The AAAI-17 Workshop on
Crowdsourcing, Deep Learning, and Artificial Intelligence Agents
WS-17-07

Unsupervised Multi-Manifold Clustering by Learning Deep Representation

Dongdong Chen and **Jiancheng Lv*** and **Zhang Yi**
Machine Intelligence Laboratory
College of Computer Science, Sichuan University
Chengdu 610065, P. R. China,
dongdongchen.scu@gmail.com, {lvjiancheng, zhangyi}@scu.edu.cn

Abstract

In this paper, we propose a novel deep manifold clustering (DMC) method for learning effective deep representations and partitioning a dataset into clusters where each cluster contains data points from a single nonlinear manifold. Different from other previous research efforts, we adopt deep neural network to classify and parameterize unlabeled data which lie on multiple manifolds. Firstly, motivated by the observation that nearby points lie on the local of manifold should possess similar representations, a locality preserving objective is defined to iteratively explore data relation and learn structure preserving representations. Secondly, by finding the corresponding cluster centers from the representations, a clustering-oriented objective is then proposed to guide the model to extract both discriminative and cluster-specific representations. Finally, by integrating two objectives into a single model with a unified cost function and optimizing it by using back propagation, we can obtain not only more powerful representations, but also more precise clusters of data. In addition, our model can be intuitively extended to cluster out-of-sample datum. The experimental results and comparisons with existing state-of-the-art methods show that the proposed method consistently achieves the best performance on various benchmark datasets.

Introduction

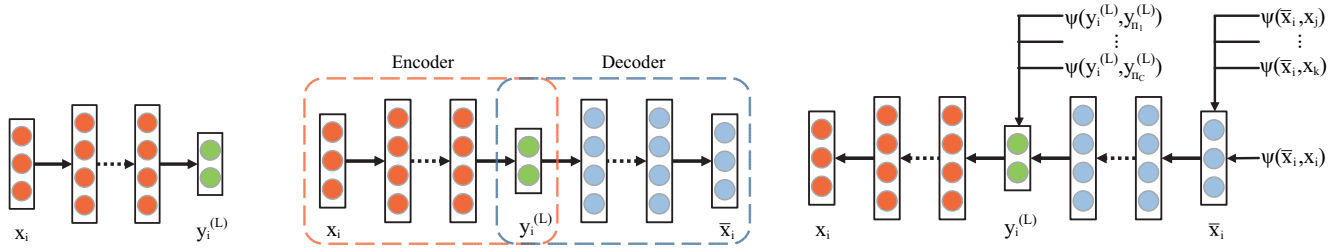
In many real-world problems, the data lie in multiple manifolds of possibly dimensions (Seung and Lee 2000). As a very challenging topic in machine learning, multi-manifold clustering (MMC) which regarding clusters as groups of points around compact manifolds, has been realized as a promising generalization of traditional clustering (Souvenir and Pless 2005).

Clustering unlabeled data, which lie on multiple low-dimensional manifolds, is well studied and numerous algorithms have been proposed (Souvenir and Pless 2005; Elhamifar and Vidal 2011). Generally, MMC starts by learning a manifold structure preserved representation, then build a similarity graph to performing spectral clustering and embedding (Yan et al. 2007). The key to such algorithms is to learn a local or global geometry preserved presentation which can characterize the non-linear structure of data. Then, based on different learnt representation, different similarity graphs of data are built to performing spectral clustering. Among these algorithms, (Polito and Perona 2002) computes a local linear representation of data, and clusters data into different clusters and computes a local linearity preserving embedding of data by solving an eigenvalue decomposition problem. (Elhamifar and Vidal 2011) solves a sparse optimization problem to determine the local neighborhood of data and build a sparse affinity graph followed by spectral clustering for embedding. (Wang et al. 2011) trains some mixture models to approximate the local tangent space at each data, then a spectral method is conducted on the affinity matrix to find the clusters.

Although the traditional MMC approaches have attained better performances than traditional clustering methods (e.g. K-means) in many scenarios, most of these methods suffer from the following three limitations. First, these algorithms are computationally expensive because it involves an eigenvalue decomposition (EVD). The complexity of a straightforward implementation of EVD is cubic to the number of nodes in the graph, while even the fastest implementation (Chen and Cai 2011) requires a super-quadratic computational complexity. Second, they can not cluster the out-of-sample data that are not used to construct the similarity graph. To group each out-of-sample data, the methods have to recalculate the similarity graph and the corresponding clusters. Third, it is not easy to select proper neighborhood of a data point in practical applications (Chen, Lv, and Yi 2014). Most of the algorithms determine the neighborhood in the original input space, and the neighborhood is fixed during manifold learning. However, the geodesic nearest neighbor on a manifold may not be the Euclidian nearest neighbor in the original space, which lead to the invalidation of such neighborhood determination.

