Multichannel Matrix Randomized Autoencoder

Shichen Zhang
Hangzhou Dianzi University
Tianlei Wang
Hangzhou Dianzi University
Jiuwen Cao (jwcao@hdu.edu.cn)
Hangzhou Dianzi University

Research Article

Keywords: Randomized Autoencoder, Matrix Representation, Matrix Neural Network, One-class classification.

Posted Date: September 23rd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2078071/v1

Multichannel Matrix Randomized Autoencoder

Shichen Zhang[a,b], Tianlei Wang[a,b], Jiuwen Cao[a,b,*]

[a]*Machine Learning and I-health International Cooperation Base of Zhejiang Province, Hangzhou Dianzi University, 310018, China.*
[b]*Artificial Intelligent Institute, Hangzhou Dianzi University, Zhejiang, 310018, China.*

Abstract

The existing randomized autoencoders (RAEs) are generally designed for vectorization data resulting in destroying the original structure information inevitably when dealing with multi-dimension data such as image and video. To address this issue, a one-side matrix randomized AE (OMRAE) is developed that takes the two-dimensional (2D) data as inputs directly by the linear mapping on one-side of inputs with matrix multiplication. For multichannel 2D (M2D) data, a multichannel OMRAE (OMMRAE) is proposed by training the output weights to rebuild each channel of inputs respectively. In this way, the structural information of each channel and the interaction between channels are explored. Then, a double-side structure using 2 OMMRAEs to simultaneously extracts the row and column structure information of M2D is developed. At last, a novel hierarchical matrix randomized neural networks is constructed for one-class classification (HMRNN-OC) where each layer passes information by bilinear mapping derived from DMMRAE. Experiments are conducted on 2 benchmark datasets for the effectiveness demonstration. Comparisons to several state-of-the-art AEs reveal that the proposed OMMRAE/DMMRAE can obtain better performance with a compact network size.

Keywords: Randomized Autoencoder, Matrix Representation, Matrix Neural Network, One-class classification.

1. Introduction

Recently, the randomized autoencoder (RAE) especially for the randomized neural network (RNN) [1–4] based autoencoder (RNN-AE) has attracted much attention due to its advantages of fast learning speed, ease of implementation and less human-intervention [5–16]. The RNN-AE can be tracked to [5] that uses random hidden-layer parameters without tuning and only trains the output weight for representation learning. Owing to that, RNN-AE showed superior performance to many other methods on generalization capacity and training speed. Subsequently, the ℓ_1-norm penalty based sparse RNN-AE (RNN-SAE) [6], the kernel RNN-AE (RNN-KAE) [7] using kernel function, the graph RNN-AE (GRNN-AE) [8, 9] by regularizing the graph Laplacian manifold, etc, were continually developed. The recent improvements of RAE focus on imposing constraints on the encoded features to obtain the desired feature distribution [12–14].

*Corresponding author.
Email addresses: zsc1023@qq.com (Shichen Zhang), tianlei.wang.cn@gmail.com (Tianlei Wang), jwcao@hdu.edu.cn (Jiuwen Cao)

However, the aforementioned RAEs are generally designed for one-dimensional (1D) vectors. The two-dimensional (2D) data (e.g., the grey-scale images and the time-frequency spectrograms) and the multichannel 2D (M2D) data (e.g., color images and videos) are more common in real-world applications. To cope with the conventional RAEs, the 2D/M2D data have to be transformed into 1D vector resulting in the inevitable destruction of the original structure information. Moreover, the direct transformation from the 2D/M2D data may lead to the curse of dimensionality dilemma. To address the aforementioned issue, a one-side matrix randomized AE (OMRAE) is developed that takes the two-dimensional (2D) data as inputs directly by the linear mapping on one-side of inputs with matrix multiplication. For multichannel 2D (M2D) data, a novel multichannel OMRAE (OMMRAE) is proposed by training the output weights to rebuild each channel of inputs respectively. Compared with the existing vectorization based RAEs, the proposed OMMRAE can reserve the structure information of each channel and meanwhile the interactions between channels are explored. The OMRAE and OMMRAE only use unilateral linear transformation on the 2D/M2D inputs and the encoded outputs are obtained by a linear transformation only on the one-hand side. Thus, a double-size multichannel MRAE (DMMRAE) by parallelly using two OMMRAEs on both side is further proposed to address the issue of OMMRAE that only performs feature learning with a unilateral projection scheme.

As discussed in [13, 14, 17, 18], the RAEs achieved encouraging performance on the one-class classification (OCC) applications, and even perform better than many deep learning based OCC algorithms [14]. Considering the successes of the RAEs on OCC, in this paper, a novel hierarchical matrix randomized neural networks is constructed for one-class classification (HMRNN-OC) where each layer passes information by bilinear mapping derived from DMMRAE. Experiments are conducted on 2 benchmark datasets for the effectiveness demonstration. Comparisons to several state-of-the-art AEs reveal that the proposed OMMRAE/DMMRAE can obtain better performance with a compact network size. The contributions of the paper are summarized as follows.

