Discriminative Partition Sparsity Analysis

Li Liu
Department of Electronic and Electrical Engineering
The University of Sheffield
Sheffield, S1 3JD
United Kingdom
Email: elp11ll@sheffield.ac.uk

Ling Shao
Department of Electronic and Electrical Engineering
The University of Sheffield
Sheffield, S1 3JD
United Kingdom
Email: ling.shao@sheffield.ac.uk

Abstract*—Effective dimensionality reduction has been an attractive research area for many large-scale vision and multimedia tasks. Several recent methods attempt to learn optimized graph-based embedding for fast and accurate applications. In this paper, we propose a novel linear unsupervised algorithm, termed Discriminative Partition Sparsity Analysis (DPSA), explicitly considering different probabilistic distributions that exist over the data points, meanwhile preserving the natural locality relationship among the data. Specifically, the Gaussian mixture model (GMM) is first applied to partition all samples into several clusters. In each cluster, a number of sparse sub-graphs are computed via the $\ell1$-norm constraint to optimally represent the intrinsic data structure. Such sub-graphs are demonstrated to be robust to data noise, automatically sparse and adaptive to the neighborhood. All the sub-graphs from the clusters are then combined into a whole discriminative optimization framework for final reduction. We have systematically evaluated our method on three image datasets: USPS digital hand-writing, CMU PIE face and CIFAR-10 tiny image, showing its accurate and robust performance for image classification.

I. Introduction

Dimensionality reduction [1], [2] has been a key problem attracting much attention in many fields of information processing, such as data mining [3], information retrieval, and pattern recognition [4], [5]. When data are represented as points in a high-dimensional space, one is often confronted with tasks like nearest neighbor search. Actually, greedily searching a dataset with N samples is infeasible because linear complexity $O(N)$ is not scalable in practical applications. To overcome this shortcoming of linear search, researchers have developed methods to index the data for fast query response, such as K-D tree, R tree, R* tree [6], which efficiently decrease the computational complexity from linear $O(N)$ to sublinear $O(N^{\frac{1}{2}})$. However, when the dimension of data points is more than 100 bits, the effectiveness and efficiency of those searching schemes will drop exponentially as the dimensionality increases, which is commonly referred to as the 'curse of dimensionality'. Therefore, to make large-scale search or classification practical, many methods have been proposed to effectively reduce the dimension of data and increase the classification speed and accuracy. Among them, linear embedding techniques show their promising efficiency and robustness for scalable data reduction tasks.

One baseline linear reduction algorithm is Principal Component Analysis (PCA) [7], which is designed to explain the variance-covariance structure of a set of variables through linear combinations. PCA is commonly applied to condense the information when the data manifold is embedded linearly or almost linearly in the ambient space. If the class information is available, Fisher Discriminative Analysis (FDA) [8] can be used to find an optimal subspace for discrimination where the projection vectors are commonly obtained by maximizing the between-class covariance and simultaneously minimizing the within-class covariance. FDA has been proved to be successful on classification problems [9], [10].

Furthermore, another popular linear technique, termed Locality Preserving Projections (LPP) [11], has been proposed for dimensionality reduction that preserves local relationships within the data set and uncovers its essential manifold structure. After that, a Semi-supervised Discriminative Analysis (SDA) [12] embedding scheme has been developed, as well. SDA inherits the advantages from both FDA and LPP to find an appropriate subspace which perceives the intrinsic data structure from the high dimensional space and also maximizes the inter-class variation. Besides, a subspace learning algorithm called Neighborhood Preserving Embedding (NPE) [13] has been used for linear reduction. Different from PCA, which aims at keeping the global Euclidean structure, NPE also aims at keeping the natural neighborhood structure on the data manifold.

Compared with non-linear methods, the above linear techniques are computationally much cheaper. Moreover, they yield projections that are not only defined on training data points, but also efficient for 'out-of-sample' extensions on novel test data. Naturally, for those 'Big Data' applications, we cannot manually annotate the ground truth for all training samples. Thus, efficient reduction methods without using label information but can still well describe the data manifold in the projected space are badly in need.

