C^{3}E: A Framework for Combining Ensembles of Classifiers and Clusterers

A. Acharya[1], E.R. Hruschka[1,2], J. Ghosh[1], and S. Acharyya[1]

[1] University of Texas (UT) at Austin, USA
[2] University of Sao Paulo (USP) at Sao Carlos, Brazil

Abstract. The combination of multiple classifiers to generate a single classifier has been shown to be very useful in practice. Similarly, several efforts have shown that cluster ensembles can improve the quality of results as compared to a single clustering solution. These observations suggest that ensembles containing both classifiers and clusterers are potentially useful as well. Specifically, clusterers provide supplementary constraints that can improve the generalization capability of the resulting classifier. This paper introduces a new algorithm named **C^{3}E** that combines ensembles of classifiers and clusterers. Our experimental evaluation of **C^{3}E** shows that it provides good classification accuracies in eleven tasks derived from three real-world applications. In addition, **C^{3}E** produces better results than the recently introduced Bipartite Graph-based Consensus Maximization (**BGCM**) Algorithm, which combines multiple supervised and unsupervised models and is the algorithm most closely related to **C^{3}E**.

Keywords: Ensembles, Classification, Clustering.

1 Introduction

The combination of multiple classifiers to generate a single classifier has been an active area of research for the last two decades [8,7]. For instance, an analytical framework to quantify the improvements in classification results due to combining multiple models has been addressed in [13]. More recently, a survey of traditional ensemble techniques — including applications of them to many difficult real-world problems such as remote sensing, person recognition, one vs. all recognition, and medicine — has been presented in [9]. In brief, the extensive literature on the subject has shown that from independent, diversified classifiers, the ensemble created is usually more accurate than its individual components. Analogously, several research efforts have shown that cluster ensembles can improve the quality of results as compared to a single clustering solution — *e.g.*, see [6]. Actually, the potential motivations and benefits for using cluster ensembles are much broader than those for using classifier ensembles, for which improving the predictive accuracy is usually the primary goal. More specifically, cluster ensembles can be used to generate more robust and stable clustering results (compared to a single clustering approach), perform distributed computing under privacy or sharing constraints, or reuse existing knowledge [12].

C. Sansone, J. Kittler, and F. Roli (Eds.): MCS 2011, LNCS 6713, pp. 269–278, 2011.

In this paper, an algorithm that combines ensembles of classifiers and clusterers is introduced. As far as we know, this topic has not been addressed in the literature. Most of the motivations for combining ensembles of classifiers and clusterers are similar to those that hold for the standalone use of either classifier ensembles or cluster ensembles. However, some additional nice properties can emerge from such a combination — *e.g.*, unsupervised models can provide a variety of supplementary constraints for classifying new data [11]. From this viewpoint, the underlying assumption is that similar new objects in the target set are more likely to share the same class label. Thus, the supplementary constraints provided by the cluster ensemble can be useful for improving the generalization capability of the resulting classifier, specially when labelled data is scarce. Also, they can be useful for designing learning methods that are aware of the possible differences between training and target distributions, thus being particularly interesting for applications in which concept drift may take place.

The remainder of this paper is organized as follows. The proposed algorithm — named **C^{3}E**, from Consensus between Classification and Clustering Ensembles — is described in the next section. Related work is addressed in Section 3. An experimental study is reported in Section 4. Finally, Section 5 concludes the paper and describes ongoing work.

Notation. Vectors and matrices are denoted by bold faced lowercase and capital letters, respectively. Scalar variables are written in italic font. A set is denoted by a calligraphic uppercase letter. The effective domain of a function $f(y)$, i.e., the set of all y such that $f(y) < +\infty$ is denoted by $dom(f)$, while the interior and the relative interior of a set $\mathcal{Y}$ are denoted by $int(\mathcal{Y})$ and $ri(\mathcal{Y})$, respectively. Also, for $\mathbf{y}_i, \mathbf{y}_j \in \mathbb{R}^k$, $\langle \mathbf{y}_i, \mathbf{y}_j \rangle$ denotes their inner product.

