Multi-local Collaborative AutoEncoder

Jielei Chu\textsuperscript{a,b}, Hongjun Wang\textsuperscript{a,h,*}, Jing Liu\textsuperscript{c}, Zhiguo Gong\textsuperscript{d}, Tianrui Li\textsuperscript{a,b}

\textsuperscript{a}School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu 611756, China
\textsuperscript{b}Institute of Artificial Intelligence, Southwest Jiaotong University, Chengdu 611756, China
\textsuperscript{c}School of Business, Sichuan University, Sichuan, 610065, Chengdu, China
\textsuperscript{d}State Key Laboratory of Internet of Things for Smart City, Department of Computer and Information Science, University of Macau, Macau, China

Abstract

The excellent performance of representation learning of autoencoders have attracted considerable interest in various applications. However, the structure and multi-local collaborative relationships of unlabeled data are ignored in their encoding procedure that limits the capability of feature extraction. This paper presents a Multi-local Collaborative AutoEncoder (MC-AE), which consists of novel multi-local collaborative representation RBM (mcrRBM) and multi-local collaborative representation GRBM (mcrGRBM) models. Here, the Locality Sensitive Hashing (LSH) method is used to divide the input data into multi-local cross blocks which contains multi-local collaborative relationships of the unlabeled data and features since the similar multi-local instances and features of the input data are divided into the same block. In mcrRBM and mcrGRBM models, the structure and multi-local collaborative relationships of unlabeled data are integrated into their encoding procedure. Then, the local hidden features converges on the center of each local collaborative block. Under the collaborative joint influence of each local block, the proposed MC-AE has powerful capability of representation learning for unsupervised clustering. However, our MC-AE model perhaps perform training process for a long time on the large-scale and high-dimensional datasets because more local collaborative blocks are integrate into it. Five most related deep models are compared with our MC-AE. The experimental results show that the

*Corresponding author at: School of Computing and Artificial Intelligence, Southwest Jiaotong University, China

Email address: wanghongjun@swjtu.edu.cn (Hongjun Wang)
The proposed MC-AE has more excellent capabilities of collaborative representation and generalization than the contrastive deep models.

**Keywords:** Restricted Boltzmann machine, autoencoder, deep collaborative representation, feature learning, unsupervised clustering.

1. Introduction

Autoencoders have shown promising capability of representation learning and attracted considerable interest in various applications (e.g., classifications [1], dictionary learning [2] and clustering [3]). Although autoencoders are capable of learning complex mappings, how to capture meaningful structure of the latent feature is a long-term challenge in deep learning. Various other deep learning methods have been successful applied in practical applications (e.g., multi-context socially-aware navigation [4]).

There are various excellent autoencoders have been proposed [1, 2, 5]. Wang et al. [1] presented a within-class scatter information constraint-based autoencoder (WSI-AE), which minimizes the within-class scatter and the reconstruction error. The WSI-AE reduces the meaningless encoded features of classical AEs and enhances the feature discriminability. For convolutional dictionary learning problems, Tolooshams et al. [2] established a link between autoencoder and dictionary learning and proposed a constrained recurrent sparse autoencoder (CRsAE) model. For domain adaptation, Yang et al. [5] developed a dual-representation autoencoder (DRAE), which has capability to learn dual-domain-invariant representations. The DRAE has three leaning phases: 1) learn global representation of all target and source data; 2) extract local representations of instances; 3) construct dual representations by aligning the local and global representations with different weights. For semi-supervised learning, Śmieja et al. [6] presented a semi-supervised Gaussian Mixture Autoencoder (SeGMA), which has the capability to learn a joint probability distribution between the data and their classes. In the latent space, a mixture of Gaussians is chosen as a target distribution. To produce better data samples and use the class-bassed discriminating features, Karatsiolis and Schizas [7] proposed a generative denoising autoencoder model, which is sampled with a Markov chain Monte Carlo process.
More recently, some autoencoders based on generative adversarial model have been presented. Ge et al. [3] developed a dual adversarial autoencoder (Dual-AAE) model, which simultaneously maximizes the mutual information and likelihood function to extract classification and structure information. To learn interpretable latent representations for undirected graphs, Kipf and Welling [8] developed the Variational Graph AutoEncoder (VGAE), a probabilistic generative model for unsupervised graph representation learning on graph-structured data. Because of its excellent representation learning capability, it is getting more and more attention for deep clustering [9] on the image data and classification [10] on the medical data.

Restricted Boltzmann Machines (RBMs) and relevant autoencoders have been proved to be provided with the capability of representation learning [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. In our previous work [17], we also proposed a powerful variant of GRBM called pcGRBM for semi-supervised representation learning. The pairwise constraints are used to guiding the encoding procedure. In many practical applications of machine learning, the labels are scarce. Hence, we proposed a multi-clustering integration RBM (MIRBM) in [18] and developed an unsupervised feature learning architecture with MIRBM. The experiments proved that our semi-supervised pcGRBM and unsupervised MIRBM have excellent capability of representation learning. However, the structure and collaborative relationships of unlabeled data have been ignored in their encoding procedure of the shallow models.

In this paper, we focus on a novel Multi-local Collaborative AutoEncoder (MC-AE) to capture hidden features and learn collaborative representation. In the structure of the MC-AE, there are two novel variants of RBM: multi-local collaborative representation RBM (mcrRBM) and multi-local collaborative representation GRBM (mcrGRBM). First, the unlabeled input data of mcrRBM and mcrGRBM models are divided into multi-local cross blocks by the Locality Sensitive Hashing (LSH) [21] method in the dimensions of instances and features simultaneously. Then, the similar local instances and features of the input data are divided into the same block. Hence, these blocks contains multi-local collaborative relationships of the unlabeled data and feature. Furthermore, the local hidden features converge on the center of each local collaborative block in the encoding procedures of the mcrRBM and mcrGRBM models. Under the
collaborative joint influence of each local block, the proposed MC-AE has powerful capability of representation learning for unsupervised clustering.

This is the first work to capture hidden features and learn collaborative representation in autoencoder from the structure perspective of unlabeled data with multi-local collaborative relationship. The contributions can be summarized as follows:

- One novel variant of RBM called multi-local collaborative representation RBM (mcrRBM) and another novel variant of GRBM called multi-local collaborative representation GRBM (mcrGRBM) are proposed by fusing the structure of unlabeled data with multi-local collaborative relationship to capture hidden features and learn collaborative representation in their encoding procedures.

- A novel Multi-local Collaborative AutoEncoder (MC-AE) based on mcrRBM and mcrGRBM are developed. For modeling real-valued data, one architecture of the MC-AE is composed of a mcrGRBM and two mcrRBMs which have Gaussian linear visible units and binary hidden layer units. For modeling binary data, another architecture of the MC-AE is composed of with three mcrRBMs which have binary visible and hidden units.

- The experiments demonstrate that the proposed MC-AE has powerful capability of collaborative representation than five contrastive models on real-valued and binary datasets. Furthermore, the hidden collaborative features of the MC-AE show generalization ability for different clustering algorithms.

The remaining of the paper is organized as follows. The related works are introduced in Section II. The theoretical background is described in Section III. The Multi-local Collaborative AutoEncoder (MC-AE) is developed in Section IV. The experimental framework is illustrated in Section V. The experimental results and discussions are shown in Section VI. Finally, our contributions are summarized in Section VII.
2. Related Work

Collaborative representation learning originates the influential sparse representation-based classification (SRC) [22]. It has attracted much attentions in collaborative filtering [23, 24].

Various deep networks based on classical RBM and variants are developed in practical applications [25, 26, 27]. There are some most relevant work: 1) deep autoencoder (DAE) [11]; 2) feature selection algorithm for Deep Boltzmann Machines (Deep-FS) [28]; 3) collaborative deep learning (CDL) [29]; 4) full GraphRBM-based DBN (fGraphDBN) [30].

The DAE [11] as a classic unsupervised deep model consists of a stack of traditional RBMs for representation learning of binary data. It is also used to model real-valued data by replacing binary visible layer units with Gaussian linear visible units. Deep Boltzmann Machines (DBMs) [31, 32, 33] have reasonable structures to learn complex relationships between features. Taherkhani et al. [28] presented a Deep-FS model, which has powerful capability of removing irrelevant features from raw data to explore the underlying representations. Reducing irrelevant features is an important strategy to prevent negative impact in the encoding procedure. Under considering the local manifold structure of the data, Chen et al. [30] developed a graph regularized RBM (GraphRBM) to learn hidden features. To obtain superior expressive power of deep model, an fGraphDBN model was developed using a stack of GraphRBM. However, none of them have collaborative representation capabilities. By adding a collaborative strategy, Wang et al. [29] proposed a popular hierarchical Bayesian model, CDL, which jointly performs collaborative filtering and deep representation learning. In this paper, the structure and multi-local collaborative relationships of unlabeled data are fused into the encoding procedure of the proposed MC-AE. To prove the effectiveness of our models, we compare them with these most related works in the experiments.
3. Theoretical Background

3.1. Restricted Boltzmann Machine

For classic RBMs [12], its architecture is a shallow two-layer structure, which consists of a binary visible and hidden layer. The RBM is an energy based model and the energy function of it is defined by:

\[
E(v, h) = - \sum_{i \in \text{visibles}} a_i v_i - \sum_{j \in \text{hiddens}} b_j h_j - \sum_{i,j} v_i h_j w_{ij},
\]

where \( v \) and \( h \) are the visible and hidden layer vectors, respectively, \( v_i \) and \( h_j \) are the binary visible and hidden units, respectively, \( w_{ij} \) is the symmetric connection weight between them, \( a_i \) and \( b_j \) are the biases of visible and hidden units, respectively.

Given a visible vector \( v \), the binary state \( h_j \) is equal to 1 with probability

\[
p(h_j = 1|v) = \sigma(b_j + \sum_i v_i w_{ij}),
\]

where \( \sigma(x) = \frac{1}{1 + \exp(-x)} \), which is a logistic sigmoid function.

Similarly, given a hidden vector \( h \), an unbiased sample of the binary state \( v_i \) is equal to 1 with probability

\[
p(v_i = 1|h) = \sigma(a_i + \sum_j h_j w_{ij}).
\]

It is difficult to get an unbiased sample of an average of the model distribution \(<v_i h_j>_{\text{model}}\) because of low computing efficiency. Hinton proposed a faster learning algorithm by Contrastive Divergence (CD) [13], [34] method. Then the update rules of parameters is given by:

\[
\Delta w_{ij} = \varepsilon(<v_i h_j>_{\text{data}} - <v_i h_j>_{\text{recon}}),
\]

\[
\Delta a_i = \varepsilon(<v_i>_{\text{data}} - <v_i>_{\text{recon}}),
\]

\[
\Delta b_j = \varepsilon(<h_j>_{\text{data}} - <h_j>_{\text{recon}}),
\]

where \( \varepsilon \) is a learning rate, \(< \cdot >_{\text{data}}\) is an average of the data distribution and \(< \cdot >_{\text{recon}}\) is an average under the distribution of reconstructed units.
3.2. Gaussian Linear Visible Units

For modeling real-valued data, the binary visible units are replaced by Gaussian linear visible units. The energy function becomes:

\[
E(v, h) = \sum_{i \in \text{visibles}} \frac{(v_i - a_i)^2}{2 \sigma_i^2} - \sum_{j \in \text{hiddens}} b_j h_j - \sum_{i,j} \frac{v_i}{\sigma_i} h_j w_{ij},
\]

(7)

where \( \sigma_i \) is the standard deviation of the Gaussian noise for visible unit \( i \). It is difficult to use CD method to learn the variance of the noise. In practice, we normalise the original data to have unit variance and zero mean. So, the reconstructed result of a Gaussian linear visible unit is equal to the input from hidden binary units plus the bias.