1. A novel OMMRAE is proposed for M2D data feature learning by multichannel interaction mechanism and thus the structural information of each channel and the interaction between channels are explored.
2. A DMMRAE by using two OMMRAEs parallelly on both side is further developed to address the issue of OMMRAE that only performs feature learning with a unilateral projection scheme.
3. A HMRNN-OC framework built in stacking DMMRAEs is developed for OCC, and the experimental results demonstrate the effectiveness of the proposed algorithms compared with several state-of-the-art AE algorithms and OCC algorithms on benchmark datasets.

2. The proposed OMMRAE and DMMRAE

2.1. OMRAE

For 2D data such as grey-scale images and time-frequency spectrograms, the OMRAE utilizes the rule of matrix multiplication to conduct linear mapping directly on the one-side of original 2D inputs. Give the 2D dataset $\left\{\mathbf{X}_i \in \mathbb{R}^{D_1 \times D_2}, i = 1, \cdots, N\right\}$ where D_1 and D_2 are the dimensions of the 2D data. OMRAE first randomly generates the input weight $\mathbf{W} \in \mathbb{R}^{L \times D_1}$ and bias $\mathbf{B} \in \mathbb{R}^{L \times D_2}$ where L is the number of the hidden neurons, and then the matrix projection is conducted on 2D input $\mathbf{X}_i$ and the hidden-layer output of i-th sample is derived as

$$\mathbf{H}_i = g(\mathbf{W}\mathbf{X}_i + \mathbf{B})^T, i = 1, \cdots, N, \tag{1}$$

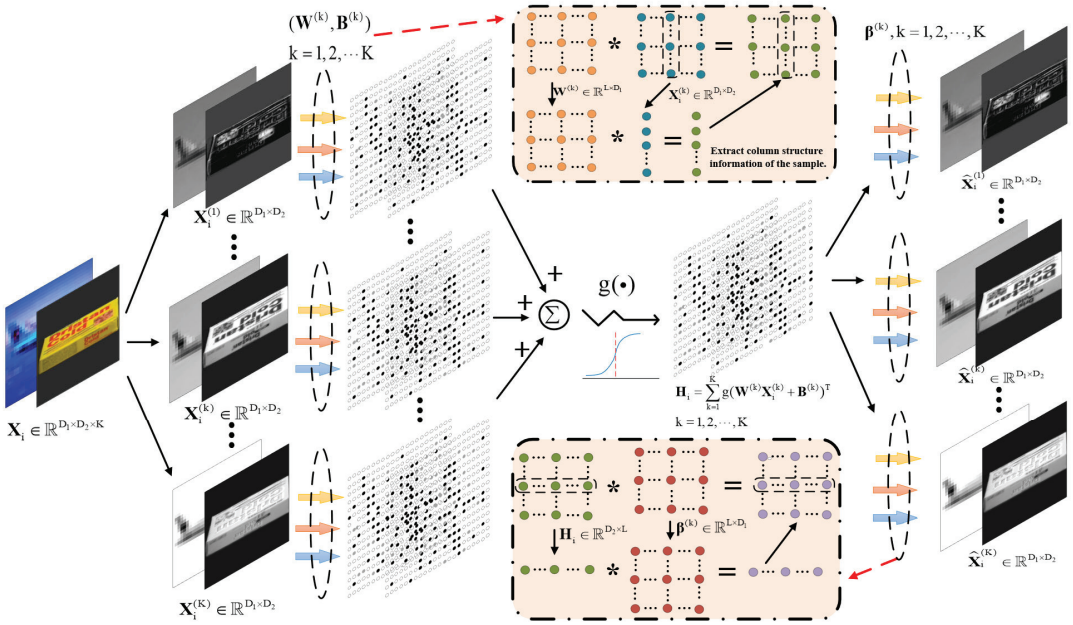


Figure 1: The architecture of the OMMRAE.

with $\mathbf{H}_i \in \mathbb{R}^{D_2\times L}$. The loss function of OMRAE is constructed as

$$\min_{\boldsymbol{\beta}} J(\boldsymbol{\beta}) = \frac{C}{2}\sum_{i=1}^{N}\left\|\mathbf{H}_i\boldsymbol{\beta} - \mathbf{X}_i^T\right\|_F^2 + \frac{1}{2}\|\boldsymbol{\beta}\|_F^2 . \tag{2}$$

Here $\boldsymbol{\beta} \in \mathbb{R}^{L\times D_1}$ is the output weight to be optimized, and it can be analytically derived by setting the derivative to 0 as

$$\boldsymbol{\beta} = \left(\sum_{i=1}^{N}\mathbf{H}_i^T\mathbf{H}_i + \frac{\mathbf{I}}{C}\right)^{-1}\sum_{i=1}^{N}\mathbf{H}_i^T\mathbf{X}_i^T . \tag{3}$$

The encoded output of OMRAE can be obtained by $\mathbf{Y}_i = \boldsymbol{\beta}\mathbf{X}_i$, $\mathbf{Y}_i \in \mathbb{R}^{L\times D_2}$. It can be readily seen from (1) that OMRAE only conducts unilateral mapping on the input and the encoded output is also obtained using the unilateral linear mapping. In this way, the column structure information is extracted.

