Available online at www.sciencedirect.com
ScienceDirect
Computational Statistics & Data Analysis 52 (2007) 325–334
COMPUTATIONAL STATISTICS & DATA ANALYSIS
www.elsevier.com/locate/csda

A stochastic approximation view of boosting

C. Andy Tsao[a,*], Yuan-chin Ivan Chang[b,c]

[a] *Department of Applied Mathematics, National Dong Hwa University, TW 97401, Taiwan*
[b] *Institute of Statistical Science, Academia Sinica, Taipei, TW 11529, Taiwan*
[c] *Department of Statistics, National Chengchi University, Taipei, TW 11605, Taiwan*

Available online 1 July 2007

Abstract

The boosting as a stochastic approximation algorithm is considered. This new interpretation provides an alternative theoretical framework for investigation. Following the results of stochastic approximation theory a stochastic approximation boosting algorithm, SABoost, is proposed. By adjusting its step sizes, SABoost will have different kinds of properties. Empirically, it is found that SABoost with a small step size will have smaller training and testing errors difference, and when the step size becomes large, it tends to overfit (i.e. bias towards training scenarios). This choice of step size can be viewed as a smooth (early) stopping rule. The performance of AdaBoost is compared and contrasted.

Keywords: Boosting; Stochastic approximation; Robbins–Monro procedure; Smooth early stopping

1. Introduction

Boosting is one of the successful ensemble classifiers and has attracted much attention recently. Earlier theoretical results show that boosting reduces the training error under the *weak base hypothesis assumption* and upper bounds of its testing errors can be obtained in some PAC (probably approximately correct) sense, see, for example, Freund and Schapire (1997). Boosting is also recognized as a large margin algorithm, see, for example, Schapire et al. (1998) and Breiman (1999). It has been observed that boosting is relatively resistant to overfitting in many practical applications with less noisy data. These empirical successes motivate theoretical investigations of possible explanations. See, for example, Schapire (1990), Grove and Schuurmans (1998), Freund and Schapire (1997), Friedman et al. (2000), Schapire et al. (1998) and Jiang (2002). The readers are referred to Meir and Rätsch (2003) for a review of more recent advancements. Recent studies provide a much clearer picture of Bayes consistency of boosting. Breiman (2004) shows that population version boosting is Bayes consistent and Jiang (2004) shows boosting is process consistent for the sample version. Other asymptotic aspects are also studied, for example Bühlmann and Yu (2003) and Zhang and Yu (2005). More recent studies can be found in, for example, Leitenstorfer and Tutz (2007), Merler et al. (2007), Tutz and Binder (2006), Gey and Poggi (2006) and Kim and Koo (2005,2006).

Although there are many modifications and interpretations of the original AdaBoost, many of its features remain unclear. The more we know about its properties, the more we can exploit its power. In this paper, the boosting-like algorithms are viewed as stochastic approximation procedures. Particularly, these algorithms are rendered by

* Corresponding author.
E-mail addresses: chtsao@mail.ndhu.edu.tw (C.A. Tsao), ycchang@sinica.edu.tw (Y.-c.I. Chang).

0167-9473/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.csda.2007.06.020

the Robbins–Monro (RM) procedure in Robbins and Monro (1951). This stochastic approximation viewpoint is then contrasted with the interpretation of (Discrete) AdaBoost given in Friedman et al. (2000).

Robbins and Monro (1951) develops and analyzes a recursive procedure for finding the root of a real-value function $g(\cdot)$ of a real variable θ. This function is assumed to be unknown, but noise-corrupted observations could be taken at values of θ selected by the experimenter. If the function $g(\cdot)$ were known and continuously differentiable, then finding its roots becomes a classical problem in numerical analysis. Various procedures such as Newton's method have been proposed in this case. However, when the observation is noise-corrupted, these procedures might not work well. Many modifications and alternatives have been introduced to account for noise. RM procedure and Kiefer–Wolfowitz (KW) procedure, respectively, proposed in Robbins and Monro (1951) and Kiefer and Wolfowitz (1952) are two of the most well-known procedures being proposed under stochastic approximation frameworks. For simplicity, only the rendering based on the deterministic RM procedure will be discussed. The relations of boosting-like algorithms to other modifications of the RM and KW procedures will be reported elsewhere.

Following the convergence theorem of the RM procedure, we propose a stochastic approximation boosting algorithm, SABoost. By adjusting the step size of SABoost, it will have different properties. Empirically, if a large step size is chosen, the training errors of SABoost will decrease quickly and tend to overfit. On the other hand, when a small step size is chosen, the training errors decrease slowly, and the phenomena of overfitting becomes negligible. In addition, the SABoost with small step size is more stable than AdaBoost in the sense that no drastic change of training and testing errors presents during boosting iterations.

The rest of this paper is organized as follows. In order to redescribe AdaBoost using a stochastic approximation procedure, we will first review AdaBoost and the RM procedure in Sections 2 and 3, respectively. The convergence theorem of the RM procedure is stated in Section 3. Empirical results of some benchmark and the synthesized data sets are shown in Section 4. Concluding remarks, discussion and some possible future studies are given in Section 5.

