TaylorBoost: First and Second-order Boosting Algorithms with Explicit Margin Control

Mohammad J. Saberian Hamed Masnadi-Shirazi Nuno Vasconcelos
Department of Electrical and Computer Engineering
University of California, San Diego
saberian@ucsd.edu, hmasnadi@ucsd.edu, nuno@ece.ucsd.edu

Abstract

A new family of boosting algorithms, denoted TaylorBoost, is proposed. It supports any combination of loss function and first or second order optimization, and includes classical algorithms such as AdaBoost, GradientBoost, or LogitBoost as special cases. Its restriction to the set of canonical losses makes it possible to have boosting algorithms with explicit margin control. A new large family of losses with this property, based on the set of cumulative distributions of zero mean random variables, is then proposed. A novel loss function in this family, the Laplace loss, is finally derived. The combination of this loss and second order TaylorBoost produces a boosting algorithm with explicit margin control.

1. Introduction

Modern solutions to many vision problems involve the design of a classifier. Boosting is a reliable tool for this design. Since the introduction of AdaBoost in [5], a number of algorithms have appeared in the literature, including LogitBoost [6], GentleBoost [6], GradientBoost [12], or TangentBoost [11]. They all minimize a risk that upper bounds the classification error, and converge asymptotically to the Bayes decision rule. However, when trained with a limited number of (possibly noisy) examples, their results vary significantly. In fact, experience has shown that different boosting algorithms can achieve significantly better performance in different classification problems.

Boosting algorithms differ along three main dimensions: the weak learners that are boosted, the optimization strategy used for weak learner selection, and the loss function that guides this optimization. Weak learners can be highly problem dependent, and are not considered in this work. Effective loss functions combine two main properties. The first is Bayes consistency, in the sense that the minimization of the associated risk converges asymptotically to the Bayes decision rule [6, 15, 3, 9]. This guarantees an optimal classifier in the large training sample regime. The second is the ability to enforce a margin, by penalizing examples correctly classified but close to the boundary. This results in improved generalization when using finite samples [14].

Despite the importance of these properties, the set of Bayes-consistent large-margin losses has remained small. In fact, its study has only recently been addressed in a systematic form in [9]. This work introduced a generic framework for the derivation of Bayes consistent losses. We have recently shown that this framework can also be used to derive losses with explicit margin control [10]. This is a family of loss functions parameterized by a scalar that controls the extent of the penalty for correctly classified examples.

Given a loss, a second important boosting dimension is the optimization strategy used for risk minimization. For example, AdaBoost has the well known interpretation of a gradient descent procedure for minimization of the risk associated with the exponential loss. This optimization strategy has been generalized to other losses, through the introduction of GradientBoost in [12]. Second order extensions, based on the Newton method, were also developed for the logistic and exponential risks, leading to LogitBoost and GentleBoost [6]. While more powerful than gradient descent, these second order extensions turned out not to be easy to reconcile with the weighting mechanism that is critical to the success of boosting. Since the straightforward application of Newton method does not produce example weighting, a somewhat arbitrary weighting mechanism was added to these algorithms. Despite its limited mathematical support, experiments show that this mechanism is crucial for classifier effectiveness. For example, Figure 1 compares the evolution of the LogitBoost risk, as a function of boosting iteration, with and without weights. It is clear that performance degrades significantly when weights are omitted. This can be problematic, since it is unclear how the weighting mechanism could be generalized to other losses. This problem has prevented the introduction of a generic second-order method, that generalizes LogitBoost in a manner similar to how GradientBoost generalizes AdaBoost.

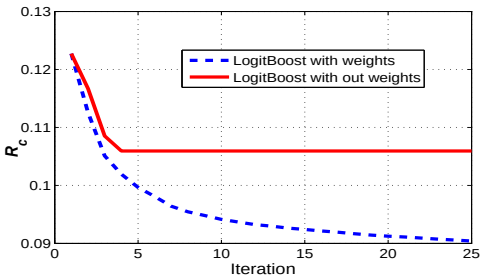

Figure 1. Classification risk of LogitBoost with and without weights.