In this paper, we develop a novel unsupervised linear dimensionality reduction algorithm, called Discriminative Partition Sparsity Analysis (DPSA). From the data probabilistic distribution point of view, samples in the high-dimensional space do not always follow the same distribution, but are naturally clustered into several groups. The data in each groupshare the same probabilistic distribution. To keep this property, we first apply the Gaussian Mixture Model (GMM) [14] to partition the training samples into different clusters. We then build a sub-graph weight matrix to describe the relationship between the data points in each cluster. Specifically, each data point is reconstructed as the linear combination of the remaining data samples in a cluster by minimizing the $\ell 1$-norm of both the reconstruction coefficients and data noise, and the combination coefficients are designated as the values in the weight matrix. Different from constructing weight matrices via a manual neighborhood constraint as in LPP and NPE, the $\ell 1$ weight matrix is more robust to data noise and can automatically realize sparsity. Besides, the neighbors selected through the $\ell 1$ are also data-adaptive, which can discover the natural locality information of the data manifold and be a nice property for applications with uneven data distributions [15]. We further align these partitioned sub-graphs and obtain the final projection via a general linear reduction framework.

The remainder of this paper is organized as follows. In Section 2, we briefly review linear reduction techniques. The architecture of DPSA and the complexity analysis are presented in Sections 3. Experiments and results are described in Section 4. In Section 5, we conclude this paper.

II. Linear Reduction framework

In this Section, we provide a general framework for the existing subspace learning algorithms from the graph embedding point of view.

Given a graph $G = \{X, W\}$ with N vertices, each of which represents a data point, let W be a symmetric $N \times N$ matrix with $W_{i,j}$ having the weight of the edge joining vertices i and j. G and W can be defined to characterize certain statistical or geometric properties of the data set. The purpose of graph embedding is to represent each vertex of the graph as a low dimensional vector that preserves similarities between the vertex pairs, where similarity is measured by the edge weight.

Let us now consider a set of samples X=$[x_1, x_2 \ldots, x_i, \ldots, x_N] \in \mathbb{R}^{m\times N}$. The problem of linear dimensionality reduction is to find a projection matrix $U \in \mathbb{R}^{m\times d}$ that maps $X \in \mathbb{R}^{m\times N}$ to Y=$[y_1, y_2, \ldots, y_i, \ldots, y_N] \in \mathbb{R}^{d\times N}$, *i.e.*, Y=U^TX, where $d < m$. The basic idea of linear reduction is to preserve the intrinsic data structure in the projected low-dimensional space. The optimal Y is given by minimizing:

$$argmin \sum_{i,j=1}^{N} ||y_i - y_j||_2^2 W_{i,j} \tag{1}$$

under an appropriate constraint. This objective function incurs a heavy penalty if neighboring vertices i and j are mapped far apart. Therefore, minimizing it is an attempt to ensure that if vertices i and j are 'close', then y_i and y_j are close as well. $W_{i,j}$ is the graph weight over the whole data set and $y_i, y_j \in Y$.

Obviously, using different graph embedding techniques will lead to different dimensionality reduction performance. In the next section, we will present our Discriminative Partition Sparsity Analysis (DPSA) via $\ell 1$ sparse graph construction.

III. Discriminative Partition Sparsity Analysis

Discriminative Partition Sparsity Analysis (DPSA) is proposed to preserve the locality information on different data distributions for dimensionality reduction. It operates in three stages. In the first stage, each sample in the dataset will be assigned to an individual cluster via the Gaussian mixture model (GMM). Thus, the whole dataset can be automatically partitioned into several groups. In the second stage, for samples in each cluster, an objective function is designed to construct a sparse sub-graph via the $\ell 1$-norm constraint, which can successfully preserve the local discriminative information. Since samples in one cluster can be seen as a part of the whole dataset, this stage is termed 'part optimization'. We then align all the part optimizations together to form a global coordinate. In the last step, termed 'global optimization', the projection matrix is obtained through a global alignment by solving a generalized eigenvalue eigenvector decomposition problem. Our proposed DPSA is outlined in Fig. 1.

