An Optimization Framework for Combining Ensembles of Classifiers and Clusterers with Applications to Non-transductive Semi-Supervised Learning and Transfer Learning

Ayan Acharya, University of Texas at Austin, USA
Eduardo R. Hruschka, University of Texas at Austin, USA; University of Sao Paulo at Sao Carlos, Brazil
Joydeep Ghosh, University of Texas at Austin, USA
Sreangsu Acharyya, University of Texas at Austin, USA

Unsupervised models can provide supplementary soft constraints to help classify new, "target" data since similar instances in the target set are more likely to share the same class label. Such models can also help detect possible differences between training and target distributions, which is useful in applications where concept drift may take place, as in transfer learning settings. This paper describes a general optimization framework that takes as input class membership estimates from existing classifiers learnt on previously encountered "source" (or training) data, as well as a similarity matrix from a cluster ensemble operating solely on the target (or test) data to be classified, and yields a consensus labeling of the target data. More precisely, the application settings considered are non-transductive semi-supervised and transfer learning scenarios where the training data is used only to build an ensemble of classifiers and is subsequently discarded before classifying the target data. The framework admits a wide range of loss functions and classification/clustering methods. It exploits properties of Bregman divergences in conjunction with Legendre duality to yield a principled and scalable approach. A variety of experiments show that the proposed framework can yield results substantially superior to those provided by naïvely applying classifiers learnt on the original task to the target data. In addition, we show that the proposed approach, even not being conceptually transductive, can provide better results compared to some popular transductive learning techniques.

Categories and Subject Descriptors: I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*; I.5.3 [**Pattern Recognition**]: Clustering—*Algorithms*; I.5.4 [**Pattern Recognition**]: Applications—*Computer vision*; *Text processing*

General Terms: Algorithms, Design, Performance, Theory

Additional Key Words and Phrases: Classification; Clustering; Ensembles; Transductive Learning; Semisupervised Learning; Transfer Learning

ACM Reference Format:
Acharya, A., Hruschka, E.R., Ghosh, J., Acharyya, S. 2012. ACM Trans. Knowl. Discov. Data. 2, 4, Article 01 (April 2012), 37 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

In several data mining applications, ranging from identifying distinct control regimes in complex plants to characterizing different types of stocks in terms of price and vol-

This work has been supported by NSF Grants (IIS-0713142 and IIS-1016614) and by the Brazilian Research Agencies FAPESP and CNPq. Author's addresses: A. Acharya, Department of Electrical and Computer Engineering, University of Texas at Austin;

ume movements, one builds an initial classification model that needs to be applied to unlabeled data acquired subsequently. Since the statistics of the underlying phenomena being modeled often changes with time, these classifiers may also need to be occasionally rebuilt if performance degrades beyond an acceptable level. In such situations, it is desirable that the classifier functions well with as little labeling of new data as possible, since labeling can be expensive in terms of time and money, and it is a potentially error-prone process. Moreover, the classifier should be able to adapt to changing statistics to some extent, given the afore-mentioned constraints.

This paper addresses the problem of combining multiple classifiers and clusterers in a fairly general setting, that includes the scenario sketched above. An ensemble of classifiers is first learnt on an initial labeled training dataset which can conveniently be denoted by "source" dataset. At this point, the training data can be discarded. Subsequently, when new, unlabeled target data is encountered, a cluster ensemble is applied to it to yield a similarity matrix. In addition, the previously learnt classifier(s) can be used to obtain an estimate of the class probability distributions for this data. The heart of our technique is an optimization framework that combines both sources of information to yield a consensus labeling of the target data. General properties of a large class of loss functions described by Bregman divergences are exploited in this framework in conjunction with Legendre duality and a notion of variable splitting that is also used in alternating direction method of multipliers [Boyd et al. 2011]) to yield a principled and scalable solution.

Note that the setting described above is different from transductive learning setups where both labeled and unlabeled data are available at the same time for model building [Silver and Bennett 2008], as well as online methods where decisions are made on one new example at a time, and after each such decision, the true label of the example is obtained and used to update the model parameters [Blum 1998]. Additional differences from existing approaches are described in the section on related works. For the moment we note that the underlying assumption is that similar new instances 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 system, specially when labeled data for training the base classifiers is scarce. Also, these supplementary constraints provided by unsupervised models can be useful for designing learning methods that help determine differences between training and target distributions, making the overall system more robust against concept drift. To highlight these additional capabilities that are useful for transfer learning, we provide a separate set of empirical studies where the target data is related to but significantly different from the initial training data.

The remainder of this paper is organized as follows. After addressing related work in Section 2, the proposed optimization framework and its associated algorithm — named $\mathbf{OAC^3}$, from **O**ptimization **A**lgorithm for **C**ombining **C**lassifiers and **C**lusterers — are described in Section 3. This particular algorithm has been briefly introduced in [Acharya et al. 2011; Acharya et al. 2012]. A convergence analysis of $\mathbf{OAC^3}$ is reported in Section 4, while Section 5 analyses its convergence rate. An experimental study illustrating the potential of the proposed framework for a variety of applications is reported in Section 6. Finally, Section 7 concludes the paper.

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. For $\mathbf{y}_i, \mathbf{y}_j \in \mathbb{R}^k$, $\langle \mathbf{y}_i, \mathbf{y}_j \rangle$

denotes their inner product. A function $f \in C^{k'}$ if all of its first k' derivatives exist and are continuous.

2. RELATED WORK

This contribution leverages the theory of classifier and cluster ensemble to solve transfer and semi-supervised learning problems in a non-transductive setting. Also, the underlying optimization framework inherits properties from alternating optimization type of algorithms. In this section, we briefly discuss the most related works in each of these different research areas. Table I shows existing machine learning algorithms for different application settings, including those where the proposed **OAC^3** can be used as well.

The combination of multiple single or base classifiers to generate a more capable ensemble classifier has been an active area of research for the past two decades [Kuncheva 2004; Oza and Tumer 2008]. Several papers provide both theoretical results [Tumer and Ghosh 1996] and empirical evidence showing the utility of such approaches for solving difficult classification problems. For instance, an analytical framework to mathematically quantify the improvements in classification results due to combining multiple models has been addressed in [Tumer and Ghosh 1996]. A survey of traditional ensemble techniques — including their applications to many difficult real-world problems such as remote sensing, person recognition, one vs. all recognition, and medicine — is presented in [Oza and Tumer 2008]. In summary, the extensive literature on the subject has shown that an ensemble created from diversified classifiers is typically 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 [Wang et al. 2011; Ghosh and Acharya 2011] and references therein. Indeed, 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 [Strehl and Ghosh 2002a]. We note however that:

- Like single classifiers/clusterers, with very few exceptions [Polikar 2007], ensemble methods assume that the test or scoring data comes from the same underlying distribution as the training (and validation) data. Thus their performance degrades if the underlying input-output map changes over time.
- There is relatively little work in incorporating both labeled and unlabeled data while building ensembles, in contrast to the substantial amount of recent interest in semi-supervised learning - including semi-supervised clustering, semi-supervised classification, clustering with constraints and transductive learning methods - using a single model [Chapelle et al. 2006; Zhu and Goldberg 2009; Cai et al. 2009; Forestier et al. 2010; Chen et al. 2009].

Transfer learning emphasizes the transfer of knowledge across related domains, tasks and distributions that are similar but not the same. The domain from which the knowledge is transferred is called the “source” domain and the domain to which the knowledge is transferred is called the “target” domain. In transfer learning scenarios, the source and target distributions are somewhat different, as they represent (potentially) related but not identical tasks. The literature on transfer learning is fairly rich and varied (*e.g.*, see [Pan and Yang 2010; Silver and Bennett 2008] and references therein), with much work done in the past 15 years [Thrun and Pratt 1997]. The tasks

Table I. Different Machine Learning Algorithms

Learning Mode	Single Model	Ensemble of Raw Data	Ensemble at Output
Unsupervised	k-means, spectral clustering, mixture models, . . .	self-supervised boosting [Welling et al. 2002]	clustering ensemble [Strehl and Ghosh 2002b], Bayesian cluster ensemble [Wang et al. 2009]
Semi-supervised	self training, transductive SVM [Joachims 1999a], measure propagation [Subramanya and Bilmes 2011], . . .	multi-view learning [Sridharan and Kakade 2008]	▨ **BGCM** [Gao et al. 2009]
Transfer	Multi-task Learning [Caruana 1997]	TrAdaBoost [Dai et al. 2007b]	▨ **LWE** [Gao et al. 2008]
Supervised	SVM, decision tree, logistic regression, . . .	bagging, boosting, Bayesian model averaging	majority voting

▨ Scope of **OAC**^{3}

may be learnt simultaneously [Caruana 1997] or sequentially [Bollacker and Ghosh 2000].

The novelty of our approach lies in the utilization of the theory of both classifier and cluster ensembles to address the challenge when there is very few labeled examples from the target class. There are certain application domains such as the problem of land-cover classification of spatially separated regions, where the setting is appropriate. Moreover, one does not always need to know *a priori* whether the target is similar to the source domain. Though there is a recent paper that uses a single clustering to modify the weights of base classifiers in an ensemble in order to provide some transfer learning capability [Gao et al. 2008], that algorithm is completely different from ours.

Semi-supervised learning is a domain of machine learning where both labeled and unlabeled data are used to train a model – typically with lot of unlabeled data and only a small amount of labeled data (see [Bengio et al. 2006; Zhu and Goldberg 2009] and the references therein for more details). There are several graph-based semi-supervised algorithms that use either the graph structure to spread labels from labeled to unlabeled samples, or optimize a loss function that includes a smoothness constraint derived from the graph [Zhang et al. 2006; Subramanya and Bilmes 2009; Subramanya and Bilmes 2011]. These approaches are typically non-parametric and transductive, needing both the labeled and unlabeled data to be simultaneously available for the entire training process. **OAC^3** can use parametric classifiers so that old labeled data can be discarded once the classifier parameters are obtained, leading to additional savings in speed and storage.

A majority of previously proposed graph-based semi-supervised algorithms [Zhu and Ghahramani 2002; Joachims 2003; Belkin et al. 2005; Bengio et al. 2006] are based on minimizing an objective consisting of squared-loss, while in [Subramanya and Bilmes 2011] (Measure Propagation – **MP**), [Corduneanu and Jaakkola 2003] and [Tsuda 2005], the authors used KL divergence. The objective function of **OAC^3** uses certain Bregman divergences [Censor and Zenios 1997], among which the KL divergence and squared loss constitute just a subset (further details are provided later, in Section 4). This allows one to use well-defined functions of measures for a specific problem in order to improve performance. Additionally, the techniques of variable splitting [Boyd et al. 2011] and alternating minimization procedure [Bezdek and Hathaway 2002] are invoked to provide a more scalable solution.

The work that comes closest to ours is by Gao *et al.* [Gao et al. 2009; Gao et al. 2011], which also combines the outputs of *multiple* supervised and unsupervised models. Here, it is assumed that each model partitions the target dataset $\mathcal{X}$ into groups, so that the instances 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 with connections only between group nodes and instance nodes. A group node and an instance node are connected if the instance is assigned to the group — no matter if it comes from a supervised or unsupervised model. The authors cast the final consensus labeling as an optimization problem on this bipartite graph. To solve the optimization problem, they introduce the Bipartite Graph-based Consensus Maximization (**BGCM**) Algorithm, which is essentially a block coordinate descent based algorithm that performs an iterative propagation of probability estimates among neighboring nodes. Note that their formulation requires *hard* classification and clustering inputs. In contrast, **OAC^3** essentially processes only two fused models, namely an ensemble of classifiers and an ensemble of clusterers, the constituents of both of which can be either hard or soft. Moreover, **OAC^3** avoids solving a difficult correspondence problem — *i.e.*, aligning cluster labels to class labels — implicitly tackled by **BGCM**.

3. DESCRIPTION OF OAC^3

The proposed framework that combines classifiers and clusterers to generate a more consolidated classification is depicted in Fig. 1. It is assumed that a set of classifiers (consisting of one or more classifiers) have been previously induced from a training set. Such classifiers could have been derived from labeled and unlabeled data, and they are part of the framework that will be used for classifying new data — *i.e.*, instances from the target set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$. The target set is a test set that has not been used to build the classifiers. The classifiers are employed to estimate initial class probabilities for every instance $\mathbf{x}_i \in \mathcal{X}$. These probability distributions are stored as a set of vectors $\{\boldsymbol{\pi}_i\}_{i=1}^n$ and will be refined with the help of the clusterer(s). From this point of view, the clusterers provide supplementary constraints for classifying the instances of $\mathcal{X}$, with the rationale that similar instances are more likely to share the same class label.

Given k classes, denoted by $C = \{C_\ell\}_{\ell=1}^k$[1], each of $\boldsymbol{\pi}_i$'s is of dimension k. In order to capture the similarities between the instances of $\mathcal{X}$, **OAC^3** also takes as input a similarity matrix **S**, which can be computed from a cluster ensemble, in such a way that each matrix entry corresponds to the relative co-occurrence of two instances in the same cluster [Strehl and Ghosh 2002a] — considering all the data partitions that form the cluster ensemble induced from $\mathcal{X}$. Alternatively, **S** can be obtained from computing pair-wise similarities between instances, or from a cophenetic matrix resulting from running a hierarchical clustering algorithm. To summarize, **OAC^3** receives as inputs a set of vectors $\{\boldsymbol{\pi}_i\}_{i=1}^n$ and a similarity matrix **S** for the target set. After processing these inputs, **OAC^3** outputs a consolidated classification — represented by a set of vectors $\{\mathbf{y}_i\}_{i=1}^n \in \mathcal{S} \subseteq \mathbb{R}^k$, where $\mathbf{y}_i \propto \hat{P}(C \mid \mathbf{x}_i)$ (estimated posterior class probability assignment) — for every instance in $\mathcal{X}$. This procedure is described in more detail in the sequel.

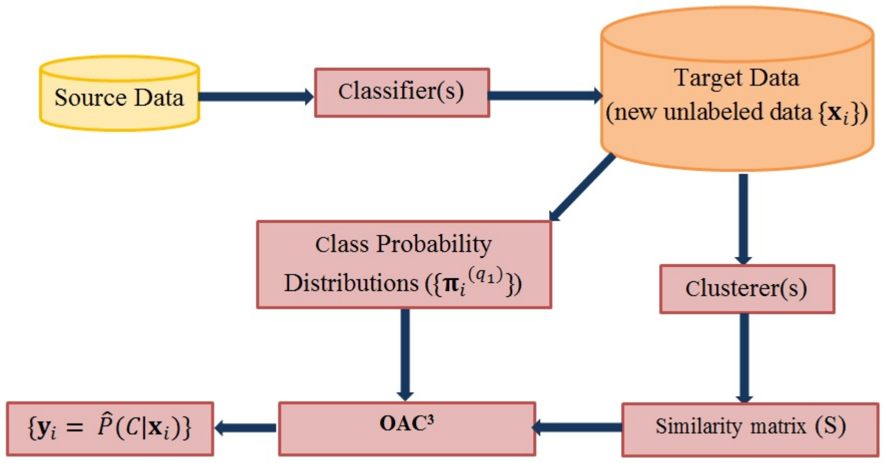
Source Data
Classifier(s)
Target Data
(new unlabeled data $\{\mathbf{x}_i\}$)
Class Probability
Distributions ($\{\boldsymbol{\pi}_i^{(q_1)}\}$)
Clusterer(s)
$\{\mathbf{y}_i = \hat{P}(C|\mathbf{x}_i)\}$
OAC^3
Similarity matrix (S)

Fig. 1. Overview of **OAC^3**.

[1]C, with an overload of notation, is used here to denote a collection of classes and should not be confused with $C^{k'}$ which is used to denote smoothness of a function.

3.1. Optimization Algorithm — OAC^3

Consider that r_1 ($r_1 \geq 1$) classifiers, indexed by q_1, and r_2 ($r_2 \geq 1$) clusterers, indexed by q_2, are employed to obtain a consolidated classification. The following steps (I-III) outline the proposed approach. Steps I and II can be seen as preliminary steps to get the inputs for **OAC^3**, while Step III is the optimization algorithm, which will be discussed in more detail.

Step I - Obtain input from classifiers. The output of classifier q_1 for instance $\mathbf{x}_i$ is a k-dimensional class probability vector $\boldsymbol{\pi}_i^{(q_1)}$. This probability vector denotes the probabilities for $\mathbf{x}_i$ being assigned to the corresponding classes (which might be soft or hard assignments). 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:

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

The reason for using an average vector will be clear after we present Theorem 3.2.

Step II - Obtain a similarity matrix. A similarity matrix can be obtained in a number of ways, such as computing pair-wise similarities between instances from the original space of features. For high-dimensional data, it is usually more appropriate to use a cluster ensemble for computing similarities between instances of the target set. In this case, after applying r_2 clustering algorithms (clusterers) to $\mathcal{X}$, a similarity matrix S is computed. Assuming that each clustering is a hard data partition (possibly obtained from a particular subspace), the similarity between two instances is simply the fraction of the r_2 clustering solutions in which those two instances lie in the same cluster[2]. Note that such similarity matrices are byproducts of several cluster ensemble solutions, e.g., the **CSPA** algorithm in [Strehl and Ghosh 2002a].

Step III - Obtain consolidated results from OAC^3. Having defined the inputs for **OAC^3**, namely the set $\{\boldsymbol{\pi}_i\}_{i=1}^n$ and the similarity matrix, S, the problem of combining classifiers and clusterers can be posed as an optimization problem whose objective is to minimize J in (2) with respect to the set of probability vectors $\{\mathbf{y}_i\}_{i=1}^n$, where $\mathbf{y}_i$ is the new and hopefully improved estimate of the aposteriori class probability distribution for a given instance in $\mathcal{X}$.[3]

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

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 **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 the sense that any Bregman divergence that satisfies some specific properties (these properties will be introduced

[2]A similarity matrix can also be defined for soft clusterings — *e.g.*, see [Punera and Ghosh 2008].
[3]From now on, for generality, we assume that we have two ensembles (a classifier ensemble and a cluster ensemble), but note that each of these ensembles may be formed by a single component.

in more detail in section 4 where the discussion is more relevant) can be used as a loss function $\mathcal{L}(\cdot,\cdot)$ in Eq. (2). So, before going into further details, the formal definition of Bregman divergence is provided.

