EURASIP Journal on Applied Signal Processing 2004:4, 503–509

Robust Face Detection in Airports

Jimmy Liu Jiang
School of Computing, National University of Singapore, Science Drive 2, Singapore 117559
Email: liujiang@pacific.net.sg

Kia-Fock Loe
School of Computing, National University of Singapore, Science Drive 2, Singapore 117559
Email: loekf@comp.nus.edu.sg

Hong Jiang Zhang
Microsoft Research Asia, Beijing Sigma Center, Beijing 100080, China
Email: hjzhang@microsoft.com

Received 25 December 2002; Revised 3 October 2003

Robust face detection in complex airport environment is a challenging task. The complexity in such detection systems stems from the variances in image background, view, illumination, articulation, and facial expression. This paper presents the S-AdaBoost, a new variant of AdaBoost developed for the face detection system for airport operators (FDAO). In face detection application, the contribution of the S-AdaBoost algorithm lies in its use of AdaBoost's distribution weight as a dividing tool to split up the input face space into inlier and outlier face spaces and its use of dedicated classifiers to handle the inliers and outliers in their corresponding spaces. The results of the dedicated classifiers are then nonlinearly combined. Compared with the leading face detection approaches using both the data obtained from the complex airport environment and some popular face database repositories, FDAO's experimental results clearly show its effectiveness in handling real complex environment in airports.

Keywords and phrases: S-AdaBoost, face detection, divide and conquer, inlier, outlier.

1. INTRODUCTION

A human face detection [1, 2, 3] system can be used for video surveillance and identity detection. Various approaches, based on feature abstraction and statistical analysis, have been proposed. Among them, Rowley and Kanade's neural network approach [4], Viola's asymmetric AdaBoost cascading approach [1], and support vector machine (SVM) approach [5] are a few of the leading ones. In the real world, the complex environment associated with the face pattern detection often makes the detection very complicated.

Boosting is a method used to enhance the performance of the weak learners (classifiers). The first provable polynomial-time boosting model [6] was developed from the probably approximately correct (PAC) theory [7], followed by the AdaBoost model [8], which has been developed into one of the simplest yet effective boosting algorithms in recent years.

In pattern detection and classification scenarios, the training input patterns are resampled in AdaBoost after every round of iteration. *Easy* patterns in the training set are assigned lower distribution weights; whereas the *difficult* patterns, which are often misclassified, are given higher distribution weights. After certain rounds of iteration, based on the values of the distribution weights assigned to the training input patterns, input training patterns can be classified into inliers (*easy* patterns) and outliers (*difficult* patterns).

When AdaBoost is used to handle scenarios in complex environment with many outliers, its limitations have been pointed out by many researchers [9, 10, 11, 12, 13, 14]. Some discussions and approaches [15, 16, 17, 18, 19] have been proposed to address these limitations.

Based on the distribution weights associated with the training patterns and applying the *divide and conquer principle*, a new AdaBoost algorithm, S-AdaBoost (suspicious AdaBoost), is proposed to enhance AdaBoost's capability of handling outliers in real-world complex environment.

The rest of the paper is organized as follows. Section 2 introduces S-AdaBoost structure, describes S-AdaBoost's divider, classifiers, and combiner, as well as compares the S-AdaBoost algorithm with other leading approaches on some benchmark databases. Section 3 introduces face detection for airport operators (FDAO) system and discusses S-AdaBoost algorithm in the domain of face pattern detection in the complex airport environment (as shown in Figure 1), where clear frontal-view potential face images cannot be assumed, and

Figure 1: Typical scenarios in complex airport environment.

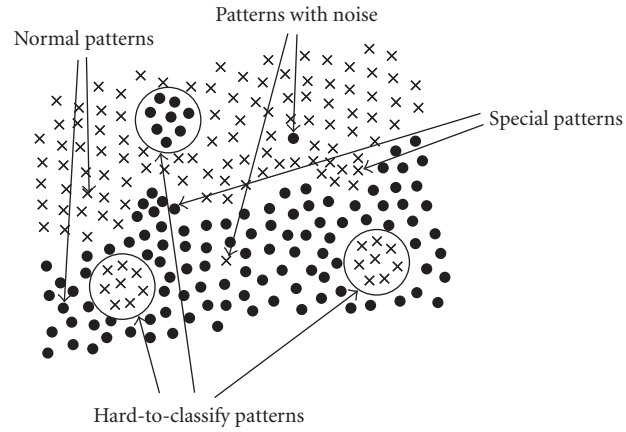

Figure 2: Input pattern space.

where minimum outliers are not norms. Section 3 also compares the performance of FDAO with other leading face detection approaches and followed by discussions in Section 4.

