Deep Subspace Clustering with Sparsity Prior*

Xi Peng[1], **Shijie Xiao**[2,3], **Jiashi Feng**[4], **Wei-Yun Yau**[1], **and Zhang Yi**[5]
[1]Institute for Infocomm Research, Singapore; [2]Nanyang Technological University, Singapore;
[3]OmniVision Technologies Singapore Pte. Ltd.; [4]National University of Singapore, Singapore
[5]College of Computer Science, Sichuan University, Chengdu, P. R. China.
{pangsaai, xiaoshijiehithonor}@gmail.com, elefjia@nus.edu.sg,
wyyau@i2r.a-star.edu.sg, zhangyi@scu.edu.cn

Abstract

Subspace clustering aims to cluster unlabeled samples into multiple groups by implicitly seeking a subspace to fit each group. Most of existing methods are based on a shallow linear model, which may fail in handling data with nonlinear structure. In this paper, we propose a novel subspace clustering method – deeP subspAce clusteRing with sparsiTY prior (PARTY) – based on a new deep learning architecture. PARTY explicitly learns to progressively transform input data into nonlinear latent space and to be adaptive to the local and global subspace structure simultaneously. In particular, considering local structure, PARTY learns representation for the input data with minimal reconstruction error. Moreover, PARTY incorporates a *prior* sparsity information into the hidden representation learning to preserve the sparse reconstruction relation over the whole data set. To the best of our knowledge, PARTY is the first deep learning based subspace clustering method. Extensive experiments verify the effectiveness of our method.

1 Introduction

Subspace clustering aims to seek a set of implicit low-dimensional subspaces to fit provided unlabeled high-dimensional data and cluster them according to their membership to the subspaces [Vidal, 2011]. During the past decades, spectral clustering based methods [Costeira and Kanade, 1998; Shi and Malik, 2000; Ng *et al.*, 2001] have been state-of-the-art, which perform subspace clustering in following two steps. First, build an affinity matrix (*i.e.* similarity graph) $\mathbf{C}$ to depict the relationship of the data, where $\mathbf{C}_{ij}$ denotes the similarity between data points $\mathbf{x}_i$ and $\mathbf{x}_j$. Second, cluster the data through clustering the eigenvectors of the graph Laplacian $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$ and $\mathbf{A} = |\mathbf{C}| + |\mathbf{C}^T|$. Clearly, performance of spectral clustering based methods critically relies on the quality of the built affinity matrix.

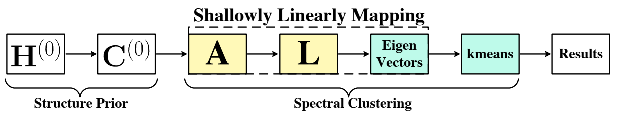
*Corresponding Author: Zhang Yi.

In recent years, many works [Nie *et al.*, 2011; Lu *et al.*, 2012; Elhamifar and Vidal, 2013; Liu *et al.*, 2013; Feng *et al.*, 2014; He *et al.*, 2015; Peng *et al.*, 2015; Zhang *et al.*, 2015] have devoted to obtaining a good affinity matrix by using the self-expression of inputs. More specifically, these methods linearly represent the input $\mathbf{X} \in \mathbb{R}^{d\times n}$ by solving

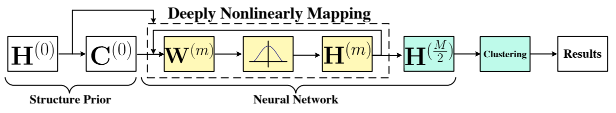
$$\min_{\mathbf{C}} \frac{1}{2}\|\mathbf{X} - \mathbf{X}\mathbf{C}\|_F^2 + \lambda\mathcal{R}(\mathbf{C}), \tag{1}$$

where d denotes the dimensionality of inputs, n is the data size, $\mathbf{C} \in \mathbb{R}^{n\times n}$ corresponds to the self-expression of $\mathbf{X}$, $\|\cdot\|_F$ denotes the Frobenius norm, and $\mathcal{R}(\mathbf{C})$ denotes *a prior* structured regularization on $\mathbf{C}$. The major difference among these methods lies in the choice of $\mathcal{R}(\mathbf{C})$. For example, sparse subspace clustering (SSC) [Elhamifar and Vidal, 2013] and low rank representation (LRR) [Liu *et al.*, 2013] encourage $\mathbf{C}$ to be sparse and low rank by adopting ℓ_1- and nuclear-norm as $\mathcal{R}(\cdot)$, respectively. Once obtaining $\mathbf{C}$, those methods build an affinity matrix as $\mathbf{A} = |\mathbf{C}| + |\mathbf{C}^T|$ and apply spectral clustering [Ng *et al.*, 2001] over $\mathbf{A}$ to cluster data.

Although those approaches have shown encouraging performance, we observe that they suffer from the following limitations. First, most of existing methods strive for building high-quality affinity matrix while ignore the importance of low-dimensional representation derived from the affinity matrix. In fact, most of them simply use several eigenvectors (*w.r.t.* leading eigenvalues) of $\mathbf{L}$ as the low-dimensional representation of data $\mathbf{X}$. This is identical to Laplacian Eigenmap (LE) [Belkin and Niyogi, 2003] in essence. In other words, those spectral clustering based methods perform subspace clustering through 1) building an affinity matrix; 2) performing LE on the affinity matrix to produce low dimensional representations; and 3) clustering the representations. It still remains open whether there is a better way to embed the affinity matrix into low-dimensional spaces and recover the latent subspaces therein, and we believe investigating data embedding schemes in depth will significantly boost the clustering performance. The second limitation of those methods is due to their intrinsic linearity and they cannot handle data with significant non-linearity well. To address this issue, some kernel extensions of existing methods have been proposed, *e.g.*, kernel sparse subspace clustering (KSSC) [Patel and Vidal, 2014] and kernel low rank representation (KLRR) [Xiao *et*



(a) Architecture of some existing spectral clustering based methods.



