

Deep Embedding Clustering Based on Residual Autoencoder

Mengli Li^{1,2} · Chao Cao^{2,3} · Chungui Li² · Shuhong Yang²

Accepted: 23 February 2024 / Published online: 30 March 2024 © The Author(s) 2024

Abstract

Clustering algorithm is one of the most widely used and influential analysis techniques. With the advent of deep learning, deep embedding clustering algorithms have rapidly evolved and yield promising results. Much of the success of these algorithms depends on the potential expression captured by the autoencoder network. Therefore, the quality of the potential expression directly determines the algorithm's performance. In view of this, researchers have proposed many improvements. Although the performance has been slightly improved, they all have one shortcoming, that is, too much emphasis is placed on the original data reconstruction ability during the process of feature expression, which greatly limits the further expression of potential features according to specific clustering tasks. Moreover, there is a large amount of noise in the original data, so blindly emphasizing reconstruction will only backfire. Hence, we innovatively propose a deep embedding clustering algorithm based on residual autoencoder (DECRA) after in-depth research. Specifically, a novel autoencoder network with residual structure is proposed and introduced into deep embedded clustering tasks. The network introduces an adaptive weight layer in feature representation z, which can make it have good robustness, generalization for specific tasks, and adaptive learning of better feature embeddings according to category classification. In this paper, the reasons for the validity of this structure are explained theoretically, and comprehensive experiments on six benchmark datasets including various types show that the clustering performance of the DECRA is very competitive and significantly superior to the most advanced methods.

Keywords Deep embedded clustering \cdot Residual structure \cdot Autoencoder network \cdot Representation learning

Shuhong Yang netysh2007@qq.com

¹ School of Technology, Guilin University, Guilin , China

² School of Computer Science and Technology, Guangxi University of Science and Technology, Liuzhou, China

³ Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou, China

1 Introduction

Clustering is an analysis technique that divides data into disjointed subclasses by using the internal structure of data. It is one of the important means in the field of data mining analysis and has attracted much attention. For a long time, it as widely used in many fields such as image segmentation [1], image recognition [2-4], network analysis [5-7] and document analysis [8, 9]. Especially the workload of data annotation is large and the cost is high in the era of big data. Cluster analysis highlights its research value because it does not need supervising information. Many research methods have emerged over the last few decades, such as K-means clustering [10], Subspace clustering [11], Spectral clustering [12], clustering based on non-negative matrix decomposition [13], etc., and have achieved important results in many fields. However, these models are shallow models, and their representation process relies heavily on hand-designed features and expert knowledge, so the representation ability of the models is limited, and they cannot represent the deep semantic information between the data. Secondly, these methods all involve very complex computational processes, such as kernel matrix calculation in kernel K-means method, decomposition process in non-negative matrix decomposition, and eigenvalue solving problem in spectral method, etc., leading to high time complexities, often on the order of $O(n^3)$, making them unsuitable for big data applications.

In response to these challenges, some studies have proposed to first use dimension reduction technology to map high-dimensional complex original features to relatively low-dimensional space [14] for the sake of obtaining low-dimensional embedding representation of the original data, and then carry out clustering. This two-step strategy has disconnected the correlation between feature representation learning and cluster allocation, and it is meaningless to have feature representation learning without specific clustering tasks. Although it seems to perfectly solve the fundamental reason why traditional methods are not applicable to big data, in fact, there are huge omission in theory and little effect. Based on this, later studies [15] Torre and Kanade proposed to combine the two to form a unified training mechanism, specifically by combining linear discriminant analysis and K-means to complete specific tasks. This research has far-reaching significance, but the shallow model used does not perform well in feature representation learning.

In recent years, thanks to deep learning, the above challenges faced by cluster analysis can be better solved. The reason is that clustering algorithms based on deep learning no longer rely on manually designed features and expert knowledge, but self-learning is more suitable for the feature expression of the big data and the current learning task. Second, compared with the shallow model, the deep network is more conducive to mining the deep semantic information of data. Finally, with the rapid development of deep learning frameworks such as Pytorch, Tensorflow, hardware and software platforms (GPUs), the emergence of efficient algorithms, for instance, random optimization and parallel computing, deep clustering algorithms can be applied to larger datasets. Therefore, a large number of studies have embraced deep learning, propelling clustering analysis into the era of deep learning. As an important unsupervised neural network model, autoencoder can effectively extract and express features of high-dimensional data, and plays an important role in deep clustering. Hence, the quality of the potential expression captured by the automatic coding network directly determines the performance of the algorithm. However, some recent studies [16, 17] have adopted the idea of IDEC [18] adding additional reconstruction loss from autoencoder in training to preserve the local structure of features. Although this can properly improve the damage to the embedded space in the learning process of clustering division, it overemphasizes the data reconstruction ability of the algorithm, greatly limiting the further expression of potential features according to specific clustering tasks. Moreover, it does not consider the large amount of noise information already existing in the original data, and blind reconstruction will only backfire. In this regard, IDCEC [19] Lu et al. proposed an improved method, which first assigned pseudo-labels to each sample through clustering division, and then selected samples with high similarity from each class according to various central points as a reliable benchmark to train additional convolutional neural networks for classification. The trained convolutional neural networks can assist the clustering networks to cluster. It is undeniable that this study reduces the influence of sample noise on clustering to a certain extent through the auxiliary pseudo-label classification mechanism, but at the same time, it also introduces new uncertainties, that is, the selection of various center points is often disturbed by noise, and fails to represent the class well, and cannot achieve good results.

