Graph Information Aggregation Cross-Domain Few-Shot Learning for Hyperspectral Image Classification

Yuxiang Zhang, *Student Member, IEEE*, Wei Li, *Senior Member, IEEE*, Mengmeng Zhang, Shuai Wang, Ran Tao, *Senior Member, IEEE*, and Qian Du, *Fellow, IEEE*

***Abstract*— Most domain adaptation (DA) methods in cross-scene hyperspectral image classification focus on cases where source data (SD) and target data (TD) with the same classes are obtained by the same sensor. However, the classification performance is significantly reduced when there are new classes in TD. In addition, domain alignment, as one of the main approaches in DA, is carried out based on local spatial information, rarely taking into account nonlocal spatial information (nonlocal relationships) with strong correspondence. A graph information aggregation cross-domain few-shot learning (Gia-CFSL) framework is proposed, intending to make up for the above-mentioned shortcomings by combining FSL with domain alignment based on graph information aggregation. SD with all label samples and TD with a few label samples are implemented for FSL episodic training. Meanwhile, intradomain distribution extraction block (IDE-block) and cross-domain similarity aware block (CSA-block) are designed. The IDE-block is used to characterize and aggregate the intradomain nonlocal relationships and the interdomain feature and distribution similarities are captured in the CSA-block. Furthermore, feature-level and distribution-level cross-domain graph alignments are used to mitigate the impact of domain shift on FSL. Experimental results on three public HSI datasets demonstrate the superiority of the proposed method. The codes will be available from the website: https://github.com/YuxiangZhang-BIT/IEEE_TNNLS_Gia-CFSL.**

***Index Terms*— Cross-scene, distribution alignment, domain adaption, few-shot learning (FSL), graph neural network (GNN), hyperspectral image classification.**

Manuscript received 2 October 2021; revised 27 March 2022 and 23 May 2022; accepted 19 June 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61922013, Grant U1833203, and Grant 62001023; in part by the Beijing Natural Science Foundation under Grant JQ20021 and Grant L191004; and in part by the China Postdoctoral Science Foundation under Grant BX20200058. *(Corresponding author: Wei Li.)*

Yuxiang Zhang, Wei Li, Mengmeng Zhang, and Ran Tao are with the School of Information and Electronics, and the Beijing Key Laboratory of Fractional Signals and Systems, Beijing Institute of Technology, Beijing 100081, China (e-mail: zyx829625@163.com; liwei089@ieee.org; mengmengzhang@bit.edu.cn; rantao@bit.edu.cn).

Shuai Wang is with the Department of Chemistry, The University of Hong Kong, Hong Kong, China (e-mail: shuaiw@connect.hku.hk).

Qian Du is with the Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39762 USA (e-mail: du@ece.msstate.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3185795.

Digital Object Identifier 10.1109/TNNLS.2022.3185795

NOMENCLATURE

Introduction of Abbreviations:

Abbreviation	Description
HSI	Hyperspectral image.
SD	Source data.
TD	Target data.
FSL	Few shot learning.
DA	Domain adaptation.
FG	Feature graph.
DG	Distribution graph.
GOT	Graph optimal transmission.
MMD	Maximum mean difference.
IDE-block	Intradomain distribution extraction block.
CSA-block	Cross-domain similarity aware block.

Explanation of Variables:

Notations	Description
$\mathbf{X}_s$ and $\mathbf{X}_t$	Source and target data.
$\mathcal{T}_s = \{\mathcal{S}_s, \mathcal{Q}_s\}$	SD FSL task $\mathcal{T}_s$ is divided into support set $\mathcal{S}_s$ and query set $\mathcal{Q}_s$.
$\mathcal{T}_t^{\text{tr}}$	TD training data with a few labeled samples.
$\mathcal{T}_t^{\text{te}}$	TD testing data with unlabeled samples.
$\mathbf{G}_f^{s/t}$ and $\mathbf{G}_d^{s/t}$	SD or TD feature graph and distribution graph.
$\mathbf{v}_{f(i)}^{s/t}$ and $\mathbf{v}_{d(i)}^{s/t}$	Node of SD or TD feature graph and distribution graph.
$\mathbf{e}_{f(i,j)}^{s/t}$ and $\mathbf{e}_{d(i,j)}^{s/t}$	Edge of SD or TD feature graph and distribution graph.
$\mathbf{m}^{s/t}$	Attentional message of SD or TD.
$\mathbf{H}_{\text{atten}}^{s/t}$	Attentional aggregation output of SD or TD.

I. INTRODUCTION

HYPERSPECTRAL imagery obtained by remote sensing system is a 3-D data cube composed of hundreds of spectral channels, with abundant spatial and spectral information. The fine classification of ground objects using HSI is one of the core contents of the application in remote sensing field,

such as land cover, resource investigation and environmental monitoring [1]–[7]. Single-scene classification [8]–[10], and multisensor single-scene classification methods [11]–[13] based on machine learning and deep learning methods have been proposed. However, it is often encountered that training sets and testing are originated from different scenes in practical remote sensing applications. This challenge is called a cross-scene classification task, in which training and testing samples are the SD with sufficient labels and TD with a small amount or even no labels, respectively. The purpose of this task is to learn the shared knowledge of SD and TD and transfer SD-trained models to TD. The acquisition process of HSI is inevitably affected by various factors, such as sensor nonlinearities, seasonal, and weather conditions [14], [15], which lead to differences in spectral reflectance between SD and TD of the same land cover classes. In the methods mentioned earlier, training and testing sets are assumed to be independent and identically distributed. Therefore, when directly using SD to classify TD, the problem of spectral shift is often encountered.

As a case of transductive transfer learning, DA attempts to reduce the spectral shift in feature-level and learn domain-invariant models, so that the model trained on SD is adapted to TD. At present, many deep learning methods design adaptive layers and metric criteria to reduce the deviation of SD and TD data distribution [16]–[18]. Deep adaptation network (DAN) applied the multikernel maximum mean discrepancy (MK-MMD) metric with better representation ability to three adaptive layers [16]. Zhu *et al.* [18] proposed the deep subdomain adaption network (DSAN) that defined the concept of subdomains and used local MDD (LMMD) to align the relevant subdomains, respectively.

Most DA frameworks aim to adapt the same classes of knowledge learned from SD to TD, and the adaptive performance decreases significantly when new classes are present in TD. In recent years, FSL has been widely concerned due to its effectiveness in identifying new unseen classes using a few training samples [19]–[26]. As one of the learning strategies in FSL, meta-learning adopts the episodic learning method [21], [27]. This training scheme creates episodes that simulate the training and testing scenes of FSL. In the training phase, the dataset is decomposed into different meta-tasks to obtain meta-knowledge and learn the generalization ability of the model for few-shot scenes with new unseen classes. The classification is completed without changing the trained model in the testing stage. There are three main meta-learning methods to solve the problem of few-shot classification (FSC), i.e., the recurrent-based frameworks [28], optimization-based schemes [29], [30], and metric-based methods [31], [32]. Among these three methods, metric-based methods have attracted much attention due to their simplicity and effectiveness. Recently, FSL has been used in HSI classification to alleviate the requirement of producing larger training sets. Liu *et al.* [33] proposed deep FSL (DFSL) to classify HSI with few labeled data. Deep residual 3-D CNN was used to extract spatial–spectral features and learned a metric space, where the nearest neighbor classifier was applied to classify the testing samples. Relation network for HSI FSC (RN-FSC) selected a few TD samples as a fine-tuning dataset to fine-tune the SD training model and utilized the remaining data of TD for testing [34].

At present, most cross-domain HSI classification methods based on DA focus on the same sensor. It is worth extending the DA method in the cases of different sensors and land cover classes by combining FSL. In addition, the current cross-scene classification task is based on local spatial information extracted by a traditional convolution operator to characterize and align the data distribution. However, in general, there is no strong correspondence of local spatial relationships between the two scenes. Therefore, it is necessary to consider nonlocal spatial information (nonlocal relationships) between land cover classes to represent the distribution of domain, and further design an effective nonlocal relationships (i.e., graph data) alignment strategy to alleviate the influence of domain shift.

To solve the above-mentioned problems, a graph information aggregation cross-domain FSL (Gia-CFSL) framework is proposed, which is based on the mechanism of combining DA and FSL. First, the episodic learning pattern of FSL is implemented on SD and TD, which is to build a meta-task (i.e., support set and query set). Their features are mapped to the same dimension through the mapping layer. Then, the similarity between the support set and query set is learned using the metric function after extracting the embedding features. Furthermore, based on the graph distribution information propagation, the IDE-block and CSA-block of SD and TD are designed, and the domain alignment is carried out from two aspects of graph feature- and graph distribution-levels. Specifically, the embedding features of SD and TD are used to construct the feature graph (FG) in the meta-task, which contains the nonlocal relationships among classes, and then IDE-block is used to generate the distribution graph (DG). Graph Optimal transport (GOT) is employed for the cross-domain alignment of feature- and distribution-level graphs. In addition, the feature-level attention and distribution-level high-order attention between SD and TD are obtained by the CSA block.

The main contributions of this work are summarized as follows.

1) In order to solve the problem of domain shift in cross-scene classification, that is, the case of spatial and spectral resolution differences and intraclass differences between domains, a strategy of graph neural network (GNN) information propagation and graph alignment is proposed.
2) To alleviate the problem that existing FSC methods deteriorate when the domain shift occurs in SD and TD, IDE-block is further designed by using the graph distribution information propagation mechanism, and the cross-domain graph alignment is carried out at feature and distribution levels.
3) A CSA-block is designed, for cross-domain attention aggregation, so that the interdomain distribution-level high-order attention and feature-level attention are coordinated to better capture the interdomain feature similarity and distribution similarity.

The rest of this article is organized as follows. Section II introduces relevant concepts of the cross-domain FSL and GNN. Section III elaborates on the proposed Gia-CFSL. The extensive experiments and analyses are presented in Section IV. Finally, conclusions are drawn in Section V.

II. Related Work

Assume that $\mathbf{X}_s \in \mathbb{R}^{d_s}$ with C_s classes and $\mathbf{X}_t \in \mathbb{R}^{d_t}$ with C_t classes are data from SD and TD, respectively, where d_s and d_t denote the dimension. The marginal distribution of SD and TD are expressed as $P(\mathbf{X}_s)$ and $P(\mathbf{X}_t)$, respectively. A graph is represented as $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{G}$ denotes an undirected graph, $\mathbf{V}$ is the node set of the graph, and $\mathbf{E}$ indicates the set of edges connected by nodes $\mathbf{v}_i$ and $\mathbf{v}_j$. Some important abbreviation and variables with their descriptions are listed in Nomenclature.