It is worthwhile to highlight several aspects of the proposed approach here. DSPA shares some similar properties with other graph embedding algorithms, all of which aim to discover the local structure of the data manifold. However, our objective function is totally different from others. Furthermore, DSPA is regarded as a linear embedding. This makes it fast and suitable for practical applications. Finally, DSPA is also an unsupervised scheme, which is better for large-scale tasks where the label information is often unavailable.

A. Part Optimization

In the data space, each data point can be naturally assigned to a potential cluster, in which all samples share the same probabilistic distribution. Meanwhile, samples from different clusters always have different probability density functions. However, in either statistics or physics, real-world data distribution basically follows the same form, *i.e.*, Gaussian distribution. Therefore, each potential cluster could be Gaussian distributed but with different probabilistic parameters. Thus, how to estimate the Gaussian parameters for different data distributions becomes a core problem.

The Gaussian Mixture Model (GMM) is one of the most popular data clustering methods that can be viewed as a linear combination of different Gaussian components. In GMM, each cluster obeys Gaussian distribution. The task of clustering is to group observations into different components through estimating each cluster's own parameters *i.e.*, ϕ, μ, Σ, under their likelihood function:

$$\begin{aligned} l(\phi, \mu, \Sigma) &= \sum_{i=1}^{m} \log p(x^{(i)}; \phi, \mu, \Sigma) \\ &= \sum_{i=1}^{m} \log \sum_{z^{(i)}=1}^{K} p(x^{(i)}|z^{(i)}; \mu, \Sigma) p(z^{(i)}; \phi) \end{aligned} \tag{2}$$

Later, the Expectation Maximization (EM) algorithm [16] is always involved in such an estimation problem. Details of GMM can be found in [17].

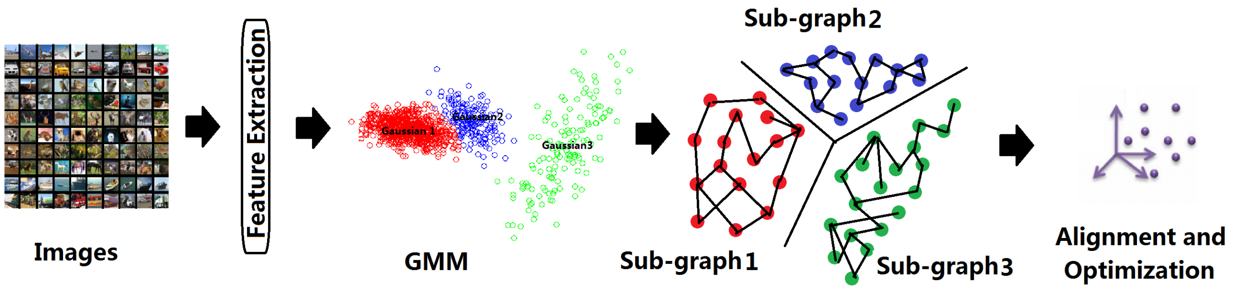

Fig. 1. The outline of DPSA. Here three clusters are used for illustration.

After finishing GMM clustering, data are partitioned into K clusters $\{C_1, C_2, \ldots, C_k, \ldots, C_K\}$, and samples belonging to a certain cluster follow the same Gaussian distribution. To discover the intrinsic data structure, samples from different Gaussians are better to be considered separately. Thus, we first construct our partitioned sub-graphs for each cluster, respectively, instead of using all the data points.

In this paper, for each cluster, we propose to obtain the sub-graph via the ℓ^1-norm constraint for data sharing the same distribution. Specifically, the neighboring samples of a data point and the corresponding similarities (graph weights) can be simultaneously calculated by solving an ℓ^1-norm optimization problem, which has been successfully utilized for spectral clustering [15], subspace learning [18], semi-supervised learning [19], *etc.*

