TOTALLY-CORRECTIVE BOOSTING USING CONTINUOUS-VALUED WEAK LEARNERS

[1]*Chensheng Sun,* [2]*Sanyuan Zhao,* [1]*Jiwei Hu,* [1]*Kin-Man Lam*

[1]Center for Signal Processing, Department of Electronic and Information Engineering,
The Hong Kong Polytechnic University, Hong Kong, China
[2]School of Information and Electronic Engineering,
Beijing Institute of Technology, Beijing, China

ABSTRACT

The Boosting algorithm has two main variants: the gradient Boosting and the totally-corrective column-generation Boosting. Recently, the latter has received increasing attention since it exhibits a better convergence property, thus resulting in more efficient strong learners. In this work, we point out that the totally-corrective column-generation Boosting is equivalent to the gradient-descent method for the gradient Boosting in the weak-learner selection criterion, but uses additional totally-corrective updates for the weak-learner weights. Therefore, other techniques for the gradient Boosting that produce continuous-valued weak learners, e.g. step-wise direct minimization and Newtons method, may also be used in combination with the totally-corrective procedure. In this work we take the well known AdaBoost algorithm as an example, and show that employing the continuous-valued weak learners improves the performance when used with the totally-corrective weak-learner weight update.

Index Terms— Boosting, totally corrective, column generation, gradient

1. INTRODUCTION

The Boosting algorithm is one of the most important techniques in machine learning. It is a supervised learning method that learns a function $F(\mathbf{x})$ to accurately predict the label y of the sample $\mathbf{x}$. The learned function $F(\mathbf{x})$, usually termed a strong learner, is a non-negative linear combination of a number of less powerful prediction functions $\{f_j(\mathbf{x})\}$, usually known as weak learners:

$$F(\mathbf{x}) = \sum_{j=1}^{N} w_j f_j(\mathbf{x}), w_j \geq 0. \quad (1)$$

By properly selecting the weak learners and determining their weights, the strong learner can be very accurate. Furthermore, feature selection is naturally embedded in the Boosting algorithm. Considering each weak learner uses only a single feature, the resulting strong learner effectively selects a subset of features to make predictions. This is especially important in some applications, e.g. visual object recognition, where a large number of features are available while the computational power is relatively limited. For example, efficient face detectors are built by the Boosting method in [1].

Training a Boosting classifier is a step-wise procedure to optimize a loss functional $L(F)$. Starting with $F_0(\mathbf{x}) = 0$, and given a pool of candidate weak learners $\mathcal{H} = \{h_1, h_2, ...\}$, in each round t a new weak learner f_t is selected from $\mathcal{H}$, and added into $F_{t-1}(\mathbf{x})$. The two main paradigms of the Boosting algorithm are the gradient Boosting methods [2] and the totally-corrective column-generation Boosting methods [3]. The biggest difference between these methods is how the weak learners' weights are adjusted. For the gradient Boosting, only the newly selected weak learner's weight is decided, and all the previous weak learners' weights are unmodified. The process draws a path in the functional space of F that approaches the optimal value of L. Many works follow this paradigm, e.g. the AdaBoost [2] using the exponential loss, the LogitBoost [2] using the logit loss, the AnyBoost [4] that generalizes to any convex loss, and the multi-instance AdaBoost [5] that deals with the multi-instance problem. In contrast, the totally-corrective Boosting methods modify all weak learners weights in every round, in order to find the optimal combination that minimizes L. The outcome is a faster convergence speed and assured convergence in finite steps, while the gradient-based methods only converge at the limit. The first totally-corrective Boosting algorithm is the LPBoost [3], which minimizes the hinge loss and introduces the column-generation technique for selecting new weak learners. The column-generation technique is adopted in later methods, e.g. [6], and is accepted as a standard for totally-corrective Boosting.

The column-generation method for selecting new weak learners is identical to the gradient descent in gradient Boosting, which usually limits the weak learners to binary classifiers. In this work we propose other techniques in gradient Boosting, e.g. step-wise direct minimization and Newton's method, that output continuous-value predictions, are equally suitable for the totally-corrective update. Experimental results show that for totally corrective Boosting, using continuous-valued weak learners also results in faster conver-

gence than using the binary classifiers does.