2 Description of C^{3}E

The proposed framework that combines classifier and cluster ensembles to generate a more consolidated classification is depicted in Fig. 1. It is assumed that an ensemble of classifiers has been previously induced from a training set. Such an ensemble is part of the framework that will be used for classifying new data — *i.e.*, objects from the target set[1] $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$. The ensemble of classifiers is employed to estimate initial class probabilities for every object $\mathbf{x}_i \in \mathcal{X}$. These probability distributions are stored as a set of vectors $\{\boldsymbol{\pi}_i\}_{i=1}^n$ and will be refined with the help of a cluster ensemble. From this point of view, the cluster ensemble provides supplementary constraints for classifying the objects of $\mathcal{X}$, with the rationale that similar objects are more likely to share the same class label. Each of $\boldsymbol{\pi}_i$'s is of dimension k so that, in total, there are k classes denoted by $C = \{C_\ell\}_{\ell=1}^k$. In order to capture the similarities between the objects of $\mathcal{X}$, **C^{3}E** also takes as input a similarity (co-association) matrix $\mathbf{S}$, where each entry corresponds to the relative co-occurrence of two objects in the same cluster [6,12] — considering all the data partitions that form the cluster ensemble induced

[1] The target set is a test set that has not been used to build the ensemble of classifiers.

from $\mathcal{X}$. To summarize, **C^{3}E** receives as inputs a set of vectors $\{\boldsymbol{\pi}_i\}_{i=1}^n$ and the similarity matrix **S**. After processing these inputs, **C^{3}E** outputs a consolidated classification — represented by a set of vectors $\{\mathbf{y}_i\}_{i=1}^n$, where $\mathbf{y}_i = \hat{P}(C \mid \mathbf{x}_i)$ — for every object in $\mathcal{X}$. This procedure is described in more detail below, where r_1 classifiers, indexed by q_1, and r_2 clusterers, indexed by q_2, are employed to obtain a consolidated classification.

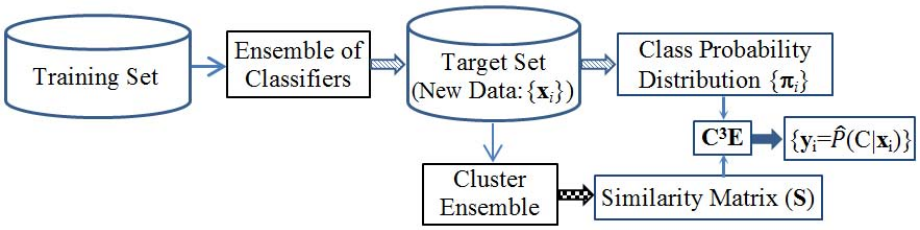

Fig. 1. Combining Ensembles of Classifiers and Clusterers

Step A: Obtain Input From Classifier Ensemble. The output of classifier q_1 for object $\mathbf{x}_i$ is a k-dimensional class probability vector $\boldsymbol{\pi}_i^{(q_1)}$. From the set of such vectors $\{\boldsymbol{\pi}_i^{(q_1)}\}_{q_1=1}^{r_1}$, an average vector can be computed for $\mathbf{x}_i$ as:

$$\boldsymbol{\pi}_i = \frac{1}{r_1} \sum_{q_1=1}^{r_1} \boldsymbol{\pi}_i^{(q_1)}. \tag{1}$$

Step B: Obtain Input From Cluster Ensemble. After applying r_2 clustering algorithms (clusterers) to $\mathcal{X}$, a similarity (co-association) matrix **S** is computed. Assuming each clustering is a hard data partition, the similarity between two objects is simply the fraction of the r_2 clustering solutions in which those two objects lie in the same cluster[2]. Note that such similarity matrices are byproducts of several cluster ensemble solutions, e.g., CSPA algorithm in [12].