Given a certain data point x with noise, a natural way to reconstruct this sample with a robust estimation of sparse representation α is formulated as:

$$x = D\alpha + \zeta = \left[\begin{array}{cc} D & I \end{array}\right] \left[\begin{array}{c} \alpha \\ \zeta \end{array}\right] \tag{3}$$

where D is an over-complete dictionary, α indicates the sparse reconstruction coefficients, and ζ is the noise term. We further set B=$\left[\begin{array}{cc} D & I \end{array}\right]$ and α'=$\left[\begin{array}{c} \alpha \\ \zeta \end{array}\right]$. Then, the ℓ^1-norm minimization problem can be solved for both the reconstruction error and data noise as follows:

$$\min_{\alpha'} ||\alpha'||_1, \quad s.t. \quad x = B\alpha' \tag{4}$$

In this paper, since the sparse coefficients of the ℓ^1 construction can be used to indicate the similarities among different samples, we use the ℓ^1 sparse representation to construct our graph through part optimization. Each ℓ^1-graph, termed as partitioned sub-graph here, summarizes all the sample behavior of the corresponding cluster in sparse representation. The construction process is formally stated as the following three main stages:

1) **Input:** Data matrix includes a set of samples X_{C_k}=$[x_{C_{k1}}, x_{C_{k2}}, \ldots, x_{C_{ki}}, \ldots, x_{C_{kn}}] \in \mathbb{R}^{m\times n}$, where X_{C_k} denotes all data samples from the cluster C_k after GMM, n indicates the number of samples in C_k and $k \in K$.

2) **Robust sparse representation:** For each data point in a cluster, its robust sparse coding is achieved by solving the ℓ^1-norm optimization problem:

$$min||\alpha_i^{C_k}||_1, \quad s.t. \quad x_{C_{ki}} = B_i^{C_k}\alpha_i^{C_k} \tag{5}$$

where matrix $B_i^{C_k}$= $[x_{C_{k1}}, \ldots, x_{C_{ki-1}}, x_{C_{ki+1}}, \ldots, x_{C_{kn}}, I]$.

3) **Graph weight setting:** Denote $G_{C_k} = \{X_{C_k}, W^{C_k}\}$ as the sub-graph with the sample set X_{C_k} from the cluster C_k, and W^{C_k} as the corresponding graph weight matrix, and we set $W_{i,j}^{C_k}$=$\alpha_{i,j}^{C_k}$, if $i > j$; and $W_{i,j}^{C_k}$=$\alpha_{i,j-1}^{C_k}$, if $i > j$.

In more detail, given a data point $x_{C_{ki}} \in X_{C_k}$, the sparse graph is embedded through the ℓ^1-norm constraint among all the training samples from the same cluster. Specifically, the data sample $x_{C_{ki}}$ is represented by using all remaining data samples in C_k (*i.e.*, $X_{C_k} \backslash x_{C_{ki}}$). In real applications, we usually cannot estimate the existing noise in training data. Thus, constructing the ℓ^1 graph is constrained by a certain hyperparameter ε, which indicates the maximum value of the reconstruction error in the sparse representation. Then Eq. 6 can be rewritten as follows:

$$min||\alpha_i^{C_k}||_1, \quad s.t. \quad ||x_{C_{ki}} - b_i^{C_k}\alpha_i^{C_k}||_2 < \varepsilon \tag{6}$$

where $b_i^{C_k}$=$[x_{C_{k1}}, \ldots, x_{C_{ki-1}}, x_{C_{ki+1}}, \ldots, x_{C_{kn}}]$ and ε is always an extremely small value.

This ℓ^1 learning technique[1] makes the sparse graph more discriminative and robust to represent the relationship between two data samples from the same cluster, since it preserves the same-distribution correlation affinity, meanwhile discarding the different-distribution correlation after embedding. Furthermore, this kind of partitioned sub-graph can better reflect the data relations with the locality information for optimization. Particularly, such ℓ^1 embedding can effectively avoid influence from noisy data distributions and lead to a more precise final classification.

