Neural Processing Letters (2021) 53:1–16
https://doi.org/10.1007/s11063-020-10375-9

Exploring Implicit and Explicit Geometrical Structure of Data for Deep Embedded Clustering

Xiaofei Zhu[1] · **Khoi Duy Do**[2] · **Jiafeng Guo**[3] · **Jun Xu**[4] · **Stefan Dietze**[5]

Accepted: 9 October 2020 / Published online: 19 October 2020

Abstract

Clustering is an essential data analysis technique and has been studied extensively over the last decades. Previous studies have shown that data representation and data structure information are two critical factors for improving clustering performance, and it forms two important lines of research. The first line of research attempts to learn representative features, especially utilizing the deep neural networks, for handling clustering problems. The second concerns exploiting the geometric structure information within data for clustering. Although both of them have achieved promising performance in lots of clustering tasks, few efforts have been dedicated to combine them in a unified deep clustering framework, which is the research gap we aim to bridge in this work. In this paper, we propose a novel approach, Manifold regularized Deep Embedded Clustering (MDEC), to deal with the aforementioned challenge. It simultaneously models data generating distribution, cluster assignment consistency, as well as geometric structure of data in a unified framework. The proposed method can be optimized by performing mini-batch stochastic gradient descent and back-propagation. We evaluate MDEC on three real-world datasets (USPS, REUTERS-10K, and MNIST), where experimental results demonstrate that our model outperforms baseline models and obtains the state-of-the-art performance.

Keywords Deep neural networks · Stacked autoencoder · Manifold constraint · Clustering

✉ Xiaofei Zhu
zxf@cqut.edu.cn

[1] College of Computer Science and Engineering, Chongqing University of Technology, Chongqing 400054, China

[2] L3S Research Center, Leibniz University of Hannover, 30167 Hannover, Germany

[3] Institute of Computing Technology, Chinese Academy of Science, Beijing 100190, China

[4] School of Information, Renmin University of China, Beijing 100872, China

[5] Knowledge Technologies for the Social Sciences, Leibniz Institute for the Social Sciences, 50667 Cologne, Germany

1 Introduction

Clustering has attracted much attention from a variety of communities, especially in scenarios where data is easily accessible and effective data analysis techniques are critical in real applications. Conventional clustering methods, such as k-means [3,13], Gaussian mixture models (GMM) [5] , and spectral clustering [27,35], aim at grouping similar patterns based on hand-crafted features. However, when the dimensionality of data is high, these methods would lead to unsatisfactory results. To tackle this issue, a number of dimensionality reduction methods have been proposed, such as Principle Component Analysis (PCA) and Latent Semantic Indexing (LSI) [7]. One shortcoming of these methods is that the reduced representation might be ineffective due to their shallow learning framework.

In recent years, inspired by the powerful representation learning ability of deep learning and its successful applications in many fields, such as face recognition [11,15,25], image classification [2,38,40], and recommender system [8,36,37], clustering algorithms based on deep neural networks have been developed, which are also referred to as *Deep Clustering* [12]. For example, Yang et al. [32] perform agglomerative clustering based on current representations to obtain clustering results, and update the representations based on the latest clustering results. Xie et al. [30] employ a stacked autoencoder (SAE) to get initial feature representations, and then use the auxiliary target distribution as supervisory signal for optimization. Guo et al. [12] further propose to simultaneously maintain the reconstruction constraint as well as auxiliary clustering constraint in a unified framework. Jabi et al. [16] conduct theoretical analysis of existing state-of-the-art models. They prove that, for the standard logistic regression posteriors, maximizing the L_2 regularized mutual information via the alternating direction method is equivalent to a soft and regularized K-means loss.

Although aforementioned deep learning-based approaches have achieved encouraging performance in many clustering tasks, the main focus of these works is to learn features for clustering while largely overlooks the rich structure information of data. Recently, a number of studies [1,29] have demonstrated that real-world data are usually sampled from a low-dimensional manifold embedded in a high-dimensional ambient space, and verify the effectiveness for clustering via integrating the underlying geometric structure of data into shallow learning models (e.g., Non-negative Matrix Factorization).

Motivated by these works, we attempt to exploit the intrinsic geometric structure of data and integrate it into the deep clustering framework. In particular, we propose a novel method, called *Manifold Regularized Deep Embedded Clustering (MDEC)*, which jointly models data generating distribution, cluster assignment consistency, as well as geometric structure of data in a unified framework. The overall architecture of our proposed model is illustrated in Fig. 1. The loss function of MDEC contains three components, including a reconstruction constraint to capture the data generating distribution, a clustering constraint to keep the cluster assignment consistency, and a manifold constraint to preserve the geometric structure of data. The proposed MDEC approach can be optimized by performing mini-batch stochastic gradient descent and back-propagation. To demonstrate the effectiveness of our approach, we conduct extensive experiments on three widely used datasets: MNIST, USPS and REUTERS-10K. Experimental results show that the proposed method outperforms all baseline methods and achieves the state-of-the-art performance on all datasets.

The main contributions of this work are summarized as follows:

(1) We exploit the underlying geometric structure of data and seamlessly incorporate it for deep clustering. To the best of our knowledge, this work is the first effort towards this

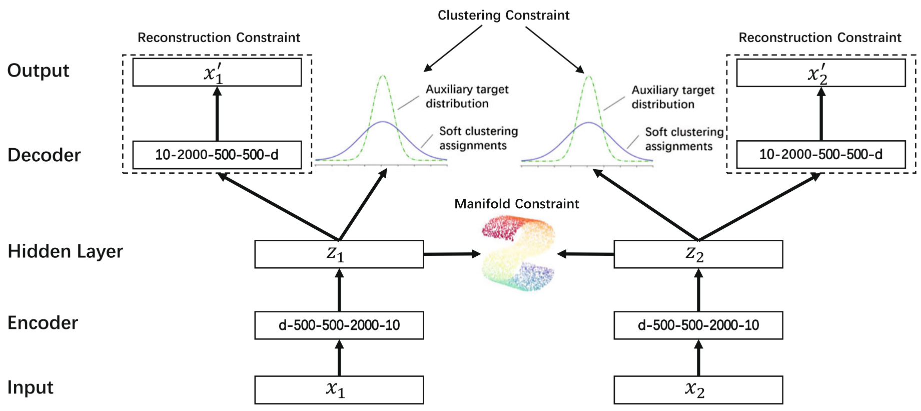


Fig. 1 The proposed architecture of MDEC

target in the deep clustering scenario. Our study shows that exploiting the geometry properties of the data can substantial improve the clustering performance.

