# Robust and Efficient Regularized Boosting Using Total Bregman Divergence [*]

Meizhu Liu and Baba C. Vemuri
CISE, University of Florida
mliu,vemuri@cise.ufl.edu

## Abstract

*Boosting is a well known machine learning technique used to improve the performance of weak learners and has been successfully applied to computer vision, medical image analysis, computational biology and other fields. A critical step in boosting algorithms involves update of the data sample distribution, however, most existing boosting algorithms use updating mechanisms that lead to overfitting and instabilities during evolution of the distribution which in turn results in classification inaccuracies. Regularized boosting has been proposed in literature as a means to overcome these difficulties.*

*In this paper, we propose a novel total Bregman divergence (tBD) regularized LPBoost, termed tBRLPBoost. tBD is a recently proposed divergence in literature, which is statistically robust and we prove that tBRLPBoost requires a constant number of iterations to learn a strong classifier and hence is computationally more efficient compared to other regularized boosting algorithms in literature. Also, unlike other boosting methods that are only effective on a handful of datasets, tBRLPBoost works well on a variety of datasets. We present results of testing our algorithm on many public domain databases along with comparisons to several other state-of-the-art methods. Numerical results depict much improvement in efficiency and accuracy over competing methods.*

## 1. Introduction

Classification is a very important task in numerous areas including but not limited to computer vision, pattern recognition and image processing. Ensemble classifiers have been in vogue for hard classification problems where use of single classifiers have not been very successful. Boosting is such an ensemble classification tool. It generates a strong classifier through a linear convex combination of weak classifiers. A weak classifier is usually only required to be better than random guessing. However, the strong classifier greatly boosts the performance of the weak classifiers and performs quite well on the whole dataset. Therefore, boosting has become a very popular method to improve the accuracy of classification and has been widely used in computer vision, pattern recognition and other areas, e.g., image and object categorization [6, 14, 15], object recognition [2] etc.

The main idea behind the boosting algorithm is that at each iteration, it will learn a weak classifier from the training samples that follow a distribution. The weighted weak classifier is then added to the strong classifier. This weight is typically related to the weak classifier's accuracy. The higher the accuracy, the larger the weight, and vise versa. After the weak classifier is added to the strong classifier, the distribution of the samples is updated. The data is reweighted following the rule that samples which are misclassified tend to gain weight and samples that are classified correctly tend to lose weight. Thus the weight will be very large if the sample has been misclassified by many previously learned weak classifiers, and the weight will be very small if the sample has been classified correctly by many previously learned weak classifiers. Therefore, the future weak classifiers to be learned will be focused more on the samples that many previous weak classifiers misclassified. This method has been proven to be effective in various classification problems and thus motivated a lot of research in the machine learning community. Schapire and Singer [16] proposed AdaBoost minimizing the exponential hinge loss, followed by the inception of LPBoost [4] which maximizes the minimum margin between classes, and others proposed many variations and improvements to AdaBoost, namely TotalBoost, SoftBoost, LogitBoost, GentleBoost [9], and entropy regularized LPBoost.

In this paper, we present a regularized LPBoost – that is based on a recently introduced robust divergence measure – for binary classification that can easily be generalized to the multiclass case. This divergence measure is called total Bregman divergence (tBD) which is based on the orthogonal distance between the convex generating function of the divergence and its tangent approximation at the second argument of the divergence. tBD is naturally robust and leads to efficient algorithms for soft and hard clustering. For more

details, we refer the reader to [17].

Based on earlier work on an upper bound for the number of update iterations in the entropy regularized LPBoost algorithm (ELPBoost), by Warmuth et al. [18], we present a constant bound on the number of iterations for our tBD-regularized boosting algorithm (tBRLPBoost). We also present empirical results that depict the efficiency of our proposed classification algorithm in comparison to ELP-Boost and others.

The rest paper is organized as follows. In Section 2, we briefly review the boosting literature. In Section 3, we introduce our algorithm, the total Bregman divergence regularized LPBoost, dubbed as tBRLPBoost, and investigate its properties, like robustness and efficiency. In Section 4, we investigate the same properties experimentally. We also validate/test our algorithm on a number of datasets and compare the results with state-of-the-art boosting algorithms and other competitive classifiers. Finally, we conclude the paper and discuss possible future work in Section 5.

