Structured AutoEncoders for Subspace Clustering

Xi Peng, *Member, IEEE*, Jiashi Feng, Shijie Xiao, Wei-Yun Yau, Joey Tianyi Zhou, and Songfan Yang

***Abstract*— Existing subspace clustering methods typically employ shallow models to estimate underlying subspaces of unlabeled data points and cluster them into corresponding groups. However, due to the limited representative capacity of the employed shallow models, those methods may fail in handling realistic data without the linear subspace structure. To address this issue, we propose a novel subspace clustering approach by introducing a new deep model—Structured AutoEncoder (StructAE). The StructAE learns a set of explicit transformations to progressively map input data points into nonlinear latent spaces while preserving the local and global subspace structure. In particular, to preserve local structure, the StructAE learns representations for each data point by minimizing reconstruction error with respect to itself. To preserve global structure, the StructAE incorporates *a prior* structured information by encouraging the learned representation to preserve specified reconstruction patterns over the entire data set. To the best of our knowledge, StructAE is one of the first deep subspace clustering approaches. Extensive experiments show that the proposed StructAE significantly outperforms 15 state-of-the-art subspace clustering approaches in terms of five evaluation metrics.**

***Index Terms*— Unsupervised deep learning, locality preservation, globality preservation, spectral clustering.**

I. Introduction

SUBSPACE clustering aims at seeking a collection of implicit subspaces to fit a given unlabeled data set and segmenting them into different groups [1]. During past years, spectral clustering based approaches [2]–[4] have achieved remarkable performance, which are conducted in two steps: 1) building an affinity matrix (*i.e.*, similarity graph) $\mathbf{C}$ to describe the affinity of data points, where $\mathbf{C}_{ij}$ quantizes the closeness between data points $\mathbf{x}_i$ and $\mathbf{x}_j$; 2) clustering data by grouping the eigenvectors of $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$. Here, $\mathbf{L}$ is termed as graph Laplacian. The diagonal matrix $\mathbf{D}$ is with the entry $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$ and $\mathbf{A} = |\mathbf{C}| + |\mathbf{C}^T|$. In general, the performance of these approaches critically depends on the quality of $\mathbf{A}$.

Although those subspace clustering approaches [5]–[21] have achieved encouraging performance, we found that they may suffer from some limitations as follows. First, most of existing algorithms strive to build the affinity matrix $\mathbf{A}$ but ignore obtaining a good representation using $\mathbf{A}$. Actually, almost all of them directly use top eigenvectors (*w.r.t.* largest eigenvalues) of $\mathbf{L}$ as the low-dimensional data representation. In essence, such a method is exactly Laplacian Eigenmap (LE) [22]. In other words, those approaches achieve subspace clustering through 1) building an affinity matrix using the self-expression, 2) getting low dimensional representations of $\mathbf{X}$ by performing LE on the affinity matrix, and 3) obtaining clustering membership by conducting the k-means algorithm on the representations. Until now, it is still an open and challenging issue whether there is a better way to embed the affinity matrix into low-dimensional spaces and discover the clustering membership therein. The second limitation can attribute to the linearity assumption adopted by those methods. The assumption leads to failure in handling data that cannot be linearly reconstructed. To overcome this disadvantage, some kernel methods have been proposed, *e.g.*, kernel low rank representation (KLRR) [23] and kernel sparse subspace clustering (KSSC) [24]. They obtain results by first mapping the input into a pre-specified kernel space and then performing subspace clustering therein. One disadvantage of these kernel subspace clustering methods is that their performance heavily depends on the used kernel function. In practice, however, it remains unclear to choose the appropriate kernel function. The third limitation is that those methods will suffer from out-of-sample and large scale clustering issues because they have to perform general eigen-decomposition over $\mathbf{L}$ of the whole data set. To solve these issue, out-of-sample and large scale extensions are required [19], usually at the cost of clustering quality.

From the above, ones could conclude that data representation plays an important role in subspace clustering. Motivated by the huge success achieved by deep representation learning [25]–[29], we propose a new deep subspace clustering method, namely, Structured AutoEncoder (StructAE) to overcome the aforementioned limitations. The proposed method achieves subspace clustering through calculating the structured reconstruction relation from raw data, training a neural network with the formulation of self-supervision based locality and self-expression based globality, and grouping the compact representation given by the propose method.

Manuscript received October 17, 2017; revised March 31, 2018 and June 3, 2018; accepted June 11, 2018. Date of publication June 18, 2018; date of current version July 16, 2018. The work of X. Peng was supported by the Fundamental Research Funds for the Central Universities under Grant YJ201748, by NFSC under Grant 61432012 and Grant U1435213, and by the Fund of Sichuan University—Tomorrow Advancing Life. The work of J. Feng was supported by NUS startup R-263-000-C08-133, MOE Tier-I R-263-000-C21-112, NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133, and MOE Tier-II R-263-000-D17-112. The work of W.-Y. Yau was supported by Singapore's RIE2020 AME-Programmatic Grant A1687b0033. The work of S. Yang was supported by NFSC under Grant 61501312. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Vishal Moga. *(Corresponding author: Joey Tianyi Zhou.)*

X. Peng is with the College of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: pangsaai@gmail.com).

J. Feng is with the Department of ECE, National University of Singapore, Singapore 119077 (e-mail: elefjia@nus.edu.sg).

S. Xiao is with 3OmniVision Technologies Singapore Pte. Ltd., Singapore 609935 (e-mail: xiaoshijiehithonor@gmail.com).

W.-Y. Yau is with the Institute for Infocomm Research, A*STAR, Singapore 138632 (e-mail: wyyau@i2r.a-star.edu.sg).

J. T. Zhou is with the Institute of High Performance Computing, A*STAR, Singapore 138632 (e-mail: joey.tianyi.zhou@gmail.com).

S. Yang is with the AI Lab, TAL Education Group, College of Electronics and Information Engineering, Sichuan University, Chengdu 610065, China (e-mail: yangsongfan@100tal.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2018.2848470

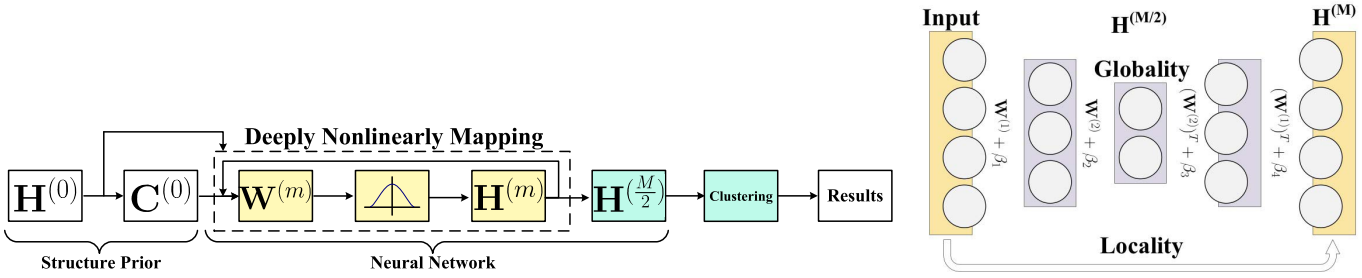


Fig. 1. The architectures of subspace clustering approaches and StructAE. In (a), $\mathbf{H}^{(0)}$ denotes inputs, $\mathbf{C}^{(0)}$ is the structure prior calculated based on $\mathbf{H}^{(0)}$ and $\mathbf{H}^{(m)}$ corresponds to the output of the m-th layer of our neural network, where $m = 1, 2, \cdots, M$. (b) a symbolic illustration, where the locality denotes the minimization of the reconstruction error between inputs and $\mathbf{H}^{(M)}$, and the globality is achieved by minimizing the reconstruction error of $\mathbf{H}^{\frac{M}{2}}$ over the whole data set with $\mathbf{C}^{(0)}$.

Fig. 1 illustrates the basic idea of StructAE, which shows that StructAE significantly differs from existing subspace clustering methods. More specifically, 1) StructAE directly obtains data representation of $\mathbf{X}$ rather than relying on the affinity matrix $\mathbf{A}$; 2) StructAE is a multi-layer nonlinear model which gives a stronger ability to capture the nonlinearity of data; 3) Comparing with the stacked AutoEncoder (SAE), StructAE not only incorporates the locality for reconstructing each data point, but also considers the structured globality *w.r.t.* the entire data set; 4) Comparing with kernel-based approaches, StructAE offers explicit transformations and a better scalability because it does not load the entire data set into the memory; 5) StructAE is compatible with both the k-means algorithm and most of existing subspace clustering approaches. When performing k-means on the representation learned by StructAE, we theoretically show that under some mild conditions, StructAE performs like the spectral clustering. When combined with subspace clustering, StructAE can be regarded as a latent subspace clustering method like [30].

The paper is a substantial extension of our conference work [31] with further improvements given below. First, we design a new algorithm by incorporating ℓ_2-norm based structure prior besides ℓ_1-norm based sparsity. Second, we present theoretical analysis to show the connections of our StructAE with some existing methods including spectral clustering [3], sparse subspace clustering (SSC) [7], low rank representation (LRR) [8], and their variants. Third, the experimental evaluations are totally different, which involve new baselines and new data sets. More specifically, in the conference version, YaleB and COIL20 with DSIFT and HOG features are used for comparisons. In contrast, this paper carries out experiments using YaleB, COIL20, and mnist with SDSIFT and LPQ features, as well as CIFAR10 with these four features and BF0502 raw data. Fourth, this paper presents the 2D visualization to show that our method could learn a distinct and compact representation which further boost the clustering performance. Fifth, besides the depth, we also investigate the influence of width and parameters of our method. Sixth, we evaluate the time cost of our method for training and inference for extensive investigations.