In view of this, we propose an innovative deep embedding clustering algorithm based on residual autoencoder after in-depth research. This algorithm proposes a novel autoencoder network, which starts with the potential representation of features and adds an adaptive weight layer to enhance robustness, task-specific generalization, and adaptive learning of better embedded representation of features according to class classification. Without introducing additional information, the clustering effect is greatly improved. We summarize the main contributions of this paper as follows:

- We propose a novel deep embedding clustering scheme, called DECRA, and design a novel autoencoder model with residual structure to enhance the robustness of potential representations, task-specific generalization, and adaptive learning of better feature embedding representations based on category classification.
- Our algorithm considers both clustering loss and reconstruction loss of residual autoencoder during clustering optimization, so that the learned feature expression is more conducive to the clustering task.
- The algorithm in this paper performs comprehensive experiments, and the results show that the algorithm is universal and competitive for multiple types of datasets. Even for difficult-to-process text dataset, performance is significantly improved.

The following contents describe related works and our method details respectively correspond to Sects. 2 and 3. Section 4 presents the experimental settings and experimental results on related datasets. Section 5 gives the conclusion.

2 Related Works

2.1 Autoencoder Network

Autoencoder is a neural network structure for unsupervised learning. The feature expression of the input sample is learned by the encoder, and then reconstructed by the decoder. After minimizing the autoencoder reconstruction loss, the feature expression of the input sample in the embedded space is learned by the intermediate state z. The process can be formalized as follows: corresponding to a set of samples $x_i \in R^D$ in the input space, the autoencoder maps $f(x_i; \theta_e)$ learns the encoding $z_i \in R^M$ for x in feature space, and then uses the decoder $g(z_i; \theta_e)$ reconstructs the original sample using the encoding in the feature space.

Encoder: $f : \mathbb{R}^D \to \mathbb{R}^M$. The features of the input x_i are condensed by the mapping function and saved as z_i . Most of them use nonlinear mapping functions, the most common

$$z_i = f(x_i; \theta_e) = \frac{1}{1 + \exp(-(Wx_i + b))}$$
(1)

where $\Theta_e = \{W, b\}, W \in \mathbb{R}^{M \times D}$ is the encoding weight, and $b \in \mathbb{R}^M$ is the corresponding deviation vector. The sigmoid function of Eq. (1) has the same input and output dimensions, so the output z_i is also M-dimensional.

Decoder: $g : \mathbb{R}^M \to \mathbb{R}^D$. The role of the decoder is to remap the encoding of the feature space to the original sample. The mapping guess the same nonlinear function as the encoder. Reconstructing the original input x_i mapping from hidden representation z_i by decoder is calculated as follows:

$$\tilde{x}_{z_i} = f(z_i; \theta_d) = \frac{1}{1 + \exp(-(W'z_i + b'))}$$
(2)

where $\Theta_d = \{W', b'\}, W' \in \mathbb{R}^{D \times M}$ is the decoding weight and $b' \in \mathbb{R}^D$ is the decoding deviation vector. This architecture can learn potential representations of features. Given the number of training samples N, the parameters W, W', b, and b can be solved by minimizing the following objective function:

$$\min \frac{1}{N} \sum_{i=1}^{N} \|x_i - \tilde{x}_{z_i}\|_2^2$$
(3)

2.2 Clustering Algorithm

Cluster analysis is essentially to divide the sample into disjoint subsets by using the similarity between the data. In recent decades, many classical clustering methods have been proposed, such as subspace clustering, K-means clustering, spectral clustering, clustering based on non-negative matrix decomposition, etc., which have had an important influence in the field of clustering. However, with the changing landscape of data complexity and volume, these methods are difficult to meet the current needs due to limited representation ability and low efficiency. Therefore, how to efficiently process large-scale data and obtain good clustering performance has aroused extensive thinking among researchers. Chang et al. [20], proposed a straightforward approach that first learns the low-dimensional embedded representation of the data by applying principal component analysis, and then performs clustering to reduce the complexity of the data. In addition, Cai et al. [21] use sparse representation method to deal with high-dimensional data and calculate the embedded representation of the data through the anchor points. Gu et al. [22] also learn the discriminative features of data based on sparse representation and improve fuzzy C-means method. With the introduction of a new fast learning algorithm (extreme learning machine), due to its stochastic nonlinear feature mapping and general fitting capabilities, [23] the algorithm's training efficiency and clustering performance have been improved compared with traditional methods. Subsequently, many researchers followed up, and Zhang et al. combined multi-view and proposed an ELM clustering framework based on multi-view fusion, called MVEC [24]. Chen et al. [25] used feature selection to remove irrelevant hidden nodes, and combined with extreme learning machine to cluster data directly. Chen et al. [26] integrated extreme learning machine into maximum interval clustering and used approximate empirical kernel mapping instead of random feature mapping, which further reduced the training time and memory consumption, and further improved the clustering accuracy.

2.3 Deep Clustering Algorithm

Recently, deep learning has made remarkable achievements. Many studies combined deep learning to gain powerful representation capabilities to acquire good embedding expressions, and achieved significant improvement in clustering performance. Such as, [27] proposed the Deep Embedding Network (DEN) in 2014 to realize the deep embedding expression of samples in the feature space. [28] proposed Deep Embedding Clustering (DEC) in 2016, which uses deep encoder network to capture the features suitable for classification as well as defines a clustering loss function to optimize the parameters of neural network and clustering centers. Although deep learning and clustering tasks are embedded in the same network framework, the algorithm only uses clustering loss to carry out feature representation learning and category classification tasks, neglecting the maintenance of local connection structure between features during training, resulting in the damage of feature space and thus affecting the clustering performance. Subsequently, [18] proposed an improved algorithm to maintain the local structure of features in the process of deep feature learning. DEC is improved by combining reconstruction losses and clustering losses in autoencoders. Soon after, [16] proposed the Convolutional Autoencoders version (DCEC) to introduce convolutional autoencoders for clustering.