Recently, deep learning, or called as deep neural network (DNN), has received increasing interest in computer vision and machine learning (Bengio, Courville, and Vincent 2013). Generally, deep learning aims to learn hierarchical feature representations by constructing high-level features from low-level features (Bengio, Courville, and Vincent 2013; Lv, Zhang, and Li 2015). One of the strategies that make training deep architectures possible and effective is the greedy layerwised unsupervised pretraining, such as restricted boltzmann machines (RBMs) (Hinton and Salakhut-

*Corresponding author: Jiancheng Lv



(a) DMC network (b) Unrolling (c) Fine-tuning

Figure 1: An illustration of the proposed DMC method. (a) The network architecture which has one input layer and L hidden layers, left layer represents the input data $\mathbf{x}$, and $\mathbf{y}^{(L)}$ denotes the deep representation learned by L-th hidden layer. (b) The network is firstly pre-trained successively as L RBMs, and then unrolled to create a deep autoencoder, the encoder and decoder are marked as two dashed rectangles (c) Unsupervised fine-tuning using back propagation of error derivatives.

dinov 2006) and autoencoder (AE) (Bengio et al. 2007). This strategy aims to learn useful representations one layer at a time, and then to set the output features to be the input of the next layer. By stacking these non-linear single layers together, deep learning are believed to yield better representations (Bengio, Courville, and Vincent 2013). Many attempts, such as a deep belief network (DBN) (Hinton, Osindero, and Teh 2006), deep auto-encoder (DAE) (Bengio 2009) and convolutional neural networks (CNN) (Lcun et al. 1998), have been made to apply deep learning in feature learning, image analysis and speech recognition (Dahl et al. 2012; Collobert et al. 2011).

More recently, a number of works have explored combing clustering with deep learning. (Chandra, Kumar, and Jawahar 2013) uses K component RBMs to learn K non-linear subspaces in the raw image space and a data point is assigned with the component RBM which best reconstruct it. (Tian et al. 2014) use stacked autoencoder to learn a representation of the affinity graph, and then run K-means in the learned representations to obtain clusters, which improves performance but at a cost of increasing memory consumption. (Chen 2015) uses DBN to learn representations, and then conducted a nonparametric maximum margin clustering upon the output of DBN. (Xie, Girshick, and Farhadi 2015) train a deep network by iteratively minimizing a Kullback-Leibler (KL) divergence between a centroid-based probability distribution and an auxiliary target distribution. A more recent work on the combination of clustering and deep learning is found in (Yang, Parikh, and Batra 2016), where the authors combined agglomerative clustering with CNN and formulate them as a recurrent process. However, to our knowledge, the adoption of deep learning in multi-manifold clustering has not been adequately investigated yet. The goal of this work is to conduct some investigations along this direction.

In this paper, we propose a deep learning based framework, called as deep manifold clustering (DMC), to solving unsupervised MMC problem. It is done by iteratively minimizing a joint cost function which consists of two objective functions. Firstly, motivated by the observation of nearby points lie on the local of manifold should possess similar representations, a locality preserving objective is minimized to iteratively explores data relation and learn structure preserving representations. Secondly, we consider a clustering-oriented constraint to guide the model to extract both discriminative and cluster-specific representations. In particular, the cluster centers can be determined by directly conducting K-means or density analysis (Rodriguez and Laio 2014) on the deep representations. The proposed method is illustrated in Figure 1. To the best of our knowledge, DMC is the first unsupervised deep learning framework that jointly learns structure preserving representations and clusters data into multi-manifolds. The model is end-to-end and simple to implement. Experimental results on various benchmark datasets demonstrate the effective of the proposed method.

Deep Manifold Clustering

Problem Formulation

Given a set of points $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^n$ derived from C non-linear manifolds, where n is the number of data points and m presents the dimensionality, the MMC problem can be formulated as follows: (**1**) learn a nonlinear mapping $F : \mathbf{x}_i \in \mathbb{R}^m \rightarrow \mathbf{y}_i \in \mathbb{R}^d$, where $d \ll m$ and $\mathbf{y}_i$ is the meaningful representation (embedding) of $\mathbf{x}_i$. Here, a meaningful representation means that the structure information of manifold should be preserved in the representation space. (**2**) output the set of labels $\{I_i\}_{i=1}^n$ where $I_i \in \{1, 2, \cdots, C\}$ is an index specifying to which output manifold a data point belongs.

To learn a meaningful representation of data, DNN is a natural choice due to their powerful representation capabilities (Bengio, Courville, and Vincent 2013) and theoretical function approximation properties to parameterize F (Hornik 1991). In this work, we devote to solving MMC by using DNN. We start by introducing the architecture of our model followed by detailing both the clustering and training processes.

Architecture

As depicted in Figure 1, DMC is a deep feedforward network which has one input layer and L hidden layers. We

first initialize the network weights by using RBMs (details in the last of this section), and then unrolled it to create a deep autoencoder which consists of two parts: an encoder and decoder (1(b)). Both the encoder and decoder have $L+1$ layers, with a shared deep hidden layer. Therefore, there are $2L+1$ layers in this deep autoencoder, and such network can be formulated as follows,

$$\begin{cases} \mathbf{y}^{(l)} & = f\left(\mathbf{z}^{(l)}\right); \\ \mathbf{z}^{(l)} & = \mathbf{W}^{(l)}\mathbf{y}^{(l-1)}; \\ \mathbf{y}^{(0)} & = \mathbf{x}; \end{cases} \tag{1}$$

where $\mathbf{z}^{(l)}$ and $\mathbf{y}^{(l)}$ are the input and representation (activation) of layer l, respectively, $l = 1, 2, \cdots, 2L$, $\mathbf{W}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ is the weight matrix[1] that connects layers l and $l-1$, $f(\cdot)$ is an element-wise nonlinear activation function, such as sigmoid and tanh.

