A Cluster-Based Semisupervised Ensemble for Multiclass Classification

Rodrigo G. F. Soares, Huanhuan Chen, *Senior Member, IEEE*, and Xin Yao, *Fellow, IEEE*

***Abstract*—Semisupervised classification (SSC) algorithms use labeled and unlabeled data to predict labels of unseen instances. Classifier ensembles have been successfully studied and employed as a SSC approach. However, the generalization of existing semisupervised ensembles can be strongly affected by incorrect label estimates produced by ensemble algorithms in order to train supervised base learners. These ensembles do not optimize the objective function present in their base learners, which causes their supervised base classifiers to be sensitive to incorrect labeling and to reinforce errors during training. We propose cluster-based boosting (CBoost), a multiclass classification algorithm with cluster regularization. In contrast to existing algorithms, CBoost and its base learners jointly perform a cluster-based semisupervised optimization, which allows base classifiers to overcome potential incorrect label estimates for unlabeled data. CBoost is effective and stable in the presence of overlapping classes and scarce labeled points in dense regions. Experiments on artificial and real-world datasets confirmed the effectiveness of our approach.**

***Index Terms*—Boosting, Clusterreg, ensemble learning, semisupervised learning.**

I. Introduction

ACQUIRING labeled data can be costly as labels may, for example, depend on human experts or expensive sensors. In contrast, collecting large amounts of unlabeled data might be straightforward and cheap. Hence, it is intuitive to use the easily available unlabeled instances to enhance generalization accuracy. Semisupervised learning (SSL) employs unlabeled and labeled instances to train learning machines with higher predictive performance. SSL has been widely studied in both theoretical and experimental machine learning research, due to its low requirements of human effort and its potentially enhanced accuracy [1]. In this work, we propose the Cluster-based Boosting (*CBoost*) algorithm for multiclass semisupervised classification (SSC).

Manuscript received December 6, 2016; revised July 6, 2017; accepted August 14, 2017. Date of current version November 22, 2017. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB1000905, in part by the National Natural Science Foundation of China under Grants 91546116 and 61673363, and in part by the CAPES Foundation, Ministry of Education of Brazil. The work of X. Yao was supported by a Royal Society Wolfson Research Merit Award. *(Corresponding author: Huanhuan Chen.)*

R. G. F. Soares is with the Department of Statistics and Informatics, Federal Rural University of Pernambuco, Recife 52171-900, Brazil (e-mail: rodrigo.gfsoares@ufrpe.br).

H. Chen is with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230000, China (e-mail: hchen@ustc.edu.cn).

X. Yao is with Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: xiny@sustc.edu.cn).

Digital Object Identifier 10.1109/TETCI.2017.2743219

A. Semisupervised Learning

The SSC training set $\mathbf{X} = \mathbf{L} \cup \mathbf{U}$ is formed of L labeled instances $\mathbf{L} = [(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_L, \mathbf{y}_L)]$, where $0 \leq y_{ni} \leq 1$, $i = 1, \ldots, C$ and $\sum_{i=1}^{C} y_{ni} = 1$, C is the number of classes and U unlabeled instances $\mathbf{U} = [\mathbf{x}_{L+1}, \ldots, \mathbf{x}_N]$ with $N = L + U$ and often $U \gg L$. The aim of SSC algorithms is to deliver better generalization than supervised classifiers, which use only labeled data $\mathbf{L}$.

In SSC, there are two types of algorithms: for transductive classifiers, unlabeled training data corresponds to test instances and they cannot generalize predictions to unseen data; while inductive learners are able to generalize predictions to new instances.

SSL algorithms assume that the true class distribution is linked to the training data distribution. There are three common assumptions for semisupervised algorithms in literature [2]. The smoothness assumption (also known as consistency assumption) states that if a pair of instances is close to each other in a high-density region, there will be a higher probability of these points sharing the same label. The cluster assumption assumes that classes are typically separated by a low-density region, that is, if two instances belong to the same cluster, they will likely lie in the same class (also known as the low-density separation assumption [2]). A SSL method can also assume that the true data distribution corresponds to a low-dimensional manifold enclosed in a high-dimensional space, which is known as the manifold assumption. This paper will concentrate on classifiers with the cluster assumption.

B. Ensemble Learning

Ensemble learning consists of training and combining several suitable single classifiers (also known as base learners) in order to mitigate individual errors from such methods and produce better generalization than each base learner independently. Such an approach has the ability to reduce the variance of individual classifiers and strengthen generalization performance [3]. We selected the gradient boosting framework [4] to implement our ensemble algorithm. This algorithm trains an ensemble in a greedy stagewise approach with steepest descent minimization. It allows the optimization of an arbitrary differentiable loss function. Such a method is relatively efficient [5], delivers accurate classifiers and has straightforward instantiation [4].

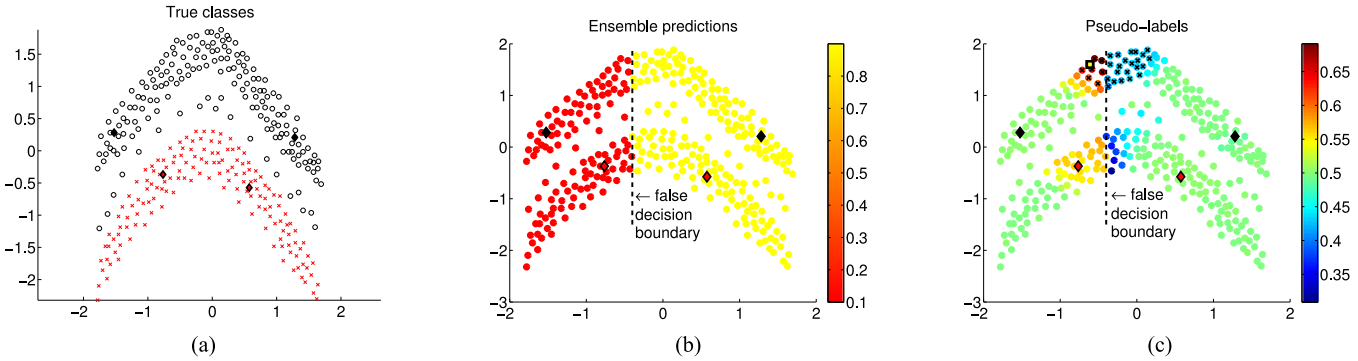

Fig. 1. Two half-moon dataset. ♦ denotes labeled points. (a) True classes. (b) The predefined incorrect decision boundary. (c) The incorrect current label estimates.

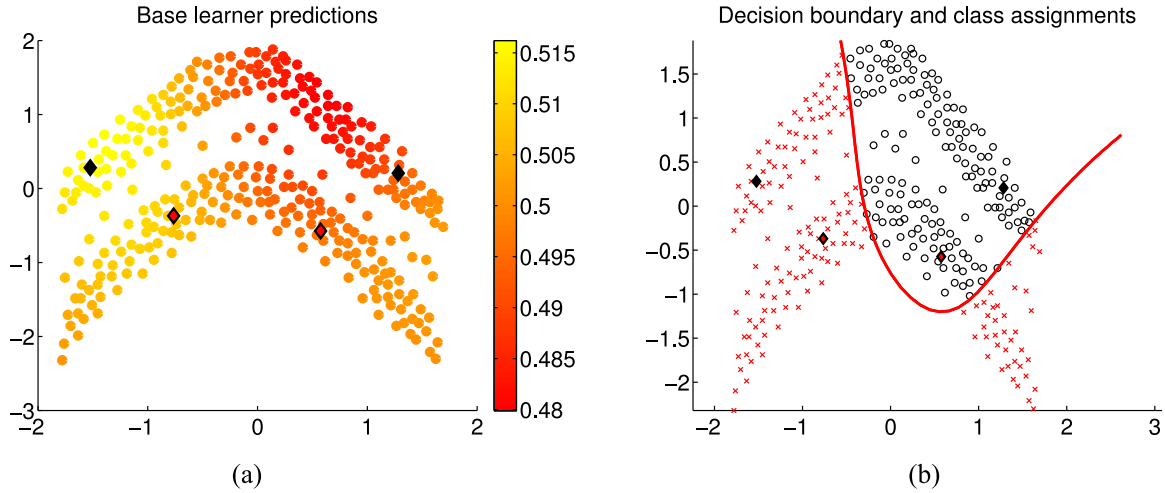

Fig. 2. Supervised base learner produced a poor decision boundary due to incorrect label estimates. (a) Posterior class probabilities of a supervised base classifier trained with label estimates from Fig. 1(c). (b) Resulting decision boundary.

C. Semisupervised Ensembles

Ensemble learning has been employed in both supervised [6] and semisupervised classification [3], [7]. This work employs ensemble learning in SSC. State-of-the-art SSC ensembles use supervised base classifiers that learn estimates of labels as supervised instances. These methods explicitly use unlabeled data only at ensemble level. We introduce an ensemble method that uses semisupervised optimization for training both ensemble and its base classifiers. And we compare its results to ensembles that explicitly employ unlabeled instances for ensemble optimization only.

Existing ensemble classifiers train their base learners with pseudo-labels[1] assigned to unlabeled data [3], [7]. When the training algorithm generates incorrect pseudo-labels, base classifiers trained with such estimates may reinforce errors. This is due to the supervised nature of the employed base classifiers, which learn pseudo-labels as real labels.

We demonstrate the impact of incorrect pseudo-labels in Fig. 1, which illustrates intermediate steps of SSC ensemble training algorithms, such as RegBoost [7] and MCSSB [3]. Fig. 1(a) shows the Two Half-moon dataset, where ♦ represents its only four labeled points. The ensemble produces label estimates for unlabeled points to be employed in the training of the next base classifier. In Fig. 1(b), we arbitrarily generate a very low-quality decision boundary (it is based on an artificial horizontal threshold) that leads to poor current label estimates. Fig. 1(c) depicts the label estimates from the current ensemble. Such pseudo-labels will compose the training set of the next base classifier.

