qquad (11)$$

because $w^{(m-1)}$ is obtained in the entire history of w and accounts for the dependencies on $z_1, \ldots, z_{m-1}$. Therefore, we have

$$\begin{aligned} p(x|y, w^{(M-1)}) \approx p(z_1|y, w^{(0)})\, p(z_2|y, w^{(1)}) \cdots \\ p(z_{M-1}|y, w^{(M-2)}) p(z'|y, w^{(M-1)}). \end{aligned} \qquad (12)$$

On the right-hand side of the above equation, all the conditional densities are fixed except for the last one $p(z'|y, w^{(M-1)})$. The densities $p(z'(x)|y, w^{(M-1)})$ for the positive class $y = +1$ and the negative class $y = -1$ can be estimated using the histograms computed from the weighted voting of the labeled training examples (x, y) with the weights $w^{(M-1)}(x, y)$.

$L_M(x)$ of (8) for the optimal weak classifier can be estimated based on (12). Using the form of the approximation formula (12), we can define the following component LLR's for the target $L_M(x)$:

$$\tilde{L}_m(x) = \frac{1}{2} \log \frac{p(z_m|y = +1, w^{(m-1)})}{p(z_m|y = -1, w^{(m-1)})} \qquad (13)$$

for the selected features, z_m's ($m = 1, \ldots, M-1$), and

$$L_k^{(M)}(x) = \frac{1}{2} \log \frac{p(z'_k(x)|y = +1, w^{(M-1)})}{p(z'_k(x)|y = -1, w^{(M-1)})} \qquad (14)$$

for features to be selected, $z'_k \in \mathcal{Z}$. Then, the target LLR function can be approximated as

$$L_M(x) = \frac{1}{2} \log \frac{p(x|y = +1, w^{(M-1)})}{p(x|y = -1, w^{(M-1)})} \approx \sum_{m=1}^{M-1} \tilde{L}_m(x) + L_k^{(M)}(x). \qquad (15)$$

Let

$$\Delta L_M(x) = L_M(x) - \sum_{m=1}^{M-1} \tilde{L}_m(x). \qquad (16)$$

The best feature is the one whose corresponding $L_k^{(M)}(x)$ best fits $\Delta L_M(x)$. It can be found as the solution to the following minimization problem

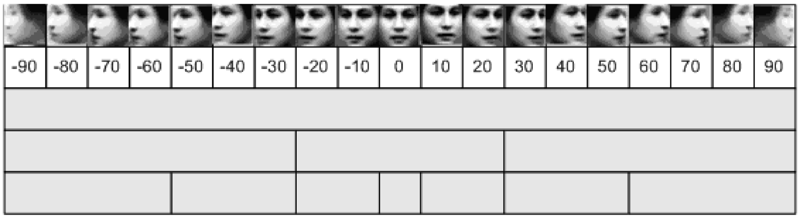


Fig. 4. Out-of-plane view partition. Out-of-plane head rotations (row 1), the facial view labels (row 2), and the coarse-to-fine view partitions at the three levels of the detector-pyramid (rows 3-5).

$$k^* = \arg\min_{k,\beta} \sum_{i=1}^{N} \Big[\Delta L_M(x_i) - \beta L_k^{(M)}(x_i)\Big]^2. \tag{17}$$

This can be done in two steps as follows: First, find k^* for which

$$(L_k^{(M)}(x_1), L_k^{(M)}(x_2), \ldots, L_k^{(M)}(x_N)) \tag{18}$$

is most parallel to

$$(\Delta L_M(x_1), \Delta L_M(x_2), \ldots, \Delta L_M(x_N)). \tag{19}$$

This amounts to finding k for which $L_k^{(M)}$ is most correlated with ΔL_M over the data distribution, and set $z_M = z'_{k^*}$. Then, we compute

$$\beta^* = \frac{\sum_{i=1}^{N} \Delta L_M(x_i) L_{k^*}(x_i)}{\sum_{i=1}^{N} [L_{k^*}(x_i)]^2}. \tag{20}$$

After that, we obtain

$$\tilde{L}_M(x) = \beta^* L_{k^*}(x). \tag{21}$$

The strong classifier is then given as

$$H_M(x) = \sum_{m=1}^{M} \big(\tilde{L}_m(x) + T\big) = \sum_{m=1}^{M} \tilde{L}_m(x) + MT. \tag{22}$$

4 Multiview Face Detection System

The face detection problem here is to classify an image of standard size (e.g., 20×20 pixels) as either face or nonface (imposter). This is essentially a one-against-rest classification problem in that everything not a face is a nonface. Here, we present engineering solutions for multiview face detection. A coarse-to-fine view-partition strategy is used and this leads to a detector-pyramid architecture consisting of several levels from the coarse level on the top to the fine level at the bottom.

