JMLR: Workshop and Conference Proceedings 20 (2011) 335–349 Asian Conference on Machine Learning

Acceleration technique for boosting classification and its application to face detection

Masanori Kawakita KAWAKITA(AT)INF.KYUSHU-U.AC.JP
Ryota Izumi IZUMI(AT)KAIRO.CSCE.KYUSHU-U.AC.JP
Jun'ichi Takeuchi TAK(AT)INF.KYUSHU-U.AC.JP
Graduate school of Information Science and Electrical Engineering, Kyushu University
Institute of Systems, Information Technologies and Nanotechnologies

Yi Hu YI_HU(AT)FUJIFILM.CO.JP
Tetsuya Takamori TETSUYA_TAKAMORI(AT)FUJIFILM.CO.JP
Hirokazu Kameyama HIROKAZU_KAMEYAMAG(AT)FUJIFILM.CO.JP
Imaging Technology Center, Research & Development Management Headquarters, FUJIFILM Corporation 798, Miyanodai, Kaisei-machi, Asigarakami-gun, Kanagawa 258-8538 Japan

Editor: Chun-Nan Hsu and Wee Sun Lee

Abstract

We propose an acceleration technique for boosting classification without any loss of classification accuracy and apply it to a face detection task. In classification task, much effort has been spent on improving the classification accuracy and the computational cost of training. In addition to them, the computational cost of classification itself can be critical in several applications including face detection. In face detection, a celebrating work by Viola and Jones (2001) developed a significantly fast face detector achieving a competitive accuracy with all preceding face detectors. In their algorithm, the cascade structure of boosting classifier plays an important role. In this paper, we propose an acceleration technique for boosting classifier. The key idea of our proposal is the fact that one can determine the sign of discriminant function before all weak learners are evaluated in general. An advantage is that our algorithm has no loss in classification accuracy. Another advantage is that our proposal is a unsupervised learning so that it can treat a covariate shift situation. We also apply our proposal to each cascaded boosting classifier in Viola and Jones type face detector. As a result, our proposal succeeds in reducing the classification cost by 20%.
Keywords: boosting, classification cost, truncation rule, face detection

1. Introduction

In the study of classification task, researchers have been mainly focused on classification accuracy and/or computational cost of training. In addition to them, the computational cost of classification (abbreviated as *classification cost* in the sequel) has attracted researchers' interests in several application areas. One of such application is face detection with digital cameras. The goal of this task is to specify where are human faces in a screen of digital cameras. Throughout the paper, we assume that only 1 CPU is available (parallel computing is not considered). One of widely used approach is preparing a face detector (classifier) and

applying it to a subwindow of a certain size. This procedure is repeatedly done with varying the location and the scale of subwindow exhaustively (illustrated in Fig. 1). As a result, the face detector is applied enormously many times (say, a few millions) to scan only an image. This task should be completed in a few tens of milliseconds for standard size images (say 600×400). If not, the user may feel very stressful because of the delay of screen refresh. Thus, the classification cost is a critical issue in this application.