2.2. OMMRAE

The effective feature representation can be obtained by OMRAE from original 2D data directly, but it fails to deal with M2D data. For M2D data, the OMRAE is extended to OMMRAE by rebuilding all channels simultaneously from hidden-layer outputs. Fig. 1 shows the detailed architecture of OMMRAE. Give the M2D dataset $\mathcal{X}_i \in \mathbb{R}^{D_1\times D_2\times K}, i = 1,\cdots,N$, where K is the number of channels and $\mathbf{X}_i^{(k)} \in \mathbb{R}^{D_1\times D_2}$ is the k-th channel. OMMRAE randomly generates the K input weights and biases $(\mathbf{W}^{(k)},\mathbf{B}^{(k)}), k = 1,\cdots,K$ for each channel, and the hidden-layer output of OMMRAE is obtained by

$$\widehat{\mathbf{H}}_i^{(k)} = g(\mathbf{W}^{(k)}\mathbf{X}_i^{(k)} + \mathbf{B}^{(k)})^T, i = 1,\cdots,N. \tag{4}$$

$$\mathbf{H}_i = \sum_{k=1}^{K} \hat{\mathbf{H}}_i^{(k)}, i = 1, \cdots, N. \tag{5}$$

It can be seen that the $\mathbf{H}_i$ integrates all information and the proposed OMMRAE tries to reconstruct all channels of the original M2D input from the integrated $\mathbf{H}_i$. It is believed that the obtained output weights can learn more intrinsic features from M2D data. Specially, the loss function can be expressed as

$$\min_{\boldsymbol{\beta}^{(k)}} J(\boldsymbol{\beta}^{(k)}) = \frac{C}{2} \sum_{k=1}^{K} \sum_{i=1}^{N} \left\| \mathbf{H}_i \boldsymbol{\beta}^{(k)} - \mathbf{X}_i^{(k)T} \right\|_F^2 + \sum_{k=1}^{K} \frac{1}{2} \left\| \boldsymbol{\beta}^{(k)} \right\|_F^2 \tag{6}$$

Similarly, the analytical weight can be obtained as

$$\boldsymbol{\beta}^{(k)} = (\sum_{i=1}^{N} \mathbf{H}_i^T \mathbf{H}_i + \frac{I}{C})^{-1} \sum_{i=1}^{N} \mathbf{H}_i^T \mathbf{X}_i^{(k)T} \tag{7}$$

The encoded output of i-th sample is $\mathbf{y}_i = \sum_{k=1}^{K} \boldsymbol{\beta}^{(k)} \mathbf{X}_i^{(k)} \in \mathbb{R}^{L \times D_2}$. As shown in (7), OMMRAE trains the output weight $\boldsymbol{\beta}^{(k)}$ by rebuilding each channel of M2D inputs, thus fully mining the structural information of each channel. Algorithm 1 summarizes the pseudo code of OMMRAE.

Algorithm 1 OMMRAE

Given:

$\mathcal{X}_i, (i = 1, \cdots, N)$, L, C, $\mathrm{g}(\bullet)$.

Steps:

1. Randomly generate $(\mathbf{W}^{(k)}, \mathbf{B}^{(k)})$,
2. Compute the hidden-layer output by (5),
3. Calculate $\sum_{i=1}^{N} \mathbf{H}_i^T \mathbf{H}_i$, $\sum_{i=1}^{N} \mathbf{H}_i^T \mathbf{x}_i^{(k)T}$,
4. Obtain the output weight $\boldsymbol{\beta}^{(k)}$ by (7),
5. Derive the encoded output $\mathbf{y}_i$.

2.3. DMMRAE

Both the OMRAE and OMMRAE are constructed only by a unilateral transformation, and the encoded outputs are also obtained by the unilateral linear mapping. In this way, OMMRAE performs feature learning on the column vectors of the M2D data but omits the correlation information among the row vectors. The disadvantages are arising as more hidden neurons are needed for feature representation, leading to a high computation complexity. To remedy this drawback, two OMMRAEs are conducted in parallel to perform feature learning on the row and column vectors simultaneously, resulting the DMMRAE. Algorithm 2 summarizes the pseudo code of DMMRAE. It is worthy pointing that the DMMRAE is actually an ensemble strategy training 2 OMMRAEs with inputs $\mathcal{X}_i \in \mathbb{R}^{D_1 \times D_2}$ and inputs $\hat{\mathcal{X}}_i \in \mathbb{R}^{D_2 \times D_1}$, where $\left\{ \hat{\mathcal{X}}_i | \hat{\mathcal{X}}_i^{(k)} = \mathbf{X}_i^{(k)T} \right\}$, respectively. The resulting output weights $\boldsymbol{\beta}_{(l)}^{(k)} \in \mathbb{R}^{L \times D_1}$ and $\boldsymbol{\beta}_{(r)}^{(k)} \in \mathbb{R}^{L \times D_2}$ are used to obtain the encoded

outputs by multiplying $\mathbf{X}_i$ left and right, respectively

$$\boldsymbol{y}_i = \sum_{k=1}^{K} \boldsymbol{\beta}_{(l)}^{(k)} \mathbf{X}_i^{(k)} \boldsymbol{\beta}_{(r)}^{(k)T} \in \mathbb{R}^{L \times L}. \tag{8}$$

From the above derivation, it can be found that the proposed OMMRAEs can effectively reduce the dimension of the output weight, which is critical to real implementations with a compact network size. For example, for a dataset $\mathbf{X} \in \mathbb{R}^{D_1 \times D_2}$, in feature learning and optimization, the dimensions of the output weight of OMMRAEs is only $L \times D_1$ or $L \times D_2$ while the dimensions of the output weight of the conventional vectorization based AE is up to $L \times D_1 \times D_2$.

