Latent Distribution Preserving Deep Subspace Clustering

Lei Zhou\(^1\), Xiao Bai\(^1\), Dong Wang\(^1\), Xianglong Liu\(^{2,1}\), Jun Zhou\(^3\) and Edwin Hancock\(^{4,1}\)

\(^1\)School of Computer Science and Engineering, Beijing Advanced Innovation Center for Big Data and Brain Computing, Jiangxi Research Institute, Beihang University, Beijing, China
\(^2\)State Key Laboratory of Software Development Environment, Beihang University, Beijing, China
\(^3\)School of Information and Communication Technology, Griffith University, Nathan, Australia
\(^4\)Department of Computer Science, University of York, York, U.K.

\{leizhou, baixiao, dongwang\}@buaa.edu.cn, xlliu@nlsde.buaa.edu.cn, jun.zhou@griffith.edu.au, edwin.hancock@york.ac.uk

Abstract

Subspace clustering is a useful technique for many computer vision applications in which the intrinsic dimension of high-dimensional data is smaller than the ambient dimension. Traditional subspace clustering methods often rely on the self-expressiveness property, which has proven effective for linear subspace clustering. However, they perform unsatisfactorily on real data with complex nonlinear subspaces. More recently, deep autoencoder based subspace clustering methods have achieved success owning to the more powerful representation extracted by the autoencoder network. Unfortunately, these methods only considering the reconstruction of original input data can hardly guarantee the latent representation for the data distributed in subspaces, which inevitably limits the performance in practice. In this paper, we propose a novel deep subspace clustering method based on a latent distribution-preserving autoencoder, which introduces a distribution consistency loss to guide the learning of distribution-preserving latent representation, and consequently enables strong capacity of characterizing the real-world data for subspace clustering. Experimental results on several public databases show that our method achieves significant improvement compared with the state-of-the-art subspace clustering methods.

1 Introduction

In many computer vision applications, such as face recognition [Liu et al., 2013; Zhou et al., 2018a], texture recognition [Lu et al., 2014; Wang et al., 2018] and motion segmentation [Elhamifar and Vidal, 2013], visual data can be well characterized by subspaces. Moreover, the intrinsic dimension of high-dimensional data is often much smaller than the ambient dimension [Vidal, 2011]. This has motivated the development of subspace clustering technique which simultaneously cluster the data into multiple subspaces and find a low-dimensional subspace for each class of data.

Many subspace clustering algorithms have been developed during the past decade, including algebraic [Vidal et al., 2005], iterative [Agarwal and Mustafa, 2004], statistical [Rao et al., 2008], and spectral clustering methods [Elhamifar and Vidal, 2013; Liu et al., 2013; Lu et al., 2012; Patel et al., 2013; Lu et al., 2014; Peng et al., 2016a; Li et al., 2017b; Zhou et al., 2018b]. Among these approaches, spectral clustering methods have been intensively studied thanks to their theoretical soundness and empirical success. These methods are based on the self-expressiveness property of data lying in a union of linear subspaces, which states that each point in a union of subspaces can be written as a linear combination of other data points in the subspaces. Two typical methods are sparse subspace clustering (SSC) [Elhamifar and Vidal, 2013] and low-rank representation (LRR) [Liu et al., 2013]. SSC uses the $\ell_1$ norm to enforce the sparsity of self-representation coefficient matrix. LRR uses nuclear norm minimization to make coefficient matrix low-rank.

Recently, deep neural network based subspace clustering methods [Peng et al., 2016b; Peng et al., 2017; Li et al., 2017a; Ji et al., 2017; Peng et al., 2018; Zhou et al., 2018c] have been proposed to learn better sample representations for subspace clustering with complex structures rather than the linear ones. However, like these conventional shallow methods [Elhamifar and Vidal, 2013; Liu et al., 2013; Favaro et al., 2011; Lu et al., 2012], they still hinge on self-expression as supervision, which may not perform well on data with inconvenient distributions. This is because both these shallow and deep subspace clustering methods only reveal the intrinsic Euclidean structure of data, and do not consider the intrinsic cluster structure which is often the union of some non-linear subspaces.