B. Global Optimization

We repetitively compute ℓ^1 sub-graphs for each cluster via the part optimization procedure. In this subsection, these partitioned graphs will be first unified as a whole: G_{whole}=$\{G_{C_1}, G_{C_2} \ldots, G_{C_k}, , \ldots, G_{C_K}\}$ and the correspond-

[1] ℓ^1-norm optimization toolbox is available at: http://sparselab.stanford.edu

ing weight matrix $W_{whole} \in \mathbb{R}^{N \times N}$,

$$W_{whole} = \begin{bmatrix} W^{C_1} & & & & \\ & \ddots & & & \\ & & W^{C_k} & & \\ & & & \ddots & \\ & & & & W^{C_K} \end{bmatrix} \quad (7)$$

G_{C_k} denotes the graph optimized from the cluster C_k. Since the unified graph G_{whole} is directed, W_{whole} will always be asymmetric. To satisfy the linear reduction framework mentioned in Eq. 1, we symmetrize W_{whole} by setting the matrix as $W = \frac{1}{2}(W_{whole} + W_{whole}^T)$. Then, after some simple algebraic formulations, we can transform Eq. 1 to:

$$argmin \sum_{i,j=1}^{N} ||y_i - y_j||_2^2 W_{i,j} = 2YLY^T \quad (8)$$

where, $L = D - W$ is the graph Laplacian [2], and D is a diagonal matrix whose entries are column (or row, since W is symmetric) sums of W, $D_{i,i} = \sum_j W_{j,i}$. It is easy to observe that L is a symmetric and semi-positive definite matrix. Finally, the minimization problem of Eq. 8 is reduced to a quadratically-constrained quadratic program:

$$\textbf{min} \ \frac{YLY^T}{YY^T}, \quad s.t. \ \ YY^T = I \quad (9)$$

where, the constraint $YY^T = I$ requires the projected data in the low-dimensional space to be uncorrelated.

Eq. 9 is a formulation of Rayleigh quotient. Thus, the optimal Y can be obtained by solving the minimum eigenvalue eigenvector problem:

$$YL = \lambda Y \quad (10)$$

However, the graph embedding approach described above only provides the mappings for the graph vertices in the training set. For classification purposes, a mapping for all samples, including new test samples, is required. If we can find a projection $U \in \mathbb{R}^{m \times d}$, we have $Y = U^T X$. Eq. 9 can be rewritten as:

$$\textbf{min} \ \frac{U^T XLX^T U}{U^T XX^T U}, \quad s.t. \ \ U^T XX^T U = I \quad (11)$$

The optimal projection U can be obtained by solving the minimum generalized eigenvalue eigenvector decomposition problem:

$$U^T XLX^T = \lambda U^T XX^T \quad (12)$$

Let the column vectors $U = \{U_0, \ldots, U_{d-1}\}$ be the solutions of Eq. 12, ordered according to their eigenvalues, $\lambda_0 \leq \ldots \leq \lambda_{d-1}$, from the smallest one to the $(d-1)$th smallest one. Therefore, $y_i \in Y$ is a d-dimensional vector after our DPSA reduction, and U is an $m \times d$ linear projection matrix.

C. Complexity Analysis

The computation of DPSA involves three steps: **1)** Data grouping into K clusters obtained by GMM; **2)** Partitioned sparse graph construction for each cluster via ℓ^1 optimization; **3)** Combining all partitioned sparse graphs and obtaining the final projection by solving the eigenvalue eigenvector problem.

Actually, the main computational cost lies in the first two phases. In the GMM phase (driven by the EM algorithm), it requires $O(KNT)$, where T is the number of iterations until convergence through EM. For the second phase, assuming each cluster after GMM has n samples, the complexity of graph construction for all the clusters is $O(Kn^2)$. Besides, the general complexity for the eigenvalue eigenvector problem is $O(N^3)$ in the last step. Thus, the total computational cost of DPSA is approximately $O(KNT) + O(Kn^2) + O(N^3)$.

IV. Experiments and Results

In this section, we systematically evaluate the proposed DPSA on different datasets, in comparison to other popular dimension reduction algorithms.

A. Datasets

Three datasets are used to evaluate our DPSA algorithm, including handwritten digit images, face images and object images. The details of the three datasets are as follows:

The **USPS** handwritten digit database is described in [20]. A popular subset 3 contains 9298 16×16 handwritten digit images belonging to 10 classes in total, which is then split into 7291 training images and 2007 test images.

The **CMU PIE** face dataset contains $41,368$ images from 68 subjects (people). Following [21], we select 11554 front face images, which are manually aligned and cropped into 32×32 pixels. Further, $7,500$ images are used as the training set and the remaining $4,054$ images are used for testing.