Step C: Obtain Consolidated Results from C^{3}E. Having defined the inputs for **C^{3}E**, namely the set $\{\boldsymbol{\pi}_i\}_{i=1}^n$ and the similarity matrix, **S**, the problem of combining the ensembles of classifiers and clusterers can be posed as an optimization problem whose objective is to minimize J in (2) w.r.t. the set of probability vectors $\{\mathbf{y}_i\}_{i=1}^n$, where $\mathbf{y}_i = \hat{P}(C \mid \mathbf{x}_i)$, *i.e.*, $\mathbf{y}_i$ is the new and hopefully improved estimate of the aposteriori class probability distribution for a given object in $\mathcal{X}$.

$$J = \sum_{i \in \mathcal{X}} \mathcal{L}(\boldsymbol{\pi}_i, \mathbf{y}_i) + \alpha \sum_{(i,j) \in \mathcal{X}} s_{ij} \mathcal{L}(\mathbf{y}_i, \mathbf{y}_j) \tag{2}$$

[2] A similarity matrix can also be defined for soft clusterings — *e.g.*, see [10].

The quantity $\mathcal{L}(\cdot,\cdot)$ denotes a loss function. Informally, the first term in Eq. (2) captures dissimilarities between the class probabilities provided by the ensemble of classifiers and the output vectors $\{\mathbf{y}_i\}_{i=1}^n$. The second term encodes the cumulative weighted dissimilarity between all possible pairs $(\mathbf{y}_i, \mathbf{y}_j)$. The weights to these pairs are assigned in proportion to the similarity values $s_{ij} \in [0,1]$ of matrix $\mathbf{S}$. The coefficient $\alpha \in \mathbb{R}_+$ controls the relative importance of classifier and cluster ensembles. Therefore, minimizing the objective function over $\{\mathbf{y}_i\}_{i=1}^n$ involves combining the evidence provided by the ensembles in order to build a more consolidated classification.

The approach taken in this paper is quite general in that any Bregman divergence (defined in the Appendix) can be used as the loss function $\mathcal{L}(\cdot,\cdot)$ in Eq. (2). Bregman divergences include a large number of useful loss functions such as the well-known squared loss, KL-divergence, logistic loss, Mahalanobis distance, and I-divergence. A specific Bregman Divergence (*e.g.* KL-divergence) can be identified by a corresponding convex function ϕ (*e.g.* negative entropy for KL-divergence), and hence be written as $d_\phi(\mathbf{y}_i, \mathbf{y}_j)$. Using this notation, the optimization problem can be rewritten as:

$$\min_{\{\mathbf{y}_i\}_{i=1}^n} J = \min_{\{\mathbf{y}_i\}_{i=1}^n} \left[\sum_{i \in \mathcal{X}} d_\phi(\boldsymbol{\pi}_i, \mathbf{y}_i) + \alpha \sum_{(i,j) \in \mathcal{X}} s_{ij} d_\phi(\mathbf{y}_i, \mathbf{y}_j) \right]. \tag{3}$$

All Bregman divergences have the remarkable property that the single best (in terms of minimizing the net loss) representative of a set of vectors, is simply the expectation of this set (!) provided the divergence is computed with this representative as the second argument of $d_\phi(\cdot,\cdot)$ — see Th. 1 in the Appendix for a more formal statement of this result. Unfortunately this simple form of the optimal solution is not valid if the variable to be optimized occurs as the first argument. In that case, however, one can work in the (Legendre) dual space, where the optimal solution has a simple form (see [1] for details). Re-examining Eq. (3), we notice that the $\mathbf{y}_i$'s to be minimized over occur both as first and second arguments of a Bregman divergence. Hence optimization over $\{\mathbf{y}_i\}_{i=1}^n$ is not available in closed form. We circumvent this problem by creating two copies for each $\mathbf{y}_i$ — the left copy, $\mathbf{y}_i^{(l)}$, and the right copy, $\mathbf{y}_i^{(r)}$. The left(right) copies are used whenever the variables are encountered in the first(second) argument of the Bregman divergences. The right and left copies are updated iteratively, and an additional constraint is used to ensure that the two copies of a variable remain close during the updates. First, keeping $\{\mathbf{y}_i^{(l)}\}_{i=1}^n$ and $\{\mathbf{y}_i^{(r)}\}_{i=1}^n \setminus \{\mathbf{y}_j^{(r)}\}$ fixed, the part of the objective function that only depends on $\mathbf{y}_j^{(r)}$ can be written as:

$$J_{[\mathbf{y}_j^{(r)}]} = d_\phi(\boldsymbol{\pi}_j^{(r)}, \mathbf{y}_j^{(r)}) + \alpha \sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}). \tag{4}$$