In this paper, in light of the above arguments, we propose a novel deep subspace clustering method based on a latent distribution-preserving autoencoder, namely Distribution-Preserving Subspace Clustering (DPSC). Motivated by the fact that the data points are drawn from the union of some low-dimensional subspaces embedded in a high-dimensional ambient space, and the subspace structure of each cluster can be described by the distribution of the cluster elements. The key idea of DPSC is to preserve the intrinsic cluster structure of data space by minimizing the inconsistency between the
original data distribution and the latent representation distribution. We first utilize a nonparametric technique, i.e. Kernel Density Estimation (KDE) [Hinneburg and Gabriel, 2007], to estimate the distribution of the data. Then we design a distribution consistency loss by minimizing the Kullback-Leibler (KL) divergence between the original data distribution and the learned latent representation distribution. The architecture of DPSC is shown in Fig. 1(a). It consists of a subspace clustering autoencoder network and a distribution consistency loss term that learns to supervise the distribution of latent representation. As the visualization result shown in Fig. 1(b), our method can better preserve the cluster structure of the original data space by introducing the distribution consistency loss which leads to a better clustering result.

The main contributions of this paper are as follows:

1. We propose a novel distribution-preserving subspace clustering method. By developing a latent distribution-preserving autoencoder, DPSC can preserve the intrinsic cluster structure of data space and supervise the encoder to produce more favorable representation for subspace clustering.

2. We design a simple but effective distribution consistency loss by minimizing the KL divergence between the original data distribution and the learned latent representation distribution, which is complementary to the autoencoder induced self-expression loss.

3. Experimental results on several public databases with different subspace applications show that our method leads to significant improvement compared with the state-of-the-arts on both linear and non-linear subspace clustering problems.

2 Related Works

In this section, we review some works related to the proposed subspace clustering method.

2.1 Traditional Subspace Clustering

Given a data matrix \( \mathbf{X} \) that contains \( N \) data points drawn from \( k \) subspaces \( \{ \mathbf{S}_i \}_{i=1}^k \), SSC [Elhamifar and Vidal, 2013] aims to find a sparse representation matrix \( \mathbf{C} \) showing mutual similarity of the points, i.e., \( \mathbf{X} = \mathbf{XC} \). Since each point in \( \mathbf{S}_i \) can be expressed in terms of the other points in \( \mathbf{S}_i \), such a sparse representation matrix \( \mathbf{C} \) always exists. As pointed out in LRR [Liu et al., 2013], SSC finds the sparsest representation of each data vector individually. There is no global constraint on its solution, so SSC method may be inaccurate when it is used to capture the global structures of data. [Liu et al., 2013] proposed that low rankness can be a more appropriate criterion. SSC and LRR methods solve the robust subspace clustering problem by removing the outliers from the original data space and obtaining a good affinity matrix based on a clean database. Thus they need prior knowledge on the structures of the errors, which usually is unknown in practice. [Peng et al., 2015] proposed a robust subspace clustering method which overcomes this limitation by eliminating the effect of errors from the projection space with a model of thresholding ridge regression (TRR).

2.2 Deep Subspace Clustering

Deep subspace clustering methods have been proposed to solve the linear subspace assumption. [Peng et al., 2017] simultaneously learned a compact representation using a neural network and a clustering assignment by minimizing the discrepancy between pair-wise sample-centers distributions. [Ji et al., 2017] proposed a deep autoencoder framework for subspace clustering by developing a self-expressive layer to mimic the “self-expressiveness” property of a union of subspaces. [Peng et al., 2018] proposed a structured autoencoder which learns a set of transformations to map input data points into nonlinear latent spaces. Most recently, [Zhou et al., 2018] adopted a GAN-alike model to supervise sample representation learning for subspace clustering. Different from these methods, the proposed DPSC introduces distribution consistency loss between the original data distribution...
3 Distribution-Preserving Deep Subspace Clustering

We propose a distribution-preserving subspace clustering (DPSC) method, which can learn more favorable data representations for subspace clustering via a latent distribution-preserving autoencoder. DPSC clusters data in three steps: 1) Initializing the parameters of autoencoder with data reconstruction and latent representations self-expression; 2) Estimating the distributions of original data and learned latent representation; 3) Updating the network parameters by adding the distribution consistency constraint, and clustering the latent data representations into multiple subspaces.

3.1 Subspace Clustering with Self-Expression Loss

Let \( X = \{x_1, \ldots, x_N\} \) denote the input data points and \( Z = \{z_1, \ldots, z_N\} \) denote their corresponding latent representation learned by the encoder. The parameters of encoder and decoder are denoted by \( \Theta_e \) and \( \Theta_d \) respectively. The number of clusters is denoted by \( k \).