In this work, we propose joint contributions along the dimensions of second order boosting, and margin controllable loss functions. The first contribution is a generic optimization procedure, based on a Taylor series expansion of the risk, that can be applied to any loss function. It leads to a family of boosting algorithms of either first or second order (depending on the approximation), denoted *TaylorBoost*. This family is shown to include GradientBoost (first-order methods) and LogitBoost (second order) when applied to the logistic loss. The final contribution is a new family of loss functions with *margin control*, derived from the set of cumulative probability distributions of zero mean random variables. A novel Bayes consistent loss in this family, *Laplace* , is introduced. The combination of this loss and second order TaylorBoost produces an algorithm with state-of-the-art results in various vision problems.

2. Boosting

A classifier is a map $h(x)$ from examples $x \in \mathcal{X}$ into labels $y \in \{-1, 1\}$. This is usually implemented as $h(x) = sgn[f(x)]$ for some real-valued classifier predictor f. The optimal predictor $f^*(x)$ minimizes the *classification risk*

$$R_c(f) = E_{X,Y}\{L[y, f(x)]\} \tag{1}$$

associated with a loss function $L(.,.)$. Boosting is a procedure to find f^*, from a training set of examples (x_i, y_i), by solving the optimization

$$\left\{ \begin{array}{ll} \min_{f(x)} & R_c(f) \simeq \sum_i L[y_i, f(x_i)] \\ s.t & f(x) \in \Omega_{\mathcal{G}}, \end{array} \right. \tag{2}$$

where $\mathcal{G} = \{g_1, .., g_m\}$ is a set of weak learners, and $\Omega_{\mathcal{G}} = span(\mathcal{G})$. The optimal classifier is found by sequentially adding to the current solution, f^k, an update $h(x) = c_g g(x)$, with $c_g \in \mathbb{R}$, $g \in \mathcal{G}$. The values of c_g and g are chosen to minimize the risk of the updated predictor, $f^{k+1}(x) = f^k(x) + h(x)$. Since $\Omega_{\mathcal{G}}$ is a convex set, the optimization problem is convex whenever $R_c(f)$ is convex in f. In this case, the boosting iterations converge to a globally optimal solution. Particular boosting algorithms differ in the loss functions adopted, the weak learner set $\mathcal{G}$, and the method used to compute the best update at each iteration. Many of these algorithms can be interpreted as gradient descent procedures in the functional space $\Omega_{\mathcal{G}}$. This requires familiarity with some analytical tools.

2.1. Analytical tools

Solving (2) by iterative descent requires the derivative of the functional $R_c[f(x)]$ along the direction $g(x)$, [7]

$$\delta R_c(f; g) = \left. \frac{\partial R_c(f + \xi g)}{\partial \xi} \right|_{\xi=0}. \tag{3}$$

This is a measure of the variation of R_c at point f along the direction specified by g. For simplicity, we sometimes omit the argument x throughout this paper. The curvature of $R_c[f]$ along g is given by the second order derivative

$$\delta^2 R_c(f; g) = \left. \frac{\partial^2 R_c(f + \xi g)}{\partial \xi^2} \right|_{\xi=0}. \tag{4}$$

Using these quantities, R_c can be approximated by a Taylor series expansion around f [7],

$$R_c(f+\epsilon g) = R_c(f) + \epsilon \delta R_c(f; g) + \frac{\epsilon^2}{2} \delta^2 R_c(f; g) + O(\epsilon^3).$$