Note that the optimization of $J_{[\mathbf{y}_j^{(r)}]}$ in (4) w.r.t. $\mathbf{y}_j^{(r)}$ is constrained by the fact that the left and right copies of $\mathbf{y}_j$ should be equal. Therefore, a soft constraint is added in (4), and the optimization problem now becomes:

$$\min_{\mathbf{y}_j^{(r)}} \left[d_\phi(\boldsymbol{\pi}_j^{(r)}, \mathbf{y}_j^{(r)}) + \alpha \sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}) + \lambda_j^{(r)} d_\phi(\mathbf{y}_j^{(l)}, \mathbf{y}_j^{(r)}) \right], \quad (5)$$

where $\lambda_j^{(r)}$ is the corresponding Lagrange multiplier. It can be shown (see the Appendix) that there is a unique minimizer for the optimization problem in (5):

$$\mathbf{y}_j^{(r)^*} = \left[\left[\boldsymbol{\pi}_j^{(r)} + \gamma_j^{(r)} \sum_{i^{(l)} \in \mathcal{X}} \delta_{i^{(l)} j^{(r)}} \mathbf{y}_i^{(l)} + \lambda_j^{(r)} \mathbf{y}_j^{(l)} \right] / \left[1 + \gamma_j^{(r)} + \lambda_j^{(r)} \right] \right], \quad (6)$$

where $\gamma_j^{(r)} = \alpha \sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}}$ and $\delta_{i^{(l)} j^{(r)}} = s_{i^{(l)} j^{(r)}} / \left[\sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} \right]$. The same optimization in (5) is repeated over all the $\mathbf{y}_j^{(r)}$'s. After the right copies are updated, the objective function is (sequentially) optimized w.r.t. all the $\mathbf{y}_i^{(l)}$'s. Like in the first step, $\{\mathbf{y}_j^{(l)}\}_{j=1}^n \setminus \{\mathbf{y}_i^{(l)}\}$ and $\{\mathbf{y}_j^{(r)}\}_{j=1}^n$ are kept fixed, and the equality of the left and right copies of $\mathbf{y}_i$ is added as a soft constraint, so that the optimization w.r.t. $\mathbf{y}_i^{(l)}$ can be rewritten as:

$$\min_{\mathbf{y}_i^{(l)}} \left[\alpha \sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}) + \lambda_i^{(l)} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)}) \right], \quad (7)$$

where $\lambda_i^{(l)}$ is the corresponding Lagrange multiplier. As mentioned earlier, one needs to work in the dual space now, using the convex function ψ (Legendre dual of ϕ) which is defined as:

$$\psi(\mathbf{y}_i) = \langle \mathbf{y}_i, \nabla\phi^{-1}(\mathbf{y}_i) \rangle - \phi(\nabla\phi^{-1}(\mathbf{y}_i)). \quad (8)$$

One can show that $\forall \mathbf{y}_i, \mathbf{y}_j \in \mathit{int}(\mathrm{dom}(\phi))$, $d_\phi(\mathbf{y}_i, \mathbf{y}_j) = d_\psi(\nabla\phi(\mathbf{y}_j), \nabla\phi(\mathbf{y}_i))$ (see [1] for more details). Thus, the optimization problem in (7) can be rewritten in terms of the Bregman divergence associated with ψ as follows:

$$\min_{\nabla\phi(\mathbf{y}_i^{(l)})} \left[\alpha \sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\psi(\nabla\phi(\mathbf{y}_j^{(r)}), \nabla\phi(\mathbf{y}_i^{(l)})) + \lambda_i^{(l)} d_\psi(\nabla\phi(\mathbf{y}_i^{(r)}), \nabla\phi(\mathbf{y}_i^{(l)})) \right]. \quad (9)$$

The unique minimizer of the problem in (9) can be computed using Corollary 1 (see the Appendix). $\nabla\phi$ is monotonic and invertible for ϕ being strictly convex and hence the inverse of the unique minimizer for problem (9) is unique and equals to the unique minimizer for problem (7). Therefore, the unique minimizer of problem (7) w.r.t. $\mathbf{y}_i^{(l)}$ is given by:

$$\mathbf{y}_i^{(l)^*} = \nabla\phi^{-1} \left[\left[\gamma_i^{(l)} \sum_{j^{(r)} \in \mathcal{X}} \delta_{i^{(l)} j^{(r)}} \nabla\phi(\mathbf{y}_j^{(r)}) + \lambda_i^{(l)} \nabla\phi(\mathbf{y}_i^{(r)}) \right] / \left[\gamma_i^{(l)} + \lambda_i^{(l)} \right] \right], \quad (10)$$

where $\gamma_i^{(l)} = \alpha \sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}}$ and $\delta_{i^{(l)} j^{(r)}} = s_{i^{(l)} j^{(r)}} / \left[\sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} \right]$.