Algorithm 2 DMMRAE

Given:

M2D data $\mathcal{X}_i$, the number of hidden-layer neurons L, the regularization parameter C, the activation function g(•).

Steps:

1. Transpose input data $\left\{\hat{\mathcal{X}}_i | \hat{\mathcal{X}}_i^{(k)} = \mathbf{X}_i^{(k)T}\right\}$ for M2D data.
2. Train the OMMRAE by Algorithm 1 with the inputs $\mathcal{X}_i$ and obtain the left-hand side encoded weight $\boldsymbol{\beta}_{(l)}^{(k)}$.
3. Train the OMMRAE by Algorithm 1 with the $\hat{\mathcal{X}}_i$ and obtain the right-hand side encoded weight $\boldsymbol{\beta}_{(r)}^{(k)}$.
4. Derive the encoded outputs by (8).

2.4. HMRNN-OC

The proposed algorithms are applied for OCC [14, 18, 19] for performance evaluation. The proposed DMMRAEs are embedded into the HMRNN-OC framework and the pseudo code of the resulting method is shown in Algorithm 3.

3. Experiments

Experiments on 2 common benchmark image datasets, COIL100 and CIFAR10, are conducted for effectiveness evaluation. Particularly, for CIFAR10, one category is chosen as the target and the rest are considered as outliers in OCC. For COIL100, we combined several classes with similar object shapes together to formulate the OCC problem. All categories will be traversed by the same rule in the experiments.

The specifications of all datasets are given below (samples are visualized in Fig. 2)

1. COIL100[1] is a color image database with 100 objects. Each object is captured in 72 different positions by rotating the object position. The size of each image is 128×128, and the dataset contains 5000 training and 2200 test samples. Due to the small number of samples in each class, we combined several classes with similar object shapes together to formulate the OCC problem. For example, we group round jars and medicine bottles together because they both look like cylinders.
2. CIFAR10[2] is an object recognition dataset including 50000 training samples and 10000

[1] https://www.kaggle.com/jessicali9530/coil100

[2] https://www.cs.toronto.edu/ kriz/cifar.html

Algorithm 3 HMRNN-OC with DMMRAE.

Given:

$\mathbf{X}_i, i = 1, \cdots, N$, stacked DMMRAEs number J, $\left(C_j, L_j\right), j = 1, \cdots, J$,

Training stage:

1. for $j = 1 : J$
   (a) Compute $\boldsymbol{\beta}_{(l)j}^{(k)}$ and $\boldsymbol{\beta}_{(r)j}^{(k)}$ by Algorithm (2).
   (b) Compute the hidden-layer output $\mathbf{y}_{ij}$ by

$$\mathbf{y}_{ij} = g(\sum_{k=1}^{K} \boldsymbol{\beta}_{(l)j}^{(k)} \mathbf{y}_{i(j-1)}^{(k)} \boldsymbol{\beta}_{(r)j}^{(k)T}), j = 1, 2, \ldots, J$$

2. Calculate the output weight $\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{C} + \hat{\mathbf{Y}}\hat{\mathbf{Y}}^T\right)^{-1} \hat{\mathbf{Y}}\mathbf{t}$, where $\hat{\mathbf{Y}} = [cs(\mathbf{y}_{1J}), \cdots, cs(\mathbf{y}_{NJ})]^T$ and $\mathbf{t} = [t, \cdots, t]^T \in \mathbb{R}^{N \times 1}$.
3. Derive training errors by $\varepsilon(\mathbf{X}_i) = \left|cs(\mathbf{y}_{iJ})^T \boldsymbol{\beta} - t\right|$.
4. Select the threshold η by rejecting a percentage of training samples as outliers.