We can clearly notice that the point ■ and its neighbours (marked as ×) should share the label from the top half-moon. However, this inadequate decision boundary leads to incorrect pseudo-labels. Fig. 2(a) and (b) presents the output and the decision boundary produced by a supervised base classifier trained with poor label estimates shown in Fig. 1(C). The newly trained base classifier learned an incorrect decision boundary and is likely to reinforce errors in the remainder of the ensemble. Existing ensemble algorithms in literature have such a drawback, that is, their base learners reinforce early errors produced by the ensemble training.

On the other hand, a semisupervised base learner can tackle incorrect pseudo-labels with the use of the data distribution information [8], instead of learning unreliable estimates as real labels. The use of data distribution can compensate the lack of

[1]Pseudo-labels are estimates of posterior class probabilities artificially assigned to unlabeled instances.

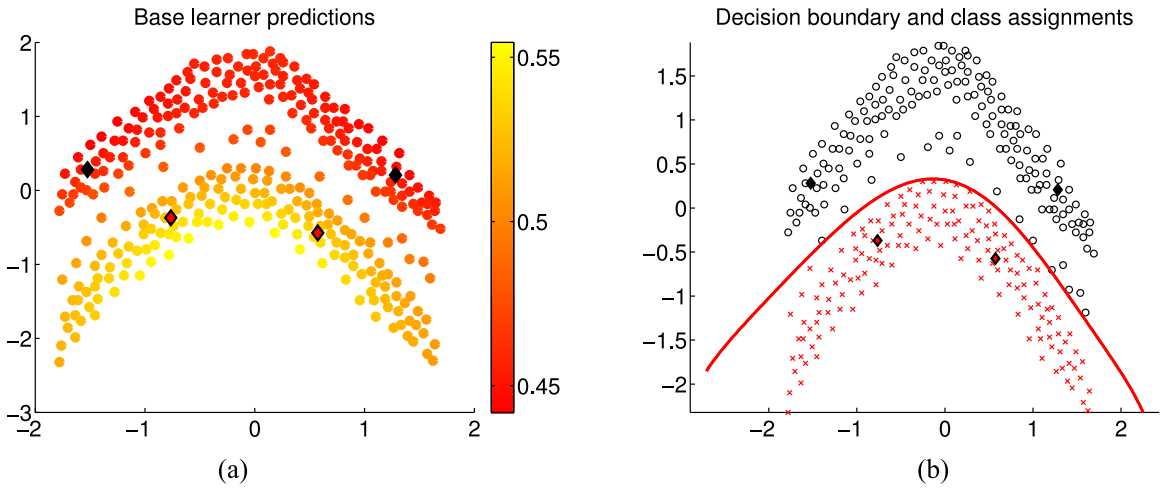

Fig. 3. Label estimates and resulting decision boundary from a semisupervised base learner. (a) Output of semisupervised base learner trained with pseudo-labels from Fig. 2(c). (b) Resulting decision boundary.

correct label estimates. In Fig. 3, we demonstrate how a cluster-based semisupervised algorithm [8] performs in the presence of incorrect pseudo-labels delivered in Fig. 1(c). Fig. 3(a) and (b) shows the output and the decision boundary, respectively, produced by a semisupervised base classifier. Such a cluster-based learner could overcome wrong estimates by avoiding generating a decision boundary in a high-density region. The data distribution information used by the base learner corresponds to the useful knowledge (that supervised base learners do not consider) of such high-density regions. Misleading pseudo-labels produced by the ensemble (Fig. 1(b)) are mitigated by penalising contrasting outputs for similar instances lying in a high-density region.

In particular, the point ■ has neighbours (similar instances denoted as $\times$) of both classes in Fig. 1(c). A cluster-based classifier can consider a weighted average of neighbouring pseudo-labels as a more robust label estimate for instance ■. Such reliable label estimates can lead to a decision boundary that avoids splitting similar instances. If ■ was in a high-density region, the penalization for incorrectly learning that robust pseudo-label would be even greater. The resulting decision boundary is shown in Fig. 3(b).

Our motivation is to tackle intermediate errors produced by ensemble training algorithm. The reinforcement of these errors degrades the predictive performance of an ensemble. Therefore, we propose the Cluster-based Boosting (CBoost), a cluster-based boosting ensemble method for multiclass SSC. This method handles intermediate errors in a more robust manner: it optimizes the cluster assumption on both ensemble and base learners training algorithms. The use of a cluster-based neighbourhood can produce a more robust training of the entire ensemble. Its loss function can effectively avoid generating decision boundaries in dense regions and is able to dilute the influence of noisy labeled instances in such regions.

We instantiated the ClusterReg framework [8] with Radial Basis Functions Networks (RBFN) as the base learners. RBFN networks can be efficient learners [9] and are trained with the Iteratively Reweighted Least Squares (IRLS) [10], as an iterative second-order algorithm is recommended for non-convex multiclass learning [10].

D. Contributions

CBoost contributions are as follows. (i) when the cluster assumption holds, *CBoost* is an effective and stable algorithm in the presence of overlapping classes and scarce labeled points in dense regions. (ii) since the intermediate decision boundaries might not align with cluster boundaries, *CBoost* uses an effective multiclass combination of semisupervised learners. (iii) multiclass ensemble algorithm and its base classifiers jointly perform a cluster-based semisupervised optimization. Ensemble and base learner training methods consider the neighbourhood of an unlabeled data point in order to calculate its desired target, so that the ensemble has stability to overcome potentially incorrect estimates.

Next section reviews existing SSC algorithms. Section III presents the proposed classifier. In Section IV, we show the experimental study and its results. Section V discusses the outcome of this work and Section VI presents its conclusions.

II. Related Works

The combination of several classifiers can deliver better performance when compared to single learners [6]. In this section, we address state-of-the-art semisupervised ensembles.

Semisupervised MarginBoost (SSMB) [11] is a generalization of MarginBoost [12]. At each iteration, it generates label estimates, drawn from the output of the current ensemble, to unlabeled data that will form the training set of the next semisupervised base learner. Its objective function consists of a monotonically decreasing loss function, the supervised and semisupervised margins.

ASSEMBLE [13] is a semisupervised margin-based boosting method. It uses a greedy optimization method to maximize the pseudo-margin. This algorithm employs pseudo-labels to train base learners. Only the most confident instances compose the training sets of base learners. This approach

may increase the margin without improving the current decision boundary. In case of early incorrect pseudo-labels, errors might be reinforced in next base classifiers and degrade generalization.

In [3], the authors proposed Multiclass SemiSupervised Boosting (MCSSB), which is a graph-based ensemble approach. MCSSB combines the similarity of instances with the current ensemble output to generate more reliable pseudo-labels. Its objective function implements all aforementioned SSL assumptions. MCSSB uses supervised base learners.

The algorithm in [14] is a gradient-based extension of the information regularization framework to semisupervised boosting. Its loss function incorporates the cluster, manifold and smoothness assumptions. The authors of [15] also introduced a boosting learner based on AdaBoost for multiclass problems. SemiBoost, MCSSB and [15] used pseudo-labels to train supervised base classifiers.

SSMB, ASSEMBLE and [14] are designed for two-class SSC. Such methods rely on suboptimal decomposition procedures to tackle multiclass datasets. SSMB, ASSEMBLE, and methods in [14] and [15] are largest-margin separators. Such an approach might be unreliable with overlapping classes and to noisy sparse labeled points in dense regions [8].

In [16], the authors introduced a multiclass SSC boosting approach that considers the proportion of classes in a given cluster. A majority class in a cluster can cause the unlabeled data to have the majority class label, which can misguide the boosting optimization. A multi-view margin-based algorithm was proposed in [17]. This algorithm uses priors generated by base learners to train another set of base classifiers. Their loss function can handle noisy priors. However, such a margin-based algorithm is sensitive to noisy classes [8].

RegBoost [7] employs all SSL assumptions in its boosting optimization. The authors propose a new cost functional that consists of a margin cost on labeled data and a regularization term based on unlabeled data. This algorithm minimizes the proposed cost functional with a greedy stagewise functional optimization procedure. It implements the cluster assumption by kernel density estimation. The manifold assumption is addressed by a Gaussian kernel that defines the affinity between instances and their labels. It will penalize a learning machine if it assigns different classes to two neighbouring points lying in a dense region. However, RegBoost might not find good margins when overlapping dense regions exist. Furthermore, it only performs binary classification. Our study confirmed our expectations that RegBoost has lower generalization when applied to real-world multiclass SSC.

ClusterReg [8] is a multiclass cluster-based single classifier. This method employs soft partitions in a regularization technique. When the cluster assumption holds, it is capable of delivering good performance when overlapping classes are present and becomes less sensitive to the small number of labeled points in dense regions.

The aforementioned ensembles can be very confident about the pseudo-labels of unlabeled instances, even though these points may have poor label estimates. Such methods are binary classifiers and depend on decomposition methods that do not exploit classes as mutually exclusive categories. And employing binary learners in multiclass classification requires mitigating imbalanced classes and different output scales from the various learners [3]. These algorithms are sensitive to the position of the few labeled points in a given cluster. Uncertain points have a great impact on the generated decision boundary. Unlabeled instances in low-density regions can also influence the learning methods. The ensembles that rely on pseudo-labels are often sensitive to incorrect pseudo-labelling produced by their base classifiers. Wrongly assigned pseudo-labels are reinforced in future training of base classifiers, which affects the decision boundary of the ensemble. We propose *CBoost* in order to overcome these limitations.

Algorithm 1: Gradient boosting.

Input: Training set $\{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$, number of iterations T and learning rate η.

Output: Predicted targets Z^t.

