Applied Intelligence (2020) 50:1155–1171
https://doi.org/10.1007/s10489-019-01581-7

Unsupervised discriminative feature representation via adversarial auto-encoder

Wenzhong Guo[1] · **Jinyu Cai**[1] · **Shiping Wang**[1]

Published online: 24 December 2019

Abstract

Feature representation is generally applied to reducing the dimensions of high-dimensional data to accelerate the process of data handling and enhance the performance of pattern recognition. However, the dimensionality of data nowadays appears to be a rapidly increasing trend. Existing unsupervised feature representation methods are susceptible to the rapidly increasing dimensionality of data, which may result in learning a meaningless feature that in turn affect their performance in other applications. In this paper, an unsupervised adversarial auto-encoder network is studied. This network is a probability model that combines generative adversarial networks and variational auto-encoder to perform variational inference and aims to generate reconstructed data similar to original data as much as possible. Due to its adversarial training, this model is relatively robust in feature learning compared with other methods. First, the architecture and training strategy of adversarial auto-encoder are presented. We attempt to learn a discriminative feature representation for high-dimensional image data via adversarial auto-encoder and take its advantage into image clustering, which has become a difficult computer vision task recently. Then amounts of comparative experiments are carried out. The comparison contains eight feature representation methods and two recently proposed deep clustering methods performed on eight different publicly available image data sets. Finally, to evaluate their performance, we utilize a K-means clustering on the low-dimensional feature learned from each feature representation algorithm, and select three evaluation metrics including clustering accuracy, adjusted rand index and normalized mutual information, to provide a comparison. Comprehensive experiments prove the usefulness of the learned discriminative feature via adversarial auto-encoder in the tested data sets.

Keywords Unsupervised learning · Image clustering · Feature representation · Adversarial auto-encoder · Generative adversarial networks

1 Introduction

Nowadays, the data information in many research areas like object detection [13], text categorization [23, 26] and face recognition [2, 40], is always provided with a high dimension. Thus many difficulties in information processing have led to extensive discussions on how to compress the big data. It is well known that a learning model could not see or hear input data directly. Instead, it needs to learn the representation of the original data information for providing useful compressed data to the model. Feature representation [42] is a method that aims to deal with such big data efficiently by representing these data in a low-dimensional form. It can be broadly divided into supervised, semi-supervised and unsupervised based methods. Although the performance of supervised methods in tests is generally better than the unsupervised one, supervised methods require all the label information of test data, which will spend a huge amount of efforts to label big data, let alone some of them are difficult to label. Hence the potential of dealing with the 'curse of dimensionality' by unsupervised feature representation methods is worth exploring.

The classic unsupervised feature representation methods contain two forms, namely linear based and non-linear

✉ Shiping Wang
shipingwangphd@163.com

Wenzhong Guo
guowenzhong@fzu.edu.cn

Jinyu Cai
jinyucai1995@gmail.com

[1] College of Mathematics and Computer Science, Fuzhou University, Fujian, 350116, China

based methods, respectively. Principal component analysis (PCA) and locality preserving projections (LPP) are two algorithms based on linear transformation [20, 43]. PCA is the most common method for linear feature representation partly due to its simplicity. Different from PCA, LPP mainly puts emphasis on retaining the neighborhood structure between input data points, not the global structure. Furthermore, isometric feature mapping (Isomap) and neighborhood preserving embedding (NPE) are two algorithms based on non-linear transformation [1, 19]. The central ideology of Isomap is to reserve geodesic distances among all the input data pairs as much as possible through defining a low-dimensional embedding. While NPE aims to seek out the local neighborhood structure between data points, it can be considered as a variant of locally linear embedding (LLE) [37].

Auto-encoder neural network [21] is another feature representation method, whose architecture consists of an input layer, a hidden layer and an output layer. The network utilizes a linear objective function for linear projection and a sigmoid function to realize non-linear mapping. Variants of auto-encoder network have received many achievements in unsupervised learning in the past decade. For instance, Xie et al. [45] proposed the deep embedded clustering method (DEC) which is based on the auto-encoder network, to learn a clustering oriented feature representation by jointly optimizing the clustering and low-dimensional representation learning, and obtained inspiring clustering performance. Furthermore, Guo et al. [17] noticed the shortcoming of DEC that it lacks the retention of local structure and may result in the distortion of feature space. To this end, they proposed the improved embedded clustering method (IDEC), which simultaneously applies the clustering loss and reconstruction loss in the training process to avoid the distortion of feature space.

According to the idea of combining the variational auto-encoder with the generative adversarial networks (GAN) [9, 15, 49], Makhzani et al. proposed a method called adversarial auto-encoder (AAE) [25, 28]. The structure of GAN consists of a generative network and a discriminative network. During the training process, the generative network is trained to fool the discriminator into believing that the generated samples are true samples, while the discriminative network aims to improve its ability to distinguish the authenticity of samples. Generally, AAE can be partitioned into two stages: the reconstruction stage and the regularization stage. Reconstruction stage aims to minimize the reconstruction loss of input data, while the target of regularization stage is to update the parameters of generator and discriminator. In recent years, the application of AAE in feature representation research field has been studied widely. Barone et al. [3] proposed a method that uses AAE to learn a cross-lingual oriented distributed representation without training the parallel text. That was different from current approaches which need an amount of parallel text to be used to learn a representation which is compatible between different languages, and it creates a potentially promising field in natural language processing research. Moreover, Zhifei Zhang et al. used the proposed conditional adversarial auto-encoder to learn a latent representation for human face through the imposed adversarial training on encoder and generator [47]. This learned latent representation well retains the personalized features of human faces so that it can achieve a glossy age progression and obtain state-of-the-art performance.

However, the application of AAE in image clustering is rarely studied. The main purpose of clustering methods [18, 31, 41] is to partition input data into multiple clusters without the guidance of label information, and join the points in the same category as close to each other as much as possible, while separating the points in different categories. Some clustering methods have been smoothly applied to text clustering for enhancing the performance of its applications, but image clustering is still a difficult computer vision task because of its the pixel-level based characteristics. Since the image data today often owns a high dimension, most images need to be processed before clustering. Taking the feature representation method as an instance, some methods could not preserve the pixel feature well after the original data is reduced to a relatively low dimension, which leads to unsatisfactory performance of image clustering. Thus how to learn an encouraging representation from high-dimensional image data is a topic worthy of further study.

In this paper, we attempt to learn a discriminative feature representation via adversarial auto-encoder. First, the architecture and training strategy of adversarial auto-encoder are presented, and for the fairness of comparative experiments, a publicly available auto-encoder network and GAN structure are utilized. Then adversarial auto-encoder is applied to learning a discriminative feature representation for image data and compared with seven feature representation methods by learning the low-dimensional feature on eight publicly available image data sets. Especially, in order to prove the usefulness of the learned discriminative feature representation, we also compare the performance of AAE with two recently proposed deep clustering methods. For the purpose of assessing their performance, we perform image clustering through K-means method [18] on the feature learned from each algorithm, and select three evaluation metrics including clustering accuracy (ACC) [22, 34], normalized mutual information (NMI) [10] and adjusted rand index (ARI) [39], to provide comparison. Finally, comprehensive experiments illustrate the effectiveness of the learned discriminative feature via an unsupervised adversarial auto-encoder algorithm in the tested data sets.

The main contribution of this work is summed up as follows:

- This paper is an attempt to apply the adversarial auto-encoder (AAE) network to learn the discriminative feature for the image clustering task, which has never been studied before. This work could provide a new perspective to study unsupervised feature representation methods.
- For the purpose of providing the readers a clearer understanding of the unsupervised feature representation, we have divided the unsupervised feature representation methods into three categories (linear transformation based, non-linear transformation based, and auto-encoder based) based on experience and comprehensively reviewed these methods in Section 2.
- Along the framework of adversarial auto-encoder, we propose a new method to conduct an unsupervised feature learning and image clustering task simultaneously, and adjust the network and some parameters to make the structure of adversarial auto-encoder more suitable for the image clustering task.
- We conduct a fairly comprehensive experiment covering eight different image data sets (handwritten fonts, human faces, object images, etc.) and nine different comparative algorithms (classic feature representation methods, recently proposed feature representation methods and novel deep clustering methods).
- The performance of the designed network outperforms other comparative algorithms in the image clustering issue of the tested data sets.

In the following sections, we review some related work on feature representation learning in Section 2. Then in Section 3, the network structure and training strategy of adversarial auto-encoder are presented. In Section 4, several comparative experiments are reported to prove the effectiveness of the discriminative feature learned from adversarial auto-encoder method. Ultimately, this paper is summed up in Section 5.

2 Related work

For the convenience of expression, we indicate the collection of input data samples and their labels as $\mathbb{X}$ and $\mathbb{L}$, respectively. Hence input data matrix is represented by $\mathbf{X} = \left[\mathbf{x}_1, \cdots, \mathbf{x}_n\right] \in \mathbb{R}^{d\times n}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is a d dimensional feature vector.

In this section, we present some classic feature representation methods, which can be broadly partitioned into three classes: linear transformation based feature representation, non-linear transformation based feature representation and variations of auto-encoder neural network.

2.1 Linear transformation based feature representation

Until now, linear feature representation is still a focal point in many fields such as image processing and pattern recognition. This may be owing to its intuitiveness, simplicity, and effectiveness in handling numerous real-world issues.

Generally speaking, for an input data matrix $\mathbf{X} = \left[\mathbf{x}_1, \cdots, \mathbf{x}_n\right] \in \mathbb{R}^{d\times n}$, performing a linear transformation on the original high-dimensional space is the simplest way to obtain a subspace with low dimension. It is able to be formalized as

$$\mathbf{X}' = \mathbf{W}^{\mathbf{T}}\mathbf{X} \tag{1}$$

where $\mathbf{W}^{\mathbf{T}} \in \mathbb{R}^{d'\times d}$ is the transformation matrix and d' is the dimensions of subspace $\left(d' < d\right)$. $\mathbf{X}' = \left[\mathbf{x}'_{\mathbf{1}}, \cdots, \mathbf{x}'_{\mathbf{n}}\right] \in \mathbb{R}^{d'\times n}$ is the representation in low-dimensional subspace, and $\mathbf{x}'_i = \mathbf{W}^{\mathbf{T}}\mathbf{x}_i$ denotes a d'-dimensional feature vector. Evidently, the features in the new low-dimensional subspace are linear combinations of features in the original high-dimensional space.