Notations: **Lower-case bold letters** denote column vectors and **UPPER-CASE BOLD ONES** denote matrices, unless otherwise stated. $\mathbf{A}^T$ and $\mathbf{A}_{ij}$ denote the transpose and an element of the matrix $\mathbf{A}$, and $\mathbf{I}$ denotes the identity matrix.

II. Related Work

In recent years, many subspace clustering works [5]–[14], [16], [23], [24], [30], [32], [33] have devoted to building a high quality affinity matrix by using self-expression of the inputs $\mathbf{X} \in \mathbb{R}^{d\times n}$. More specifically, these algorithms represent $\mathbf{X}$ as its linear combination by:

$$\min_{\mathbf{C}} \frac{1}{2}\|\mathbf{X} - \mathbf{X}\mathbf{C}\|_F^2 + \lambda\mathcal{R}(\mathbf{C}), \tag{1}$$

where d is the dimension of $\mathbf{X}$, n is the number of data points, $\|\cdot\|_F$ denotes the Frobenius norm, $\mathbf{C} \in \mathbb{R}^{n\times n}$ denotes the self-expression of $\mathbf{X}$, and $\mathcal{R}(\mathbf{C})$ denotes a desirable *prior* on $\mathbf{C}$. Those methods differ from each other in the choice of $\mathcal{R}(\mathbf{C})$. For example, LRR [8], SSC [7], and least square regression (LSR) [6], [10], [34] assume that $\mathbf{C}$ is low rank, sparse, and dense, respectively. Accordingly, nuclear-, ℓ_1-, and Frobenius-norm are formulated into $\mathcal{R}(\cdot)$. Once getting $\mathbf{C}$, the affinity matrix is obtained via $\mathbf{A} = |\mathbf{C}| + |\mathbf{C}^T|$ and clustering membership is achieved by applying spectral clustering [3] over $\mathbf{A}$.

To further boost the performance of subspace clustering, we propose a deep neural network based method. Our work is complementary to existing works in deep learning and subspace clustering, since it incorporates the merits (*i.e.* structure prior) of subspace clustering into deep neural network. Thus, it is well expected that StructAE can obtain satisfactory results in clustering unlabeled data. Unlike existing subspace clustering approaches, the proposed StructAE learns a set of nonlinear transformations to obtain the low-dimensional representation using a neural network instead of the manifold learning (*e.g.*, LE). Benefiting from stronger nonlinearity and representational capacity of deep neural networks, our method could achieve a better clustering performance. Comparing with existing deep learning methods [35]–[37], the proposed StructAE is a *novel* neural network for subspace clustering by preserving both globality and locality of the data set. To be exact, StructAE guarantees the globality by minimizing the reconstruction error of embedding structure prior, and the locality by minimizing the reconstruction error between each

data point and the reconstruction given by our neural network. One of pioneer works in deep subspace clustering is [38] which is remarkably different from our method. In brief, our method uses the structured globality including affinity (e.g., sparsity) as a prior to learn a compact representation, whereas [38] learns the sparsity in the latent space.

III. Structured AutoEncoders for Subspace Clustering

StructAE consists of following three steps: first calculating the structure prior from inputs, then training a neural network to project the input into another space, and finally clustering the representation into multiple subspaces. In this section, we will first elaborate on the detail of StructAE for these three steps and then introduce how to optimize the StructAE model. Moreover, we also give some theoretical analysis to show the connections between StructAE and existing methods.

A. The Deep Model of StructAE

StructAE employs a neural network ($M+1$ layers) to perform M nonlinear transformations, where the first layer corresponds to input, the first $M/2$ hidden layers perform as an encoder to learn a compact representation and the last $M/2$ layers perform as a decoder to reconstruct the input $\mathbf{X}$. For ease of presentation, some of the used notations are defined given below. We use $\mathbf{h}_i^{(0)} = \mathbf{x}_i \in \mathbb{R}^d$ to denote a data point and

$$\mathbf{h}_i^{(m)} = g(\mathbf{W}^{(m)}\mathbf{h}_i^{(m-1)} + \mathbf{b}^{(m)}) \in \mathbb{R}^{d_m} \tag{2}$$

to denote the output of the m-th layer. Here, $m = 1, 2, \cdots, M$ is the index of layer, d_m is the number of neurons of the corresponding layer, and $g(\cdot)$ denotes the used activation function. For a given input $\mathbf{x}_i$, the corresponding reconstruction and low-dimensional representation are $\mathbf{h}_i^{(M)}$ and $\mathbf{h}_i^{(\frac{M}{2})}$, respectively. Moreover, for the data set $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$, the corresponding reconstructions are:

$$\mathbf{H}^{(M)} = [\mathbf{h}_1^{(M)}, \mathbf{h}_2^{(M)}, \cdots, \mathbf{h}_n^{(M)}]. \tag{3}$$

StructAE aims to simultaneously preserve self-supervision based locality $\mathcal{J}_1$ (*i.e.* data reconstruction) and self-expression based globality $\mathcal{J}_2$ (*i.e.* the global structure prior $\mathbf{C}$.[1]) in learning representation. With the aforementioned definitions, we propose the following objective function:

$$\min_{\mathbf{W}^{(m)},\mathbf{b}^{(m)}} \underbrace{\frac{1}{2}\|\mathbf{X} - \mathbf{H}^{(M)}\|_F^2}_{\mathcal{J}_1,\ \text{Locality}} + \underbrace{\frac{\lambda_1}{2}\|\mathbf{H}^{(\frac{M}{2})} - \mathbf{H}^{(\frac{M}{2})}\mathbf{C}\|_F^2}_{\mathcal{J}_2,\ \text{Globality}} + \underbrace{\frac{\lambda_2}{2}\sum_{m=1}^{M}\left(\|\mathbf{W}^{(m)}\|_F^2 + \|\mathbf{b}^{(m)}\|_2^2\right)}_{\mathcal{J}_3,\ \text{Regularization}}, \tag{4}$$

where λ_1 and λ_2 are positive tradeoff parameters.

We design the terms $\{\mathcal{J}_i\}_{i=1}^3$ for different goals. To be exact, the first term $\mathcal{J}_1$ aims to keep the locality by minimizing the errors between the input $\mathbf{X}$ and the reconstruction $\mathbf{H}^{(M)}$. Clearly, each data point actually performs as its supervisor to learn a compact representation $\mathbf{H}^{(\frac{M}{2})}$. $\mathcal{J}_2$ is derived from the manifold learning [39] which assumes that some properties on manifold are invariant to different projection spaces.[2] Based on the assumption, ones could learn representation by preserving the specific property from input space into another space. Clearly, the key of manifold learning is to seek the invariance. In this paper, the reconstruction relationship (*i.e.*, the structure prior) is regarded as a type of invariance, which has shown effectiveness in [7], [8], and [40]. In particular, we adopt two popular cases, *i.e.,* the sparsity prior by

$$\begin{aligned}\min_{\mathbf{C}}\ & \sum_{i=1}^{n}\|\mathbf{x}_i - \mathbf{Y}\mathbf{c}_i\|_2^2 + \lambda\|\mathbf{c}_i\|_1 \\ \text{s.t. }& \mathbf{c}_{ii} = 0,\end{aligned} \tag{5}$$

and the non-sparse prior by

$$\begin{aligned}\min_{\mathbf{C}}\ & \sum_{i=1}^{n}\|\mathbf{x}_i - \mathbf{Y}\mathbf{c}_i\|_2^2 + \lambda\|\mathbf{c}_i\|_2 \\ \text{s.t. }& \mathbf{c}_{ii} = 0,\end{aligned} \tag{6}$$

where $\|\cdot\|_1$ and $\|\cdot\|_2$ denote ℓ_1- and ℓ_2-norm, and the corresponding StructAE is denoted by StructAE-L1 and StructAE-L2, respectively. $\mathbf{c}_{ii}$ denotes to the i-th element of the vector $\mathbf{c}_i$, and the constraint is used to avoid degenerated solutions. Note that the ℓ_2-norm based reconstruction coefficient is provable to give low-rank property [40], and it is more computationally efficient than the nuclear-norm based objective function. In Eq.(6), $\mathbf{Y}$ denotes a dictionary, which can be pre-determined using dictionary learning methods or simply specified to be the data set itself. For simplicity, we directly let $\mathbf{Y} = \mathbf{X}$ in this paper.

With the optimal solution of Eq. (5) or Eq. (6), $\mathcal{J}_2$ guarantees the globality since the reconstruction relation of using the entire data set is kept from input space into the hidden representation. $\mathcal{J}_3$ is a popular regularization, which is used to avoid over-fitting. It should be pointed out that, the objective function Eq. (4) with the nonlinear function will not give the trivial solution $\mathbf{W}^{(m)} = \mathbf{0}$ and $\mathbf{W}^m = \mathbf{I}$ thanks to the existence of $\mathcal{J}_1$ and the updating rule Eq. (10)–(13), where $\mathbf{0}$ and $\mathbf{I}$ denote an all-zero matrix and identity matrix, respectively. Clearly, either all-zero matrix or identity matrix will not give the minimal $\mathcal{J}_1$.

StructAE enforces the learned representation to keep the global structure prior and simultaneously uses the input as the supervisor to learn a compact data representation. These global and local structures are integrated into the data representation, and thus resulting in favorable clustering results.

B. Optimization

In this subsection, we show how efficiently optimize our StructAE using the stochastic sub-gradient descent algorithm (SGD). For convenience, Eq. (4) is rewritten in the following

[1] In this paper, we mainly investigate ℓ_1-norm based sparsity and ℓ_1-norm based self-expression prior as calculated in Eq. (5)–(6).

