Variational Empirical Bernstein Boosting

Pannagadatta Shivaswamy
pannaga@cs.cornell.edu
Department of Computer Science
Cornell University, Ithaca NY

Tony Jebara
jebara@cs.columbia.edu
Department of Computer Science
Columbia University, New York NY

1 Introduction

Many machine learning algorithms implement empirical risk minimization or a regularized variant of it. For example, the popular AdaBoost [1] algorithm minimizes exponential loss on the training examples. Recently, another alternative called sample variance penalization [2] has been proposed. Sample variance penalization is motivated from empirical Bernstein bounds that involve both the empirical mean and the empirical variance of a sample. A new boosting algorithm is proposed in this paper which also leverages the empirical Bernstein inequality. The algorithm minimizes a weighted combination of the empirical mean and the empirical variance of the risk. The algorithm proposed in this work does not require exhaustive enumeration of the weak learners (unlike an earlier algorithm by [3]).

It is assumed that a training set $(X_i, y_i)_{i=1}^n$ where $X_i \in \mathbf{R}^m$ and $y_i \in \{\pm 1\}$ is drawn independently and identically distributed (*iid*) from a fixed but unknown distribution $\mathcal{D}$. The goal is to learn a classifier or a function $f : \mathcal{X} \to \{\pm 1\}$ that performs well on test examples drawn from the same distribution $\mathcal{D}$. In the rest of this paper, $G : \mathcal{X} \to \{\pm 1\}$ denotes the so-called weak learner. The notation G^s denotes the weak learner in a particular iteration s. Further, the two indices sets I_s and J_s respectively denote examples that the weak learner G^s correctly classified or misclassified, *i.e.,* $I_s := \{i | G^s(X_i) = y_i\}$ and $J_s := \{j | G^s(X_j) \neq y_j\}$.

2 Algorithm

The proposed algorithm appears in Figure 1. This algorithm has a user specified parameter λ which lies in the range $[0, 1]$. With the exponential loss, it was shown by [3] that sample variance penalization is equivalent to minimizing the following cost,

$$\left(\sum_{i=1}^n e^{-y_i f(X_i)}\right)^2 + \lambda \left(n \sum_{i=1}^n e^{-2y_i f(X_i)} - \left(\sum_{i=1}^n e^{-y_i f(X_i)}\right)^2 \right).$$

The following lemma and the theorems together show that VEB-Boost minimizes the above cost iteratively.

Lemma 1 *The update of α_s in VEB-Boost minimizes the cost*

$$\left(\sum_{i \in I_s} \left(\lambda n w_i^2 + (1-\lambda) w_i\right)\right) e^{-2\alpha} + \left(\sum_{j \in J_s} \left(\lambda n w_j^2 + (1-\lambda) w_j\right)\right) e^{2\alpha}, \quad (1)$$

where $w_i = e^{-y_i \sum_{t=1}^{s-1} \alpha_t G^t(X_i)}$

Theorem 2 *Assume that $0 \leq \lambda \leq 1$ and $\sum_{i=1}^n w_i = 1$ (i.e. normalized weights). Then, the cost in the above lemma (*i.e. (1)*) is a variational upper bound on the cost*

$$\left(\sum_{i \in I_s} w_i e^{-\alpha} + \sum_{j \in J_s} w_j e^{\alpha}\right)^2 + \lambda \left(n \sum_{i \in I_s} w_i^2 e^{-2\alpha} + n \sum_{j \in J_s} w_j^2 e^{2\alpha} - \left(\sum_{i \in I_s} w_i e^{-\alpha} + \sum_{j \in J_s} w_j e^{\alpha}\right)^2 \right).$$

with equality being achieved at $\alpha = 0$.

Require: $(X_i, y_i)_{i=1}^n$, scalar parameter $\lambda \geq 0$, weak learners $\mathcal{H}$
Initialize the weights: $w_i \leftarrow \frac{1}{n}$;
Initialize f to predict 0 on all inputs.
for $s \leftarrow 1$ to S **do**
  $u_i \leftarrow \lambda n w_i^2 + (1-\lambda) w_i$
  Estimate a weak learner $G^s(\cdot)$ from training examples weighted by $(u_i)_{i=1}^n$.
  $\alpha_s = \frac{1}{4} \log\left(\frac{\sum_{j \in J_s} u_j}{\sum_{i \in I_s} u_i}\right)$
  if $\alpha_s < 0$ **then** break **end if**
  $f(\cdot) \leftarrow f(\cdot) + \alpha_s G^s(\cdot)$
  $w_i \leftarrow w_i \exp(-y_i G^s(X_i) \alpha_s)/Z_s$ where Z_s is such that $\sum_{i=1}^n w_i = 1$.
end for

