Dictionary Learning Based Nonlinear Classifier Training from Distributed Data

Zahra Shakeri, Haroon Raja, and Waheed U. Bajwa
{zahra.shakeri, haroon.raja, waheed.bajwa}@rutgers.edu
Dept. of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ 08854

Abstract— **This paper addresses the problem of collaborative training of nonlinear classifiers using big, distributed training data. The supervised learning strategy considered in this paper corresponds to data-driven joint learning of a nonlinear transformation that maps the (training) data to a higher-dimensional feature space and a ridge regression based linear classifier in the feature space. The key aspect of this paper, which distinguishes it from related prior work, is that it assumes: (i) the training data are distributed across a number of interconnected sites, and (ii) sizes of the local training data as well as privacy concerns prohibit exchange of individual training samples between sites. The main contribution of this paper is formulation of an algorithm, termed *cloud D-KSVD*, that reliably, efficiently and collaboratively learns both the nonlinear map and the linear classifier under these constraints. In order to demonstrate the effectiveness of cloud D-KSVD, a number of numerical experiments on the MNIST dataset are also reported in the paper.**

I. Introduction

Classification is one of the most important information processing tasks. There exists an extensive body of literature on training classifiers from labeled data, but much of that work assumes the training data to be available at a centralized location [1], [2]. On the other hand, many disciplines in the world today—ranging from search engines to medical informatics—are increasingly faced with scenarios in which the training data are geographically distributed across different interconnected locations (sites). While each one of the sites in this setting can rely only on its local data for supervised learning, such an approach can be suboptimal due to issues ranging from noisy local data and labels to local class imbalance. At the same time, it might be infeasible in many of these cases to gather all the distributed data at a centralized location for supervised learning due to the massive nature of these data and/or privacy concerns. The challenge in this setting then is design of a *collaborative* supervised learning framework in which individual sites collaborate with each other to approach centralized classification performance *without* exchange of individual training samples between the sites.

In this paper, we undertake this challenge and develop a framework that collaboratively learns a nonlinear classifier at individual sites from the distributed training data. Our collaborative supervised learning strategy in this regard corresponds to data-driven collaborative and joint learning of a nonlinear transformation that maps (training) data in the input space $\mathbb{R}^n$ to a higher-dimensional feature space and a ridge regression based linear classifier in the feature space. In order to learn the nonlinear mapping, we resort to the framework of *dictionary learning* [3], [4] in computational harmonic analysis. Specifically, the task of dictionary learning corresponds to obtaining an overcomplete basis $D \in \mathbb{R}^{n \times K}, K \gg n$, such that each sample in the training data is well approximated by no more than $T_0 \ll n$ columns (*atoms*) of D. Such a dictionary, which is a linear map from $\mathcal{F}_{T_0} = \{x \in \mathbb{R}^K : \|x\|_0 \leq T_0\}$ to $\mathbb{R}^n$, in turn (under suitable conditions on D and T_0) induces a nonlinear map Φ_D from the *input space* $\mathbb{R}^n$ to the *feature space* $\mathcal{F}_{T_0}$ as follows:

$$\Phi_D(y) = \arg\min_{x \in \mathcal{F}_{T_0}} \|y - Dx\|_2. \qquad (1)$$

In the literature, evaluation of nonlinear maps of the form (1) for a given $y \in \mathbb{R}^n$ is termed *sparse coding* [4]. We can now use this terminology to formally describe the goal of this paper as follows: *collaborative exploitation of labeled training data distributed across sites for joint learning of a dictionary D (equivalently, the nonlinear map $\Phi_D : \mathbb{R}^n \to \mathcal{F}_{T_0}$) and a linear classification rule in $\mathcal{F}_{T_0}$.*

A. Our Contributions and Related Work

There are two main contributions of this paper. First, it develops a collaborative supervised learning framework for joint dictionary learning and linear classification rule from distributed training data. Our development in this regard leverages the centralized framework of [5] for joint dictionary and classifier learning, termed *discriminative K-SVD* (D-KSVD), and the collaborative framework of [6] for *reconstructive* dictionary learning from distributed data, termed *cloud K-SVD*. We accordingly term the framework developed in this paper as *cloud D-KSVD*. The second main contribution of this paper is that it evaluates the performance of cloud D-KSVD by carrying out a series of numerical experiments on the MNIST dataset of handwritten digits [7]. The results of these experiments confirm that collaborative supervised learning is superior to local supervised learning, especially in the presence of class imbalance at (some of the) individual sites. These experiments also demonstrate that the classification performance of our proposed framework not only comes very close to that of centralized supervised learning, but is also better than the classification performance of a collaborative framework based on cloud K-SVD alone.

