A Simple Multi-Class Boosting Framework with Theoretical Guarantees and Empirical Proficiency

Ron Appel [1] **Pietro Perona** [1]

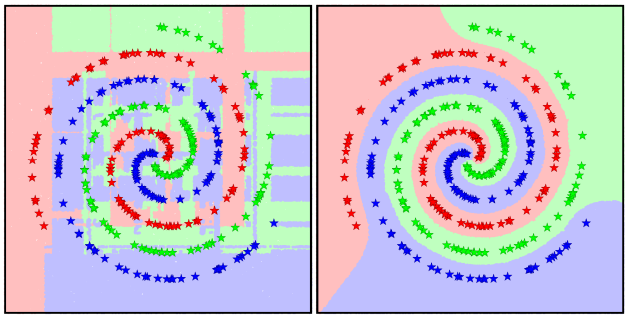
Abstract

There is a need for simple yet accurate white-box learning systems that train quickly and with little data. To this end, we showcase REBEL, a multi-class boosting method, and present a novel family of weak learners called localized similarities. Our framework provably minimizes the training error of any dataset at an exponential rate. We carry out experiments on a variety of synthetic and real datasets, demonstrating a consistent tendency to avoid overfitting. We evaluate our method on MNIST and standard UCI datasets against other state-of-the-art methods, showing the empirical proficiency of our method.

1. Motivation

The past couple of years have seen vast improvements in the performance of machine learning algorithms. Deep Nets of varying architectures reach almost *(if not better than)* human performance in many domains (LeCun et al., 2015). A key strength of these systems is their ability to transform the data using complex feature representations to facilitate classification. However, there are several considerable drawbacks to employing such networks.

A first drawback is that validating through many architectures, each of which may have millions of parameters, requires a lot of data and time. In many fields (e.g. pathology of not-so-common diseases, expert curation of esoteric subjects, etc.), gathering large amounts of data is expensive or even impossible (Yu et al., 2015). Autonomous robots that need to learn on the fly may not be able to afford the large amount of processing power or time required to properly train more complex networks simply due to their hardware constraints. Moreover, most potential users (e.g. non-machine-learning scientists, small business owners, hobbyists, etc.) may not have the expertise or artistry required to hypothesize a set of appropriate models.



(a) Old: Decision Stumps (b) New: Localized Similarities

Figure 1. (a) The typical decision stumps commonly used in boosting lead to classification boundaries that are axis aligned and not representative of the data. Although these methods can achieve perfect training accuracy, it is apparent that they heavily overfit. (b) Our method uses *localized similarities*, a novel family of simple weak learners (see Sec. 5.1). Paired with a procedure that provably guarantees exponential loss minimization, our classifiers focus on smooth, well-generalizing boundaries.

A second drawback is that the complex representations achieved by these networks are difficult to interpret and to analyze. For many riskier applications (e.g. self-driving cars, robotic surgeries, military drones, etc.), a machine should only run autonomously if it is able to *explain* its every decision and action. Further, when used towards the scientific analysis of phenomena (e.g. understanding animal behavior, weather patterns, financial market trends, etc.), the goal is to extract a causal interpretation of the system in question; hence, to be useful, a machine should be able to provide a clear explanation of its internal logic.

For these reasons, it is desirable to have a simple white-box machine learning system that is able to train quickly and with little data. With these constraints in mind, we showcase a multi-class boosting algorithm called *REBEL* and a novel family of weak learners called *similarity stumps*, leading to much better generalization than decision stumps, as shown in Fig. 1. Our proposed framework is simple, efficient, and is able to perfectly train on *any* dataset (i.e. fully minimize the training error in a finite number of iterations).

[1]Caltech, Pasadena, USA. Correspondence to: Ron Appel <appel@vision.caltech.edu>, Pietro Perona <perona@vision.caltech.edu>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

The main contributions of our work are as follows:

1. a simple multi-class boosting framework using localized similarities as weak learners (see Sec. 3)
2. a proof that the training error is fully minimized within a finite number of iterations (see Sec. 5)
3. a procedure for selecting an adequate learner at each iteration (see Sec. 5.2)
4. empirical demonstrations of state-of-the-art results on a range of datasets (see Sec. 7)

2. Background

Boosting is a fairly mature method, originally formulated for binary classification (e.g. AdaBoost and similar variants) (Schapire, 1990; Freund, 1995; Freund & Schapire, 1996). Multi-class classification is more complex than its binary counterpart, however, many advances have been made in both performance and theory in the context of boosting. Since weak learners come in two flavors, binary and multi-class, two corresponding families of boosting methods have been explored.

The clever combination of multiple binary weak learners can result in a multi-class prediction. AdaBoost.MH reduces the K-class problem into a single binary problem with a K-fold augmented dataset (Schapire & Singer, 1999). AdaBoost.MO and similar methods reduce the K-class problem into C one-versus-all binary problems using Error-Correcting Output Codes to select the final hypothesized class (Allwein et al., 2001; Sun et al., 2005; Li, 2006). More recently, CD-MCBoost and CW-Boost return a K-dimensional vector of class scores, focusing each iteration on a (binary) problem of improving the margin of one class at a time (Saberian & Vasconcelos, 2011; Shen & Hao, 2011). REBEL also returns a vector of class scores, increasing the margin between dynamically-selected binary groupings of the K classes at each iteration (Appel et al., 2016).

When multi-class weak learners are acceptable (and available), a reduction to binary problems is unnecessary. AdaBoost.M1 is a straightforward extension of its binary counterpart (Freund & Schapire, 1996). AdaBoost.M2 and AdaBoost.MR make use of a K-fold augmented dataset to estimate output label probabilities or rankings for a given input (Freund & Schapire, 1996; Schapire & Singer, 1999). More recent methods such as SAMME, AOSO-LogitBoost, and GD-MCBoost are based on linear combinations of a fixed set of codewords, outputting K-dimensional score vectors (Zhu et al., 2009; Sun et al., 2011; Saberian & Vasconcelos, 2011).