Definition 3.1 (*[Bregman 1967], [Banerjee et al. 2005]*). Let $\phi : \mathcal{S} \to \mathbb{R}, \mathcal{S} = \text{dom}(\phi)$ be a strictly convex function defined on a convex set $\mathcal{S} \subseteq \mathbb{R}^k$ such that ϕ is differentiable on $\text{ri}(\mathcal{S})$, which is assumed to be nonempty. The Bregman divergence $d_\phi : \mathcal{S} \times \text{ri}(\mathcal{S}) \to [0,\infty)$ is defined as $d_\phi(p,q) = \phi(p) - \phi(q) - \langle p-q, \boldsymbol{\nabla}_\phi(q)\rangle$, where $\boldsymbol{\nabla}_\phi(q)$ represents the gradient vector of ϕ evaluated at q.

A specific Bregman Divergence (*e.g.* KL-divergence) between two vectors $\mathbf{y}_i$ and $\mathbf{y}_j$ can be identified by a corresponding strictly convex function ϕ (*e.g.* negative entropy for KL-divergence), and hence be written as $d_\phi(\mathbf{y}_i,\mathbf{y}_j)$. Following from Definition 3.1, $d_\phi(\mathbf{y}_i,\mathbf{y}_j) \geq 0 \;\forall \mathbf{y}_i \in \mathcal{S}, \mathbf{y}_j \in \text{ri}(\mathcal{S})$ and equality holds if and only if $\mathbf{y}_i = \mathbf{y}_j$. Using this notation, the objective function of $\mathbf{OAC^3}$, that is going to be minimized over $\{\mathbf{y}_i\}_{i=1}^n$, can be rewritten as:

$$J_0 = \left[\sum_{i\in\mathcal{X}} d_\phi(\boldsymbol{\pi}_i,\mathbf{y}_i) + \alpha \sum_{(i,j)\in\mathcal{X}} s_{ij} d_\phi(\mathbf{y}_i,\mathbf{y}_j)\right]. \tag{3}$$

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 Theorem 3.2 in the sequel 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 [Banerjee et al. 2005] 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. In what follows, it will be clear that the right and left copies are updated iteratively, and an additional soft constraint is used to ensure that the two copies of a variable remain "close enough" during the updates. With this modification, we propose minimizing the following objective $J : \mathcal{S}^n \times \mathcal{S}^n \to [0,\infty)$:

$$J(\mathbf{y}^{(l)},\mathbf{y}^{(r)}) = \left[\sum_{i=1}^n d_\phi(\boldsymbol{\pi}_i,\mathbf{y}_i^{(r)}) + \alpha \sum_{i,j=1}^n s_{ij} d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_j^{(r)}) + \lambda \sum_{i=1}^n d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_i^{(r)})\right], \tag{4}$$

where, $\mathbf{y}^{(l)} = \left(\mathbf{y}_i^{(l)}\right)_{i=1}^n \in \mathcal{S}^n$ and $\mathbf{y}^{(r)} = \left(\mathbf{y}_i^{(r)}\right)_{i=1}^n \in \mathcal{S}^n$.

To solve the optimization problem in an efficient way, we first keep $\{\mathbf{y}_i^{(l)}\}_{i=1}^n$ and $\{\mathbf{y}_i^{(r)}\}_{i=1}^n \setminus \{\mathbf{y}_j^{(r)}\}$ fixed, and minimize the objective w.r.t. $\mathbf{y}_j^{(r)}$ only. The problem can, therefore, be written as:

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

where $\lambda_j^{(r)}$ is the corresponding penalty parameter that is used to keep $\mathbf{y}_j^{(r)}$ and $\mathbf{y}_j^{(l)}$ close to each other. For every valid assignment of $\{\mathbf{y}_i^{(l)}\}_{i=1}^n$, it can be shown that there is a unique minimizer $\mathbf{y}_j^{(r)^*}$ for the optimization problem in (5). For that purpose, a new Corollary is developed from the results of Theorem 3.2 [Banerjee et al. 2005] that is stated below.

THEOREM 3.2 ([BANERJEE ET AL. 2005]). *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 $d_\phi \colon \mathcal{S} \times ri(\mathcal{S}) \to [0, \infty)$, the optimization problem $\min_{\mathbf{s} \in ri(\mathcal{S})} \mathbb{E}_v[d_\phi(Y, \mathbf{s})]$ has a unique minimizer given by $\mathbf{s}^* = \boldsymbol{\mu} = \mathbb{E}_v[Y]$.*

To solve the problem formulated in Eq. (5), the following corollary is required:

COROLLARY 3.3. *Let $\{Y_i\}_{i=1}^m$ be a set of random variables, each of which takes values in $\mathcal{Y}_i = \{\mathbf{y}_{ij}\}_{j=1}^{n_i} \subset \mathcal{S} \subseteq \mathbb{R}^k$ following a probability measure v_i such that $\mathbb{E}_{v_i}[Y_i] \in ri(\mathcal{S})$. Consider a Bregman divergence $d_\phi \colon \mathcal{S} \times ri(\mathcal{S}) \to [0, \infty)$ 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. Since $\mathbb{E}_{v_i}[Y_i] \in \mathrm{ri}(\mathcal{S}) \ \forall i$, their convex combination should also belong to $\mathrm{ri}(\mathcal{S})$, implying that $\boldsymbol{\mu} \in \mathrm{ri}(\mathcal{S})$. Now $\forall s \in \mathrm{ri}(\mathcal{S})$ we have:

$$J_\phi(\boldsymbol{s}) - J_\phi(\boldsymbol{\mu}) = \sum_{i=1}^m \alpha_i \mathbb{E}_{v_i}[d_\phi(Y_i, \boldsymbol{s})] - \sum_{i=1}^m \alpha_i \mathbb{E}_{v_i}[d_\phi(Y_i, \boldsymbol{\mu})]$$

$$= \sum_{i=1}^m \alpha_i \left[\phi(\boldsymbol{\mu}) - \phi(\boldsymbol{s})\right] \ - \ \sum_{i=1}^m \alpha_i \Big\langle \sum_{j=1}^n v_{ij} y_{ij} - \boldsymbol{s}, \boldsymbol{\nabla}_\phi(\boldsymbol{s}) \Big\rangle$$
$$+ \ \sum_{i=1}^m \alpha_i \Big\langle \sum_{j=1}^n v_{ij} y_{ij} - \boldsymbol{\mu}, \boldsymbol{\nabla}_\phi(\boldsymbol{\mu}) \Big\rangle$$

$$= \sum_{i=1}^m \alpha_i \left[\phi(\boldsymbol{\mu}) - \phi(\boldsymbol{s}) - \langle \boldsymbol{\mu} - \boldsymbol{s}, \boldsymbol{\nabla}_\phi(\boldsymbol{s}) \rangle\right] = d_\phi(\boldsymbol{\mu}, \boldsymbol{s}) \sum_{i=1}^m \alpha_i \geq 0$$

with equality only when $\boldsymbol{s} = \boldsymbol{\mu}$ following the strict convexity of ϕ. Hence, $\boldsymbol{\mu}$ is the unique minimizer of the objective function J_ϕ. $\square$

We are now in a position to explain why an avergae vector is used in Eq. 1. The summation $\frac{1}{r_1} \sum_{q_1=1}^{r_1} d_\phi(\pi_i^{(q_1)}, y_i)$, according to Theorem 3.2, is lower bounded by the term $d_\phi(\pi_i, y_i)$. Therefore, minimizing the objective in Eq. 3 is equivalent to minimizing an objective consisting of the terms involving the above summation.

From the results of Corollary 3.3, the unique minimizer of the optimization problem in (5) is obtained as:

$$\mathbf{y}_j^{(r)^*} = \frac{\boldsymbol{\pi}_j^{(r)} + \gamma_j^{(r)} \sum_{i^{(l)} \in \mathcal{X}} \delta_{i^{(l)}j^{(r)}} \mathbf{y}_i^{(l)} + \lambda_j^{(r)} \mathbf{y}_j^{(l)}}{1 + \gamma_j^{(r)} + \lambda_j^{(r)}}, \tag{6}$$

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 with respect to 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 difference between the left and right copies of $\mathbf{y}_i$ is penalized, so that the optimization with respect to $\mathbf{y}_i^{(l)}$ can be rewritten as:

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

where $\lambda_i^{(l)}$ is the corresponding penalty parameter. 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:

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

One can show that $\forall \mathbf{y}_i, \mathbf{y}_j \in \text{int}(\text{dom}(\phi))$, $d_\phi(\mathbf{y}_i, \mathbf{y}_j) = d_\psi(\boldsymbol{\nabla}_\phi(\mathbf{y}_j), \boldsymbol{\nabla}_\phi(\mathbf{y}_i))$ — see [Banerjee et al. 2005] for more details. Thus, the optimization problem in (7) can be rewritten in terms of the Bregman divergence associated with ψ as follows:

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

The unique minimizer of the problem in (9) can be computed using Corollary 3.3. $\boldsymbol{\nabla}_\phi$ is monotonic and invertible for ϕ being strictly convex and hence the inverse of the unique minimizer for the problem in (9) is also unique and equals to the unique minimizer for the problem in (7). Therefore, the unique minimizer of the problem in (7) with respect to $\mathbf{y}_i^{(l)}$ is given by:

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

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:

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

Algorithm 1 — OAC^3

```
Inputs: $\{\boldsymbol{\pi}_i\}$, 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 Eq. (6) $\forall j \in \{1, 2, \cdots, n\}$.
    Step 2: Update $\boldsymbol{y}_i^{(l)}$ using Eq. (10) $\forall i \in \{1, 2, \cdots, n\}$.
End Loop
Step 3: Compute $\boldsymbol{y}_i = 0.5[\boldsymbol{y}_i^{(l)} + \boldsymbol{y}_i^{(r)}]$ $\forall i \in \{1, 2, \cdots, n\}$.
Step 4: Normalize $\boldsymbol{y}_i$ $\forall i \in \{1, 2, \cdots, n\}$.
```

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

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

where part of the superscript "$, I$" indicates that the optimal value corresponds to I-divergence. Optimization over the left and right arguments of all the instances constitutes one pass (iteration) of the algorithm, and these two steps are repeated till convergence (a detailed proof for convergence will be given in Section 4). Upon convergence, all the $\mathbf{y}_i$'s are normalized to unit L_1 norm after averaging over the respective left and right copies, to yield the individual class probability distributions for every instance $\mathbf{x}_i \in \mathcal{X}$. The main steps of **OAC^3** are summarized in Algorithm 1.

The update procedure captured by Eq. (10) deserves some special attention. Depending on the divergence used, the update might not ensure that the left copies returned are in the correct domain. For example, if KL divergence is used, Eq. (10) will not necessarily produce probabilities. In that case, one needs to use another Lagrangian multiplier to make sure that the returned values lie on simplex as it has been done in [Subramanya and Bilmes 2011].

3.2. Time Complexity Analysis of OAC^3

Considering that a trained ensemble of classifiers is available, the computation of the set of vectors $\{\boldsymbol{\pi}_i\}_{i=1}^{n}$ requires $O(nr_1k)$, where n is the number of instances in the target set, r_1 is the number of components of the classifier ensemble, and k is the number of class labels. Computing the similarity matrix, S, is $O(r_2n^2)$, where r_2 is the number of components of the cluster ensemble. Finally, having $\{\boldsymbol{\pi}_i\}_{i=1}^{n}$ and S available, the computational cost (per iteration) of **OAC^3** is $O(kn^2)$. In practice, usually the number of classes k is small and thus the dominant variables are the number of target instances to be classified, n, and the number of supervised and unsupervised models, r. From this standpoint, note that the time complexity of **OAC^3** is quadratic with n and linear with the number of the components of the ensembles r. Actually, the computational bottleneck of **OAC^3** is not the optimization algorithm itself, whose main steps (1 and 2) can be parallelized (this can be identified by a careful inspection of Eq. (6) and (10)), but the computation of the similarity matrix. Note that low values in the similarity

Table II. Examples of Bregman divergences that satisfy properties (a) to (f)

Domain	$\phi(\mathbf{p})$	$d_\phi(\mathbf{p},\mathbf{q})$	Divergence
$\mathbb{R}$	p^2	$(p-q)^2$	Squared Loss
$[0,1]$	$p\text{log}(p)+(1-p)\text{log}(1-p)$	$p\text{log}(\frac{p}{q})+(1-p)\text{log}(\frac{1-p}{1-q})$	Logistic Loss
$\mathbb{R}_+$	$p\text{log}(p)-(1+p)\text{log}(1+p)$	$p\text{log}(\frac{p}{q})-(1+p)\text{log}(\frac{1+p}{1+q})$	Bose-Einstein Entropy
$\mathbb{R}_{++}$	$-\text{log}(p)$	$\frac{p}{q}-\text{log}(\frac{p}{q})-1$	Itakura-Saito Distance
$\mathbb{R}^k$	$\|\|p\|\|^2$	$\|\|p-q\|\|^2$	Squared Euclidean Distance
k-simplex	$\sum_{i=1}^{k} p_i\text{log}_2(p_i)$	$\sum_{i=1}^{k} p_i\text{log}_2(\frac{p_i}{q_i})$	KL-Divergence
$\mathbb{R}_+^k$	$\sum_{i=1}^{k} p_i\text{log}(p_i)$	$\sum_{i=1}^{k} p_i\text{log}(\frac{p_i}{q_i})-\sum_{i=1}^{k}(p_i-q_i)$	Generalized I-Divergence

matrix can often be zeroed out to further speed up the computation, without having much impact on the results.

4. CONVERGENCE ANALYSIS OF OAC^3

We claim that **OAC^3** makes the objective J in Eq. 4 converge to some unique minimizer when Bregman divergences with the following properties are used as loss functions:

(a) $d_\phi(\mathbf{p},\mathbf{q})$ is strictly convex in $\mathbf{p}$ and $\mathbf{q}$ separately.
(b) $d_\phi(\mathbf{p},\mathbf{q})$ is jointly convex w.r.t $\mathbf{p}$ and $\mathbf{q}$.
(c) The level sets $\{\mathbf{q} : d_\phi(\mathbf{p},\mathbf{q}) \leq r\}$ are bounded for any given $\mathbf{p} \in \mathcal{S}$.
(d) $d_\phi(\mathbf{p},\mathbf{q})$ is lower-semi-continuous in $\mathbf{p}$ and $\mathbf{q}$ jointly.
(e) If $d_\phi(\mathbf{p}^t,\mathbf{q}^t) \to 0$ and $\mathbf{p}^t$ or $\mathbf{q}^t$ is bounded, then $\mathbf{p}^t \to \mathbf{q}^t$ and $\mathbf{q}^t \to \mathbf{p}^t$.
(f) If $\mathbf{p} \in \mathcal{S}$ and $\mathbf{q}^t \to \mathbf{p}$, then $d_\phi(\mathbf{p},\mathbf{q}^t) \to 0$.

Bregman divergences that satisfy the above properties include a large number of useful loss functions such as the well-known squared loss, KL-divergence, generalized I-divergence, logistic loss, Itakura-Saito distance and Bose-Einstein entropy [Wang and Schuurmans 2003a]. These divergences along with their associated strictly convex functions $\phi(.)$ and domains are listed in Table II.

An alternating optimization algorithm, in general, is not guaranteed to converge. Even if it converges it might not converge to the locally optimal solution. Some authors [Cheney and Goldstein 1959; Zangwill 1969; Wu 1982; Bezdek and Hathaway 2003] have shown that the convergence guarantee of alternating optimization can be analyzed using the topological properties of the objective and the space over which it is optimized. Others have used information geometry [Csiszár and Tusnády 1984; Wang and Schuurmans 2003b; Subramanya and Bilmes 2011] to analyze the convergence as well as a combination of both information geometry and topological properties of the objective [Gunawardana and Byrne 2005]. In this paper, the information geometry approach is utilized to show that the proposed optimization procedure converges to the global minima of the objective J in 4.

At this point it is worth mentioning the connection of the optimization framework with other related approaches. Label Propagation ([Zhu and Ghahramani 2002] – **LP**) is one of the related algorithms which works on minimizing squared-loss defined over the reals. Harmonic Function ([Zhu et al. 2003] – **HF**) algorithms also optimize the same objective as **LP**. However, **LP** uses an iterative solution and **HF** uses closed form solution and both algorithms attain the same optimal solution. Quadratic Cost criterion ([Bengio et al. 2006] – **QC**) augments a regularizer to the objective functions used in both **LP** and **HF**. **LP**, **HF** and, **QC** are only suitable for binary classification problems and multi-class extension is entirely based on one-vs-all strategy. Interestingly,

QC has been shown to generalize **HF**. **MP** [Subramanya and Bilmes 2011], on the other hand, employs KL divergence as a loss function and hence is suitable for multi-class problems. Moreover, it provides guard against degenerate solutions (those that assign equal confidence to all classes) – similar to the regularizer term used in **QC**. From that perspective, if the objective consisting of KL divergence in **MP** is replaced by the squared loss term, it generalizes **QC** to multi-class problems. Graph transduction, proposed in [Wang et al. 2008], employs alternating minimization type algorithms to solve a squared loss based objective. However, the convergence guarantee of the algorithm is not proven and one of the updates involves solving an NP-complete optimization problem. The objective function used in **OAC^3** does not guard against degenerate solutions but can easily be extended to alleviate the same problem with the addition of a single tuning parameter (and its associated regularizer). In the experiments reported, no significant difference in performance is observed with this extension and hence it is discarded to help tune one less model parameter. However, it should be noted that with such regularizer augmented, the objective of **OAC^3** generalizes that of **MP** to a larger class of Bregman divergences. In [Subramanya and Bilmes 2011], the authors proved that their algorithm converges (for KL divergence) but the convergence rate (for KL divergence) is not proven and only empirical evidence is given for a linear rate. In this paper, apart from generalizing these algorithms and establishing their convergence with a larger class of Bregman divergences, we provide proofs for linear rate of convergence for generalized I divergence and KL divergence (the proof for squared loss follows directly from the analysis of [Subramanya and Bilmes 2011]). Note that although the objective for the optimization problem shares some commonality with the graph based semi-supervised algorithms, their application settings are different.