In terms of connections to prior work, note that a number of dictionary learning based classifiers have been developed in the literature in recent years [5], [8]–[14]. Some of these works are based on reconstructive dictionary learning [8], [9], while others are based on discriminative dictionary learning [5], [10]–[14]. To the best of our knowledge, however, all of these works assume the (labeled or unlabeled) training data to be available at a centralized location. Recently, we proposed the collaborative framework of cloud K-SVD in [6] for reconstructive dictionary learning. In this regard, this paper can be viewed as a demonstration of the usefulness of some of the principles underlying cloud K-SVD for collaborative discriminative dictionary learning. While our focus in this paper has been on combining the ideas in cloud K-SVD and D-KSVD due to the superior classification performance of D-KSVD in a centralized setting, it is plausible that the D-KSVD part of our collaborative framework can be replaced with some of the other (centralized) discriminative dictionary learning approaches in the literature.

Outside the realm of dictionary learning, distributed classification has been studied in the literature in various guises. Some of the

This work is supported in part by the Army Research Office under grant W911NF-14-1-0295 and by an Army Research Lab Robotics CTA subaward.

earliest interest in this topic arose in the context of distributed sensor networks [15]–[19]. But the distributed classification problems studied in works like [15]–[18] primarily focus on *fusion* of distributed data for classification, rather than collaborative training of classifiers at individual sites from distributed data. Similarly, the focus in works like [19] is on collaborative *decision making*, rather than collaborative training, using related (but different) distributed measurements of the same object. In recent years, there has also been an interest in parallelizing supervised learning algorithms [20]–[23]. Such works, however, are based on the premise that training (labeled) data is initially available at a centralized location.

In terms of the distribution of labeled training data (see Fig. 1), our work is most closely related to [24]–[35]. In [24], [25], the authors collaboratively learn kernel-linear least-squares regression estimators from training data, which can in principle also be used for classification. In [26]–[35], the focus is on the collaborative training of (linear and/or kernel) *support vector machines* (SVMs). Although works [26], [27] require the sites to be connected in either a fully connected [26] or a ring [27] topology, other works [28]–[35] can deal with more general topologies. The fundamental difference between these works and our work is that we are interested in collaborative learning of both a nonlinear map and a classifier. In the context of kernel SVM training, this would be akin to joint, collaborative learning of a kernel and an SVM. To the best of our knowledge, however, none of the earlier works address such a problem.

Notational Convention: We use lowercase and uppercase letters to represent scalars/vector and matrices, respectively. Given a vector v, $[v]_i$ denotes its i-th element, $\|v\|_2$ represents its ℓ_2 norm, and $\|v\|_0$ counts the number of nonzero entries in it. Given a matrix A, $\|A\|_F$ denotes its Frobenius norm, while A^T denotes its transpose.

II. Problem Formulation

Consider a collection of N sites that are interconnected to each other. We express this collection through an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ and $\mathcal{E} = \{(i,j) \in \mathcal{V} \times \mathcal{V} :$ sites i and j are connected$\}$. Each of these N sites is interested in classifying n-dimensional data into one of L possible classes. In order to facilitate this classification task, we assume each site i has access to S_i labeled training samples $\{(y_i^j, \ell_i^j)\}_{j=1}^{S_i}$, where $y_i^j \in \mathbb{R}^n$ denotes a training sample, $\ell_i^j \in \mathcal{L}$ denotes the label of y_i^j, and $\mathcal{L} = \{1, \dots, L\}$. Given these $S = \sum_{i \in \mathcal{V}} S_i$ labeled training samples distributed across different sites, we are interested in collaboratively and jointly learning a nonlinear (feature) map Φ_D and a linear classifier $\mathcal{C}$ such that (ideally) $\mathcal{C}(\Phi_D(x)) = \ell_x$ for any sample $x \in \mathbb{R}^n$ that belongs to class $\ell_x \in \mathcal{L}$. Note that the composition $\mathcal{C} \circ \Phi_D : \mathbb{R}^n \to \mathcal{L}$ in this case is a nonlinear classifier in the input space.