3 Proposed Method

Our proposed algorithm, named deep embedding clustering based on residual autoencoder (DECRA), will be introduced in details in this section. For the convenience of representation, the variables are defined as follows: Suppose the dataset *X* has *m*samples, each sample $x_i \in \mathbb{R}^n$, where *n* is the dimension. The prior knowledge *K* represents number of clusters, and the *j*-th cluster center is represented by $\mu_j \in \mathbb{R}^d$. With $s_i \in \{1, 2, ..., K\}$ represents the assigned clustering label about x_i . $Z = \{z_1, ..., z_m\}$ represents their corresponding potential representation, where $z_i = f(x_i; \theta_e) \in \mathbb{R}^d$ is the output of the encoder after processing. The residual structure $\hat{z} = H(z, W) + z$ is formed by applying an adaptive weight layer to the potential feature *z* and with the original z. $\tilde{x}_{z_i} = g(\hat{z}_i; \theta_d)$ represents the reconstructed data of the original sample and the decoder output.

Our research is dedicated to find a model structure that makes the embedding point $\{z_i\}_{i=1}^m$ more suitable for clustering tasks. For this reason, adding residual structure on the basis of autoencoder can learn the feature representation of samples more robust under unsupervised conditions, and enable the learned feature representation to self-adapt and learn better feature embedding representation according to class classification. The algorithm is divided into two parts: pre-training and fine-tuning. The overall network architecture is shown in Fig. 1.

3.1 Network Pre-Training

Consistent with DEC, IDEC and other deep embedded clustering algorithms, DECRA algorithm also needs to be pre-trained on the network to get the mapping of the samples in the low-dimensional space. The main purpose is to get a good feature expression, and the second is to reduce the influence of latitude on the algorithm. However, the traditional autoencoder learns the abstract representation of each sample through the neural network. When the network is complex to a certain degree or the training blindly emphasizes reconstruction, the



Fig. 1 The framework of the proposed DECRA

autoencoder will remember all the information of each sample, including the noise, and there is a serious risk of overfitting. The generated potential expression greatly reduces the discriminability and generalization for specific clustering tasks.

Therefore, a novel residual autoencoder is proposed in this paper, which is expected to solve the above problems. The structure is shown in Fig. 2. After encoding, the network flow direction is divided into two branches: one is activated by the adaptive weight layer $z \rightarrow H(z, W)$, see Eq. (4); The other one does not carry out special processing $z \rightarrow z$, and retains the original data. Combine the results of the two paths and pass them to the decoder, keeping the original feature dimension unchanged, as shown in Eq. (5). w_i is the weight of the neural network; \hat{z} is the output vector of the residual block. b_i is the offset term of the neural network.

$$H(z, W) = w_i \times z_i + b \tag{4}$$

$$\hat{z} = H(z, W) + z \tag{5}$$

Finally, the decoder processes \hat{z} to reconstruct x. An autoencoder with a residual structure introduces an adaptive weight layer on the latent feature z and forms a residual structure with the previous z. In the reconstruction training, it has the following two advantages: (1) the weight layer is equivalent to the "noise" added on z, preventing overfitting; (2) The residual structure makes the potential representation of the network tend to be between z and the



Fig. 2 The framework of the residual autoencoder

noise version of z in the gradient backpropagation, so that the applied weight does not worry about destroying the original feature information, but also ensures the great flexibility of the network, so that it can be optimized according to specific clustering tasks.

A basic network is trained by the reconstruction loss of the residual autoencoder as shown in Eq. (6). The initial cluster center $\{\mu_j\}_{j=1}^K$ of z is obtained by K-means clustering.

$$L_r(\tilde{x}_{\hat{z}_i}, x_i) = \|x_i - \tilde{x}_{\hat{z}_i}\|_2^2$$
(6)

3.2 Network Fine-Tuning

In the network fine-tuning phase, we also borrowed the idea of the t-SNE algorithm [29], but no longer focused on measuring the similarity between data points in the initial space of data. As with DEC, the similarity is measured using the student t-SNE distribution function between the embedding point z_i in the pre-trained potential space and the cluster center μ_i :

$$q_{ij} = \frac{\left(1 + \|z_i - \mu_j\|_2^2\right)^{-1}}{\sum\limits_{j' \neq i} \left(1 + \|z_i - \mu_{j'}\|_2^2\right)^{-1}}$$
(7)

where $z_i = f_{\theta}(x_i)$ corresponds to x_i after embedding, and μ_j represents the *j*-th cluster center in the embedding space. q_{ij} can be understood as the probability of classifying the *i*-th feature z_i into class *j*, and $q_i = [q_{i1}, q_{i2}, \dots, q_{iK}]^T$ is the soft assignment probability of the data point x_i .

Consistent with the general deep embedding clustering algorithm, the objective allocation function P is used to learn the clustering division with higher certainty in the clustering soft distribution function Q. In the distribution function Q, the meaning of the probability value is the similarity between z_i and μ_j . The more similar the two are, the higher the probability that the data point x_i will be assigned to j-th category. Therefore, the distribution with high confidence in the distribution function usually has higher clustering accuracy and can better reflect the real clustering results. The objective allocation function P is defined as follows:

$$p_{ij} = \frac{q_{ij}^2 / \Sigma_i q_{ij}}{\sum\limits_{j' \neq i} \left(q_{ij'}^2 / \Sigma_i q_{ij'} \right)}$$
(8)

Using the above two distribution functions, we construct the loss function of the cluster by the KL divergence of them:

$$L_c = KL(P || Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$
(9)

Combined with Eqs. (6) and (9), the autoencoder with residual structure and clustering module are jointly optimized. We use the fully connected layer to implement the encoder and decoder modules. The overall loss function is defined as:

$$L = L_r + \gamma L_c \tag{10}$$

where L_r means reconstruction loss and L_c is clustering loss, $\gamma > 0$ is the proportional coefficient controlling reconstruction and clustering. Through fine-tuning training of the

model, the final clustering result is a label that assigns potential feature point z_i to cluster center μ_i with the maximum probability q_{ii} :

$$s_i = \arg\max_j q_{ij} \tag{11}$$

3.3 Network Optimization

We analyze the optimization part of the network more details in this section. We use smallbatch optimization algorithm combined with backpropagation algorithm for Eq. (10), and iteratively optimize cluster center μ_j , residual autoencoder parameters θ_e , θ_d , and adaptive weight layer parameters W. It mainly includes two parts: reconstruction loss optimization and clustering loss optimization. For a complete description of the DECRA algorithm in this paper, see Algorithm 1.