## 2. Previous work

AdaBoost enjoys the stature of being at the "top" of all adaptive boosting algorithms. AdaBoost performs very well in many cases, however, it does not do well on noisy datasets [5]. BrownBoost [8] however overcomes this noise problem. Because in BrownBoost, it is assumed that only noisy examples will be misclassified frequently, thus the samples that are repeatedly misclassified are assumed to be noisy data samples and removed. BrownBoost works well on a variety of problems, but it does not maximize the margin between different classes which limits its performance in many cases.

LPBoost [4] is a linear programming formulation of the boosting algorithm that maximizes the minimum soft margin between different classes, and performs very well on many datasets. However, it does not provide any iteration upper bound and at times requires linear time ($O(N)$, $N$ is the size of the training dataset) to get a good strong classifier. This can be computationally expensive when the datasets are large. SoftBoost [19] on the other hand can maximize the minimum margin up to $\epsilon$ accuracy, with $O(\log N/\epsilon^2)$ number of iterations. This is a great improvement except it suffers from the common annoying problem of a slow start, which makes SoftBoost rather unsatisfactory. More recently, ELPBoost was introduced in [18], which is a regularized version of LPBoost. It uses relative entropy, the Kullback-Leibler divergence (KL) between the updated distribution and the orginal distribution to regularize the conventional LPBoost. ELPBoost overcomes the slow start issue and it performs as well as LPBoost on many datasets with an iteration bound of $O(\log N/\epsilon^2)$. Nevertheless, it is not robust to noise and $O(\log N/\epsilon^2)$ is still an expensive computational requirement.

To overcome all of the aforementioned problems, we present a robust and efficient boosting algorithm in this paper. We show that it has constant upper bound for the number of iterations required to achieve a good strong classifier. The algorithm uses the total Bregman divergence (tBD) to regularize LPBoost. tBD was recently proposed in [12, 17] and it has three salient features. First, it is invariant to rigid transformations of the coordinate system used to specify the convex generating function. Secondly, it is intrinsically robust to noise and outliers. Finally, the representative of a set of objects (whether they are scalars, vectors or functions) has a closed form expression, which makes it computationally attractive. These properties have been verified in the applications of image clustering, retrieval [12] and medical image segmentation [17]. The tBD $\delta$ associated with a real valued strictly convex and differentiable function $f$ defined on a convex set $\mathcal{X}$ between points $d, \tilde{d} \in \mathcal{X}$ is defined by,

$$\delta_f(d, \tilde{d}) = \frac{f(d) - f(\tilde{d}) - \langle d - \tilde{d}, \nabla f(\tilde{d}) \rangle}{\sqrt{1 + \|\nabla f(\tilde{d})\|^2}}, \quad (1)$$

where, $\langle \cdot, \cdot \rangle$ denotes the standard inner product, $\nabla f(y)$ is the gradient of $f$ at $y$, and $\|\nabla f(y)\|^2 = \langle \nabla f(y), \nabla f(y) \rangle$. tBD is a class of divergences which has the specific form as in (1). In this paper, we will focus on the total Kullback-Leibler (tKL) divergence – which is in the class of tBDs – and use it to regularize the LPBoost. We show that the maximum number of iterations for tBRLPBoost to learn a strong classifier is a constant i.e. the iteration upper bound is independent of the number of the training samples. This makes it easily scalable and computationally efficient in comparison to the existing methods. Also, this algorithm performs well on noisy datasets due to the intrinsic robustness of tBD.

## 3. tBRLPBoost: Total Bregman divergence regularized LPBoost

tBRLPBoost uses tBD to regularize the conventional LPBoost. The purpose of regularization is to make the boosting algorithm converge quickly and smoothly, and increase its robustness to noise and outliers. We now briefly present a mathematical description of the conventional boosting. Given the input $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the domain and $\mathcal{Y}$ is the range, the goal is to learn a function $\bar{H} \colon \mathcal{X} \mapsto \mathcal{Y}$. In the binary classification problem, $\mathcal{X}$ is a set of feature vectors, and $\mathcal{Y} = \{-1, 1\}$. The training samples are $\{(x_n, y_n)\}_{n=1}^N$, $x_n \in \mathcal{X}$ is the feature vector, and $y_n \in \{1, -1\}$ is the class label for $x_n$. Given the training samples, the task of boosting is to learn the strong classifier $H(x) = \mathbf{sign}(\sum_{t=1}^T w_t h_t(x))$ to approximate $\bar{H}$. Here, $h_t$ belongs to the weak classifier space $\mathcal{H}$, and $h_t \colon \mathcal{X} \mapsto \mathbb{R}$. In addition, $h_t$ is learned at the $t^{th}$ iteration with respect to the distribution $\mathbf{d}^{t-1}$. Furthermore, $\mathbf{sign}(h_t(x_n))$ predicts the class