DPSC first learns the latent representations for the input data through a traditional autoencoder with the reconstruction loss:

\[
L_r(\Theta_e; \Theta_d) = \|X - \hat{X}\|_F^2
\]

where \( \hat{X} = \Theta_d(\Theta_e(X)) \) denotes the reconstruction of \( X \) by the autoencoder. Then the latent representations \( Z = \Theta_e(X) \).

As done in [Ji et al., 2017], we add a self-expressive layer in the autoencoder to learn better latent representations for subspace clustering. The self-expression loss is defined as:

\[
L_s(C) = \|Z - ZC\|_F^2 + \lambda \|C\|_F^2
\]

where \( C \in \mathbb{R}^{N \times N} \) is the self-expression coefficient matrix. We adopt an F-norm penalty on \( C \), since compared with the non-smooth penalty term, e.g. the \( \ell_1 \) norm, it can be learned more easily while also achieving comparable or even better performance.

With the learned latent representations \( Z_i \), we can use a spectral clustering algorithm on the affinity matrix \( W = \frac{1}{2}(|C| + |C^T|) \) to obtain the subspace clustering result \( c_i (i = 1, \ldots, k) \).

3.2 Distribution-Preserving with Distribution Consistency Loss

To address inconvenient sample distributions, we then design a distribution consistency loss as a complementary unsupervised solution to the self-expression loss.

The proposed DPSC clusters data by simultaneously learning a set of \( k \) cluster centers \( \{c_i\}_{i=1}^k \) in the latent feature space \( Z \) and the parameters \( \Theta_e \) of the encoder that map the data points into \( Z \). Given an initial estimate of the non-linear mapping \( \Theta_e \), \( \Theta_d \) and the initial cluster centroids \( \{c_i\}_{i=1}^k \), we propose to improve the clustering in an unsupervised manner. We first estimate the probability density distribution of the original data space and the latent representational space initialized by the autoencoder. Then a distribution consistency loss between these two distributions is learned to guide the update of the distribution-preserving autoencoder network.

![Figure 2: Clustering accuracy of the proposed method on the MNIST database: (a) different kernel functions; (b) different loss functions.](image-url)

**Kernel Density Estimation**

Minimizing the inconsistency of distributions of data between the original space and the latent representational space is a challenging task because in practice there is little prior information to help us to identify the distribution of the data. To solve this problem, we propose to use kernel density estimation, which is a nonparametric technique to estimate the real distribution of the data.

The kernel density estimation of a given database \( X = \{x_i\}_{i=1}^N \) is:

\[
\hat{f}(x) = \frac{1}{N} \sum_{i=1}^{N} \kappa_H(x - x_i)
\]

where \( H = h^2 I \) is a matrix of smoothness and specifies the width of the kernel around each sample point \( x_i \), \( h \) is the bandwidth of the neighbourhood. \( \kappa(x) \) is the kernel function. The experimental result in Fig. 2(a) shows that the Guassian kernel is better than both the Cauchy kernel and the exponential kernel. Thus the Gaussian kernel is chosen in our method. Then the kernel density estimation of the database \( X = \{x_i\}_{i=1}^N \) is:

\[
\hat{f}(x) = \frac{1}{N} \sum_{i=1}^{N} \|H\|^{-1/2} \kappa(H^{-1/2}(x - x_i))
\]

**Distribution Consistency Loss**

Our aim is to find the mapping function to preserve the distribution of the original database so that \( \hat{f}(x_i) = \hat{f}(\Theta_e(x_i)) \) for each data point in \( X \), where \( \Theta_e(x_i) = z_i \). This can be achieved by minimizing the KL divergence based distribution consistency loss:

\[
L_d(\Theta_e) = \sum_{i=1}^{N} \hat{f}(x_i) \log \frac{\hat{f}(x_i)}{\hat{f}(z_i)}
\]
which is a well-known criterion for describing the dissimilarity between two distributions. Here, \( f(x_i) \) and \( f(z_i) \) are the probability density distributions of \( X = [x_1, \ldots, x_N] \) and \( Z = [z_1, \ldots, z_N] \), respectively. Fig. 2(b) shows the superiority of KL divergence compared with squared error. Clearly, our objective function is proposed to achieve invariance of distribution by minimizing the discrepancy between the target distribution \( f(x) \) and the predicted distribution \( f(z) \).