In another related domain, spectral graph transduction Joachims 2003 provides an approximate solution to the NP-hard norm-cut problem. However, this algorithm requires eigen-decomposition of a matrix of size $n \times n$, where n is the number of instances, which is inefficient for very large data sets. Manifold regularization [Belkin et al. 2005] is a general framework in which a parametric loss function is defined over the labeled samples and is regularized by graph smoothness term defined over both the labeled and unlabeled samples. In the algorithms proposed therein, one either needs to invert an $n \times n$ matrix or use optimization techniques for general SVM in case there is no closed form solution. Both **OAC^3** and **MP**, on the other hand, have closed form solutions corresponding to each update and hence are perfectly suitable for large scale applications. Information regularization [Corduneanu and Jaakkola 2003], in essence, works on the same intuition as **OAC^3**, but does not provide any proof of convergence and one of the steps of the optimization does not have a closed form solution – a concern for large data applications. [Tsuda 2005] extended the works of [Corduneanu and Jaakkola 2003] to hyper-graphs and used closed form solutions in both steps of the alternating minimization procedure which, surprisingly, can be seen as a special case of **MP**.

We now give a sketch of the proof of convergence of **OAC^3**. The so-called 5-points property (5-pp) [Csiszár and Tusnády 1984] of the objective function J is essential to analyze the convergence. If J satisfies the 3-points property (3-pp) and the 4-points property (4-pp), then it satisfies the 5-pp. All of these properties are explained in details in Appendix A. To prove 5-pp of J, we will try to prove that it satisfies both 3-pp and 4-pp. However, this proof is not easy for any arbitrary Bregman divergence. In [Subramanya and Bilmes 2011], the authors followed the procedure of [Csiszár and Tusnády 1984] to prove the convergence of a slightly different objective that involves KL-divergence as a loss function. The proof there is specific to KL-divergence and does not generalize to Bregman divergences with properties (a) to (f). Therefore, we take a more subtle route in proving the 3-pp and 4-pp of J. We show that the objective function J, which is a sum of Bregman divergences of different pairs of variables, can itself be thought of as a Bregman divergence in some joint space. This Bregman divergence also satisfies the properties (a) to (f), which then allows one to use the convergence tools developed by [Wang and Schuurmans 2003a]. The formal proof for convergence is placed in appendix A to facilitate an easy perusal of the paper.

5. ANALYSIS OF RATE OF CONVERGENCE FOR OAC^3

In practical applications, the rate of convergence of any optimization algorithm is of great importance. To analyze the same, we use some formulations that were derived in [Bezdek and Hathaway 2003] to characterize the local convergence rate of alternating minimization type of algorithms in general. In this section, we will first explain the tools and then show that the analysis applies to the objective function J seamlessly. The details of the tools are skipped here though and only the main lemmata and theorems are provided.

5.1. Tools for Analyzing Local Rate of Convergence

Let us consider a variable $\mathbf{z} \in \mathcal{S}^{2n}$ where $\mathbf{z} = (\mathbf{z}_{n'})_{n'=1}^{2n}$ and $\mathbf{z}_{n'} \in \mathcal{S}\ \forall n'$. Assume functions $M_{n'} : \mathcal{S}^{2n-1} \rightarrow \mathcal{S}\ \forall n'$ which are defined as:

$$M_{n'}(\tilde{\mathbf{z}}_{n'}) = \underset{\mathbf{z}_{n'} \in \mathcal{S}}{\text{argmin}} f(\mathbf{z}_1, \cdots, \mathbf{z}_{n'-1}, \mathbf{z}_{n'}, \mathbf{z}_{n'+1}, \cdots, \mathbf{z}_{2n}) \tag{13}$$

Here, $\tilde{\mathbf{z}}_{n'} = (\mathbf{z}_1, \cdots, \mathbf{z}_{n'-1}, \mathbf{z}_{n'+1}, \cdots, \mathbf{z}_{2n})$. Corresponding to each $M_{n'}$ we also define a function $C_{n'} : \mathcal{S}^{2n} \rightarrow \mathcal{S}^{2n}$ as:

$$C_{n'}(\mathbf{z}_1, \cdots, \mathbf{z}_{n'-1}, \mathbf{z}_{n'}, \mathbf{z}_{n'+1}, \cdots, \mathbf{z}_{2n}) = (\mathbf{z}_1, \cdots, \mathbf{z}_{n'-1}, M_{n'}(\tilde{\mathbf{z}}_{n'}), \mathbf{z}_{n'+1}, \cdots, \mathbf{z}_{2n}) \tag{14}$$

Moreover, one complete execution of alternating minimization step can conveniently be represented by a function $\mathbb{S} : \mathcal{S}^{2n} \rightarrow \mathcal{S}^{2n}$:

$$\mathbb{S}(\mathbf{z}) = C_1 \circ C_2 \circ \cdots C_{2n}(\mathbf{z}). \tag{15}$$

LEMMA 5.1. *Let $f : \mathcal{S}^{2n} \rightarrow \mathbb{R}$ satisfy the following conditions:*

(a) f is C^2 in a neighborhood of $\mathbf{z}^*$, $\mathbf{z}^*$ being a local minimizer of f;
(b) $\nabla^2 f(\mathbf{z}^*)$ is positive definite;
(c) There is a neighborhood $\mathcal{N}$ of $\mathbf{z}^*$ on which f is strictly convex, and such that for $n' \in \{1, 2, \cdots, 2n\}$ if $\mathbf{z} = \mathbf{z}^*_{\mathbf{z}_{n'}}$ locally minimizes $g_{n'}(\mathbf{z}_{\mathbf{z}_{n'}}) = f(\mathbf{z}_{\mathbf{z}_{n'}})$ with $\mathbf{z}_{\mathbf{z}_{n'}}$ indicating that all variables except $\mathbf{z}_{n'}$ are held fixed, then $\mathbf{z}^*_{\mathbf{z}_{n'}}$ is also the unique global minimizer of $g_{n'}(\mathbf{z}_{\mathbf{z}_{n'}})$.

Then in some neighborhood of $\mathbf{z}^*$, the minimizing function $M_{n'}$ exists and is continuously differentiable $\forall n' \in \{1, 2, 3, \cdots, 2n\}$.

LEMMA 5.2. *Let $f : \mathcal{S}^n \rightarrow \mathbb{R}$ be differentiable and satisfy the conditions of Lemma 5.1. Then $\rho(\nabla_{\mathbb{S}}(\mathbf{z}^*)) < 1$ where $\nabla_{\mathbb{S}}(\mathbf{z}^*)$ is the Jacobian of the mapping $\mathbb{S}$ evaluated at $\mathbf{z}^*$ and ρ is the spectral radius of the Jacobian.*

Before presenting the main theorem from [Bezdek and Hathaway 2003], the formal definition of q-linear rate of convergence is provided below. The "q" in this definition stands for quotient.

Definition 5.3 (*q-linear rate of convergence*). A sequence $\{\mathbf{z}^{(t)}\} \rightarrow \mathbf{z}^*$ q-linearly iff $\exists t_0 \geq 0$ and $\exists \rho \in [0, 1)$ such that $\forall t \geq t_0$, $||\mathbf{z}^{(t+1)} - \mathbf{z}^*|| \leq \rho ||\mathbf{z}^{(t)} - \mathbf{z}^*||$

THEOREM 5.4. *Let* $\mathbf{z}^*$ *be a local minimizer of* $f : \mathcal{S}^n \to \mathbb{R}$ *for which* $\nabla^2 f(\mathbf{z}^*)$ *is positive definite and let* f *be* C^2 *in a neighborhood of* $\mathbf{z}^*$. *Also let assumption (c) of Lemma 5.2 hold for* $\mathbf{z}^*$. *Then there is a neighborhood* $\mathcal{N}$ *of* $\mathbf{z}^*$ *such that for any* $\mathbf{z}^{(0)} \in \mathcal{N}$, *the corresponding iteration sequence* $\{\mathbf{z}^{(t+l)} = \mathbb{S}(\mathbf{z}^{(t)}) : t = 0, 1, ...\}$ *converges q-linearly to* $\mathbf{z}^*$.

5.2. Hessian Calculation of J

From the theorems and lemmata presented in the previous subsection, one can observe that the Hessian of the objective being positive definite is a critical condition. Therefore, we will try to show that $\nabla^2 J$ is positive definite for some of the Bregman divergences. According to Eq. (4), ∇J involves the following terms:

$$\nabla_{\mathbf{y}_i^{(l)}} J = \alpha \sum_{j=1;j\neq i}^{n} s_{ij} \left[\nabla_\phi(\mathbf{y}_i^{(l)}) - \nabla_\phi(\mathbf{y}_j^{(r)})\right] + \lambda \left[\nabla_\phi(\mathbf{y}_i^{(l)}) - \nabla_\phi(\mathbf{y}_i^{(r)})\right]$$

$$\nabla_{\mathbf{y}_j^{(r)}} J = \Bigg[(\nabla_\phi(\mathbf{y}_j^{(r)}) - \nabla_\phi(\boldsymbol{\pi}_j)) + \alpha \sum_{j=1;j\neq i}^{n} (\nabla_\phi(\mathbf{y}_j^{(r)}) - \nabla_\phi(\mathbf{y}_i^{(l)})) + \lambda \left[\nabla_\phi(\mathbf{y}_j^{(r)}) - \nabla_\phi(\mathbf{y}_i^{(l)})\right]\Bigg]^\dagger \nabla^2_\phi(\mathbf{y}_j^{(r)}).$$

$\nabla^2 J$, derived from the above equations, has the following terms:

$$\nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_i^{(l)}} J = \big(\alpha \sum_{j=1;j\neq i}^{n} s_{ij} + \lambda\big) \nabla^2_\phi(\mathbf{y}_i^{(l)})$$

$$\nabla^2_{\mathbf{y}_j^{(r)},\mathbf{y}_j^{(r)}} J = \left[\big(1 + \alpha \sum_{i=1;j\neq i}^{n} s_{ij} + \lambda\big)\mathbf{y}_j^{(r)} - \boldsymbol{\pi}_j - \alpha \sum_{i=1;j\neq i}^{n} s_{ij}\mathbf{y}_i^{(l)} - \lambda \mathbf{y}_j^{(l)}\right]^\dagger \nabla^3_\phi(\mathbf{y}_j^{(r)}) + \big(1 + \alpha \sum_{i=1;j\neq i}^{n} s_{ij} + \lambda\big)\nabla^2_\phi(\mathbf{y}_j^{(r)})$$

$$\nabla^2_{\mathbf{y}_j^{(r)},\mathbf{y}_i^{(l)}} J = \nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_j^{(r)}} J = -\alpha s_{ij} \nabla^2_\phi(\mathbf{y}_j^{(r)}) (i \neq j)$$

$$\nabla^2_{\mathbf{y}_i^{(r)},\mathbf{y}_i^{(l)}} J = \nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_i^{(r)}} J = -\lambda \nabla^2_\phi(\mathbf{y}_i^{(r)})$$

$$\nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_j^{(l)}} J = \nabla^2_{\mathbf{y}_i^{(r)},\mathbf{y}_j^{(r)}} J = 0, (i \neq j)$$

Note that this calculation is valid for any Bregman divergence within the assumed family.

5.3. Hessian Calculation for KL and Generalized I divergence

We are now in a position to show that the Hessian of the objective J is positive definite when KL or I-divergence is used as Bregman divergence. Recall from table II that the generating functions $\phi(.)$'s for KL and I-divergence differ only by a linear term and hence the Hessian of the objective J would be the same for these two cases. We list different terms of the Hessian here:

$$\nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_i^{(l)}} J = \left(\alpha \sum_{j=1;j\neq i}^{n} s_{ij} + \lambda\right) \mathbf{diag}\big((1/\mathbf{y}_{i\ell}^{(l)})_{\ell=1}^{k}\big) \tag{16}$$

$$\nabla^2_{\mathbf{y}_j^{(r)},\mathbf{y}_j^{(r)}} J = \mathbf{diag}\left(\left(\frac{\boldsymbol{\pi}_{j\ell} + \alpha \sum_{i=1;j\neq i}^{n} s_{ij}\mathbf{y}_{i\ell}^{(l)} + \lambda \mathbf{y}_{j\ell}^{(l)}}{\mathbf{y}_j^{(r\ell)^2}}\right)_{\ell=1}^{k}\right) \tag{17}$$

$$\nabla^2_{\mathbf{y}_j^{(r)},\mathbf{y}_i^{(l)}} J = \nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_j^{(r)}} J = -\alpha s_{ij}\mathbf{diag}\big((1/\mathbf{y}_{j\ell}^{(r)})_{\ell=1}^{k}\big)(i \neq j) \tag{18}$$

$$\nabla^2_{\mathbf{y}_i^{(r)},\mathbf{y}_i^{(l)}} J = \nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_i^{(r)}} J = -\lambda\mathbf{diag}\big((1/\mathbf{y}_{i\ell}^{(r)})_{\ell=1}^{k}\big) \tag{19}$$

$$\nabla^2_{\mathbf{y}_i^{(l)},\mathbf{y}_j^{(l)}} J = \nabla^2_{\mathbf{y}_i^{(r)},\mathbf{y}_j^{(r)}} J = 0, (i \neq j). \tag{20}$$

Using Eqs. (16) to (20) and some simple algebra, the following lemma can be proved.

LEMMA 5.5. *$\mathcal{H} = \nabla^2 J$ is positive definite over the domain of J under the assumption $\sum_{i=1}^{n}\sum_{\ell=1}^{k} \pi_{i\ell} > 0$ when KL or generalized I divergence is used as a Bregman divergence.*

The proof is placed in Appendix B.

5.4. Convergence Rate of OAC³ with KL and I-divergence

Following Lemma 5.5, $\mathcal{H}$ is positive definite if $\sum_{i=1}^{n}\sum_{\ell=1}^{k} \pi_{i\ell} > 0$. This is always the case as $\boldsymbol{\pi}_i$ represents some probability assignment. Also, if generalized I divergence or KL divergence is used as the Bregman divergence, $J \in C^{\infty}$ (*i.e.* J is a smooth function). From Lemma A.1, we have that J is jointly strictly convex and hence has a unique minimizer. From the same Lemma, J is separately strictly convex w.r.t each of its arguments. Therefore, with other variables fixed at some value, J has a unique minimizer w.r.t one particular variable. Hence, all the conditions mentioned in Lemma 5.1 are satisfied for J in its entire domain. Therefore, following Theorem 5.4 we can conclude that J converges globally (implying that $\mathcal{N} = \text{dom}(J)$) to its unique minimizer q-linearly using **OAC³**. Note that when the Bregman divergence is the squared Euclidean distance, variable splitting is not required at all. The updates involve only one set of copies (*i.e.* there is no need to maintain left and right copies) and the q-linear rate of convergence of the objective J can be proved following the same method as done in [Subramanya and Bilmes 2011]. The proof uses Perron-Frobenius theorem to bound the maximum eigen-value of the transformation matrix used to update the values of the probability assignments. Thus, **OAC³** converges q-linearly at least when squared Euclidean, KL or I divergence is used as loss function. One needs to compute the Hessian or use some other tricks for other Bregman divergences having properties (a) to (f).

6. EXPERIMENTAL EVALUATION

First we provide a simple pedagogical example that illustrates how the supplementary constraints provided by clustering algorithms can be useful for improving the generalization capability of classifiers. Section 6.2 reports sensitivity analyses on the **OAC^3** parameters. Then, in Section 6.3, we compare the performance of **OAC^3** with the recently proposed **BGCM** [Gao et al. 2009; Gao et al. 2011]. This comparison is straightforward and fair, since it uses the same datasets, as well as the same outputs of the base models, which were kindly provided by the authors of this paper. For a comparison with other semi-supervised methods, the design space is much larger, since we are now faced with a variety of classification and clustering algorithms to choose from as the base models in **OAC^3**, as well as a variety of semi-supervised methods to compare with. In Section 6.4 we use simple (linear) base methods, and pick the popular Semi-Supervised Linear Support Vector Machine (**S^{3}VM**) [Sindhwani and Keerthi 2006] for comparison. Finally, in Section 6.5 we report empirical results for transfer learning settings.

6.1. Pedagogical Example

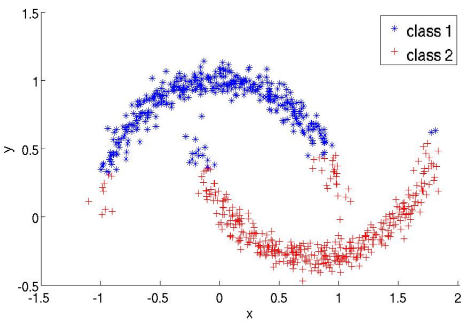

Fig. 2. Class Labels from the Classifier Ensemble.

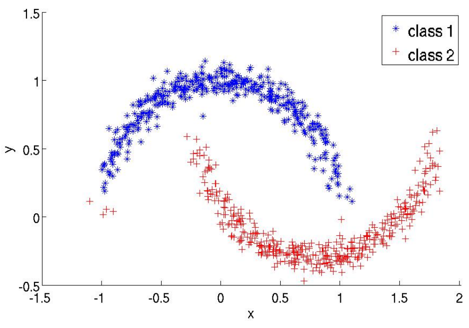

Fig. 3. Class Labels from **OAC^3**.