[2] Note that, it is also feasible and quite easy to formulate the graph Laplacian into $\mathcal{J}_2$ to utilize the local consistency based on pairwise distance.

sample-wise form:

$$\mathcal{J} = \frac{1}{2}\sum_{i=1}^{n}\left(\|\mathbf{x}_i - \mathbf{h}_i^{(M)}\|_2^2 + \lambda_1\|\mathbf{h}_i^{(\frac{M}{2})} - \mathbf{H}^{(\frac{M}{2})}\mathbf{c}_i\|_2^2\right) + \frac{\lambda_2}{2}\sum_{m=1}^{M}\left(\|\mathbf{W}^{(m)}\|_F^2 + \|\mathbf{b}^{(m)}\|_2^2\right). \quad (7)$$

Recall the definition of $\mathbf{h}_i^{(m)}$ in Eq. (2) and use the back-propagation algorithm, the sub-gradients of Eq. (7) *w.r.t.* $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ is as follows:

$$\frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(m)}} = \left(\Delta^{(m)} + \lambda_1\Lambda^{(m)}\right)(\mathbf{h}_i^{(m-1)})^T + \lambda_2\mathbf{W}^{(m)} \quad (8)$$

$$\frac{\partial \mathcal{J}}{\partial \mathbf{b}^{(m)}} = \Delta^{(m)} + \lambda_1\Lambda^{(m)} + \lambda_2\mathbf{b}^{(m)}, \quad (9)$$

where $\Delta^{(m)}$ is given by:

$$\begin{cases} -\left(\mathbf{x}_i - \mathbf{h}_i^{(M)}\right) \odot g'(\mathbf{z}_i^{(M)}), & m = M \\ (\mathbf{W}^{(m+1)})^T\Delta^{(m+1)} \odot g'(\mathbf{z}_i^{(m)}), & \text{otherwise} \end{cases} \quad (10)$$

and $\Lambda^{(m)}$ is defined by

$$\begin{cases} (\mathbf{W}^{m+1})^T\Lambda^{(m+1)} \odot g'(\mathbf{z}_i^{(m)}), & m = 1, \cdots, \frac{M-2}{2} \\ \left(\mathbf{h}_i^{(\frac{M}{2})} - \mathbf{H}^{(\frac{M}{2})}\mathbf{c}_i\right) \odot g'(\mathbf{z}_i^{(\frac{M}{2})}), & m = \frac{M}{2} \\ \mathbf{0}, & m = \frac{M+2}{2}, \cdots, M \end{cases} \quad (11)$$

Here $\odot$ is the element-wise multiplication, $g'(\cdot)$ denotes the derivative of the activation $g(\cdot)$, $\mathbf{h}_i^{(0)} = \mathbf{x}_i$, and $\mathbf{z}_i^{(m)} = \mathbf{W}^{(m)}\mathbf{h}_i^{(m-1)} + \mathbf{b}^{(M)}$.

Using the SGD algorithm, $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^M$ are updated as follows:

$$\mathbf{W}^{(m)} = \mathbf{W}^{(m)} - \mu\frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(m)}}, \quad (12)$$

$$\mathbf{b}^{(m)} = \mathbf{b}^{(m)} - \mu\frac{\partial \mathcal{J}}{\partial \mathbf{b}^{(m)}}. \quad (13)$$

Algorithm 1 summarizes the optimization procedure of our method.

Algorithm 1 Structured AutoEncoders for Subspace Clustering

Input: A input data $\mathbf{X}$, and the parameters λ_1, λ_2.
// Initialization:
Let $\mathbf{H}^0 = \mathbf{X}$ and initialize our neural network $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^M$.
// Compute the structure prior:
Obtain the structured prior $\mathbf{C}$ of $\mathbf{X}$ by solving Eq.(5) or Eq.(6).
for $m = 1, 2 \cdots, M$ **do**
  Get $\{\mathbf{H}^{(m)}\}_{m=1}^M$ via Eq.(2).
// Optimizations:
while *not converge* **do**
  for $i = 1, 2, \cdots, n$ **do**
    Randomly select a data point $\mathbf{x}_i$ and let $\mathbf{h}_i^0 = \mathbf{x}_i$,
    // Forward propagation:
    for $m = 1, 2 \cdots, M$ **do**
      Compute $\mathbf{h}_i^{(m)}$ via Eq.(2).
    // Compute gradients:
    for $m = M, M-1 \cdots, 1$ **do**
      Compute the gradient via Eqns.(8)–(11).
    // Update neural network:
    for $m = 1, 2, \cdots, M$ **do**
      Update $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ via Eqns.(12)–(13).
// Clustering:
Obtain the clustering membership by clustering $\mathbf{H}^{(\frac{M}{2})}$.
Output: The clustering membership and the parametric neural network $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^M$.

C. Implementation Details

In our implementation, $g(\cdot) = tanh(\cdot)$ is used as the activation function. To initialize our neural network, the pre-training and fine-tuning strategy [25] is adopted. More specifically, the entire neural network is decomposed into multiple smaller ones and then lay-wise pre-training is performed on these single hidden layer network. In our experiments, for a five-layer neural network 300-200-150-200-300, we first train the shallow networks of 300-200-300 and 200-150-200, respectively. After that, we use the pretrained weights to initialize the deeper network.

It should be pointed out that the fully connected layers in StructAEs could be replaced with other neural networks including but not limited to convolutional neural networks, restricted Boltzmann machines, long short term memory AutoEncoders. With such a new module, our method may give a better performance. In our implementation, however, we adopt the fully connected layers (*i.e.*, the classic AutoEncoder) due to following reasons: 1) we experimentally found that even with such a simple neural network, our method could remarkably outperform many popular subspace clustering methods; 2) it is relatively easy to tune parameters for the classic AutoEncoder comparing with other neural networks; 3) the classic AutoEncoder is computationally efficient, which is more friendly to our hardware environment than other networks.

D. Connection to Previous Works

In this subsection, we discuss the relationship between our StructAE and previous works from two different angles. First, we show that StructAE can be treated as a variant of the classical AutoEncoder (AE). Moreover, with several simplifications, StructAE can be deemed as a deep extension of spectral clustering (SC) algorithms.

1) Connection Between StructAE and AE: The well-known AE utilizes each data point as the supervisor to guide representation learning, which ignores the relationship with other data. Our StructAE will reduce to AE if λ_1 (Eq. (4)) is fixed to 0, *i.e.* the structure prior is not incorporated into our objective function. In this sense, StructAE augments AE with formulation of the valuable relationships among different data points (*i.e.,* structure global prior) and such relationships can give encouraging performance as shown in previous subspace clustering works [7], [8].

2) Connection Between StructAE and SC: Most spectral clustering based methods obtain the segmentation of data by conducting the k-means algorithm on leading eigenvectors (*w.r.t.* largest eigenvalues) of the Laplacian matrix $\mathbf{L}$ as aforementioned. StructAE can be regarded as providing a deep framework to generalize existing SCs.

Theorem 1: Let $g(z) = z$, $\lambda_1 \to +\infty$, $M = 2$ *in Eq. (4) and with the following mild constraint for avoiding trivial solutions, then the compact representation* $\mathbf{H}^\star$ *learnt by StructAE will be the solution to*

$$\min_{\mathbf{H}} \frac{1}{2}\|\mathbf{H} - \mathbf{H}\mathbf{C}\|_F^2 \quad \text{s.t. } \mathbf{H}\mathbf{H}^T = \mathbf{I}. \tag{14}$$

Interestingly, $\mathbf{H}^*$ *is also optimal to*

$$\max_{\mathbf{H}} \frac{\mathbf{H}\mathbf{L}\mathbf{H}^T}{\mathbf{H}\mathbf{H}^T}, \tag{15}$$

which is exactly the objective of spectral clustering based methods. The only one difference is the choice of $\mathbf{L}$*. To be exact,* $\mathbf{L} = \mathbf{C} + \mathbf{C}^T - \mathbf{C}\mathbf{C}^T$ *for our method, whereas* $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ *(*$\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$*) for the spectral clustering method [3].*

Proof: The optimal solution of Eq. (14) is achieved by solving the following eigen-decomposition problem:

$$\mathbf{H}^* = \text{argmin}\frac{\mathbf{H}(\mathbf{I} - \mathbf{C} - \mathbf{C}^T + \mathbf{C}\mathbf{C}^T)\mathbf{H}^T}{\mathbf{H}\mathbf{H}^T} \tag{16}$$

which is identical to

$$\mathbf{H}^* = \text{argmax}\frac{\mathbf{H}(\mathbf{C} + \mathbf{C}^T - \mathbf{C}\mathbf{C}^T)\mathbf{H}^T}{\mathbf{H}\mathbf{H}^T}, \tag{17}$$

as desired. □

Based on Theorem 1, we can further bridge StructAE and some existing subspace clustering methods as below:

Remark 1: StructAE is a deep framework of most existing spectral clustering based subspace clustering methods if the same structure prior is adopted. With the simplifications shown in Theorem 1, for example, StructAE will degrade to SSC [7] with the sparsity prior, to LRR [8] with the low rank prior, to LSR [6] and L2graph [14] with the L2-norm based prior.

IV. Experiments

In this section, we report the performance of the proposed StructAE comparing with 15 popular subspace clustering methods regarding to five evaluation metrics.

A. Experimental Settings

1) Baseline Algorithms: We compare the proposed StructAEs with SSC [7], LRR [8], low rank based subspace clustering (LRSC) [41], LSR [6], smooth representation clustering (SMR) [42], kernel SSC (KSSC) [24], kernel LRR (KLRR) [23], latent subspace sparse subspace clustering (LS3C) [30] and stacked sparse autoencoders (SAE) [43]. Among the compared approaches, KSSC, LS3C, and KLRR have two variants that are based on the radial basis function / the polynomial function, denoted by KSSC-R/ KSSC-P, LS3C-R / LS3C-P and KLRR-R / KLRR-P, respectively. Moreover, LS3C-L denotes LS3C with the linear kernel function. LSR has also two variants which are with and without a diagonal constraint. We denote them as LSR1 and LSR2. Moreover, we examine the performance of SAEs and SAEg which are SAE with the saturating linear transfer function and the sigmoid function, respectively. All evaluated methods are implemented in MATLAB.