(b) Architecture of the proposed PARTY.

Figure 1: Comparison on architectures of PARTY and subspace clustering methods: (a) a popular architectures of existing subspace clustering methods with $\mathbf{L}$ being the graph Laplacian and (b) the architecture of PARTY. In (b), $\mathbf{H}^{(m)}$ denotes the output of the m-th layer, with $m = 1, 2, \cdots, M$ and $\mathbf{H}^{(0)}$ denotes inputs. PARTY is complementary to existing subspace clustering methods. Thus, one can directly perform k-means or other existing clustering methods on the compact representation $\mathbf{H}^{(\frac{M}{2})}$. The major difference between these two architectures is colored.

al., 2015]. They compute the representation in a kernel space rather than the original space. However, the performance of these kernel based methods heavily relies on the choice of kernel function, while a golden rule for choosing the kernel functions is still absent in practice.

Motivated by the fact that representation learning is vital to the subspace clustering and deep learning is one of the most powerful representation learning approaches [Hinton and Salakhutdinov, 2006; Krizhevsky *et al.*, 2012], we propose a novel subspace clustering framework based on a deep learning model, namely, deeP subspAce clusteRing with sparsiTY prior (PARTY). The proposed method achieves subspace clustering by computing the sparse reconstruction relation from the input space, learning a neural network with locality and globality of data set, and clustering the compact representation learned by our neural network. The basic idea of PARTY is illustrated in Fig. 1. One can observe that PARTY significantly differs from the existing subspace clustering approaches in the following aspects: 1) PARTY directly learns low-dimensional representation of data instead of relying on an affinity matrix; 2) PARTY possesses a multilayer structure, which offers stronger ability to model nonlinearity of the data; 3) Although being partially similar to stacked Auto-Encoder (SAE), PARTY not only considers the locality for reconstructing input data, but also incorporates the structured global prior in representation learning; 4) In contrast to kernel-based methods, PARTY provides explicit transformations and better scalability as it avoids loading all the data to calculate the kernel function; 5) PARTY is compatible with both k-means and most of existing subspace clustering methods. When applying k-means on the outputs of PARTY, we theoretically demonstrate that under mild conditions, PARTY performs in a similar way to spectral clustering. When combined with spectral clustering, PARTY actually performs latent subspace clustering like [Patel *et al.*, 2013], by learning the latent space via hierarchically stacked nonlinear transformations.

2 Related Works

In this section, we briefly discuss some existing works in subspace clustering and deep learning, respectively.

Subspace Clustering: Recently, many subspace clustering methods have been proposed [Elhamifar and Vidal, 2013; Liu *et al.*, 2013; Patel *et al.*, 2013; Patel and Vidal, 2014; Peng *et al.*, 2015; Lu *et al.*, 2012; Favaro *et al.*, 2011; Yuan *et al.*, 2016; Yu *et al.*, 2015; Hu *et al.*, 2014b], of which the major difference is the way to obtain the affinity matrix. More specifically, these methods use the linear reconstruction coefficients to build the affinity matrix, by enforcing different constraints on the coefficients. However, these approaches are linear models, which cannot model the nonlinearity of data in many practical situations. To address this problem, [Patel and Vidal, 2014] and [Xiao *et al.*, 2015] proposed kernel SSC (KSSC) and kernel LRR (KLRR), respectively. However, how to choose appropriate kernel for these kernel-based methods is usually unclear in practice.

Unlike these subspace clustering approaches, our method learns multiple hierarchical nonlinear transformations (in the neural network) to map the input into another space so that the nonlinearity can be incorporated into the obtained low-dimensional representation, thus resulting a better clustering performance. To the best of our knowledge, this is *the first deep subspace clustering framework*.

Deep Learning: With the impressive power of learning representations, deep learning has achieved huge success in numerous applications, especially in the scenario of supervised learning, *e.g.* image classification [Krizhevsky *et al.*, 2012], metric learning [Hu *et al.*, 2014b], image super-resolution [Wang *et al.*, 2015], etc. In contrast, less works investigated the applications with unsupervised learning scheme, such as subspace clustering. To the best of our knowledge, there are only two recent works [Ma *et al.*, 2014; Tian *et al.*, 2014] that apply the existing Auto-Encoder (AE) for clustering. They mainly differ in the input of the neural network. Specifically, [Ma *et al.*, 2014] directly encodes the representation from raw data, whereas [Tian *et al.*, 2014] feeds a predefined affinity matrix to the neural network. Due to the absence of comparison with state-of-the-art subspace clustering such as SSC [Elhamifar and Vidal, 2013] and LRR [Liu *et al.*, 2013], it remains unknown whether these two works are relatively effective for subspace clustering. Since unsupervised deep learning remains an open issue, it could be better to incorporate domain knowledge and the merits of previous works to develop new unsupervised deep model as suggested by [Bengio *et al.*, 2013].

