A Comprehensive Study for Asymmetric AdaBoost and Its Application in Object Detection

GE Jun-Feng[1] LUO Yu-Pin[1]

Abstract Asymmetry is inherent in tasks of object detection where rare positive targets need to be distinguished from enormous negative patterns. That is, to achieve a higher detection rate, the cost of missing a target should be higher than that of a false positive. Cost-sensitive learning is a suitable way for solving such problems. However, most cost-sensitive extensions of AdaBoost are realized by heuristically modifying the weights and confidence parameters of the discrete AdaBoost. It remains unclear whether there is a unified framework to interpret these methods as AdaBoost, clarify their relationships, and further derive the superior real-valued cost-sensitive boosting algorithms. In this paper, according to the three different upper bounds of the asymmetric training error, we not only give a detailed discussion about the various discrete asymmetric AdaBoost algorithms and their relationships, but also derive the real-valued asymmetric boosting algorithms in the form of additive logistic regression with analytical solutions, which are denoted by Asym-Real AdaBoost and Asym-Gentle AdaBoost. Experiments on both face detection and pedestrian detection demonstrate that the proposed approaches are efficient and achieve better performance than the previous AdaBoost methods and discrete asymmetric extensions.

Key words Cost-sensitive learning, ensemble learning, asymmetric AdaBoost, object detection, pedestrian detection

In recent years, the popular machine learning method named AdaBoost has achieved tremendous practical success on object detection in computer vision field. The general principle of AdaBoost[1] is to linearly combine a series of weak classifiers to produce a superior classifier. Each weak classifier consists of a prediction and a confidence value and each sample in the training set has an associated weight. At each iteration, AdaBoost chooses the best weak classifier to minimize the upper bound of training error, increases the weights of wrongly classified training samples, and decreases the weights of correctly classified samples. Benefiting from this scheme, many AdaBoost based object detecting algorithms for face[2−5] and pedestrian[6−9] have been proposed with impressive performance in both speed and accuracy, following the famous face detector introduced by Viola[10].

However, in the tasks of object detection, rare positive targets have to be distinguished from enormous negative patterns. Therefore, asymmetry is usually inherent in such domain which requires more attention to the positive targets or to make the algorithm cost-sensitive to achieve a higher detection rate and moderate low false positive rate. Thus, there should be different treatment for false negatives (FN) and false positives (FP), that is, FN samples are penalized more than FP samples. Since AdaBoost aims at minimizing the bound of classification error which treats FP and FN equally, the symmetric AdaBoost algorithm is not optimal for object detection tasks.

To deal with the class imbalance problem in classification[11], various asymmetric extensions of AdaBoost have been proposed in the literature. However, most of them[5, 11−13] directly modify the weights and confidence parameters of discrete AdaBoost[14], where the weak classifier holds $h_t \in \{-1, +1\}$, heuristically to achieve cost-sensitivity without clarifying the relations to the loss minimization of AdaBoost. Although some [15−17] derive the cost-sensitive boosting algorithms from minimizing the asymmetric loss, they only focus on one kind of upper bound of asymmetric loss for discussion and resort to search procedures to get the optimal solution.

It remains unclear whether there is a unified framework to interpret all the asymmetric methods as AdaBoost, clarify their relationships, and further derive the superior real-valued cost-sensitive boosting algorithms which adopt confidence-rated weak learners to reduce the upper bound of training error.

In this paper, we give a detailed discussion about the various discrete asymmetric extensions, divide them into three groups according to the different upper bounds of the asymmetric training error and clarify their relations to the loss minimization of AdaBoost with some reformulations and improvements. Then, the real-valued asymmetric AdaBoost algorithms are derived in the form of additive logistic regression following the way described in [18] with analytical solutions and consideration of different exponential loss functions, which are denoted by Asym-Real AdaBoost and Asym-Gentle AdaBoost.

The rest of the paper is organized as follows. Section 1 presents a discussion about the various discrete asymmetric extensions and their relationship. In Section 2, we derive the real-valued asymmetric AdaBoost in the form of additive logistic regression. Sections 3 and 4 give the experimental results and the conclusion, respectively.

1 Discrete AdaBoost and asymmetric extensions

The discrete AdaBoost proposed by Freund[14] takes as input a training set $S = \langle(x_1, y_1), \cdots, (x_m, y_m)\rangle$ where each instance, x_i, belongs to a certain domain or instance space X, and each label y_i belongs to a finite label space Y. In the object detection application, we will only focus on the binary case, where $Y = \{-1, +1\}$. The pseudo-code for discrete AdaBoost is shown as follows. Essentially, AdaBoost is actually an iterative procedure to reduce the upper bound of training error in (3), where $F(x) = \sum_t \alpha_t h_t(x)$. There are three key issues in the AdaBoost algorithm:

The Pseudo-code for discrete AdaBoost

Given $(x_1, y_1), \cdots, (x_m, y_m)$, $x_i \in X$, and $y_i \in \{-1, +1\}$

Initialize sample weights $D_1(i) = 1/m$.

Iterate $t = 1, \cdots, T$:

1) Train weak leaner $h_t \to Y$ using distribution D_t.

Received May 20, 2008; in revised form March 19, 2009

1. Tsinghua National Laboratory for Information Science and Technology(TNList), Department of Automation, Tsinghua University, Beijing 100084, P. R. China

DOI: 10.3724/SP.J.1004.2009.1403

2) Get the best weak classifier h_t with minimal weighted training error.