2. S-ADABOOST IN CLASSIFICATION

2.1. Input pattern analysis in S-AdaBoost

The *divide and conquer principle* is used in S-AdaBoost to *divide* the input pattern space **S** into a few subspaces and *conquer* the subspaces through simple fittings (decision boundaries) to the patterns in the subspaces. Input space can be denoted by

$$\mathbf{S} = \{\mathbf{P} = (\mathbf{X}, \mathbf{Y})\}, \tag{1}$$

where

$\mathbf{X} = \{\mathbf{x}_i\}$ denotes the input patterns,
$\mathbf{Y} = \{\mathbf{y}_i\}$ denotes the classification results,
$\mathbf{P} = \{\mathbf{p}_i = \{(\mathbf{x}_i, \mathbf{y}_i)\}\}$ denotes the input pattern and classification result pairs.

In S-AdaBoost, patterns in **S** can be divided into a few subsets relative to a classifier $\mathbf{T}(\mathbf{x})$:

$$\mathbf{S} = \mathbf{S}_{\mathbf{no}} + \mathbf{S}_{\mathbf{sp}} + \mathbf{S}_{\mathbf{ns}} + \mathbf{S}_{\mathbf{hd}}, \tag{2}$$

where,

$\mathbf{S}_{\mathbf{no}} = \{\mathbf{P}_{\mathbf{no}}\}$: normal patterns (patterns can be easily classified by $\mathbf{T}(\mathbf{x})$),
$\mathbf{S}_{\mathbf{sp}} = \{\mathbf{P}_{\mathbf{sp}}\}$: special patterns (patterns can be classified correctly by $\mathbf{T}(\mathbf{x})$ with bearable adjustment),
$\mathbf{S}_{\mathbf{ns}} = \{\mathbf{P}_{\mathbf{ns}}\}$: patterns with noise (noisy patterns),
$\mathbf{S}_{\mathbf{hd}} = \{\mathbf{P}_{\mathbf{hd}}\}$: hard-to-classify patterns (patterns hard to be classified by $\mathbf{T}(\mathbf{x})$).

A typical input pattern space is shown in Figure 2. The first two subspaces are further collectively referred to as ordinary pattern space (inlier space), and the last two are collectively called outliers space in S-AdaBoost:

$$\begin{aligned} \mathbf{S}_{\mathbf{od}} &= \mathbf{S}_{\mathbf{no}} + \mathbf{S}_{\mathbf{sp}}, \\ \mathbf{S}_{\mathbf{ol}} &= \mathbf{S}_{\mathbf{ns}} + \mathbf{S}_{\mathbf{hd}}. \end{aligned} \tag{3}$$

As shown in Figure 2, it is noticed that classifying all patterns in **S** using a single classifier $\mathbf{T}(\mathbf{x})$ with a simple decision boundary can be difficult sometimes. Nevertheless, after dividing **S** into $\mathbf{S}_{\mathbf{od}}$ and $\mathbf{S}_{\mathbf{ol}}$, it is relatively easier for an algorithm like AdaBoost to classify $\mathbf{S}_{\mathbf{od}}$ well with a not very complicated decision boundary. However, to correctly classify both $\mathbf{S}_{\mathbf{od}}$ and $\mathbf{S}_{\mathbf{ol}}$ well using only one classifier $\mathbf{T}(\mathbf{x})$ in **S**, the trade-off between the complexity and generalization of the algorithm needs to be considered. It is well understood that more complex $\mathbf{T}(\mathbf{x})$ yields lower training errors yet runs the risk of poor generalization [1]. It is confirmed by a number of researchers [4, 5, 6, 7, 8, 9] that if a system is to use AdaBoost alone to classify both $\mathbf{S}_{\mathbf{od}}$ and $\mathbf{S}_{\mathbf{ol}}$ well, $\mathbf{T}(\mathbf{x})$ will focus intensively on $\mathbf{P}_{\mathbf{ns}}$ and $\mathbf{P}_{\mathbf{hd}}$ in $\mathbf{S}_{\mathbf{ol}}$ and the generalization characteristic of the system will be affected in real-world complex environment.

2.2. S-AdaBoost machine

During training, instead of using single classifier (as shown in Figure 3) to fit all the training samples (often with outliers) as done in AdaBoost, S-AdaBoost uses an AdaBoost $\mathbf{V}(\mathbf{v})$ as a divider to divide the patterns in the training input space **S** into two separate sets in $\mathbf{S}_{\mathbf{od}}$ and $\mathbf{S}_{\mathbf{ol}}$. One set in $\mathbf{S}_{\mathbf{od}}$ is used to train the AdaBoost classifier $\mathbf{T}_{\mathbf{od}}(\mathbf{x})$, which has good generalization characteristic, and the other set in $\mathbf{S}_{\mathbf{ol}}$ is used to train a dedicated outlier classifier $\mathbf{T}_{\mathbf{ol}}(\mathbf{x})$, which has good localization capability. The structure of the S-AdaBoost machine is shown in Figure 4.