Among the feature representation methods based on linear transformation, PCA and LPP are two mainstream methods. Though linear discriminant analysis (LDA) [30] is another mainstream approach, it is a supervised one. In this paper, We mainly focus on unsupervised feature representation, hence LDA will not be discussed.

2.1.1 Principal component analysis

The central idea of PCA [7, 32, 36] is to utilize a linear transformation to project the input data in the directions of maximum variances, which aims to maintain the features of input data when reducing to a subspace with a low dimension, and minimize the reconstruction loss. For a collection of input data matrix $\mathbf{X} = \left[\mathbf{x}_1, \cdots, \mathbf{x}_n\right]$, denote a transformation matrix as $\mathbf{W}$ and $\mathbf{y_i} = \mathbf{W}^T\mathbf{x}_i$, the objective function of PCA can be represented as follows

$$\underset{\|\mathbf{W}\|=1}{\mathbf{argmax}} \sum_{i=1}^{n} (\mathbf{y}_i - \bar{\mathbf{y}})^2 \tag{2}$$

where $\bar{\mathbf{y}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{y}_i$ indicates the mean of $\{\mathbf{y}_i\}_{i=1}^{n}$, then (2) can be represented in another form

$$\underset{\|\mathbf{W}\|=1}{\mathbf{argmax}} \mathbf{W}^T\mathbf{C}\mathbf{W} \tag{3}$$

where $\mathbf{C}$ indicates the covariance matrix about input samples, and the eigenvectors of $\mathbf{C}$ relevant to the largest eigenvalues cross the most favorable subspace which generated by PCA.

2.1.2 Locality preserving projections

Different from PCA, LPP mainly puts emphasis on retaining the local neighborhood architecture between input data points, while PCA mainly revolves around retaining the global architecture of input data.

LPP contains the advantages of manifold learning and linear feature representation [36, 46], which is based on a linear approximation of Laplacian Eigenmaps (LE) method [4]. In simple terms, it is a linear version of LE that utilizes a linear approximation method for non-linear feature representation.

Assume input data matrix $\mathbf{X} = \left[\mathbf{x}_1, \cdots, \mathbf{x}_n\right]$ and a linear transformation matrix $\mathbf{W}$ which projects $\mathbf{X}$ into a subspace $\mathbf{Y} = \left[\mathbf{y}_1, \cdots, \mathbf{y}_n\right]$ with low dimension, the objective function can be denoted as follows

$$\min_{\mathbf{W}} \sum_{i=1}^{n} \sum_{j=1}^{n} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|^2 \mathbf{P}(i, j) \tag{4}$$

where $\mathbf{y}_i = \mathbf{W}^T \mathbf{x}_i$, and $\mathbf{P}(i, j)$ which created by the nearest-neighbor graph is a similarity matrix. When $\mathbf{x}_i$ is among kNN of $\mathbf{x}_j$ or $\mathbf{x}_i$, $\mathbf{P}(i, j)$ can be formalized as

$$\mathbf{P}(i, j) = \mathbf{e}^{-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{t}} \tag{5}$$

2.2 Non-linear transformation based feature representation

In the real world, not all data is linear structure. Data structures such as trees and graphs, are very common, but they are non-linear structures and therefore difficult to be handled by linear methods. Hence, non-linear methods are also widely studied. Isometric feature mapping (Isomap) and neighborhood preserving embedding (NPE) are two typical unsupervised feature representation methods based on non-linear transformation.

2.2.1 Isometric feature mapping

Isomap [1, 38] is based on the method that retains geodesic distances among all pairs of the data points as much as possible by defining a low-dimensional embedding.

Isomap creates a neighborhood graph G to calculate the geodesic distances among the input data. Then a shortest path among two points in G is able to be calculated through applying Dijkstra or Floyd algorithm [11, 24]. This path matrix constitutes an evaluation about the geodesic distance between two input data points. Finally, the geodesic distances among all data points are calculated, thus it could create a pairwise geodesic distance matrix.

But isomap method still has some disadvantages, such as its topological instability. In a neighborhood graph G, it may establish wrong connections between data points. Although with some shortcomings, isomap is also widely applied in many real-world problems successfully.

2.2.2 Neighborhood preserving embedding

The purpose of NPE [19, 36] is to seek out an approach to implement local neighborhood structure retention [48] for input data, which is similar to locally linear embedding (LLE) [37]. NPE makes use of a local least squares approximation method to assess the affinity weight matrix $\mathbf{W}$. For an input data matrix $\mathbf{X} = \left[\mathbf{x}_1, \cdots, \mathbf{x}_n\right]$, the local geometry of these data points can be characterized through the reconstruction from their neighbors. Thus the reconstruction error is able to be formalized by the cost function as follows

$$\phi(\mathbf{W}) = \sum_{i=1}^{n} \left\|\mathbf{x}_i - \sum_{j=1}^{n} \mathbf{W}_{ij} \mathbf{x}_j\right\|^2 \tag{6}$$

where $\mathbf{x}_j$ is one of the neighbors of $\mathbf{x}_i$, $\mathbf{W}_{ij} \in \mathbf{W}$ represents the verge weight from $\mathbf{x}_i$ to $\mathbf{x}_j$, and a constraint of $\mathbf{W}_{ij}$ is defined as

$$\sum_{j=1}^{n} \mathbf{W}_{ij} = 1 \tag{7}$$

2.3 Feature representation based on auto-encoder neural network

Auto-encoder (AE) neural network is another method for feature representation. In the past decade, variations of auto-encoder neural network have received many achievements. In this subsection, an unsupervised sparse auto-encoder and a semi-supervised label consistent auto-encoder are introduced.

2.3.1 Sparse auto-encoder

As a variant of auto-encoder, sparse auto-encoder (SAE) assumes that its network is a sparse network. The difference between SAE and auto-encoder is the definition of loss function. SAE has a constraint on the output of the hidden layer, it adds sparsity constraints to the hidden neurons, so as to induce the mean value of hidden layer output close to zero as much as possible. It indicates that most of the hidden layer neurons are in a non-activate state. Therefore, the optimization function of SAE is defined as

$$\min_{\mathbf{W}, \mathbf{b}} J(\mathbf{W}, \mathbf{b}) + \beta \sum_{j=1}^{d'} \mathrm{KL}(\rho || \hat{\rho}_j) \tag{8}$$

where d' is the dimensions of hidden layer, $J(\mathbf{W}, \mathbf{b})$ denotes the optimization function of the typical AE network, ρ

is a sparsity parameter whose value is generally tiny, and $\mathrm{KL}(\rho||\hat{\rho_j}) = \rho\log(\frac{\rho}{\hat{\rho_j}}) + (1-\rho)\log(\frac{1-\rho}{1-\hat{\rho_j}})$ is the KL-divergence that induces the output of hidden layer close to the sparsity parameter ρ which is predefined. Besides, $\hat{\rho_j}$ can be formulated as

$$\hat{\rho_j} = \frac{1}{n}\sum_{i=1}^{n}\sigma(\mathbf{w}_j^T\mathbf{X} + \mathbf{b}_j^{(1)})\mathbf{x}_i \tag{9}$$

2.3.2 Label consistent auto-encoder

Label consistent auto-encoder (LCAE) which proposed by Gogna et al. [14], is an approach for learning feature representation in semi-supervised. The structure of LCAE is a two-layer stacked auto-encoder network, and it is based on the idea of how to gain an approximate inverse for a linear problem, which can be formalized as

$$\mathbf{x}' = \mathbf{A}^T\mathbf{y} = \mathbf{A}^T\mathbf{A}\mathbf{x} \tag{10}$$

where $\mathbf{x}'$ is a noisy version of $\mathbf{x}$, and $\mathbf{x}$ presents the practical solution. LCAE focuses on seeking a linear mapping from deepest layer of network to category labels, so as to form the penalty about class-label consistency. Thus the objective function of LCAE is defined as follows

$$\begin{aligned}&\min_{\mathbf{W}_1,\mathbf{W}_1{}',\mathbf{W}_2,\mathbf{W}_2{}',D}\left\|\mathbf{X} - \mathbf{W}_1{}'\phi(\mathbf{W}_2{}'\phi(\mathbf{W}_2\phi(\mathbf{W}_1\mathbf{X})))\right\|_F^2\\&+\lambda\left\|L - D\phi(\mathbf{W}_2\phi(\mathbf{W}_1\mathbf{X}_S))\right\|_F^2\end{aligned} \tag{11}$$

where L is the labels, D is the linear map. And the weight matrices of encoder and decoder in the first layer are $\mathbf{W}_1$ and $\mathbf{W}_1{}'$, while $\mathbf{W}_2$ and $\mathbf{W}_2{}'$ are the encoder and decoder matrices in the second layer, respectively. $\mathbf{X}$ presents the input data and part of it have labels while the rest are not. Therefore $\mathbf{X} = [\mathbf{X}_U|\mathbf{X}_S]$, where $\mathbf{X}_S$ is the data which have supervised information.

2.3.3 Deep embedded clustering

Deep clustering is an emerging research field in recent years. Different from the two-stage framework in a classic way, deep clustering attempts to learn the low-dimensional representation and cluster allocations simultaneously.

The deep embedded clustering method proposed by Xie et al. [45] applied an auto-encoder network and defined a clustering loss to learn the clustering allocations and low-dimensional feature simultaneously. After the pre-training process, DEC only reserve the encoder and discard the decoder, then the encoder is fine-tuned according to the defined loss function as follows

$$L = KL(P \parallel Q) = \sum_i\sum_j p_{ij}\log\frac{p_{ij}}{q_{ij}} \tag{12}$$

where KL means the KullbackLeibler divergence, Q is the soft labels distribution and P is the target distribution produced from Q, their detailed definitions can be found in [45]. In addition, the training process of DEC can be considered as a self-training due to the relation of P and Q.

2.3.4 Improved Deep Embedded Clustering

Although the deep clustering method which aims to learn the low-dimensional feature representation of inputs for the clustering issue had obtained inspiring performance, Guo et al. [17] noticed a shortcoming of the DEC method that it lacks the consideration about the retention of local structure. To this end, they proposed the improved deep embedded clustering method (IDEC) to address this issue.