label for $x_n$, and $|h_t(x_n)|$ is the confidence factor in the prediction. $w_t$ is the weight for $h_t$, and $T$ is the number of weak classifiers. Given the distribution of the training samples $\mathbf{d}^{t-1}$, the accuracy of the weak classifier $h_t$ is given by $\rho^t = \sum_{n=1}^{N} d_n^{t-1} \mathbf{sign}(y_n h_t(x_n))$. For ease of exposition and avoid notation clutter, we introduce a vector $\mathbf{u}^t$ and call it the "edge". We let $u_n^t = \mathbf{sign}(h_t(x_n)y_n)$, i.e., $\mathbf{u}^t$ measures the accuracy of the weak classifier $h_t$. $u_n^t \in \{-1, 1\}$, and $\mathbf{u}^t = \mathbf{1}$ implies $h_t$ is a perfect classifier while $\mathbf{u}^t = -\mathbf{1}$ implies $h_t$ is a very poor classifier. Now we can rewrite the accuracy $\rho^t$ of $h_t$ as $\rho^t = \sum_{n=1}^{N} d_n^{t-1} u_n^t = \mathbf{d}^{t-1} \cdot \mathbf{u}^t$.

During boosting, if the samples are linearly separable, we will maximize the hard margin between different classes. The hard margin is the width of the area separating the positive from the negative samples. By maximizing the hard margin, the classification accuracy of the strong classifier on the testing dataset will probably be maximized. The LPBoost formulation to maximize the hard margin at iteration $t$ is given by,

$$\max_{\mathbf{w} \in \Delta_t, \rho} \rho, \text{ s.t. } \sum_{i=1}^{t} u_n^i \omega_i \geq \rho, n = 1, \cdots, N, \qquad (2)$$

where, $\rho$ is the minimum hard margin between the two classes. $\Delta_t$ is the $t$-simplex and $\mathbf{w} \in \Delta_t$ implies that $\sum_{j=1}^{t} w_j = 1$ and $w_j \geq 0$, for $j = 1, \cdots, t$. The effect of constraining $\mathbf{w} \in \Delta_t$ is to prevent $\mathbf{w}$ from scaling.

When the samples can not be linearly separated, we can not find a hard margin that clearly separates the positive from the negative classes. In this case, soft boosting is an alternative choice, where we maximize the soft margin at each iteration and allow samples to fall below the margin. Samples can fall below the margin up to some slack factors, but we can levy a penalty for falling below the margin to make sure the slack factors don't become extremely large. LPBoost to maximize the soft margin can be expressed by the following function,

$$\max_{\mathbf{w} \in \Delta_t, \zeta \geq \mathbf{0}, \rho} \rho - D \sum_{n=1}^{N} \zeta_n$$

$$\text{s.t. } \sum_{i=1}^{t} u_n^i w_i \geq \rho - \zeta_n, n = 1, \cdots, N, \qquad (3)$$

where, $\zeta$ is the slack variable vector, and $D$ is the constant factor which penalizes the slack variables. If $D$ is very large, say $\infty$, then (3) becomes the hard margin maximization problem (2); if $D$ is small enough, then it will always leads to feasible solutions for (3).

The dual problem of (3) at step $t$ minimizes the maximum accuracy resulting from the weak classifiers learned thus far, and is given by,

$$\min_{\mathbf{d} \in \Delta_N, \mathbf{d} \leq D\mathbf{1}} \max_{i=1, \cdots, t} \mathbf{u}^i \cdot \mathbf{d} \qquad (4)$$

Note that adding the weight $D$ to the slack variables of the primal (3) results in $\mathbf{d} \leq D\mathbf{1}$. To make such a $\mathbf{d}$ exist, we require $D \geq 1/N$. It was shown in [18] that $D = 1/s$, and $s \in \{1, \cdots, N\}$ is a favorable choice. LPBoost works well and has been extensively used, however, it converges very slowly, and the number of iterations for the algorithm is at best $O(\log N)$ with a large constant factor. When $N$ is large, it will be computationally expensive to use LPBoost. Also, the evolution of $\mathbf{d}$ might have serious instabilities, which reduces the efficiency significantly and might result in overfitting. To overcome these downsides, we add a regularization term based on the total Kullback-Leibler (tKL) divergence to (4). This regularization makes the evolution of $\mathbf{d}$ smooth, and more interestingly the number of iterations will be reduced to a constant. At the same time, the soft margin can be maximized without loss. The regularized LPBoost (tBRLPBoost) is

$$\min_{\mathbf{d} \in \Delta_N, \mathbf{d} \leq D\mathbf{1}} (\max_{i=1, \cdots, t} \mathbf{u}^i \cdot \mathbf{d} + \lambda \delta(\mathbf{d}, \mathbf{d}^0)) \qquad (5)$$