1: Initialize the ensemble with a constant $Z^0 = 0$.
2: **for** $t = 1$ to T, $n = 1$ to N **do**
3: Find residuals of $\mathcal{L}$ w.r.t. Z_n with rule

$$r_n = -\left[\frac{\partial \mathcal{L}(Z_n^{t-1}, y_n)}{\partial Z_n^{t-1}}\right]$$

4: Fit a base learner z_n to r_n
5: Compute multiplier β^t by solving

$$\beta^t = \text{argmin}_\beta \sum_i \mathcal{L}(Z_n^{t-1} + \beta^t z_n, y)$$

6: Update the ensemble $Z_n^t = Z_n^{t-1} + \eta\beta^t z_n$
7: **end for**

III. *CBoost* Algorithm

In this section, we present our gradient boosting approach. Then, we describe the cluster-based loss function and instantiate the gradient boosting framework. We also introduce an instantiation of ClusterReg for RBFN as the base classifier for *CBoost*.

A. Gradient Boosting

We instantiate gradient boosting in order to implement *CBoost*. In this framework, each base learner is trained with the residuals of the opposite direction of the gradient $r_n = -[\frac{\partial \mathcal{L}(Z_n^{t-1}, y_n)}{\partial Z_n^{t-1}}]$ of a loss function $\mathcal{L}$ (superscript denotes the iteration number). The weight β^t is the result of a line-search $\beta^t = \text{argmin}_\beta \sum_i \mathcal{L}(Z_n^{t-1} + \beta^t z_n, y)$ along the direction produced by a new base classifier. New base learners are added to the ensemble proportionally to β^t with the rule $Z_n^t = Z_n^{t-1} + \eta\beta^t z_n$, where η is a learning rate, which avoids overfitting. At each iteration, gradient boosting finds the steepest descent, trains a base learner that helps the ensemble to follow the gradient, performs a line-search along that direction and includes a base learner that minimizes the loss function. This framework is described in Algorithm 1.

In multiclass classification, the ensemble outputs posterior class probabilities $\mathbf{F} = \{\mathbf{F}_n\}_{n=1}^{N}$, where $\mathbf{F}_n = \{F_{ni}\}_{i=1}^{C}$ and $\sum_{i=1}^{C} F_{ni} = 1$, which is a transformation of the linear combination Z. The residual r_n should be transformed into posterior probabilities $\tilde{\mathbf{y}}_n$. And the multiplier β^t becomes a vector with a weight associated to each class.

B. Multiclass Gradient Boosting Algorithm

We used gradient boosting framework [4] for minimizing the loss function $\mathcal{L}(\mathbf{F}, \mathbf{Y})$, where matrices $\mathbf{F}$ and $\mathbf{Y}$ are $N \times C$, N is the number of instances and C is the number of classes. $\mathbf{F}$ represents the output of the ensemble in the form of posterior class probabilities and Y denotes the desired class probabilities.

CBoost employs the cross-entropy loss function for multiclass classification [10]. In order to produce posterior class probabilities F_{ni}, the linear output signals Z_{ni} of the ensemble are transformed by the softmax function as in (1). This function allows each class, associated to an output node, to be trained dependently to all other nodes, which may improve the decision boundary between one given class and all others. Therefore, $0 \leq F_{ni} \leq 1, i = 1, \ldots, C, \sum_{i=1}^{C} F_{ni} = 1$ and

$$F_{ni} = F(Z_{ni}) = \frac{\exp(Z_{ni})}{\sum_{j}^{C} \exp(Z_{nj})}. \tag{1}$$

We first assign a constant to the initial ensemble $\mathbf{Z}^0$, which is a $N \times C$ matrix, where each Z_{ni} is the linear output of node i for instance n; and Z_{nj} represents the linear outputs of the other nodes j for that instance. At each iteration t, residuals r_{ni} of the gradient descent of the loss function $\mathcal{L}$, w.r.t. the output of the current ensemble Z_{ni}^{t-1}, are assigned to instance n and class i, as shown in (2). These residuals represent the opposite direction of the gradient of $\mathcal{L}$. Feeding these residuals to the base learners allows the ensemble to move the optimization towards the opposite steepest gradient. New base learners are trained with such residuals as labels.

$$r_{ni}^{t} = -\left[\frac{\partial \mathcal{L}(F^{t-1}, Y)}{\partial Z_{ni}^{t-1}}\right] \tag{2}$$

A new base learner z_{ni}, $n = 1, \cdots, N$ and $i = 1, \cdots, C$, is trained with the newly generated pseudo-labels. Since the ensemble is a weighted sum of all base classifiers, we perform a line search for each new base learner with a weight vector $\boldsymbol{\beta}^t = [\beta_1^t, \ldots, \beta_C^t]$ in order to include it in the ensemble. The weight $\boldsymbol{\beta}^t$ represents the influence of the base learner at iteration t in the ensemble. Higher $\boldsymbol{\beta}^t$ denotes a higher generalization performance of such base learner, which should have a higher weight in the predictions produced by the ensemble. The $\boldsymbol{\beta}^t$ for a new base learner is as (3).

$$\boldsymbol{\beta}^t = \text{argmin}_{\boldsymbol{\beta}} \sum_{n}^{N} \sum_{i}^{C} \mathcal{L}\left(F\left(Z_{ni}^{t-1} + \beta_i * z_{ni}\right), y_{ni}\right) \tag{3}$$

A new base learner is added to the ensemble as (4), where η is the learning rate, which reduces the influence of this base classifier on the ensemble and, hence, diminishes overfitting [4]. This equation denotes the stagewise learning of gradient boosting, which adds and weights new base learners trained with the current residuals.

$$Z_{ni}^{t} = Z_{ni}^{t-1} + \eta \beta_i^t z_{ni} \tag{4}$$

A number of greedy gradient descent steps are performed. An increase in the training or validation error rates, or maximum number of iterations T could be stopping criteria.

C. Multiclass Cluster-Based Loss Function

The loss function is composed of supervised loss and cluster regularization [8]. It uses soft partitions generated by clustering algorithms to penalize the assignment of different classes to similar instances in a cluster neighbourhood. The minimization of this function produces good predictors despite very few labeled instances. The trained learners can handle noisy sparse labeled data in dense regions by naturally considering neighbourhood structures arising from soft partitions. Learners are also able to deliver good generalization when classes or clusters overlap [8].

Label estimates of unlabeled instances are weighted averages of current putative labels (these can be either true labels or label estimates) in the neighbourhood of each point. Weights are penalty values and the neighbourhood is defined according to cluster posterior probabilities. The estimated desired label for an unlabeled point follows (5).

$$u_{ni} = \frac{\sum_{k \in \nu(n)} p(\mathbf{q}_k, \mathbf{q}_n) \hat{y}_{ki}}{\sum_{k \in \nu(n)} p(\mathbf{q}_k, \mathbf{q}_n)}, \tag{5}$$

where

$$\hat{y}_{ki} = \begin{cases} y_{ki}, & \text{if } k \text{ is labeled} \\ F_{ki}, & \text{if } k \text{ is unlabeled.} \end{cases}$$

Penalties $p(\mathbf{q}_k, \mathbf{q}_n)$ are calculated according to a soft partition $\mathbf{Q} = [q_{nk}]_{N \times K}$, where $\sum_{i=1}^{K} q_{ni} = 1$ and K is the number of clusters. They are proportional to the cluster membership similarity of a pair of points. Current label estimates u_{ni} are updated at each iteration. $\hat{y}_{ki}$ represents the pseudo-label of neighbour k. If k is labeled, $\hat{y}_{ki}$ is its true label. Otherwise, $\hat{y}_{ki}$ is the ensemble prediction for k. The set $\nu(n)$ denotes the nearest neighbours of n. Then, u_{ni} is a weighted average of current label estimates lying in the neighbourhood of n. The estimated desired label u_{ni} implements the cluster neighbourhood that will help to tackle intermediate errors produced by the ensemble training algorithm, which will lead to a more robust ensemble learning.

Equation (6) maps the similarity, $s(\mathbf{q}_n, \mathbf{q}_k)$, between instances n and k into penalization. As the similarity measure $s(\mathbf{q}_n, \mathbf{q}_k))$, we adopt the product of the inverse of Euclidean distance and Pearson correlation between vectors $\mathbf{q}_n$ and $\mathbf{q}_k$. Such a measure will ensure that similar instances have close and highly correlated cluster memberships.

$$p(\mathbf{q}_n, \mathbf{q}_k) = \sin\left(\frac{\pi}{2}\left(s(\mathbf{q}_n, \mathbf{q}_k)\right)^{\kappa}\right). \tag{6}$$

The steepness of $p(\mathbf{q}_n, \mathbf{q}_k)$ is controlled by the parameter κ. This technique is flexible and allows various degrees

of penalization: higher κ signifies that only most similar instances will have high penalty, while low values will also penalize less similar points. It regulates the degree in which the training should avoid decision boundaries in dense regions. Higher κ relaxes the cluster assumption by allowing decision boundaries in high-density regions, whereas lower values force the classifier to avert decision boundaries that split clusters.

The nearest neighbours of n are the V instances with the highest $p(\mathbf{q}_k, \mathbf{q}_n)$ scores. In this sense, the soft partition helps to implement the cluster assumption by producing a cluster-based neighbourhood structure.

Equation (7) shows the *CBoost* objective function with cross entropy.

$$\mathcal{L}(\mathbf{F}, \mathbf{y}) = -\sum_{n=1}^{N}\sum_{i=1}^{C}\left\{\frac{I_{nL}}{L} y_{ni}\log\left(F_{ni}\right) + \frac{I_{nU}\lambda}{U} m(\mathbf{q}_n) u_{ni}\log\left(F_{ni}\right)\right\}, \tag{7}$$

where I_{nL} and I_{nU} are binary masks that indicate if n is either labeled or unlabeled, respectively. F_{ni} is the predicted class probability for class i and instance n. The parameter λ controls the magnitude of the semisupervised regularization. $m(\mathbf{q}_n)$ is the highest score in $\mathbf{q}_n$, and it indicates the most likely cluster for point n.