3.3. Locality Sensitive Hashing

The Locality Sensitive Hashing (LSH) [21] exploits the probability that two similar samples likely collide by mapping with a weak hash function. In fact, the probability of the collision is proportional to their similarity. One classic hash function is the Minwise Independent Permutation (Minhash) [35] which defines the probability of collision is proportional to the Jaccard similarity of two hashed objects. The Jaccard similarity varies from 0 to 1. The value of it is 1 means that the two hashed objects are equal.

4. Multi-local Collaborative AutoEncoder

In this section, we firstly present the key basics of unsupervised Multi-local Collaborative AutoEncoder (MC-AE) that is the mcrRBM and mcrGRBM models (novel variants of RBM and GRBM). Then, we show the inference, learning algorithm and complexity analysis of the mcrRBM model. Finally, we propose two MC-AE deep architectures based on the mcrRBM and mcrGRBM models for modeling real-valued and binary data, respectively.

4.1. The mcrRBM and mcrGRBM Models

In this section, we present the key basics of the MC-AE deep architecture that is the mcrRBM (see Fig.1) and mcrGRBM models (see Fig.2). Here, we use the LSH [21]
method to divide the input data into multi-local cross blocks with the perspective of instances and features simultaneously. Furthermore, the similar multi-local instances and features of the input data are divided into the same block. Then, we expect the hidden layer feature units of each block converges on the block center as much as possible in the encoding procedure of mcrRBM and mcrGRBM models. By this way, the correlations between the instances and features (multi-local collaborative relations) are fused in the hidden layer features. Due to the same mapping relations from the visible layer units to hidden layer units (sigmoid transformation) between the mcrRBM and mcrGRBM models, we only present the mcrRBM model and its inference and learning algorithm.

Let \( \mathcal{D} = \{ \mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_N \} \) be an original data set with \( N \) vectors and \( M \) features of each vector. The visible layer vector \( \mathbf{v}_s = (v_{s1}, v_{s2}, \cdots, v_{sM}) \), \( (i = 1, 2, \cdots, M, \text{and } s = 1, 2, \cdots, N) \). The hidden layer vector \( \mathbf{h}_s = (h_{s1}, h_{s2}, \cdots, h_{sM}) \), \( (j = 1, 2, \cdots, M', \text{and } s = 1, 2, \cdots, N) \). The reconstructed visible layer vector \( \mathbf{v}^{(r)}_s = (v^{(r)}_{s1}, v^{(r)}_{s2}, \cdots, v^{(r)}_{sM}) \), \( (i = 1, 2, \cdots, M, \text{and } s = 1, 2, \cdots, N) \). The reconstructed hidden layer vector \( \mathbf{h}^{(r)}_s = (h^{(r)}_{s1}, h^{(r)}_{s2}, \cdots, h^{(r)}_{sM}) \), \( (j = 1, 2, \cdots, M', \text{and } s = 1, 2, \cdots, N) \). The matrix \( (\mathbf{v}_1^T \mathbf{v}_2^T \cdots \mathbf{v}_N^T)^T \) is partitioned into \( K \) row clusters by LSH and each cluster has a serial number set of vectors \( \mathcal{R}_k \), \( (k = 1, 2, \cdots, K \text{ and } \mathcal{R}_1 \cup \mathcal{R}_2 \cdots \cup \mathcal{R}_K = \{1, 2, \cdots, N\}) \). Simultaneously, the matrix is partitioned into \( L \) column clusters by LSH and each cluster has a serial number set of vectors \( \mathcal{S}_l \).
vectors $\ell_l, (l = 1, 2, \cdots, L)$ and $\ell_1 \cup \ell_2 \cdots \cup \ell_K = \{1, 2, \cdots, M\}$. So, the matrix is divided into $K \times L$ blocks.

Based on the expectations of our collaborative representation method and the training objective of classic RBM, our novel objective function takes the form:

$$G(\mathbf{v}, \theta) =$$

$$- \frac{\eta}{N} \sum_{\mathbf{v}_i} \log p(\mathbf{v}_i; \theta) + \frac{(1 - \eta)}{K \times L} \left[ \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{s \in \mathbb{R}_k} \sum_{t \in \ell_l} d(h_{st}, u_{kl}) + \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{s \in \mathbb{R}_k} \sum_{t \in \ell_l} d(h_{st}^{(r)}, u_{kl}^{(r)}) \right],$$

where $\theta = \{a, b, W\}$ are the model parameters, $\eta$ is an adjusting parameter, $d(h_{st}, u_{kl})$ and $d(h_{st}^{(r)}, u_{kl}^{(r)})$ are the Bregman divergences \cite{36} distances, which are defined as:

$$d(h_{st}, u_{kl}) = (h_{st} - u_{kl})^2$$

and

$$d(h_{st}^{(r)}, u_{kl}^{(r)}) = (h_{st}^{(r)} - u_{kl}^{(r)})^2,$$

respectively. $u_{kl}$ and $u_{kl}^{(r)}$ are the centers of block $(\mathbb{R}_k, \ell_l)$ in hidden layer and reconstructed hidden layer, respectively. They take the form:

$$u_{kl} = \frac{\sum_{s \in \mathbb{R}_k} \sum_{t \in \ell_l} h_{st}}{|\mathbb{R}_k| |\ell_l|}, u_{kl}^{(r)} = \frac{\sum_{s \in \mathbb{R}_k} \sum_{t \in \ell_l} h_{st}^{(r)}}{|\mathbb{R}_k| |\ell_l|}.$$

We expect all units close to their center of each local collaborative block in the representation learning process.
4.2. The Inference

In this subsection, we use the gradient descent algorithm to obtain the update rules of the parameters of the mcrRBM model. The detailed inference is shown as follows.

Suppose that

\[ C_{data} = \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{s \in \mathcal{R}_k} \sum_{t \in \ell_l} (h_{st} - u_{kl})^2, \]  \hspace{1cm} (10)

\[ C_{recon} = \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{s \in \mathcal{R}_k} \sum_{t \in \ell_l} (h_{st}^{(r)} - u_{kl}^{(r)})^2. \] \hspace{1cm} (11)

Using the introduced variables \( C_{data} \) and \( C_{recon} \), the objective function have another concise equivalent form:

\[ G(V, \theta) = -\eta \frac{1}{N} \sum_{v_i} \log p(v_i; \theta) + (1 - \eta) \frac{1}{KL} (C_{data} + C_{recon}). \] \hspace{1cm} (12)

The following crucial problem is how to solve this multi-objective optimization problem. For the average log-likelihood \( \frac{1}{N} \sum_{v_i} \log p(v_i; \theta) \), the CD method was presented to approximately follow the gradient of two divergences \( \text{CD}_n = \text{KL}(p_0 || p_\infty) - \text{KL}(p_n || p_\infty) \) to avoid enormous difficulties of the log-likelihood gradient computing.

Normally, we run the Markov chain from the data distribution \( p_0 \) to \( p_1 \) (one step) in CD learning. So, the following key task is how to obtain the approximative gradient of \( C_{data} + C_{recon} \).

Suppose that \( J_{data} = h_{st} - u_{kl} \), \( J_{recon} = h_{st}^{(r)} - u_{kl}^{(r)} \), then

\[ J_{data} = h_{st} - \sum_{s \in \mathcal{R}_k} \sum_{t \in \ell_l} h_{st} \frac{\sigma (\sum_{m=1}^{M} v_{sm} w_{mt} + b_{st})}{|\mathcal{R}_k| |\ell_l|} = \sigma (\sum_{m=1}^{M} v_{sm} w_{mt} + b_{st}) \]  \hspace{1cm} (13)

and

\[ J_{recon} = h_{st}^{(r)} - \sum_{s \in \mathcal{R}_k} \sum_{t \in \ell_l} h_{st}^{(r)} \frac{\sigma (\sum_{m=1}^{M} v_{sm}^{(r)} w_{mt} + b_{st}^{(r)})}{|\mathcal{R}_k| |\ell_l|} = \sigma (\sum_{m=1}^{M} v_{sm}^{(r)} w_{mt} + b_{st}^{(r)}) \] \hspace{1cm} (14)
where \( \sigma \) is a sigmoid function.

When \( t = j \in \ell_1 \), the partial derivative of \( J_{data} \) is given by:

\[
\frac{\partial J_{data}}{\partial w_{ij}} = e^{-\left( \sum_{m=1}^{M} v_{sm} w_{mj} + b_{sj} \right)} v_{si} \left[ 1 + e^{-\left( \sum_{m=1}^{M} v_{sm} w_{mj} + b_{sj} \right)} \right]^{-2} \sum_{s \in \mathbb{R}_k} e^{-\left( \sum_{m=1}^{M} v_{sm} w_{mj} + b_{sj} \right)} v_{si} \left[ 1 + e^{-\left( \sum_{m=1}^{M} v_{sm} w_{mj} + b_{sj} \right)} \right]^{-2} \sum_{s \in \mathbb{R}_k} (1 - h_{sj}) h_{sj} v_{si} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}) h_{sj} v_{si} |\mathbb{R}_k|}{|\mathbb{R}_k|}. \tag{15}
\]

Obviously, if \( t \neq j \), then \( \frac{\partial J_{data}}{\partial w_{ij}} = 0 \).

Similarly, if \( t = j \in \ell_1 \), the partial derivative of \( J_{recon} \) is given by:

\[
\frac{\partial J_{recon}}{\partial w_{ij}} = (1 - h_{sj}(r)) h_{sj}(r) v_{si} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}(r)) h_{sj}(r) v_{si}}{|\mathbb{R}_k|}. \tag{16}
\]

As for model parameter \( b \), if \( t = j \), the partial derivative takes the forms:

\[
\frac{\partial J_{data}}{\partial b_j} = (1 - h_{sj}) h_{sj} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}) h_{sj}}{|\mathbb{R}_k|}, \tag{17}
\]

\[
\frac{\partial J_{recon}}{\partial b_j} = (1 - h_{sj}(r)) h_{sj}(r) - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}(r)) h_{sj}(r)}{|\mathbb{R}_k|}.
\]

It is obvious that model parameter \( a \) is independent of \( J_{data} \) and \( J_{recon} \). So, we can obtain that \( \frac{\partial J_{data}}{\partial a_i} = 0 \) and \( \frac{\partial J_{recon}}{\partial a_i} = 0 \). Then, the partial derivative of the \( C_{data} \) in terms of \( w_{ij} \) takes the form:

\[
\frac{\partial C_{data}}{\partial w_{ij}} = 2 \sum_{k=1}^{K} \sum_{s \in \mathbb{R}_k} \left( h_{sj} - \frac{\sum_{s \in \mathbb{R}_k} h_{sj}}{|\mathbb{R}_k|} \right) \left( 1 - h_{sj} \right) h_{sj} v_{si} - \frac{\sum_{s \in \mathbb{R}_k} \left( 1 - h_{sj} \right) h_{sj} v_{si} |\mathbb{R}_k|}{|\mathbb{R}_k|}. \tag{18}
\]
And the partial derivative of the $C_{\text{recon}}$ in terms of $w_{ij}$ takes the form:

$$\frac{\partial C_{\text{recon}}}{\partial w_{ij}} = 2\sum_{k=1}^{K} \sum_{s \in \mathbb{R}_k} \left( h_{ij}^{(r)} - \frac{\sum_{s \in \mathbb{R}_k} h_{sj}^{(r)}}{|\mathbb{R}_k|} \right) \left[ (1 - h_{sj}^{(r)}) h_{sj}^{(r)} v_{si}^{(r)} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}^{(r)}) h_{sj}^{(r)} v_{si}^{(r)}}{|\mathbb{R}_k|} \right].$$

(19)

Similarly, the partial derivative of the $C_{\text{data}}$ in terms of $b_j$ is given by:

$$\frac{\partial C_{\text{data}}}{\partial b_j} = 2\sum_{k=1}^{K} \sum_{s \in \mathbb{R}_k} \left( h_{sj} - \frac{\sum_{s \in \mathbb{R}_k} h_{sj}}{|\mathbb{R}_k|} \right) \left[ (1 - h_{sj}) h_{sj} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}) h_{sj}}{|\mathbb{R}_k|} \right].$$