In order to solve this problem, we resort to the framework of discriminative dictionary learning in which the nonlinear map Φ_D is induced by a dictionary $D \in \mathbb{R}^{n \times K}$ according to (1). We motivate that framework by collecting the training samples $\{y_i^j\}_{j=1}^{S_i}$ into a matrix $Y_i \in \mathbb{R}^{n \times S_i}$ and writing $Y = [Y_1, Y_2, \dots, Y_N]$. In addition, we associate with each label ℓ_i^j a *label unit-vector* $h_i^j = \mathrm{e}_{\ell_i^j} \in \mathbb{R}^L$, where $\mathrm{e}_{\ell_i^j}$ denotes the ℓ_i^j-th column of the $L \times L$ identity basis. Then, collecting the label vectors $\{h_i^j\}_{j=1}^{S_i}$ into a matrix $H_i \in \mathbb{R}^{L \times S_i}$ and writing $H = [H_1, H_2, \dots, H_N]$, the problem of joint learning of a dictionary $D \in \mathbb{R}^{n \times K}$ and a linear classifier $\mathcal{C}$ can be posed in terms of the following optimization problem [5]:

$$(D, W, X) = \arg\min_{D,W,X} \|Y - DX\|_F^2 + \gamma \|H - WX\|_F^2 + \beta \|W\|_F^2 \quad \text{such that} \quad \forall s = 1, \dots, S, \|x_s\|_0 \leq T_0. \quad (2)$$

Here, $x_s \in \mathbb{R}^K$ denotes the s-th column of the *coefficient matrix* $X \in \mathbb{R}^{K \times S}$, $W \in \mathbb{R}^{L \times K}$ denotes the classification matrix, and the final classification rule $\mathcal{C}$ is defined in terms of the matrix W as $\mathcal{C}(\Phi_D(x)) = \arg\max_{\ell \in \mathcal{L}} |[W \Phi_D(x)]_\ell|$. Note that the regularization parameters γ and β in (2) control the discriminative power and the complexity of the classifier, respectively.

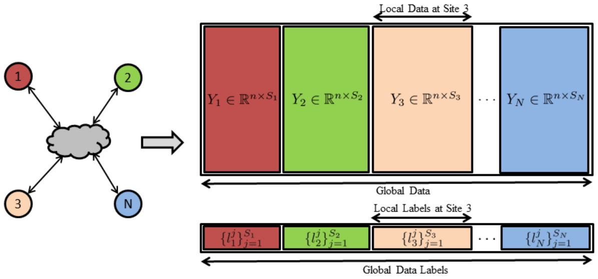


Fig. 1. An illustration of the distribution of labeled training data across sites.

While (2) is a non-convex problem, [5] provides a solution to this problem under the rubric of *discriminative K-SVD* (D-KSVD). But the D-KSVD framework, which relies on the K-SVD algorithm of [4] for dictionary learning, assumes (Y, H) to be available at one location. In contrast, our goal in this paper is to collaboratively solve (2) at each individual site when the training data is split across N sites (see Fig. 1). Given the nature of this problem, we can in fact only learn N different dictionary–classifier pairs $(\widetilde{D}_i, \widetilde{W}_i)$, one pair at each site, but our goal is to ensure that the classification performances of these pairs remain close to each other.

III. Proposed Collaborative Framework

In this section, we present our approach to collaborative learning of $(\widetilde{D}_i, \widetilde{W}_i)$ at each individual site from distributed training data. We term our proposed approach *cloud D-KSVD*, which is based on the centralized D-KSVD solution to (2) proposed in [5]. Before discussing cloud D-KSVD, however, we first provide a brief review of (centralized) D-KSVD for discriminative dictionary learning.

A. Review of Centralized D-KSVD

The key to the D-KSVD solution of [5] is transformation of the discriminative dictionary learning problem (2) into the classical reconstructive dictionary learning problem [4]. Specifically, notice that (2) can be rewritten in the following form:

$$(D, W, X) = \arg\min_{D,W,X} \left\| \begin{pmatrix} Y \\ \sqrt{\gamma} H \end{pmatrix} - \begin{pmatrix} D \\ \sqrt{\gamma} W \end{pmatrix} X \right\|_F^2 + + \beta \|W\|_F^2 \quad \text{such that} \quad \forall s, \|x_s\|_0 \leq T_0. \quad (3)$$

Next, define $\widehat{Y} \in \mathbb{R}^{n \times (S+L)} = \begin{bmatrix} Y^T & \sqrt{\gamma} H^T \end{bmatrix}^T$ as "training data" and $\widehat{D} \in \mathbb{R}^{(n+L) \times K} = \begin{bmatrix} D^T & \sqrt{\gamma} W^T \end{bmatrix}^T$ as "reconstructive dictionary." Then it is argued in [5] that making $\widehat{D}$ have unit ℓ_2-norm columns in (3) is heuristically sufficient to remove the $\beta \|W\|_F^2$ term in (3). In other words, [5] promotes the use of the following optimization program as a surrogate for (3):

$$(\widehat{D}, X) = \underset{\widehat{D}, X}{\operatorname{argmin}} \|\widehat{Y} - \widehat{D} X\|_F^2 \text{ such that } \forall s, \|x_s\|_0 \leq T_0. \quad (4)$$