As the divider is used to separate the training input patterns to train the two dedicated classifiers, it is no longer needed in testing phase. The dedicated classifiers can make their independent classifications for any new inputs from the entire pattern space.

2.3. S-AdaBoost divider

An AdaBoost $\mathbf{V}(\mathbf{v})$ in the S-AdaBoost machine divides the original training set into two separate sets contained in $\mathbf{S}_{\mathbf{od}}$ and $\mathbf{S}_{\mathbf{ol}}$, respectively. The same AdaBoost algorithm is used in both the divider $\mathbf{V}(\mathbf{v})$ and the classifier $\mathbf{T}_{\mathbf{od}}(\mathbf{x})$ to ensure the optimal performance of the classifier $\mathbf{T}_{\mathbf{od}}(\mathbf{x})$.

In AdaBoost, input patterns are associated with distribution weights. The distribution weights of the more "outlying"

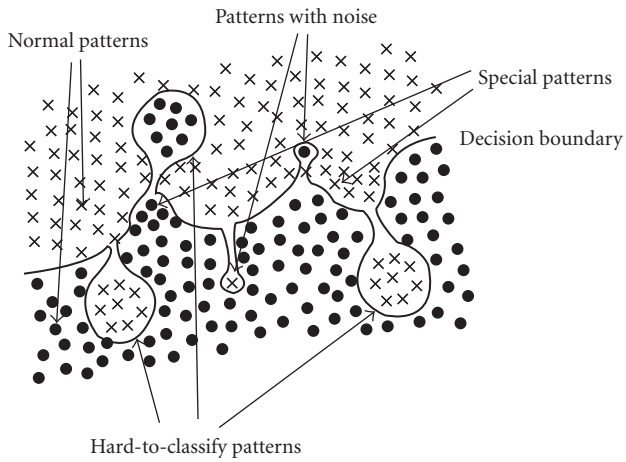

Figure 3: Single classifier for the input pattern space.

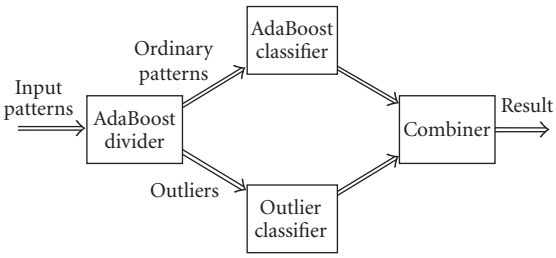

Figure 4: S-AdaBoost machine in training.

patterns increase after each iteration; and the distribution weights of the more "inlying" (or more "ordinary") patterns decrease after every iteration. When the distribution weight of a pattern reaches certain threshold, the chance of the pattern being an "outlier" is high. This property is used in $\mathbf{V}(\mathbf{v})$ to divide the input patterns into inliers (ordinary patterns) and outliers. The pseudocode of the AdaBoost divider $\mathbf{V}(\mathbf{v})$ based on a given weak learning algorithm $\mathbf{W}$ for a two-class classification can be described as in Algorithm 1.

It is task specific to choose the optimal value for the threshold $\mathbf{v}$. The implication of the optimal value will be discussed in the following sections.

2.4. S-AdaBoost's classifiers and combiner

After the training sets in input space $\mathbf{S}$ being divided into $\mathbf{S_{od}}$ and $\mathbf{S_{ol}}$, $\mathbf{P_{no}}$ and $\mathbf{P_{sp}}$ are used to train the $\mathbf{T_{od}}(\mathbf{x})$ classifier, whereas $\mathbf{P_{ns}}$ and $\mathbf{P_{hd}}$ are used to train the $\mathbf{T_{ol}}(\mathbf{x})$ classifier in the S-AdaBoost machine.

After certain rounds of iteration, $\mathbf{T_{od}}(\mathbf{x})$ classifier focuses more on the *relative difficult* $\mathbf{P_{sp}}$ and less on the *relative easy* $\mathbf{P_{no}}$ in forming the decision boundary. As $\mathbf{P_{sp}}$ are not outliers, the accuracy and generalization of the classifier $\mathbf{T_{od}}(\mathbf{x})$ is maintained. Making use of the randomness nature of $\mathbf{P_{ns}}$, $\mathbf{T_{ol}}(\mathbf{x})$, a classifier with good localization characteristic, can identify the local clustering of $\mathbf{P_{hd}}$ and at the same time isolate $\mathbf{P_{ns}}$ from $\mathbf{P_{hd}}$.

Given: Weak learning algorithm $\mathbf{W}$.
Training patterns: $\mathbf{S} = \mathbf{P} = \{\mathbf{p}_i = (\mathbf{x}_i, \mathbf{y}_i)\}$ for $i = 1$ to M,
where M stands for the number of the training patterns;
$\mathbf{x}_i \in \mathbf{X}$ stands for the input patterns;
$\mathbf{y}_i \in \mathbf{Y} = \{-1, 1\}$ stands for the targeted output;
number of iteration T;
the threshold value $\mathbf{v}$.