(20)

And the partial derivative of the $C_{\text{recon}}$ in terms of $b_j$ is given by:

$$\frac{\partial C_{\text{recon}}}{\partial b_j} = 2\sum_{k=1}^{K} \sum_{s \in \mathbb{R}_k} \left( h_{sj}^{(r)} - \frac{\sum_{s \in \mathbb{R}_k} h_{sj}^{(r)}}{|\mathbb{R}_k|} \right) \left[ (1 - h_{sj}^{(r)}) h_{sj}^{(r)} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}^{(r)}) h_{sj}^{(r)}}{|\mathbb{R}_k|} \right].$$

(21)

Combined with the CD learning with 1 step Gibbs sampling, the update rule of the proposed model parameter $W$ takes the forms:

$$w_{ij}^{(r+1)} = w_{ij}^{(r)} + \eta \varepsilon \left( \langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{recon}} \right) + 2(1 - \eta) \left\{ \sum_{k=1}^{K} \sum_{s \in \mathbb{R}_k} h_{sj} - \frac{\sum_{s \in \mathbb{R}_k} h_{sj}}{|\mathbb{R}_k|} \right\} \left[ (1 - h_{sj}) h_{sj} v_{si} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}) h_{sj} v_{si}}{|\mathbb{R}_k|} \right] + \sum_{k=1}^{K} \sum_{s \in \mathbb{R}_k} \left( h_{sj}^{(r)} - \frac{\sum_{s \in \mathbb{R}_k} h_{sj}^{(r)}}{|\mathbb{R}_k|} \right) \left[ (1 - h_{sj}^{(r)}) h_{sj}^{(r)} v_{si} - \frac{\sum_{s \in \mathbb{R}_k} (1 - h_{sj}^{(r)}) h_{sj}^{(r)} v_{si}}{|\mathbb{R}_k|} \right],$$

(22)

where $\varepsilon$ is learning rate, the average $\langle v_i h_j \rangle_{\text{data}}$ and $\langle v_i h_j \rangle_{\text{recon}}$ are computed using the sample data and reconstructed data, respectively.
For the parameters of the biases $\mathbf{a}$ and $\mathbf{b}$, the update rules of them take the forms:

$$a_i^{(\tau+1)} = a_i^{(\tau)} + \eta \varepsilon (< v_i >_{data} - < v_i >_{recon})$$  \hspace{1cm} (23)$$

and

$$b_j^{(\tau+1)} = b_j^{(\tau)} + \eta \varepsilon (< h_j >_{data} - < h_j >_{recon}) + \frac{2(1-\eta)}{K \times L} \left\{ \sum_{k=1}^{K} \sum_{s \in \mathcal{R}_k} (h_{sj} - \frac{\sum_{s \in \mathcal{R}_k} h_{sj}}{|\mathcal{R}_k|}) (1 - h_{sj}) h_{sj} - \sum_{s \in \mathcal{R}_k} h_{sj}^{(r)} \right\} + \sum_{k=1}^{K} \sum_{s \in \mathcal{R}_k} \left( h_{sj}^{(r)} - \frac{\sum_{s \in \mathcal{R}_k} h_{sj}^{(r)}}{|\mathcal{R}_k|} \right)$$  \hspace{1cm} (24)$$

4.3. The Algorithm

**Algorithm 1** Learning algorithm of mcrRBM with 1 step Gibbs sampling

---

**Input:** $\mathcal{D}$: input data sets;  
$\mathcal{B}$: training batch sets;  
$\varepsilon$: learning rate;  
$(\mathcal{R}_k, \ell_l)$: matrix blocks of $\mathcal{D}$, $k \in [1, K]$ and $l \in [1, L]$;

**Output:** $\theta$: model parameters of mcrRBM.

---

Initialize: $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{W}$.

**while** $\tau$ not exceeding maximum iteration **do**

**for** all training batch $\mathcal{B}$ **do**

Encoder: sample the states of hidden units by

$$p(h_j = 1|\mathbf{v}) = \sigma (b_j + \sum_i v_i w_{ij}).$$

Decoder: sample the reconstructed states of visible units using

$$p(v_i = 1|h) = \sigma (a_i + \sum_j h_j w_{ij}).$$

**for** all $\mathcal{R}_k$ **do**

---

13
for all $\ell_i$ do
  Compute the partial derivative $\frac{\partial C_{\text{data}}}{\partial w_{ij}}$ using Eq. (18).
  Compute the partial derivative $\frac{\partial C_{\text{recon}}}{\partial w_{ij}}$ using Eq. (19).
  Compute the partial derivative $\frac{\partial C_{\text{data}}}{\partial b_j}$ using Eq. (20).
  Compute the partial derivative $\frac{\partial C_{\text{recon}}}{\partial b_j}$ using Eq. (21).
end for
end for

t = t + 1.
end while
return $a$, $b$ and $W$.

In the reconstruction process of mcrGRBM model, a linear reconstruction method replaces the nonlinear reconstruction method of mcrRBM model. The steps of the learning algorithms of our mcrRBM and mcrGRBM models are almost the same, except the reconstruction process. So, we omit the learning algorithm of mcrGRBM model.

4.4. Complexity Analysis

In this subsection, we analyze the time complexity of above learning algorithm. Supposing that the input data sets $D$ is divided into $TB$ training batch. Then the time complexities of the encoder and decoder steps are $O(TB)$ in each iteration. When partial derivatives $\frac{\partial C_{\text{data}}}{\partial w_{ij}}$, $\frac{\partial C_{\text{recon}}}{\partial w_{ij}}$, $\frac{\partial C_{\text{data}}}{\partial b_j}$ and $\frac{\partial C_{\text{recon}}}{\partial b_j}$ are calculated, they take $O(TB \times (K \times L))$ in each iteration. The complexities of update parameters $W$, $a$ and $b$ are $O(TB)$ in each iteration. Supposing that the maximum iteration is $IT$. Then, the time complexity of the mcrRBM learning algorithm with 1 Step Gibbs sampling is $O(IT \times TB \times (K \times L))$. 

14
4.5. The MC-AE Deep Architecture

A novel Multi-local Collaborative AutoEncoder (MC-AE) architecture is developed with one visible layer and three hidden layers (see Fig. 3). To learn collaborative representation of two types of input data (binary and real-valued), the visible layer units can be designed as binary and linear units, respectively. In other words, one architecture of MC-AE for modeling binary data consists of three mcrRBM. And another architecture of MC-AE for modeling real-valued data consists of one mcrGRBM and two mcrRBMs. In the encoding procedure, the first multi-local collaborative blocks (MCB 1) comes from raw data by LSH method. Then, local collaborative relationships of the unlabeled data and feature force the local hidden features to converge on the center of each local collaborative block. The second multi-local collaborative blocks (MCB 2) is generated by LSH method from the first hidden layer. Similarly, they are fused into the second hidden layer, and so forth. The next experiments confirm the collaborative joint influence of each local block to improve the capability of representation learning of the proposed MC-AE.

5. Experimental Framework

This section introduces the experimental datasets used in the current work, the experimental settings and the evaluation metrics.

5.1. Datasets

To explore the collaborative representation capability of the proposed MC-AE for real-valued data, we do experiments on ten image datasets from MSRA-MM 2.0 [37]. The summaries of them are listed from No. 1 to No. 10 in Table 1. All of them have same class, but different instances and features. The datasets of banner, beret, bugatti and building have 892 features, but the vista, vistawallpaper, water, wing and worldmap have 899 features. To explore the collaborative representation capability of our MC-AE for binary data further, we do experiments on ten UCI datasets 1. The

1http://archive.ics.uci.edu/ml/index.php
Figure 3: Multi-local Collaborative AutoEncoder (MC-AE). One architecture of MC-AE (Linear visible layer and binary hidden layer) consists of one mcrGRBM and two mcrRBMs for modeling real-valued data. Another architecture of MC-AE (visible and hidden layer units are both binary) consists of three mcrRBMs for modeling binary data.

summaries of them are listed from No. 11 to No. 20 in Table 1. They have different classes, instances and features.

5.2. Experimental Settings

To validate the capability of collaborative representation of the proposed MC-AE, we compare it with the DAE [11], Deep-FS [28], fGraphDBN [30] and VGAE [8], which have not collaborative representation strategy. Furthermore, we compare our MC-AE with CDL [29] model, which has collaborative representation strategy. For modeling real-valued data, the MC-AE architecture consists of three binary hidden layer and one linear visible layer in Fig. 3. The transformation functions of all hidden layers between encoding and decoding are sigmoid functions. But, the transformation
functions of encoding and decoding between visible layer and the first hidden layer are sigmoid and linear functions, respectively. For modeling binary data, the MC-AE architecture consists of three binary hidden layers and one binary visible layer in Fig. 3. In other words, all transformation functions are both sigmoid function.

To compare the generalization capabilities of our MC-AE for representation learning, two different unsupervised clustering algorithms: K-means [38] and Spectral Clustering (SC) [39] are applied to clustering task with the representation of the deepest hidden layers of all contrastive deep models. The clustering algorithms based on the DAE, Deep-FS, fGraphDBN, CDL, VGAE and our MC-AE models using K-means are called DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM, VGAE+KM and MC-AE+KM, respectively. Similarly, the clustering algorithms based on the DAE, Deep-FS, fGraphDBN, CDL, VGAE and the proposed MC-AE models using SC are called DAE+SC, Deep-FS+SC, fGraphDBN+SC, CDL+SC, VGAE+SC and MC-AE+SC, respectively.
In two frameworks of the proposed MC-AE, the dimensionality of each hidden and visible layer is same as the raw data. The learning rate and $\eta$ of our MC-AE are set to 0.001 and 0.1, respectively. The parameters of other contrastive models adopt the values suggested in their papers.

Our MC-AE+KM and MC-AE+SC methods are implemented in Matlab 2016 (a). All contrastive methods have run on a Server with Core i9 CPU and 64 GB RAM.

5.3. Evaluation Metrics

In this paper, three classical clustering evaluation metrics: clustering accuracy (ACC) [40], Jaccard index (Jac) [21] and Fowlkes-Mallows index (FMI) [41] are utilized to evaluate the performance of the proposed MC-AE model. Furthermore, the Friedman Aligned Ranks test statistic [42] is used to report significant differences of all contrastive algorithms. The ACC evaluation metric takes the form:

$$\text{accuracy} = \frac{\sum_{i=1}^{n} \delta(s_i, \text{map}(r_i))}{n},$$

where $\text{map}(r_i)$ maps label $r_i$ of each cluster to the equivalent label and $n$ is the total number of instances. If $x = y$, then $\delta(x, y)$ equals to 1. Otherwise, it is zero. The Jac evaluation metric is defined by:

$$\text{Jac} = \frac{|A \cap B|}{|A \cup B|},$$

where $A$ and $B$ are finite sample sets and $0 \leq J(A, B) \leq 1$. The FMI evaluation metric is given by:

$$\text{FMI} = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP + FN}},$$

where $TP$ is the number of true positives, $FP$ is the number of false positives and and $FN$ is the number of false negatives.

The Friedman Aligned Ranks test statistic [42] takes the form:

$$T = \frac{(n_a - 1)(\sum_{j=1}^{n_a} \hat{r}_{ij}^2 - n_a n_d (n_a n_d + 1)^2 / 4)}{n_a n_d (n_a n_d + 1)(2n_a n_d + 1) / 6 - \sum_{i=1}^{n_d} \hat{r}_{ij}^2 / n_a},$$

where $\hat{r}_{i,j}$ and $\hat{r}_{i,j}$ are the ranks total of the $j$th algorithm and $i$th data set, respectively, $n_a$ and $n_d$ are the numbers of algorithm and data set, respectively. For $n_a - 1$ degrees
of freedom, the test statistic $T$ is compared for significance with a chi-square distribution.