Consider the two-dimensional dataset known as *Half-Moon*, which has two classes, each of which represented by 400 instances. From this dataset, 2% of the instances are used for training, whereas the remaining instances are used for testing (target set). A classifier ensemble formed by three well-known classifiers (Decision Tree, Linear Discriminant, and Generalized Logistic Regression) are adopted. In order to get a cluster ensemble, a single linkage (hierarchical) clustering algorithm is chosen. The cluster ensemble is then obtained from five data partitions represented in the dendrogram, which is cut for different number of clusters (from 4 to 8). Fig. 2 shows the target data class labels obtained from the standalone use of the classifier ensemble, whereas Fig. 3 shows the corresponding results achieved by **OAC^3**. The parameter values were set by using cross-validation. In particular, we set $\alpha = 0.0001$ and $\lambda_i^{(r)} = \lambda_i^{(l)} = \lambda = 0.1$ for all i. Comparing Fig. 2 to Fig. 3, one can see that **OAC^3** does a better job, especially with the most difficult objects to be classified, showing that the information provided by the similarity matrix can improve the generalization capability of classifiers.

We also evaluate the performance of **OAC^3** for different proportions (from 1% to 50%) of training data. Fig. 4 summarizes the average accuracies (over 10 trials) achieved by **OAC^3**. The accuracies provided by the classifier ensemble, as well as by its *best* individual component, are also shown for comparison purposes. The results obtained by **OAC^3** are consistently better than those achieved by the classifier ensemble. As expected, the curve for **OAC³** shows that the less the amount of labeled objects, the greater are the benefits of using the information provided by the cluster ensemble. With 2% of training data, the accuracies observed are 100% in nine trials and 95% in one trial. The mean and standard deviation are 99.5 and 1.59 respectively. This explains why the error bar exceeds 100%.

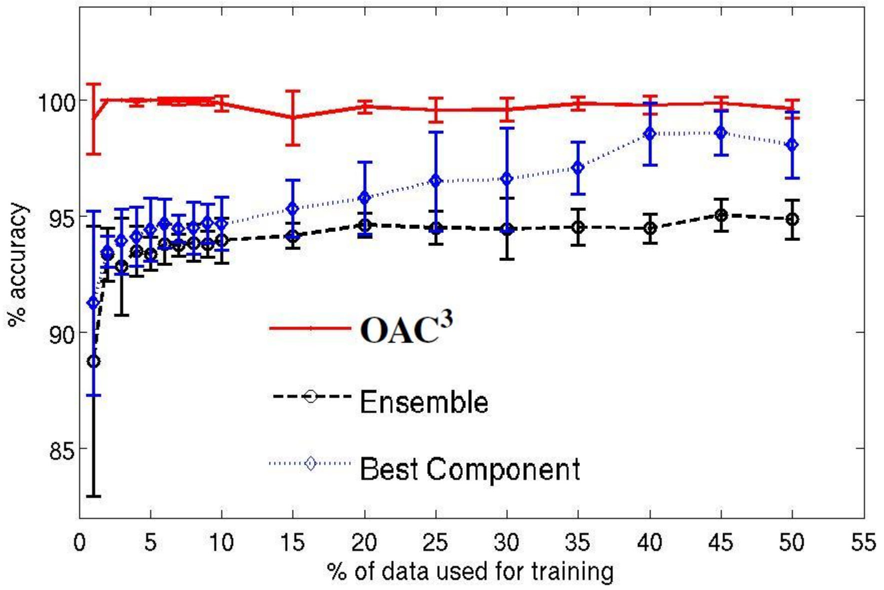

Fig. 4. Average Accuracies and Standard Deviations.

6.2. Sensitivity Analysis

We perform a sensitivity analysis on the **OAC³** parameters by using the same classification datasets employed in [Gao et al. 2009]. These datasets represent eleven classification tasks from three real-world applications (*20 Newsgroups*, *Cora*, and *DBLP*). There are six datasets (News1 — News6) for *20 Newsgroups* and four datasets (Cora1 — Cora4) for *Cora*. In each task, there is a target set on which the class labels should be predicted. In [Gao et al. 2009], two supervised models and two unsupervised models were used to obtain (on the target sets) class and cluster labels, respectively. These same class and cluster labels are used as inputs to **OAC³**. Then, we vary the **OAC³** parameters and observe their respective accuracies.

In order to analyze the influence of the parameters α and λ (recall that we set $\lambda_i^{(r)} = \lambda_i^{(l)} = \lambda$ for all i), we consider that the algorithm converges when the relative difference of the objective function in two consecutive iterations is less than $\varepsilon = 10^{-10}$. By adopting this criterion, **OAC³** usually converges after nine iterations (on average). The algorithm has shown to be robust with respect to λ. As far as α is concerned, for most of the datasets — News1, News3, News4, News6, Cora1, Cora3, Cora4, and DBLP — the classification accuracies achieved from **OAC³** are better than those found by the classifier ensemble — no matter the value chosen for α. Figure 5 illustrates a typical accuracy surface for different values of λ and α. It is worth mentioning that

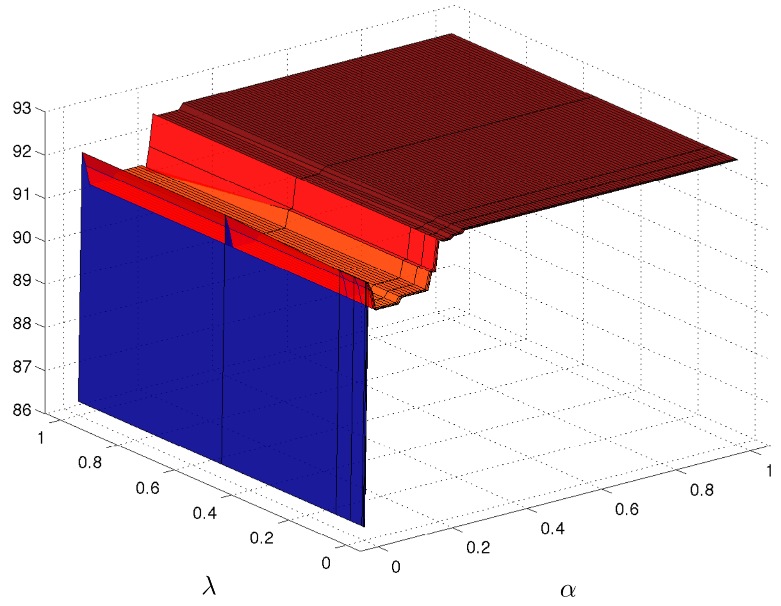

Fig. 5. Accuracy Surface – News6

the accuracy surface tends to keep steady for $\alpha > 1$ (*i.e.*, the accuracies do not change significantly). In particular, **OAC^3** was run for $\alpha = \{10; 20; ...; 100; 200; ...; 1000; 100000\}$, for which the obtained results are the same as those achieved for $\alpha = 1$ for any value of λ. This same observation holds for all the assessed datasets. The interpretation for such results is that there is a threshold value for α that makes the second term of the objective function in (2) dominating — *i.e.*, the information provided by the cluster ensemble is much more important than the information provided by the classifier ensemble.

We observed that for five datasets (News3, News6, Cora1, Cora3, and DBLP) any value of $\alpha > 0.30$ provides the best classification accuracy. Thus, the algorithm can be robust with respect to the choice of its parameters for some datasets. For the datasets News2 and News5, some α values yield to accuracy deterioration, thereby suggesting that, depending on the value chosen for α, the information provided by the cluster ensemble may hurt — *e.g.*, see Figure 7. Finally, for Cora2, accuracy improvements were not observed, *i.e.*, the accuracies provided by the classifier ensemble were always the best ones. This result suggests that the assumption that classes can be represented by means of clusters does not hold.

As expected, our experiments also show that the number of iterations may influence the performance of the algorithm. In particular, depending on the values chosen for α, a high number of iterations may prejudice the obtained accuracies. Considering the best values obtained for α in our sensitivity analysis, we observed that, for all datasets, the best accuracies were achieved for less than 10 iterations.

By taking into account the results obtained in our sensitivity analyses, and recalling that fine tuning of the **OAC^3** parameters can be done by means of cross-validation, in the next section we compare the performance of **OAC^3** with the recently proposed **BGCM** [Gao et al. 2009; Gao et al. 2011].

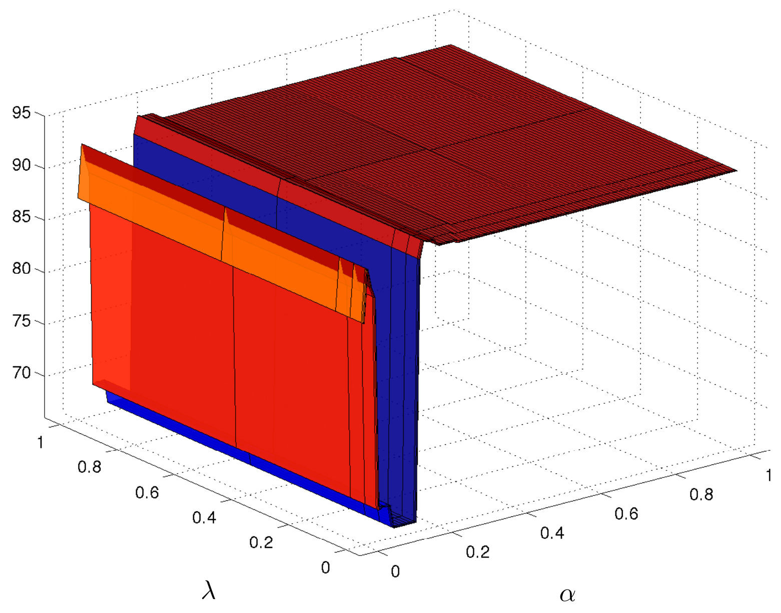

Fig. 6. Accuracy Surface – News2

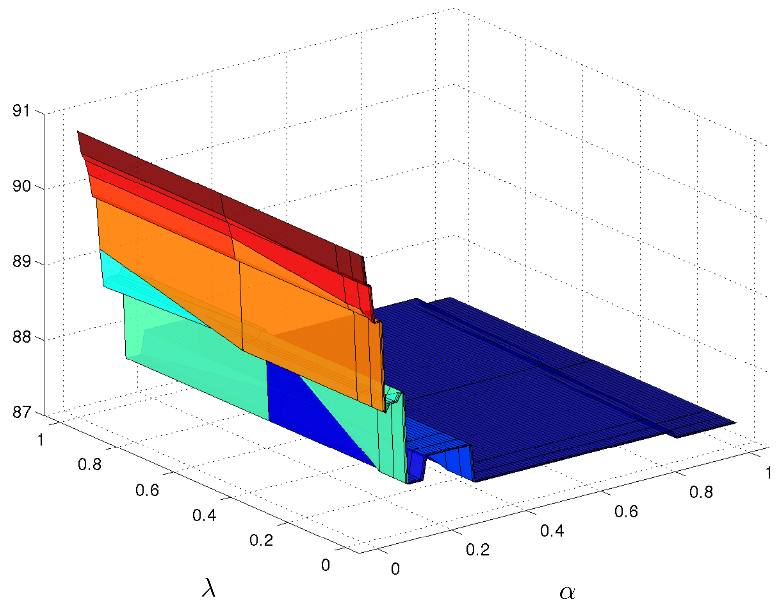

Fig. 7. Accuracy Surface – Cora2

Table III. Comparison of **OAC^3** with Other Algorithms — Classification Accuracies (Best Results in Boldface).

Method	News1	News2	News3	News4	News5	News6	Cora1	Cora2	Cora3	Cora4	DBLP
M_1	79.67	88.55	85.57	88.26	87.65	88.80	77.45	88.58	86.71	88.41	93.37
M_2	77.21	86.11	81.34	86.76	83.58	85.63	77.97	85.94	85.08	88.79	87.66
M_3	80.56	87.96	86.58	89.83	87.16	90.20	77.79	88.33	86.46	88.13	93.82
M_4	77.70	85.71	81.49	84.67	85.43	85.78	74.76	85.94	78.10	90.16	79.49
MCLA	75.92	81.73	82.53	86.86	82.95	85.46	87.03	83.88	88.92	87.16	89.53
HBGF	81.99	92.44	88.11	91.52	89.91	91.25	78.34	91.11	84.81	89.43	93.57
BGCM	81.28	91.01	86.08	91.25	88.64	90.88	86.87	**91.55**	89.65	90.90	94.17
OAC^3	**85.01**	**93.64**	**89.64**	**93.80**	**91.22**	**92.59**	**88.54**	90.79	**90.60**	**91.49**	**94.38**

6.3. Comparison with BGCM

As discussed in Section 2, **BGCM** is the algorithm most closely related to **OAC^3**. We evaluate **OAC^3** on the same classification datasets employed to assess **BGCM** [Gao et al. 2009; Gao et al. 2011]. These datasets are those addressed in Section 6.2. In [Gao et al. 2009], 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 are used as inputs to **OAC^3**. In doing so, comparisons between **OAC^3** and **BGCM** are performed using exactly the same base models, which were trained in the same datasets[4]. In other words, both **OAC^3** and **BGCM** receive the same inputs with respect to 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 [Gao et al. 2009] are not reproduced here, and the interested reader is referred to that paper for further details. However, the results for their four base models ($\mathbf{M}_1$,...,$\mathbf{M}_4$), for **BGCM**, and for two well-known cluster ensemble approaches — **MCLA** [Strehl and Ghosh 2002a] and **HBGF** [Fern and Brodley 2004] — are reproduced here for comparison purposes. Being cluster ensemble approaches, **MCLA** and **HBGF** 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 which favors the best possible class predictions. In order to run **OAC^3**, the supervised models ($\mathbf{M}_1$ and $\mathbf{M}_2$) are fused to obtain class probability estimates for every instance in the target set. Also, the similarity matrix used by **OAC^3** is calculated by fusing the unsupervised models ($\mathbf{M}_3$ and $\mathbf{M}_4$).

The parameters of **OAC^3** have been chosen from the sensitivity analysis performed in Section 6.2. However, for the experiments reported in this section we do not set particular values for each of the (eleven) studied datasets. Instead, we have chosen a set of parameter values that result in good accuracies across related datasets. In particular the following pairs of (α, λ) are respectively used for the datasets *News*, *Cora*, and *DBLP*: $(4 \times 10^{-2}, 10^{-2})$; $(10^{-4}, 10^{-2})$; $(10^{-7}, 10^{-3})$. Such choices will hopefully show that one can get good results by using **OAC^3** without being (necessarily) picky about its parameter values — thus these results are also complementary to the ones provided in Section 6.2.

The classification accuracies achieved by the studied methods are summarized in Table III, where one can see that **OAC^3** shows 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 in [Demsar 2006], are also reported. In brief, multiple algorithms are compared on multiple datasets by

[4]For these datasets, comparisons with **S^{3}VM** [Sindhwani and Keerthi 2006] have not been performed because the raw data required for learning is not available.

using the Friedman test, with a corresponding Nemenyi post-hoc test. The Friedman test is a non-parametric statistic test equivalent to the repeated-measures ANOVA. If the null hypothesis, which states that the algorithms under study have similar performances, is rejected, then the Nemenyi post-hoc test is used for pairwise comparisons between algorithms. The adopted statistical procedure indicates that the null hypothesis of equal accuracies — considering the results obtained by the ensembles — can be rejected at 10% significance level. In pairwise comparisons, significant statistical differences are only observed between **OAC^3** and the other ensembles, *i.e.*, there is no evidence that the accuracies of **MCLA**, **HBGF**, and **BGCM** are statistically different from one another.

6.4. Comparison with S^{3}VM

We also compare **OAC^3** to a popular semi-supervised algorithm known as **S^{3}VM** [Sindhwani and Keerthi 2006]. This algorithm is essentially a Transductive Linear Support Vector Machine (**SVM**) which can be viewed as a large scale implementation of the algorithm introduced in [Joachims 1999b]. For dealing with unlabeled data, it appends an additional term in the **SVM** objective function whose role is to drive the classification hyperplane towards low data density regions [Sindhwani and Keerthi 2006]. The default parameter values have been used for **S^{3}VM**.

Six datasets are used in our experiments: *Half-Moon* (see Section 6.1), *Circles* (which is a synthetic dataset that has two-dimensional instances that form two concentric circles — one for each class), and four datasets from the *Library for Support Vector Machines* [5] — *Pima Indians Diabetes*, *Heart*, *German Numer*, and *Wine*. In order to simulate real-world classification problems where there is a very limited amount of labeled instances, small percentages (e.g., 2%) of the instances are randomly selected for training, whereas the remaining instances are used for testing (target set). The amount of instances for training is chosen so that the pooled covariance matrix of the training set is positive definite. This *restriction* comes from the use of an **LDA** classifier in the ensemble, and it imposes a lower bound on the number of training instances (7% for *Heart* and 10% for *German Numer*). We perform 10 trials for every proportion of instances in the training/target sets. The number of features are 2, 2, 8, 13, 24, 24 for Half-moon, Circles, Pima, Heart, German Numer and Wine respectively.

Considering **OAC^3**, the components of the classifier ensemble are chosen as previously described in Section 6.1. Cluster ensembles are generated by means of multiple runs of k-means (10 data partitions for the two-dimensional datasets and 50 data partitions for *Pima*, *Heart*, *German Numer*, and *Wine*).

The parameters of **OAC^3** (α and λ) are optimized for better performance in each dataset using 5-fold cross-validation. The optimal values of (α, λ) for *Half-moon*, *Circles*, *Pima*, *Heart*, *German Numer*, and *Wine* are (0.05,0.1), (0.01,0.1), (0.002,0.1), (0.01,0.2), (0.01,0.1) and (0.01,0.1) respectively. For comparing against squared loss, we also present the results from **OAC^{3}SQ** – a formulation of **OAC^3** with squared loss used as the objective function and **GT** – the graph transduction based formulation proposed in [Wang et al. 2008]. Note that, as discussed in Section 2, there are important differences between transductive settings and our approach. Differently from **OAC^3**, **GT** needs both labeled and unlabeled data at classification time. In particular, the labels used by **GT** are the true labels and not the ones predicted by the classifiers. Thus, **GT** and related methods would be handicapped in a non-transductive setting. From this standpoint, results from **GT** have been included as a baseline for comparison purposes only.

[5]`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Table IV. Comparison of **OAC^3** with **BGCM**, **S^{3}VM** and **GT** — Average Accuracies ±(Standard Deviations).