In the noteworthy paper "A Theory of Multiclass Boosting" (Mukherjee & Schapire, 2010), many of the existing boosting methods were shown to be inadequate at training; either because they require their weak learners to be too strong, or because their loss functions are unable to deal with some training data configurations. (Mukherjee & Schapire, 2010) outline the appropriate *Weak Learning Condition* that a boosting algorithm must require of its weak learners in order to guarantee training convergence. However, no method is prescribed with which to find an adequate set of weak learners.

The goal of our work is to propose a multi-class boosting framework with a simple family of binary weak learners that guarantee training convergence and are easily interpretable. Using REBEL (Appel et al., 2016) as the multi-class boosting method, our framework is meant to be as straightforward as possible so that it is accessible and practical to more users; outlining it in Sec. 3 below.

3. Our Framework

In this section, we define our notation, introduce our boosting framework, and describe our training procedure.

Notation	
scalars (regular), vectors (bold):	$x, \quad \mathbf{x} \equiv [x_1, x_2, ...]$
constant vectors:	$\mathbf{0} \equiv [0,0,...], \ \mathbf{1} \equiv [1,1,...]$
indicator vector:	$\boldsymbol{\delta}_k$ ($\mathbf{0}$ with a 1 in the k^{th} entry)
logical indicator function:	$\mathbb{1}_{(\text{LOGICAL EXPRESSION})} \in \{0,1\}$
inner product:	$\langle \mathbf{x}, \mathbf{v} \rangle$
element-wise multiplication:	$\mathbf{x} \odot \mathbf{v}$
element-wise function:	$\mathbf{F}[\mathbf{x}] \equiv [F(x_1), F(x_2), ...]$

In the multi-class classification setting, a datapoint is represented as a feature vector $\mathbf{x}$ and is associated with a class label y. Each point is comprised of d features and belongs to one of K classes: $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d, \quad y \in \mathcal{Y} \equiv \{1,2,...,K\}$

A good classifier reduces the training error while generalizing well to potentially-unseen data. We use REBEL (Appel et al., 2016) due to its support for binary weak learners, its mathematical simplicity (i.e. closed-form solution to loss minimization), and its strong empirical performance. REBEL returns a vector-valued output $\mathbf{H}$, the sum of T {weak learner f, accumulation vector $\mathbf{a}$} pairs, where $f_t : \mathcal{X} \to \{\pm 1\}$ and $\mathbf{a}_t \in \overline{\mathbb{R}}^K$:

$$\mathbf{H}(\mathbf{x}) \equiv \sum_{t=1}^{T} f_t(\mathbf{x})\, \mathbf{a}_t$$

The hypothesized class is simply the index of the maximal entry in $\mathbf{H}$:

$$F(\mathbf{x}) \equiv \arg\max_{y \in \mathcal{Y}} \{\langle \mathbf{H}(\mathbf{x}), \boldsymbol{\delta}_y \rangle\}$$

The average misclassification error ε can be expressed as:

$$\varepsilon \ \equiv \ \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}_{(F(\mathbf{x}_n) \neq y_n)} \tag{1}$$

REBEL uses an exponential loss function to upper-bound the average training misclassification error:

$$\varepsilon \;\le\; \mathcal{L} \;\equiv\; \frac{1}{2N}\sum_{n=1}^{N}\langle \mathbf{exp}[\mathbf{y}_n \odot \mathbf{H}(\mathbf{x}_n)], \mathbf{1}\rangle \qquad (2)$$

where: $\mathbf{y}_n \equiv \mathbf{1}{-}2\,\boldsymbol{\delta}_{y_n}$ (i.e. all $+1$s with a -1 in the y_n^{th} index)

Being a greedy, additive model, all previously-trained parameters are fixed and each iteration amounts to jointly optimizing a new weak learner f and accumulation vector $\mathbf{a}$. To this end, the loss at iteration $I{+}1$ can be expressed as:

$$\mathcal{L}_{I+1} = \frac{1}{N}\sum_{n=1}^{N}\langle \mathbf{w}_n, \mathbf{exp}[f(\mathbf{x}_n)\,\mathbf{y}_n \odot \mathbf{a}]\rangle \qquad (3)$$

$$\text{where:}\quad \mathbf{w}_n \equiv \frac{1}{2}\,\mathbf{exp}[\mathbf{y}_n \odot \mathbf{H}_I(\mathbf{x}_n)]$$

Given a weak learner f, we define true and false (i.e. correct and incorrect) multi-class weight sums ($\mathbf{s}_f^{\mathrm{T}}$ and $\mathbf{s}_f^{\mathrm{F}}$) as:

$$\mathbf{s}_f^{\mathrm{T}} \equiv \frac{1}{N}\sum_{n=1}^{N}\mathbb{1}[f(\mathbf{x}_n)\mathbf{y}_n<\mathbf{0}]\odot\mathbf{w}_n, \quad \mathbf{s}_f^{\mathrm{F}} \equiv \frac{1}{N}\sum_{n=1}^{N}\mathbb{1}[f(\mathbf{x}_n)\mathbf{y}_n>\mathbf{0}]\odot\mathbf{w}_n$$

$$\text{thus:}\quad \mathbf{s}_f^{\mathrm{T}}+\mathbf{s}_f^{\mathrm{F}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{w}_n, \quad \mathbf{s}_f^{\mathrm{T}}-\mathbf{s}_f^{\mathrm{F}} = \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{x}_n)\,\mathbf{w}_n\odot\mathbf{y}_n$$

Using these weight sums, the loss can be simplified to:

$$\mathcal{L}_{I+1} \equiv \mathcal{L}_f \equiv \langle \mathbf{s}_f^{\mathrm{T}}, \mathbf{exp}[-\mathbf{a}]\rangle + \langle \mathbf{s}_f^{\mathrm{F}}, \mathbf{exp}[\mathbf{a}]\rangle \qquad (4)$$

In this form, it is easily shown that with the optimal accumulation vector $\mathbf{a}^*$, the loss has an explicit expression:

$$\mathbf{a}^* = \frac{1}{2}\big(\mathbf{ln}[\mathbf{s}_f^{\mathrm{T}}] - \mathbf{ln}[\mathbf{s}_f^{\mathrm{F}}]\big) \quad \therefore\; \mathcal{L}_f^* = 2\langle\sqrt{\mathbf{s}_f^{\mathrm{T}}\odot\mathbf{s}_f^{\mathrm{F}}}, \mathbf{1}\rangle \qquad (5)$$

At each iteration, growing decision trees requires an exhaustive search through a pool of decision stumps (which is tractable but time-consuming), storing the binary learner that best reduces the multi-class loss in Eq. 5. In some situations, axis-aligned trees are simply unable to reduce the loss any further, thereby stalling the training procedure.