In our experiments, SAE and our StructAEs are five-layer neural networks which consist of 300-200-150-200-300 neurons and use $tanh$ as the activation function. This structure is introduced in [44], which has shown efficacy in face verification. Like [44], we train our network by adopting the standard SGD with the batch-size of 1 and using different learning rate for different data sets as elaborated latter. In implementations, we employ the Homotopy algorithm [45] to solve ℓ_1-minimization problem of SSC, KSSC, and StructAE (in the case of sparsity). For a fair comparison, we compare the best performance of all the tested approaches with the tuned parameters. In other words, we do not split the data into different subsets for training, validation, and testing as supervised works did. Instead, we tune parameters for all tested methods using the whole data set, as [6], [7] did. Regarding to the baselines, we seek optimal parameters by referring to the setting in the corresponding paper. Regarding to our StructAE, we fix $\lambda_2 = 10^{-3}$ in all cases and experimentally determine the value of λ_1. In details, we choose an optimal λ_1 from $\{10^{-4}, 5\times10^{-4}, 10^{-3}, 5\times10^{-3}, 10^{-2}, 5\times10^{-2}\}$. Moreover, SAE and StructAEs are regarded to achieve convergence if the their loss is less than 10^{-3} or the training epoch reaches to 100.

2) Data Sets: We perform experiments using five different data sets, *i.e.* the COIL20 image data set [46], the Extended Yale database B (YaleB) [47], the BF0502 data set [48], the CIFAR10 image database [49], and the mnist handwritten digits [50]. The COIL20 data set consists of 1,440 object images distributed over 20 subjects, where the dimension of each image is 32×32. The YaleB data set contains 2,414 facial images captured from 38 persons, where the dimension of each image is 192×168. The used BF0502 data set [23] includes 1,200 facial images cropped from the TV series "Buffy the Vampire Slayer", where each subject contains 200 data points. The CIFAR10 data set includes 60,000 32×32 color images of which the first 200 images for each class are converted into gray images and used in our experiments. The used mnist is a subset by following the setting of [30].

For comprehensive studies, we conduct experiments not only using raw data (*i.e.* gray value), but also using the following four different features, *i.e.* dense scale-invariant feature transform (DSIFT) [51], square root of DSIFT (SDSIFT) [52], the histogram of oriented gradients (HOG) [53], and local phase quantization (LPQ) [54]. To present more new results comparing with our conference paper [31], in this paper, we carry out experiments using YaleB and COIL20 with SDISFT and LPQ, and CIFAR10 with these four features. The details for extracting these features are introduced as follows:

- DSIFT and SDISFT: We divide each image into multiple patches and then densely sample SIFT descriptors from each patch. The patch size of YaleB is 12×12. For CIFAR10, COIL20, and mnist, the patch size is set as

TABLE I

RESULTS ON **CIFAR10**. THE NUMBERS IN **BOLDFACE** INDICATE THE BEST RESULT ACCORDING TO THE T-TEST, WHERE THE SIGNIFICANCE LEVEL IS 0.05. STRUCTAE-L1 AND STRUCTAE-L2 DENOTE THE VARIANT WITH ℓ_1- AND ℓ_2-NORM BASED PRIOR, RESPECTIVELY. STRUCTAE-L*S AND STRUCTAE-L*L DENOTE ACHIEVING CLUSTERING WITH K-MEANS AND SSC (OR LSR1), RESPECTIVELY. IN OTHER WORDS, STRUCTAE-L*S PERFORMS CLUSTERING IN SUBSPACE, AND STRUCTAE-L*L PERFORMS CLUSTERING IN THE LATENT SUBSPACE

Features	DSIFT					HOG				
Methods	Accuracy	NMI	ARI	Precision	Fscore	Accuracy	NMI	ARI	Precision	Fscore
StructAE-L1S	**18.05±0.07**	**4.63±0.19**	**1.89±0.05**	**11.67±0.04**	**11.76±0.05**	**25.92±0.95**	**12.85±0.53**	**6.62±0.37**	**15.88±0.35**	**16.02±0.31**
StructAE-L1L	18.00±0.08	4.10±0.07	1.76±0.04	11.55±0.03	11.67±0.03	15.50±0.02	1.69±0.02	0.51±0.01	10.42±0.01	10.75±0.01
StructAE-L2S	17.53±0.15	4.31±0.32	1.74±0.11	11.53±0.10	11.63±0.11	23.32±1.38	11.33±0.84	5.59±0.58	14.95±0.50	15.11±0.55
StructAE-L2L	17.09±0.14	4.17±0.12	1.60±0.03	11.40±0.03	11.58±0.02	15.35±0.43	2.42±0.33	0.84±0.23	10.72±0.20	10.82±0.24
SAEg	12.75±0.00	0.93±0.00	0.03±0.00	9.95±0.00	11.07±0.01	17.54±0.15	4.58±0.02	1.90±0.01	11.67±0.01	11.76±0.01
SAEs	13.25±0.08	1.13±0.02	0.10±0.01	10.06±0.01	11.15±0.02	17.80±0.00	4.54±0.00	1.94±0.00	11.72±0.00	11.76±0.00
SSC	15.92±0.08	3.93±0.04	1.54±0.02	11.32±0.02	11.66±0.02	18.40±0.07	4.66±0.03	2.01±0.02	11.66±0.01	12.45±0.07
KSSC-R	17.00±0.00	4.12±0.01	1.52±0.01	11.30±0.01	11.66±0.00	23.60±0.00	11.22±0.01	6.04±0.01	14.01±0.01	14.57±0.01
KSSC-P	16.68±0.06	3.90±0.01	1.57±0.00	11.37±0.00	11.55±0.01	23.70±0.00	11.86±0.00	6.47±0.00	14.26±0.00	15.00±0.00
LS3C-L	14.46±0.02	3.86±0.01	1.38±0.01	11.20±0.01	11.34±0.01	14.83±0.15	2.30±0.11	0.75±0.06	10.64±0.05	10.72±0.05
LS3C-P	16.80±0.00	4.05±0.00	1.74±0.00	11.51±0.00	11.68±0.01	20.55±0.00	10.57±0.00	4.80±0.00	13.22±0.00	14.46±0.00
LS3C-R	16.73±0.24	3.50±0.05	1.46±0.04	11.28±0.03	11.40±0.04	21.30±0.00	11.39±0.00	5.59±0.00	13.94±0.00	15.24±0.00
LRR	16.35±0.00	4.25±0.00	1.53±0.00	10.99±0.00	11.44±0.00	15.25±0.11	2.06±0.03	0.69±0.02	10.58±0.02	10.86±0.02
KLRR-R	16.85±0.00	3.93±0.00	1.37±0.00	11.15±0.00	11.63±0.00	23.35±0.00	11.90±0.00	0.70±0.00	14.09±0.00	15.14±0.00
KLRR-P	16.84±0.02	4.28±0.02	1.74±0.01	11.53±0.01	11.57±0.01	23.30±0.00	12.00±0.00	6.66±0.00	14.24±0.00	14.32±0.00
LRSC	16.44±0.13	3.24±0.01	1.21±0.03	11.03±0.02	11.43±0.03	15.51±0.40	2.45±0.26	0.86±0.20	10.74±0.18	10.83±0.21
LSR1	16.21±0.05	3.97±0.02	1.49±0.01	11.29±0.01	11.53±0.01	15.61±0.33	2.50±0.12	0.89±0.06	10.77±0.06	10.84±0.05
LSR2	16.16±0.15	3.66±0.04	1.32±0.02	11.14±0.02	11.10±0.02	15.52±0.49	2.43±0.14	0.85±0.10	10.74±0.09	10.81±0.09
SMR	15.15±0.00	2.16±0.00	0.57±0.00	10.48±0.00	10.63±0.00	20.92±0.04	6.18±0.02	3.02±0.02	12.54±0.02	13.23±0.01

4×4. By concatenating these SIFT descriptors of all patches of each image, we obtain a feature vector with the dimension of 32,256 for YaleB, 2,048 for CIFAR10, and 2,048 for COIL20, respectively.

- HOG: We first divide each image into multiple blocks with two scales, *i.e.* 4×4 and 2×2 for CIFAR10. Then, we extract a 9-dimensional HOG feature vector from each block. After concatenating the features of all patches of each image, the feature dimension is 2,880 for CIFAR10.
- LPQ: We first divide each image into multiple non-overlapping blocks and then extract the LPQ feature from each patch. The patch size is set as 8×8 for CIFAR10 and COIL20 and 15×15 for YaleB. For all the tested data sets, we set the size of LPQ window as 3, 5, and 7. By concatenating the features of all patches of each image, the dimension of each feature is 12,288 for CIFAR10 and COIL20 and 101,376 for YaleB.

Like [44], we perform dimension reduction using PCA on all the tested data sets and the reduced dimension is fixed to 300 for computational efficiency.

3) Evaluation Criteria: To evaluate the clustering quality of our method, five measurements are adopted, including *Accuracy* (or called *Purity*), normalized mutual information (denoted by *NMI*), the adjusted rand index (denoted by *ARI*), *Precision*, and *Fscore*. Moreover, we repeat all evaluated methods five times and report their mean and the standard deviation regarding to these metrics.

B. Investigation on Different Variants of StructAE

StructAE can incorporate various structured priors as discussed above. In this subsection, we compare the clustering results of StructAE with sparse (denoted by StructAE-L1) and non-sparse prior (denoted by StructAE-L2) on the CIFAR10 data set. Specifically, StructAE-L1 adopts the sparsity by solving Eq. (5), and StructAE-L2 utilizes the prior derived from Eq. (6). In our experiments, we directly use the tuned parameters from SSC for StructAE-L1 and from LSR2 for StructAE-L2. This implies that the performance of StructAEs might be further improved if the parameters for the structure prior are specifically tuned.