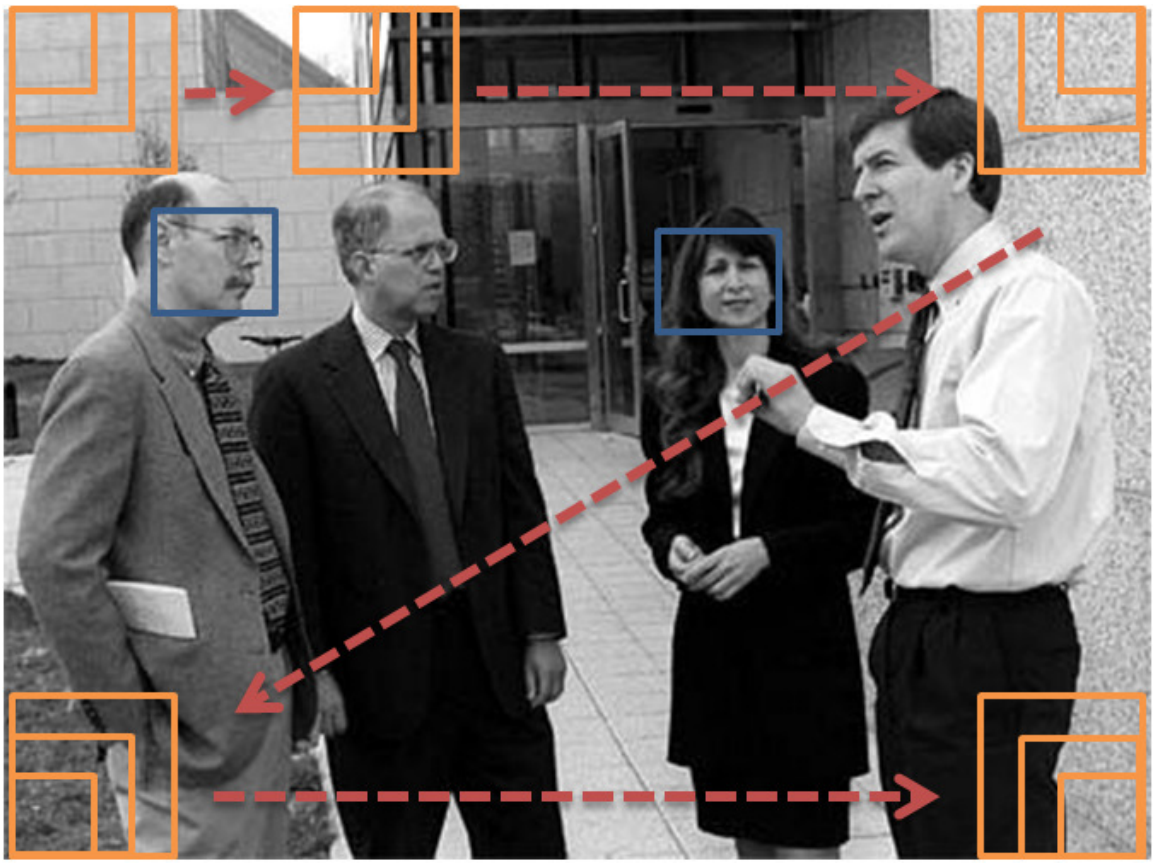

Figure 1: Scanning an image by shifting and scaling the subwindow. This image is chosen from CMU Profile Face Images (Schneiderman and Kanade, 2000). The blue subwindows detected some faces.

Motivated by this back ground, we propose an acceleration technique for boosting classifiers (Freund and Schapire, 1997; Mason et al., 2000; Friedman et al., 2000). Boosting is one of the most successful classifiers in the last decade. Currently, most of face detectors use boosting classifier. Given an input vector x, the boosting classifier predicts its label y according to a weighted majority vote (i.e., linear combination) of some weak classifiers. Taking this structure into account, we propose two new simple acceleration techniques. The key idea is that one does not necessarily need to evaluate all weak classifiers to obtain the exact classification result. Based on this idea, our first proposal (referred to as γ-truncation

rule) is to truncate the evaluation of weak classifiers up to the necessary minimum number. When γ-truncation rule is applied, the number of weak classifier to be evaluated depends on the order of weak classifiers and the input x. Therefore, our second proposal is to optimize the order of weak classifiers such that evaluation of weak classifiers is truncated as early in average (with respect to x) on an available data set as possible. We refer to this technique as order structure learning (OSL).

An advantage is that our proposal (γ-truncation rule combined with OSL) achieves the acceleration without no loss of classification accuracy because the classification result is exactly the same as the original boosting classifier. Another advantage is that our proposal can treat a covariate shift situation (e.g., Shimodaira (2000)). The covariate shift indicates that the input distribution of training data differs from that of test data. If additional unlabeled data (i.e., only x) generated from the test distribution are available, then OSL can be trained with this data set because OSL is an unsupervised learning. The performance of our proposal is illustrated by some simulations.

We apply our proposal to an existing face detection system. The state-of-the-art face detectors originated from a celebrating work by Viola and Jones (2001). Face detection task is essentially involved with covariate shift situation. For example, there are usually few faces in an image of digital cameras. Thus, when scanning an image by subwindow, most subwindows are negative samples (non-face). In contrast, a face detector is usually trained with a data set containing faces with larger proportions to detect a variety of faces. Based on this fact, Viola and Jones (2001) proposed a fast face detection system as follows. They prepare a sequence of boosting classifiers and make a decision tree of specific structure (so-called cascade) with boosting classifiers as nodes. Each input (subwindow) x is judged as face if and only if all nodes judge it as "face." The key point is that the more ascendant node is designed to be computationally simper and to have extremely high detection rate. In other words, many definitely non-face subwindows are rejected early with low computational cost. Their face detector is approximately 15 times faster than preceding face detectors while keeping the detection rates comparable with them. As a result, many current face detectors are variants of their method. We employ one of such variants (Lienhart and Maydt, 2002a). Their algorithm is available at Lienhart and Maydt (2002b). As a result of applying our proposal to each node (boosting classifier), we achieve 20% decrease in classification cost.

We should mention a preceding study. After this paper was accepted, we found that Utsumi et al. (2010) proposed an acceleration technique for face recognition. It turned out that their proposal contained the γ-truncation rule. To our best knowledge, however, the order structure learning is exactly original.

This paper is organized as follows. In Section 2, we propose a new acceleration technique for boosting classification. Section 3 illustrates the performance of our proposal by simulations. In section 4, we apply our proposal to a face detection system. Section 5 describes the conclusion and the future work.

2. Acceleration technique

We propose an acceleration technique for boosting classification by learning from a data set.