Dataset	$\|\mathcal{X}\|$	Ensemble	Best	S^{3}VM	BGCM	OAC^3	OAC^{3}SQ	GT
Half-moon(2%)	784	92.53(±1.83)	93.02(±0.82)	99.61(±0.09)	92.16(±1.47)	**99.64**(±0.08)	97.12(±0.12)	97.01(±0.24)
Circles(2%)	1568	60.03(±8.44)	95.74(±5.15)	54.35(±4.47)	78.67(±0.54)	**99.61**(±0.83)	96.12(±1.02)	96.11(±0.28)
Pima(2%)	745	68.16(±5.05)	69.93(±3.68)	61.67(±3.01)	69.21(±4.83)	**70.31**(±4.44)	67.54(±0.31)	68.21(±0.18)
Heart(7%)	251	77.77(±2.55)	79.22(±2.20)	77.07(±4.77)	82.78(±4.82)	**82.85**(±5.25)	81.10(±0.47)	80.22(±0.31)
G. Numer(10%)	900	70.96(±1.00)	70.19(±1.52)	73.00(±1.50)	73.70(±1.06)	**74.44**(±3.44)	73.22(±0.58)	72.93(±0.44)
Wine(10%)	900	79.87(±5.68)	80.37(±5.47)	80.73(±4.49)	75.37(±13.66)	**83.62**(±6.27)	81.89(±0.68)	82.11(±0.47)

It is easy to show that solving for $\mathbf{OAC^3SQ}$ one does not need to maintain left and right copies and the update equations are available in closed form solution for each $\mathbf{y}_i$. This implies there is only one parameter α left to be cross-validated from the training data. A 5-fold cross-validation yields values 0.05, 0.025, 0.005, 0.04, 0.03, and 0.01 for *Half-moon*, *Circles*, *Pima*, *Heart*, *German Numer*, and *Wine* respectively. Table IV shows that the accuracies obtained by $\mathbf{OAC^3}$ are good and consistently better than those achieved by both the classifier ensemble and its *best* individual component. In addition, $\mathbf{OAC^3}$ shows better accuracies than $\mathbf{OAC^3SQ}$, $\mathbf{S^3VM}$ and **BGCM** — from the adopted statistical procedure [Demsar 2006], $\mathbf{OAC^3}$ exhibits significantly better accuracies at a significance level of 10% compared to $\mathbf{S^3VM}$ and **BGCM** only, i.e., there is no significant statistical difference between $\mathbf{OAC^3}$ and $\mathbf{OAC^3SQ}$. To make the computations faster, similarity values below 0.1 are also set to zero and the results do not change from the figures reported in the above table, implying that $\mathbf{OAC^3}$ can be made faster without affecting the accuracy significantly.

6.5. Transfer Learning

Transfer learning emphasizes the transfer of knowledge across domains, tasks, and distributions that are similar but not the same [Silver and Bennett 2008]. We focus on learning scenarios where training and test distributions are different, as they represent (potentially) related but not identical tasks. It is assumed that the training and test domains involve the same class labels. The real-world datasets employed in our experiments are:

a) Text Documents — [Pan and Yang 2010]: From the well-known text collections *20 newsgroup* and *Reuters-21758*, nine cross-domain learning tasks are generated. The two-level hierarchy in both of these datasets is exploited to frame a learning task involving a top category classification problem with training and test data drawn from different sub categories — *e.g.*, to distinguish documents from two top newsgroup categories (rec and talk), the training set (or the source domain) is built from “rec.autos”, “rec.motorcycles”, “talk.politics”, and “talk.politics.misc”, and the test set (or the target domain) is formed from the sub-categories “rec.sport.baseball”, “rec.sport.hockey”, “talk.politics.mideast”, and “talk.religions.misc”. The *Email spam* data set, released by ECML/PKDD 2006 discovery challenge, contains a training set of publicly available messages and three sets of email messages from individual users as test sets. The 4000 labeled examples in the training set and the 2500 test examples for each of the three different users differ in the word distribution. A spam filter learned from public sources are used to test transfer capability on each of the users.

b) Botswana — [Rajan et al. 2006]: This is an application of transfer learning to the pixel-level classification of remotely sensed images, which provides a real-life scenario where such learning will be useful — in contrast to the contrived setting of text classification, which is chosen as it has been used previously in [Dai et al. 2007a]. It is relatively easy to acquire an image, but expensive to label each pixel manually, where images typically have about a million pixels and represent inaccessible terrain. Thus typically only part of an image gets labeled. Moreover, when the satellite again flies over the same area, the new image can be quite different due to change of season, thus a classifier induced on the previous image becomes significantly degraded for the new task. These hyperespectral data sets used are from a 1476×256 pixel study area located in the Okavango Delta, Botswana. It has nine different land-cover types consisting of seasonal swamps, occasional swamps, and drier woodlands located in the distal portion of the delta. Data from this region for different months (May, June and July) were obtained by the Hyperion sensor of the NASA EO-1 satellite for the calibration/validation portion of the mission in 2001. Data collected for each month was further segregated into two different areas. While the May scene (Fig. 8) is characterized by the onset of the annual flooding cycle and some newly burned areas, the progression of the flood and the corresponding vegetation responses are seen in the June (Fig. 9) and July (Fig. 10) scenes. The acquired raw data was further processed to produce 145 features. From each area of Botswana, different transfer learning tasks are generated: the classifiers are trained on either May, June or $\{\text{May} \cup \text{June}\}$ data and tested on either June or July data.

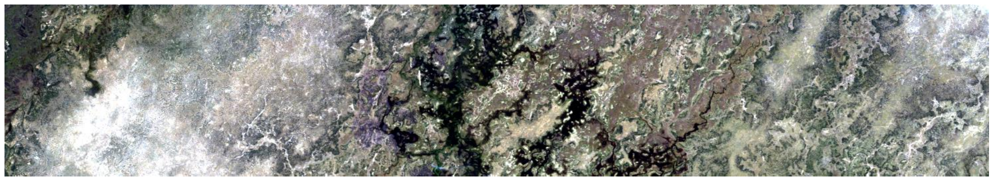
Fig. 8. *Botswana* May 2001.

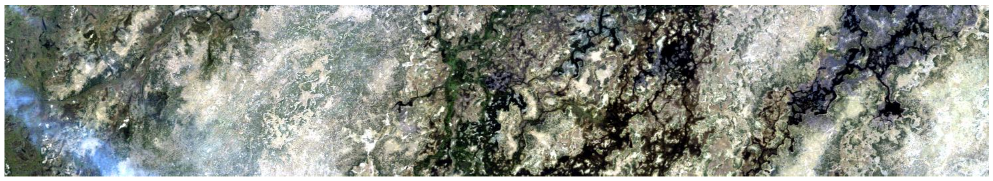
Fig. 9. *Botswana* June 2001.

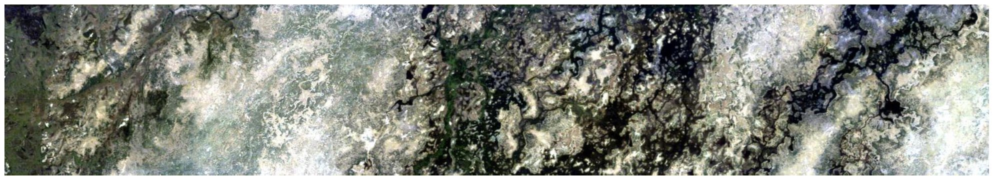
Fig. 10. *Botswana* July 2001.

For text data, we use logistic regression (**LR**), **SVM**, and Winnow (**WIN**) [Gao et al. 2008] as baseline classifiers. The CLUTO package (`http://www.cs.umn.edu/~karypis/cluto`) is used for clustering the target data into two clusters. We also compare **OAC^3** with two transfer learning algorithms from the literature — Transductive Support Vector Machines (**TSVM**) [Joachims 1999a] and the Locally Weighted Ensemble (**LWE**) [Gao et al. 2008]. We use Bayesian Logistic Regression `http://www.bayesianregression.org/` for running the logistic regression classifier, LIBSVM (`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`) for **SVM**, SNoW Learning Architecture `http://cogcomp.cs.illinois.edu/page/software_view/1` for **Winnow**, and SVM$^{\text{light}}$ `http://svmlight.joachims.org/` for transductive **SVM**. The posterior class probabilities from **SVM** are also obtained using the LIBSVM package with linear kernel. For SNoW, "-S 3 -r 5" is used and the remaining parameters of all the packages are set to their default values. The values of (α, λ), obtained by 10-fold cross-validation in source domain, are set as $(0.008, 0.1)$ and $(0.11, 0.1)$ for the transfer learning tasks corresponding to 20 *Newsgroup* and *Spam* datasets, respectively. For *Reuters*-21578, the best values of the parameters (α, λ) are found as $(0.009, 0.1)$, $(0.0001, 0.1)$, and $(0.08, 0.1)$

for *O vs Pe*, *O vs Pl*, and *Pe vs Pl*, respectively (see Table VI). For the hyperspectral data, we use two baseline classifiers: the well-known Naïve Bayes Wrapper (**NBW**) and the Maximum Likelihood (**ML**) classifier, which performs well when used with a best bases feature extractor [Kumar et al. 2001]. The target set instances are clustered by k-means, varying k from 50 to 70. **PCA** is also used for reducing the number of features employed by **ML**. In particular, for the hyperspectral data, cross-validation in the source domain does not result in very good performance. Therefore, we take 5% labeled examples from each of the nine classes of the target data and tune the values of α and λ based on the performance on these examples. The classifiers **NBW** or **ML**, however, are not retrained with these examples from the target domain and the accuracies reported in Table VI are on the unlabeled examples only from the target domain.

Table V. Classification of *20 Newsgroup*, *Reuters-21758* and *Spam* Data.

Dataset	Mode	WIN	LR	SVM	Ensemble	TSVM	LWE	OAC^3
20 Newsgroup	C vs S	66.61	67.17	67.02	69.58	76.97	77.07	**91.25**
	R vs T	60.43	68.79	63.87	65.98	89.95	87.46	**90.11**
	R vs S	80.11	76.51	71.40	77.39	89.96	87.81	**92.90**
	S vs T	73.93	72.16	71.51	75.11	85.59	81.99	**91.83**
	C vs R	89.00	77.36	81.50	85.18	89.64	91.09	**93.75**
	C vs T	93.41	91.76	93.89	93.48	88.26	**98.90**	98.70
Reuters-21758	O vs Pe	70.57	66.19	69.25	73.30	76.94	76.77	**80.97**
	O vs Pl	65.10	67.87	69.88	69.21	**70.08**	67.59	68.91
	Pe vs Pl	56.75	56.48	56.20	57.59	59.72	59.90	**67.46**
Spam	spam 1	79.15	56.92	66.28	68.64	76.92	65.60	**80.29**
	spam 2	81.15	59.76	73.15	75.07	84.92	73.36	**87.05**
	spam 3	88.28	64.43	78.71	81.87	90.79	**93.79**	91.27

The results for text data are reported in Table V. The different learning tasks corresponding to different pairs of categories are listed as "Mode". **OAC^3** improves the performance of the classifier ensemble (formed by combining **WIN**, **LR** and **SVM** via output averaging) for all learning tasks, except for *O vs Pl*, where apparently the training and test distributions are similar. Also, the **OAC^3** accuracies are better than those achieved by both **TSVM** and **LWE** in most of the datasets. Except for **WIN**, the performances of the base classifiers and clustereres (and hence of **OAC^3**) are quite invariant, thereby resulting in very low standard deviations. The **OAC^3** accuracies are significantly better than those obtained by both **TSVM** and **LWE** (at 10% significance level).

Table VI. Classification of Hyperspectral Data — *Botswana*.

Dataset	Original to Target	NBW	NBW+OAC^3	ML	ML+OAC^3	α	λ	PCs
Area 1	may to june	70.68	**72.61** ($\pm$0.42)	74.47	**81.93** ($\pm$0.52)	0.0010	0.1	9
	may to july	61.85	**63.11** ($\pm$0.29)	58.58	**64.32** ($\pm$0.53)	0.0001	0.2	12
	june to july	70.55	**72.47** ($\pm$0.17)	79.71	**80.06** ($\pm$0.26)	0.0012	0.1	127
	may+june to july	75.53	**80.53** ($\pm$0.31)	85.78	**85.91** ($\pm$0.23)	0.0008	0.1	123
Area 2	may to june	66.10	**71.02** ($\pm$0.28)	70.22	**81.48** ($\pm$0.43)	0.0070	0.1	9
	may to july	61.55	**63.74** ($\pm$0.14)	52.78	**64.15** ($\pm$0.22)	0.0001	0.2	12
	june to july	54.89	**57.65** ($\pm$0.53)	75.62	**77.04** ($\pm$0.37)	0.0060	0.1	80
	may+june to july	63.79	**64.58** ($\pm$0.16)	77.33	**79.59** ($\pm$0.23)	0.0040	0.1	122

Table VI reports the results for the hyperspectral data. The parameter values (α, λ) for best performance of **OAC^3** are also presented alongside. Note that **OAC^3** provides consistent accuracy improvements for both **NBW** and **ML** [6]. In pairwise comparisons, the accuracies provided by **OAC^3** are significantly better than those obtained by both **NBW** and **ML** (at 10% significance level). The column "PCs" indicates the number of principal components used to project the data.

7. CONCLUDING REMARKS

We presented a general framework for combining classifiers and clusterers to address semi-supervised and transfer learning problems in non-transductive settings. The optimization algorithm yields closed form updates, facilitates parallelization of the same and, therefore, is convenient for handling large scale data – with a linear rate of convergence. The proofs for the convergence are novel and generalize across a wide variety of Bregman divergences, allowing one to use a suitable divergence measure based on the application domain. The proposed framework has been empirically shown to outperform a variety of algorithms [Gao et al. 2011; Sindhwani and Keerthi 2006; Gao et al. 2008] in both semi-supervised and transfer learning problems. More significantly, it can operate even in settings where there are no labeled data in the target domain and the labeled data from the source domain is also no longer available. Such settings are very challenging for existing graph-based semi-supervised learning approaches.

There are a few aspects that can be further explored. For example, the impact of the number of classifiers and clusterers in **OAC^3** deserves further investigation. Also, the relative relevance of each component of the ensemble can be incorporated into **OAC^3** via weights on the components, thus possibly leading to incremental accuracy improvements. The weights of the components in the classifier ensemble can either be estimated from domain knowledge or can iteratively be refined using importance sampling in functional space, as suggested in [Xie et al. 2012]. In another possible extension, if labeled data is scarce in the source domain but unlabeled data is easily accessible, one could use the labels of the unlabeled data inferred by **OAC^3** to retrain the classifiers and predict on the unlabeled data again in the source domain in an iterative process.

APPENDIX

A. PROOFS FOR CONVERGENCE OF OAC^3

LEMMA A.1. *The objective function J used in Eq. (4) is separately and jointly strictly convex over $\mathcal{S}^n \times \mathcal{S}^n$. Also, J is jointly lower-semi-continuous w.r.t $\mathbf{y}^{(l)}$ and $\mathbf{y}^{(r)}$.*

PROOF.

(a) From the property (a) in Section 4, one can see that J is strictly convex w.r.t $\mathbf{y}^{(l)}$ and $\mathbf{y}^{(r)}$ separately. From the same property the first term $f_1(\mathbf{y}^{(r)}) = \sum_{i=1}^{n} d_\phi(\boldsymbol{\pi}_i, \mathbf{y}_i^{(r)})$ in J is strictly convex w.r.t. $\mathbf{y}^{(r)}$. The 2nd and 3rd terms in the objective function can collectively be represented by $f_2(\mathbf{y}^{(l)}, \mathbf{y}^{(r)})$. This function is jointly convex by property (b) but is not necessarily jointly strictly convex. Suppose $(\mathbf{y}^{1,(l)}, \mathbf{y}^{1,(r)}), (\mathbf{y}^{2,(l)}, \mathbf{y}^{2,(r)}) \in \mathcal{S}^n \times \mathcal{S}^n$ and $0 < w < 1$. Then, we have:

$$\begin{aligned} f_1(w\mathbf{y}^{1,(r)} + (1-w)\mathbf{y}^{2,(r)}) \;&<\; wf_1(\mathbf{y}^{1,(r)}) + (1-w)f_1(\mathbf{y}^{2,(r)}) \\ f_2(w(\mathbf{y}^{1,(l)}, \mathbf{y}^{1,(r)}) + (1-w)(\mathbf{y}^{2,(l)}, \mathbf{y}^{2,(r)})) \;&\leq\; wf_2(\mathbf{y}^{1,(l)}, \mathbf{y}^{1,(r)}) + (1-w)f_2(\mathbf{y}^{2,(l)}, \mathbf{y}^{2,(r)}). \end{aligned}$$

[6]Standard deviations of the accuracies from **NBW** and **ML** are close to 0 and hence not shown.

Now, it follows that:

$$\begin{aligned} & J(w(\mathbf{y}^{1,(l)}, \mathbf{y}^{1,(r)}) + (1-w)(\mathbf{y}^{2,(l)}, \mathbf{y}^{2,(r)})) \\ = \; & f_1(w\mathbf{y}^{1,(r)} + (1-w)\mathbf{y}^{2,(r)}) + f_2(w(\mathbf{y}^{1,(l)}, \mathbf{y}^{1,(r)}) + (1-w)(\mathbf{y}^{2,(l)}, \mathbf{y}^{2,(r)})) \\ < \; & wf_1(\mathbf{y}^{1,(r)}) + (1-w)f_1(\mathbf{y}^{2,(r)}) + wf_2(\mathbf{y}^{1,(l)}, \mathbf{y}^{1,(r)}) + (1-w)f_2(\mathbf{y}^{2,(l)}, \mathbf{y}^{2,(r)}) \\ = \; & wJ(\mathbf{y}^{1,(l)}, \mathbf{y}^{1,(r)}) + (1-w)J(\mathbf{y}^{2,(l)}, \mathbf{y}^{2,(r)}), \end{aligned} \tag{21}$$

which implies that J is jointly strictly convex.