The core of IDEC is to keep the decoder of auto-encoder network after the pre-training process instead of discarding it like DEC. The retained decoder can provide the reconstruction loss to guide the clustering task jointly with the clustering loss, thereby avoiding the distortion of feature space when only the clustering loss is applied. Hence the objective function IDEC can be formalized as follows

$$L = L_r + \gamma L_c \tag{13}$$

where γ is a coefficient which can control the degree of distortion in feature space. If $\gamma = 1$ and $L_r = 0$, then IDEC can be regarded as DEC method. Besides, L_r and L_c are reconstruction loss and clustering loss, respectively. The reconstruction loss L_r can be defined as

$$L_r = \left\|\mathbf{X} - \mathbf{X}'\right\|_2^2 \tag{14}$$

where $\mathbf{X}'$ is the reconstruction of the input data matrix $\mathbf{X}$. And The clustering loss L_c is the same with (12).

3 Adversarial auto-encoder

Adversarial auto-encoder (AAE) is a probability-based auto-encoder. It combines generative adversarial networks (GAN) which is popular recently with variational auto-encoder (VAE), and focuses on seeking out a discriminative feature representation for high-dimensional data.

3.1 Architecture

In order to comprehend the architecture of AAE, we must understand the structure of generative adversarial networks (GAN) and auto-encoder (AE) firstly.

A typical auto-encoder network includes two phases: an encoder and a decoder. The former focuses on learning a subspace feature representation of inputs, while the latter aims to reconstruct the inputs and minimize the reconstruction error. The network structure of auto-encoder which consists of three layers is shown in Fig. 1.

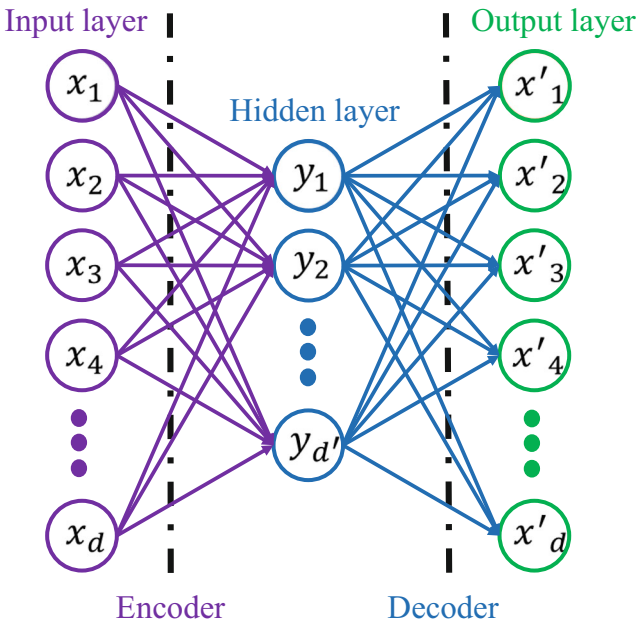

Fig. 1 The network structure of auto-encoder. A typical auto-encoder network usually comprises an input layer, a hidden layer, and an output layer

With regard to GAN, the structure of it contains a generative network G and a discriminative network D, respectively. A function $D(\mathbf{x})$ used in the discriminative network calculates the possibility that sample $\mathbf{x}$ is a positive sample that comes from the original data distribution, instead of a negative sample that is produced by generator. Meanwhile, assume a sample $\mathbf{z}$ sources from the prior distribution and a function $G(\mathbf{z})$ is utilized to map $\mathbf{z}$ to the data space, then a visualization about the structure of GAN is presented in Fig. 2.

During the training procedure, we could leverage the gradient of $D(\mathbf{x})$ to train the generator G and revise its parameters, while the aim of $G(\mathbf{z})$ in training is to fool the discriminative network D into trusting that the samples it generates are positive samples as much as possible. This process is formalized as follows

$$\min_G \max_D \mathrm{E}_{\mathbf{x}\sim p_{data}}[\log D(\mathbf{x})] + \mathrm{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))] \tag{15}$$

thus it is able to divide the training process into two phases roughly. First, the discriminator D is trained to differentiate the real samples from those fakes produced by generator G, and the second is to make the samples produced by generator G can confuse discriminator D through training.

The adversarial auto-encoder (AAE) network is a combination of generative adversarial networks and variational auto-encoder. The two training targets of AAE are: minimize reconstruction error and match the aggregated posterior distribution of the representation in hidden layer to an arbitrary prior distribution through an adversarial training. Assume an input data $\mathbf{x}$ and a hidden unit $\mathbf{z}$ of an auto-encoder, the structure of AAE is presented in Fig. 3.

The upper part of this figure is an auto-encoder network which first transforms the input data $\mathbf{x}$ into hidden vector $\mathbf{z}$, then reconstructs $\mathbf{x}$ from the hidden vector $\mathbf{z}$. The encoder of auto-encoder network can be considered as a generator. The bottom part of this figure is a discriminative network which is trained to distinguish a sample whether it is from the hidden layer of auto-encoder network (fake sample), or from the original data distribution (true sample).

3.2 Training strategy

Define the original data distribution as $p_{data}(\mathbf{x})$, the prior distribution as $p(\mathbf{z})$, and the model distribution as $p(\mathbf{x})$. Meanwhile, we also define $q(\mathbf{z}|\mathbf{x})$ and $q(\mathbf{x}|\mathbf{z})$ as a distribution of encoder and decoder, respectively. By the

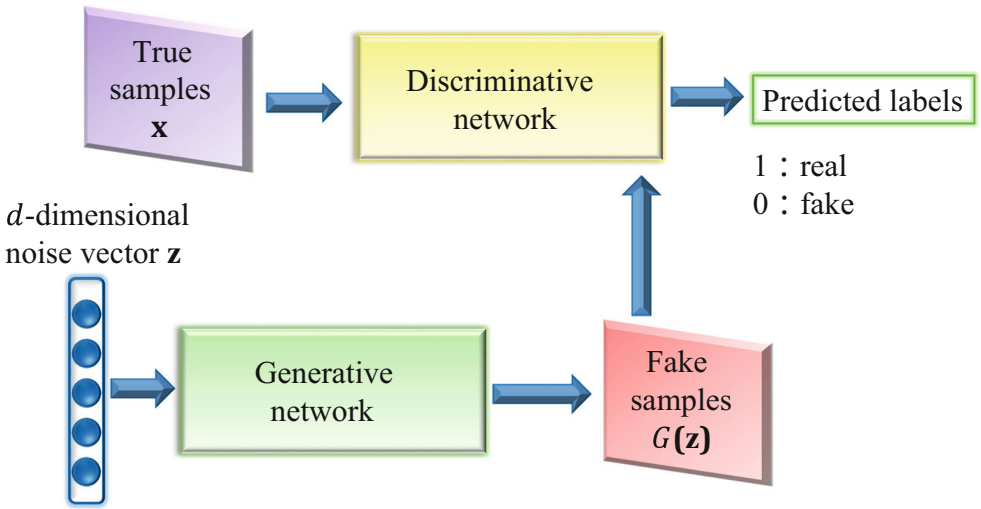

Fig. 2 The structure of generative adversarial networks. The generative network aims to generate fake samples that can deceive the discriminative network, while the discriminative network puts emphasis on enhancing its ability to distinguish between true and false samples through training

encoding function in encoder distribution $q(\mathbf{z}|\mathbf{x})$, we can define $q(\mathbf{z})$ on hidden layer through the following formula

$$q(\mathbf{z}) = \int_{\mathbf{x}} q(\mathbf{z}|\mathbf{x}) p_{data}(\mathbf{x}) d\mathbf{x} \tag{16}$$

where $q(\mathbf{z})$ in hidden layer is an aggregated posterior distribution. In the training process, one goal of AAE is to match $q(\mathbf{z})$ with $p(\mathbf{z})$ to realize regularization. Consequently, AAE adds an adversarial network in training, and considers the encoder $q(\mathbf{z}|\mathbf{x})$ as a generator in the adversarial network. From Fig. 3, it can be easily known that it is the adversarial network that leads this process. In the meantime, another goal of AAE is the minimization of reconstruction error about auto-encoder network. The encoder guarantees that the aggregated posterior distribution, $q(\mathbf{z})$, could confuse the discriminative network into trusting that $q(\mathbf{z})$ comes from $p(\mathbf{z})$.

For the selection of encoder $q(\mathbf{z}|\mathbf{x})$ in AAE, there are several possible methods: First is the deterministic method, in this method we suppose $q(\mathbf{z}|\mathbf{x})$ as a deterministic function of $\mathbf{x}$, thus the randomness of $q(\mathbf{z})$ only comes from the data distribution $p_{data}(\mathbf{x})$. The encoder can be considered the same as the encoder of a typical auto-encoder. Another is the universal approximator posterior method. In this method, the origin of randomness about $q(\mathbf{z})$ is the random noise of encoder inputs and the data-distribution $p_{data}(\mathbf{x})$. The Gaussian posterior method is also a choice of encoder. We suppose $q(\mathbf{z}|\mathbf{x})$ as a Gaussian distribution in this method, and its means and standard deviations are computed by encoder. Besides, the randomness of $q(\mathbf{z})$ sources from the stochasticity of the Gaussian distribution in the output of encoder and the data-distribution $p_{data}(\mathbf{x})$.

The main difference in the training process between AAE and VAE is that in AAE, we only require to learn a sampling from the prior distribution for inducing the process of matching $q(\mathbf{z})$ to $p(\mathbf{z})$, while VAE demands to learn the precise function of the prior distribution for the back-propagation of KL divergence. Thus AAE could impose a complex distribution without learning the precise function of the prior distribution. In addition, there are also some differences between AAE and GAN, for instance, AAE learns to catch the data distribution through the training process of auto-encoder network. However, for an output layer in network, a pixel level data distribution is imposed by GAN on it.