Figure 1: VEB-Boost

	Decision Stumps			CART	
Dataset	AdaBoost	VEB-Boost	EBBoost	AdaBoost	VEB-Boost
image	4.28 ± 0.8	3.75 ± 0.7	4.04 ± 0.8	2.61 ± 0.6	2.36 ± 0.5
musklarge	7.80 ± 1.0	6.82 ± 0.6	6.89 ± 0.6	6.67 ± 0.7	6.47 ± 0.5
nist38	5.68 ± 0.6	5.23 ± 0.5	5.34 ± 0.4	3.30 ± 0.3	3.19 ± 0.3
nist56	3.64 ± 0.5	3.42 ± 0.4	3.38 ± 0.4	2.26 ± 0.3	2.17 ± 0.2
ringnorm	15.05 ± 3.1	12.65 ± 2.8	13.45 ± 2.4	7.65 ± 2.4	7.35 ± 2.4
splice	10.57 ± 1.1	10.71 ± 0.9	10.27 ± 0.9	5.10 ± 0.8	4.43 ± 0.6
twonorm	4.30 ± 0.4	4.08 ± 0.4	4.00 ± 0.2	4.08 ± 0.5	3.86 ± 0.4

Table 1: Results with decision stumps and decision trees.

At each step of boosting, the variational upper bound touches the cost function at its value from the previous iteration. This upper bound is minimized in each iteration by VEB-Boost. Thus, it is guaranteed to monotonically decrease the cost in each step. The variational bound is at its loosest at $\lambda = 0$. Despite this, the following theorem shows that VEB-Boost is still efficient compared to AdaBoost at $\lambda = 0$.

Theorem 3 *Let O_A denote the squared cost obtained by AdaBoost after S iterations. For weak learners with a fixed error rate $\epsilon < 0.5$, VEB-Boost with the setting $\lambda = 0$ attains a cost at least as low as O_A in no more than $2S$ iterations.*

The EBBoost algorithm proposed by [3] suffers from a severe limitation: it requires enumeration and evaluation of every possible weak learner per iteration. Typically, weak learners take weights on the training examples and output a rule such that performance on that weighted set of examples is guaranteed to be better than random guessing. However, with the EBBoost algorithm, the weight on all the misclassified examples is $\sum_{i \in J_s} w_i^2 + \left(\sum_{i \in J_s} w_i\right)^2$ and the weight on correctly classified examples is $\sum_{i \in I_s} w_i^2 + \left(\sum_{i \in I_s} w_i\right)^2$; these aggregate weights on misclassified examples and correctly classified examples do not translate into weights on the individual examples. Thus, it becomes necessary to exhaustively enumerate weak learners with EBBoost.

3 Experiments

The experiment setup was similar to that in [3]. Experiments were conducted with decision stumps and decision trees (CART) as the weak learners. Note that EBBoost could not be run with decision trees as weak learners since it requires an exhaustive enumeration of all possible configurations which is unrealistic. A subset of the results are presented in Table 1. For each dataset, and for each weak learner, the algorithm with the best percentage test error is represented by a dark shaded cell. All lightly shaded entries for that dataset and that weak learner are not significantly different from the minimum error. With decision stumps, both EBBoost and VEB-Boost have comparable performance and significantly outperform AdaBoost. With CART as the weak learner, VEB-Boost is significantly better than AdaBoost.

4 Conclusions

We proposed a new algorithm called VEB-Boost to overcome the limitations of EBBoost. VEB-Boost minimizes a well motivated cost by iterating a variational upper bound. Even when the bound is at its loosest, the number of iterations required by VEB-Boost is a small constant factor more than the number of iterations required by AdaBoost. Experimental results showed that VEB-Boost outperforms AdaBoost in terms of classification accuracy and efficiently extends to any family of weak learners.

References

[1] 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*, vol. 55, no. 1, pp. 119–139, 1997.

[2] A. Maurer and M. Pontil, "Empirical Bernstein bounds and sample variance penalization," in *COLT*, 2009.

[3] P. K. Shivaswamy and T. Jebara, "Empirical bernstein boosting," in *AISTATS*, 2010.

Topic: learning algorithms/learning theory **Preference: oral/poster**