The **CIFAR-10** dataset is a labeled subset of the 80-million tiny images collection [22]. It consists of a total of 60000 32×32 color images in 10 classes. The entire dataset is partitioned into two parts: a training set with 50000 samples and a test set with 10000 samples and then we use a 384-d Gist descriptor to represent each image.

B. Results

We show the results of our DPSA algorithm on the three datasets compared with other state-of-the-art dimensionality reduction methods including PCA, Fisherface, LDA, LPP, NPE and SDA. Here, the Fisherface indicates the Fisher discriminative analysis, while LDA denotes the method of PCA+Fisherface.

Since our DPSA is a linear unsupervised reduction method, the methods we compare with are all linear unsupervised (or semi-supervised) except for Fisherface and LDA. For SDA, only a quite small number of labeled samples, with the rest of data unlabeled, are used to train the final projection. We compute the best recognition results for each method via the same linear SVM classifier. Table I lists the top recognition accuracies of the seven methods and their corresponding

TABLE I
THE TOP RECOGNITION PERFORMANCE (%) ON THE USPS DATASET, CMU PIE DATASET AND CIFAR-10 DATASET, RESPECTIVELY.

Dataset \ Methods	Original feature	PCA	Fisherface	LDA	LPP	NPE	SDA	DPSA
USPS	93.04 (256)	93.16 (51)	89.74 (9)	95.28 (98,9)	94.79 (47)	94.67 (56)	95.24 (5,10)	**96.43 (30,39)**
CMU PIE	96.30 (1024)	96.42 (98)	96.65 (67)	97.88 (96,67)	97.04 (82)	97.47 (79)	97.82 (5,68)	**98.26 (28,54)**
CIFAR-10	82.26 (384)	83.21 (95)	82.22 (9)	84.98 (98,9)	83.71 (73)	83.88 (68)	84.74 (5,10)	**86.19 (120,60)**

Note that the numbers in parentheses are the corresponding feature dimensions with the best results after dimensionality reduction. For LDA, the first number is the percentage of energy retained in the PCA step, and the second number is the length of the final vector via Fisherface. For SDA, the first number is the K, which means the percentage (we fix it to a very small number, *i.e.*, 5%) of the labeled data used in the training phase, and the second number is the final feature length. For DPSA, the first number is the number of clusters by GMM and the second number is the reduced feature length.

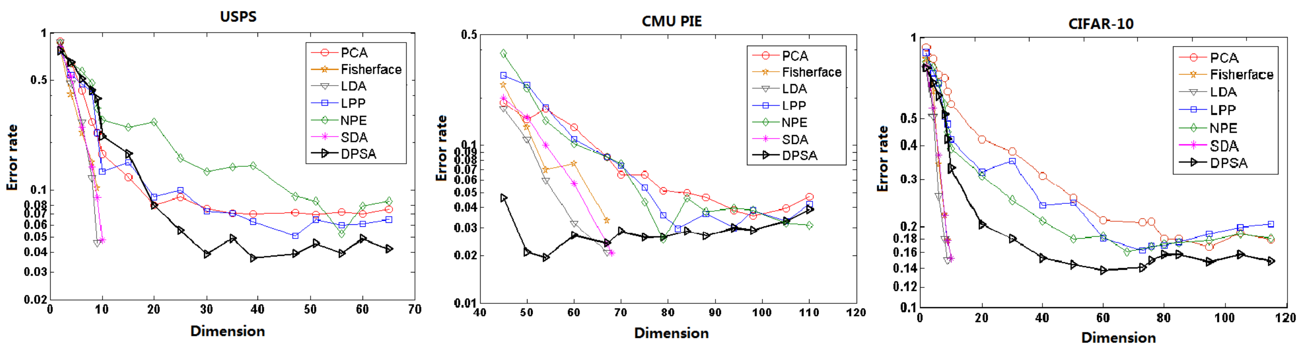

Fig. 2. The recognition error rate vs. number of dimensions on three datasets.