Our proposed framework circumvents such situations. At each iteration, instead of exhaustively searching for an adequate learner, we first determine an appropriate "binarization" of the multi-class data (i.e. a separation of the K-class data into two distinct groups) and then find a weak learner with a guaranteed reduction in loss, foregoing the need for an exhaustive search.

4. Binarizing Multi-Class Data

At each iteration, the first step in determining an adequate weak learner is *binarizing* the data, i.e. assigning a temporary binary label to each data point by placing it into one of two groups. The following manipulations result in a procedure for binarizing datapoints given their boosting weights.

Eq. 5 can be upper-bounded as follows:

$$\mathcal{L}_f^* \;=\; 2\langle\sqrt{\mathbf{s}_f^{\mathrm{T}}\odot\mathbf{s}_f^{\mathrm{F}}}, \mathbf{1}\rangle \;\le\; \langle \mathbf{s}_f^{\mathrm{T}}+\mathbf{s}_f^{\mathrm{F}}, \mathbf{1}\rangle - \frac{1}{2}\Big\langle \overbrace{\frac{[\mathbf{s}_f^{\mathrm{T}}-\mathbf{s}_f^{\mathrm{F}}]^2}{[\mathbf{s}_f^{\mathrm{T}}+\mathbf{s}_f^{\mathrm{F}}]}}^{\mathcal{U}}, \mathbf{1}\Big\rangle \qquad (6)$$

$$\text{since: } \sqrt{x(1-x)} \le \frac{1}{2}-\Big(\frac{1}{2}-x\Big)^2 \;\;\forall x, \quad \text{using: } x = \frac{s^{\mathrm{T}}}{s^{\mathrm{T}}+s^{\mathrm{F}}}$$

By expanding $\mathbf{s}_f^{\mathrm{T}} \pm \mathbf{s}_f^{\mathrm{F}}$, $\mathcal{U}$ is expressed as a squared norm:

$$\mathcal{U} \;=\; \left\langle \frac{\Big[\frac{1}{N}\sum_{n=1}^{N} f(\mathbf{x}_n)\,\mathbf{w}_n\odot\mathbf{y}_n\Big]^2}{\Big[\frac{1}{N}\sum_{n=1}^{N}\mathbf{w}_n\Big]}, \mathbf{1}\right\rangle \;=\; \Big\|\sum_{n=1}^{N} f(\mathbf{x}_n)\,\mathbf{u}_n\Big\|^2 \qquad (7)$$

$$\text{where: } \mathbf{u}_n \;\equiv\; \frac{1}{\sqrt{N}}\frac{\mathbf{w}_n\odot\mathbf{y}_n}{\sqrt{\sum_{n=1}^{N}\mathbf{w}_n}}$$

Eq. 7 can be written as a product of matrices by stacking all of the $\mathbf{u}_n$ as column vectors of a $K\times N$ matrix $\underline{\mathbf{U}}$ and defining $\mathbf{f}$ as a row vector with elements $f(\mathbf{x}_n)$:

$$\mathcal{U} \;=\; \mathbf{f}\,[\underline{\mathbf{U}}^{\top}\underline{\mathbf{U}}]\,\mathbf{f}^{\top}$$

Note that the trace of $\underline{\mathbf{U}}^{\top}\underline{\mathbf{U}}$ can be lower-bounded:

$$\mathrm{tr}(\underline{\mathbf{U}}^{\top}\underline{\mathbf{U}}) = \sum_{n=1}^{N}\|\mathbf{u}_n\|^2 = \left\langle \frac{\sum_{n=1}^{N}[\mathbf{w}_n]^2}{N\Big[\sum_{n=1}^{N}\mathbf{w}_n\Big]}, \mathbf{1}\right\rangle \ge \frac{1}{N^2}\sum_{n=1}^{N}\langle \mathbf{w}_n, \mathbf{1}\rangle$$

$$\text{since by Jensen's inequality: } \sum_{n=1}^{N} x_n^2 \;\ge\; \frac{1}{N}\Big(\sum_{n=1}^{N} x_n\Big)^2$$

Furthermore, $\underline{\mathbf{U}}^{\top}\underline{\mathbf{U}}$ has N (not-necessarily unique) non-negative eigenvalues, each associated with an independent eigenvector. Let $\hat{\mathbf{v}}_n$ be the eigenvector corresponding to the n^{th} largest eigenvalue λ_n. Hence, $\mathbf{f}$ can be decomposed as:

$$\mathbf{f} \;=\; \langle \mathbf{f}, \hat{\mathbf{v}}_1\rangle\,\hat{\mathbf{v}}_1 + \sum_{n=2}^{N}\langle \mathbf{f}, \hat{\mathbf{v}}_n\rangle\,\hat{\mathbf{v}}_n \qquad (8)$$

$$\therefore\;\; \mathcal{U} \;=\; \lambda_1\langle \mathbf{f}, \hat{\mathbf{v}}_1\rangle^2 + \sum_{n=2}^{N}\lambda_n\langle \mathbf{f}, \hat{\mathbf{v}}_n\rangle^2 \;\ge\; \lambda_1\langle \mathbf{f}, \hat{\mathbf{v}}_1\rangle^2$$