4.1 Dealing with Head Rotations

Our system deals with three types of head rotations in the following ranges: 1) out-of-plane rotations in the range of $\Theta = [-90, +90]$ (in degrees), 2) in-plane rotations in the range of $\Phi = [-45, +45]$, and 3) up-and-down nodding rotations approximately in the range of $[-20, +20]$. A detector-pyramid is constructed to detect the presence of up-right faces, for a certain range of out-of-plane rotations in Θ and in-plane rotations in $[-15, +15]$. The design of such a detector-pyramid will be described shortly. In-plane rotations are handled as follows: 1) Divide Φ into three subranges $\Phi_1 = [-45, -15]$, $\Phi_2 = [-15, +15]$, and $\Phi_3 = [+15, +45]$. 2) Apply the detector-pyramid on two images in-plane-rotated by ± 30 as well on the original image. This will effectively cover in-plane-rotations in $[-45, +45]$. The up-and-down nodding rotations are dealt with by tolerances of the face detectors to them.

4.2 Detector-Pyramid

The design of the detector-pyramid adopts the coarse-to-fine and simple-to-complex (top-down in the pyramid) strategy [46], [47], and generalizes the cascade structure of Viola and Jones's system [22] to suit the multiview case.

Coarse-to-Fine. The full range Θ of out-of-plane rotations is partitioned into increasingly narrower ranges and, thereby, the whole face space is divided into increasingly smaller subspaces. Our current implementation of the detector-pyramid consists of three levels. The partitions of the out-of-plane rotation for the three levels are illustrated in Fig. 4. Although there are no overlaps between the partitioned view subranges at each level, a face detector trained for one view can usually cope with faces in an extended view range.

The detector-pyramid architecture is illustrated in Fig. 5 for the detection of faces with out-of-plane rotation in $\Theta = [-90, +90]$. (In-plane rotations in $\Phi = [-45, +45]$ is dealt with by applying the detector-pyramid on the images rotated ± 30, as discussed earlier.) The figure shows a detector pyramid of n levels. D_{LV} denotes a detector which detects faces of view range V at pyramid level L. The one, D_{11}, on the top is for the coarsest classification of faces in the whole range of out-of-plane rotation. Those at the bottom (level n) are for the finest classification. The current implementation according to the partition of Fig. 4 consists of 11 detectors. The final result is obtained after merging the subwindows that pass the seven channels at the bottom level. This is schematically illustrated in Fig. 6.

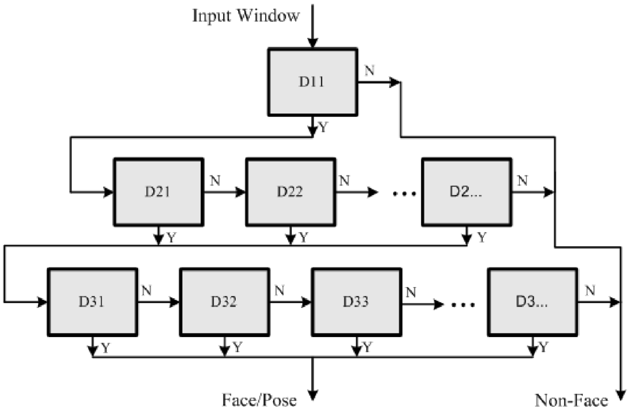


Fig. 5. Detector-pyramid for multiview face detection.

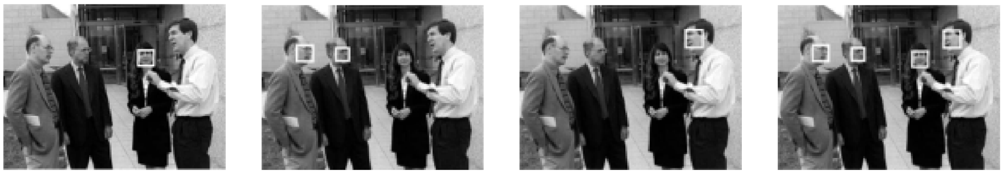
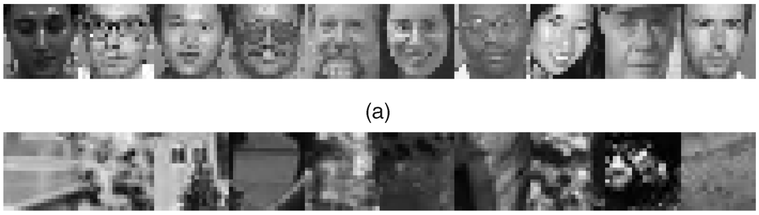


Fig. 6. Schematic illustration of merging from different channels. From left to right: Outputs of frontal, left, right view channels, and the final result after merge.



(a)

(b)

Fig. 7. (a) Face and (b) nonface examples.

Simple-to-Complex. A vast number of subwindows result from the scan of the input image. For the purpose of efficiency, it is crucial to discard as many nonface subwindows as possible at the earliest possible stage so that as few subwindows as possible will be processed further by later stages. Therefore, the detectors in the early stages are simpler so as to reject a vast number of nonface subwindows more quickly with little computation, whereas those in the later stage are more complex and spend more time. Therefore, a detector, i.e., a block in the pyramid, consists of a cascade of strong classifiers for efficient classification, following the idea of Viola and Jones [22].

4.3 Summary of the System

Now, we summarize how to construct a detector pyramid for multiview face detection: The full range of out-of-plane rotation is partitioned into one (for the top level of the pyramid) or several subranges (for the lower levels). The detectors in the pyramid are learned independently of one another, using face examples for the corresponding view range and bootstrapped nonface examples. The learning of a detector is done as follows:

1. A set of simple Haar wavelet *features* are used as candidate features. There are tens of thousands of such features for a 20×20 window.
2. A subset of them are selected and the corresponding *weak classifiers* are constructed, using FloatBoost learning.
3. A *strong classifier* is constructed as a linear combination of the weak ones.
4. A *detector* is composed of one or several strong classifiers in cascade.

