A Robust Boosting Algorithm

Richard Nock and Patrice Lefaucheur

Université des Antilles-Guyane
GRIMAAG-Dépt Scientifique Interfacultaire, Campus de Schoelcher
97233 Schoelcher, Martinique, France
{Richard.Nock,Patrice.Lefaucheur}@martinique.univ-ag.fr

Abstract. We describe a new Boosting algorithm which combines the base hypotheses with symmetric functions. Among its properties of practical relevance, the algorithm has significant resistance against noise, and is efficient even in an agnostic learning setting. This last property is ruled out for voting-based Boosting algorithms like ADABOOST. Experiments carried out on thirty domains, most of which readily available, tend to display the reliability of the classifiers built.

1 Introduction and Motivations

Recent advances in Machine Learning (ML) have shown experimentally and theoretically the power of ensemble methods, that is, algorithms combining the predictions of multiple classifiers to make a single classifier [BK99]. Some of the most popular and widely used techniques are Arcing [Bre96b], Bagging [Bre96a], and Boosting [Sch90], in increasing order of the quantity of dedicated works. Arcing and Bagging are voting methods; they differ essentially by a scheme which iteratively modifies the training sample to build the voters. In Bagging [Bre96a], a new sample is generated at each step by bootstrap sampling the initial learning sample. Arcing [Bre96b], on the other side, keeps the initial examples but modifies their weights according to a rule which reweights higher the examples that have been difficult to classify by the voters built so far.
Finally, Boosting is related to a general methodology in which an algorithm, called "strong" learner, requests and combines the output of so-called "weak" learners. The weak and strong adjectives are used advisedly, since the weak hypotheses are only required to perform a little bit better than the unbiased coin (but for any distribution over the learning sample). The strong learner combines them and outputs an hypothesis of arbitrary accuracy provided a sufficiently large number of weak hypotheses have been combined. Boosting draws its roots in the weak and strong learning frameworks [KV89, KV94], and further on the PAC model of Valiant [Val84]. One of the very first argumentation in favor of Boosting is due to [Kea88]. Historically, this paper is most interesting because it proposes, without proofs though, three potential Boosting algorithms, that are all *voting* procedures. The first evidence that Boosting is theoretically viable does not exactly use this combination scheme, but a recursive, decision-tree type majority vote combination [Sch90]. Beyond theory, the first evidences

T. Elomaa et al. (Eds.): ECML, LNAI 2430, pp. 319–331, 2002.

that the practical importance of Boosting is much more than "possible" (quote from [Kea88]) and can actually be of great help to solve challenging problems, culminates in the paper of [FS97] and its algorithm, AdaBoost, and more recently in refined analyzes of AdaBoost [FHT00, SS98]. Interestingly, this approach follows the voting approach proned by [Kea88], but with a powerful reweighting scheme, that Arcing further studied [Bre96b]. This scheme is a stepwise multiplicative update of the training example's weights, so as to bring higher importance to those that have been hard to classify for the last hypothesis.

Most approaches derived from Boosting are voting procedures (see *e.g.* the papers [Bre96b, FS97, SS98]), and more generally many ensemble methods are also voting procedures [Bre96a]. A set of voters is grown, which is a way to cast the initial examples onto a new representation space of different dimension, space into which each hypothesis built defines a new variable. Afterwards, a linear separator on this new set of variables is used to classify observations. Linear separators have certain desirable properties. They have good VC-dimension ($n+1$ for a n-dimensional space), that is, they bring discriminative power at affordable sample complexity for learning [Vap98]. Furthermore, provided the learning sample is separable, *i.e.* provided there exists a theoretical hyperplane separating the examples, they are efficiently learnable [Val84, NG95], which means the existence of polynomial-time, accurate induction algorithms. However, a major drawback is that whenever the dimension is fixed, if no assumption can be made about the separability of the data, then achieving the error of the optimal hyperplane is hard; Even constant factor approximations of this error are intractable [HS92]. This is a drawback that Support Vector Machines avoid by projecting the data into a very high dimensional space in which they are separable [Vap98]. Boosting, however, cannot guarantee such a separability property. Thus, it may face (in)approximability results depending on the nature of the target concept. Because seldom are the real domains for which assumptions can be made on this target concept, one should consider that learning occurs in a relaxed, sort of "agnostic" learning setting. Fortunately, a model of agnostic (or robust) learning, making no assumption on the target concept as well as on the unknown (but fixed) distribution used to generate the examples, has been receiving much attention and an extensive theoretical cover [KSS94]. Obviously, such an ideally relaxed learning setting is also inherently hard, and many results obtained are actually negative, precluding in one sense the use of most interesting classes of concept representations (even simple rules are not agnostically learnable [HS92]). Fortunately, most, but not all.

This paper exhibits the Boosting abilities a class of concept representations among the computationally easiest to manipulate, which allows agnostic learning as well as a handles record noise levels (another crucial issue in Boosting [BK99]): symmetric functions.