The two paradigms of Boosting algorithms will be detailed in Section 2. Then in Section 3 we take the AdaBoost algorithm as an example, and show the details of using continuous-valued weak learners with totally-corrective update. Section 4 gives the experimental results proving the efficacy of the method. We conclude the paper in Section 5.

2. TWO PARADIGMS OF THE BOOSTING ALGORITHM

Training a Boosting classifier involves optimizing a loss functional L. Given a training set with ground-true labels $\{(\mathbf{x}_i, y_i), i = 1, ..., M\}$, we optimize the empirical loss, which is often the sum of the loss on individual samples:

$$\min_{F} \quad L(F) = \sum_{i=1}^{M} l(F(\mathbf{x}_i), y_i). \tag{2}$$

For example, the exponential loss $l(F(\mathbf{x}_i), y_i) = e^{-y_i F(\mathbf{x}_i)}$ is used in AdaBoost, and the Hinge loss $l(F(\mathbf{x}_i), y_i) = \max(0, -y_i F(\mathbf{x}_i))$ is used in LPBoost.

2.1. Gradient Boosting

The gradient Boosting deals with differentiable convex loss. Consider the strong-learner score $\mathbf{F} = [F(\mathbf{x}_1), ..., F(\mathbf{x}_M)]$ as the variables, the gradient-based Boosting solves an unconstrained optimization problem for the M-D vector variable $\mathbf{F}$, using a step-wise additive update in the form of $F_{t+1} \leftarrow F_t(\mathbf{x}) + w_{t+1} f_{t+1}(\mathbf{x})$. The three major techniques for solving this problem are the gradient-descent method, the step-wise direct minimization method, and the Newtons method.

2.1.1. Gradient descent

For the gradient-descent method, a new weak learner is found to approximate the negative gradient direction of $L(F)$:

$$f_{t+1}(\mathbf{x}) = \arg\max_{h} \sum_{i=1}^{M} -\frac{\partial L}{\partial F(\mathbf{x}_i)} h(\mathbf{x}_i). \tag{3}$$

For 2-class classification, the loss is usually a function of the margin $z = yF(\mathbf{x})$. Therefore, approximation of the negative gradient direction can be implemented by minimizing the 0-1 error under the sample weights $\omega_i = -\partial L / \partial z_i$:

$$f_{t+1}(\mathbf{x}) = \arg\min_{h} \sum_{i=1}^{M} \omega_i (-y_i h(\mathbf{x}_i)). \tag{4}$$

Then the weight w_{t+1} of the new weak learner, i.e. the step size in the approximate negative gradient direction, is determined by line search.

2.1.2. Step-wise direct minimization

For direct minimization, a new weak learner is found to directly minimize $L(F + f)$. The form of the new weak learner can be obtained by setting the gradient to 0:

$$\frac{\partial L(F + f)}{\partial f} = 0. \tag{5}$$

2.1.3. Newton's method

For the Newton's method, the new weak-learner is obtained from:

$$f = H^{-1} \frac{\partial L}{\partial F}, \tag{6}$$

where H is the Hessian of the loss functional, evaluated at the current strong learner F. For the step-wise direct minimization and the Newton's method, the new weak-learner's weight is 1.

All these gradient Boosting methods update the strong learner F in an additive manner, keeping the weights of previously selected weak learners unchanged. In the functional space the strong learner F gradually moves towards the optimal position.