For the experiments reported in this paper, the generalized I-divergence, defined as:

$$d_\phi(\mathbf{y}_i, \mathbf{y}_j) = \sum_{\ell=1}^{k} y_{i\ell} \log(\frac{y_{i\ell}}{y_{j\ell}}) - \sum_{\ell=1}^{k} (y_{i\ell} - y_{j\ell}), \forall \mathbf{y}_i, \mathbf{y}_j \in \mathbb{R}^k, \tag{11}$$

has been used. The underlying convex function is given by $\phi(\mathbf{y}_i) = \sum_{\ell=1}^{k} y_{i\ell} \log(y_{i\ell})$ so that $\nabla\phi(\mathbf{y}_i) = (1 + \log(y_{i\ell}))_{\ell=1}^{k}$. Thus, Eq. (10) can be rewritten as:

$$\mathbf{y}_i^{(l)*,I} = \exp\left(\left[\gamma_i^{(l)} \sum_{j^{(r)} \in \mathcal{X}} \delta_{i^{(l)} j^{(r)}} \nabla\phi(\mathbf{y}_j^{(r)}) + \lambda_i^{(l)} \nabla\phi(\mathbf{y}_i^{(r)})\right] / \left[\gamma_i^{(l)} + \lambda_i^{(l)}\right]\right) - 1. \tag{12}$$

Optimization over the left and right arguments of all the data points constitutes one pass (iteration) of the algorithm. These two steps are repeated till convergence. Since, at each step, the algorithm minimizes the objective in (3) and the minimizer is unique due to the strict convexity of ϕ, the algorithm is guaranteed to converge. On convergence, all $\mathbf{y}_i$'s are normalized to unit L_1 norm, to yield the individual class probability distributions for every object $\mathbf{x}_i \in \mathcal{X}$. The main steps of $\mathbf{C^3E}$ are summarized in Algorithm 1.

Algorithm 1 - $\mathbf{C^3E}$

Input: $\{\boldsymbol{\pi}_i\}, \mathbf{S}$
Output: $\{\boldsymbol{y}_i\}$
Step 0: Initialize $\{\boldsymbol{y}_i^{(r)}\}, \{\boldsymbol{y}_i^{(l)}\}$ so that $\boldsymbol{y}_{i\ell}^{(r)} = \boldsymbol{y}_{i\ell}^{(l)} = \frac{1}{k}$ $\forall i \in \{1, 2, \cdots, n\}$, $\forall \ell \in \{1, 2, \cdots, k\}$
Loop until convergence
  Step 1:
    Update $\boldsymbol{y}_j^{(r)}$ using equation (6) $\forall j \in \{1, 2, \cdots, n\}$
  Step 2:
    Update $\boldsymbol{y}_i^{(l)}$ using equation (10) $\forall i \in \{1, 2, \cdots, n\}$
End Loop
Step 4: Compute $\boldsymbol{y}_i = 0.5[\boldsymbol{y}_i^{(l)} + \boldsymbol{y}_i^{(r)}]$ $\forall i \in \{1, 2, \cdots, n\}$
Step 5: Normalize $\boldsymbol{y}_i$ $\forall i \in \{1, 2, \cdots, n\}$