Furthermore, we also examine the performance of StructAE with subspace clustering and latent subspace clustering. To this end, four variants of StructAE are proposed and investigated, *i.e.* StructAE-L1S, StructAE-L2S, StructAE-L1L, and StructAE-L2L. The first two methods correspond to StructAE-L1 and StructAE-L2 with k-means (clustering in subspace). The latest two approaches correspond to StructAE-L1 and StructAE-L2 with SSC and LSR (clustering in the latent subspace), respectively. In experiments, the learning rate μ of StructAE-L1S is set as 2^{-11} for DSIFT, SDSIFT, and LPQ, and 2^{-10} for HOG. Regarding to StructAE-L1L, μ is set as 2^{-11} for DSIFT and LPQ, and 2^{-10} for SDSIFT and HOG. Regarding to StructAE-L2S, μ is 2^{-11} for DSIFT and LPQ, and 2^{-12} for SDSIFT and HOG. Regarding to StructAE-L2L, we set $\mu = 2^{-12}$ in these four tests.

From Table I and II, the following observations can be obtained. 1) StructAE-L1 generally outperforms StructAE-L2, *i.e.* L1-norm based sparsity prior gives a better performance to StructAE comparing with L2-norm based non-sparsity prior. 2) In most cases, StructAE-L1L and StructAE-L2L perform better than StructAE-L1S and StructAE-L2S, respectively. This verifies the conclusion in [30], *i.e.* latent subspace clustering could be more competitive than subspace clustering. 3) StructAE-L1/StructAE-L2 significantly outperforms SSC/LSR, where the former could be regarded as the deep extension of the latter. Such a result supports our motivation that subspace clustering could benefit from deep representation learning. As StructAE-L1L remarkably outperforms other variants of StructAE, we will only report its performance in the following experiments and denote it as StructAE for simplicity.

TABLE II
RESULTS ON **CIFAR10**

Features	SDSIFT					LPQ				
Methods	Accuracy	NMI	ARI	Precision	Fscore	Accuracy	NMI	ARI	Precision	Fscore
StructAE-L1S	16.55±0.08	2.93±0.08	1.05±0.04	10.91±0.04	11.05±0.03	**21.05±0.56**	**7.54±0.78**	3.57±0.71	**12.23±0.46**	**16.72±0.57**
StructAE-L1L	**18.00±0.00**	3.86±0.00	**1.56±0.00**	**11.35±0.00**	**11.57±0.00**	17.65±0.09	4.23±0.04	1.90±0.02	11.65±0.02	11.86±0.02
StructAE-L2S	16.10±0.11	3.00±0.04	1.07±0.02	10.93±0.02	11.05±0.01	18.58±0.79	6.17±0.64	2.60±0.40	11.61±0.27	15.95±0.37
StructAE-L2L	17.18±0.34	**3.99±0.22**	1.48±0.10	11.27±0.09	11.56±0.09	16.85±0.00	4.37±0.00	1.91±0.00	11.67±0.00	11.79±0.00
SAEg	15.50±0.00	3.28±0.00	1.02±0.00	10.85±0.00	11.37±0.00	12.69±0.11	0.91±0.04	0.01±0.00	9.97±0.01	11.20±0.04
SAEs	15.34±0.05	3.61±0.01	1.14±0.00	10.95±0.00	11.44±0.01	13.13±0.07	1.08±0.02	0.09±0.01	10.05±0.01	10.80±0.03
SSC	15.92±0.12	2.96±0.04	1.01±0.02	10.86±0.01	11.19±0.01	16.02±0.03	3.44±0.03	1.47±0.02	11.27±0.01	11.54±0.01
KSSC-R	16.64±0.05	3.38±0.02	1.27±0.02	11.10±0.01	11.30±0.01	20.47±0.04	6.78±0.00	**4.48±0.00**	11.85±0.00	14.36±0.00
KSSC-P	6.65±0.00	2.87±0.01	1.11±0.00	10.96±0.00	11.09±0.00	18.92±0.11	5.72±0.09	2.64±0.05	11.34±0.04	12.42±0.05
LS3C-L	16.69±0.14	3.84±0.04	1.51±0.02	11.29±0.02	11.28±0.02	17.20±0.06	3.76±0.03	1.57±0.01	11.38±0.01	11.45±0.01
LS3C-P	16.85±0.02	2.83±0.01	1.10±0.01	10.94±0.01	11.22±0.01	17.65±0.00	6.03±0.00	2.90±0.00	11.44±0.00	13.10±0.00
LS3C-R	15.30±0.00	2.52±0.00	0.98±0.00	10.82±0.00	11.28±0.00	19.10±0.05	5.65±0.02	2.74±0.02	11.39±0.01	12.66±0.01
LRR	15.05±0.09	3.64±0.03	0.89±0.01	10.55±0.01	11.16±0.04	15.06±0.09	2.53±0.02	0.73±0.01	10.54±0.01	12.09±0.02
KLRR-R	16.47±0.03	3.59±0.01	1.37±0.00	11.19±0.00	11.31±0.00	20.35±0.00	6.79±0.00	4.52±0.00	12.02±0.00	14.11±0.00
KLRR-P	17.13±0.03	3.44±0.01	1.39±0.01	11.18±0.01	11.48±0.00	19.81±0.19	6.73±0.04	3.15±0.03	11.76±0.03	13.01±0.03
LRSC	17.09±0.07	3.42±0.10	1.50±0.04	11.28±0.04	11.49±0.03	18.63±0.04	4.09±0.01	1.80±0.00	11.57±0.00	11.75±0.00
LSR1	15.88±0.10	3.28±0.07	1.41±0.03	11.22±0.02	11.48±0.03	16.67±0.03	3.88±0.00	1.75±0.00	11.52±0.00	11.76±0.00
LSR2	16.47±0.23	3.68±0.04	1.23±0.03	11.05±0.03	11.36±0.03	16.65±0.00	3.79±0.00	1.69±0.00	11.47±0.00	11.70±0.00
SMR	16.15±0.00	2.30±0.00	1.11±0.00	10.96±0.00	11.12±0.00	17.30±0.00	3.98±0.00	1.71±0.00	11.28±0.00	13.09±0.00

TABLE III
RESULTS ON **COIL20**

Features	SDSIFT					LPQ				
Methods	Accuracy	NMI	ARI	Precision	Fscore	Accuracy	NMI	ARI	Precision	Fscore
StructAE	**91.04±2.23**	**94.67±0.99**	**88.61±1.95**	**85.90±2.36**	**89.19±1.83**	**84.38±1.67**	**89.72±0.57**	**81.51±2.02**	**81.65±2.83**	**82.43±1.90**
SAEg	79.71±2.86	91.90±1.05	78.09±2.10	70.36±2.84	79.31±1.97	60.42±1.14	70.71±0.86	50.64±1.39	49.74±1.51	53.26±1.33
SAEs	78.94±1.15	91.85±0.04	78.43±1.19	70.77±3.27	79.63±1.09	54.35±1.63	65.63±1.05	42.99±1.79	41.77±1.64	46.09±1.68
SSC	82.17±1.59	91.13±0.69	78.86±1.52	70.18±3.32	80.05±1.40	75.53±1.82	88.01±0.85	74.53±1.91	70.86±1.63	75.88±1.82
KSSC-R	80.58±3.14	92.33±1.05	79.70±2.21	71.67±3.30	80.83±2.07	71.89±1.54	84.23±1.19	66.84±2.51	63.64±3.55	68.60±2.34
KSSC-P	85.56±1.00	94.27±0.13	83.36±0.62	75.58±0.31	84.27±0.59	71.39±0.51	84.08±0.36	67.85±0.72	63.76±1.80	69.59±0.66
LS3C-L	34.04±1.78	52.24±1.75	19.09±2.61	16.85±2.03	25.05±2.22	50.33±0.79	62.67±0.38	39.67±0.47	36.20±1.01	43.22±0.43
LS3C-P	67.28±2.95	80.97±1.23	60.30±3.34	55.05±4.23	62.53±3.09	57.74±2.11	75.14±0.74	53.52±1.40	49.04±1.67	56.14±1.30
LS3C-R	21.92±0.48	31.76±0.66	14.75±0.45	15.39±0.28	20.30±0.55	55.97±3.27	71.63±1.62	47.36±3.51	43.42±4.35	50.39±3.19
LRR	87.51±1.76	94.28±0.61	84.34±1.88	77.40±3.17	85.19±1.77	75.99±4.01	88.64±1.37	76.12±3.03	69.67±3.92	74.58±2.83
KLRR-R	79.94±0.88	89.02±0.74	77.75±1.32	75.95±2.33	78.89±1.24	68.58±0.91	79.64±0.60	61.44±1.21	58.33±1.12	63.51±1.14
KLRR-P	77.21±1.95	87.43±0.88	74.32±1.72	71.09±2.05	75.67±1.62	64.28±1.18	76.55±0.63	56.37±1.61	54.36±2.40	58.69±1.48
LRSC	55.08±0.12	71.26±0.19	47.95±0.40	45.55±0.42	50.81±0.38	61.85±1.44	72.85±0.96	53.15±1.86	52.04±1.95	55.63±1.75
LSR1	64.86±0.83	74.10±0.55	53.78±2.03	50.98±2.72	56.30±1.88	69.57±1.54	77.81±0.88	61.53±1.38	60.66±1.22	63.51±1.31
LSR2	65.00±1.67	74.34±0.73	54.31±1.91	50.91±2.80	56.83±1.75	69.82±2.66	77.10±1.24	61.20±2.33	60.72±2.94	63.20±2.19
SMR	80.22±1.07	89.97±0.19	78.27±0.26	75.30±1.70	79.40±0.22	77.88±1.17	86.80±0.12	73.63±1.38	71.99±2.30	74.98±1.29

C. Comparison With State-of-the-Art

In this subsection, we compare StructAE with some state-of-the-art approaches using COIL20, YaleB, and mnist. Moreover, we have also illustrated the 2D-visualization of COIL20 and YaleB by conducting t-sne [55] on the input data and features learned by our StructAE.