Algorithm 1 Deep embedding clustering based on residual autoencoder Algorithm

```
Input:
          Input dataset: X; Category number: K; Maximum iteration: MaxIter; Stop threshold: ε; Update interval: T.
Output: Encoder network parameter \theta_{a}; Decoder network parameter \theta_{d}; Adaptive weight layer parameter W;
          Cluster allocation s
Initialize \Theta_e, \Theta_d, \mu and W on the basis of Section 3.1
for iter = 0 to MaxIter
    if iter\%T == 0 then
        i. Calculate all embedded points z_i
        ii.Update P according to Eqs. (7-8)
        iii. Save the last label assignment: s_{old} = s
        iv. Calculate the new label assignment s according to Eq. (11).
        v. if sum(s_{old} \neq s)/n < \varepsilon then
                Stop
     Select a small-batch of data S \in X
     Update \Theta_e, \Theta_d, \mu and W on S according to Section 3.3.1 and Section 3.3.2
end for
```

3.3.1 Reconstruction Loss Optimization

The residual autoencoder proposed in this paper can be divided into two steps for the backpropagation process of reconstruction loss:

(1) The fist branch gradient backpropagation without adaptive weight layer, that is, the traditional autoencoder derivation process:

$$\frac{\partial L_{\mathbf{r}}}{\partial \theta_{\mathbf{d}}} = 2 \sum_{i=1}^{m} \left(x_i - \tilde{x}_{\hat{z}_i} \right) \times \frac{\partial \tilde{x}_{\hat{z}_i}}{\partial \theta_{\mathbf{d}}}$$
(12)

$$\frac{\partial L_{\mathbf{r}}}{\partial \theta_{e}} = 2 \sum_{i=1}^{m} \left(x_{i} - \tilde{x}_{\hat{z}_{i}} \right) \times \frac{\partial \tilde{x}_{\hat{z}_{i}}}{\partial z_{i}} \times \frac{\partial z_{i}}{\partial \theta_{e}}$$
(13)

(2) The novel branch gradient backpropagation with adaptive weight layer proposed in this paper is derived for θ_e , θ_d and W respectively as follows:

$$\frac{\partial L_{\mathbf{r}}}{\partial \theta_{\mathbf{d}}} = 2 \sum_{i=1}^{m} \left(x_i - \tilde{x}_{\hat{z}_i} \right) \times \frac{\partial \tilde{x}_{\hat{z}_i}}{\partial \theta_{\mathbf{d}}}$$
(14)

$$\frac{\partial L_{\mathbf{r}}}{\partial W} = 2 \sum_{i=1}^{m} \left(x_i - \tilde{x}_{\hat{z}_i} \right) \times \frac{\partial \tilde{x}_{\hat{z}_i}}{\partial H(z, W)} \times \frac{\partial H(z, W)}{\partial W}$$
(15)

$$\frac{\partial L_{\mathbf{r}}}{\partial \theta_{e}} = 2 \sum_{i=1}^{m} \left(x_{i} - \tilde{x}_{\hat{z}_{i}} \right) \times \frac{\partial \tilde{x}_{\hat{z}_{i}}}{\partial H(z, W)} \times \frac{\partial H(z, W)}{\partial z_{i}} \times \frac{\partial z_{i}}{\partial \theta_{e}}$$
(16)

Then, the corresponding parameters are updated. It can be seen that branch (1) constantly adjusts the network parameters according to the original sample to make the reconstructed data approximate to it. In this process, the potential features can be learned and the feature transformation of the input raw data can be realized. Branch (2) is introduced in this paper with an adaptive weight layer, which is equivalent to introducing a "noise" term and selectively controlling the expression of z, so as to prevent the network from fully characterizing the sample in the reconstruction training, and enhance the network generalization ability. The combination of the noise version of z, so that the network does not have to worry about the applied weight destroying the original feature information, but also ensures the great flexibility of the network, so that it can be optimized for specific clustering tasks.

3.3.2 Clustering Loss Optimization

In the process of clustering loss optimization, for the sake of avoiding the numerical instability caused by the simultaneous updating of P and Q, we adopt the strategy of alternate updating. When P is fixed, the gradient of L_c with respect to z_i , μ_i and θ_e can be calculated as:

$$\begin{split} \frac{\partial L_{c}}{\partial z_{i}} &= \frac{\partial \mathrm{KL}(\mathbf{P}||\mathbf{Q})}{\partial z_{i}} = \frac{\partial \sum_{i=1}^{m} \sum_{j=1}^{K} \left(p_{ij} \log p_{ij} - p_{ij} \log q_{ij} \right)}{\partial z_{i}} = -\sum_{j=1}^{K} p_{ij} \frac{\partial \left(\log q_{ij} \right)}{\partial z_{i}} \\ &= \sum_{j=1}^{K} p_{ij} \left[-\partial \left(\log \left(1 + \left\| z_{i} - \mu_{j} \right\|^{2} \right)^{-1} \right) \times \frac{1}{\partial z_{i}} + \partial \left(\log \sum_{j'} \left(1 + \left\| z_{i} - \mu_{j'} \right\|^{2} \right)^{-1} \right) \times \frac{1}{\partial z_{i}} \right] \\ &= \sum_{j=1}^{K} p_{ij} \frac{2(z_{i} - \mu_{j})}{1 + \left\| z_{i} - \mu_{j} \right\|^{2}} - \sum_{j=1}^{K} p_{ij} \times 2 \left(\sum_{j'} \left(z_{i} - \mu_{j'} \right) \left(1 + \left\| z_{i} - \mu_{j'} \right\|^{2} \right)^{-2} \right) \times \frac{1}{\sum_{j'} \left(1 + \left\| z_{i} - \mu_{j'} \right\|^{2} \right)^{-1}} \\ &= \sum_{j=1}^{K} p_{ij} \frac{2(z_{i} - \mu_{j})}{1 + \left\| z_{i} - \mu_{j} \right\|^{2}} - 2 \left(\sum_{j'} \left(z_{i} - \mu_{j'} \right) \left(1 + \left\| z_{i} - \mu_{j'} \right\|^{2} \right)^{-2} \right) \times \frac{1}{\sum_{j'} \left(1 + \left\| z_{i} - \mu_{j'} \right\|^{2} \right)^{-1}} \\ &= \sum_{j=1}^{K} p_{ij} \frac{2(z_{i} - \mu_{j})}{1 + \left\| z_{i} - \mu_{j} \right\|^{2}} - 2 \sum_{j'} \left(z_{i} - \mu_{j'} \right) \left(1 + \left\| z_{i} - \mu_{j'} \right\|^{2} \right)^{-2} \times \frac{1}{\sum_{j'} \left(1 + \left\| z_{i} - \mu_{j'} \right\|^{2} \right)^{-1}} \end{split}$$