A. Cross-Domain Few-Shot Learning

Following the common few-shot setups, each FSL task $\mathcal{T}$ consists of support set $\mathcal{S}$ and a query set $\mathcal{Q}$, where $\mathcal{S}$ is a labeled set and $\mathcal{Q}$ is an unlabeled set on which the learned classifier is evaluated. When making a meta-task, the support set $\mathcal{S}$ contains C classes with K samples in each class, and the FSL task is called C-way K-shot, which is denoted as an episode. The query set $\mathcal{Q}$ contains samples from the same classes with the support set $\mathcal{S}$ in a meta-task.

There are two cross-domain FSL tasks: SD FSL task $\mathcal{T}_s$ and TD FSL task $\mathcal{T}_t$. All data in the SD FSL task is labeled, and the TD FSL task is separated into TD training data $\mathcal{T}_t^{\text{tr}}$ with a few labeled samples and TD testing data $\mathcal{T}_t^{\text{te}}$ with unlabeled samples. In general setting, C_s is greater than C_t to ensure the diversity of training samples [33]. $\mathcal{T}_s$ and $\mathcal{T}_t^{\text{tr}}$ are used for the FSL episodic training procedure of SD and TD, respectively. In the case of SD episodic training procedure, the SD support set $\mathcal{S}_s = \{(\mathbf{x}_n^s, y_n^s)\}_{n=1}^{C\times K}$ is composed of randomly selected C classes with K samples per class from $\mathcal{T}_s$, and the SD query set has T samples $\mathcal{Q}_s = \{(\mathbf{x}_m^s, y_m^s)\}_{m=C\times K}^{C\times K+T}$ (y_n^s and y_m^s are the corresponding class labels). The support set $\mathcal{S}_s$ in each episode serves as the labeled training set on which the model is trained to minimize the loss of its predictions over the query set $\mathcal{Q}_s$. This training procedure is iteratively carried out episode by episode until convergence [35]. The TD episodic training procedure is the same as earlier, but due to the few labeled samples in $\mathcal{T}_t^{\text{tr}}$, a data augment is necessary so that a meta-task is constructed.

B. Graph Neural Network

The excellent ability of GNN in processing unstructured data has made new breakthroughs in recommendation systems and natural language processing. GNN was first designed for tasks on processing graph-structured data [36], which mainly represents the characteristics of central nodes by neighborhood aggregation and transforming neighboring nodes recursively. The general GNN framework can be expressed as

$$\mathbf{H}_{\mathbf{v}}^{\ell} = \text{Combine}^{\ell}\left(\mathbf{H}_{\mathbf{v}}^{\ell-1}, \text{Aggregator}^{\ell}\left(\left\{\mathbf{H}_{\mathbf{u}}^{\ell-1}, \forall \mathbf{u} \in \mathcal{N}(\mathbf{v})\right\}\right)\right) \tag{1}$$

where $\mathbf{v}$ and $\mathbf{u}$ are the central and neighboring nodes, respectively, $\mathcal{N}(\mathbf{v})$ is the neighborhood function, which controls the order of selecting neighboring nodes. In each iteration, only the features of the first order neighbors are aggregated, and the GNN framework is composed of two functions, i.e., combine and aggregator. After ℓ rounds of aggregation, each node $\mathbf{v} \in \mathbf{V}$ obtains its representation vector $\mathbf{H}_{\mathbf{v}}^{\ell}$, which is input into a mapping function, e.g., a fully connected layer, to obtain the final results for a specific task such as node classification. Many GNN methods have been developed from the point of aggregate functions. Hamilton *et al.* [37] proposed GraphSAGE, which randomly sampled the adjacent nodes, making the number of adjacent nodes of each node smaller than a given number of samples. A recursive distributed GNN implementation method of autoregressive moving average (ARMA) filter was proposed [38], in which the convolution layer was trained efficiently and localized in the node space.

C. Graph Optimal Transport

GOT, focusing on cross-domain alignment, was first proposed for tasks, such as image-text retrieval, visual question answering, and machine translation [39]. Cross-domain alignment is expressed as a graph matching problem. GOT uses the cosine distance to measure the similarity between nodes and uses two optimal transport distances: Wasserstein distance (WD) for node matching, and Gromov-WD (GWD) for edge matching. WD is calculated as

$$\begin{aligned} \mathcal{D}_{\text{WD}}[P(\mathbf{X}_s), P(\mathbf{X}_t)] &= \inf_{\gamma \in \Pi[P(\mathbf{X}_s), P(\mathbf{X}_t)]} \mathrm{E}_{(\mathbf{x}^s, \mathbf{x}^t)\sim\gamma}\left[c(\mathbf{x}^s, \mathbf{x}^t)\right] \\ c\left(\mathbf{x}_i^s, \mathbf{x}_j^t\right) &= 1 - \frac{\left(\mathbf{x}_i^s\right)^T \mathbf{x}_j^t}{\left\|\mathbf{x}_i^s\right\|_2 \left\|\mathbf{x}_j^t\right\|_2} \end{aligned} \tag{2}$$

where $c(\mathbf{x}^s, \mathbf{x}^t)$ is the cross-domain cost matrix obtained by the cosine distance, $\Pi[P(\mathbf{X}_s), P(\mathbf{X}_t)]$ denotes all the joint distributions $\gamma(\mathbf{x}^s, \mathbf{x}^t)$. GWD is described as

$$\begin{aligned} \mathcal{D}_{\text{GWD}}[P(\mathbf{X}_s), P(\mathbf{X}_t)] &= \inf_{\gamma \in \Pi[P(\mathbf{X}_s), P(\mathbf{X}_t)]} \\ &\quad \times \mathrm{E}_{\substack{(\mathbf{x}^s, \mathbf{x}^t)\sim\gamma \\ (\hat{\mathbf{x}}^s, \hat{\mathbf{x}}^t)\sim\gamma}}\left[L\left(\mathbf{x}_i^s, \mathbf{x}_j^t, \hat{\mathbf{x}}_i^s, \hat{\mathbf{x}}_j^t\right)\right] \\ L\left(\mathbf{x}_i^s, \mathbf{x}_j^t, \hat{\mathbf{x}}_i^s, \hat{\mathbf{x}}_j^t\right) &= \left\|c_1\left(\mathbf{x}_i^s, \hat{\mathbf{x}}_i^s\right) - c_2\left(\mathbf{x}_j^t, \hat{\mathbf{x}}_j^t\right)\right\| \end{aligned} \tag{3}$$

where $L(\mathbf{x}_i^s, \mathbf{x}_j^t, \hat{\mathbf{x}}_i^s, \hat{\mathbf{x}}_j^t)$ is the cost function evaluating the intragraph structural similarity between two pairs of nodes $(\mathbf{x}_i^s, \hat{\mathbf{x}}_i^s)$ and $(\mathbf{x}_j^t, \hat{\mathbf{x}}_j^t)$. A transport plan shared by WD and GWD is employed to integrate the WD and GWD [39] and obtain the GOT distance.

III. Proposed Cross-Domain Few-Shot Learning Method

The flowchart of the proposed Gia-CFSL is shown in Fig. 1, which contains three components, i.e., feature extraction, FSL, and domain alignment. The feature extraction component includes the mapping layer of SD and TD, and the embedding feature extractor with shared weights, in which the mapping layer is used to map the original features of SD and TD to the same dimension. In the FSL component, the Euclidean

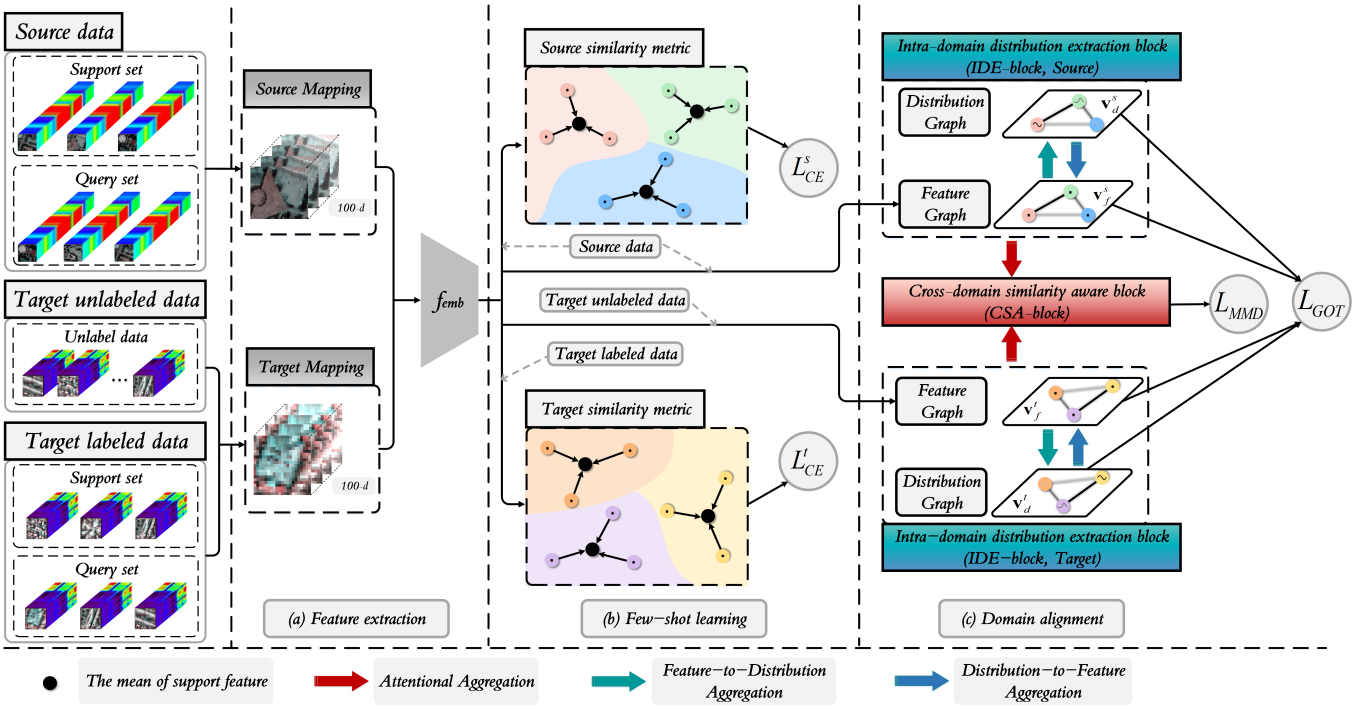

Fig. 1. Flowchart of the proposed Gia-CFSL, including feature extraction, FSL, and domain alignment.

distance is used to measure similarity, and the potential relationship between support samples and query samples in each meta-task of SD and TD is learned, respectively. Domain alignment is composed of IDE-block and CSA-block, in which the domain distribution characteristics are obtained through aggregation and information propagation, and domain shifts can be reduced at feature and distribution levels.