$\delta(., .)$ is the tKL and

$$\delta(\mathbf{d}, \mathbf{d}^0) = \left( \sum_{n=1}^{N} d_n \log \frac{d_n}{d_n^0} \right) \Big/ \sqrt{1 + \sum_{n=1}^{N} d_n^0 (1 + \log d_n^0)^2} \qquad (6)$$

$\mathbf{d}^0$ is the initialized distribution, which is set to a uniform distribution, i.e., $d_n^0 = 1/N$, $n = 1, \cdots, N$. $\lambda > 0$ is the regularization parameter that controls the smoothness of estimated distribution.

At each iteration, tBRLPBoost computes the weak classifier $h_t$, along with its corresponding weight $w_t$ in building the strong classifier, and based on the performance of the weaker classifier, tBRLPBoost updates $\mathbf{d}^{t-1}$ to $\mathbf{d}^t$. The algorithm for tBRLPBoost is depicted in Algorithm 1.

---
**Algorithm 1** Total Bregman divergence regularized LP-Boost
---
    **Input**: $\{(x_n, y_n)\}_{n=1}^{N}$, $x_n \in \mathcal{X}$, and $y_n \in \{1, -1\}$,
    **Output**: $H(x) = \mathbf{sign}(\sum_{t=1}^{T} w_t h_t(x))$.
    **Initialization:** $d_n^0 = 1/N$, $n = 1, \cdots, N$
    **for** $t = 1$ to $T$ **do**
       {Find the weak classifier}
       $h_t \leftarrow \arg\max_h \sum_{n=1}^{N} d_n^{t-1} \mathbf{sign}(y_n h(x_n))$.
       Update the distribution from $\mathbf{d}^{t-1}$ to $\mathbf{d}^t$ according to
       (5) and the weight $\mathbf{w}$ according to (3)
    **end for**
    Return $H(x) = \mathbf{sign}(\sum_{t=1}^{T} w_t h_t(x))$
---

### 3.1. Computation of $\mathbf{d}^t$ and $\mathbf{w}$

To directly compute $\mathbf{d}^t$ from (5) is complicated, instead, we will first get the Lagrangian dual of (5) and accordingly compute $\mathbf{d}^t$. To find the Lagrangian dual, we rewrite (5) into the following form

$$\min_{\beta, \mathbf{d}} \beta + \lambda \delta(\mathbf{d}, \mathbf{d}^0), \text{ s.t. } \mathbf{d} \in \Delta_N, \mathbf{d} \leq D\mathbf{1},$$

$$\mathbf{u}^i \cdot \mathbf{d} \leq \beta, i = 1, \cdots, t \qquad (7)$$

Now, the Lagrangian $\Phi$ of (5) is easy to see, and given by,

$$\Phi(\mathbf{d}, \beta, \mathbf{w}, \xi, \gamma) = \beta + \lambda\delta(\mathbf{d}, \mathbf{d}^0) + \sum_{i=1}^{t} w_i(\mathbf{u}^i \cdot \mathbf{d} - \beta)$$
$$+ \sum_{n=1}^{N} \xi_n(d_n - D) + \gamma(\mathbf{d} \cdot \mathbf{1} - 1), \quad (8)$$

where, $w_i$ and $\gamma$ are non-negative regularizers. Differentiating $\Phi$ with respect to $\beta$, we get

$$\frac{\partial \Phi}{\partial \beta} = 1 - \sum_{i=1}^{t} w_i = 0, \quad (9)$$

and by enforcing $\sum_{i=1}^{t} w_i = 1$ manually (done by normalizing $\mathbf{w}$), we can eliminate $\beta$ from (8). Also since,

$$\frac{\partial \Phi}{\partial \gamma} = \mathbf{d} \cdot \mathbf{1} - 1 = 0, \quad (10)$$

by enforcing $\mathbf{d} \cdot \mathbf{1} = 1$, we can eliminate $\gamma$ from (8). Moreover, using the KKT condition [3], $\xi_n(d_n - D) = 0$. Therefore, we can simplify (8) and get the partial Lagrangian

$$\Phi(\mathbf{d}, \mathbf{w}) = \lambda\delta(\mathbf{d}, \mathbf{d}^0) + \sum_{i=1}^{t} w_i \mathbf{u}^i \cdot \mathbf{d}. \quad (11)$$