2.1. Setting

We introduce some notations to describe our setting. Let $\mathscr{X} \subset \Re^d$ and $\mathscr{Y} := \{-1, 1\}$. Suppose that we are given a boosting classifier $g(x)$ for a classification task in which one has to predict a label $y \in \mathscr{Y}$ of a given input $x \in \mathscr{X}$. The boosting classifier takes the following form:

$$g(x) := \mathrm{sign}(F(x)), \quad F(x) := \alpha_1 f_1(x) + \alpha_2 f_2(x) + \cdots, \alpha_J f_J(x), \tag{1}$$

where $\alpha_j \in \Re_+$ and $f_j(x)$ is any types of weak classifier for each j. Note that each f_j also takes its value in $\{-1, 1\}$ and that $\mathrm{sign}(x)$ is defined as a function taking 1 if $x \geq 0$ and 0 otherwise. The function $F(x)$ is called a *discriminant function*. The classification result of boosting classifier is decided by the sign of discriminant function $F(x)$. Taking into account that only 1 CPU is available, we postulate that weak classifiers in $F(x)$ is sequentially evaluated in the order in Eq. 1. Thus, F corresponds to not only a function of x but also its representation as a list (ordered sequence) $(\alpha_1 f_1, \alpha_2 f_2, \cdots, \alpha_J f_J)$. Let us write a computational cost to evaluate $F(x)$ as $C(x; F)$. Assume that the test data to be classified are subject to the distribution $p(x)$. Note that $p(x)$ is not necessarily the same with the distribution of training data used for boosting. Our goal is to reduce the average computational cost $E_{p(x)}[C(X; F)]$ as much as possible without loss of classification accuracy when we are given $F(x)$ and a set of samples $D := \{x_1, x_2, \cdots, x_n\}$ generated from $p(x)$. To our knowledge, this kind of problem is somewhat new to a field of machine learning.

2.2. γ-truncation rule

In this section, we introduce the first acceleration technique. The idea is so simple. To decide the sign of boosting discriminant function $F(x) = \sum_{j=1}^{J} \alpha_j f_j(x)$, one needs not to evaluate all weak classifiers $\{f_j(x)\}$ in general. To explain this, we introduce the following notations for each integer $M \in \{1, 2, \cdots, J\}$:

$$\begin{aligned} \overline{F}_M(x) &:= \sum_{j=1}^{M} \alpha_j f_j(x) \\ \widetilde{F}_M(x) &:= \sum_{j=M+1}^{J} \alpha_j f_j(x) \end{aligned}$$

Suppose a situation that we evaluated the first M weak classifiers at a fixed x. That is, we have the value of $\overline{F}_M(x)$. If the condition

$$\text{Truncate Condition:} \quad |\overline{F}_M(x)| > \max_{x' \in \mathscr{X}} |\widetilde{F}_M(x')| = \sum_{j=M+1}^{J} \alpha_j \tag{2}$$

holds, then it clearly holds that $\mathrm{sign}(\overline{F}_M(x)) = \mathrm{sign}(F(x))$. Therefore, we need not to evaluate the remaining weak classifiers $\{f_{M+1}(x), f_{M+2}(x), \cdots, f_J(x)\}$. Besides, we need not to calculate $\sum_{j=M+1}^{J} \alpha_j$ of the right-hand side of Eq. (2) in detection because this

quantity can be calculated for every M in advance and kept in memory. Define the minimum of integers satisfying the truncate condition as $\gamma(x, F)$, i.e.,

$$\gamma(x, F) := \min \left\{ M \in \{1, 2, \cdots, J\} \,\middle|\, \overline{F}_M(x) > \sum_{j=M+1}^{J} \alpha_j \right\}. \tag{3}$$

We refer to $\gamma(x, F)$ as "minimum evaluation number" at x. For each x, we need to evaluate only the first $\gamma(x, F)$ weak classifiers. Then, the classification result is obtained as $\mathrm{sign}(\overline{F}_{\gamma(x,F)}(x))$. We refer to this method as γ-truncation rule. Clearly, γ-truncation rule causes exactly no loss in classification accuracy compared to $g(x)$. Now, we define a computational cost $C(x, F)$ as the number of evaluated weak classifiers (ignoring a difference of the computational cost among weak classifiers). Then, as for the usual boosting classification, $C(x, F)$ is always J so that the average computational cost $E_{p(x)}[C(x, F)]$ is also J while $E_{p(x)}[\gamma(x, F)]$ for γ-truncation rule, which is smaller than J in general.

2.3. Order structure learning

In this section, we propose one more acceleration technique. In γ-truncation rule, the minimum truncating number $\gamma(x, F)$ depends on x and the order of weak classifiers in F. Thus, our second proposal is to optimize the order of weak classifiers such that $\gamma(x, F)$ is minimum on the average with respect to $p(x)$. To explain this, let us introduce some notations. Let $\mathscr{K}$ be a set of all permutations of $\{1, 2, \cdots, J\}$, i.e.,

$$\mathscr{K} := \{k : \{1, 2, \cdots, J\} \to \{1, 2, \cdots, J\} \,|\, k \text{ is one-to-one correspondence}\}.$$

For each $k \in \mathscr{K}$, a discriminant function F ordered by k is denoted by

$$F^k(x) := \sum_{j=1}^{J} \alpha_{k(j)} f_{k(j)}(x).$$

Our goal is to learn the order of weak classifiers as minimizing $E_{p(x)}[\gamma(x, F)]$. Since p is unknown, we consider its empirical estimates defined by

$$\Gamma(F) := \frac{1}{n} \sum_{i=1}^{n} \gamma(x_i, F).$$

We refer to Γ as "averaged minimum evaluation number." Our second proposal is to learn the order of weak classifiers as minimizing Γ, i.e.,

$$\hat{k} := \operatorname*{argmin}_{k \in \mathscr{K}} \Gamma(F^k). \tag{4}$$

We refer to this algorithm as an *order structure learning* (OSL) in the sequel. If one wants to optimize by the brute force search, the evaluation of Γ should be done $J!$ times. When we use a PC, this can be done at most for $J = 15$. Thus, we propose an approximation algorithm using transposition for search task. Let $\overline{\mathscr{K}}$ be a set of all transpositions:

$$\overline{\mathscr{K}} := \{k(j_1, j_2) \,|\, j_1, j_2 \in \{1, 2, \cdots, J\}\},$$

where

$$k(j_1, j_2) : j \mapsto \begin{cases} j_2 & j = j_1 \\ j_1 & j = j_2 \\ j & \text{otherwise} \end{cases}.$$