1) Training Algorithm: The formulation in (4) reduces the problem of learning (D, W) from the training data to that of learning a reconstructive dictionary $\widehat{D}$ from $\widehat{Y}$. In the D-KSVD formulation, (4) is solved using the K-SVD dictionary learning algorithm [4]. This involves initialization with some $\widehat{D}^{(0)}$, followed by an alternate-minimization procedure that alternates between solving (4) first for

X by fixing $\widehat{D}$ and then for $\widehat{D}$ by fixing X. Specifically, assuming K-SVD has started iteration $t > 0$, it estimates $X^{(t)}$ by carrying out *sparse coding* as follows:

$$X^{(t)} = \arg\min_{X} \|\widehat{Y} - \widehat{D}^{(t-1)}X\|_F^2 \text{ such that } \forall s, \|x_s\|_0 \leq T_0. \tag{5}$$

Note that (5) can be efficiently solved using a number of greedy or optimization-based algorithms [4].

Next, K-SVD estimates $\widehat{D}^{(t)}$ by carrying out *dictionary update* as:

$$\widehat{D}^{(t)} = \arg\min_{D} \|\widehat{Y} - DX^{(t-1)}\|_F^2. \tag{6}$$

The main novelty of K-SVD lies in the manner it efficiently solves (6). To this end, K-SVD fixes all but the k-th column $\widehat{d}_k^{(t)}$, $k = 1, \dots, K$, of $\widehat{D}^{(t)}$ and then (dropping the iteration count for ease of notation) defines the representation error matrix $E_k = \widehat{Y} - \sum_{j \neq k} \widehat{d}_j x_T^j$, where x_T^j denotes the j-th row of $X^{(t)}$. Next, it obtains a column submatrix E_k^R of the matrix E_k by retaining those columns of E_k whose indices match the indices of the samples in $\widehat{Y}$ that utilize $\widehat{d}_k^{(t)}$. Finally, it updates $\widehat{d}_k^{(t)}$ by setting it equal to the dominant left singular vector of E_k^R. In addition, it is advocated in [4] to simultaneously update the k-th row of $X^{(t)}$ at this point by setting its nonzero entries equal to $\sigma_1 v_1^T$, where σ_1 and v_1 denote the largest singular value and right singular vector of E_k^R, respectively.

2) Classification Algorithm: Since K-SVD is guaranteed to converge (under appropriate conditions [4]), the D-KSVD algorithm obtains $\widehat{D}$ from (4). The next challenge then becomes splitting $\widehat{D} = \begin{bmatrix} D^T & \sqrt{\gamma} W^T \end{bmatrix}^T$ into a desired discriminative dictionary $\widetilde{D}$ and a classification matrix $\widetilde{W}$. One of the main contributions of [5] is establishing this relationship between the desired $(\widetilde{D}, \widetilde{W})$ and the (D, W) learned using (4). Specifically, [5] shows that

$$\widetilde{D} = \begin{bmatrix} \frac{d_1}{\|d_1\|_2} & \frac{d_2}{\|d_2\|_2} & \cdots & \frac{d_K}{\|d_K\|_2} \end{bmatrix}, \text{ and} \tag{7}$$

$$\widetilde{W} = \begin{bmatrix} \frac{w_1}{\|d_1\|_2} & \frac{w_2}{\|d_2\|_2} & \cdots & \frac{w_K}{\|d_K\|_2} \end{bmatrix}. \tag{8}$$

Once the pair $(\widetilde{D}, \widetilde{W})$ is obtained, the classification proceeds as follows. Given a test sample $\widetilde{y} \in \mathbb{R}^n$ that belongs to one of the L classes in $\mathcal{L}$, we first obtain

$$\widetilde{x} = \Phi_{\widetilde{D}}(\widetilde{y}) = \operatorname*{argmin}_{x} \|\widetilde{y} - \widetilde{D}x\|_2^2 \text{ such that } \|x\|_0 \leq T_0. \tag{9}$$

Next, we define $\widetilde{h} = \widetilde{W}\widetilde{x}$ and then use the classification rule $\mathcal{C}(\Phi_{\widetilde{D}}(\widetilde{y})) = \arg\max_{\ell \in \mathcal{L}} |[\widetilde{h}]_\ell|$, where $[\widetilde{h}]_\ell$ is the ℓ-th entry of $\widetilde{h}$.

B. Cloud D-KSVD

We are now ready to discuss our proposed collaborative framework for discriminative dictionary learning. Similar to D-KSVD, we are interested in solving (4) for D at each site. But the major difference is that $\widehat{Y} = [\widehat{Y}_1, \widehat{Y}_2, \dots, \widehat{Y}_N]$ is now distributed across N sites, where $\widehat{Y}_i = \begin{bmatrix} Y_i^T & \sqrt{\gamma} H_i^T \end{bmatrix}^T$.