Let $\theta = \{\mathbf{W}^{(l)}\}_{l=1}^{L}$ and $\theta' = \{\mathbf{W}^{(l)}\}_{l=L+1}^{2L}$ denote the parameters of encoder and decoder, respectively. Given an input $\mathbf{x}$, its deep representation $\mathbf{y}^{(L)}$ and reconstruction $\bar{\mathbf{x}}$ can be subsequently formulated as follows:

$$\mathbf{y}^{(L)} = F_\theta(\mathbf{x}) = f(\mathbf{W}^{(L)} f(\mathbf{W}^{(L-1)} \cdots f(\mathbf{W}^{(1)}\mathbf{x}))), \tag{2}$$

$$\bar{\mathbf{x}} = F_{\theta'}(\mathbf{y}^{(L)}) = f(\mathbf{W}^{(2L)} f(\mathbf{W}^{(2L-1)} \cdots f(\mathbf{W}^{(L+1)}\mathbf{y}^{(L)}))). \tag{3}$$

where F_θ and $F_{\theta'}$ represent the nonlinear transformations learned by encoder and decoder, respectively.

Cost Function

Locality preserving objective To make the learned representations meaningful, they are expected to be embedded into their intrinsic manifold, which preserves their local property of manifold (Roweis and Saul 2000). Generally, the locality of manifold can be interpreted as similar inputs should have similar representations. That is, a data point can be recovered using the representation of its nearby point than using that of non-nearby point, i.e., $\mathbf{x}_i \sim \mathbf{x}_j \rightarrow F_{\theta'}(\mathbf{y}_i) \sim \mathbf{x}_j \rightarrow \bar{\mathbf{x}}_i \sim \mathbf{x}_j$. Based on the observation, a locality preserving objective is defined as follows:

$$\min_{\theta \cup \theta'} (1-\alpha)\Psi(\bar{\mathbf{x}}_i, \mathbf{x}_i) + \frac{\alpha}{k} \sum_{j \in \mathcal{K}_i} \Psi(\bar{\mathbf{x}}_i, \mathbf{x}_j), \tag{4}$$

where $\alpha \in [0, 1]$ balances the importance between the reconstruction of $\mathbf{x}_i$ itself and its local neighborhood, $\Psi(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2$ is the error function, $\mathcal{K}_i$ is the indices set of k nearest neighbors of $\mathbf{x}_i$. In particular, in order to recover a reasonable local neighborhoods, $\mathcal{K}_i$ is not keep fixed but iteratively updated according to the deep hidden representations $\{\mathbf{y}_i^{(L)}\}_{i=1}^n$ derived from DMC, which is totally different from the traditional MMC approaches in the determination of local neighborhood.

[1]To simplify the notation, we ignore the bias terms of the neural network

Finding of centers Motivated by the fact that the centers of multiple cluster are mostly surrounded by nearby points with lower local density and that they are at a relatively large distance from any points with a higher local density (Rodriguez and Laio 2014), we can determine the centers of manifold by finding of density peaks of representations.

To approximate the density of representations, we compute two metrics: its local density ρ_i and its distance ξ_i from representations of higher density. Firstly, the local density ρ_i of a representation $\mathbf{y}_i$ is defined as

$$\rho_i = \sum_{j \in \Omega \setminus \{i\}} e^{-\frac{\Delta_{ij}}{\hat{\Delta}}}, \tag{5}$$

where $\Omega = \{i\}_{i=1}^n$, Δ_{ij} is the distance between $\mathbf{y}_i^{(L)}$ and $\mathbf{y}_j^{(L)}$, $\hat{\Delta}$ is a cutoff distance.

The representations $\{\mathbf{y}_i^{(L)}\}_{i=1}^n$ are then sorted in order of decreasing density, and denote $\{\lambda_i\}_{i=1}^n$ with $\rho_{\lambda_1} \geq \rho_{\lambda_2} \geq \cdots \geq \rho_{\lambda_n}$. The distance metric ξ_i is then defined as follows:

$$\xi_{\lambda_i} = \begin{cases} \min\limits_{\substack{\lambda_j \\ j<i}} \{\Delta_{\lambda_i \lambda_j}\}, & i \geq 2; \\ \max\limits_{j \geq 2} \{\xi_{\lambda_j}\}, & i = 1. \end{cases} \tag{6}$$

The centers are then recognized as representations for which both the values of ρ_i and ξ_i are anomalously large. We assume the number of desired clusters C is given to us as is standard in clustering. For the convenience of research, we define $\gamma_i = \rho_i \xi_i$ and sort it in descending order, and define $\pi = \{\pi_i\}_{i=1}^n$ with $\gamma_{\pi_1} \geq \gamma_{\pi_2} \geq \cdots \geq \gamma_{\pi_n}$. Therefore, the centers can be determined by the C biggest γ, i.e.,

$$\mathcal{Y}_\pi = \{\mathbf{y}_{\pi_c}^{(L)} | \pi_c \in \pi, c = 1, 2, \cdots, C\}. \tag{7}$$

Note that there are many strategies to determine the centers, such as K-means and the density analysis aforementioned. Both of them can obtain the centers set which can be formulated by Eqs. (7), the main difference between them is that the centers given by the later are always the points in the dataset while the former can not grantees that.