Deringer

1.

$$= \sum_{j=1}^{K} p_{ij} \frac{2(z_i - \mu_j)}{1 + ||z_i - \mu_j||^2} - 2 \sum_{j'} (z_i - \mu_{j'}) (1 + ||z_i - \mu_{j'}||^2)^{-1} \times q_{ij'}$$

= $2 \times \sum_{j=1}^{K} \frac{(z_i - \mu_j)(p_{ij} - q_{ij})}{1 + ||z_i - \mu_j||^2}$ (17)

By the same token, we can get:

$$\frac{\partial L_c}{\partial \mu_j} = -2 \times \sum_{j=1}^K \frac{(z_i - \mu_j)(p_{ij} - q_{ij})}{1 + \|z_i - \mu_j\|^2}$$
(18)

Finally, the gradient for the encoder parameter θ_e is obtained:

$$\frac{\partial L_{\rm c}}{\partial \theta_{\rm e}} = \frac{\partial L_{\rm c}}{\partial z_i} \times \frac{\partial z_i}{\partial \theta_{\rm e}} \tag{19}$$

3.3.3 Update Target Distribution P

The update of P is different from the above process. The distribution of cluster labels is determined by the soft label distribution Q, but the definition of P itself is also related to Q. Therefore, in order to prevent excessive fluctuation in the iteration process, this paper updates the target distribution P with all data points in the potential space in every T iteration. The update rules are shown in Eqs. (7) and (8). When P is updated, the label assigned to x_i is calculated according to Eq. (11). If the label allocation change (percentage) is less than the threshold ε between two consecutive updates, the training terminates.

4 Experiment

4.1 Datasets

For verifying the effectiveness of our DECRA algorithm, we experimentally evaluated it on six high-dimensional datasets covering the domains of handwritten digits, objects, and text as follows:

- MNIST [30]: This dataset consists of a total of 70,000 images of size 28 × 28 pixels handwritten digits.
- Fashion-MNIST [31]: This dataset contains 60,000 training images and 10,000 testing images, each of which is displayed in grayscale with a size of 28 × 28 pixels
- REUTERS-10 K [32]: REUTERS dataset has about 810,000 English news stories annotated with classification trees. The REUTERS-10 k dataset used in this paper, derived from DEC, consists of 10,000 samples.
- MNIST-test [30]: REUTERS dataset consists of a total of 10,000 images of size 28 × 28 pixels handwritten digits.
- USPS [33]: This dataset consists of 9298 grayscale images of size 16 × 16 pixels handwritten digits.
- COIL20 [34]: The dataset consists of 1440 pieces of data from 20 different objects (K = 20) 32 × 32 image composition, 72 images per object, taken at 5-degree pose intervals.

The detail information about those datasets is presented in Table

Dataset	Examples	Dimension	Classes
MNIST	70,000	784	10
Fashion-MNIST	70,000	784	10
REUTERS-10 K	10,000	2000	4
MNIST-test	10,000	784	10
USPS	9298	256	10
COIL20	1440	1024	20
	Dataset MNIST Fashion-MNIST REUTERS-10 K MNIST-test USPS COIL20	DatasetExamplesMNIST70,000Fashion-MNIST70,000REUTERS-10 K10,000MNIST-test10,000USPS9298COIL201440	Dataset Examples Dimension MNIST 70,000 784 Fashion-MNIST 70,000 784 REUTERS-10 K 10,000 2000 MNIST-test 10,000 784 USPS 9298 256 COIL20 1440 1024

4.2 Clustering Index

In order to evaluate the clustering results, we adopt three evaluation indexes that are widely used in cluster analysis: Accuracy (ACC), Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI).

The ACC evaluation indicator is defined as follow:

$$ACC = \max_{m} \frac{\sum_{i=1}^{n} \mathbf{1}\{l_{i} = m(c_{i})\}}{n}$$
(20)

where l_i and c_i are the true labels and predictive clustering of data points x_i , respectively. For unsupervised clustering algorithms, the Hungarian algorithm represented by m(.) is used to compute the best mapping between the cluster assignment and the real label.

The NMI normalization metric measures the similarity between the true label and the predicted label of data, defined as:

$$NMI = \frac{I(l;c)}{\max\{H(l), H(c)\}}$$
(21)

where I(l, c) represents the mutual information between l and c, where l is the true label of the data, c is the label predicted by the clustering algorithm, and H represents their entropy. The results of the NMI are not changed by the arrangement of clusters (classes), they are normalized to a range of [0,1], where '0' means that there is no correlation between the real label and the predicted label, and '1' means that there is a complete correlation between them.

The ARI evaluation indicator is defined as:

$$ARI = \frac{\text{RI} - E(\text{RI})}{\max(\text{RI}) - E(\text{RI})}$$
(22)

where E(RI) represents the expectation of the Rand Index (RI). Each three indicators measure the effectiveness of the algorithm from different angles. Combining the above three indicators, we can fully understand the performance of all aspects of the algorithm.