Since $R_c[f]$ only depends on the example space $\mathcal{X}$ through the training points x_i, there is no loss in mapping the space of functions $f(x)$ into the finite vector space $\mathcal{V} \in \mathbb{R}^n$ of vectors $[f(x_1), \ldots, f(x_n)]$, where n is the training set size. In $\mathcal{V}$, the dot product is usually defined as

$$\langle g_1, g_2 \rangle = \sum_i g_1(x_i) g_2(x_i). \tag{5}$$

Without loss of generality, we assume that the weak learners, $g \in \mathcal{G}$, are normalized so that $\langle g, g \rangle = 1$. Defining the indicator function $I(x) = 1$ if x holds and zero otherwise, the functional *gradient* of R_c is the vector of components

$$\begin{aligned} \nabla_{R_c(f)}(x_i) &= \left. \frac{\partial}{\partial \xi} R_c[f + \xi I(x = x_i)] \right|_{\xi=0} \\ &= \left. \frac{\partial L[y_i, f(x_i) + \xi]}{\partial \xi} \right|_{\xi=0} \end{aligned} \tag{6}$$

and the *second order gradient*, or Hessian, is the matrix

$$\begin{aligned} \nabla^2_{R_c(f)}(x_i, x_j) &= \left. \frac{\partial^2 R_c(f + \xi_1 I(x=x_i) + \xi_2 I(x=x_j))}{\partial \xi_1 \partial \xi_2} \right|_{\xi_1, \xi_2 = 0} \\ &= \left\{ \begin{array}{ll} \left. \frac{\partial^2}{\partial \xi^2} L[y_i, f(x_i) + \xi] \right|_{\xi=0} & \text{if } i = j \\ 0 & \text{otherwise} \end{array} \right. \end{aligned} \tag{7}$$

For simplicity, we denote $\nabla^2_{R_c(f)}(x_i, x_i)$ as $\nabla^2_{R_c(f)}(x_i)$. R_c is convex whenever

$$\nabla^2_{R_c(f)}(x_i) \geq 0, \quad \forall f, \ \forall x_i. \tag{8}$$

Projection into $\mathcal{V}$ simplifies (3) into

$$\delta R_c(f;g) = \frac{\partial}{\partial \xi} \sum_j L[y_j, f(x_j) + \xi g(x_j)] \Big|_{\xi=0} \quad (9)$$

$$= \sum_j g(x_j) \frac{\partial}{\partial \zeta} L[y_i, f(x_j) + \zeta)] \Big|_{\zeta=0} = \left\langle \nabla_{R_c(f)}, g \right\rangle . \quad (10)$$

Similarly, it can be shown that

$$\delta^2 R_c(f;g) = \left\langle \nabla^2_{R_c(f)}, g^2 \right\rangle = \langle g, g \rangle_{\nabla^2_{R_c(f)}}, \quad (11)$$

where $< f, g >_w = \sum_i w_i f(x_i) g(x_i)$. Gradient descent iteratively updates f by selecting, at each iteration, a step in the direction of the negative gradient. The Newton method is a second order method that also considers the curvature of R_c, using the update [6]

$$\mathcal{J}_{R_c(f)}(x_i) = -\frac{\nabla_{R_c(f)}(x_i)}{\nabla^2_{R_c(f)}(x_i)}. \quad (12)$$

2.2. AdaBoost

AdaBoost [5] is a gradient descent method for minimization of the exponential risk, $L[y, f(x)] = e^{-yf(x)}$. Since

$$\nabla^2_{R_c(f)}(x_i) = y_i^2 e^{-y_i f(x_i)} > 0,$$

R_c is strictly convex and (2) has a unique solution. Given the predictor at iteration k, $f^k(x)$, the descent direction is

$$-\nabla_{R_c(f^k)}(x_i) = y_i e^{-y_i f^k(x_i)} = y_i w_i, \quad (13)$$

where $w_i = e^{-y_i f^k(x_i)}$ is the weight of the i^{th} example. This direction may not belong to $\mathcal{G}$, and is approximated by

$$g^* = \arg\max_{g \in \mathcal{G}} \langle -\nabla_{R_c}, g \rangle = \arg\max_{g \in \mathcal{G}} \sum_i y_i g(x_i) w_i \quad (14)$$

The optimal step size, in the direction of g^*, is then

$$c_{g^*} = \arg\min_c R_c(f^k(x) + c g^*(x)), \quad (15)$$

The predictor is updated to $f^{k+1} = f^k + c_{g^*} g^*$ and the procedure iterated. Since (14) and (15) are independent of both $\mathcal{G}$ and the loss used, they are valid, and called as GradBoost [12], for any proper loss and set of weak learners.