Symmetric function are Boolean functions whose outputs are invariant under permutation of the input bits [KL88]. Their discriminative power is the same as linear separators: they have the same VC-dimension. Computationally speaking,

most the learning algorithms require record times or space when compared to many other classes of concept representations [Bou92]. Efficient learning algorithms are known to learn in the PAC or agnostic model, even under malicious noise [Bou92, KL88]. These algorithms have two common-points: first, they are purely theoretical, and studied with absolutely no experiment in the aforementioned papers or books. Second, they follow a similar, simple induction scheme which, informally, proceeds by giving to each of the $n+1$ possible summations the most frequent class in the corresponding subset of the learning sample.
Boosting consists in our case in an algorithm creating a symmetric function of potentially high dimension, by stepwise additions of so-called weak hypotheses whose input is the set of initial variables, and whose output is the set $\{0,1\}$. The set of weak hypotheses defines the new binary observation of each example into this new representation space.
To be more precise, our contribution is twofold. First, we provide the Boosting algorithm and a theoretical study on its capabilities. Among all, we show an interesting relationship with previous works on symmetric functions: Boosting also suggests to build each symmetric function with the robust schemata of [Bou92, KL88]. Armed with an original margin definition adapted to symmetric functions, we prove various results on the algorithm. One is used to establish the theoretical Boosting ability of our algorithm, and makes it, among the quantity of works and algorithms around "Boosting", one of the few proven to have the Boosting property in its original acceptation [Sch90, FS97]. Our algorithm is also the first to be, for any possible set of weak hypotheses, a theoretically efficient robust learner. Such a property is definitely ruled out by [HS92] for the approaches of [SFBL98, SS98] and related. Second, we provide numerous experiments on our algorithm, and compare it with other approaches on thirty domains, most of which are readily available. Dedicated experiments are also devoted to studying noise handling, and a criterion to stop Boosting.
The following section presents our Boosting algorithm, SFboost. The two next sections study and discuss SFboost respectively from a theoretical and an experimental point of view.

2 From Boosting to SFboost

Due to the space constraints, we shall assume basic knowledge of Boosting algorithms, and in particular of their main representative: AdaBoost. All this basic knowledge is included in the clever paper of [SS98], for example. Let us give some general definitions. We let $LS = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$ denote a set of $|LS| = m$ training examples, where $|.|$ denotes the cardinal. Here, each instance x_i belongs to a domain $\mathcal{X}$, and is described using n variables. Each y_i is a class, belonging to a set $\{-1, +1\}$, where -1 (resp. $+1$) is called the negative (resp. positive) class. Sometimes, the classes shall also be noted "$-$" and "$+$" respectively. This paper is mainly focused on binary classification problems, but multiclass classification problems can be handled with the AdaBoost technique, by making from a c-class classification problem c binary

problems, discriminating between one class and all others. They can also be handled with symmetric functions, as these classifiers are naturally multi-class, which is not the case for linear separators. Note that we do not require that examples be initially described using binary variables. Only the weak hypotheses shall be required to have their output in $\{0,1\}$. Such binary output hypotheses can be decision trees, decision lists, monomials (simple rules), etc.

Algorithm 1: SFBOOST (LS)

Input: a learning sample $LS = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$
for $i = 1$ ***to*** m **do** $D_0(i) = 1/m$;
$\boldsymbol{v}_0[0] = \frac{1}{2} \ln \frac{D^+_{0,0}}{D^-_{0,0}}$;
$Z_0 = \sum_{i=1}^m D_0(i) \exp(-y_i H_0(x_i))$;
for $i = 1$ ***to*** m **do** $D_1(i) = \frac{D_0(i) \exp(-y_i H_0(x_i))}{Z_0}$;
for $t = 1$ ***to*** T **do**
  $h_t = \mathit{Weak\_learn}(LS, D_t)$;
  for $j = 0$ ***to*** t **do**

$$\boldsymbol{v}_t[j] = \begin{cases} \boldsymbol{v}_{t-1}[0] + \frac{1}{2} \ln \frac{D^+_{(t-1,0)0}}{D^-_{(t-1,0)0}} & \text{iff } j = 0, \\ \boldsymbol{v}_{t-1}[t-1] + \frac{1}{2} \ln \frac{D^+_{(t-1,t-1)1}}{D^-_{(t-1,t-1)1}} & \text{iff } j = t, \\ \frac{1}{2} \ln \frac{D^+_{(t-1,j-1)1} \exp(\boldsymbol{v}_{t-1}[j-1]) + D^+_{(t-1,j)0} \exp(\boldsymbol{v}_{t-1}[j])}{D^-_{(t-1,j-1)1} \exp(-\boldsymbol{v}_{t-1}[j-1]) + D^-_{(t-1,j)0} \exp(-\boldsymbol{v}_{t-1}[j])} & \text{otherwise} \end{cases} \quad (1)$$