(b) To prove that $J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)})$ is lower-semi-continuous in $\mathbf{y}^{(l)}$ and $\mathbf{y}^{(r)}$ jointly, we observe that

$$\begin{aligned} & \liminf_{(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}) \to (\mathbf{y}^{0,(l)}, \mathbf{y}^{0,(r)})} J(\mathbf{y}^{0,(l)}, \mathbf{y}^{0,(r)}) \\ = \; & \left[\sum_{i=1}^{n} \liminf_{\mathbf{y}_i^{(r)} \to \mathbf{y}_i^{0,(r)}} d_\phi(\boldsymbol{\pi}_i, \mathbf{y}_i^{(r)}) + \alpha \sum_{i,j=1}^{n} s_{ij} \liminf_{(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}) \to (\mathbf{y}_i^{0,(l)}, \mathbf{y}_j^{0,(r)})} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}) \right. \\ + \; & \left. \lambda \sum_{i=1}^{n} \liminf_{(\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)}) \to (\mathbf{y}_i^{0,(l)}, \mathbf{y}_i^{0,(r)})} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)}) \right] \\ \geq \; & \left[\sum_{i=1}^{n} d_\phi(\boldsymbol{\pi}_i, \mathbf{y}_i^{0,(r)}) + \alpha \sum_{i,j=1}^{n} s_{ij} d_\phi(\mathbf{y}_i^{0,(l)}, \mathbf{y}_j^{0,(r)}) + \lambda \sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{0,(l)}, \mathbf{y}_i^{0,(r)}) \right] \\ = \; & J(\mathbf{y}^{0,(l)}, \mathbf{y}^{0,(r)}). \end{aligned} \tag{22}$$

The inequality in the 3rd step follows from the lower semi continuity of $d_\phi(.,.)$ in Section 4 (Property (d)).

□

The following theorem helps prove that the objective function J can be seen as part of a Bregman divergence.

THEOREM A.2 ([BANERJEE ET AL. 2005]). *A divergence $d : \mathcal{S} \times ri(\mathcal{S}) \to [0, \infty)$ is a Bregman divergence if and only if $\exists \mathbf{a} \in ri(\mathcal{S})$ such that the function $\phi_\mathbf{a}(\mathbf{p}) = d(\mathbf{p}, \mathbf{a})$ satisfies the following conditions:*

(a) $\phi_\mathbf{a}$ is strictly convex on $\mathcal{S}$.
(b) $\phi_\mathbf{a}$ is differentiable on $ri(\mathcal{S})$.
(c) $d(\mathbf{p}, \mathbf{q}) = d_{\phi_\mathbf{a}}(\mathbf{p}, \mathbf{q}), \forall \mathbf{p} \in \mathcal{S}, \mathbf{q} \in ri(\mathcal{S})$ where $d_{\phi_\mathbf{a}}$ is the Bregman divergence associated with $\phi_\mathbf{a}$.

We now introduce a function $\tilde{J} : \mathcal{S}^n \times \mathcal{S}^n \to [0, \infty)$ that is defined as follows:

$$\tilde{J}(\mathbf{y}^{(r)\prime}, \mathbf{y}^{(r)}) = \left[\sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{(r)\prime}, \mathbf{y}_i^{(r)}) + \alpha \sum_{i,j=1}^{n} s_{ij} d_\phi(\mathbf{y}_i^{(r)\prime}, \mathbf{y}_j^{(r)}) + \lambda \sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{(r)\prime}, \mathbf{y}_i^{(r)}) \right]. \tag{23}$$

Note that $\tilde{J}$ is different from J defined in Eq. (4). The left arguments in the divergences of the first term of J are $\boldsymbol{\pi}_i$'s which are assumed to be fixed.

LEMMA A.3. *$\tilde{J}$ satisfies properties (a) and (b) in Section 4.*

PROOF. The proof is direct from the definition of $\tilde{J}$. □

Further assume:

$$\begin{aligned}\mathbf{p} &= \big((\boldsymbol{\pi}_i)_{i=1}^n, (\mathbf{y}_i^{(l)})_{i=1}^n, ((\mathbf{y}_i^{(l)})_{j=1}^{n-1})_{i=1}^n\big),\\ \mathbf{q} &= \big((\mathbf{y}_i^{(r)})_{i=1}^n, (\mathbf{y}_i^{(r)})_{i=1}^n, ((\mathbf{y}_j^{(r)})_{j=1,j\neq i}^{n-1})_{i=1}^n\big),\\ \mathbf{q}' &= \big((\mathbf{y}_i^{(r)\prime})_{i=1}^n, (\mathbf{y}_i^{(r)\prime})_{i=1}^n, ((\mathbf{y}_j^{(r)\prime})_{j=1,j\neq i}^{n-1})_{i=1}^n\big),\end{aligned}$$

with $\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)}, \mathbf{y}_i^{(r)\prime} \in \mathcal{S}\ \forall i$. The vectors $\mathbf{p}$, $\mathbf{q}$ and $\mathbf{q}'$ are each of dimension $kn(n+1)$ and formed by concatenating vectors from the set $\mathcal{S}$. $(\mathbf{y})_{i=1}^n$ implies that a new vector is created by repeating $\mathbf{y}$ for n times. For ease of understanding, we also define $\mathcal{A} = \{\mathbf{p} : \mathbf{y}^{(l)} \in \mathcal{S}^n\}$, $\mathcal{B} = \{\mathbf{q} : \mathbf{y}^{(r)} \in \mathcal{S}^n\}$. We will assume that whenever a point $\mathbf{y}^{(l)} \in \mathcal{S}^n$ is mapped to a point $\mathbf{p} \in \mathcal{A}$, $\mathbf{p} = \mathbb{A}(\mathbf{y}^{(l)})$. Similarly, $\mathbf{q} = \mathbb{B}(\mathbf{y}^{(r)})$ whenever $\mathbf{y}^{(r)} \in \mathcal{S}^n$ is mapped to $\mathbf{q} \in \mathcal{B}$. Indeed, both $\mathbb{A}$ and $\mathbb{B}$ are bijective mappings.

Example A.4. To explain the mappings $\mathbb{A}$ and $\mathbb{B}$ more clearly, we consider the following example. Let $n = 3$ and $\mathbf{y}^{(l)}, \mathbf{y}^{(r)}, \mathbf{y}^{(r)\prime} \in \mathcal{S}^3$. Here, $\mathbf{y}^{(l)} = \big(\mathbf{y}_1^{(l)}, \mathbf{y}_2^{(l)}, \mathbf{y}_3^{(l)}\big)$ – a concatenation of three vectors $\mathbf{y}_1^{(l)}$, $\mathbf{y}_2^{(l)}$ and $\mathbf{y}_3^{(l)}$ (corrpesponding to three instances) each of which belongs to $\mathcal{S} \subseteq \mathbb{R}^k$. Similarly, $\mathbf{y}^{(r)} = \big(\mathbf{y}_1^{(r)}, \mathbf{y}_2^{(r)}, \mathbf{y}_3^{(r)}\big)$ and $\mathbf{y}^{(r)\prime} = \big(\mathbf{y}_1^{(r)\prime}, \mathbf{y}_2^{(r)\prime}, \mathbf{y}_3^{(r)\prime}\big)$. The vector $\mathbf{p}$, formed by the transformation $\mathbb{A}$ on $\mathbf{y}^{(l)}$, takes the following form:

$$\mathbf{p} = \big(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \boldsymbol{\pi}_3, \mathbf{y}_1^{(l)}, \mathbf{y}_2^{(l)}, \mathbf{y}_3^{(l)}, \mathbf{y}_1^{(l)}, \mathbf{y}_1^{(l)}, \mathbf{y}_2^{(l)}, \mathbf{y}_2^{(l)}, \mathbf{y}_3^{(l)}, \mathbf{y}_3^{(l)}\big)$$

Note that this vector has 12 elements each of dimension k and hence the dimension of the whole vector is of the form $kn(n+1)$. Similarly,

$$\mathbf{q} = \mathbb{B}(\mathbf{y}^{(r)}) = \big(\mathbf{y}_1^{(r)}, \mathbf{y}_2^{(r)}, \mathbf{y}_3^{(r)}, \mathbf{y}_1^{(r)}, \mathbf{y}_2^{(r)}, \mathbf{y}_3^{(r)}, \mathbf{y}_2^{(r)}, \mathbf{y}_3^{(r)}, \mathbf{y}_1^{(r)}, \mathbf{y}_3^{(r)}, \mathbf{y}_1^{(r)}, \mathbf{y}_2^{(r)}\big),$$

and,

$$\mathbf{q}' = \mathbb{B}(\mathbf{y}^{(r)\prime}) = \big(\mathbf{y}_1^{(r)\prime}, \mathbf{y}_2^{(r)\prime}, \mathbf{y}_3^{(r)\prime}, \mathbf{y}_1^{(r)\prime}, \mathbf{y}_2^{(r)\prime}, \mathbf{y}_3^{(r)\prime}, \mathbf{y}_2^{(r)\prime}, \mathbf{y}_3^{(r)\prime}, \mathbf{y}_1^{(r)\prime}, \mathbf{y}_3^{(r)\prime}, \mathbf{y}_1^{(r)\prime}, \mathbf{y}_2^{(r)\prime}\big).$$

□

Now, in light of Theorem A.2, the following corollary is introduced.

COROLLARY A.5. *If a mapping* $d : (\mathcal{A} \cup \mathcal{B}) \times \mathcal{B} \to [0, \infty)$ *is defined as:*

$$d(\mathbf{r}, \mathbf{q}) = \begin{cases} d(\mathbf{p}, \mathbf{q}) = J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}) & \textit{if } \mathbf{r} = \mathbf{p} \in \mathcal{A} \\ d(\mathbf{q}', \mathbf{q}) = \tilde{J}(\mathbf{y}^{(r)\prime}, \mathbf{y}^{(r)}) & \textit{if } \mathbf{r} = \mathbf{q}' \in \mathcal{B} \end{cases} \tag{24}$$

then d is a Bregman divergence.

PROOF. We show that conditions (a), (b) and (c) of Theorem A.2 are satisfied for d.

(a) Since d_ϕ is a Bregman divergence, $\exists \mathbf{a} \in \mathrm{ri}(\mathcal{S})$ such that conditions (a), (b) and (c) are satisfied in corollary A.2 pertaining to this divergence. Note that $\mathbf{p} \in \mathcal{A}$ and $\mathbf{q}', \mathbf{q} \in \mathcal{B}$. Assume $\mathbf{a}' = \mathbb{B}((\mathbf{a})_{i=1}^n) \in \mathcal{B} \subset (\mathcal{A} \cup \mathcal{B})$. We now define

$$\psi_{\mathbf{a}'}(\mathbf{r}) = \begin{cases} \psi_{\mathbf{a}'}(\mathbf{p}) = d(\mathbf{p}, \mathbf{a}') = J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}) & \text{if } \mathbf{r} = \mathbf{p} \in \mathcal{A} \\ \psi_{\mathbf{a}'}(\mathbf{q}') = d(\mathbf{q}', \mathbf{a}') = \tilde{J}(\mathbf{y}^{(r)\prime}, \mathbf{y}^{(r)}) & \text{if } \mathbf{r} = \mathbf{q}' \in \mathcal{B} \end{cases} \tag{25}$$

Since each of $d_\phi(., \mathbf{a})$ is strictly convex over $\mathcal{S}^n$ in Eq. (4) and Eq. (23), $\psi_{\mathbf{a}'}$ is also strictly convex on $\mathcal{A} \cup \mathcal{B}$. Note the emphasis on $\mathcal{B} \subset (\mathcal{A} \cup \mathcal{B})$ in the definition of $\mathbf{a}'$ which just ensures that all conditions in Theorem A.2 are satisfied.

(b) Again, this is a direct consequence from Eq. (4) and Eq. (23). Since, by the strict convexity of $\phi(.)$, each of $d_\phi(., \mathbf{a})$ is differentiable over $\mathrm{ri}(\mathcal{S}^n)$, $\psi_{\mathbf{a}'}$ is also differentiable over $\mathrm{ri}(\mathcal{A} \cup \mathcal{B})$. Note that we have a bijective mapping of elements from $\mathcal{S}^n$ to $\mathcal{A} \cup \mathcal{B}$, and hence $\mathrm{ri}(\mathcal{S}^n)$ gets mapped to $\mathrm{ri}(\mathcal{A} \cup \mathcal{B})$.

(c) We have $\forall\ (\mathbf{p},\mathbf{q}) \in \mathcal{A}\times\mathcal{B}$,

$$\begin{aligned}
d_{\psi_{\mathbf{a}'}}(\mathbf{p},\mathbf{q}) &= \left[\psi_{\mathbf{a}'}(\mathbf{p}) - \psi_{\mathbf{a}'}(\mathbf{q}) - \langle \boldsymbol{\nabla}_{\psi_{\mathbf{a}'}}(\mathbf{q}), (\mathbf{p}-\mathbf{q})\rangle\right] \\
&= \sum_{i=1}^{n}\left[d_\phi(\boldsymbol{\pi}_i,\mathbf{a}) - d_\phi(\mathbf{y}_i^{(r)},\mathbf{a})\right] + \alpha\sum_{i,j=1}^{n} s_{ij}\left[d_\phi(\mathbf{y}_i^{(l)},\mathbf{a}) - d_\phi(\mathbf{y}_j^{(r)},\mathbf{a})\right] \\
&+ \lambda\sum_{i=1}^{n}\left[d_\phi(\mathbf{y}_i^{(l)},\mathbf{a}) - d_\phi(\mathbf{y}_i^{(r)},\mathbf{a})\right] - \langle \boldsymbol{\nabla}_{\psi_{\mathbf{a}'}}(\mathbf{q}), (\mathbf{p}-\mathbf{q})\rangle \\
&= \sum_{i=1}^{n}\left[d_\phi(\boldsymbol{\pi}_i,\mathbf{y}_i^{(r)}) + \alpha\sum_{i,j=1}^{n} s_{ij} d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_j^{(r)}) + \lambda\sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_i^{(r)})\right] \\
&= d(\mathbf{p},\mathbf{q}).
\end{aligned}$$

The second step follows from the definition of $\psi(.)$ in Eq. (25) and the last step follows from the definition of $d(\mathbf{p},\mathbf{q})$ in Eq. (24). The equality $d_{\psi_{\mathbf{a}'}}(\mathbf{q}',\mathbf{q}) = d(\mathbf{q}',\mathbf{q})$ $\forall(\mathbf{q}',\mathbf{q}) \in \mathcal{B}\times\mathcal{B}$ can similarly be proved. Therefore, combining the two results, we have $d_{\psi_{\mathbf{a}'}}(\mathbf{r},\mathbf{q}) = d(\mathbf{r},\mathbf{q})\ \forall(\mathbf{r},\mathbf{q}) \in (\mathcal{A}\cup\mathcal{B})\times\mathcal{B}$. With a slight abuse of notation, henceforth, we will denote the mapping $\psi_{\mathbf{a}'}$ by ψ with an implicit assumption of the existence of an $\mathbf{a}' \in \mathcal{B}$ as described before.

We will see next that we require some definition of $\psi_{\mathbf{a}'}(\mathbf{q})$ for $\mathbf{q}\in\mathcal{B}$ and this explains the definition of $d(\mathbf{r},\mathbf{q})$ in Eq. (24) for the case when $\mathbf{r} = \mathbf{q}' \in \mathcal{B}$. □

LEMMA A.6. *d_ψ satisfies properties (a) and (b) in Section 4.*

PROOF.

(a) One can see that d_ψ is strictly convex separately w.r.t its arguments from its definition in Eq. (24). Since each of J and $\tilde{J}$ is strictly convex separately w.r.t the arguments and $\mathbb{A}$ and $\mathbb{B}$ are bijective mappings, d_ψ is strictly convex separately w.r.t. r and q.

(b) The joint convexity of d_ψ also follows directly from its definition and the joint convexity of J and $\tilde{J}$.

□

At this point, we reiterate that defining d_ψ as in Eq. (24) helps in proving some interesting properties of J in a very elegant way. We, in fact, treat d_ψ as a surrogate for J, establish two specific properties of d_ψ and then show that these properties, by the definition of d_ψ, translates to the same properties of J. The first of them is the 3-Points Property (3-pp) which is introduced in the following definition.

Definition A.7 (*3-pp*). Let $\mathcal{P}$ and $\mathcal{Q}$ be closed convex sets of finite measures. A function $d : \mathcal{P}\times\mathcal{Q} \to \mathbb{R}\cup\{-\infty,+\infty\}$ is said to satisfy the 3-points property (3-pp) if for a given $q\in\mathcal{Q}$ for which $d(p,q) < \infty\ \forall p\in\mathcal{P}$, $\delta(p,p^*) \le d(p,q) - d(p^*,q)$ where $p^* = \underset{p\in\mathcal{P}}{\operatorname{argmin}}\, d(p,q)$ and $\delta : \mathcal{P}\times\mathcal{P} \to \mathbb{R}_+$ with $\delta(p,p') = 0$ iff $p = p'$.