The composite map of two transpositions $k_1, k_2 \in \overline{\mathscr{K}}$ is denoted by $k_1 \cdot k_2$. We further define a set of all permutations which is realized by a composite of q transpositions:

$$\overline{\mathscr{K}}(q) := \{k_1 \cdot k_2 \cdots k_q \,|\, \forall \ell,\ k_\ell \in \overline{\mathscr{K}}\}.$$

Using these notations, the proposed algorithm is described in Algorithm 1. The number q

Algorithm 1: Approximating algorithm OSL(q)

Input: $F(x)$, q, D, initial order $z \in \mathscr{K}$
Output: $\hat{k}$

1. Initialize $k_0 := z$.
2. For $t = 1, 2, \cdots$,
   (a) Search $k' := \mathrm{argmin}_{k \in \overline{\mathscr{K}}(q)} \Gamma(F^{k \cdot k_{t-1}})$ exhaustively.
   (b) Set $k_t := k' \cdot k_{t-1}$.
   (c) If $\Gamma(F^{k_{t-1}}) = \Gamma(F^{k_t})$, then output $\hat{k} := k_t$ and halt. Otherwise, back to (a).

should be chosen as large as possible unless OSL(q) does not finish in practical time. This algorithm is a sort of hill-climbing optimization and the obtained solution can not be a global minimum for some initial orders. We investigated the dependency on initial order by some numerical experiments. As a result, the choice of initial order did not affect the acceleration performance much but affected the convergence speed to some extent. Taking this into account, we prepare two candidates of initial order z. One is an decreasing order with respect to the weights $\{\alpha_j\}$. The other is just an order chosen by boosting. These orders are somewhat effective and close to the optimal order in some cases because the truncate condition is likely to hold. Starting from these initial order, OSL is expected to converge with relatively fewer steps.

Finally, we emphasize that OSL is an unsupervised learning because OSL does not require the label information in its training process. This allows OSL to treat covariate shift situation. Suppose that the training data $D'_L := \{(x'_i, y'_i) \,|\, i = 1, 2, \cdots, n\}$ to construct a boosting detector are generated from $p'(x)p(y \,|\, x)$ and the test data in detection is generated from $p(x)p(y \,|\, x)$. Covariate shift indicates that $p \neq p'$. In this case, it is not guaranteed that the above initial orders are close to optimal. Generally, it is much easier to collect the unlabeled data (i.e., only x) than the labeled data (i.e., a pair of x and y), as is often said in semi-supervised learning. Thus, it is possible that some unlabeled data $D_U := \{x_1, x_2, \cdots, x_n\}$ generated from $p(x)$ are available. Since OSL is unsupervised learning, OSL can be trained with D_U. Therefore, OSL can optimize the order taking a target distribution $p(x)$ into account.

3. Simulation

We illustrate the performance of our proposal by two simulations.

3.1. Simulation 1

To illustrate the performance of γ-truncation rule and OSL, we did a simple simulation according to the following procedure.

1. Construct a discriminant function $F^*(x)$ with 50 different weak classifiers ($x \in \Re^2$). Its shape is shown in Fig. 2. In this $F^*(x)$, few weak classifiers (group 1) have large coefficients, while the remaining weak classifiers (group 2) have small coefficients. In the region R_1, only group 1 classifiers are evaluated because γ-truncation rule is applied. In R_2, however, each output of group 1 classifiers cancels out one another. Therefore, we need to evaluate group 2 weak classifiers. Thus, we can speed up the classification in R_1 more than R_2. (See also Fig. 3).

2. Let $f_1(x)$ and $f_2(x)$ be uniform distributions in R_1 and R_2. The marginal distribution of x is defined as
$$f(x;p) := pf_1(x) + (1-p)f_2(x),$$
where $p \in [0,1]$ is the occurrence probability of $x \in R_1$.

3. The conditional distribution of y is defined as
$$f(y \,|\, x) := \frac{\exp(2y'F^*(x))}{1+\exp(2F^*(x))} \text{ with } y' := (y+1)/2.$$

4. For each $p = 0.1, 0.2, \cdots, 1$,
   (a) Generate 2000 i.i.d. samples $D_L := \{(x_i, y_i) \,|\, i = 1, 2, \cdots, 2000\}$ from the joint distribution $f(x, y; p) := f(x; \beta) f(y \,|\, x)$.
   (b) Construct a discriminant function F by AdaBoost with D_L.
   (c) Learn the order structure by $OSL(2)$ with the x-part of D_L.
   (d) Generate D'_L similarly to D_L and calculate the averaged minimum evaluation number Γ for the three orders, say, the weight decreasing order, the boosting order and the order learned by OSL.