3) Choose the weight updating parameter:

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{1}$$

where $\epsilon_t = \sum_{i=1}^{m} D_t(i)[\![h_t(x_i) \neq y_i]\!]$

4) Update and normalize sample weights:

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t} \tag{2}$$

where $Z_t = \sum_i D_t(i)\exp(-\alpha_t y_i h_t(x_i))$

Output the final classifier: $H(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$

1) The sample reweighting scheme. The main effect of the update rule is to decrease or increase the weights of training samples classified correctly or incorrectly by h_t and makes the subsequent weak learners focus on the hard classified samples. Therefore, this affects the training of weak classifiers directly.

2) The choice of combination coefficient α_t. The α_t is chosen to minimize or approximately minimize Z_t in (2) on each round of boosting because the training error satisfies the bound described in (3). Different approaches for calculating α_t analytically or numerically can also affect the performance of AdaBoost.

3) The discriminative weak learners. Different feature sets or different output schemes of the weak learners can produce distinct results.

$$\frac{1}{m}|\{i : H(x_i) \neq y_i\}| \leq \sum_i D_1(i)\exp(-y_i F(x_i)) = \prod_t Z_t \tag{3}$$

Most current asymmetric extensions of AdaBoost attempt to achieve cost-sensitivity by direct modification of the weight updating scheme and the combination coefficient of AdaBoost heuristically. The main idea behind the adjustments is to give more weights on the positive samples, and then the FN will be penalized more than the FP.

Viola[19] first tried to achieve the asymmetric AdaBoost for object detection called AsymBoost by pre-weighting the positive samples K times as large as the negative ones. But, unfortunately the initial asymmetric weights were immediately absorbed by the first selected weak classifier as the AdaBoost process was too greedy. Instead of pre-weighting initially, they altered the weight update rule in each round for applying the asymmetric weight to avoid the weight absorbing phenomenon. The modified weight update rule is

$$\text{AsymBoost:} \quad D_{t+1}(i) = \frac{CD_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}, \tag{4}$$

where $C = (\sqrt{K})^{(1/N)}$ for positive samples and $C = (\sqrt{K})^{(-1/N)}$ for negative samples. K is the cost ratio and N is the total number of the rounds for training the final strong classifier. And AsymBoost relies on (1) to compute α_t.

AdaCost[12] proposed by Fan adopts the similar approach to make AdaBoost cost-sensitive. They incorporated a cost adjustment function $\beta_\delta(i)$ into the weight updating rule (2) and the computation of α_t (1). The weight updating formula was modified into

$$\text{AdaCost:} \quad D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i)\beta_\delta(i))}{Z_t} \tag{5}$$

α_t is computed by

$$\text{AdaCost:} \quad \alpha_t = \frac{1}{2}\ln\frac{1+r_t}{1-r_t} \tag{6}$$

where $r_t = \sum_i D_t(i)y_i h_t(x_i)\beta_\delta(i)$, $\delta = +1$ if $y_i = h_t(x_i)$ and $\delta = -1$ otherwise. Finally, $\beta_{+1}(i) = -0.5C_i + 0.5$ and $\beta_{-1}(i) = 0.5C_i + 0.5$. C_i is the cost factor assigned to the i-th sample and has to be restricted to $[0 \quad 1]$, as $y_i h_t(x_i)\beta_\delta(i) \in [-1 \quad 1]$ was assumed in the derivation of α_t.

CSB0, CSB1, and CSB2 introduced by Ting[20] are similar asymmetric extensions of AdaBoost, which also only modify the weight updating method and again use (1) for calculating α_t without taking the cost items into consideration. The corresponding three different weight update rules are shown as follows.

$$\text{CSB0:} \quad D_{t+1}(i) = \frac{C_\delta(i)D_t(i))}{Z_t} \tag{7}$$

$$\text{CSB1:} \quad D_{t+1}(i) = \frac{C_\delta(i)D_t(i)\exp(-y_i h_t(x_i))}{Z_t} \tag{8}$$

$$\text{CSB2:} \quad D_{t+1}(i) = \frac{C_\delta(i)D_t(i)\exp(-y_i h_t(x_i)\alpha_t)}{Z_t} \tag{9}$$

where $C_{-1}(i)$ is the cost of misclassifying the i-th sample, and $C_{+1}(i) = 1$. As CSB0 and CSB1 are quite different from AdaBoost, they will be excluded in further discussion.

Considering the importance of choosing α_t, Sun[11] not only fed the cost items into the weight updating formula but also re-derived the weight updating parameter α_t to guarantee the boosting efficiency by minimizing the corresponding Z_t according to the approaches in [21].

Three modifications of (2) are shown as (10) ~ (12). Each modification is taken as a new boosting algorithm denoted as AdaC1, AdaC2, and AdaC3, respectively.

$$\text{AdaC1:} \quad D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t C_i h_t(x_i)y_i)}{Z_t} \tag{10}$$

$$\text{AdaC2:} \quad D_{t+1}(i) = \frac{C_i D_t(i)\exp(-\alpha_t h_t(x_i)y_i)}{Z_t} \tag{11}$$

$$\text{AdaC3:} \quad D_{t+1}(i) = \frac{C_i D_t(i)\exp(-\alpha_t C_i h_t(x_i)y_i)}{Z_t} \tag{12}$$

where C_i is an associated cost item of the i-th sample and in the range of $(0 \quad 1]$ for the assumption of $C_i y_i h_t(x_i) \in [-1 \quad 1]$ in the calculation of α_t.