2.2. Totally-corrective column-generation Boosting

To derive the totally-corrective column-generation Boosting, we assume all weak learners can be obtained in advance, and we introduce the matrix H such that the j-th column $H_{:j}$ is the score of the j-th weak learner from the pool $\mathcal{H} = \{h_1, h_2, ...\}$, i.e.:

$$H_{:j} = [h_j(\mathbf{x}_1), ..., h_j(\mathbf{x}_M)]^{\mathrm{T}}. \tag{7}$$

Then the Boosting problem is to find an optimal linear combination that minimizes the loss, subject to proper constraints on the weights to make the problem well formed:

$$\min_{\mathbf{w}} \quad L(H\mathbf{w}) \quad s.t. \quad \mathbf{w} \geq 0, \ \mathbf{1}^{\mathrm{T}}\mathbf{w} = 1/T\,, \tag{8}$$

where $\mathbf{1}$ is a column vector of all 1s, such that we constrain the sum of weak-learner weights to a fixed value. The L-1 norm constraint on $\mathbf{w}$ introduces sparsity into the solution, such that only a few elements of $\mathbf{w}$ are non-zero, and efficient feature selection can be achieved. T is a hyper parameter set by the user. In experiments we set T according to the sum of weak learner weights in non-totally-corrective Boosting.

Since the number of columns in H is usually very large, the problem cannot be directly solved. The column-generation technique is suitable for such problems. After introducing the auxiliary variable $\mathbf{F} = H\mathbf{w}$, the dual problem can be written as follows:

$$\max_{\mathbf{u}, r} \quad \inf_{\mathbf{F}, \mathbf{w}} \mathcal{L} \quad s.t. \quad \sum_{i=1}^{M} u_i H_{ij} \leq r, \ \cdots \tag{9}$$

where $\mathcal{L}$ is the Lagrangian:

$$\mathcal{L}(\mathbf{F},\mathbf{w},\mathbf{u},\mathbf{q},r) = L(\mathbf{F}) + \sum_{i=1}^{M} u_i (F_i - H_{i:}\mathbf{w}) \\ -\mathbf{q}^{\mathrm{T}}\mathbf{w} + r\left(1^{\mathrm{T}}\mathbf{w} - 1/T\right). \tag{10}$$

$\mathbf{u}$ is the Lagrange multiplier corresponding to the equality constraint $\mathbf{F} = H\mathbf{w}$. $H_{i:}$ is the i-th row in H. From the KKT condition $\partial\mathcal{L}/\partial\mathbf{F} = 0$, we can obtain $u_i = -\partial L/\partial F_i$, i.e. u_i is the i-th negative partial derivative of the loss L. "$\cdots$" stands for the other constraints implied by the KKT conditions.

In the dual problem, each column of H corresponds to a constraint $\sum_{i=1}^{M} u_i H_{ij} \leq r$. The column-generation technique works with a reduced matrix $\hat{H}$ containing only a subset of columns from H, equivalent to relaxing the dual problem. Then in each iteration the most violated constraint that is not included in $\hat{H}$ is found and added to $\hat{H}$:

$$H_{:t+1} = \arg\max_{h} \sum_{i=1}^{M} u_i h(\mathbf{x}_i),\ \hat{H} \leftarrow \left[\hat{H}, H_{:t+1}\right]. \tag{11}$$

Thus, a new weak learner is selected, and then the weak-learner weights are updated by solving the primal problem (8), resulting in a totally-corrective Boosting algorithm.

3. CONTINUOUS-VALUED WEAK LEARNERS WITH TOTALLY-CORRECTIVE UPDATES

It can be observed that the column-generation criterion for selecting the new weak learner is identical to the gradient-descent criterion in the gradient Boosting. Indeed, most of the previous column-generation Boosting methods also limit the weak learners to binary classifiers, and select the new weak learner to minimize the weighted 0-1 loss.

Previous works have shown that the step-wise direct minimization and Newtons method can make the gradient Boosting algorithm converge faster. An intuitive thought is that the convergence speed of the totally-corrective Boosting can be further increased by using these methods for weak-learner selection.