L0: Initialize the two subspaces:
$\mathbf{S_{od}} = \mathbf{S}$; $\mathbf{S_{ol}} = \{\cdot\}$;
$m = M$.

L1: Initialize distribution D (*distribution weights of training patterns*):
set $D_1(i) = \dfrac{1}{m}$ for all $i = 1$ to m;
set iteration count $t = 1$;
set divide = 0;
set initial error rate $\epsilon_1 = 0$.

L2: Iterate while $\epsilon_t < 0.5$ and $t \le T$. Call $\mathbf{W}$ algorithm with distribution D_i:
obtain from $\mathbf{W}$ the hypothesis
$$\mathbf{h}_t : X \longrightarrow Y;$$
calculate the weighted error rate:
$$\epsilon_t = \sum_{i:h_t(\mathbf{x}_i)!=\mathbf{y}_i} D_t(i);$$
set $\beta_t = \dfrac{\epsilon_t}{(1-\epsilon_t)}$;
update the new distribution D for $i = 1$ to m:
$$D_{t+1}(i) = \frac{D_t(i)\beta_t^{\mathrm{Sign}(h_t(\mathbf{x}_i)=\mathbf{y}_i)}}{z_t},$$
where Z_t is a normalization factor chosen such that the new distribution D_{t+1} is a normalized distribution.
$t{+}{+}$.
For $i = 1$ to m,
BEGIN
If $D_t(i) >$ the threshold value $\mathbf{v}$,
BEGIN
$m = m - 1$;
$\mathbf{S_{od}} = \mathbf{S_{od}} - P_i$;
$\mathbf{S_{ol}} = \mathbf{S_{ol}} + P_i$;
divide = 1.
END
If divide = 1,
go to L1.
END

L3: Export the ordinary pattern subspace $\mathbf{S_{od}}$ and the outlier subspace $\mathbf{S_{ol}}$.

Algorithm 1

Noticing that classifiers $\mathbf{T_{od}}(\mathbf{x})$ and $\mathbf{T_{ol}}(\mathbf{x})$ are of different structure and nature, a nonlinear combiner Ç instead of a linear one is used to combine the classification results from $\mathbf{T_{od}}(\mathbf{x})$ and $\mathbf{T_{ol}}(\mathbf{x})$ to generate the final classification result.

If threshold $\mathbf{v} \leq 0$, *then*
{ $\mathbf{S_{od}} = \{\cdot\}$;
all the patterns in **S** are treated as outliers;
the S-AdaBoost becomes a large memory network;
$\mathbf{T_{ol}}(\mathbf{x})$ determines the performance of S-AdaBoost.
}
If threshold $\mathbf{v} \geq 1$, *then*
{ $\mathbf{S_{ol}} = \{\cdot\}$;
no patterns in **S** are treated as outliers;
the performance of S-AdaBoost is determined by $\mathbf{T_{od}}(\mathbf{x})$;
S-AdaBoost machine becomes AdaBoost machine.
}

Algorithm 2

2.5. Choose threshold v value in S-AdaBoost divider

Threshold **v** plays a very important role in S-AdaBoost. This is noticed from Algorithm 2. AdaBoost can be considered as a special implementation of S-AdaBoost when threshold **v** value is greater than or equal to 1.

The optimal value of threshold **v** is associated with the classification task itself and the nature of patterns in **S**. Experiments were conducted to determine the optimal value for threshold **v** (as shown in Sections 2.6 and 3). From the experiments conducted, as a guideline, S-AdaBoost performed reasonably well when the value of threshold **v** was around $1/(M \times \partial^2)$, where M is the number of training patterns and ∂ is the false positive rate of S-AdaBoost when threshold $\mathbf{v} = 1$ (the AdaBoost's false positive rate).

2.6. Experiments on benchmark databases

From the "soft margin" approach, the regularized AdaBoost [19] has been regarded as one of the most effective classifiers handling outliers; mistrust is introduced to be associated with the training patterns to alleviate the distortion that an outlier can cause to the margin distribution. The mistrust values are calculated based on the weights calculated for those training patterns. Considering that the regularized AdaBoost approach demands vast computational resources to obtain the optimal parameters, S-AdaBoost is simpler, faster, and easy to be implemented.

Experiments were conducted to test the effectiveness of the S-AdaBoost algorithm on the GMD benchmark databases [20], which include samples from UCI [21], DELVE [22], and Statlog [23] benchmark repositories. The test results obtained from some leading algorithms, namely, AdaBoost, SVM, regularized AdaBoost [19], and S-AdaBoost (when threshold **v** is set to $1/(M \times \partial^2)$, where ∂ is the error rate of AdaBoost machine) were shown in Table 1. Ten cross-validation method was used in all the experiments, the means and standard deviations of the results are both listed.