3 Related Work

Of late there has been substantial interest in exploiting both labeled and unlabeled data for a variety of learning scenarios, including works on semi-supervised learning and transductive learning [11,2,14]. In almost all cases, these approaches use a *single* (clustering, classification or regression) model, which is then tempered by additional data or other constraints. A notable exception is a recent

work by Gao et al. [5], in which the outputs of *multiple* supervised and unsupervised models are combined. Here, it is assumed that each model partitions the target dataset $\mathcal{X}$ into groups, so that the objects in the same group share either the same predicted class label or the same cluster label. The data, models and outputs are summarized by a bipartite graph. In this graph, on one side the nodes denote the groups output by the models, whereas on the other side the nodes denote objects. A group node and an object node are connected if the object is assigned to the group — no matter if it comes from a supervised or unsupervised model. From the resulting graph, the goal is to predict the class labels so that they agree with the supervised models and also satisfy the constraints enforced by the clustering models, as much as possible. In other words, the authors in [5] aim at consolidating a classification solution by maximizing the consensus among both supervised predictions and unsupervised constraints, casting it as an optimization problem on a bipartite graph. The objective function is designed to maximize such a consensus by promoting smoothness of label assignment over the graph and consistency with the initial labeling. To solve the optimization problem, they introduce the Bipartite Graph-based Consensus Maximization (**BGCM**) Algorithm.

The **C^{3}E** algorithm can also be viewed as a semi-supervised ensemble working at the output level. Unlike **BGCM**, however, it does not receive as input several supervised and unsupervised models. Instead, **C^{3}E** ultimately processes only two fused models, namely: (i) an ensemble of classifiers that delivers a class probability vector for every object in $\mathcal{X}$; and (ii) an ensemble of clusterers that provides a similarity matrix, where each entry corresponds to the relative co-ocurrence of two objects in the same cluster of $\mathcal{X}$ (considering all the available data partitions). Contrary to **BGCM**, which is based on hard classification inputs from supervised models, **C^{3}E** can deal with class probability distributions obtained by the ensemble of classifiers, and caters to both hard and soft clusterings. Moreover, **C^{3}E** avoids solving a difficult correspondence problem — *i.e.*, aligning cluster labels to class labels — implicitly tackled by **BGCM**.

4 Experimental Evaluation

The **C^{3}E** algorithm has been evaluated on the same classification datasets employed by Gao *et al.* [5] to assess their **BGCM** algorithm[3]. Following Gao *et al.* [5], eleven classification tasks from three real-world applications (20 Newsgroups, Cora, and DBLP) have been used. In each task, there is a target set on which the class labels should be predicted. In [5], two supervised models ($\mathbf{M}_1$ and $\mathbf{M}_2$) and two unsupervised models ($\mathbf{M}_3$ and $\mathbf{M}_4$) were used to obtain (on the target sets) class and cluster labels, respectively. These same labels have been used as inputs to **C^{3}E**. In doing so, comparisons between **C^{3}E** and **BGCM** are performed using exactly the same base models, which were trained in the same datasets. In other words, both **C^{3}E** and **BGCM** receive the same inputs

[3] Datasets available at http://ews.uiuc.edu/ jinggao3/nips09bgcm.htm.

w.r.t. the components of the ensembles, from which consolidated classification solutions for the target sets are generated.

For the sake of compactness, the description of the datasets and learning models used in [5] are not reproduced here, and the interested reader is referred to that paper for further details. However, the results achieved by Gao *et al.* [5] from their four base models ($\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, and $\mathbf{M}_4$), from **BGCM** [5], and from two well-known cluster ensemble approaches — **MCLA** [12] and **HBGF** [4] — are reproduced here for comparison purposes. Being cluster ensemble approaches, **MCLA** [12] and **HBGF** [4] ignore the class labels, considering that the four base models provide just cluster labels. Therefore, to evaluate classification accuracy obtained by these ensembles, the cluster labels are matched to the classes through an Hungarian method that favors the best possible class predictions. In order to run $\mathbf{C^3E}$, the supervised models ($\mathbf{M}_1$ and $\mathbf{M}_2$) have been fused to obtain class probability estimates for every object in the target set. Also, the co-association matrix used by $\mathbf{C^3E}$ was achieved by fusing the unsupervised models ($\mathbf{M}_3$ and $\mathbf{M}_4$). The parameters of $\mathbf{C^3E}$ have been manually optimized for better performance in each dataset. In particular the following pairs of (α, λ)[4] have been respectively used for the datasets News, Cora, and DBLP: (4×10^{-2},10^{-2}); (10^{-4},10^{-2}); (10^{-7}, 10^{-3}).