Recently, Hou[15] and Hamed Masnadi-Shirazi[16] attempted to derive asymmetric boosting classifiers from the upper bounds of normalized asymmetric classifier error and the empirical risk of cost-sensitive loss, respectively. Actually, their upper bounds are equivalent. Minimizing the upper bound as is done by original AdaBoost leads to minimizing Z_t (13) at each round as AdaC1.

$$Z_t = \sum_i D_t(i)\exp(-C_i y_i \alpha_t h_t(x_i)) \tag{13}$$

However, Hou embedded the cost items into the label y_i and constrained the cost items to meet the condition $C_1 > C_2$ and $C_1 + C_2 = 1$[15] (cost C_1 to misses and cost C_2 to FP). Fortunately, this made the algorithm more attractive with the ability to select the best cost parameters adaptively. Finally, they achieved the AsymAda (asymmetric AdaBoost algorithm) that consists of the following equations, when $h_t \in \{-1, +1\}$.

$$\begin{aligned} \alpha_t &= \frac{1}{2}\ln\frac{A_t D_t}{B_t C_t} \\ C_1^t &= \frac{\ln\frac{A_t}{B_t}}{2\alpha_t} \end{aligned} \tag{14}$$

where $A_t = P(y_i = 1, h_t(x_i) = 1)$, $B_t = P(y_i = 1, h_t(x_i) = -1)$, $C_t = P(y_i = -1, h_t(x_i) = 1)$, and $D_t = P(y_i = -1, h_t(x_i) = -1)$.

Similar approaches were taken in [16] to derive ABoosting (asymmetric boosting). But, they tried to directly minimize (13) by solving (15). Unfortunately, their optimal solution is not analytical and standard scalar search procedures are required to find the optimal α_t.

$$\begin{aligned} 2C_1 \cdot b \cdot \cosh(C_1\alpha) + 2C_2 \cdot d \cdot \cosh(C_2\alpha) = \\ C_1 \cdot T_+ \cdot \mathrm{e}^{-C_1\alpha} + C_2 \cdot T_- \cdot \mathrm{e}^{-C_2\alpha} \end{aligned} \tag{15}$$

where $T_+ = A_t + B_t$, $T_- = C_t + D_t$, $b = B_t$, and $d = C_t$.

1.1 Unified framework for asymmetric extensions and improvements

Generally, the natural way to obtain cost-sensitive extensions of AdaBoost is to start from the asymmetric loss function or the asymmetric classification error, and then minimize the upper bound of the asymmetric training error as is done by AdaBoost.

For example, an asymmetric loss, with a cost of C_1 for false rejected samples and C_2 for false accepted samples, can be formulated as (16). Accordingly, the asymmetric training error can be represented by FN and FP or asymmetric loss in (17), where $N_{\rm FN}$ and $N_{\rm FP}$ indicate the number of FN and FP, respectively.

$$ALoss(i) = \begin{cases} C_1, & \text{if } y_i = 1 \text{ and } H(x_i) = -1 \\ C_2, & \text{if } y_i = -1 \text{ and } H(x_i) = 1 \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

$$\varepsilon_{\rm asym}(C_1, C_2) = C_1 N_{\rm FN} + C_2 N_{\rm FP} = \sum_i ALoss(i) \tag{17}$$

Considering the different positions between cost items and exponent, there are three kinds of upper bounds of the asymmetric training error as follows:

$$AB_1 = \sum_i \exp(-C_i y_i \sum_t \alpha_t h_t(x_i)) \geq \varepsilon_{\rm asym} \tag{18}$$

$$AB_2 = \sum_i C_i \exp(-y_i \sum_t \alpha_t h_t(x_i)) \geq \varepsilon_{\rm asym} \tag{19}$$

$$AB_3 = \sum_i C_i \exp(-C_i y_i \sum_t \alpha_t h_t(x_i)) \geq \varepsilon_{\rm asym} \tag{20}$$

where $C_i = C_1$ if $y_i = 1$ and $C_i = C_2$ if $y_i = -1$.

According to the different upper bounds, all the above asymmetric extensions of AdaBoost can be divided into three groups as shown in Table 1.

Table 1 The three groups of asymmetric extensions of discrete AdaBoost

Upper bound	Extension	Cost item*	Weight update	α_t
	AdaC1	P← C_1, N← C_2	(10)	approximate
AB_1	AdaCost	TP← β_+^+, FN← β_-^+ TN← β_+^-, FP← β_-^-	(5)	approximate (6)
	AsymAda	P← C_1, N← C_2	(10)	optimal (14)
	ABoosting	P← C_1, N← C_2	(10)	optimal (15)
	AdaC2	P← C_1, N← C_2	(11)	optimal
AB_2	AsymBoost	P← $\sqrt{k}$, N← $\frac{1}{\sqrt{k}}$	(4)	irrational (1)
	CSB2	TP← 1, FN← C_1 TN← 1, FP← C_2	(9)	irrational (1)
AB_3	AdaC3	P← C_1, N← C_2	(12)	approximate

* The cost items refer to the C_i parameters in weight update rules. Only AdaCost and CSB2 treat the correctly and wrongly classified samples differently.

Under the unified framework, the relations between the asymmetric extensions can be clarified, and their main differences locate in the computation of α_t and the selection of cost items. For example, AdaCost can be regarded as a variant of AdaC1 due to a different choice of C_i. And comparing the AdaC1 with ABoosing, the only difference is the way to compute α_t. The optimal α_t in ABoosing outperforms the approximate solution in AdaC1[11].