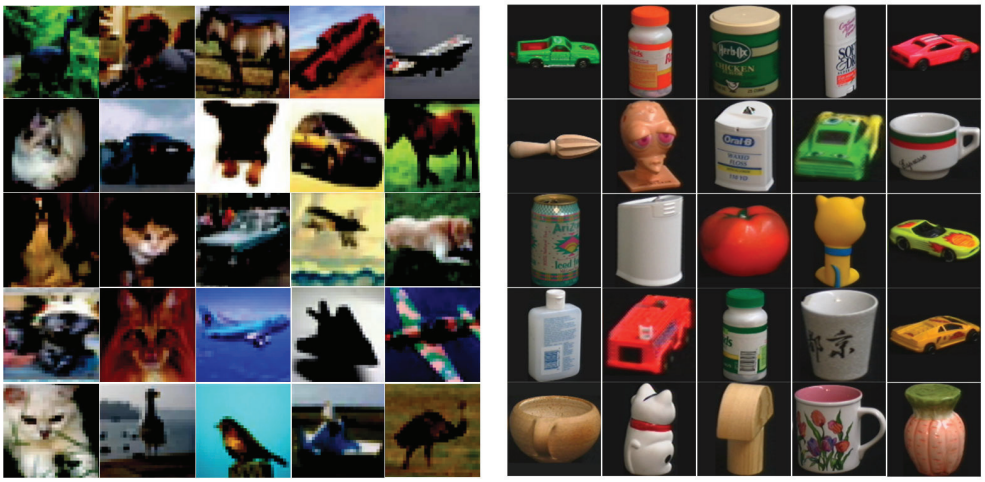
Testing stage:

A testing sample $\mathbf{X}_p$

1. for $j = 1 : J$,
   $\mathbf{y}_{pj} = g(\sum_{k=1}^{K} \boldsymbol{\beta}_{(l)j}^{(k)} \mathbf{X}_{p(j-1)}^{(k)} \boldsymbol{\beta}_{(r)j}^{(k)T})$.
2. Compute the output $O_p = cs(\mathbf{y}_{iJ})^T \boldsymbol{\beta}$
3. Derive $\varepsilon(\mathbf{X}_p) = \left|O_p - t\right|$ and perform the OCC

$$\begin{cases} \varepsilon(\mathbf{X}_p) \leq \eta & \rightarrow \quad \text{target} \\ \varepsilon(\mathbf{X}_p) > \eta & \rightarrow \quad \text{outlier} \end{cases}$$

testing samples from 4 vehicle and 6 animal classes, and each sample is 32×32 color images. consists of images from 10 classes. Out of the considered datasets, CIFAR10 is the most challenging dataset due to it diverse content and complexity. Specifically, it should be noted that all other datasets are very well aligned, without a background. In comparison, CIFAR10 is not an aligned dataset and it contains objects of the given class across very different settings. As a result, one-class novelty detection results for this dataset are comparatively weaker for all methods. Out of the baseline methods, [21] has done considerably better than other methods.



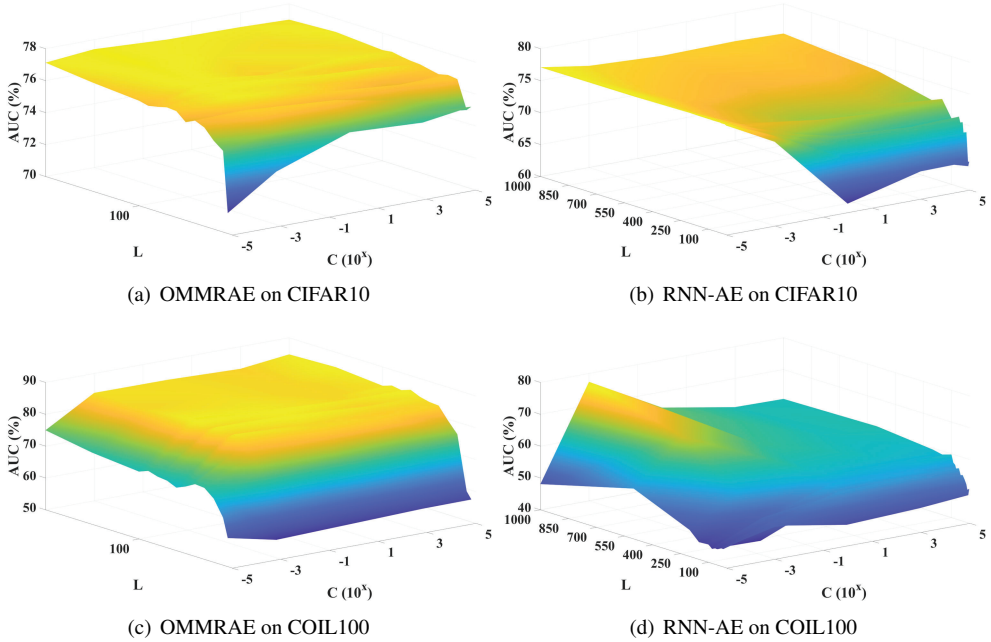
(a) CIFAR10 (b) COIL100

Figure 2: Visualization of benchmark datasets.

We compared the performances to several existed state-of-the-art (SOTA) AEs, namely, RNN-AE [5], sparse feature encoding based RNN (RNN-SAE) [6], random sparse matrix based AE (SMA) [20], and stacked convolutional AE (CoAE) [21]. For traditional scalar/vector AEs, the hidden nodes L and the regularized parameter C were optimized on the grid {100, 500, 700, 1000, } $\times$ $\{10^{-3}, 10^{0}, 10^{3}\}$. For our proposed AEs, the hidden nodes L and the regularized parameter C were optimized on the grid $\{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 120, 150, 200\} \times \left\{10^{-3}, 10^{0}, 10^{3}\right\}$. In each case, one AE was embedded in HLS-OC for feature learning. The regularized parameter in the last OCC layer of HLS-OC was optimized on the grid { 10^{-5}, 10^{-3}, 10^{0}, 10^{3}, 10^{5} }, and the threshold η was set to exclude the 10% of samples with the largest training error as outliers. The performance of each algorithm was quantified by AUC [22].