  $Z_t = \sum_{i=1}^m D_t(i) \exp\left(-y_i \left(H_t(x_i) - H_{t-1}(x_i)\right)\right)$;
  for $i = 1$ ***to*** m **do** $D_{t+1}(i) = \frac{D_t(i) \exp\left(-y_i \left(H_t(x_i) - H_{t-1}(x_i)\right)\right)}{Z_t}$;
Output: $H_T(x) = \boldsymbol{v}_T \left[\sum_{t=1}^T h_t(x_i)\right]$

We consider that a symmetric function is a function $H : \{0,1\}^n \to \mathbb{R}$ which is invariant under permutation of its input. Note that in a two-class framework, symmetric functions have their output generally restricted to $\{0,1\}$ [KL88]. We prefer to adopt our slightly more general definition which casts its output in $\mathbb{R}$, so as to make the output give both a label (its sign) and a confidence (its absolute value), thereby rejoining the convention of [SS98]. Suppose we have T weak hypotheses, $h_1, h_2, ..., h_T$. Building a symmetric function H_T using this intermediate set of hypotheses is actually building a symmetric function over the transformed set of examples $\{(x'_1, y_1), (x'_2, y_2), ..., (x'_m, y_m)\}$, with $x'_i = \wedge_{t=1}^T h_t(x_i)$. H_T makes a partition of $\mathcal{X}$ into what we call buckets, the j^{th} bucket ($j = 0, 1, ..., T$) receiving the examples (x_i, y_i) for which $\sum_{t=1}^T h_t(x_i) = j$. The output of H_T can be represented by a $T+1$-dimension "bucket vector" $\boldsymbol{v}_T$, such that $H_T(x_i) = \boldsymbol{v}_T \left[\sum_{t=1}^T h_t(x_i)\right]$.

Suppose H_T is built stepwise for $t = 0, 1, ..., T$, by adding weak hypotheses one at a time, so that at the very beginning, when no such hypothesis exists ($t = 0$), all examples share the same (empty) description and the symmetric function consists of a single bucket. At $t = 1$, we dispose of h_1, and two buckets, receiving examples (x_i, y_i) for which $h_1(x) = 0$, and $h_1(x) = 1$ respectively, and so on for the next steps ($t = 2, ..., T$). Each weak hypothesis is built by a so-called weak learner, *Weak_learn*, which takes as input LS, and a distribution D_t. Note that the distribution takes t as index, which means that the weak learner shall be trained on D_t to output hypothesis h_t. We consider that D_0 is the uniform distribution, where $D_0(i)$, the weight of example $(x_i, y_i) \in LS$, is $1/m$ for all $i = 1, 2, ..., m$. We also adopt the notation $D^b_{t,j}$ ($b \in \{+, -\}$) to denote the sum of the weights of the examples at time t (*i.e.* computed with D_t) falling in bucket j, and belonging to class b (for $0 \leq j \leq t \leq T$). Finally, $D^b_{(t,j)b'}$ ($b' \in \{0, 1\}$) is the sum of weights of examples at time t falling in bucket j, belonging to class b, and for which $h_{t+1}(.) = b'$ (Therefore, we have $D^b_{t,j} = D^b_{(t,j)0} + D^b_{(t,j)1}$). Algorithm 1 presents our approach to Boosting with symmetric functions, called SFboost. Note that it does not unveil how *Weak_learn* works: though any learning algorithm can be used in place of *Weak_learn* as long as the output of its hypotheses is $\{0, 1\}$, the next section presents in particular a criterion that *Weak_learn should* optimize in its synergy with SFboost.

3 Analysis of SFboost

SFboost Repeatedly Levels Out the Distributions. SFboost proceeds by repeatedly leveling out the weights of the classes. Indeed, it is easy to show that after the computation of distribution D_{t+1}, *i.e.* the distribution onto which h_{t+1} shall be built, we have $\forall j = 0, 1, ..., t$, $D^+_{t+1,j} = D^-_{t+1,j}$.

SFboost Is a Boosting Algorithm. For $t = 1, 2, ..., T$, we define a new distribution D'_t such that $\forall (x_i, y_i) \in LS$, we have $D'_0 = D_0$, $D'_1 = D_1$, and $D'_{t>1}(i) = D_t(i) \exp(-y_i H_t(x_i))/Z'_t$ (with Z'_t its normalization coefficient). We call D' the "AdaBoost distribution", as we would have in AdaBoost $D_{t+1} = D'_t$. With D'_t, the algorithm SFboost can be much simplified. For example, it is a simple matter of fact to show that $\boldsymbol{v}_t$ admits the much simpler expression $\boldsymbol{v}_t[j] = \frac{1}{2} \ln(D'^+_{t,j}/D'^-_{t,j})$. $\forall b \in \{+, -\}, j = 0, 1, ..., t$, fix $D'^b_{|t,j} = D'^b_{t,j}/D'_{t,j}$, $D'_{t,j} = D'^+_{t,j} + D'^-_{t,j}$, and $D'^b_t = \sum_{j=0}^t D'^b_{t,j}$. Then Z_t (for $t = 0, 1, ..., T$) can be computed and upperbounded as follows:

$$Z_t = \sum_{j=0}^{t} 2\sqrt{D'^+_{t,j} D'^-_{t,j}} = 2 \sum_{j=0}^{t} D'_{t,j} \sqrt{D'^+_{|t,j}(1 - D'^+_{|t,j})}$$

$$\leq 2 \sqrt{\sum_{j=0}^{t} D'_{t,j} D'^+_{|t,j} \left(1 - \sum_{j=0}^{t} D'_{t,j} D'^+_{|t,j}\right)} = 2\sqrt{D'^+_t (1 - D'^+_t)} \ . \qquad (2)$$

We now show a theorem which displays the ability of H_T to separate the classes in the training sample. In the case of SFBOOST, [SFBL98] show that not only does the error decreases exponentially as t increases, but also the fraction of "risky" examples, close to the frontier. More precisely, [SFBL98] define in the case of a linear separator the following notion of *margin* for an example (x_i, y_i): $\mu(x_i) = \left(y_i \sum_{t=1}^{T} \alpha_t h_t(x)\right) / \sum_{t=1}^{T} \alpha_t$. If this margin is positive, the classifier assigns the right label to the example, and the larger its magnitude, the more confident is the classification given. [SFBL98] have shown that the accuracy of a linear separator depends on the margins over the training sample LS, and one should already strive to maximize them to optimize the quality of the classifier over the whole domain. If the training error rates ϵ_t of each weak hypotheses does not exceed $1/2 - \gamma$ (for any possible D_t), then we have [SFBL98]:

$$\mathbf{Pr}_{LS}[\mu(x_i) \leq \theta] \leq \left(\sqrt{(1-2\gamma)^{1-\theta}(1+2\gamma)^{1+\theta}}\right)^T \quad . \tag{3}$$

The subscript LS in $\mathbf{Pr}$ denotes the probability w. r. t. random uniform choice in LS. Fix $S'^b = \sum_{t=0}^{T} D_t'^b$ $(b \in \{0,1\})$. Fix $V_T = (1/2)|\ln(S'^+/S'^-)|$. V_T quantifies a deviation between the average distributions generated throughout the growth of H_T. This is a separation parameter to which LS contributes: indeed, when there is no weak hypothesis in H_T, a symmetric function can already be constructed, and its accuracy only depends on the balance between the classes in LS. For any example (x_i, y_i), its margin $\mu_{SF}(x_i)$ equals

$$\mu_{SF}(x_i) = \frac{y_i H_T(x_i)}{|V_T|} = \frac{y_i \boldsymbol{v}_T \left[\sum_{t=1}^{T} h_t(x_i)\right]}{|V_T|} \quad . \tag{4}$$

Like $\mu(.)$, if $\mu_{SF}(x_i)$ is positive, then the classifier gives the right class to the example. Furthermore, its magnitude quantifies a relative confidence of bucket $\sum_t h_t(x_i)$ in the last ADABOOST distribution. The larger it is, the more useful is the bucket partitioning generated by H_T (w. r. t. x_i). Armed with our margin definition, we are able to prove the following theorem:

Theorem 1. *Fix* $b_t = \arg\max_{b \in \{+,-\}} D_t'^b$. *We have:*

$$\boldsymbol{Pr}_{LS}[\mu_{SF}(x_i) \leq \theta] \leq 2^{T+1} \prod_{t=0}^{T} \sqrt{(D_t'^{b_t})^{1+\frac{\theta}{T+1}} (1 - D_t'^{b_t})^{1-\frac{\theta}{T+1}}} \quad . \tag{5}$$

Proof sketch: If $\mu_{SF}(x_i) \leq \theta$, then

$$\begin{aligned} y_i H_T(x_i) &\leq \frac{\theta}{2} \ln \frac{\max\{S'^+, S'^-\}}{\min\{S'^+, S'^-\}} \leq \frac{\theta}{2} \ln \frac{\frac{1}{T+1} \sum_{t=0}^{T} D_t'^{b_t}}{1 - \frac{1}{T+1} \sum_{t=0}^{T} D_t'^{b_t}} \\ &\leq \frac{\theta}{2(T+1)} \ln \frac{\prod_{t=0}^{T} D_t'^{b_t}}{\prod_{t=0}^{T} (1 - D_t'^{b_t})} \quad . \end{aligned} \tag{6}$$