Table 1. Classification Accuracies — Best Results in Boldface

Method	News1	News2	News3	News4	News5	News6	Cora1	Cora2	Cora3	Cora4	DBLP
M_1	0.7967	0.8855	0.8557	0.8826	0.8765	0.8880	0.7745	0.8858	0.8671	0.8841	0.9337
M_2	0.7721	0.8611	0.8134	0.8676	0.8358	0.8563	0.7797	0.8594	0.8508	0.8879	0.8766
M_3	0.8056	0.8796	0.8658	0.8983	0.8716	0.9020	0.7779	0.8833	0.8646	0.8813	0.9382
M_4	0.7770	0.8571	0.8149	0.8467	0.8543	0.8578	0.7476	0.8594	0.7810	0.9016	0.7949
MCLA	0.7592	0.8173	0.8253	0.8686	0.8295	0.8546	0.8703	0.8388	0.8892	0.8716	0.8953
HBGF	0.8199	0.9244	0.8811	0.9152	0.8991	0.9125	0.7834	0.9111	0.8481	0.8943	0.9357
BGCM	0.8128	0.9101	0.8608	0.9125	0.8864	0.9088	0.8687	0.9155	0.8965	0.9090	0.9417
C^3E	**0.8501**	**0.9364**	**0.8964**	**0.9380**	**0.9122**	**0.9180**	**0.8854**	**0.9171**	**0.9060**	**0.9149**	**0.9438**

The classification accuracies achieved by the studied methods are summarized in Table 1, where one can see that the proposed $\mathbf{C^3E}$ has shown the best accuracies for all datasets. In order to provide some reassurance about the validity and non-randomness of the obtained results, the outcomes of statistical tests, following the study of Demsar [3], are also reported. In brief, multiple algorithms have been compared on multiple datasets by using the Friedman test, with a corresponding post-hoc test. The adopted statistical procedure indicates that the null hypothesis of equal accuracies — considering the results obtained by the ensembles — can be rejected at $\alpha = 0.05$. In pairwise comparisons, significant statistical differences have only been observed between $\mathbf{C^3E}$ and the other ensembles, *i.e.*, there is no evidence that the accuracies of **MCLA**, **HBGF**, and **BGCM** are statistically different from one to another.

[4] $\lambda_i^{(r)} = \lambda_i^{(l)} = \lambda$ has been set for all i.

5 Conclusions

The **C^{3}E** algorithm, which combines ensembles of classifiers and clusterers, was introduced. **C^{3}E** has shown better accuracies than the recently proposed **BGCM** Algorithm [5], which combines the outputs of multiple supervised and unsupervised models and is the most closely related algorithm to **C^{3}E**. The asymptotic time complexity of **C^{3}E** is quadratic with the number of objects in the target set and linear with the number of ensemble components, whereas **BGCM** has cubic time complexity with respect to these input sizes.

There are several aspects that can be investigated in future work. For example, the impact of the number of classifiers and clusterers in **C^{3}E** deserves further investigations. Also, the relative relevance of each component of the ensemble can be straightforwardly incorporated into **C^{3}E**. Finally, a more comprehensive experimental evaluation, specially considering comparisons with other semi-supervised algorithms, is in order.

Acknowledgments

This work has been supported by NSF Grants (IIS-0713142 and IIS-1016614) and by the Brazilian Research Agencies FAPESP and CNPq.