Clustering-oriented objective Now we discuss how to guide DMC to extract discriminative features and learn cluster-specific representation given the recognized centers. We consider an objective function that penalizes representations based on their proximity to each centers. That is, a representation $\mathbf{y}_i^{(L)}$ that are closer to center $\mathbf{y}_j^{(L)}$ get lower penalty than to centers that are farther away, where $j \in \pi$. We thus consider the following weighted least squares fitting optimization,

$$\min_\theta \sum_{c=1}^{C} S_{ic} \Psi(\mathbf{y}_i^{(L)}, \mathbf{y}_{\pi_c}^{(L)}) \tag{8}$$

where S_{ic} is the coefficient of which should be lager if $\mathbf{y}_i^{(L)}$ is closer to $\mathbf{y}_{\pi_c}^{(L)}$. We compute S_{ic} as

$$S_{ic} = \frac{\varphi_\sigma(\mathbf{y}_i^{(L)}, \mathbf{y}_{\pi_c}^{(L)})}{\sum\limits_{c' \in \pi} \varphi_\sigma(\mathbf{y}_i^{(L)}, \mathbf{y}_{\pi_{c'}}^{(L)})} \tag{9}$$

where $\varphi_\sigma(\cdot)$ is a kernel function with a bandwidth σ. Here we use the Gaussian kernel $\varphi_\sigma(\mathbf{y}_i^{(L)}, \mathbf{y}_j^{(L)}) = \exp(-\|\mathbf{y}_i^{(L)} - \mathbf{y}_j^{(L)}\|^2/2\sigma)$ which is one of the most commonly used.

Joint cost function By applying the objective in Eq.(4) and Eq.(8) on each data point from $\mathbf{X}$, we formulate the following joint optimization problem for our DMC model:

$$\begin{aligned}\min_{\theta\cup\theta'} \mathcal{J} &= \mathcal{J}_1 + \mathcal{J}_2 \\ &= \sum_{i=1}^{n}\left((1-\alpha)\Psi(\mathbf{x}_i, \bar{\mathbf{x}}_i) + \frac{\alpha}{k}\sum_{j\in\mathcal{K}_i}\Psi(\mathbf{x}_j, \bar{\mathbf{x}}_i)\right) \\ &+ \beta\sum_{i=1}^{n}\sum_{c=1}^{C} S_{ic}\Psi(\mathbf{y}_i^{(L)}, \mathbf{y}_{\pi_c}^{(L)})\end{aligned} \tag{10}$$

where $\mathcal{J}_1$ is defined on the reconstruction layer, which serves to encourage DMC to reconstructs the input effectively and simultaneously preserves the manifold locality, and $\mathcal{J}_2$ is defined on the deep hidden layer to ensure the learned representations are discriminative and cluster-specific, β is a parameter to balance the contribution of $\mathcal{J}_1$ and $\mathcal{J}_2$.

Training

To minimize the cost function in Eqs. (10), we use the stochastic gradient descent (SGD) to obtain the parameters $\{\mathbf{W}^{(l)}\}_{l=1}^{2L}$. The gradient of the cost function $\mathcal{J}$ with respect to the parameters $\mathbf{W}^{(l)}$ can be computed as follows:

$$\frac{\partial\mathcal{J}}{\partial\mathbf{W}^{(l)}} = \delta_i^{(l+1)}(\mathbf{a}_i^{(l)})^\top, \tag{11}$$

where $l = 1, 2, \cdots, 2L$ and $\delta_i^{(l)}$ for each layer can be computed using following updating equations.

$$\delta_i^{(l)} = \begin{cases} \dot{f}(\mathbf{z}_i^{(l)}) \otimes (\bar{\mathbf{x}}_i - \mathbf{x}_i + \alpha\Lambda_i), & l = 2L; \\ \dot{f}(\mathbf{z}_i^{(l)}) \otimes ((\mathbf{W}^{(l)})^\top\delta_i^{(l+1)} + \beta\Theta_i), & l = L; \\ \dot{f}(\mathbf{z}_i^{(l)}) \otimes ((\mathbf{W}^{(l)})^\top\delta_i^{(l+1)}), & otherwise; \end{cases} \tag{12}$$

where $\otimes$ denotes element-wise multiplication, $\dot{f}(\cdot)$ is the derivative of the activation function, $\Lambda_i = \mathbf{x}_i - \frac{1}{k}\sum_{j\in\mathcal{K}_i}\mathbf{x}_j$ and $\Theta_i = \mathbf{y}_i^{(L)} - \sum_{c=1}^{C} S_{ic}\mathbf{y}_{\pi_c}^{(L)}$.