4.3 Experimental Settings

In this paper, Our DECRA algorithm is based on the deep learning framework Tensorflow-1.15 and the operating system is ubuntu-20.04. NVIDIA Graphics processing Units (GPUs) are used to accelerate deep learning calculations. we use the fully connected layer to implement the encoder and decoder modules, and set the encoder nodes in every layer are *d*-500-500-1000-10, where *d* is the feature dimension. The decoder nodes are symmetrical with the encoder, that is, 10-1000-500-500-*d*. Except the input, output and embedding layers, the activation functions used in all hidden layers of the encoder are ReLu functions [35]. In the experiments, for pre-training on all datasets, the batch size is set to be 256. We adopt Adam [36] as the optimization algorithm, and set the two parameters of Adam to be $\beta 1 = 0.9$ and $\beta 2 = 0.999$. The learning rate of the REUTERS-10K dataset is set to $\lambda = 0.0001$, while on other four datasets is set to be $\lambda = 0.001$. And the convergence threshold is set to be $\epsilon = 0.01\%$. For more code details, see the Availability of data and materials section, where we share the complete, executable code.

We select two traditional algorithms and several typical deep clustering algorithms to compare the clustering performance with our DECRA algorithm. That includes K-means [37], SC [12], AE + K-means [38], DEC [28], SDEC [39], IDEC [18], DCEC [16] and IDCEC [19] algorithms.

4.4 Experimental Results

The performance comparison of each algorithm on the six benchmark datasets are shown in Tables 2, 3 and 4. The table values are derived from the results reported in the corresponding paper or by running their code. If an algorithm is not suitable for a specific dataset, its corresponding result is "N/A". We highlight the best results in bold. From these results, we can draw the following points.

- Generally, all deep clustering algorithms, including AE + k-means, DEC, SDEC, IDEC, DCEC, IDCEC and DECRA, outperform the two traditional clustering methods, k-means and spectral clustering (SC) algorithm, which reveals that deep clustering algorithms has more application prospects than traditional.
- 2. When the strategies of feature learning are basically the same, the clustering performance of the deep clustering algorithms using the unified optimization framework, including DEC, SDEC, IDEC, DCEC, IDCEC and DECRA, are generally better than that of the methods which carry out feature learning and clustering analysis in separated steps, such as AE + k-means method. In AE + k-means, the features Learning and clustering analysis are carried out separately, so that the feature learning process cannot be guided by clustering analysis, and thus the learned features may not be optimal for subsequent clustering tasks.
- 3. Since IDEC proposed to add reconstruction loss in training to maintain the local structure of data, deep-embedded clustering algorithms have followed this line, such as DCEC, IDCEC and so on. Of course, it plays a positive role in preventing the inherent structural damage of data caused by clustering partitioning learning. However, it also leads to a new problem, that is, overemphasis on data reconstruction may hinder the generalization of potential expressions. As there are often a large amount of noise in the data, blind pursuit of reconstruction will greatly limit the further expression of potential features according to specific clustering tasks. The comparison results fully reflect that our algorithm achieves the best performance on all datasets, and achieves a substantial improvement on some datasets, such as USPS increasement of about 14%. These results demonstrate that the residual autoencoder proposed in this paper can learn high-quality latent expressions, which are more flexible and conducive to generalization compared to previous methods.
- 4. It is worth mentioning that our algorithm not only works well on image datasets, but also achieves remarkable results on text dataset that is difficult to process. The text dataset Reuters-10K uses a simple tf-idf feature extraction method, resulting in a very sparse

METHODS	MNIST	Fashion-mnist	Reuters-10 k	MNIST-test	USPS	COIL20
V	0.522	NT/ A	NT/ A	0.542	0.00	NI/A
K-means	0.532	N/A	N/A	0.542	0.008	IN/A
SC	0.680	0.551	N/A	0.667	0.656	0.488
AE + k-means	0.741	N/A	0.749	N/A	0.710	N/A
DEC	0.847	0.536	0.711	0.776	0.733	0.576
SDEC	0.861	0.569	0.723	N/A	0.764	0.592
IDEC	0.881	0.552	0.729	0.791	0.762	0.599
DCEC	0.874	0.508	N/A	0.852	0.739	0.683
IDCEC	0.948	0.579	N/A	0.883	0.812	0.695
DECRA	0.964	0.663	0.825	0.890	0.951	0.735

Table 2 ACC of compared algorithms on six benchmark datasets

The bold text represent our algorithm's results

Table 3 NMI of compared algorithms on six benchmark datasets

METHODS	MNIST	Fashion-mnist	Reuters-10 k	MNIST-test	USPS	COIL20
K-means	0.499	N/A	N/A	0.5	0.626	N/A
SC	0.759	0.63	N/A	0.712	0.796	0.772
AE + k-means	0.723	N/A	0.498	N/A	0.675	N/A
DEC	0.791	0.591	0.501	0.724	0.706	0.765
SDEC	0.829	0.628	0.586	N/A	0.777	0.772
IDEC	0.867	0.604	0.469	0.736	0.756	0.770
DCEC	0.851	0.600	N/A	0.809	0.739	0.798
IDCEC	0.906	0.641	N/A	0.877	0.858	0.810
DECRA	0.917	0.656	0.592	0.879	0.885	0.829

The bold text represent our algorithm's results

Table 4 ARI of compared algorithms on six benchmark datasets

METHODS	MNIST	Fashion-mnist	Reuters-10 k	MNIST-test	USPS	COIL20
K-means	0.365	N/A	N/A	0.378	0.545	N/A
SC	0.625	0.422	N/A	0.563	0.650	0.407
AE + k-means	0.647	N/A	0.496	N/A	0.588	N/A
DEC	0.749	0.420	0.498	0.66	0.637	0.556
SDEC	0.792	0.451	0.556	N/A	0.699	0.565
IDEC	0.850	0.442	0.538	0.666	0.679	0.575
DCEC	0.819	0.418	N/A	0.773	0.655	0.613
IDCEC	0.889	0.468	N/A	0.841	0.772	0.652
DECRA	0.924	0.530	0.642	0.846	0.907	0.681

The bold text represent our algorithm's results

feature matrix with most elements being zero and only a few elements containing simple word frequency information. Some of the previous algorithms could not handle text features or did not achieve very good performance. Even so, our algorithm achieved a significant improvement of 10%. It shows that the proposed algorithm is universal to all kinds of datasets.