(2) We propose a novel deep clustering algorithm, which is an end-to-end learning framework by simultaneously modeling data generating distribution, cluster assignment consistency, as well as geometric structure of data.

(3) Extensive experiments on three widely used datasets show that the proposed approach outperforms all baseline methods and achieves the state-of-the-art performance. The rest of the paper is organized as follows: In Sect. 2, we briefly review the related work, including manifold preservation and deep clustering; In Sect. 3, we describe our proposed approach in detail; Sect. 4 presents the experimental results on three real-world datasets. Finally, we conclude this paper in Sect. 5.

2 Related Work

In this section, we present a brief description of some existing studies on manifold preservation and deep clustering, respectively.

Manifold Preservation Most conventional clustering algorithms, such as k-means [3,13], Gaussian mixture model (GMM) [5], and spectral clustering [27,35] assume the data lies in an Euclidean space. However, recent studies [9,31] have shown that many real world datasets reside on a low-dimensional manifold which is embedded in a high-dimensional ambient space. Applying conventional clustering algorithms on this kind of data will likely result in suboptimal performance.

To address this issue, a number of algorithms have been proposed in the hope of preserving the intrinsic manifold structure for better clustering results. For example, Liu et al. [20] apply a local consistency regularizer, which assumes that similar observations should have similar conditional probability distributions, to revise the objective function of the GMM-based clustering methods. Zheng et al. [39] propose to incorporate manifold constraint into the objective function of sparse coding to obtain powerful sparse representations for clustering. Cai et al. [1] utilize the intrinsic geometry of the data distribution and adopt it as an additional regularization term in the standard Non-negative Matrix Factorization (NMF) algorithm. Wu et al. [29] employ a L_{21} norm in the objective function NMF to measure the quality of factorization and utilize the geometric structure of data to preserve local invariance. Xu et al. [31] introduce manifold regularizations into a zero-shot learning framework based on matrix tri-factorization in order to capture the geometric structure residing in both visual and semantic spaces. Ye et al. [34] address the multi-manifold clustering problem, and they assume that each sub-manifold is a probability distribution defined in the manifold space with a deep neural network. Inspired by the observations, in this work, we consider to leverage the underlying geometric structure of data as a supervisory signal and seamlessly integrating it into the deep clustering framework.

Deep Clustering In recent years, deep learning has been widely used in clustering problems due to its strong capability to exploit the unknown structure in the data space for learning good representations. We can roughly group these works into two categories. The first category includes two-stage methods, which first conduct feature learning based on deep neural networks, and then employ traditional clustering methods to obtain final results. For example, Tian et al. [26] use a stacked autoencoder to learn a non-linear embedding of the original graph, and then conduct k-means to get clustering results. Chen et al. [4] train a DBN for feature representation and then apply a non-parametric maximum-margin clustering over the new representation to cluster data. Peng et al. [24] model data generating distribution information by minimizing the reconstruction error of the input data and incorporate a global structure prior into the deep neural networks for learning reliable representations. After that, traditional clustering algorithms, such as k-means, will be used to obtain clustering results.

In contrast, the second category includes methods which jointly conduct feature learning and clustering in a unified framework. For an instance, Wang et al. [28] explore the possibility to combine the sparse coding domain expertise into deep learning framework for clustering. The proposed model is jointly optimized by a task-specific clustering-oriented loss functions from end to end. Yang et al. [32] utilize intermediate clustering results as supervisory signals to guide representation learning, and in turn, the learned representations will be leveraged for obtaining more reliable clustering results. These two alternative steps will be implemented until a stopping criterion is reached. Xie et al. [30] propose a method, named Deep Embedded Clustering (DEC), which simultaneously learns feature representations and cluster assignments using deep neural networks. It uses a stacked autoencoder to pre-train initial feature representations, and then fine-tune them by utilizing a cluster-oriented loss. Although DEC has achieved promising results, it ignores the reconstruction loss which may distort the distribution of the data during the training procedure. To deal with this problem, Guo et al. [12] further propose to simultaneously consider reconstruction loss and clustering loss in a unified framework, which achieves the state-of-the-art clustering performance.

Our work falls into the second category. Since deep clustering still remains an open issue, we believe it will significantly boost the clustering performance by combining the geometric structure of data and the merits of previous works to develop new deep clustering models. Different to previous research, we explore the geometric structure of data and seamlessly integrate it into the deep embedded clustering framework. It is worth noting that the information of the data manifold structure is substantially different from that of the data locality information as used in IDEC. The latter focuses on the property of data in a point-wise manner, while the former relies on the intrinsic geometric relationships among data points and captures the data property in a pair-wise manner.

3 Proposed Approach

In this section, we first elaborate on the formulation of the objective function for the proposed approach, Manifold regularized Deep Embedded Clustering (MDEC). Then we discuss the optimization procedure of MDEC.

3.1 The General Framework of MDEC

The main idea behind our approach MDEC is shown by the diagram in Fig. 1. It contains three essential parts: reconstruction constraint, clustering constraint, and manifold constraint. The reconstruction constraint, which is derived from a stacked autoencoder, is used to preserve the data generating distribution. The clustering constraint, which is based on an auxiliary target distribution as proposed in [30], is utilized to maintain the cluster assignment consistency. The manifold constraint is used to constrain the learned feature representations to preserve the underlying geometric structure of data.