On the whole, each mini-batch training process of AAE, including the auto-encoder and the adversarial network which are trained jointly with SGD, can be separated into a reconstruction stage and a regularization stage. In the reconstruction stage, the auto-encoder aims at the minimization of the reconstruction error for inputs, by updating the parameters of encoder and decoder network, ϕ and θ. Define objective function as follows

$$\underset{\theta,\phi}{\operatorname{argmin}} \left\| \mathbf{x} - \mathbf{x}' \right\|^2 \tag{17}$$

where $\mathbf{x}'$ is the reconstructed data. While in the regularization stage, there have two aims in it. First, the adversarial

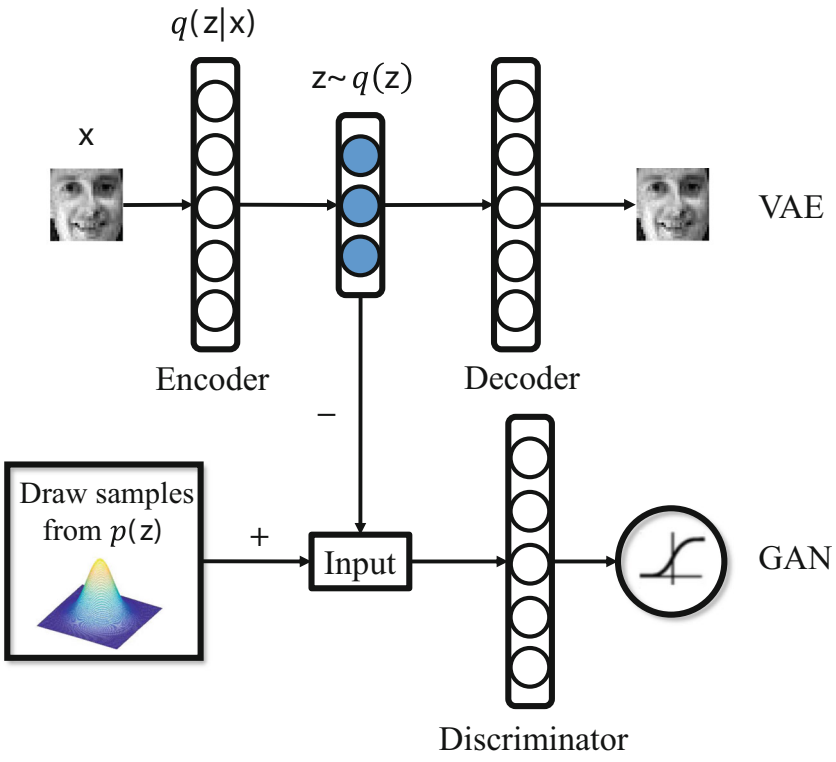

Fig. 3 A visualization of the architecture of adversarial auto-encoder. The upper half is an auto-encoder, its encoder can be considered as a generator, while the bottom half is a discriminative network

network differentiates the true samples that were produced by the prior distribution, from those fake samples generated in the hidden layer through updating its discriminator. Then the generator in adversarial network is updated to fool the discriminative network, and its objective function is similar to (15). The discriminator $D(\mathbf{x})$ calculates the possibility that sample $\mathbf{x}$ is a true sample which we attempt to model, instead of a fake sample. While $\mathbf{z}$ is mapped by $G(\mathbf{z})$ from the prior distribution into the data space. The goal of $G(\mathbf{z})$ is to maximally fool the discriminative network through training, so as to force the discriminator to trust the samples generated by it are true samples, and this training process of G is realized by taking the advantage of the gradient of $D(\mathbf{x})$ with regard to $\mathbf{x}$. G also uses it to update its parameters.

The training procedure of adversarial auto-encoder is presented at Algorithm 1.

Algorithm 1 Algorithm of adversarial auto-encoder.

Input: The learning rate α, the input data $\mathbf{X} = \left[\mathbf{x}_1, \cdots, \mathbf{x}_n\right] \in \mathbb{R}^{d\times n}$ and the number of hidden units d'
Output: The low-dimensional discriminative feature representation of hidden layer, $\mathbf{Z} \in \mathbb{R}^{d'\times n}$.

1: Randomly initialize the parameters of auto-encoder network: ϕ and θ;
2: For input data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$, randomly generate mini-batches $\{\mathbf{X}_i \in \mathbb{R}^{d\times n_i}\}_{i=1}^{k}$, $(n = \sum_{j=1}^{k} n_j)$, to form a partition of $\mathbf{X}$;
3: **repeat**
4: **for** each epoch $(i = 1, 2\cdots, k)$ **do**
5: The parameters of encoder and decoder network, ϕ and θ, are updated by the reconstruction loss function $\underset{\theta,\phi}{\operatorname{argmin}} \left\|\mathbf{x} - \mathbf{x}'\right\|^2$;
6: Generate the fake samples $\mathbf{z}_i$ from the encoder layer, and update the discriminative network to differentiate between real and fake samples via the following formula $\nabla_{\theta_d} \frac{1}{k} \sum_{i=1}^{k} [\log D(\mathbf{x}_i) + \log(1 - D(G(\mathbf{z}_i)))]$;
7: The generative network is updated by descending its stochastic gradient according to the following formula $\nabla_{\theta_g} \frac{1}{k} \sum_{i=1}^{k} \log(1 - D(G(\mathbf{z}_i)))$;
8: **end for**
9: **until** convergence
10: **return** The low-dimensional discriminative feature representation $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n] \in \mathbb{R}^{d'\times n}$.

4 Experimental analysis

In this section, we first present the eight tested image data sets. Then illustrate the parameter settings for each feature representation algorithm and evaluation metrics used in experiment. After that, a number of comparative experiments are performed to provide a comparison for the low-dimensional discriminative feature learned via adversarial auto-encoder with other feature representation methods. Extensive experiments prove the efficiency of adversarial auto-encoder.

4.1 Data sets

USPS is a popular subset of the USPS handwritten digit database [35], it includes 9,298 handwritten images for a total of 10 categories,[1] and splits them into 7,291 images for training and 2,007 images for testing. Each of them is denoted as a feature vector with 256 dimensions.

Chars74K database consists of two different versions (English and Kannada) [8]. The English version contains 62 different categories of images.[2] It could be divided into three parts: the first part is 7,705 characters acquired from natural scenes, the second part is 3,410 hand-drawn characters generated by tablet PC, and the third part is 62,992 synthesized character generated by computer fonts. In this paper, we select the English version in our experiments. Apart from this, we barely employ the third part of this database and eliminate all numerals data. Thus the database applied in experiment consists of 44,044 training samples and 8,788 testing samples with 52 categories where each image could be stacked as a 1,024-dimensional feature vector.

Yale-B [6, 12] is a subversion of The Extended Yale Face Database B.[3] For this data set, we simply utilize the cropped images and unify the size to 32×32 pixels for all images. Each of them is denoted as a feature vector with 1,024 dimensions. This data set is now composed of 2,414 gray-scale face images with 38 categories, and under 64 different illuminations conditions per class.

ORL [5, 16] face database is a collection of 10 different types of face images and each type of images contains 40 distinctive subjects.[4] These images were obtained at varied times on some subjects, differing the facial details and expressions and the lighting. Every image is reshaped as a 1,024-dimensional feature vector.

COIL-20 data set is comprised of 20 objects in total [33], and each object has 72 images.[5] These images were generated by rotating the object on a turntable and producing at intervals of 5 degrees. All images are gray-scale with the size 32×32, and stacking as a 1,024-dimensional feature vector.

CIFAR-10 database is composed of 60,000 colour images, including 10 categories such as airplane, dog and

[1] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/multiclass.html
[2] http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/
[3] http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html
[4] http://www.uk.research.att.com/facedatabase.html
[5] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php

truck, etc. It is a subset gained from the 80 million tiny images data set [27] and each category contains 6,000 images.[6] We split the data into 50,000 images for training and 10,000 images for testing, and each of them is reshaped as a feature vector with 3,072 dimensions.

Fashion-MNIST is a data set which contains 60,000 samples for training and 10,000 samples for testing, including 10 categories such as Trouser, Dress and Sneaker, etc.[7] [44]. Each example is a 28×28 gray-scale image, with 256 grey levels per pixel, and it could be denoted as a 784-dimensional feature vector.

SMSHP (Sebastien Marcel Static Hand Posture) data set [29] consists of 6 different hand posture styles (v, a, b, c, point, five).[8] The categories in this database are generated by the 6 hand postures. For convenience of experiment, the size of all images is unified as 32×32 pixels, and each of them could be reshaped as a feature vector with 1,024 dimensions.

The primary information of each data set aforementioned, including the amount of samples, features and classes, are summed up in Table 1. In addition, we arbitrarily select 30 samples from every data set to provide a visualization in Fig. 4.

Table 1 A concise illustration about the tested data sets applied in experiments

ID	data sets	# samples	# features	# classes
1	USPS	9,298	256	10
2	Chars74K	52,832	1,024	52
3	Yale-B	2,414	1,024	38
4	ORL	400	1,024	40
5	COIL-20	1,440	1,024	20
6	CIFAR-10	60,000	3,072	10
7	Fashion-MNIST	70,000	784	10
8	SMSHP	5,531	1,024	6

4.2 Parameter settings

In order to prove the effectiveness of adversarial auto-encoder (AAE) for unsupervised feature representation, several unsupervised feature representation methods are introduced to provide a comparison, including principal component analysis (PCA), Isometric feature mapping (Isomap), locality preserving projection (LPP), neighborhood preserving embedding (NPE) auto-encoder (AE) and sparse auto-encoder (SAE). Apart from this, we also take a semi-supervised label consistent auto-encoder (LCAE) method and two recently proposed deep clustering methods (deep embedded clustering (DEC) and improved deep embedded clustering (IDEC)) into comparison, to show the effectiveness of AAE.

Regarding the parameter settings, we illustrate the architecture and training strategy about AAE in Section 3. In the following experiment, a three layers AAE is employed, the hidden layer units are set to $\{20, 40, 60, \cdots, 200\}$, and the number of training epochs is fixed as 1000. Apart from this, the learning rate α is set as $\alpha = 0.001$. With regard to PCA, the only parameter of it is the subspace dimensions. For Isomap, LPP and NPE, they search their neighbors by utilizing the kNN method, the amount of their nearest k neighbors is fixed to 20, and if a singular matrix with the size of $k \times k$ appears, k will be reset to a larger one to solve this problem. For AE, SAE and LCAE, the parameters of them including learning rate α and training epochs, are identical to the settings of AAE. Besides, the range of subspace dimensions about all compared feature representation methods is all set to $\{20, 40, 60, \cdots, 200\}$, unanimous to AAE. For the two deep clustering methods (DEC and IDEC), the performance of DEC and IDEC in USPS data set are obtained from the original paper [17], and for other data sets, we train them with the same parameter settings. The network structure of DEC and IDEC is fixed as d-500-500-2000-10 dimensions and the learning rate is set as $\alpha = 0.001$. Also, the pre-training iterations are set to 200 and the coefficient γ is set as $\gamma = 0.1$ after pre-training.