We take the well-studied AdaBoost as an example. The direct-minimization step and Newton's method step for AdaBoost are known as Real-AdaBoost and Gentle-AdaBoost, respectively. In Real-AdaBoost, the direct minimization step results in weak learners in the form of the half log-odds:

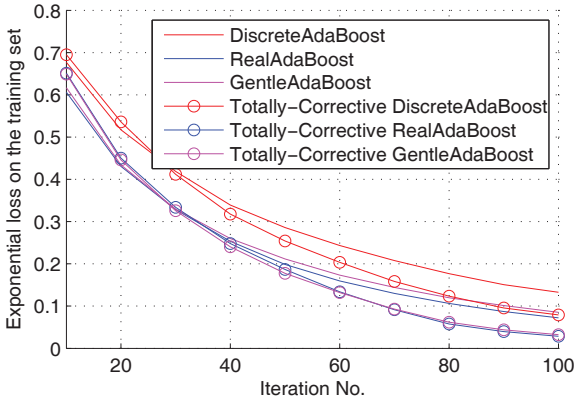
$$f_j(\mathbf{x}) = \frac{1}{2}\log\frac{P(y=1,\mathbf{x})\,e^{-F(\mathbf{x})}}{P(y=-1,\mathbf{x})\,e^{F(\mathbf{x})}} \\ = \frac{1}{2}\log\frac{P_\omega(y=1|\mathbf{x})}{P_\omega(y=-1|\mathbf{x})}. \tag{12}$$

In Gentle-AdaBoost, the new weak learner is the difference between the posteriors:

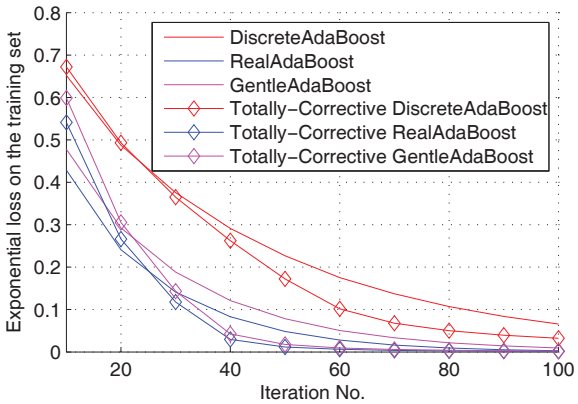
$$f_j(\mathbf{x}) = P_\omega(y=1|\mathbf{x}) - P_\omega(y=-1|\mathbf{x}), \tag{13}$$

where the subscript ω denotes a sample-weighted distribution using the weights $\omega_i = -\partial L/\partial z_i$. For both methods, the weak learners are regression functions that take continuous values. Following the traditional methods, the weak learner that leads to the greatest loss decrease is selected.

The newly selected weak learner is going to be used as a new column of H. Due to the L-1 norm constraint on w in the totally-corrective primal problem, regularization on the columns of H is also necessary to keep the problem well formed. For binary weak learners $f(\mathbf{x}) \in \{-1,+1\}$, the columns of H are implicitly restricted by $\|H_{:j}\|_1 = M$. Therefore, we also process the continuous-valued weak learners such that the new column of H has a fixed L-1 norm M.



(a) Using stump weak learners



(b) Using LUT weak learners

Fig. 1. Convergence of loss on the training data

4. EXPERIMENTS

We evaluate the totally-corrective AdaBoost using various criteria for weak-learner selection, i.e. the gradient-descent method used in Discrete-AdaBoost and column-generation Boosting; the direct minimization, which leads to the log-odds estimators used in Real-AdaBoost; and Newton's

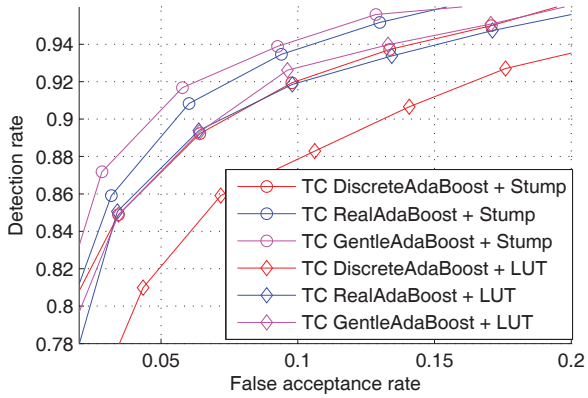


Fig. 2. ROC curves on the test set

method, which leads to the posterior-difference estimators used in Gentle-AdaBoost. The totally-corrective update is performed by solving the Boosting primal problem (8), using the MOSEK optimization package.