1) Results on COIL20: The results of StructAE on COIL20 are reported in Table III. In experiments, we set $\lambda_1 = 10^{-4}$ and $\mu = 2^{-10}$ of StructAE for the SDSIFT feature, and $\lambda_1 = 5 \times 10^{-3}$ and $\mu = 2^{-11}$ for the LPQ feature.

From the results, we can observe that:

- StructAE outperforms the other methods by a considerable margin in terms of the five evaluation metrics. When using the LPQ feature, for example, its performance is at least 6.50%, 1.08%, 5.39%, 9.66%, and 6.55% higher than the second best result regarding to these five metrics.
- Considering different features, the best performance is achieved by our StructAE on SDSIFT. The Accuracy of StructAE increases from 84.38% (using the LPQ features) to 91.04%.
- Among the tested approaches, StructAE and LS3C are only two methods to perform subspace clustering in the learned latent space. The results show that LS3C is inferior to our StructAE. This may be because LS3C is a shallow model while ours are deep models.
- StructAE also remarkably outperforms SAE. This result is not surprising because our model incorporates locality and globality together, while SAE only considers the locality and ignores the relations among data points.
- Fig. 2 shows that our method could learn a compact and distinctive representation which makes it superior to the tested methods.

2) Results on YaleB: We set $\lambda_1 = 10^{-4}$ and $\mu = 2^{-10}$ for StructAE in this experiment. Table IV reports results of the tested methods on the YaleB data set, which shows that: 1) In all tests, StructAE achieves the best performance. When using the SDISFT feature, for example, it outperforms the second best method with a performance gain of 6.38% in *Accuracy*, 3.43% in *NMI*, 7.61% in *ARI*, 9.50% in *Precision*, and 7.40% in *Fscore*; 2) All the investigated methods achieve the worst results on the LPQ features. Especially, the performance of SAEg and SAEs decreases more than a half. 3) Fig. 3 provides a visualized evidence for the superior performance of our method.

TABLE IV
RESULTS ON **YALEB**

Features	SDSIFT					LPQ				
Methods	Accuracy	NMI	ARI	Precision	Fscore	Accuracy	NMI	ARI	Precision	Fscore
StructAE	**94.70±2.50**	**96.58±0.76**	**92.29±2.44**	**89.72±3.90**	**92.50±2.37**	**45.65±1.06**	**50.24±1.45**	**20.84±0.28**	**22.25±0.28**	**26.61±0.23**
SAEg	88.32±2.60	93.15±0.86	84.68±2.30	79.73±3.74	85.10±2.22	27.77±0.58	32.88±0.45	7.40±0.51	7.88±0.47	10.68±0.43
SAEs	85.95±0.77	92.77±0.52	83.00±1.90	77.28±2.76	83.48±1.84	24.95±0.37	32.21±0.33	8.52±0.20	9.60±0.29	11.32±0.16
SSC	86.35±1.13	92.82±0.48	84.04±1.06	80.22±1.58	84.47±1.03	37.71±0.61	32.46±0.81	9.90±0.35	8.89±0.27	13.45±0.32
KSSC-R	85.31±1.68	91.02±0.37	78.47±0.82	74.00±1.19	79.07±0.80	41.93±1.11	37.50±0.42	13.09±0.69	11.16±0.64	16.45±0.60
KSSC-P	62.39±1.19	68.07±0.56	38.25±3.10	33.27±3.79	40.19±2.91	42.39±1.17	38.08±1.10	13.85±0.85	11.72±0.71	15.23±0.77
LS3C-L	51.28±1.02	61.60±0.46	29.66±0.65	26.08±0.83	31.89±0.60	44.06±0.79	46.05±0.37	9.85±0.23	8.66±0.18	13.49±0.20
LS3C-P	48.82±1.39	54.76±0.39	17.08±1.42	13.91±1.29	20.22±1.26	14.65±0.39	21.11±0.40	0.98±0.03	3.14±0.02	5.42±0.04
LS3C-R	10.56±0.29	13.15±0.19	1.10±0.07	3.50±0.05	4.11±0.10	6.72±0.10	6.90±0.20	0.05±0.03	2.63±0.02	3.49±0.05
LRR	85.62±0.55	92.99±0.43	83.25±2.13	77.27±3.08	83.73±2.07	38.69±1.31	49.93±0.51	14.38±3.24	17.73±3.50	21.84±3.03
KLRR-R	70.27±1.10	75.84±0.17	49.12±1.15	43.24±1.60	50.67±1.10	36.30±0.58	46.49±0.53	13.70±0.90	10.97±1.12	15.93±0.81
KLRR-P	69.76±1.36	75.81±0.52	52.18±1.06	47.70±1.91	53.57±0.99	19.87±0.52	35.73±0.54	9.99±0.28	12.06±0.27	12.39±0.27
LRSC	74.74±0.62	79.63±0.73	58.06±1.67	53.04±2.19	59.28±1.60	45.65±0.97	49.64±0.62	17.44±0.89	15.74±0.83	20.98±0.86
LSR1	76.02±1.21	80.14±0.47	58.27±1.60	53.00±2.42	59.48±1.53	41.50±1.22	47.00±0.54	11.00±1.30	17.39±1.31	17.65±1.18
LSR2	74.70±1.32	79.39±0.80	56.14±0.34	50.41±0.64	57.43±0.33	41.40±0.89	47.76±0.78	11.93±1.67	18.32±1.76	18.10±1.53
SMR	77.76±0.44	83.44±0.39	64.71±1.03	59.51±1.72	65.72±0.99	44.77±0.46	48.12±1.00	18.55±2.62	14.04±3.42	19.11±2.46

TABLE V
RESULTS ON **MNIST**

Features	SDSIFT					LPQ				
Methods	Accuracy	NMI	ARI	Precision	Fscore	Accuracy	NMI	ARI	Precision	Fscore
StructAE	**65.70±0.00**	**68.98±0.00**	**57.99±0.00**	**57.31±0.00**	**62.60±0.00**	**61.10±0.02**	**55.95±0.06**	**45.40±0.04**	**49.12±0.04**	**51.10±0.04**
SAEg	52.03±0.47	55.93±0.07	38.71±0.09	36.83±0.05	46.47±0.09	31.65±0.00	25.10±0.00	13.96±0.00	21.41±0.00	23.39±0.00
SAEs	48.63±1.05	53.73±0.31	37.04±1.18	37.21±2.32	44.64±0.69	31.65±0.00	25.10±0.00	13.96±0.00	21.41±0.00	23.39±0.00
SSC	63.85±0.00	65.33±0.00	53.36±0.00	54.81±0.00	60.48±0.00	58.17±0.00	53.98±0.00	42.03±0.00	46.78±0.00	49.90±0.00
KSSC-R	60.90±0.00	64.60±0.00	51.65±0.00	51.70±0.00	57.00±0.00	55.10±0.00	50.23±0.00	37.66±0.00	42.59±0.00	44.15±0.00
KSSC-P	61.20±0.00	65.06±0.00	52.06±0.00	52.08±0.00	57.36±0.00	54.93±0.00	49.81±0.00	37.35±0.00	42.35±0.00	43.86±0.00
LS3C-L	35.17±0.22	40.47±0.11	25.73±0.12	28.12±0.10	35.01±0.10	42.15±0.01	42.29±0.03	27.31±0.02	32.46±0.01	35.23±0.02
LS3C-P	47.01±0.50	48.28±0.09	33.46±0.07	37.22±0.04	40.78±0.08	41.73±0.01	39.09±0.01	25.25±0.00	30.77±0.00	33.39±0.00
LS3C-R	64.05±0.00	66.71±0.00	54.94±0.00	55.91±0.00	61.09±0.00	58.98±0.00	53.79±0.00	43.14±0.00	47.37±0.00	49.47±0.00
LRR	63.69±0.02	66.37±0.04	53.88±0.03	53.59±0.03	58.97±0.02	60.20±0.00	55.26±0.00	44.52±0.00	48.98±0.00	50.24±0.00
KLRR-R	58.67±0.00	57.16±0.00	45.06±0.00	49.57±0.00	50.70±0.00	54.93±0.00	50.31±0.00	38.42±0.00	43.49±0.00	44.79±0.00
KLRR-P	58.67±0.00	58.90±0.00	46.22±0.00	50.36±0.00	51.77±0.00	53.27±0.00	46.59±0.00	35.34±0.00	41.05±0.00	41.98±0.00
LRSC	56.96±0.02	56.72±0.03	44.58±0.04	46.82±0.04	50.60±0.04	50.71±0.05	46.42±0.06	34.69±0.05	39.49±0.04	41.59±0.05
LSR1	62.48±0.00	60.05±0.00	48.38±0.00	52.17±0.00	53.71±0.00	50.92±0.00	46.53±0.00	34.93±0.00	39.94±0.00	41.76±0.00
LSR2	62.50±0.00	59.95±0.00	48.32±0.00	52.13±0.00	53.66±0.00	50.88±0.00	46.60±0.00	34.96±0.00	39.91±0.00	41.80±0.00
SMR	62.77±0.00	67.22±0.83	56.19±0.00	56.83±0.00	61.62±0.00	51.20±0.00	49.01±0.00	36.21±0.00	40.65±0.00	42.98±0.00

3) Results on mnist: In the experiments, λ_1 and μ of StructAE are fixed to 0.05 and 2^{-10}, respectively. Table V reports the results which show that the proposed StructAE consistently outperforms the evaluated methods on the mnist and LS3C-R achieves the second best result in most cases. Noted that, [30] has experimentally shown that the highest Accuracy of SSC, LRR, LS3C, and nonlinear LS3C is about 41.41%, 24.52%, 41.32%, and 45.37% on the mnist. Clearly, these four methods remarkably improve their performance in our experimental setting.