See the bottom of Section 2.3 for the definitions of weight decreasing order, and the boosting order. If there is a weak classifier appears more than one time, then their weights are summarized into that of first weak classifier in preparing these two orders.

The right panel of Fig. 3 shows the results. As p approaches to one, the classification cost is reduced to a half. This is because as the data distribute in R_1 with larger probability as p gets larger. It is easy to see that only the γ-truncation rule with initial orders (weight decreasing order, boosting order) works well to some extent. However, OSL succeeds in reducing the classification cost by 10% more when p is large.

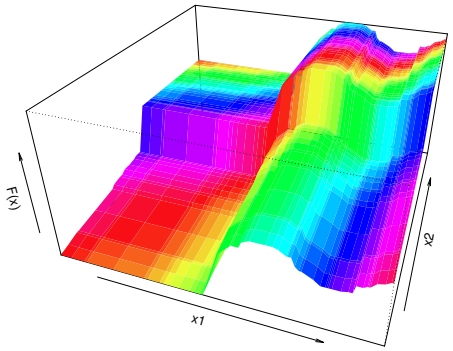

Figure 2: Plot of $F^*(x)$. The zero height ($F(x) = 0$) is just the center of z-axis.

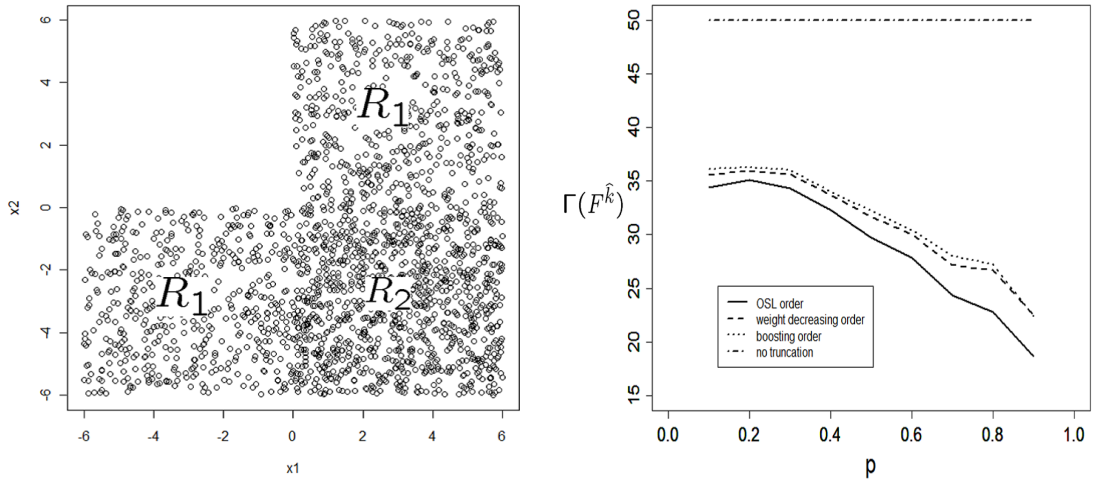

Figure 3: Simulation 1: the left panel shows a plot of the training data D_L, while the right panel shows a plot of averaged minimum evaluation number Γ against p, the occurrence probability of region R_1.

3.2. Simulation 2

Next, we assume covariate shift situation. First, we construct a boosting classifier $F(x)$ in the same way with the previous section with $p = 0.5$. Then, OSL is done with the following procedure:

1. Generate 2000 i.i.d. samples $D_U := \{x'_1, x'_2, \cdots, x'_{2000}\}$ from a distribution

$$p(x) := \frac{p}{2}N(x; \mu_1, 0.3I) + \frac{p}{2}N(x; \mu_2, 0.3I) + (1-p)N(x; \mu_3, 0.3I),$$

   where $N(x; \mu, \Sigma)$ is a normal distribution with mean μ and a covariance matrix Σ, I denotes an identity matrix, $\mu_1 = (3, 3)^T, \mu_2 = (-3, -3)^T$ and $\mu_3 = (3, -3)^T$.

2. Learn the order structure by $OSL(2)$ with D_U.

3. Generate D'_U similarly to D_U and calculate the averaged minimum evaluation number Γ for the weight decreasing order, the boosting order and the OSL order.