1) Initialization: Unlike D-KSVD, initialization of $\widehat{D}^{(0)}$ in (4) is also a function of the training data at individual sites. In cloud D-KSVD, we proceed with the initialization of the dictionary $\widehat{D}_i^{(0)}$ locally at the i-th site as follows. First, we initialize a dictionary $D_i^{(0)} \in \mathbb{R}^{n \times K}$ and carry out local sparse coding using $D_i^{(0)}$, i.e.,

$$X_i = \arg\min_{X \in \mathbb{R}^{K \times S_i}} \|Y_i - D_i^{(0)} X\|_F^2 \text{ such that } \forall s, \|x_s\|_0 \leq T_0. \tag{10}$$

Next, we initialize a local classifier matrix $W_i^{(0)}$ by solving

$$W_i^{(0)} = \arg\min_{W} \|H_i - WX_i\|_F^2 + \beta \|W\|_F^2. \tag{11}$$

Note that (11) is simply a multivariate ridge regression problem, with the closed-form solution given by

$$W_i^{(0)} = (X_i X_i^T + \beta I)^{-1} X_i H_i^T. \tag{12}$$

Finally, we set the initial dictionary at the i-th site, $i \in \mathcal{V}$, as follows: $\widehat{D}_i^{(0)} = \begin{bmatrix} D_i^{(0)T} & \sqrt{\gamma} W_i^{(0)T} \end{bmatrix}^T$.

2) Training Algorithm: After initialization, each site $i \in \mathcal{V}$ has access to $\widehat{D}_i^{(0)}$ that is obtained using local data only. Our next goal is to solve (4) at each site for $\widehat{D}_i \in \mathbb{R}^{(n+L) \times K}$ by relying on a collaborative variant of K-SVD that alternates between solving (4) first for (global) X by fixing $\widehat{D}_i$ at each site and then for $\widehat{D}_i$ by fixing $X = [X_1, X_2, \dots, X_N]$, which will always be partitioned across the N sites. In our recent work [6], we have proposed such a collaborative variant using the moniker of *cloud K-SVD*. Specifically, assuming cloud K-SVD has started iteration $t > 0$ in the network, each site only updates the sparse representation of its local $\widehat{Y}_i$ through sparse coding as follows:

$$X_i^{(t)} = \arg\min_{X} \|\widehat{Y}_i - \widehat{D}_i^{(t-1)} X\|_F^2 \text{ s.t. } \forall s, \|x_s\|_0 \leq T_0. \tag{13}$$

Next, sites focus on collaboratively updating their individual dictionary estimates $\{\widehat{D}_i^{(t)}\}_{i \in \mathcal{V}}$. In this regard, cloud K-SVD takes its cue from K-SVD and fixes all but the k-th column $\widehat{d}_{i,k}^{(t)}$ of $\widehat{D}_i^{(t)}$ at each site. The next challenge then is defining the *global, reduced* representation error matrix E_k^R (we are once again dropping the iteration count for ease of notation), since there are N different versions of dictionaries in the network. In order to address this challenge, cloud K-SVD first defines *local* representation error matrices $E_{i,k} = \widehat{Y}_i - \sum_{j \neq k} \widehat{d}_{i,j} x_{i,T}^j$, where $x_{i,T}^j$ denotes the j-th row of $X_i^{(t)}$. It then obtains a submatrix $E_{i,k}^R$ of $E_{i,k}$ by retaining the columns of $E_{i,k}$ whose indices match the indices of the samples in $\widehat{Y}_i$ that utilize $\widehat{d}_{i,k}^{(t)}$. Finally, it defines the global, reduced representation error matrix as $E_k^R = \begin{bmatrix} E_{1,k}^R, E_{2,k}^R, \dots, E_{N,k}^R \end{bmatrix}$, which is distributed across the network. Cloud K-SVD then advocates to update $\widehat{d}_{i,k}^{(t)}$ by setting it equal to the dominant left singular vector u_1 of E_k^R. Note that u_1 is also equal to the dominant eigenvector of $M = E_k^R E_k^{R\,T} = \sum_{i \in \mathcal{V}} M_i$, where M_i denotes $E_{i,k}^R E_{i,k}^{R\ T}$. One of the main novelties of cloud K-SVD in this regard is formulation of a collaborative variant of the classical power method [36] for estimation of the dominant eigenvector of M. This variant, which is described and rigorously analyzed in [6], relies on a finite number of iterations of distributed consensus averaging [37]. While more details of this part of cloud K-SVD can be found in [6], including a discussion of the doubly-stochastic mixing matrix needed for distributed consensus, the end result is that each site obtains an updated $\widehat{d}_{i,k}^{(t)}$ that can come arbitrarily close to u_1. Finally, cloud K-SVD also simultaneously updates the k-th row of $X_i^{(t)}$ at this point by setting its nonzero entries equal to $\widehat{d}_{i,k}^T E_{i,k}^R$.