A. Feature Extraction Preprocess

Due to different sensors of SD and TD, the spectral resolutions of samples are inconsistent. Therefore, the mapping layer is used to map two domains to the same dimension, and then the deep residual 3-D CNN network is used to extract the spatial–spectral embedding features. The parameter settings of the above-mentioned two modules are the same as deep cross-domain FSL (DCFSL) [40] for a fair comparison.

The two mapping layers denoted as $\mathbf{M}_s(\bullet)$ for SD and $\mathbf{M}_t(\bullet)$ for TD, are implemented by a layer of 2-D convolution, where the size of the convolution kernel is 9×9, and map the SD and TD in their original spectral dimensions (d_s and d_t) to the mapping dimension d_{map} (d_{map} is set to 100 in the experiment). The deep residual 3-D CNN network, denoted as $f_{\text{emb}}(\bullet)$, consists of two 3-D residual blocks (three 3-D convolution layers and a shortcut layer), two maximum pooling layers and a 3-D convolution layer. The convolution kernel size is $3 \times 3 \times 3$, and the output dimension of $f_{\text{emb}}(\bullet)$ is denoted as d_{emb} (the embedding feature dimension).

B. Source and Target Few-Shot Learning

The embedding features are input into the FSL part to perform FSL task learning for HSI classification using the Euclidean distance. FSL of SD and TD is performed simultaneously in each iteration. Taking SD as an example, the support set and query set are input into $f_{\text{emb}}(\mathbf{M}_s(\bullet))$ to the embedding features $\mathbf{Z}_s$. In each episode, FSL is performed by calculating the Euclidean distance between the embedding features of the query set and the prototype of each class (i.e., the mean value of embedding features of the support set), and the prediction loss of the query set is minimized. The predicted probability output of a query sample $\mathbf{x}^s_m$ is obtained by the calculated Euclidean distance through the softmax function

$$P\left(\hat{y}^s_m \middle| \mathbf{x}^s_m\right) = \text{Softmax}\left(-\text{ED}\left(\mathbf{z}^s_m, \mathbf{z}^{\text{mean}(c)}_n\right)\right) \tag{4}$$

where $\text{ED}(\bullet)$ denotes an Euclidean distance function, $\mathbf{z}^s_m$ is the query set embedding features, $\mathbf{z}^{\text{mean}(c)}_n$ is the mean value of the cth class support set embedding features, also known as the cth class prototype ($c \in C_s$). The FSL loss of a query sample in SD is calculated by cross entropy loss

$$\mathcal{L}^s_{\text{CE}}\left(P\left(\hat{y}^s_m \middle| \mathbf{x}^s_m\right), y^s_m\right) = -y^s_m \log P\left(\hat{y}^s_m \middle| \mathbf{x}^s_m\right). \tag{5}$$

The calculation process for $\mathcal{L}^t_{\text{CE}}$ is similar to that of earlier, but note that L^t_{CE} is obtained based on the meta-task constructed using $\mathcal{T}^{\text{tr}}_t$ after data augment.

C. Domain Alignment With IDE- and CSA-Block

Given the effect of domain shift on classification performance in the procedure of FSL episodic training, the IDE-block and CSA-block are designed for domain alignment in the proposed Gia-CFSL framework.

1) Intradomain Distribution Extraction Block: The FG containing nonlocal relationships is constructed by using the embedding features of SD and TD for each meta-task (C-way

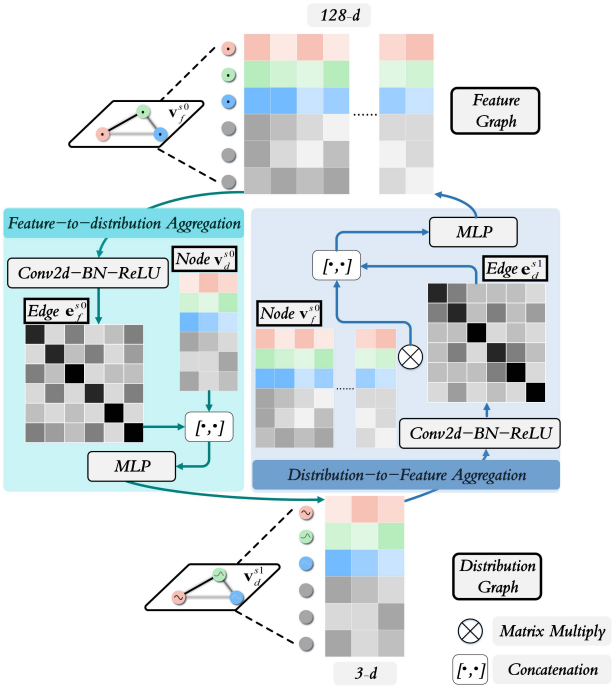

Fig. 2. Calculation process of IDE-block, we take the 128 embedding feature dimension (d_{emb}) and three-way one-shot of SD as example, where the dimension d_{emb} of $\mathbf{v}_f^{s0}$ is 128 and the dimension of $\mathbf{v}_d^{s1}$ is $C \times K = 3$ (three support samples with different colors and three query samples with gray in FG $\mathbf{v}_f^{s0}$).The green arrow represents feature-to-distribution aggregation, and the blue arrow represents distribution-to-feature aggregation.

K-shot) $\mathbf{G}_f^{s/t} = (\mathbf{V}_f^{s/t}, \mathbf{E}_f^{s/t})$ (s/t represents SD or TD and f represents feature), and IDE-block is designed to generate the DG, $\mathbf{G}_d^{s/t} = (\mathbf{V}_d^{s/t}, \mathbf{E}_d^{s/t})$ (d represents distribution). Taking SD as an example, the internal structure and calculation process of IDE-block are illustrated in Fig. 2, where the flowchart of IDE-block of SD under the three-way one-shot episode is also shown.

a) Feature graph: The embedding features are separated into support set $\mathcal{S}_s$ and query set $\mathcal{Q}_s$ as the initialization of node features in FG

$$\mathbf{v}_{f(i)}^{s0} = f_{\text{emb}}\left(\mathbf{M}_s\left(\mathbf{x}_i^s\right)\right) \tag{6}$$

where the node $\mathbf{v}_{f(i)}^{s0} \in \mathbb{R}^{d_{\text{emb}}}$ is the feature-level information and represents a land cover class that contains 9×9 spatial patch information. The edge in FG is constructed by calculating the similarity between two nodes

$$\mathbf{e}_{f(i,j)}^{s0} = f_{\text{FG}}\left(\left(\mathbf{v}_{f(i)}^{s0} - \mathbf{v}_{f(j)}^{s0}\right)^2\right) \tag{7}$$

where $f_{\text{FG}}(\bullet)$ consists of two Conv2d-BN-ReLU blocks to enhance the nonlinear representation of nonlocal relationships.

b) Feature-to-distribution aggregation: In order to capture the intradomain distribution-level information, the similarity between nodes on the edges of FG is aggregated as the node features of DG

$$\mathbf{v}_{d(i)}^{s1} \leftarrow \text{MLP}\left(\left[\mathbf{e}_{f(i,j)}^{s0}, \mathbf{v}_{d(i)}^{s0}\right]\right) \tag{8}$$

where multilayer perceptron (MLP) consists of a fully connected layer and an ReLU, which is used to aggregate the FG edge features $\mathbf{e}_{f(i,j)}^{s0}$ and the initialization of DG node features $\mathbf{v}_{d(i)}^{s0}$ to obtain the new DG node features $\mathbf{v}_{d(i)}^{s1}$, the jth value in $\mathbf{v}_{d(i)}^{s1} \in \mathbb{R}^{C\times K}$ represents the nonlocal relationship between node i and j, $[\bullet, \bullet]$ is the concatenation operator, and $\mathbf{v}_{d(i)}^{s0}$ is the initialization of DG node features

$$\mathbf{v}_{d(i)}^{s0} = \begin{cases} \left[\delta\left(y_i^s, y_j^s\right)\right]_{j=1}^{C\times K}, & \mathbf{x}_i^s \in \mathcal{S}_s \\ \left[\dfrac{1}{\text{CK}}, \ldots, \dfrac{1}{CK}\right], & \mathbf{x}_i^s \in \mathcal{Q}_s \end{cases} \tag{9}$$

where $\delta(\bullet, \bullet)$ denotes the Kronecker delta function which outputs one when the labels of two support samples are consistent (i.e., $y_i^s = y_j^s$) and zero when $y_i^s \neq y_j^s$.

c) Distribution graph: The similarity between nonlocal relationships is acquired by calculating the DG node features $\mathbf{v}_{d(i)}^{s1}$

$$\mathbf{e}_{d(i,j)}^{s1} = f_{\text{DG}}\left(\left(\mathbf{v}_{d(i)}^{s1} - \mathbf{v}_{d(j)}^{s1}\right)^2\right) \tag{10}$$

where $f_{\text{DG}}(\bullet)$ represents two Conv2d-BN-ReLU blocks. In the produced DG, the high-order distribution information is accessed based on the nonlocal relationships among samples in a meta-task.

d) Distribution-to-feature aggregation: The FG node features are updated by aggregating the DG edge features $\mathbf{e}_{d(i,j)}^{s1}$ and the initialization of FG node features $\mathbf{v}_{f(i)}^{s0}$

$$\mathbf{v}_{f(i)}^{s1} \leftarrow \text{MLP}\left(\left[\left(\sum_j \mathbf{e}_{d(i,j)}^{s1}\mathbf{v}_{p(j)}^{s0}\right), \mathbf{v}_{f(i)}^{s0}\right]\right) \tag{11}$$

where MLP is two Conv2d-BN-ReLU blocks. IDE-block corresponding to TD is the same. In order to make full use of the unlabeled data in TD, the query samples are randomly selected from $\mathcal{T}_t^{\text{te}}$, and the support samples are consistent with the FSL of TD.