From Table 1, it is shown that S-AdaBoost performs the best in terms of general performance and achieves the best results in 10 out of 13 tests; S-AdaBoost outperforms AdaBoost in all the 13 tests as well as outperforms SVM and regularized AdaBoost, which are the two leading approaches in handling complex environment.

Table 1: Error rates of some leading approaches on benchmark databases.

Database	AdaBoost	SVM	Reg. AdaBoost	S-AdaBoost
Banana	10.8 ± 0.8	11.0 ± 0.7	10.9 ± 0.7	10.6 ± 0.5
B. Cancer	30.8 ± 4.0	26.3 ± 4.5	26.5 ± 4.3	26.1 ± 4.3
Diabetes	26.8 ± 2.0	23.7 ± 2.0	23.8 ± 2.3	23.5 ± 1.6
German	27.5 ± 2.4	22.8 ± 2.0	24.3 ± 2.3	23.8 ± 2.4
Heart	20.8 ± 3.2	16.4 ± 3.2	16.5 ± 3.3	15.9 ± 3.1
Image	2.9 ± 0.9	2.8 ± 0.5	2.7 ± 0.4	2.7 ± 0.5
Ringnorm	1.9 ± 0.4	1.6 ± 0.2	1.6 ± 0.1	1.7 ± 0.2
F. Sonar	35.7 ± 1.6	32.0 ± 1.6	34.2 ± 1.8	31.6 ± 1.8
Splice	10.4 ± 1.1	10.6 ± 0.7	9.5 ± 1.0	9.3 ± 0.8
Thyroid	4.5 ± 2.1	4.9 ± 1.8	4.6 ± 2.0	4.3 ± 2.0
Titanic	23.1 ± 1.4	22.2 ± 1.2	22.6 ± 1.2	22.2 ± 1.1
Twonorm	3.0 ± 0.2	2.7 ± 0.2	2.7 ± 0.3	2.7 ± 0.2
Waveform	10.6 ± 1.3	9.8 ± 1.3	9.8 ± 1.1	9.6 ± 1.0
Average	16.1	14.5	14.6	14.1

3. S-ADABOOST FOR FACE DETECTION IN AIRPORT

3.1. FDAO

Real-time surveillance cameras are used in FDAO (as shown in Figure 5) to scan crowds and detect potential face images. An international airport has been chosen as the piloting complex environment to test the effectiveness of FDAO. Potential face images are to be detected in complex airport backgrounds, which include different configurations of illumination, pose, occlusion, and even make-up.

3.2. FDAO system training

Two CCD cameras with a resolution of 320×256 pixels were installed in the airport to collect training images for FDAO. Out of all the images collected, 5000 images with one or multiple face images were selected for this experiment. The 5000 raw images were further divided into two separate datasets, one of the datasets contained 3000 raw images and the other contained the remaining 2000 raw images. More than 7000 face candidates were cropped by hand from the 3000-image dataset as the training set for FDAO, and the 2000-image dataset was chosen as the test set. Five thousand nonface images (including images of carts, luggage, and pictures from some public image banks, etc.) were used (2500 images as the training set and the remaining 2500 images as the test set) as nonface image dataset. All the above training images were resized to 20×20 pixels and the brightness of the images were normalized to the mean of zero and standard deviation of one before being sent for training.

The preprocessor (as shown in Figure 5) acts as a filter to generate a series of potential face patches with 20×20-pixel resolution from the input image with the brightness normalized to the mean of zero and the standard deviation of one.

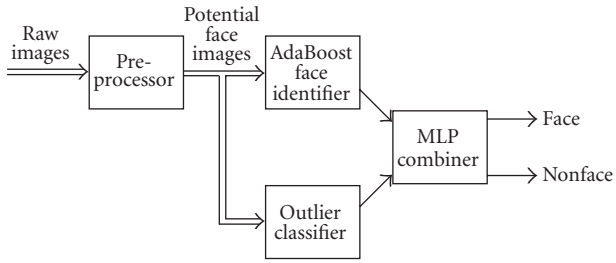

Figure 5: FDAO.

Simple edge detection techniques are used to remove some of the obvious nonface patches. The preprocessor is designed in such a way to generate extra candidates than the real number of faces from the original images to avoid face images not being detected.

The ordinary pattern (inlier) classifier $\mathbf{T_{od}}(\mathbf{x})$ and the AdaBoost divider $\mathbf{V}(\mathbf{v})$ (as shown in Figure 5) share the same structure. The base classifier is implemented by a fully connected three-layer (400 input nodes, 15 hidden nodes, and 1 output node) back-propagation (BP) neural network. BP neural network is chosen due to its good generalization capability. As face patterns are highly nonlinear, the nonlinear distributed representation and the highly connected structure of the BP base classifier suit the nature of the face detection problem.