(the last ineq. follows from Jensen's inequality). Fix K as the right-hand side parameter in ineq. 6. Then we have $\exp(-y_i H_T(x_i)+K) \geq 1$, and $\mathbf{Pr}_{LS}[\mu_{SF}(x_i) \leq \theta] \leq \mathbf{E}_{LS}\left[\exp(-y_i H_T(x_i) + K)\right]$ ($\mathbf{E}$ denotes the expectation). Remarking that $H_T(x_i) = H_0(x_i) + \sum_{t=1}^{T}\left(H_t(x_i) - H_{t-1}(x_i)\right)$, we easily obtain by unraveling D_{T+1}:

$$\mathbf{Pr}_{LS}[\mu_{SF}(x_i) \leq \theta] \leq \exp(K) \prod_{t=0}^{T} Z_t \sum_{i=1}^{m} D_{T+1}(i) \ . \tag{7}$$

Plugging in the upperbound of each Z_t in ineq. 2 and the expression of K, we obtain the statement of the theorem. □

Theorem 1 has two incidences. First, ineq. 7 shows that at each round, one should strive to select the weak hypothesis h_t which minimizes the corresponding Z_t. Second, let us consider that the training error rate ϵ_t of each h_t on its associated AdaBoost distribution is no more than $1/2 - \gamma$ for some constant $\gamma > 0$, *i.e.* h_t performs only slightly better than random. Then, theorem 1 says that the fraction of examples having margin upperbounded by θ decreases as:

$$\mathbf{Pr}_{LS}[\mu_{SF}(x_i) \leq \theta] \leq \left(\sqrt{(1+2\gamma)^{1+\frac{\theta}{T+1}}(1-2\gamma)^{1-\frac{\theta}{T+1}}}\right)^{T+1} \ . \tag{8}$$

If $\theta < \gamma(T+1)$, then the right-hand side of ineq. 8 decreases exponentially with T. This result is stronger than the one we actually need to bring the Boosting ability of SFboost. Indeed, the Occam's razor argument of [Fre95] (section 3.2) can be used to show that if the weak learner is such that it returns with probability $1-\delta_0$ a hypothesis whose error is no more than $\epsilon_0 = 1/2 - \gamma < 1/2$, then SFboost returns with high probability $1-\delta$ ($\forall \delta > 0$) a symmetric function whose error is no more than ϵ ($\forall \epsilon > 0$), after a reasonable number of rounds (T), and provided $|LS|$ is large enough. The sample size is lowerbounded by a quantity almost linear in $(1/\epsilon)(\ln(1/\delta) + \gamma^{-2}\ln Q)$, where Q is the quantity of weak hypotheses available to the weak learner at each call. For classes used in practice with a reasonable exponential cardinal (depth-bounded decision trees, monomials, etc.), our lowerbound can be very small, and make SFboost an efficient *Boosting* algorithm in the sense of [Sch90]. For infinite-cardinality classes, a more complicated argument is needed which integrates the VC dimension. The emphasis on the Boosting ability of SFboost is important, as throughout the literature, a rapidly increasing number of so-called "boosting" algorithms have been developed. However, with respect to the original theory of [KV89, Sch90], only a few of them (such as [Fre95, FS97, Sch90, SS98]) are really Boosting algorithms.

SFboost Is an Agnostic/Robust Learning Algorithm. The agnostic learning model of [KSS94] (cast in approximation complexity on the form of a robust learning model [HS92]) is a relaxed variant of the PAC learning model [Val84], which virtually makes no assumption about the target concept used to label the

examples. In this model, the learner draws examples according to a fixed but unknown distribution $\mathcal{D}$, and is given two parameters $\epsilon, \delta > 0$. In time polynomial in $1/\epsilon, 1/\delta, n$, the learner has to return a hypothesis from its class of concept representation (*e.g.* bounded-depth decision trees), such that with probability $> 1 - \delta$, the error of this hypothesis is no more than the best achievable in the class plus ϵ. If we look further into the formula computing $\boldsymbol{v}_T$, it admits another expression which is actually

$$\boldsymbol{v}_T[j] = \frac{1}{2} \ln \frac{|\{(x_i, +) \in LS : \sum_{t=1}^{T} h_t(x_i) = j\}|}{|\{(x_i, -) \in LS : \sum_{t=1}^{T} h_t(x_i) = j\}|} \ .$$

Our way to compute the class associated to the buckets is the same as a well known agnostic learning algorithm for symmetric functions [Bou92] with record complexity. Therefore, for each possible t, SFBOOST agnostically learns the target concept. To our knowledge, SFBOOST is the first Boosting algorithm which is also an agnostic/robust learning algorithm. As linear separators are not robustly learnable [HS92] (modulo adequate complexity hypotheses), such a property is definitely out of reach for ADABOOST and all its related algorithms.