2. FHT's interpretation

Boosting (refer Freund and Schapire, 1997; Schapire, 1990, for more details) is a learning procedure that starts with a weak base learner and assigns the misclassified data higher weights in each iteration. Then it takes a weighted majority vote to predict labels of examples with given feature variables. The process is then repeated T iterations. This T is known as the boosting time and can be configured by users for early stopping. Although many variations and modifications of boosting have been proposed over the years, here we will mainly focus on the Discrete AdaBoost for easy exposition.

To fix notations, let us consider a supervised learning problem with binary responses where training data $(x_i, y_i)_1^n$, testing data $(x'_j, y'_j)_1^m$ and $y_i, y'_j = \in \mathcal{Y} = \{\pm 1\}$ are given. Here x's and x''s are "input" or "explanatory" variables and y's and y''s are output or response variables. It is desired to find a good *machine or classifier* F based on the information of training data such that F can predict y' for each x' as correctly as possible. Usually the performance of F is measured through its training error (TE) and testing/generalization error (GE) defined as

$$TE = \frac{1}{n}\sum_i 1_{[y_i \neq F(x_i)]} \quad \text{and} \quad GE = \frac{1}{m}\sum_j 1_{[y'_j \neq F(x'_j)]}.$$

Note that it is often assumed that X and Y arised from some (unknown) joint distribution. Rigorously, testing error should be $E_{Y,X}1_{[Y\neq F(X)]}$ rather than GE we defined above. However, the sampling analog GE can be considered as an estimate of $E_{Y,X}1_{[Y\neq F(X)]}$. For large m, GE is a satisfactory estimator in the sense of (statistical) consistency.

Now, the Discrete AdaBoost algorithm:

Discrete AdaBoost.

(1) Start with weights $D_t(i) = 1/n$, $i = 1$ to n.
(2) Repeat for $t = 1$ to T
- Obtain $h_t(x)$ from weak learner h using weighted training data with respect to D_t.
- Compute $\varepsilon_t = E_{D_t}1_{[y\neq h_t(x)]}$, $\alpha_t = \log\frac{1-\varepsilon_t}{\varepsilon_t}$.

- Update $i = 1$ to n,

$$D_{t+1}(i) = \frac{1}{Z_t} D_t(i) \exp[\alpha_t 1_{[y_i \neq h_t(x_i)]}],$$

where Z_t is the normalizer.

(3) Output the classifier $F(x) = \text{sgn}[\sum_{t=1}^{T} \alpha_t h_t(x)]$.

Friedman et al. (2000) (FHT) show that

Remark 1. The Discrete AdaBoost (population version) builds an additive logistic regression model via Newton-like updates for minimizing $E(e^{-YF(X)})$.

This is an insightful and powerful observation. Immediately, it motivates construction of new boosting-like algorithms and provides a statistical framework for further theoretical investigations.

Specifically, let $Y \in \{\pm 1\}$ denote the labels and $\mathcal{X}$ be the domain of feature variables. Suppose F is a real-valued function mapping $\mathcal{X}$ to $\mathcal{R}$. Then in binary classification problems, our goal is to predict Y by the sign of estimated F based on a set of labeled examples. Friedman et al. (2000) render AdaBoost as a Newton update minimizing an approximate risk $J(F) \equiv E_{X,Y}[e^{-YF(X)}] \approx E_{X,Y} 1_{[YF(X)<0]}$ and obtain the update formula

$$F(x) \leftarrow F(x) + \frac{1}{2} \log\left(\frac{1-err}{err}\right) f(x), \tag{1}$$

$$w(x, y) \leftarrow w(x, y) \exp\left[\log\left(\frac{1-err}{err}\right) 1_{[y \neq f(x)]}\right], \tag{2}$$

where $f(x) = \text{sgn}(E_w(Y|x))$ and $err = E_w[1_{[Yf(x)<0]}|x]$.

Alternatively,

$$F_{t+1}(x) = F_t(x) + \alpha_t \ f(x), \tag{3}$$

where $\alpha_t = \frac{1}{2} \log(\frac{1-\varepsilon_t}{\varepsilon_t})$, $w_t(x, y) = \exp(-yF_t(x))$, $E_{w_t} g(x, Y)$ denotes $E_{Y|x}[w_t(x, Y)g(x, Y)]/(E_{Y|x} w_t(x, Y))$ and $\varepsilon_t = E_{w_t} 1_{[yf(x)<0]}$.

Note that this interpretation is based on minimization of an (approximated) conditional risk where X is given. Furthermore, it is in nature a population-version theory in the sense that X and Y are considered as random vector/variable. And when X is conditioned, the optimization problem is essentially a problem of sample size 1. This conditional risk approach greatly simplifies the theoretical analysis.