The detector pyramid is then built upon the learned detectors. More detailed specifications will be given in Section 5.2.2.

5 Experimental Results

The following experiments compare FloatBoost (FB) and AdaBoost (AB) learning algorithms in their performances for nonlinear classification and face detection.

5.1 Comparisons in Boosting Learning for Classification

5.1.1 On Single Strong Classifiers

This set of experiments compares single strong classifiers learned by using FB and AB algorithms in their classification performance. While a cascade of stronger classifiers are needed to achieve a very low false alarm rate for face detection [22], [13], this is for the comparison on the effectiveness of the two boosting learning algorithms, rather than on overall system performance.

The data set is composed of face and nonface images of size 20×20. A set of 5,000 frontal face images is collected from various sources. The faces are cropped and rescaled to images of size 20×20. Another set of 5,000 nonface examples of the same size are collected from images containing no faces. See Fig. 7 for a random sample of face and nonface images. The 5,000 examples in each set are divided into a training set of 4,000 examples and a test set of 1,000 examples.

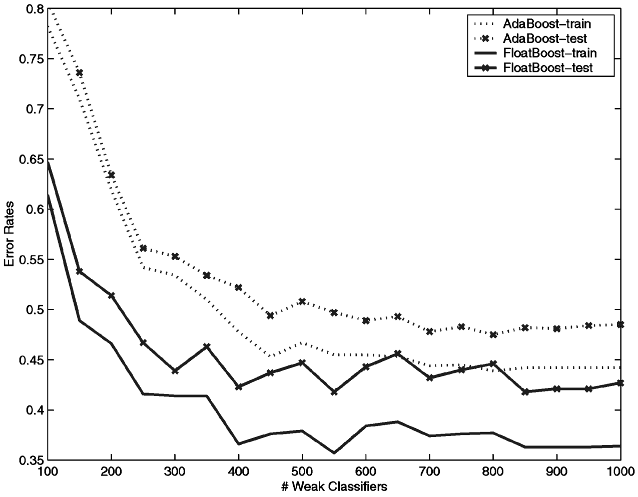


Fig. 8. The false alarm error rates of FB and AB algorithms on frontal face training and test sets, as a function of the number of weak classifiers. Here, an FB or AB strong classifier is composed of up to 1,000 weak classifiers.

TABLE 1
Comparison of Cascades of AB and FB Classifiers

AB-1										
Stage	1	2	3	4	5	6	7	8	9	10
# WC	6	9	26	37	53	66	89	151	200	200
Tot # WC	6	15	41	78	131	197	286	437	637	837
DR	99.5%	99%	98.5%	98%	97.5%	97%	96.5%	96%	95.5%	95%
FA	31.64%	26.64%	29.68%	29.00%	29.20%	28.96%	28.52%	29.88%	32.44%	45.64%
Overall FA	0.316	0.0843	0.0250	0.00726	0.00212	6.14e-4	1.75e-4	5.23e-5	1.70e-5	7.74e-6

FB-1										
Stage	1	2	3	4	5	6	7	8	9	10
# WC	5	8	23	29	36	46	78	99	133	200
Tot # WC	5	13	36	65	101	147	225	324	457	657
DR	99.5%	99%	98.5%	98%	97.5%	97%	96.5%	96%	95.5%	95%
FA	29.94%	27.66%	28.30%	28.98%	29.12%	29.04%	29.76%	29.20%	30.02%	46.21%
Overall FA	0.299	0.0828	0.0234	0.00679	0.00198	5.74×e-4	1.71e-4	4.99e-5	1.50e-5	6.92e-6

FB-2										
Stage	1	2	3	4	5	6	7	8	9	10
# WC	6	9	26	37	53	66	89	151	200	200
Tot # WC	6	15	41	78	131	197	286	437	637	837
DR	99.5%	99%	98.5%	98%	97.5%	97%	96.5%	96%	95.5%	95%
FA	25.6%	21.8%	22.1%	30.1%	24.0%	28.4%	28.3%	26.9%	39.2%	52.3%
Overall FA	0.256	0.0558	0.0123	0.0037	0.00089	2.53e-4	7.16e-5	1.93e-5	7.55e-6	3.95e-6

The performance is measured by the false alarm (FA) error rate, given that the detection rate (DR) on the training set is fixed at 99.5 percent. The FA curves for the training and test sets for the two algorithms are shown in Fig. 8. The following conclusions can be made from these curves: 1) Given the same number of learned features or weak classifiers, FloatBoost always achieves lower error rates than AdaBoost for both training and test data sets. For example, on the test set, the FB false alarm rate with an ensemble of 1,000 weak classifiers is 0.427, as opposed to that of 0.485 made by AB. 2) Even the false alarm rate on the *test* set is almost consistently lower than that of AB on the *training* set. 3) FB needs many fewer weak classifiers than AB in order to achieve the same false alarm rate. For example, AB needs 800 weak classifiers to achieve its lowest FA rate of 0.481 on the test set, whereas FB needs only 230 to achieve the same performance. These clearly demonstrate the strength of FloatBoost in statistical learning to achieve good classification performance.