Different from these two methods, our framework is based on a *new* neural network which preserves both locality and globality. To be exact, the locality is considered by minimizing the reconstruction error of the sample itself, while the globality is guaranteed by minimizing the reconstruction error of using the whole data set to reconstruct each sample. Such a framework is complementary to existing subspace clustering methods and deep learning approaches, because it incorporates the advantages (*i.e.* structure prior) of existing subspace clustering methods into the framework deep learning. As a result, it is well expected that the proposed neural network can achieve satisfactory performance while handling unlabeled data. To our best of knowledge, this is *the first work to introduce the global structure prior into neural network for unsupervised learning*.

3 Deep Subspace Clustering with Sparsity Prior

In this section, we elaborate on the details of the proposed PARTY model for subspace clustering. PARTY clusters data in the following three steps: computing the sparsity prior from raw data, learning a neural network to map the input into a latent space, and clustering the low-dimensional data representations into multiple subspaces. We will first explain how PARTY is specifically designed for these steps and then present the algorithm for optimizing the PARTY model.

3.1 The Deep Model of PARTY

The neural network within PARTY consists of $M + 1$ layers for performing M nonlinear transformations, where M is an even number, the first $M/2$ hidden layers are encoders to learn a set of compact representation (*i.e.*, dimension reduction) and the last $M/2$ layers are decoders to progressively reconstruct the input. For ease of presentation, we first provide the following definitions. Let $\mathbf{h}_i^{(0)} = \mathbf{x}_i \in \mathbb{R}^d$ denote one input sample to the first layer and

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

be the output of the m-th layer, where $m = 1, 2, \cdots, M$ indexes the layer of network and $g(\cdot)$ is a nonlinear activation function, d_m denotes the dimension of the output at the m-th layer, $\mathbf{W}^{(m)} \in \mathbb{R}^{d_m \times d_{m-1}}$ and $\mathbf{b}^{(m)} \in \mathbb{R}^{d_m}$ denote the weights and bias associated with the m-th layer, respectively. Thus, given $\mathbf{x}_i$ as the input of the first layer, $\mathbf{h}_i^{(M)}$ (*i.e.* the output of the top layer) is the reconstruction of $\mathbf{x}_i$, while $\mathbf{h}_i^{(\frac{M}{2})}$ is the desired low-dimensional representation of $\mathbf{x}_i$. Furthermore, for a collection of n given samples $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$, the corresponding outputs of our neural network is denoted by

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

The objective of PARTY is to minimize the data reconstruction error and simultaneously preserve the global sparsity prior $\mathbf{C}$ in the representation learning. With the definitions above, these targets can be formally stated as:

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

where λ_1 and λ_2 are positive tradeoff parameters.

The terms $\{\mathcal{J}_i\}_{i=1}^3$ are designed for different goals. Intuitively, the first term $\mathcal{J}_1$ is designed to consider the locality by minimizing the reconstruction errors *w.r.t.* the input itself. In other words, the input acts as a supervisor to learn a compact representation $\mathbf{H}^{(\frac{M}{2})}$. $\mathcal{J}_2$ is designed based on the so-called manifold assumption [Roweis and Saul, 2000] which states that the reconstruction relation (*e.g.*, the structure prior) is invariant to different feature spaces. In this paper, we mainly consider the sparsity prior in $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_n]$ which is obtained by solving the following problem:

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

where $\|\cdot\|_1$ denotes the ℓ_1-norm that is usually used to achieve sparsity, $\mathbf{c}_{ii}$ is the i-th entry of the column vector $\mathbf{c}_i$, and the constraint avoids the degenerated solution.

With the optimal solution of (5), $\mathcal{J}_2$ guarantees the globality because the reconstruction relation over the whole data set is preserved into the hidden representation. Lastly, $\mathcal{J}_3$ is a regularization term to avoid over-fitting. Noticed that, the objective (4) with nonlinear function $g(\cdot)$ can intrinsically avoid trivial solutions such as $\mathbf{W}^{(m)} = [\mathbf{I}\ \mathbf{O}]$ and $\mathbf{W}^{(m)} = [\mathbf{I}\ \mathbf{O}]^T$ where $\mathbf{I}$ is an identity matrix and $\mathbf{O}$ is an all-zero matrix.

Our neural network model uses the input as self-supervisor to learn compact representation and simultaneously exploits the sparsity prior for ensuring invariance of the underlying manifold (non-linear subspace) structure in the progressively learned representations. The learned representation is fully adaptive to these local and global structures and favorable for the following clustering process.

3.2 Optimization

We now demonstrate how PARTY model can be optimized efficiently via stochastic sub-gradient descent. For convenience of developing the algorithm, we rewrite (4) in the following sample-wise form:

$$\mathcal{J} = \frac{1}{2}\sum_{i=1}^{n}\left(\|\mathbf{x}_i - \mathbf{h}_i^{(M)}\|_2^2 + \lambda_1\|\mathbf{h}_i^{(\frac{M}{2})} - \mathbf{H}^{(\frac{M}{2})}\mathbf{c}_i\|_2^2\right) + \frac{\lambda_2}{2}\sum_{m=1}^{M}\left(\|\mathbf{W}^{(m)}\|_F^2 + \|\mathbf{b}^{(m)}\|_2^2\right), \tag{6}$$