e) Cross-domain graph alignment: After producing FG and DG corresponding to SD and TD, respectively, WD and GWD of feature and distribution levels are calculated by $\mathbf{v}_{f(i)}^{s1}$ and $\mathbf{v}_{d(i)}^{s1}$. It is worth noting that WD and GWD measure the distance of node and edge between two graphs. The fusion of these two distances effectively considers the node and edge information, in order to achieve better graph alignment. In order to achieve the best integration of WD and GWD, a transformation matrix $\mathbf{T}$ shared by WD and GWD is considered [39]. GOT distance is defined as

$$\begin{aligned}\text{GOT}\left(\mathbf{G}^s, \mathbf{G}^t\right) = \sum_{i,\hat{i},j,\hat{j}} &\left(\mathbf{T}_{ij} \cdot c\left(\mathbf{v}_i^s, \mathbf{v}_j^t\right)\right. \\ &\left. + \mathbf{T}_{\hat{i}\hat{j}} \cdot L\left(\mathbf{v}_i^s, \mathbf{v}_j^t, \hat{\mathbf{v}}_i^s, \hat{\mathbf{v}}_j^t\right)\right)\end{aligned} \tag{12}$$

where $\mathbf{T}_{ij}$ and $\mathbf{T}_{\hat{i}\hat{j}}$ are the transformation matrix of two pairs of nodes (i.e., $(\mathbf{v}_i^s, \mathbf{v}_j^t)$ and $(\hat{\mathbf{v}}_i^s, \hat{\mathbf{v}}_j^t)$). Then, GOT loss for cross-domain graph alignment is formulated as

$$L_{\text{GOT}} = \text{GOT}\left(\mathbf{G}_f^s, \mathbf{G}_f^s\right) + \text{GOT}\left(\mathbf{G}_d^s, \mathbf{G}_d^s\right). \tag{13}$$

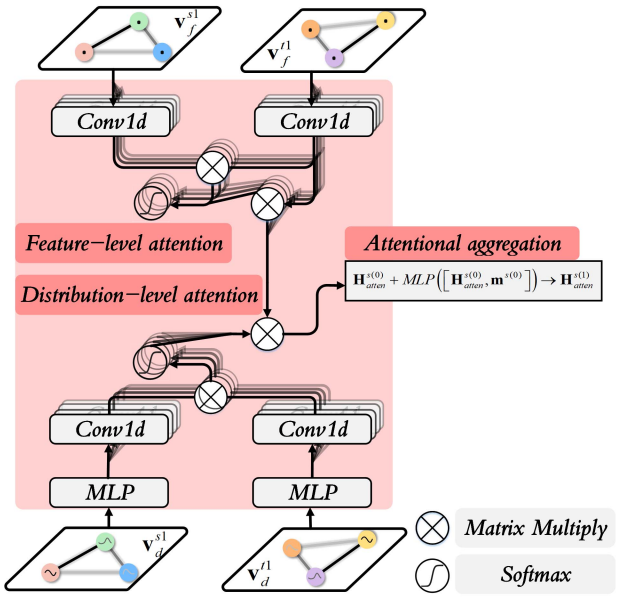

Fig. 3. Calculation process of CSA-block. The feature- and distribution-level attention features are calculated successively by $(\mathbf{v}_{f(i)}^{s1}, \mathbf{v}_{f(i)}^{t1})$ and $(\mathbf{v}_{d(i)}^{s1}, \mathbf{v}_{d(i)}^{t1})$, and the aggregation operation is carried out by MLP to output $\mathbf{H}_{\text{atten}}^{s(1)}$.

2) Cross-Domain Similarity Aware Block: The main purpose of IDE-block is to characterize and aggregate the non-local relationships among samples in SD and TD, as well as to align feature- and distribution-level graphs. However, the lack of cross-domain information interaction is not sufficient to reduce the influence of domain shift in FSL only by minimizing GOT loss. Therefore, CSA-block is further designed to generate feature-level attention and distribution-level high-order attention between SD and TD, which captures interdomain feature similarity and distribution similarity, respectively, and makes them collaborate through attentional aggregation, to reduce cross-domain discrepancy together with IDE-block. The calculation process of the CSA block for SD is shown in Fig. 3.

a) Feature-level attention: Generally, in the attention mechanism, a task-related query vector is given, and the attention map calculated by query and key is attached to the value, in order to obtain the attention feature. In the feature-level attention stage for SD, the generated FG $\mathbf{v}_{f(i)}^{s1}$ of SD is used as a query, and the FG $\mathbf{v}_{f(i)}^{t1}$ of TD is used as key and value. Specifically, the linear self-representation of $\mathbf{v}_{f(i)}^{s1}$ and $\mathbf{v}_{f(i)}^{t1}$ is carried out, and the feature-level attention map of SD relative to TD is calculated as

$$\alpha_{f(i,j)}^{s\to t} = f\left(\mathbf{v}_{f(i)}^{s1}, \mathbf{v}_{f(j)}^{t1}\right) = \text{Softmax}\left(\left(C\left(\mathbf{v}_{f(i)}^{s1}\right)\right)^T C\left(\mathbf{v}_{f(j)}^{t1}\right)\right) \tag{14}$$

where $C(\bullet)$ represents 1-D convolution. In Fig. 3, multiple Conv1d layers represent multihead attention mechanism (four heads are set in the experiment), in which each head generates independent attention to improve model expressivity. The feature-level attention feature is acquired by weighted averaging of each element in $\mathbf{v}_{f(i)}^{t1}$ according to the weight $\alpha_{f(i,j)}^{s\to t}$

$$f\left(\mathbf{v}_{f(i)}^{s1}, \mathbf{v}_{f(j)}^{t1}\right)\mathbf{v}_{f}^{t(l)} = \sum_j \alpha_{f(i,j)}^{s\to t}\mathbf{v}_{f(j)}^{t1}. \tag{15}$$

b) Distribution-level attention: In the distribution-level attention stage for SD, the extracted DG $\mathbf{v}_{d(i)}^{s1}$ and $\mathbf{v}_{d(i)}^{t1}$ of SD and TD are used as query and key. First, the MLP consisting of two Conv1d-BN-ReLU blocks is used to increase dimension of DG from $C \times K$ to 128, and the 1-D convolution is used for self-representation. The distribution-level attention map of SD relative to TD is also calculated by multihead attention mechanism

$$\alpha_{d(i,j)}^{s\to t} = f\left(\mathbf{v}_{d(i)}^{s1}, \mathbf{v}_{d(j)}^{t1}\right) = \text{Softmax}\left(\left(C\left(\mathbf{v}_{d(i)}^{s1}\right)\right)^T C\left(\mathbf{v}_{d(j)}^{t1}\right)\right). \tag{16}$$

Taking the feature-level attention feature as a Value, and applying the distribution-level high-order attention map to obtain the final attention feature,

$$f\left(\mathbf{v}_{d(i)}^{s1}, \mathbf{v}_{d(j)}^{t1}\right)\left(f\left(\mathbf{v}_{f(i)}^{s1}, \mathbf{v}_{f(j)}^{t1}\right)\mathbf{v}_{f(j)}^{t1}\right) = \sum_j \alpha_{d(i,j)}^{s\to t}\alpha_{f(i,j)}^{s\to t}\mathbf{v}_{f(j)}^{t1}. \tag{17}$$

Since FG and DG are mutually generated in IDE-block, Eq.17 enables the distribution-level high-order attention map to collaborate with the above feature-level attention map, in order to capture both feature and distribution similarity of SD relative to TD.

c) Attentional aggregation: The generated attention feature is regarded as the attentional message $\mathbf{m}^{s(\ell-1)}$ of the $(\ell-1)$th layer, which propagates to the ℓth layer through aggregation operation with the $(\ell-1)$th output $\mathbf{H}_{\text{atten}}^{s(\ell-1)}$,

$$\mathbf{H}_{\text{atten}}^{s(\ell)} \leftarrow \mathbf{H}_{\text{atten}}^{s(\ell-1)} + \text{MLP}\left(\left[\mathbf{H}_{\text{atten}}^{s(\ell-1)}, \mathbf{m}^{s(\ell-1)}\right]\right) \tag{18}$$

where ℓ is the number of layers ($\ell = 1$ in the experiment), and the FG $\mathbf{v}_{f(i)}^{s1}$ of SD is the input of the initial layer $\mathbf{H}_{\text{atten}}^{s0}$.

CSA-block corresponding to TD is the same as the above-mentioned process. Different from the CSA-block corresponding to SD, FG $\mathbf{v}_{f(i)}^{t1}$ of TD is used as the query, and FG $\mathbf{v}_{f(i)}^{s1}$ of SD is used as the Key and Value in feature-level attention stage. In addition, FG $\mathbf{v}_{f(i)}^{t1}$ of TD acts as the input of the initial layer in attentional aggregation. The linear MMD distance is used to align the attentional aggregation output of SD and TD as

$$\mathcal{L}_{\text{MMD}}\left(\mathbf{H}_{\text{atten}}^{s1}, \mathbf{H}_{\text{atten}}^{t1}\right) = \left\| \frac{1}{N}\sum_{i=1}^{N}\phi\left(\mathbf{h}_{\text{atten}(i)}^{s1}\right) - \frac{1}{N}\sum_{j=1}^{N}\phi\left(\mathbf{h}_{\text{atten}(j)}^{t1}\right) \right\|_{\text{H}}^2 \tag{19}$$

where $N = C \times K + T$. The mean discrepancy is calculated by (19) in the reproducing kernel Hilbert space.

Then, the total loss of Gia-CFSL is defined as

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}}^{s} + \mathcal{L}_{\text{CE}}^{t} + \lambda_1\mathcal{L}_{\text{GOT}} + \lambda_2\mathcal{L}_{\text{MMD}} \tag{20}$$

where λ_1 and λ_2 are the regularization parameters. In the proposed Gia-CFSL, embedding features are used in the training stage of SD and TD, respectively, to conduct the FSL episode

