[[1]](#footnote-1)

Neighboring Envelope Embedded Stacked Autoencoder for Deep Learning on Hierarchically Structured Samples

Chuanyan Zhou1, Jie Ma1, Fan Li, Yongming Li1\*, *Member*, *IEEE*, Pin Wang1

*Abstract*—A stacked autoencoder (SAE) is a widely used deep network. However, existing deep SAEs focus on original samples without considering the hierarchical structural information between samples. This limits the accuracy of the SAE. In recent years, state-of-the-art SAEs have suggested improvements in network structure, cost function, parameter optimization, and thereby the accuracy has been enhanced. However, the problem mentioned above is still not solved. Therefore, this paper is concerned with how to design a SAE that can conduct deep learning on hierarchically structured samples. This proposed SAE - neighboring envelope embedded stacked autoencoder (NE\_ESAE) mainly consists of two parts. The first is the neighboring sample envelope learning mechanism (NSELM) that constructs sample-pairs by combining neighboring samples. In addition, the NSELM constructs multilayer sample spaces by multilayer iterative mean clustering, which considers similar samples and generates layers of envelope samples with hierarchical structural information. The second is an embedded stacked autoencoder (ESAE) to consider the original samples during training and in network structure, thereby finding the relationship of the samples with original features and deep features in a better manner. The experimental results show that our method has significantly better performance than some representative methods. Different from existing SAEs, the proposed NE\_ESAE realizes deep learning on hierarchical structured samples, and makes SAE able to conduct cooperative deep sample and feature learning. The advantage that has been gained can be applied to other deep neural networks.

*Impact Statement* — Existing deep SAEs do not consider the relationship between neighboring samples, similar samples, and the relationship between samples with deep features and original features (we call them hierarchical structural information). This limitation makes SAEs only obtain deep features for every original sample (we call this ‘flat’ deep learning). With an improvement of 2.5% - 34% over the existing representative SAEs, the proposed NE\_ESAE can realize deep learning on hierarchical structured samples, thereby obtaining deep features for different layers of structured samples and with more complementarity with the original features. In addition, the proposed NE\_ESAE can realize cooperative deep sample and feature transformation. More importantly, this proposed model can be considered to be a framework rather than concrete model, in that the innovations achieved in this framework can be applied to other deep neural networks and transform their current work patterns.

*Index Terms*—Deep learning, Envelope learning, Feature learning, Stacked autoencoder.

# INTRODUCTION

S

AE, as an widely used deep learning network, can automatically learn effective features from a large amount of unlabeled data and is widely used. To date, various revised SAEs have also been successively proposed by researchers. Rui Li *et al*. [1] proposed a supervised autoencoder, by adding an additional classification layer on top of the representation layer to jointly predict the target and reconstruct the input. To solve the problems of finding feature representations that minimize the distance between source and target domains and avoiding data redundancy that may lead to the performance degradation of transfer learning, Yi Zhu *et al*. [2] proposed a new SAE framework for semisupervised representations for transfer learning. To solve the problem of imbalanced learning, Nima Farajian *et al*. [3] proposed an SAE-based imbalanced learning method that includes feature learning and classification steps. Ahmad M. Karim[4] integrated the postprocessing procedure into the data classification framework and proposed a new data classification framework that combines a postprocessing system composed of sparse SAEs and a linear system model based on particle swarm optimization (PSO). Wenjuan Wang *et al*. [5] proposed an efficient stacked contractive autoencoder (SCAE) method for unsupervised feature extraction in view of the large-scale, high-dimensional, and highly redundant network traffic characteristics in cloud computing environments. Weifeng Liu *et al*. [6] proposed a large-capacity autoencoder (LMAE) to further improve the discriminability by enforcing a large marginal distribution of samples of different classes in the hidden feature space. Wang Y *et al*. [7] proposed a cascaded supervised autoencoder to pretrain deep networks and obtain deep fault-related features from raw input data.

# motivation and contribution

Although SAEs have achieved great success in many applications [1]-[9], it is still a challenging problem to design a deep autoencoder for effective feature learning [10]. Existing SAEs mainly consider minimizing the error between each input sample and its output with reconstructed deep features, but the structural information between samples is ignored. As shown in Fig. 1, the existing autoencoder does not consider the relationship (structure information) between the ith sample and the jth sample. However, this will lead to a decrease in separability between samples, limit the search for optimal samples, and limit the quality of optimal features, thereby affecting the classification performance of the algorithm [11]. Therefore, it is necessary to consider the structural information among samples for SAE.

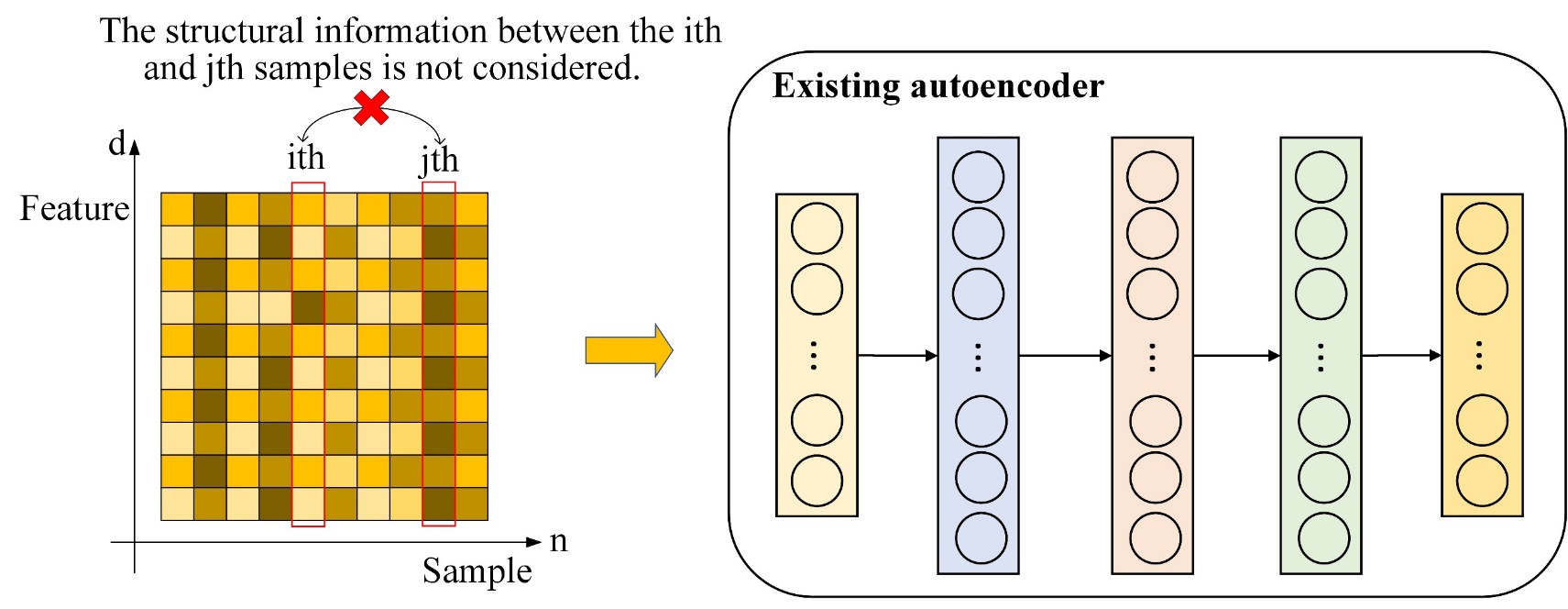
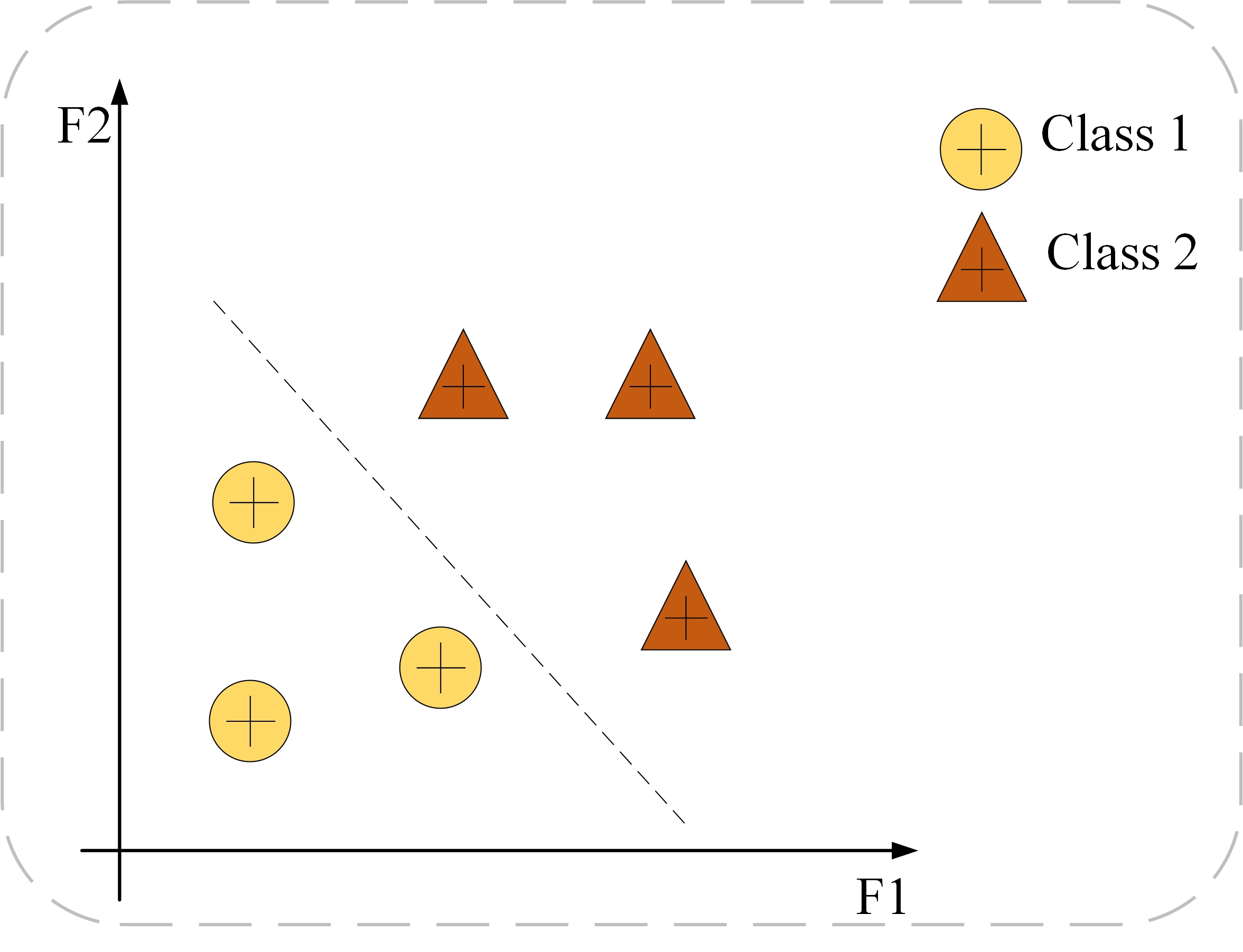
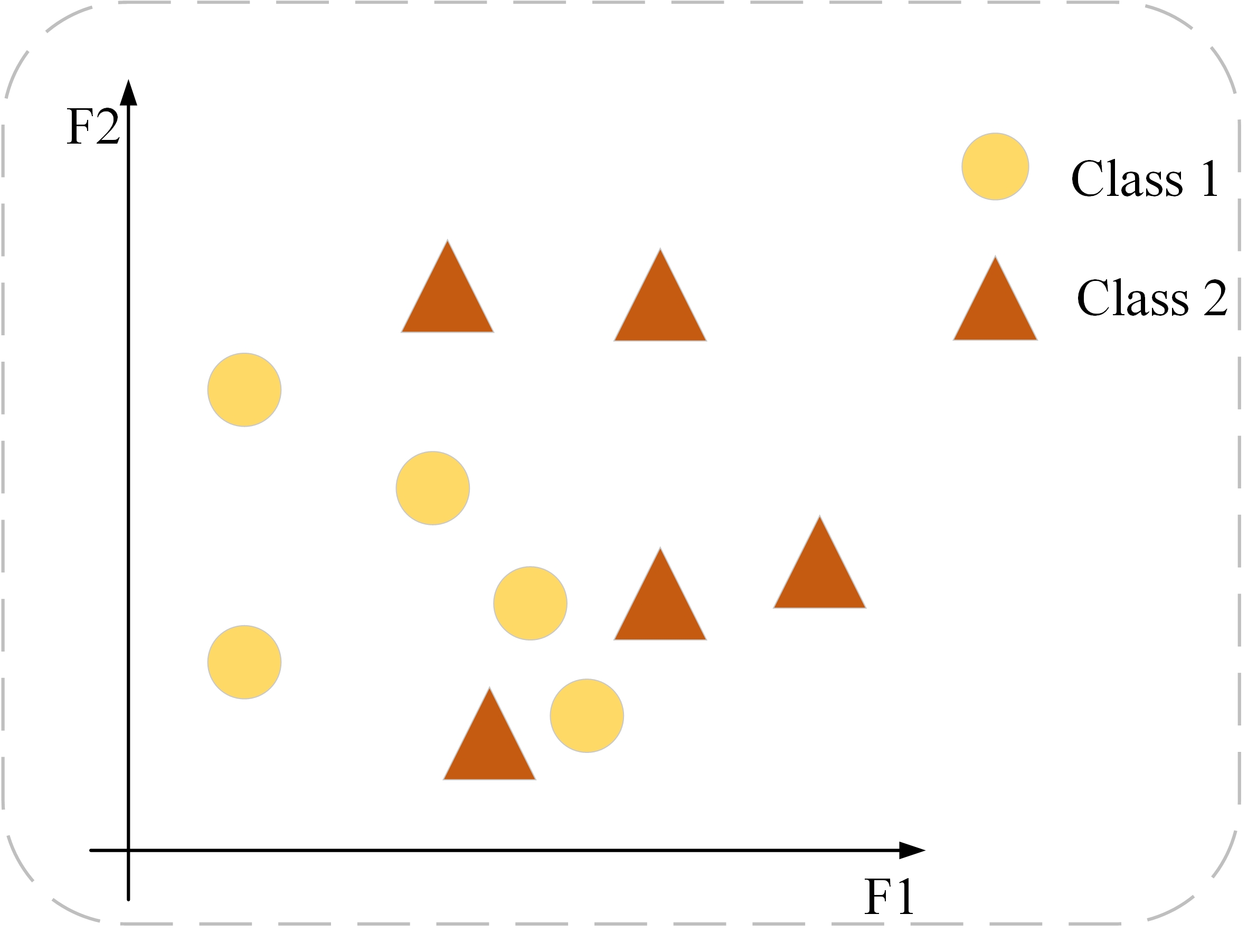


Fig. 1. The existing autoencoder model.

It is challenging to find the optimal features for the original samples. Fig. 2 shows the distribution of the original samples. Fig. 2 (a) shows the distribution of the original samples under feature spaces F1 and F2. As shown in Fig. 2 (a), it is difficult to achieve linear separability and to find a reliable optimal feature set. Fig. 2 (b) shows the distribution of the clustering-based transformed samples under F1 and F2. As shown in Fig. 2 (b), the separability is significantly improved by sample transformation under the same features. Sample transformation is helpful to mine the structural information between samples and construct ‘bigger’ samples with better separability. This means that the transformed samples should be considered for deep learning, apart from the original samples.



(a) Before sample transformation. (b) After sample transformation.

Fig. 2. Separability before and after sample transformation.