Now differentiating $\Phi$ with respect to $\mathbf{d}$, setting it to 0, and normalizing $\mathbf{d}$, we get,

$$d_n^t = \frac{d_n^0 \exp\left(-c \sum_{i=1}^{t} u_n^i w_i\right)}{Z_t}, \quad (12)$$

where, $c = \frac{1}{\lambda}\sqrt{1 + \sum_{n=1}^{N} d_n^0(1 + \log d_n^0)^2}$, and $Z_t$ is the normalization parameter to make $\sum_{n=1}^{N} d_n^t = 1$.

The weight vector $\mathbf{w}$ for the weak classifiers should satisfy (3), and can be solved using column generation [4] or a gradient based method. We used the former in this paper.

### 3.2. Bounding the number of iterations

Let the tolerance be $\epsilon$, i.e., we allow the maximum soft margin resulting from tBRLPBoost to be different from the optimal soft margin by $\epsilon$. If the regularization parameter $\lambda$ in (5) is set to $\frac{\epsilon\sqrt{1+(\log N-1)^2}}{2\log(ND)}$, then the number of iterations for tBRLPBoost is at most $\mathbf{b} = 32\sqrt{2}/\epsilon^2 + 1$.

**Theorem 3.1.** *Let $\lambda$ in (5) be $\frac{\epsilon\sqrt{1+(\log N-1)^2}}{2\log(ND)}$, then the tBRLPBoost will converge in constant 'b' number of iterations given by $\mathbf{b} = 32\sqrt{2}/\epsilon^2 + 1$.*

*Proof.* Since the initialized distribution $\mathbf{d}^0$ is set to $d_n^0 = 1/N$, $n = 1, \cdots, N$, therefore, $\sqrt{1 + \sum_{n=1}^{N} d_n^0(1 + \log d_n^0)^2} = \sqrt{1 + (\log N - 1)^2}$. Also according to (5), $d_n \leq D$, $n = 1, \cdots, N$, thus

$$\delta(\mathbf{d}, \mathbf{d}^0) \leq \frac{\log(ND)}{\sqrt{1 + (\log N - 1)^2}}. \quad (13)$$

Consequently, if $\lambda = \frac{\epsilon\sqrt{1+(\log N-1)^2}}{2\log(ND)}$, we have $\lambda\delta(\mathbf{d}, \mathbf{d}^0) \leq \frac{\epsilon}{2}$. This means that the optimal margin of the regularized problem (5) is at most $\epsilon$ different from the margin of the unregularized problem (3). Using similar arguments as in [18], we can show the the algorithm converges in

$$T \leq \frac{16}{\lambda\epsilon} + 1 = \frac{32\log(ND)}{\epsilon^2\sqrt{1 + (\log N - 1)^2}} + 1. \quad (14)$$

Since $\sqrt{1 + (\log N - 1)^2} \geq \log N/\sqrt{2}$, $N \in \mathbb{Z}^+$, and $D \leq 1$, thus, setting $D = 1$, we get

$$T \leq 32\sqrt{2}/\epsilon^2 + 1. \quad (15)$$

From (5), $D = 1$ means there is no constraint on the distribution, in other words, $\mathbf{d}$ can get to any point in the $N$-simplex, so the algorithm can achieve the optimum. $\square$

The upper bound for the number of iterations for tBRLPBoost is $32\sqrt{2}/\epsilon^2 + 1$, which is a constant and hence independent of the size of the training dataset. Actually, the number of iterations in real applications can be much smaller than $32\sqrt{2}/\epsilon^2 + 1$. Because $D$ is usually not set to 1 manually, instead, it is learned during the training and is set to be the number that can maximize the classification accuracy on the training dataset. Therefore, $D$ can be much smaller than 1, Hence, the iteration number will be much smaller. Therefore, the algorithm will converge in much fewer iterations. This was verified in our experiments.

### 3.3. Weak classifiers

The type of weak classifiers is very important in determining the accuracy of the final strong classifier. To emphasize the performance strength of tBRLPBoost, we use the simplest type of weak classifier, namely, the decision stump, which is a two level binary decision tree.