5.1.2 On Cascades of Strong Classifiers

This set of experiments compare classification performances for cascades of 10 FB and AB strong classifiers. The training face data is the same as used in Section 5.1.1. Nonface images are collected by stagewise bootstrapping from 100,000 images containing no faces.

To evaluate this, we trained and compared three cascade face detectors: AdaBoost (AB-1), FloatBoost 1 (FB-1), and FloatBoost 2 (FB-2). AB-1 is trained in such a way that it achieves about 30 percent false alarm rate in each stage. FB-1 has the same detection false alarm rates as AB-1 for each stage but different numbers of weak classifiers. FB-2 has the same detection rate and the same numbers of weak classifiers as AB-1 but different false alarm rate.

Table 1 compares the three cascade classifiers in terms of the number of weak classifiers (WCs) for each stage and the total number of WCs, the detection (DR) and false alarm (FA) rates for each stage, and overall false alarm rate of the cascade. The AB and FB-based classifiers here are trained to have about the same target DR rates for the corresponding stages, but allow different FA rates and numbers of WC. While the rates in this table are for the training sets, the rates are for the training sets and the error rates for test sets are generally higher. Comparing FB-1 with AB-1, we see that the FloatBoost method needs fewer WCs than the AdaBoost method to achieve about the same DR and FA. Comparing FB-2 with AB-1, we see that FloatBoost can achieve a lower overall FA than AdaBoost (3.95e-6 against 7.74e-6), given the same number of WCs and DR. These further demonstrate the adavantages of FloatBoost as opposed to AdaBoost.

A comment follows: Table 1 compares the results of the cascades of strong classifiers, each cascade composed of up to 10 strong classifiers (10 stages), obtained with AB-1, FB-1, and FB-2 settings, and these results are for the training set only. Differently, Fig. 8 compares the single strong classifiers (1 stage) learned by using AB and FB, and the results are for both training and test data sets. These data sets are face icons rather than large images, and these experiments are meant to compare AB and FB learning algorithms. Performances of resulting face detection systems (including both cascade of strong classifiers and postprocessing mergers) performed on large images are presented the next.

5.2 Comparisons of AB and FB-Based Systems

These experiments compare face detection systems performing on large images, rather than 20×20 icons used for Table 1.

TABLE 2
Comparison of Face Detection Rate on the MIT+CMU Test Set

	FB	AB (20)	AB (24)	CMU-NN
DR(#FA=10)	83.6%	82.7%	76.1%	83.2%
DR(#FA=31)	90.2%	89.2%	88.4%	86%
#WC	2546	3872	6061	N/A

TABLE 3
Frontal Face Detection Comparison on Home-Brew Test Set

	AB-1	FB-1	FB-2
DR	85.31%	84.88%	86.18%
#FA	913	862	639
#WC	837	657	837

5.2.1 Frontal Faces

For the training of the systems, about 3,000 frontal face examples are collected from various sources and these faces are subject to slight out-of-plane rotation in the range of $[-20, +20]$. They are aligned by setting the eye and mouth coordinates to fixed positions. For each aligned face example, a synthesized face example is generated by a random in-plane-rotation in the range of $[-15, +15]$. This generates a training set of 6,000 face examples. The 6,000 images are then cropped and rescaled to the size of 20×20. Sufficient nonface examples are bootstrapped from 100,000 images containing no faces.

Two sets of experiments are performed, one with the MIT + CMU test set and the other with a home-brew test set. The MIT + CMU test set, which was used in [17], consists of 125 images containing 481 faces. For this data set, four detectors are compared:

1. Floatboost (FB),
2. AB(20) (AdaBoost of Viola-Jones as implemented by ourselves using training examples of size 20×20),
3. AB(24) (AdaBoost with training examples of size 24×24, as reported in [22]), and
4. CMU neural network-based system of Rowley et al. [17] as the baseline system.

Table 2 compares the detection rates (DR) for the four systems and the numbers of weak classifiers (#WC) for the three boosting-based systems, given the numbers of false alarms (#FA). The thresholds of FB, AB(20), and AB(24) classifiers are adjusted to have the same numbers of false alarms as reported in [17] on the MIT + CMU test set. Again, the results show that FB can achieve higher detection rate, given the same FA. Also, the FB-based system needs fewer weak classifiers than the AB-based systems, about 66 percent of that required by AB(20) and 42 percent of AB(24). If the same number of features (i.e., WCs) are used and a target detection rate is fixed, FB generally achieved lower FA rates than AB. Fig. 9 shows ROC curves of the FB and AB(20) based systems. The reader is also referred to [48] for performances of other systems on the MIT + CMU data set.

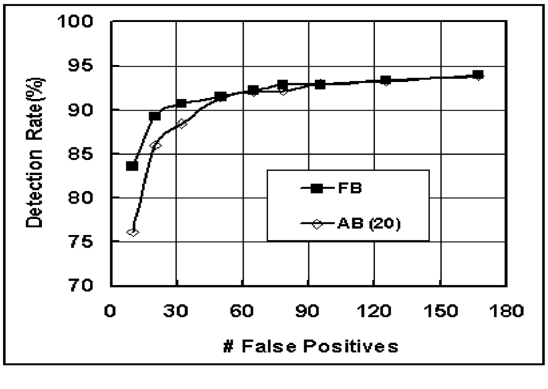


Fig. 9. Comparison of detection rates of FB and AB (20) methods on the MIT + CMU test set.