Regarding the smoothness assumption, the regularization term in (7) regularizes the classifier when it predicts different classes to a pair of similar points, as denoted by $-u_{ni}\log\left(F_{ni}\right)$. *CBoost* implements the cluster assumption by calculating putative labels u_{ni} with the weighting of influences of neighbouring labels $\hat{y}_{ki}$ according to their density information in $\mathbf{Q}$. Besides u_{ni}, the probability $m(\mathbf{q}_n)$ is an estimate of the density of instance n. It also helps to weight the impact of point n in the training. The penalty function will regularize the optimization if it delivers distinct predictions to instance n and its neighbour k (in case the classifier generates a decision boundary between these potentially similar points). The regularization will have an even greater impact on the training if n lies in a dense region. Thus, the algorithm will avert producing decision boundaries that divide clusters.

D. Multiclass Boosting With Cluster Regularization

The initial ensemble, $\mathbf{Z}^0$, consists of a base classifier trained with label estimates produced by the initialization method in Section III-B.

A new base classifier $\mathbf{f}$ is trained with the current residuals r_{nj}, which are the opposite direction of the gradient $\mathcal{L}(\mathbf{F}_n^{t-1}, \mathbf{y}_n)$ w.r.t. Z_{nj}^{t-1}. Each new base classifier helps to minimize the ensemble loss function $\mathcal{L}$ by minimizing the same $\mathcal{L}$ w.r.t. r_{nj}. Residuals are as in (8).

$$r_{nj} = -\frac{I_{nL}}{L} * \left(F_{nj}^{t-1} - y_{nj}\right) - \frac{I_{nU}\lambda m(\mathbf{q}_n)}{U} * \left(F_{nj}^{t-1} - u_{nj}\right) \tag{8}$$

Algorithm 2: *CBoost* algorithm.

Input: $\mathbf{X} = \mathbf{L} \cup \mathbf{U}$, $\mathbf{L} = [(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_L, \mathbf{y}_L)]$, $\mathbf{U} = [\mathbf{x}_{L+1}, \ldots, \mathbf{x}_N]$ and $N = L + U$, often $U \gg L$.

Output: Posterior class probabilities $\mathbf{F}^T$.

1: Train base learner f_{nj} with initial pseudo-labels $\hat{y}_{nj}$ as in (16).
2: Assign initial ensemble $Z_{nj}^0 = z_{nj}$ and $F_{nj}^0 = \text{softmax}(Z_{nj}^0)$.
3: **for** $t = 1$ to T, $n = 1$ to N and $j = 1$ to C **do**
4: Calculate $\hat{y}_{nj}$ using F_{nj}^t as in (2).
5: Calculate class probabilities u_{nj} for unlabeled instances with (5).
6: Calculate residuals of $\mathcal{L}$ w.r.t. Z_{nj} with (8).
7: Calculate pseudo-labels $\tilde{y}_{nj} = \text{softmax}(r_{nj})$.
8: Train semisupervised learner f_{nj} to $\tilde{y}_{nj}$.
9: Find multiplier β_j^t using (3).
10: Update ensemble and its posterior class probabilities as

$$Z_{nj}^t = Z_{nj}^{t-1} + \eta\beta_j^t z_{nj}$$
$$F_{nj}^t = \text{softmax}(Z_{nj}^t)$$

11: **end for**

The residuals r_{nj} are mapped into posterior class probabilities, $\tilde{y}_{nj} = \text{softmax}(r_{nj})$, which are employed for training the next base learner $\mathbf{f}$.

Since there is no closed form in the line search for (3), we optimize the base learner weight $\boldsymbol{\beta}^t$ with a single Newton-Raphson step [4], as shown in (9).

$$\beta_j^t = -\mathbf{H}^{-1} * \left[\frac{\partial\mathcal{L}(\mathbf{F}^{t-1}, \mathbf{y})}{\partial\beta_j}\right]. \tag{9}$$

We derive β_j^t w.r.t $\mathbf{F}_n^{t-1}$ since initially $\beta_j^t = 0$ and hence $\mathbf{F}_n^t = \mathbf{F}_n^{t-1}$. The gradient of each instance is

$$\left[\frac{\partial\mathcal{L}(\mathbf{F}_n^{t-1}, \mathbf{y}_n)}{\partial\beta_j^t}\right] = \frac{I_{nL}}{L}\left(F_{nj}^{t-1} - y_{nj}\right) z_{nj} + \frac{I_{nU}\lambda m(\mathbf{q}_n)}{U}\left(F_{nj}^{t-1} - u_{nj}\right) z_{nj}, \tag{10}$$

and Hessian matrix $\mathbf{H}$ is

$$H_{jk} = \sum_{n=1}^{N}\left(\frac{I_{nL}}{L} + \frac{I_{nU}\lambda m(\mathbf{q}_n)}{U}\right) * F_{nj}^{t-1}(\delta_{jk} - F_{nk}^{t-1}) * z_{nj} z_{nk}, \tag{11}$$

where $\delta_{jk} = 1$ if $j = k$ and 0 otherwise.

The ensemble is updated as (4) until a stopping criterion is met, e.g. increase of validation error. Ensemble outputs are also transformed into posterior class probabilities by Equation (1). *CBoost* is outlined in Algorithm 2.

E. RBFN as Base Learner

We chose RBFN as the base classifier as it can be efficient and easily adapted to our method. In our preliminary experiments, it

was faster than Multilayer Perceptron (MLP) networks. Its loss function is as (7) with the addition of a weight regularization term. The output function is $f_{nj} = \text{softmax}(z_{nj})$. Equation (13) shows the loss function for RBFN.

$$\mathcal{L}(\mathbf{f}, \mathbf{y}) = -\sum_{n=1}^{N}\sum_{i=1}^{C}\left\{\frac{I_{nL}}{L} y_{ni}\log\left(f_{ni}\right) + \frac{I_{nU}\lambda m(\mathbf{q}_n)}{U} u_{nj}\log\left(f_{ni}\right) - \alpha\frac{\mathbf{w}_i^T\mathbf{w}_i}{2}\right\}, \quad (12)$$

where $\mathbf{w}_i$ is the weight vector for node i and α regulates the weight decay.

Since there is no closed form of RBFN training algorithm for multiclass classification, we use the IRLS method to train the RBFN [10]. The weights are updated by several Newton-Raphson steps as in (13).

$$\Delta\mathbf{w}_j = -\mathbf{H}^{-1} * \left[\frac{\partial\mathcal{L}(\mathbf{f}, \mathbf{y})}{\partial\mathbf{w}_j}\right], \quad (13)$$

where $\mathbf{H}$ denotes the Hessian matrix and $[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_j}]$ is the gradient vector. Equation (14) shows the gradient of $\mathcal{L}$ w.r.t. weight $\mathbf{w}_j$.

$$\left[\frac{\partial\mathcal{L}(\mathbf{f}_n, \mathbf{y}_n)}{\partial\mathbf{w}_j}\right] = \frac{I_{nL}}{L}\left(f_{nj} - y_{nj}\right)\Phi_n + \frac{I_{nU}\lambda m(\mathbf{q}_n)}{U}\left(f_{nj} - u_{nj}\right)\Phi_n + \alpha\mathbf{w}_j \quad (14)$$

where Φ_n is the output column vector of hidden nodes.

The Hessian in (13) is a block matrix $\mathbf{H} = [H_{jk}]_{MC\times MC}$ where M is the number of RBF centres and each block is

$$H_{jk} = \left[\frac{\partial^2\mathcal{L}}{\partial\mathbf{w}_j\partial\mathbf{w}_k}\right] = \sum_{n=1}^{N}\left\{\left(\frac{I_{nL}}{L} + \frac{I_{nU}\lambda m(\mathbf{q}_n)}{U}\right) * f_{nj}(\delta_{jk} - f_{nk}) * \Phi_n\Phi_n^T + \alpha\right\}. \quad (15)$$

Weights are trained with (13) until a stopping criterion is met, e.g. validation error starts to increase.

F. Initialization Procedure

The initialization procedure assigns initial label estimates to unlabeled points [7] in order to train the first base learner. Pseudo-labels of instances in a given cluster are weighted averages of true labels present in that dense region. Weights are $p(\mathbf{q}_n, \mathbf{q}_k)$. If no labeled point lies in a cluster, classes will have equal probabilities. We aim to provide the algorithm with more reliable initial pseudo-labels than simple equal probabilities. For each cluster Ψ, class i and unlabeled point n, the estimated desired labels $\hat{y}_{ni}$ for unlabeled data is:

$$\hat{y}_{ni} = \frac{\sum_{k\in\Psi} I_{kL} * p(\mathbf{q}_n, \mathbf{q}_k) * y_{ki}}{\sum_{k\in\Psi} I_{kL} * p(\mathbf{q}_n, \mathbf{q}_k)}. \quad (16)$$

Most of exiting semisupervised classifiers assign similar labels to all unlabeled instances, hence the training error might be extremely low. This is caused by their loss function comparing the predicted output with itself. This initialization procedure avoids this problem by using the distribution of labeled instances in a given cluster.

IV. Experiments

We present experimental studies performed with both transductive and inductive approaches using artificial and real-world datasets.

A. Parameter Tuning

Parameter tuning is performed with a grid search and 10-fold cross-validation. The highest generalization is presented. In order to provide fairness of comparison, we follow the tuning suggestions of the authors of each existing algorithm analysed in this work in both inductive and transductive scenarios.