Since the trace of a matrix is equal to the sum of its eigenvalues and $\underline{\mathbf{U}}^\top\underline{\mathbf{U}}$ has at most K non-zero eigenvalues (λ_1 being the largest), hence:

$$\lambda_1 \;\geq\; \frac{1}{K}\,\mathrm{tr}(\underline{\mathbf{U}}^\top\underline{\mathbf{U}}) \;\geq\; \frac{\mathcal{L}_0}{KN} \tag{9}$$

$$\text{since:}\;\; \frac{1}{N}\sum_{n=1}^{N}\langle \mathbf{w}_n, \mathbf{1}\rangle = \mathcal{L}_0$$

Based on this formulation, binarization is achieved by setting the *binarized class* b_n of each sample n as the sign of its corresponding element in $\hat{\mathbf{v}}_1$: $b_n \equiv \mathrm{sign}(\langle \hat{\mathbf{v}}_1, \boldsymbol{\delta}_n\rangle)$

Accordingly, if $\mathbf{b}$ is the vector with elements b_n, then:

$$\langle \mathbf{b}, \hat{\mathbf{v}}_1\rangle^2 \;=\; \langle \mathbf{sign}[\hat{\mathbf{v}}_1], \hat{\mathbf{v}}_1\rangle^2 \;=\; \langle |\hat{\mathbf{v}}_1|, \mathbf{1}\rangle^2 \;\geq\; 1 \tag{10}$$

(please refer to supplement for proof)

Finally, by combining Eq. 6, Eq. 9, and Eq. 10, with perfect binarized classification (i.e. when the binary weak learner perfectly classifies the binarized data), the loss ratio at any iteration is bounded by:

$$\frac{\mathcal{L}_{f^*}}{\mathcal{L}_0} \;\leq\; 1 - \frac{1}{2KN}$$

In general, there is no guarantee that any weak learner can achieve perfect binarized classification. In the following section, we show that with the ability to *isolate* any single point in space (i.e. to classify an inner point as $+1$ and all outer points as -1), the loss decreases exponentially.

5. Isolating Points

Assume that we have a weak learner f_i that can isolate a single point $\mathbf{x}_i$ in the input space $\mathcal{X}$. Accordingly, denote $\mathbf{f}_i = 2\,\boldsymbol{\delta}_i - \mathbf{1}$ as a vector of -1s with a $+1$ in the i^{th} entry, corresponding to classification using the isolating learner $f_i(\mathbf{x}_n)$. If $N \geq 4$, then for any unit vector $\hat{\mathbf{v}} \in \mathbb{R}^N$:

$$\max_i\{\langle \mathbf{f}_i, \hat{\mathbf{v}}\rangle^2\} \;\geq\; \frac{4}{N} \tag{11}$$

(please refer to supplement for proof)

Combining Eq. 6, Eq. 9, and Eq. 11, the loss ratio at each iteration is upper-bounded as follows:

$$\frac{\min_i\{\mathcal{L}_{f_i}\}}{\mathcal{L}_0} \;\leq\; 1 - \frac{2}{KN^2}$$

Before the first iteration, the initial loss $\mathcal{L}_0 = K/2$. Each iteration decreases the loss exponentially. Since the training error is discrete and is upper bounded by the loss (as in Eq. 2), our framework attains minimal training error on *any*[1] training set after a finite number of iterations:

$$\text{define:}\;\; T_0 \equiv \left\lceil \frac{\ln(2/KN)}{\ln\!\left(1-\frac{2}{KN^2}\right)} \right\rceil \approx \left\lceil \frac{KN^2}{2}\ln\!\left(\frac{KN}{2}\right) \right\rceil$$

$$\therefore\;\; T \geq T_0 \;\;\Rightarrow\;\; \frac{K}{2}\left(1-\frac{2}{KN^2}\right)^T < \frac{1}{N} \;\;\Rightarrow\;\; \varepsilon = 0$$

Although this bound is too weak to be of practical use, it is a bound nonetheless (and can likely be improved). In the following section, we specify a suitable family of weak learners with the ability to isolate single points.

5.1. One/Two-Point Localized Similarities

Classical decision stumps compare a single feature to a threshold, outputting $+1$ or -1. Instead, our proposed family of weak learners (called *localized similarities*) compare points in the input space using a similarity measure. Due to its simplicity and effectiveness, we use negative squared Euclidean distance $-\|\mathbf{x}_i - \mathbf{x}_j\|^2$ as a measure of similarity between points $\mathbf{x}_i$ and $\mathbf{x}_j$. A localized similarity has two modes of operation:

1. In one-point mode, given an anchor $\mathbf{x}_i$ and a threshold τ, the input space is classified as positive if it is more *similar* to $\mathbf{x}_i$ than τ, and negative otherwise; ranging between $+1$ and -1:

$$f_i(\mathbf{x}) \;\equiv\; \frac{\tau - \|\mathbf{x}_i - \mathbf{x}\|^2}{\tau + \|\mathbf{x}_i - \mathbf{x}\|^2}$$

2. In two-point mode, given supports $\mathbf{x}_i$ and $\mathbf{x}_j$, the input space is classified as positive if it is more *similar* to $\mathbf{x}_i$ than to $\mathbf{x}_j$ (and vice-versa), with maximal absolute activations around $\mathbf{x}_i$ and $\mathbf{x}_j$; falling off radially away from the midpoint $\mathbf{m}$:

$$f_{ij}(\mathbf{x}) \;\equiv\; \frac{\langle \mathbf{d}, \mathbf{x}-\mathbf{m}\rangle}{4\|\mathbf{d}\|^4 + \|\mathbf{x}-\mathbf{m}\|^4}$$

$$\text{where:}\;\; \mathbf{d} \equiv \frac{1}{2}\,[\mathbf{x}_i - \mathbf{x}_j] \quad \text{and:}\;\; \mathbf{m} \equiv \frac{1}{2}\,[\mathbf{x}_i + \mathbf{x}_j]$$

One-point mode enables the isolation of any single datapoint, guaranteeing a baseline reduction in loss. However, it essentially leads to pure memorization of the training data; mimicking a nearest-neighbor classifier. Two-point mode adds the capability to generalize better by providing margin-style functionality. The combination of these

[1] There may be situations in which multiple samples belonging to different classes are coincident in the input space. These cases can be dealt with (before or during training) either by assigning all such points as a special "mixed" class (to be dealt with at a later stage), or by setting the class labels of all coincident points to the single label that minimizes the error.

two modes enables the flexibility to tackle a wide range of classification problems. Furthermore, in either mode, the functionality of a localized similarity is easily interpretable: "which of these fixed training points is a given query point more similar to?"