After all feature representation methods are performed, the low-dimensional features learned by them are assessed by K-means clustering method. The process of dimensionality reduction and K-means clustering for AAE is summarized in Fig. 5. The main trouble in assessing the effectiveness of unsupervised and semi-supervised approaches is the missing of supervised information. In the absence of complete supervised information, it is difficult to seek for the corresponding relation between predictive labels and ground truths after clustering. With a few searches and selections, we finally select three evaluation metrics, which are popular recently, including clustering accuracy (ACC), normalized mutual information (NMI) and adjusted rand index (ARI), to evaluate the performance of K-means method. In addition, for different initial values, the performance of K-means fluctuates greatly, it mainly due to its high sensitivity. Thus we repeat it 10 times then record their means and standard deviations. The higher scores of ACC, ARI and NMI in K-means method indicate the better performance.

4.3 Evaluation metrics

After learning the subspace feature with low dimension by those feature representation methods, we assess their quality

[6] http://www.cs.toronto.edu/~kriz/cifar.html

[7] https://github.com/zalandoresearch/fashion-mnist

[8] http://www.idiap.ch/resource/gestures/

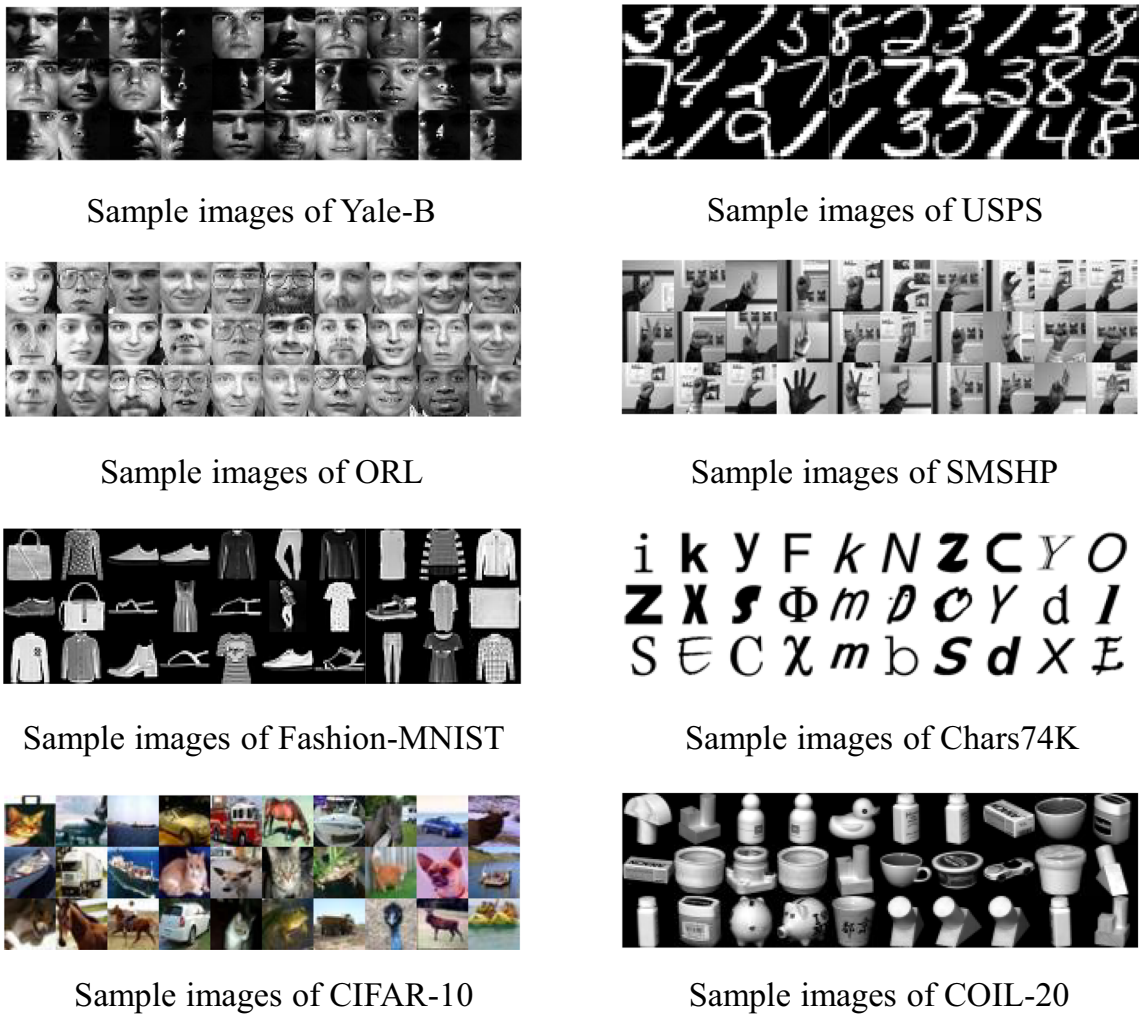

Fig. 4 A visualization of data sets utilized in experiment, we arbitrarily select 30 images from each database for display

through K-means clustering. With regard to AAE, AE, SAE and LCAE, we obtain the subspace representation of them in the hidden layer, as presented in Fig. 5.

As mentioned above, it is difficult to seek out a corresponding relation between the labels predicted by K-means method and the ground truths without the guidance of supervised information. Thus in performance evaluation, we suppose the information of true class labels for data sets is available, though at the unsupervised situation. In the following, the definition of ACC, ARI and NMI is introduced.

4.3.1 Clustering accuracy

Define $\{\mathbf{x}_i\}_{i=1}^n$ as the data points, $\{\mathbf{y}_i\}_{i=1}^n$, $\{\hat{\mathbf{y}}_i\}_{i=1}^n$ as the labels of ground truths and the predictive labels of K-means clustering, respectively. From this, ACC can be formalized as follow

$$ACC = \frac{\sum_{i=1}^n \delta(\mathbf{y}_i, map(\hat{\mathbf{y}}_i))}{n} \tag{18}$$

where $map(\cdot)$ is a permutation mapping which could best match the labels of K-means clustering with the labels of ground truths. Besides, $\delta(\mathbf{y}_i, map(\hat{\mathbf{y}}_i)) = 1$ when $\mathbf{y}_i = map(\hat{\mathbf{y}}_i)$, as for other cases, $\delta(\mathbf{y}_i, map(\hat{\mathbf{y}}_i)) = 0$.

4.3.2 Adjusted rand index

Assume $\mathbb{T} = \{\mathbf{T}_j\}_{j=1}^t$ stands for the ground truths, $\mathbb{C} = \{\mathbf{C}_i\}_{i=1}^c$ stands for the clustering result. ARI computes the resemblance between two clustering samples, the contingency table regards to $\mathbb{C}$ and $\mathbb{T}$ is represented as follows

$\mathbb{C}\backslash\mathbb{T}$	$\mathbf{T}_1$	$\cdots$	$\mathbf{T}_t$	sums
$\mathbf{C}_1$	n_{11}	$\cdots$	n_{1t}	a_1
$\vdots$	$\vdots$		$\vdots$	$\vdots$
$\mathbf{C}_c$	n_{c1}	$\cdots$	n_{ct}	a_c
sums	b_1	$\cdots$	b_t	

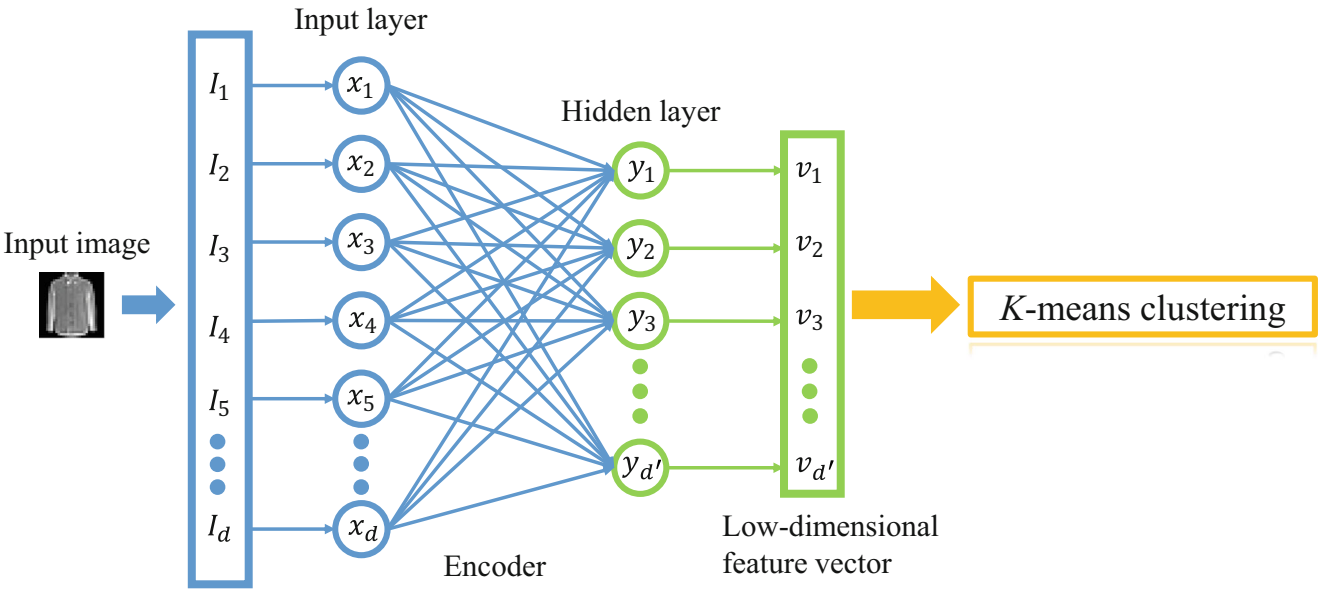

Fig. 5 The process of dimensionality reduction and K-means clustering on AAE. Through encoding, a d-dimensional original data is denoted as a feature vector with d' dimensions in the hidden layer, then adopt it as the input of K-means method. Finally, the clustering performance is assessed by three popular evaluation metrics including ACC, ARI and NMI

where $n_{ij} = \left|\mathbf{C}_i \cap \mathbf{T}_j\right|$, hence, $ARI(\mathbb{C}, \mathbb{T})$ can be defined according to the contingency table.

$$\frac{\sum_{ij}\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}} \tag{19}$$

4.3.3 Normalized mutual information

With regard to NMI, define d_i as the amount of flows in category i, c_j as the amount of flows in category j, and $d_{i,j}$ as the amount of flows both in category i and category j, then NMI is able to be calculated as follows

$$NMI = \frac{\sum d_{i,j}\log(\frac{|\Omega|.d_{i,j}}{d_i c_j})}{\sqrt{(\sum_i d_i \log(\frac{d_i}{d}))(\sum_j c_j \log(\frac{c_j}{d}))}} \tag{20}$$

where NMI is based on the metric mutual information(MI), the value of denominator indicates the information entropy, and the value of numerator is MI. The value of NMI is range from 0 to 1.