Another advantage of the unified framework is that it provides a simple way to improve the heuristically modified methods. For example, in AsymBoost, the cost items in each round are related to the number of weak learners (N), which is unavailable in training cascaded detectors. However, since we know its essence, we can re-derive the algorithm from AB_2 to solve this problem.

Minimizing (19) is equivalent to minimizing AB_2' in (21) at each boosting iteration.

$$\begin{aligned} AB_2' = &\sum_{i \in \{i|y_i=1\}} C_1 D_t'(i) \exp(-\alpha_t h_t(x_i)) + \\ &\sum_{i \in \{i|y_i=-1\}} C_2 D_t'(i) \exp(\alpha_t h_t(x_i)) \end{aligned} \tag{21}$$

where $D_t'(i) = \exp(-y_i \sum_{j=1}^{t-1} \alpha_j h_j(x_i))$. Thus, the best weak learner h_t can be obtained by minimizing (21) and the corresponding α_t can be determined by the same method in discrete AdaBoost. The optimal α_t is uniquely selected as

$$\alpha_t = \frac{1}{2}\ln\frac{\sum_{i, y_i = h_t(x_i)} C_i D_t'(i)}{\sum_{i, y_i \neq h_t(x_i)} C_i D_t'(i)} \tag{22}$$

Each sample weight for learning of the next iteration is updated as $D_{t+1}'(i) = D_t'(i)\exp(-\alpha_t y_i h_t(x_i))$.

The similar ideas were also discussed by Sun[11], but not considered in the reasoning process of AdaC2. In fact, the cost items for FN and FP in AdaC2 are $(C_i)^N$, varying during the training process with different number of iterations. Fortunately, the problems of determining N before training in AdaC2 and AsymBoost can be solved by taking advantage of the above asymmetric learning scheme.

Moreover, the computation of α_t in CSB2 and AsymBoost are the same as that of original AdaBoost, which is unreasonable. We can replace them with the optimal one in the same category easily. The inappropriate choice of the

α_t parameter will frequently give disproportionate weights to weak learners and degrade the performance[16].

2 Real-valued asymmetric AdaBoost

Although the discrete asymmetric boosting algorithms achieve better performance than the original discrete ones, the computation of α_t is approximate[11] or requires search procedures[16]. These problems trigger us to find out whether the superior real-valued methods like Real and Gentle AdaBoost can be extended to cost-sensitive ones and if there are simple and analytical ways to compute the optimal parameters.

From a statistical perspective, AdaBoost acts as a method for fitting an additive model $F(x)=\sum_j f_j(x)$ in a forward stage-wise manner. Both discrete and real-valued AdaBoosts can be derived in the form of additive logistic regression by minimizing the expectation of exponential loss function (23), which is usually motivated as an upper bound on misclassification error[18].

$$J(F))=\mathrm{E}\left[\mathrm{e}^{-yF(x)}\right]\geq\varepsilon=N_{\mathrm{FP}}+N_{\mathrm{FN}}\tag{23}$$

Consequently, different loss functions lead to different boosting algorithms. As mentioned in the above section, by integrating the cost item C into (23) in different ways, we can obtain the following expected cost-sensitive exponential loss functions, and then derive various real-valued asymmetric AdaBoost algorithms accordingly:

$$J_{\mathrm{asym}}(F(x),C_i)=\mathrm{E}\left[\mathrm{e}^{-C_i yF(x)}\right]\tag{24a}$$

$$J_{\mathrm{asym}}(F(x),C_i)=\mathrm{E}\left[C_i\mathrm{e}^{-yF(x)}\right]\tag{24b}$$

$$J_{\mathrm{asym}}(F(x),C_i)=\mathrm{E}\left[C_i\mathrm{e}^{-C_i yF(x)}\right]\tag{24c}$$

As parameter C_i is different for positive and negative samples, the above equations are not equivalent.

Taking (24a) for example, if the exponential loss function is

$$ALoss(F(x))=\begin{cases}\mathrm{e}^{-C_1 y_i F(x)}, & \text{if } y_i=1\\ \mathrm{e}^{-C_2 y_i F(x)}, & \text{if } y_i=-1\end{cases}\tag{25}$$

We can notice that it is the upper bound of the simple loss described in (16), thus the upper bound of asymmetric training error is held by expected asymmetric exponential loss defined in (26) ($C_1, C_2\in(0\quad 1]$) like other boosting algorithms. Similarly, asymmetric extensions of AdaBoost can be derived through fitting the additive logistic regression model by stage-wise optimization of (27).

$$\begin{aligned}J_{\mathrm{asym}}(F(x),C_1,C_2)=\mathrm{E}\Big[&I(y=1)\mathrm{e}^{-C_1F(x)}+\\&I(y=-1)\mathrm{e}^{C_2F(x)}\Big]\geq\varepsilon_{\mathrm{asym}}(C_1,C_2)\end{aligned}\tag{26}$$