5.2. Finding Adequate Localized Similarities

Given a dataset with N samples, there are about N^2 possible localized similarities. The following procedure efficiently selects an adequate localized similarity:

0. Using Eq. 5, calculate the base loss $\mathcal{L}_1$ for the *homogeneous* stump f_1 (i.e. the one-point stump with any $\mathbf{x}_i$ and $\tau \equiv \infty$, classifying all points as $+1$).
1. Compute the eigenvector $\hat{\mathbf{v}}_1$ (as in Eq. 8); label the points based on their binarized class labels b_n.
2. Find the optimal isolating localized similarity f_i (i.e. with $\mathbf{x}_i$ and appropriate τ, classifying point i as $+1$ and all other points as -1).
3. Using Eq. 5, calculate the corresponding loss $\mathcal{L}_i$. Of the two stumps f_1 and f_i, store the one with smaller loss as best-so-far.
4. Find point $\mathbf{x}_j$ most similar[2] to $\mathbf{x}_i$ among points of the opposite binarized class:

$$\mathbf{x}_j = \arg\min_{b_j=-b_i} \{\|\mathbf{x}_i - \mathbf{x}_j\|^2\}$$

5. Calculate the loss achieved using the two-point localized similarity f_{ij}. If it outperforms the previous best, store the newer learner and update the best-so-far loss.
6. Find all points that are *similar enough* to $\mathbf{x}_j$ and remove them from consideration for the remainder of the current iteration. In our implementation, we remove all $\mathbf{x}_n$ for which:

$$f_{ij}(\mathbf{x}_n) \leq f_{ij}(\mathbf{x}_j)/2$$

If all points have been removed, return the best-so-far localized similarity; otherwise, loop back to step 4.

Upon completion of this procedure, the best-so-far localized similarity is guaranteed to lead to an adequate reduction in loss, based on the derivation in Sec. 4 above.

6. Generalization Experiments

Our boosting method provably reduces the loss well after the training error is minimized. In this section, we demonstrate that the continual reduction in loss serves only to improve the decision boundaries and not to overfit the data.

We generated 2-dimensional synthetic datasets in order to better visualize and get an intuition for what the classifiers are doing. The results shown in this chapter are based on a dataset composed of 500 points belonging to one of three classes in a spiral formation, with a $(2/3, 1/3)$ train/test split. Fig. 2 shows the hypothesized class using a classifier trained for 1000 iterations.

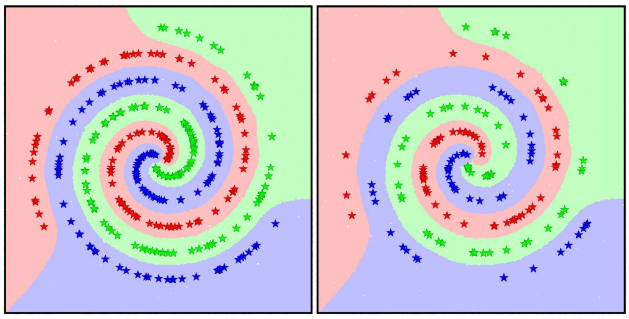


Figure 2. A 500-point 2-dimensional synthetic dataset with a $(2/3, 1/3)$ split of train data (left plot) to test data (right plot). Background shading corresponds to the hypothesized class using our framework.

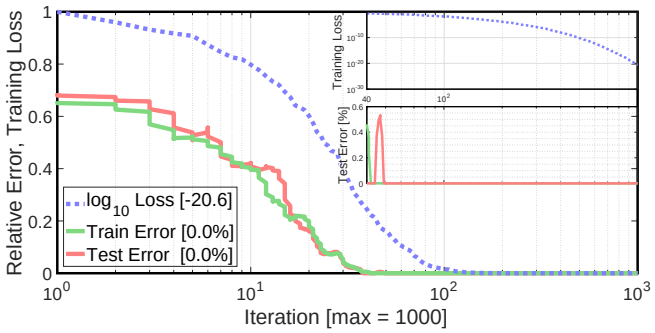


Figure 3. A plot of training loss, training error, and test error as a classifier is trained for 1000 iterations. Note that the test error does not increase even after the training error drops to zero. The lower inset is a zoomed-in plot of the train and test error, the upper inset is a plot of training loss using a log-scaled y-axis; both inset plots are congruous with the original x-axis.

Our classifier achieves perfect training (left) and test classification (right), producing a visually simple *well-generalizing* contour around the points. Training curves are given in Fig. 3, tracking the loss and classification errors per training iteration. Note that the test error does not increase even after the training error drops to zero.

The following experiments explore the functionality of our framework (i.e. REBEL using localized similarities) in two scenarios that commonly arise in practice: (1) varying sparsity of training data, and, (2) varying amounts of mislabeled training data.

[2] "most similar" need not be exact; approximate nearest-neighbors also works with negligible differences.