Due to the high sensitivity of K-means clustering, it will be performed 10 times repeatedly to receive their means and standard deviations as the final evaluation result of our experiments. The higher scores of ACC, ARI and NMI denote the better performance for K-means clustering.

4.4 Experimental visualization

The main target of unsupervised feature representation methods is to seek for a subspace representation which could well represent the feature of original data, without the guidance of supervised information. In this subsection, we provide a concise visual experimental result for each feature representation algorithm used in experiments.

Taking the data set COIL-20 as an instance, part of aforementioned feature representation algorithms are selected to be performed to reduce the dimensions of it into 2. Then executing K-means clustering for each low-dimensional feature learned by them, and visualize the result as Fig. 6.

From this figure, we can roughly offer an intuitive comparison about the performance of image clustering for adversarial auto-encoder and other feature representation methods. The goal of clustering is to join the points in a same category close to each other as much as possible, while separating the points in different categories. The clustering result of adversarial auto-encoder outperforms the other compared algorithms clearly, it can join the points which have the same label more closely, while separating the points in different categories relatively better. The favorable performance of adversarial auto-encoder may be due to the added adversarial network during its training process, it takes encoder as its generator, and induces encoder to generate a sample to confuse the discriminative network into trusting this sample is a true sample, the discriminative learning process motivates encoder to generate a better

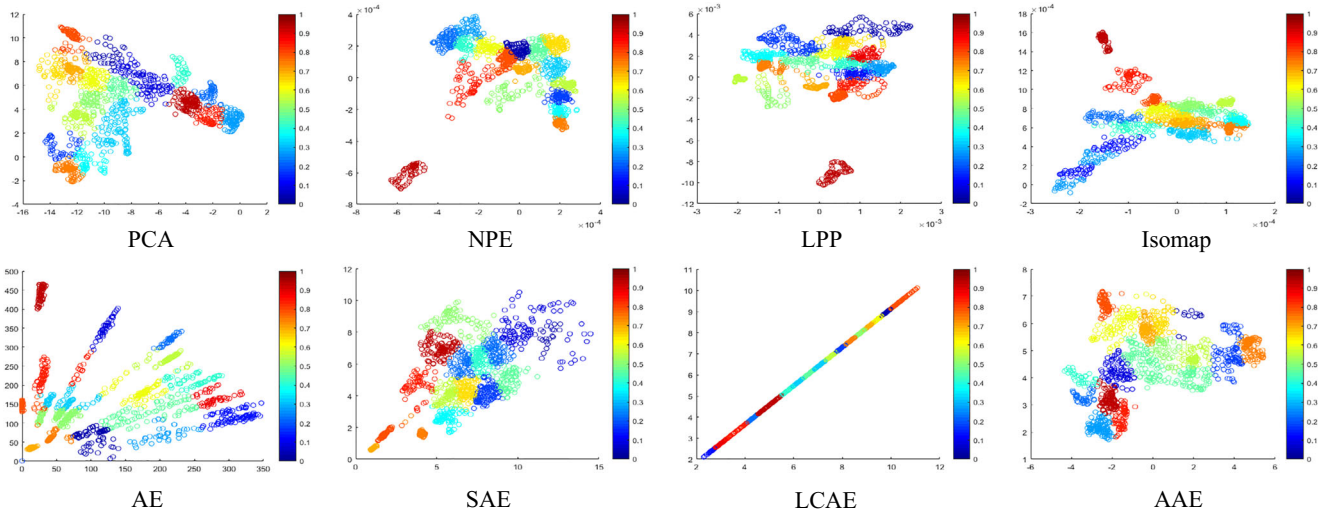

Fig. 6 A concise visualization of feature representation on COIL-20 data set, part of aforementioned algorithms are selected to performed to reduce the dimensions of original input data into 2, then we execute K-means clustering for each low-dimensional feature representation learned, and visualize the result in this figure

sample in every iteration of training. Hence, the low-dimensional feature representation generated by encoder can still represent the original data well.

Other than this, for the discriminative feature learned by AAE, we also provide a brief visualization in Fig. 7, the reconstructed images come from the discriminative feature obtained in the hidden layer. We can notice that the result of reconstructed images is not encouraging after 10 iterations from this figure. When the number of iterations exceeds 100, we can obtain a relatively ideal result of reconstructed images, which demonstrates the efficiency of the discriminative feature learned from AAE and its ability to accelerate the convergence process of the network.

4.5 Experimental result

In this subsection, comprehensive experiments are carried out. We choose the K-means method to compare the image clustering performance between AAE and other feature representation methods including PCA, LPP, Isomap, NPE, AE, SAE, and LCAE. In particular, for demonstrating the high efficiency of the learned discriminative feature representation, we also compared with DEC and IDEC, which are two recently proposed deep clustering methods. Then, three popular evaluation metrics consisting of ACC, ARI, and NMI are selected to assess their performance. Besides, the raw data of each data set is

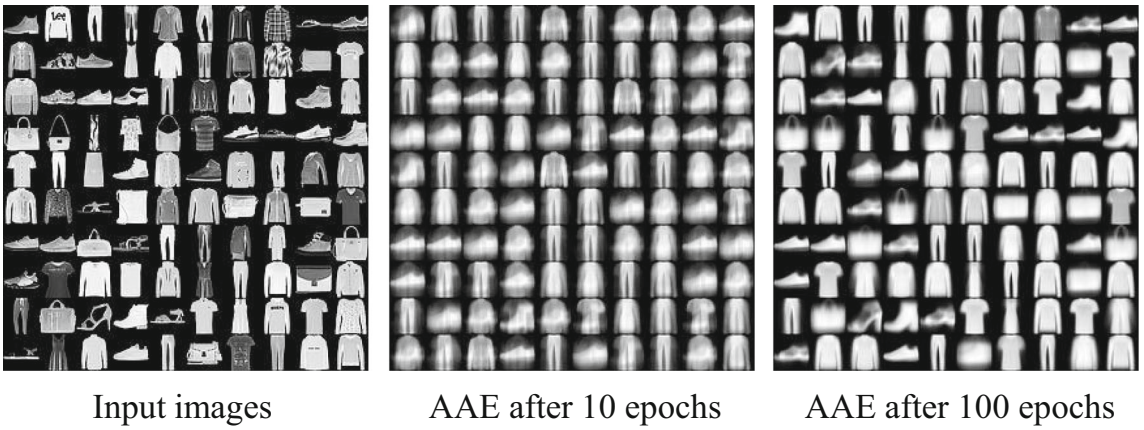

Fig. 7 A brief visualization for the discriminative feature learned by AAE. We first reduce the dimensions of Fashion-MNIST data sets into 200 to learn a discriminative feature representation, then reconstruct it to make this visualization. So as to prove the fast convergence of AAE, we provide a intuitive comparison from the reconstructed images after 10 and 100 iterations

Table 2 Clustering accuracy obtained by applying different feature representation approaches. The results (mean% + std%) shown in **red** and **blue** mean the best two performance. The symbol "—" denotes that the scores and source codes of these algorithms in corresponding data set were not available

Dataset/method	USPS	Chars74K	Yale-B	ORL	COIL-20	CIFAR-10	Fashion-MNIST	SMSHP
Baseline	67.3±2.2	34.1±0.7	10.5±0.5	63.1±2.6	57.8±3.6	22.6±0.4	56.4±1.1	37.0±1.0
PCA	70.8±1.2	35.4±0.7	11.0±0.3	65.3±1.6	64.6±1.7	23.4±1.5	58.1±0.9	**38.8±0.6**
NPE	71.1±1.2	**38.2±1.1**	32.9±1.4	**73.1±1.2**	60.2±1.2	22.5±0.5	54.3±1.8	38.6±1.9
LPP	68.4±1.9	37.6±1.4	**33.8±1.7**	68.3±1.5	**68.8±0.3**	23.3±0.4	59.4±3.6	38.3±0.5
Isomap	69.7±1.4	5.4±0.1	32.2±1.1	57.6±1.4	67.3±2.4	22.9±0.5	57.1±2.8	33.9±0.3
AE	69.1±3.1	9.5±0.3	9.0±0.2	40.8±4.1	63.2±2.4	21.3±1.6	57.6±2.7	34.3±0.8
SAE	74.1±0.7	8.5±0.2	11.3±0.2	64.0±3.1	68.5±0.8	23.2±0.7	**60.8±1.6**	35.1±0.4
LCAE	71.2±1.7	34.8±2.1	9.6±0.3	56.1±0.6	65.3±1.2	**23.5±1.2**	58.1±1.8	36.4±0.5
DEC	74.1±0.0	37.0±1.1	17.5±1.6	—	64.9±2.1	22.2±1.0	58.6±0.2	32.2±0.5
IDEC	**76.1±0.0**	37.8±0.2	17.9±0.8	—	67.5±1.8	23.0±1.5	58.9±0.2	33.1±1.0
AAE	**74.5±2.7**	**38.4±0.8**	**34.7±0.4**	**71.1±1.2**	**76.3±1.5**	**24.9±0.4**	**61.1±1.3**	**39.2±0.8**

performed image clustering directly to obtain the baseline in our experiments.

The K-means performance of all compared feature representation approaches in these eight image data sets are displayed in Tables 2, 3 and 4. The performance of each method can be appraised through comparing the scores of three metrics between the learned low-dimensional feature and the baseline. From the three tables, it is intuitive to notice that the feature representation methods are mostly superior to the traditional k-means clustering in all tested data sets, and in most image data sets, AAE is better than other compared feature representation methods in clustering performance. In Table 2, AAE ranks first at clustering accuracy on seven image data sets and second on two image data sets. Especially in the COIL-20 database, the discriminative feature learned by AAE reaches a favorable performance. The clustering accuracy of AAE in this data set is 76.3%, and the baseline is 57.8%, with an 18.5% advancement. While the second-best performance is the LPP method, its clustering accuracy is 68.8%, AAE exceeds it more than 7%.