## 4. Experiments

We evaluated our algorithm on numerous public domain datasets, including the UCI machine learning repository [7], the OASIS MRI brain database [13], the Epilepsy dataset, the CNS dataset, the Colon tumor dataset and Leukemia cancer dataset [11]. We compared our method with several state-of-the-art boosting techniques.

### 4.1. UCI datasets

The UCI repository [7] is a collection of databases that have been extensively used for analyzing machine learning techniques. The repository contains very noisy data (e.g. waveform) as well as relatively clean data, which is optimal for testing classification algorithms. We selected 13 datasets from the UCI repository. The selected datasets include noisy and clean datasets, cover small size to large size datasets in terms of number of samples in the datasets, and range from low dimension to high dimension in terms of number of attributes per sample of the datasets. We
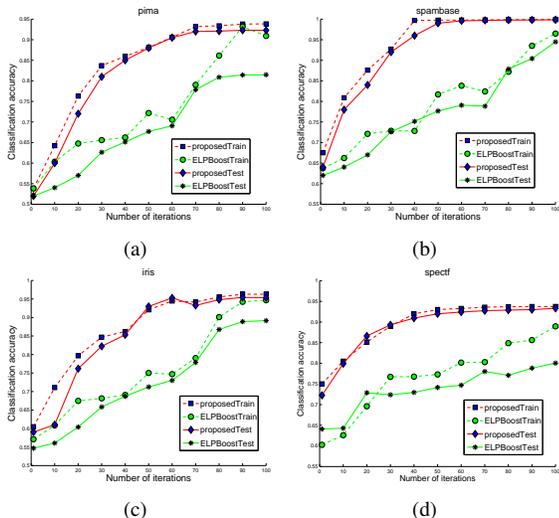
Figure 1. Classification accuracy vs. number of iterations. Comparision of ELPBoost and tBRLPBoost on the training and testing sets of the pima (a), spambase (b), iris (c) and spectf (d) data.

| dataset | AdaBoost | LPBoost | ELPBoost | tBRLPBoost |
|---------|----------|---------|----------|------------|
| Y vs. M | 0.960 | 0.980 | 0.973 | **0.985** |
| M vs. O | 0.962 | 0.972 | 0.974 | **0.996** |
| O vs. Y | 0.987 | 0.988 | 0.988 | **1.000** |
| AD vs. Con. | 0.923 | 0.967 | 0.968 | **0.977** |

Table 2. Classification accuracy on the OASIS dataset.

compared our results with those generated from other techniques in the literature, including AdaBoost, LPBoost, SoftBoost, BrownBoost and ELPBoost. For the implementation of most methods, we used the existing codes on the website (http://sourceforge.net/projects/jboost/, http://www.kyb.mpg.de/bs/people/nowozin/gboost/#pubs), and for ELPBoost, we adapted it from LPBoost (http://www.kyb.mpg.de/bs/people/nowozin/gboost/#pubs). We use the same experimental settings as in [18]. Each dataset is separated into 100 predefined splits as training and testing sets. We use 5-fold cross-validation for each split, and determine the parameters for each of the boosting algorithms during the training and validation period. The results obtained are an average taken over 100 runs of the estimation of the classification accuracy for each algorithm and dataset. The means and standard deviations are reported in Table 1, from which it is evident that tBRLPBoost has up to 13% higher accuracy than other boosting algorithms on various kinds of datasets.

We also report the change of the classification accuracy as a function of the number of iterations on the pima, spambase, iris and spectf datasets, and compare our results with the results from an application of ELPBoost [18]. The results are obtained using 5-fold cross-validation and we show the average classification accuracy vs. the number of iterations in Fig. 4.1. The figure shows that the accuracy of tBRLPBoost increases much faster during the first few iterations and gets to satisfactory accuracy in far less number of iterations in comparison to ELPBoost.