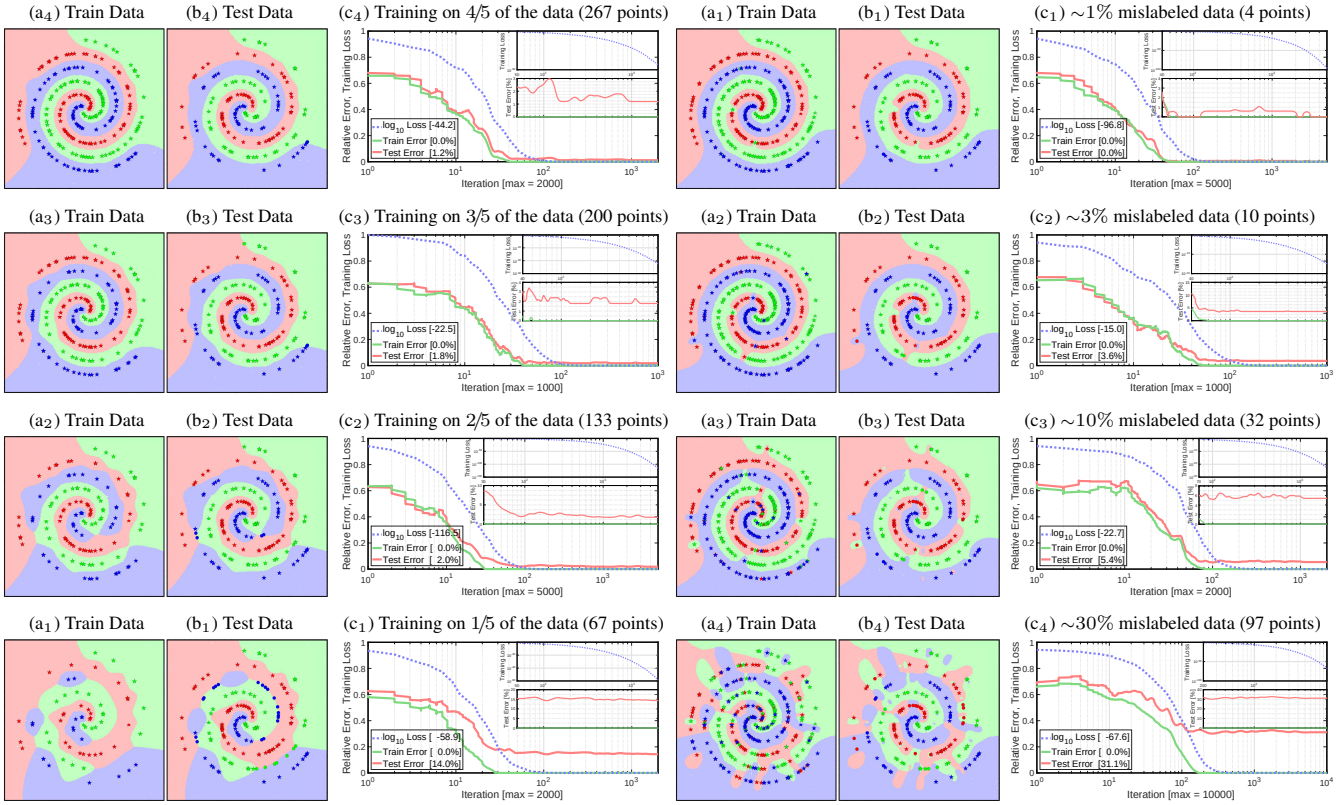


Figure 4. Classification boundaries (a,b), and training curves (c) when a classifier is trained on varying amounts of data. Stars are correctly-classified, circles are misclassified. In all cases, the test error is fairly stable once reaching its minimum.

Figure 5. Classification boundaries (a,b), and training curves (c) when a classifier is trained on varying fractions of mislabeled data. In all cases, the test error is fairly stable once reaching its minimum. Even with 30% mislabeled data, the classification boundaries are reasonable given the training labels.

6.1. Sparse Training Data

In this section of experiments, classifiers were trained using varying amounts of data, from $4/5$ to $1/5$ of the total training set. Fig. 4 shows the classification boundaries learned by the classifier (a_i,b_i), and the training curves (c_i). In all cases, the boundaries seem to aptly fit (and not *overfit*) the training data (i.e. being satisfied with isolated patches *without* overzealously trying to connect points of the same class together). This is more rigorously observed from the training curves; the test error does not increase after reaching its minimum (for hundreds of iterations).

6.2. Mislabeled Training Data

In this section of experiments, classifiers were trained with varying fractions of mislabeled data; from 1% to 30% of the training set. Fig. 5 shows the classification boundaries (a_i,b_i) and the training curves (c_i). All classifiers seem to degenerate gracefully, isolating rogue points and otherwise maintaining smooth boundaries. Even the classifier trained on 30% mislabeled data (which we would consider to be unreasonably noisy) is able to maintain smooth boundaries. In all cases, the training curves still show that the test error is fairly stable once reaching its minimum value. Moreover, test errors approximately equal the fraction of mislabeled data, further validating the generalization of our method.

6.3. Real Data

Although the above observations are promising, they could result from the fact that the synthetic datasets are 2-dimensional. In order to rule out this possibility, we performed similar experiments on several UCI datasets (Bache & Lichman, 2013) of varying input dimensionalities (from 9 to 617). From the training curves in Fig. 6, we observe that once the test errors saturate, they no longer increase, even after hundreds of iterations.

In Fig. 7, we plot the training losses on a log-scaled y-axis. The linear trend signifies an exponential decrease in loss per iteration. Our proven bound predicts a much slower (exponential) rate than the actual trend observed during training. Note that within the initial $\sim 10\%$ of the iterations, the loss drops at an even faster rate, after which it settles down

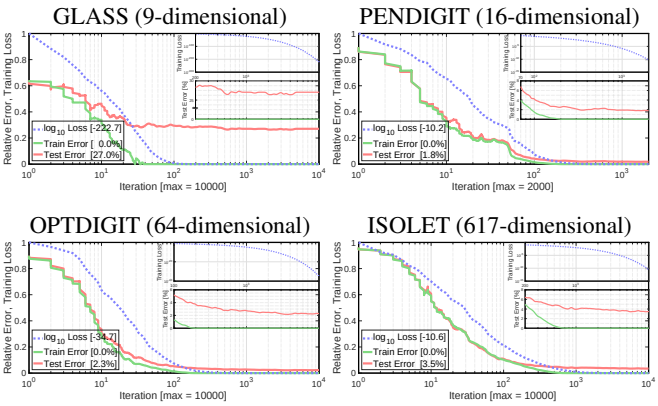


Figure 6. Training curves for classifiers trained on UCI datasets with a range of dimensionalities. In all cases, the test error is stable once it reaches its minimum.