Moreover, AAE still obtains encouraging results in the scores of ARI and NMI. In Tables 3 and 4, the scores of AAE are also very competitive. Furthermore, in comparison with the semi-supervised LCAE method and deep clustering methods DEC and IDEC, AAE also achieves a better performance. Nevertheless, other methods also perform well on some data sets. For instance, NPE ranks first on three metrics of the data set ORL. LPP and Isomap show their favorable performance in the data set Yale-B. DEC and IDEC are also very competitive on USPS and Chars74K.

Table 3 Adjusted rand index obtained by applying different feature representation approaches. The results (mean% + std%) shown in **red** and **blue** mean the best two performance. The symbol "—" denotes that the scores and source codes of these algorithms in corresponding data set were not available

Dataset/method	USPS	Chars74K	Yale-B	ORL	COIL-20	CIFAR-10	Fashion-MNIST	SMSHP
Baseline	57.4±1.8	20.4±0.7	1.7±0.2	46.5±3.7	52.9±4.1	4.7±0.2	39.5±0.8	3.9±0.8
PCA	60.7±1.4	22.8±0.9	1.9±0.5	48.3±1.0	58.2±2.6	5.1±0.9	40.7±0.5	5.2±0.6
NPE	58.3±0.5	24.2±1.2	**13.1±1.3**	**54.8±3.4**	52.9±1.2	5.1±0.1	32.9±1.9	5.3±1.9
LPP	62.4±1.0	22.9±0.5	12.6±0.5	47.2±1.6	**62.7±4.3**	4.9±0.2	**45.8±3.5**	5.6±0.4
Isomap	62.2±3.6	1.2±0.4	**13.2±0.8**	36.6±0.9	60.8±0.5	4.8±0.5	41.7±0.8	0.7±0.2
AE	58.7±3.1	2.4±0.1	1.4±0.2	23.6±2.7	57.6±1.8	4.3±0.9	40.1±1.8	2.4±0.1
SAE	**64.8±1.1**	2.0±0.4	2.3±0.5	47.9±1.6	60.4±1.4	4.8±0.4	44.2±1.2	2.7±0.2
LCAE	60.9±2.5	20.8±0.7	1.7±0.4	38.5±0.5	56.5±2.3	5.2±0.2	43.5±2.1	4.4±0.5
DEC	—	**24.6±1.1**	7.6±0.8	—	58.7±1.0	5.2±0.4	44.3±0.4	5.7±0.5
IDEC	—	**25.6±0.3**	8.9±0.2	—	61.2±2.6	**5.4±0.4**	44.9±0.2	**5.9±0.3**
AAE	**65.9±2.0**	23.8±0.3	11.6±0.3	**54.2±1.3**	**63.8±0.9**	**5.9±0.4**	**46.0±0.4**	**5.9±0.9**

Table 4 Normalized mutual information obtained by applying different feature representation approaches. The results (mean% + std%) shown in **red** and **blue** mean the best two performance. The symbol "—" denotes that the scores and source codes of these algorithms in corresponding data set were not available

Dataset/method	USPS	Chars74K	Yale-B	ORL	COIL-20	CIFAR-10	Fashion-MNIST	SMSHP
Baseline	62.4±1.3	48.7±0.4	10.6±0.7	78.1±1.7	74.1±1.7	8.1±0.3	52.2±0.4	6.9±0.9
PCA	63.0±0.5	50.2±1.2	11.1±0.2	78.6±0.7	77.6±1.1	8.5±0.2	54.3±1.1	8.9±1.1
NPE	64.1±0.6	54.6±0.9	**42.2±2.6**	**85.3±1.5**	74.7±0.8	8.5±0.3	50.5±1.7	**12.1±1.6**
LPP	68.7±0.9	54.9±0.9	**42.8±1.3**	79.0±0.4	78.1±0.6	9.3±0.2	**58.4±2.1**	9.1±2.0
Isomap	69.5±1.7	13.2±0.2	39.3±1.0	72.4±1.2	76.2±1.5	9.4±0.6	55.9±0.9	**9.3±0.6**
AE	65.1±1.9	11.3±0.2	9.6±0.3	55.2±2.2	76.4±3.1	6.9±0.6	55.3±1.8	5.9±0.8
SAE	69.6±1.5	9.1±1.1	13.2±1.0	78.5±1.6	**79.8±1.7**	8.7±1.1	58.1±0.5	3.2±0.3
LCAE	65.7±1.8	49.6±2.3	11.4±0.6	72.5±0.5	75.5±0.9	9.2±0.9	55.6±2.3	5.7±0.3
DEC	**75.2±0.0**	55.8±1.0	29.6±2.0	—	76.6±0.8	**9.8±0.7**	57.5±0.3	8.6±0.9
IDEC	**78.4±0.0**	**56.2±0.2**	30.8±0.5	—	77.4±2.0	9.7±1.2	57.9±0.7	8.5±0.7
AAE	70.3±1.5	**56.3±0.5**	40.9±0.8	**82.9±2.6**	**83.2±0.4**	**10.1±0.9**	**58.7±0.6**	8.1±0.3

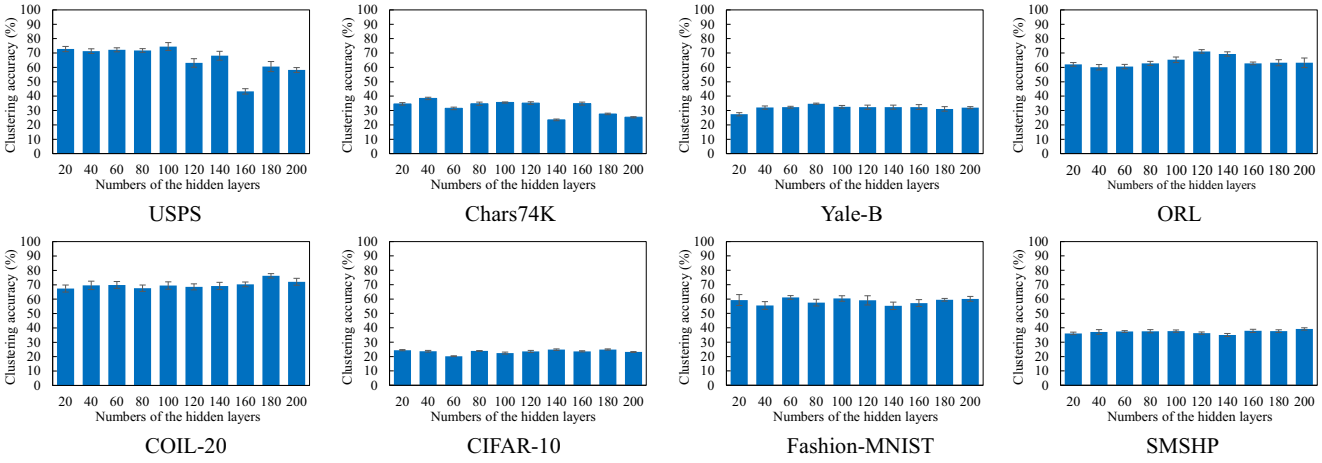

Fig. 8 The influence about different numbers of hidden layers on the clustering accuracy scores of AAE, the numbers of hidden layers is set as {20, 40, 60, ⋯ , 200}

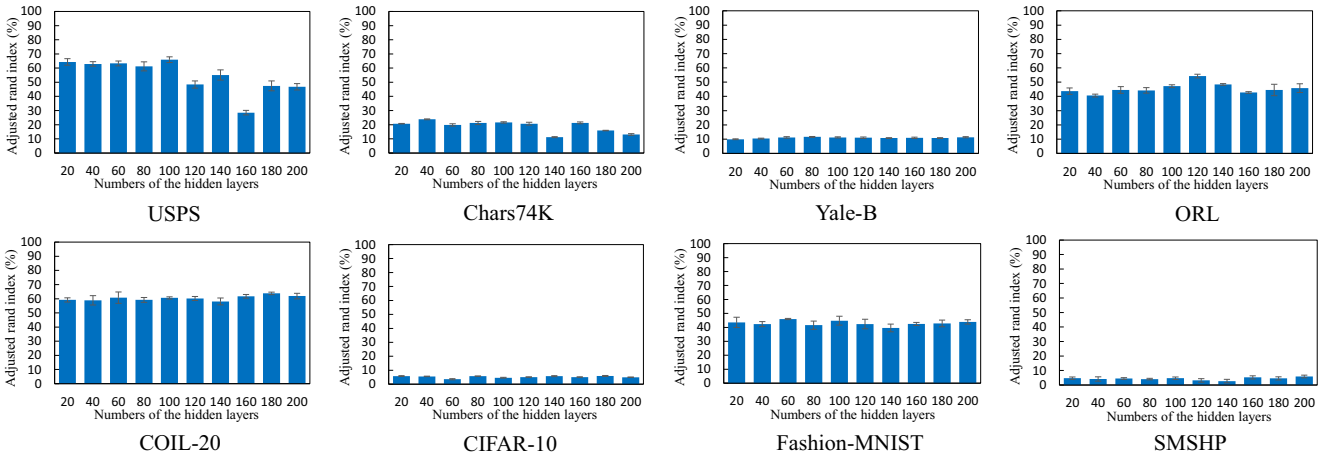

Fig. 9 The influence about different numbers of hidden layers on the adjusted rand index scores of AAE, the numbers of hidden layers is set as {20, 40, 60, ⋯ , 200}

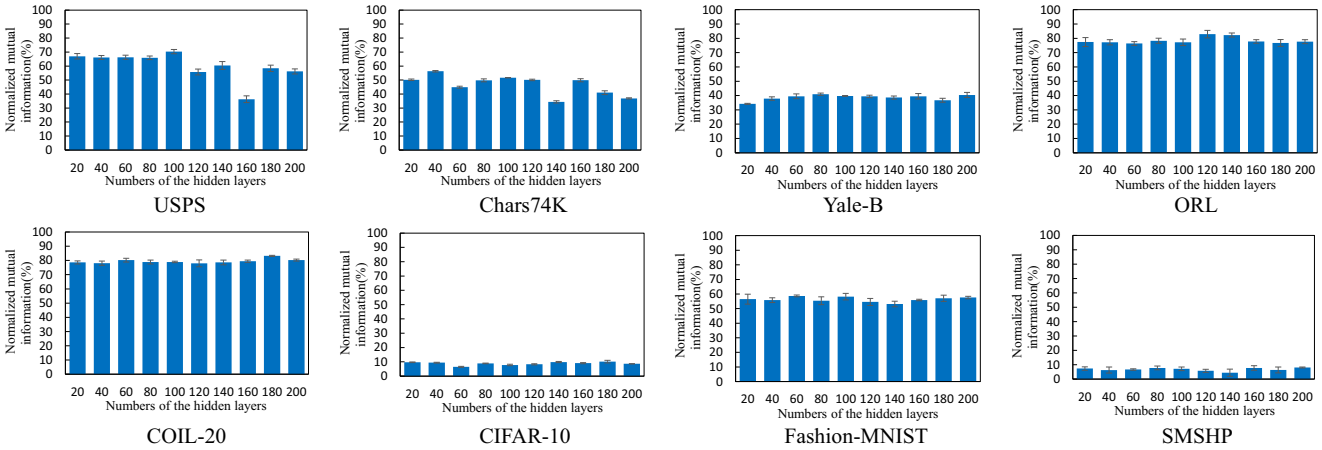

Fig. 10 The influence about different numbers of hidden layers on the normalized mutual information scores of AAE, the numbers of hidden layers is set as $\{20, 40, 60, \cdots, 200\}$

In particular, for demonstrating the influence about different numbers of hidden layers on the clustering performance of AAE, the fluctuations about the scores of three metrics with the numbers of hidden layers are shown in (Figs. 8, 9 and 10). To preserve the original feature after the reduction of dimensions is an arduous task, a low-dimensional feature that can well represent the original data is able to gain higher scores in experiments. Overall, the clustering results indicate that AAE obtains a competitive clustering performance in unsupervised learning, and comprehensive experiments demonstrate the effectiveness of learned feature representation via AAE in the tested data sets.