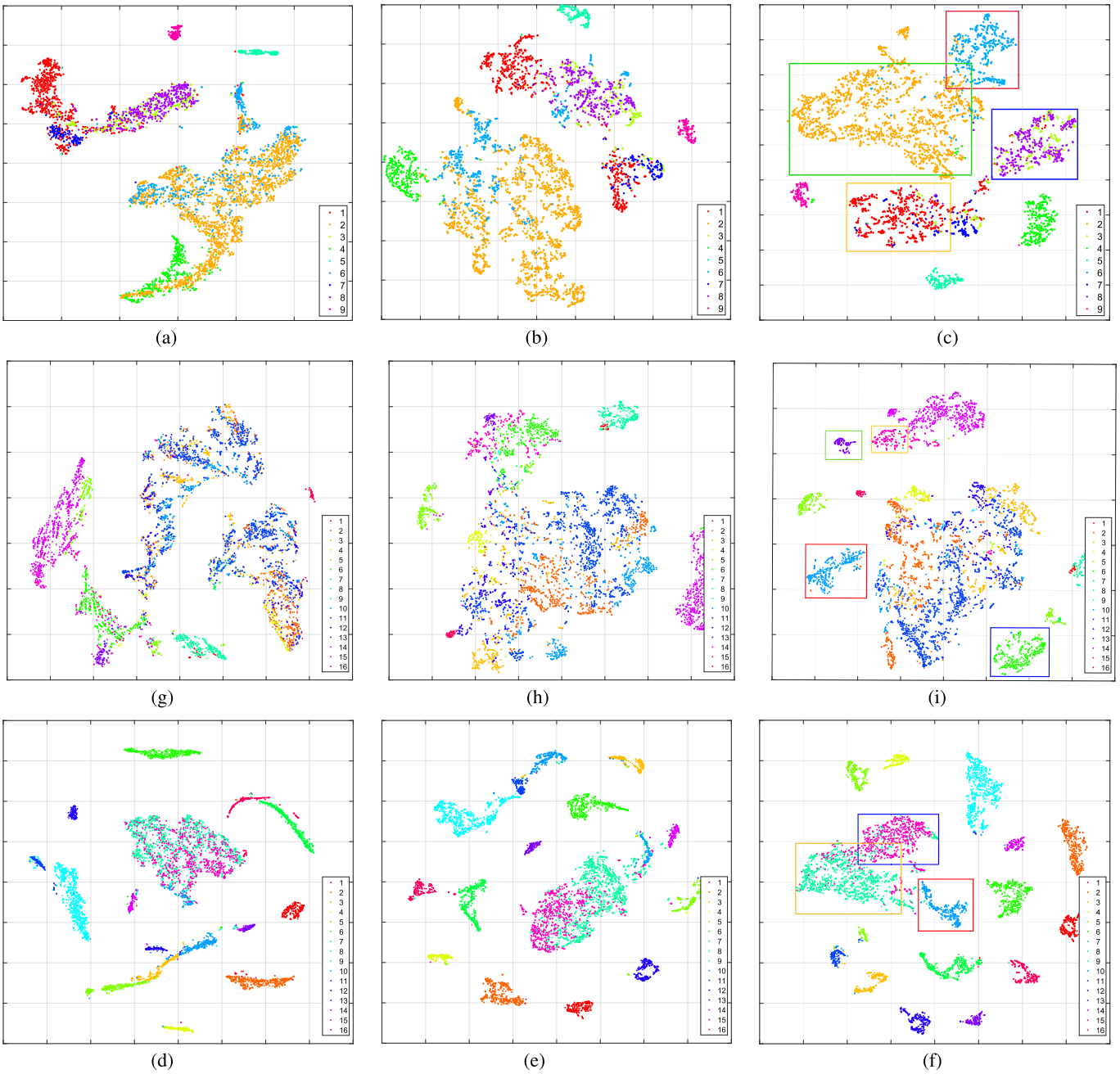

Fig. 4. Class separability of the proposed Gia-CFSL using Pavia University, Indian Pines, and Salinas, and it is obvious that features by Gia-CFSL have the best class separability. (a) Original samples from Pavia University. (b) Features by FSL. (c) Features by Gia-CFSL. (g) Original samples from Indian Pines. (h) Features by FSL. (i) Features by Gia-CFSL. (d) Original samples from Salinas. (e) Features by FSL. (f) Features by Gia-CFSL.

training and domain alignment, $\mathcal{T}_t^{\text{tr}}$ is regarded as the support set and $\mathcal{T}_t^{\text{te}}$ as the query set in the testing stage, and nearest neighbor classifier is applied to predict labels for the query features of TD output by the embedding feature extractor.

D. Alignment Performance of Gia-CFSL

In real HSI, different land cover classes may have similar characteristics in the spectral dimension, resulting in poor class separability using the original samples. Fig. 4(a), (d), and (g) are 2-D visualization of the original samples obtained from Pavia University, Indian Pines, and Salinas (the data are detailed in Section IV-A), which is visualized by t-SNE. Specifically, after training Gia-CFSL, the three TDs are input into the feature extractor to obtain the features with dimension d_{emb}, and then t-SNE is used to project the features for 2-D visualization. It is obvious that the distribution of samples in some classes is too scattered, and there are mixed samples between these classes, such as the second class (meadows) and sixth class (bare soil) in Pavia University, the sixth class (grass-tree) and 15th class (buildings-grass-trees-drives) in Indian Pines, and the eighth class (grape untrained) and 15th class (Vinyard untrained) in Salinas.

The objective of FSL is to learn the common parts of different meta-tasks through a large number of episodic training procedures, such as the sample similarity measurement. In the proposed Gia-CFSL, FSL and domain alignment are combined to alleviate the negative impact of domain shift, in order to learn a better class separability metric space on TD. The features output by Gia-CFSL, as shown in Fig. 4(c), (f), and (i), are much more separable than the features' output by FSL without domain alignment in Fig. 4(b), (e), and (h). It confirms that the domain alignment part composed of IDE-block and CSA-block does improve the separability of metric spaces during the episodic training.

IV. Experimental Results and Discussion

Considering different SD and TD sensors and land cover classes in practical applications, the Chikusei dataset is selected as the source scene, Pavia University, Indian Pines, and Salinas are used as target scenes to verify the effectiveness of Gia-CFSL. Several state-of-the-art deep domain adaptive and FSL-based algorithms are employed for comparison algorithms, including SVM, with the radial basis function (RBF) kernel, 3-dimensional CNN (3DCNN) [41], patch-to-patch CNN (PToP CNN) [11], DSAN [18], dynamic adversarial adaptation network (DAAN) [42], distribution propagation graph network (DPGN) [43], DFSL [33], RN-FSC [34], and DCFSL [40]. The class-specific accuracy (CA), the overall accuracy (OA), and the Kappa coefficient (KC) are employed to evaluate the classification performance.

A. Experiment Data

1) Source Data: Chikusei: This airborne hyperspectral data were captured by the Headwall Hyperspec-Vnir-C sensor in Chikusei, Japan, on July 29, 2014. The Chikusei data have 128 bands in the spectral range from 363 to 1018 nm, with a size of 2517 × 2335 and a spatial resolution of 2.5 m. There are 19 classes, including urban and rural areas. The name of land cover classes and the number of samples are listed in Table I. Additionally, their pseudocolor picture and ground truth maps are shown in Fig. 5.

2) Target Data:

a) Pavia University: The data were gathered by reflective optics spectrographic image system (ROSIS), with spectral coverage of 430 to 860 nm, including 103 spectral bands, 610 × 340 pixels, and 1.3 m the spatial resolution. The ground truth contains nine classes, and the name of land cover classes and the number of samples are listed in Table II. The pseudocolor picture is shown in Fig. 6(a).

b) Indian Pines: The data were selected from the scene at the Indiana Pines proving ground in northwest Indiana and captured the AVIRIS sensor in June 1992. The image scene includes crops and natural vegetation, with 145 × 145 pixels, a spatial resolution of 20 m. A total of 220 bands cover the visible to middle infrared spectrum. After bad band removal, 202 bands were used. There were 16 different land cover classes. The class-specific numbers are listed in Table III. Additionally, a pseudocolor picture is shown in Fig. 6(b).

TABLE I
Number of Source Samples for the Chikusei Dataset

Class	Name	Number of Samples
1	Water	2345
2	Bare soil (school)	2859
3	Bare soil (park)	236
4	Bare soil (farmland)	48525
5	Natural plants	4297
6	Weeds m farmland	1108
7	Forest	20516
8	Grass	6515
9	Rice field (groim)	13369
10	Rice field (first stage)	1268
11	Row crops	5961
12	Plastic house	2193
13	Manmade (nan-dark)	1220
14	Manmade (dark)	7664
15	Manmade (blue)	431
16	Manmade (red)	222
17	Manmade grass	1040
13	Asphalt	801
19	Paved ground	145
Total		77592

TABLE II
Number of Source Samples for the Pavia University Data

Class	Name	Number of Samples
1	Asphalt	6631
2	Meadows	18549
3	Gravel	2099
4	Trees	3064
5	Sheets	1345
6	Bare soil	5029
7	Bitumen	1330
8	Bricks	3682
9	Shadow	947
Total		42776

c) Salinas: The Salinas data were collected by the AVIRIS sensor in Salinas Valley, CA, USA, including 512 × 217 pixels, with a spatial resolution of 3.7 m. It also contains 224 bands. After deleting invalid bands, the number of bands used in the experiment was 200. The numbers of samples for 16 crop classes are listed in Table IV, and a pseudocolor picture is shown in Fig. 6(c).

B. Experimental Setting

Gia-CFSL is implemented on the PyTorch platform. The input is set as a patch size of 9 × 9. Adaptive moment estimation (Adam) is used as the optimization scheme. All convolutional layers and linear layers are initialized by the Xavier normalization, and the batch norm layers use the normal distribution $N(1, 0.02)$. The default value for ℓ_2-norm regularization of feature extraction is 0, and the other modules are set to 1e-4 for weight decay, which stabilizes network training to reduce overfitting, and based on the mini meta-task, the number of training iterations is 10 000 to prevent too few training times from being able to fully learn the prototype of each class.

For each meta-task of C-way K-shot in episodic training, C is set as the number of classes of TD, nine for the Pavia University, and 16 for the Indian Pines and Salinas. K is

TABLE III
NUMBER OF TRAINING AND TESTING SAMPLES FOR THE INDIAN PINES DATA

Class	Name	Number of Samples
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-nimtill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-tree	730
7	Grass-pasture-mowed	20
8	Hay-windrowed	473
9	Oats	20
10	Soybean-notill	972
11	Soybean-miiitill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	336
16	Stone-Steel-Towers	93
Total		10249

TABLE IV
NUMBER OF TRAINING AND TESTING SAMPLES FOR THE SALINAS DATA

Class	Name	Number of Samples
1	Brocoli green weeds 1	2009
2	Brocoli green weeds 2	3726
3	Fallow	1976
4	Fallow rough plow	1394
5	Fallow smooth	2678
6	Stubble	3959
7	Celery	3579
8	Grape untrained	11271
9	Soil vinyard develop	6203
10	Corn senesced green weeds	3278
11	Lettuce romaine 4wk	1063
12	Lettuce romaine 5wk	1927
13	Lettuce romaine 6wk	916
14	Lettuce romaine 7wk	1070
15	Vinyard untrained	7268
16	Vinyard vertical trellis	1807
Total		54129