Suppose a current hypothesis $F(x)$ has been obtained in the additive model. The next step is to learn an optimal weak classifier $f(x)$ to add in. Thus, the overall training loss turns into

$$\begin{aligned}J_{\mathrm{asym}}(F(x)+f(x))=\mathrm{E}_w\Big[&I(y=1)\mathrm{e}^{-C_1f(x)}+\\&I(y=-1)\mathrm{e}^{C_2f(x)}\Big]=\\&P_w(y=1|x)\mathrm{e}^{-C_1f(x)}+P_w(y=-1|x)\mathrm{e}^{C_2f(x)}\end{aligned}\tag{27}$$

where $w=w(x,y)=\mathrm{e}^{-yCF(x)}$, $C=C_1$ if $y=1$, and $C=C_2$ if $y=-1$. $\mathrm{E}_w[\cdot]$ is the weighted expectation defined by

$$\mathrm{E}_w\left[g(x,y)\right]=\frac{\mathrm{E}\left[w(x,y)g(x,y)\right]}{\mathrm{E}\left[w(x,y)\right]}\tag{28}$$

Thus, the optimal weak classifier can be obtained from minimizing (27) and described as follows:

$$\begin{aligned}\hat{f}(x)=\arg\min_f J_{\mathrm{asym}}(F(x)+f(x),C_1,C_2)=\\\frac{1}{C_1+C_2}\ln\frac{C_1P_w(y=1|x)}{C_2P_w(y=-1|x)}\end{aligned}\tag{29}$$

We denote the above algorithm as Asym-Real AdaBoot-1. Similarly, the optimal weak hypothesis of Asym-Gentle AdaBoot-1 can be obtained using Newton steps as is done by Gentle AdaBoost and described as follows:

$$f_t(x)=\frac{C_1P_w(y=1|x)-C_2P_w(y=-1|x)}{C_1^2P_w(y=1|x)+C_2^2P_w(y=-1|x)}\tag{30}$$

Both the above two algorithms use $w_i^{(t+1)}=w_i^{(t)}\mathrm{e}^{-Cyf_t(x)}$ for updating weights.

If we start from (24b), the similar steps can be taken to explore extensions of AdaBoost. The optimal weak classifier selected in each iteration should satisfy

$$\begin{aligned}\hat{f}(x)=\arg\min_f C_1P(y=1|x)\mathrm{e}^{-(F(x)+f(x))}+\\C_2P(y=-1|x)\mathrm{e}^{(F(x)+f(x))}\end{aligned}\tag{31}$$

Here, if we let $w=w(x,y)=\mathrm{e}^{-yF(x)}$, then the best weak classifiers for Asym-Real AdaBoost-2 and Asym-Gentle AdaBoost-2 can be obtained respectively as follows, with, the corresponding weight updating rule $w_i^{(t+1)}=w_i^{(t)}\mathrm{e}^{-yf_t(x)}$.

$$f_t(x)=\frac{1}{2}\ln\frac{C_1P_w(y=1|x)}{C_2P_w(y=-1|x)}\tag{32a}$$

$$f_t(x)=\frac{C_1P_w(y=1|x)-C_2P_w(y=-1|x)}{C_1P_w(y=1|x)+C_2P_w(y=-1|x)}\tag{32b}$$

If we let $w'=w'(x,y)=C\mathrm{e}^{-yF(x)}$ and consider w' as the new distribution of the training samples, then the rest steps of the asymmetric algorithms will be the same as the symmetric AdaBoost algorithms, as discussed in Section 1. Thus, the weak classifiers in each round can be determined by

$$f_t(x)=\frac{1}{2}\ln\frac{P_{w'}(y=1|x)}{P_{w'}(y=-1|x)}\tag{33a}$$

$$f_t(x)=P_{w'}(y=1|x)-P_{w'}(y=-1|x)\tag{33b}$$

The corresponding weight updating scheme is ${w'}_i^{(t+1)}={w'}_i^{(t)}\mathrm{e}^{-yf_t(x)}$. However, this approach is equivalent to the direct pre-weighting method proposed by Viola and Jones, so it is unworkable because of the weight absorbing phenomenon.

The resulting asymmetric extensions derived from (24c) contain two cases, too. In one case, $w_i^{(t+1)}=w_i^{(t)}\mathrm{e}^{-Cyf_t(x)}$, the optimal weak classifiers are

$$f_t(x)=\frac{1}{C_1+C_2}\ln\frac{C_1^2P_w(y=1|x)}{C_2^2P_w(y=-1|x)}\tag{34a}$$

$$f_t(x) = \frac{C_1^2 P_w(y=1|x) - C_2^2 P_w(y=-1|x)}{C_1^3 P_w(y=1|x) + C_2^3 P_w(y=-1|x)} \tag{34b}$$

In the other case, $w_i^{'(t+1)} = C\mathrm{e}^{-Cy(F(x)+f_t(x))} = w_i^{'(t)}\mathrm{e}^{-Cyf_t(x)}$ is chosen as the weight updating rule, and the corresponding weak classifiers can be obtained similarly as (35a) and (35b). Obviously, these two extensions are like the Asym-Real Adaboost-1 and Asym-Gentle AdaBoost-1, respectively, except the initial weights of training samples, hence no more discussions will be given about these extensions in the following sections.

$$f_t(x) = \frac{1}{C_1 + C_2} \ln \frac{C_1 P_{w'}(y=1|x)}{C_2 P_{w'}(y=-1|x)} \tag{35a}$$

$$f_t(x) = \frac{C_1 P_{w'}(y=1|x) - C_2 P_{w'}(y=-1|x)}{C_1^2 P_{w'}(y=1|x) + C_2^2 P_{w'}(y=-1|x)} \tag{35b}$$

The extensions referring to (34a) and (34b) will be denoted by Asym-Real AdaBoost-3 and Asym-Gentle AdaBoost-3 in further discussions.