3.2 Reconstruction Loss

The reconstruction loss L_r is handled by employing a stacked autoencoder, which aims to learn a compressed representation of the input data. Generally, a stacked autoencoder contains two parts: an encoder function $z_i = f_\theta(x_i)$ to map the input x_i to a hidden representation z_i and a decoder function $x_{i'} = g_{\theta'}(z_i)$ to reconstruct the input x_i from z_i. Formally, the reconstruction loss is measured by Mean Square Error (MSE):

$$L_r = \sum_{i=1}^{n} ||x_i - g_{\theta'}(z_i)||_2^2 \tag{1}$$

where θ and θ' are the parameters of the encoder and decoder, respectively.

3.3 Clustering Loss

The clustering loss is first proposed by Xie et al. [30], and is designed to minimize the matching discrepancy between the soft cluster assignment and the auxiliary target distribution. Following Xie et al. [30], we measure the soft cluster assignment based on the similarity between embedded point z_i and the cluster centroid μ_j by using the Student's t-distribution [22]:

$$q_{ij} = \frac{(1 + ||z_i - \mu_j||_2^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'}(1 + ||z_i - \mu_{j'}||_2^2/\alpha)^{-\frac{\alpha+1}{2}}} \tag{2}$$

where α is the degree of freedom of the Student's t-distribution, which is set to 1 as suggested in [30].

Then, the auxiliary target distribution p_i is derived by manipulating the obtained soft cluster assignment to strengthen high confidence predictions. It can be formalized as follows:

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_{j'}(q_{ij'}^2 / \sum_i q_{ij'})} \tag{3}$$

Subsequently, the clustering loss can be formally defined by the KL divergence between the soft cluster assignment q_i and the auxiliary target distribution p_i:

$$L_c = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{4}$$

3.4 Manifold Loss

The goal of incorporating the manifold regularizer term into the objective function is to learn embeddings for the input data which can vary smoothly along the geodesics on the manifold. In order to exploit the underlying geometric structure of data, we resort to constructing a graph, such as anchor graph, which describes similarities between data points and landmarks [21] or a k-nearest neighbor graph. In this work, we focus on adopting the k-nearest neighbor graph, because it has been successfully used in several recent studies [1,33] to capture the intrinsic geometric structure of data.

Suppose there are n data points, represented as: $X = \{x_i\}_{i=1}^n$ where x_i is the feature vector of the i-th data point. $Z = \{z_i\}_{i=1}^n$ denotes the embedded points extracted through an encoder mapping $z_i = f_\theta(x_i)$. Considering a graph with n data points, a weight matrix $W = [w_{ij}]_{n\times n}$ on the graph is defined as follows:

$$w_{ij} = \begin{cases} 1 & \text{if } x_j \in N_i \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where N_i denotes the k nearest neighbors of x_i.

With the weight matrix W defined above, we use the following term to measure the smoothness of the embedded representations.

$$L_m = \frac{1}{k} \sum_{i=1}^{n} \sum_{j \in N_i} W_{ij} \phi(z_i, z_j) \tag{6}$$

where $\phi(z_i, z_j)$ is a measurement of the distance between the two embedded representations z_i and z_j. It is worth noting that there are many options to define $\phi(\cdot, \cdot)$, and some commonly used distance measurements are Euclidean distance and cosine distance. In this paper, we leverage the cosine distance, which has shown superior performance in many applications like information retrieval and text mining, to measure the discrepancy between z_i and z_j, which is formalized as follows:

$$\phi(z_i, z_j) = 1 - cos(z_i, z_j) \tag{7}$$

$$= 1 - \frac{z_i^T z_j}{||z_i||_2 ||z_j||_2} \tag{8}$$

We have also implemented our approach based on the euclidean distance, and the results were considerably worse than that of adopting cosine distance. By minimizing L_m, we expect that if two data points x_i and x_j are close in the original data space, their corresponding embedded representations z_i and z_j should also be close to each other.

3.5 Objective Function

MDEC seamlessly integrates these three parts into a unified end-to-end framework, and each part can provide useful complementary information for other parts during the training procedure. The objective function of MDEC can be expressed as

$$L = L_r + \delta L_m + \gamma L_c \tag{9}$$

where L_r, L_m and L_c are reconstruction loss, manifold loss, and clustering loss, respectively. δ and γ are the weights to control the trade-off among these three terms.

It is worth noting that the manifold constraint is different to the reconstruction constraint for modeling the property of data. In particular, the reconstruction constraint attempts to model data property in a point-wise manner, i.e., reconstructing each data point via capturing data generating distribution. However, the manifold constraint tends to capture pair-wise relationships among data points via exploiting geometric structure of data. Hence, incorporating the manifold constraint is considered to be complementary to the reconstruction constraint. Our experimental results also verify the effectiveness of exploiting the geometric structure of data.

3.6 Optimization

In our MDEC model, the parameters for learning contain the encoder parameters θ, the decoder parameters θ' and the cluster centroids $\{\mu_j\}_{j=1}^{K}$. We optimize Eq. (9) using a mini-batch stochastic gradient descent and back-propagation. Specifically, in each iteration we sample a mini-batch of data points from the whole dataset, and then conduct learning based on the sampled set. The gradients of L_m with respect to z_i are computed as:

$$\frac{\partial L_m}{\partial z_i} = -\frac{1}{k} \sum_{j \in N_i} W_{ij} \left(\frac{z_j}{||z_i||_2 \cdot ||z_j||_2} \right) - \left(\frac{z_i cos(z_i, z_j)}{||z_i||_2^2} \right), \tag{10}$$

and the gradients of L_c with respect to z_i and μ_j are computed as:

$$\frac{\partial L_c}{\partial z_i} = \frac{\alpha+1}{\alpha} \sum_{j=1}^{K} \left(1 + \frac{||z_i - \mu_j||_2^2}{\alpha} \right)^{-1} \times (p_{ij} - q_{ij})(z_i - \mu_j) \tag{11}$$

$$\frac{\partial L_c}{\partial \mu_j} = \frac{\alpha+1}{\alpha} \sum_{i=1}^{n} \left(1 + \frac{||z_i - \mu_j||_2^2}{\alpha} \right)^{-1} \times (p_{ij} - q_{ij})(\mu_j - z_i) \tag{12}$$

We can then update μ_j with

$$\mu_j = \mu_j - \frac{\lambda}{m} \sum_{i=1}^{m} \frac{\partial L_c}{\partial \mu_j} \tag{13}$$

and use the chain rule to update the encoder parameters θ and the decoder parameters θ' by:

$$\theta = \theta - \frac{\lambda}{m} \sum_{i=1}^{m} \left(\frac{\partial L_r}{\partial \theta} + \delta \frac{\partial L_m}{\partial \theta} + \gamma \frac{\partial L_c}{\partial \theta} \right) \tag{14}$$

$$\theta' = \theta' - \frac{\lambda}{m} \sum_{i=1}^{m} \frac{\partial L_r}{\partial \theta'} \tag{15}$$

where λ is the learning rate, and m is the number of data point in the mini-batch.

In order to maintain stability, we update the target distribution P which serves as "ground truth" soft label in every T iterations. We choose to stop training when the algorithm has converged, i.e., the label assignment s between two consecutive updates for the target distribution is less than a predefined threshold ϵ. The label assignment of x_i is computed by:

$$s_i = \arg\max_j q_{ij} \tag{16}$$

The whole learning algorithm of MDEC is summarized in Algorithm 1. Please note that before training, we first construct a k-nearest neighbor graph based on the original feature space. We may also generate a k-nearest neighbor graph for every T iterations according to the latest embedded point $\{z_i\}_{i=1}^n$ during the training procedure. However, computing a k-nearest neighbor graph for every T iterations is expensive, and our experiments demonstrate that adopting the k-nearest neighbor graph constructed based on the original feature space can well capture the geometric structure of data.

Algorithm 1 Manifold Regularized Deep Embedded Clustering

```
Require: Train data: $X$; Number of Clusters: $K$; Number of nearest neighbors: $k$; Target distribution update
    interval: $T$; Stopping threshold: $\delta$; Maximum iterations: $MaxIter$.
Ensure: Parameters $\theta$ and $\theta'$ of autoencoder; Clustering center $\mu$ ; Label assignment $s$.
Initialization: Initialize cluster center $\mu$ using k-means; Initialize parameters $\theta$ and $\theta'$ by pre-training a
    stacked denoising autoencoder as conducted in [30].
1: Construct a $k$-nearest neighbor graph $G$ based on the input data $X$
2: for $i = 0 \rightarrow MaxIter$ do
3:     if i % T == 0 then
4:         compute all embedded points $\{z_i = f_\theta(x_i)\}_{i=1}^n$;
5:         update $q$ and $p$ using (2) and (3), respectively;
6:         save last label assignment: $s_{old} = s$;
7:         compute new label assignment s with (16);
8:         if $sum(s_{old} \neq s)/n < \delta$ then
9:             break.
10:    Sample a mini-batch of $m$ points from $X$, and for
       each point $x_i$, perform the following operation:
11:        Compute its embedded representation $z_i$.
12:        Compute its $k$ nearest neighbors' embedded
           representations $\{z_j\}_{j \in N_i}$ according to $G$.
13:    Calculate the gradients according to (10), (11), and
       (12).
14:    Update parameters $\theta, \theta', \mu$ by using back propoga-
       tion.
```

4 Experiments

4.1 Datasets

We compare our model with several baselines on three widely used benchmark datasets: MNIST, USPS and REUTERS-10K. The table The dataset statistics are summarized in Table 1.

– *MNIST* [18]: The MNIST dataset contains 70,000 handwritten digits of 28 × 28 pixel size. The digits are centered and size-normalized.
– *USPS* [14]: The USPS dataset consists of 9,298 gray-scale handwritten digit images with size of 16 × 16 pixels. The features are floating point in [0,2].
– *REUTERS-10K* [19]: The REUTERS-10K dataset contains nearly 810,000 English news stories labeled with a category tree [19]. As in [30], we adopted four root categories

Table 1 Statistics of the three datasets: MNIST, USPS and REUTERS-10K

Dataset	#Points	#Classes	#Dimension
MNIST	70,000	10	784
REUTERS-10K	10,000	4	2000
USPS	9298	10	256

(*corporate/industrial*, government/social, *markets*, *economics*) as labels and skipped all documents with multiple labels. We further randomly sampled a subset of 10,000 examples and computed tf-idf features on the 2000 most frequent occurring words. The sampled dataset is referred to as REUTERS-10K.

4.2 Baseline Methods

We compare our MDEC framework with five baseline approaches, which are summarized as follows. To the best of our knowledge, our proposed approach is the first work that incorporating the underlying geometric structure into the deep embedded clustering framework. Hence, for the evaluation, we adopt two state-of-the-art deep embedded clustering methods, i.e., DEC and IDEC, as baseline methods. Besides, since our method is relying on the underlying geometric structure of the input data, thus we also compare with the spectral embedded clustering method SEC [23].

It is worth noting that some recent efforts show that leveraging a complicated deep neural network (e.g., a multi-layer convolutional denoising autoencoder) can further improve the performance on visual data [16]. For fair comparison, in this paper, we employ the same base deep neural network (i.e., a stacked autoencoder) as used in IDEC [12], which is a general framework and can handle both textual and visual data. We leave the exploration of specified and complicated deep neural network as future work.

Since KMeans is one of the most classical clustering methods, we implement two variants of KMeans based on different input feature representations: the original representation and the embedded representation pre-trained with an autoencoder. For all compared methods except KMeans based approaches, we used their default parameters recommended by the corresponding authors. More details about these baseline methods are summarized as follows:

- *KMeans* This is a widely established clustering algorithm. We run KMeans 20 times with different initialization and choose the best objective value as the result.
- *AE-KMeans* This method is a two-stage deep clustering algorithm. It first obtains embedded representations by pre-training a stacked autoencoder, and then runs the k-means algorithm on them.
- *Spectral Embedded Clustering (SEC)* [23] SEC is a variant of spectral clustering with a linearity regularization. It outperforms traditional spectral clustering methods on a wide range of datasets. We use the default parameters of SEC provided in [23].
- *Deep Embedded Clustering (DEC)* [30] DEC leverages deep neural networks to simultaneously handle the cluster assignment consistency as well as the underlying feature representation.
- *Improved Deep Embedded Clustering (IDEC)* [12] This method is a variant of DEC. It jointly maintains the cluster assignment consistency and preserves the data generating distribution.

4.3 Evaluation Metrics

To evaluate the performance of the clusters produced by different approaches, we use two evaluation metrics as follows:

– *Clustering Accuracy (ACC)* is defined as:

$$ACC = \frac{\sum_{i=1}^{n} \delta(l_i, map(c_i))}{n},$$

where l_i is the true class label and c_i is the obtained cluster label of x_i, $\delta(x, y)$ is the delta function, and $map(\cdot)$ is the best mapping function. Note $\delta(x, y) = 1$ if $x = y$, $\delta(x, y) = 0$ otherwise. The mapping function $map(\cdot)$ matches the true class label and the obtained cluster label and the best mapping is solved by the Kuhn-Munkres algorithm, also called Hungarian method.

– *Normalized Mutual Information (NMI)* is calculated by:

$$NMI = \frac{MI(C, C')}{\max(H(C), H(C'))},$$

where C is a set of clusters obtained from the true labels and C' is a set of clusters obtained from the clustering algorithm. $MI(C, C')$ is the mutual information metric, and $H(C)$ and $H(C')$ are the entropies of C and C' respectively.

4.4 Implementation

Similar to [30], we implement the encoder network and decoder network in a symmetrical way. Specifically, we set the encoder network as a fully multilayer perceptron (MLP) with dimension d-500-500-2000-10 for all datasets, and set the decoder network in a reverse mode, i.e., a MLP with dimensions 10-2000-500-500-d, where d is the dimension of the input data space. All internal layers are activated by the nonlinearity function ReLU [10]. In order to accelerate the convergence of the model's learning progress, we pre-train a stacked autoencoder network before using back-propagation to fine-tune the entire deep neural networks. There are three parameters γ, δ and k to control the impact of clustering loss and manifold loss. We empirically set $\gamma = 0.1$, $\delta = 0.3$, and $k = 3$. More discussion about the impact of the two parameters can be found in the section of parameter sensitivity analysis. For the MNIST dataset, we adopt the optimizer Adam [17] with an initial learning rate $\lambda = 0.001$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For both the USPS and REUTERS-10K datasets, we use the optimizer SGD with learning rate $\lambda = 0.1$ and momentum $\beta = 0.99$. We set the batch size to 256, and the convergence threshold ϵ to 0.001. The update intervals T for MNIST, USPS and REUTERS-10K are set to 140, 30, 3 iterations, respectively. The implementation of our model is based on Python and Keras [6], and run on a GPU server with GeForce GTX 1080Ti and 11GB GPU memory.

4.5 Experimental Results

As shown in Table 2, the worst performance (in terms of Accuracy) can be observed for the conventional clustering method KMeans on all datasets except USPS. The spectral embedded clustering method SEC demonstrates better performance than KMeans on both REUTERS-10k and MNIST. However, it shows a worse performance than KMeans on USPS, which

Table 2 Comparison of clustering performance of different methods on three datasets in terms of Accuracy(%)

Method	USPS	REUTERS-10K	MNIST
KMeans	66.80	51.48	53.24
SEC	48.02	58.62	80.37
AE + Kmeans	69.13	70.43	81.85
DEC	74.08	73.68	86.55
IDEC	75.98	75.02	88.05
MDEC	**77.88**	**76.82**	**88.46**

The bold numbers indicates the best results among all methods

Table 3 Comparison of clustering performance of different methods on three datasets in terms of NMI(%)

Method	USPS	REUTERS-10K	MNIST
KMeans	62.64	30.87	49.98
SEC	40.14	34.34	N/A
AE + Kmeans	66.17	39.72	74.76
DEC	75.29	49.76	83.72
IDEC	78.36	49.37	86.53
MDEC	**80.60**	**52.65**	**87.31**

The bold numbers indicates the best results among all methods

may be due to the fact that SEC adopts a linear mapping to project the original data into a low dimensional subspace. AE + KMeans outperforms both KMeans and SEC on all datasets. It demonstrates that adopting the embedded representations from a pre-trained stacked autoencoder is helpful as it can preserve the data generating distribution. It is worth noting that autoencoder employs a nonlinear embedding strategy, which differentiates it from the linear mapping based method SEC. Among all baseline methods, the unified end-to-end framework DEC and its variant IDEC achieve superior performance. This is mainly due to the fact that they jointly conduct feature learning as well as clustering. The difference between DEC and IDEC is that IDEC simultaneously minimizes the reconstruction loss and clustering loss, while DEC ignores the reconstruction constraint and only uses an autoencoder for pre-training an initial feature representation.

Compared to all baseline methods, our proposed method MDEC demonstrates the best performance, substantially outperforming the state-of-the-art methods, such as DEC and IDEC. This result verifies the effectiveness of exploiting the geometric structure of data as supervision. Please note that IDEC can be considered as a special case of our proposed MDEC (i.e., MDEC will reduce to IDEC if γ is set to zero). In Table 3, similar results are observed in terms of NMI.

4.6 Parameter Sensitivity Analysis

In this section, we study the sensitivity of the proposed MDEC framework to the three parameters γ, δ, and k. When we vary one parameter, the other two parameters are fixed to the optimal values so that only one of them would influence the results.[1] We only report results in terms of Accuracy as similar findings are observed in terms of NMI. Figure 2 shows the performance of MDEC under different settings of k, γ and δ.

[1] Optimal values of parameters can be found using grid search.

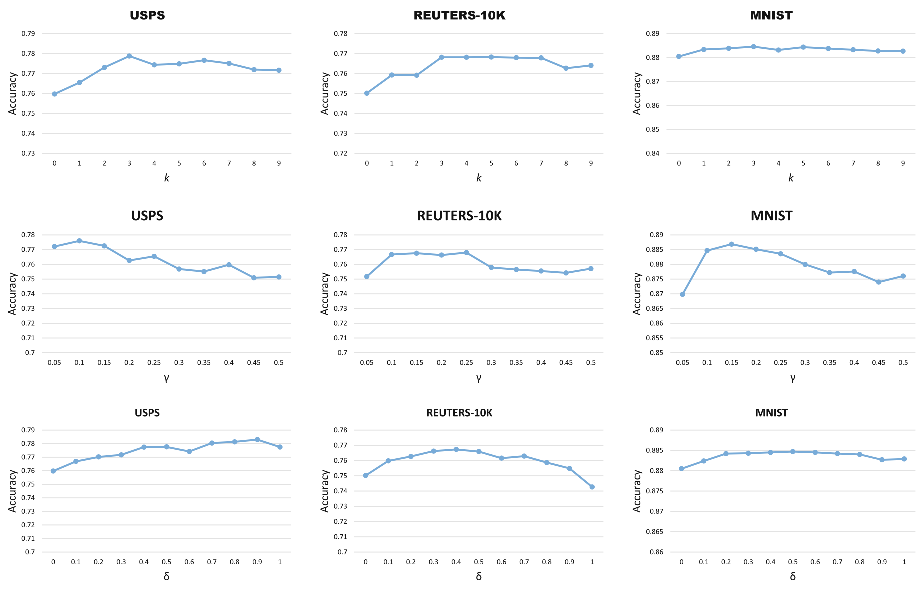
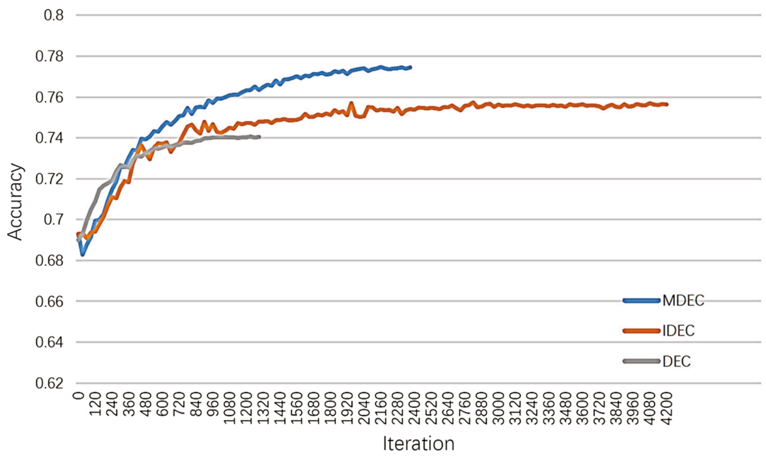


Fig. 2 The performance of MDEC under different settings of k (top row), γ (middle row) and δ (bottom row) on three datasets (USPS, REUTERS-10K, MNIST)

We first look into the parameter k which is number of nearest neighbors for constructing the k-nearest neighbor graph. We vary k values from 0 to 9 with γ and δ fixed as their optimal values (0.1 and 0.3, respectively). The top row of Fig. 2 shows the clustering performance, we can observe that incorporating a manifold constraint, i.e., when k is larger than 0, can help to boost the Accuracy performance by a considerable margin, especially when k varies from 3 to 7. If we keep increasing k, the performance will start to decline. This changing trend is reasonable because useful geometric structure information is leveraged when k is larger than 0. When k becomes too large, unreliable manifold constraint would be injected into the training procedure and hurt the performance of our method.

We then study the impact of the other two parameters γ, δ in Eq. (9). γ is the weight parameter used to determine the importance of utilizing the cluster-oriented supervisory information for guiding the training procedure. The higher the value of γ, the more emphasis MDEC puts on the cluster-oriented supervisory signal. The middle row of Fig. 2 demonstrates the Accuracy values when varying γ from 0.05 to 5 with interval 0.05 (k and δ are fixed to 3 and 0.3, respectively). We observe that the highest accuracy is achieved when γ is around 0.1 for all datasets. When γ becomes larger, there is a slight decrease of the performance. Similarly, δ is the weight parameter to control the influence of manifold constraint. A higher δ value indicates that more emphasis of MDEC will be put on the geometric structure-oriented supervision. The bottom row of Fig. 2 show the Accuracy values obtained by varying δ from 0 to 0.9 by fixing other two parameters ($k = 3$ and $\gamma = 0.1$). We can see that increasing the δ value can generally lead to improvements of performance. When δ gets too large the performances will start to drop. Generally speaking, there is a wide range of δ (i.e., from 0.2 to 0.8) that MDEC performs well with.



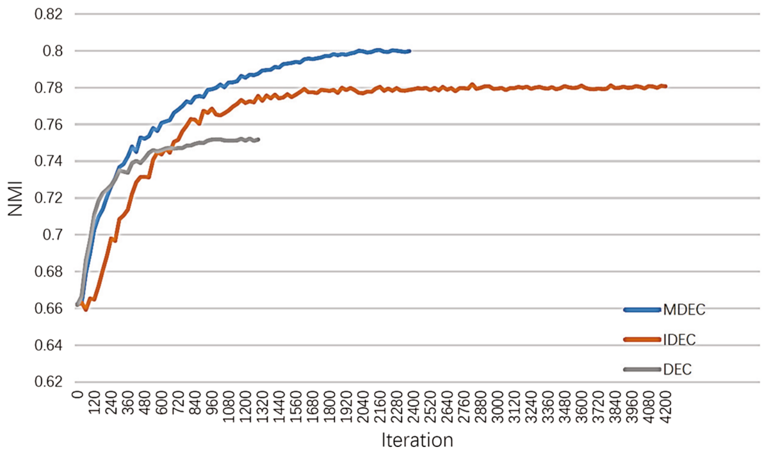


Fig. 3 Accuracy and NMI versus epochs

4.7 Convergence Study

To explore the convergence speed, we plot the results on the dataset USPS through different epochs. Figure 3 shows the learning curve of DEC, IDEC and MDEC. As shown in Fig. 3, DEC converges very fast, but with the lowest performance on both metrics Accuracy and NMI. IDEC shows a slower convergence speed compared with DEC as it needs to simultaneously manipulate both clustering loss as well as the reconstruction loss during the training procedure. The convergence speed of MDEC is faster than IDEC due to the incorporation of the additional manifold loss.

We can also observe that, during the initial iterations (i.e., from 0 to 720), IDEC performs worse than DEC on both metrics. This is mainly caused by the fact that, in the beginning, the parameters of both clustering constraint and the reconstruction constraint are just initialized and most likely disagree with each other. After sufficient iterations (e.g., approximately 750 iterations), IDEC demonstrates consistently superior performance than DEC. The performance of MDEC is better than IDEC as it surpasses DEC with less iterations, e.g., only after

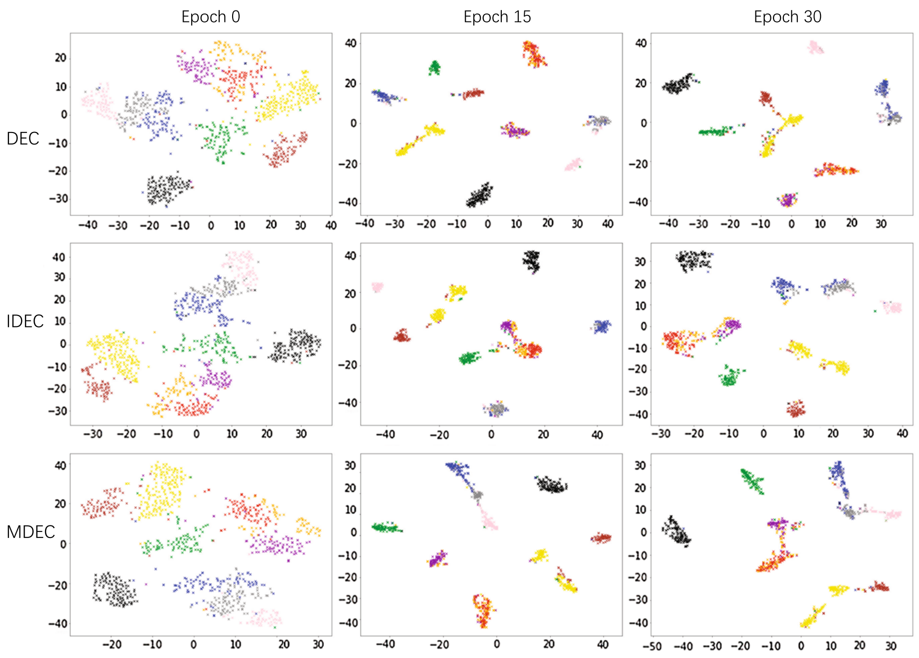


Fig. 4 t-SNE visualization comparison of the embedding feature space learned by three different methods DEC, IDEC and MDEC as the number of epochs increases (each color denotes a cluster). (Color figure online)

around 360 iterations. It is also interesting to see that at for all iterations, MDEC demonstrates a consistent superior performance than IDEC, which shows that further leveraging the underlying geometric structure of the input data can considerably benefit the clustering capacity of MDEC.

4.8 Effect of the Learned Embedding

Figure 4 shows the 2-D embedding feature space of three different methods DEC, IDEC and MDEC by using t-SNE [22] on a random subset of USPS with 1000 samples. From Fig. 4, we can observe that DEC would lead to undesirable embedding results, for example, the cluster colored by yellow (digit 0) will become less differentiable (i.e., mixed with more points from other clusters) when the training epochs increase from 15 to 30. This would caused by the fact that the training procedure of DEC is only guided by the cluster-oriented constraint while ignoring the intrinsic properties of the data space, such as the intrinsic structural signals. IDEC addresses this issue by resorting to incorporate the reconstruction loss which is considered to be able to preserve the local structure of the data-generating distribution. The learned embedded feature space from IDEC is more separable as compared with that of DEC, e.g., these improper data points mixed with yellow cluster are largely reduced. For our proposed method MDEC, the learned embedded feature space has better a differentiating capability. For an instance, only a very few improper data points, which are mixed with the yellow cluster, are observed. This is because MDEC benefits from exploiting the underlying geometric properties of the data.

5 Conclusions

In this paper, we present MDEC, a unified end-to-end framework for unsupervised clustering problems. Different to conventional deep embedded clustering methods such as DEC and IDEC, MDEC incorporates a manifold constraint to preserve the underlying geometric structure of data. The objective of MDEC simultaneously considers data reconstruction, cluster assignment consistency, as well as geometric structure of data. Empirical experimental results on three real-world datasets (i.e., USPS, REUTERS-10K and MNIST) demonstrate that our approach outperforms the current state-of-the-art approaches.

Acknowledgements The work was partially supported by the National Natural Science Foundation of China (No. 61722211), the Federal Ministry of Education and Research (No. 01LE1806A), the Natural Science Foundation of Chongqing (No. cstc2017jcyjBX0059), and the Beijing Academy of Artificial Intelligence (No. BAAI2019ZD0306).

References

1. Cai D, He X, Han J, Huang T (2011) Graph regularized non-negative matrix factorization for data representation. IEEE Trans Pattern Anal Mach Intell 33(8):1548–1560
2. Chang P, Zhang J, Hu J, Song Z (2018) A deep neural network based on ELM for semi-supervised learning of image classification. Neural Process Lett 48(1):375–388
3. Chao G (2019) Discriminative k-means Laplacian clustering. Neural Process Lett 49(1):393–405
4. Chen G (2015) Deep learning with nonparametric clustering. arXiv:1501.03084
5. Chen X, Liu X, Jia Y (2011) Discriminative structure selection method of Gaussian mixture models with its application to handwritten digit recognition. Neurocomputing 74:954–961
6. Chollet F (2015) Keras Technical report
7. Deerwester S, Dumais ST, Furnas GW, Landauer TK, Harshman R (1990) Indexing by latent semantic analysis. J Am Soc Inf Sci 41(6):391–407
8. Deng Z, Huang L, Wang C, Lai J, Yu PS (2019) Deepcf: a unified framework of representation learning and matching function learning in recommender system. CoRR arXiv:1901.04704
9. Gan H, Sang N, Huang C (2015) Manifold regularized semi-supervised Gaussian mixture model. J Opt Soc Am A Opt Image Sci 32(4):566–575
10. Glorot X, Bordes A, Bengio Y (2011) Deep sparse rectifier neural networks. J Mach Learn Res 15:315–323
11. Guo G, Zhang N (2019) A survey on deep learning based face recognition. Comput Vis Image Underst 189:102805
12. Guo X, Gao L, Liu X, Yin J (2017) Improved deep embedded clustering with local structure preservation. In: Proceedings of the twenty-sixth international joint conference on artificial intelligence, pp 1753–1759
13. Hartigan JA, Wong MA (1979) A k-means clustering algorithm. Appl Stat 28:100–108
14. Hastie T, Tibshirani R, Friedman J (eds) (2003) The elements of statistical learning. Springer, New York
15. Hu W, Hu H (2019) Fine tuning dual streams deep network with multi-scale pyramid decision for heterogeneous face recognition. Neural Process Lett 50(2):1465–1483
16. Jabi M, Pedersoli M, Mitiche A, Ayed IB (2019) Deep clustering: on the link between discriminative models and k-means. arXiv:1810.04246
17. Kingma D, Ba J (2014) Adam: a method for stochastic optimization. arXiv:1412.6980
18. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324
19. Lewis DD, Yang Y, Rose TG, Li F (2004) Rcv1: a new benchmark collection text categorization research. J Mach Learn Res 5:361–397
20. Liu J, Cai D, He X (2010a) Gaussian mixture model with local consistency. In: Proceeding of the 24th AAAI conference on artificial intelligence, pp 512–517
21. Liu W, He J, Chang SF (2010b) Large graph construction for scalable semi-supervised learning. In: Proceedings of the 27th international conference on machine learning, pp 679–686
22. Maaten LvD, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9(85):2579–2605
23. Nie F, Zeng Z, Tsang IW, Zhang C (2011) Spectral embedded clustering: a framework for in-sample and out-of-sample spectral clustering. IEEE Trans Neural Netw 22(11):1796–1808

24. Peng X, Xiao S, Feng J, Yau WY, Yi Z (2016) Deep subspace clustering with sparsity prior. In: Proceedings of the twenty-fifth international joint conference on artificial intelligence, pp 1925–1931
25. Song L, Zhang M, Wu X, He R (2018) Adversarial discriminative heterogeneous face recognition. In: Proceedings of the 32nd AAAI conference on artificial intelligence
26. Tian F (2014) Learning deep representations for graph clustering. In: Proceedings of the 27th international conference on neural information processing systems, pp 2429–2437
27. Wada Y, Miyamoto S, Nakagama T, Andéol L, Kumagai W, Kanamori T (2019) Spectral embedded deep clustering. Entropy 21(8):795
28. Wang Z, Chang S, Zhou J, Wang M, Huang TS (2016) Learning a task-specific deep architecture for clustering. In: Proceedings of the 16th SIAM international conference on data mining 2016, pp 369–377
29. Wu B, Wang E, Zhu Z, Chen W, Xiao P (2018) Manifold nmf with l_{21} norm for clustering. Neurocomputing 273:78–88
30. Xie J, Girshick R, Farhadi A (2016) Unsupervised deep embedding for clustering analysis. In: Proceedings of the 33rd international conference on machine learning, pp 478–487
31. Xu X, Shen F, Yang Y, Zhang D, Shen HT, Song J (2017) Matrix tri-factorization with manifold regularizations for zero-shot learning. In: Proceedings of the 30th IEEE conference on computer vision and pattern recognition, pp 2007–2016
32. Yang J, Parikh D, Batra D (2016b) Joint unsupervised learning of deep representations and image clusters. In: Proceeding of the 29th IEEE conference on computer vision and pattern recognition, pp 5147–5156
33. Yang Z, Cohen WW, Salakhutdinov R (2016a) Revisiting semi-supervised learning with graph embeddings. In: Proceedings of the 33rd international conference on machine learning, pp 40–48
34. Ye X, Zhao J (2019) Multi-manifold clustering: a graph-constrained deep nonparametric method. Pattern Recogn 93:215–227
35. Yu SX, Shi J (2003) Multiclass spectral clustering. In: Proceedings of the 9th IEEE international conference on computer vision, pp 313–319
36. Zhang S, Tay Y, Yao L, Wu B, Sun A (2019) Deeprec: an open-source toolkit for deep learning based recommendation. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI 2019, Macao, China, August 10–16, 2019, pp 6581–6583
37. Zhang S, Yao L, Sun A, Tay Y (2019) Deep learning based recommender system: a survey and new perspectives. ACM Comput Surv 52(1):5:1–5:38
38. Zhe X, Chen S, Yan H (2019) Directional statistics-based deep metric learning for image classification and retrieval. Pattern Recogn 93:113–123
39. Zheng M, Bu J, Chen C, Wang C, Zhang L, Qiu G, Cai D (2011) Graph regularized sparse coding for image representation. IEEE Trans Image Process 20(5):1327–1336
40. Zhu X, Li Z, Zhang X, Li P, Xue Z, Wang L (2019) Deep convolutional representations and kernel extreme learning machines for image classification. Multimed Tools Appl 78(20):29271–29290