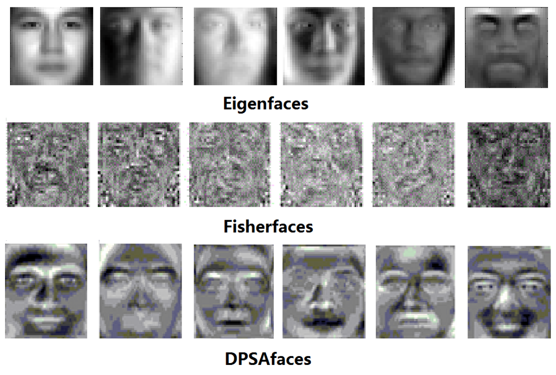

Fig. 3. The first 6 basis vectors of Eigenfaces, Fisherfaces, and DPSAfaces calculated from the face images in the CMU PIE dataset.

numbers of dimensions on the USPS, CMU and CIFAR-10 datasets, respectively. In addition, Fig. 2 also plots the corresponding curves of the recognition error rates of the seven comparable methods vs. the numbers of the projected dimensions.

In terms of the classification accuracy, our unsupervised DPSA approach consistently outperforms all the unsupervised methods, *i.e.*, PCA, LPP, NPE, and supervised (semi-supervised) methods, *i.e.*, Fisherface, LDA and SDA, on all three datasets. From Table I, for both USPS and CMU PIE datasets, DPSA achieves 1.15% and 0.38% higher than the LDA, which gives the second best performance, and 1.64% and 0.79% higher than the best unsupervised methods on these two datasets, respectively. For the larger and more complex CIFAR-10 dataset, DPSA also reaches 1.21% higher than the best supervised method and 2.31% higher than the best unsupervised one. Regarding reduction effects, DPSA achieves lower dimensional representations on all three datasets, compared with other unsupervised techniques in Table I.

This is because that PCA produces a set of linearly uncorrelated principal component as the low-dimension projections, which only maximizes the variance of data features, but misses their intrinsic data structures in the original feature space. For LPP, the graph Laplacian indeed helps to keep the data locality structure in high dimension, and tries to preserve the same structure in the low-dimensional space as well. The locality information in LPP is always manually constructed via a neighborhood constraint, such that, if the two data points' pairwise distance exceeds a certain threshold, the value of graph Laplacian will be set as zero. However, this kind of construction is sensitive to data noise and one noisy feature may dramatically change the data's relationship. Furthermore, when data's distribution is not even, the weight matrix based on the pairwise-distance may also involve the far-distance inhomogeneous data together, if the threshold is large. The same drawbacks also exist in Neighborhood Perceiving Embedding (NPE) and Semi-supervised Discriminative Embedding (SDA).

For those supervised methods (*i.e.*, Fisherface and LDA), in our experiments, they involve the label information in their training phase and build an objective function to maximize the inter-class variation and minimize the intra-class variation. The advantage of Fisherface and LDA is that they can project the data into a very low-dimensional space with $C-1$, where C is the number of classes in the training data. However,

TABLE II
COMPUTATIONAL COSTS FOR DPSA WITH THE HIGHEST CLASSIFICATION ACCURACIES.

Datasets \ Computational time	Learning time	Coding time	Classification time
USPS (96.33%)	207 seconds (7291 training samples)	0.0032 seconds/sample	0.35 seconds/sample
CMU PIE (98.06%)	245 seconds (7500 training samples)	0.0048 seconds/sample	0.48 seconds/sample
CIFAR-10 (86.19%)	1521 seconds (50000 training samples)	0.0035 seconds/sample	0.83 seconds/sample

the variation of the values of data from the same class is impaired and ignored in the reduced space. For instance, two far-away data points from the same class may be very close after projection, which would easily lead to over-fitting on testing data.

In contrast to all mentioned above, our DPSA treats the data samples on different distributions separately and focuses more on the natural relationship among samples, instead of manually setting a threshold to break the intrinsic data properties. Therefore, our DPSA is first clustered via GMM to partition all data into several groups, each of which shares the same Gaussian distribution. Then an ℓ^1 sub-graph is constructed to preserve the data locality structure in each cluster and finally all the sub-graphs are merged to compute the projection. DPSA shows its discriminative advantages for subspace learning as follows: (1) great robustness to data noise, (2) automatic sparsity instead of manual setting, and (3) adaptive neighborhood for each individual data point. Fig. 3 also visualizes the DPSA projection vectors as feature images on the CMU PIE face dataset, together with Eigenfaces and Fisherfaces.