2.3. LogitBoost

LogitBoost [6] is an implementation of Newton's method for the minimization of the logistic risk loss, $L[y, f(x)] = \log(1 + e^{-2yf(x)})$, and real-valued learners. Application of (6), (7) and (12) leads to the update

$$\nabla_{R_c(f^k)}(x_i) = \frac{2 y_i e^{-2 y_i f^k(x_i)}}{1 + e^{-2 y_i f^k(x_i)}}, \quad (16)$$

$$\nabla^2_{R_c(f^k)}(x_i) = \frac{4 y_i^2 e^{-2 y_i f^k(x_i)}}{\left(1 + e^{-2 y_i f^k(x_i)}\right)^2} > 0, \quad (17)$$

$$\mathcal{J}_{R_c(f^k)}(x_i) = \frac{1}{2} y_i (1 + e^{-2 y_i f^k(x_i)}), \quad (18)$$

at iteration k. As in AdaBoost, R_c is strictly convex, and $\mathcal{J}_{R_c(f^k)}$ is approximated by the closest weak learner

$$g^* = \arg\max_{g \in \mathcal{G}} \left\langle \mathcal{J}_{R_c(f^k)}, g \right\rangle . \quad (19)$$

However, as shown in Figure 1 , this results in boosted classifiers with weak performance. Noting that, when $g \in \mathcal{G}$ are normalized, (19) is the solution of

$$(g^*, c_g^*) = \arg\min_{g \in \mathcal{G}} \left| \mathcal{J}_{R_c(f^k)} - c_g g \right|^2, \quad (20)$$

[6] suggested to overcome this limitation by solving, instead, the weighted least squares problem

$$(g^*, c_g^*) = \arg\min_{g \in \mathcal{G}} \left| \mathcal{J}_{R_c(f^k)} - c_g g \right|_w^2 \quad (21)$$

$$= \arg\min_{g \in \mathcal{G}} \sum_i w_i [\mathcal{J}_{R_c(f^k)}(x_i) - c_g g(x_i)]^2 \quad (22)$$

with a set of weights defined as

$$w_i = 4 \left(e^{f^k(x_i)} + e^{-f^k(x_i)} \right)^{-2}. \quad (23)$$

(21) results in

$$g^* = \arg\max_{g \in \mathcal{G}} \frac{\left\langle \mathcal{J}_{R_c(f^k)}, g \right\rangle_w^2}{\langle g, g \rangle_w}, \quad (24)$$

and perform a line search, using (15), to find the optimal step size. While this improves the performance, it has various problems. First, unlike AdaBoost, the weights do not follow naturally from the optimization. Second, they are somewhat arbitrary, defined so as to guarantee that the final algorithm has boosting-like properties. Third, it is not clear that the addition of such weights would benefit other boosting algorithms, or how they should be defined.

3. TaylorBoost

The distinguishing feature of TaylorBoost is that, rather than computing the best update (in the gradient or Newton sense) and selecting the closest weak learner, as in (14) or (21), it selects the best weak learner directly. This is done

by defining the latter as the minimizer of the Taylor series expansion of the risk around the current predictor, f^k,

$$(c^*, g^*) = \arg\min_{c,g} \overline{R_c}(f^k + cg) \quad (25)$$

$$\overline{R_c}(f^k + cg) = R_c(f^k) + c\delta R_c(f^k; g) + \frac{c^2}{2}\delta^2 R_c(f^k; g).$$

By controlling the order of the approximation, it is possible to obtain different boosting algorithms.