3. Stochastic approximation

Minimization of conditional risk is hardly a new task for statisticians. Specifically, now the objective is to find F_* such that

$$F_*(x) = \arg\min_{F(x)} \Psi(F(x)) \tag{4}$$

where $\Psi(F(x)) = E_{Y|x} L(Y, F(x))$. Under some regularity conditions, this F_* is also a solution to

$$\psi(F(x)) \equiv \Psi'(F(x)) = E_{Y|x} L'(Y, F(x)) = 0 \tag{5}$$

where $L'(Y, F) = \partial L(Y, F)/\partial F$. This problem has long been studied from a stochastic approximation viewpoint. See Lai (2003) for a review on stochastic approximation. We consider here the most well-known RM procedure and restate the RM deterministic (RM-D) algorithm given in Duflo (1997) in our current context and notations. The readers are referred to Duflo (1997) and Robbins and Monro (1951) for their original formulation.

RM-D Algorithm.

- Choose F_0 arbitrarily.

- Iterate $t = 0, 1, \ldots,$

$$g_{t+1}(x) = \psi(F_t(x)),$$
$$F_{t+1}(x) = F_t(x) - \gamma_t(g_{t+1} + \delta_{t+1}), \quad \gamma_t, \delta_{t+1} \geqslant 0.$$

Note that in usual boosting-like algorithms, there is no δ_t terms, that is, $\delta_t = 0$. However, as we will see immediately in the coming proposition, with suitable regularization, such δ's can be introduced without affecting the convergence.

Although Friedman et al. (2000) provide a good interpretation of boosting, their Result 1 leaves us no guidance about how to check the convergence of Newton updates. In contrast, the convergence of stochastic approximation algorithms have been established under various conditions. The following proposition is an adaption of the convergence theorem of the RM-D algorithm; for example, see Sections 1.2 and 1.4 in Duflo (1997). Here we restate Proposition 1.2.3 in Duflo (1997) in our notations. Note that the dependency of x is notationally suppressed.

Proposition 2 (*RM-D convergence*). *If ψ is a monotonic continuous real function such that $\psi(F_*) = 0$ and for all F*

$$\psi(F)\ (F - F_*) > 0,$$
$$|\psi(F)| \leqslant K(1 + |F|) \quad \textit{for some } K.$$

Suppose $\{\delta_t\}$, $\{\gamma_t\}$ are two sequences of real numbers and $\gamma_t \geqslant 0$. Define

$$F_{t+1} = F_t - \gamma_t(\psi(F_t) + \delta_{t+1}). \tag{6}$$

If γ_t decreases to 0 such that

$$\sum_t \gamma_t = \infty; \quad \sum_t \gamma_t \delta_t < \infty, \tag{7}$$

then F_t converges to F_* for any initial F_0.

Sequences γ_t and δ_t control the speed of approximation of the stochastic approximation algorithm. In particular, γ_t controls the "step size" of stochastic approximation. The common choice for γ_t is γ/t and $\delta_t = 0$, with some $\gamma > 0$.

When the exponential criterion is employed, that is $L(Y, F) = \mathrm{e}^{-YF}$, and then $L'(Y, F) = -Y\mathrm{e}^{-YF}$, the algorithms can be further simplified and render forms similar to the original AdaBoost.

RM-D Algorithm with Exponential Criterion.

- Choose F_0 arbitrarily.
- Iterate $t = 0, 1, \ldots,$

$$\begin{aligned} g_{t+1}(x) &= -E_{Y|x} Y \exp(-Y F_t(x)) \\ F_{t+1}(x) &= F_t(x) - \gamma_t g_{t+1}(x) \\ &= F_t(x) + \gamma_t (E_{Y|x} w_t(x, Y)) E_{w_t}(Y_{t+1}|x), \quad \gamma_t \geqslant 0 \end{aligned} \tag{8}$$

where $w_t(x, y) = \exp(-y F_t(x))$ and again

$$E_{w_t} g(x, Y) = \frac{E_{Y|x}[w_t(x, Y) g(x, Y)]}{E_{Y|x} w_t(x, Y)}.$$

The original form (6) suggests our approach is closely related to Friedman (2001,2002) since the gradient of the conditional risk is employed in updating iteration. However, Friedman (2001,2002) are of the sample-version theories. Both account for the finite sample in choosing the optimal parameters in the classifier F. Furthermore, the form of the possible classifiers are explicitly given as the addition of some simple functions (base learners). Our approach is a population-version theory. Our formulation is more related to Friedman et al. (2000). This approach does not specify the explicit form of the classifiers and therefore not directly allows optimization of the parameters. Although this population approach simplifies the notations and analytical derivation, it provides no clear guideline as how to use

the data and how to utilize the base learners. Nonetheless, our results suggest stochastic approximation can serve as an alternative framework for investigating boosting-like algorithms.

The iterative algorithms (8) and (3) bear formal similarity. Under the conditions of Proposition 2 the convergence of (8) can be established. On the hand, the convergence of the population-version Discrete AdaBoost (3) is less clear. It should be noted that the population-version results do not imply the corresponding algorithms will have the same convergence when working with finite sample data.