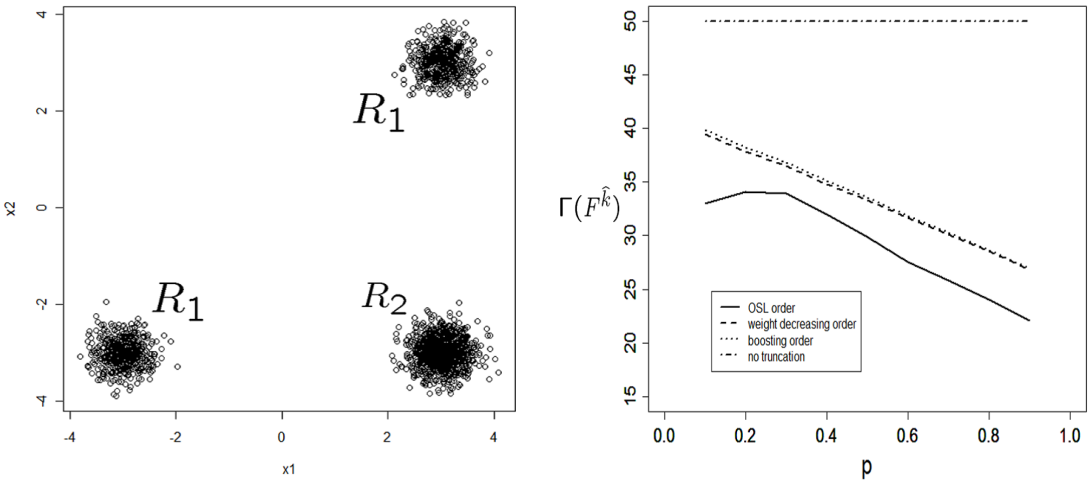

Figure 4: Plot of average minimum evaluation number Γ against p in simulation 2.

The data D_U are plotted in the left panel in Fig. 4. As seen in the right panel in Fig. 4, γ-truncation rule with the weight decreasing order and the boosting order are improved by OSL for any p. In covariate shift situation, it is not guaranteed that these initial order is close to the optimal order in general. The reason is simple. Clearly, the optimal order satisfying Eq. (4) depends on $p(x)$ strongly. Even when $p(x)$ varies, if we have unlabeled data D_U, then OSL learns the nearly optimal order. However, both the weight decreasing order and the boosting order are fixed when $F(x)$ is fixed. Thus, OSL is indispensable in covariate shift situation.

4. Application to face detection

In this section, we apply our acceleration technique to an existing face detector. First, we review one of current face detectors, which is a variant of Viola and Jones (2001). Second, we propose a slight extension of our acceleration technique to apply it to this face detector. Third, we observe by experiments how much is the face detector accelerated by our proposal.

4.1. Conventional face detection

In this section, we briefly review the conventional face detector. Many current face detectors originated from an celebrating work by Viola and Jones (2001). We refer to their face detector as VJ face detector in short. VJ face detector consists of multiple classifiers $\{g_1(x), g_2(x), \cdots, g_n(x)\}$ with a cascade structure illustrated in Figure 5. When an input

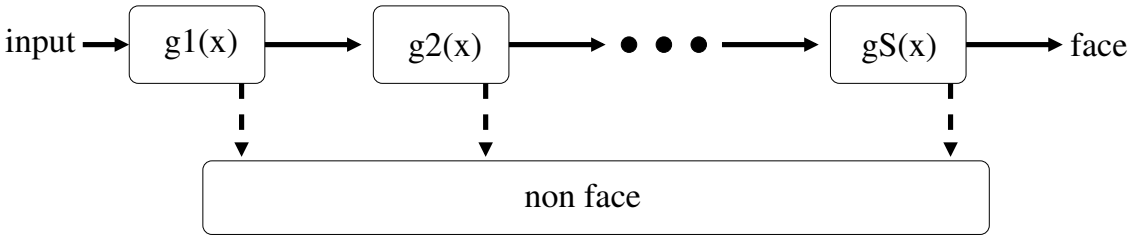

Figure 5: Cascade structure of boosting classifiers. The solid arrow adjoined with each classifier $g_i(x)$ indicates that g_i judges x as "face", while the dotted line indicates that g_i judges as "non face".

subwindow is given, classifiers are evaluated sequentially. Only a positive result (judged as face) of the previous classifiers triggers the next classifier. In other words, a negative result (non face) of any classifier truncates the evaluation and the final judgement is negative. Each classifier is trained by AdaBoost (Freund and Schapire, 1997) with decision stumps as weak classifiers with Haar-like features (Papageorgiou et al., 1998).

Viola and Jones (2001) provided a fast algorithm to calculate Haar-like features. Their most important contribution is the introduction of the cascade structure. In the cascade structure, each boosting classifier is designed such that the former classifier is computationally simpler and has higher detection rate. For this purpose, boosting algorithm is quite suitable because it enables us to control the computational complexity easily by tuning the iteration number. As described in Section 1, an image of digital cameras usually contain only few faces so that a majority of subwindows are non faces. VJ face detector rejects such non face subwindows quickly by the former classifiers (of low computational cost) without many errors (high detection rate). As a result, VJ face detector is approximately 15 times faster than previous face detectors while keeping the detection rates comparable with them.

In this paper, we employ a similar but different face detector by Lienhart and Maydt (2002a) (referred to as a LM face detector). There are two reasons. One reason is that their implementation is open at Lienhart and Maydt (2002b) so that it is somewhat widely used. In addition, conventional face detectors involve many parameters to be tuned manually in general. The details of such parameters are usually not described. Their code offers such informations so that comparisons can be made. Another reason is that LM face detector is an improved version of Viola and Jones (2001). The differences between them are the following three points:

(i) features: In addition to Haar-like features used in VJ face detector, LM face detector employs a rotated version of Haar-like features to achieve higher detection rate.

(ii) boosting algorithms: In stead of AdaBoost, LM face detector employs Gentle AdaBoost (Friedman et al., 2000), which is often more stable than AdaBoost.