For the cost-sensitive boosting algorithms, the costs C_1 and C_2 are used to characterize the identification importance of different samples, naturally specified from domain knowledge.

In practice, there are two approaches to determine the cost items. One is to specify the costs through empirical methods which let cost factors chosen from certain intervals and then test the performance of the various costs, finally select the best cost items according to the desired performance. The other approach is adjusting the cost items to satisfy the given criterion during the training process, such as minimizing the upper bound of training error mentioned in the previous section and Neyman-Pearson criterion that minimizes the overall risk subject to a given detection rate or FP rate[22]. Note that because we can always set $C_1 = 1$, the search for optimal cost is one-dimensional. In this paper, we let $C_2 \in [0.1 \quad 1]$ while fixing $C_1 = 1$ for comparing the performance of various real-valued asymmetric extensions.

The weak learners widely used for real-valued AdaBoost are domain-partitioning weak hypotheses, such as classification and regression trees (CART)[23] and look-up table (LUT) type weak classifier[4]. We will use these two kinds of weak classifiers in evaluation for face detection and pedestrian detection, respectively.

The training process of Asym-Gentle AdaBoost-1 with domain-partitioning weak learners can be described as following and the other real-valued extensions can be trained similarly.

The training process of Asym-Gentle AdaBoost-1 with domain-partitioning weak learners

Given $(x_1, y_1), \cdots, (x_m, y_m)$, $x_i \in X$, and $y_i \in \{-1, +1\}$

Initialize sample weights $D_1(i) = 1/m$.

Choose cost items C_1, C_2. Let $C_1 > C_2$, $C_1, C_2 \in (0, 1]$,

Iterate $t = 1, \cdots, T$:

1) For each weak learner $f_j(x)$, it will partition X into several disjoint subregions $X_1, \cdots, X_n$. Compute the output of $f_j(x)$ in each subregion using (30) and the normalization factor Z_t^j.

$$Z_t^j = \sum_i D_t(i) \exp(-C_i y_i f_j(x_i)) \tag{36}$$

2) Choose the weak classifier $f_j(x)$ with smallest Z_t^j as the best function at this round $f_t(x)$.

3) Update and normalize sample weights:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-C_i y_i f_t(x_i))}{Z_t} \tag{37}$$

Output the final classifier: $H(x) = \mathrm{sign}(\sum_{t=1}^{T} f_t(x))$.

3 Experimental results

We evaluated the real-valued asymmetric algorithms on both pedestrian detection and face detection tasks.

Regarding the comparisons of cost-sensitive classification performance, we used the pedestrian benchmark data set proposed by Munder and Gavrila[24] for evaluation, which consists of three training sets and two test sets. Every set contains 4 800 pedestrian examples and 5 000 non-pedestrian examples. For each real-valued or discrete asymmetric AdaBoost algorithm, the weak learners were based on a combination of decision stumps and histograms of oriented gradients (HOG) features[25], and the strong classifier was trained with 100 iterations, by selecting one out of the three training sets, and using the remaining for validation. After the parameters (cost items C_i) were optimized via cross validation, testing the classifier on the two test sets yielded two different results, and the mean value was used for final comparison.

The evaluation of cost-sensitive classification requires a metric that weights some errors more than others. We adopted the common metric described in (38) for comparison, where p is the FP, m is the number of missing positives (FN), and $\beta > 1$ is a cost factor that weights misses more heavily than FP[16].

$$\epsilon = p + \beta \times m \tag{38}$$

Three cost factors ($\beta = 2, 5, 10$) were considered and Table 2 presents the corresponding best results of each algorithm with different parameters C. Note that symmetric AdaBoost methods performed well for small cost factors. And real-valued asymmetric algorithms outperformed the discrete ones a lot.

Table 2 The best results of different methods with various cost ratios C under different criteria with $\beta = 2, 5, 10$

Method	$\beta = 2$	$\beta = 5$	$\beta = 10$
Discrete AdaBoost	4 452 $p = 1\,232, m = 1\,610$	9 282	17 732
AdaC1	4 438 $C_2 = 0.5$	7 965 $C_2 = 0.3$	11 573 $C_2 = 0.2$
CSB2	4 526 $C_{-1}^P = 2$	8 378 $C_{-1}^P = 5$	12 438 $C_{-1}^P = 8$
Gentle AdaBoost	2 790 $p = 542, m = 1\,124$	6 162	11 782
Asym-Real-1	2 786 $C_2 = 0.5$	5 241 $C_2 = 0.3$	8 054 $C_2 = 0.2$
Asym-Real-3	2 924 $C_2 = 0.7$	5 382 $C_2 = 0.4$	8 985 $C_2 = 0.2$
Asym-Gentle-1	2 682 $C_2 = 0.5$	4 996 $C_2 = 0.2$	7 346 $C_2 = 0.2$
Asym-Gentle-2	2 835 $C_2 = 0.6$	5 182 $C_2 = 0.3$	7 947 $C_2 = 0.2$

For face detection, a set of 2 170 frontal face samples and 3 000 non-face samples were collected from Internet and normalized to 24×24 for training single stage classifiers using the real-valued symmetric and asymmetric AdaBoost.

All the single stage classifiers were trained until they got the zero training error using CART as the weak learner. And the MIT+CMU test which consists of 130 images and contains 507 frontal faces was used for evaluating the performance. The ROC curves of these algorithms are shown in Figs. 1 and 2 according to the different node numbers of CART. We can see that the Asym-Gentle AdaBoost was more robust than Asym-Real AdaBoost, and the Asym-Gentle AdaBoost-1 always achieved the best performance.