SFboost Has Optimal Malicious-Noise Tolerance. [KL88] have studied the learnability of concepts when data can be corrupted by errors from whom absolutely no assumption can be made. Their "malicious noise" model takes place in the same setting as the PAC learning model, but with an adversary which manipulates any requested example with probability β, to return something from which nothing can be assumed. This adversary has unbounded computational resources, knows everything about the task (the target concept to be learned, the distribution $\mathcal{D}$), *and* knows the internal state of the learner. [KL88] show that the maximal amount of such malicious noise is $\Omega(\epsilon)$, where ϵ is the error parameter of the PAC-learning model (see before, or [Val84]). They also show that the class of symmetric functions admit an algorithm which does not only tolerate this optimal bound, but also with a minimal sample complexity. It turns out that at each time t, the symmetric function SFBOOST builds is the same as the one which would be chosen in theorem 11 of [KL88]. To the best of our knowledge, no other Boosting algorithm is known to bring such a noise resistance. Note that noise handling is one of the main problems of Boosting algorithms [BK99].

4 Experiments

Numerical Problems. As SFBOOST proceeds by repeatedly splitting the training sample into subsamples, there may be some problems to compute the components of $\boldsymbol{v}_T$ (eq. 1) whenever the weight of one class approaches zero, or equals zero in a bucket, which in turn would severely bias the update of distributions D_t and D'_t. To avoid such situations, we have chosen to follow experimentally the setup proned by [SS98], which boils down to replacing eq. 1 by what follows: $\boldsymbol{v}_t[0] = \frac{1}{2} \ln \frac{D^+_{(t-1,0)0} \exp(\boldsymbol{v}_{t-1}[0]) + \epsilon}{D^-_{(t-1,0)0} \exp(-\boldsymbol{v}_{t-1}[0]) + \epsilon}$, $\boldsymbol{v}_t[t] = \frac{1}{2} \ln \frac{D^+_{(t-1,t-1)1} \exp(\boldsymbol{v}_{t-1}[t-1]) + \epsilon}{D^-_{(t-1,t-1)1} \exp(-\boldsymbol{v}_{t-1}[t-1]) + \epsilon}$, and

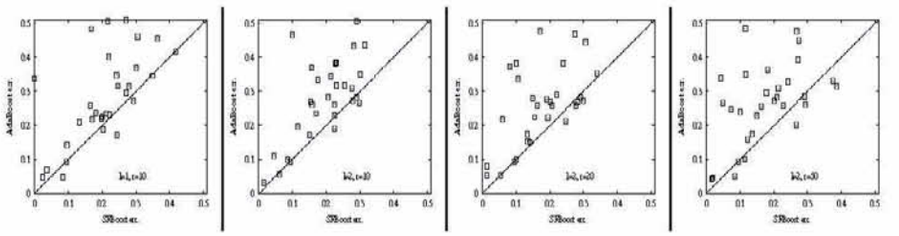

Fig. 1. Scatterplots of the errors of SFBOOST (x) vs ADABOOST (y). Points above the $y = x$ line indicate datasets for which SFBOOST performs better. See text for details

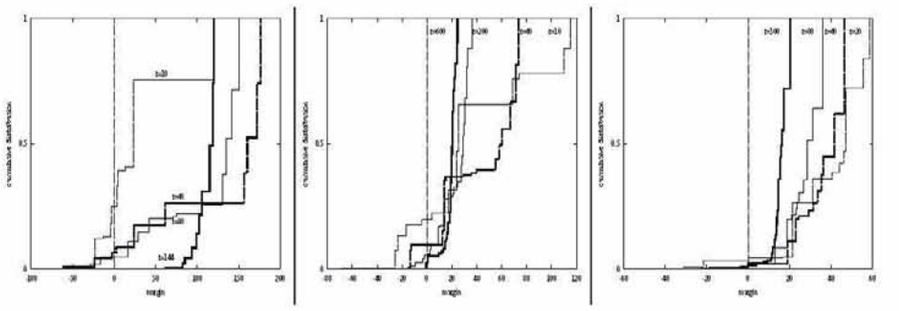

Fig. 2. Cumulative distributions of the margin $\mu_{SF}(.)$ as in eq. 4, for three problems. The "$t = x$" values show the respective margin distribution curves when the symmetric function contains $r = x$ rules

$\boldsymbol{v}_t[j] = \frac{1}{2} \ln \frac{D^+_{(t-1,j-1)1} \exp(v_{t-1}[j-1]) + D^+_{(t-1,j)0} \exp(v_{t-1}[j]) + \epsilon}{D^-_{(t-1,j-1)1} \exp(-v_{t-1}[j-1]) + D^-_{(t-1,j)0} \exp(-v_{t-1}[j]) + \epsilon}$ otherwise ($0 < j < t$). We also fix $\epsilon = 1/m$, as proposed by [SS98].