6. Results and Discussion

For fairness of comparisons, our MC-AE+KM algorithms based on the proposed MC-AE model are compared with DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM and VGAE+KM, respectively. Similarly, our MC-AE+SC algorithms based on the proposed MC-AE model are compared with DAE+SC, Deep-FS+SC, fGraphDBN+SC, CDL+SC and VGAE+SC, respectively. Moreover, the results (ACC, Jac and FMI) of K-means and SC algorithms on original real-valued datasets and UCI datasets are listed in Table 10 and Table 11 for comparisons, respectively.

6.1. Representation Learning for clustering on Real-valued Datasets

6.1.1. Accuracy

Table 2 shows the results of the ACC (mean±std) of each contrastive algorithm on each dataset and the average ACC ($\overline{ACC}$) of each algorithm is listed in the last column. The MC-AE+KM algorithm based on the proposed MC-AE model shows the best performance on the banner, beret, building, vista, voituretuning and wing datasets. The ACC of them are 0.9372, 0.6895, 0.7164, 0.6308, 0.6394 and 0.6192, respectively. The MC-AE+SC algorithm based on the proposed MC-AE shows the best performance on the bugatti, vistawallpaper, water and woldmap datasets. The ACC of them are 0.7007, 0.6320, 0.5705 and 0.7134, respectively. The average ACC of MC-AE+KM and MC-AE+SC algorithms are 0.6335 and 0.6500, respectively. They show the best performance in the corresponding comparative grouping.

An intuitive comparison of the overall performance (average ACC) is shown in Fig. 5 (left one). From Table 2 and Fig. 5, we can draw the conclusion that the proposed MC-AE shows the better performance than other deep models (DAE, Deep-FS, fGraphDBN, CDL and VGAE) in terms of the ACC metric.
6.1.2. Jaccard Index

Table 4 shows the results of the Jac of each contrastive algorithm on each dataset and the average Jac ($\bar{Jac}$) of each algorithm is listed in the last column. The MC-AE+KM algorithm based on the proposed MC-AE shows the best performance on the banner, beret, building, vista, voituretuning and wing datasets. The Jac of them are 0.8820, 0.5348, 0.5574, 0.4738, 0.4760 and 0.4714, respectively. The MC-AE+SC algorithm based on the proposed MC-AE shows the best performance on the bugatti, vistawallpaper, water and worldmap datasets. The Jac of them are 0.5420, 0.4736, 0.4351 and 0.5602, respectively. The average Jac of MC-AE+KM and MC-AE+SC algorithms are 0.4771 and 0.5311, respectively. They also show most excellent performance in the corresponding comparative grouping.

The intuitive comparison of overall performance (average Jac) is shown in Fig. 5 (middle one). Therefore, we can draw the conclusion that the proposed MC-AE shows the best performance among all contrastive deep models in terms of the Jac metric from Table 4 and Fig. 5.

6.1.3. Fowlkes and Mallows Index

Table 5 shows the results of the FMI of each contrastive algorithm on each dataset and the average FMI ($\bar{FMI}$) of each algorithm is listed in the last column. The MC-AE+KM algorithm based on the proposed MC-AE shows the best performance on the banner, beret, building, vista, voituretuning and wing datasets. The FMI of them are 0.9392, 0.7313, 0.7466, 0.6870, 0.6899 and 0.6866, respectively. The MC-AE+SC algorithm based on the proposed MC-AE shows the best performance on the bugatti, vistawallpaper, water and worldmap datasets. The FMI of them are 0.7279, 0.6865, 0.6585 and 0.7460, respectively. The average FMI of MC-AE+KM and MC-AE+SC algorithms are 0.6542 and 0.7172, respectively. They show the best performance in the corresponding comparative grouping.

An intuitive comparison of average FMI is shown in Fig. 5 (right one). In terms of the FMI metric, we also can draw the conclusion that the proposed MC-AE shows the better performance than DAE, Deep-FS, fGraphDBN, CDL and VGAE deep models from Fig. 5 and Table 5.
6.1.4. The Friedman Aligned Ranks Test Statistic

Table 3 shows the ranks (in the parentheses) and average ranks of all contrastive algorithms. The smaller rank means the better performance of the algorithm on the corresponding dataset. The average ranks of MC-AE+SC and MC-AE+KM algorithms based on our MC-AE are 15.7 and 16.8, respectively. However, the average ranks of DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM and VGAE+KM are 95.2, 74.9, 71.45, 77.25 and 76.7, respectively. And the ranks of DAE+SC, Deep-FS+SC, fGraphDBN+SC, CDL+SC and VGAE+SC are 110.9, 53.9, 65.9, 37.8 and 29.5, respectively. Clearly, MC-AE+SC and MC-AE+KM algorithms show the best performance in the corresponding comparative grouping. By means of the Friedman Aligned test statistic, \( T = 7.7217 \) is the chi-square distribution with 11 degrees of freedom. The \( p \)-value is 0.00000457 which is computed by \( \chi^2(11) \) distribution for one tailed test and the two-tailed probability is 0.00000913. Then, the null hypothesis is rejected at a high level significance. The \( p \)-values are far less than 0.05, so the experimental results of algorithms are different.

6.1.5. Friedman + Post-hoc Nemenyi Tests

The results of Friedman test + post-hoc Nemenyi test [43] are shown in Fig. 4 among all contrastive methods on real-valued datasets. It is obvious that the test values of MC-AE+KM based on our MC-AE model versus DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM and VGAE+KM are less than significance level (5%). Hence, there are striking differences between MC-AE+KM and five related contrastive methods (DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM and VGAE+KM). In Fig. 4, most of the test values between our MC-AE+SC and five related contrastive methods (DAE+SC Deep-FS+SC, fGraphDBN+SC, CDL+SC and VGAE+SC) are less than 5% significance level expect for the results of MC-AE+SC versus CDL+SC and VGAE+SC methods. So, although the MC-AE+SC method based on MC-AE model has better performance than CDL+SC and VGAE+SC methods, there are no significant difference between MC-AE+SC and them.
Figure 4: The results of Friedman + post-hoc Nemenyi tests among all contrastive algorithms on real-valued datasets.

Figure 5: The performance comparisons of DAE, Deep-FS, fGraphDBN, CDL, VGAE and our MC-AE using average ACC, Jac and FMI metrics on the real-valued datasets.
6.2. Representation Learning for Clustering on UCI Datasets

6.2.1. Accuracy

Table 6 shows the results of the ACC (mean±std) of each contrastive algorithm on each UCI dataset and the average ACC (\(\overline{ACC}\)) of each algorithm is listed in the last column. The MC-AE+KM algorithm based on the proposed MC-AE shows the best performance on the balance, biodegradation, dermatology, Kdd, OLD, parkinsons and secom datasets. The ACC of them are 0.6224, 0.6673, 0.4754, 0.9877, 0.9369, 0.8051 and 0.9336, respectively. The MC-AE+SC algorithm based on the proposed MC-AE shows the best performance on the car and HabermanSurvival datasets. The ACC of them are 0.6985 and 0.7255. The average ACC of MC-AE+KM and MC-AE+SC algorithms are 0.7049 and 0.7108, respectively. They show the most excellent performance in the corresponding comparative grouping.

An intuitive comparison of the overall performance (average ACC) is shown in Fig. 7 (left one). From Table 6 and Fig. 7, we can draw the conclusion that the proposed MC-AE shows the better performance than other deep models (DAE, DeepFS, fGraphDBN, CDL and VGAE) in terms of the ACC metric.

6.2.2. Jaccard Index

Table 8 shows the results of the Jac of each contrastive algorithm on each dataset and the average Jac (\(\overline{Jac}\)) of each algorithm is listed in the last column. The MC-AE+KM algorithm based on the proposed MC-AE shows the best performance on the dermatology, Kdd, OLD and secom datasets. The Jac of them are 0.3555, 0.9806, 0.8816 and 0.7634, respectively. The MC-AE+SC algorithm based on the proposed MC-AE shows the best performance on the balance, biodegradation, car and parkinsons datasets. The Jac of them are 0.4285, 0.5525, 0.5408 and 0.6319, respectively. For the ClimateMode and HabermanSurvival datasets, the fGraphDBN+SC and DAE+KM algorithms show the best performance, respectively. Nevertheless the MC-AE+KM and MC-AE+SC algorithms show the best performance in the corresponding comparative grouping. The average Jac of them are 0.5137 and 0.5602, respectively.

The intuitive comparison of overall performance (average Jac) is shown in Fig. 7 (middle one). Therefore, we can draw the conclusion that the proposed MC-AE shows
the best performance among all contrastive deep models in terms of the Jac metric from Table 8 and Fig. 7.

6.2.3. Fowlkes and Mallows Index

Table 9 shows the results of the FMI of each contrastive algorithm on each UCI dataset and the average FMI ($FMI$) of each algorithm is listed in the last column. The MC-AE+KM algorithm based on the proposed MC-AE shows the best performance on the dermatology, Kdd, OLD, and secom datasets. The FMI of them are 0.5249, 0.9853, 0.9390, and 0.8658, respectively. The MC-AE+SC algorithm based on the proposed MC-AE shows the best performance on the balance, biodegradation, car, and parkinsons datasets. The FMI of them are 0.6533, 0.7429, 0.7346, and 0.7938, respectively. For the ClimateMode and HabermanSurvival datasets, the fGraphDBN+SC and DAE+KM algorithms show the best performance, respectively. Nevertheless, the MC-AE+KM and MC-AE+SC algorithms show the best performance in the corresponding comparative grouping. The average FMI of them are 0.6498 and 0.7315, respectively.

An intuitive comparison of average FMI is shown in Fig. 7 (right one). In terms of the FMI metric, we also can draw the conclusion that the proposed MC-AE shows the better performance than DAE, Deep-FS, fGraphDBN, CDL, and VGAE deep models from Fig. 7 and Table 9.

6.2.4. The Friedman Aligned Ranks Test Statistic

Table 7 shows the ranks (in the parentheses) and average ranks of all contrastive algorithms. The smaller rank means the better performance of the algorithm on the corresponding dataset. The average ranks of MC-AE+SC and MC-AE+KM algorithms based on our MC-AE are 29.8.7 and 35.5, respectively. However, the average ranks of DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM, and VGAE+KM are 71.7, 84.5, 74.75, 78.1, and 78.4, respectively. And the ranks of DAE+SC, Deep-FS+SC, fGraphDBN+SC, CDL+SC, and VGAE+SC are 47.8, 49.7, 60.45, 59.7, and 54.6, respectively. Clearly, MC-AE+SC and MC-AE+KM algorithms show the most excellent performance in the corresponding comparative grouping. By means of the Friedman Aligned test statistic, $T=7.2492$ is the chi-square distribution with 11 degrees of free-
The $p$-value is 0.00000823 which is computed by $\chi^2(11)$ distribution for one tailed test and two-tailed probability is 0.00001646. Then, the null hypothesis is rejected at a high level significance. The $p$-values are far less than 0.05, so the experimental results of algorithms are different.

6.2.5. Friedman + Post-hoc Nemenyi Tests

The results of Friedman test + post-hoc Nemenyi test [43] are shown in Fig. 6 among all contrastive methods on UCI datasets. It is obvious that the test values of MC-AE+KM based on our MC-AE model versus DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM and VGA+KM are less than 0.05. Hence, there are striking differences between MC-AE+KM and five related contrastive methods (DAE+KM, Deep-FS+KM, fGraphDBN+KM, CDL+KM and VGA+KM). In Fig. 6, the test values between our MC-AE+SC and five related contrastive methods (DAE+SC, Deep-FS+SC, fGraphDBN+SC, CDL+SC and VGA+SC) are 0.4865, 0.2108, 0.0852, 0.1224, 0.4422, respectively. Although the MC-AE+SC method based on our MC-AE model has better performance than five related contrastive methods, there are no significant difference between MC-AE+SC and them.