For an approximation of first order,

$$(c_{g^*}, g^*) = \arg\min_{c,g} R_c(f^k) + c_g \delta R_c(f^k; g) \quad (26)$$

the optimal weak learner is

$$g^* = \arg\min_g \delta R_c(f^k; g) = \arg\min_g \langle \nabla_{R_c}, g \rangle \quad (27)$$

This is identical to the selection rule of (14). The optimal step is then defined as in (15), and the proposed procedure reduces to GradBoost. For a second order approximation

$$(c^*, g^*) = \arg\min_{c,g} R_c(f^k) + c\delta R_c(f^k; g) + \frac{c^2}{2}\delta^2 R_c(f^k; g).$$

Given a weak learner, the optimal weighting of the two derivatives is obtained by setting to zero the derivative with respect to c. This leads to

$$c_g^* = -\frac{\delta R_c(f^k; g)}{\delta^2 R_c(f^k; g)}. \quad (28)$$

The optimal weak learner is then using (11)

$$g^* = \arg\max_{g\in\mathcal{G}} \frac{[\delta R_c(f^k; g)]^2}{\delta^2 R_c(f^k; g)} = \arg\max_{g\in\mathcal{G}} \frac{\langle \nabla_{R_c(f)}, g \rangle^2}{\langle g, g \rangle_w} \quad (29)$$

where $w_i = \nabla^2_{R_c(f^k)}(x_i)$. The optimal step size can finally be found by a line search according to (15). We denote this method *Quadratic boosting*, or QuadBoost. Note that if logistic loss is used, these weights are the same as (23) and it follows from (12) that $\mathcal{J}_{R_c(f^k)}(x_i) \times w_i = -\nabla_{R_c f^k}(x_i)$. The optimal weak learner of (29) can then be written as

$$g^* = \arg\max_{g\in\mathcal{G}} \frac{\langle \nabla_{R_c(f^k)}, g \rangle^2}{\langle g, g \rangle_w} = \arg\max_{g\in\mathcal{G}} \frac{\langle \mathcal{J}_{R_c(f^k)}, g \rangle^2_w}{\langle g, g \rangle_w}. \quad (30)$$

Since this is identical to (24), LogitBoost is a special case of QuadBoost. First and second order TaylorBoost are presented in Algorithm 1.

Algorithm 1 TaylorBoost

Input: Training set S_t, Number of weak learners in the final classifier, N, Loss function, $L[y, f(x)]$.
Initialization: Set $k = 0$, $f^k(x) = 0$
while $k < N$ **do**
  Compute $\nabla_{R_c(f^k)}(x_i)$ using (6)
  For first order TaylorBoost $w_i = 1$
  For second order method $w_i = \nabla^2_{R_c(f^k)}(x_i)$ by (7)
  Find the best weak learner, g^*, by (29)
  Find the optimal step, c_{g^*}, size by (15)
  $f^{k+1} = f^k + c_{g^*} g^*$ and $k = k + 1$
end while
Output: decision rule: $sign[f^N(x)]$

3.1. Properties

The TaylorBoost family has a number of interesting properties. First, it generalizes classical boosting. GradBoost is the family of algorithms derived from the first-order Taylor series expansion, and LogitBoost is the QuadBoost derived with the logistic loss. Second, it does not require arbitrary weight specification. Instead, for both first and second-order algorithms, the weights follow naturally from the optimization. Third, any convex $R_c(f)$ is also analytical, has a convergent Taylor series, and makes the problem of (2) convex. Hence, all TaylorBoost methods converge to the global optimum, if $R_c(f)$ is convex. Fourth, due to the more accurate approximation of QuadBoost, it selects better weak learners and has faster convergence rate. Finally, although second order methods are sometimes expensive (need to invert the Hessian), QuadBoost has complexity equivalent to that of GradBoost.