In the second set of system experiments, the systems are built using the same cascade detectors AB-1, FB-1, and FB-2 as described in Section 5.1.2, without any tuning, and the tests are performed using the home-brew face image set. This set contains 463 faces in 296 pictures, most of which are taken outdoors, with complex backgrounds and arbitrary illumination conditions. The comparison is shown in Table 3. FB-1 can achieve a performance comparable to AB-1, with fewer weak classifiers. FB-2, consisting of the same number of weak classifiers as AB-1, can achieve a higher detection rate and lower false alarm rate than the latter. These confirm the conclusions made in Section 5.1.2. Fig. 10 shows some FB-2 detection results.

5.2.2 Multiview Faces

This section demonstrates a multiview face detection system based on the method presented in Section 4. More than 6,000 original face samples are collected for the training, covering the out-of-plane rotation range $[-90, +90]$. A total number of about 25,000 multiview face images are generated from the 6,000 by randomly shifting or rotating the original images by a small amount.

The top level of the detector pyramid is trained with face examples in the view range $[-90, +90]$. At the second level, the face training set is divided into three view groups, corresponding to the subranges of $[-90, -30]$, $[-30, +30]$ and $[+30, +90]$. At the third level, the full range of $[-90, 90]$ is partitioned into nine view groups: $[-90, -70]$, $[-70, -50]$, $[-50, -30]$, $[-30, -10]$, $[-10, +10]$, $[+10, +30]$, $[+30, +50]$, $[+50, +70]$, $[+70, +90]$. The system consists of 13 detectors, but only eight of them need be trained due to the symmetry of the face, while the other side view detectors at the second and third levels can be constructed by mirroring features used in one side view detectors. This way, the number of cascade detectors and the training time are reduced to about $(K+1)/2$ for each level where K is the number of view groups for that level. These cascade detectors are trained independently. However, we believe that using bootstrapped nonface examples for training the detectors at later levels could lead to an improvement.

The cascades are trained in the following way: The top-level detector consists of a cascade of three strong classifiers, using 5, 13, and 20 features, respectively. It rejects about 50 percent of nonfaces, while retaining 99 percent of training faces. At the second level, each detector has a cascade of six strong classifiers and Rejects about 97 percent of nonfaces which passed through the top level, and retains 98 percent train faces. At the bottom level, each detector is a cascade of

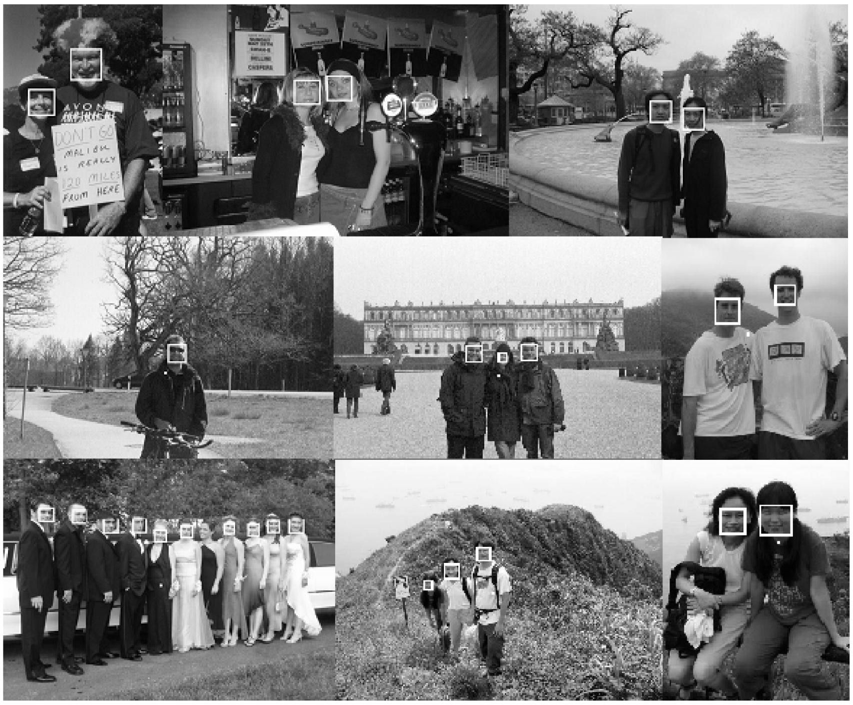


Fig. 10. Results of frontal face detection obtained using the FB-2-based detector.