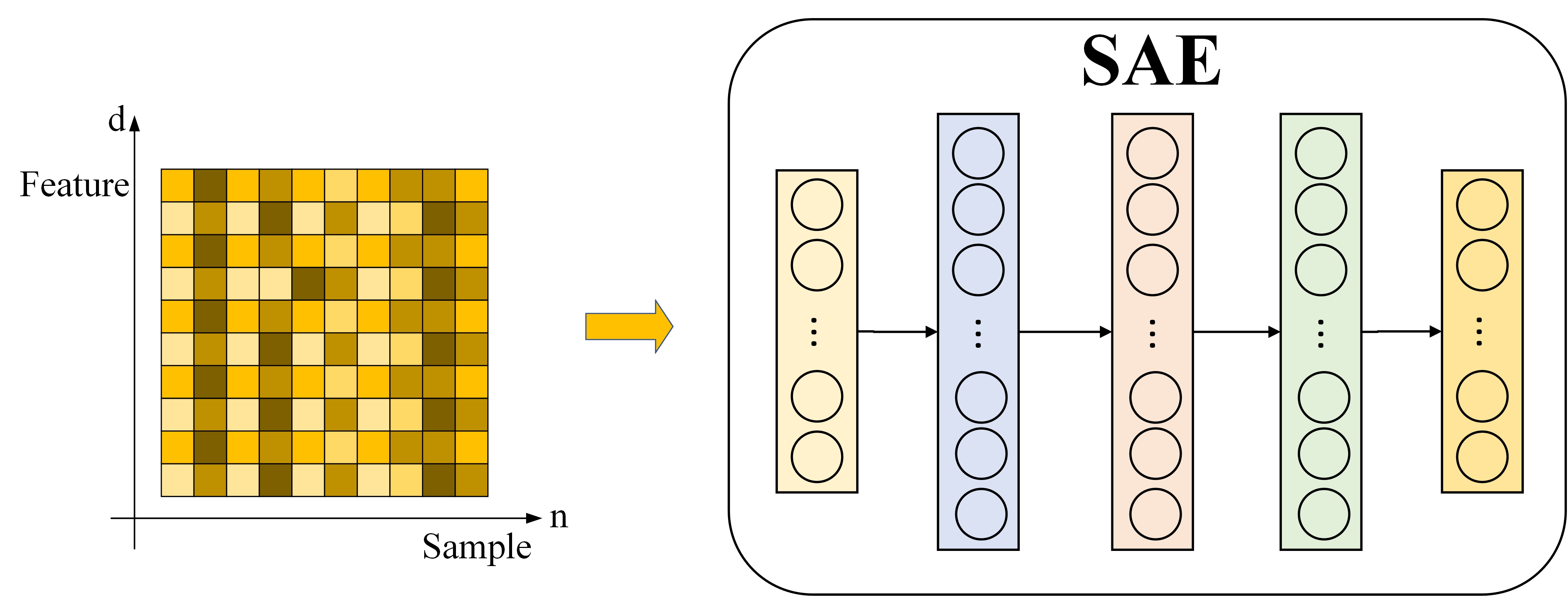
The existing sample transformation methods mainly include the nearest neighbor rule based [12], the density based [13], the random sampling based [14] and the clustering based [15]. Clustering-based methods can separate a finite unlabeled dataset into a finite and discrete set of data structures [16], analyze the internal relations of the data, and gather similar samples together, thus reducing the difficulty of classification in the face of confusing boundaries [17]. Clustering can be considered to mine sample structure information and takes the generated cluster center as a new sample. Existing clustering methods mainly include the k-means algorithm [18], hierarchical clustering [19], density-based clustering [20], grid-based clustering[21], self-organizing map (SOM) [22], transformed subspace clustering [23], transformed locally linear manifold clustering [24], Kernelized transformed subspace clustering with geometric weights for non-linear manifolds [25], etc. Among them, k-means clustering method has the advantages of simple, easy to implement, fast convergence, good clustering effect, strong interpretation and so on. Therefore, we consider using k-means clustering to construct multilayer sample spaces to obtain effective structural information and mine the hierarchical structure of sample space. However, there will be differences in the distribution of samples before and after clustering [26]. Hence we introduce the maximum mean difference (MMD) [27] to reduce the distribution difference of samples before and after clustering. The simple MMD is not sufficient and needs to be combined with the local and global distribution difference. The measured distribution differences can be used to impose constraints on the training model, and the model can be optimized [28].

We propose an algorithm to construct multilayer sample transformation, and then combine MMD to achieve consistency between sample layers. The whole algorithm is called neighboring sample envelope learning mechanism (NSELM). In addition, in order to improve the complementarity between the original feature and the deep feature of the sample, a new embedded stacked autoencoder (ESAE) is designed here. We combine the NSELM and MSEM and train several base classifiers on the sample layer, a multilayer space ensemble mechanism (MSEM) is designed to fuse the multilayer classification results, forming the algorithm in this paper" Neighboring Envelope Embedded Stacked Autoencoder". The main contribution of this paper can be expressed as follows.

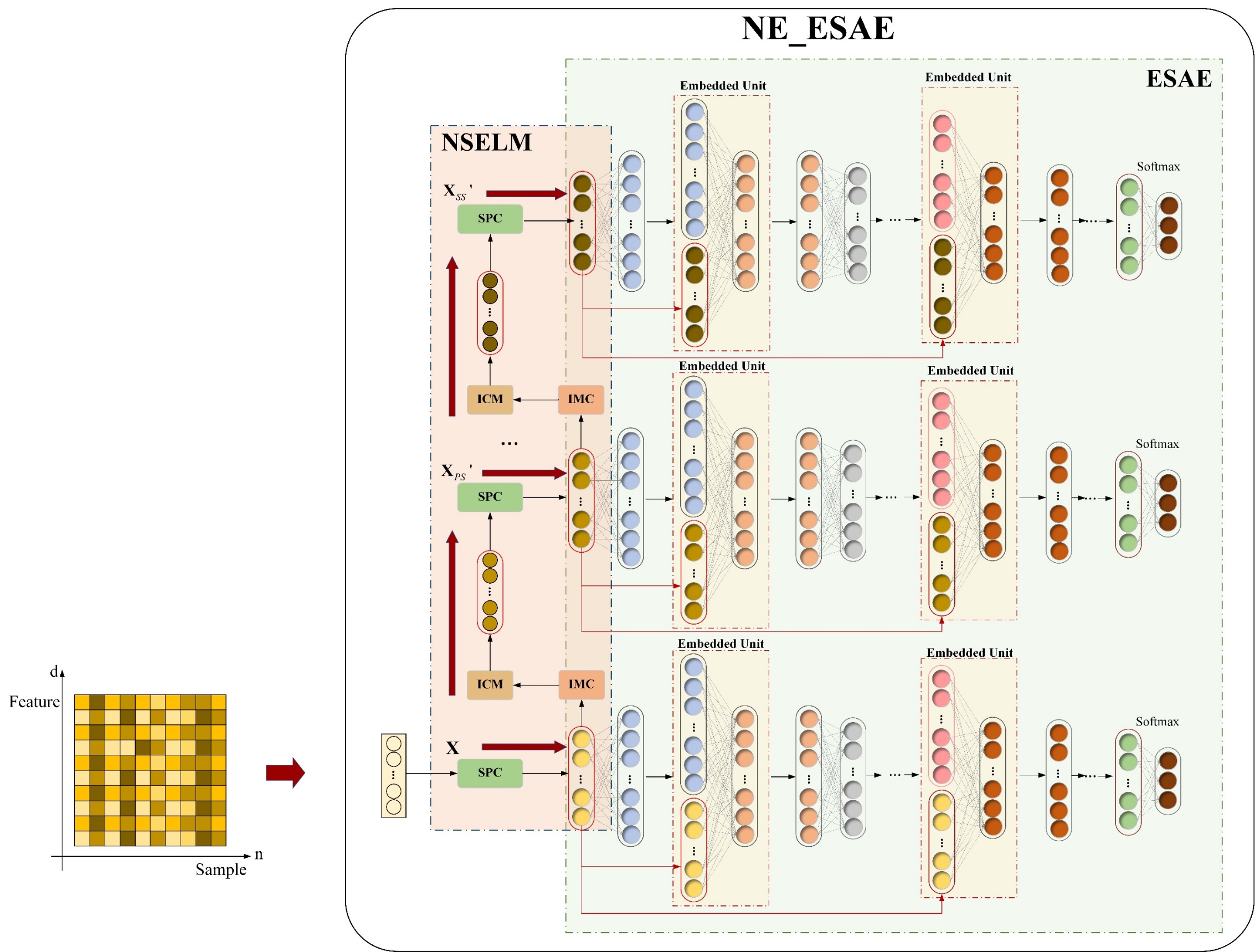
1. The neighboring sample envelope learning mechanism (NSELM) is proposed to construct hierarchical new samples (called ‘deep samples’ or ‘envelope samples’). First, a sample-pair connection mechanism (SPC) is proposed to construct a sample-pair for neighboring samples. Then, multilayer iterative mean clustering with an interlayer consistency mechanism (ICMC) is proposed to achieve deep sample transformation and mine the structural information of the sample. This mechanism deeply mines the information among the samples, thereby constructing more powerful ‘bigger’ samples (called ‘envelope samples’).
2. The embedded stacked autoencoder (ESAE) is designed in this paper to mine relationship between samples with original features and samples with deep features during training and within network. In other words, the ESAE is able to mine the structure information with different feature sets, thereby achieving high-quality deep features.
3. Based on the NSELM and ESAE, a new SAE model is proposed and is called the neighboring envelope embedded stacked autoencoder (NE\_ESAE). This model realizes the sample-feature cooperative transformation, while existing SAEs only realize the feature transformation.

# Related works

Here, we introduce some related works on SAE. In the case of using the sparse autoencoder and considering their stacking, this method is called the stacked sparse autoencoder (SSAE) [29]. A sparse autoencoder is obtained by adding some sparse constraints to the traditional autoencoder, which can suppress most of the output of hidden-layer neurons. Shi Y *et al*. [30] introduced distance constraints into sparse stacked autoencoders to construct distance-constrained sparse stacked autoencoder (DCSSAE). Liu L *et al*. [31] improved the feature learning ability and extracted more representative feature representations than the basic model by proposing the latent relationship guided stacked sparse autoencoder (LRSSAE). Pelin Görgel *et al*. [32] proposed a stacked denoising sparse autoencoder (SDSAE) that stacks the denoising and sparse autoencoders together to form a robust deep mode. Zhu *et al*. [33] proposed a new stacked pruned sparse autoencoder (SPSAE) model. Different from traditional auto encoders, this model contains a fully connected autoencoder, which utilizes the dominant features extracted in all layers to participate in subsequent layers thereby reducing the loss of information. Sun *et al*.[34] proposed a gated stacked target-related autoencoder (GSTSAE), which is a novel deep feature extraction and layer wise ensemble method.



(a) The traditional SAE



(b) The proposed NS\_ESAE.

Fig. 3. Flowchart of the traditional SAE and NE\_ESAE.

# Proposed Method

This section describes the two aspects of the proposed method. In the following, Section A describes the proposed NE\_ESAE. Section B describes the classification algorithm based on NE\_ESAE.

## Neighboring Envelope Embedded Stacked Autoencoder (NE\_ESAE)

Fig. 3 (a) shows the flowchart of the traditional SAE. Fig. 3 (b) shows the flowchart of the proposed NE\_ESAE, which mainly includes the neighboring sample envelope learning mechanism (NSELM) and embedded stacked autoencoder (ESAE). The NSELM mines the spatial information of the samples to realize the multilayer transformation of samples. ESAE is designed and trained to extract multilayer deep features. Different from the traditional SAE in Fig. 3 (a), the NE\_ESAE can mine the hierarchical structure information of the original samples and the complementarity between the original features and the deep features.

## 1) Neighboring Sample Envelope Learning Mechanism (NSELM)

The proposed NSELM consists of two parts. The first is the sample-pair connection mechanism (SPC) based on a near-neighbor sample, which constructs a sample-pair by connecting the original sample with the nearest-neighbor sample. The second is ICMC, which can be seen as a combination of iterative mean clustering (IMC) and the interlayer consistency mechanism (ICM). ICMC constructs multilayer sample spaces and achieves local and global consistency of structural information between interlayers to better realize more powerful and ‘bigger’ samples.

*A) Sample-pair connection mechanism based on near-neighbor (SPC):* To mine the neighborhood relationship between samples, this paper proposes SPC. We propose the  operator for transforming (reconstructing) the original sample. Given the original training data, the sample-pair is constructed as follows. First, we randomly pick from the same class sample set of  in  and calculate the Euclidean distance between  and. Based on this, we can find the neighboring sample, which is closest to . We use  to represent the distance, and the objective function can then be expressed as

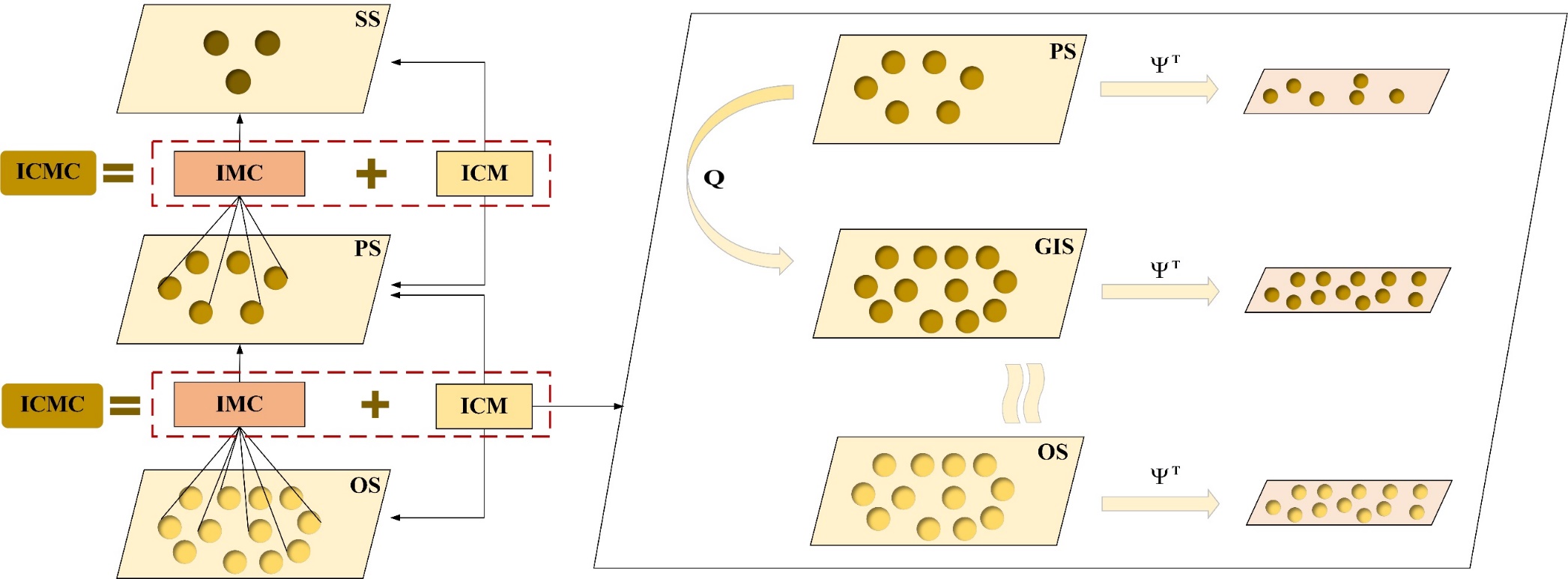


Fig. 5. Flowchart of the proposed ICMC. ICMC consists of IMC and ICM. IMCs are used to construct the multilayer sample spaces, and ICM is designed to effectively reduce the distribution difference of interlayer samples. In ICM, PS is used to generate GIS, and its sample distribution is similar to OS. GIS generation is carried out by the generation matrix  in an unsupervised manner.