(iii) weak classifiers: Instead of decision stump, LM face detector employs a decision tree. As a result, Lienhart and Maydt (2002a) reported that they succeeded in reducing the computational cost with a competitive classification accuracy.

By modifications (ii) and (iii), a discriminant function of LM face detector takes a form similar to but different from that of AdaBoost. In case of AdaBoost, $F(x)$ in Eq. (1) consists of a linear combination of (± 1-valued) weak classifier $f_j(x)$ and its linear coefficient α_j. In LM face detector, $F(x)$ consists of a sum of decision trees of depth 1 (without linear coefficient). That is, $F(x)$ takes a form:

$$F(x) := \sum_{j=1}^{J} f_j(x),$$

where $f_j(x)$ takes not only $-1, 1$ but possibly any two real values $\{\ell_j, u_j\}$ decided by CART (Breiman et al., 1984). To apply to this discriminant function, γ-truncation rule needs to be modified slightly.

4.2. Extension of γ-truncation rule

In this section, we extend γ-truncation rule. As described in the previous section, $F(x)$ used in LM face detector consists of

$$F(x) = \sum_{j=1}^{J} f_j(x), \text{where } f_j(x) \text{ takes } \{\ell_j, u_j\}.$$

Thus, γ-truncation condition in Eq. (2) should be modified as follows. Define the following notations again:

$$\overline{F}_M(x) := \sum_{j=1}^{M} f_j(x), \ \widetilde{F}_M(x) := \sum_{j=M+1}^{J} f_j(x).$$

The truncate condition is written as $\overline{F}_M(x) + \min_{x' \in \mathcal{X}} \widetilde{F}(x') > 0$ or $\overline{F}_M(x) + \max_{x' \in \mathcal{X}} \widetilde{F}(x') < 0$. Note that

$$\min_{x \in \mathcal{X}} \widetilde{F}_M(x) = \sum_{j=M+1}^{J} \min\{\ell_j, u_j\}, \ \max_{x \in \mathcal{X}} \widetilde{F}_M(x) = \sum_{j=M+1}^{J} \max\{\ell_j, u_j\}.$$

Thus, truncate conditions should be

$$\overline{F}_M(x) \ > \ -\sum_{j=M+1}^{J} \min\{\ell_j, u_j\} \tag{5}$$

$$\overline{F}_M(x) \ < \ -\sum_{j=M+1}^{J} \max\{\ell_j, u_j\}. \tag{6}$$

If Eq. (5) holds, then $\text{sign}(F(x))$ is exactly positive, while $\text{sign}(F(x))$ is exactly negative if Eq. (6) holds regardless of $\widetilde{F}_M(x)$.

Table 1: The evaluation number of weak classifiers in the whole process of face detection. LM face detector indicates evaluating all weak classifiers in each stage.

	LM face detector	the boosting order	OSL
total evaluation number	$314,270,657$	$277,023,438$	$248,739,106$

4.3. Experiments

In this section, we apply our acceleration technique to LM face detector Lienhart and Maydt (2002b). LM face detector consists of 25 boosting classifiers in the cascade. A left panel of Fig. 7 provides the information how many weak classifiers each boosting classifier consists of. The initial size of subwindow is 40×40 and is enlarged by 11% up to the whole image size. The subwindows moves by the pixels of expanding magnification from the initial size.

We applied the proposed acceleration technique to each stage (boosting classifier) in the cascade. OSL was done by training data ($948,139$ subwindows) that were chosen randomly from a benchmark data set made by (Rowley et al., 1998). We chose 120 images ($7,190,806$ subwindows) from CMU Profile Face Images (Schneiderman and Kanade, 2000) as the test data on which we evaluated the averaged minimum evaluation number Γ. Some examples of test data images are shown in Fig. 6. We remark that we cannot specify which training data are used for constructing the boosting classifiers. This implies a covariate shift situation. The results are shown in Fig. 7. The right panel in Fig. 7 shows that γ-truncation rule

Figure 6: Some examples of images in CMU Profile Face images.

with OSL order decreases the classification cost of each stage by around 20%. It is observed that γ-truncation rule with the boosting order performed poorly in many stages. Under covariate shift situation, the boosting order does not perform enough so that OSL is quite necessary, as was already confirmed in simulations. We did not use the weight decreasing order because each weak classifier in LM face detector has no weight. It is also observed that the latter stage has the larger acceleration. In general, our acceleration technique improve the classification cost if the number of weak classifiers is large. In the cascade, the latter classifier contains the more weak classifiers. To see the effectiveness of acceleration in the whole, we need to count how many subwindows are received by each stage. It is seen in Fig. 8 that a majority of subwindows are rejected in the early stages. Based on this information, the evaluation number of weak classifier in total is summarized in Table 1. The total classification cost is also reduced by around 20% compared to LM face detector.