As MCSSB [3] implements three SSC assumptions, it is expected that *CBoost* outperforms MCSSB solely on problems where there is a relevant cluster structure. We chose decision trees as base learners and $C = 10000$ as suggested in [3]. Our preliminary experiments confirmed that these values produce best generalization. The proportion σ of distances employed to calculate the kernel was searched in $\{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.5, 0.8, 1\}$. The sample size s was searched in $\{0.1, 0.5, 0.8, 1\}$. We used 20 and 50 base classifiers in the experiments.

For RegBoost [7], the number of neighbours was searched in $\{3, 4, 5, 6\}$. The number of iterations was either 20 or 50. The resampling rate was fixed at 0.1 for the first iteration. And, for the remaining iterations, this parameter was searched in $\{0.1, 0.25, 0.5\}$. We selected SVM as its base learner following the recommendation in [7].

We analyse *CBoost* in two versions: *CBoost-Semi* is the proposed ensemble and *CBoost-Sup* is a similar boosting approach with supervised base learners. Both versions have analogous parameter tuning. We also compare *CBoost-Semi* to a single ClusterReg with RBFN. Its parameters are selected following the settings of the base classifiers from *CBoost-Semi*.

The trade-off λ can be in $[0, 1]$ and optimized in $\{0.2, 0.4, 0.6, 0.8, 1\}$. For each base learner, λ was uniformly drawn from $[0.2, 1]$ to provide diversity to the ensemble.

Parameter V should be 30 for most datasets due to their size. With $V = 30$ in datasets with less than 1500 instances, *CBoost-Semi* can exploit a comprehensive amount of labels in the neighbourhood of a point. We can set V to 2% of the number of points for datasets with more than 1500 instances.

Self-Tuning Spectral Clustering (STSC) [18] produces cluster memberships $\mathbf{Q}$ with arbitrary shapes. We also used K-means [19] for transductive setting.

We should set the number of clusters K to, at least, the number of classes [8]. When the partition produced by the clustering algorithm does not resemble the class structure, the number of clusters can be increased. In this case, a single class will be formed of multiple clusters. The classifier will avert generating contrasting labels in such dense region and may produce a decision boundary that does not split that class. K was searched

TABLE I
SUMMARY OF TUNED PARAMETERS FOR *CBoost*

Tuned Parameters of *CBoost*	
Clustering algorithm (produce matrix $\mathbf{Q}$)	K-means and STSC
λ (controls the amount of regularization)	Grid search in $\{0.2, 0.4, 0.6, 0.8, 1\}$
K (number of clusters)	Grid search with $\{1, 2, 3, 4\}$ times the number of classes

in $1, \ldots, 4$ times the number of classes. We fixed κ at 5 for all settings. Further tuning of κ can be performed.

The centres of the hidden nodes of the RBFN were the instances in the training set. At each iteration, centre widths were randomly chosen from between 20% and 80% of the median of pairwise Euclidean distances among all training instances. The weight decay α was randomly selected from $[0.2, 0.5]$ and λ was randomly chosen from $[0.2, 1]$.

We used 20 base learners, η was fixed at 0.5 and the number of IRLS iterations was 50. The tuning of the remaining parameters of RBFN follows the respective settings of *Cboost-Semi*. The parameter tuning is summarized in Table I.

B. Transductive Setting

With the transductive setting, we demonstrate the superior predictive accuracy of *CBoost-Semi* over single classifiers and the influence of using semisupervised base learners in a semisupervised ensemble. Thus, we compare *CBoost-Semi* to ClusterReg, *CBoost-Sup* and other state-of-the-art classifiers.

1) Datasets: In [2], the authors designed several transductive benchmarks. Among those, we used three artificial (g241c, g241d and Digit1) and three real-world datasets (USPS, COIL and BCI) to assess the generalization of these algorithms. The cluster assumption holds in g241c, that is, its class distribution corresponds to clusters. Whilst g241d has a misleading cluster structure and does not have manifolds. Digit1 possesses a manifold structure and does not hold the cluster assumption. USPS has cluster and low-dimensional manifold structures. All datasets are binary problems with 1500 instances and 241 attributes. The exceptions are BCI, that has 400 instances and 114 dimensions, and COIL with six classes. Further details of these datasets are described in [2, Ch 21]. The datasets have 12 subsets of 10 and 100 labeled points.

2) Algorithms and Results: Algorithms were trained with 12 disjoints subsets with distinct 10 and 100 labels. The average error of each algorithm was reported. We compared *CBoost-Sup* and *CBoost-Semi* with various methods described in [2], [7], [8], [20]. As suggested in [2], test sets are predefined and we directly compare the average of predictive errors. Further details of these learners are in [2, Ch 21] and [7], [8], [20].

Tables II and III present the results. Algorithms are grouped by the implemented assumptions: manifold and cluster assumptions, and ensembles of multiple assumptions. Fig. 4 presents box plots of the test errors of these classifiers. It shows the superior generalization of *CBoost-Semi* in comparison to ClusterReg and *CBoost-Sup*.

C. Inductive Setting

Inductive algorithms can classify unseen data points. In this scenario, we evaluate the generalization ability of *CBoost*, ClusterReg, MCSSB and RegBoost.

1) Datasets: We selected 13 datasets from the UCI machine learning repository [21]. Table IV summarizes these datasets.

We generated three versions of each dataset. Each version has 5%, 10% or 20% of labeled data. In order to use these versions as semisupervised problems, we uniformly drawn the respective amount of points to be labeled. Each version has a distinct set of labeled instances. In this sense, each version of a dataset presents itself as a particular SSC task. We used a 10-fold cross-validation procedure for all datasets.

2) Algorithms and Results: We compare *CBoost-Semi* to ensemble methods, MCSSB and RegBoost, which use all SSC assumptions. We also study ClusterReg with RBFN to assess the difference in performance of *CBoost-Semi* over a single classifier. Experiments with *CBoost-Sup* help to elucidate the impact of using semisupervised base classifiers instead of fully supervised base learners.

Table V(a) and (b) presents the predictive results of the evaluated classifiers. Specifically, Table V(a)–(c) shows the obtained results for different proportions of labeled data: 5%, 10% and 20%, respectively. We performed pairwise t-tests with 95% of significance level to compare methods. The $\bullet/\circ$ denote whether *CBoost-Semi* was significantly superior or inferior to other algorithms. The scores of wins, ties and losses are the number of cases where *CBoost-Semi* was statistically superior, comparable or inferior to the other classifiers. The Friedman test [22] with 5% of significance provided statistical evidence of the difference between the means of errors in Table V. After the Friedman test, we performed the Bonferroni-Dunn test [22] with 5% of significance level. Such a post-hoc test confirmed that *CBoost-Semi* was superior to all other algorithms, including state-of-the-art methods, across all amounts of labeled data.

D. Efficiency Investigation

We studied the time efficiency of *CBoost-Semi*, ClusterReg and *CBoost-Sup*. Fig. 5 reports the average time and standard deviation of the executions that produced the highest generalization in Table V(a)–(c). We show the CPU time spent on problems with 5%, 10% and 20% of labels.

We used an Intel Core 2 Quad CPU Q8200 with 2 gigabytes of RAM. All implementations were in Matlab. Implementations of *CBoost* can be further optimized.

V. DISCUSSIONS

Tables II and III demonstrate that *CBoost-Semi* was superior to other methods (as in g241c, for 10 and 100 labeled instances). This fact demonstrates that *CBoost-Semi* was able to properly employ data distribution, estimated by clustering algorithm, to produce a classifier with robustness to the few labeled instances

TABLE II
MEAN OF THE PERCENTAGE OF PREDICTION ERRORS WITH 12 DISJOINT SUBSETS OF 10 LABELED INSTANCES

Algorithm	g241c	g241d	Digit1	USPS	COIL	BCI	Text
Manifold-based algorithms							
1NN	44.05	43.22	23.47	19.82	65.91	48.74	39.44
MVU+1NN	48.68	47.28	11.92	14.88	65.72	50.24	39.40
LEM+1NN	47.47	45.34	12.04	19.14	67.96	49.94	40.48
QC+CMN	39.96	46.55	9.80	**13.61**	59.63	50.36	40.79
Discrete Reg.	49.59	49.05	12.64	16.07	63.38	49.51	40.37
SGT	**22.76**	**18.64**	8.92	25.36	*n/a*	49.59	**29.02**
Laplacian RLS	43.95	45.68	**5.44**	18.99	**54.54**	48.97	33.68
CHM (normed)	39.03	43.01	14.86	20.53	*n/a*	**46.90**	*n/a*
Cluster-based algorithms							
SVM	47.32	46.66	30.60	20.03	68.36	49.85	45.37
TSVM	24.71	50.08	17.77	25.20	67.50	49.15	31.21
Cluster-Kernel	48.28	42.05	18.73	19.41	67.32	48.31	42.72
Data-Rep. Reg.	41.25	45.89	12.49	17.96	63.65	50.21	*n/a*
LDS	28.85	50.63	15.63	**15.57**	**61.90**	49.27	**27.15**
ClusterReg (MLP)	**16.90**	40.82	12.06	19.42	65.51	**45.36**	40.48
ClusterReg (RBFN)	26.94	**27.95**	**10.64**	19.98	69.13	49.19	40.48
Ensembles and multiple-assumptions algorithms							
AdaBoost	40.12	43.05	28.92	25.57	71.16	47.08	47.42
SAMME	50.09	50.07	50.07	19.98	70.25	50.30	*n/a*
ASSEMBLE	40.62	44.41	23.49	21.77	65.49	48.96	49.13
RegBoost	38.22	42.90	17.94	**17.41**	65.39	**46.73**	**34.96**
CBoost-Sup	44.65	45.76	15.64	19.98	77.61	47.37	44.49
CBoost-Semi	**22.76**	**23.07**	**14.72**	19.98	**64.33**	48.50	43.77

Note: Test sets are fixed. As in [2, Ch. 21], we present only the mean errors of these results. Bold face indicates the highest predictive performance within each group of classifiers. And *n/a* is unavailable results in [2, Ch. 21].