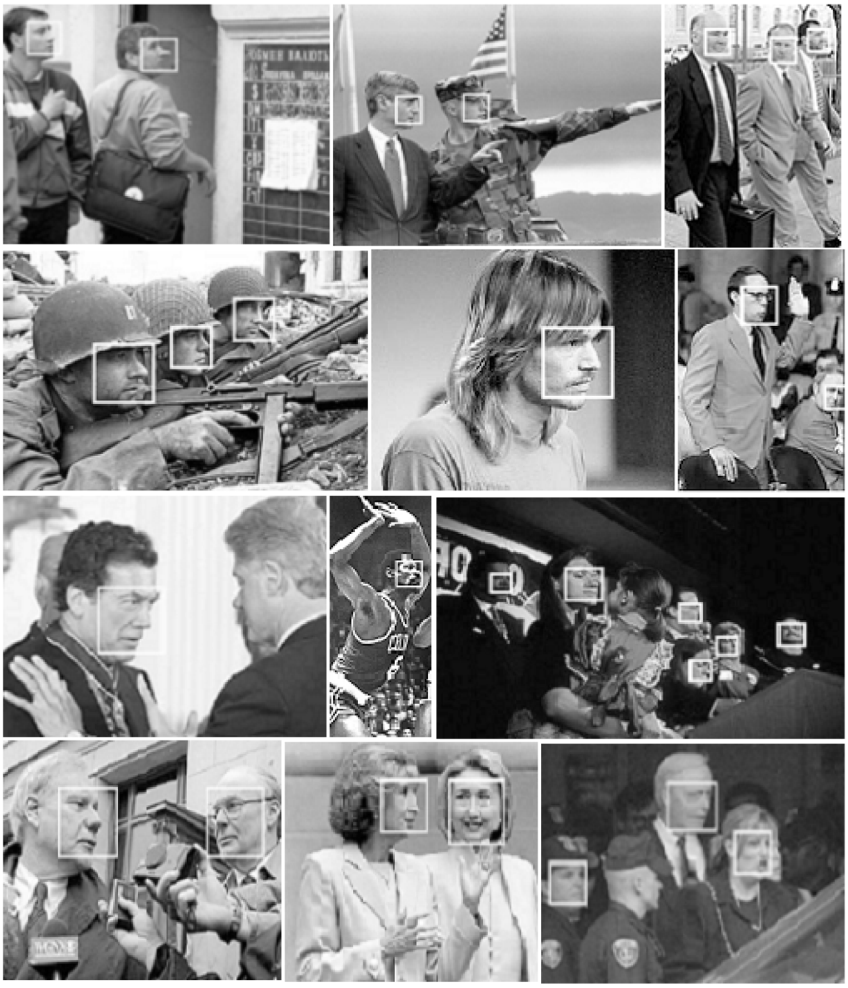


Fig. 11. Some multiview face detection results.

about 20 strong classifiers, and has a detection rate of about 94 percent and a false positive rate of about 4×10^{-6}.

The CMU profile face set [36] is used to test the algorithm. (The database can be downloaded at http://vasc.ri.cmu.edu/idb/html/face/profile_images/index.html.) This data set consists of 208 images with 441 faces of which 347 are profile views, which are not restricted in terms of subject matter or background scenery. They were collected from various news Web sites. Some results are shown in Fig. 11.

The detector-pyramid architecture is effective for speeding up multiview face detection. For an image of 320×240 pixels, there are a total of 70,401 subwindows to be classified. The full-view detector at the top of the pyramid needs 110 ms to process these subwindows, and rejects about 50 percent of them. The second level needs a total of 77 ms to process the remaining subwindows. The third level needs only 15 ms to process the remaining subwindows. So, the total time of the detector-pyramid processing is about 202 ms. Because only a small fraction of all the subwindows are processed by the third level, it will not increase computation much when the full view range is partitioned into smaller intervals. In contrast, the view-based approach applying all the nine detectors would cost 976 ms.

The system runs at 200 ms per image of size 320×240 pixels on a Pentium-III CPU of 700 MHz. This is the first real-time multiview face detection system in the world. Some face detection and tracking demos can be found at http://research.microsoft.com/~szli/Demos/MV-FaceDet.html.

6 Conclusion and Future Work

The contribution of this paper is summarized in the following: 1) A novel learning procedure, FloatBoost, is proposed to improve AdaBoost learning. 2) A novel statistical model is provided for stagewise approximation needed for learning weak classifiers. 3) The FloatBoost learning algorithm is applied to face detection and a detector pyramid architecture is presented for efficient detection of multiview faces.

By incorporating the idea of Floating Search [12] into AdaBoost [1], [4], FloatBoost learning results in a strong classifier which needs fewer weaker classifiers than AdaBoost to achieve a similar error rate, or achieves a lower error rate with the same number of weak classifiers. Real-time multiview face detection is achieved by incorporating the idea of the detector pyramid with the detectors learned using FloatBoost.

The performance improvement brought about by FloatBoost is achieved with the cost of longer training time, about five times longer for the FloatBoost classifiers evaluated in this paper. Several methods can be used to make the training more efficient. For example, noticing that only examples with large weight values are influential, Friedman et al. [10] propose to select only examples with large weights, i.e., those examples which are wrongly classified by the previously learned classifiers, and use them for the subsequent training. Top examples within a fraction of $1 - \beta$ of the total weight mass may be used, where $\beta \in [0.01, 0.1]$.

Currently, the cascade structure is adopted in a face detector. This is for computational efficiency in the run time. However, the overall detection rate of a cascade detector is approximately the product of the individual detection rates, resulting in a drop in the overall detection rate. A possible amendment is not to use cascade, but rather to use a single strong classifier consisting of a long sequence of weak classifiers. Such a "noncascade" detector should have many "exits" for rejecting nonfaces subwindows whenever possible. The learning of such a noncascade classifier should inherit sample weights learned previously, rather than starting from fresh new weights as in cascade learning. Our preliminary results show that this idea is effective.