### 4.2. OASIS datasets

We also evaluated our algorithm on the OASIS MRI brain database [13]. The OASIS database contains a cross-

sectional collection of 416 subjects aged 18 to 96. Out of the 416 subjects, 175 subjects are younger than 40 that we designated as young (Y), and 195 subjects are above 60 that we designated as old (O). The other 66 subjects are designated as middle aged (M). Each subject is represented using a 3D histogram describing the non-rigid registration required to co-register an emerging atlas (in a groupwise registration process) to a subject MR brain scan. This groupwise registration was accomplished by using the method described in [10] on the OASIS dataset. The number of bins in each direction was set to $(6 \times 6 \times 6)$ for constructing the histograms of the displacement vectors describing the non-rigid registration. Further, among the old aged people, we took 70 subjects, and 35 of them were diagnosed with very mild to moderate Alzheimer disease (AD) while the rest were controls. We did four groups of experiments on this dataset. First classified the age groups (Y vs. M, M vs. O, and O vs. Y), and then classified the healthy and the very mild to moderate AD patients (AD vs. Control). For each experiment, we use 5-fold cross validation and report the average classification accuracy. The parameters of each of the algorithms are set to be those maximizing the accuracy of the training dataset. We compared our results with those from AdaBoost, LPBoost and ELPBoost. The results are shown in Table 2, from which it is evident that our method (tBRLPBoost) outperforms the competitive methods. Also tBRLPBoost converges much faster than the other algorithms. All algorithms are run on a laptop with Intel(R) Core(TM)2 CPU L7500 @1.6GHz, 4GB memory, GNU Linux and MATLAB (Version 2010a). The average CPU time taken to converge for our algorithm is 0.2778s, while AdaBoost takes 1.8082s, LPBoost takes 2.1430s and ELPBoost takes 2.1164s.

### 4.3. Other datasets

The Epilepsy dataset [11] consists of 3D histograms $(6 \times 6 \times 6)$ of displacement vector fields representing the registration deformation field between the left and right hippocampi in 3D. The goal is to distinguish between left and right anterior temporal lobe (L/RATL) epilepsy. The CNS dataset contains the treatment outcomes of 60 subjects for central nervous system embryonal tumor, which includes 21 survivors and 39 failures. The Colon tumor dataset consists of 22 normal and 40 tumor colon tissue features. The Leukemia cancer dataset contains 25 acute myeloid leukemia (AML) and 47 acute lymphoblas-

| dataset | AdaBoost | LPBoost | SoftBoost | BrownBoost | ELPBoost | tBRLPBoost |
|---|---|---|---|---|---|---|
| breast cancer | $0.702 \pm 0.039$ | $0.734 \pm 0.050$ | $0.721 \pm 0.044$ | $0.699 \pm 0.037$ | $0.728 \pm 0.042$ | $\mathbf{0.864 \pm 0.008}$ |
| diabetes | $0.722 \pm 0.015$ | $0.765 \pm 0.018$ | $0.764 \pm 0.017$ | $0.728 \pm 0.014$ | $0.756 \pm 0.018$ | $\mathbf{0.783 \pm 0.015}$ |
| german credit | $0.730 \pm 0.021$ | $0.754 \pm 0.023$ | $0.752 \pm 0.023$ | $0.752 \pm 0.019$ | $0.758 \pm 0.223$ | $\mathbf{0.805 \pm 0.102}$ |
| heart disease | $0.798 \pm 0.025$ | $0.812 \pm 0.033$ | $0.807 \pm 0.024$ | $0.793 \pm 0.027$ | $0.827 \pm 0.031$ | $\mathbf{0.895 \pm 0.016}$ |
| ionosphere | $0.879 \pm 0.016$ | $0.843 \pm 0.011$ | $0.914 \pm 0.021$ | $0.875 \pm 0.015$ | $0.838 \pm 0.014$ | $\mathbf{0.957 \pm 0.006}$ |
| liver disorders | $0.753 \pm 0.026$ | $0.765 \pm 0.032$ | $0.805 \pm 0.038$ | $0.802 \pm 0.036$ | $0.783 \pm 0.027$ | $\mathbf{0.875 \pm 0.031}$ |
| sonar | $0.861 \pm 0.053$ | $0.857 \pm 0.060$ | $0.872 \pm 0.102$ | $0.861 \pm 0.042$ | $0.873 \pm 0.071$ | $\mathbf{0.902 \pm 0.018}$ |
| pendigits | $0.921 \pm 0.010$ | $0.942 \pm 0.012$ | $0.957 \pm 0.009$ | $0.093 \pm 0.012$ | $0.921 \pm 0.008$ | $\mathbf{0.985 \pm 0.003}$ |
| waveform | $0.892 \pm 0.004$ | $0.899 \pm 0.005$ | $0.901 \pm 0.006$ | $0.900 \pm 0.004$ | $0.895 \pm 0.006$ | $\mathbf{0.923 \pm 0.003}$ |

Table 1. Classification accuracy (mean $\pm$ deviation) for different boosting algorithms.