Therefore, concatenating  and, the SPC can be expressed as



The calculation process of  is shown in Fig. 4. In the figure, sample-pair  is generated by SPC. We refer to the sample space consisting of  as the original sample-pair space (OS), which contains neighborhood structural information among the original samples (relationship between neighboring original samples).

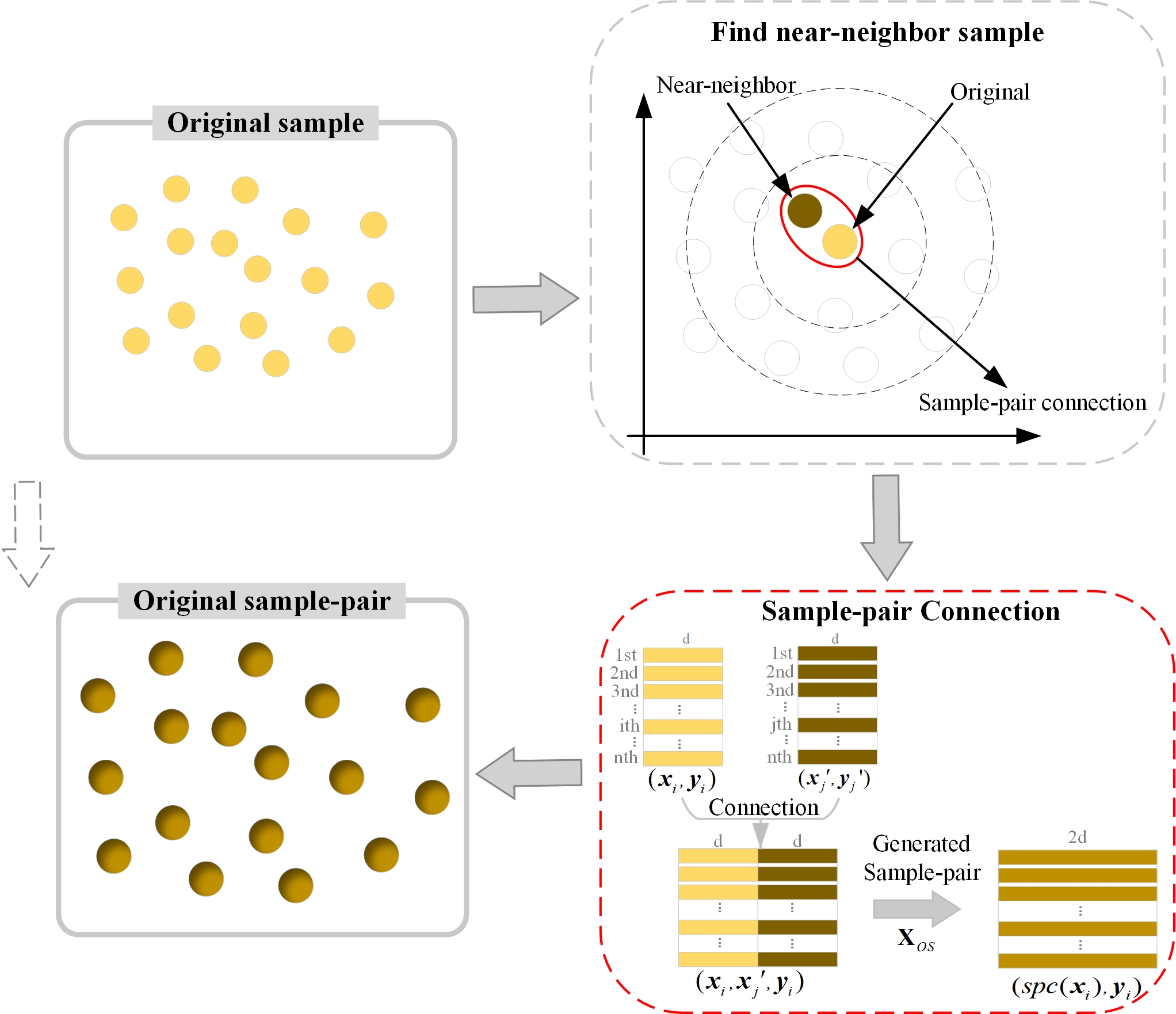


Fig. 4. Sample-p air connection mechanism. First, an original sample is selected; then, the intrinsic structure of similar neighbor samples is captured; and finally, the original sample and the near-neighbor sample are connected to generate a sample-pair.

*B) Multilayer iterative mean clustering based on the interlayer consistency mechanism (ICMC):* To mine the relationship betwee n similar samples, this paper proposes the ICMC and is shown in Fig. 5. As shown in the figure, ICMC mainly consists of IMC and ICM. IMCs are used to construct the multilayer sample spaces, and ICM is designed to effectively reduce the difference in the distribution of interlayer samples. The proposed ICMC method is described as follows.

*I) IMC:* Let  be the  cluster centers. The formula of IMC is as follows:



where  is the sample set with  as the cluster center. To determine, the following two steps need to be repeated continuously:

Step 1: For each sample-pair , the Euclidean distance between  and each cluster center is calculated in turn, and  is divided into the cluster corresponding to the nearest sample center. Let the *p*th () sample cluster generated in the *t*th clustering process be recorded as . Its allocation result can be expressed as



Step 2: After completing step 1,  sample clusters are obtained, and the sample center of each cluster is recalculated by taking the mean value of all dimensions of all samples in the current cluster.

where  represents the number of samples in . The above two steps are repeated until convergence, and the final  sample centers are taken as new samples (a new sample set).

Based on the above steps, the primary clustering sample space (PS) can be obtained based on the original sample-pair space (OS). By repeating the IMC process, we can obtain a secondary clustering sample space (SS). The multilayer sample spaces constructed by ICMs are shown in (6).



*II) ICM:* To ensure the consistency of the sample distribution among multilayer sample spaces, we design an ICM between neighboring sample spaces. Before doing so, we need to transpose the matrix in each sample space (i.e., ,) to ensure the subsequent work. As shown in Fig. 5, in ICM, we believe that PS can generate an intermediate sample space (GIS) with a similar distribution to OS by the generative transfer matrix . Our main purpose is to align the PS and OS sample distributions through GIS.

We use the implicit but generic transformation  to represent the training set of PS, OS and GIS, which are defined as, and . For convenience,  and  are defined as a sample in GIS and OS, respectively, considering that . Therefore, ICM expression is



Overall, ICM consists of three items. The first is the global distribution difference (GDD) loss, which minimizes the global difference in the edge distribution between GIS and OS. The second item is local distribution difference (LDD) loss, which uses the locality of OS to measure the local area difference from GIS. The third part is LRC regularization, which is performed to maintain the generalization of the generator matrix  and can effectively display the global structure of samples in OS and PS. Considering that the nonconvexity of the rank function is NP-hard, this paper uses the nuclear norm  as the rank approximation. The ICM is described as follows.

*a) Global distribution difference (GDD):* The first part in ICM is the GDD loss to minimize the global difference in marginal distributions between the GIS and the OS. We use a linearly transformed projection matrix  to find the potential common subspace of  and . At the same time, we introduce the generative transfer matrix ; by substituting it in (7), the GDD loss after projection can be rewritten as



where  can be expressed as some linear combinations, i.e.,.denotes the linear combination coefficient matrix and,. Then, the projected PS can be expressed as , and the projected OS can be expressed as. Let  and ; the PS and OS can be expressed simply as  and , respectively. Therefore, the GDD loss is formulated as



*b) Local distribution difference (LDD):* The second part in ICM is the LDD. The LDD loss exploits the locality of the OS to measure local differences and indirectly enhances the distribution consistency between the GIS and the OS. The LDD loss in (7) can be rewritten as



The matrix  is a diagonal matrix with entries**,** and  is the trace of a matrix. is the affinity matrix, described as



where  represents the *z*th nearest neighbors of sample . It is the same as GDD. By substituting  and using the linearly transformed projection matrix  to find the potential common subspace of  and, the LDD loss can be rewritten as



*c) ICM based on a combination of GDD and LDD:* Finally, combining the above three points and introducing the trade-off parameters  and  at the same time, the objective of our ICM can be rewritten as



The rows of  must be orthogonal and normalized to the unit norm to prevent trivial solutions by enforcing, which can be further rewritten as , an equality constraint.

*d) Optimization:* Obviously, there are two variables  and  to solve in ICM. Generally, it can be effectively solved by the alternating directional method of multipliers (ADMM) [35]. By introducing auxiliary variable, furthermore, using the augmented Lagrangian method (ALM) [36], ICM can be rewritten as



where  and  are the augmented Lagrangian multipliers and penalty parameters, respectively, and  represents a full one matrix. Then, the variables ,  and  in ICM can be alternatively solved as follows.

Step 1: Update for : When  and  are fixed, the objective function in (14) is converted into



we can derive the solution of the *j*th iterations column wise. To obtain the *i*th column vector in  by setting the partial derivative of (15) with respect to  to be zero yields



Step 2: Update for : When  and  are fixed, the objective function in (14) is converted into



we can use the gradient descent algorithm [37] to solve the closed-form solution of , and after (*j+1)*th iterations,  can be updated as



Step 3: Update for : When  and  are fixed, the objective function in (14) is converted into



after the (*j+1)*th iteration, can be updated as



|  |
| --- |
| **Algorithm 1** ICMC |
| **Input:** Data matrix |
| **Procedure:** |
| 1. Setting parameters: 2. Calculate  by (4) 3. Get  by (5) 4. Constructing multilayer sample spaces as shown in (6) |
| 1. Get new transposed data, |
| 1. Compute , ,, |
| 1. Initialize: |
| 1. **While** not converge **do** |
| 1. Update , fix  and  by (15)-(16) 2. Update , fix  and  by (17)-(18) 3. Update , fix  and  by (19)-(20) 4. Update the multiplier  and parameter: 6. : 7. Check convergence |
| 1. **End while** |
| 1. Compute |
| **Output:** |

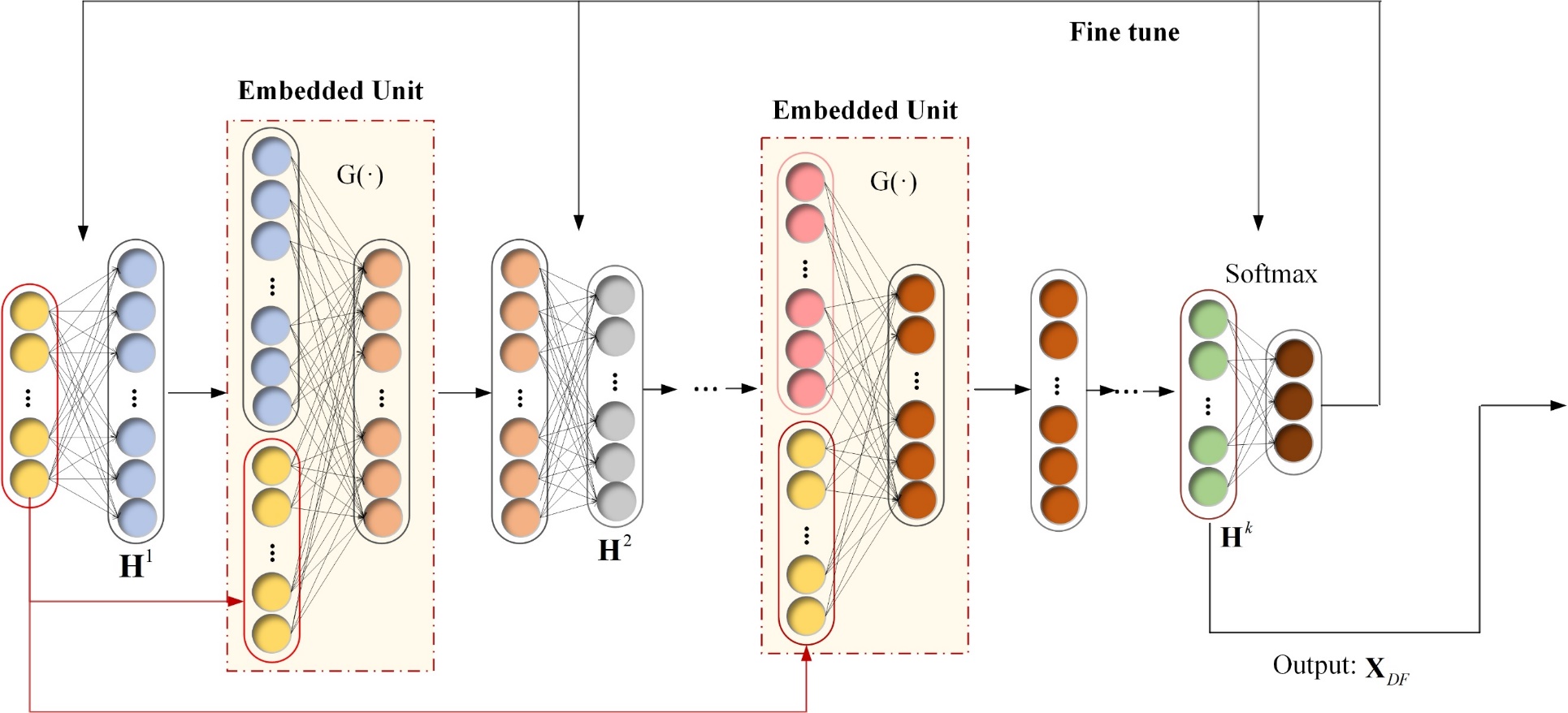


Fig. 6. Structure of the proposed ESAE. The embedded unit introduces the original input feature information into the training process of the encoder network and performs feature transformation through embedding criteria.

We can use the singular value threshold (SVT) operator [38] to solve . The specific optimization process of variables ,  and can be referred to[39].

The pseudocode of Algorithm 1 - the proposed ICMC is described as follows.

The projected PS can be represented as , where  denotes sample dimensionality after ICMC. ICMC allows us to maintain the local and global consistency of the sample data structure information to better realize efficient sample transformation.

## 2) Embedded Stacked Autoencoder (ESAE)

The ESAE model is shown in Fig. 6. Different from the traditional SAE, the key part of the ESAE is the embedded unit between two adjacent hidden layers. Assume that the input data of the ESAE network are , where the output matrix of the hidden layer in the *k*th encoder is , 1<*k*<*K*, and  represents the number of hidden layer neurons of the *k*th encoder. The embedded unit can be expressed as



where ,  means to concatenate the original input feature  and the hidden layer output feature  of the encoder, and  is the corresponding transformation matrix, consisting of 0's and 1's. After transformation, the hidden layer output vector of the *k*th encoder in the ESAE network can be expressed as

where  and  are the connection weight matrix and bias vector between the input layer and the hidden layer in the *k*th AE, respectively.  represents the activation function. The activation function of the encoder in this paper adopts the sigmoid function, which is . The purpose of the embedding unit is to introduce certain original information constraints in the layer-by-layer training process of the AE network and filter some hidden layer outputs with weak class representation capabilities. The objective function of the proposed embedded unit is expressed as



We then calculate the covariance matrix  of the feature matrix , sort its diagonal elements in descending order, and take the first  values to form a vector . The transformation matrix  is obtained according to the following formula:



where  is the *i*th diagonal element of the covariance matrix ,  is the *j*th element of the vector, and . By combining (21)-(24), we can obtain the hidden layer output of the *k*th encoder. Then, the decoding function of the *k*th encoder can be rewritten as



where  is the data obtained by reconstructing the output of the embedded unit.  and  are the connection weight matrix and bias vector between the hidden and output layers in the kth encoder, respectively.

We introduce KL divergence to enable the hidden layer of the encoder to learn the sparse representation. The formula for KL divergence is:



where  is the activation value of the *i*th input vector on the *j*th neuron in the hidden layer. After introducing the embedded unit in the structure and introducing the sparse criterion in the training process, the optimization objective function of the *k*th encoder of ESAE is written as



whereandare the penalty parameters of the regularization item and the sparse criterion item, respectively.