3.3 The Formulation and Training Strategy
Finally, we obtain the loss function for the DPSC by combining the data reconstruction loss, the self-expression loss and the distribution consistency loss:

\[
L(\Theta_c; C; \Theta_d) = L_r + \lambda_1 L_s + \lambda_2 L_d \tag{6}
\]

The first term of the loss function denotes the reconstruction loss of the autoencoder in Eq. (1). The second term corresponds to self-expression loss in Eq. (2). The last term is the distribution consistency loss as shown in Eq. (5) which preserves the cluster structure between original data space and latent representational space.

Due to the large amount of parameters, it is intractable to directly train the network from scratch. To address this, we design a two-stages training strategy. First we pre-train the deep autoencoder without considering the distribution consistency loss term, i.e., minimizing the reconstruction loss and self-expression loss in Eq. (6) while discarding the last term. In this way, the deep autoencoder can produce a reasonable good initial representation. After that, we estimate the probability density distributions of the original data space and the latent representational space initialized by the deep autoencoder. Then a distribution consistency loss between these two distributions is learned to guide the update of the autoencoder network. We train the whole DPSC network by minimizing the loss function (6) with the Adam algorithm [Kingma and Ba, 2015]. The learning rate is set as \( 1 \times 10^{-3} \) for all experiments.

After the whole network is trained, we can use the parameters of the self-expressive layer, i.e. \( C \), to construct an affinity matrix for spectral clustering. During testing, we perform spectral clustering on the constructed affinity matrix \( W = \frac{1}{2} (C + [C^T]) \). For fairness of comparing with other methods, we use the NCut algorithm as in [Elhamifar and Vidal, 2013; Ji et al., 2017].

4 Experiments
We conducted experiments on three subspace clustering tasks: a) handwritten digit clustering, b) face recognition, and c) object clustering. The first two tasks are relatively easier since handwriting digit and face images approximately lie on a union of linear subspaces. The last task is more challenging with the non-linear subspace representation.

The baseline subspace clustering methods include sparse subspace clustering (SSC) [Elhamifar and Vidal, 2013], kernel SSC (KSSC) [Patel and Vidal, 2014], elastic net subspace clustering (ENSC) [You et al., 2016], efficient dense subspace clustering (EDSC) [Ji et al., 2014], low-rank representation (LRR) [Liu et al., 2013], low-rank subspace clustering (LRSC) [Favaro et al., 2011], deep subspace clustering network (DSC-Net) [Ji et al., 2017], structured autoencoder (StructAE) [Peng et al., 2018], deep adversarial subspace clustering (DASC) [Zhou et al., 2018c] and SSC with the pre-trained convolutional autoencoder features (AE+SSC). Among these methods, AE+SSC only uses the features from pre-trained autoencoder without self-expression layer and distribution consistency loss. DSC-Net adds self-expression layer in autoencoder but has no distribution consistency loss. These two methods can better show the superiority of our latent distribution-preserving autoencoder with distribution consistency loss.

We adopted the following widely used clustering metrics to measure the clustering performance: accuracy (ACC), normalized mutual information (NMI) and the purity (PUR).

4.1 Handwritten Digit Clustering
We test the proposed method on handwritten digit clustering using the MNIST database [Lecun et al., 1998]. This database contains 10 clusters, including handwritten digits 0-9 as shown in Fig. 3(a). Each cluster contains 6,000 images for training and 1,000 images for testing, with a size of 28x28 pixels in each image. We randomly selected 1,000 images from each digit for our experiment. We fixed the number of clusters \( k = 10 \) and chose different numbers of data points for each cluster. Each cluster contained \( N_i \) data points randomly chosen from the corresponding 1,000 images, where \( N_i \in \{100, 500, 1000\} \), so that the number of total points \( N \in \{1000, 5000, 10000\} \). For DPSC, we set the bandwidth \( h = 2 \), \( \lambda = \lambda_1 = 1 \) and \( \lambda_2 = 2 \). Then we applied all methods on this database for comparison.
Table 1 shows the clustering results on MNIST. Here DPSC outperforms the baselines in all three metrics given different numbers of data points. Specifically, when each cluster contains 1,000 data points, our method outperforms the second best method DASC by 3.32%, 3.08% and 3.12% in terms of ACC, NMI and PUR, respectively. Moreover, DPSC achieves a significant improvement over the shallow subspace clustering methods, e.g., SSC and LRR. This is because compared with shallow methods, DPSC uses a multi-layer convolutional autoencoder as the feature extractor. Therefore, DPSC can better handle translation, rotation and shift in the handwritten images while mapping the input data into a union of linear subspaces.