Recall the definition of $\mathbf{h}_i^{(m)}$ in (2). Applying chain rule, we can express the sub-gradients of (6) *w.r.t.* $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ as follows:

$$\frac{\partial\mathcal{J}}{\partial\mathbf{W}^{(m)}} = \left(\mathbf{\Delta}^{(m)} + \lambda_1\mathbf{\Lambda}^{(m)}\right)(\mathbf{h}_i^{(m-1)})^T + \lambda_2\mathbf{W}^{(m)} \tag{7}$$

$$\frac{\partial\mathcal{J}}{\partial\mathbf{b}^{(m)}} = \mathbf{\Delta}^{(m)} + \lambda_1\mathbf{\Lambda}^{(m)} + \lambda_2\mathbf{b}^{(m)}, \tag{8}$$

where $\mathbf{\Delta}^{(m)}$ is defined as

$$\begin{cases} -\left(\mathbf{x}_i - \mathbf{h}_i^{(M)}\right) \odot g'(\mathbf{z}_i^{(M)}), & m = M \\ (\mathbf{W}^{(m+1)})^T\mathbf{\Delta}^{(m+1)} \odot g'(\mathbf{z}_i^{(m)}), & \text{otherwise} \end{cases} \tag{9}$$

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

$$\begin{cases} (\mathbf{W}^{m+1})^T \mathbf{\Lambda}^{(m+1)} \odot g'(\mathbf{z}_i^{(m)}), & m = 1, \cdots, \frac{M-2}{2} \\ \left(\mathbf{h}_i^{(\frac{M}{2})} - \mathbf{H}^{(\frac{M}{2})}\mathbf{c}_i\right) \odot g'(\mathbf{z}_i^{(\frac{M}{2})}), & m = \frac{M}{2} \\ \mathbf{0}, & m = \frac{M+2}{2}, \cdots, M \end{cases} \tag{10}$$

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

Using the stochastic sub-gradient descent algorithm, we update $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^M$ as follows until convergence:

$$\mathbf{W}^{(m)} = \mathbf{W}^{(m)} - \mu \frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(m)}}, \tag{11}$$

and

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

where $\mu > 0$ is the learning rate which is typically set to a small value such as 2^{-10} in our experiments.

Algorithm 1 summarizes the detailed procedure for optimizing PARTY.

Algorithm 1 Deep Subspace Clustering with Sparsity Prior

```
Input: A given data set $\mathbf{X}$, and the tradeoff parameters $\lambda_1$, $\lambda_2$.
Initialize $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^M$, and $\mathbf{H}^0 = \mathbf{X}$.
// Obtain the sparsity prior:
Compute the sparsity prior $\mathbf{C}$ over $\mathbf{X}$ via solving (5).
// Optimization:
for $m = 1, 2 \cdots, M$ do
    Do forward propagation to get $\{\mathbf{H}^{(m)}\}_{m=1}^M$ via (2).
end
while not converge do
    for $i = 1, 2, \cdots, n$ do
        Randomly select a data point $\mathbf{x}_i$ and let $\mathbf{h}_i^0 = \mathbf{x}_i$,
        for $m = 1, 2 \cdots, M$ do
            Compute $\mathbf{h}_i^{(m)}$ via (2).
        end
        for $m = M, M-1 \cdots, 1$ do
            Calculate the gradient using (7)–(10).
        end
        for $m = 1, 2, \cdots, M$ do
            Update $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ via (11)–(12).
        end
    end
end
// Clustering:
Obtain the segmentation of data by clustering based on $\mathbf{H}^{(\frac{M}{2})}$.
Output: $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^M$ and the clustering result.
```

3.3 Implementation Details

In our experiments, we use $g = tanh$ as the activation function which is defined as follows:

$$g(z) = tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}, \tag{13}$$

and the corresponding derivative is calculated by

$$g'(z) = tanh'(z) = 1 - tanh^2(z). \tag{14}$$

Regarding the initialization of $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^M$, we adopt the pre-training and fine-tuning strategy [Hinton and Salakhutdinov, 2006].

3.4 Connection to Prior Works

We here provide two ways to understand our proposed PARTY method. Firstly, PARTY can be treated as a general form of the classical Auto-Encoder. Moreover, with several simplifications, PARTY can be deemed as a variant of spectral clustering (SC) algorithms.

Connection between AE and PARTY:
The AE has been widely used in various applications including clustering [Tian *et al.*, 2014; Ma *et al.*, 2014] which use the input as the supervisor to learn a compact representation and then perform kmeans over the representation to obtain the segmentation of data. The proposed PARTY reduces to the the standard AE if λ_1 in (4) is set as 0, *i.e.* without the sparsity prior. In this sense, PARTY augments AE by considering the valuable relations among different samples (*i.e.* structure prior) and can provide superior performance in the scenario of unsupervised learning as shown in our experiments.

Connection between SC and PARTY:
Most spectral clustering based methods obtain the segmentation of data by performing kmeans on the eigenvectors corresponding to the largest eigenvalues of the Laplacian $\mathbf{L}$ as aforementioned.

Remark 1 *If we adopt $g(z) = z$, $\lambda_1 \to +\infty$, $M = 2$ in (4) and add a mild constraint for avoiding trivial solutions, then our learnt compact representation $\mathbf{H}^\star$ will be the solution of*

$$\min_{\mathbf{H}} \frac{1}{2}\|\mathbf{H} - \mathbf{HC}\|_F^2 \quad \text{s.t. } \mathbf{HH}^T = \mathbf{I}, \tag{15}$$

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

$$\max_{\mathbf{H}} \frac{\mathbf{HLH}^T}{\mathbf{HH}^T}, \tag{16}$$

which is exactly the problem of spectral clustering [Ng et al.*, 2001] but with different choice of $\mathbf{L}$. Specifically, $\mathbf{L} = \mathbf{C} + \mathbf{C}^T - \mathbf{CC}^T$ for our method, whereas $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{AD}^{-\frac{1}{2}}$ ($\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$) for the spectral clustering method.*

4 Experiments

In this section, we compare the proposed PARTY with 13 popular subspace clustering methods on several image data sets in terms of 5 evaluation metrics.