Furthermore, a brief comparison of computational complexity is shown in Table II. The results show that DPSA always needs a few hundred seconds for learning the projection. The learning speed highly depends on the size of the training set. Once the projection is obtained, it is very fast for the DPSA to code a new sample and classify it (always with the total time less than 0.9 seconds/sample in the Matlab environment).

V. CONCLUSION

In this paper, we have presented a new unsupervised linear subspace learning approach, named Discriminative Partition Sparsity Analysis (DPSA). DPSA explicitly considers different distributions that exist in data points and also keeps the natural locality relationship among the data on each sub-distribution. Specifically, we introduced the Gaussian mixture model(GMM) clustering and sparsity optimization for dimensionality reduction tasks, in which each data point can be embedded via the ℓ^1-constraint to construct the graph and then the final projection is computed by solving the eigenvalue eigenvector problem .

We have systematically evaluated our method on the USPS, CMU PIE and CIFAR-10 datasets and produced the image classification accuracies of 96.43%, 98.26% and 86.19%, respectively. In all three datasets, our DPSA achieves better results compared with other popular supervised and unsupervised methods.

REFERENCES

[1] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[2] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering." in *Advances in Neural Information Processing Systems*, vol. 14, 2001, pp. 585–591.

[3] L. Shao, L. Liu, and X. Li, "Feature learning for image classification via multiobjective genetic programming," *IEEE Transactions on Neural Networks and Learning Systems*, 2014.

[4] F. Zhu and L. Shao, "Weakly-supervised cross-domain dictionary learning for visual recognition," *International Journal of Computer Vision*, pp. 1–18, 2014.

[5] L. Liu, L. Shao, X. Zhen, and X. Li, "Learning discriminative key poses for action recognition." *IEEE Transactions on Cybernetics*, 2013.

[6] V. Gaede and O. Günther, "Multidimensional access methods," *ACM Computing Surveys (CSUR)*, vol. 30, no. 2, pp. 170–231, 1998.

[7] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1, pp. 37–52, 1987.

[8] A. M. Martínez and A. C. Kak, "Pca versus lda," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228–233, 2001.

[9] S. Chakrabarti, S. Roy, and M. V. Soundalgekar, "Fast and accurate text classification via multiple linear discriminant projections," *The VLDB Journal*, vol. 12, no. 2, pp. 170–185, 2003.

[10] K. Fukunaga, *Introduction to statistical pattern recognition*, 1990.

[11] X. He and P. Niyogi, "Locality preserving projections," in *Neural Information Processing Systems*, vol. 16, 2003, p. 153.

[12] D. Cai, X. He, and J. Han, "Semi-supervised discriminant analysis," in *IEEE International Conference on Computer Vision*, 2007, pp. 1–7.

[13] X. He, D. Cai, S. Yan, and H.-J. Zhang, "Neighborhood preserving embedding," in *IEEE International Conference on Computer Vision*, vol. 2, 2005, pp. 1208–1213.

[14] G. McLachlan and D. Peel, *Finite mixture models*, 2004.

[15] B. Cheng, J. Yang, S. Yan, Y. Fu, and T. Huang, "Learning with ℓ^1-graph for image analysis," *IEEE Transactions on Image Processing*, pp. 858–866, 2010.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

[17] J. A. Bilmes *et al.*, "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International Computer Science Institute*, vol. 4, no. 510, p. 126.

[18] L. Zhang, P. Zhu, Q. Hu, and D. Zhang, "A linear subspace learning approach via sparse coding," in *IEEE International Conference on Computer Vision*, 2011, pp. 755–761.

[19] S. Yan and H. Wang, "Semi-supervised learning by sparse representation," in *SIAM International Conference on Data Mining*, 2009, pp. 792–801.

[20] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[21] D. Cai, X. He, and J. Han, "Speed up kernel discriminant analysis," *VLDB*, vol. 20, no. 1, pp. 21–33, 2011.

[22] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1958–1970, 2008.