6.3. Computational Efforts

The results of computational efforts (CPU time) of our MC-AE+KM and MC-AE+SC methods on real-valued datasets and UCI datasets are listed in Table 12 and Table 13, respectively. The CPU times of MC-AE+KM algorithm consists of the training time of MC-AE model and clustering time of K-means algorithm. Similarly, the CPU times of MC-AE+SC algorithm consists of the training time of MC-AE model and clustering time of SC algorithm. It is obvious that the training time of MC-AE model occupies the most CPU times of MC-AE+KM and MC-AE+SC algorithms, especially on the high-dimensional datasets. The result was not unexpected because the more local collaborative blocks participate in training process of our MC-AE model on the high-dimensional datasets.
Figure 6: The results of Friedman + post-hoc Nemenyi tests among all contrastive algorithms on UCI datasets.

Figure 7: The performance comparisons of DAE, Deep-FS, fGraphDBN, CDL, VGAE and our MC-AE using average ACC, Jac and FMI metrics on the UCI datasets.
7. Conclusions

In this study, we developed a new Multi-local Collaborative AutoEncoder architecture, MC-AE, which is based on the proposed novel mcrRBM and mcrGRBM shallow models. The structure and multi-local collaborative relationships of unlabeled data are integrated into the encoding procedure of our MC-AE that force the multi-local hidden features to converge on the their centers of each local collaborative block. The proposed MC-AE is evaluated on ten real-valued datasets and ten UCI datasets with linear and binary visible layer units, respectively. Through extensive experiments, our MC-AE has consistently outperformed the existing related deep models. Furthermore, the proposed architecture showed more excellent generalization capability for different unsupervised clustering algorithms. In the future work, it is necessary to study multi-local collaborative representation learning for semi-supervised clustering, classification and computer vision.

8. Acknowledgement

This work was supported by the National Natural Science Foundation of China (No. 62176221, 71901158) and Sichuan Science and Technology Program (2021YFS0178).

9. Appendix

References

References

[1] T. Wang, J. Cao, X. Lai, Q. M. J. Wu, Hierarchical one-class classifier with within-class scatter-based autoencoders, IEEE Transactions on Neural Networks and Learning Systems (2020) 1–7doi:10.1109/TNNLS.2020.3015860.

[2] B. Tolooshams, S. Dey, D. Ba, Deep residual autoencoders for expectation maximization-inspired dictionary learning, IEEE Transactions on Neural Networks and Learning Systems (2020) 1–15doi:10.1109/TNNLS.2020.3005348.
[3] P. Ge, C. X. Ren, D. Q. Dai, J. Feng, S. Yan, Dual adversarial autoencoders for clustering, IEEE Transactions on Neural Networks and Learning Systems 31 (4) (2020) 1417–1424. doi:10.1109/TNNLS.2019.2919948.

[4] S. B. Banisetty, V. Rajamohan, F. Vega, D. Feil-Seifer, A deep learning approach to multi-context socially-aware navigation, CoRR abs/2104.10197. arXiv:2104.10197. URL https://arxiv.org/abs/2104.10197

[5] S. Yang, K. Yu, F. Cao, H. Wang, X. Wu, Dual-representation-based autoencoder for domain adaptation, IEEE Transactions on Cybernetics (2021) 1–14 doi:10.1109/TCYB.2020.3040763.

[6] M. Śmieja, M. Wolczyk, J. Tabor, B. C. Geiger, Segma: Semi-supervised gaussian mixture autoencoder, IEEE Transactions on Neural Networks and Learning Systems (2020) 1–12 doi:10.1109/TNNLS.2020.3016221.

[7] S. Karatsiolis, C. N. Schizas, Conditional generative denoising autoencoder, IEEE Transactions on Neural Networks and Learning Systems 31 (10) (2020) 4117–4129. doi:10.1109/TNNLS.2019.2952203.

[8] T. N.Kipf, M.Welling, Variational graph auto-encoders, in: NIPS Workshop on Bayesian Deep Learning, 2016.

[9] C. Xu, Y. Dai, R. Lin, S. Wang, Deep clustering by maximizing mutual information in variational auto-encoder, Knowledge-Based Systems 205 (2020) 106260.

[10] M. Pesteie, P. Abolmaesumi, R. N. Rohling, Adaptive augmentation of medical data using independently conditional variational auto-encoders, IEEE Transactions on Medical Imaging 38 (12) (2019) 2807–2820. doi:10.1109/TMI.2019.2914656.

[11] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[12] G. Hinton, T. Sejnowski, Learning and relearning in boltzmann machines, Parallel distributed processing: Explorations in the microstructure of cognition 1 (1986) 282–317.

[13] G. E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Computation 14 (8) (2002) 1771–1800.

[14] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images.

[15] J. Zhang, H. Wang, J. Chu, S. Huang, T. Li, Q. Zhao, Improved gaussian-bernoulli restricted boltzmann machine for learning discriminative representations, Knowledge-Based Systems 185 (Dec.1) (2019) 104911.1–104911.13.

[16] L. Bao, X. Sun, Y. Chen, D. Gong, Y. Zhang, Restricted boltzmann machine-driven interactive estimation of distribution algorithm for personalized search, Knowledge-Based Systems 200 (2020) 106030.

[17] J. Chu, H. Wang, H. Meng, P. Jin, T. Li, Restricted boltzmann machines with gaussian visible units guided by pairwise constraints, IEEE Transactions on Cybernetics 49 (2019) 4321–4334.

[18] J. Chu, H. Wang, J. Liu, Z. Gong, T. Li, Unsupervised feature learning architecture with multi-clustering integration rbm, IEEE Transactions on Knowledge and Data Engineering (2020) 1–1 doi:10.1109/TKDE.2020.3015959.

[19] J. Yu, G. Liu, Knowledge extraction and insertion to deep belief network for gearbox fault diagnosis - sciencedirect, Knowledge-Based Systems 197 (June.7) (2020) 105883.

[20] Hu, Junying, Zhang, Chunxia, Jiangshe, Ji, Nannan, A new regularized restricted boltzmann machine based on class preserving, Knowledge Based Systems 123 (May.1) (2017) 1–12.

[21] F. O. D. França, A hash-based co-clustering algorithm for categorical data, Expert Systems with Applications 64 (2016) 24–35.
[22] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2) (2009) 210–227.

[23] M. Fu, H. Qu, Z. Yi, L. Lu, Y. Liu, A novel deep learning-based collaborative filtering model for recommendation system, IEEE Transactions on Cybernetics 49 (3) (2019) 1084–1096.

[24] X. Zeng, Y. Lin, Y. He, L. Lv, X. Min, A. Rodríguez-Paton, Deep collaborative filtering for prediction of disease genes, IEEE/ACM Transactions on Computational Biology and Bioinformatics (2019) 1–1.

[25] K. H. Cho, T. Raiko, A. Ilin, Gaussian-bernoulli deep boltzmann machine, in: the 2013 International Joint Conference on Neural Networks, IEEE, 2013, pp. 1–7.

[26] J. Zhang, G. Tian, Y. Mu, W. Fan, Supervised deep learning with auxiliary networks, in: the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2014, pp. 353–361.

[27] Y. Sun, X. Wang, X. Tang, Hybrid deep learning for face verification, IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (10) (2016) 1997–2009.

[28] A. Taherkhani, G. Cosma, T. M. McGinnity, Deep-fs: A feature selection algorithm for deep boltzmann machines, Neurocomputing 322 (2018) 22–37.

[29] H. Wang, N. Wang, D. Yeung, Collaborative deep learning for recommender systems, in: SIGKDD, 2015, pp. 1235–1244.

[30] D. Chen, J. Lv, Z. Yi, Graph regularized restricted boltzmann machine, IEEE Transactions on Neural Networks and Learning Systems 29 (6) (2018) 2651–2659.

[31] N. Srivastava, R. R. Salakhutdinov, G. E. Hinton, Modeling documents with deep boltzmann machines, arXiv preprint arXiv:1309.6865.

[32] R. Salakhutdinov, G. E. Hinton, Deep boltzmann machines, in: International Conference on Artificial Intelligence and Statistics, 2009, pp. 448–455.
[33] R. Salakhutdinov, G. Hinton, An efficient learning procedure for deep boltzmann machines, Neural Computation 24 (8) (2012) 1967–2006.

[34] M. A. Carreira-Perpinan, G. E. Hinton, On contrastive divergence learning, in: Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, Citeseer, 2005, pp. 33–40.

[35] S. Har-Peled, P. Indyk, R. Motwani, Approximate nearest neighbor: Towards removing the curse of dimensionality, Theory of Computing 8 (14) (2012) 321–350. doi:10.4086/toc.2012.v008a014.

[36] A. Banerjee, S. Merugu, I. S. Dhillon, J. Ghosh, Clustering with bregman divergences, Journal of Machine Learning Research 6 (4) (2005) 1705–1749.

[37] H. Li, M. Wang, X.-S. Hua, Msra-mm 2.0: A large-scale web multimedia dataset, in: IEEE International Conference on Data Mining Workshops, IEEE, 2009, pp. 164–169.

[38] S. Lloyd, Least squares quantization in pcm, IEEE Transactions on Information Theory 28 (2) (1982) 129–137.

[39] A. Y. Ng, M. I. Jordan, Y. Weiss, et al., On spectral clustering: Analysis and an algorithm, Advances in Neural Information Processing Systems 2 (2002) 849–856.

[40] D. Cai, X. He, J. Han, Document clustering using locality preserving indexing, IEEE Transactions on Knowledge and Data Engineering 17 (12) (2005) 1624–1637.

[41] R. Liu, H. Wang, X. Yu, Shared-nearest-neighbor-based clustering by fast search and find of density peaks, Information Sciences 450 (2018) 200 – 226. doi:https://doi.org/10.1016/j.ins.2018.03.031.

[42] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence
and data mining: Experimental analysis of power, Information Sciences 180 (10) (2010) 2044–2064.

[43] C. Zhang, S. Ding, A stochastic configuration network based on chaotic sparrow search algorithm, Knowledge-Based Systems 220 (10) (2021) 106924.
Table 2: The results of the ACC between our MC-AE and the contrastive deep models on ten real-valued datasets. The best performance on each data set is bolded.