3) Classification Algorithm: The classification algorithm in cloud D-KSVD is identical to that in D-KSVD. Specifically, each site at this point obtains a dictionary $\widehat{D}_i = \begin{bmatrix} D_i^T & \sqrt{\gamma} W_i^T \end{bmatrix}^T$, which is then transformed into the final pair $(\widetilde{D}_i, \widetilde{W}_i)$ according to (7) and (8). Using this pair, each site can then individually classify any test sample $\widetilde{y} \in \mathbb{R}^n$ according to the procedure described in Sec. III-A2.

IV. Numberical Results

In this section, we demonstrate the effectiveness of cloud D-KSVD. We use the MNIST database [7], which consists of 28×28 pixel images of handwritten digits. For simplicity, each image is down-sampled to have only 256 features. We work on digits 0 to 4 in experiments ($L = 5$) and we consider a total of 10 sites. We perform

5-fold cross validation on the database by treating $\frac{1}{5}$ of the data as test data and the rest as training data in each fold. In the first set of experiments, we divide the training data uniformly between the 10 sites. We train dictionaries using centralized D-KSVD (assuming all data is available at a single location), cloud D-KSVD, local D-KSVD (assuming each site performs training on local training data only) and cloud K-SVD (sites collaboratively learn purely representative dictionaries, one for each class). We also train a linear SVM for the centralized data for comparison with cloud D-KSVD.

To initial the discriminative dictionaries, we first perform 10 iterations of K-SVD for the centralized and local setting and cloud K-SVD in the distributed setting. We then initial the classifiers according to (12) using these initial dictionaries. Then, we perform 50 iterations of D-KSVD for centralized and local setting and cloud D-KSVD for distributed setting. The parameters selected in these experiments correspond to a sparsity constraint of $T_0 = 10$, $\gamma = 0.83$ and $K = 500$ number of dictionary atoms (100 atoms for each class).

For the representative dictionary, we train a separate dictionary for each data label by performing 60 iterations of cloud K-SVD. We set $T_0 = 10$ and $K = 100$ for each dictionary (total of 500 atoms for 5 dictionaries). To classify a test data sample, the coefficient vector of the test sample is obtained for each dictionary using sparse coding. The assigned class to the sample is the index of the dictionary that best represents the sample (has the least representation error).

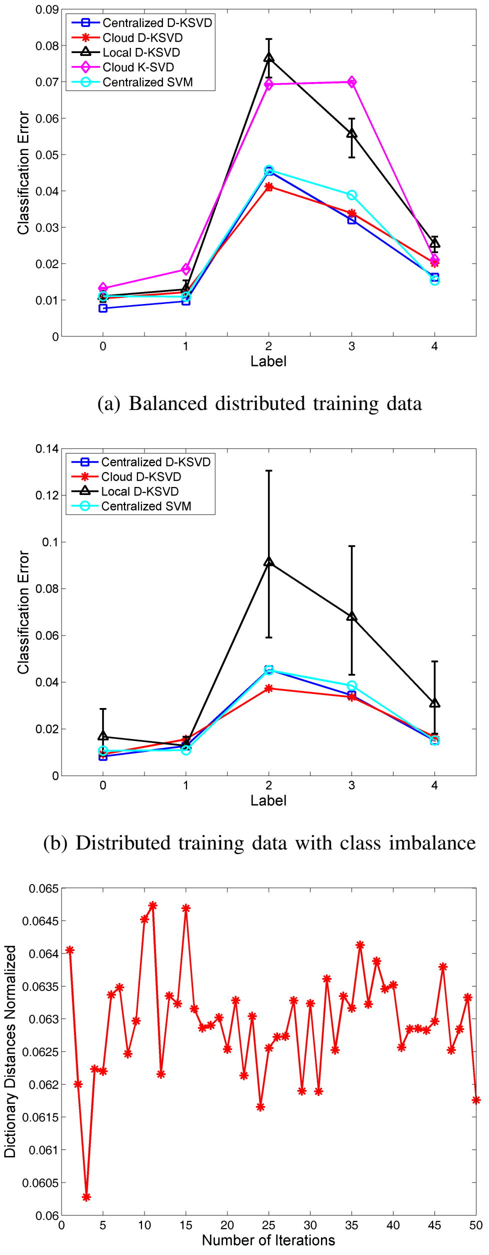
The test data classification results are shown in Fig. 2(a) where the sites' average classification error is plotted along with the worst case and best case error for each label for cloud D-KSVD, local D-KSVD and cloud K-SVD. The results demonstrate that cloud D-KSVD outperforms local D-KSVD and has a performance close to the centralized D-KSVD and centralized linear SVM. Also, the classification performance of various sites is approximately identical when using cloud D-KSVD due to the fact that they are collaborating with one another. Note that non-linear SVM will likely outperform linear SVM, but we do not make the comparison with non-linear SVM here as our parameters are not optimally chosen. Finally, observing the classification error of cloud D-KSVD and cloud K-SVD, it is evident that cloud D-KSVD outperforms cloud K-SVD for all the class labels.