3.1. Sensitive analysis of hyper-parameters

Fig. 3 shows the parameter sensitivities of OMMRAE and RNN-AE on the number of hidden layer neurons and regularization parameters (L, C). As can be seen, the proposed MMAREs are generally less sensitivity to the number of hidden layer neurons and the regularization parameters than RNN-AE. For OMMRAE, the overall performance becomes stable and convincing when $L \geq 100$. We use the HLS-OC algorithm embedded with a MMRAE to conduct optimization respectively in the parameter grid established above.



(a) OMMRAE on CIFAR10 (b) RNN-AE on CIFAR10

(c) OMMRAE on COIL100 (d) RNN-AE on COIL100

Figure 3: Visualization of the parameters optimization on the COIL100 and CIFAR10 datasets, obtained by grid optimization in OMMRAE (left) and RNN-AE (right).

3.2. Compared with typical AE algorithms

Table 1 compares the AUC obtained by the 4 SOTA AEs between the proposed OMMRAE and DMMRAE algorithms. All methods have been applied to the HLS-OC framework with a single AE for feature learning. The best results are marked in bold in the table. As highlighted, the proposed matrix AEs can effectively improve the classification performance and have a high accuracy in many categories. Among all 15 testing cases, OMMRAE and DMMRAE wins the highest AUC on 10 cases, while RNN-SMA, RNN-AE, and RNN-SAE only offer the best results on 2, 1, and 2 cases, respectively. Besides, the comparison between OMMRAE and DMMRAE shows that DMMRAE performs overwhelmingly better than OMMRAE, as adopting double-side MMRAE wins the highest AUC on 9 cases.

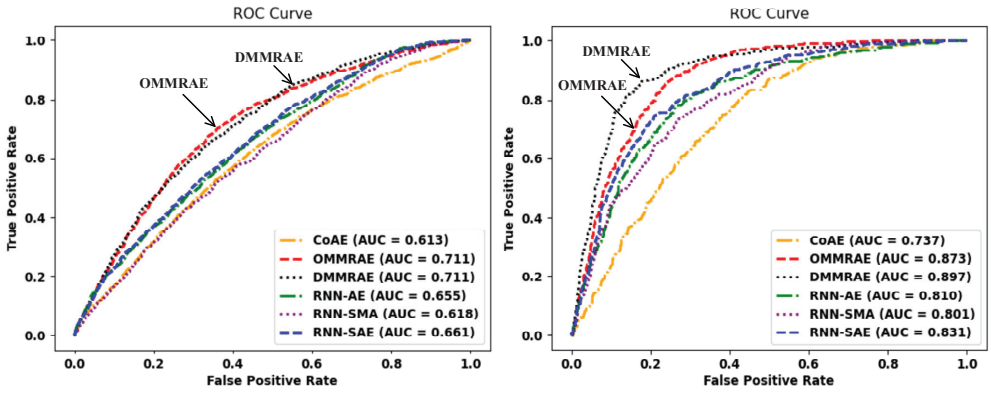
In Fig. 4, the ROC curves are also depicted to visually demonstrate the advantages of the proposed matrix AEs in feature learning. It is clearly observed that: 1) in Fig. 4 (a) and (b), our proposed MMRAEs generally achieve the best performance among all compared AEs, 2) among our proposed MMRAEs, adopting the double-side feature learning (namely DMMRAE) performs normally better than the single-side feature learning (namely OMMRAE).

Network complexity is an important factor for real implementation. Comparing with conventional vector/scalar AEs, a key merit of MMRAEs is the compact network structure to achieve a comparable or better performance than SOTA AEs. Fig. 5 shows the comparison of used number of hidden nodes to achieve the best performance for our proposed OMMRAE/DMMRAE and the comparesed SOTA AEs. The comparisons are presented on the experiments of 3 category data from CIFAR10 and COIL100, respectively. As clearly depicted, for both CIFAR10 and COIL100 datasets, to achieve the best performance, the needed hidden nodes by our proposed

OMMRAE/DMMRAE is generally far less than conventional AEs. In summary, MMRAEs not only improve the overall accuracy, but also effectively reduce the network size to guarantee a compact network structure.

Table 1: AUC (%) Comparisons with SOTA AE algorithms.