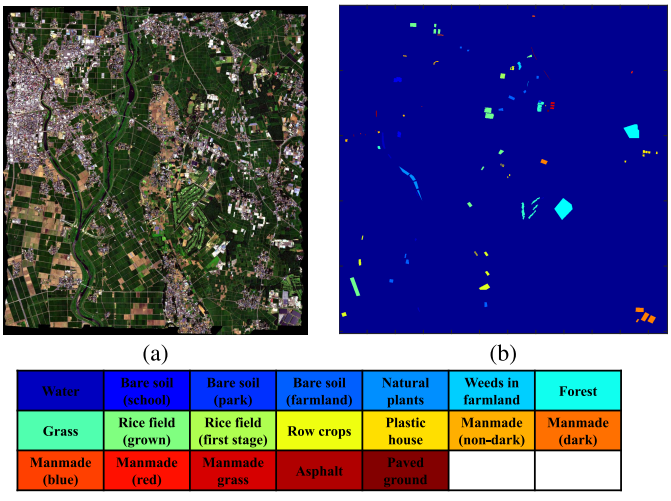

Fig. 5. Pseudocolor image and ground truth map of Chikusei: (a) pseudocolor image of Chikusei and (b) ground truth map of Chikusei.

regarded as that K labeled samples are randomly selected to form the support sets of $\mathcal{T}_s$ and $\mathcal{T}_t^{\text{tr}}$ in each episode for model training, and K for SD FSL and TD FSL is set to 1 in all experiments. In addition, the query sets are used to evaluate the learned classifier, and the more samples it contains, the better it is to simulate small sample classes in TD. For a fair comparison, the number of samples in $\mathcal{Q}$ is set to 19, which is consistent with that in DCFSL [40]. Note that there are 19 classes in SD ($C_s > C_t$, which ensures the diversity of SD training samples), and C classes are randomly extracted from SD when constructing a meta-task. Furthermore, five labeled samples are selected from each class of TD for TD FSL in all experiments. However, five labeled samples are too few for $\mathcal{S}_t$ and $\mathcal{Q}_t$ of TD to be constructed, so the data is augmented by adding Gaussian random noise. For the testing phase, $\mathcal{T}_t^{\text{tr}}$ and $\mathcal{T}_s$ are treated as support set and query set, respectively. Only feature extraction and FSL in GiA-CFSL are retained to obtain the embedding features of the support set (prototype) and query set. The nearest neighbor classifier is used to compare the distance between query embedding features and class prototypes to predict query labels.

Fig. 6. Pseudocolor image: (a) Pavia University, (b) Indian Pines, and (c) Salinas.

TABLE V
PARAMETER TUNING OF THE BASE LEARNING RATE η FOR THE PROPOSED GIA-CFSL USING THE THREE EXPERIMENTAL DATA

Target data	Base learning rate η				
	1e-5	1e-4	1e-3	1e-2	1e-1
University of Pavia	73.20	77.53	79.56	**83.53**	80.07
Indian Pines	53.29	54.78	65.61	**66.58**	60.97
Salinas	84.38	85.31	86.73	**90.27**	90.20

C. Parameter Tuning

A parameter sensitivity analysis is conducted to evaluate the sensitivity of Gia-CFSL on the three TDs. The regularization parameters, base learning rate and embedding feature dimension, i.e., λ_1, λ_2, η, and d_{emb}, regarded as adjustable hyperparameters are selected from $\{1e-3, 1e-2, 1e-1, 1e+0, 1e+1, 1e+2\}$, $\{1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}$, and $\{64, 128, 256\}$, respectively. The tenfold cross-validation is adopted to select the optimal parameters.

In three experimental datasets with different combinations of λ_1 and λ_2, the variation trend of Gia-CFSL classification

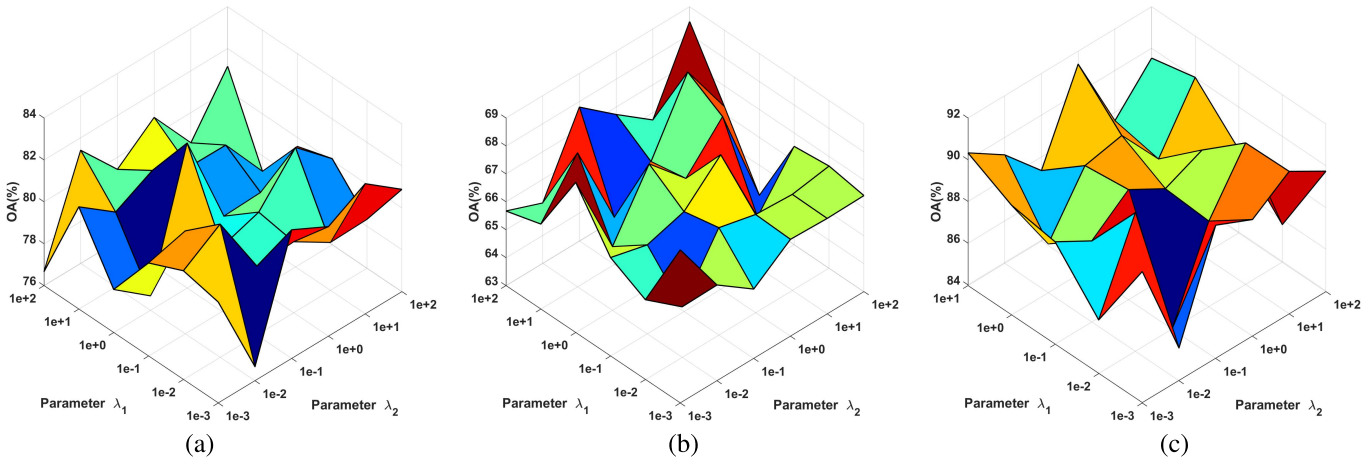

Fig. 7. Parameter tuning of λ_1 and λ_2 for the proposed Gia-CFSL using all the three experimental data. (a) Pavia University. (b) Indian Pines. (c) Salinas.

TABLE VI
PARAMETER TUNING OF d_{emb} FOR THE PROPOSED GIA-CFSL (FIVE LABELED SAMPLES FROM TD)

Data set	Pavia University			Indian Pines			Salinas		
d_{emb}	64	128	256	64	128	256	64	128	256
OA (%)	82.14±2.93	**84.47±1.84**	81.67±2.34	**67.42±1.80**	65.29±2.85	63.51±2.13	88.69±1.84	**89.95±0.93**	89.18±2.06
KC (κ)	76.76±3.60	**79.71±2.18**	76.14±2.82	**63.01±2.03**	60.59±3.18	58.05±2.64	87.43±2.03	**88.82±1.03**	87.57±2.27

TABLE VII
ABLATION COMPARISON OF EACH VARIANT OF GIA-CFSL (FIVE LABELED SAMPLES FROM TD)

Model	FSL	Gia-CFSL (no FSL)	Gia-CFSL (no GOT)	Gia-CFSL (no MMD)	Gia-CFSL
Data set	Pavia University				
OA (%)	81.76±3.08	69.31±2.42	83.14±1.34	83.61±2.82	**84.47±1.84**
KC (κ)	76.42±3.63	59.45±2.86	78.05±2.84	78.71±3.36	**79.71±2.18**
	Indian Pines				
OA (%)	65.03±3.91	51.08±2.74	65.67±1.45	66.36±2.21	**67.42±1.80**
KC (κ)	60.45±4.38	43.95±3.05	60.96±1.94	61.86±2.74	**63.10±2.03**
	Salinas				
OA (%)	87.53±1.76	74.11±2.17	87.92±2.01	88.63±2.30	**89.95±0.93**
KC (κ)	86.12±1.93	72.84±2.61	86.65±2.28	87.36±2.55	**88.82±1.03**

accuracy is shown in Fig. 7. In order to reduce the pressure on hyperparametric optimization for different experimental datasets, suboptimal parameters are selected. For Pavia University, Indian Pines, and Salinas, the suboptimal parameters λ_1 and λ_2 are $1e+0$ and $1e-1$, respectively. In gradient descent, the gradient of the loss function is used to estimate hyperparameters of the model weight after being adjusted by the learning rate. Table V provides classification results corresponding to different base learning rates in three datasets. The optimal base learning rate corresponding to three datasets is $1e-2$. In addition, the OA of all experimental datasets in three embedding feature dimensions is listed in Table VI, and the optimal d_{emb} is 128 for Pavia University and Salinas and 64 for the Indian Pines.

D. Ablation Study

The proposed framework mainly consists of FSL, IDE-block, and CSA-block for domain alignment. To assess the contribution of key components of Gia-CFSL, ablation analyses are conducted by removing each component from the entire framework.

There are four variants in the ablation analyses: 1) "FSL:" only the feature extraction and FSL are retained; 2) "Gia-CFSL (no FSL):" the FSL is deleted from Gia-CFSL; 3) "Gia-CFSL (no GOT):" the GOT loss is deleted; and 4) "Gia-CFSL (no MMD):" the MMD loss is removed. The mean and standard deviation of ten experiments for both ablation models are listed in Table VII. It is obvious that the overall Gia-CFSL outperforms other variants and gains large improvements. The OA drops sharply from 2%–3% in FSL, where it is difficult to achieve the expected performance on TD without domain alignment. There are only MMD loss and GOT loss in Gia-CFSL (no FSL), and the classification performance drops significantly due to the lack of supervised classification loss in FSL. Compared with Gia-CFSL (no GOT), the cross-domain graph alignment in Gia-CFSL (no MMD) provides a greater improvement, and further Gia-CFSL achieves relatively high accuracy, indicating CSA-block in domain alignment plays a role as similarity aware by cross-domain attentional aggregation, and further cooperate with IDE-block to reduce the cross-domain discrepancy.