LEMMA A.8. *J satisfies 3-pp.*

PROOF. The proof is based on the works of [Wang and Schuurmans 2003a]. First, we will show that 3-pp is valid for $d_\psi(.,.)$ over $\mathcal{A}\times\mathcal{B}$. As mentioned earlier, this is where the introduction of d_ψ becomes useful and elegant. Assume that $\mathbf{p} = \mathbb{A}(\mathbf{y}^{(l)}) \in \mathcal{A}$ corresponding to some $\mathbf{y}^{(l)} \in \mathcal{S}^n$, $\mathbf{q} = \mathbb{B}(\mathbf{y}^{(r)}) \in \mathcal{B}$ corresponding to some $\mathbf{y}^{(r)} \in \mathcal{S}^n$ and

$\mathbf{p}^* = \operatorname*{argmin}_{\mathbf{p}\in\mathcal{A}} d_\psi(\mathbf{p},\mathbf{q}) = \operatorname*{argmin}_{\mathbf{y}^{(l)}\in\mathcal{S}^n} J(\mathbf{y}^{(l)},\mathbf{y}^{(r)}) = \mathbb{A}\left(\mathbf{y}^{(l)^*}\right)$ (the fact that the minimizers are just transformations of each other under $\mathbb{A}$ or $\mathbb{A}^{-1}$ follows directly from the separately strict convexity of J and d_ψ). Therefore,

$$\begin{aligned} & d_\psi(\mathbf{p},\mathbf{q}) - d_\psi(\mathbf{p}^*,\mathbf{q}) \\ =\ & \psi(\mathbf{p}) - \psi(\mathbf{p}^*) - \langle \nabla_\psi(\mathbf{q}), \mathbf{p}-\mathbf{p}^*\rangle \\ =\ & \delta_\psi(\mathbf{p},\mathbf{p}^*) + \langle \nabla_\psi(\mathbf{p}^*) - \nabla_\psi(\mathbf{q}), \mathbf{p}-\mathbf{p}^*\rangle \end{aligned}$$

where, $\delta_\psi : \mathcal{A}\times\mathcal{A}\to\mathbb{R}$ is defined as follows:

$$\delta_\psi(\mathbf{p},\mathbf{p}^*) = \psi(\mathbf{p}) - \psi(\mathbf{p}^*) - \langle \nabla_\psi(\mathbf{p}^*), \mathbf{p}-\mathbf{p}^*\rangle. \tag{26}$$

Since $\mathbf{p}^* = \operatorname*{argmin}_{\mathbf{p}\in\mathcal{A}} d_\psi(\mathbf{p},\mathbf{q})$, $\langle \nabla_{\mathbf{p}} d_\psi(\mathbf{p}^*,\mathbf{q}), (\mathbf{p}-\mathbf{p}^*)\rangle \geq 0$, then $\langle \nabla_\psi(\mathbf{p}^*) - \nabla_\psi(\mathbf{q}), (\mathbf{p}-\mathbf{p}^*)\rangle \geq 0$ which implies $d_\psi(\mathbf{p},\mathbf{q}) - d_\psi(\mathbf{p}^*,\mathbf{q}) \geq \delta(\mathbf{p},\mathbf{p}^*)$. Now, by some simple algebra, we can show $\delta_\psi(\mathbf{p},\mathbf{p}^*) = \sum_{i=1}^{n}\left(\lambda + \alpha\sum_{j=1;j\neq i}^{n}\right) d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_i^{(l)^*})$. By assumption, $d_\phi\big(\mathbf{y}_i^{(l)},\mathbf{y}_i^{(l)^*}\big) \geq 0$ and hence $\delta_\psi(\mathbf{p},\mathbf{p}^*) \geq 0$ with 0 achieved iff $\mathbf{y}^{(l)} = \mathbf{y}^{(l)^*}$. If we define $\delta_\psi(\mathbf{p},\mathbf{p}^*) = \delta_J(\mathbb{A}^{-1}(\mathbf{p}),\mathbb{A}^{-1}(\mathbf{p}^*))$ then $\delta_J(\mathbf{y}^{(l)},\mathbf{y}^{(l)^*}) \geq 0$ with 0 achieved iff $\mathbf{y}^{(l)} = \mathbf{y}^{(l)^*}$. Note that

$$\delta_J(\mathbf{y}^{1,(l)},\mathbf{y}^{2,(l)}) = \sum_{i=1}^{n}\left(\lambda + \alpha\sum_{j=1;j\neq i}^{n}\right) d_\phi(\mathbf{y}_i^{1,(l)},\mathbf{y}_i^{2,(l)}). \tag{27}$$

Therefore, following 3-pp of d_ψ over $\mathcal{A}\times\mathcal{B}$, we can conclude that

$$J(\mathbf{y}^{(l)},\mathbf{y}^{(r)}) - J(\mathbf{y}^{(l)^*},\mathbf{y}^{(r)}) \geq \delta_J(\mathbf{y}^{(l)},\mathbf{y}^{(l)^*}), \tag{28}$$

which is the 3-pp for J. □

LEMMA A.9. *δ_J satisfies properties (c) and (f) mentioned in Section 4.*

PROOF.

(a) Since level sets of each of the terms in Eq. (27) are bounded following the property (c) in Section 4, we conclude that the level set $\{\mathbf{y}^{(r)} : \delta_J(\mathbf{y}^{(l)},\mathbf{y}^{(r)}) \leq \ell\}$ for a given $\mathbf{y}^{(l)} \in \mathcal{S}^n$ is also bounded.

(b) We refer to Eq. (27). As, $\mathbf{y}^{2,(l)} \to \mathbf{y}^{1,(l)}$, each of the $d_\phi(.,.)$'s goes to 0 by the property (f) in Section 4. Therefore, $\delta_J \to 0$ as $\mathbf{y}^{2,(l)} \to \mathbf{y}^{1,(l)}$.

□

Next, 4-Points Property (4-pp) is introduced.

Definition A.10 (*4-pp*). Let $\mathcal{P}$ and $\mathcal{Q}$ be closed convex sets of finite measures. A function $d : \mathcal{P}\times\mathcal{Q} \to \mathbb{R}\cup\{-\infty,+\infty\}$ is said to satisfy 4-pp if for a given $p \in \mathcal{P}$, $d(p,q^*) \leq \delta(p,p^*) + d(p,q)$ where $q^* = \operatorname*{argmin}_{q\in\mathcal{Q}} d(p^*,q)$ and $\delta : \mathcal{P}\times\mathcal{P} \to \mathbb{R}_+$ with $\delta(p,p') = 0$ iff $p = p'$.

LEMMA A.11. *J satisfies 4-pp.*

PROOF. Assume $\mathbf{u} = \mathbb{A}(\mathbf{y}^{1,(l)}) \in \mathcal{A}$, $\mathbf{p} = \mathbb{A}(\mathbf{y}^{2,(l)}) \in \mathcal{A}$, $\mathbf{q} = \mathbb{B}(\mathbf{y}^{3,(r)}) \in \mathcal{B}$, and $\mathbf{q}^* = \operatorname*{argmin}_{\mathbf{q}\in\mathcal{B}} d_\psi(\mathbf{p},\mathbf{q}) = \mathbb{B}(\mathbf{y}^{4,(r)^*})$. Here, $\mathbf{y}^{1,(l)},\mathbf{y}^{2,(l)},\mathbf{y}^{3,(r)} \in \mathcal{S}^n$ and $\mathbf{y}^{4,(r)^*} =$

$\underset{\mathbf{y}^{(r)} \in \mathcal{S}^n}{\operatorname{argmin}} J(\mathbf{y}^{2,(l)}, \mathbf{y}^{(r)})$. From the joint convexity of d_ψ (established in Lemma A.6) w.r.t both of its arguments we have:

$$d_\psi(\mathbf{u}, \mathbf{v}) \geq d_\psi(\mathbf{p}, \mathbf{q}^*) + \langle \nabla_{\mathbf{p}} d_\psi(\mathbf{p}, \mathbf{q}^*), \mathbf{u} - \mathbf{p} \rangle + \langle \nabla_{\mathbf{q}} d_\psi(\mathbf{p}, \mathbf{q}^*), \mathbf{v} - \mathbf{q}^* \rangle. \tag{29}$$

Since $\mathbf{q}^*$ minimizes $d_\psi(\mathbf{p}, \mathbf{q})$ over $\mathbf{q} \in \mathcal{B}$, we have $\langle \nabla_{\mathbf{q}} d_\psi(\mathbf{p}, \mathbf{q}^*), \mathbf{v} - \mathbf{q}^* \rangle \geq 0$ which, in turn, implies:

$$d_\psi(\mathbf{u}, \mathbf{p}) - d_\psi(\mathbf{p}, \mathbf{q}^*) - \langle \nabla_{\mathbf{p}} d_\psi(\mathbf{p}, \mathbf{q}^*), \mathbf{u} - \mathbf{p} \rangle \geq 0.$$

Now we have:

$$\begin{aligned} & \delta_\psi(\mathbf{u}, \mathbf{p}) - d_\psi(\mathbf{u}, \mathbf{q}^*) \\ = \; & \psi(\mathbf{q}^*) - \psi(\mathbf{p}) - \langle \nabla \psi(\mathbf{q}^*), \mathbf{u} - \mathbf{q}^* \rangle - \langle \nabla \psi(\mathbf{p}), \mathbf{u} - \mathbf{p} \rangle \\ = \; & -d_\psi(\mathbf{p}, \mathbf{q}^*) - \langle \nabla \psi(\mathbf{p}) - \nabla \psi(\mathbf{q}^*), \mathbf{u} - \mathbf{p} \rangle \\ = \; & -d_\psi(\mathbf{p}, \mathbf{q}^*) - \langle \nabla_{\mathbf{p}} d_\psi(\mathbf{p}, \mathbf{q}^*), \mathbf{u} - \mathbf{p} \rangle \end{aligned}$$

Combining the above two equations, we have,

$$\delta_\psi(\mathbf{u}, \mathbf{p}) + d_\psi(\mathbf{u}, \mathbf{v}) \geq d_\psi(\mathbf{u}, \mathbf{q}^*) \tag{30}$$

Eq. (30) gets translated for J as follows (using definitions of δ_ψ and d_ψ):

$$\delta_J(\mathbf{y}^{1,(l)}, \mathbf{y}^{2,(l)}) + J(\mathbf{y}^{1,(l)}, \mathbf{y}^{3,(r)}) \geq J(\mathbf{y}^{1,(l)}, \mathbf{y}^{4,(r)^*}) \tag{31}$$

Hence, J satisfies 4-pp. □

We now introduce the main theorem that establishes the convergence guarantee of **OAC³**.

THEOREM A.12. *If* $\mathbf{y}^{(l,t)} = \underset{\mathbf{y}^{(l)} \in \mathcal{S}^n}{\operatorname{argmin}} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r,t-1)})$, $\mathbf{y}^{(r,t)} = \underset{\mathbf{y}^{(r)} \in \mathcal{S}^n}{\operatorname{argmin}} J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r)})$*, then* $\lim_{t \to \infty} J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t)}) = \inf_{\mathbf{y}^{(l)}, \mathbf{y}^{(r)} \in \mathcal{S}^n} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)})$.

PROOF. The proof here follows the same line of argument as given in [Wang and Schuurmans 2003a] and [Eggermont and LaRiccia 1998]. Since, $\mathbf{y}^{(r,t+1)} = \underset{\mathbf{y}^{(r)} \in \mathcal{S}^n}{\operatorname{argmin}} J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r)})$, we have, $J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t)}) - J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t+1)}) \geq 0$. By the 3-pp, $J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t+1)}) - J(\mathbf{y}^{(l,t+1)}, \mathbf{y}^{(r,t+1)}) \geq \delta_J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(l,t+1)})$. Then,

$$\begin{aligned} & J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t)}) - J(\mathbf{y}^{(l,t+1)}, \mathbf{y}^{(r,t+1)}) \\ = \; & J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t)}) - J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t+1)}) + J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t+1)}) - J(\mathbf{y}^{(l,t+1)}, \mathbf{y}^{(r,t+1)}) \\ \geq \; & \delta_J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(l,t+1)}) \geq 0. \end{aligned}$$

This implies that the sequence $J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t)})$ is non-increasing and non-negative. Let, $(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,\infty)}) = \underset{\mathbf{y}^{(l)}, \mathbf{y}^{(r)} \in \mathcal{S}^n}{\operatorname{argmin}} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)})$. From 4-pp and 3-pp, we can derive the following two inequalities:

$$\begin{aligned} J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,t+1)}) & \leq \delta_J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(l,t)}) + J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,\infty)}) \qquad (32) \\ \delta_J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(l,t+1)}) & \leq J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,t+1)}) - J(\mathbf{y}^{(l,t+1)}, \mathbf{y}^{(r,t+1)}). \qquad (33) \end{aligned}$$

Combining the above two inequalities, we get:

$$\delta_J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(l,t)}) - \delta_J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(l,t+1)}) \geq J(\mathbf{y}^{(l,t+1)}, \mathbf{y}^{(r,t+1)}) - J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,\infty)}) \geq 0, \tag{34}$$

which is the 5-points property (5-pp) of J. From (34), the sequence $\delta_J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(l,t)})$ is non-increasing and non-negative. Therefore, it must have a limit (from the Monotone Convergence Theorem) and consequently the left hand side of (34) approaches 0 as $t \to \infty$. Hence, $\lim_{t\to\infty} J(\mathbf{y}^{(l,t)}, \mathbf{y}^{(r,t)}) = J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,\infty)})$ (by the Pinching Theorem).

Finally, we must show that $\mathbf{y}^{(l,t)}$ and $\mathbf{y}^{(r,t)}$ themselves converge. From the boundedness of $\delta_J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(l,t)})$ (established in Lemma A.9), it follows that $\mathbf{y}^{(l,t)}$ is bounded. Therefore, it has a convergent subsequence $\{\mathbf{y}^{(l,t_i)}\}$ – the limit of which can be denoted by $\mathbf{y}^{0,(l)}$ (by the Bolzano-Weierstrass Theorem). Similarly, it can be shown that the subsequence $\{\mathbf{y}^{(r,t_i)}\}$ also converges to some limit. Let that limit be denoted by $\mathbf{y}^{0,(r)}$. By the lower-semi-continuity of J (established in Lemma A.1), we have:

$$J(\mathbf{y}^{0,(l)}, \mathbf{y}^{0,(r)}) \leq \liminf_i J(\mathbf{y}^{(l,t_i)}, \mathbf{y}^{(r,t_i)}) = J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,\infty)}). \tag{35}$$

We denote $\mathcal{Y}_l^\infty = \{\mathbf{y}^{(l)} : \underset{\mathbf{y}^{(l)},\mathbf{y}^{(r)}\in\mathcal{S}^n}{\arg\min} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)})\}$ and $\mathcal{Y}_r^\infty = \{\mathbf{y}^{(r)} : \underset{\mathbf{y}^{(l)},\mathbf{y}^{(r)}\in\mathcal{S}^n}{\arg\min} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)})\}$. Therefore, from the joint strict convexity of J, we should have $\mathcal{Y}_l^\infty = \{\mathbf{y}^{0,(l)}\} = \{\mathbf{y}^{(l,\infty)}\}$ and $\mathcal{Y}_r^\infty = \{\mathbf{y}^{0,(r)}\} = \{\mathbf{y}^{(r,\infty)}\}$.

To prove the convergence of the entire sequence, we apply the same logic as above with $\mathbf{y}^{(l,\infty)}$ replaced by $\mathbf{y}^{0,(l)}$. Then the sequence $\{\delta_J(\mathbf{y}^{0,(l)}, \mathbf{y}^{(l,t)})\}$ is bounded and non-increasing and by using Lemma A.9, we conclude that it has a convergent subsequence $\{\delta_J(\mathbf{y}^{0,(l)}, \mathbf{y}^{(l,t_i)})\}$ that goes to 0 as $\mathbf{y}^{(l,t_i)} \to \mathbf{y}^{0,(l)}$. This, from Monotone Convergence Theorem, implies that $\{\delta_J(\mathbf{y}^{0,(l)}, \mathbf{y}^{(l,t)})\} \to 0$ and again using Lemma A.9, we can conclude that $\mathbf{y}^{(l,t)} \to \mathbf{y}^{0,(l)}$. Since $\mathbf{y}^{(r,t)}$ is also bounded, it should have a convergent subsequence (by the Bolzano-Weierstrass Theorem). We denote this limit by $\mathbf{y}^{(0),(r)}$. Again, by the lower-semi-continuity of J, we have:

$$J(\mathbf{y}^{0,(l)}, \mathbf{y}^{(0),(r)}) \leq J(\mathbf{y}^{(l,\infty)}, \mathbf{y}^{(r,\infty)}). \tag{36}$$

Hence, $\mathbf{y}^{(0),(r)} = \underset{\mathbf{y}^{(r)}\in\mathcal{S}^n}{\arg\min} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r,\infty)})$ and $\mathbf{y}^{(r,t)} \to \mathbf{y}^{(0),(r)} = \mathbf{y}^{0,(r)}$. $\square$

There is another interesting aspect of J that was discovered in [Subramanya and Bilmes 2011] for a slightly different objective function with KL divergence used as a loss function. The same property also holds for J if the loss function is constructed from the assumed family of Bregman divergences. This property is concerned with the equality of solutions of J and J_0 and explores under what conditions these two objectives become equal. To establish the theorem that explores this condition, the following lemmata are essential.

LEMMA A.13. *If* $\mathbf{y}^{(r)} = \mathbf{y}^{(l)} = \mathbf{y}$ *then* $J_0 = J$.

PROOF. This proof immediately follows from the definitions of J_0 and J in Eq. (3) and Eq. (4) respectively. $\square$

LEMMA A.14. $\underset{(\mathbf{y}^{(l)},\mathbf{y}^{(r)})\in\mathcal{S}^n\times\mathcal{S}^n}{\arg\min} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}; \lambda = 0) \leq \underset{\mathbf{y}\in\mathcal{S}^n}{\arg\min} J_0(\mathbf{y})$.

PROOF.

$$\min_{\mathbf{y}\in\mathcal{S}^n} J_0(\mathbf{y}) = \min_{\left(\mathbf{y}^{(l)},\mathbf{y}^{(r)}\right)\in\mathcal{S}^n\times\mathcal{S}^n;\mathbf{y}^{(r)}=\mathbf{y}^{(l)}} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}; \lambda = 0) \geq \min_{\left(\mathbf{y}^{(l)},\mathbf{y}^{(r)}\right)\in\mathcal{S}^n\times\mathcal{S}^n} J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}; \lambda = 0)$$

The last step is due to the fact that the unconstrained minima is never larger than the constrained minima. $\square$

LEMMA A.15. *Given any* $\mathbf{y}^{(l)}$, $\mathbf{y}^{(r)}$, $\mathbf{y} \in \mathcal{S}^n$ *such that* $\mathbf{y}^{(l)}$, $\mathbf{y}^{(r)}$, $\mathbf{y} > \mathbf{0}$ *and* $\mathbf{y}^{(l)} \neq \mathbf{y}^{(r)}$ (i.e. *not all components are equal) then there exists a finite* λ *such that* $J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}) \geq J(\mathbf{y}, \mathbf{y}) = J_0(\mathbf{y})$.