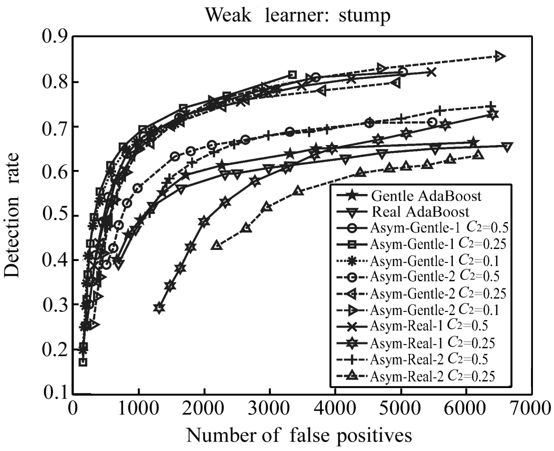

Fig. 1 Detection performances of single stage classifiers with stump on the MIT+CMU database

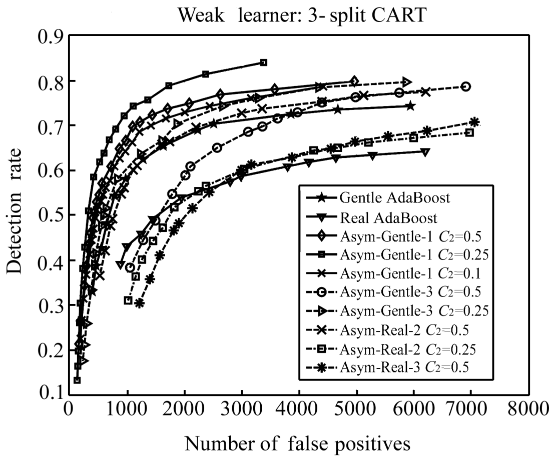

Fig. 2 Detection performances of single stage classifiers with 3-split CART on the MIT+CMU database

While training the cascaded face detector which follows the flowchart in [10] using Asym-Real and Asym Gentle AdaBoost algorithms, 6 930 normalized frontal face samples and 8 400 images containing no faces were collected from various sources. Finally, the best detector we obtained was trained by Asym-Gentle AdaBoost-1 with the CART of two splits and the cost factor $C_2 = 0.25$. It contains 18 layers and total 1 876 features, and achieved a detection rate of 98.2 % and false accept rate of 2.1×10^{-6} on the training set. Its performance on MIT+CMU test set is listed in Table 3 compared with the results of other asymmetric algorithms.

To compare the performance of various algorithms on pedestrian detection task, we evaluated all the algorithms on the well organized INRIA database[25]. In this database, pedestrians are mostly upright standing or walking with an image size of 64×128. The training set consists of 1 208 pedestrian images (2 416 with their left-right reflections) and 1 218 person-free images. Meanwhile, there are 288 images containing 563 pedestrians and 453 images without pedestrians in the testing set.

The framework used in our asymmetric detectors is the same as the one described in [7]. However, we further normalized the output of the weighted Fischer linear discriminant (WFLD) learner to make it suitable for constructing the LUT type weak classifier. On the basis of the LUT type weak learner, we could directly choose the best weak classifier to minimize the upper bound of the training error. The performances of Real AdaBoost, Gentle AdaBoost, and the best asymmetric extension on the testing set are shown in Fig. 3, where the best result was still obtained by Asym-Genlte AdaBoost-1 with the cost factor $C_2 = 0.25$.

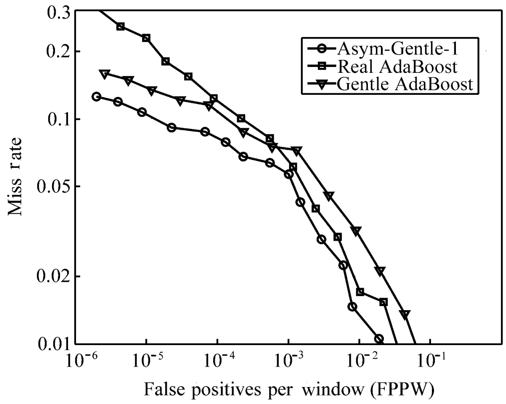

Fig. 3 Detection performances on INRIA database

4 Conclusion

For object detection, the asymmetric costs for FP and FN are required for higher performance. Recent asymmetric extensions of AdaBoost were realized based on the discrete AdaBoost to solve this problem. But most of them relied on tricks or heuristic alterations of the AdaBoost algorithm to obtain cost-sensitivity.

In this paper, we presented a detailed discussion about the various discrete cost-sensitive extensions and their relationship. Moreover, we pointed out that the asymmetric extensions can be derived naturally by minimizing the upper bound of the asymmetric training error in a step-wise manner. These ideas were adopted during deriving the real-valued asymmetric extensions of AdaBoost. Three different asymmetric exponential loss functions which were motivated as the upper bound of the asymmetric training error have led to three kinds of cost-sensitive algorithms.

The comparative experiments have demonstrated that the proposed real-valued extensions outperform the discrete ones on classification performance and computation of parameters. Moreover, the Asym-Gentle AdaBoost methods are more robust than Asym-Real AdaBoost and achieve better performance than the previous symmetric and asymmetric AdaBoost algorithms on both face detection and pedestrian detection.

For future work, a notable task is to fix the cost factors through some more efficient methods like the adaptive approach proposed by Hou[15], since both the current empirical and searching methods are time-consuming.