Then, $\mathbf{W}^{(l)}$ can be updated using the following gradient descent algorithm until convergence:

$$\mathbf{W}^{(l)} = \mathbf{W}^{(l)} - \mu\frac{\partial\mathcal{J}}{\partial\mathbf{W}^{(l)}} \tag{13}$$

where μ is the learning rate and the joint optimization of θ and $\mathcal{Y}_\pi$ is summarized in Algorithm 1.

Clustering

Given data points, we first trained the DMC to obtain the transformed representations, and then conduct traditional K-means algorithm to perform clustering. In addition, our DMC model can be intuitively extended to cluster out-of-sample datum. Motivated by collaborative representation based out-of-sample clustering (Peng et al. 2015), we compute the linear coding of an out-of-sample data $\mathbf{x}_t \notin \mathbf{X}$ by solving

$$\min_{\mathbf{u}_j} \|\mathbf{y}_t^{(L)} - \mathbf{Y}\mathbf{u}_j\|^2 + \tau\|\mathbf{u}_j\|^2, \tag{14}$$

where $\mathbf{y}_t^{(L)}$ is the representation of $\mathbf{x}_t$, $\mathbf{Y} = [\mathbf{y}_i^{(L)}|\mathbf{x}_i \in \mathbf{X}]_{i=1}^n$ and $\tau > 0$ is a tuning parameter. The point $\mathbf{x}_t$ is then assigned to the cluster which produce the minimal linear reconstruction residual, i.e.,

$$I_t = \arg\min_c \|\mathbf{y}_t^{(L)} - \mathbf{Y}\mathbf{u}_j^{(c)}\|^2 \tag{15}$$

where the vector $\mathbf{u}_j^{(c)}$ includes the nonzero elements in $\mathbf{u}_j$ associating with the c-th cluster.

Algorithm 1: Joint Optimization on θ and $\mathcal{Y}_\pi$

Input: Training set $\{\mathbf{x}_i\}_{i=1}^n$; network layer number L, learning rate μ, parameter α, β and k.
Output: Network weights $\{\mathbf{W}^{(l)}\}_{l=1}^L$.
Step 1 (Initialization):
Pre-train $\{\mathbf{W}^{(l)}\}_{l=1}^L$ using RBMs.
Step 2 (Optimization by back prorogation):
while *not convegence* **do**
  for $l = 1, 2, \cdots, L$ **do**
    Compute $\{\mathbf{y}_i\}_{i=1}^n$ according to (2);
    Determine neighbors $\{\mathcal{K}_i\}_{i=1}^n$ from $\{\mathbf{y}_i\}_{i=1}^n$;
    Update centers $\mathcal{Y}_\pi$ according to (7);
  end
  for $l = L+1, 2, \cdots, 2L$ **do**
    Compute $\{\bar{\mathbf{x}}_i\}_{i=1}^n$ according to (3);
  end
  for $l = 1, 2, \cdots, 2L$ **do**
    Calculate the gradient according to (11)-(12);
  end
  for $l = 2L, 2L-1, \cdots, 1$ **do**
    Update $\mathbf{W}^{(l)}$ according to (13);
  end
end

Initialization

We initialize DMC with a stacked RBMs. An RBMs is a generative undirected graphical model with a bipartite structure of two sets of binary stochastic nodes termed as the visible $\{v_i\}_{i=1}^{n_v}$ and the hidden layer units $\{h_i\}_{i=1}^{n_h}$, The two layers are connected by $\mathbf{W} \in \mathbb{R}^{n_v\times n_h}$, the activation probabilites of units in each layer are computed by:

$$\begin{cases} P(h_j = 1|\mathbf{v}) &= f(\sum_{i=1}^{n_v} w_{ij}v_i), \\ P(v_i = 1|\mathbf{h}) &= f(\sum_{j=1}^{n_h} w_{ij}h_j), \end{cases} \tag{16}$$

where f is the sigmoid activation function.

We train RBMs by minimizing the contrastive divergence objective (CD-1, (Hinton 2002)). In particular, we use Gaussian visible units (Welling, Rosen-Zvi, and Hinton 2004) to model distribution of real-valued input data.

Table 1: Clustering Results (NMI, %) on different datasets.

Algorithm	COIL-20	MNIST	ExtYaleB
NWMDS (2005)	67.3 ± 0.3	42.8 ± 0.1	52.2 ± 0.7
LSA (2006)	68.2 ± 2.1	35.3 ± 1.8	64.3 ± 3.5
SSC (2009)	81.0 ± 1.8	68.4 ± 0.3	83.6 ± 2.0
MUMC (2010)	71.4 ± 2.8	55.1 ± 3.0	64.7 ± 1.6
SMCE (2011)	83.3 ± 1.1	72.0 ± 0.6	80.6 ± 1.2
LRSC (2011)	71.5 ± 3.4	51.8 ± 2.2	75.2 ± 1.8
LRR (2013)	78.6 ± 0.4	46.2 ± 0.9	79.4 ± 0.9
SDC (2014)	66.7 ± 0.1	51.5 ± 0.0	81.4 ± 0.1
GSAE (2014)	73.6 ± 1.3	74.2 ± 1.5	63.5 ± 2.8
DMC($L = 2$)	84.7 ± 0.7	79.5 ± 1.1	85.7 ± 1.6
DMC($L = 4$)	$\mathbf{89.1 \pm 0.8}$	$\mathbf{86.4 \pm 1.3}$	$\mathbf{87.6 \pm 0.5}$