4. Loss and Risk function design

The discussion above is applicable to any choice of loss function $L[y, f]$ and weak learner set $\mathcal{G}$. In this section, we consider the choice of loss function. The design of loss functions suitable for classification is an extensively studied topic [6, 15, 3, 9]. Losses commonly used in machine learning are upper-bounds on the zero-one loss. When the classifier that minimizes the risk associated with a loss L converges asymptotically to the Bayes rule, as training samples increase, L is said to be *Bayes consistent* [15, 9].

Bayes consistency is a necessary condition for effective learning. It guarantees that, with infinite training data, the algorithm produces the optimal decision rule. It is, however, not sufficient, as it does not guarantee that the decision rule is the best possible for small training samples. The classification margin plays an important role in this regime, as larger margins guarantee improved generalization [14]. SVM and boosting guarantee a large margin by relying on margin enforcing losses. These are losses that assign a penalty to correctly classified examples which are near the classification border. The use of losses that are both Bayes consistent and margin enforcing guarantees good performance in both the small and large sample size regimes.

4.1. Margin control

The design of Bayes consistent loss functions has most recently been studied in [9]. This work introduced a generic framework for the derivation of such losses, showing that it is possible to augment Bayes consistency with other desirable loss properties. We have recently considered the problem of controlling the margin enforced by a loss, and shown that it is possible to derive Bayes consistent losses with explicit margin control [10]. This builds on a specialization of the loss design procedure of [9]. The procedure is based on a decomposition of the optimal predictor into $f(x) = l[\eta(x)]$, where $\eta(x) = P_{Y|X}(1|x)$ and $l(\eta)$ a link function. The specialization is as follows [10]

1. select any real, invertible, and increasing link function $l : [0, 1] \to R$ that satisfies

$$l^{-1}(-v) = 1 - l^{-1}(v). \tag{31}$$

2. compute $C(\eta) = -\int l(\eta)d\eta$

3. define the loss as $L(y, f) = \phi(yf)$ where

$$\phi(v) = C[l^{-1}(v)] - (1 - l^{-1}(v))v \tag{32}$$

It can be shown that, minimization of the risk associated with L, produces a classifier of conditional risk $C(\eta)$ [9]. The loss L is denoted *canonical* since $l(\eta) = -\frac{\partial C}{\partial \eta}$ [4, 10].

The ability to control the margin properties of canonical losses follows from the fact that they have a predicable form when l^{-1} is a sigmoidal function. In this case, it can be shown that $\phi(v)$ is convex, monotonically decreasing, linear (slope -1) for large negative v, constant for large positive v, and has slope $-1/2$ at the origin [10]. These properties can be seen in Figure 2, and are frequently used to justify the success of the logistic loss (a member of canonical losses) and LogitBoost. The only degrees of freedom are in the behavior of the loss around $v = 0$, i.e. its margin enforcing properties. These properties can be controlled by controlling the slope of the sigmoid l^{-1} around the origin. While the details depend on the selected sigmoidal function, there is usually a parameter which controls this slope. This parameter thus controls the margin properties of the loss [10].

4.2. A new family of canonical losses

Two losses with explicit margin control, the CBoost loss

$$\phi(v; a) = \frac{1}{2a}(\sqrt{4 + (av)^2} - av) \tag{33}$$

and the canonical logistic loss

$$\phi(v; a) = \frac{1}{a}[\log(1 + e^{(av)}) - av] \tag{34}$$

Table 1. Classification error of various combinations of boosting and loss functions on car, face, and pedestrian detection.