Table 3 Frontal face detection rates for various numbers of FP on the MIT+CMU test set

FP	10	21	31	49	50	65	78	97	110
Asym-Gentle-1	91.2 %	92.4 %	-	93.7 %	-	95.2 %	-	96.4 %	-
Hou	90.5 %	-	91.9 %	-	93.1 %	93.9 %	94.1 %	-	94.7 %
Ma-Ding	90.1 %	-	91.3 %	-	92.5 %	93.1 %	93.3 %	-	-
Viola-Jones (Asym)	-	-	88.5 %	-	91.5 %	91.9 %	92.1 %	-	93.1 %

References

1 Schapire R E. A brief introduction to boosting. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence. Stockholm, Sweden: Morgan Kaufmann Publishers, 1999. 1401−1406

2 Viola P, Jones M J. Robust real-time face detection. *International Journal of Computer Vision*, 2004, **57**(2): 137−154

3 Huang C, Ai H Z, Li Y, Lao S H. High-performance rotation invariant multiview face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007, **29**(4): 671−686

4 Wu B, Ai H Z, Huang C, Lao S H. Fast rotation invariant multi-view face detection based on real adaboost. In: Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition. Seoul, Korea: IEEE, 2004. 79−84

5 Ma Y, Ding X Q. Real-time rotation invariant face detection based on cost-sensitive adaboost. In: Proceedings of IEEE International Conference on Image Processing. Barcelona, Spain: IEEE, 2003. 921−924

6 Viola P, Jones M J, Snow D. Detecting pedestrians using patterns of motion and appearance. In: Proceedings of IEEE International Conference on Computer Vision. Nice, France: IEEE, 2003. 734−741

7 Laptev I. Improvements of object detection using boosted histograms. In: Proceedings of British Machine Vision Conference. Edinburgh, UK: Springer, 2006. 949−958

8 Zhu Q, Yeh M C, Cheng K T, Avidan S. Fast human detection using a cascade of histograms of oriented gradients. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. New York, USA: IEEE, 2006. 1491−1498

9 Hou C, Ai H Z, Lao S H. Multiview pedestrian detection based on vector boosting. In: Proceedings of Asian Conference on Computer Vision. Tokyo, Japan: Springer, 2007. 210−219

10 Viola P, Jones M J. Rapid object detection using a boosted cascade of simple features. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Hawaii, USA: IEEE, 2001. 511−518

11 Sun Y M, Kamel M S, Wong A K C, Wang Y. Costsensitive boosting for classification of imbalanced data. *Pattern Recognition*, 2007, **40**(12): 3358−3378

12 Fan W, Stolfo S J, Zhang J X, Chan P K. Adacost: misclassification cost-sensitive boosting. In: Proceedings of International Conference on Machine Learning. San Francisco, USA: Morgan Kaufmann, 1999. 97−105

13 Pham M T, Cham T J. Online learning asymmetric boosted classifiers for object detection. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Minnesota, USA: IEEE, 2007. 1−8

14 Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting. In: Proceedings of European Conference on Computational Learning Theory. Barcelona, Spain: Springer, 1995. 23−37

15 Hou X W, Liu C L, Tan T N. Learning boosted asymmetric classifiers for object detection. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. New York, USA: IEEE, 2006. 330−338

16 Masnadi-Shirazi H, Vasconcelos N. Asymmetric boosting. In: Proceedings of International Conference on Machine Learning. Oregon, USA: ACM Press, 2007. 609−619

17 Masnadi-Shirazi H, Vasconcelos N. Cost-Sensitive Boosting, Technical Report SVCL-TR-2007-06, Department of Electrical and Computer Engineering, University of California, USA, 2007

18 Friedman J, Hastie T, Tibshirani R. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 2000, **28**(2): 337−407

19 Viola P, Jones M J. Fast and robust classification using asymmetric adaboost and a detector cascade. In: Proceedings of Advances in Neural Information Processing Systems. British Columbia, Canada: MIT Press, 2001. 1311−1318

20 Ting K M. A comparative study of cost-sensitive boosting algorithms. In: Proceedings of International Conference on Machine Learning. California, USA: Morgan Kaufmann, 2000. 983−990

21 Schapire R E, Singer Y. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 1999, **37**(3): 297−336

22 Duda R O, Hart P E, Stork D G. *Pattern Classification*. Beijing: China Machine Press, 2000. 26−29

23 Jia H X, Zhang Y J. Fast human detection by boosting histograms of oriented gradients. In: Proceedings of International Conference on Image and Graphics. Chengdu, China: IEEE, 2007. 683−688

24 Munder S, Gavrila D M. An experimental study on pedestrian classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006, **28**(11): 1863−1868

25 Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. California, USA: IEEE, 2005. 886−893

GE Jun-Feng Received his B. Sc. degree from Huazhong University of Science and Technology in 2003. He is currently a Ph. D. candidate with the Department of Automation, Tsinghua University. His research interest covers computer vision and machine learning. Corresponding author of this paper.
E-mail: gejf03@mails.thu.edu.cn

LUO Yu-Pin Received his B. Sc. degree from Changsha University in 1982, and his M. Sc. and Ph. D. degrees from Nagoya Institute of Technology, Japan in 1987 and 1990, respectively. He is currently a professor with the Department of Automation, Tsinghua University. His research interest covers image processing, computer vision, and pattern recognition.
E-mail: luo@tsinghua.edu.cn