Model Analysis

Thus far we have discussed how DMC proceeds given estimates of the network parameters θ and the clusters center $\mathcal{Y}_\pi$. Now we discuss the the connections between DMC and four prior works: Principal Component Analysis (PCA) (Jolliffe 1986), Denoising AutoEncoder (dAE) ((Vincent et al. 2008)) and Landmark-based Spectral Clustering (LSC) (Chen and Cai 2011).

Connection to PCA Suppose the activation function is linear, let $\mathcal{J} = \mathcal{J}_1$ and $\alpha = 0$, then a hidden layer with d units actually learns a projection of the input on the subspace spanned by the first d principal components of the data.

Connection to dAE If the hidden layer is nonlinear and $\alpha = 1$, the criteria in (4) can be regraded as an average of k dAEs. That is, for an input $\mathbf{x}_i$, its corrupted version is determined by its neighbors in $\mathcal{K}_i$ rather than by assigning zero values to some randomly selected dimensions of $\mathbf{x}_i$.

Connection to LSC LSC start by producing p landmark points (centers) using K-means, then constructs a affinity matrix $\mathbf{S} \in \mathbb{R}^{p \times n}$ between data points and landmark points, finally computes the representation by performing spectral clustering on the normalized affinity graph matrix. That is to say, our DMC is related to LSC to some extend, except nonlinear transformation implemented by DNN.

Note that, if the hidden layer is nonlinear and we use the cost function incorporating $\mathcal{J}_1$ and $\mathcal{J}_2$ (i.e., (10)), and both the values of α and β are non-zero, the DMC behaves differently from PCA, dAE and LSC with the ability to capture multi-modal aspects of the input distribution. This departure from prior works becomes even more important when we consider stacking multiple layers ($L > 1$) to build a deep representation learning scheme, and thus our DMC is capable of learning very complex multiple nonlinear manifolds.

Experimental results

Experimental Setup

In our experiments, all the neural networks use an element-wise sigmoid function, to activate each layer during the DMC training phase, the neighborhood size k is set to 10, the parameters $\alpha = 10^{-2}$, $\beta = 10^{-3}$, the learning rate $\mu = 10^{-3}$.

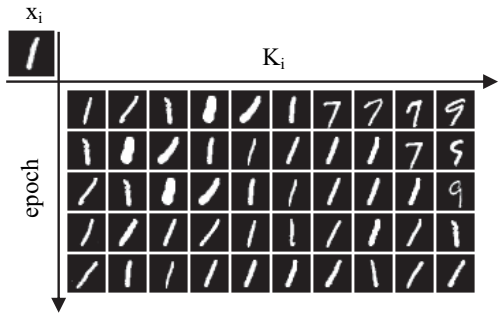


Figure 2: The covergence of local neighborhood $\mathcal{K}_i$ during the iterative optimization. Each row represents the 10 nearest neighbors determined from deep representations derived from DMC in one training epoch.

Baselines and Evaluation Metrics

We compare our DMC method with nine state-of-the-art multi-manifold clustering methods, including Node-Weighted Multi-Dimensional Scaling (NWMDS) (Souvenir and Pless 2005), local subspace affinity (LSA) (Yan and Pollefeys 2006), Sparse Subspace Clustering (SSC) (Elhamifar and Vidal 2009), MUlti-Manifold Cluster (MUMC) (Wang et al. 2010), Sparse Manifold Clustering and Embedding (SMCE) (Elhamifar and Vidal 2011), Low-Rank Subspace Clustering (LRSC) (Favaro, Vidal, and Ravichandran 2011), Low-Rank Representation (LRR) (Liu et al. 2013), Spatial-Density based Clustering (SDC) (Rodriguez and Laio 2014) and Graph based Stacked AutoEncoder (GSAE) (Tian et al. 2014).

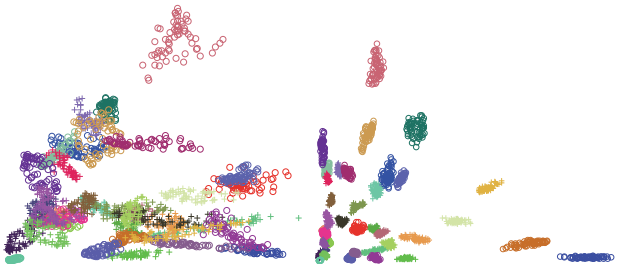
We evaluate the quality of the competing algorithm via Normalized Mutual Information (NMI) (Cai, He, and Han 2005). The range of NMI values is $[0, 1]$. The higher the NMI , the better the corresponding clustering results.