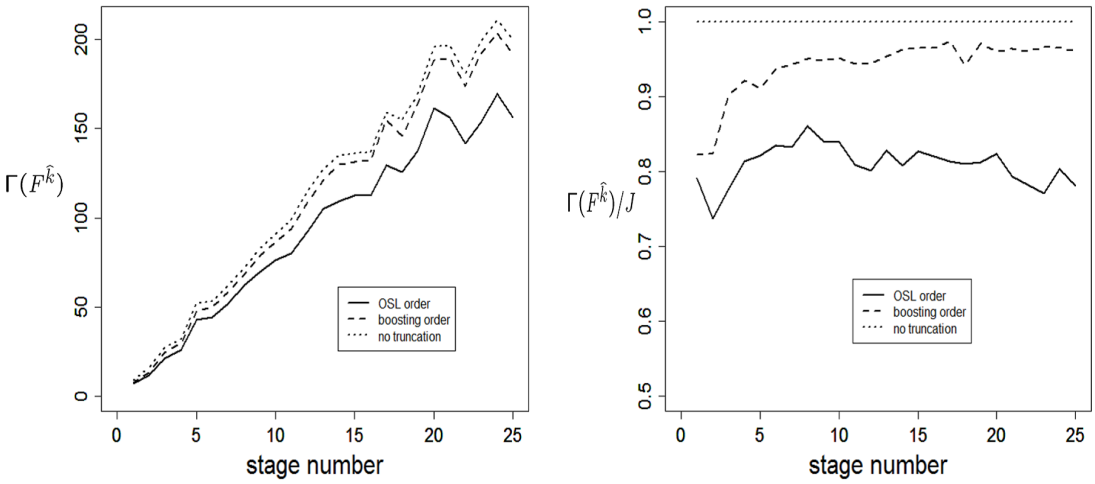

Figure 7: Results of experiments: the left panel shows the plot of Γ for each stage (classifier $g_i(x)$) in the face detector cascade. The right panel shows the same plot of a normalized average minimum evaluation number divided by the number of all weak classifiers in each stage.

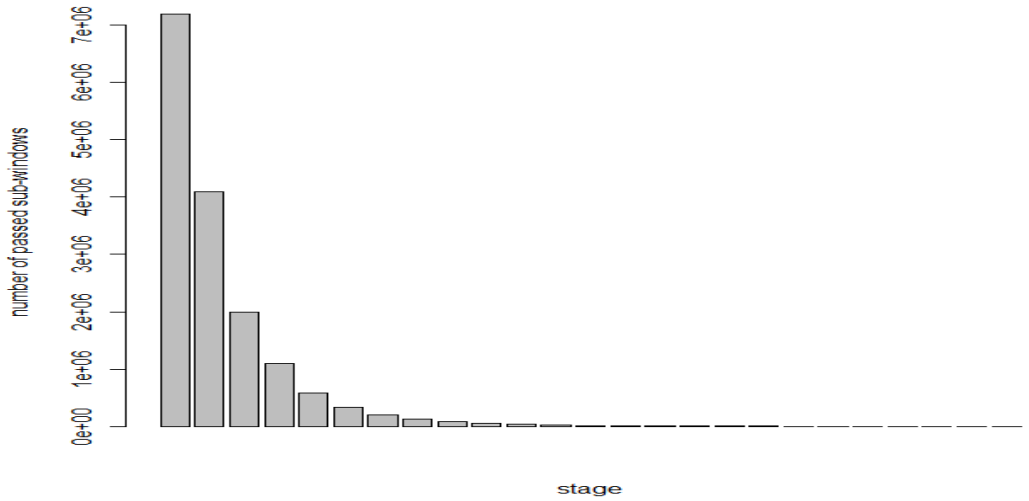

Figure 8: Plot of how many subwindows are received by each stage (classifier $g_i(x)$ in the cascade.

5. Conclusion and future work

We have proposed a new acceleration technique for evaluating the boosting classifier. This technique has no loss in classification accuracy. In application to the existing face detector, we achieve 20% decrease in classification cost.

An interesting future work is admitting a small loss in classification accuracy. By this, more acceleration would be possible. A more interesting future work is learning the order of all weak classifiers beyond the cascade structure. The cascade structure works significantly well but it is quite heuristic method. Another way is to construct a strong complicated boosting classifier and to learn the order structure by our proposal. This way is more adaptive to data and is expected to perform better with respect to the tradeoff between classification cost and accuracy.

Acknowledgments

The authors thank Dr. Yuzuko Utsumi for giving us the information of preceding studies. The authors also thank anonymous reviewers for various useful comments.

References

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees. Wadsworth, Belmont*, 1984.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28:337–407, 2000.

Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. *2009 IEEE International Conference on Image Processing*, 1:900–903, 2002a.

Rainer Lienhart and Jochen Maydt. Open computer vision library, 2002b.

L. Mason, J. Baxter, P. L. Bartlett, and M. Frean. Functional gradient techniques for combining hypotheses. *In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans editors, Advances in Large Margin Classifiers, MIT Press, Cambridge, MA*, pages 221–247, 2000.

C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. *International Conference on Computer Vision*, 1998.

H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 20:22–38, 1998. URL http://vasc.ri.cmu.edu/idb/html/face/profile_images/index.html.

Henry Schneiderman and Takeo Kanade. A statistical model for 3d object detection applied to faces and cars. *IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, 2000.

Yuzuko Utsumi, Yuta Matsumoto, and Yoshio Iwai. Effective face recognition by branch and bound method. *IPSJ*, 51:217–228, 2010.

P. Viola and M. Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. *Neural Information Processing Systems*, 14, December 2001.