Acknowledgments

The authors would like to thank Long Zhu for doing part of the experiments in the early development of this work, and Steve Lin for his help in proofreading the final manuscript.

References

[1] Y. Freund and R. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application To Boosting," *J. Computer and System Sciences,* vol. 55, no. 1, pp. 119-139, Aug. 1997.

[2] L. Valiant, "A Theory of the Learnable," *Comm. ACM,* vol. 27, no. 11, pp. 1134-1142, 1984.

[3] M.J. Kearns and U. Vazirani, *An Introduction to Computational Learning Theory.* Cambridge, Mass.: MIT Press, 1994.

[4] R.E. Schapire and Y. Singer, "Improved Boosting Algorithms Using Confidence-Rated Predictions," *Proc. 11th Ann. Conf. Computational Learning Theory,* pp. 80-91, 1998.

[5] L. Breiman, "Arcing Classifiers," *The Annals of Statistics,* vol. 26, no. 3, pp. 801-849, 1998.

[6] R. Schapire, Y. Freund, P. Bartlett, and W.S. Lee, "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," *The Annals of Statistics,* vol. 26, no. 5, pp. 1651-1686, Oct. 1998.

[7] J. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," *The Annals of Statistics,* vol. 29, no. 5, Oct. 2001.

[8] L. Mason, J. Baxter, P. Bartlett, and M. Frean, "Functional Gradient Techniques for Combining Hypotheses," *Advances in Large Margin Classifiers,* A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, eds., pp. 221-247, Cambridge, Mass.: MIT Press, 1999.

[9] R. Zemel and T. Pitassi, "A Gradient-Based Boosting Algorithm for Regression Problems," *Advances in Neural Information Processing Systems,* vol. 13, 2001.

[10] J. Friedman, T. Hastie, and R. Tibshirani, "Additive Logistic Regression: A Statistical View of Boosting," *The Annals of Statistics,* vol. 28, no. 2, pp. 337-374, Apr. 2000.

[11] P. Buhlmann and B. Yu, "Invited Discussion on 'Additive Logistic Regression: A Statistical View of Boosting (friedman, hastie and tibshirani)'," *The Annals of Statistics,* vol. 28, no. 2, pp. 377-386, Apr. 2000.

[12] P. Pudil, J. Novovicova, and J. Kittler, "Floating Search Methods in Feature Selection," *Pattern Recognition Letters,* vol. 15, no. 11, pp. 1119-1125, 1994.

[13] S.Z. Li, L. Zhu, Z.Q. Zhang, A. Blake, H. Zhang, and H. Shum, "Statistical Learning of Multi-View Face Detection," *Proc. European Conf. Computer Vision,* vol. 4, pp. 67-81, 2002.

[14] S.Z. Li, Z.Q. Zhang, H.-Y. Shum, and H. Zhang, "FloatBoost Learning for Classification," *Proc. Neural Information Processing Systems,* Dec. 2002.

[15] M. Bichsel and A.P. Pentland, "Human Face Recognition and the Face Image Set's Topology," *CVGIP: Image Understanding,* vol. 59, pp. 254-261, 1994.

[16] P.Y. Simard, Y.A.L. Cun, J.S. Denker, and B. Victorri, "Transformation Invariance in Pattern Recognition—Tangent Distance and Tangent Propagation," *Neural Networks: Tricks of the Trade,* G.B. Orr and K.-R. Muller, eds., Springer, 1998.

[17] H.A. Rowley, S. Baluja, and T. Kanade, "Neural Network-Based Face Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 1, pp. 23-28, Jan. 1998.

[18] K.-K. Sung and T. Poggio, "Example-Based Learning for View-Based Human Face Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 1, pp. 39-51, Jan. 1998.

[19] E. Osuna, R. Freund, and F. Girosi, "Training Support Vector Machines: An Application to Face Detection," *Computer Vision and Pattern Recognition*, pp. 130-136, 1997.

[20] M.-H. Yang, D. Roth, and N. Ahuja, "A SNoW-Based Face Detector," *Proc. Neural Information Processing Systems*, pp. 855-861, 2000.

[21] *Handbook of Face Recognition*, S.Z. Li and A.K. Jain, eds. Springer-Verlag, (in press), 2004.

[22] P. Viola and M. Jones, "Robust Real Time Object Detection," *IEEE ICCV Workshop Statistical and Computational Theories of Vision*, July 2001.

[23] P. Viola and M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, Dec. 2001.

[24] K. Tieu and P. Viola, "Boosting Image Retrieval," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 228-235, 2000.

[25] H. Schneiderman, "A Statistical Approach to 3D Object Detection Applied to Faces and Cars (cmu-ri-tr-00-06)," PhD dissertation, R.I., 2000.

[26] C. Liu, "A Bayesian Discriminating Features Method for Face Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 6, pp. 725-740, June 2003.

[27] B. Moghaddam and A. Pentland, "Probabilistic Visual Learning for Object Representation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 696-710, July 1997.