	Car		*Face*		*Pedestrian*	
Method	*Grad*	*Quad*	*Grad*	*Quad*	*Grad*	*Quad*
Exponential	27.54%	17.59%	9.14%	8.74%	7.45%	6.98%
Logistic	21.68%	12.59%	8.68%	8.62%	5.86%	5.45%
CBoost	34.77%	12.44%	10.15%	8.23%	7.34%	5.29%
CBoost(a^*)	13.76%	**12.44**%	8.29%	8.23%	5.47%	5.29%
Laplace	29.87%	12.97%	9.56%	8.16%	6.94%	5.28%
Laplace(a^*)	13.02%	12.97%	8.19%	**8.16**%	5.54%	**5.28**%

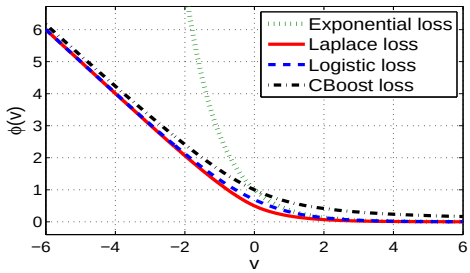

Figure 2. Laplace and other convex Bayes consistent loss functions.

were proposed in [10]. These losses are canonical extensions of those used by AdaBoost and LogitBoost, whose margin properties are controlled by the parameter a. In this work, we note that the family of canonical losses with margin control is much broader, using the fact that the cumulative distributions of many zero mean random variables 1) are sigmoidal functions with the symmetry of (31), and 2) enable control of the sigmoid slope by a simple parameter.

For example, the cumulative distribution of a the zero mean Laplacian random variable of variance $2a^2$, exhibits these properties and enables margin control through the variance parameter. Using this distribution as l^{-1} leads to

$$\begin{aligned} l(\eta) &= \begin{cases} a\log(2\eta) & \text{if } \eta < 0.5 \\ -a\log(-2\eta + 2) & \text{if } \eta \geq 0.5 \end{cases} \\ &= -asgn(2\eta - 1)\log(1 - |2\eta - 1|). \end{aligned} \tag{35}$$

Application of the procedure of Section 4.1 then produces the loss

$$\phi(v) = \frac{1}{2}[ae^{\frac{-|v|}{a}} + |v| - v], \tag{36}$$

which we denote *Laplace loss*, and is compared with a number of losses discussed above in Figure 2 (for $a = 1$).

5. Evaluation

We start with the problems of car, face, and pedestrian detection. The face dataset consists of $9,000$ face and $9,000$ non-face images, of size 24×24. Car detection is based on the UIUC dataset [1] of $1,100$ positives and $10,000$ negatives, of size 20×50. Pedestrian detection is based on the

MIT Pedestrian dataset [13] of $1,000$ positives and $10,000$ negatives, of size 40×20. In all cases, the data was split into five folds, four of which were used for training and one for testing. All experiments were repeated with each fold taking the role of test set, and the results averaged. Also regression on Haar wavelets were used as weak learners.

Table 1 presents the error rates of detectors learned with 25 iterations of TaylorBoost and the losses discussed above. Underlined entries are methods that were available before this work: GradBoost (for logistic and exponential loss), LogitBoost (QuadBoost + logistic loss), GentleBoost (QuadBoost + exponential loss) and (GradBoost + CBoost loss) [10]. For the losses with margin control, results are presented for both $a = 1$ and the cross-validated margin parameter a^*. A number of observations can be made.

First, best results are always obtained with a combination of TaylorBoost, a canonical loss, and margin cross-validation. The gains can be very substantial. For example, in Car detection, the lowest error of GradBoost with classical losses is 21.7%. Its combination with the Laplace loss and margin cross-validation reduces the error to 13%. Second, QuadBoost achieves better performance for all losses and datasets. Again, the gains can be substantial, e.g. from 34.8% to 12.4% on Car detection with the CBoost loss ($a = 1$). This is due to the better convergence rate of QuadBoost. Third, among the canonical losses, Laplace outperforms CBoost in 5 of 6 cases when $a = 1$, and 4 of 6 when the margin parameter is cross-validated. Fourth, when compared to the classic (Logistic and Exponential) losses, at least one of the canonical losses (with a^*) achieves smaller error on all cases, and this holds for both losses on 11 of 12. Fifth, margin cross-validation leads to more uniform performance across losses and optimization strategies.