4.1 Experimental Settings

Baseline Algorithms: We compare PARTY with SSC [Elhamifar and Vidal, 2013], LRR [Liu *et al.*, 2013], low rank based subspace clustering (LRSC) [Favaro *et al.*, 2011], least square regression (LSR) [Lu *et al.*, 2012], smooth representation clustering (SMR) [Hu *et al.*, 2014a], kernel SSC (KSSC) [Patel and Vidal, 2014], kernel LRR (KLRR) [Xiao *et al.*, 2015], latent subspace sparse subspace clustering (LS3C) [Patel *et al.*, 2013] and stacked sparse autoencoders (SAE) [Ma *et al.*, 2014; Tian *et al.*, 2014]. Among these methods, both KSSC and KLRR have two versions based on the radial basis function / the polynomial function, which are denoted by KSSC1 / KSSC2 and KLRR1 / KLRR2, respectively. LSR has also two variants which are denoted by LSR1 and LSR2. Moreover, we investigate the performance of SAE with the sigmoid function (SAEg) and the saturating linear transfer function (SAEs). For a fair comparison, we adopt the same ℓ_1-solver (*i.e.* the Homotopy solver [Yang *et al.*, 2010]) to achieve sparsity for SSC, KSSC, and our method.

In all experiments, we train the SAE and our PARTY with five layers at which consists of 300-200-150-200-300 neurons. As PARTY is complementary to existing clustering methods, we investigate the performance of PARTY with two clustering methods for an extensive investigation, *i.e.* kmeans and SSC, where the corresponding methods are denoted by PARTYk and PARTYs. For fair comparisons, we report the best result of all the evaluated methods achieved with their optimal parameters. For our PARTY with the tradeoff parameters, λ_1 and λ_2. We fix $\lambda_2 = 10^{-3}$ for all data sets and experimentally choose λ_1.

Data Sets: We carry our experiments using three real-world data sets, *i.e.* Extended Yale database B (YaleB) [Georghiades *et al.*, 2001], COIL20 image data set [Nene *et al.*, 1996], and the BF0502 data set [Sivic *et al.*, 2009]. The used YaleB consists of 2,414 samples from 38 individuals, where each image is with size of 192×168. The COIL20 data set contains 1,440 samples distributed over 20 objects, where each image is with the size of 32×32. The used BF0502 data set [Xiao *et al.*, 2015] contains 1200 facial images detected from the TV series "Buffy the Vampire Slayer", where each category includes 200 samples.

For a comprehensive study, other than the raw data (*i.e.* gray value), we also extract the following two features from YaleB and COIL20 for evaluations, *i.e.* dense scale-invariant feature transform (DSIFT) [Lowe, 2004] and the histogram of oriented gradients (HOG) [Dalal and Triggs, 2005]. For computational efficiency, we perform PCA to reduce the feature dimension of all data sets to 300.

Evaluation Criteria: We adopt five metrics to evaluate the clustering quality, *i.e.* *Accuracy*, normalized mutual information (*NMI*), the adjusted rand index (*ARI*), *Precision*, and *Fscore*. Higher value of these metrics indicates better performance. For each data set, we repeat each algorithm five times and report the mean and the standard deviation of these metrics.

4.2 Comparison with State of The Art

In this section, we first evaluate the performance of PARTY on two image databases, *i.e.* the COIL20 object images and the YaleB facial images. Tables 1 and 2 report the results which show the superiority of our method. 1) On the COIL20, the Accuracy of PARTYs is 2.36% and 4.42% at least higher than that of the other methods regarding to DISFT and HOG. 2) On the YaleB, PARTYs again achieves the best results, of which the Accuracy is 4.78% and 4.14% higher than the second best method *w.r.t.* these two features. 3) In most cases,

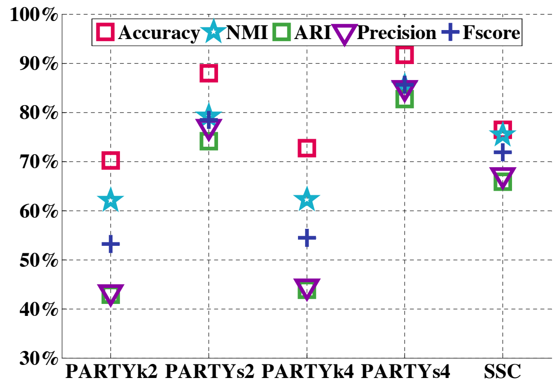


Figure 2: Deep vs. Shallow models on the BF0502 data set. PARTYk2, PARTYs2, PARTYk4, and PARTYs4 correspond to PARTY with kmeans (M=2), with SSC (M=2), with kmeans (M=4), and with SSC (M=4), respectively.

PARTYs outperforms PARTYk, which may attribute to that the former performs subspace clustering in the latent space instead of the original space and the latent space is more discriminative than the original one.

4.3 Deep Model vs. Shallow Models

We investigate the performance of our neural network with different depths on the BF0502 data set. Specifically, we report the performance of PARTY with five layers ($M = 4$) and three layers ($M = 2$), respectively. In the case of $M = 2$, the network is with the neurons of 300-150-300. In the experiment, we fix the parameter λ_1 of PARTY as 10^{-2} and 10^{-3} for $M = 2$ and $M = 4$, respectively. Moreover, we report the result of SSC as a baseline.