It follows from Proposition 2 that we can manipulate sequences $\{\gamma_t\}$, $\{\delta_t\}$ such that the boosting procedures will have different kinds of properties. These properties will be illustrated with some bench mark data sets and synthesized data in the next section. In contrast with the original AdaBoost, we call this procedure SABoost. SABoost is similar to Discrete AdaBoost except that the "step size" α_t is replaced by $s_t = \gamma_t E_{Y|x} w_t(x, Y)$ where $\gamma_t = \gamma/t$ for some $\gamma > 0$. This choice of γ_t is also supported by a widely stated sufficient condition for convergence of a stochastic approximation procedure

$$\sum_{t=1}^{\infty} \gamma_t = \infty; \quad \sum_{t=1}^{\infty} \gamma_t^2 < \infty \tag{9}$$

under quite general settings. See, for example, Duflo (1997). With choices of γ and thus the step size, SABoost can have quite different performances.

4. Empirical results

4.1. Bench mark data sets

For evaluating the performances of SABoost, we conduct experiments with both synthesized and some bench mark data sets. Here we first compare the performances of SABoost with Discrete AdaBoost with decision stumps (DS) as base learners for both procedures. Different step sizes are used in SABoost. In our experiments, we set $\delta_t = 0$, $\gamma_t = \gamma/t$ and $\gamma > 0$ is a positive real number and choose $\gamma = 0.5$, 1, and 1.5. All experiments are done on PC's running Window XP system. The statistical programming language R is used for all experiments. (cf. Verzani, 2005, and www.r-project.org for the details of language R.)

In order to prevent the slow convergence rate of SABoost due to the small step size γ, we propose a hybrid procedure combining SABoost and AdaBoost together in the following way: (1) Let T be the pre-chosen total iteration number and conduct the original Discrete AdaBoost procedure in the first $[\mu * T]$ ($\mu \in (0, 1)$) iterations, where $[A]$, $A \in R$, denotes the largest integer less than A. (2) Switch to SABoost for the rest $T - [\mu * T]$ of iterations. It will be called *Hybrid SABoost* here. In all our experiments, we set $\mu = 0.25$. Specifically

SABoost. (1) Start with weights $D_t(i) = 1/n$, $i = 1$ to n and $C_t = 1$.
(2) Repeat for $t = 1$ to T

- Obtain $h_t(x)$ from weak learner h using weighted training data with respect to D_t.
- Compute $\varepsilon_t = E_{D_t} 1_{[y \neq h_t(x)]}$, $s_t = \text{sgn}(0.5 - \varepsilon_t)\gamma_t * C_t$.
- Update $i = 1$ to n,

$$D_{t+1}(i) = \frac{1}{Z_t} D_t(i) \exp[s_t 1_{[y_i \neq h_t(x_i)]}],$$

  where Z_t is the normalizer.
- Update $C_t = \left(\frac{1}{n}\sum_{i=1}^{n} \exp[-y_i * F_t(x_i)]\right)^{-1}$

(3) Output the classifier $F(x) = \text{sgn}[\sum_{t=1}^{T} s_t h_t(x)]$.

Hybrid-SABoost. (1) Start with weights $D_t(i) = 1/n$, $i = 1$ to n and $C_t = 1$.
(2) Repeat for $t = 1$ to $[\mu * T]$

- Obtain $h_t(x)$ from weak learner h using weighted training data with respect to D_t.
- Compute $\varepsilon_t = E_{D_t} 1_{[y \neq h_t(x)]}$, $s_t = \log \frac{1-\varepsilon_t}{\varepsilon_t}$.

Table 1
Testing errors of SABoost, AdaBoost with Decision Stumps (DS) as weak base learners

Method	SABoost $\gamma = 1.0$	AdaBoost with DS
Ionosphere	10.11 ± 5.58	10.83 ± 4.96
Bupa	27.14 ± 6.98	28.74 ± 6.62
Pima	24.36 ± 4.37	27.17 ± 4.84
WDBC	4.28 ± 3.02	2.52 ± 2.15
Diabetes	24.75 ± 1.65	24.92 ± 1.63
German	24.38 ± 2.41	24.52 ± 2.08
Heart	19.49 ± 3.78	20.49 ± 3.84
Splice	8.62 ± 0.51	9.14 ± 0.62
Twonorm	12.14 ± 3.03	4.61 ± 0.41
Waveform	13.04 ± 0.83	13.04 ± 0.68
Breast	29.51 ± 4.79	29.58 ± 4.7

There are 50 replications and 1000 iterations for Ionosphere, Bupa liver-disorder, Pima Indian-Diabetes, and WDBC breast-cancer using random sampling. For Breast, Diabetes, German, Heart, Splice Twonorm, and Waveform, we follow the original partitions in IDA Benchmark Repository. Thus, there are 100 replications and 200 iterations for each of them except Splice. There are only 20 combinations in Splice.