| dataset | CSC-M | CSC | KFD | SVM | tBRLPBoost |
|---|---|---|---|---|---|
| Epilepsy | 0.932 | 0.886 | 0.864 | 0.864 | **0.941** |
| CNS | 0.700 | 0.733 | 0.650 | 0.683 | **0.757** |
| Colon | 0.871 | 0.871 | 0.758 | 0.823 | **0.892** |
| Leukemia | 0.972 | 0.986 | 0.986 | 0.972 | **0.990** |

Table 3. Classification accuracy for different methods on the Epilepsy, CNS, Colon tumor and Leukemia datasets.

tic leukemia (ALL) samples. We compared our method with the recently published competitive classifiers, including conic section classifier (CSC) [1], CSC with margin pursuit (CSC-M) [11], kernel Fisher Discriminants (KFD), and kernel SVMs (SVM). The results are shown in Table 3, which depicts that our algorithm performs better than the competing classifiers.

## 5. Conclusions

In this paper, we proposed a new boosting algorithm dubbed tBRLPBoost, which is total Bregman divergence (tBD) regularized LPBoost. tBRLPBoost is robust to noise and outliers due to the intrinsic robustness property of tBD. tBRLPBoost is able to maximize the soft margin for linearly inseparable dataset. We showed that tBRLPBoost requires only a constant number of iterations to converge and hence is independent of the size of the training set, which makes it very efficient. In comparison, most efficient boosting algorithm in the published literature cost logarithmic number of iterations. We showed several comparisons to other existing boosting methods and depicted better performance of our algorithm in comparison to the state-of-the-art methods in literature. Even though in this paper we focus on tKL regularized LPBoost, we can easily extend this and use other classes of tBD to regularize the LPBoost. Further, tBRLPBoost can be very easily generalized to multiclass input as well to the confidence-related input, where the label is not binary but takes values in the interval $[-1, 1]$.

## References

[1] A. Banerjee and et al. A conic section classifier and its application to image datasets. *IEEE CVPR*, 2006. 6

[2] A. Bar-Hillel, T. Hertz, and D. Weinshall. Object class recognition by boosting a part based model. *IEEE CVPR*, pages 702–709, 2005. 1

[3] S. Boyd and L. Vandenberghe. Convex optimization. *Cambridge University Press, Cambridge, England*, 2004. 4

[4] A. Demiriz and et al. Linear programming boosting via column generation. *Mach. Learn.*, 2002. 1, 2, 4

[5] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Mach. Learn.*, 2000. 2

[6] P. Dollar, Z. Tu, H. Tao, and S. Belongie. Feature mining for image classification. *IEEE CVPR*, pages 1–8, 2007. 1

[7] A. Frank and A. Asuncion. UCI machine learning repository, 2010. 4

[8] Y. Freund. An adaptive version of the boost by majority algorithm. *Mach. Learn.*, 43:293–318, 2001. 2

[9] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Ann. Statist.*, pages 337–374, 2000. 1

[10] S. Joshi and et al. Unbiased diffeomorphic atlas construction for computational anatomy. *NeuroImage*, 2004. 5

[11] S. Kodipaka and et al. Large margin pursuit for a conic section classifier. *IEEE CVPR*, 2008. 4, 5, 6

[12] M. Liu and et al. Total Bregman divergence and its applications to shape retrieval. *IEEE CVPR*, 2010. 2

[13] D. S. Marcus and et al. Open Access Series of Imaging Studies (OASIS): Cross-Sectional MRI Data in Young, Middle Aged, Nondemented, and Demented Older Adults. *J. Cogn. Neurosci.*, 19:1498–1507, 2007. 4, 5

[14] X. Perrotton and et al. Implicit hierarchical boosting for multi-view object detection. *IEEE CVPR*, 2010. 1

[15] A. Saffari, C. Leistner, and H. Bischof. Regularized multiclass semi-supervised boosting. *IEEE CVPR*, 2009. 1

[16] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.*, 37(3):297–336, 1999. 1

[17] B. C. Vemuri and et al. Total Bregman divergence and its applications to DTI analysis. *IEEE TMI*, 2010. 2

[18] M. K. Warmuth and et al. Entropy regularized LPBoost. *Int. Conf. Alg. Learn. Theory*, 2008. 2, 3, 4, 5

[19] M. K. Warmuth, K. A. Glocer, and G. Raetsch. Boosting algorithms for maximizing the soft margin. *NIPS*, 2007. 2