E. Performance on Cross-Scene HSI Classification

To evaluate the performance of Gia-CFSL when SD and TD are obtained from different sensors and the classes included are different, relevant algorithms, including SVM, 3DCNN, PToP CNN, DAAN, DSAN, DPGN, DFSL, RN-FSC, and DCFSL, are used for comparison. The training samples are set as follows. SVM, 3DCNN, and PToP CNN, regarded as supervised methods, need training, and testing samples with

TABLE VIII

CLASS-SPECIFIC AND OVERALL CLASSIFICATION ACCURACY (%) OF DIFFERENT METHODS FOR THE TARGET SCENE PAVIA UNIVERSITY DATA (FIVE LABELED SAMPLES FROM TD)

Class	Classification algorithms									
	SVM	3DCNN [41]	PToP CNN [11]	DAAN [42]	DSAN [18]	DPGN [43]	DFSL [33]	RN-FSC [34]	DCFSL [40]	Gia-CFSL
1	57.31	51.31	46.21	93.38	91.15	84.09	69.05	69.42	81.06	81.16±8.58
2	52.14	70.69	90.42	95.10	94.10	62.90	85.45	92.70	87.74	89.58±6.30
3	56.31	36.10	67.04	43.26	29.16	0.05	56.83	49.95	63.33	59.77±9.38
4	72.03	68.03	66.33	92.30	96.90	89.1	89.52	92.09	92.56	93.17±3.88
5	93.31	92.91	100	99.63	99.63	99.70	99.29	98.45	99.01	99.46±0.73
6	65.68	43.67	68.14	54.32	27.36	56.33	70.50	57.66	74.58	76.95±8.91
7	95.71	69.74	70.46	11.28	58.35	65.94	71.36	69.36	77.74	79.58±5.63
8	70.51	57.52	80.11	0.46	54.02	66.54	58.29	64.07	62.43	74.36±8.47
9	99.58	97.77	96.89	88.38	94.72	0.00	97.02	98.81	98.22	98.56±1.66
OA (%)	61.44	62.57	76.64	76.54	78.43	64.42	78.23	79.84	82.40	**84.47±1.84**
KC (κ)	52.95	53.33	69.41	68.10	70.58	55.42	70.02	72.41	77.11	**79.71±2.18**

TABLE IX

CLASS-SPECIFIC AND OVERALL CLASSIFICATION ACCURACY (%) OF DIFFERENT METHODS FOR THE TARGET SCENE INDIAN PINES (FIVE LABELED SAMPLES FROM TD)

Class	Classification algorithms									
	SVM	3DCNN [41]	PToP CNN [11]	DAAN [42]	DSAN [18]	DPGN [43]	DFSL [33]	RN-FSC [34]	DCFSL [40]	Gia-CFSL
1	41.30	92.68	100	39.13	97.83	95.12	97.44	96.65	95.61	91.60±7.69
2	42.86	38.44	46.33	26.82	28.29	47.15	38.34	45.95	50.44	50.46±7.85
3	39.04	44.85	54.19	90.96	94.70	27.03	43.35	41.25	48.42	44.88±9.45
4	59.49	36.64	64.42	91.14	97.89	56.47	68.45	59.06	79.57	81.61±10.76
5	59.63	71.13	54.76	68.53	90.89	39.75	70.21	65.90	73.89	70.76±8.08
6	84.79	72.69	58.34	13.29	38.08	61.38	76.38	69.51	88.26	84.25±6.89
7	92.86	100	100	25.00	96.43	100	99.77	99.65	99.57	97.10±5.75
8	90.79	83.93	99.14	59.83	81.59	92.39	75.67	76.91	88.44	91.12±4.23
9	90.00	33.33	95.28	20.00	100	100	99.00	100	100	99.26±2.22
10	34.57	64.84	61.48	49.49	52.37	57.91	47.90	26.05	61.71	62.68±7.84
11	0.00	58.04	51.35	45.87	38.17	41.18	57.80	65.36	57.82	66.54±5.99
12	15.01	23.64	43.29	91.57	96.12	47.96	38.13	26.31	40.34	42.06±10.39
13	89.76	91.00	100	15.12	14.63	89.00	98.04	99.28	99.25	97.11±4.58
14	90.91	53.97	82.37	99.60	98.66	78.65	83.08	75.66	87.26	87.10±6.11
15	17.36	56.96	71.01	67.36	57.51	46.72	62.86	69.90	68.71	68.74±10.56
16	86.02	100	99.18	96.77	94.62	98.86	99.94	99.88	98.52	97.47±4.01
OA (%)	42.80	55.93	60.52	57.45	60.74	53.68	58.94	57.84	65.76	**67.42±1.80**
KC (κ)	37.40	50.65	56.06	52.66	56.69	48.12	54.22	53.01	61.43	**63.10±2.03**

TABLE X

CLASS-SPECIFIC AND OVERALL CLASSIFICATION ACCURACY (%) OF DIFFERENT METHODS FOR THE TARGET SCENE SALINAS DATA (FIVE LABELED SAMPLES FROM TD)

Class	Classification algorithms									
	SVM	3DCNN [41]	PToP CNN [11]	DAAN [42]	DSAN [18]	DPGN [43]	DFSL [33]	RN-FSC [34]	DCFSL [40]	Gia-CFSL
1	98.95	95.11	100	99.75	92.83	87.72	94.86	96.45	99.55	99.88±0.17
2	97.02	93.90	97.14	96.16	99.49	99.49	98.68	99.15	99.71	99.34±1.25
3	83.55	81.48	92.33	50.86	61.13	79.76	93.65	85.85	93.68	95.04±6.58
4	92.18	95.90	100	99.71	98.71	98.34	99.39	98.49	99.45	96.99±2.73
5	83.38	75.38	69.75	86.56	99.22	80.13	90.60	82.67	90.39	89.02±6.40
6	89.67	93.42	100	99.92	99.97	99.92	97.64	97.29	99.27	99.52±0.67
7	99.16	98.43	88.22	96.51	97.29	99.86	99.33	99.39	99.04	99.44±0.73
8	68.15	76.42	82.36	91.79	89.22	50.84	78.34	71.59	72.61	80.89±8.73
9	98.23	95.18	97.24	83.15	85.44	89.03	90.19	88.16	99.74	99.55±0.40
10	33.13	55.85	86.31	70.99	78.92	81.24	61.84	69.72	84.51	76.47±7.46
11	75.47	81.09	88.67	46.25	99.34	89.46	96.45	89.29	98.17	97.74±1.89
12	94.29	95.58	96.42	95.49	71.35	99.17	93.11	94.03	99.04	98.39±2.60
13	98.47	95.06	92.32	99.34	99.67	99.56	99.50	99.45	98.97	98.87±0.61
14	92.90	95.02	82.77	98.60	72.52	98.87	97.53	96.58	97.77	95.69±5.37
15	62.29	66.03	30.54	57.18	62.36	59.75	77.46	69.30	74.12	75.76±9.00
16	97.45	80.41	92.26	83.40	74.60	77.69	85.98	81.86	90.62	94.08±4.06
OA (%)	80.43	82.74	81.83	84.06	85.35	78.67	87.67	83.84	88.53	**89.95±0.93**
KC (κ)	78.36	80.77	79.71	82.23	83.66	76.42	86.02	81.75	87.27	**88.82±1.03**

the same dimension and the same classes, so only five label samples from each class in TD are selected as the training set, and the rest are used as the testing set. For the unsupervised domain adaptive methods, DAAN and DSAN, the projection layer consistent with Gia-CFSL is added, and all data of SD with labels and TD data without labels are used for training. For DPGN, DFSL, RN-FSC, and DCFSL developed based on FSL, the selection of training samples is the same as Gia-CFSL. Specifically, DPGN only uses SD construction meta-tasks for training and does not contact TD, DFSL, and

TABLE XI
CLASSIFICATION ACCURACY (%) OF GIA-CFSL IN DIFFERENT TD TRAINING SAMPLES $\mathcal{T}_t^{\text{tr}}$

$\mathcal{T}_t^{tr}$	1	2	3	4	5
Data set	Pavia University				
OA (%)	59.04±4.42	69.46±5.02	75.91±3.65	81.74±4.84	**84.47±1.84**
KC (κ)	49.34±4.31	60.47±5.26	68.92±4.63	76.22±5.11	**79.71±2.18**
	Indian Pines				
OA (%)	46.72±4.12	57.21±4.36	60.33±3.55	64.25±3.20	**67.42±1.80**
KC (κ)	40.25±4.30	51.88±4.66	54.98±3.94	59.40±3.61	**63.10±2.03**
	Salinas				
OA (%)	73.75±3.52	81.63±2.70	85.77±2.02	87.84±1.17	**89.95±0.93**
KC (κ)	71.24±3.81	79.91±2.97	83.06±2.40	86.04±1.53	**88.82±1.03**

RN-FSC use GRBS [44] to select 100 bands in SD and TD to ensure the consistency of input dimension. The optimal base learning rate and regularization parameters of all comparison algorithms are selected from $\{1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}$ and $\{1e-3, 1e-2, 1e-1, 1e+0, 1e+1, 1e+2\}$, respectively, and using cross-validation to find the corresponding optimal parameters.

Tables VIII–X report the CA, OA, and KC of the above-mentioned methods in three TDs. For all the classification methods, five labeled samples are randomly selected for each class as supervised samples in the TD, and the average accuracy and standard deviation of ten experiments are reported. The following analyses are obtained from Tables VIII–X.

1) Compared with SVM, 3DCNN, and PToP CNN, which only use a few TD samples for training, DAAN, DSAN, DFSL, RN-FSC, DCFSL, and Gia-CFSL, which utilize SD knowledge and design domain alignment blocks, provide 5%–10% improvement in OA. It suggests that when there are a few samples in TD, transferring SD knowledge to TD helps improve the classification accuracy of the model on TD.
2) The performance of FSL-based methods, such as DFSL, RN-FSC, DCFSL, and Gia-CFSL, is improved by 4%–8% over unsupervised DA methods, such as DAAN and DSAN, because the learning method of constructing meta-tasks in FSL for episodic training has stronger generalization ability and learns more complete SD knowledge for TD. This demonstrates that the FSL methods trained by the meta-learning idea can better solve the problem of a few labeled samples.
3) Due to the design of domain alignment blocks, Gia-CFSL and DCFSL perform better than DPGN, DFSL, and RN-FSC that only focusing on FSL. Furthermore, IDE-block for feature- and distribution-level alignments is developed in Gia-CFSL, and it is assisted by the CSA-block that can perceive the interdomain similarities to reduce the cross-domain discrepancy. This alignment method is superior to the conditional adversarial DA strategy in DCFSL, and the OA of the three experimental data is improved by approximately 2%.

In order to analyze the influence of different TD training samples on classification performance, $\mathcal{T}_t^{\text{tr}}$ is selected from (1, 2, 3, 4, 5), and the final performance of Gia-CFSL on three datasets is reported in Table XI. With the increase of TD training samples, the classification accuracy of TD improved significantly. Even if each class has only a small amount of supervised information, it is fully utilized by Gia-CFSL to learn the prototype representation of each class.