| Methods       | banner  | beret   | bugatti | building | vista   | vistawallpaper | voituretuning | water   | wing   | worldmap | ACC     |
|---------------|---------|---------|---------|----------|---------|----------------|---------------|---------|--------|----------|---------|
| DAE+KM        | 0.5105±0.0004 | 0.3893±0.0018 | 0.4161±0.0010 | 0.4501±0.0013 | 0.3842±0.0006 | 0.3780±0.0008 | 0.3788±0.0013 | 0.3482±0.0009 | 0.3680±0.0010 | 0.4834±0.0020 | 0.4107   |
| Deep-FS+KM    | 0.4364±0.0003 | 0.4234±0.0000 | 0.4305±0.0005 | 0.5159±0.0001 | 0.4251±0.0003 | 0.4251±0.0003 | 0.4331±0.0002 | 0.4140±0.0000 | 0.4151±0.0003 | 0.4731±0.0002 | 0.4394   |
| fGraphDBN+KM  | 0.4802±0.0148 | 0.4361±0.0010 | 0.3515±0.0853 | 0.5642±0.0011 | 0.4030±0.0311 | 0.4155±0.0103 | 0.4892±0.0014 | 0.4121±0.0002 | 0.4334±0.0027 | 0.5209±0.0091 | 0.4506   |
| CDL+KM        | 0.5267±0.0000 | 0.4304±0.0001 | 0.4390±0.0000 | 0.4918±0.0014 | 0.4030±0.0000 | 0.4030±0.0000 | 0.4020±0.0001 | 0.3955±0.0002 | 0.4065±0.0000 | 0.5077±0.0001 | 0.4406   |
| VGAE+KM       | 0.4905±0.0237 | 0.4410±0.0188 | 0.3875±0.0242 | 0.5463±0.0343 | 0.4750±0.0198 | 0.4707±0.0176 | 0.3985±0.0141 | 0.3830±0.0329 | 0.4311±0.0444 | 0.3969±0.0227 | 0.4429   |
| MC-AE+KM      | 0.9372±0.0006 | 0.6895±0.0002 | 0.4989±0.0003 | 0.7146±0.0000 | 0.6308±0.0005 | 0.4906±0.0002 | 0.6394±0.0009 | 0.5108±0.0004 | 0.6192±0.0053 | 0.6043±0.0030 | 0.6335   |
| DAE+SC        | 0.3570±0.0000 | 0.3642±0.0001 | 0.3639±0.0002 | 0.3649±0.0001 | 0.3567±0.0001 | 0.3717±0.0002 | 0.3481±0.0001 | 0.3449±0.0001 | 0.3540±0.0001 | 0.3455±0.0001 | 0.3553   |
| Deep-FS+SC    | 0.5093±0.0001 | 0.4001±0.0000 | 0.4947±0.0001 | 0.4812±0.0000 | 0.4927±0.0002 | 0.4927±0.0002 | 0.4391±0.0000 | 0.4638±0.0000 | 0.5456±0.0001 | 0.5672±0.0000 | 0.4886   |
| fGraphDBN+SC  | 0.4035±0.0013 | 0.4132±0.0012 | 0.3662±0.0021 | 0.5269±0.0010 | 0.4618±0.0063 | 0.4718±0.0014 | 0.5154±0.0021 | 0.4656±0.0019 | 0.4871±0.0024 | 0.4824±0.0103 | 0.4534   |
| CDL+SC        | 0.6031±0.0000 | 0.4063±0.0000 | 0.4769±0.0051 | 0.4877±0.0002 | 0.5424±0.0053 | 0.5924±0.0053 | 0.5631±0.0074 | 0.5201±0.0000 | 0.5668±0.0002 | 0.5455±0.0001 | 0.5304   |
| VGAE+SC       | 0.7672±0.0049 | 0.5696±0.0104 | 0.6604±0.0037 | 0.5950±0.0224 | 0.5111±0.0579 | 0.4631±0.0282 | 0.5354±0.0575 | 0.4086±0.0203 | 0.5550±0.0447 | 0.5388±0.0897 | 0.5604   |
| MC-AE+SC      | 0.9349±0.0089 | 0.4224±0.0112 | 0.7007±0.0089 | 0.7058±0.0026 | 0.5720±0.0068 | 0.6320±0.0054 | 0.6359±0.0025 | 0.5705±0.0044 | 0.6121±0.0046 | 0.7134±0.0081 | 0.6500   |
Table 3: The results of the Rank between our MC-AE and the contrastive deep models on ten real-valued datasets. The smaller the better.

| Methods       | banner     | beret      | bugatti    | building   | vista      | vistawallpaper | voilretuning | water      | wing       | worldmap   | Total  | Avg  |
|---------------|------------|------------|------------|------------|------------|----------------|--------------|------------|------------|------------|--------|------|
| DAE+KM        | -0.0692 (92) | -0.0597 (87) | -0.0494 (82) | -0.0854 (100) | -0.0873 (102) | -0.0892 (104) | -0.1027 (109) | -0.0832 (98) | -0.1148 (112) | -0.0315 (66) | 952    | 95.2 |
| Deep-FS+KM    | -0.1433 (116) | -0.0236 (62) | -0.0350 (68) | -0.0196 (60) | -0.0464 (76) | -0.0421 (72) | -0.0484 (78) | -0.0174 (57) | -0.0677 (89) | -0.0418 (71) | 749    | 74.9 |
| fGraphDBN+KM  | -0.0995 (107) | -0.0129 (56) | -0.1140 (110) | 0.0287 (37) | -0.0085 (90.5) | -0.0517 (83) | 0.0077 (45) | -0.0193 (59) | -0.0494 (81) | 0.0060 (46) | 714.5  | 71.45|
| CDL+KM        | -0.0530 (85) | -0.0186 (58) | -0.0265 (64) | -0.0437 (75) | -0.0685 (90.5) | -0.0642 (88) | -0.0795 (96) | -0.0359 (70) | -0.0763 (94) | -0.0072 (52) | 772.5  | 77.25|
| Vgae+KM       | -0.0892 (103) | -0.0080 (53) | -0.0780 (95) | 0.0108 (44) | 0.0035 (49) | 0.0035 (50) | -0.0830 (97) | -0.0484 (79) | -0.0517 (84) | -0.1180 (117) | 767    | 76.7 |
| MC-AE+KM      | 0.3575 (1) | 0.2405 (3) | 0.0334 (33) | 0.1791 (8) | 0.1593 (11) | 0.0234 (41) | 0.1579 (12) | 0.0794 (24) | 0.1364 (15) | 0.0894 (20) | 168    | 16.8 |
| DAE+SC        | -0.2227 (120) | -0.0848 (99) | -0.1016 (108) | -0.1886 (119) | -0.1148 (111) | -0.0955 (105) | -0.1334 (115) | -0.0865 (101) | -0.1288 (114) | -0.1694 (117) | 1109   | 110.9|
| Deep-FS+SC    | -0.0704 (93) | -0.0489 (80) | 0.0292 (36) | -0.0543 (86) | 0.0212 (42) | 0.0255 (38) | -0.0424 (73) | 0.0324 (34) | 0.0628 (27) | 0.0523 (30) | 539    | 53.9 |
| fGraphDBN+SC  | -0.1762 (118) | -0.0358 (69) | -0.0993 (106) | -0.0066 (54) | -0.0097 (55) | 0.0046 (47) | 0.0339 (32) | -0.0258 (63) | 0.0043 (48) | -0.0325 (67) | 659    | 65.9 |
| CDL+SC        | 0.0234 (40) | -0.0427 (74) | 0.0114 (43) | -0.0478 (77) | 0.0709 (26) | 0.1252 (17) | 0.0816 (23) | 0.0887 (21) | 0.0840 (22) | 0.0306 (35) | 378    | 37.8 |
| Vgae+SC       | 0.1875 (7) | 0.1206 (18) | 0.1949 (6) | 0.0595 (28) | 0.0396 (31) | -0.0041 (51) | 0.0559 (29) | -0.0228 (61) | 0.0722 (25) | 0.0239 (39) | 295    | 29.5 |
| MC-AE+SC      | 0.3552 (2) | -0.0266 (65) | 0.2352 (4) | 0.1703 (9) | 0.1005 (19) | 0.1648 (10) | 0.1544 (13) | 0.1391 (14) | 0.1293 (16) | 0.1985 (5) | 157    | 15.7 |
| Total         | 884 | 724 | 755 | 697 | 703 | 706 | 722 | 681 | 727 | 661 | 7260 |
Table 4: The results of the Jac between our MC-AE and the contrastive deep models on ten real-valued datasets. The best performance on each data set is bolded.

| Methods       | banner | beret | bugatti | building | vista | vista wallpaper | voituretuning | water | wing | worldmap | Jac  |
|---------------|--------|-------|---------|----------|-------|-----------------|---------------|-------|------|----------|------|
| DAE+KM        | 0.3720 | 0.2825| 0.2774  | 0.3070   | 0.2582| 0.2796          | 0.2515        | 0.2321| 0.2491| 0.3145   | 0.2824|
| Deep-FS+KM    | 0.3363 | 0.2755| 0.2792  | 0.3480   | 0.2780| 0.2780          | 0.2765        | 0.2495| 0.2727| 0.3181   | 0.2912|
| fGraphDBN+KM  | 0.3561 | 0.2900| 0.2614  | 0.3880   | 0.2708| 0.2677          | 0.3117        | 0.2706| 0.2744| 0.3490   | 0.3040|
| CDL+KM        | 0.3955 | 0.3082| 0.3090  | 0.3406   | 0.2751| 0.2751          | 0.2584        | 0.2473| 0.2663| 0.3555   | 0.3031|
| VGAE+KM       | 0.3613 | 0.2886| 0.2731  | 0.3397   | 0.2832| 0.2814          | 0.2589        | 0.2393| 0.2732| 0.2809   | 0.2880|
| MC-AE+KM      | **0.8820** | **0.5348** | 0.3194 | **0.5574** | **0.4738** | 0.2714 | **0.4760** | 0.3561 | **0.4714** | 0.4290 | **0.4771** |
| DAE+SC        | 0.3192 | 0.2633| 0.2606  | 0.2629   | 0.2435| 0.2432          | 0.2435        | 0.2321| 0.2430| 0.2647   | 0.2576|
| Deep-FS+SC    | 0.4672 | 0.3496| 0.3530  | 0.2717   | 0.3314| 0.3314          | 0.3236        | 0.3085| 0.3433| 0.3885   | 0.3468|
| fGraphDBN+SC  | 0.3441 | 0.2750| 0.2624  | 0.3502   | 0.3226| 0.3221          | 0.3432        | 0.2603| 0.3305| 0.3272   | 0.3138|
| CDL+SC        | 0.4906 | 0.3679| 0.2694  | 0.3657   | 0.3318| 0.3318          | 0.3271        | 0.4351| 0.4694| 0.3887   | 0.3777|
| VGAE+SC       | 0.6274 | 0.4076| 0.5046  | 0.4096   | 0.3759| 0.3464          | 0.3706        | 0.2589| 0.3893| 0.3924   | 0.4083|
| MC-AE+SC      | 0.8779 | 0.5328| 0.5420  | 0.5470   | 0.4120| **0.4736**      | 0.4709        | **0.4551** | 0.4598 | **0.5602** | 0.5311|
Table 5: The results of the FMI between our MC-AE and the contrastive deep models on ten real-valued datasets. The best performance on each data set is bolded.

| Methods       | banner | beret | bugatti | building | vistavista | vistawallpaper | voituretuning | water | wing | worldmap | Mean |
|---------------|--------|-------|---------|----------|------------|----------------|---------------|-------|------|-----------|------|
| DAE+KM        | 0.5876 | 0.4488| 0.4430  | 0.4765   | 0.4143     | 0.4380         | 0.4065        | 0.3802| 0.4038| 0.4849    | 0.4484|
| Deep-FS+KM    | 0.5577 | 0.4410| 0.4467  | 0.5223   | 0.4386     | 0.4386         | 0.4361        | 0.4016| 0.4330| 0.4915    | 0.4607|
| fGraphDBN+KM  | 0.5727 | 0.4558| 0.4261  | 0.5591   | 0.4280     | 0.4248         | 0.4755        | 0.4259| 0.4327| 0.5188    | 0.4719|
| CDL+KM        | 0.6070 | 0.4771| 0.4778  | 0.5123   | 0.4338     | 0.4338         | 0.4160        | 0.3986| 0.4256| 0.5266    | 0.4709|
| VAE+KM        | 0.5764 | 0.4550| 0.4382  | 0.5144   | 0.4442     | 0.4422         | 0.4156        | 0.3891| 0.4316| 0.4493    | 0.4556|
| MC-AE+KM      | 0.9392 | 0.7313| 0.4902  | 0.7466   | 0.6870     | 0.4321         | 0.6899        | 0.5344| 0.6866| 0.6043    | 0.6542|
| DAE+SC        | 0.5424 | 0.4285| 0.4259  | 0.4303   | 0.3979     | 0.3979         | 0.3979        | 0.3803| 0.3969| 0.4333    | 0.4231|
| Deep-FS+SC    | 0.6629 | 0.5184| 0.5223  | 0.4416   | 0.4980     | 0.4980         | 0.4890        | 0.4276| 0.5113| 0.5603    | 0.5129|
| fGraphDBN+SC  | 0.5633 | 0.4401| 0.4272  | 0.5202   | 0.4880     | 0.4874         | 0.5116        | 0.4133| 0.4972| 0.4967    | 0.4845|
| CDL+SC        | 0.6805 | 0.5380| 0.4345  | 0.5360   | 0.4984     | 0.4984         | 0.4931        | 0.4585| 0.6840| 0.5600    | 0.5381|
| VAE+SC        | 0.7756 | 0.5830| 0.6947  | 0.5831   | 0.5595     | 0.5207         | 0.5487        | 0.4119| 0.5714| 0.5638    | 0.5812|
| MC-AE+SC      | 0.9367 | 0.7288| 0.7279  | 0.7353   | 0.6028     | 0.6865         | 0.6808        | 0.6585| 0.6692| 0.7460    | 0.7172|
Table 6: The results of the ACC between our MC-AE and the contrastive deep models on ten UCI datasets. The best performance on each data set is bolded.