In Fig. 3, a more intuitive comparison is shown. It can also be seen that the DECRA algorithm obtains impressive effects in the same type of algorithm, and acquires a large gap with other algorithms. In summary, the experimental results show the superiority of DECRA compared with the most advanced deep clustering algorithm.



Fig. 3 Visualization of algorithm performance on different datasets



Fig. 4 ACC curves of IDEC and DECRA on MNIST

We show the evolution process of both clustering performances on MNIST in Fig. 4 so as to further compare the performance improvement of our improved DECRA algorithm with IDEC algorithm. It can be seen that the final accuracy is the same as Table 2, that is, DECRA is better than IDEC, and after about 20 iterations, the accuracy of IDEC algorithm is no longer improved, indicating that the potential expression at this time can no longer learn discriminant features for clustering. The reasons are as follows: (1) It may be that the autoencoder reconstructs the noise of the sample into z without distinction, resulting in poor clustering due to the influence of noise. (2) It may be that the reconstruction is too strong, limiting the further expression of z according to specific clustering. In contrast, due to its unique and flexible structure, DECRA algorithm has robustness to noise and generalization to clustering tasks, and its accuracy has been improved until the 150th iteration, and finally converges stably to 0.964.

Figure 5 shows the visualization effect of evolution process of the learned latent representation on training. Taking the MNIST dataset for example, we randomly select 1000 sample and map the latent representation z to a two-dimensional space by t-SNE. From the trend of evolution, we can see that with the increase of training epochs, data points belong to different clusters are separated away and easier to distinguish, while the distribution of data points from same cluster are more compact. This suggests that the learned feature space is more suitable for clustering tasks.



Fig. 5 Visualization of MNIST during training process with t-SNE

Method	Lr	Res	MNIST	Fashion-mnist	Reuters-10 k	MNIST-test	USPS	COIL20
Benchmark	×	×	0.847	0.536	0.711	0.776	0.733	0.576
-	\checkmark	×	0.881	0.552	0.729	0.791	0.762	0.599
DECRA	\checkmark	\checkmark	0.964	0.663	0.825	0.890	0.951	0.735

Table 5 Comparison of ACC in ablation experiment

The bold text represent our algorithm's results

Table 6 Comparison of NMI in ablation experiment

Method	Lr	Res	MNIST	Fashion-mnist	Reuters-10 k	MNIST-test	USPS	COIL20
Benchmark	×	×	0.791	0.591	0.501	0.724	0.706	0.765
-	\checkmark	×	0.867	0.604	0.469	0.736	0.756	0.770
DECRA	\checkmark	\checkmark	0.917	0.656	0.592	0.879	0.885	0.829

The bold text represent our algorithm's results

Table 7 Comparison of ARI in ablation experiments

Method	Lr	Res	MNIST	Fashion-mnist	Reuters-10 k	MNIST-test	USPS	COIL20
Benchmark	×	×	0.749	0.420	0.498	0.66	0.637	0.556
-	\checkmark	×	0.850	0.442	0.538	0.666	0.679	0.575
DECRA	\checkmark	\checkmark	0.924	0.530	0.642	0.846	0.907	0.681

The bold text represent our algorithm's results

4.5 Ablation Experiment

In order to further analyze and demonstrate the validity of the residual structure proposed in this paper, an ablation experiment is carried out in this section. As shown in Tables 5, 6, 7, we showed the comparison results of clustering performance after ablation of key structures on three quantitative indicators for the 6 datasets used. Where "Lr" and "Res" respectively indicate that the training maintains reconstruction constraints and residual structures, and whether there is a corresponding structure is marked with " \times " or " $\sqrt{}$ ". It can be intuitively seen that maintaining reconstruction constraints in training is conducive to maintaining the local intrinsic structure of the original data to a certain extent, avoiding damage during clustering partitioning learning, and improving performance to a certain extent. However, we have also noticed that there is insufficient performance improvement or even retrogressive phenomenon in some indicators, such as Reuters-10k retrogressive 4% in NMI indicators. This is most likely due to the original characterization of the data (including noise) during auto-encoder reconstruction, which makes the model not robust enough. In contrast, our autoencoder with residual structure has achieved optimal performance on all datasets and indicators in the table, which reflects that the autoencoder with residual structure in this paper has good robustness and generalization for specific clustering tasks.

5 Conclusion

In this paper, we propose a novel residual autoencoder framework for deep clustering task, named deep embedding clustering algorithm based on residual autoencoder (DECRA). This model can effectively address the problem that the feature generalization ability is insufficient due to overemphasis on reconstruction loss training, and the clustering effect is not good due to the existence noise. At the same time, we introduce an adaptive weight layer that enables adaptive learning based on category classification, resulting in higher-quality feature representations. Feature learning and cluster analysis are carried out under a unified framework, so that they promote each other to achieve the overall optimal clustering result. The effectiveness of the proposed method is verified by comprehensive experiments on 6 public benchmark datasets.

Acknowledgements Thanks to Guilin University, Guangxi University of Science and Technology and other relevant researchers for their support of this paper.

Authors' Contributions ML: Conceptualization, Methodology, Software, Writing draft and Visualization. CC: Software Validation and Editing draft. SY: Reviewing and Editing, Supervision. CL: Funding acquisition, Project administration. All authors read and approved the final manuscript.