References

1. Banerjee, A., Merugu, S., Dhillon, I., Ghosh, J.: Clustering with Bregman divergences. In: JMLR (2005)
2. Schlkopf, B., Zien, A., Chapelle, O.: Semi-Supervised Learning. MIT Press. Cambridge (2006)
3. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7, 1–30 (2006)
4. Fern, X.Z., Brodley, C.E.: Solving cluster ensemble problems by bipartite graph partitioning. In: Proc. of the ICML, pp. 36–43 (2004)
5. Gao, J., Liang, F., Fan, W., Sun, Y., Han, J.: Graph-based consensus maximization among multiple supervised and unsupervised models. In: Proc. of NIPS, pp. 1–9 (2009)
6. Ghosh, J., Acharya, A.: Cluster ensembles. WIREs Data Mining and Knowledge Discovery 1, 1–12 (to appear 2011)
7. Kittler, J., Roli, F. (eds.): IPSN 2003. LNCS, vol. 2634. Springer, Heidelberg (2003)
8. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley, Chichester (2004)
9. Oza, N., Tumer, K.: Classifier ensembles: Select real-world applications. Information Fusion 9(1), 4–20 (2008)
10. Punera, K., Ghosh, J.: Consensus based ensembles of soft clusterings. Applied Artificial Intelligence 22, 109–117 (2008)
11. Davidson, I., Basu, S., Wagstaff, K.L. (eds.): Clustering with Balancing Constraints. CRC Press, Boca Raton (2008)
12. Strehl, A., Ghosh, J.: Cluster ensembles – a knowledge reuse framework for combining multiple partitions. In: JMLR, vol. 617, pp. 583–617 (2002)

13. Tumer, K., Ghosh, J.: Analysis of decision boundaries in linearly combined neural classifiers. Pattern Recognition 29, 341–348 (1996)
14. Goldberg, A., Zhu, X.: Introduction to Semi-Supervised Learning. Morgan and Claypool Publishers, San Rafael (2009)

Appendix

Definition 1. *Let $\phi : \mathcal{S} \to \mathbb{R}$ be a strictly convex, differentiable function defined on a convex set $\mathcal{S} = dom(\phi) \subseteq \mathbb{R}^k$. Then the* ***Bregman divergence*** *$d_\phi : \mathcal{S} \times ri(\mathcal{S}) \to [0, \infty)$ between $\mathbf{y}_i \in \mathcal{S}$ and $\mathbf{y}_j \in ri(\mathcal{S})$ is defined as: $d_\phi(\mathbf{y}_i, \mathbf{y}_j) = \phi(\mathbf{y}_i) - \phi(\mathbf{y}_j) - \langle \mathbf{y}_i - \mathbf{y}_j, \nabla\phi(\mathbf{y}_j) \rangle$.*

From this Definition, it follows that $d_\phi(\mathbf{y}_i, \mathbf{y}_j) \geq 0 \; \forall \mathbf{y}_i \in \mathcal{S}, \mathbf{y}_j \in ri(\mathcal{S})$ and equality holds *iff* $\mathbf{y}_i = \mathbf{y}_j$. Then, it can be shown that there is a unique minimizer for the optimization problem in (5) by considering the following theorem:

Theorem 1 (from [1]). *Let Y be a random variable that takes values in $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1}^n \subset \mathcal{S} \subseteq \mathbb{R}^k$ following a probability measure v such that $\mathbb{E}_v[Y] \in ri(\mathcal{S})$. Given a Bregman divergence $d_\phi : \mathcal{S} \times ri(\mathcal{S}) \to [0, \infty)$, the optimization problem $\min_{\mathbf{s} \in ri(\mathcal{S})} \mathbb{E}_v[d_\phi(Y, \mathbf{s})]$ has a unique minimizer given by $\mathbf{s}^* = \boldsymbol{\mu} = \mathbb{E}_v[Y]$.*

Corollary 1. *Let $\{Y_i\}_{i=1}^n$ be a set of random variables, each of which takes values in $\mathcal{Y}_i = \{\mathbf{y}_{ij}\}_{j=1}^{n_i} \subset \mathcal{S} \subseteq \mathbb{R}^d$ following a probability measure v_i such that $\mathbb{E}_{v_i}[Y_i] \in ri(\mathcal{S})$. Consider a Bregman divergence d_ϕ and an objective function of the form $J_\phi(\mathbf{s}) = \sum_{i=1}^m \alpha_i \mathbb{E}_{v_i}[d_\phi(Y_i, \mathbf{s})]$ with $\alpha_i \in \mathbb{R}_+ \; \forall i$. This objective function has a unique minimizer given by $\mathbf{s}^* = \boldsymbol{\mu} = \left[\sum_{i=1}^m \alpha_i \mathbb{E}_{v_i}[Y_i]\right] / \left[\sum_{i=1}^m \alpha_i\right]$.*

Proof. The proof for this corollary is pretty straightforward and similar to that of Theorem 1 as given in [1] but omitted here for space constraints. ■