5 Conclusion and further discussion

An unsupervised adversarial auto-encoder was applied to learning a discriminative feature representation for image data in this paper, and the learned feature was evaluated by K-means clustering. Extensive experiments showed that adversarial auto-encoder obtained a competitive performance in image clustering. The discriminative feature learned by it could well represent the original feature after appropriate iterations. Compared to other feature representation methods, the added discriminative process led the adversarial auto-encoder to be more robust. However, when the categories increase, the clustering performance may suffer from a bottleneck without any guidance of label information. Thus in further work, we will focus on how to add weakly label information during the feature learning process, to break through this bottleneck. Besides, we will also put more emphasis on the network structure improvement about adversarial auto-encoder which may further enhance the network performance.

Acknowledgments This work was partially supported by the Technology Innovation Platform Project of Fujian Province under Grant (Nos. 2014H2005 and 2009J1007), the National Natural Science Foundation of China (Nos. 61502104 and 61672159), the Fujian Collaborative Innovation Center for Big Data Application in Governments, the Fujian Engineering Research Center of Big Data Analysis and Processing.

References

1. Balasubramanian M, Schwartz EL (2002) The isomap algorithm and topological stability. Science 295(5552):7–7
2. Bao S, Song X, Hu G, Yang X, Wang C (2017) Colour face recognition using fuzzy quaternion-based discriminant analysis. International Journal of Machine Learning and Cybernetics, pp 1–11
3. Barone AVM (2016) Towards cross-lingual distributed representations without parallel text trained with adversarial autoencoders. ACL 2016:121
4. Belkin M, Niyogi P (2002) Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in Neural Information Processing Systems, pp 585–591
5. Cai D, He X, Han J, Zhang HJ (2006) Orthogonal laplacianfaces for face recognition. IEEE Trans Image Process 15(11):3608–3614
6. Cai D, He X, Hu Y, Han J, Huang T (2007) Learning a spatially smooth subspace for face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition
7. Cui Z, Li F, Zhang W (2018) Bat algorithm with principal component analysis. International Journal of Machine Learning and Cybernetics, pp 1–20
8. De Campos TE, Babu BR, Varma M et al (2009) Character recognition in natural images. International Conference on Computer Vision Theory and Applications 7(2):273–280
9. Denton EL, Chintala S, Fergus R et al (2015) Deep generative image models using a laplacian pyramid of adversarial networks. In: Advances in Neural Information Processing Systems, pp 1486–1494
10. Estévez PA, Tesmer M, Perez CA, Zurada JM (2009) Normalized mutual information feature selection. IEEE Trans Neural Netw 20(2):189–201

11. Floyd RW (1962) Algorithm 97: shortest path. Commun ACM 5(6):345
12. Georghiades A, Belhumeur P, Kriegman D (2001) From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Trans Pattern Anal Mach Intell 23(6):643–660
13. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 580–587
14. Gogna A, Majumdar A, Ward R (2017) Semi-supervised stacked label consistent autoencoder for reconstruction and analysis of biomedical signals. IEEE Trans Biomed Eng 64(9):2196–2205
15. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp 2672–2680
16. Guo G, Li SZ, Chan K (2000) Face recognition by support vector machines. In: Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition, IEEE, pp 196–201
17. Guo X, Gao L, Liu X, Yin J (2017) Improved deep embedded clustering with local structure preservation. In: International Joint Conference on Artificial Intelligence, pp 1753–1759
18. Hartigan JA, Wong MA (1979) Algorithm as 136: a k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics) 28(1):100–108
19. He X, Cai D, Yan S, Zhang HJ (2005) Neighborhood preserving embedding. In: Tenth IEEE International Conference on Computer Vision, vol 2, IEEE, pp 1208–1213
20. He X, Niyogi P (2004) Locality preserving projections. In: Advances in Neural Information Processing Systems, pp 153–160
21. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
22. Huang JZ, Ng MK, Rong H, Li Z (2005) Automated variable weighting in k-means type clustering. IEEE Trans Pattern Anal Mach Intell 27(5):657–668
23. Joachims T (1998) Text categorization with support vector machines: Learning with many relevant features. In: European Conference on Machine Learning, Springer, pp 137–142
24. Johnson DB (1973) A note on dijkstra's shortest path algorithm. Journal of the ACM (JACM) 20(3):385–388
25. Kadurin A, Aliper A, Kazennov A, Mamoshina P, Vanhaelen Q, Khrabrov K, Zhavoronkov A (2017) The cornucopia of meaningful leads: Applying deep adversarial autoencoders for new molecule development in oncology. Oncotarget 8(7):10883
26. Khan J, Alam A, Hussain J, Lee YK (2019) Enswf: effective features extraction and selection in conjunction with ensemble learning methods for document sentiment classification. Appl Intell, pp 1–23
27. Krizhevsky A, Hinton GE (2009) Learning multiple layers of features from tiny images. Tech. rep., Citeseer
28. Makhzani A, Shlens J, Jaitly N, Goodfellow I, Frey B (2015) Adversarial autoencoders. arXiv:1511.05644
29. Marcel S, Bernier O (1999) Hand posture recognition in a body-face centered space. In: International Gesture Workshop, Springer, pp 97–100
30. Mika S, Ratsch G, Weston J, Scholkopf B, Mullers KR (1999) Fisher discriminant analysis with kernels. In: Neural Networks for Signal Processing IX. Proceedings of the 1999 IEEE Signal Processing Society Workshop, IEEE, pp 41–48
31. Mojarad M, Nejatian S, Parvin H, Mohammadpoor M (2019) A fuzzy clustering ensemble based on cluster clustering and iterative fusion of base clusters. Appl Intell 49(7):2567–2581
32. Moore B (1981) Principal component analysis in linear systems: Controllability, observability, and model reduction. IEEE Trans Autom Control 26(1):17–32
33. Nene SA, Nayar SK, Murase H et al (1996) Columbia object image library (coil-20)
34. Park HS, Jun CH (2009) A simple and fast algorithm for k-medoids clustering. Expert Syst Appl 36(2):3336–3341
35. Proedrou K, Nouretdinov I, Vovk V, Gammerman A (2002) Transductive confidence machines for pattern recognition. In: European Conference on Machine Learning, . Springer, pp 381–390
36. Qiao L, Chen S, Tan X (2010) Sparsity preserving projections with applications to face recognition. Pattern Recogn 43(1):331–341
37. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2326
38. Samko O, Marshall AD, Rosin PL (2006) Selection of the optimal parameter value for the isomap algorithm. Pattern Recogn Lett 27(9):968–979
39. Steinley D (2004) Properties of the hubert-arable adjusted rand index. Psychol Methods 9(3):386
40. Turk MA, Pentland AP (1991) Face recognition using eigenfaces. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, pp 586–591
41. Wang S, Guo W (2017) Robust co-clustering via dual local learning and high-order matrix factorization. Knowl-Based Syst 138:176–187
42. Wang S, Guo W (2017) Sparse multi-graph embedding for multimodal feature representation. IEEE Transactions on Multimedia 99:1–1
43. Wold S, Esbensen K, Geladi P (1987) Principal component analysis. Chemometr Intell Lab Syst 2(1-3):37–52
44. Xiao H, Rasul K, Vollgraf R (2017) Fashion-mnist:, a novel image dataset for benchmarking machine learning algorithms. arXiv:1708.07747
45. Xie J, Girshick R, Farhadi A (2016) Unsupervised deep embedding for clustering analysis. In: International Conference on Machine Learning, pp 478–487
46. Yu W, Teng X, Liu C (2006) Face recognition using discriminant locality preserving projections. Image Vis Comput 24(3):239–248
47. Zhang Z, Song Y, Qi H (2017) Age progression/regression by conditional adversarial autoencoder. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 5810–5818
48. Zheng Y, Doermann D (2006) Robust point matching for nonrigid shapes by preserving local neighborhood structures. IEEE Trans Pattern Anal Mach Intell 28(4):643–649
49. Zhu JY, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv:1703.10593

Wenzhong Guo received the B.S. and M.S degrees in computer science from Fuzhou University, Fuzhou, China, in 2000 and 2003, respectively, and the Ph.D. degree in communication and information system from Fuzhou University in 2010. He is currently a Full Professor with the College of Mathematics and Computer Science, Fuzhou University. His research interests include cloud computing, mobile computing, and evolutionary computation. Currently, he leads the Network Computing and Intelligent Information Processing Laboratory, which is a Key Laboratory of Fujian Province, China.

Jinyu Cai received the B.S. degree in computer science and technology from Fuzhou University, Fuzhou, China, in 2018. He is currently pursuing the M.S. degree with the College of Mathematics and Computer Science, Fuzhou University. His research interests include machine learning, computer vision and pattern recognition.

Shiping Wang received his Ph.D. degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China in 2014. He worked as a research fellow in Nanyang Technological University from August 2015 to August 2016. He is currently a Full Professor and Qishan Scholar with the College of Mathematics and Computer Science, Fuzhou University. His research interests include machine learning, computer vision and granular computing.