The outlier classifier $\mathbf{T_{ol}}(\mathbf{x})$ is implemented by a three-layer radial basis function (RBF) neural network (400 input nodes, dynamic number of hidden nodes, and 1 output node). The RBF neural network is chosen due to its good localization characteristic. The radii of the hidden nodes in the RBF neural network are also set to be very small to enhance RBF network's good local clustering characteristic, which helps to isolate the noisy patterns $\mathbf{P_{ns}}$ from the hard-to-classify patterns $\mathbf{P_{hd}}$.

Two confidence-values outputs from the above classifiers are used as the inputs to the combiner **Ç**. The combiner **Ç** is implemented by a three-layer BP neural network (2 input nodes, 3 hidden nodes, and 1 output node).

The reason of choosing a nonlinear network to implement the combiner **Ç** instead of using a linear one is due to the consideration that the hidden layer nodes in nonlinear network enable the neural network to learn the complex relationship between the two confidence-values outputs by the two different neural network classifiers. As the RBF network and BP-based AdaBoost used to implement the dedicated classifiers are of different structure and nature, a nonlinear combiner is able to learn their complex relationship better than a linear one.

3.3. Testing result analysis

To test the effectiveness of S-AdaBoost's face detection capability, the performance of FDAO (when threshold **v** was set at $1/(M \times \partial^2)$) was compared with other leading approaches. Rowley and Kanade's neural network approach [4], Viola's asymmetric AdaBoost cascading approach [1], and SVM approach [5] were implemented. To compare various approaches using consistent methodology, the *detection error rate* $\boldsymbol{\delta}$ of the four algorithms is computed in our test: *detection error rate* $\boldsymbol{\delta}$ = (number of face images wrongly classified as nonface images + number of nonface images wrongly classified as face images)/ number of faces in the test set.

Table 2: Error rates of different approaches.

Approach	Rowley	Viola	SVM	S-AdaBoost
Detection error rate	29.4% ± 3.2%	27.1% ± 2.9%	27.7% ± 3.0%	25.5% ± 3.5%

To compare the effectiveness of different approaches in real complex airport environment, the same training and testing face as well as nonface datasets (as used in FDAO) were used in our experiment. During testing, the preprocessed data (20×20 images) were fed directly to $\mathbf{T_{od}}(\mathbf{x})$ and $\mathbf{T_{ol}}(\mathbf{x})$. The testing results obtained from various approaches are listed in Table 2.

Compared with the other three leading approaches on FDAO databases, it is shown that the S-AdaBoost approach performs the best in the experiment. Detail analysis of the S-AdaBoost in FDAO reviews that quite a number of "noisy" patterns and outliers are actually filtered to the $\mathbf{T_{ol}}(\mathbf{x})$, which results in optimal performance of $\mathbf{T_{od}}(\mathbf{x})$. The nonlinear combiner also contributes to the good performance of the system.

SVM-based face detection approaches use a small set of support vectors to minimize the structure risk. A linearly constrained quadratic programming problem, which is time and memory intensive, needs to be solved in the same time to estimate the optimal hyperplane. In the real world, the outliers are often misclassified as the support vectors in SVM-based approaches. Compared with the SVM-based approaches, S-AdaBoost is faster and divides the input patterns into inliers (ordinary patterns) and outliers to make sure the outliers are not influencing the classification of the ordinary patterns. Viola and Jones' approach is a rapid approach able to process the 15 fps (frame per second) 384×288 pixel gray-level input images in real time. Through introducing "integral image" representation scheme and using cascading multi-AdaBoost for feature selection and background-clearing, the system achieves very good performance. Compared with the Viola and Jones' approach, which uses more than 30 layers of AdaBoost machines in their implementation, S-AdaBoost uses just two layers of AdaBoost machine. It is less complex and can work in the normal CCD camera's rate of 60 fps.

Further comparison between the results in Table 1 and those in Table 2 shows that S-AdaBoost outperforms other methods more in Table 2 than in Table 1, which might be due to the fact that the data collected in FDAO is more "raw" and "real" than the data collected in the benchmark datasets in Table 1.

To further compare, 50 testing images (http://vasc.ri.cmu.edu/demos/faceindex/ Submissions 1–13 on 19, October, 2002 and Submissions 4–40 on 18, October, 2002) were

sent to CMU face detection test program (http://www.vasc.ri.cmu.edu/cgi-bin/demos/findface.cgi) for analysis. The false positive rate obtained from the 50 testing images set was 58% and the number of false face images detected was 28. In FDAO system, the false positive rate obtained on the same 50 testing images set was 20% and the number of false face images detected was 8. Some of the detected faces by CMU (left two pictures) and S-AdaBoost system (right two pictures) are shown in Figure 6 (CMU program has 2 correct detections and 1 wrong detection in the first picture and 1 wrong detection in the second picture, whereas, S-AdaBoost has 3 correct detections in the first picture and no wrong detection in the second picture).