Dataset	CoAE [21]	RNN-AE [5]	RNN-SAE [6]	RNN-SMA [20]	OMMRAE	DMMRAE
CIFAR10-1	71.04 ± 2.58	73.64 ± 0.60.	75.18 ± 1.58	74.23 ± 0.27	**77.83 ± 1.14**	76.20 ± 0.54
CIFAR10-2	61.78 ± 3.86	65.13 ± 0.35	66.05 ± 0.63	63.35 ± 0.72	70.73 ± 2.55	**71.47 ± 0.21**
CIFAR10-3	52.13 ± 1.70	57.65 ± 0.65	59.31 ± 1.33	58.87 ± 1.20	57.89 ± 1.54	**60.20 ± 0.29**
CIFAR10-4	52.92 ± 2.62	59.03 ± 0.55	59.6 ± 0.27	**62.79 ± 0.49**	60.37 ± 0.13	61.95 ± 0.87
CIFAR10-5	52.26 ± 3.02	69.98 ± 0.56	68.27 ± 0.40	70.19 ± 0.39	69.90 ± 0.78	**70.56 ± 0.38**
CIFAR10-6	61.14 ± 4.57	61.22 ± 0.49	61.68 ± 0.14	**67.90 ± 1.20**	64.15 ± 3.00	64.21 ± 0.70
CIFAR10-7	53.61 ± 2.14	73.71 ± 0.25	72.17 ± 0.20	74.38 ± 0.51	75.13 ± 2.11	**76.09 ± 0.16**
CIFAR10-8	55.06 ± 0.75	60.72 ± 0.45	61.76 ± 0.58	62.72 ± 0.53	64.09 ± 0.94	**67.36 ± 0.29**
CIFAR10-9	71.09 ± 5.09	**78.67 ± 0.29**	77.89 ± 0.17	78.37 ± 0.75	77.94 ± 0.27	64.01 ± 0.65
CIFAR10-10	61.79 ± 5.30	75.79 ± 0.05	77.38 ± 0.12	75.2 ± 0.70	77.42 ±0.14	**77.85 ± 0.44**
Average	59.28 ± 3.16	67.55 ± 0.42	67.93 ± 0.54	68.80 ± 0.68	**69.55 ± 1.26**	69.19 ± 0.43
COIL100-1	78.05 ± 3.20	77.37 ± 1.39	81.51 ± 1.72	79.90 ± 1.38	86.26 ± 0.48	**88.70 ± 2.37**
COIL100-2	80.72 ± 7.60	98.17 ± 0.18	**98.19 ± 0.73**	97.97 ± 0.69	97.61 ± 0.28	97.81 ± 0.47
COIL100-3	86.12 ± 4.82	92.25 ± 1.23	94.26 ± 1.09	92.03 ± 1.20	94.05 ± 0.78	**95.08 ± 0.45**
COIL100-4	79.76 ± 5.01	84.97 ± 1.96	85.23 ± 1.61	82.58 ± 2.47	87.75 ± 2.28	**89.99 ± 1.53**
COIL100-5	75.21 ± 9.51	88.71 ± 1.59	**92.71 ± 1.07**	92.05 ± 1.82	89.54 ± 2.65	91.17 ± 1.38
Average	79.97 ± 6.03	88.29 ± 1.27	90.38 ± 1.24	88.91 ± 1.51	91.04 ± 1.29	**92.55 ± 1.24**

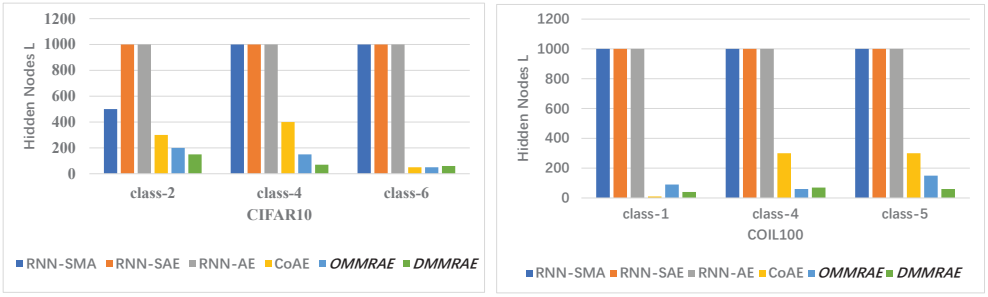


(a) Comparison in CIFAR10 (b) Comparison in COIL100

Figure 4: ROC Comparisons with SOTA AEs.

4. Conclusions

Unlike the conventional scalar/vector based autoencoder that performs feature learning on matrix or high-dimensional data generally relying on concatenating data by paying the price of breaking the structural information, the proposed multichannel matrix randomized autoencoder (MMRAE) can effectively exploiting the structural information and reduce the number of parameters in feature learning. The further extended double-side MMRAE is flexible learning both



(a) Hidden nodes in CIFAR10 (b) Hidden nodes in COIL100

Figure 5: Hidden nodes comparisons with SOTA AE algorithms to achieve the best performance.

the column and row structure information in matrix data. Experiments on 2 benchmark datasets show that the proposed MMRAEs are not only effective in classification accuracy but also have a compact network size. The future work will focus on constrained modeling based MMRAEs by exploiting the feature correlation within the same class as well as cross-classes..

Acknowledgment

This work was supported by the National Key Research and Development Program of China (2021YFE0205400, 2021YFE0100100), the Fundamental Research Funds for the Provincial Universities of Zhejiang (GK219909299001-017), the Zhejiang Provincial Natural Science Foundation (LQ22F030006, LZ22F030002), the National Natural Science Foundation of China (U1909209), and the Key Research and Development Program of Zhejiang Province (2020C03038).