| Methods          | balance | biodegradation | car     | ClimateModel | dermatology | HabermanSurvival | Kdd     | OLD     | parkinsons | secom | ACC     |
|------------------|---------|----------------|---------|--------------|-------------|------------------|---------|---------|------------|-------|---------|
| DAE+KM           | 0.3904 ± 0.0000 | 0.6199 ± 0.0002 | 0.3414 ± 0.0000 | 0.5130 ± 0.0001 | 0.4372 ± 0.0010 | 0.6242 ± 0.0009 | 0.5523 ± 0.0000 | 0.9065 ± 0.0002 | 0.6359 ± 0.0002 | 0.7709 ± 0.0006 | 0.5792 ± 0.0000 |
| Deep-FS+KM       | 0.4768 ± 0.0017 | 0.6398 ± 0.0000 | 0.4199 ± 0.0000 | 0.5407 ± 0.0001 | 0.2377 ± 0.0000 | 0.6157 ± 0.0000 | 0.3875 ± 0.0000 | 0.7616 ± 0.0000 | 0.6051 ± 0.0000 | 0.7620 ± 0.0072 | 0.5447 ± 0.0000 |
| KGraphDBN+KM     | 0.4176 ± 0.0039 | 0.5768 ± 0.0007 | 0.3163 ± 0.0012 | 0.8444 ± 0.0943 | 0.3689 ± 0.0541 | 0.5686 ± 0.0139 | 0.3695 ± 0.0110 | 0.8346 ± 0.0028 | 0.6795 ± 0.0036 | 0.7625 ± 0.0221 | 0.5739 ± 0.0000 |
| CDL+KM           | 0.4208 ± 0.0000 | 0.5223 ± 0.0000 | 0.4132 ± 0.0000 | 0.5037 ± 0.0000 | 0.2705 ± 0.0002 | 0.5196 ± 0.0000 | 0.6164 ± 0.0204 | 0.9017 ± 0.0000 | 0.5641 ± 0.0040 | 0.8283 ± 0.0000 | 0.5561 ± 0.0000 |
| Vgae+KM          | 0.4886 ± 0.0993 | 0.6051 ± 0.0159 | 0.3510 ± 0.0432 | 0.5420 ± 0.0304 | 0.4019 ± 0.0257 | 0.6101 ± 0.1009 | 0.3650 ± 0.0030 | 0.9009 ± 0.0018 | 0.6764 ± 0.0936 | 0.6570 ± 0.0216 | 0.5598 ± 0.0000 |
| MC-AE+SC         | 0.6224 ± 0.0001 | 0.6673 ± 0.0004 | 0.4426 ± 0.0001 | 0.5500 ± 0.0004 | 0.4754 ± 0.0005 | 0.6275 ± 0.0012 | 0.5977 ± 0.0007 | 0.9369 ± 0.0002 | 0.8051 ± 0.0003 | 0.9336 ± 0.0003 | 0.7049 ± 0.0000 |
| DAE+SC           | 0.4640 ± 0.0001 | 0.6645 ± 0.0000 | 0.5434 ± 0.0000 | 0.5926 ± 0.0003 | 0.3552 ± 0.0007 | 0.7255 ± 0.0000 | 0.6414 ± 0.0002 | 0.7932 ± 0.0001 | 0.7641 ± 0.0000 | 0.8175 ± 0.0002 | 0.6361 ± 0.0000 |
| Deep-FS+SC       | 0.4584 ± 0.0000 | 0.6190 ± 0.0000 | 0.6325 ± 0.0022 | 0.8093 ± 0.0000 | 0.2814 ± 0.0006 | 0.7055 ± 0.0005 | 0.7211 ± 0.0009 | 0.8481 ± 0.0020 | 0.7487 ± 0.0000 | 0.5333 ± 0.0168 | 0.6357 ± 0.0000 |
| KGraphDBN+SC     | 0.4616 ± 0.0011 | 0.6062 ± 0.0023 | 0.5966 ± 0.1457 | 0.9120 ± 0.0013 | 0.3538 ± 0.0097 | 0.6977 ± 0.0023 | 0.3680 ± 0.0122 | 0.6434 ± 0.0036 | 0.6795 ± 0.0036 | 0.7618 ± 0.0573 | 0.6081 ± 0.0000 |
| CDL+SC           | 0.4524 ± 0.0000 | 0.6445 ± 0.0000 | 0.6485 ± 0.0000 | 0.5385 ± 0.0599 | 0.3087 ± 0.0000 | 0.6755 ± 0.0000 | 0.6586 ± 0.0000 | 0.8833 ± 0.0000 | 0.7341 ± 0.0000 | 0.5124 ± 0.0000 | 0.6037 ± 0.0000 |
| Vgae+SC          | 0.5232 ± 0.0634 | 0.6269 ± 0.0165 | 0.4064 ± 0.0659 | 0.6744 ± 0.1573 | 0.4557 ± 0.0643 | 0.6229 ± 0.0772 | 0.4155 ± 0.0427 | 0.9224 ± 0.0123 | 0.7118 ± 0.0620 | 0.8204 ± 0.0049 | 0.6180 ± 0.0000 |
| MC-AE+SC         | 0.4640 ± 0.0003 | 0.6645 ± 0.0002 | 0.6985 ± 0.0002 | 0.9111 ± 0.0011 | 0.4508 ± 0.0004 | 0.7255 ± 0.0007 | 0.8156 ± 0.0005 | 0.9013 ± 0.0001 | 0.7641 ± 0.0000 | 0.7128 ± 0.0004 | 0.7108 ± 0.0000 |
| Methods      | balance    | biodegradation | car   | ClimateModel | dermatology | HabermanSurvival | Kidd | OLD | parkinsons | secom | Total | Avg |
|-------------|------------|----------------|-------|---------------|-------------|------------------|------|-----|------------|------|-------|-----|
| DAE+KM      | -0.0796 (94) | -0.0015 (59) | -0.1428 (109) | -0.1463 (110) | 0.0708 (25) | -0.0190 (75) | -0.0226 (78) | 0.0537 (33) | -0.0065 (88) | 0.0315 (46) | 717   | 71.7 |
| Deep-FS+KM  | 0.0068 (56) | 0.0184 (53) | -0.0643 (89) | -0.1186 (103) | -0.1287 (105) | -0.0275 (80) | -0.1874 (114) | -0.0912 (97) | -0.0923 (98) | 0.0226 (50) | 845   | 84.5 |
| fGraphDBN+KM| -0.0524 (85) | -0.0446 (83) | -0.1679 (113) | 0.1851 (7) | 0.0025 (58.0) | -0.0746 (92) | -0.2054 (115) | -0.0182 (74) | -0.0179 (72.5) | 0.0231 (48) | 747.5 | 74.75 |
| CDL+KM      | -0.0492 (84) | -0.0991 (100) | -0.0710 (91) | -0.1556 (111) | -0.0959 (99.0) | -0.1236 (104) | 0.0415 (42) | 0.0489 (36) | -0.1333 (107) | 0.0889 (17) | 791   | 79.1 |
| VGAE+KM     | 0.0386 (52) | -0.0163 (70) | -0.1332 (106) | -0.1173 (102) | 0.0355 (44) | -0.0331 (81) | -0.2099 (119) | 0.0481 (38) | -0.0210 (77) | -0.0824 (95) | 784   | 78.4 |
| MC-AE+KM    | 0.1524 (9)  | 0.0459 (39) | -0.0416 (82) | -0.1093 (101) | 0.1090 (14) | -0.0157 (69) | 0.4128 (1)  | 0.0841 (19) | 0.1077 (15) | 0.1942 (6)  | 355   | 35.5 |
| DAE+SC      | -0.0060 (62.5) | 0.0431 (40.5) | 0.0592 (31) | -0.0667 (90) | -0.0112 (65) | 0.0823 (21.5) | 0.0665 (29) | 0.0596 (87) | 0.0667 (27.5) | 0.0781 (24) | 478   | 47.8 |
| Deep-FS+SC  | -0.0116 (66) | -0.0024 (60) | 0.1483 (11) | 0.1500 (10) | -0.0550 (96) | 0.0623 (30) | 0.1462 (12) | -0.0047 (61) | 0.0513 (35) | -0.2061 (116) | 497   | 49.7 |
| fGraphDBN+SC| -0.0084 (64) | -0.0152 (68) | 0.1124 (13) | 0.2527 (2) | -0.0126 (67) | 0.0545 (32) | -0.2069 (117) | -0.2094 (118) | -0.0179 (72.5) | 0.0224 (51) | 604.5 | 60.45 |
| CDL+SC      | -0.0176 (71) | 0.0231 (49) | 0.1643 (8) | -0.1408 (108) | -0.0577 (86) | 0.0323 (45) | 0.0837 (20) | 0.0305 (47) | 0.0367 (43) | -0.2270 (120) | 597   | 59.7 |
| VGAE+SC     | 0.0532 (34) | 0.0095 (57) | -0.0778 (93) | 0.0151 (54) | 0.0893 (16) | -0.0203 (76) | -0.1594 (112) | 0.0696 (26) | 0.0144 (55) | 0.0810 (23) | 546   | 54.6 |
| MC-AE+SC    | -0.0060 (62.5) | 0.0431 (40.5) | 0.2143 (5) | 0.2518 (3) | 0.0844 (18) | 0.0823 (21.5) | 0.2407 (4) | 0.0485 (37) | 0.0667 (27.5) | -0.0266 (79) | 298   | 29.8 |
| Total       | 740         | 719           | 751        | 801          | 693         | 727            | 763          | 673          | 718          | 675          | 7260  |      |

Table 7: The results of the Rank between our MC-AE and the contrastive deep models on ten UCI datasets. The smaller the better.
Table 8: The results of the Jac between our MC-AE and the contrastive deep models on ten UCI datasets. The best performance on each data set is bolded.