3.4. AdaBoost divider and the threshold v value in FADO

The AdaBoost divider plays a very important role in the S-AdaBoost architecture. From the algorithm described in Section 2.3, it is observed that initially all the training patterns are assigned equal distribution weights (in L1). After certain rounds of iterations, the *difficult* patterns are assigned higher distribution weight (in L2); if the distribution weights exceed a threshold value **v**, S-AdaBoost treats those training pattern as outliers (in L3), which include the patterns with noise and the hard-to-classify patterns.

To test how good AdaBoost is at separating the patterns and to further analyze the influence of the threshold **v** on the overall performance of the system, a series of experiments was conducted. Through choosing different threshold **v** values, different sets of $\mathbf{T_{od}}(\mathbf{x})$ and $\mathbf{T_{ol}}(\mathbf{x})$ were generated, and different S-AdaBoost machines were thus trained to generate the corresponding test results. To measure the effectiveness of the S-AdaBoost machine, two error rates were measured, namely, the false positive rate as well as the detection error rate $\boldsymbol{\delta}$ defined in Section 3.3. The experimental results are shown in Figure 7.

In Figure 7, the Y-axis denotes the error rate, while X-axis (not proportional) denotes the value of threshold **v**. It is found that with the threshold **v** gradually increased from 0 (when all patterns were treated as outliers), the error rates of S-AdaBoost decreased slowly, then the error rates dropped faster and became stable for a while before they went up slowly (finally, the false positive rate reached $\boldsymbol{\partial}$ and the detection error rate reached $\boldsymbol{\delta}$). After examining the patterns in $\mathbf{S_{ol}}$ for different threshold values, it was observed that when threshold **v** was small, most of the patterns in **S** were in $\mathbf{S_{ol}}$, and the system's generalization characteristic was poor, which resulted in high error rates. Along with the increment of threshold **v**, more and more $\mathbf{P_{no}}$ and $\mathbf{P_{sp}}$ were divided into $\mathbf{S_{od}}$ and more genuine clusterings of $\mathbf{P_{hd}}$ were detected in $\mathbf{S_{ol}}$; the error rates went down faster and then reached an optimal range with threshold **v** increased further; some $\mathbf{P_{hd}}$ and $\mathbf{P_{ns}}$ patterns divided into $\mathbf{S_{od}}$; $\mathbf{T_{od}}(\mathbf{x})$ tried progressively harder to adopt these outlying patterns, which resulted in slow rising of error rates. The false positive rate reached $\boldsymbol{\partial}$ and detection error rate reached $\boldsymbol{\delta}$ when all the patterns in **S** were divided into $\mathbf{S_{od}}$ like the experiments described in Section 2.6. Testing results showed that S-AdaBoost performed reasonably well when the value of threshold **v** was around $1/(M \times \boldsymbol{\partial}^2)$, where M was the number of training patterns.

Figure 6: Faces detected by CMU program and S-AdaBoost.

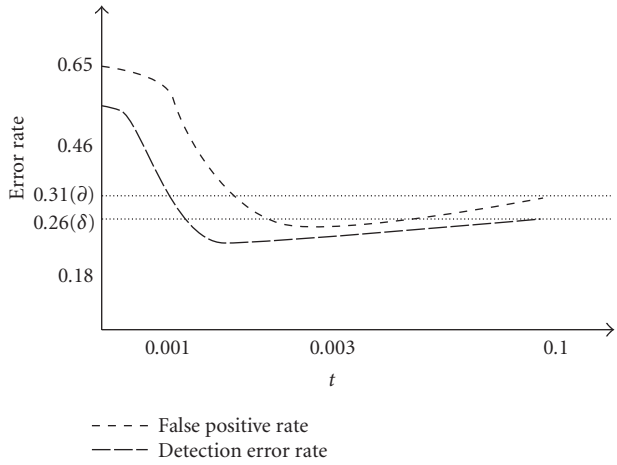

Figure 7: Error rates.

4. DISCUSSION AND CONCLUSIONS

S-AdaBoost, a new variant of AdaBoost, is more effective than the conventional AdaBoost in handling outliers in real-world complex environment. FDAO is introduced as a practical system to support the above claim. Experimental results on benchmark databases and comparison with other leading face detection methods on FDAO datasets clearly show S-AdaBoost's effectives in handling pattern classification application in complex environment and FDAO's capability in boosting face detection in airport environment. Future improvements will focus on theory exploration of the threshold value and better understanding of the dividing mechanism in the S-AdaBoost architecture.

REFERENCES

[1] P. Viola and M. Jones, "Fast and robust classification using asymmetric AdaBoost and a detector cascade," in *Neural Information Processing Systems*, pp. 1311–1318, Vancouver, British Columbia, Canada, December 2001.

[2] M.-H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, 2002.