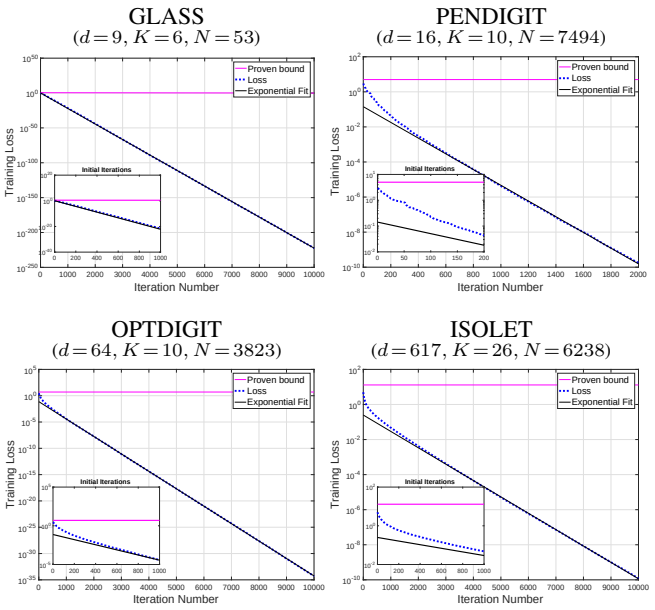


Figure 7. Training losses for classifiers trained on UCI datasets. The linear trend (visualized using a log-scaled y-axis) signifies an exponential decrease in loss, albeit at a much faster rate than established by our proven bound.

to a seemingly-constant rate of exponential decay. We have not yet determined the characteristics (i.e. the theoretically justified rates) of these observed trends, and relegate this endeavor to future work.

7. Comparison with Other Methods

In Sec. 5 we proved that our framework adheres to theoretical guarantees, and in Sec. 6 above, we showed that it has promising empirical properties. In this section, we compete against several state-of-the-art boosting baselines. Specifically, we compared 1-vs-All AdaBoost and AdaBoost.MH (Schapire & Singer, 1999), AdaBoost.ECC (Dietterich & Bakiri, 1995), Struct-Boost (Shen et al., 2014), CW-Boost (Shen & Hao, 2011), AOSO-LogitBoost (Sun et al., 2011), REBEL (Appel et al., 2016) using shallow decision trees, REBEL using only 1-point (isolating) similarities, and our full framework, REBEL using 2-point localized similarities.

Based on the same experimental setup as in (Shen et al., 2014; Appel et al., 2016), competing methods are trained to a maximum of 200 decision stumps. For each dataset, five random splits are generated, with 50% of the samples for training, 25% for validation (i.e. for setting hyperparameters where needed), and the remaining 25% for testing.

REBEL using localized similarities is the most accurate method on five of the six datasets tested. In the Vowel dataset, it achieves almost half of the error as the next best method. Note that although our framework uses REBEL as its boosting method, the localized similarities add an extra edge, beating REBEL with decision trees in all runs.

Further, when limited to only using 1-point (i.e. isolating) localized similarities, the performance is extremely poor, validating the need for 2-point localized similarities as prescribed in Sec. 5.2. Overall, these results demonstrate the ability of our framework to produce easily interpretable classifiers that are also empirically proficient.

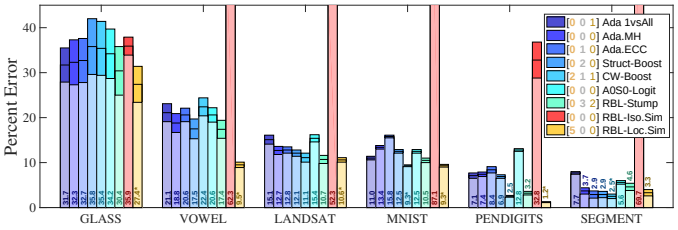


Figure 8. Test errors of various state-of-the-art and baseline classification methods on MNIST and several UCI datasets. REBEL using localized similarities (shown in yellow) is the best-performing method on all but one of the datasets shown. When constrained to use only 1-point (isolating) similarities (shown in red), the resulting classifier is completely inadequate.

7.1. Comparison with Neural Networks and SVMs

Complex neural networks are able to achieve remarkable performance on large datasets, but they require an amount of training data proportional to their complexity. In the regime of small to medium amounts of data (within which UCI and MNIST datasets belong, i.e. $10 < N < 10^6$ training samples), such networks cannot be too complex. Accordingly, in Fig. 9, we compare our method against fully-connected neural networks.

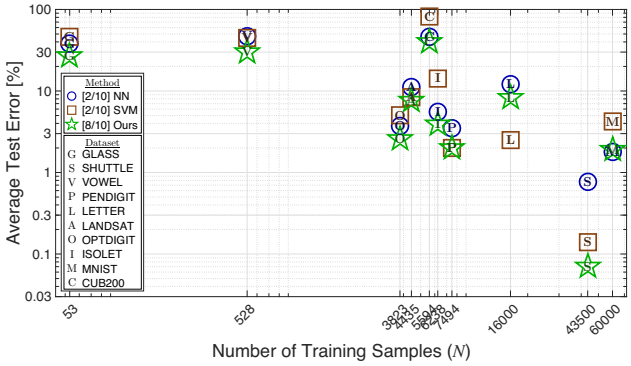


Figure 9. Comparison of our method versus Neural Networks and Support Vector Machines on ten datasets of varying sizes and difficulties. Our method is the most accurate on almost all datasets.