Fig. 2 shows that a deeper model will bring a better performance to our method. For example, PARTYs4 is 3.75%, 6.05%, 8.59%, 7.87%,7.11% higher than PARTYs2 regarding to *Accuracy*, *NMI*, *ARI*, *Precision*, and *Fscore*, respectively. Moreover, PARTYs remarkably outperforms SSC and the advantage is more distinct when a deeper network is built.

4.4 Influence of Activation Functions

In this section, we report the performance of our model with four different activation functions on the YaleB data set. The used activation functions includes *tanh*, *sigmoid*, non-saturating sigmoid (*nssigmoid*), and *softplus* which is the smooth version of the rectified linear unit (ReLU) [Krizhevsky *et al.*, 2012]. From Fig. 3, we can see that the *tanh* function outperforms the other three activation functions in the tests and the *nssigmoid* function achieves the second best result which is very close to the best one.

5 Conclusion

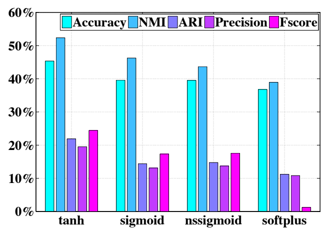
In this paper, we proposed a deep learning based subspace clustering method which simultaneously considers the locality and globality of data set. Extensive experimental results have shown that our method remarkably outperforms 13 state-of-the-art subspace clustering methods in terms of five

Table 1: Performance comparison on the **COIL20** data set. Results in boldface are significantly better than the others, according to the t-test with a significance level at 0.05.

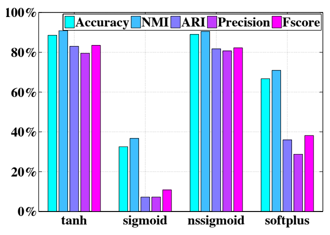
Features	DISIFT					HOG				
Methods	Accuracy	NMI	ARI	Precision	Fscore	Accuracy	NMI	ARI	Precision	Fscore
PARTYk	76.18±2.31	81.62±1.11	68.65±1.80	68.50±2.17	70.24±1.70	72.08±4.09	77.79±2.10	61.08±6.36	59.51±7.89	63.80±5.84
PARTYs	**85.76±4.70**	**91.19±0.89**	**84.80±3.79**	**82.45±5.97**	**85.58±3.54**	**85.50±2.37**	**91.19±0.66**	**81.92±1.18**	**79.12±2.44**	**82.86±1.10**
SAEg	65.36±3.70	77.09±1.57	56.59±3.61	51.47±5.40	59.07±3.28	74.93±2.61	89.26±0.78	74.25±1.75	66.65±2.18	75.70±1.63
SAEs	56.57±4.78	65.06±3.76	39.82±8.23	34.73±8.11	43.65±7.32	71.93±2.37	87.01±1.22	70.43±0.22	63.47±2.16	72.08±2.09
SSC	83.40±2.78	91.04±1.03	82.58±1.97	74.60±2.63	83.54±1.84	81.08±1.10	90.12±0.18	80.20±1.34	72.80±3.85	81.30±1.23
KSSC1	82.42±4.20	90.39±1.55	80.62±4.20	74.86±6.26	81.67±3.95	70.94±0.88	84.08±0.66	66.37±1.46	61.23±2.45	68.22±1.35
KSSC2	76.43±2.17	90.11±0.81	77.05±1.09	69.95±2.38	78.32±1.03	75.17±1.16	86.53±0.65	69.53±1.43	63.63±1.23	71.21±1.35
LS3C	30.98±1.18	49.23±1.06	17.91±1.19	16.20±0.97	23.90±0.98	30.37±1.16	40.50±0.66	16.36±0.54	16.13±0.46	21.98±0.50
LRR	79.08±1.98	89.72±0.59	78.17±1.70	69.63±2.11	79.40±1.59	58.42±1.10	76.91±0.90	39.64±1.96	29.82±2.15	44.05±1.69
KLRR1	70.28±1.27	81.49±0.42	62.69±1.01	59.06±0.66	64.71±0.97	73.72±0.67	81.28±0.51	68.39±0.76	69.52±0.67	69.96±0.72
KLRR2	78.58±1.04	83.88±0.99	72.69±1.99	72.70±2.46	74.07±1.88	74.19±1.32	83.80±0.70	67.27±1.73	62.92±2.24	69.04±1.62
LRSC	71.15±0.98	78.38±1.20	62.13±2.30	61.05±2.14	64.09±2.18	44.03±1.32	57.23±0.77	31.73±0.93	28.96±1.20	35.91±0.79
LSR1	61.54±2.45	71.27±0.94	51.06±2.94	50.05±3.56	53.65±2.74	64.74±1.73	73.01±1.27	52.44±1.96	48.88±2.42	55.09±1.81
LSR2	64.72±2.59	72.78±1.29	54.77±2.93	54.57±3.41	57.11±2.74	61.75±1.43	71.11±1.31	50.17±1.59	46.41±1.77	52.98±1.49
SMR	80.49±2.00	89.42±0.75	78.94±1.37	77.40±2.12	80.01±1.29	74.86±1.57	84.47±1.43	68.72±1.60	64.73±1.61	70.40±1.51

Table 2: Performance comparison on the **YaleB** data set.