TABLE VI
COMPARISON WITH DEEP FEATURES. THE NUMBERS IN **BOLDFACE** INDICATE THE BEST RESULT ACCORDING TO THE T-TEST, WHERE THE SIGNIFICANCE LEVEL IS 0.05

Methods	CIFAR10	COIL20	YaleB	mnist
StructAE	**25.92±0.95**	**91.04±2.23**	**94.70±2.50**	**65.70±0.00**
SSC+VGG16	20.40±0.10	28.49±0.53	35.85±2.64	56.55±0.00
SSC+ResNet50	21.56±0.10	41.01±0.57	40.47±1.49	62.87±0.01
LRR+VGG16	11.68±0.16	76.08±1.87	55.14±1.99	11.81±0.09
LRR+ResNet50	12.75±0.15	64.42±2.65	6.55±0.11	64.35±0.03

D. Comparison With Deep Features

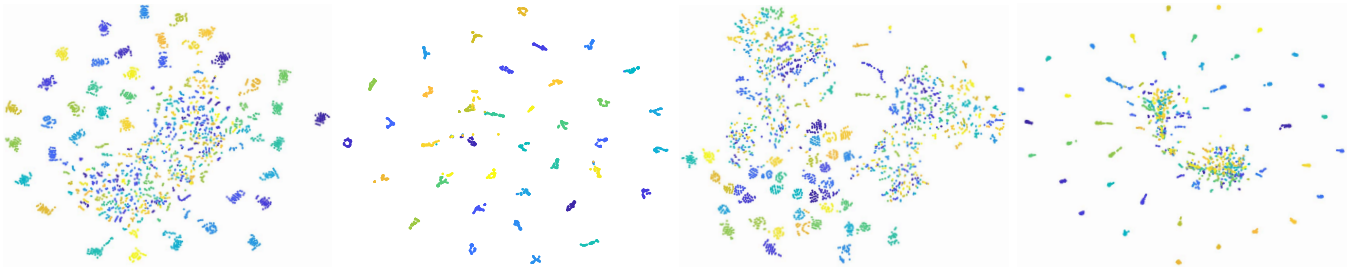
In this section, we show that our method with handcrafted features could significantly outperform state-of-the-art subspace clustering approaches with deep features. Specifically, we first resize each image to 224×224 so that the images could be passed through the pretrained VGG16 [56] and ResNet50 [57]. After obtaining deep features, we conduct SSC and LRR to achieve data clustering. For simplicity, here, we report the best result achieved by our method in the above setting and use the Accuracy as the evaluation metric. Table VI demonstrates that the proposed StructAE is superior to the baselines by a considerable performance margin. To be exact, it is 4.36%, 14.96%, 39.56%, and 1.35% higher than the best baseline on these four data sets. Furthermore, one could observe that LRR+VGG16 does not give an acceptable result on CIFAR10 and mnist. The possible reason is that VGG16 features do not show low rank property.

E. Influence of Depths, Widths, and Parameter

In this subsection, we examine the clustering results of our StructAE with different depths, widths, and λ_1 using the BF0502 raw data set in terms of clustering Accuracy. Regarding the evaluation on depths and widths, two variants of StructAE ($\lambda_1 = 10^{-3}$ and $\mu = 2^{-10}$) are investigated, *i.e.* StructAE4 and StructAE2, which correspond to five-layer case ($M = 4$) and three-layer case ($M = 2$). For $M = 2$, the used neural network is with 300-150-300 neurons. Moreover, StructAE100 and StructAE50 have similar structure with StructAE4. The only one difference is that the middle layer of StructAE100 and StructAE50 consists of 100 and 50 neurons,

(a) (b) (c) (d)

Fig. 2. Visualization on the COIL20 data set. See the electronic version for a better look. (a) SDSIFT. (b) StructAE + SDSIFT. (c) LPQ. (d) StructAE + LPQ.



(a) (b) (c) (d)

Fig. 3. Visualization on the YaleB data set. See the electronic version for a better look. (a) SDSIFT. (b) StructAE + SDSIFT. (c) LPQ. (d) StructAE + LPQ.

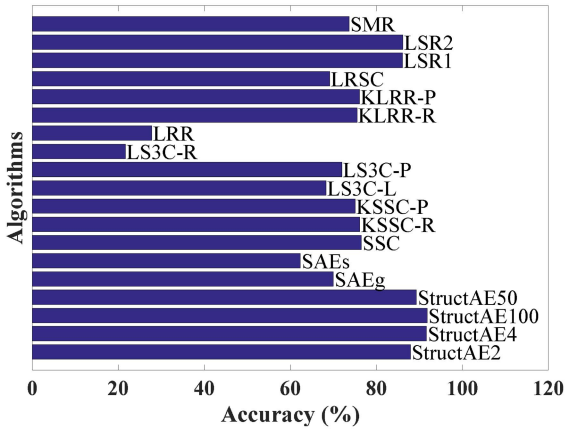


Fig. 4. The performance of StructAE with different depths and widths on the BF0502 raw data set, where StructAE2 denotes the network with 2 hidden layers, and StructAE4 denotes the 4-hidden-layer network. StructAE100 and StructAE50 denote the networks with 100 and 50 neurons at the middle layer, respectively.

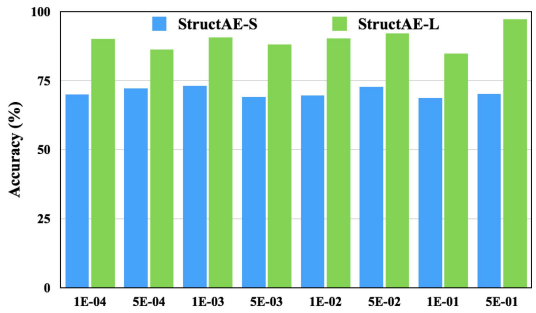


Fig. 5. Influence of the parameter λ_1 on the BF0502 raw data set.

respectively. In experiments, λ_1 is set as 10^{-2} for StructAE2 and 10^{-3} for StructAE4, StructAE100 and StructAE50. Regarding the evaluation on parameters, StructAE-S and StructAE-L are investigated, which correspond to StructAE-L1S and StructAE-L1L, respectively.

From Fig. 4–5, some observations are summarized as follows:

- Deeper network leads to better clustering results for our method. For example, StructAE4 is 3.75% higher than StructAE2 in *Accuracy*.
- The best performance is achieved by our StructAE4 which outperforms the best baseline method by 5.50%. Moreover, when the size of encoding layer reduces from 150 to 100, the clustering performance of StructAE remains unchanged. If this number further reduces to 50, *i.e.*, StructAE50 performs slightly worse, but it still outperforms the baselines.
- With increasing λ_1, the Accuracy of StructAE-S and StructAE-L varies in $[68.67\%, 73.08\%]$ and $[84.83\%, 97.25\%]$, respectively. Clearly, a specifically tuned λ_1 will remarkably improve the performance of our method.

F. Performance With Different Activation Functions

We examine the clustering performance of StructAE with four nonlinear functions by conducting experiments on the YaleB database with DSIFT and HOG. The investigated activation functions include *tanh*, *sigmoid*, non-saturating sigmoid (*nssigmoid*), and *softplus* (*i.e.* ReLU). For extensive investigations, we again investigate the performance of StructAE in

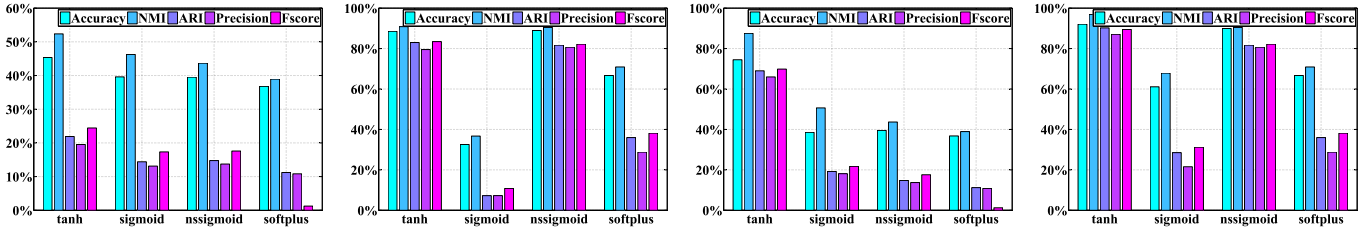


Fig. 6. The clustering results of StructAE with four nonlinear activation functions on the YaleB database, where StructAE-S denotes clustering in subspace and StructAE-L denotes clustering in the latent subspace. (a) StructAE-S on DSIFT. (b) StructAE-L on DSIFT. (c) StructAE-S on HOG. (d) StructAE-L on HOG.

TABLE VII
TIME COST ON THE **CIFAR10** DATA SET WITH THE DSIFT FEATURE

Methods	Training Time (s)	Inference Time (s)
StructAE-L1S	16434.84±61.89	258.30±19.31
StructAE-L1L	31468.28±72.26	295.07±54.49
StructAE-L2S	13542.07±82.62	185.41±10.61
StructAE-L2L	21503.82±69.96	129.20±13.91
SAEg	11311.43±67.13	311.43±34.40
SAEs	11313.16±59.26	313.16±27.13
SSC	106.26±6.93	-
KSSC-R	5613.10±29.94	-
KSSC-P	7393.17±75.11	-
LS3C-L	1166.00±12.43	-
LS3C-P	406.62±0.55	-
LS3C-G	652.72±1.01	-
LRR	168.07±4.64	-
KLRR-R	4139.38±98.83	-
KLRR-P	25.44±0.49	-
LRSC	35.34±2.37	-
LSR1	48.11±2.23	-
LSR2	50.96±1.95	-
SMR	197.88±4.63	-

the case of clustering in subspace (denoted by StructAE-S) and latent subspace (denoted by StructAE-L).