PROOF. For $J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}) \geq J(\mathbf{y}, \mathbf{y})$, we should have:

$$\left[\sum_{i=1}^{n} d_\phi(\boldsymbol{\pi}_i, \mathbf{y}_i^{(r)}) + \alpha \sum_{i,j=1}^{n} s_{ij} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}) + \lambda \sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)})\right] - J(\mathbf{y}, \mathbf{y}) \geq 0$$

$$\Rightarrow \lambda \geq \frac{J(\mathbf{y}, \mathbf{y}) - \sum_{i=1}^{n} d_\phi(\boldsymbol{\pi}_i, \mathbf{y}_i^{(r)}) - \alpha \sum_{i,j=1}^{n} s_{ij} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)})}{\sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)})}$$

$$\Rightarrow \lambda \geq \frac{J_0(\mathbf{y}) - J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}; \lambda = 0)}{\sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)})} \geq 0.$$

where the last inequality follows from Lemma A.14. □

The theorem that formulates the conditions for equality of solutions of J and J_0 is given below:

THEOREM A.16 (EQUALITY OF SOLUTIONS OF J AND J_0). *Let* $\mathbf{y}^* = \underset{\mathbf{y} \in \mathcal{S}^n}{\arg\min}\, J_0(\mathbf{y})$ *and* $(\mathbf{y}^{\tilde{\lambda},(l)^*}, \mathbf{y}^{\tilde{\lambda},(r)^*}) = \underset{\mathbf{y} \in \mathcal{S}^n}{\arg\min}\, J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}; \tilde{\lambda})$ *for an arbitrary* $\lambda = \tilde{\lambda} > 0$. *Then there exists a finite* $\hat{\lambda}$ *such that at convergence of OAC*^{3}*, we have* $\mathbf{y}^* = \mathbf{y}^{\hat{\lambda},(l)^*} = \mathbf{y}^{\hat{\lambda},(r)^*}$. *Further, if* $\mathbf{y}^{\tilde{\lambda},(l)^*} \neq \mathbf{y}^{\tilde{\lambda},(r)^*}$, *then*

$$\hat{\lambda} \geq \frac{J_0(\mathbf{y}^*) - J(\mathbf{y}^{\tilde{\lambda},(l)^*}, \mathbf{y}^{\tilde{\lambda},(r)^*}; \lambda = 0)}{\sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{\tilde{\lambda},(l)}, \mathbf{y}_i^{\tilde{\lambda},(r)})}$$

and if $\mathbf{y}^{\tilde{\lambda},(l)^*} = \mathbf{y}^{\tilde{\lambda},(r)^*}$, *then* $\hat{\lambda} \geq \tilde{\lambda}$.

PROOF. If $\mathbf{y}^{\tilde{\lambda},(l)^*} = \mathbf{y}^{\tilde{\lambda},(r)^*}$, then from the strict convexity of both J_0 and J, $J_0(\mathbf{y}^*) = J(\mathbf{y}^{\tilde{\lambda},(l)^*}, \mathbf{y}^{\tilde{\lambda},(r)^*}; \lambda = 0)$. Also, since for any $\mathbf{y}^{(l)} \neq \mathbf{y}^{(r)}$, $J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}; \hat{\lambda}) > J(\mathbf{y}^{(l)}, \mathbf{y}^{(r)}; \tilde{\lambda})$, whenever $\hat{\lambda} \geq \tilde{\lambda}$, then $\forall \hat{\lambda} \geq \tilde{\lambda}$ $J_0(\mathbf{y}^*) = J(\mathbf{y}^{\hat{\lambda},(l)^*}, \mathbf{y}^{\hat{\lambda},(r)^*}; \lambda = 0)$. Also, if $\mathbf{y}^{\tilde{\lambda},(l)^*} \neq \mathbf{y}^{\tilde{\lambda},(r)^*}$, then from Lemma A.15, if

$$\infty > \hat{\lambda} \geq \frac{J_0(\mathbf{y}^*) - J(\mathbf{y}^{\tilde{\lambda},(l)^*}, \mathbf{y}^{\tilde{\lambda},(r)^*}; \lambda = 0)}{\sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{\tilde{\lambda},(l)}, \mathbf{y}_i^{\tilde{\lambda},(r)})}$$

then it is guaranteed that $\mathbf{y}^{\hat{\lambda},(l)^*} = \mathbf{y}^{\hat{\lambda},(r)^*}$. □

B. PROOF FOR ANALYSIS OF RATE OF CONVERGENCE

LEMMA B.1. *$\mathcal{H} = \nabla^2 J$ is positive definite over the domain of J under the assumption $\sum_{i=1}^{n}\sum_{\ell=1}^{k} \pi_{i\ell} > 0$ when KL or generalized I divergence is used as a Bregman divergence.*

PROOF. Assume $\mathbf{z} = \left(\left(\mathbf{y}_i^{(l)^\dagger}\right)_{i=1}^{n}, \left(\mathbf{y}_i^{(r)^\dagger}\right)_{i=1}^{n}\right)^\dagger$. Now,

$$\begin{aligned}
&\mathbf{z}^\dagger \mathcal{H}\mathbf{z} \qquad (37)\\
&= \sum_{i=1}^{n} \mathbf{y}_i^{(l)^\dagger} \nabla_{\mathbf{y}_i^{(l)},\mathbf{y}_i^{(l)}} \mathbf{y}_i^{(l)} + \sum_{j=1}^{n} \mathbf{y}_j^{(r)^\dagger} \nabla_{\mathbf{y}_j^{(r)},\mathbf{y}_j^{(r)}} \mathbf{y}_j^{(r)} + 2 \sum_{i,j=1;i\neq j}^{n} \mathbf{y}_i^{(l)^\dagger} \nabla_{\mathbf{y}_i^{(l)},\mathbf{y}_j^{(r)}} \mathbf{y}_j^{(r)}\\
&+ 2\sum_{i=1}^{n} \mathbf{y}_i^{(l)^\dagger} \nabla_{\mathbf{y}_i^{(l)},\mathbf{y}_i^{(r)}} \mathbf{y}_i^{(r)}\\
&= \sum_{i=1}^{n} \big(\alpha \sum_{j=1;j\neq i}^{n} s_{ij} + \lambda\big) \sum_{\ell=1}^{k} \mathbf{y}_{i\ell}^{(l)} + \sum_{j=1}^{n}\sum_{\ell=1}^{k} \big(\pi_{j\ell} + \alpha \sum_{i=1;i\neq j}^{n} s_{ij} y_{i\ell}^{(l)} + \lambda y_{j\ell}^{(l)}\big) - 2\lambda \sum_{i=1}^{n}\sum_{\ell=1}^{k} y_{i\ell}^{(l)}\\
&- 2\alpha \sum_{i,j=1;i\neq j}^{n} s_{ij} \sum_{\ell=1}^{k} y_{i\ell}^{(l)}\\
&= \sum_{i=1}^{n}\sum_{\ell=1}^{k} \pi_{i\ell} > 0.
\end{aligned}$$

Therefore, if $\sum_{i=1}^{n}\sum_{\ell=1}^{k} \pi_{i\ell} > 0$, $\nabla^2 J$ is positive definite over the domain of J. □

ACKNOWLEDGMENTS

We are grateful to Luiz F. S. Coletta for running the experiments described in Section 6.2. We also thank Ambuj Tewari and Ali Jalali for pointing us to relevant literature for analyzing the rate of convergence of the optimization framework. This work has been supported by NHARP, NSF Grants (IIS-0713142 and IIS-1016614) and by the Brazilian Research Agencies FAPESP and CNPq.

REFERENCES

ACHARYA, A., HRUSCHKA, E., GHOSH, J., AND ACHARYYA, S. 2012. Transfer learning with cluster ensembles. *JMLR Workshop and Conference Proceedings 27*, 123–132.

ACHARYA, A., HRUSCHKA, E. R., GHOSH, J., AND ACHARYYA, S. 2011. C^{3}E: A Framework for Combining Ensembles of Classifiers and Clusterers. In *10th Int. Workshop on MCS*.

BANERJEE, A., MERUGU, S., DHILLON, I. S., AND GHOSH, J. 2005. Clustering with bregman divergences. *J. Machine Learning Res. 6*, 1705–1749.

BELKIN, M., NIYOGI, P., AND SINDHWANI, V. 2005. On Manifold Regularization. In *AISTAT*.

BENGIO, Y., DELALLEAU, O., AND LE ROUX, N. 2006. Label Propagation and Quadratic Criterion. In *Semi-Supervised Learning*, O. Chapelle, B. Schölkopf, and A. Zien, Eds. MIT Press, 193–216.

BEZDEK, J. AND HATHAWAY, R. 2002. Some notes on alternating optimization. In *Advances in Soft Computing AFSS 2002*, N. Pal and M. Sugeno, Eds. Lecture Notes in Computer Science Series, vol. 2275. Springer Berlin / Heidelberg, 187–195.

BEZDEK, J. C. AND HATHAWAY, R. J. 2003. Convergence of alternating optimization. *Neural, Parallel Sci. Comput. 11,* 4, 351–368.

BLUM, A. 1998. On-line algorithms in machine learning. In *Online Algorithms: The State of the Art*, Fiat and Woeginger, Eds. LNCS Vol.1442, Springer.

BOLLACKER, K. D. AND GHOSH, J. 2000. Knowledge transfer mechanisms for characterizing image datasets. In *Soft Computing and Image Processing*. Physica-Verlag, Heidelberg.

BOYD, S., PARIKH, N., CHU, E., PELEATO, B., AND ECKSTEIN, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Tech Report.

BREGMAN, L. M. 1967. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics 7,* 3, 200 – 217.

CAI, W., CHEN, S., AND ZHANG, D. 2009. A simultaneous learning framework for clustering and classification. *Pattern Recogn. 42*, 1248–1259.

CARUANA, R. 1997. Multitask learning. *Machine Learning 28*, 41–75.

CENSOR, Y. A. AND ZENIOS, S. A. 1997. *Parallel Optimization: Theory, Algorithms and Applications*. Oxford University Press.

CHAPELLE, O., SCHÖLKOPF, B., AND ZIEN, A. 2006. *Semi-Supervised Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

CHEN, S., GUO, G., AND CHEN, L. 2009. Semi-supervised classification based on clustering ensembles. In *Proc. of AICI '09*. Springer-Verlag, 629–638.

CHENEY, W. AND GOLDSTEIN, A. A. 1959. Proximity maps for convex sets. *Proceedings of the American Mathematical Society 10,* 3, pp. 448–450.

CORDUNEANU, A. AND JAAKKOLA, T. 2003. On information regularization. In *UAI*. 151–158.

CSISZÁR, I. AND TUSNÁDY, G. 1984. Information geometry and alternating minimization procedures. *Statistics aand Decisions, Supplement Issue 1,* 1, 205–237.

DAI, W., XUE, G., YANG, Q., AND YU, Y. 2007a. Co-clustering based classification for out-of-domain documents. In *Proceedings of Knowledge Discovery and Data Mining*. New York, NY, USA, 210–219.

DAI, W., YANG, Q., RONG XUE, G., AND YU, Y. 2007b. Boosting for transfer learning. In *In ICML*.

DEMSAR, J. 2006. Statistical comparison of classifiers over multiple data sets. *Journal of Machine Learning Research 7,* 7, 1–30.

EGGERMONT, P. AND LARICCIA, V. 1998. On em-like algorithms for minimum distance estimation. Unpublished manuscript, University of Delaware.

FERN, X. AND BRODLEY, C. 2004. Solving cluster ensemble problems by bipartite graph partitioning. In *Proc. of International Conference on Machine Learning*. 281–288.

FORESTIER, G., GANÇARSKI, P., AND WEMMERT, C. 2010. Collaborative clustering with background knowledge. *Data Knowl. Eng. 69*, 211–228.

GAO, J., FAN, W., JIANG, J., AND HAN, J. 2008. Knowledge transfer via multiple model local structure mapping. In *Proceedings of Knowledge Discovery and Data Mining*. 283–291.

GAO, J., LIANG, F., FAN, W., SUN, Y., AND HAN, J. 2009. Graph-based consensus maximization among multiple supervised and unsupervised models. In *Proc. of Neural Information Processing Systems*. 1–9.

GAO, J., LIANG, F., FAN, W., SUN, Y., AND HAN, J. 2011. A graph-based consensus maximization approach for combining multiple supervised and unsupervised models. *IEEE Transactions on Knowledge and Data Engineering accepted for publication*.

GHOSH, J. AND ACHARYA, A. 2011. Cluster ensembles. *WIREs Data Mining and Knowledge Discovery 1*, 1–12.

GUNAWARDANA, A. AND BYRNE, W. 2005. Convergence theorems for generalized alternating minimization procedures. *J. Mach. Learn. Res. 6*, 2049–2073.

JOACHIMS, T. 1999a. Making large-scale SVM learning practical. In *Advances in Kernel Methods: Support Vector Learning*, C. B. B. Scholkopf and A. Smola, Eds. MIT Press, Cambridge, USA, 169–184.

JOACHIMS, T. 1999b. Transductive inference for text classification using support vector machines. In *Proc. of International Conference on Machine Learning*. 200–209.

JOACHIMS, T. 2003. Transductive learning via spectral graph partitioning. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*.

KUMAR, S., GHOSH, J., AND CRAWFORD, M. M. 2001. Best-bases feature extraction algorithms for classification of hyperspectral data. *IEEE TGRS 39,* 7, 1368–79.

KUNCHEVA, L. I. 2004. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, Hoboken, NJ.

OZA, N. C. AND TUMER, K. 2008. Classifier ensembles: Select real-world applications. *Inf. Fusion 9*, 4–20.

PAN, S. J. AND YANG, Q. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering 22*, 1345–1359.

POLIKAR, R. 2007. Bootstrap-inspired techniques in computational intelligence. *IEEE SIGNAL PROCESSING MAGAZINE*.

PUNERA, K. AND GHOSH, J. 2008. Consensus based ensembles of soft clusterings. In *Applied Artificial Intelligence*. Vol. 22. 109–117.

RAJAN, S., GHOSH, J., AND CRAWFORD, M. M. 2006. Exploiting class hierarchies for knowledge transfer in hyperspectral data. *IEEE TGRS 44,* 11, 3408–3417.

SILVER, D. L. AND BENNETT, K. P. 2008. Guest editor's introduction: special issue on inductive transfer learning. *Machine Learning 73*, 215–220.

SINDHWANI, V. AND KEERTHI, S. S. 2006. Large scale semi-supervised linear SVMs. In *Proc. of the 29th Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval*. NY, USA, 477–484.

SRIDHARAN, K. AND KAKADE, S. M. 2008. An information theoretic framework for multi-view learning. In *COLT*. 403–414.

STREHL, A. AND GHOSH, J. 2002a. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research 3 (Dec)*, 583–617.

STREHL, A. AND GHOSH, J. 2002b. Cluster ensembles – a knowledge reuse framework for combining partitionings. In *Proceedings of AAAI 2002, Edmonton, Canada*. AAAI, 93–98.

SUBRAMANYA, A. AND BILMES, J. 2011. Semi-supervised learning with measure propagation. *Journal of Machine Learning. Research 12*, 3311–3370.

SUBRAMANYA, A. AND BILMES, J. A. 2009. Entropic graph regularization in non-parametric semi-supervised classification. In *Proc. of Neural Information Processing Systems*. Vancouver, Canada.

THRUN, S. AND PRATT, L. 1997. *Learning To Learn*. Kluwer Academic, Norwell, MA.

TSUDA, K. 2005. Propagating distributions on a hypergraph by dual information regularization. In *Proceedings of the 22nd international conference on Machine learning*. ICML '05. ACM, New York, NY, USA, 920–927.

TUMER, K. AND GHOSH, J. 1996. Analysis of decision boundaries in linearly combined neural classifiers. *Pattern Recognition 29,* 2, 341–348.

WANG, H., SHAN, H., AND BANERJEE, A. 2009. Bayesian cluster ensembles. In *Proceedings of the Ninth SIAM International Conference on Data Mining*. 211–222.

WANG, H., SHAN, H., AND BANERJEE, A. 2011. Bayesian cluster ensembles. *Statistical Analysis and Data Mining 1*, 1–17.

WANG, J., JEBARA, T., AND CHANG, S. 2008. Graph transduction via alternating minimization. In *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, A. Mccallum and S. Roweis, Eds. Omnipress, 1144–1151.

WANG, S. AND SCHUURMANS, D. 2003a. Learning continuous latent variable models with bregman divergences. *2842*, 190–204.

WANG, S. AND SCHUURMANS, D. 2003b. Learning latent variable models with Bregman divergences. In *IEEE International Symposium on Information Theory*.

WELLING, M., ZEMEL, R. S., AND HINTON, G. E. 2002. Self supervised boosting. In *In Advances in Neural Information Processing Systems 15*. MIT Press, 665–672.

WU, C. F. J. 1982. On the convergence properties of the EM algorithm. *Annals of Statistics*.

XIE, S., FAN, W., AND YU, P. S. 2012. An iterative and re-weighting framework for rejection and uncertainty resolution in crowdsourcing. 1–12.

ZANGWILL, W. 1969. *Nonlinear Programming: a Unified Approach*. Prentice-Hall International Series in Management, Englewood Cliffs: N.J.

ZHANG, T., POPESCUL, A., AND DOM, B. 2006. Linear prediction models with graph regularization for web-page categorization. In *Proc. of the 12th ACM SIGKDD*. ACM, New York, NY, USA, 821–826.

ZHU, X. AND GHAHRAMANI, Z. 2002. Learning from labeled and unlabeled data with label propagation. Tech. rep., Carnegie Mellon University.

ZHU, X., GHAHRAMANI, Z., AND LAFFERTY, J. D. 2003. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *ICML*, T. Fawcett, N. Mishra, T. Fawcett, and N. Mishra, Eds. AAAI Press, 912–919.

ZHU, X. AND GOLDBERG, A. B. 2009. *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers.

Received April 2012; revised April 2012; accepted April 2012