TABLE III
MEAN OF THE PERCENTAGE OF PREDICTION ERRORS WITH 12 DISJOINT SUBSETS OF 100 LABELED INSTANCES

Algorithm	g241c	g241d	Digit1	USPS	COIL	BCI	Text
Manifold-based algorithms							
1NN	40.28	37.49	6.12	7.64	23.27	44.83	30.77
MVU+1NN	44.05	43.21	3.99	6.09	32.27	47.42	30.74
LEM+1NN	42.14	39.43	**2.52**	6.09	36.49	48.64	30.92
QC+CMN	22.05	28.20	3.15	6.36	10.03	46.22	25.71
Discrete Reg.	43.65	41.65	2.77	**4.68**	**9.61**	47.67	24.00
SGT	**17.41**	**9.11**	2.61	6.80	*n/a*	45.03	**23.09**
Laplacian RLS	24.36	26.46	2.92	**4.68**	11.92	**31.36**	23.57
CHM (normed)	24.82	25.67	3.79	7.65	*n/a*	36.03	*n/a*
Cluster-based algorithms							
SVM	23.11	24.64	5.53	9.75	22.93	34.31	26.45
TSVM	18.46	22.42	6.15	9.77	25.80	**33.25**	24.52
Cluster-Kernel	13.49	4.95	3.79	9.68	21.99	35.17	24.38
Data-Rep. Reg.	20.31	32.82	**2.44**	5.10	**11.46**	47.47	*n/a*
LDS	18.04	28.74	3.46	**4.96**	13.72	43.97	**23.15**
ClusterReg (MLP)	**13.38**	**4.36**	3.45	5.25	24.73	33.92	32.09
ClusterReg (RBFN)	19.54	17.07	7.20	16.53	36.35	48.11	32.09
Ensembles and multiple-assumptions algorithms							
AdaBoost	24.82	26.97	9.09	9.68	22.96	24.02	26.31
SAMME	36.75	38.70	19.55	16.94	53.79	41.64	*n/a*
ASSEMBLE	27.19	27.42	6.71	8.12	21.84	28.75	27.77
RegBoost	20.54	23.56	4.58	**6.31**	**21.78**	**23.69**	**23.25**
CBoost-Sup	20.92	28.35	4.87	8.78	63.78	40.25	30.76
CBoost-Semi	**12.71**	**6.99**	**4.34**	7.20	30.67	38.83	25.58

Note: Test sets are fixed. As in [2, Ch. 21], we present only the mean errors of these results. Bold face indicates the highest predictive performance within each group of classifiers. And *n/a* is unavailable results in [2, Ch. 21].

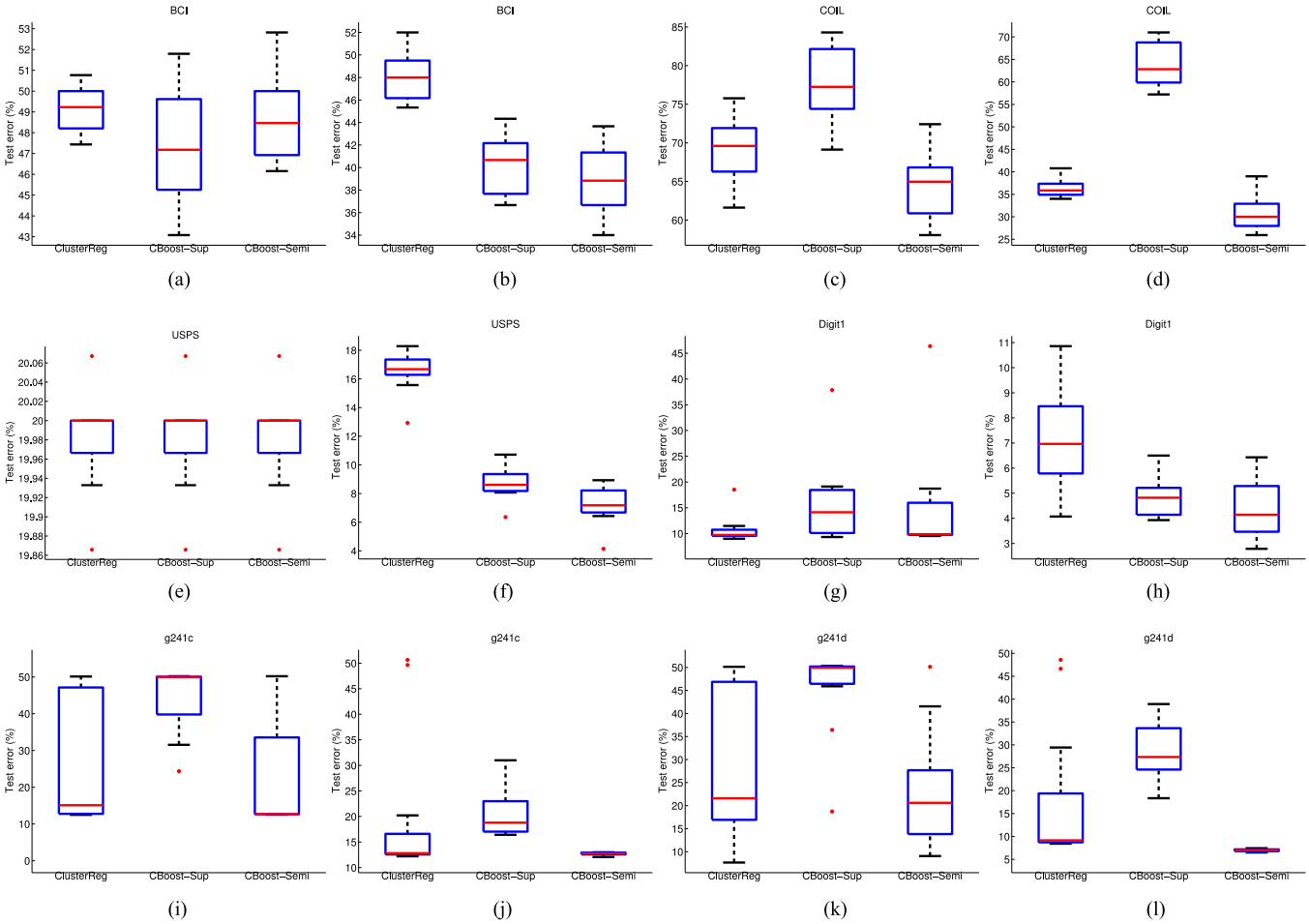

Fig. 4. Box plot of test errors (%) of ClusterReg, *CBoost-Sup* and *CBoost-Semi*. (a) BCI - 10 labels. (b) BCI - 100 labels. (c) COIL - 10 labels. (d) COIL - 100 labels. (e) USPS - 10 labels. (f) USPS - 100 labels. (g) Digit1 - 10 labels. (h) Digit1 - 100 labels. (i) g241c - 10 labels. (j) g241c - 100 labels. (k) g241d - 10 labels. (l) g241d - 100 labels.

available. The proposed method was also superior to existing ensemble methods. Differently from other algorithms, *CBoost-Semi* was able to tackle poor label estimates.

CBoost-Semi could overcome misleading cluster structures (as in g241d) by using the few available labeled instances more effectively than all other algorithms (except for Cluster-Kernel on g241d with 100 labeled instances).

As expected, methods with manifold assumption produced the highest accuracy for Digit1 and USPS with 10 and 100 labels. Nevertheless, *CBoost-Semi* obtained superior generalization compared to other cluster-based algorithms. *CBoost-Semi* also produced comparable generalization with other cluster-based and multiple-assumptions methods for USPS. In this sense, when the assumed data distribution did not relate to the class distribution, the proposed method was able to weight the impact of labeled data more severely and dilute the misguiding cluster assumption.

The superior performance of manifold-based algorithms in COIL in comparison with cluster-based methods indicates that COIL has an underlying manifold that relates to the classes. As RegBoost implements all SSC assumptions, its performance in BCI suggests that this dataset has relevant cluster and manifold structures [7].

TABLE IV
SUMMARY OF DATASETS

Datasets	# instances	# attributes	# classes
Australian credit	690	14	2
BUPA	345	6	2
Contraceptive	1473	9	3
Dermatology	366	34	6
Ecoli	336	6	5
Glass	214	9	6
Horse Colic	368	27	2
House Votes	435	16	2
Ionosphere	351	34	2
SPECT	267	22	2
Statlog	846	18	4
Yeast	1479	8	9
WDBC	569	32	2

According to Fig. 4, *CBoost-Semi* was statistically more accurate than ClusterReg in most problems, especially for g241d and COIL, where there was no meaningful cluster structure. When cluster assumption held, *CBoost-Semi* also produced higher accuracy. This indicates that ensemble approaches can also minimize errors (variance) of individual classifiers in SSC.