From Fig. 6, ones could observe that:

- By comparing StructAE-S and StructAE-L, the *sigmoid* function gives a remarkable performance change, *i.e.*, the second best result with StructAE-S versus the worst result with StructAE-L. The possible reason is that clustering in latent space makes the *sigmoid* function saturated, and lead to barely learning of StructAE-L. In contrast, with the non-saturating version of sigmoid (*i.e. nssigmoid*), StructAE-L performs stable in different settings.
- Discarding the *sigmoid* function, the activation with stronger nonlinearity gives better performance. From Fig. 6, ones could see that the *tanh* function is with the strongest nonlinearity which may result in its superior performance.

G. Time Cost Analysis

In this subsection, we investigate the computational efficiency of our method using the CIFAR10 data set with the DSIFT feature. The inference time denotes the cost taken by StructAEs and SAEs to perform representation learning and clustering after the networks converging. Table VII shows that the proposed StructAEs are less efficient than the baselines. The training time of StructAEs consists of three parts, *i.e.*, for training AutoEncoder, for computing the structure prior, and for training StructAEs by solving Eq.(4). In fact, the first part is the most time-consuming, which takes about 60% of the entire cost. Noticed that, once StructAE converges, it will achieve clustering result quickly.

V. CONCLUSION

This paper proposed a novel neural network for subspace clustering by simultaneously preserving the locality and globality of data sets. Extensive experimental results have demonstrated that our method is remarkably superior to 15 recently-proposed approaches in terms of five clustering evaluation metrics.

ACKNOWLEDGMENT

The authors would like to thank Prof. Vishal Monga and anonymous reviewers for their valuable comments and constructive suggestions to improve the quality of this paper.

REFERENCES

[1] R. Vidal, "Subspace clustering," *IEEE Signal Process. Mag.*, vol. 28, no. 2, pp. 52–68, Mar. 2011.
[2] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[3] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. 14th Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2001, pp. 849–856.
[4] H. Liu, T. Liu, J. Wu, D. Tao, and Y. Fu, "Spectral ensemble clustering," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Sydney, NSW, Australia, Aug. 2015, pp. 715–724.
[5] F. P. Nie, H. Wang, H. Huang, and C. Ding, "Unsupervised and semi-supervised learning via ℓ_1-norm graph," in *Proc. 12th IEEE Conf. Comput. Vis.*, Barcelona, Spain, Nov. 2011, pp. 2268–2273.
[6] C.-Y. Lu, H. Min, Z.-Q. Zhao, L. Zhu, D.-S. Huang, and S. Yan, "Robust and efficient subspace segmentation via least squares regression," in *Proc. 12th Eur. Conf. Comput. Vis.*, Florence, Italy, Oct. 2012, pp. 347–360.
[7] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2765–2781, Nov. 2013.
[8] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.
[9] R. He, Y. Zhang, Z. Sun, and Q. Yin, "Robust subspace clustering with complex noise," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4001–4013, Nov. 2015.
[10] X. Peng, Z. Yi, and H. Tang, "Robust subspace clustering via thresholding ridge regression," in *Proc. 29th AAAI Conf. Artif. Intell.*, Austin, TX, USA, Jan. 2015, pp. 3827–3833.

[11] C. Zhang, H. Fu, S. Liu, G. Liu, and X. Cao, "Low-rank tensor constrained multiview subspace clustering," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1582–1590.

[12] Y. Yang, J. Feng, N. Jojic, J. Yang, and T. S. Huang, "ℓ^0-sparse subspace clustering," in *Proc. 14th Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, Oct. 2016, pp. 731–747.

[13] C.-G. Li and R. Vidal, "A structured sparse plus structured low-rank framework for subspace clustering and completion," *IEEE Trans. Signal Process.*, vol. 64, no. 24, pp. 6557–6570, Dec. 2016.

[14] X. Peng, Z. Yu, Z. Yi, and H. Tang, "Constructing the L2-graph for robust subspace learning and subspace clustering," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 1053–1066, Apr. 2017.

[15] C. You, C.-G. Li, D. P. Robinson, and R. Vidal, "Oracle based active set algorithm for scalable elastic net subspace clustering," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3928–3937.

[16] Y. Yang, F. Shen, Z. Huang, H. T. Shen, and X. Li, "Discrete nonnegative spectral clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 9, pp. 1834–1845, Sep. 2017.

[17] E. Elhamifar, "High-rank matrix completion and clustering under self-expressive models," in *Proc. 28th Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, 2016, pp. 73–81.

[18] P. Ji, M. Salzmann, and H. Li, "Shape interaction matrix revisited and robustified: Efficient subspace clustering with corrupted and incomplete data," in *Proc. 15th IEEE Conf. Comput. Vis.*, Santiago, Chile, Dec. 2015, pp. 4687–4695.

[19] X. Peng, H. Tang, L. Zhang, Z. Yi, and S. Xiao, "A unified framework for representation-based subspace clustering of out-of-sample and large-scale data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2499–2512, Dec. 2016.

[20] Y. Yuan, J. Lin, and Q. Wang, "Dual-clustering-based hyperspectral band selection by contextual analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1431–1445, Mar. 2016.

[21] X. Li, J. Lv, and Z. Yi, "An efficient representation-based method for boundary point and outlier detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 51–62, Jan. 2018.

[22] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, 2003.

[23] S. Xiao, M. Tan, D. Xu, and Z. Y. Dong, "Robust kernel low-rank representation," *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2268–2281, Nov. 2016.

[24] V. M. Patel and R. Vidal, "Kernel sparse subspace clustering," in *Proc. 21st IEEE Int. Conf. Image Process.*, Paris, France, Oct. 2014, pp. 2849–2853.

[25] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[26] C. Deng, Z. Chen, X. Liu, X. Gao, and D. Tao, "Triplet-based deep hashing network for cross-modal retrieval," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3893–3903, Aug. 2018.

[27] H. Zhu *et al.*, "YoTube: Searching action proposal via recurrent and static regression networks," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 2609–2622, Jun. 2018.

[28] J. T. Zhou, H. Zhao, X. Peng, M. Fang, Z. Qin, and R. S. M. Goh, "Transfer hashing: From shallow to deep," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published.

[29] F. Shen, Y. Xu, L. Liu, Y. Yang, Z. Huang, and H. T. Shen, "Unsupervised deep hashing with similarity-adaptive and discrete optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.

[30] V. M. Patel, H. Van Nguyen, and R. Vidal, "Latent space sparse subspace clustering," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 225–232.

[31] X. Peng, S. Xiao, J. Feng, W. Yau, and Z. Yi, "Deep subspace clustering with sparsity prior," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, New York, NY, USA, Jul. 2016, pp. 1925–1931.

[32] J. Feng, Z. Lin, H. Xu, and S. Yan, "Robust subspace segmentation with block-diagonal prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 3818–3825.

[33] C. Hou, F. Nie, D. Yi, and D. Tao, "Discriminative embedded clustering: A framework for grouping high-dimensional data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 6, pp. 1287–1299, Jun. 2015.

[34] X. Peng, J. Lu, Z. Yi, and R. Yan, "Automatic subspace learning via principal coefficients embedding," *IEEE Trans. Cybern.*, vol. 47, no. 11, pp. 3583–3596, Nov. 2017.

[35] J. Y. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. 33th Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2016, pp. 478–487.

[36] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proc. 29th IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5147–5156.

[37] X. Peng, J. Feng, J. Lu, W.-Y. Yau, and Z. Yi, "Cascade subspace clustering," in *Proc. 31st AAAI Conf. Artif. Intell.* San Francisco, CA, USA, Feb. 2017, pp. 2478–2484.

[38] P. Ji, T. Zhang, H. Li, M. Salzmann, and I. Reid, "Deep subspace clustering networks," in *Proc. 29th Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, Dec. 2017, pp. 24–33.

[39] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.

[40] X. Peng, C. Lu, Z. Yi, and H. Tang, "Connections between nuclear-norm and frobenius-norm-based representations," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 218–224, Jan. 2018.

[41] P. Favaro, R. Vidal, and A. Ravichandran, "A closed form solution to robust subspace estimation and clustering," in *Proc. 24th IEEE Conf. Comput. Vis. Pattern Recognit.*, Colorado Springs, CO, USA, Jun. 2011, pp. 1801–1807.

[42] H. Hu, Z. Lin, J. Feng, and J. Zhou, "Smooth representation clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 3834–3841.

[43] F. Tian, B. Gao, Q. Cui, E. Chen, and T. Y. Liu, "Learning deep representations for graph clustering," in *Proc. 28th AAAI Conf. Artif. Intell.*, Qubec, QC, Canada, Jul. 2014, pp. 1293–1299.

[44] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *Proc. IEEE Conf. CVPR*, Jun. 2014, pp. 1875–1882.

[45] A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Fast l1-minimization algorithms and an application in robust face recognition: A review," Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2010-13, Feb. 2010.

[46] S. A. Nene *et al.*, "Columbia object image library (coil-20)," Tech. Rep. CUCS-005-96, 1996.

[47] A. S. Georghiades, P. N. Belhumeur, and D. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun. 2001.

[48] J. Sivic, M. Everingham, and A. Zisserman, "'Who are you?'—Learning person specific classifiers from video," in *Proc. 22th IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami, FL, USA, Jun. 2009, pp. 1145–1152.

[49] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[50] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[51] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[52] L. Wolf, T. Hassner, and Y. Taigman, "Similarity scores based on background samples," in *Proc. 9th Asian Conf. Comput. Vis.*, Xi'an, China, vol. 5995, Sep. 2010, pp. 88–97.

[53] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, vol. 1, no. 1, pp. 886–893.

[54] V. Ojansivu and J. Heikkilä, "Blur insensitive texture classification using local phase quantization," in *Image and Signal Processing*. Cherbourg, France: Springer, 2008, pp. 236–243.

[55] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[56] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, Sep. 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[57] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.