- Update $i = 1$ to n,

$$D_{t+1}(i) = \frac{1}{Z_t} D_t(i) \exp[s_t 1_{(y_i \neq h_t(x_i))}],$$

where Z_t is the normalizer.

(3) Repeat for $t = [\mu * T] + 1$ to T.

- Obtain $h_t(x)$ from weak learner h using weighted training data with respect to D_t.
- Compute $\varepsilon_t = E_{D_t} 1_{[y \neq h_t(x)]}$, $s_t = \text{sgn}(0.5 - \varepsilon_t)\gamma_t * C_t$.
- Update $i = 1$ to n,

$$D_{t+1}(i) = \frac{1}{Z_t} D_t(i) \exp[s_t 1_{[y_i \neq h_t(x_i)]}],$$

where Z_t is the normalizer.
- Update $C_t = \left(\frac{1}{n}\sum_{i=1}^{n} \exp(-y_i * F_t(x_i))\right)^{-1}$.

(4) Output the classifier $F(x) = \text{sgn}[\sum_{t=1}^{T} s_t h_t(x)]$.

The purpose of the Hybrid method is to accelerate the convergence speed by replacing part of iteration in the early stages of SAboost by AdaBoost. Nonetheless, as we expected, it also inherits some properties of AdaBoost. Thus we expect it to have performance between the AdaBoost and SABoost. We do test for other μ (say, $\mu = 0.5$), but the difference is rather small for the case of $T = 200$ and is omitted in this report. It may have bigger differences for larger T, but we did not conduct an exhaustive simulation for this point.

Our bench mark data sets are from the following two sources: (1) *UCI machine learning repository* (Blake and Merz, 1998) and (2) *IDA Benchmark Repository*. Table 1 shows testing error rates of SABoost and the AdaBoost with DS as weak base learner. The data sets Ionosphere, Bupa liver-disorder, Pima Indian-diabetes and Wisconsin breast cancer (Wdbc) are from UCI. We conduct 50 replications and 1000 iterations for each of them. In each replication, we randomly sample one-tenth of the data for testing and use the rest of the data for training.

The other data sets in Table 1 are from IDA Benchmark Repository, which have been used in Rätsch et al. (2001) and Mika et al. (1999) (cf. www.boosting.org). There are 100 pre-partitioned training and testing pairs for each of them. (With only one exception—Splice, which has only 20 pre-partitioned combinations.) The results in this portion of Table 1 are obtained under their original training-testing setup. Overall, SABoost (with $\gamma = 1$) delivers comparable or better performances than AdaBoost.

Table 2
SA Boosting with different step sizes

Data set	$\gamma = 0.5$	$\gamma = 1.5$	Hybrid ($\gamma = 0.5$)
Ionosphere	9.94 ± 5.08	11.50 ± 5.47	10.00 ± 4.86
Bupa	26.46 ± 6.65	27.26 ± 6.66	28.51 ± 6.64
Pima	25.76 ± 5.00	26.68 ± 4.11	25.53 ± 4.81
WDBC	4.21 ± 2.77	3.79 ± 2.85	2.45 ± 2.09
Diabetes	24.44 ± 1.74	25.18 ± 1.90	24.61 ± 1.64
German	24.13 ± 2.26	24.81 ± 2.40	23.97 ± 2.31
Heart	17.92 ± 3.89	20.10 ± 3.84	19.75 ± 3.50
Splice	8.96 ± 0.49	8.91 ± 0.56	8.77 ± 0.56
Twonorm	5.67 ± 0.40	13.03 ± 3.46	10.39 ± 2.26
Waveform	12.86 ± 0.54	14.62 ± 1.61	12.89 ± 0.58
Breast	28.58 ± 4.72	29.84 ± 4.62	29.09 ± 4.65

Table 3
The mean and standard deviation of the absolute values of differences between training and testing errors(in percentage) for AdaBoost(DS) and SABoost(DS with $\gamma = 0.5$)

Data set	AdaBoost(DS)	SABoost(DS)
Ionosphere	10.83 ± 4.96	6.82 ± 4.69
Bupa	24.54 ± 7.19	8.49 ± 6.11
Pima	16.52 ± 5.17	6.75 ± 5.33
WDBC	2.52 ± 2.15	3.08 ± 2.14
Diabetes	11.19 ± 2.20	5.27 ± 2.66
German	6.08 ± 2.73	3.66 ± 2.61
Heart	19.98 ± 4.05	7.78 ± 5.03
Splice	4.69 ± 1.05	1.18 ± 0.84
Twonorm	4.61 ± 0.41	5.21 ± 0.55
Waveform	13.04 ± 0.68	8.24 ± 1.39
Breast	8.97 ± 5.08	7.60 ± 4.80

To study the effect of step size, we also try different γ's with SABoost. Table 2 shows the testing errors of SABoost(DS) with different step sizes, and of the hybrid method with $\mu = 0.25$. Note that the performances of SABoost with $\gamma = 1.5$ are very similar to AdaBoost. The results of Hybrid stay between SABoost with $\gamma = 0.5$ and AdaBoost as expected.