Four neural networks were implemented, each having one of the following architectures: $[d-4d-K]$, $[d-4K-K]$, $[d-2d-d-K]$, $[d-4K-2K-K]$, where d is the number of input dimensions and K is the number of output classes. Only the one with the best test error is shown in the plot. A multi-class SVM (Chang & Lin, 2011) was validated using a 5×6 parameter sweep for C and γ. Our method was run until the training loss fell below $1/N$. Overall, REBEL using localized similarities achieves the best results on eight of the ten datasets, decisively marking it as the method of choice for this range of data.

8. Discussion

In Sec. 6, we observed that our classifiers tend to smoothen the decision boundaries in the iterations beyond zero training error. In Fig. 10, we see that this is not the case with the typically-used axis-aligned decision stumps. Why does this happen with our framework?

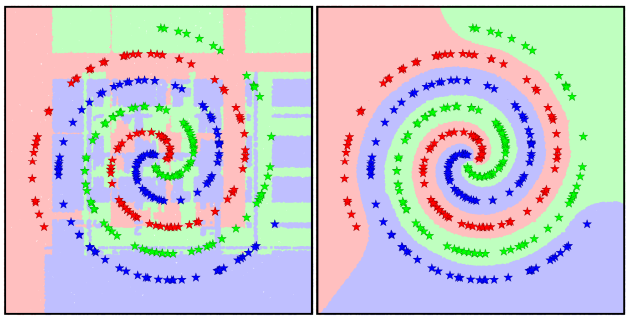


Figure 10. The contrasted difference between overtraining using (a) classical decision stumps and (b) localized similarities. (a) leads to massive overfitting of the training data, whereas (b) leads to smoothening of the decision boundaries.

Firstly, we note that the largest-margin boundary between two points is the hyperplane that bisects them. Every two-point localized similarity acts as such a bisector. Therefore, it is not surprising that with only a pool of localized similarities, a classifier should have what it needs to place good boundaries. Further, not all pairs need to be separated (since many neighboring points belong to the same class); hence, only a small subset of the $\sim N^2$ possible learners will ever need to be selected.

Secondly, we note that if some point (either an outlier or an unfortunately-placed point) continues to increase in weight until it can no-longer be ignored, it can simply be isolated and individually dealt with using a one-point localized similarity, there is no need to combine it with other "innocent-bystander" points. This phenomenon is observed in the mislabeled training experiments in Sec. 6.2.

Together, the two types of localized similarities complement each other. With the guarantee that every step reduces the loss, each iteration focuses on either further smoothening out an existing boundary, or reducing the weight of a single unfit point.

9. Conclusions

We have presented a novel framework for multi-class boosting that makes use of a simple family of weak learners called localized similarities. Each of these learners has a clearly understandable functionality; a test of similarity between a query point and some pre-defined samples.

We have proven that the framework adheres to theoretical guarantees: the training loss is minimized at an exponential rate, and since the loss upper-bounds the training error (which can only assume discrete values), our framework is therefore able to achieve maximal accuracy on any dataset.

We further explored some of the empirical properties of our framework, noting that the combination of localized similarities and guaranteed loss reduction tend to lead to a non-overfitting regime, in which the classifier focuses on smoothing-out its decision boundaries. Finally, we compare our method against several state-of-the-art methods, outperforming all of the methods in most of the datasets.

Altogether, we believe that we have achieved our goal of presenting a simple multi-class boosting framework with theoretical guarantees and empirical proficiency.

Acknowledgements

The authors would like to thank anonymous reviewers for their feedback and Google Inc. and the Office of Naval Research MURI N00014-10-1-0933 for funding this work.

References

Allwein, E. L., Schapire, R. E., and Singer, Y. Reducing multiclass to binary: a unifying approach for margin classifiers. *JMLR*, 2001.

Appel, R., Burgos-Artizzu, X. P., and Perona, P. Improved multi-class cost-sensitive boosting via estimation of the minimum-risk class. *arXiv*, (1607.03547), 2016.

Bache, K. and Lichman, M. UCI machine learning repository (uc irvine), 2013. URL http://archive.ics.uci.edu/ml.

Chang, C. and Lin, C. LIBSVM: A library for support vector machines. *Transactions on Intelligent Systems and Technology*, 2011.

Dietterich, T. G. and Bakiri, G. Solving multiclass learning problems via error-correcting output codes. *arXiv*, (9501101), 1995.

Freund, Y. Boosting a weak learning algorithm by majority. *Information and Computation*, 1995.

Freund, Y. and Schapire, R. E. Experiments with a new boosting algorithm. In *Machine Learning International Workshop*, 1996.

LeCun, Y., Bengio, Y., and Hinton, G. E. Deep learning. *Nature Research*, 2015.

Li, L. Multiclass boosting with repartitioning. In *ICML*, 2006.

Mukherjee, I. and Schapire, R. E. A theory of multiclass boosting. In *NIPS*, 2010.

Saberian, M. and Vasconcelos, N. Multiclass boosting: Theory and algorithms. In *NIPS*, 2011.

Schapire, R. E. The strength of weak learnability. *Machine Learning*, 1990.

Schapire, R. E. and Singer, Y. Improved boosting algorithms using confidence-rated predictions. In *Conference on Computational Learning Theory*, 1999.

Shen, C. and Hao, Z. A direct formulation for totally-corrective multi-class boosting. In *CVPR*, 2011.

Shen, G., Lin, G., and van den Hengel, A. Structboost: Boosting methods for predicting structured output variables. *PAMI*, 2014.

Sun, P., Reid, M. D., and Zhou, J. Aoso-logitboost: Adaptive one-vs-one logitboost for multi-class problem. *arXiv*, (1110.3907), 2011.

Sun, Y., Todorovic, S., Li, J., and Wu, D. Unifying the error-correcting and output-code adaboost within the margin framework. In *ICML*, 2005.

Yu, F., Zhang, Y., Song, S., Seff, A., and Xiao, J. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv*, (1506.03365), 2015.

Zhu, J., Zou, H., Rosset, S., and Hastie, T. Multi-class adaboost. *Statistics and its Interface*, 2009.