TABLE V
MEAN AND STANDARD DEVIATION (%) OF PREDICTION ERRORS ON 10-FOLD CROSS-VALIDATION WITH (A) 5% , (B) 10% AND (C) 20% OF LABELED INSTANCES

Datasets	MCSSB	RegBoost	ClusterReg	*CBoost-Sup*	*CBoost-Semi*
australian	44.52 ± 4.87 •	18.15 ± 3.74	41.88 ± 17.14 •	21.47 ± 3.47 •	18.67 ± 1.26
bupa	38.91 ± 10.85 ○	47.45 ± 10.83 •	30.50 ± 2.21 ○	49.59 ± 9.77 •	38.90 ± 4.90
contraceptive	57.07 ± 4.59 •	67.76 ± 8.20 •	49.85 ± 1.27 ○	49.38 ± 4.02 ○	52.74 ± 2.84
dermatology	11.12 ± 5.82 •	58.24 ± 5.63 •	23.40 ± 7.44 •	19.80 ± 4.87 •	5.34 ± 5.31
ecoli	18.66 ± 5.96 •	37.62 ± 6.83 •	16.54 ± 4.74 •	14.59 ± 4.25 •	11.68 ± 2.81
glass	60.31 ± 10.60 •	77.53 ± 17.87 •	58.40 ± 9.29 •	38.96 ± 12.01	36.31 ± 10.36
horse-colic	30.38 ± 10.08	48.44 ± 19.57 •	31.06 ± 5.61 •	25.87 ± 6.27	26.23 ± 6.17
house-votes-84	61.57 ± 7.24 •	56.10 ± 12.64 •	7.81 ± 3.06	10.73 ± 3.69 •	7.84 ± 2.30
ionosphere	35.64 ± 12.78 •	50.55 ± 19.80 •	12.97 ± 2.51	9.05 ± 2.04	13.35 ± 7.78
spect	79.51 ± 10.71 •	31.99 ± 4.27 •	11.09 ± 1.78	10.69 ± 2.74	11.08 ± 3.12
statlog	49.47 ± 6.09 •	69.71 ± 5.89 •	52.11 ± 5.51 •	45.60 ± 3.18 •	35.33 ± 5.59
yeast	56.58 ± 3.03 •	68.63 ± 3.68 •	53.35 ± 2.12 •	53.06 ± 1.76 •	48.78 ± 0.99
WDBC	37,25 ± 5,37 •	18,93 ± 5,67 •	8,69 ± 1,17 •	7,02 ± 1,86	6,86 ± 2,68
Win/Tie/Loss	11/2/0	12/1/0	8/3/2	7/5/1	–

(a) Results for 5% of labeled data.

Datasets	MCSSB	RegBoost	ClusterReg	*CBoost-Sup*	*CBoost-Semi*
australian	44.58 ± 6.90 •	13.38 ± 2.54 ○	12.76 ± 1.46 ○	19.96 ± 2.67 •	16.18 ± 2.73
bupa	43.64 ± 9.92 •	47.11 ± 12.00 •	33.22 ± 2.43 •	26.80 ± 3.20 •	23.55 ± 4.31
contraceptive	53.35 ± 3.51 •	61.00 ± 4.59 •	45.71 ± 1.91	45.37 ± 2.97	46.65 ± 1.80
dermatology	9.97 ± 6.31 •	69.25 ± 5.95 •	20.12 ± 6.85 •	5.49 ± 4.84 •	1.26 ± 1.64
ecoli	18.59 ± 6.63	35.11 ± 7.51 •	18.90 ± 5.93	25.98 ± 4.51 •	19.17 ± 4.57
glass	52.54 ± 11.18 •	67.30 ± 12.24 •	43.09 ± 9.44 •	19.01 ± 7.31	19.54 ± 4.36
horse-colic	25.35 ± 9.32	57.12 ± 18.39 •	30.10 ± 6.89 •	23.45 ± 5.23	22.52 ± 5.19
house-votes-84	61.35 ± 8.08 •	58.12 ± 11.63 •	11.76 ± 1.24 •	4.86 ± 1.93 •	1.78 ± 1.23
ionosphere	35.90 ± 6.75 •	44.85 ± 15.40 •	10.48 ± 2.08 •	6.41 ± 4.06	8.27 ± 2.20
spect	79.60 ± 8.61 •	49.55 ± 32.36 •	15.45 ± 1.49 •	12.12 ± 1.06	11.70 ± 1.58
statlog	43.46 ± 7.23 •	74.44 ± 2.74 •	55.63 ± 3.75 •	49.38 ± 3.32 •	37.90 ± 2.31
yeast	53.90 ± 3.70 •	68.63 ± 2.94 •	52.09 ± 3.50 •	50.26 ± 0.98 •	47.57 ± 1.66
WDBC	37,37 ± 7,19 •	13,86 ± 6,47 •	2,77 ± 1,49 ○	5,12 ± 1,99	5,31 ± 2,05
Win/Tie/Loss	11/2/0	12/0/1	9/2/2	7/6/0	–

(b) Results for 10% of labeled data.

Datasets	MCSSB	RegBoost	ClusterReg	*CBoost-Sup*	*CBoost-Semi*
australian	44.34 ± 7.04 •	17.37 ± 5.21	16.14 ± 3.12	18.35 ± 3.03 •	15.82 ± 3.44
bupa	38.25 ± 10.96 •	52.16 ± 11.77 •	20.41 ± 5.00	20.24 ± 6.20	21.22 ± 4.65
contraceptive	54.15 ± 6.38 •	57.22 ± 6.41 •	45.80 ± 3.09 •	48.45 ± 6.44 •	43.52 ± 2.12
dermatology	6.52 ± 3.99	59.61 ± 8.20 •	14.71 ± 4.92 •	3.54 ± 3.00	4.13 ± 2.24
ecoli	17.59 ± 7.73	37.52 ± 13.14 •	18.37 ± 3.34 •	12.64 ± 4.19	12.93 ± 7.42
glass	61.69 ± 12.82 •	67.06 ± 9.44 •	19.42 ± 6.93	18.84 ± 6.88	20.32 ± 7.66
horse-colic	40.87 ± 10.84 •	47.22 ± 14.98 •	37.05 ± 3.09 •	29.11 ± 4.99	29.12 ± 5.04
house-votes-84	61.29 ± 7.43 •	50.04 ± 10.84 •	6.87 ± 2.88 •	9.68 ± 5.17 •	3.11 ± 1.99
ionosphere	36.03 ± 10.85 •	38.46 ± 13.65 •	8.59 ± 1.78	8.27 ± 1.85	8.59 ± 1.78
spect	79.53 ± 5.20 •	30.85 ± 12.03 •	8.07 ± 2.53	8.07 ± 2.53	8.23 ± 4.06
statlog	33.47 ± 4.32	72.05 ± 5.28 •	50.83 ± 5.46 •	37.02 ± 4.43	34.26 ± 4.72
yeast	52.47 ± 4.27 •	68.65 ± 2.65 •	51.35 ± 2.79 •	49.54 ± 2.95 •	46.57 ± 2.61
WDBC	37,28 ± 6,42 •	28,99 ± 5,33 •	1,32 ± 1,14	2,41 ± 1,65	1,97 ± 1,28
Win/Tie/Loss	10/3/0	12/1/0	7/6/0	4/9/0	–

(c) Results for 20% of labeled data.

Symbols •/○ denote significantly superior and inferior performance, respectively, of *CBoost-Semi* against other algorithms according to t-test with 95% of significance level. The scores of wins, ties and losses indicate the number of problems in which *CBoost-Semi* was statistically superior, comparable, inferior to the other algorithms, respectively.

When the class distribution does not relate to clusters, the generalization of cluster-based methods will strongly rely on learning the labeled data. Thus, we expected *CBoost-Sup* to outperform *CBoost-Semi* in datasets with misleading cluster structure, since *CBoost-Sup* only raises this assumption for the ensemble algorithm and base learner are supervised. Results of USPS (with 10 labels) and Digit1 demonstrated this intuition. In contrast, *CBoost-Semi* was superior in g241d and USPS (with 100 labels). This fact may be due to the base classifiers of *CBoost-Semi* diminish their cluster bias with parameter λ and κ, whilst the supervised base classifiers in *CBoost-Sup* learn all the misleading signals from wrong label estimates. When there was an useful cluster structure (as in g241c and COIL), *CBoost-Semi* was superior. This result suggests the effectiveness of training ensemble and its base classifiers to jointly optimize a single cluster-based loss function.

In the inductive setting, *CBoost-Semi* was statistically superior to ClusterReg in most datasets. In Australian Credit, the ensemble might have produced large amounts of incorrect pseudo-labels. These results might denote that ensembles can deliver better generalization than single classifiers in SSC.

CBoost-Semi delivered better predictive accuracy than algorithms with all SSC assumptions for most problems. Results of RegBoost and MCSSB might denote the sensitiveness of

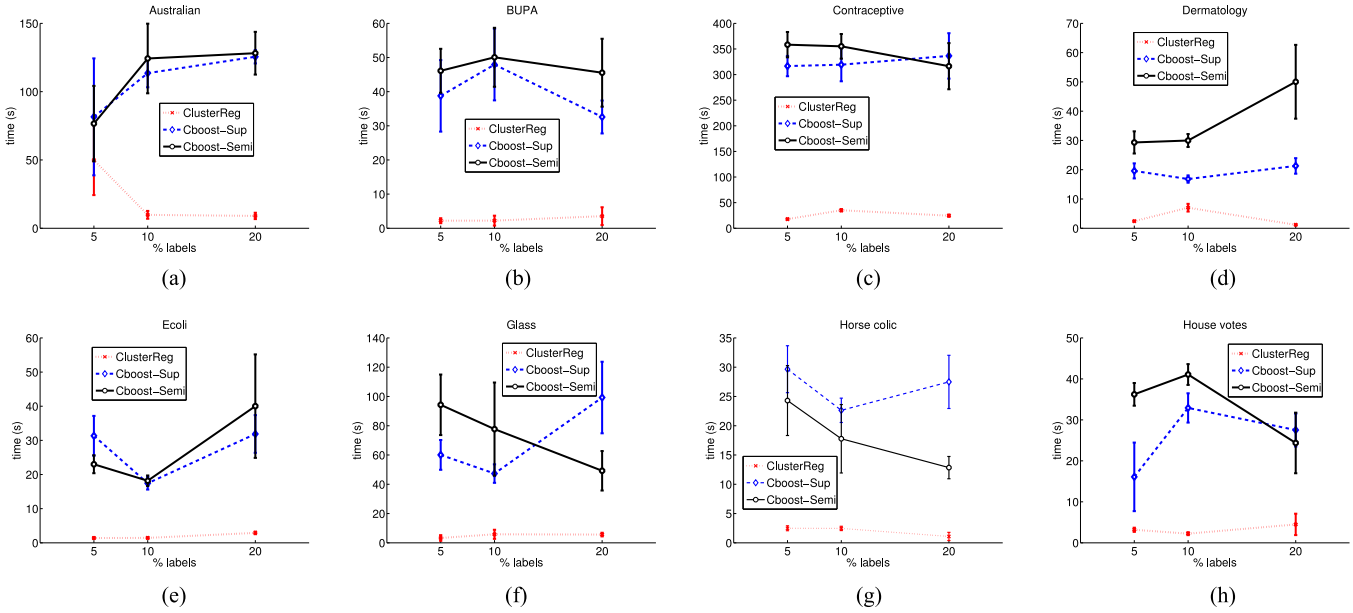

Fig. 5. Plots of average CPU time and standard deviation consumed in 10-fold cross-validation executions with proportions of 5%, 10% and 20% of labeled data. (a) Australian. (b) BUPA. (c) Contraceptive. (d) Dermatology. (e) Ecoli. (f) Glass. (g) Horse colic. (h) House votes.

algorithms based on largest-margin separator to overlapping classes and noisy labeled data in dense regions. They are more likely to propagate errors in pseudo-labels. Larger number of classes had a greater impact on RegBoost. It was significantly inferior to MCSSB for all multiclass datasets.