Although the testing errors are almost always the focus in performance evaluation of classifiers, it is important to look at the other properties. For boosting-like algorithms, the phenomenon of overfitting is a long standing issue and of practical implication. For practitioners, the smaller the differences between training and testing error, the better. It will be a waste of time to run an interminable boosting iteration while overfitting occurs. Although the overfitting may not be well defined, here we try to identify it using differences between training and testing errors. Table 3 summarizes the means and variations of the differences between training and testing errors at each iterations for both AdaBoost and SABoost ($\gamma = 0.5$). It is clear to see that the means and variances (standard deviations) of SABoost are almost always smaller than AdaBoost.

Fig. 1 shows both training and testing errors at each iteration (average of 50 replications) of SABoost, Hybrid SABoost and AdaBoost applying to the Bupa data set. Fig. 2 shows the error rate curves of all three procedures. It is clear that the SABoost have the smallest testing error among three methods in most of cases. In addition, it also have the smallest differences between training and testing errors. This is a good property for practitioners if they want to decide when to stop the boosting iteration based on the progress of training error. The results for other data sets are similar and then omitted. These patterns are also observed in experiments using synthesized data (see Table 4).

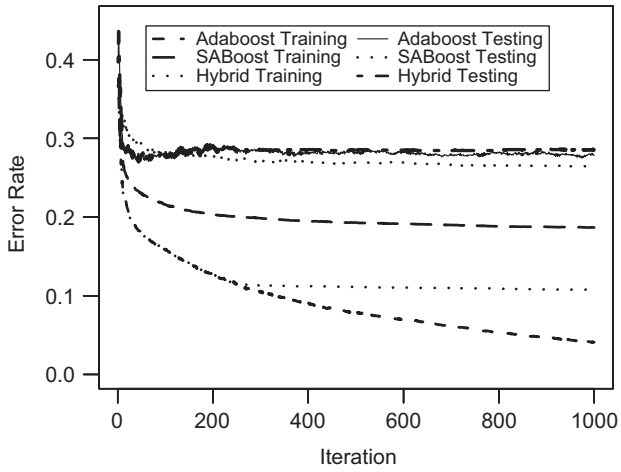

Fig. 1. Bupa liver-disorder: The solid lines are Training and testing error curves of AdaBoost with decision stumps as weak base learner. The dot lines are training and testing errors of SABoost with $\gamma = 0.5$. The dash lines are for hybrid method which first has 250 AdaBoost iterations and then switches to SABoost for the rest of the iterations.

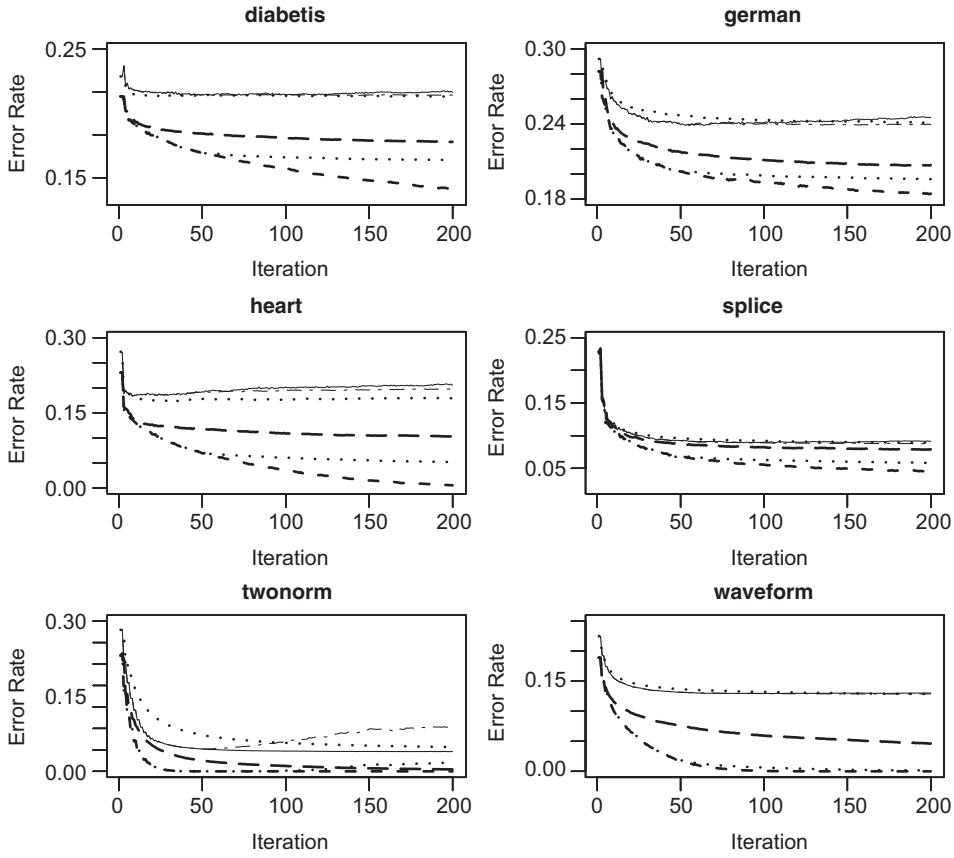

Fig. 2. Error rate curves of Diabetes, German, Heart, Splice, Twonorm and Waveform. All methods here use decision stumps as weak base learners. Please see Fig. 1 for graph legend.