After two stages of network pretraining and fine-tuning, for the *i*th input vector , each hidden layer in the network outputs a new feature vector, representing different levels of information. We take the output of the last hidden layer as the deep feature learned by the network, which can be expressed as follows:



where  is the number of neurons in the last hidden layer. The ESAE is shown in Algorithm 2.

3*) Division of training and test set*

The construction method of the training data and test data in each layer of the sample space are described as follows. First, in OS, sample-pair  is obtained from the original training data  by training the first SPC. As described in above, two layers of ICMC construct multilayer samples , , and  and obtain the optimal linear combination coefficient matrix  and . After that,  and  are combined with the second and third SPC to obtain  and  in the PS and SS, respectively. Finally, three layers of envelope samples (,  and ) will be input into each ESAE layer for training.

Then, the performance of NE\_ESAE is verified by test data. First, the test data  are input into the first trained SPC to obtain . Then, in PS,  is input into the first layer of trained ICMC to obtain , where . In SS,  is input into the second layer of the trained ICMC to obtain , where . Then,  and  are input into the second and third trained SPC to obtain  and  in the PS and SS, respectively. Finally, three layers of envelope samples (,and) will be input into each layer of the trained ESAE to extract deep mixed features F1, F2 and F3 and to obtain classification results. The results from three layers of envelope samples (,and) are fused to obtain the final results.

|  |
| --- |
| **Algorithm 2** ESAE |
| **Input:** Data matrix |
| **Procedure:** |
| 1. Setting parameters: , ,  and the number of neurons in each hidden layer, the number of iterations of network training, etc. |
| 1. **Pretraining:** |
| 1. Training the first layer of ESAE, and the output of the hidden layer is recorded as |
| 1. **For**  **do** |
| 1. Calculate  according to (23)-(34) |
| 1. Embed the hidden layer output  of the previous encoder according to the original features of (21) |
| 1. Train the *k*th layer of the network with (27) as the objective function |
| 1. Obtain the encoder output of the kth hidden layer and denote it as |
| 1. **End for** |
| 1. **End pretraining** |
| 1. Stack the hidden layers and add a softmax layer on top |
| 1. Network Fine-tuning |
| **Output** |

## Classification algorithm based on NE\_ESAE

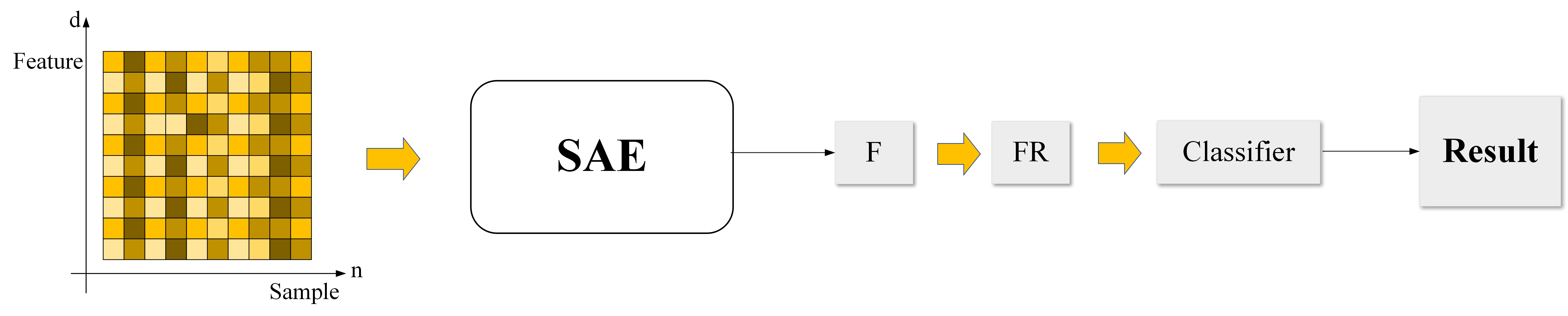
To improve the complementarity of multilayer sample space, this paper proposes the multilayer space ensemble mechanism (MSEM). MSEM fuses the results under each layer of sample space to achieve final result. This MSEM will involve two commonly used ensemble mechanisms: Weighted fusion (WF) and majority voting (MV). By NSELM, NE\_ESAE contains three layers of sample spaces, which are OS, PS and SS, and which contain structured samples (or called envelope samples). Suppose that the prediction result of the model in the OS is , the prediction result of the model in the PS is , the prediction result of the model in the SS is , the realization process of multilayer space ensemble mechanism is as follows. For detail of the WF and MV, please see the paper [40]-[41]. By the MSEM, the three results from the NE\_ESAE are fused.

The traditional classification algorithm based on SAE is shown in Fig. 7 (a), which directly inputs the deep features learned by SAE into the classifier to obtain the final results. In contrast, our NE\_ESAE outputs three layers of deep features F1, F2 and F3 by constructing a multilayer of envelope samples. To improve the complementarity among multilayer features, we combine NE\_ESAE with MSEM to construct a classification algorithm based on NE\_ESAE (Fig. 7 (b)). As shown in the figure, MSEM fuses the classification results from the three layers of ESAEs to realize the sample-feature cooperative transformation.

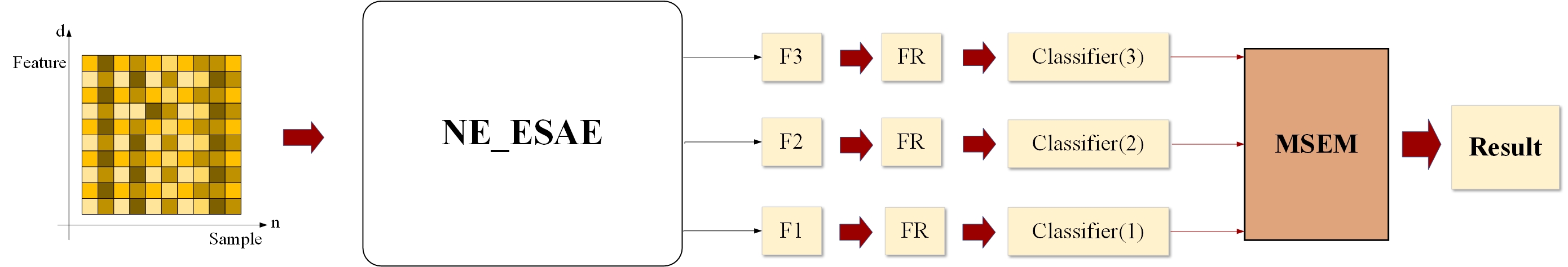
TABLE I

Basic information of the datasets used in the study

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset | Instances | Attributes | Class |
| Alzheimer's disease (AD) | 90 | 32 | 3 |
| LSVT Voice Rehabilitation Dataset (LSVT) | 126 | 310 | 2 |
| Parkinson Speech Dataset (PD) | 1040 | 26 | 2 |
| Pen-Based Recognition of Handwritten Digit (Pendigits) | 10992 | 16 | 10 |
| Statlog Landsat\_Satellite (Statlog) | 6435 | 36 | 6 |
| Statlog Vehicle Silhouettes (Vehicle) | 846 | 18 | 4 |
| Statlog Heart Dataset (Heart) | 270 | 13 | 2 |
| Maxlettle Parkinson Dataset (Maxlettle) | 195 | 22 | 2 |
| Urban land cover (Urban) | 675 | 147 | 9 |
| Breast Cancer Wisconsin Diagnostic (WDBC) | 569 | 30 | 2 |
| Breast Cancer Wisconsin Original (Wisconsin) | 683 | 9 | 2 |
| Pima Indians Diabetes Dataset (PID) | 768 | 8 | 2 |
| Letter Recognition (LR) | 20000 | 16 | 26 |
| Gas Sensor Array Drift (GSAD) | 13910 | 128 | 6 |
| Human Activity Recognition (HAR) | 10299 | 561 | 6 |



(a) The classification algorithm based on SAE



(b) The classification algorithm based on NE\_ESAE

Fig. 7. The classification algorithm based on traditional SAE and proposed NE\_ESAE.

# Experimental results and analysis

To verify the effectiveness of the proposed NE\_ESAE, six groups of experiments were conducted. The first group of experiments based on an ablation study was conducted to verify the effectiveness of SPC and ICMC, the effectiveness of MSEM, and the effectiveness of NSELM and ESAE. The second group of experiments compares the proposed algorithm with existing representative feature learning algorithms, the state-of-the-art kernel feature methods and representative SAE models. The third group of experiments analyzes the effects of some parameters, including the type of classifier and the proportion of cluster center samples. The fourth group of experiments involves the confusion matrix. In the fifth experiment, the computational complexity and time of the algorithm were analyzed. In the sixth experiment, the model visualization and convergence analysis were carried out. The data and codes can be found in <https://github.com/ChuanyanZhou/NEESAE>.

## Experimental conditions

The performance of the proposed algorithm was tested on 15 representative datasets [42]-[53]. These datasets have both large and small samples, binary and multiple class labels, and high-, medium- and low-dimensional features. The main information concerning all datasets is shown in Table I.

In the following experiments, the number of layers of the encoder in the proposed NE\_ESAE network was set to 3 for all experiments since the sample size was not large for some datasets; the number of hidden layer neurons of each encoder was determined according to the number of samples and feature dimensions of the dataset, and the optimal structure was determined by the grid search method. The experiment adopted the hold-out cross-validation method, and each dataset was divided into three equal parts. The classifier used in the experiment was a support vector machine (SVM) for fair comparison.

## Ablation study

*1) Effectiveness analysis of SPC:* We designed three sets of experiments. They were experiments that only used original features (OF), experiments that used features learned by PCA (OF&PCA), experiments that used features learned by LDA (OF&LDA). The experimental results are listed in Table II.

The experimental results in Table II show that most of the sample-pairs are better than the original sample for most datasets. The result demonstrates that the sample-pair can learn more efficient and discriminative feature representations than the original sample. Sample-pairs performed better in small and medium sample datasets, especially in 90 samples with AD, which improved accuracy by approximately 10% in all cases. This shows that our proposed SPC method has obvious advantages on small and medium sample size datasets.

*2) Effectiveness analysis of ICMC:* To verify the effectiveness of ICMC, we used the ablation method to compare the classification accuracy between sample-pair and sample-pair after ICMC processing (sample-pair &ICMC). The experimental results are listed in Table III. According to the experimental results shown in Table III, it is obvious that the sample-pair &ICMC outperforms the sample-pair on 13 of all datasets. Among them, the classification accuracy of the heart dataset is improved by 6.44%. This shows that our proposed ICMC mechanism has obvious advantages on most datasets.

TABLE II

Comparison of the original sample and sample-pair

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset |  | OF (%) | OF&PCA (%) | OF&LDA (%) |
| AD | Original sample | 54.00±9.55 | 60.00±8.50 | 62.67±4.94 |
| Sample-pair | **64.67±4.47** | **70.67±4.35** | **72.00±6.91** |
| LSVT | Original sample | 80.48±6.39 | 90.48±3.37 | 87.14±3.61 |
| Sample-pair | **94.29±3.98** | **96.67±1.30** | **95.71±3.91** |
| PD | Original sample | 62.70±1.86 | 64.94±1.95 | 64.20±1.74 |
| Sample-pair | **70.75±1.74** | **71.90±1.06** | **73.22±1.67** |
| Pendigits | Original sample | **98.13±0.05** | **98.07±0.13** | **97.87±0.23** |
| Sample-pair | 66.36±0.98 | 68.56±2.40 | 64.63±1.84 |
| Statlog | Original sample | **86.13±0.53** | **87.23±0.54** | **87.09±0.63** |
| Sample-pair | 82.07±1.10 | 85.85±0.36 | 82.89±1.07 |
| Vehicle | Original sample | 80.35±1.31 | 82.34±1.05 | 82.55±0.92 |
| Sample-pair | **83.90±0.19** | **85.32±0.78** | **85.53±1.05** |
| heart | Original sample | 80.89±4.26 | 85.11±3.30 | 83.33±3.60 |
| Sample-pair | **85.56±2.83** | **90.67±2.02** | **90.67±2.30** |
| Maxlittle | Original sample | 85.54±4.01 | 88.00±2.28 | 88.62±4.16 |
| Sample-pair | **86.77±2.57** | **91.08±3.15** | **90.78±1.09** |
| Urban | Original sample | **79.91±3.87** | **82.40±2.80** | **83.38±2.10** |
| Sample-pair | 73.42±2.48 | 75.73±2.87 | 75.02±2.02 |
| WDBC | Original sample | 95.66±1.52 | 97.88±0.84 | 97.46±0.69 |
| Sample-pair | **97.57±1.38** | **98.73±1.16** | **99.58±0.44** |
| Wisconsin | Original sample | 96.30±1.72 | 97.18±1.19 | 96.83±1.26 |
| Sample-pair | **97.18±1.48** | **98.06±0.74** | **97.89±1.26** |
| PID | Original sample | 70.39±2.74 | 72.34±1.98 | 75.78±3.49 |
| Sample-pair | **74.14±4.27** | **80.16±1.78** | **82.34±1.88** |
| LR | Original sample | 85.23±0.18 | 85.26±0.26 | 85.51±0.27 |
| Sample-pair | **88.72±0.25** | **90.07±0.15** | **89.82±0.21** |
| GSAD | Original sample | 99.24±0.19 | 99.27±0.14 | 99.43±0.05 |
| Sample-pair | **99.43±0.02** | **99.51±0.08** | **99.62±0.08** |
| HAR | Original sample | 98.23±0.35 | 98.37±0.30 | 98.39±0.26 |
| Sample-pair | **98.70±0.08** | **98.81±0.19** | **98.86±0.14** |

*3) Effectiveness analysis of MSEM:* This section discusses the effectiveness of the MSEM. Specifically, the ESAE is trained in the OS, PS and SS, and the prediction results of each model are fused to obtain the final classification result. The fusion strategy MSEM adopts the MV method and the WF method.

As seen from Table IV, the classification accuracies of the three different sample spaces are different. The prediction results of each model can be fused by MSEM to improve the classification accuracy to a certain extent. As shown in Table IV, the classification performance of all the datasets was significantly improved after the fusion of MSEM. Compared with OS, MSEM improves the performance of all datasets by 17.34%, 0.48%, 4.34%, 9.48%, 0.47%, 1%, 2.27%, 6.92%, 2.38%, 1.84%, 0.7%, and 9.3%, respectively. Compared with the PS, the MSEM improves the performance of all datasets by 6.00%, 6.67%, 6.38%, 5.94%, 16.74%, 9.64%, 29.45%, 1.72%, 0.26%, 5.54%, 1.21% and 4.07%. Compared with the SS, the MSEM improves the performance of all datasets by 7.34%, 9.05%, 4.98%, 3.51%, 13.28%, 7.20%, 33.18%, 0.27%, 2.14%, 0.78% and 2.97%, respectively.

TABLE III

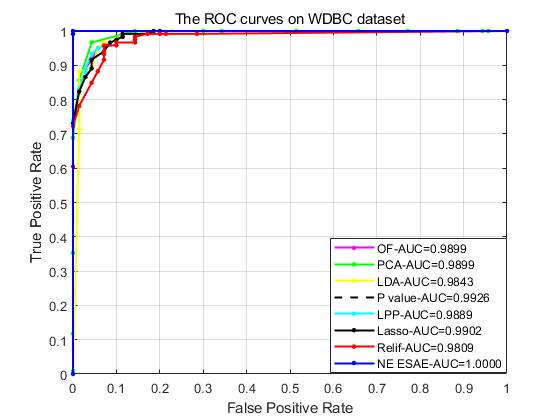
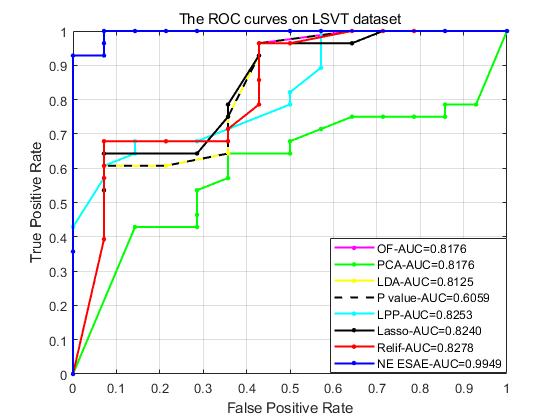
Comparison of sample-pair and sample-pair &ICMC

|  |  |  |
| --- | --- | --- |
| Dataset | Sample-pair (%) | Sample-pair &ICMC (%) |
| AD | 64.67±4.47 | **66.00±5.96** |
| LSVT | 94.29±3.98 | **95.71±3.91** |
| PD | 70.75±1.74 | **70.86±1.63** |
| Pendigits | 66.36±0.98 | **69.61±1.62** |
| Statlog | 82.07±1.10 | **85.13±0.30** |
| Vehicle | **83.90±0.19** | 77.66±0.43 |
| heart | 85.56±2.83 | **92.00±3.08** |
| Maxlittle | 86.77±2.57 | **87.69±3.61** |
| Urban | 73.42±2.48 | **76.49±1.90** |
| WDBC | 97.57±1.38 | **99.58±0.24** |
| Wisconsin | 97.18±1.48 | **98.24±0.82** |
| PID | 74.14±4.27 | **79.14±1.42** |
| LR | **88.72±0.25** | 84.77±0.66 |
| GSAD | 99.43±0.02 | **99.53±0.18** |
| HAR | 98.70±0.08 | **99.25±0.12** |

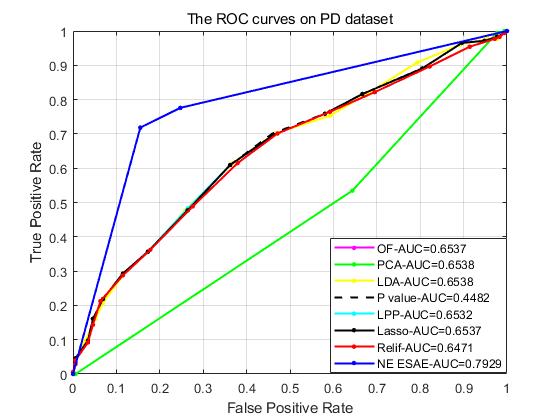
In addition, it can be seen from Table IV that the improvement is apparent. For example, the MV method in AD performs 2.67% better than the WF method. Combined with the above experimental results, it is proven that the proposed MSEM is effective and can significantly improve the accuracy and stability of the classification model.

*4) Effectiveness analysis of the NSELM and ESAE:* To verify the effectiveness of the proposed NSELM and ESAE, we used the ablation study to compare the classification performance of NE\_ESAE, NE\_ESAE without NSELM (no NSELM), and NE\_ESAE without ESAE (no ESAE) under the same network structure. The experimental results are shown in Table V.

As we can see from Table V, NSELM leads to higher classification accuracy for most datasets. This result means that the proposed NSELM can deeply mine the structural information of samples, thereby effectively improving the classification accuracy. For small sample datasets such as AD, LSVT, Maxlittle, and heart, our method shows clear advantages; at the highest level, the classification accuracy of the AD dataset is improved by 30.67%. Furthermore, the comparison between NE\_ESAE and NE\_ESAE without ESAE verifies that the proposed ESAE can effectively enhance the feature extraction ability.



(a) LSVT (b) WDBC



(c) The description of the ROC curves on PD

Fig. 8. Description of the ROC curves on LSVT, WDBC and PD.

## Algorithm comparison

*1)* *Comparison with classical feature-learning algorithms:* To evaluate the performance of the NE\_ESAE algorithm, its performance was compared to those representative feature-learning algorithms, including feature selection algorithms (PCA [54], LDA [55], LPP [56], Relief [57], Lasso [63] and P\_value [64]). The results are listed in Table VI. In addition to accuracy, receiver operating characteristic (ROC) curves were also obtained in this experiment to better compare the sample recognition ability of the proposed algorithm and the comparison algorithm.

TABLE IV

MSEM effectiveness analysis experimental comparison

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | OS (%) | PS (%) | SS (%) | MSEM (%) | |
| MV | WF |
| AD | 73.33±6.24 | 84.67±10.70 | 83.33±8.50 | **90.67±6.41** | 88.00±7.30 |
| LSVT | 96.19±3.61 | 90.00±13.08 | 87.62±12.19 | **96.67±3.61** | 93.33±4.26 |
| PD | 74.43±1.61 | 72.36±1.68 | 73.76±6.64 | 77.76±3.79 | **78.74±4.22** |
| Vehicle | 80.08±2.49 | 83.59±4.96 | 86.02±3.79 | 87.66±2.97 | **89.53±1.96** |
| Pendigits | 83.74±0.62 | 67.47±5.11 | 70.93±6.92 | 79.61±7.99 | **84.21±0.83** |
| Statlog | 88.33±0.18 | 78.29±4.52 | 80.73±4.41 | 86.03±1.05 | **87.93±0.93** |
| Urban | 82.02±2.50 | 54.84±6.66 | 51.11±5.63 | 65.69±6.06 | **84.29±3.38** |
| heart | 92.52±3.19 | 94.72±2.46 | 99.17±1.06 | 97.78±2.57 | **99.44±1.11** |
| WDBC | 97.62±2.68 | 99.74±0.31 | 100.00±0.00 | 99.87±0.26 | **100.00±0.00** |
| Maxlittle | 95.08±2.28 | 91.38±6.21 | 94.78±4.56 | **96.92±1.09** | 96.25±5.07 |
| Wisconsin | 99.19±1.28 | 98.68±0.88 | 99.11±0.87 | **99.89±0.21** | 98.88±3.07 |
| PID | 80.08±2.49 | 85.31±1.58 | 86.41±3.80 | 88.20±2.63 | **89.38±1.88** |
| LR | 92.59±2.46 | 94.77±1.60 | 90.72±3.67 | **96.78±1.16** | 94.77±1.45 |
| GSAD | 98.82±0.39 | 99.54±0.25 | 99.51±0.11 | 99.72±0.14 | **99.76±0.14** |
| HAR | 99.10±0.06 | 99.25±0.13 | 99.65±0.22 | 99.65±0.08 | **99.70±0.11** |

The experimental results in Table VI further demonstrate the advantages of the proposed NE\_ESAE algorithm. For example, in dataset AD, the accuracy of our algorithm is 28% higher than the best LDA and 42% higher than the worst P\_value among other classical feature-learning algorithms. In dataset PD, the accuracy of our algorithm is 13.8% higher than that of PCA and 22.13% higher than that of P\_value. The results also show that simultaneous sample learning and feature learning are more effective than only feature learning in the original space. The multilayer sample spaces contain different structural information from the original space, which can provide more potential sample distribution information. In addition, it can be seen from the variance in the table that the proposed algorithm has better stability than other feature learning algorithms in most cases.

TABLE V

NSELM and ESAE effectiveness analysis experimental comparison

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset | NE\_ESAE without NSELM (%) | NE\_ESAE  without  ESAE (%) | NE\_ESAE (%) |
| AD | 60.00±3.33 | 79.33±14.41 | **90.67±6.41** |
| LSVT | 80.48±8.81 | 96.19±5.98 | **96.67±3.61** |
| PD | 63.33±4.80 | 72.82±3.18 | **78.74±4.22** |
| Vehicle | 78.58±6.70 | 80.99±2.85 | **89.53±1.96** |
| Pendigits | **98.51±0.27** | 67.40±1.49 | 84.21±0.83 |
| Statlog | 85.32±2.83 | 83.53±0.63 | **87.93±0.93** |
| Urban | 74.29±1.32 | 74.82±5.66 | **84.29±3.38** |
| Heart | 84.44±5.88 | 96.89±2.53 | **99.44±1.11** |
| WDBC | 96.17±1.43 | 99.79±0.29 | **100.00±0.00** |
| Maxlittle | 86.15±3.08 | 89.23±5.76 | **96.92±1.09** |
| Wisconsin | 97.35±0.88 | 99.50±0.48 | **99.89±0.21** |
| PID | 73.44±7.14 | 82.50±2.28 | **89.38±1.88** |
| LR | 72.23±0.90 | 89.20±0.90 | **96.78±1.16** |
| GSAD | 83.50±4.77 | 99.67±0.09 | **99.76±0.14** |
| HAR | 98.71±0.45 | 99.70±0.12 | **99.70±0.11** |

To visually show the accuracy of the algorithms, Fig. 8 (a) to (b) describe the ROC curves of the methods on the LSVT and WDBC datasets. The AUC (area under the curve) is the area enclosed by the ROC curve and the coordinate axes. It can be seen from the figure that the AUC of NE\_ESAE is higher than that of the comparison method, which appears that the prediction model of the proposed method is optimal in comparison.

*2)* *Comparison with* *state-of-the-art kernel feature methods:* Because our algorithm uses kernel methods in ICMC, we compare our proposed method with state-of-the-art kernel feature methods: Kernel PCA (KPCA)[58], Kernel LDA (KLDA)[59], Kernel Dictionary learning (KDL) [60], Kernel transform learning (KTL) [61], and Kernel label consistent transform learning (KLCTL) [62]). The experimental results are shown in Table VII.

The results show a certain trend. First, internalization helps; We always get better results with nuclearization than with linearization. For example, the classification accuracy of PCA and LDA after nuclearization for the same data set is obviously improved. At the same time, we can find that most of the data sets our method shows better classification effect, which can explain the superiority of our proposed method. In addition, we can find that our method is 1.2% higher than the best KLDA on the HAR dataset and 4.68% higher than the best SKTL on the LR dataset. The results show that the kernel method we used in combination with ESAE has certain advantages.

TABLE VI

Comparison of different feature-learning algorithms

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset | OF (%) | PCA (%) | LDA (%) | LPP (%) | Relief (%) | Lasso (%) | P\_value (%) | NE\_ESAE (%) |
| AD | 54.00±9.55 | 60.00±8.50 | 62.67±4.94 | 60.67±7.60 | 54.00±10.90 | 54.00±9.55 | 48.67±7.67 | **90.67±6.41** |
| LSVT | 80.48±6.39 | 90.48±3.37 | 87.14±3.61 | 90.48±3.76 | 88.57±3.10 | 76.67±5.16 | 75.71±5.93 | **96.67±3.61** |
| PD | 62.70±1.86 | 64.94±1.95 | 64.20±1.74 | 65.29±1.47 | 64.08±1.13 | 62.70±1.86 | 56.61±3.92 | **78.74±4.22** |
| Vehicle | 80.35±1.31 | 82.34±1.05 | 82.55±0.92 | 81.77±0.89 | 78.16±0.19 | 80.781.05 | 75.39±0.89 | **89.53±1.96** |
| Pendigits | **98.13±0.05** | 98.07±0.13 | 97.87±0.23 | 97.97±0.11 | 97.82±0.23 | 98.13±0.05 | 92.87±1.46 | 84.21±0.83 |
| Statlog | 86.13±0.53 | 87.23±0.54 | 87.09±0.63 | 87.27±0.48 | 86.10±0.40 | 86.15±0.51 | 86.13±0.68 | **87.93±0.93** |
| Urban | 79.91±3.87 | 82.40±2.80 | 83.38±2.10 | 82.93±2.53 | 80.18±2.53 | 79.91±3.84 | 80.44±2.67 | **84.29±3.38** |
| heart | 78.89±3.42 | 85.33±2.53 | 84.67±2.14 | 84.22±3.37 | 81.78±3.00 | 78.29±3.42 | 62.00±4.67 | **99.44±1.11** |
| WDBC | 95.66±1.52 | 97.88±0.84 | 97.46±0.69 | 97.35±0.53 | 97.04±1.43 | 95.78±1.54 | 88.78±2.19 | **100.00±0.00** |
| Maxlettle | 84.62±3.92 | 88.62±2.79 | 88.62±4.16 | 86.77±4.43 | 85.85±5.03 | 85.23±4.16 | 76.62±2.75 | **96.92±1.09** |
| Wisconsin | 96.30±1.72 | 97.18±1.19 | 96.83±1.26 | 97.00±1.18 | 96.74±1.83 | 96.30±1.72 | 93.66±1.67 | **99.89±0.21** |
| PID | 70.40±2.74 | 72.34±1.98 | 75.78±3.49 | 69.67±5.54 | 73.29±4.51 | 70.39±2.74 | 73.91±3.55 | **89.38±1.88** |
| LR | 85.23±0.18 | 85.26±0.26 | 85.51±0.27 | 85.39±0.27 | 84.89±0.07 | 84.43±0.15 | 85.84±0.16 | **96.78±1.16** |
| GSAD | 99.24±0.19 | 99.27±0.14 | 99.43±0.05 | 99.29±0.14 | 99.19±0.16 | 99.19±0.16 | 99.23±0.13 | **99.76±0.14** |
| HAR | 98.23±0.35 | 98.37±0.30 | 98.39±0.26 | 98.37±0.26 | 98.28±0.30 | 98.23±0.31 | 97.92±0.40 | **99.70±0.11** |

TABLE VII

Comparison of different feature-learning algorithms

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Dataset | KPCA (%) | KLDA (%) | KDL (%) | KTL (%) | KLCTL (%) | NE\_ESAE (%) |
| AD | 69.33±3.65 | 64.00±6.41 | 63.48±3.63 | 77.91±2.97 | 84.54±2.91 | **90.67±6.41** |
| LSVT | 90.61±3.53 | 84.29±4.58 | 80.32±0.86 | 86.80±3.15 | 88.81±2.86 | **96.67±3.61** |
| PD | 68.62±1.63 | 69.70±2.14 | 66.13±2.81 | 69.22±2.17 | 70.73±1.47 | **78.74±4.22** |
| Vehicle | 81.42±1.50 | 87.00±0.59 | 71.49±2.55 | 73.99±1.56 | 85.06±1.80 | **89.53±1.96** |
| Pendigits | **99.19±1.21** | 98.38±0.10 | 94.69±0.93 | 96.51±0.92 | 97.11±0.91 | 84.21±0.83 |
| Statlog | 89.84±3.52 | 87.37±0.40 | 74.00±1.26 | 85.14±0.97 | 85.68±0.82 | **87.93±0.93** |
| Urban | 81.07±3.12 | 88.35±3.49 | 79.00±1.21 | 74.76±2.70 | 80.35±1.09 | **84.29±3.38** |
| heart | 85.11±3.30 | 81.11±2.36 | 80.79±0.47 | 84.07±0.93 | 86.94±0.87 | **99.44±1.11** |
| WDBC | 97.28±1.44 | 97.91±2.53 | 92.04±1.40 | 96.07±1.25 | 96.24±0.29 | **100.00±0.00** |
| Maxlettle | 86.46±2.96 | 83.69±4.43 | 84.07±3.59 | 83.59±4.70 | 87.35±1.09 | **96.92±1.09** |
| Wisconsin | 97.47±1.31 | 94.01±2.03 | 94.36±0.13 | 95.23±1.16 | 96.94±0.67 | **99.89±0.21** |
| PID | 76.95±1.44 | 74.30±0.75 | 65.00±0.21 | 77.14±0.23 | 82.41±0.92 | **89.38±1.88** |
| LR | 89.54±2.16 | 87.51±0.69 | 84.90±0.99 | 86.93±0.63 | 92.10±0.61 | **96.78±1.16** |
| GSAD | 99.65±0.44 | 99.54±0.40 | 92.67±1.24 | 93.11±0.55 | 95.05±0.33 | **99.76±0.14** |
| HAR | 97.49±0.51 | 98.50±0.22 | 98.43±0.54 | 98.18±0.49 | 98.20±0.89 | **99.70±0.11** |

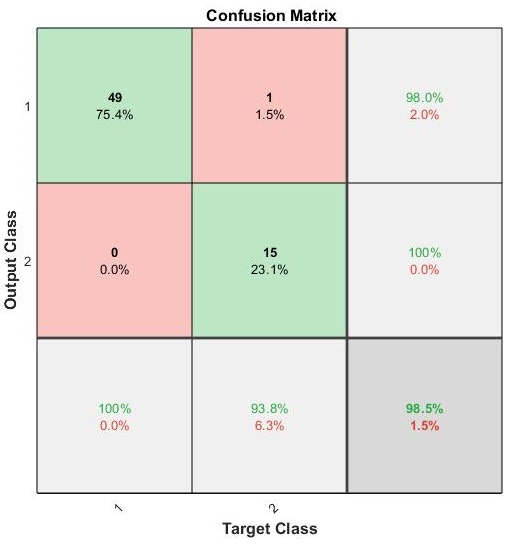
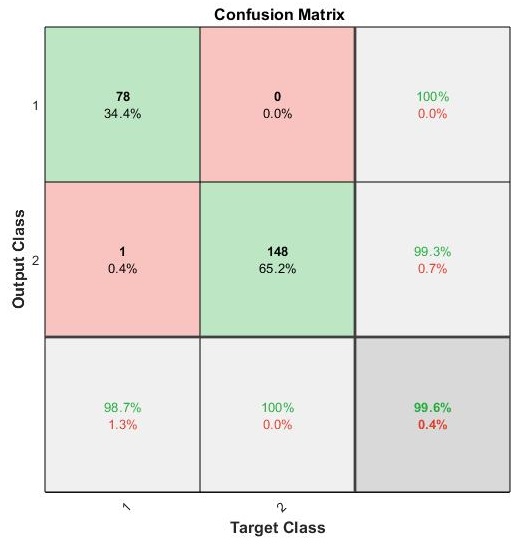
TABLE VIII

Classification accuracy (mean ±variance) of different deep autoencoder classifiers

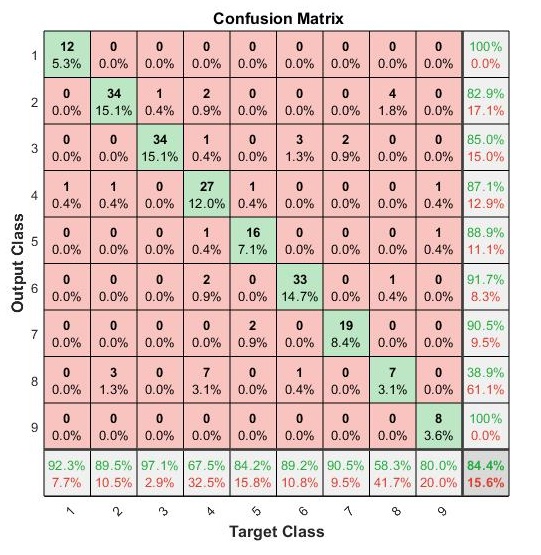
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset | SAE (%) | SSAE (%) | SDSAE (%) | SPSAE (%) | ESGSAE\_FF (%) | GSTAE (%) | NE\_ESAE (%) |
| AD | 50.67±7.95 | 56.67±5.27 | 55.58±4.36 | 57.78±4.27 | 67.33±2.49 | 71.11±8.16 | **90.67±6.41** |
| LSVT | 83.80±5.16 | 83.33±5.83 | 76.62±5.29 | 84.33±5.36 | 92.76±0.62 | 84.66±4.32 | **96.67±3.61** |
| PD | 61.15±2.91 | 64.48±2.05 | 64.88±1.84 | 64.22±2.34 | 66.72±0.87 | 73.89±4.27 | **78.74±4.22** |
| Urban | 74.48±3.33 | 79.73±0.67 | 75.17±1.88 | 77.81±1.17 | 83.20±1.01 | 76.98±0.73 | **84.29±3.38** |
| WDBC | 97.06±1.21 | 98.89±0.17 | 98.60±0.34 | 98.13±0.50 | 99.81±0.45 | 99.34±1.27 | **100.00±0.00** |
| Vehicle | 67.30±3.33 | 70.00±2.99 | 72.00±2.25 | 74.76±2.93 | 81.91±0.42 | 79.71±2.93 | **89.53±1.96** |
| Pendigits | 89.64± 1.44 | 93.80± 0.51 | 94.58 ± 0.53 | 91.60 ± 0.57 | **98.00 ±0.12** | 93.53±0.77 | 84.21±0.83 |
| Statlog | 83.67± 0.36 | 84.85± 0.84 | 83.65± 0.71 | 85.87± 0.86 | 87.28±0.12 | 85.42±0.38 | **87.93±0.93** |
| LR | 84.42±2.63 | 91.38±1.71 | 93.20±1.17 | 94.88±0.12 | 95.55±0.78 | 92.10±0.99 | **96.78±1.16** |
| GSAD | 89.37±1.82 | 95.78±0.63 | 95.77±0.17 | 98.89±0.59 | 99.07±0.36 | 97.42±0.43 | **99.76±0.14** |

*3)* *Comparison with the representative stacked autoencoders:* To verify the advantage of the proposed NE\_ESAE over the existing SAEs, some representative SAEs were considered for comparison. They included the stacked autoencoder (SAE), stacked sparse autoencoder (SSAE) [29], stacked denoising sparse autoencoder (SDSAE) [32], stacked pruning sparse autoencoder (SPSAE) [33], embedded stacked group sparse autoencoder ensemble with L1 regularization and manifold reduction (ESGSAE\_FF) [65] and gated stacked target-related autoencoder (GSTAE) [34]. Since most of the SAEs had tested ton he seven datasets, including AD, LSVT, PD, Urban, Vehicle, Pendigits, and Statlog, these datasets were also used for comparison. The experimental results are shown in Table VIII.

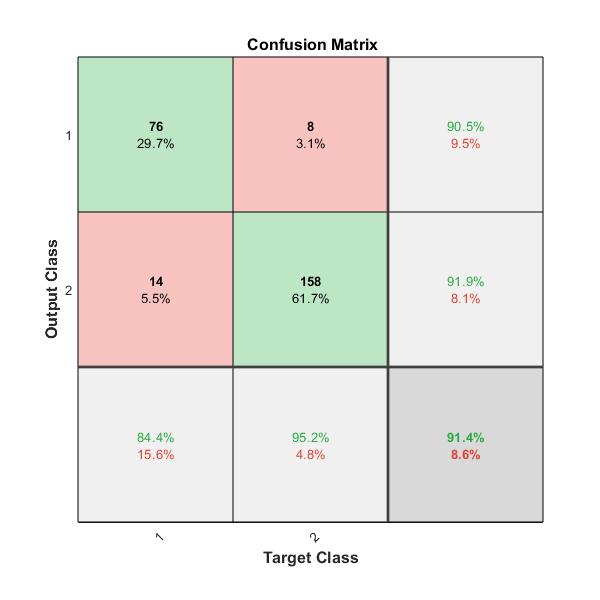
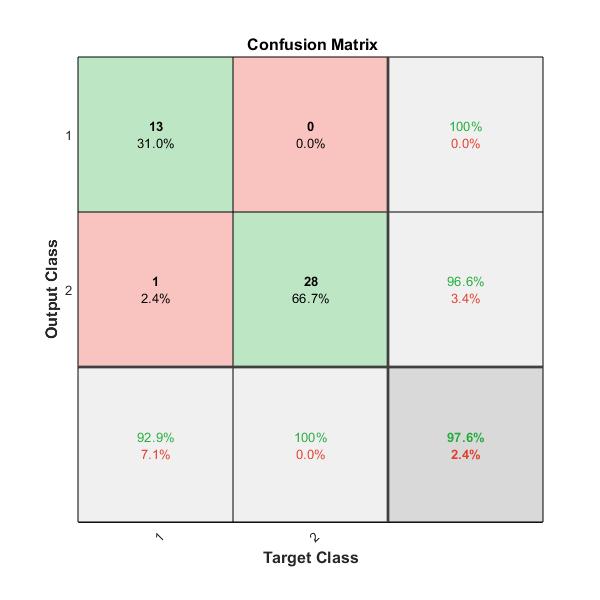
As Table VIII shows, the proposed NE\_ESAE has the best classification accuracy compared with other stacked autoencoders in most cases. For example, the classification accuracy of NE\_ESAE in AD dataset is 40.00% higher than SAE, 34.00% higher than SSAE, 35.09% higher than SDSAE, 32.89% higher than SPSAE, 23.34% higher than ESGSAE\_FF, and 19.56% higher than GSTAE, respectively. The experimental results prove that it is feasible and effective to consider the sample structure information and introduce the complementary ideas of original features and deep features in pretraining. For datasets with a small number of samples and high features, such as LSVT and Urban, the NE\_ESAE model still shows better performance.

****

(a) Maxlittle (b) Wisconsin

****

(c) Statlog (d) Urban

****

(e) PID (f) LSVT

Fig. 10. The confusion matrix under different datasets

## Parametric analysis

*1)* *Effects of classifier type:* The impact of different classifiers (extreme learning machines (ELM), SVM, and random forest (RF)) was experimentally studied. The results are presented in Table VIII.

|  |  |
| --- | --- |
| (a)AD | (b)PD |

Fig. 9. The effect of the cluster ratio

Table VIII shows that half of the datasets have the highest classification accuracy when SVM is used as the classifier. For example, on the AD dataset, the classification accuracy obtained by using SVM is 90.67%, which is 2.67% and 8.00% higher than those of RF and ELM, respectively, and has better stability. For dataset PD, the classification accuracy of SVM is the highest, with an accuracy of 78.74%, which is 1.07% higher than that of RF and 2.07% higher than that of ELM.

TABLE IX

Classification accuracy (mean ±variance) of the proposed

algorithm with different classifiers

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset | SVM (%) | RF (%) | ELM (%) |
| AD | **90.67±6.41** | 88.00±8.69 | 82.6714.61 |
| LSVT | 96.67±3.61 | **97.62±2.38** | 97.14±3.91 |
| PD | **78.74±4.22** | 77.67±5.25 | 76.67±6.30 |
| Vehicle | **89.53±1.96** | 80.00±8.11 | 80.99±7.34 |
| Pendigits | 84.21±0.83 | 82.78±1.79 | **85.17±1.35** |
| Statlog | **87.93±0.93** | 81.61±2.49 | 82.99±1.93 |
| Urban | **84.29±3.38** | 75.18±5.45 | 82.14±4.60 |
| heart | **99.44±1.11** | 98.89±1.28 | 98.33±2.13 |
| WDBC | **100.00±0.00** | **100.00±0.00** | 99.87±0.26 |
| Maxlittle | 96.92±1.09 | **98.75±1.71** | **98.75±1.71** |
| Wisconsin | **99.89±0.21** | 99.41±0.51 | 99.71±0.51 |
| PID | 89.38±1.88 | **89.84±2.47** | 89.38±2.25 |
| LR | 96.78±1.16 | **97..51±0.84** | 92.98±0.77 |
| GSAD | **99.76±0.14** | 99.72±0.09 | 97.42±0.10 |
| HAR | **99.70±0.11** | 98.16±0.22 | 94.56±0.67 |

For datasets with a large number of samples, such as Pendigits, the ELM algorithm has better performance than SVM and RF. The possible reason is that ELM is based on a feedforward neural network, which requires a certain number of samples to optimize the weights. Hence it can show better performance for large sample datasets. Overall, the classification performance of RF is between that of SVM and ELM, but one disadvantage of RF is that as the number of decision trees increases, its training process requires more time and space. On the other hand, by comparing the variance of the classification accuracy obtained from the twelve experiments in Table 10, it can be found that the variance is always the smallest when SVM is used. Overall, the experimental results show that the proposed algorithm has the best stability when SVM is used as the classifier.

*2)* *Effects of the cluster ratio:* This section studies the influence of the cluster ratio. The cluster ratio means the ratio of the number of samples in the sample space before and after clustering (i.e., the ratio of PS to OS and the ratio of SS to PS). The experiment was carried out on the AD and PD datasets. Considering the different numbers of samples in the AD and PD datasets, the cluster ratio of dataset AD was set as 1/3, 1/2, 2/3, and 3/4, and that of dataset PD was set as 1/4, 1/3, 1/2, and 2/3. The results are presented in Fig. 9. In the figure, the PS curve represents the model’s classification accuracy in the PS sample space, the SS curve represents the model’s classification accuracy in the SS sample space, and the MSEM curve represents the classification accuracy after fusion of the multilayer sample space by MSEM.

TABLE X

Comparison of complexity of different methods

|  |  |
| --- | --- |
| Method | SVM (%) |
| SAE |  |
| SSAE |  |
| SDSAE |  |
| SPSAE |  |
| ESGSAE\_FF |  |
| GSTAE |  |
| NE\_ESAE |  |

Fig. 9 shows that the cluster ratio has a significant impact on the experimental results. According to the MSEM curve, no matter how large is the cluster ratio selected, the classification accuracy after MSEM will be higher than that of PS and SS, indicating that the information contained in each sample space is complementary. As shown in Fig. 9 (a), the classification accuracy of PS and SS increases with increasing cluster ratio at the beginning. When the cluster ratio reaches 2/3, the classification accuracy is the best and then decreases. In addition, the classification accuracy of the MSEM increases with increasing cluster ratio at the beginning. When the cluster ratio reaches 1/2, the classification accuracy reaches the best value and then decreases. By comparing these three curves, we find that although the classification accuracy of PS and SS is the highest when the ratio is 2/3, the classification accuracy of MSEM is lower than that when the ratio is 1/2.

Fig. 9 (b) is similar as Fig. 9 (a). With the increase in the cluster ratio, the classification accuracy of these three curves decreases. When the cluster ratio reaches 1/3, the classification accuracy of PS and MSEM increases with increasing cluster ratio, and the classification accuracy of SS is the lowest at 2/3. However, the cluster ratio should not be too large; otherwise, it will increase the time complexity of the algorithm. In summary, we find that the algorithm performs best when the cluster of the dataset is 1/2, which makes the algorithm have high classification accuracy but does not increase the algorithm complexity.

## Confusion matrix

To better show the performance of the proposed NE\_ESAE model, we introduced the confusion matrix. We selected six datasets with apparent imbalanced distributions: Maxlittle, PID, LSVT, Statlog, Urban and Wisconsin. Part of the results are shown in Fig. 10.

As shown in Fig. 10 (a), the ratio of the two classes of samples in the dataset Maxlittle is 3:1. The prediction accuracy of the proposed model is 100% for the larger size class and 93.8% for the smaller size class. Wrong prediction only occurs in the second class of samples, and the main reason for the error is that the second class is predicted to be the first class. The prediction accuracy of the model for larger size classes is higher than that for smaller size classes. A similar pattern can be found in Fig. 10 (b) dataset Wisconsin. As seen from Fig. 10 (c), for the dataset Statlog, the overall accuracy is 84.7%, while the accuracy of a single class can reach 99.4%. The number of samples in the second and fifth classes is similar, but the classification accuracy is quite different, which are 94.5% and 50.2%, respectively. Likewise, as seen from Fig. 10 (d), for the dataset Urban, the overall accuracy is 84.4%, while the recognition rate for a single class can reach 97.1%. By comparing the classification accuracy of each classification class with the overall accuracy, it can be seen that the difference between the classification accuracy of a single class and the overall classification accuracy is small, and the proposed model has certain advantages as to the class imbalance problem.

TABLE XI

Comparison of training time and classification accuracy of different methods in seconds.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset | SAE | SSAE | SDSAE | SPSAE | ESGSAE\_FF | GSTAE | NE\_ESAE |
| AD | 11.07s (50.67%) | 17.00s (56.67%) | 15.80s (55.58%) | 18.50s (57.78%) | 28.90s (67.33%) | 13.60s (71.11%) | 37.70s (90.67%) |
| Urban | 165.2s (74.48%) | 365.8s (79.73%) | 178.0s (75.17%) | 225.0s (77.81%) | 320.0s (83.20%) | 126.2s (76.98%) | 178.7s (84.29%) |
| Vehicle | 34.88s (67.30%) | 50.09s (70.00%) | 43.21s (72.00%) | 150.9s (74.76%) | 130.8s (81.91%) | 30.22s (79.71%) | 125.7s (89.53%) |
| Statlog | 75.00s (83.67%) | 98.60s (84.85%) | 70.00s (83.65%) | 125.0s (85.87%) | 149.0s (87.28%) | 213.1s (85.42%) | 309.6s (87.93%) |
| LR | 860.7s (84.42%) | 1975s (91.38%) | 2039s (93.20%) | 1864s (94.88%) | 2487s (95.55%) | 642.7s (92.10%) | 117.9s (96.78%) |
| HAR | 600.8s (91.65%) | 17122s (98.82%) | 9208s (97.36%) | 8762s (98.13%) | 6318s (97.81%) | 4089s (98.22%) | 15245s (99.70%) |

## Computational Complexity and Time Analysis

*1)* *Computational complexity:* The computational complexity of the proposed approach can be computed as a sum of four tasks: i) complexity of SPC, ii) complexity of ICMC and iii) complexity of ESAE.

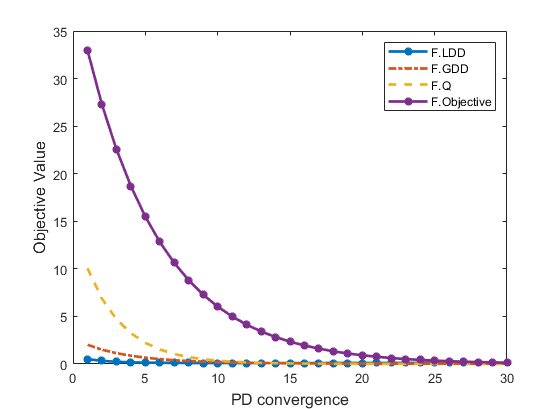
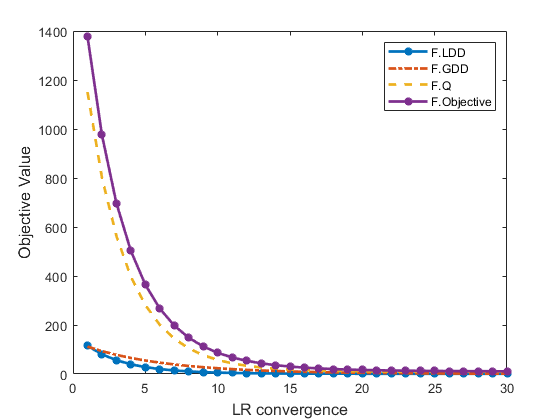


where the , and  is the time complexity of SPC, ICMC and ESAE.

i) The computational complexity of SPC is , where  is the number of samples.

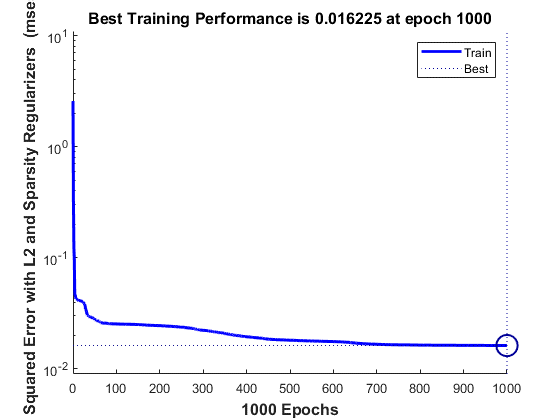
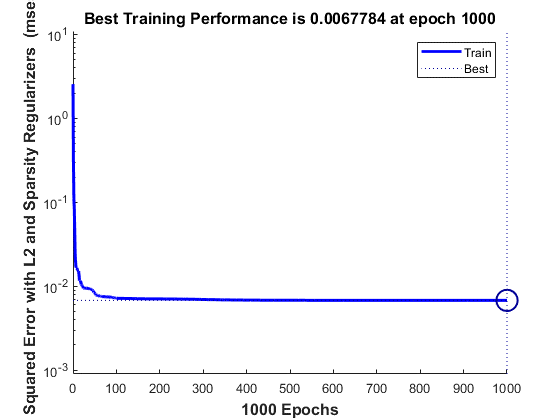
ii) The computational complexity of ICMC consists of four basic steps: IMC, update , update , and update . IMC is calculated as , where  is the number of iterations,  is the number of clustering clusters, and  is the number of samples. The complexity of computing and  is. The calculation of  involves feature decomposition and matrix multiplication, and the complexity is . Assuming that the number of iterations is , the total computational complexity of ICMC can be expressed as . It is worth noting that the computational complexity of the Gram matrix is not considered here, as it can be computed in advance without having to compute in ICMC.

iii) The computational complexity of ESAE is related to the number of hidden layer neurons, which are  respectively, so the computational complexity of ESAE is . At the same time, there is the back propagation part, whose computational complexity is , where  is the number of iterations,  is the number of samples, and  is the computational complexity of calculating the gradient of a single sample. So the total computational complexity of ESAE can be expressed as

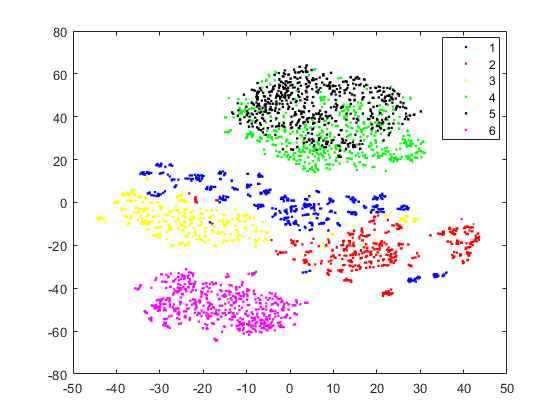
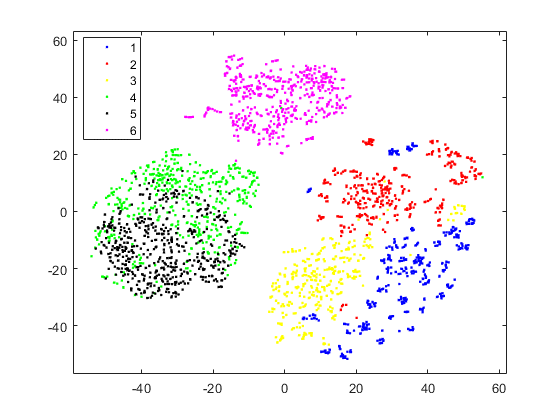
(a) PD (b) LR

Fig. 11. Convergence of ICM algorithm.

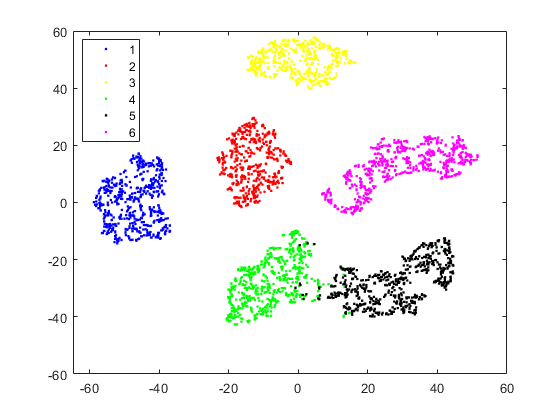
 

(a) PD (b) LR

Fig. 12. Convergence of ESAE algorithm.

(a) Original (b) NSELM



(c) NE\_ESAE

Fig. 10. Visualizations of original dataset and projected spaces obtained by NSELM, NE\_ESAE on HAR using t-SNE technique.

Therefore, the complexity of the proposed approach is the sum of all given as



Maximize the computational complexity, the computational complexity approximation of NE\_ESAE can be expressed as



Table X (Those in parentheses are classification accuracy) is a comparison of computational complexity between our algorithm and other autoencoder models. It can be seen from the table that although the complexity of our algorithm is the highest, it is similar to that of other methods. The algorithm complexity of our method is mainly added to the ICMC part, but it is worth it, because ICMC has deeply mined the hierarchical information between samples and improved the accuracy of sample classification.

*2) Time Analysis:* To further analyze the time complexity of our algorithm, we compare the training time and classification accuracy of NE\_ESAE with that of a typical deep encoder network, and these times are used to train different presentation learning tools other than classifiers. Also, all networks have the same number of hidden layers.

As shown in Table XI, the classification accuracy of NE\_ESAE is always significantly higher than that of other autoencoder models. Although the training time of NE\_ESAE is not the shortest, the improvement in its performance shows that the extra time cost is well worth it. For example, for small sample datasets, we can find that the accuracy of NE\_ESAE on AD dataset is 40%, 34%, 35.09%, 32.89%, 23.34% and 19.56% higher than that of SAE, SSAE, SDSAE, SPSAE, ESGSAE\_FF, GSTAESAE, respectively. However, the training time of NE\_ESAE was only 3.41%, 2.22%, 2.39%, 2.04%, 8.8% and 2.78% longer. For large sample datasets, it can be found that the accuracy of NE\_ESAE on LR dataset is 12.36%, 5.4%, 5.58%, 1.9%, 1.23% and 4.68% higher than that of SAE, SSAE, SDSAE, SPSAE, ESGSAE\_FF, GSTAESAE, respectively. However, the training time of NE\_ESAE was not the longest, only 1.40%, 1.10% and 1.83% longer than that of SAE, SSAE and GSTAE. Experimental results show that NE\_ESAE is superior to other encoder networks in representing deep feature categories. Although the training time is not the shortest, the improvement of classification accuracy proves that the improvement of time cost is meaningful.

## Model Visualization and Convergence Analysis

As can be seen from Figure 10 (a), many samples from different categories are mixed together and difficult to separate, which also leads to a decline in classification effect. It can be seen from Figure 10 (b) that the separability of samples processed by our NSELM mechanism is improved, the distance between similar samples is shorter, the distance between different categories is expanding, and the effect is more obvious from Figure10 (c). We can obviously see that after our method, the separability between different categories of samples is significantly enhanced, and the visualization result of NE\_ESAE almost retains the structure of the original dataset. The different categories of samples were well separated. The results show that the effectiveness of NE\_ESAE is obvious, and the visualization effect explains the effectiveness of our algorithm.

The convergence curves of NE\_ESAE on some datasets are shown in Figure 11 and Figure 12. Other datasets showed the same trend. In ICMC experiment, the number of iterations is set to 30, and the changes of the objective function equation (14) are shown in the figure. The objective function (14) can be expressed as, where is the auxiliary matrix. In addition, the convergence of the ICM method is discussed by observing the change of the objective function. It can be seen that by running the algorithm, the objective function is reduced to a constant value after several iterations on the dataset AD and the dataset LR. In addition, the convergence of all items in ICM, such as F.LDD (i.e.), F.GDD (i.e. ), F.Q (i.e. LRC regularization), is also shown in Figure 11. We can observe the fast convergence of each part after many iterations. For details, please refer to article [39].

In ESAE experiment, the number of iterations was set to 1000, and the changes of the objective function were shown in Figure 12. It can be seen that, by running the algorithm, the objective function is reduced to a constant value on the dataset PD and LR after several iterations. Figure 11 and Figure 12 ensure the convergence of our overall algorithm.

# Conclusion

SAE is a very important and widely used algorithm among them. However, existing SAEs only focus on original samples without considering the hierarchical structure between samples. This limits the construction of more powerful samples and the improvement of accuracy. To solve this problem, the NE\_ESAE model is proposed. First, the NSELM is proposed for preprocessing SAE. NSELM constructs a sample-pair and a multilayer sample space, which considers the hierarchical structure between neighboring and similar samples and generates layers of envelope samples with better quality. Second, ESAE is proposed and trained in each layer of the sample space to consider the original features during training and in the network structure, thereby better finding the relationship between the samples with original features and deep features. In summary, the proposed NE\_ESAE can consider the hierarchical structure between samples and obtain a multilayer of deep features, thereby overcoming the limitation of the existing SAEs.

Experimental results show that the proposed NE\_ESAE has obvious advantages over most algorithms. Taking the AD dataset as an example, the classification accuracy of NE\_ESAE is 40.00%, 34.00%, 35.09%, 32.89%, and 23.34% higher than that of the other SAEs. Besides, the influence of the classifier type and the cluster ratio in the IMC of the NE\_ESAE model was analyzed, which provides a reference for parameter setup.

This proposed NSELM method not only deeply mines the structural information of the data but also learns the structural information of each layer of samples, which increases the complementarity of the learned features and enhances the representation ability of the learned features. The ESAE proposed in this paper is a lightweight deep network. The network introduces the original features into the structure and training process of the autoencoder, which improves the complementarity between the deep features and the original features, thereby achieving high-quality features. By combining the NSELM and ESAE, the NE\_ESAE model is proposed to realize the sample-feature cooperative transformation. As seen from further analysis, the proposed NE\_ESAE model has the significant advantages of better generalization ability, more stability, and higher classification accuracy. In summary, different from the existing SAEs, the proposed NE\_ESAE realizes structured samples and the sample-feature cooperative transformation.

However, there are still some work to do. In the future, we plan to extend the proposed approach in two directions. One is to explore the network structure to enhance the feature learning ability, and anther is to incorporate more types of data.

Acknowledgments

The authors would like to thank the editors and reviewers for their valuable comments and suggestions. The authors would also like to thank those individuals or institutions that have provided support for this research.

References

1. Li R, Wang X, Quan W, *et al*. “Stacked Fusion Supervised Auto-encoder with an Additional Classification Layer,” *Neural Processing Lett.*, vol. 51, no.3, pp. 2649-2667, 2020.
2. Yang S, Zhang Y, Zhu Y, *et al*. “Representation learning via serial autoencoders for domain adaptation.” *Neurocomputing*, vol. 351, pp. 1-9, 2019.
3. Farajian N, Adibi P. “Minority manifold regularization by stacked auto-encoder for imbalanced learning.” *Expert Syst. Appl.* , vol. 169, pp. 114317, 2021.
4. Karim A M, Kaya H, Güzel M S, *et al.* “A novel framework using deep auto-encoders based linear model for data classification.” *Sensors*, vol. 20, no. 21, pp. 6378, 2020.
5. W. Wang, X. Du, D. Shan, R. Qin and N. Wang, “Cloud Intrusion Detection Method Based on Stacked Contractive Auto-Encoder and Support Vector Machine,” *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 1634-1646, Sept. 2022, DOI. 10.1109/TCC.2020.3001017.
6. Liu W, Ma T, Tao D, *et al*. “HSAE: A Hessian regularized sparse auto-encoders.” *Neurocomputing*, vol. 187, pp. 59-65, 2016.
7. Wang Y, Yang H, Yuan X, *et al*. “Deep learning for fault-relevant feature extraction and fault classification with stacked supervised auto-encoder.” *J. Process Control*, vol. 92, pp. 79-89, 2020.
8. T. Su, Y. Liu, J. Zhao and J. Liu, “Probabilistic Stacked Denoising Autoencoder for Power System Transient Stability Prediction With Wind Farms,” *IEEE Trans. Power Syst.* vol. 36, no. 4, pp. 3786-3789, July 2021, DOI. 10.1109/TPWRS.2020.3043620.
9. J. Yang *et al*., “No Reference Quality Assessment for Screen Content Images Using Stacked Autoencoders in Pictorial and Textual Regions,” *IEEE Trans Cybern.*, vol. 52, no. 5, pp. 2798-2810, May 2022, DOI. 10.1109/TCYB.2020.3024627.
10. X. Wang, K. Tan, Q. Du, Y. Chen and P. Du, “CVA2E: A Conditional Variational Autoencoder With an Adversarial Training Process for Hyperspectral Imagery Classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 8, pp. 5676-5692, Aug. 2020, DOI. 10.1109/TGRS.2020.2968304.
11. X. Xie, Z. Li, P. Zhang, and G. Zhang, “Information Structures and Uncertainty Measures in an Incomplete Probabilistic Set-Valued Information System,” *IEEE Access*, vol. 7, pp. 27501-27514, 2019.
12. F. Li, C. Shang, Y. Li, J. Yang, and Q. Shen, “Interpolation With Just Two Nearest Neighboring Weighted Fuzzy Rules,” *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 9, pp. 2255-2262, Sept. 2020.
13. Rodriguez A, Laio A. “Clustering by fast search and find of density peaks.” *Science*, vol. 344, no. 6191, pp. 1492-1496, 2014.
14. Y. Wang, S. T. Xia, Q. Tang, J. Wu, and X. Zhu, “A Novel Consistent Random Forest Framework: Bernoulli Random Forests,” *IEEE Trans. Neural Networks Learn. Syst*. vol. 29, no. 8, pp. 3510-3523, Aug. 2018.
15. Liangzhao Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam and H. Chang, “QoS-aware middleware for Web services composition,” *IEEE Trans. Software Eng.*, vol. 30, no. 5, pp. 311-327, May 2004, doi: 10.1109/TSE.2004.11.
16. R. Xu, and D. Wunsch, 2nd, “Survey of clustering algorithms,” *IEEE Trans. Neural Networks*, vol. 16, no. 3, pp. 645-78, May, 2005.
17. Cui S, Wang Y, Yin Y, *et al*. “A cluster-based intelligence ensemble learning method for classification problems,” *Inf. Sci.*, vol. 560, pp. 386-409, 2021.
18. M. -S. Yang and K. P. Sinaga, “A Feature-Reduction Multi-View k-Means Clustering Algorithm,” *IEEE Access*, vol. 7, pp. 114472-114486, 2019, DOI. 10.1109/ACCESS.2019.2934179.
19. Y. Lin, and S. Chen, “A Centroid Auto-Fused Hierarchical Fuzzy c-Means Clustering,” *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 7, pp. 2006-2017, July 2021, DOI. 10.1109/TFUZZ.2020.2991306.
20. S. Jianbing, H. Xiaopeng, L. Zhiyuan, L. Yu, W. Wenguan, and S. Ling, “Real-Time Superpixel Segmentation by DBSCAN Clustering Algorithm,” *IEEE Trans Image Process*, vol. 25, no. 12, pp. 5933-5942, Dec, 2016.
21. J. Zhang, X. Feng, and Z. Liu, “A Grid-Based Clustering Algorithm via Load Analysis for Industrial Internet of Things,” *IEEE Access*, vol. 6, pp. 13117-13128, 2018, DOI. 10.1109/ACCESS.2018.2797885.
22. S. Misbahulmunir, V. K. Ramachandaramurthy and Y. H. M. Thayoob, “Improved Self-Organizing Map Clustering of Power Transformer Dissolved Gas Analysis Using Inputs Pre-Processing,” *IEEE Access*, vol. 8, pp. 71798-71811, 2020, DOI. 10.1109/ACCESS.2020.2986726.
23. J. Maggu, A. Majumdar and E. Chouzenoux, "Transformed Subspace Clustering," *IEEE Trans. Knowl. Data En.*, vol. 33, no. 4, pp. 1796-1801, 2021, DOI: 10.1109/TKDE.2020.2969354.
24. J. Maggu, A. Majumdar and E. Chouzenoux, "Transformed Locally Linear Manifold Clustering," in *2018 26th EUSIPCO*, Rome, Italy, pp. 1057-1061,2018, DOI: 10.23919/EUSIPCO.2018.8553061.
25. Maggu J, Majumdar A. “Kernelized transformed subspace clustering with geometric weights for non-linear manifolds,” *Neurocomputing*, vol.520, pp. 141-151, 2023.
26. Li F, Zhang X, Wang P, *et al*. “Deep instance envelope network-based imbalance learning algorithm with multilayer fuzzy C-means clustering and minimum interlayer discrepancy,” *Appl. Soft Comput.*, vol. 123, pp. 108846, 2022.
27. Long M, Cao Y, Wang J, *et al*. “Learning transferable features with deep adaptation networks,” in *ICML.*, Lille, Fr., 2015, pp. 97-105.
28. B. Yang, Y. Lei, F. Jia, N. Li, and Z. Du, “A Polynomial Kernel Induced Distance Metric to Improve Deep Transfer Learning for Fault Diagnosis of Machines,” *IEEE Trans. Ind. Electron.*, vol. 67, no. 11, pp. 9747-9757, Nov. 2020, DOI. 10.1109/TIE.2019.2953010.
29. Praveen G B, Agrawal A, Sundaram P, *et al*. “Ischemic stroke lesion segmentation using stacked sparse autoencoder,” *Comput. Biol. Med.*, vol. 99, pp. 38-52, 2018.
30. Y. Shi, J. Lei, Y. Yin, K. Cao, Y. Li and C. -I. Chang, “Discriminative Feature Learning With Distance Constrained Stacked Sparse Autoencoder for Hyperspectral Target Detection,” *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 9, pp. 1462-1466, Sept. 2019, DOI. 10.1109/LGRS.2019.2901019.
31. L. Liu, Y. Wang, J. Peng, L. Zhang, B. Zhang and Y. Cao, “Latent Relationship Guided Stacked Sparse Autoencoder for Hyperspectral Imagery Classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 5, pp. 3711-3725, May 2020, DOI. 10.1109/TGRS.2019.2961564.
32. Görgel P, Simsek A. “Face recognition via deep stacked denoising sparse autoencoders (DSDSA),” *Appl. Math. Comput*., vol. 355, pp. 325-342, 2019.
33. Zhu H, Cheng J, Zhang C, *et al*. “Stacked pruning sparse denoising autoencoder based intelligent fault diagnosis of rolling bearings,” *Appl. Soft Comput.*, vol. 88, pp. 106060, 2020.
34. Q. Sun and Z. Ge, “Gated Stacked Target-Related Autoencoder: A Novel Deep Feature Extraction and Layerwise Ensemble Method for Industrial Soft Sensor Application,” *IEEE Trans. Cybern.,* vol. 52, no. 5, pp. 3457-3468, May 2022, DOI. 10.1109/TCYB.2020.3010331.
35. Martınez-Munoz G, Suárez A, “Aggregation ordering in bagging,” in *Proc. of the IASTED International Conference on Artificial Intelligence and Applications*. Citeseer, Innsbruck, Austria, 2004, pp. 258-263
36. Mika S, Ratsch G, Weston J, *et al*. “Fisher discriminant analysis with kernels,” in *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98th8468)*, 1999, pp. 41-48.
37. Q. Qu, Y. Zhang, Y. C. Eldar and J. Wright, “Convolutional Phase Retrieval via Gradient Descent,” *IEEE Trans. Inf. Theory*, vol. 66, no. 3, pp. 1785-1821, March 2020, DOI. 10.1109/TIT.2019.2950717.
38. Cai J F, Candès E J, Shen Z. “A singular value thresholding algorithm for matrix completion,” SIAM J. Optim., vol.20, no. 4, pp. 1956-1982, 2010.
39. L. Zhang, S. Wang, G. -B. Huang, W. Zuo, J. Yang and D. Zhang, “Manifold Criterion Guided Transfer Learning via Intermediate Domain Generation,” IEEE Trans. Neural Networks Learn. Syst., vol. 30, no. 12, pp. 3759-3773, Dec. 2019, DOI. 10.1109/TNNLS.2019.2899037.
40. F. Xie, Z. Pang, H. Wen, W. Lei and X. Xu, “Weighted Voting in Physical Layer Authentication for Industrial Wireless Edge Networks,” *IEEE Trans. Ind. Inf*., vol. 18, no. 4, pp. 2796-2806, April 2022, DOI. 10.1109/TII.2021.3103780.
41. M. Bibi, W. Aziz, M. Almaraashi, I. H. Khan, M. S. A. Nadeem and N. Habib, "A Cooperative Binary-Clustering Framework Based on Majority Voting for Twitter Sentiment Analysis," *IEEE Access*, vol. 8, pp. 68580-68592, 2020, DOI. 10.1109/ACCESS.2020.2983859.
42. Tan X, Liu Y, Li Y, *et al*. “Localized instance fusion of MRI data of Alzheimer’s disease for classification based on instance transfer ensemble learning,” *Biomed. Eng. Online*, vol. 17, no. 1, pp. 1-17, 2018.
43. A. Tsanas, M. A. Little, C. Fox and L. O. Ramig, “Objective Automatic Assessment of Rehabilitative Speech Treatment in Parkinson's Disease,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 1, pp. 181-190, Jan. 2014, DOI. 10.1109/TNSRE.2013.2293575.
44. B. E. Sakar *et al*., “Collection and Analysis of a Parkinson Speech Dataset With Multiple Types of Sound Recordings,” *IEEE J. Biomed. Health. Inf.*, vol. 17, no. 4, pp. 828-834, July 2013, DOI. 10.1109/JBHI.2013.2245674.
45. Hoti F, Holmström L. “A semiparametric density estimation approach to pattern classification,” *Pattern Recognit.*, vol. 37, no.3, pp. 409-419, 2004.
46. Dua, D. Graff, C. UCI Machine Learning Repository, University of California, School of Information and Computer Science. Irvine, CA, 2019. [Online]. Available: http://archive.ics.uci.edu/ml.
47. M. A. Little, P. E. McSharry, E. J. Hunter, J. Spielman and L. O. Ramig, “Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's Disease,” *IEEE Trans. Biomed. Eng.*, vol. 56, no. 4, pp. 1015-1022, April 2009, DOI. 10.1109/TBME.2008.2005954.
48. Johnson B, Xie Z. “Classifying a high resolution image of an urban area using super-object information,” *ISPRS J. Photogramm. Remote Sens.*, vol. 83, pp. 40-49, 2013.
49. Wolberg W H, Street W N, Mangasarian O L. “Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates,” Cancer Lett., vol. 77, no. 2-3, pp. 163-171,1994.
50. Wolberg W H, Mangasarian O L. “Multisurface method of pattern separation for medical diagnosis applied to breast cytology,” *PNAS*, vol. 87, no. 23, pp. 9193-9196, 1990.
51. Smith J W, Everhart J E, Dickson W C, *et al*. “Using the ADAP learning algorithm to forecast the onset of diabetes mellitus,” in *Proceedings of the annual symposium on computer application in medical care*, 1988, pp. 261.
52. I Rodriguez Lujan, J Fonollosa, A Vergara, M Homer, R Huerta. "On the calibration of sensor arrays for pattern recognition using the minimal number of experiments." *Chemometr Intell Lab Syst.* no.130, pp. 123-134,2014.
53. Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. “A Public Domain Dataset for Human Activity Recognition Using Smartphones,” in *ESANN* vol.3,2013.
54. A. -K. Seghouane, N. Shokouhi and I. Koch, “Sparse Principal Component Analysis With Preserved Sparsity Pattern,” *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3274-3285, July 2019, DOI. 10.1109/TIP.2019.2895464.
55. C. -H. Li, B. -C. Kuo and C. -T. Lin, “LDA-Based Clustering Algorithm and Its Application to an Unsupervised Feature Extraction,” *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 1, pp. 152-163, Feb. 2011, DOI. 10.1109/TFUZZ.2010.2089631.
56. R. Wang, F. Nie, R. Hong, X. Chang, X. Yang and W. Yu, “Fast and Orthogonal Locality Preserving Projections for Dimensionality Reduction,” *IEEE Trans. Image Process.*, vol. 26, no. 10, pp. 5019-5030, Oct. 2017, DOI. 10.1109/TIP.2017.2726188.
57. Z. Deng, F. -L. Chung and S. Wang, “Robust Relief-Feature Weighting, Margin Maximization, and Fuzzy Optimization,” *IEEE Trans. Fuzzy Syst*., vol. 18, no. 4, pp. 726-744, Aug. 2010, DOI. 10.1109/TFUZZ.2010.2047947.
58. J. B. O. Souza Filho and P. S. R. Diniz, "Improving KPCA Online Extraction by Orthonormalization in the Feature Space," *IEEE Trans Neural Netw Learn Syst*, vol. 29, no. 4, pp. 1382-1387, April 2018, DOI: 10.1109/TNNLS.2017.2660441.
59. F. Peng, W. Peng, C. Zhang and D. Zhong, "IoT Assisted Kernel Linear Discriminant Analysis Based Gait Phase Detection Algorithm for Walking with Cognitive Tasks," *IEEE Access*, vol. 7, pp. 68240-68249, 2019, DOI: 10.1109/ACCESS.2019.2915290.
60. Z. Miao, H. Zhang and S. Ma, "Nonlinear Kernel Dictionary Learning Algorithm Based on Analysis Sparse Model," *IEEE Access*, vol. 8, pp. 12456-212466, 2020, DOI: 10.1109/ACCESS.2020.3038695.
61. Maggu J, Majumdar A. “Kernel transform learning,” *Pattern Recognition Letters*, vol. 98, pp.117-122, 2017.
62. Maggu J, Majumdar A. “Supervised kernel transform learning,” in *IJCNN*. *IEEE*, pp. 1-8, 2019.
63. Yamada M, Jitkrittum W, Sigal L, *et al*. “High-dimensional feature selection by feature-wise kernelized lasso,” *Neural Comput.*, vol. 26, no. 1, pp. 185-207, 2014.
64. Donoho D, Jin J. “Higher criticism thresholding: Optimal feature selection when useful features are rare and weak,” *Proceedings of the National Academy of Sciences*, vol.105, no.39, pp. 14790-14795, 2008.
65. Y. Li, Y. Lei, P. Wang, M. Jiang, and Y. Liu, “Embedded stacked group sparse autoencoder ensemble with L1 regularization and manifold reduction,” *Appl. Soft Comput.*, vol. 101, 2021.

1. This work was sup-ported in part by the National Natural Science Foundation of China (NSFC) under grants 61771080 and U21A20448, the Key Project of Technology Innovation and Application De-velopment in Chongqing (cstc2019jsxz-mbdxX0050), the Natural Science Foundation of Chongqing (cstc2020jscx-msxm0369, cstc2020jcyj-msxmX0100, cstc2020jscx-fyzx0212, cstc2020jcyj-msxmX0523, cstc2020jscx-gksbx0009, the Chongqing Social Science Plan-ning Project (2018YBYY133), and the fund of Sichuan (21ZDYF3646). (Corresponding author: Yongming Li.)

   Chuanyan Zhou is with the School of Microelectornics and Communication Engineering, Chongqing University, Chongqing, 400044, P. R. China (e-mail: 1347289136@qq.com).

   Jie Ma is with the School of Microelectornics and Communication Engineering, Chongqing University, Chongqing, 400044, P. R. China (e-mail:  [609290268@qq.com](mailto:979940181@qq.com)).

   Fan Li is with the School of Microelectornics and Communication Engineering, Chongqing University, Chongqing, 400044, P. R. China (e-mail: [979940181@qq.com](mailto:979940181@qq.com)).

   Yongming Li is with the School of Microelectornics and Communication Engineering, Chongqing University, Chongqing, 400044, P. R. China (e-mail: yongmingli@cqu.edu.cn).

   Pin Wang is with the School of Microelectornics and Communication Engineering, Chongqing University 400044, P. R. China (e-mail: wangpin@cqu.edu.cn). [↑](#footnote-ref-1)