In the second set of experiments, we consider the case of the sites not having the same number of training data. In real world applications, some sites may have access to a smaller number of training data and there may be class imbalance in some sites (different class sizes). We consider that 80% of the labeled data is distributed among half of the sites, while the other 20% is distributed among the other half of the sites. The chosen parameters are similar to the previous simulations. The classification errors for this case are plotted in Fig. 2(b). It is apparent that distributed learning of the dictionary and classifier has a great advantage over training based on locally available data for sites with a smaller number of training data.

In the case of balanced data across sites, the normalized distance of the dictionary learned by centralized D-KSVD, $\widehat{D}_C$, and the one learned by cloud D-KSVD at site i, $\widehat{D}_{D,i}$, as a function of the number of dictionary learning iterations is defined as

$$d^{(t)} = \frac{1}{K} \left\| \widehat{D}_C^{(t)} - \widehat{D}_{D,i}^{(t)} \right\|_F^2, \; t = \{1, 2, ..., 50\}, \; i \in \mathcal{V}. \qquad (14)$$

Fig. 2(c) plots this normalized distance averaged over 10 sites along with the least and most normalized distance as a function of the number of iterations. It is evident that the average normalized distance does not vary significantly across different iterations and sites obtain similar dictionaries.



(a) Balanced distributed training data

(b) Distributed training data with class imbalance

(c) Iterative behavior of cloud D-KSVD

Fig. 2. Performance summary of cloud D-KSVD. (a) and (b) compare the classification performance of cloud D-KSVD with that of centralized and local D-KSVD, centralized linear SVM, and cloud K-SVD. The results for cloud D-KSVD, local D-KSVD and cloud K-SVD are displayed using bars to highlight the best, worst, and average error across sites. (c) displays the average normalized distance along with the least and most normalized distance between the dictionaries obtained using cloud D-KSVD and centralized D-KSVD as a function of the number of dictionary learning iterations.

V. Conclusion

In this paper, we developed a collaborative framework for learning a nonlinear classifier from distributed data. Our framework corresponded to joint learning of a dictionary and a linear classifier by leveraging recent results on discriminative and collaborative dictionary learning. In order to verify the effectiveness of our approach, we carried out numerical experiments that showed that the performance of our framework comes very close to that of centralized methods.

References

[1] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, "Machine learning, neural and statistical classification," *Ellis Horwood Series in Artificial Intelligence, Ellis Horwood*, 1994.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*, vol. 2, Springer, 2009.

[3] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural computation*, vol. 15, no. 2, pp. 349–396, 2003.

[4] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[5] Q. Zhang and B. Li, "Discriminative K-SVD for dictionary learning in face recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2010)*, 2010, pp. 2691–2698.

[6] H. Raja and W.U. Bajwa, "Cloud K-SVD: Computing data-adaptive representations in the cloud.," in *Proc. 51st Annu. Allerton Conf. Communication, Control, and Computing*, 2013, pp. 1474–1481.

[7] Y. LeCun and C. Cortes, "The MNIST database of handwritten digits," http://yann.lecun.com/exdb/mnist/, 1998.

[8] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Discriminative learned dictionaries for local image analysis," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2008)*, 2008, pp. 1–8.

[9] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. 26th Annu. Int. Conf. Machine Learning*, 2009, pp. 689–696.

[10] F. Rodriguez and G. Sapiro, "Sparse representations for image classification: Learning discriminative and reconstructive non-parametric dictionaries," Tech. Rep., DTIC Document, 2008, http://www.dsp.ece.rice.edu/cs/.

[11] D. Pham and S. Venkatesh, "Joint learning and dictionary construction for pattern recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVRP 2008)*, 2008, pp. 1–8.

[12] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. Bach, "Supervised dictionary learning," in *Proc. Advances in Neural Information Processing Systems*, 2009, pp. 1033–1040.

[13] Z. Jiang, Z. Lin, and L. S. Davis, "Learning a discriminative dictionary for sparse coding via label consistent K-SVD," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2011)*, 2011, pp. 1697–1704.