Overall, these results support the following conclusions: 1) QuadBoost is more effective than GradBoost, 2) with margin cross validation, Laplace is usually the best of the losses, and 3) margin cross-validation increases both the classification accuracy and the robustness, by guaranteeing less sensitivity to the loss and the optimization strategy.

5.1. Discriminant Tracking

Discriminant tracking is a state of the art object tracking approach. A classifier is trained to distinguish the target object from the background in each video frame, and used in the next frame for tracking [2]. Various methods have been proposed to learn the classifier, including AdaBoost [2], discriminant saliency [8], and a combination of discriminant saliency and TangentBoost [11]. The latter achieved the best results in the literature. We implemented trackers that combine QuadBoost with the Laplace and CBoost losses. In all cases $a = 1$ since cross validation is too time consuming for the tracking application. We have also implemented trackers by LogitBoost, GentleBoost, and TangentBoost. Table 2 presents the error rates (as defined in [8]) of each method on five video clips. The combination of QuadBoost and canonical losses achieves the best performances. In particular, Laplace+QuadBoost and CBoost+QuadBoost each achieve the best result on two clips, while TangentBoost has the lowest error rate on the last.

Table 2. Tracking accuracy for five video clips.

Method	Tangent	Logit	Gentle	Lap+Quad	CBoost+Quad
Gravel	18.6 %	18.8 %	19.2%	18.8 %	**17.8** %
Athlete	36.6 %	37.2 %	36.4%	35.8 %	**35.4** %
Karls	79.9 %	31.7 %	33.7%	**31.4** %	35.3 %
Montins	86.9 %	92.2 %	92.5%	**83.4** %	87.7 %
Plush	**8.9**%	9.4 %	9.5%	10.2 %	10.9 %

References

[1] S. Agarwal, A. Awan, D. Roth, and I. C. Society. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. PAMI*, 26:1475 –1490, 2004. 2933

[2] S. Avidan. Ensemble tracking. *IEEE PAMI*, 29(2):261–271, 2007. 2934

[3] P. Bartlett, M. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *JASA*, 2006. 2929, 2932

[4] A. Buja, W. Stuetzle, and Y. Shen. Loss functions for binary class probability estimation and classification: Structure and applications. *(Tech. Rep.) Univ. of Pennsylvania*, 2005. 2933

[5] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Comp. and Sys. Science*, 1997. 2929, 2931

[6] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28, 1998. 2929, 2931, 2932

[7] B. A. Frigyik, S. Srivastava, and M. R. Gupta. An introduction to functional derivatives. *Technical Report(University of Washington)*, 2008. 2930

[8] V. Mahadevan and N. Vasconcelos. Saliency-based discriminant tracking. In *CVPR*, 2009. 2934

[9] H. Masnadi-Shirazi and N. Vasconcelos. On the design of loss functions for classification: theory, robustness to outliers, and savageboost. In *NIPS*, 2008. 2929, 2932, 2933

[10] H. Masnadi-Shirazi and N. Vasconcelos. Variable margin losses for classifier design. In *NIPS*, 2010. 2929, 2933, 2934

[11] H. Masnadi-Shirazi, N. Vasconcelos, and V. Mahadevan. On the design of robust classifiers for computer vision. In *CVPR*, 2010. 2929, 2934

[12] L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithms as gradient descent. In *NIPS*, 2000. 2929, 2931

[13] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *CVPR*, pages 193–99, 1997. 2934

[14] V. N. Vapnik. *Statistical Learning Theory*. John Wiley Sons Inc, 1998. 2929, 2932

[15] T. Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, 2004. 2929, 2932