| Methods     | balance | biodegradation | car  | ClimateModel | dermatoLOGY | HabermanSurvival | Kdd  | OLD  | parkinsons | secom | Jac  |
|-------------|---------|----------------|------|--------------|-------------|------------------|------|------|------------|-------|------|
| DAE+KM      | 0.2683  | 0.3893         | 0.3050| 0.4573       | 0.2239      | **0.6881**       | 0.4266| 0.8215| 0.4035     | 0.5766| 0.4487|
| Deep-FS+KM  | 0.3036  | 0.5126         | 0.2957| 0.4582       | 0.1099      | 0.5316           | 0.3385| 0.6213| 0.4583     | 0.4926| 0.4122|
| fGraphDBN+KM| 0.2577  | 0.4353         | 0.2194| 0.7367       | 0.1883      | 0.4049           | 0.2597| 0.7180| 0.5232     | 0.6594| 0.4403|
| CDL+KM      | 0.2585  | 0.3916         | 0.2444| 0.4573       | 0.1192      | 0.3817           | 0.5040| 0.8215| 0.4134     | 0.7061| 0.4298|
| VGAE+KM     | 0.3097  | 0.3991         | 0.2329| 0.4633       | 0.1983      | 0.4382           | 0.3240| 0.8215| 0.4882     | 0.6056| 0.4281|
| MC-AE+KM    | 0.2488  | 0.3898         | 0.2143| 0.4601       | 0.3555      | 0.4324           | 0.9806| 0.8816| 0.4103     | 0.7634| 0.5137|

| Methods     | balance | biodegradation | car  | ClimateModel | dermatoLOGY | HabermanSurvival | Kdd  | OLD  | parkinsons | secom | Jac  |
|-------------|---------|----------------|------|--------------|-------------|------------------|------|------|------------|-------|------|
| DAE+SC      | 0.4285  | 0.5506         | 0.4204| 0.4719       | 0.2082      | 0.5983           | 0.5492| 0.4961| 0.6319     | 0.6383| 0.4993|
| Deep-FS+SC  | 0.4135  | 0.3715         | 0.4657| 0.8307       | 0.1748      | 0.5748           | 0.5104| 0.7387| 0.6108     | 0.7149| 0.5406|
| fGraphDBN+SC| 0.4267  | 0.4674         | 0.4480| **0.8392**   | 0.2054      | 0.5933           | 0.2852| 0.5222| 0.5232     | 0.6660| 0.4923|
| CDL+SC      | 0.4255  | 0.5495         | 0.5376| 0.4543       | 0.1942      | 0.5953           | 0.5576| 0.8724| 0.6289     | 0.4639| 0.5279|
| VGAE+SC     | 0.3203  | 0.4145         | 0.2663| 0.5799       | 0.2454      | 0.4543           | 0.3354| 0.8564| 0.5525     | 0.7524| 0.4777|
| MC-AE+SC    | 0.4285  | 0.5525         | 0.5408| 0.8370       | 0.1987      | 0.5983           | 0.5707| 0.7560| 0.6319     | 0.4875| 0.5602|
Table 9: The results of the FMI between our MC-AE and the contrastive deep models on ten UCI datasets. The best performance on each data set is bolded.

| Methods    | balance | biodegradation | car | ClimateModel | dermatology | HabermanSurvival | Kdd | OLD | parkinsons | secem | FMI    |
|------------|---------|----------------|-----|--------------|-------------|----------------|-----|-----|------------|-------|--------|
| DAE+KM     | 0.4263  | 0.5551         | 0.3743 | 0.6491     | 0.3659     | **0.7778**     | 0.6097 | 0.9023 | 0.5843     | 0.7419 | 0.5987 |
| Deep-FS+KM | 0.4570  | 0.6862         | 0.4587 | 0.6400     | 0.1882     | 0.6868         | 0.5718 | 0.7634 | 0.6188     | 0.6711 | **0.5742** |
| fGraphDBN+KM| 0.4103 | 0.6103         | 0.3826 | 0.8474     | 0.3244     | 0.5771         | 0.4229 | 0.8369 | 0.6907     | 0.7632 | **0.5866** |
| CDL+KM     | 0.4138  | 0.5632         | 0.4217 | 0.6490     | 0.2132     | 0.5547         | 0.6893 | 0.9023 | 0.5858     | 0.8293 | **0.5822** |
| VGAE+KM    | 0.4695  | 0.5710         | 0.3975 | 0.6538     | 0.3307     | 0.6085         | 0.5503 | 0.9023 | 0.6549     | 0.7264 | **0.5865** |
| MC-AE+KM   | 0.4010  | 0.5615         | 0.3798 | 0.6527     | **0.5249** | 0.6038         | **0.9853** | **0.9390** | 0.5839     | **0.8658** | **0.6498** |
| DAE+SC     | 0.6533  | 0.7431         | 0.5977 | 0.6626     | 0.4452     | 0.7699         | 0.7308 | 0.6864 | **0.7938** | 0.7844 | **0.6867** |
| Deep-FS+SC | 0.6323  | 0.5429         | 0.6366 | 0.9066     | 0.3675     | 0.7462         | 0.6769 | 0.8467 | 0.7761     | 0.8312 | **0.6963** |
| fGraphDBN+SC| 0.6500 | 0.6479         | 0.6295 | **0.9156** | 0.4068     | 0.7075         | 0.4742 | 0.7011 | 0.6907     | 0.7633 | **0.6587** |
| CDL+SC     | 0.6183  | 0.7079         | 0.6995 | 0.6143     | 0.3981     | 0.7349         | 0.7101 | 0.9002 | 0.7588     | 0.6268 | **0.6769** |
| VGAE+SC    | 0.4851  | 0.5894         | 0.4343 | 0.7360     | 0.3926     | 0.6254         | 0.5601 | 0.9236 | 0.7169     | 0.8213 | **0.6285** |
| MC-AE+SC   | **0.6533** | **0.7429**     | **0.7346** | **0.9140** | **0.4398** | **0.7699**     | **0.7282** | **0.8612** | **0.7938** | **0.6771** | **0.7315** |
Table 10: results of the K-means and SC algorithms on ten real-valued datasets.

| Dataset  | ACC         | Jac         | FMI         |
|----------|-------------|-------------|-------------|
|          | K-means     | SC          | K-means     | SC          |
| banner   | 0.4667 ± 0.0192 | 0.3713 ± 0.0158 | 0.3535 | 0.3251 | 0.5702 | 0.5470 |
| beret    | 0.4482 ± 0.0111 | 0.3893 ± 0.0060 | 0.3031 | 0.2631 | 0.4746 | 0.4282 |
| bugatti  | 0.4240 ± 0.0346 | 0.3813 ± 0.0114 | 0.2760 | 0.2643 | 0.4423 | 0.4302 |
| building | 0.5799 ± 0.0026 | 0.4724 ± 0.0023 | 0.3582 | 0.2917 | 0.5385 | 0.4661 |
| vista    | 0.4698 ± 0.0026 | 0.4693 ± 0.0185 | 0.2840 | 0.2802 | 0.4479 | 0.4436 |
| vistawallpaper | 0.4781 ± 0.0118 | 0.4610 ± 0.0158 | 0.2896 | 0.2807 | 0.4545 | 0.4442 |
| voituretuning | 0.4407 ± 0.0112 | 0.3561 ± 0.0039 | 0.2867 | 0.2444 | 0.4469 | 0.3990 |
| water    | 0.4071 ± 0.0014 | 0.4255 ± 0.0006 | 0.2385 | 0.2411 | 0.3883 | 0.3920 |
| wing     | 0.4677 ± 0.0039 | 0.3929 ± 0.0208 | 0.2800 | 0.2481 | 0.4398 | 0.4035 |
| worldmap | 0.4595 ± 0.0127 | 0.3872 ± 0.0047 | 0.3014 | 0.2760 | 0.4710 | 0.4456 |
| Average  | 0.4642       | 0.4106       | 0.2971       | 0.2715       | 0.4674       | 0.4399 |

Table 11: The results of the K-means and SC algorithms on ten UCI datasets.

| Dataset       | ACC         | Jac         | FMI         |
|---------------|-------------|-------------|-------------|
|               | K-means     | SC          | K-means     | SC          |
| balance       | 0.5221 ± 0.0544 | 0.5173 ± 0.0406 | 0.3747 | 0.2727 | 0.5492 | 0.4321 |
| biodgradation | 0.5886 ± 0.0000 | 0.6253 ± 0.0005 | 0.4179 | 0.5174 | 0.5906 | 0.7046 |
| car           | 0.4282 ± 0.0412 | 0.4088 ± 0.0724 | 0.2864 | 0.2716 | 0.4736 | 0.4573 |
| ClimateModel  | 0.5383 ± 0.0312 | 0.7827 ± 0.0011 | 0.4578 | 0.6464 | 0.6495 | 0.7869 |
| dermatology   | 0.2942 ± 0.0274 | 0.3087 ± 0.0047 | 0.1488 | 0.1370 | 0.2594 | 0.2412 |
| HabermanSurvival | 0.5131 ± 0.0113 | 0.5218 ± 0.0019 | 0.3776 | 0.3787 | 0.5509 | 0.5520 |
| Kdd           | 0.6622 ± 0.0005 | 0.6845 ± 0.0004 | 0.5644 | 0.5725 | 0.7490 | 0.7684 |
| OLD           | 0.9017 ± 0.0000 | 0.9056 ± 0.0001 | 0.8215 | 0.8253 | 0.9023 | 0.9031 |
| parkinsons    | 0.5436 ± 0.0000 | 0.5795 ± 0.0000 | 0.4152 | 0.4134 | 0.5872 | 0.5861 |
| secom         | 0.7571 ± 0.0004 | 0.5110 ± 0.0075 | 0.6157 | 0.4667 | 0.7688 | 0.6616 |
| Average       | 0.5749       | 0.5845       | 0.4480       | 0.4502       | 0.6080       | 0.6111 |
Table 12: The CPU times (second) of the MC-AE+KM and MC-AE+SC algorithms on ten real-valued datasets. The CPU times of MC-AE+KM algorithm consists of the training time of MC-AE model and clustering time of K-means algorithm. The CPU times of MC-AE+SC algorithm consists of the training time of MC-AE model and clustering time of SC algorithm.

| Dataset     | K-means | SC   | MC-AE | MC-AE+KM | MC-AE+SC |
|-------------|---------|------|-------|----------|----------|
| banner      | 2.4688  | 1.5938 | 7216.2500 | 7218.7188 | 7217.8438 |
| beret       | 2.0156  | 1.8594 | 7215.3750 | 7217.3906 | 7217.2344 |
| bugatti     | 2.8438  | 2.2500 | 7154.9531 | 7157.7969 | 7157.2031 |
| building    | 3.4688  | 1.4063 | 7481.1875 | 7484.6563 | 7482.5938 |
| vista       | 2.1875  | 1.2656 | 6745.1563 | 6747.3438 | 6746.4219 |
| vistawallpaper | 2.1875 | 1.4688 | 6718.3281 | 6720.5156 | 6719.7969 |
| voituretuning | 2.0625 | 2.0781 | 7235.9219 | 7237.9844 | 7238.0000 |
| water       | 3.0313  | 1.5469 | 7807.7969 | 7810.8281 | 7809.3348 |
| wing        | 2.3438  | 1.4844 | 7052.0000 | 7054.3438 | 7053.4844 |
| worldmap    | 2.1250  | 1.8750 | 7804.4063 | 7806.5313 | 7806.2813 |

Table 13: The CPU times (second) of the MC-AE+KM and MC-AE+SC algorithms on ten UCI datasets. The CPU times of MC-AE+KM algorithm consists of the training time of MC-AE model and clustering time of K-means algorithm. The CPU times of MC-AE+SC algorithm consists of the training time of MC-AE model and clustering time of SC algorithm.

| Dataset              | K-means | SC   | MC-AE | MC-AE+KM | MC-AE+SC |
|----------------------|---------|------|-------|----------|----------|
| balance              | 0.0469  | 0.5156 | 6.1719 | 6.2188   | 6.6875   |
| biodegradation       | 0.3594  | 0.5313 | 47.7500 | 48.1094  | 48.2813  |
| car                  | 0.3125  | 2.0156 | 17.7969 | 18.1094  | 19.8125  |
| ClimateModel         | 0.3750  | 0.8438 | 11.0625 | 11.4375  | 11.9063  |
| dermatology          | 0.2344  | 0.6094 | 12.4844 | 12.7188  | 13.0938  |
| HabermanSurvival     | 0.0321  | 0.3594 | 7.6250  | 7.6571   | 7.9844   |
| Kdd                  | 0.0625  | 0.0469 | 2.4688  | 2.5313   | 2.5156   |
| OLD                  | 1.2031  | 4.6875 | 213.8750 | 215.0781 | 218.5625 |
| parkinsons           | 0.6094  | 0.7813 | 51.3750 | 51.9844  | 52.1563  |
| secom                | 2.6406  | 2.8438 | 6417.4219 | 6420.0625 | 6420.2656 |

42