References

[1] W. F. Schmidt, M. A. Kraaijveld, R. P. Duin, et al., Feed forward neural networks with random weights, in: Proceedings of the International Conference on Pattern Recognition Methodology and Systems, Vol. 2, 1992, pp. 1–4.
[2] G.-B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (2) (2012) 513–529.
[3] M. Eshaghnezhad, S. Effati, F. Rahbarnia, Projection recurrent neural network model: A new strategy to solve maximum flow problem, IEEE Transactions on Circuits and Systems II: Express Briefs 67 (11) (2020) 2747–2751. doi:10.1109/TCSII.2020.2977862.
[4] H. Tang, P. Dong, Y. Shi, A construction of robust representations for small data sets using broad learning system, IEEE Transactions on Systems, Man, and Cybernetics: Systems 51 (10) (2021) 6074–6084. doi:10.1109/TSMC.2019.2957818.
[5] L. L. C. Kasun, H. Zhou, G.-B. Huang, C. M. Vong, Representational Learning with Extreme Learning Machine for Big Data, IEEE Intelligent Systems 28 (6) (2013) 31–34.
[6] J. Tang, C. Deng, G. B. Huang, Extreme Learning Machine for Multilayer Perceptron, IEEE Transactions on Neural Networks and Learning Systems 27 (4) (2016) 809–821.
[7] C. M. Wong, C. M. Vong, P. K. Wong, J. Cao, Kernel-Based Multilayer Extreme Learning Machines for Representation Learning, IEEE Transactions on Neural Networks and Learning Systems 29 (3) (2018) 757–762.
[8] K. Sun, J. Zhang, C. Zhang, J. Hu, Generalized extreme learning machine autoencoder and a new deep neural network, Neurocomputing 230 (2017) 374–381.
[9] H. Ge, W. Sun, M. Zhao, Y. Yao, Stacked Denoising Extreme Learning Machine Autoencoder Based on Graph Embedding for Feature Representation, IEEE Access 7 (2019) 13433–13444.
[10] L. Yang, S. Song, S. Li, Y. Chen, G. Huang, Graph Embedding-Based Dimension Reduction With Extreme Learning Machine, IEEE Transactions on Systems, Man, and Cybernetics: Systems 51 (7) (2021) 4262–4273.

[11] R. Ma, T. Wang, J. Cao, F. Dong, Minimum error entropy criterion-based randomised autoencoder, Cognitive Computation and Systems 3 (4) (2021) 332–341.
[12] T. Wang, J. Cao, X. Lai, Q. M. Wu, Within-Class Scatter Constraint Based Randomized Autoencoder for One-Class Classification, in: Proceedings of 2nd China Symposium on Cognitive Computing and Hybrid Intelligence, CCHI 2019, 2019, pp. 111–116.
[13] J. Du, C. M. Vong, C. Chen, P. Liu, Z. Liu, Supervised Extreme Learning Machine-Based Auto-Encoder for Discriminative Feature Learning, IEEE Access 8 (2020) 11700–11709.
[14] T. Wang, J. Cao, X. Lai, Q. M. Wu, Hierarchical One-Class Classifier with Within-Class Scatter-Based Autoencoders, IEEE Transactions on Neural Networks and Learning Systems 32 (8) (2021) 3770–3776.
[15] A. Iosifidis, V. Mygdalis, A. Tefas, I. Pitas, One-class classification based on extreme learning and geometric class information, Neural Processing Letters 45 (2) (2017) 577–592.
[16] H. Yıldırım, M. R. Özkale, An enhanced extreme learning machine based on liu regression, Neural Processing Letters 52 (1) (2020) 421–442.
[17] L. Yang, S. Song, S. Li, Y. Chen, G. Huang, Graph embedding-based dimension reduction with extreme learning machine, IEEE Transactions on Systems, Man, and Cybernetics: Systems (2019).
[18] J. Cao, H. Dai, B. Lei, C. Yin, H. Zeng, A. Kummert, Maximum Correntropy Criterion-Based Hierarchical One-Class Classification, IEEE Transactions on Neural Networks and Learning Systems 32 (8) (2021) 3748–3754.
[19] H. Dai, J. Cao, T. Wang, M. Deng, Z. Yang, Multilayer one-class extreme learning machine, Neural Networks 115 (2019) 11–22.
[20] T. Wang, X. Lai, J. Cao, C.-M. Vong, B. Chen, An enhanced hierarchical extreme learning machine with random sparse matrix based autoencoder, in: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019, pp. 3817–3821.
[21] J. Masci, U. Meier, D. Cireşan, J. Schmidhuber, Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction, in: Proceedings of International Conference on Artificial Neural Networks, 2011, pp. 52–59.
[22] S. J. Mason, N. E. Graham, Areas beneath the relative operating characteristics (roc) and relative operating levels (rol) curves: Statistical significance and interpretation, Quarterly Journal of the Royal Meteorological Society 128 (584) (2002) 2145–2166.