4.2. Experiments with synthesized data

In this simulation study, data are generated from a 5 dimension concentric circles. The sample size is 500 and the ratio of size of positive responses y's to the size of negative y's is approximately equal to 1. Only the first two

Table 4
Error rates (in %) of AdaBoost, SABoost ($\gamma = 0.5, 1$) with different base learner (SVM with Linear Kernel Polynomial Kernel with degree = 3 and coef. = 1 and RBF kernel default parameter)

Base learner	SVM-Linear	
Methods	GE (TE)	$\lvert TE - GE \rvert$
AdaBoost	29.6 (27.3)	2.93 ± 2.27
SABoost $\gamma = 0.5$	32.7 (32.0)	2.30 ± 1.57
SABoost $\gamma = 1$	32.1 (31.9)	2.42 ± 1.63
Base learner	SVM-Polynomial	
AdaBoost	22.02 (20.35)	2.45 ± 1.65
SABoost $\gamma = 0.5$	21.87 (20.87)	2.45 ± 1.57
SABoost $\gamma = 1$	21.95 (20.89)	2.32 ± 1.63
Base learner	SVM-RBF	
AdaBoost	24.67 (18.11)	6.56 ± 2.33
SABoost $\gamma = 0.5$	23.27 (19.76)	3.79 ± 2.34
SABoost $\gamma = 1.0$	24.75 (20.38)	4.59 ± 2.40

Testing error (GE) and Training Error (TE).

dimensions of data are used as feature variables. The "weak learner"s used here are SVM with linear, polynomial (degree 3) and RBF kernels. The number of boosting iterations is 200 with 50 replications for each setup. We use step sizes $\gamma = 0.5$ and 1 for SABoost. The results are summarized in Table 4. We can see that the standard deviation of the differences of absolute value of training and testing errors for SABoost with $\gamma = 0.5$ is the smallest one among all boosting procedures. For reference purposes, the testing errors of SVM with correspondent kernels are also recorded therein. The comparison of SVM using different base learners is to know (empirically) how the choice of the base learners affect the performance of AdaBoost and SABoost. It is observed that SABoost with the SVM-Polynomial and $\gamma = 0.5$ yields the best performance in the sense that it has the smallest testing error and a small difference of training and testing errors. In a way, SVM-Polynomial lies between the SVM-Linear and SVM-RBF. Possibly, it is suitably simple base learner for the synthesized data sets.

5. Conclusion and discussion

In this study, we raise a stochastic approximation interpretation for boosting-like algorithms. This stochastic approximation viewpoint provides some new insights and tools for boosting-like algorithms. It also suggests some more flexible ways to construct new boosting algorithms such as choices of weak base learners with suitable sequences of γ_t's. In addition, an SABoost algorithm is proposed based on the convergence theory of the RM procedure. For suitable choices of step size, SABoost performs similarly or better than the original AdaBoost with the same weak base learner in both synthesized and bench mark data sets.

It should be emphasized that the focus of current paper is on the understanding of boosting rather than proposing a better or best boosting algorithm. There are quite a few variations of boosting-like algorithms improved upon AdaBoost for variety of problems to date. We do not claim that SABoost is the best one. Nonetheless, under the similar boosting scheme, the simple choice of γ_t, for example, γ/t can deliver comparable and sometimes better performance than the one with the original α_t. Our observation suggests that the choice of step size deserves a closer investigation. Interestingly, the condition (9) which leads to γ/t is also noted in Zhang and Yu (2005). In light of recent results, for example, Jiang (2004) and Meir and Rätsch (2003) (Section 2 and references therein) suggest *regularization* is needed for boosting, particularly for noisy data. Regularization, loosely speaking, can be achieved by imposing constraints on the base functions or through early stopping. Note that our choice of $\gamma_t = \gamma/t$ decreases to 0 and thus can be viewed as a smooth stopping rule.

Although fine tuning of γ or μ can improve the performance of SABoost or Hybrid SABoost, it also raises the concern that the improvement is made at the cost of introducing extra parameters. We recognize the possibility. Nonetheless, we emphasize that even SABoost with naive choice of γ, say $\gamma = 1 (\gamma_t = 1/t)$ have comparative performance with AdaBoost in many examples we studied. Our observation suggests that the property of boosting-like algorithm can be

investigated under a simpler, data-independent γ_t contrasting to the original α_t. Our experiments with Hybrid SABoost reveal that the step sizes in early stage of boosting can greatly affect the performances of boosting. Therefore, the boosting iterations should be understood at least from two parts: the early iterations which determines large part of the reduction of TE and GE; the latter iterations should be suitably regularized or early stopped preventing overfitting. Two ensuing questions are under current investigation.