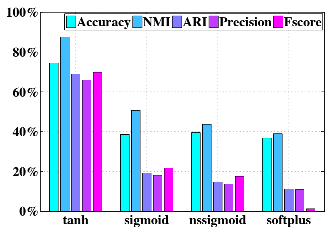
Features	DSIFT					HOG				
Methods	Accuracy	NMI	ARI	Precision	Fscore	Accuracy	NMI	ARI	Precision	Fscore
PARTYk	45.36±2.04	52.36±1.08	21.90±1.11	19.53±0.25	24.43±1.03	74.48±2.92	87.55±1.29	69.00±2.90	65.97±2.78	69.85±2.81
PARTYs	**88.55±1.76**	**90.85±0.23**	**83.00±0.81**	**79.52±1.68**	**83.45±0.78**	**92.08±2.42**	**96.91±0.77**	**90.25±2.85**	**87.07±4.34**	**89.46±2.76**
SAEg	82.30±1.69	87.54±0.47	75.82±1.40	70.90±2.37	76.50±1.35	84.78±3.54	93.43±1.48	82.57±4.61	75.86±6.79	83.07±4.46
SAEs	80.73±1.93	85.92±0.73	73.37±1.77	68.04±2.71	74.12±1.70	81.43±3.77	92.48±1.46	80.45±3.79	73.35±5.11	81.01±3.67
SSC	83.77±1.37	90.02±0.55	78.83±1.10	74.74±1.48	79.42±1.11	85.10±0.98	92.82±0.61	81.62±1.38	76.73±2.23	82.13±1.34
KSSC1	81.44±0.96	89.07±0.41	68.01±1.52	61.25±2.08	68.94±1.47	80.51±1.83	88.66±0.72	70.51±2.77	64.06±4.16	71.37±2.67
KSSC2	77.66±1.77	84.43±0.69	60.55±1.71	52.44±2.14	61.76±1.64	75.39±2.24	80.39±0.87	60.87±1.25	56.81±0.89	61.98±1.22
LS3C	49.90±1.26	59.84±0.78	28.80±0.89	25.76±0.82	31.03±0.86	49.11±0.78	53.53±0.80	17.94±1.20	15.01±1.10	20.91±1.09
LRR	81.62±2.79	89.16±0.78	73.51±2.05	65.94±2.81	74.29±1.98	81.00±2.73	93.01±1.19	78.21±4.81	68.03±6.22	78.87±4.65
KLRR1	69.99±1.08	74.72±0.41	48.23±1.67	42.46±2.28	49.80±1.58	78.91±3.67	86.18±2.16	70.23±3.89	67.10±3.76	71.05±3.78
KLRR2	66.15±0.87	72.37±0.46	46.18±1.67	41.24±2.37	47.80±1.59	60.19±1.25	68.97±0.73	33.77±1.48	27.93±1.83	35.99±1.37
LRSC	68.29±1.42	73.40±1.03	51.33±1.37	47.74±1.57	52.72±1.32	68.68±1.37	73.20±0.75	40.03±3.45	32.99±4.09	42.03±3.25
LSR1	72.86±0.21	77.66±0.24	55.29±1.02	50.53±3.04	56.59±2.00	76.55±0.93	81.05±0.59	55.68±2.32	48.67±3.31	57.03±2.21
LSR2	73.38±1.35	77.42±0.57	54.03±1.02	48.77±1.21	55.38±0.98	76.06±1.11	80.41±0.37	55.47±3.03	49.04±4.70	56.81±2.88
SMR	81.49±2.78	85.29±0.61	68.18±1.31	62.82±1.64	69.08±1.27	87.94±1.24	92.76±0.77	82.84±1.31	79.95±1.18	83.31±1.27



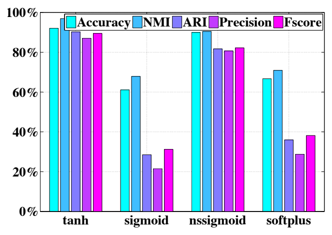
(a) PARTYk on DISIFT



(b) PARTYs on DISIFT



(c) PARTYk on HOG



(d) PARTYs on HOG

Figure 3: The performance of PARTY with four different activation functions on the YaleB data set.

evaluation metrics. Our deep learning based framework is general and not limited to the sparsity prior. We plan to incorporate other structure priors such as low rank in future.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of this paper. X. Peng is supported by A*STAR Industrial Robotics Programme - Distributed Sensing and Perception under SERC grant 1225100002, J. Feng is supported by NUS startup grant R263000C08133, and Z. Yi is supported by National Nature Science Foundation of China under grant No. 61432012.

References

[Belkin and Niyogi, 2003] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, 2003.

[Bengio *et al.*, 2013] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, Aug. 2013.

[Costeira and Kanade, 1998] J. P. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *Int. J. Comput. Vis.*, 29(3):159–179, 1998.

[Dalal and Triggs, 2005] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of 18th IEEE Conf. Comput. Vis. and Pattern Recognit.*, volume 1, pages 886–893 vol. 1, San Diego, CA, Jun. 2005.

[Elhamifar and Vidal, 2013] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2765–2781, 2013.

[Favaro *et al.*, 2011] P. Favaro, R. Vidal, and A Ravichandran. A closed form solution to robust subspace estimation and clustering. In *Proc. of 24th IEEE Conf. Comput. Vis. and Pattern Recognit.*, pages 1801–1807, Colorado Springs, CO, Jun. 2011.

[Feng *et al.*, 2014] J. Feng, Z. C. Lin, H. Xu, and S. C. Yan. Robust subspace segmentation with block-diagonal prior. In *Proc. of 27th IEEE Conf. Comput. Vis. and Pattern Recognit.*, pages 3818–3825, Columbus, OH, Jun. 2014.

[Georghiades *et al.*, 2001] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(6):643–660, 2001.

[He *et al.*, 2015] R. He, Y.Y. Zhang, Z.N Sun, and Q.Y. Yin. Robust subspace clustering with complex noise. *IEEE Trans. on Image Process.*, 24(11):4001–4013, Nov 2015.