### 4.2 Face Recognition

Since subspaces are commonly used to capture the appearance of faces under varying illuminations, we also test the performance of our method on face clustering with the CMU PIE database [Sim et al., 2001]. The CMU PIE database contains 41,368 images of 68 people under 13 different poses, 43 different illumination conditions, and 4 different expressions as shown in Fig. 3(b). In our experiment, we used the face images in five near frontal poses (P05, P07, P09, P27, P29).

Then each person had 170 face images under different illuminations and expressions. Each image was manually cropped and normalized to a size of $32 \times 32$ pixels. We randomly picked $k \in \{5, 10, 20\}$ individuals to investigate the performance of all methods. For our method, we set the bandwidth $h = 2$, $\lambda = \lambda_1 = 1$ and $\lambda_2 = 2$.

Table 2 reports the clustering results on the CMU PIE face database. It can be observed that DPSC consistently outperforms the baselines on all three metrics. For $k = \{5, 10, 20\}$ individuals, DPSC improves the performance by 2.15%, 2.36% and 2.14% over the second best method DASC on ACC. For both NMI and PUR metrics, respectively, DPSC also brings about 3% improvement over the state-of-the-art DASC. All these results clearly prove the superior effectiveness and robustness of DPSC.

These results also clearly demonstrate that deep clustering methods perform much better than the shallow ones, benefiting from integrating representation learning with self-expression learning. The deep autoencoder extracts more powerful representations and the following self-expression layer enforces the representations to favorably locate in a union of linear subspaces, effectively getting rid of strict linear
Table 3: Clustering results on COIL-20.

Table 4: Clustering results on COIL-100.

Figure 4: (a) The clustering accuracy of our DPSC with different bandwidth $h$. (b) The NMI of our DPSC with different bandwidth $h$.

4.3 Object Clustering

We further evaluated DPSC on the challenging object clustering task using the COIL-20 and COIL-100 [Nene et al., 1996] databases which provide various objects as shown in Fig. 3 (c). COIL-20 has 1,440 toy images from 20 classes, and COIL-100 contains 7,200 images of 100 objects. In both databases, each object was taken with poses varying at an interval of 5 degrees, producing a total of 72 images per object. This implies that the images are not distributed in a union of linear subspaces and thus are more challenging. In contrast with the previous human face databases, in which faces are well aligned and have similar structures, the object images from COIL-20 and COIL-100 are more diverse. Samples from the same object differ from each other due to the change of viewing angle, introducing additional challenge for subspace clustering techniques. For these databases, we downsample the images to $32 \times 32$ and set the bandwidth $h = 3$, $\lambda = 1$ and $\lambda_2 = 2$.

Table 3 and table 4 depict the ACC, NMI and PUR of different methods on clustering 20 classes for COIL-20 and 100 classes for COIL-100, respectively. Note that, in both cases and metrics, our DPSC achieves the best performance. In particular, for COIL-100, our method obtains an accuracy of 75.40%, which improves 3.25% over the best-performing baseline DASC.

4.4 Parameter Sensitivity

In our DPSC, there are four hyper-parameters $\lambda$, $\lambda_1$, $\lambda_2$ and $h$. Experiments show that the balancing parameters $\lambda$, $\lambda_1$ and $\lambda_2$ have slight influence to the performance. We set them by several attempts. Due to the limited space, we do not show the varying results with different balancing parameters. We focus on the effect of $h$, bandwidth of the kernel density estimator, which is important to the proposed method. Here, we report the clustering accuracy and NMI on four databases with different bandwidth $h$. Fig. 4 shows that our DPSC is insensitive with different $h$ in a certain range.

5 Conclusion

In this paper, we propose a novel distribution-preserving deep subspace clustering (DPSC) method. The distribution consistency loss of DPSC guarantees that the posterior latent representation distribution matches the prior original data space distribution and preserves the cluster structure of high-dimensional data space. This solves the inconvenient subspace distribution conditions. Extensive experiments on MNIST, CMU PIE and COIL-20/100 show the superiority of DPSC on both linear and non-linear subspace clustering problems over state-of-the-arts. In the future work, it is interesting to further investigate the cluster structure of complex sample distributions and try different distribution-preserving strategies.

Acknowledgments

This work was supported by the National Natural Science Foundation of China project no. 61772057 and the support funding from State Key Lab. of Software Development Environment and Jiangxi and Qingdao Research Institute of Beihang University.
References