1. How to choose a good γ_t for SABoost for a given base learner (with some distribution information of the data)?
2. What base learners will produce α_t in AdaBoost which are essentially equivalent to γ_t in SABoost (in terms of the convergence rate of step size)?

In addition, the stochastic versions and the KW parallels will be investigated in another paper.

Acknowledgments

The research is supported by NSC-95-2118-M-259-002-MY2, Taiwan. We would also like to thank Lin, Sung-Chiang for his assistance in numerical computation and simulation.

References

Blake, C., Merz, C., 1998. UCI repository of machine learning databases.
Breiman, L., 1999. Predicting games and arcing algorithms. Neural Comput. 11, 1493–1518.
Breiman, L., 2004. Population theory for boosting ensembles. Ann. Statist. 32, 1–11.
Bühlmann, P., Yu, B., 2003. Boosting with the l_2-loss: regression and classification J. Amer. Statist. Assoc. 98 324–339.
Duflo, M., 1997. Random iterative models. Series in Applications of Mathematics. Springer, New York.
Freund, Y., Schapire, R., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. J. of Comput. System Sci. 55, 119–139.
Friedman, J., 2001. Greedy function approximation: a gradient boosting machine. Ann. Statist. 29, 1189–1232.
Friedman, J., 2002. Stochastic gradient boosting. Comput. Statist. Data Anal. 38, 367–378.
Friedman, J., Hastie, T., Tibishirani, R., 2000. Additive logistic regression: a statistical view of boosting (with discussion). Ann. of Statist. 28, 337–407.
Gey, S., Poggi, J.-M., 2006. Boosting and instability for regression trees Comput. Statist. Data Anal. 50, 533–550.
Grove, A., Schuurmans, D., 1998. Boosting in the limit: maximizing the margin of learned ensembles. in: Proceedings of the Fifteenth National Conference on Artificial Intelligence. Madison, Wisconsin.
Jiang, W., 2002. On weak base hypotheses and their implications for boosting regression and classification. Ann. Statist. 30, 51–73.
Jiang, W., 2004. Process consistency for AdaBoost. Ann. Statist. 32, 13–29.
Kiefer, J., Wolfowitz, J., 1952. Stochastic estimation of the maximum of a regression function. Ann. Math. Statist. 23, 462–466.
Kim, Y., Koo, J.-Y., 2005. Inverse boosting for monotone regression functions. Comput. Statist. Data Anal. 49, 757–770.
Kim, Y., Koo, J.-Y., 2006. Erratum to "Inverse boosting for monotone regression functions" Comput. Statist. Data Anal. 50, 583(Comput. Statist. Data Anal. 49 (2005) 757–770).
Lai, T.Z., 2003. Stochastic approximation: invited paper. Ann. Statist. 31, 391–406.
Leitenstorfer, F., Tutz, G., 2007. Knot selection by boosting techniques. Comput. Statist. Data Anal. 51, 4605–4621.
Meir, R., Rätsch, G., 2003. An introduction to boosting and leveraging. In: Mendelson, S., Smola, A. (Eds.), Advanced Lectures on Machine Learning. Springer, New York, pp. 118–183.
Merler, S., Caprile, B., Furlanello, C., 2007. Parallelizing AdaBoost by weights dynamics. Comput. Statist. Data Anal. 51, 2487–2498.
Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Müller, K.-R., 1999. Fisher discriminant analysis with kernels. Neural Networks for Signal Processing, vol. IX. IEEE, New York, pp. 41–48.
Rätsch, G., Onoda, T., Müller, K.-R., 2001. Soft margins for AdaBoost. Machine Learning, 42, 287–320. Also NeuroCOLT Technical Report NC-TR-1998-021.
Robbins, H., Monro, S., 1951. A stochastic approximation method. Ann. Math. Statist. 22, 400–407.
Schapire, R., 1990. The strength of weak learnability. Mach. Learning 5, 197–227.
Schapire, R., Freund, Y., Bartlett, P., Lee, W., 1998. Boosting the margin: a new explanation for the effectiveness of voting methods. Ann. Statist. 26, 1651–1686.
Tutz, G., Binder, H., 2006. Boosting ridge regression. Computational Statistics and Data Analysis. Corrected Proof, Available online 22 December 2006, in press.
Verzani, J., 2005. Using R for Introductory Statistics. Chapman & Hall, CRC Press, London, Boca Raton, FL. ISBN 1-584-88450-9.
Zhang, T., Yu, B., 2005. Boosting with early stopping: convergence and consistency. Ann. Statist. 33, 1538–1579.