The two most commonly used types of weak learners, the stumps and the LUTs are used in our experiments. A stump consists of a threshold and two values on the two sides of the threshold:

$$f(x) = \begin{cases} v_1 & x > th \\ v_2 & x \leq th \end{cases} \tag{14}$$

For the three different weak-learner selection criteria, different stumps are trained. The decision stump sets the value on one side to 1, and on the other side to -1, according to the optimality criterion of (3). The regression stump for the log-odds and posterior difference computes the values according to the samples falling on each side. For the LUTs, the range of feature values is partitioned into bins of equal size, and for each bin the value is decided similarly to the stumps. We partition the value range into 64 bins according to the observed minimum and maximum values for each feature. The partition is fixed, but the bin values are updated in the learning process.

We evaluate the methods in the pedestrian-detection problem, using the INRIA dataset. The HOG feature [7] is used. We extract the HOG feature for 619 blocks of size 16x16, 32x32 and 64x64, resulting in a feature vector of length 22,284. Then, each weak learner is based on a single feature. We train the Boosting strong classifier for an intermediate stage of a cascaded detector, where the negative training set consists of fairly difficult samples. We use 2,474 positive samples and 5,000 negative samples to train the classifiers.

The AdaBoost objective is to minimize the exponential loss on the training data. Therefore, we plot the loss function value against the number of weak learners in Fig.1. It can be observed that for non-totally-corrective AdaBoost, the stage-wise direct minimization and Newtons method, both using continuous-valued weak learners, cause the loss decreases faster than the binary classifiers used in Discrete-AdaBoost. But updating the weak learner weights in the totally-corrective manner significantly improves the convergence speed for the binary classifier weak learners. In addition, combining the continuous-valued weak learners with the totally-corrective update further accelerates the convergence speed, for both stumps and LUTs. In Fig.2, the ROC curve on the test set is shown for totally-corrective AdaBoost with different types of weak learners. It can be seen that the generalization performance is also improved by using continuous-valued weak learners. The results also indicate that the LUT weak learners are more easily overfitted with the training data than the stumps are.

5. CONCLUSIONS

The totally-corrective Boosting is an important advance on the existing Boosting algorithms, and also provides new insights into the nature of the Boosting algorithm generally. Though the column-generation method implies binary classifiers as weak learners, thus emulating the gradient-descent in gradient Boosting, our experiments show that other techniques used in gradient Boosting that outperform gradient descent also lead to better results in combination with totally-corrective updates. In our future research, we hope that the theoretical proof can be found to support our algorithm, and we will also try to apply the hybrid method to other variants of Boosting algorithms.

6. REFERENCES

[1] Paul Viola and Michael J. Jones, "Robust real-time face detection," *International Journal on Computer Vision*, vol. 57(2), pp. 137–154, 2004.

[2] Jorome Friedman, Trevor Hastie, and Robert Tibshirani, "Additive logistic regression: A statistical view of boosting," *The Annals of Statistics*, vol. 28, pp. 337–407, 2000.

[3] Ayhan Demiriz, Kristin P. Bennett, and John Shawe-Taylor, "Linear programming boosting via column generation," *Machine Learning*, vol. 46, no. 1-3, pp. 225–254, 2002.

[4] Llew Mason, Jonathan Baxter, Peter L. Bartlett, and Marcus R. Frean, "Boosting algorithms as gradient descent," in *NIPS*, 1999, pp. 512–518.

[5] Paul A. Viola, John C. Platt, and Cha Zhang, "Multiple instance boosting for object detection," in *NIPS*, 2005.

[6] Chunhua Shen and Hanxi Li, "On the dual formulation of boosting algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2216–2231, 2010.

[7] Navneet Dalal and Bill Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005, pp. 886–893.