SFboost vs. AdaBoost. In this experiments, we have chosen to test the behavior of SFBOOST against its principal opponent: ADABOOST. Each algorithm was tested on a set of 30 problems, most of which come from the UCI repository of ML database [BKM98]. The error is evaluated by averaging over a ten-fold stratified cross validation procedure [Qui96]. Finally, on each couple (training set, test set) generated, both algorithms SFBOOST and ADABOOST are ran.
For the sake of comparison, we have chosen for the weak learners a simple class of concept representations: monomials (rules) with a maximal number of literals $\leq l$ for some $l > 0$. Note that whenever $l = 1$ we induce decision stumps [SFBL98]. Each algorithm is ran with a fixed value for l, and requests a number of rules equal to r, for some $r > 0$. As suggested by theory, the weak learners are de-

signed to optimize respectively Z_t (ineq. 2 for SFBOOST) and the Z of ADABOOST (section 3 in [SS98] for ADABOOST). The weak learners are also stepwise greedy optimization procedures for their corresponding Z criterion, building each monomial from scratch. Figure 1 summarizes the results obtained over each of the 30 datasets, for couples of values $(l, r) \in \{(1, 10), (2, 10), (2, 20), (2, 50)\}$. They clearly depicts the ability of SFBOOST to beat ADABOOST on many of the datasets. We have also observed that, as r increases for fixed l, the gap between SFBOOST and ADABOOST tends to increase, but with the same best algorithm: domains for which SFBOOST performs better than ADABOOST at fixed r, l tend to be domains for which SFBOOST shall perform even better when increasing r, and reciprocally for ADABOOST. We emphasize the fact that our choice to use monomials was simply for implementation purposes. Only theory, and the weak learning assumptions of ADABOOST or SFBOOST, could guide reliably through the choice of a more or less complicated class of concept representation to address a domain. Unfortunately, nothing can *a priori* states on an arbitrary domain that some algorithm satisfies the weak learning hypothesis better than another one. So far, only an induction scheme has seemingly brought an experimental accurate answer to the building of weak hypotheses, and has been supported by theoretical comparison studies [KM96]. This scheme has previously been successful to build formulas such as decision trees [Qui94], decision lists [NJ98], and, of course, our simple rules in our experiments with SFBOOST and ADABOOST. Figure 2 presents the margin distribution for SFBOOST over one run, for three problems of the UCI (Monks 1, 2, 3) [BKM98] over which we ran SFBOOST for a maximum of $r = 800$ iterations (with $l = 3$). They display nicely the decreasing of the training error. They also display the decreasing of the maximal margin with r, but the fractions of examples whose margin is no more than reasonable positive thresholds *also* decreases, which accounts for a concentration of the examples near positive, reasonably large margins.

Noise Handling. Usual boosting algorithms are well known to be sensitive to noise [BK99]. In the case of SFBOOST, theory suggests that the algorithm should handle reasonable noise, and be at least as good as ADABOOST, if not better. On 28 out of the 30 problems (for lisibility purposes), we have ran SFBOOST and ADABOOST again with $(l = 4, r = 20)$, either on the original datasets, or when plugging 10% class noise on the examples. Figure 3 (left table) shows the results obtained. The plot for each dataset gives three indications: the error comparison before noise, that after, and which algorithm is the most resistant to noise addition (if the slope is > 1, it is SFBOOST). There are two parts on the plot: datasets plotted before noise with an (x, y) such that approximately $x, y \leq .3$, and the others. The second set contains problems that were so "hard" to handle without noise that noise addition sometimes even reduces the errors. A more reliable study can be carried out with the first set of problems. In that set, out of 17 problems, only 3 are problems for which the segment slope is < 1. In other words, there are 14 problems on which SFBOOST is more resistant to noise addition. A simple sign test reveals a $p = 0.00636$ threshold probability to reject the

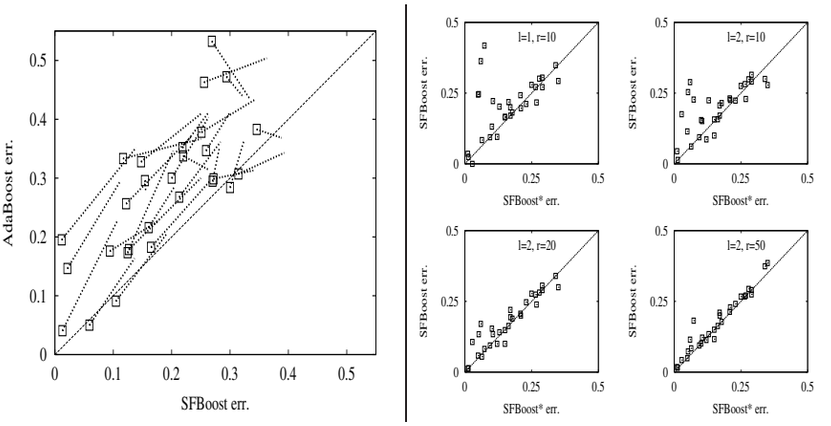

Fig. 3. *Left table*: scatterplots for the errors of SFBOOST (solid line) vs ADABOOST (dashed line) on 28 out of the 30 datasets, with and without 10% class noise ($l = 4, r = 20$). The squares depicts the errors without noise; dashed lines link them with the errors on their corresponding noisy dataset. *Right table*: error scatterplots of SFBOOST* (x) vs SFBOOST (y) for the 30 datasets with $(l, r) = (1, 10), (2, 10), (2, 20)$ and $(2, 50)$ for SFBOOST (see text for details)

hypothesis for an identical behavior against noise. Therefore, SFBOOST seems to handle noise in our experiments in a better way than ADABOOST does.