Datasets

We conduct our experiments on three widely used benchmark datasets including the COIL-20 (Nene et al. 1996), Extended Yale database B (ExtYaleB) (Georghiades, Belhumeur, and Kriegman 2001) and MNIST (LeCun and Cortes 2010). The used COIL-20 dataset includes 20 objects. There are 72 images of each object in different positions, and the size of each image is 192×168 pixels. The ExtYaleB contains $2,414$ frontal face images of 38 individuals, with an image size of 192×168 pixels. The used MNIST dataset includes 60,000 handwritten digits samples used for training and 2000 samples for testing, with an image size of 28×28 binarized grayscale pixels. For all datasets, the input was normalized between 0 and 1.

Results and Analysis

Multi-Manifold Clustering We first compare our DMC model with nine state-of-the-art MMC approaches. The hidden layers for DMC ($L = 4$) and DMC $L = 2$ are with structure of 1000-500-250-10 and 1000-10, respectively. Table 1 report the average clustering results and standard deviations of different clustering methods on all three datasets.



(a) epoch=20 (b) epoch=300

Figure 3: 2D visualization of the learned deep representations at different training epoches on ExtYaleB dataset. (a) and (b) present representations learnt by DMC at beginning stage and final stage, respectively.

We clearly see that our DMC achieves higher clustering accuracy than all the other compared state-of-the-art methods on all the three datasets. Compared to those unsupervised MMC methods such as LSA, SSC, LRR and LRSC, our DMC can extract more precise object-specific information in the learned deep networks. Compared deep learning method, i.e., GSAE, our DMC could explicitly addresses the nonlinear separation problem by learning locality preserved and clustering oriented nonlinear transformations, so that more discriminative, nonlinear, and cluster specific information can be exploited to improve the clustering performance.

In addition, we also explore the ability of the DMC in more challenging local manifold structure discovery (LMSD). By following the tricks proposed in (Wang et al. 2014), we evaluate the LMSD performance on MNIST dataset by checking the changes of the local neighborhood $\mathcal{K}_i$ of a input data point $\mathbf{x}_i$. Figure 2 shows the changes of $\mathcal{K}_i$ of a digit 1's 10 nearest neighbors with different training epochs, i.e., $epoch = 1, 5, 10, 30, 50$, respectively. As can be seen, $\mathcal{K}_i$ converges to the manifold where it belongs to.

Structure Preserving Emebdding We test our DMC in terms of its ability to learn the structure preserving representation (manifold embedding). We visualize the 2D manifold clustering results of DMC on the ExtYaleB dataset. For visualization, we map the face images into a two-dimensional deep representation space ($L = 4$) learned from DMC. Our DMC is with layers of size (192×168)-1000-500-250-2. The 2D manifolds are shown in Figure 3. The data points of the 38 persons are shown with different colors and symbols. As can be seen, by iteratively minimizing the joint cost function that incorporates the reconstruction error and the proposed structure preserving criteria, the proposed DMC causes data points of different samples to form more distinctive coordinates and results in more distinguishable representations of different clusters.

Power of Deep Architecture We are interested in knowing whether the depth of DMC has a effect on the clustering performance. We test our DMC with different number of layers, i.e., $L = 4, 3, 2$, on the tree datasets. We report the results of SMCE as baseline. Figure 4 shows the results. As

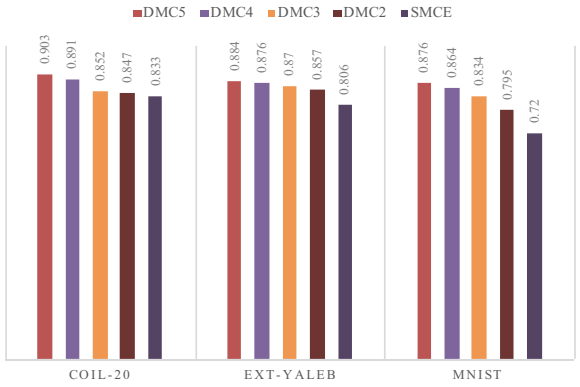


Figure 4: Power of deep architecture. DMC5, DMC4, DMC3 and DMC2 correspond to DMC with $L = 4, 3, 2, 1$, respectively.

can be seen, the performance of our method increases when the depth go to deeper. Moreover, DMC with different number of layers always performs better than SMCE. Therefore, we can conclude that the deep architectures play an important role in extracting meaningful representation to solving MMC problems.

Conclusion

In this paper, we propose a novel deep manifold clustering method which adopt deep neural network to classify and parameterize unlabeled data which lie on multiple nonlinear manifolds. By integrating a locality-preserving objective and a clustering-oriented objective into a single model with a unified cost function and optimizing it by using back propagation, we can obtain not only more powerful representations, but also more precise clusters of data. The model is end to end and easy to implement. The experimental results demonstrate the effective of the proposed method.

Acknowledgments

This work was supported by the National Science Foundation of China (Grant No. 61375065 and 61625204), partially supported by the State Key Program of National Science Foundation of China (Grant No. 61432012 and 61432014).

References

Bengio, Y.; Lamblin, P.; Popovici, D.; Larochelle, H.; et al. 2007. Greedy layer-wise training of deep networks. *NIPS* 19:153.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 35(8):1798–1828.