[28] A. Kuchinsky, C. Pering, M.L. Creech, D. Freeze, B. Serra, and J. Gwizdka, "FotoFile: A Consumer Multimedia Organization and Retrieval System," *Proc. ACM SIG CHI'99 Conf.*, May 1999.

[29] A.P. Pentland, B. Moghaddam, and T. Starner, "View-Based and Modular Eigenspaces for Face Recognition," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 84-91, 1994.

[30] J. Feraud, O. Bernier, and M. Collobert, "A Fast and Accurate Face Detector for Indexation of Face Images," *Proc. Fourth IEEE Int'l Conf. Automatic Face and Gesture Recognition*, 2000.

[31] L. Wiskott, J. Fellous, N. Kruger, and C.V. Malsburg, "Face Recognition By Elastic Bunch Graph Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 775-779, July 1997.

[32] S. Gong, S. McKenna, and J. Collins, "An Investigation into Face Pose Distribution," *Proc. IEEE Int'l Conf. Face and Gesture Recognition*, 1996.

[33] J. Ng and S. Gong, "Performing Multi-View Face Detection and Pose Estimation Using a Composite Support Vector Machine Across The View Sphere," *Proc. IEEE Int'l Workshop Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pp. 14-21, Sept. 1999.

[34] Y.M. Li, S.G. Gong, and H. Liddell, "Support Vector Regression And Classification Based Multi-View Face Detection and Recognition," *IEEE Int'l Conf. Face and Gesture Recognition*, pp. 300-305, Mar. 2000.

[35] J. Huang, X. Shao, and H. Wechsler, "Face Pose Discrimination Using Support Vector Machines (SVM)," *Proc. Int'l Conf. Pattern Recognition*, 1998.

[36] H. Schneiderman and T. Kanade, "A Statistical Method for 3D Object Detection Applied to Faces and Cars," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, 2000.

[37] H. Schneiderman and T. Kanade, "Object Detection Using the Statistics of Parts," *Int'l J. Computer Vision*, vol. 56, no. 3, pp. 151-177, Feb. 2004.

[38] S D. Stearns, "On Selecting Features for Pattern Classifiers," *Proc. Int'l Conf. Pattern Recognition*, pp. 71-75, 1976.

[39] J. Kittler, "Feature Set Search Algorithm," *Pattern Recognition in Practice*, C.H. Chen, ed., Sijthoff and Noordhoof: North Holland, pp. 41-60, 1980.

[40] A. Jain and D. Zongker, "Feature Selection: Evaluation, Application, and Samll Sample Performance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 153-158, Feb. 1997.

[41] P. Somol, P. Pudil, J. Novoviova, and P. Paclik, "Adaptive Floating Search Methods in Feature Selection," *Pattern Recognition Letters*, vol. 20, pp. 1157-1163, 1999.

[42] C.P. Papageorgiou, M. Oren, and T. Poggio, "A General Framework for Object Detection," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 555-562, 1998.

[43] P.Y. Simard, L. Bottou, P. Haffner, and Y.L. Cun, "Boxlets: A Fast Convolution Algorithm for Signal Processing and Neural Networks," *Advances in Neural Information Processing Systems*, M. Kearns, S. Solla, and D. Cohn, eds., vol. 11, MIT Press, pp. 571-577, 1998.

[44] F. Crow, "Summed-Area Tables for Texture Mapping," *Proc. SIGGRAPH*, vol. 18, no. 3, pp. 207-212, 1984.

[45] R. Lienhart and J. Maydt, "An Extended Set of Haar-Like Features for Rapid Object Detection," *Proc. IEEE Int'l Conf. Image Processing*, vol. 1, pp. 900-903, 2002.

[46] Y. Amit, D. Geman, and K. Wilder, "Joint Induction of Shape Features and Tree Classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp. 1300-1305, 1997.

[47] F. Fleuret and D. Geman, "Coarse-to-Fine Face Detection," *Int'l J. Computer Vision*, vol. 20, pp. 1157-1163, 2001.

[48] B.K.L. Erik Hjelmas, "Face Detection: A Survey," *Computer Vision and Image Understanding*, vol. 3, no. 3, pp. 236-274, Sept. 2001.

Stan Z. Li received the BEng degree from Hunan University, the MEng degree from the National University of Defense Technology, and the PhD degree from Surrey University where he also worked as a research fellow. All the degrees are in electrical and electronic engineering, He is a researcher at Microsoft Research Asia in Beijing. He joined Microsoft Research China in May 2000 after his post as an associate professor of Nanyang Technological University Singapore. His current research interest is in pattern recognition and machine learning, image analysis, and face technologies. He has published two books, including *Markov Random Field Modeling in Image Analysis* (Springer-Verlag, second edition in 2001) and *Handbook of Face Recognition* (editing with Anil K. Jain, Springer-Verlag, 2004), and more than 160 refereed papers and book chapters in these areas. He is a senior member of IEEE and currently serves on the editorial board of *Pattern Recognition* and on the program committees of various international conferences.

ZhenQiu Zhang received the BS degree from the Department of Electrical Engineering Tsinghua University, China, in 1999, and the MS degree in electrical engineering from the Chinese Academy of Science in 2002. He is currently a PhD candidate in the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. His research interests include computer vision, machine learning and multimodal human-computer interaction.