[Hinton and Salakhutdinov, 2006] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[Hu *et al.*, 2014a] H. Hu, Z. C. Lin, J.J. Feng, and J. Zhou. Smooth representation clustering. In *Proc. of 27th IEEE Conf. Comput. Vis. and Pattern Recognit.*, pages 3834–3841, Columbus, OH, Jun. 2014.

[Hu *et al.*, 2014b] J. L. Hu, J. W. Lu, and Y. P. Tan. Discriminative deep metric learning for face verification in the wild. In *Proc. of 27th IEEE Conf. Comput. Vis. and Pattern Recognit.*, pages 1875–1882, Columbus, OH, Jun. 2014.

[Krizhevsky *et al.*, 2012] A. Krizhevsky, L. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. of 25th Adv. in Neural Inf. Process. Syst.*, pages 1097–1105, Lake Tahoe, CA, Dec. 2012.

[Liu *et al.*, 2013] G. C. Liu, Z. C. Lin, S. C. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):171–184, 2013.

[Lowe, 2004] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, 2004.

[Lu *et al.*, 2012] C. Y. Lu, H. Min, Z. Q. Zhao, L. Zhu, D. S. Huang, and S. C. Yan. Robust and efficient subspace segmentation via least squares regression. In *Proc. of 12th Eur. Conf. Comput. Vis.*, pages 347–360, Florence, Italy, Oct. 2012.

[Ma *et al.*, 2014] Y. J. Ma, C. Shang, F. Yang, and D. X. Huang. Latent subspace clustering based on deep neural networks. In *Proc. of 5th Int. Symp. on Adv. Control of Ind. Proc.*, Hiroshima, Japan, May 2014.

[Nene *et al.*, 1996] S. A Nene, S. K. Nayar, H. Murase, et al. Columbia object image library (coil-20). Technical report, Technical Report CUCS-005-96, 1996.

[Ng *et al.*, 2001] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proc. of 14th Adv. in Neural Inf. Process. Syst.*, pages 849–856, Vancouver, Canada, Dec. 2001.

[Nie *et al.*, 2011] F. P. Nie, H. Wang, H. Huang, and C. Ding. Unsupervised and semi-supervised learning via l1-norm graph. In *Proc. of 12th IEEE Conf. Comput. Vis.*, pages 2268–2273, Barcelona, Nov. 2011.

[Patel and Vidal, 2014] V. M. Patel and R. Vidal. Kernel sparse subspace clustering. In *Proc. of Int. Conf. on Image Process.*, pages 2849–2853, Paris, Oct. 2014.

[Patel *et al.*, 2013] V. M. Patel, Hien V. N., and R. Vidal. Latent space sparse subspace clustering. In *Proc. of 14th IEEE Conf. Comput. Vis.*, pages 225–232, Sydney, VIC, Dec. 2013.

[Peng *et al.*, 2015] X. Peng, Z. Yi, and H. J. Tang. Robust subspace clustering via thresholding ridge regression. In *Proc. of 29th AAAI Conf. Artif. Intell.*, pages 3827–3833, Austin Texas, USA, Jan. 2015.

[Roweis and Saul, 2000] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[Shi and Malik, 2000] J. B. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

[Sivic *et al.*, 2009] J. Sivic, M. Everingham, and A. Zisserman. Who are you? - learning person specific classifiers from video. In *Proc. of 22th IEEE Conf. Comput. Vis. and Pattern Recognit.*, pages 1145–1152, Miami, FL, Jun. 2009.

[Tian *et al.*, 2014] F. Tian, B. Gao, Q. Cui, E. H. Chen, and T. Y. Liu. Learning deep representations for graph clustering, Jul. 2014.

[Vidal, 2011] R. Vidal. Subspace clustering. *IEEE Signal Proc. Mag.*, 28(2):52–68, 2011.

[Wang *et al.*, 2015] Z. W. Wang, D. Liu, J. C. Yang, W. Han, and T. Huang. Deep networks for image super-resolution with sparse prior. In *Proc. of 15th IEEE Conf. Comput. Vis.*, pages 370–378, Santiago, Chile, Dec. 2015.

[Xiao *et al.*, 2015] S. Xiao, M. Tan, D. Xu, and Z. Y. Dong. Robust kernel low-rank representation. *IEEE Trans. Neural. Netw. Learn. Syst.*, PP(99):1–1, 2015.

[Yang *et al.*, 2010] A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Fast l1-minimization algorithms and an application in robust face recognition: A review. Technical Report UCB/EECS-2010-13, University of California, Berkeley, Feb. 2010.

[Yu *et al.*, 2015] Z. D. Yu, W. Y. Liu, W.B. Liu, X. Peng, Z. Hui, and B.V.K. Vijaya Kumar. Generalized transitive distance with minimum spanning random forest. In *Proc. of 24th Int. Joint Conf. on Artif. Intell.*, pages 2205–2211, Buenos Aires, Argentina, Jul. 2015.

[Yuan *et al.*, 2016] Y. Yuan, J. Lin, and Q. Wang. Dual-clustering-based hyperspectral band selection by contextual analysis. *IEEE Trans. Geosci. Remote Sens.*, 54(3):1431–1445, Mar. 2016.

[Zhang *et al.*, 2015] C. Zhang, H. Fu, S. Liu, G. Liu, and X. Cao. Low-rank tensor constrained multiview subspace clustering. In *Proc. the 15th Int. Conf. on Comput. Vis.*, pages 1582–1590, Santiago, Dec. 2015.