CBoost-Semi could tackle the scarce and noisy labeled data and overlapping classes. It was specifically designed for multiclass SSC. The training algorithm of both ensemble and base classifiers jointly minimize a semisupervised loss function that uses the cluster-based neighbourhood of a point to assess its desired label. Such characteristics and the inductive results provide evidences that our approach allows base learners to overcome possible incorrect label estimates.

RegBoost and MCSSB produced better generalization in Australian Credit with 10% and BUPA with 5% of labels, respectively. This fact indicates that such problems have meaningful manifolds that favors manifold-based methods.

Our approach had significantly better generalization than *CBoost-Sup* in most datasets. It was especially superior to *CBoost-Sup* in problems with very few labels (5% and 10% of labeled data). With 20% of labeled data, *CBoost-Semi* produced higher accuracy in four datasets and was not statistically inferior in any problem. These results confirmed that designing a base learner that explicitly helps the ensemble to minimize its loss function improves the ensemble predictive performance.

If no relevant cluster structure is present, the generalization of *CBoost-Semi* might degenerate. As depicted in Fig. 5, the trade-off for obtaining high quality predictions with *CBoost-Semi* is the increase of computational time. The use of semisupervised base learners in *CBoost-Semi* causes the computation of neighbourhoods, hence it is more time-consuming than *CBoost-Sup*. Our experiments confirmed that, similarly to other SSC methods, *CBoost-Semi* handles the lack of label information by raising assumptions on the data distribution and its relation to the class distribution.

CBoost-Semi delivered higher predictive performance in the real-world datasets with larger number of classes in all proportions of labels. *CBoost-Semi* was able to form good decision boundaries for each class with respect to the others. The ensemble learning algorithm and base learners are inherently multiclass: each class is learned dependently of the others, as described in Section III. The other methods, with the exception of MCSSB, are binary and depend on sub-optimal decomposition techniques. Such decomposition procedures are sensitive to different scales and imbalanced classes [3].

VI. Conclusion

In this work, we proposed Cluster-based Boosting, a fully semisupervised ensemble approach to multiclass SSC. When the cluster assumption holds, *CBoost* can handle noisy classes and the scarce labeled points lying in dense regions does not severely affect *CBoost*'s decision boundaries. It employs semisupervised base learners that consider the surroundings of a point when assessing its label estimates. In this sense, base classifiers can tackle potentially incorrect label estimates and produce a more robust ensemble classifier.

Our experiments validated the relevance of our cluster-based boosting approach and the use of semisupervised base classifiers. The results supported the significantly superior generalization of *CBoost* over ClusterReg, *CBoost-Sup*, and other state-of-the-art ensembles. The trade-off for the better generalization of *CBoost-Semi* when compared to *CBoost-Sup* is the additional computational time required by the calculation of neighbourhoods in its semisupervised base learner.

In the semisupervised context, the availability of unlabeled data often incurs in large datasets. As future work, we aim to investigate techniques to improve the efficiency of the proposed ensemble method. We also intend to employ other parameter selection procedures for *CBoost-Semi* in order to improve its predictive performance.

REFERENCES

[1] X. Zhu, "Semi-supervised learning literature survey," Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 1530, 2008.

[2] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. Cambridge, MA, USA: MIT Press, 2006.

[3] H. Valizadegan, R. Jin, and A. K. Jain, "Semi-supervised boosting for multi-class classification," in *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*. Berlin, Germany: Springer-Verlag, 2008, pp. 522–537.

[4] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, 2001.

[5] P. Sun and X. Yao, "Sparse approximation through boosting for learning large scale kernel machines," *IEEE Trans. Neural Netw.*, vol. 21, no. 6, pp. 883–894, Jun. 2010.

[6] M. H. Nguyen, H. A. Abbass, and R. I. Mckay, "A novel mixture of experts model based on cooperative coevolution," *Neurocomputing*, vol. 70, no. 1–3, pp. 155–163, 2006.

[7] K. Chen and S. Wang, "Semi-supervised learning via regularized boosting working on multiple semi-supervised assumptions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 129–143, Jan. 2011.

[8] R. G. F. Soares, H. Chen, and X. Yao, "Semisupervised classification with cluster regularization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 11, pp. 1779–1792, Nov. 2012.

[9] I. T. Nabney, "Efficient training of RBF networks for classification," in *Proc. 9th Int. Conf. Artif. Neural Netw.*, 1999, vol. 1, pp. 210–215.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[11] F. dAlche Buc, Y. Grandvalet, and C. Ambroise, "Semi-supervised marginboost," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 553–560.

[12] Y. Grandvalet, F. d'Alché Buc, and C. Ambroise, "Boosting mixture models for semi-supervised learning," in *Proceedings of the International Conference on Artificial Neural Networks*. New York, NY, USA: Springer-Verlag, 2001, pp. 41–48.

[13] K. P. Bennett, A. Demiriz, and R. Maclin, "Exploiting unlabeled data in ensemble methods," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2002, pp. 289–296.

[14] L. Zheng, S. Wang, Y. Liu, and C.-H. Lee, "Information theoretic regularization for semi-supervised boosting," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2009, pp. 1017–1026.

[15] E. Song, D. Huang, G. Ma, and C.-C. Hung, "Semi-supervised multi-class adaboost by exploiting unlabeled data," *Expert Syst. Appl.*, vol. 38, no. 6, pp. 6720–6726, 2011.

[16] A. Saffari, C. Leistner, and H. Bischof, "Regularized multi-class semi-supervised boosting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 967–974.

[17] A. Saffari, C. Leistner, M. Godec, and H. Bischof, "Robust multi-view boosting with priors," in *Proceedings of the 11th European Conference on Computer Vision Conference on Computer Vision: Part III (ECCV)*. Berlin, Germany: Springer-Verlag, 2010, pp. 776–789.

[18] L. Z. Manor and P. Perona, "Self-tuning spectral clustering," in *Proceedings of the Advances in Neural Information Processing Systems 2004*. Cambridge, MA, USA: MIT Press, 2004, pp. 1601–1608.

[19] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.

[20] S. R. Ji Zhu, H. Zou, and T. Hastie, "Multi-class adaboost," *Statist. Interface*, vol. 2, pp. 349–360, 2009.

[21] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[22] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.

Rodrigo G. F. Soares received the B.Sc. degree in computer engineering from the Federal University of Rio Grande do Norte, Natal, Brazil, in 2005, and the M.Phil. degree in computer science from the Federal University of Pernambuco, Recife, Brazil, in 2008. He received the Brazilian Council for Scientific and Technological Development (CNPq) scholarships in both degrees. He received the Ph.D. degree in computer science from the University of Birmingham, Birmingham, U.K., in 2014, with a scholarship from the Capes Foundation, Brazil. He is currently a Lecturer at the Federal Rural University of Pernambuco, Recife, Brazil. His research interests include data mining, semisupervised learning, neural networks, clustering, and evolutionary computation.

Huanhuan Chen (SM'16) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 2004, and the Ph.D. degree in computer science from the University of Birmingham, Birmingham, U.K., in 2008.

He is currently a Professor with UBRI, School of Computer Science and Technology, USTC. His current research interests include statistical machine learning, data mining, fault diagnosis, and evolutionary computation. He received the 2015 International Neural Network Society Young Investigator Award, IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation Award (the only winner), and the 2009 CPHC/British Computer Society Distinguished Dissertations Award (the runner up). His work "Probabilistic Classification Vector Machines" on Bayesian machine learning has been awarded the IEEE Transactions on Neural Networks Outstanding Paper Award (bestowed in 2011 and only one paper in 2009).

Xin Yao (F'03) is a Chair Professor of computer science at the Southern University of Science and Technology, Shenzhen, China, and a Professor of computer science at the University of Birmingham, Birmingham, U.K. He is a Distinguished Lecturer of the IEEE Computational Intelligence Society (CIS). His major research interests include evolutionary computation, ensemble learning, and their applications in software engineering. He has been working on multiobjective optimization since 2003, when he published a well-cited EMO03 paper on many objective optimization. He received the 2001 IEEE Donald G. Fink Prize Paper Award, 2010, 2016, and 2017 IEEE Transactions on Evolutionary Computation Outstanding Paper Awards, 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), 2011 IEEE Transactions on Neural Networks Outstanding Paper Award, and many other best paper awards. He received the prestigious Royal Society Wolfson Research Merit Award in 2012 and the IEEE CIS Evolutionary Computation Pioneer Award in 2013. He was the President (2014–2015) of IEEE CIS, and the Editor-in-Chief (2003–2008) of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION.