Multi-View Deep Clustering based on AutoEncoder

Shihao Dong\(^{1,a}\), Huiying Xu\(^{2,b*}\), Xinzong Zhu\(^{3,c*}\), XiFeng Guo\(^{4,d}\), Xinwang Liu\(^{5,e}\), Xia Wang\(^{6,f}\)

\(^{1,2,3,6}\)School of Mathematics and Computer Science, ZheJiang Normal University, Jinhua, ZheJiang, China

\(^{4,5}\)National University of Defense Technology, Changsha, HuNan, China

\(^{a}\)e-mail: louisdong@zjnu.edu.cn

\(^{b*}\)Corresponding author: xhy@zjnu.edu.cn

\(^{c*}\)Corresponding author: zxz@zjnu.edu.cn

\(^{d}\)e-mail: guoxifeng1990@163.com, \(^{e}\)e-mail: xinwangliu@nudt.edu.cn, \(^{f}\)e-mail: wangxia@zjnu.edu.cn

(Shihao Dong and Huiying Xu contributed equally to this work.)

( Corresponding authors: Huiying Xu; Xinzong Zhu.)

Abstract—In recent years, with the development of deep learning, replacing traditional clustering methods with subspaces extracted by deep neural networks will help better clustering performance. However, due to the instability of unsupervised learning, the features extracted each time are different even if the same data is processed. In order to improve the stability and performance of clustering, we propose a novel unsupervised deep embedding clustering multi-view method, which treats multiple different subspaces as different views through some data expansion methods for the same data. Specifically, our method uses a variety of different deep autoencoders to learn the latent representation of the original data and constrain them to learn different features. Our experimental evaluations on several natural image datasets show that this method has a significant improvement compared to existing methods.

1. Introduction

Cluster analysis is a basic problem in many fields, such as machine learning, data mining, pattern recognition, image analysis, and biological information. Clustering is to divide similar objects into different groups or more subsets through static classification, so that the member objects in the same subset have similar attributes. Generally, data clustering is summarized as a non-Supervised learning. Some common clustering methods\(^{[1]}\)–\(^{[5]}\) However, because the similarity measurement methods used by traditional clustering methods are inefficient, their performance on high-dimensional data is usually poor. In addition, these methods usually have high computational complexity on large-scale data sets. Therefore, people have extensively studied dimensionality reduction and feature conversion methods to map the original data to a new feature space, in which the generated data can be more easily separated by existing classifiers. Generally speaking, the existing data conversion methods include linear transformation (such as principal component analysis\(^{[6]}\)) and non-linear transformation (such as nuclear method\(^{[7]}\) and spectral method\(^{[8]}\)). Nevertheless, the highly complex underlying structure of the data still challenges the effectiveness of existing clustering methods.
Due to the development of deep learning[9], and the inherent characteristics of the highly non-linear transformation of deep neural networks, it can be used to transform data into a representation that is easier to cluster. In recent years, deep embedding clustering has been proposed[9], followed by other novel methods[10]–[16], making deep clustering a popular research field. Examples include stacked autoencoders[17], variable autoencoders[18] and convolutional autoencoders[19], which are proposed for unsupervised learning. The neural network-based clustering method defeats the traditional method to a certain extent. The method is an effective method to learn complex nonlinear transformations to obtain powerful features. However, the single view method of acquiring features through neural networks, that is, first extracting view features, and then using traditional clustering, such as K-means or spectral clustering, does not fully extract all the features of the data, and does not make good use of multi-view features the relationship between learning and clustering, so this separate learning strategy may bring unsatisfactory clustering results. To solve this problem, we propose a multi-view depth clustering method based on autoencoder.

2. Materials and Methods
The multi-view depth clustering (MDEC) structure based on the autoencoder is shown in Figure 1. The structure consists of four parts: 1. An encoder composed of an AutoEncoder(AE), a Convolutional AutoEncoder(CAE), and a Convolutional Variational AutoEncoder(CVAE). 2. Multi-view fusion layer. 3. Deeply embedded clustering layer. 4. Decoder.

2.1 Network structure

2.1.1 AutoEncoder
In the model, we use the representation data set to obtain potential features through the non-linear mapping of the autoencoder, convolutional autoencoder, and variational autoencoder. The encoder can convert high-dimensional data into low-dimensional features. The expression is as follows:

\[ h(X; \theta_m) = Z_m \]  

(1)

2.1.2 Multi-view Fusion Layer
After mapping from the self-encoder layer, we get \( M \) latent feature spaces \( Z_m \). In order to obtain more comprehensive information of the original data, we fuse different features \( Z_m \) obtained by different autoencoders into the common subspace \( Z \). The formula is as follows:

\[ Z = \int (\{Z_m\}_{m=1}^M; \beta) = \sum_{m=1}^M \beta_m Z_m / \sum_{m=1}^M \beta_m \]  

(2)

\( \int (\cdot; \beta) \) represents the fusion function, parameter \( \beta = [\beta_1, \beta_2, ..., \beta_m] \).

2.1.3 Deep embedded clustering layer
With the help of DEC[9] in the clustering layer, we divide \( n \) points \( \{x_i \in X\}_{i=1}^n \) into \( k \) clusters. The center of each class is represented by \( \mu_j, \ j = 1, ..., k \). We cluster the fusion feature \( Z \), first initialize

Figure 1: Structure of MDEC
the cluster centers \( \{ \mu_j \}_{j=1}^k \), then calculate the soft assignment of feature points and cluster centers, calculate the KL divergence of the soft assignment and auxiliary target distribution to update the cluster center \( \mu_j \), parameter \( \theta \) and \( \beta_m \).

2.1.4 Decoder
In order to better learn the feature \( Z \) of the original data \( X \), we use a symmetrical structure decoding with the encoder:

\[
\overline{X} = g(Z; \theta_m)
\]  

(3)

\( \overline{X} \) represents the reconstruction of sample \( X \).

2.2 Loss Function
The loss function consists of three parts: (1) The reconstruction loss \( L_R \) is used to update the encoder, convolutional autoencoder, and convolutional variational autoencoder network parameters. (2) The fusion loss \( L_F \) is used to update the fusion parameters \( \beta \). (3) The clustering loss \( L_C \) is used to update the clustering results and autoencoder parameters and fusion parameters.

2.2.1 Fusion Loss
After obtaining different potential features \( Z_m \) from multiple encoders, we adopt a multi-view fusion method, and use the square difference function of the common subspace feature and the feature before fusion as the fusion loss, so that the common subspace feature can represent the original data to the maximum. Defined as follows:

\[
L_F = \min_{\theta, \beta} \sum_{m=1}^M \| \sum_{m=1}^M \beta_m Z_m \|^2
\]

(4)

2.2.2 Restructure Loss
Our model uses the square difference function of the encoder input and the decoder output as the reconstruction loss, pre-trains the self-encoder, and obtains a good initialization model:

\[
L_R = \min_{\theta, \beta} \sum_{m=1}^M \| \overline{X} - X \|^2
\]

(5)

2.2.3 Clustering Loss
According to van der Maaten & Hinton(2008)[20], we use Student’s t-distribution as a kernel to measure the similarity between embedded point \( Z_i \) and centroid \( \mu_j \):

\[
q_{ij} = \frac{(1 + \|Z_i - \mu_j\|^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_j (1 + \|Z_j - \mu_j\|^2/\alpha)^{-\frac{\alpha+1}{2}}}
\]

(6)

Where \( Z_i = \int_A (X_i) \in Z, \alpha \) are the degrees of freedom of the Student’s t-distribution and \( q_{ij} \) can be interpreted as the probability of assigning sample \( i \) to cluster \( j \). The clustering is optimized by learning from the high-confidence assignment of the cluster with the help of the auxiliary target distribution. Specifically, we train our model by matching the soft allocation to the target distribution. To this end, we define the goal as the KL divergence loss between the soft assignment \( q_{ij} \) and the auxiliary distribution \( p_{ij} \), as shown below:

\[
L = KL(P \parallel Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}
\]

(7)

We compute \( p_{ij} \) by first raising \( q_{ij} \) to the second power and then normalizing by frequency per:

\[
p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_j q_{ij}^2 / f_j}
\]

(8)
Where \( \int_j = \sum_i q_{ij} \).

We divide the training into two stages, namely the pre-training initialization stage and the clustering optimization stage. In the pre-training initialization phase, we use the following loss function to train the model:

\[
\mathcal{L}_1 = \mathcal{L}_R + \mathcal{L}_F
\]  

(9)

In the cluster optimization stage, we use the loss function:

\[
\mathcal{L}_2 = \mathcal{L}_R + \mathcal{L}_F + \mathcal{L}_C
\]  

(10)

2.3 Optimization

We jointly optimize the cluster centers \( \{\mu_j\} \) and network parameters \( \theta \), using Stochastic Gradient Descent (SGD) with momentum. The gradients of \( \mathcal{L} \) with respect to feature space embedding of each data point \( Z_i \) and each cluster centroid \( \mu_j \) are computed as:

\[
\frac{\partial \mathcal{L}}{\partial Z_i} = \frac{a+1}{a} \sum_l (1 + \frac{\|Z_i - \mu_j\|^2}{a})^{-1} \times (p_{ij} - q_{ij})(Z_i - \mu_j)
\]  

(11)

\[
\frac{\partial \mathcal{L}}{\partial \mu_j} = -\frac{a+1}{a} \sum_l (1 + \frac{\|Z_i - \mu_j\|^2}{a})^{-1} \times (p_{ij} - q_{ij})(Z_i - \mu_j)
\]  

(12)

The gradient \( Z_i \) respect to latent space of every model \( Z_i^m \) and fusion parameter \( \beta_m \) are followed as:

\[
\frac{\partial Z_i}{\partial Z_i^m} = \beta_m / \sum_{m=1}^{M} \beta_m
\]  

(13)

\[
\frac{\partial Z_i}{\partial \beta_m} = \frac{Z_i^m \times \sum_{m=1}^{M} \beta_m - \sum_{m=1}^{M} (\beta_m \times Z_i^m)}{(\sum_{m=1}^{M} \beta_m)^2}
\]  

(14)

The gradients \( \partial \mathcal{L}/\partial Z_i \) are then passed down to the DNN and used in standard backpropagation to compute the DNN’s parameter gradient \( \partial \mathcal{L}/\partial \theta \), For the purpose of discovering cluster assignments, we stop our procedure when less than \( 0.01 \% \) of points change cluster assignment between two consecutive iterations.

We extract different latent features through different encoders, and merge the features into a common subspace. After pre-training, we get the initial fusion parameters \( \beta_m \) and model parameters \( \theta_m \), Then perform K-means clustering on the fused public subspace \( Z \) to initialize the cluster centroid \( \mu_j \).

3. Results & Discussion

3.1 Datasets

We validated our proposed method on multiple data sets and compared with multiple excellent methods:

**MNIST**: The MNIST data set consists of 70,000 handwritten digits and is 28 x 28 pixels in size. These numbers have been centered and normalized in size[21].

**FASHION-MNIST**: Contains 70,000 fashion product pictures from 20 categories, and the picture size is the same as MNIST[22].

**COIL-20**: Collect 20 categories of 1440 128 x 128 gray-scale object images viewed from different angles[24].

Data set specific information and sample view Table 1 and Figure 2.

Table 1: Information of datasets

| Datasets     | Number | Class | Size          |
|--------------|--------|-------|---------------|
| MNIST        | 70000  | 10    | (28,28,1)     |
| FASHION-MNIST| 70000  | 10    | (28,28,1)     |
| COIL-20      | 1440   | 20    | (128,128,1)   |
3.2 Evaluation Metric
We use standard unsupervised evaluation indicators and protocols to evaluate and compare other algorithms. For all algorithms, we set the number of clusters to the number of true categories, and use unsupervised clustering accuracy (ACC) to evaluate performance:

\[
ACC = \max_m \frac{\sum_{i=1}^{n} \mathbb{1}(\ell_i = m(c_i))}{n}
\]

Where \(\ell_i\) is the ground-truth label, \(c_i\) is the cluster assignment produced by the algorithm, and \(m\) ranges over all possible one-to-one mappings between clusters and labels.

Intuitively this metric takes a cluster assignment from an unsupervised algorithm and a ground truth assignment and then finds the best matching between them. The best mapping can be efficiently computed by the Hungarian algorithm[27].

3.3 Implementation
We use autoencoder, convolutional variational autoencoder and convolutional autoencoder as the three single-view deep network branches for the original image. For specific network configuration, see Table 2.

| Branch | Structure |
|--------|-----------|
| AE     | 500-500-2000-10 |
| CAE    | Conv1(5 × 5 × 32,strides=2)-Conv2(5 × 5 × 64,strides=2)-Conv3(3 × 3 × 128,strides=2)-flatten-10 |
| CVAE   | Conv1(2 × 2 × 1)-Conv2(2 × 2 × 6)-Conv3(3 × 3 × 20)-Conv3(3 × 3 × 60)-Flatten-256-10 |
3.4 Algorithm comparison

Table 3: Comparison of clustering performance of different algorithms on three data sets

| Methods | MNIST  | FASHION-MNIST | COIL-20 |
|---------|--------|---------------|---------|
|         | ACC    | NMI           | ACC     | NMI    | ACC     | NMI    |
| K-means | 0.546  | 0.495         | 0.512   | 0.499  | 0.574   | 0.732  |
| DEC     | 0.844  | 0.816         | 0.518   | 0.546  | 0.680   | 0.802  |
| RMKMC   | 0.592  | 0.658         | 0.533   | 0.528  | 0.609   | 0.749  |
| DCCA    | 0.480  | 0.397         | 0.527   | 0.538  | 0.557   | 0.649  |
| DCCAE   | 0.467  | 0.392         | 0.518   | 0.530  | 0.561   | 0.653  |
| DGCCA   | 0.632  | 0.581         | 0.562   | 0.570  | 0.540   | 0.624  |
| DMJC    | 0.960  | 0.931         | 0.620   | 0.647  | 0.714   | 0.808  |
| MDEC    | 0.989  | 0.962         | 0.586   | 0.652  | 0.789   | 0.831  |

We choose two monomodal clustering methods: K-means[5], Deep embedding clustering(DEC)[9]; traditional large-scale multi-view clustering method: Robust Multi-View K-Means Clustering (RMKMC)[28]; two deep two-modal clustering methods: Deep Canonical Correlation Analysis(DCCA)[29], Deep Canonically Correlated Auto-Encoders (DCCAE)[30]; two deep multi-modal clustering methods: Deep Generalized Canonical Correlation Analysis (DGCCA)[31], Joint framework for Deep Multi-view Clustering (DMJC)[32] as a comparison with our proposed algorithm Table 3.

4. Conclusions

In this paper, we propose a novel multi-view depth clustering framework (MDEC), which includes a multi-modal encoder, a fusion network and a deep clustering layer. Through the multi-modal encoder and the fusion layer, MDEC extracts original data features through nonlinear mapping, reduces the dimensionality of high-dimensional data, optimizes the common subspace of data features, and finally constrains the subspace clustering with KL divergence. Experimental results on three public data sets prove that our model is superior to several latest models.

Acknowledgment

This work was supported by Outstanding Talents of “Ten Thousand Talents Plan” in Zhejiang Province (project no. 2018R51001), the Natural Science Foundation of China (project no. 61976196). Shihao Dong and Huiying Xu equally contribute to the paper.

References

[1] Kohonen, T. "The self-organizing map. Neurocomputing 21, 1e6." (1998).
[2] Reynolds, Douglas A. "Gaussian Mixture Models." Encyclopedia of biometrics 741 (2009).
[3] Ester, Martin, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." Kdd. Vol. 96. No. 34. 1996.
[4] Arthur, David, and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Stanford, 2006.
[5] Hartigan, John A., and Manchek A. Wong. "Algorithm AS 136: A k-means clustering algorithm." Journal of the royal statistical society. series c (applied statistics) 28.1 (1979): 100-108.
[6] Wold, Svante, Kim Esbensen, and Paul Geladi. "Principal component analysis." Chemometrics and intelligent laboratory systems 2.1-3 (1987): 37-52.
[7] Hofmann, Thomas, Bernhard Schölkopf, and Alexander J. Smola. "Kernel methods in machine learning." The annals of statistics (2008): 1171-1220.
[8] Ng, Andrew Y., Michael I. Jordan, and Yair Weiss. "On spectral clustering: Analysis and an algorithm." Advances in neural information processing systems. 2002.
[9] Schmidhuber, Jürgen. "Deep learning in neural networks: An overview." Neural networks 61 (2015): 85-117.
[10] Xie, Junyuan, Ross Girshick, and Ali Farhadi. "Unsupervised deep embedding for clustering analysis." International conference on machine learning. 2016.
[11] Li, Fengfu, Hong Qiao, and Bo Zhang. "Discriminatively boosted image clustering with fully convolutional auto-encoders." Pattern Recognition 83 (2018): 161-173.
[12] Yang, Bo, et al. "Towards k-means-friendly spaces: Simultaneous deep learning and clustering." International conference on machine learning. PMLR, 2017.
[13] Ghasedi Dizaji, Kamran, et al. "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization." Proceedings of the IEEE international conference on computer vision. 2017.
[14] Jiang, Zhuxi, et al. "Variational deep embedding: An unsupervised and generative approach to clustering." arXiv preprint arXiv:1611.05148 (2016).
[15] Yang, Jianwei, Devi Parikh, and Dhruv Batra. "Joint unsupervised learning of deep representations and image clusters." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.
[16] Hsu, Chih-Chung, and Chia-Wen Lin. "Cnn-based joint clustering and representation learning with feature drift compensation for large-scale image data." IEEE Transactions on Multimedia 20.2 (2017): 421-429.
[17] Wang, Zhangyang, et al. "Learning a task-specific deep architecture for clustering." Proceedings of the 2016 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2016.
[18] Vincent, Pascal Vincent., Larochelle Larocheh, and H. Stacked Denoising Autoencoders. "Learning Useful Representations in a Deep Network with a Local Denoising Criterion Pierre-Antoine Manzagol." J. Mach. Learn Res 11 (2010): 3371-3408.
[19] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).
[20] Guo, Xifeng, et al. "Deep clustering with convolutional autoencoders." International conference on neural information processing. Springer, Cham, 2017.
[21] Maaten, Laurens van der, and Geoffrey Hinton. "Visualizing data using t-SNE." Journal of machine learning research9.Nov (2008): 2579-2605.
[22] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.
[23] Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.
[24] Li, Fengfu, Hong Qiao, and Bo Zhang. "Discriminatively boosted image clustering with fully convolutional auto-encoders." Pattern Recognition 83 (2018): 161-173.
[25] Coates, Adam, Andrew Ng, and Honglak Lee. "An analysis of single-layer networks in unsupervised feature learning." Proceedings of the fourteenth international conference on artificial intelligence and statistics. 2011.
[26] Krizhevsky, Alex, and Geoff Hinton. "Convolutional deep belief networks on cifar-10." Unpublished manuscript 40.7 (2010): 1-9.
[27] Xiao, Jianxiong, et al. "Sun database: Large-scale scene recognition from abbey to zoo." 2010 IEEE computer society conference on computer vision and pattern recognition. IEEE, 2010.
[28] Kuhn, Harold W. "The Hungarian method for the assignment problem." Naval research logistics quarterly 2.1-2 (1955): 83-97.
[29] Cai, Xiao, Feiping Nie, and Heng Huang. "Multi-view k-means clustering on big data." Twenty-Third International Joint conference on artificial intelligence. 2013.

[30] Andrew, Galen, et al. "Deep canonical correlation analysis." International conference on machine learning. PMLR, 2013.

[31] Wang, Weiran, et al. "On deep multi-view representation learning." International conference on machine learning. 2015.

[32] Benton, Adrian, et al. "Deep generalized canonical correlation analysis." arXiv preprint arXiv:1702.02519 (2017).

[33] Lin, Bingqian, et al. "Deep Multi-View Clustering via Multiple Embedding." CoRR (2018).