Funding This work was supported by the National Natural Science Foundation of China under Grants (62062010, 62061003); and the Basic Ability Promotion Project of Guangxi Middle and Young University Teachers (2023KY1638).

Data and Materials Availability The datasets and codes are available at https://github.com/lml-mengli/DECRA.

Declarations

Conflict of interest We promise that this manuscript is the authors' original work and has not been published nor has it been submitted simultaneously elsewhere. All authors have checked the manuscript and have agreed to the submission.

Ethical Approval Not applicable.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

- Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell 22:888–905
- Chen Y, Wang JZ, Krovetz R (2005) Clue: cluster-based retrieval of images by unsupervised learning. IEEE Trans Image Process 14:1187–1201
- Goldberger J, Gordon S, Greenspan H (2006) Unsupervised image-set clustering using an information theoretic framework. IEEE Trans Image Process 15:449–458
- 4. Xie P, Xing EP (2015) Integrating image clustering and codebook learning. In: Proceedings of the AAAI conference on artificial intelligence, 29(1)

- Xu Z, Yan F, Qi Y (2013) Bayesian nonparametric models for multiway data analysis. IEEE Trans Pattern Anal Mach Intell 37:475–487
- Zhe S, Xu Z, Chu X, Qi Y, Park Y (2015) Scalable nonparametric multiway data analysis. In: Artificial intelligence and statistics, pp. 1125–1134
- Xu Z, Liu B, Zhe S, Bai H, Wang Z, Neville J (2018) Variational random function model for network modeling. IEEE Trans Neural Netw Learn Syst 30:318–324
- Kim HK, Kim H, Cho S (2017) Bag-of-concepts: comprehending document representation through clustering words in distributed representation. Neurocomputing 266:336–352
- Huang S, Xu Z, Lv J (2018) Adaptive local structure learning for document co-clustering. Knowl-Based Syst 148:74–84
- MacQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, pp. 281–297
- Agrawal R, Gehrke J, Gunopulos D, Raghavan P (1998) Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of the 1998 ACM SIGMOD international conference on Management of data, pp. 94–105
- 12. Ng A, Jordan M, Weiss Y (2001) On spectral clustering: analysis and an algorithm. Adv Neural inf Process Syst 14
- Xu W, Liu X, Gong Y (2003) Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 267–273
- 14. He X, Cai D, Niyogi P (2005) Laplacian score for feature selection. Adv Neural Inf Process Syst 18
- De la Torre F, Kanade T (2006) Discriminative cluster analysis. In: Proceedings of the 23rd international conference on Machine learning, pp. 241–248
- Guo X, Liu X, Zhu E, Yin J (2017) Deep clustering with convolutional autoencoders. In: Neural Information Processing: 24th International conference, pp. 373–382
- Chen R, Tang Y, Zhang W, Feng W (2022) Deep multi-view semi-supervised clustering with sample pairwise constraints. Neurocomputing 500:832–845
- Guo X, Gao L, Liu X, Yin J (2017) Improved deep embedded clustering with local structure preservation. Ijcai 17:1753–1759
- Lu H, Chen C, Wei H, Ma Z, Jiang K, Wang Y (2022) Improved deep convolutional embedded clustering with re-selectable sample training. Pattern Recogn 127:108611
- Chang M-C, Bus P, Schmitt G (2017) Feature extraction and k-means clustering approach to explore important features of urban identity. In: IEEE International conference on machine learning and applications (ICMLA), pp. 1139–1144
- Cai D, Chen X (2014) Large scale spectral clustering via landmark-based sparse representation. IEEE Trans Cybern 45:1669–1680
- Gu J, Jiao L, Yang S, Liu F (2017) Fuzzy double c-means clustering based on sparse self-representation. IEEE Trans Fuzzy Syst 26:612–626
- Huang G, Song S, Gupta JN, Wu C (2014) Semi-supervised and unsupervised extreme learning machines. IEEE Trans Cybern 44:2405–2417
- Zhang Y, Wu J, Zhou C, Cai Z, Yang J, Yu PS (2019) Multi-view fusion with extreme learning machine for clustering. ACM Trans Intell Syst Technol (TIST) 10:1–23
- Chen J, Zeng Y, Li Y, Huang G-B (2020) Unsupervised feature selection based extreme learning machine for clustering. Neurocomputing 386:198–207
- Chen C, Vong C-M, Wong P-K, Tai K-I (2020) Approximate empirical kernel map-based iterative extreme learning machine for clustering. Neural Comput Appl 32:8031–8046
- 27. Huang P, Huang Y, Wang W, Wang L (2014) Deep embedding network for clustering.In: International conference on pattern recognition, pp. 1532–1537
- Xie J, Girshick R, Farhadi A (2016) Unsupervised deep embedding for clustering analysis. In: International conference on machine learning, pp. 478–487
- Van Der Maaten L (2009) Learning a parametric embedding by preserving local structure. In: Artificial intelligence and statistics, pp. 384–391
- LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324
- Xiao H, Rasul K, Vollgraf R (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747
- Lewis DD, Yang Y, Russell-Rose T, Li F (2004) Rcv1: A new benchmark collection for text categorization research. J Mach Learn Res 5:361–397
- Bard JF, Morton DP, Wang YM (2007) Workforce planning at USPS mail processing and distribution centers using stochastic optimization. Ann Oper Res 155:51–78

- 34. Nene SA, Nayar SK, Murase H (1996) Columbia object image library (coil-20), 223-303
- 35. Glorot X, Bordes A, Bengio Y (2011) Deep sparse rectifier neural networks. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics, pp. 315–323
- 36. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980
- Arthur D, Vassilvitskii S (2007) K-means++ the advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 1027–1035
- Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313:504–507
- Ren Y, Hu K, Dai X, Pan L, Hoi SC, Xu Z (2019) Semi-supervised deep embedded clustering. Neurocomputing 325:121–130

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.