Bengio, Y. 2009. Learning deep architectures for ai. *Foundations and trends® in Machine Learning* 2(1):1–127.

Cai, D.; He, X.; and Han, J. 2005. Document clustering using locality preserving indexing. *IEEE Transactions*

on Knowledge and Data Engineering (TKDE) 17(12):1624–1637.

Chandra, S.; Kumar, S.; and Jawahar, C. 2013. Learning multiple non-linear sub-spaces using k-rbms. In *CVPR*, 2778–2785. IEEE.

Chen, X., and Cai, D. 2011. Large scale spectral clustering with landmark-based representation. In *AAAI*.

Chen, D.; Lv, J. C.; and Yi, Z. 2014. A local non-negative pursuit method for intrinsic manifold structure preservation. In *AAAI*, volume 3, 1745–1751.

Chen, G. 2015. Deep learning with nonparametric clustering. *Computer Science*.

Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. Natural language processing (almost) from scratch. *JMLR* 12:2493–2537.

Dahl, G. E.; Yu, D.; Deng, L.; and Acero, A. 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on* 20(1):30–42.

Elhamifar, E., and Vidal, R. 2009. Sparse subspace clustering. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 2790–2797.

Elhamifar, E., and Vidal, R. 2011. Sparse manifold clustering and embedding. *Advances in Neural Information Processing Systems (NIPS)* 24:55–63.

Favaro, P.; Vidal, R.; and Ravichandran, A. 2011. A closed form solution to robust subspace estimation and clustering. 1801–1807. IEEE.

Georghiades, A. S.; Belhumeur, P. N.; and Kriegman, D. J. 2001. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 23(6):643–660.

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

Hinton, G.; Osindero, S.; and Teh, Y.-W. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18(7):1527–1554.

Hinton, G. 2002. Training products of experts by minimizing contrastive divergence. *Neural computation* 14(8):1771–1800.

Hornik, K. 1991. Approximation capabilities of multilayer feedforward networks. *Neural Networks* 4(2):251–257.

Jolliffe, I. T. 1986. *Principal component analysis*, volume 487. Springer-Verlag New York.

LeCun, Y., and Cortes, C. 2010. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*.

Liu, G.; Lin, Z.; Yan, S.; Sun, J.; Yu, Y.; and Ma, Y. 2013. Robust recovery of subspace structures by low-rank representation. *IEEE TPAMI* 35(1):171–184.

Lv, J. C.; Zhang, Y.; and Li, Y. 2015. Non-divergence of stochastic discrete time algorithms for pca neural networks. *IEEE Transactions on Neural Networks & Learning Systems* 26(2):394–9.

Lcun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

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

Peng, X.; Tang, H.; Zhang, L.; Yi, Z.; and Xiao, S. 2015. A unified framework for representation-based subspace clustering of out-of-sample and large-scale data. *IEEE Transactions on Neural Networks and Learning Systems*.

Polito, M., and Perona, P. 2002. Grouping and dimensionality reduction by locally linear embedding. *NIPS* 14:1255–1262.

Rodriguez, A., and Laio, A. 2014. Machine learning. clustering by fast search and find of density peaks. *Science* 344(6191):1492–6.

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

Seung, H. S., and Lee, D. D. 2000. The manifold ways of perception. *Science* 290(5500):2268–2269.

Souvenir, R., and Pless, R. 2005. Manifold clustering. In *ICCV*, 648–653 Vol. 1.

Tian, F.; Gao, B.; Cui, Q.; Chen, E.; and Liu, T.-Y. 2014. Learning deep representations for graph clustering. In *AAAI*, 1293–1299.

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P.-A. 2008. Extracting and composing robust features with denoising autoencoders. In *ICML*, 1096–1103. ACM.

Wang, Y.; Jiang, Y.; Wu, Y.; and Zhou, Z.-H. 2010. *Multi-manifold Clustering*. Springer Berlin Heidelberg.

Wang, Y.; Jiang, Y.; Wu, Y.; and Zhou, Z.-H. 2011. Spectral clustering on multiple manifolds. *IEEE Transactions on Neural Networks (TNN)* 22(7):1149–1161.

Wang, W.; Huang, Y.; Wang, Y.; and Wang, L. 2014. Generalized autoencoder: A neural network framework for dimensionality reduction. In *CVPR*, 496–503.

Welling, M.; Rosen-Zvi, M.; and Hinton, G. E. 2004. Exponential family harmoniums with an application to information retrieval. In *NIPS*, 1481–1488.

Xie, J.; Girshick, R.; and Farhadi, A. 2015. Unsupervised deep embedding for clustering analysis. *Computer Science*.

Yan, J., and Pollefeys, M. 2006. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *ECCV*. Springer. 94–106.

Yan, S.; Xu, D.; Zhang, B.; Zhang, H.-J.; Yang, Q.; and Lin, S. 2007. Graph embedding and extensions: a general framework for dimensionality reduction. *IEEE TPAMI* 29(1):40–51.

Yang, J.; Parikh, D.; and Batra, D. 2016. Joint unsupervised learning of deep representations and image clusters. 5147–5156.