[3] S. Z. Li, L. Zhu, Z. Q. Zhang, A. Blake, H. J. Zhang, and H. Shum, "Statistical learning of multi-view face detection," in *Proc. 7th European Conference on Computer Vision*, pp. 67–81, Copenhagen, Denmark, May 2002.

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

[5] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 130–136, San Juan, Puerto Rico, June 1997.

[6] L. G. Valiant, "A theory of the learnable," *Communications of the ACM*, vol. 27, no. 11, pp. 1134–1142, 1984.

[7] R. E. Schapire, "The strength of weak learnability," *Journal of Machine Learning Research*, vol. 5, no. 2, pp. 197–227, 1990.

[8] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. 13th International Conference on Machine Learning*, pp. 148–156, Bari, Italy, July 1996.

[9] T. G. Dietterich and E. B. Kong, "Machine learning bias, statistical bias, and statistical variance of decision tree algorithms," Tech. Rep., Department of Computer Science, Oregon State University, Corvallis, Ore, USA, 1995, http://web.engr.oregonstate.edu/~tgd/publications/index.html.

[10] J. R. Quinlan, "Bagging, boosting, and C4.5," in *Proc. 13th National Conference on Artificial Intelligence*, pp. 725–730, Portland, Ore, USA, August 1996.

[11] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization," *Journal of Machine Learning Research*, vol. 40, no. 2, pp. 139–157, 2000.

[12] A. J. Grove and D. Schuurmans, "Boosting in the limit: maximizing the margin of learned ensembles," in *Proc. 15th National Conference on Artificial Intelligence*, pp. 692–699, Madison, Wis, USA, July 1998.

[13] G. Rätsch, "Ensemble learning methods for classification," M.S. thesis, Department of computer Science, University of Potsdam, April 1998.

[14] W. Jiang, "Some theoretical aspects of boosting in the presence of noisy data," in *Proc. 18th International Conference on Machine Learning*, pp. 234–241, San Francisco, Calif, USA, June 2001.

[15] A. Krieger, C. Long, and A. Wyner, "Boosting noisy data," in *Proc. 18th International Conference on Machine Learning*, pp. 274–281, Williamstown, Mass, USA, January 2001.

[16] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," Tech. Rep., Stanford University, Stanford, Calif, USA, 1998.

[17] Y. Freund, "An adaptive version of the boost by majority algorithm," in *Proc. 12th Annual Conference on Computational Learning Theory*, pp. 102–113, Santa Cruz, Calif, USA, 1999.

[18] C. Domingo and O. Watanabe, "MAdaBoost: a modification of AdaBoost," in *Proc. 13th Annual Conference on Computational Learning Theory*, pp. 180–189, Sydney, Australia, December 2000.

[19] G. Rätsch, T. Onoda, and K.-R. Müller, "Soft margins for AdaBoost," *Journal of Machine Learning Research*, vol. 42, no. 3, pp. 287–320, 2001.

[20] G. Rätsch, http://www.first.gmd.de/~raetsch/.

[21] UCI Machine Learning Repository, http://www1.ics.uci.edu/~mlearn/MLRepository.html.

[22] Data for Evaluating Learning in Valid Experiments, http://www.cs.toronto.edu/~delve/.

[23] The StatLog Repository, http://www.liacc.up.pt/ML/statlog/.

Jimmy Liu Jiang received his B.S. degree in Computer Science from the University of Science and Technology of China in 1988, and his M.S. degree in computer science from the National University of Singapore in 1992, specialized in pattern recognition and artificial intelligence. From 1999 to 2003, he completed the Ph.D. degree study in the National University of Singapore, specialized in imperfect data learning. His current research interests include image understanding and bioinformatics.

Kia-Fock Loe is an Associate Professor in the Department of Computer Science at the National University of Singapore. He obtained his Ph.D. degree from the University of Tokyo. His current research interests are neural network, machine learning, pattern recognition, computer vision, and uncertainty reasoning.

Hong Jiang Zhang received his Ph.D. degree from the Technical University of Denmark and his B.S. from Zhengzhou University, China, both in electrical engineering, in 1991 and 1982, respectively. From 1992 to 1995, he was with the Institute of Systems Science, National University of Singapore, where he led several projects in video and image content analysis and retrieval and computer vision. He also worked at MIT Media Lab in 1994 as a Visiting Researcher. From 1995 to 1999, he was a Research Manager at Hewlett-Packard Labs, where he was responsible for research and technology transfers in the areas of multimedia management, intelligent image processing, and Internet media. In 1999, he joined Microsoft Research Asia, where he is currently a Senior Researcher and Assistant Managing Director in charge of media computing and information processing research. Dr. Zhang has authored 3 books, over 260 referred papers, 7 special issues of international journals on image and video processing, content-based media retrieval, and computer vision, as well as over 50 patents or pending applications. He currently serves on the editorial boards of five IEEE/ACM journals and a dozen committees of international conferences.