Stopping Boosting. There is a lack of criteria to choose the T parameter for Boosting. In the case of SFBOOST, we have tried a very simple alternative, suggested by ineq. 7. When putting $\theta = K = 0$, ineq. 7 shows that the training error is upperbounded by $P = \prod_{t=0}^{T} Z_t$. But each Z_t can sometimes be > 1, on hard enough domains. This suggests that, out of a classifier containing T weak hypotheses $h_1, h_2, ..., h_T$, one could choose the one containing $h_1, h_2, ..., h_{T^* \leq T}$ which minimizes P, out of the $T+1$ possible subclassifiers (with the empty one). This is a simple, yet reasonable test to carry out. Figure 3 (right table) reports the results of SFBOOST against this variant called SFBOOST* on the 30 datasets, where H_{T^*} is built after $T = 50$ iterations. SFBOOST* beats SFBOOST on most of the datasets, even when the points gather around the $y = x$ line as r increases in SFBOOST : for $r = 50$, SFBOOST* still beats SFBOOST on 21 datasets, and is beaten only on 3 of them.

References

[BK99] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning Journal*, 36:105–139, 1999. 319, 320, 326, 328

[BKM98] C. L. Blake, E. Keogh, and C. J. Merz. UCI repository of machine learning databases, 1998. `http://www.ics.uci.edu/~mlearn/MLRepository.html`. 327, 328

[Bou92] S. Boucheron. *Théorie de l'apprentissage, de l'approche formelle aux enjeux cognitifs*. Hermes, 1992. 321, 326

[Bre96a] L. Breiman. Bagging predictors. *Machine Learning Journal*, 24:123–140, 1996. 319, 320

[Bre96b] L. Breiman. Bias, Variance and Arcing classifiers. Technical Report 460, UC Berkeley, 1996. 319, 320

[FHT00] J. Friedman, T. Hastie, and R. Tibshirani. Additive Logistic Regression: A Statistical View of Boosting. *Annals of Statistics*, 28:337–374, 2000. 320

[Fre95] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121:256–285, 1995. 325

[FS97] Y. Freund and R. E. Schapire. A Decision-Theoretic generalization of on-line learning and an application to Boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997. 320, 321, 325

[HS92] K-U. Höffgen and H. U. Simon. Robust trainability of single neurons. In *Proceedings of the 5 th International Conference on Computational Learning Theory*, 1992. 320, 321, 325, 326

[Kea88] M. J. Kearns. Thoughts on Hypothesis Boosting, 1988. ML class project. 319, 320

[KL88] M. J. Kearns and M. Li. Learning in the presence of malicious errors. In *Proceedings of the 20 th ACM Symposium on the Theory of Computing*, pages 267–280, 1988. 320, 321, 322, 326

[KM96] M. J. Kearns and Y. Mansour. On the boosting ability of top-down decision tree learning algorithms. *Proceedings of the* 28^{th} *Annual ACM Symposium on the Theory of Computing*, pages 459–468, 1996. 328

[KSS94] M. J. Kearns, R. E. Schapire, and L. M. Sellie. Toward efficient agnostic learning. *Machine Learning Journal*, 17:115–141, 1994. 320, 325

[KV89] M. J. Kearns and L. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Proceedings of the 21 th ACM Symposium on the Theory of Computing*, pages 433–444, 1989. 319, 325

[KV94] M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. M. I. T. Press, 1994. 319

[NG95] R. Nock and O. Gascuel. On learning decision committees. In *Proceedings of the 12 th International Conference on Machine Learning*, pages 413–420, 1995. 320

[NJ98] R. Nock and P. Jappy. On the power of decision lists. In *Proceedings of the 15 th International Conference on Machine Learning*, pages 413–420, 1998. 328

[Qui94] J. R. Quinlan. *C4.5 : programs for machine learning*. Morgan Kaufmann, 1994. 328

[Qui96] J. R. Quinlan. Bagging, Boosting and C4.5. In *Proceedings of AAAI'96*, pages 725–730, 1996. 327

[Sch90] R. E. Schapire. The strength of weak learnability. *Machine Learning Journal*, pages 197–227, 1990. 319, 321, 325

[SFBL98] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the Margin: A new explanation for the effectiveness of Voting methods. *Annals of statistics*, 26:1651–1686, 1998. 321, 324, 327

[SS98] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Proceedings of the 11 th International Conference on Computational Learning Theory*, pages 80–91, 1998. 320, 321, 322, 325, 326, 327, 328

[Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984. 319, 320, 325, 326

[Vap98] V. Vapnik. *Statistical Learning Theory*. John Wiley, 1998. 320