[14] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Trans. Pattern Anal. and Mach. Intell.,*, vol. 34, no. 4, pp. 791–804, 2012.

[15] A. D'Costa and A. M. Sayeed, "Data versus decision fusion for classification in sensor networks," in *Proc. Int. Conf. Information fusion*, 2003.

[16] A. D'Costa, V. Ramachandran, and A. M. Sayeed, "Distributed classification of Gaussian space-time sources in wireless sensor networks," *IEEE J. Sel. Areas Commun.,*, vol. 22, no. 6, pp. 1026–1036, 2004.

[17] J. H. Kotecha, V. Ramachandran, and A. M. Sayeed, "Distributed multitarget classification in wireless sensor networks," *IEEE J. Sel. Areas Commun.,*, vol. 23, no. 4, pp. 703–713, April 2005.

[18] E. Kokiopoulou and P. Frossard, "Distributed SVM applied to image classification," in *Proc. IEEE Int. Conf. Multimedia and Expo*, July 2006, pp. 1753–1756.

[19] E. Kokiopoulou and P. Frossard, "Distributed classification of multiple observation sets by consensus," *IEEE Trans. Signal Process.,*, vol. 59, no. 1, pp. 104–114, Jan 2011.

[20] H. P. Graf, E. Cosatto, L. Bottou, I. Dourdanovic, and V. Vapnik, "Parallel support vector machines: The cascade SVM," in *Advances in neural information processing systems*, 2004, pp. 521–528.

[21] T. Do and F. Poulet, "Classifying one billion data with a new distributed SVM algorithm," in *Proc. Int. Conf. Research, Innovation and Vision for the Future*, 2006, pp. 59–66.

[22] K. Zhu, H. Wang, H. Bai, J. Li, Z. Qiu, H. Cui, and E. Y. Chang, "Parallelizing support vector machines on distributed computers," in *Advances in Neural Information Processing Systems*, 2008, pp. 257–264.

[23] D. Mahajan, S. S. Keerthi, and S. Sundararajan, "A distributed algorithm for training nonlinear kernel machines," *arXiv preprint arXiv:1405.4543*, 2014.

[24] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: An efficient framework for modeling sensor network data," in *Proc. 3rd Int. Symp. Information Processing in Sensor Networks*, 2004, pp. 1–10.

[25] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "A collaborative training algorithm for distributed learning," *IEEE Trans. Inf. Theory*, vol. 55, no. 4, pp. 1856–1871, April 2009.

[26] A. Navia-Vazquez, D. Gutierrez-Gonzalez, E. Parrado-Hernandez, and J.J. Navarro-Abellan, "Distributed support vector machines," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 1091–1097, July 2006.

[27] K. Flouri, B. Beferull-Lozano, and P. Tsakalides, "Training a SVM-based classifier in distributed sensor networks," in *Proc.14th European Signal Processing Conf.*, 2006, pp. 1–5.

[28] K. Flouri, B. Beferull-Lozano, and P. Tsakalides, "Distributed consensus algorithms for SVM training in wireless sensor networks," in *Proc. 16th European Signal Processing Conf.*, 2008, pp. 25–29.

[29] K. Flouri, B. Beferull-Lozano, and P. Tsakalides, "Optimal gossip algorithm for distributed consensus SVM training in wireless sensor networks," in *Proc. 16th Int. Conf. Digital Signal Processing,*, July 2009, pp. 1–6.

[30] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed linear support vector machines," in *Proc. 9th ACM/IEEE Int. Conf. Information Processing in Sensor Networks*, 2010, pp. 35–46.

[31] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *J. Mach. Learn. Res.*, vol. 11, pp. 1663–1707, Aug. 2010.

[32] M. R. Guarracino, A. Irpino, N. Radziukyniene, and R. Verde, "Supervised classification of distributed data streams for smart grids," *Energy Systems*, vol. 3, no. 1, pp. 95–108, 2012.

[33] S. Lee and A. Nedic, "DrSVM: Distributed random projection algorithms for SVMs," in *Proc. IEEE 51st Annu. Conf. Decision and Control (CDC)*, Dec 2012, pp. 5286–5291.

[34] S. Lee and A. Nedic, "Distributed random projection algorithm for convex optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 2, pp. 221–229, April 2013.

[35] Y. Lu, V. Roychowdhury, and L. Vandenberghe, "Distributed parallel support vector machines in strongly connected networks," *IEEE Trans. Neural Netw.,*, vol. 19, no. 7, pp. 1167–1178, July 2008.

[36] G. H. Golub and C. F. Van Loan, *Matrix computations*, vol. 3, Baltimore MD: John Hopkins University Press, 2012.

[37] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Lett.*, vol. 53, no. 1, pp. 65–78, 2004.