Soft-orthogonal constrained dual-stream encoder with self-supervised clustering network for brain functional connectivity data

Hu Lu[a] (luhu@ujs.edu.cn), TingTing Jin[a] (2222008028@stmail.ujs.edu.cn), Hui Wei[b] (weihui@fudan.edu.cn), Michele Nappi[c] (mnappi@unisa.it), Hu Li[d] (20180042@lixin.edu.cn) , ShaoHua Wan[e] (shaohua.wan@uestc.edu.cn)

[a] School of Computer Science and Communication Engineering, Jiangsu University, Jiangsu 212013, China

[b] School of Computer Science, Fudan University, Shanghai 200433, China

[c] Department of Computer Science, University of Salerno, Salerno, Italy

[d] School of information management,Shanghai Lixin University of Accounting and Finance, Shanghai 201209, China

[e] Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China

Corresponding Author:

ShaoHua Wan

Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China.

Email: shaohua.wan@uestc.edu.cn

Soft-orthogonal constrained dual-stream encoder with self-supervised clustering network for brain functional connectivity data

Hu Lu[a], TingTing Jin[a], Hui Wei[b], Michele Nappi[c], Hu Li[d], ShaoHua Wan[e,*]

[a]*School of Computer Science and Communication Engineering, Jiangsu University, Jiangsu 212013, China*
[b]*School of Computer Science, Fudan University, Shanghai 200433, China*
[c]*Department of Computer Science, University of Salerno, Salerno, Italy.*
[d]*School of information management,Shanghai Lixin University of Accounting and Finance, Shanghai 201209, China*
[e]*Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China*

Abstract

In many brain network studies, brain functional connectivity data is extracted from neuroimaging data and then used for disease prediction. For now, brain disease data not only has a small sample but also has the problem of high dimensional and nonlinear. Therefore, deep clustering on brain functional connectivity data is very challenging. To solve these problems, we propose a Soft-orthogonal Constrained Dual-stream Encoder with Self-supervised clustering network (SSCDE), which consists of a pretext task and downstream task, which can fully mine the effective information in brain disease data. In the pretext task, we use two brain disease data under the same category to do cross-domain learning to obtain effective information from the same dataset. In the downstream task, to reduce redundancy and avoid negative coding, we propose a soft-orthogonal constrained dual-stream encoder to encode features separately. At the same time, we use the pseudo labels given by the pretext task as prior information for self-supervised learning. We conduct validation on different brain disease

*ShaoHua Wan

Email addresses: luhu@ujs.edu.cn (Hu Lu), 2222008028@stmail.ujs.edu.cn (TingTing Jin), weihui@fudan.edu.cn (Hui Wei), mnappi@unisa.it (Michele Nappi), 20180042@lixin.edu.cn (Hu Li), shaohua.wan@uestc.edu.cn (ShaoHua Wan)

recognition tasks, and the result have proved that the proposed framework has achieved good performance compared with the unsupervised clustering analysis algorithms. To our knowledge, this is the first cross-domain assisted recognition study on brain functional connectivity data.
Keywords: brain functional connectivity data, graph convolutional networks, deep clustering

---

1. Introduction

Resting-state functional Magnetic Resonance Imaging (rs-fMRI) (Cox & Savoy, 2003; Vergun et al., 2016; Biswal et al., 1995) is a powerful neuroimaging technique to study human brain function and dysfunction, widely used to diagnose and identify brain diseases. Brain functional connectivity data, generated from magnetic resonance imaging, is represented by nodes and edges. Nodes represent Regions Of Interest (ROI) predefined by relevant templates, and edges represent functional connectivity relationships between ROI. Directly analyzing brain functional connectivity data and building a brain functional connectivity network for analysis are common research methods that can be used to diagnose brain diseases at this stage. In recent years, many studies have used brain functional connectivity data (Cui et al., 2018) for unsupervised analytical studies. If the traditional clustering method (Ng et al., 2001; Baumgartner et al., 1997) is used directly on the brain functional connectivity data, the ideal clustering performance often cannot be achieved. Thanks to the popularity of deep learning, traditional clustering methods are gradually being replaced by deep learning methods. Researchers extract low-dimensional features based on the deep learning method, then cluster them. Therefore, how to combine deep learning with clustering (Guo et al., 2017; Caron et al., 2018) for the study of brain diseases is of great significance.

Although the clustering algorithm based on deep learning has become a research hotspot, how to extract the ideal feature information under the curse of dimensionality is still a process that requires continuous research. Most deep

learning methods use autoencoders (Ji et al., 2017; Ng et al., 2011) to find the optimal low-dimensional subspace. Still, they are often unable to do so for high-dimensional and nonlinear data, resulting in unsatisfactory results. In particular, some methods utilize multiple encoding networks (Zhuang & Ma, 2018) to extract as much implicit information as possible from the data but do not consider the redundant information generated between these encoders, which leads to adverse effects.

In recent years, the emergence of self-supervised learning ideas (Misra & Maaten, 2020; Liu et al., 2021) has enhanced the extensive analytical ability of unsupervised analysis, and self-supervised learning can further improve the learning ability by fully excavating the inner expression of characteristics.

DENs-SCC (Lu & Jin, 2022) proposes a dual-stream encoder model to mine different information from brain functional connectivity data to achieve fusion coding. This paper implements an improvement based on it. Considering the possibility of redundant information being extracted by the dual-stream encoders, we imposed orthogonal constraints on these two encoders to learn dissimilar information and ensure they are complementary and unique. Additionally, self-supervised learning and cross-domain learning can introduce prior knowledge. Building upon this, we designed an auxiliary learning upstream model to assist the primary model in better understanding the intrinsic information within the data, thereby facilitating the further exploration of deeper-level features. The proposed framework can be divided into two parts.

The first part is the pretext task. Inspired by CDNE (Shen & Chung, 2019) and MEDA (Wang et al., 2018), we propose a pretext task based on Unsupervised Domain Adaptation (UDA). The pretext task transfers knowledge by reducing the distribution difference between the source and target domains. Specifically, due to the different feature dimensions of brain disease datasets, in order to bridge the feature space gap, we map data from the source and target domains to a common shared space. At the same time, in this shared feature space, the data distribution of the source and target domains is matched (Long et al., 2013) to complete knowledge transfer. After training, we cluster the fea-

tures of the extracted target domain to generate trusted pseudo labels. These pseudo-labels can be used as label information in downstream tasks to guide the training of the model.

The second part is the downstream task. Inspired by SDCN (Bo et al., 2020) and (Yang et al., 2019), we propose a soft-orthogonal constrained dual-stream encoder with self-supervised network. The dual-stream encoder encodes the raw features to capture more prosperous and effective features. Specifically, we utilize the stacked encoder to capture in-depth features. At the same time, graph convolutional networks (GCN) (Kipf & Welling, 2016a) are used to capture node-level information. We attach a soft-orthogonal network to constrain the dual-stream encoder to generate dissimilar information. Then, we utilize the decoder to minimize the reconstruction loss, which helps preserve the local structure of the data points. Meanwhile, we use a spectral network to guide the embedding node for clustering optimization and a classifier based on pseudo labels of the pretext task to self-supervise the embedding node. Finally, the label information of the brain functional connectivity data is obtained.

In summary, our main contributions are summarized as follows:

- We propose a soft-orthogonal constrained dual-stream encoder framework with self-supervised network for clustering brain functional connectivity data. It can be divided into a pretext task and a downstream task. We demonstrate that the framework can improve learning ability through joint learning and improve clustering performance.

- We propose a soft-orthogonal encoding network that utilizes the orthogonality-constrained dual-stream encoder to generate features with dissimilar information, reducing the redundancy of generated information.

- We propose a pretext task based on the unsupervised domain adaptive method to assist the downstream task in self-supervised learning, which utilizes its information to supervise the model to generate valuable representations.

- This framework is still an unsupervised recognition task because it does not need the real labels of the raw data. To show the performance of the proposed framework, we have performed multiple validations on eight public brain disease datasets to prove that the algorithm proposed in this paper shows excellent performance.

2. Related work

2.1. Deep clustering methods

Compared with traditional clustering methods, clustering algorithms based on deep learning have apparent advantages in complex data. Deep clustering is done after mapping raw features into the low-dimensional subspace, which has been favored by many researchers and has become a current research hotspot. DEC (Xie et al., 2016) proposed a classical deep clustering algorithm, which implemented an end-to-end deep clustering model based on the autoencoder, and performed k-means clustering after learning low-dimensional and high-quality features. In the graph-encoder method, Tian *et al.* (Tian et al., 2014) proposed a graph clustering method based on deep learning, which learned the nonlinear embedding of the original graph through a stack encoder, then clusters the features, indicating the value of graph clustering in deep learning. DAEGC (Wang et al., 2019) utilized a graph attention network for encoding topology and node information according to neighboring nodes' importance and used a clustering module to generate soft labels to supervise the update of embedded nodes. AGE (Cui et al., 2020a) used an unsupervised graph representation learning method to design a filter to remove high-frequency noise and an adaptive encoder to learn better embedding features. AdaGAE (Li et al., 2021) used the popular regularization method to preserve the local structure of the data to design a new way of learning graph data. AdaGAE believes that the strategy of constructing graphs plays a vital role in clustering performance. Furthermore, ARGE (Pan et al., 2018) realized a way to represent graph data in a low-dimensional space from an adversarial learning perspective. DSCNSS (Chen

et al., 2022) utilizes clustering labels in subspace to guide deep networks.

Self-supervised learning mainly uses the pretext task to mine its supervised information from unsupervised data, and the info constructed in this way provides valuable information for the downstream task. There are many ways to implement self-supervised learning. Some researchers have adopted the method of context-instance contrast for learning. For example, DGI (Velickovic et al., 2019) used node representations as local features and used the mean value of randomly sampled two-hop neighbor nodes as context vectors, retaining the structure of the original context nodes but generating negative samples by shuffling the order. Hendrycks *et al.* (Hendrycks et al., 2019) combined adversarial learning with self-supervision to improve the robustness of the model. Cluster-based methods like learning by assigning pseudo labels to samples. For example, Caron *et al.* (Caron et al., 2018) generated pseudo labels through k-means clustering. In addition, S2ConvSCN (Zhang et al., 2019) introduced dual self-supervision utilizing spectral clustering output to supervise the learned features. IDCEC (Lu et al., 2022) proposed a new reliable sample selection mechanism based on self-supervised learning for deep clustering.

2.2. Brain functional connectivity data clustering methods

Various clustering learning methods have been applied to the clustering analysis of brain functional connectivity data. Traditional clustering methods, such as k-means (MacQueen et al., 1967) and spectral clustering (Snyder et al., 1997), are often difficult to achieve ideal clustering effects on high-dimensional nonlinear data. Borlea *et al.* (Borlea et al., 2022) proposed a new quality improvement method for k-means clustering, and the proposed method can significantly improve the performance of complex datasets and high-dimensional datasets. Zhao *et al.* (Zhao et al., 2019) used hierarchical clustering, Ordering Points To Identify the Clustering Structure (OPTICS), and Density Peak Clustering (DPC) methods to cluster brain functional connectivity data. As far as brain structure data is concerned, it is highly complex. Deep neural networks can mine intrinsic representation information.

Brain functional connectivity data suffers from high dimensionality and non-linear structure, and the use of neural network models allows for the processing of similarly complex data (Tan et al., 2014). For example, Pehlivan *et al.* (Pehlivan & Turksen, 2021) proposed a multiplicative fuzzy regression function. The regression function uses a multiplicative model that can better accommodate the non-linear relationships present in the data. Also, the clustering algorithm based on this function allows for the clustering of data taking into account the ambiguity between the data. These methods are of wide value in dealing with the complex problems that exist in the real world.

Over the years, many researchers have used deep learning to study brain disease data. Suk *et al.* (Suk et al., 2015) implemented unsupervised learning of brain diseases using stack autoencoders. Cui *et al.* (Cui et al., 2020b) learned the global and local information of brain disease networks based on measuring attribute similarity and structural similarity, respectively, and then integrated them into a matrix and performed spectral clustering. The clustering performance has been dramatically improved through the deep learning method. DMACN (Lu et al., 2021) proposed a deep multicore autoencoder with a self-expression layer capable of training neural networks that tend to cluster. Lu *et al.* (Lu et al., 2020) used multiple hidden information of the stack encoder to build different kernels and realizes the fuzzy multi-kernel clustering method of the autoencoder. DENs-SCC (Lu & Jin, 2022) is able to mine node-level information and discriminative information of the data separately and encode them together while performing dimensionality reduction. In this way, deep and effective features can be extracted in the low-dimensional space, which allows for better clustering optimisation in the model. Therefore, it is crucial to fully mine valuable representations from brain functional connectivity data using deep neural networks.

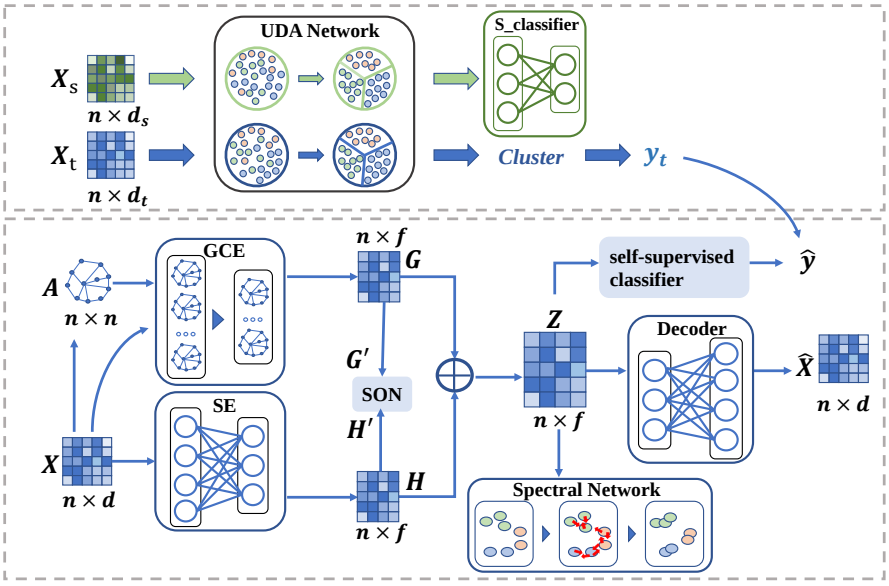


Figure 1: The framework of the proposed model. SSCDE is divided into two parts: the pretext task and the downstream task. In the pretext task, input data are the source dataset X_s and the target dataset X_t. Green represents the training process of X_s, and blue represents the training process of X_t (same as X of the downstream task). The pretext task generates pseudo labels y_t and sends them to the downstream task for self-supervised learning. In the downstream task, input data are the target dataset X and the adjacency matrix A. SON performs orthogonal constraints on GCE and SE to generate features G, H, and the embedded feature Z is calculated by G and H. The spectral network performs cluster-guided optimization of the embedding matrix Z. The decoder reconstructs the raw data X to generate $\hat{X}$.

3. Proposed method

It is important that different models and algorithms are used to solve specific problems in different application scenarios (Pozna & Precup, 2012; Verma et al., 2022; Precup et al., 2022). Brain functional connectivity data is high-dimensional and nonlinear, and learning effective feature representations in brain networks is challenging and meaningful. In this section, for the problem of brain functional connectivity data, we propose a soft-orthogonal constrained dual-stream encoder with self-supervision. The overall framework is shown in

Fig. 1. The overall framework consists of a pretext task and a downstream task. The pretext task is based on UDA for cross-domain learning, and the features of the extracted target domain are clustered to generate pseudo labels. The downstream task mainly consists of four modules: Soft-orthogonal Encoding Network (SEN), Decoder module, Self-supervised module, and Spectral Network.

3.1. Problem definition

Given a labeled source domain $D_s = \{X_{s_i}, \boldsymbol{y_{s_i}}\}_{i=1}^{n_s}$ and an unlabeled target domain $D_t = \{X_{t_j}\}_{j=1}^{n_t}$ under the assumptions that marginal distributions $P_s(X_s) \neq P_t(X_t)$, conditional distributions $Q_s(\boldsymbol{y_s} \mid X_s) \neq Q_t(\boldsymbol{y_t} \mid X_t)$. X_s represents the features of the source domain, and X_t represents the features of the target domain. n_s and n_t represent the number of samples in the source domain and target domain, respectively.

Uppercase and lowercase letters denote matrices and vectors, respectively. The graph can be represented as $G =< V, E, W >$, where V represents the set of vectors, each vertex is a sample, E represents the set of edges, and W_{ij} represents the weight between the edges, that is, the similarity between node i and node j. A is an adjacency matrix. For each node $\boldsymbol{v_i} \in V$, it can be represented by $\boldsymbol{x_i} \in R^d$, that is, $V = [\boldsymbol{x_1}, \boldsymbol{x_2} \ \ldots \ \boldsymbol{x_n}] \in R^{N \times d}$. X represents the sample feature matrix, n represents the number of samples, c represents the number of cluster classes.

3.2. The pretext task

Datasets related to brain diseases often suffer from the drawback of small sample size, and extracting useful feature information from such limited data poses a significant challenge. In the domain adaptive method, source domains and target domains' feature space and category space are required to be consistent (Long et al., 2013), but the distributions are different. Benefiting from the fact that each allogeneous brain disease had two datasets, we try to learn effective feature information from the same type of dataset by UDA. First, to

solve the problem of the same feature space, we reduce the dimensionality of the source and target domains so that they can align their features. The encoding rules are as follows:

$$H_s^{(l)} = \phi(W_s^{(l)} H_s^{(l-1)} + b_s^{(l)}), l = 1 \dots mid_s, \tag{1}$$

$$H_t^{(l)} = \phi(W_t^{(l)} H_t^{(l-1)} + b_t^{(l)}), l = 1 \dots mid_t, \tag{2}$$

where, $H_s^{(0)} = X_s \in R^{n_s \times d_s}$, $H_t^{(0)} = X_t \in R^{n_t \times d_t}$, $d_s \neq d_t$, $H_s^{(l)}$ and $H_t^{(l)}$ are the representation learned by the *l-th* layer in the UDA network, $H_s^{(mid_s)} \in R^{n_s \times f_s}$, $H_t^{(mid_t)} \in R^{n_t \times f_t}$, $f_s = f_t$. ϕ is a non-linear activation function.

Next, to transfer the knowledge in the source domain into the target domain, we use Maximum Mean Difference (MMD) to compute the distributional difference between the two domains empirically. The essence of MMD is to find a transformation function so that the distance between the transformed source domain and target domain data becomes smaller. In the new feature space, we use Eq. 3 to calculate the marginal distribution difference between the source and target domains and Eq. 4 to calculate the conditional distribution difference between the source and target domains.

$$\mathcal{L}_{MMD} = \| \frac{1}{n_s} \sum_{i=1}^{n_s} \Phi(X_{s_i}) - \frac{1}{n_t} \sum_{j=1}^{n_t} \Phi(X_{t_j}) \|_H^2, \tag{3}$$

$$\mathcal{L}_{CMMD} = \sum_{c=1}^{C} \| \frac{1}{n_s^c} \sum_{i=1}^{n_s^c} \Phi(X_{s_i}^c) - \frac{1}{n_t^c} \sum_{j=1}^{n_t^c} \Phi(X_{t_j}^c) \|_H^2, \tag{4}$$

where, X_s^c and X_t^c denote the feature of source and target data belonging to the *c-th* class, respectively. n_s^c and n_t^c are the numbers of the corresponding features. Φ ($\cdot$) is a function that maps features into Hilbert space satisfying the condition of $< K(x, \cdot), K(\cdot, y) >_H = K(x, y)$.

When the source domain features are extracted through the UDA network, we use the cross-entropy loss function in the source domain classifier ($S\_classifier$) to label all the samples in the source domain. The loss function

is shown in Eq. 5.

$$\mathcal{L}_{s\_cls} = -\frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{k=1}^{C} y_{s_{ik}} \log \hat{y}_{s_{ik}}, \tag{5}$$

where, $\boldsymbol{y_s}$ represents the ground-truth label of the source domain, and $\hat{\boldsymbol{y}}_{\boldsymbol{s}}$ represents the predicted label generated by the $S\_classifier$.

After training, the low-dimensional target domain features are extracted, clustered, and the generated pseudo labels y_t are sent to the downstream task.

3.3. The downstream task

The downstream task consists of four modules: Soft-orthogonal Encoding Network (SEN), Decoder module, Self-supervised module, and Spectral Network.

First, the SEN is divided into three parts: the Graph Convolutional Encoder (GCE), the Stack Encoder (SE), and the Soft-Orthogonal Network (SON). GCN (Kipf & Welling, 2016a) can learn a good feature representation for raw data by using topological structure, and it has become a popular research tool. In GCE, in order to use the relationship between nodes to learn a good feature representation, we use GCN to learn node-level information, and the encoding rules are:

$$G^{(l)} = \phi(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}G^{(l-1)}W_g^{(l)} + b_g^{(l)}), l = 1, 2 \dots mid_1, \tag{6}$$

where, $G^{(l)}$ is the representation learned by the *l-th* layer in the GCE and $G^{(0)} = X$. ϕ is a non-linear activation function. $\hat{A} = A + I$ is the adjacency matrix with self-connection added and $\hat{D}_{ii} = \sum_j \hat{A}_{ij}$. mid_1 is the number of network layers of graph convolution, and $W_g^{(l)}$, $b_g^{(l)}$ respectively represent the weight and bias of the *l-th* layer in the GCE.

Autoencoders can learn the latent features of the input data and play an important role in the unsupervised domain. In the SE, in order to extract low-dimensional and highly correlated features, we use a stack encoder to encode the original nodes. The encoding rules are:

$$H^{(l)} = \phi(W_e^{(l)}H^{(l-1)} + b_e^{(l)}), l = 1, 2 \dots mid_2, \tag{7}$$

where, $H^{(l)}$ is the representation learned by the $l\text{-}th$ layer in the SE and $H^{(0)} = X$. ϕ is a non-linear activation function. mid_2 is the number of layers of the stack encoder, $W_e^{(l)}$, $b_e^{(l)}$ respectively represent the weight and bias of the $l\text{-}th$ layer in the SE.

In SON, in order to reduce redundancy and avoid negative encoding, we impose an orthogonal constraint on G, H. However, if G, H are directly orthogonal, it will be too violent. To reduce the negative encoding effect of brute force constraints (two encoders will make some attributes of the feature have a direct membership of 0. In other words, when an important attribute of G, H directly returns to 0, it will cause negative encoding), we design a soft orthogonal network indirect constraint that allows G, H to have an orthogonal constraint tendency. The rules we impose on SON are as follows:

$$L_{soft} = G^{'T} H^{'}, \tag{8}$$

where, $G^{'}$ and $H^{'}$ are the features generated by G, H in the soft-orthogonal network.

After the soft-orthogonal encoding network encoding, the features G and H extracted are mapped to a new subspace using Eq. 9.

$$Z = \alpha G + (1 - \alpha) H, \tag{9}$$

where Z is the embedded node in the low-dimensional space and α is the weight coefficient.

Second, in an autoencoder, the decoder plays an important role. The decoder can try to recover raw data and use the value of the loss function to backpropagate so that the embedding node retains the important information of raw data. Therefore, in order to reconstruct the raw input data, preserving the local structure information, in the decoder, we utilize Eq. 10 to encode the network.

$$\hat{H}^{(l)} = \phi(W_d^{(l)} \hat{H}^{(l-1)} + b_d^{(l)}), l = mid + 1 \ldots L, \tag{10}$$

where, $\hat{H}^{(l)}$ is the representation learned by the $l\text{-}th$ layer in the decoder, $\hat{H}^{(mid)} = Z$, $\hat{H}^{(L)} = \hat{X}$. ϕ is a non-linear activation function. $W_d^{(l)}$, $b_d^{(l)}$ respectively represent the weight and bias of the $l\text{-}th$ layer in the decoder.

Then, the popularity of self-supervised learning is an inevitable trend, and more meaningful information can be learned through simple settings. We use the same type of dataset to assist the current dataset in learning to generate pseudo labels for self-supervised learning. Specifically, the self-supervised module is composed of a self-supervised classifier. On the premise that pseudo labels obtained by clustering the features generated by the pretext task are credible, the pseudo label y_t is introduced to join the training.

$$\mathcal{L}_{self} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{C} y_{t_{ik}} \log \hat{y}_{ik}, \tag{11}$$

where, $\boldsymbol{y_t}$ is the pseudo-label generated by the pretext task, and $\hat{\boldsymbol{y}}$ represents the label generated by the classifier of the features extracted in the downstream task.

Finally, we want to guide nodes in a low-dimensional space to decrease intra-cluster distances and increase inter-cluster distances in the spectral network (Yang et al., 2019). Specifically, the model wants to make two adjacent points in the low-dimensional space close to each other based on the manifold assumption. The feature $Z \in R^{N\times f}$ is first encoded into the graph Laplacian feature space to generate the feature $Y \in R^{N\times C}$, and then use the idea of spectral clustering to optimize nodes according to the similarity of embedded nodes as the weight. But to prevent all nodes from being mapped to the same cluster, the feature Y is generated by QR decomposition. Therefore, The encoding rules are:

$$W_{ij} = e^{-\frac{\|Z_i - Z_j\|^2}{2\sigma^2}}, \tag{12}$$

$$\begin{aligned}\mathcal{L}_{SC} &= \frac{1}{2}\sum_{i,j}^{N} W_{ij}(y_i - y_j)^2 \\ &= \sum_{i=1}^{N} y_i^2 d_i - \sum_{i,j=1}^{N} y_i y_j W_{ij} \\ &= Tr(Y^T DY) - Tr(Y^T WY) \\ &= Tr(Y^T LY),\ s.t.\ Y^T Y = I,\end{aligned} \tag{13}$$

where, $W \geq 0$ is a non-negative matrix constructed with embedded feature Z, L is the Laplacian matrix, $D_{ii} = \sum_j^N W_{ij}$ denotes the degree matrix, and W_{ij} represents the weight between point i and point j.

3.4. Loss function

In the pretext task, we use $\mathcal{L}_{MMD}$, $\mathcal{L}_{CMMD}$ to reduce the domain difference, use $\mathcal{L}_{s\_cls}$ to label the source domain data, the design of the total loss function is shown in Eq. 14.

$$\mathcal{L}_{pretext} = \mathcal{L}_{MMD} + \beta\mathcal{L}_{CMMD} + \gamma\mathcal{L}_{s\_cls}, \tag{14}$$

In the downstream task, the raw data is reconstructed using Eq. 15 based on Mean Square Error (MSE) in the decoder so that the embedded node retains important information. $\mathcal{L}_{soft}$ is used to implicitly constrain the dual-stream encoder to generate dissimilar information to reduce redundancy. $\mathcal{L}_{SC}$ is used for cluster-guided optimization of Embedding Nodes. $\mathcal{L}_{self}$ is used for self-supervised learning of features. The design of the overall loss function is shown in Eq. 16:

$$\mathcal{L}_{dec} = \frac{1}{2N}\sum_{i=1}^{N} \parallel x_i - \hat{x}_i \parallel_2^2 = \frac{1}{2N} \parallel X - \hat{X} \parallel_F^2, \tag{15}$$

$$\mathcal{L}_{down} = \mathcal{L}_{dec} + \delta\mathcal{L}_{SC} + \epsilon\mathcal{L}_{soft} + \mu\mathcal{L}_{self}, \tag{16}$$

β, γ, δ, ϵ, μ are coefficients of the loss function. $\mathcal{L}_{pretext}$ is the loss function of the pretext task. $\mathcal{L}_{down}$ is the loss function of the downstream task. In the pretext, $\mathcal{L}_{MMD}$, $\mathcal{L}_{CMMD}$ are the marginal distribution difference loss function

and the conditional distribution difference loss function, respectively. $\mathcal{L}_{s\_cls}$ is the loss function of the $S\_classifier$. In the downstream task, $\mathcal{L}_{dec}$ is the loss function of the decoder, $\mathcal{L}_{soft}$ is the loss function of the SEN, $\mathcal{L}_{SC}$ is the loss function of the spectral network, and $\mathcal{L}_{self}$ is the loss function of the self-supervised classifier.

3.5. Algorithm description

Algorithm 1 The pretext task algorithm

Input: source feature: X_s, source label: y_s, target feature: X_t, learning rate: lr_1, maximum number of iterations: MI_1;

Output: predicted labels of target domains: $\boldsymbol{y_t}$;

for i=0 to MI_1 **do**
  The source domain generates features H_s according to Eq. 1 and feeds the features into the source classifier;
  The target domain generates features H_t according to Eq. 2;
  Calculate the value of the loss function $\mathcal{L}_{pretext}$ according to Eq. 14;
  Backpropagation, update parameters;
end for
Perform spectral clustering on H_t, and generate pseudo-label $\boldsymbol{y_t}$.

Algorithm 2 The downstream task algorithm

Input: feature: X, learning rate: lr_2, maximum number of iterations: MI_2, number of graph neighbors: K, hyperparameters:σ, pseudo labels: y_t;

Output: clustering results R;

Calculate topology A based on features X;
for i=0 to MI_2 **do**
  GCE generates the representation G according to Eq. 6;
  SE generates the representation H according to Eq. 7;
  Generate features $G^{'}$ and $H^{'}$ by orthogonal constraints using G and H;
  Generate embedded features Z using features G and H according to Eq. 9;

Given the feature Z, the decoder reconstructs the raw data X according to Eq. 10;
Given the feature Z, obtain Y by means of a spectral network;
Given the feature Z, obtain the predicted label $\hat{\boldsymbol{y}}$ through the classifier;
Given feature Z, get the predicted label $\hat{\boldsymbol{y}}$ through the classifier;
Calculate the value of the loss function $\mathcal{L}_{down}$ according to Eq. 16;
Backpropagation, update parameters;
end for
Perform k-means on Z and return clustering result in R.

4. Validation and results

We conduct extensive performance tests on the brain functional connectivity datasets. In the beginning, we introduce the description of the dataset, show our preprocessing, and introduce the performance of evaluation metrics. We compare the proposed framework with different clustering algorithms, analyze and discuss the performance of the pretext task and the downstream task, and then do the ablation study of the loss function. Finally, we did other studies and discussions to analyze the performance of the proposed model.

Table 1: Description of the adopted eight different datasets. n represents the number of samples, d represents the number of features, and c represents the number of clusters.

dataset	Alzheimer's Disease		Post Traumatic Stress Disorder		Attention Deficit Hyperactivity Disorder		Autism Spectrum Disorder	
	AD_1	AD_2	$PTSD_1$	$PTSD_2$	$ADHD_1$	$ADHD_2$	ASD_1	ASD_2
n	96	132	87	174	487	917	454	988
d	888	687	340	677	672	1179	1722	1357
c	4	4	3	3	3	3	3	3

4.1. Datasets

In this paper, we evaluate the proposed framework using eight different brain disease datasets shown in Tab. 1. It can be divided into four categories of diseases, Alzheimer's Disease (AD), Post Traumatic Stress Disorder

(PTSD), Attention Deficit Hyperactivity Disorder (ADHD), and Autism Spectrum Disorder (ASD). The four datasets AD_1, $PTSD_1$, $ADHD_1$, ASD_1 (Zhao et al., 2019, 2018) are obtained from `https://github.com/xinyuzhao/identification-of-brain-based-disorders`, and the four datasets AD_2, $PTSD_2$, $ADHD_2$, ASD_2 (Lanka et al., 2020a,b) are obtained from `https://github.com/pradlanka/malini`. In the proposed framework, we use two datasets under the same broad category of diseases to learn from each other.

4.2. Preprocessing

In the pretext task, the precondition of the UDA method is that the category space and feature space are the same, but the distributions are different. We can deal with the feature space by dimensionality reduction, but we need to filter out the interference items in the category space manually. In addition, the GCN method needs to provide the connection relationship between the data, which is not available in the raw data, so it needs to construct the adjacency matrix artificially.

Our preprocessing consists of the following two main areas.

- Processing of datasets: In the dataset $ADHD_1$, the number of samples with the category $ADHD\text{-}h$ is less than 10, which the author screened out. So the number of label categories in datasets ($ADHD_1$, $ADHD_2$) is different, and there will be a problem with different category spaces as the source and target domains. Therefore, this paper also removes the $ADHD\text{-}h$ with only 13 samples in the dataset $ADHD_2$.

- Construction of Adjacency Matrix: The graph is constructed using the KNN method. Through cross-validation, a small value of K is selected to start, and the value of K is continuously increased, and then a more appropriate value of K is finally found by calculating the verification set. In the adjacency matrix, if there is an edge between two nodes, $A_{ij} = 1$, otherwise $A_{ij} = 0$.

4.3. Evaluation metrics

We use three performances to evaluate the clustering performance of the framework, which are often used in unsupervised learning. There are accuracy ($ACC \in [0, 1]$), adjusted rand index ($ARI \in [-1, 1]$), and F1-score ($F \in [0, 1]$). For all performances, the higher the value means the better.

$$ACC = \max_{m} \frac{\sum_{i=1}^{N} 1\{l_i = m(c_i)\}}{N}, \tag{17}$$

where l_i and c_i are the true label and predicted cluster of data point $\boldsymbol{x_i}$, and m ranges over all possible one-to-one mappings between clusters and labels.

$$RI = \frac{TP + TN}{C_N^2}, \tag{18}$$

$$ARI = \frac{RI - E(RI)}{max(RI) - E(RI)}, \tag{19}$$

where TP (group 2 similar nodes in the same cluster) and TN (group 2 dissimilar nodes in different clusters) are both correct decisions. FP (group 2 dissimilar nodes in the same clusters) and FN (group 2 similar nodes in different clusters) are both wrong decisions. ARI reflects the degree of overlap between the two divisions.

$$F = 2 * \frac{Pre * Recall}{Pre + Recall} = \frac{2TP}{2TP + FP + FN}, \tag{20}$$

where F is a comprehensive analysis of whether TP is large enough from a subjective (predicted) and objective (actual) perspective.

4.4. Performance comparison with the state-of-art methods

Our proposed algorithm is evaluated on eight brain disease datasets and compared with four existing mainstream traditional datasets and nine deep learning clustering algorithms. The four traditional clusters are k-means, Spectral Clustering (SC), OPTICS, and Agglomerative Clustering (Agg). Traditional clustering comes directly from the Scikit-learn library [https://scikit-learn.org/stable]. The nine deep learning clusters are graphencoder (Tian et al.,

Table 2: Performance (%) comparison with proposed methods on eight different datasets. **Bold** indicates the best performance.

Methods	Metric	AD_1	AD_2	$PTSD_1$	$PTSD_2$	$ADHD_1$	$ADHD_2$	ASD_1	ASD_2
SC	ACC	42.71	28.79	100	48.28	57.7	62.49	51.76	56.58
	ARI	10.61	0.64	100	0.56	3.67	0.06	0.34	0.99
	F	35.76	15.42	100	25.23	29.33	25.95	31.68	26.91
k-means	ACC	48.96	55.3	100	70.69	54.83	51.58	53.52	47.77
	ARI	17.93	15.52	100	35.65	16.86	9.86	6.62	3.6
	F	45.75	55.9	100	69.94	46.6	45.14	39.34	36.86
OPTICS	ACC	31.25	26.52	49.43	48.85	55.44	62.27	56.61	56.07
	ARI	-0.79	-0.1	0.85	0.41	-0.41	-0.26	0.3	-0.17
	F	13.45	11.82	24.35	23.04	23.78	25.58	24.5	23.95
Agg	ACC	44.79	45.45	100	59.77	57.49	47.22	53.96	48.68
	ARI	15.61	12.41	100	15.2	17.54	7.46	8.06	0.74
	F	31.7	43.18	100	54.51	48.01	41.91	39.35	33.25
graphencoder	ACC	41.67	39.39	100	42.53	52.57	49.62	49.12	46.86
	ARI	8.22	5.05	100	1.63	7.17	2.68	0.15	1.67
	F	33.8	36.61	100	37.73	39.4	34.96	33.54	34.42
GAE	ACC	57.29	53.03	94.25	56.9	54.62	57.47	50.88	50
	ARI	32.84	15.67	95.4	20.76	16.88	**18.17**	6.65	2.02
	F	52.72	53.17	71.21	55.18	**51.26**	45.66	38.31	35.11
DEC	ACC	44.79	41.67	89.66	69.54	55.44	52.78	56.61	48.99
	ARI	13.04	6.47	81.87	**39.03**	-0.62	11.73	0.3	8.17
	F	43.59	36.91	83.38	64.3	23.78	45.53	24.5	43.34
SDCN	ACC	38.54	46.97	78.16	41.95	45.79	47.44	52.2	45.65
	ARI	3.43	8.76	66.17	-2.08	4.41	-1.85	7	4.84
	F	38.52	45.46	57.51	24.44	41.14	31.82	38.15	36.86
DAEGC	ACC	54.17	46.97	100	68.97	58.52	43.29	49.78	49.78
	ARI	33.69	14.91	100	32.4	17.56	2.56	10.57	10.57
	F	40.01	42.86	100	67.44	46.21	40.03	40.45	40.45
AGE	ACC	53.12	47.73	97.7	64.94	58.73	53.87	51.32	52.63
	ARI	29.47	14.04	95.07	26.54	**19.82**	13.09	10.34	**13.18**
	F	41.63	40.09	96.77	61.94	49.24	45.86	39.3	**43.96**
AdaGAE	ACC	31.25	26.52	77.01	48.85	55.44	62.27	56.61	57.27
	ARI	-0.79	-0.1	67.67	0.41	-0.41	-0.26	0.3	0.99
	F	13.45	11.82	57.67	23.04	23.78	25.58	24.5	26.42
ARGE	ACC	63.54	49.24	100	62.07	52.77	53.44	54.63	49.49
	ARI	**40.25**	11.84	100	20.32	15.46	11.95	9.01	1.4
	F	**61.98**	49.74	100	61.14	46.69	**47.7**	40.4	34.21
R-DGAE	ACC	56.25	46.21	90.8	63.21	49.69	62.05	51.10	55.87
	ARI	14.63	8.37	72.22	18.25	-4.78	2.55	-2.97	-0.09
	F	58.31	44.75	91.54	44.92	22.63	30.8	23.99	24.96
SSCDE	ACC	**65.62**	**57.58**	**100**	**74.71**	**60.99**	**64.45**	**64.1**	**62.04**
	ARI	32.2	**18.32**	**100**	38.07	18.55	9.15	**13.5**	12.12
	F	53.99	**57.55**	**100**	**71.99**	47.54	36.84	**43.19**	43.69

2014), GAE (Kipf & Welling, 2016b), DEC (Xie et al., 2016), SDCN (Bo et al., 2020), DAEGC (Wang et al., 2019), AGE (Cui et al., 2020a), AdaGAE (Li et al., 2021), ARGE (Pan et al., 2018), R-DGAE (Mrabah et al., 2022).

As shown in Tab. 2, the proposed clustering framework performs well on brain disease functional connectivity datasets. The vast majority of datasets achieve the highest accuracy values, showing that our proposed method of using cross-domain learning to generate pseudo labels to supervise the downstream task for self-supervision is feasible. But slightly worse in ARI and F performance. After analysis, it may be due to the redundant expression of high-dimensional small sample data features, but our framework does not handle this well.

4.5. Ablation study of pretext and downstream tasks

Table 3: Performance (%) on eight brain functional connectivity datasets in the pretext task.

	$AD_2 \rightarrow AD_1$	$AD_1 \rightarrow AD_2$	$PTSD_2 \rightarrow PTSD_1$	$PTSD_1 \rightarrow PTSD_2$
ACC	62.5	52.27	100	71.26
ARI	26.76	12.91	100	33
F	61.87	51.26	100	68.43
	$ADHD_2 \rightarrow ADHD_1$	$ADHD_1 \rightarrow ADHD_2$	$ASD_2 \rightarrow ASD_1$	$ASD_1 \rightarrow ASD_2$
ACC	58.73	62.49	63.88	60.43
ARI	16.92	0.81	12.96	12.29
F	43.87	27.33	42.88	43.52

In the pretext task, we use datasets under a broad category of diseases to learn from each other. A→B, that is, A is the auxiliary dataset, and B is the target dataset. To learn more useful information, we use domain adaptive learning to map the source and target domains into a common feature subspace so that we can apply the knowledge on the source domain to the target domain. Transfer information from the source domain to the target domain by labeling the source domain and making the two domains similar in distribution. After cross-domain learning, we cluster the extracted target domain features. The

performance is shown in Tab. 3. The results show that the performance is good, so the design of the pretext task is feasible, and the obtained pseudo labels are credible.

Table 4: Performance (%) on eight brain functional connectivity datasets in the downstream task without self-supervision.

	AD_1	AD_2	$PTSD_1$	$PTSD_2$	$ADHD_1$	$ADHD_2$	ASD_1	ASD_2
ACC	58.33	53.79	100	67.82	59.55	60.52	58.59	57.59
ARI	21.43	14.33	100	26.90	19.83	10.19	4.14	10.30
F	54.15	53.61	100	66.68	45.19	37.98	30.78	38.71

In the downstream task, we propose a soft-orthogonal encoding network that exploits orthogonality to capture features as dissimilar as possible through two encoders. We also use the decoder to reconstruct the features. At the same time, the spectral network module will perform clustering optimization on the features. After that, the features in the low-dimensional space are clustered. The framework is tested without adding pseudo labels in the pretext task. Some performances are higher than the pretext task, and some are slightly inferior to the pretext task. The performances are shown in Tab. 4. The results show that the performance is good, so the design of the downstream task is feasible.

Table 5: Performance (%) on eight brain functional connectivity datasets in the self-supervised downstream task.

	AD_1	AD_2	$PTSD_1$	$PTSD_2$	$ADHD_1$	$ADHD_2$	ASD_1	ASD_2
ACC	65.62	57.58	100	74.71	60.99	64.45	64.1	62.04
ARI	32.20	18.32	100	38.07	18.55	9.15	13.5	12.12
F	53.99	57.55	100	71.99	47.54	36.84	43.19	43.69

Inspired by the pretext and downstream tasks and self-supervision, we add pseudo labels for self-supervised training from the pretext task to the downstream task. Based on the original without self-supervision, a self-supervision module is added. The performance is shown in Tab. 5, which has been greatly improved, and the effect of one plus one is greater than two is achieved. There-

fore, obtaining pseudo labels based on cross-domain learning for self-supervised learning.

4.6. Ablation study of loss function

The value of the loss function can be used to express the gap between the actual and expected data. In deep learning, the model often uses the value of the loss function to backpropagate and optimize the network. In order to verify the rationality of the proposed framework in the design of the loss function, we conduct ablation study on the loss function. J1 represents the reconstruction loss function in the decoder, J2 represents the loss function in the SEN, J3 represents the loss function guided by clustering optimization in the spectral network, and J4 represents the loss function of the self-supervised classifier in the self-supervised module.

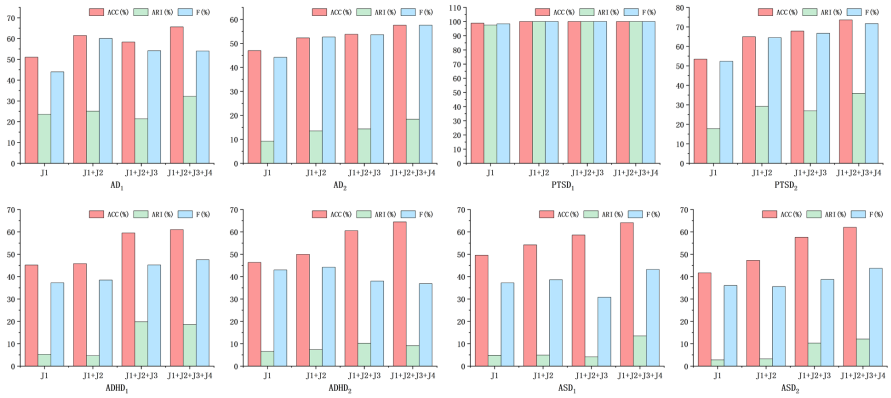


Figure 2: Ablation study of the loss function.

As shown in Fig. 2, we perform ablation study on the loss function to verify the rationality of the proposed framework loss function design. After analysis and comparison, it is found that on the whole, the fusion of these loss functions has greatly improved the performance. Specifically, in the case of reconstruction loss J1, the framework can extract features that preserve local structural depth information. Based on adding J1 and then adding the loss function J2 of the

SEN, we make the two features captured by the dual-stream encoder as dissimilar as possible. On the eight datasets, except for the slight decrease in AD_1, all show an improvement effect, so it can be justified that the design with J2 is added. More, in the case of adding J1 and J2, in order to make the nodes in the same cluster closer in the low-dimensional space and the nodes in different clusters away from each other, we also added the loss function J3 of clustering optimization, which can be found that the performance on the dataset is improved. Finally, we add the self-supervised loss function J4 based on pseudo labels given by the pretext task of cross-domain learning, and the performance is improved. The design of our loss function is reasonable and reliable, and some complex information can be mined from the brain functional connectivity data.

4.7. Performance analysis of SEN

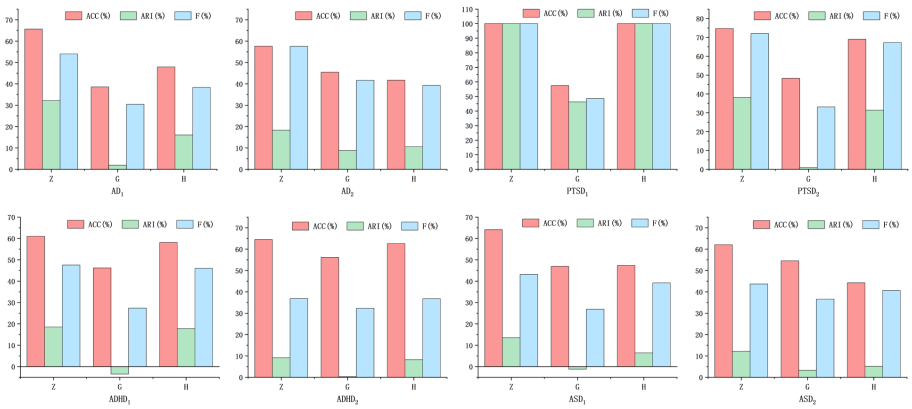


Figure 3: Performance (%) of ACC, ARI, F on eight datasets for features Z, G, H captured in SEN.

The features captured by a single encoder are usually limited, and the information obtained is insufficient. At present, many researchers have studied two or more encoders. Although multiple encoders can capture more information, many researchers do not consider the adverse effects of encoders. In our proposed dual-stream encoding network, SE is used to capture highly correlated features, and GCE is used to capture node-level information. And an orthog-

onal network is used to implicitly constrain the encoder so that the captured nodes are dissimilar. Then map these two feature codes to a new subspace to obtain the features we want.

In the SEN, G is the feature captured by GCE, H is the feature captured by SE, and Z is the feature that maps G and H to a new subspace after weighted encoding. As shown in Fig. 3, it can be seen that Z outperforms the features in the other two subspaces on all three evaluation metrics after clustering. Therefore, our encoding network design is reasonable and feasible.

4.8. Implementation Details

All experimental codes are implemented on the Windows system using the Pytorch framework. For the pretext task of the model, this paper adopts domain adaptive learning. The model is trained using an Adam optimizer with a maximum number of iterations of 200. UDA, which in the pretext task, consists of two parallel networks, both of which are composed of nonlinear functions. The activation function corresponding to each layer is a Relu function, which is finally reduced to 64 dimensions. For the downstream task, the graph convolution encoder consists of two layers of graph convolution. The feature dimension after the first layer of image convolution is 256, and the feature dimension after the second layer of image convolution is 32; The stack encoder consists of three layers of nonlinear transformation functions. The corresponding activation function of each layer is a Relu function, which is reduced to 256, 128, and 64 dimensions, respectively. Using the Adam optimizer to optimize the model, the maximum number of iterations of the model is 200. Our proposed model is able to converge gradually as the epoch increases during the training process. When the loss function converges, we consider that the trained neural network model has stabilized.

4.9. Comparison of sample visualizations for different algorithms

In order to clearly show the clustering effect of the algorithm of this thesis, taking the dataset AD_2 as an example, this thesis presents the samples under

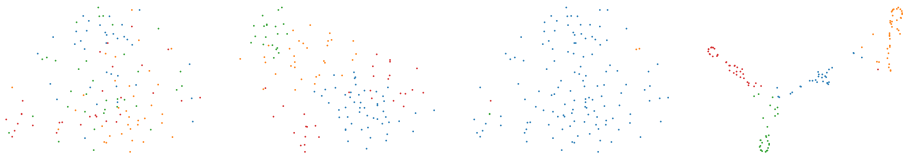


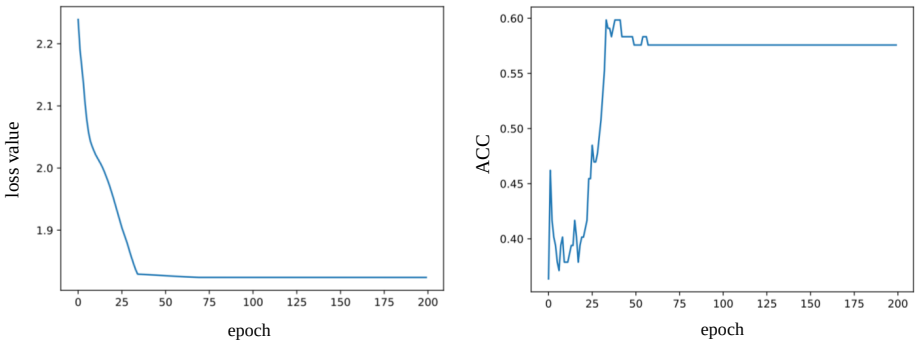
Figure 4: Visualization of the different methods on the AD_2 dataset.

the feature space and their clustering effect by the t-SNE method, as shown in Fig. 4, Fig. 4(a) represents the data distribution under the raw feature space. Fig. 4(b) represents the distribution of samples after clustering with k-means clustering. Fig. 4(c) shows the distribution of samples after clustering with Spectral clustering. Fig. 4(d) shows the sample distribution after clustering with the SSCDE model. SSCDE can automatically identify and analyze the similarities and differences between data points and classify them into different clusters.

Furthermore, it is clear from the graph that the raw data distribution, the traditional clustering distribution, is poor. However, in SSCDE, the division of clusters is visible. It shows that the SSCDE model can extract effective low-dimensional features and optimize them according to the similarity between the data, thus making clustering effective.

4.10. Convergence analysis of the model

As shown in Fig. 5, the model initially converges at a high-speed rate and then flattens. It may be related to the fact that the AD_2's sample is too small so that it can converge very quickly. However, it is also formal because the data set is too small, and there is some difficulty in mining information from small sample data. In addition, the ACC change curve is unstable in the first 50 epochs and becomes stable when the loss converges. It indicates that the model has found the optimal solution according to the given objective function.



(a) The variation of the loss (b) The variation of the ACC

Figure 5: Loss plot and ACC plot on the AD_2 dataset.

4.11. Discussion of parameters

Several hyperparameters are covered in our paper, such as learning rate, coefficients of the loss function, σ, k. σ controls the local scope of the Gaussian kernel function and plays an important role in the optimization of clustering. A good initial graph helps us learn a good feature representation, and k determines the number of adjacencies for each node. Therefore, we introduce the parameter σ and parameter k in detail.

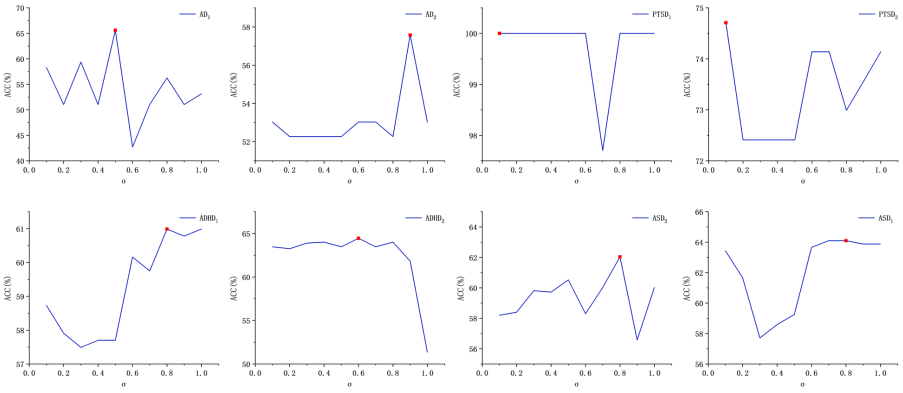


Figure 6: The effect of parameter σ on ACC.

We pay special attention to the value of the parameter σ in the spectral network. It is found that the value of σ plays an important role in the perfor-

mance. In Fig. 6, according to experience, we set the value of σ on [0.1, 0.2 ... 1] to analyze its influence on ACC. When ACC gets the highest value, we pick the value of σ, and the final obtained value is shown in Tab. 6.

More, we also pay attention to the value of parameter k. In preprocessing, we pick the optimal number of adjacencies k for each dataset. As shown in Fig. 7, when the error rate reaches the lowest, we determine the optimal k. The optimal k obtained is shown in Tab. 6.

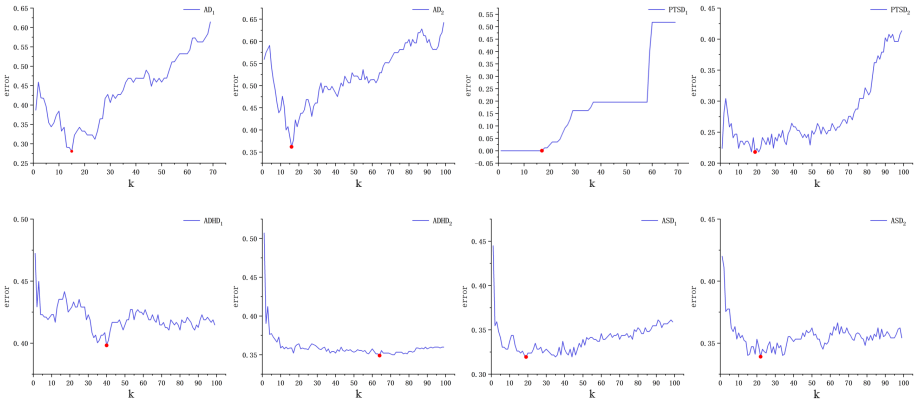


Figure 7: The error rate of parameter k during preprocessing.

Table 6: The value of the parameter for each dataset.

	AD_1	AD_2	$PTSD_1$	$PTSD_2$	$ADHD_1$	$ADHD_2$	ASD_1	ASD_2
σ	0.5	0.9	0.1	0.1	0.8	0.6	0.8	0.8
k	15	16	17	19	40	64	19	22

Another important parameter is α. When features G and H are captured by the dual-stream encoder, the formula $Z = \alpha G + (1 - \alpha)H$ is used for encoding to generate feature Z. And, α uniformly takes the value 0.5. But we believe that α also plays an important role in the framework, and subsequent research can conduct adaptive learning research on the role of α in the network.

Conclusion

This paper proposes a soft-orthogonal constrained dual-stream encoder framework with a self-supervised network for clustering brain functional connectivity data. In the proposed algorithm, the dual-stream encoder can capture richer and non-redundant information under the constraint of the soft-orthogonal network. The design of the soft-orthogonal network ensures that the two encoders are complementary and unique, improving the accuracy of the model clustering. Furthermore, we generate pseudo labels based on cross-domain learning in the pretext task to assist the downstream task for self-supervised learning. The results show that the design of the proposed clustering framework, which can capture in-depth information, is reasonable and effective. Compared with multiple existing clustering algorithms, our proposed framework exhibits promising performance. The model can improve the accuracy and interpretability of clustering and has excellent performance and application prospects. This algorithmic model can assist doctors and others in their work to a certain extent and has some clinical reference value. However, the performance of self-supervised learning is affected by the pretext task. In future research, we will continue to expand the clustering and self-supervised learning models to improve brain functional connectivity data recognition performance. (Xia et al., 2021).

References

Baumgartner, R., Scarth, G., Teichtmeister, C., Somorjai, R., & Moser, E. (1997). Fuzzy clustering of gradient-echo functional mri in the human visual cortex. part i: Reproducibility. *Journal of Magnetic Resonance Imaging*, *7*, 1094–1101.

Biswal, B., Zerrin Yetkin, F., Haughton, V. M., & Hyde, J. S. (1995). Functional connectivity in the motor cortex of resting human brain using echo-planar mri. *Magnetic resonance in medicine*, *34*, 537–541.

Bo, D., Wang, X., Shi, C., Zhu, M., Lu, E., & Cui, P. (2020). Structural deep clustering network. In *Proceedings of The Web Conference 2020* (pp. 1400–1410).

Borlea, I.-D., Precup, R.-E., & Borlea, A.-B. (2022). Improvement of k-means cluster quality by post processing resulted clusters. *Procedia Computer Science*, *199*, 63–70.

Caron, M., Bojanowski, P., Joulin, A., & Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 132–149).

Chen, C., Lu, H., Wei, H., & Geng, X. (2022). Deep subspace image clustering network with self-expression and self-supervision. *Applied Intelligence*, (pp. 1–15).

Cox, D. D., & Savoy, R. L. (2003). Functional magnetic resonance imaging (fmri)"brain reading": detecting and classifying distributed patterns of fmri activity in human visual cortex. *Neuroimage*, *19*, 261–270.

Cui, G., Zhou, J., Yang, C., & Liu, Z. (2020a). Adaptive graph encoder for attributed graph embedding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 976–985).

Cui, X., Xiang, J., Guo, H., Yin, G., Zhang, H., Lan, F., & Chen, J. (2018). Classification of alzheimer's disease, mild cognitive impairment, and normal controls with subnetwork selection and graph kernel principal component analysis based on minimum spanning tree brain functional network. *Frontiers in Computational Neuroscience*, *12*, 31.

Cui, X., Xiao, J., Guo, H., Wang, B., Li, D., Niu, Y., Xiang, J., & Chen, J. (2020b). Clustering of brain function network based on attribute and structural information and its application in brain diseases. *Frontiers in Neuroinformatics*, *13*, 79.

Guo, X., Liu, X., Zhu, E., & Yin, J. (2017). Deep clustering with convolutional autoencoders. In *International conference on neural information processing* (pp. 373–382). Springer.

Hendrycks, D., Mazeika, M., Kadavath, S., & Song, D. (2019). Using self-supervised learning can improve model robustness and uncertainty. *Advances in Neural Information Processing Systems*, *32*.

Ji, P., Zhang, T., Li, H., Salzmann, M., & Reid, I. (2017). Deep subspace clustering networks. *Advances in neural information processing systems*, *30*.

Kipf, T. N., & Welling, M. (2016a). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, .

Kipf, T. N., & Welling, M. (2016b). Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, .

Lanka, P., Rangaprakash, D., Dretsch, M. N., Katz, J. S., Denney, T. S., & Deshpande, G. (2020a). Supervised machine learning for diagnostic classification from large-scale neuroimaging datasets. *Brain imaging and behavior*, *14*, 2378–2416.

Lanka, P., Rangaprakash, D., Gotoor, S. S. R., Dretsch, M. N., Katz, J. S., Denney Jr, T. S., & Deshpande, G. (2020b). Malini (machine learning in neuroimaging): A matlab toolbox for aiding clinical diagnostics using resting-state fmri data. *Data in brief*, *29*, 105213.

Li, X., Zhang, H., & Zhang, R. (2021). Adaptive graph auto-encoder for general data clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, .

Liu, X., Zhang, F., Hou, Z., Mian, L., Wang, Z., Zhang, J., & Tang, J. (2021). Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, .

Long, M., Wang, J., Ding, G., Sun, J., & Yu, P. S. (2013). Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision* (pp. 2200–2207).

Lu, H., Chen, C., Wei, H., Ma, Z., Jiang, K., & Wang, Y. (2022). Improved deep convolutional embedded clustering with re-selectable sample training. *Pattern Recognition*, *127*, 108611.

Lu, H., & Jin, T. (2022). Dual-stream encoder neural networks with spectral constraint for clustering functional brain connectivity data. *Neural Computing and Applications*, (pp. 1–11).

Lu, H., Liu, S., Wei, H., Chen, C., & Geng, X. (2021). Deep multi-kernel auto-encoder network for clustering brain functional connectivity data. *Neural Networks*, *135*, 148–157.

Lu, H., Liu, S., Wei, H., & Tu, J. (2020). Multi-kernel fuzzy clustering based on auto-encoder for fmri functional network. *Expert Systems with Applications*, *159*, 113513.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (pp. 281–297). Oakland, CA, USA volume 1.

Misra, I., & Maaten, L. v. d. (2020). Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 6707–6717).

Mrabah, N., Bouguessa, M., Touati, M. F., & Ksantini, R. (2022). Rethinking graph auto-encoder models for attributed graph clustering. *IEEE Transactions on Knowledge and Data Engineering*, .

Ng, A., Jordan, M., & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, *14*.

Ng, A. et al. (2011). Sparse autoencoder. *CS294A Lecture notes*, *72*, 1–19.

Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., & Zhang, C. (2018). Adversarially regularized graph autoencoder for graph embedding. *arXiv preprint arXiv:1802.04407*, .

Pehlivan, N. Y., & Turksen, I. B. (2021). A novel multiplicative fuzzy regression function with a multiplicative fuzzy clustering algorithm. *Romanian Journal of Information Science and Technology*, *24*, 79–98.

Pozna, C., & Precup, R.-E. (2012). Aspects concerning the observation process modelling in the framework of cognition processes. *Acta Polytechnica Hungarica*, *9*, 203–223.

Precup, R.-E., Duca, G., Travin, S., & Zinicovscaia, I. (2022). Processing, neural network-based modeling of biomonitoring studies data and validation on republic of moldova data. *PROCEEDINGS OF THE ROMANIAN ACADEMY SERIES A-MATHEMATICS PHYSICS TECHNICAL SCIENCES INFORMATION SCIENCE*, *23*, 403–410.

Shen, X., & Chung, F. L. (2019). Network embedding for cross-network node classification. *arXiv preprint arXiv:1901.07264*, .

Snyder, L. H., Batista, A., & Andersen, R. A. (1997). Coding of intention in the posterior parietal cortex. *Nature*, *386*, 167–170.

Suk, H.-I., Lee, S.-W., & Shen, D. (2015). Latent feature representation with stacked auto-encoder for ad/mci diagnosis. *Brain Structure and Function*, *220*, 841–859.

Tan, G. W.-H., Ooi, K.-B., Leong, L.-Y., & Lin, B. (2014). Predicting the drivers of behavioral intention to use mobile learning: A hybrid sem-neural networks approach. *Computers in Human Behavior*, *36*, 198–213.

Tian, F., Gao, B., Cui, Q., Chen, E., & Liu, T.-Y. (2014). Learning deep representations for graph clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*. volume 28.

Velickovic, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., & Hjelm, R. D. (2019). Deep graph infomax. *ICLR (Poster)*, *2*, 4.

Vergun, S., Gaggl, W., Nair, V. A., Suhonen, J. I., Birn, R. M., Ahmed, A. S., Meyerand, M. E., Reuss, J., DeYoe, E. A., & Prabhakaran, V. (2016). Classification and extraction of resting state networks using healthy and epilepsy fmri data. *Frontiers in neuroscience*, *10*, 440.

Verma, A., Meenpal, T., & Acharya, B. (2022). Computational cost reduction of convolution neural networks by insignificant filter removal. *SCIENCE AND TECHNOLOGY*, *25*, 150–165.

Wang, C., Pan, S., Hu, R., Long, G., Jiang, J., & Zhang, C. (2019). Attributed graph clustering: A deep attentional embedding approach. *arXiv preprint arXiv:1906.06532*, .

Wang, J., Feng, W., Chen, Y., Yu, H., Huang, M., & Yu, P. S. (2018). Visual domain adaptation with manifold embedded distribution alignment. In *Proceedings of the 26th ACM international conference on Multimedia* (pp. 402–410).

Xia, W., Gao, Q., Yang, M., & Gao, X. (2021). Self-supervised contrastive attributed graph clustering. *arXiv preprint arXiv:2110.08264*, .

Xie, J., Girshick, R., & Farhadi, A. (2016). Unsupervised deep embedding for clustering analysis. In *International conference on machine learning* (pp. 478–487). PMLR.

Yang, X., Deng, C., Zheng, F., Yan, J., & Liu, W. (2019). Deep spectral clustering using dual autoencoder network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 4066–4075).

Zhang, J., Li, C.-G., You, C., Qi, X., Zhang, H., Guo, J., & Lin, Z. (2019). Self-supervised convolutional subspace clustering network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5473–5482).

Zhao, X., Rangaprakash, D., Denney Jr, T. S., Katz, J. S., Dretsch, M. N., & Deshpande, G. (2019). Identifying neuropsychiatric disorders using unsupervised clustering methods: Data and code. *Data in brief*, *22*, 570–573.

Zhao, X., Rangaprakash, D., Yuan, B., Denney Jr, T. S., Katz, J. S., Dretsch, M. N., & Deshpande, G. (2018). Investigating the correspondence of clinical diagnostic grouping with underlying neurobiological and phenotypic clusters using unsupervised machine learning. *Frontiers in applied mathematics and statistics*, *4*, 25.

Zhuang, C., & Ma, Q. (2018). Dual graph convolutional networks for graph-based semi-supervised classification. In *Proceedings of the 2018 World Wide Web Conference* (pp. 499–508).