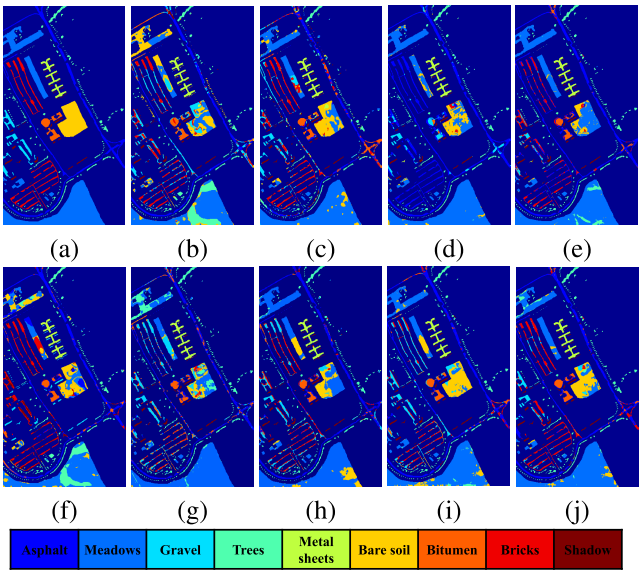

Fig. 8. Data visualization and classification maps for target scene Pavia University data obtained with different methods, including: (a) ground truth map of Pavia University, (b) 3DCNN (62.57%), (c) PToP CNN (76.64%), (d) DAAN (76.54%), (e) DSAN (78.43%), (f) DPGN (64.42%), (g) DFSL (78.23%), (h) RN-FSC (79.84%), (i) DCFSL (82.40%), and (j) Gia-CFSL (85.20%).

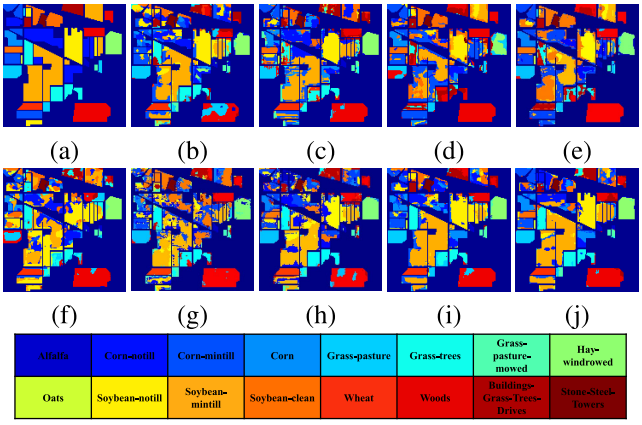

Fig. 9. Data visualization and classification maps for target scene Indian Pines data data obtained with different methods, including: (a) Ground truth map of Indian Pines, (b) 3DCNN (55.93%), (c) PToP CNN (60.52%), (d) DAAN (57.45%), (e) DSAN (60.74%), (f) DPGN (53.68%), (g) DFSL (58.94%), (h) RN-FSC (57.84%), (i) DCFSL (65.76%), and (j) Gia-CFSL (68.48%).

For further visual comparison, the ground truth map of three TDs and classification maps corresponding to five labeled samples of each class in TD are illustrated in Figs. 8–10. In these maps, labeled samples are displayed as ground truth and unlabeled samples as backgrounds. In contrast, the proposed Gia-CFSL obtains less noisy and more accurate results in some areas of the classification maps, such as the second class (Meadows) and sixth class (Bare soil) in Pavia University data, the eighth class (Grape untrained) and 15th

TABLE XII
NUMBER OF PARAMETERS (M, MILLION) AND EXECUTION TIME (IN SECONDS) OF TRAINING IN DIFFERENT METHODS

method		SVM	3DCNN	PToP CNN	DPGN	DAAN	DSAN	DFSL	RN-FSC	DCFSL	Gia-CFSL
#params(M)		-	0.03	10.60	4.41	45.56	24.52	0.03	0.19	4.26	0.31
Computing time	Pavia University	1.42	161.22	9105.32	5912.74	1634.27	2988.45	1305.09	930.84	3905.62	4858.44
	Indian Pines	1.64	157.05	7485.09	7001.41	3204.78	4800.31	2120.13	1351.02	5055.06	6572.23
	Salinas	1.88	189.09	8839.83	8648.24	5487.25	6480.25	2839.89	1485.26	6381.95	7228.99

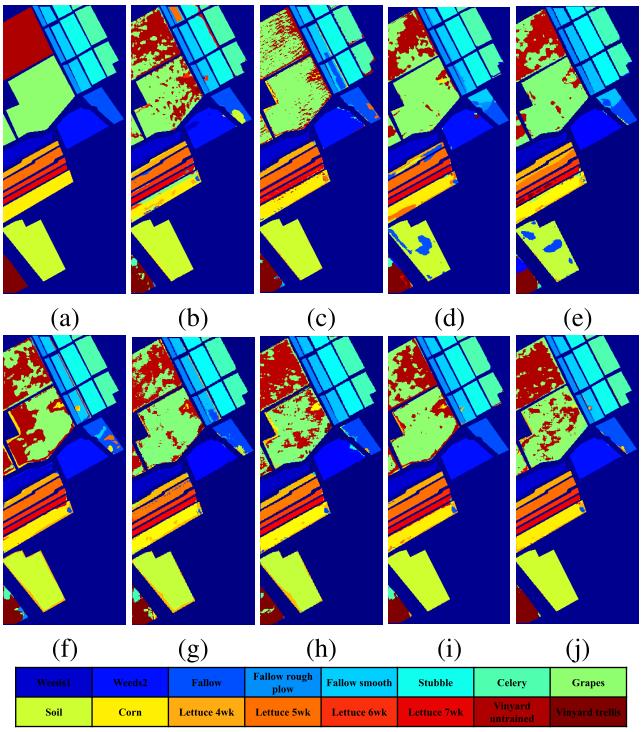

Fig. 10. Data visualization and classification maps for target scene Salinas data data obtained with different methods, including: (a) Ground truth map of Salinas, (b) 3DCNN (82.74%), (c) PToP CNN (81.83%), (d) DAAN (84.06%), (e) DSAN (85.35%), (f) DPGN (78.67%), (g) DFSL (87.67%), (h) RN-FSC (83.84%), (i) DCFSL (88.53%), and (j) Gia-CFSL (90.35%).

class (Vinyard untrained) in Salinas data, the sixth (Grass-tree) and 11th (Soybean-miiitill) in Indian Pines, and the generated map is most similar to the ground truth map. It is basically consistent with the results of class separability analyses of aligned features of Gia-CFSL, as shown in Fig. 4.

To show the computational complexity of different methods, the number of parameters of different methods and training time on all experimental data are listed in Table XII, and the number of training samples in the TD is five labeled samples. All the experiments are carried out using PyTorch on an Intel Core i7-8700 (64-GB RAM) powered with NVIDIA GTX 1080ti GPU with 11-GB memory. As listed in Table XII, PToP CNN has the highest computational cost, which is due to the unsupervised pretraining in patch-to-patch mode. SVM and 3DCNN do not include domain alignment or FSL, but train the classifier directly in the case of a few samples, with small computational complexity. The complexity of Gia-CFSL is similar to DAAN, DCFSL, and other cross-domain classification methods, which indicates that knowledge transfer in cross-domain classification increase computational cost to ensure the performance in TD.

V. CONCLUSION

In this article, Gia-CFSL has been proposed to solve the issues of cross-scene classification, such as the difference in spatial and spectral resolution from separate sensors and one of the land-cover classes. It performs both FSL and domain alignment under the condition that all labeled samples in SD are available and a few labeled samples are in TD. Specifically, the episodic learning pattern of FSL is used to learn the class separability metric space of SD and TD, and DA is considered to reduce the negative impact of domain shift on FSL. Based on the graph distribution information propagation mechanism, an IDE-block is designed in Gia-CFSL to characterize and aggregate intradomain nonlocal relationships, and graph alignment is performed at feature and distribution levels. Furthermore, a CSA-block is developed to capture the interdomain similarity and enhance information interaction between SD and TD. Extensive experiments have been conducted on three datasets, and the proposed method has presented significant improvements over the state-of-the-art models.

REFERENCES

[1] Z. Gong, P. Zhong, and W. Hu, "Statistical loss and analysis for deep learning in hyperspectral image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 322–333, Jan. 2021.
[2] K. Wei, Y. Fu, and H. Huang, "3-D quasi-recurrent neural network for hyperspectral image denoising," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 363–375, Jan. 2021.
[3] C. Liu, J. Li, L. He, A. J. Plaza, S. Li, and B. Li, "Naive Gabor networks for hyperspectral image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 376–390, Jan. 2021.
[4] J. Lu, H. Liu, Y. Yao, S. Tao, Z. Tang, and J. Lu, "HSI ROAD: A hyper spectral image dataset for road segmentation," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2020, pp. 1–6.
[5] R. Dian, S. Li, and L. Fang, "Learning a low tensor-train rank representation for hyperspectral image super-resolution," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2672–2683, Sep. 2019.
[6] R. Dian, S. Li, A. Guo, and L. Fang, "Deep hyperspectral image sharpening," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5345–5355, Nov. 2018.
[7] R. Dian, S. Li, and X. Kang, "Regularizing hyperspectral and multispectral image fusion by CNN denoiser," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 3, pp. 1124–1135, Mar. 2020.
[8] P. Duan, P. Ghamisi, X. Kang, B. Rasti, S. Li, and R. Gloaguen, "Fusion of dual spatial information for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 8, pp. 7726–7738, Sep. 2021.
[9] P. Duan, X. Kang, S. Li, P. Ghamisi, and J. A. Benediktsson, "Fusion of multiple edge-preserving operations for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 12, pp. 10336–10349, 2019.
[10] X. Wang, K. Tan, Q. Du, Y. Chen, and P. Du, "CVA^{2}E: 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.

[11] M. Zhang, W. Li, Q. Du, L. Gao, and B. Zhang, "Feature extraction for classification of hyperspectral and LiDAR data using patch-to-patch CNN," *IEEE Trans. Cybern.*, vol. 50, no. 1, pp. 100–111, Jan. 2019.

[12] X. Xu, W. Li, Q. Ran, Q. Du, L. Gao, and B. Zhang, "Multisource remote sensing data classification based on convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 2, pp. 937–949, Feb. 2018.

[13] X. Zhao *et al.*, "Joint classification of hyperspectral and LiDAR data using hierarchical random walk and deep CNN architecture," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 10, pp. 7355–7370, Oct. 2020.

[14] L. Bruzzone and D. F. Prieto, "Unsupervised retraining of a maximum likelihood classifier for the analysis of multitemporal remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 2, pp. 456–460, Feb. 1999.

[15] Y. Zhang, W. Li, M. Zhang, Y. Qu, R. Tao, and H. Qi, "Topological structure and semantic information transfer network for cross-scene hyperspectral image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Sep. 16, 2021, doi: 10.1109/TNNLS.2021.3109872

[16] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 97–105.

[17] B. Sun and K. Saenko, "Deep CORAL: Correlation alignment for deep domain adaptation," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 443–450.

[18] Y. Zhu *et al.*, "Deep subdomain adaptation network for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 4, pp. 1713–1722, Apr. 2020.

[19] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 3630–3638.

[20] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[21] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 4077–4087.

[22] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.

[23] V. Garcia and J. Bruna, "Few-shot learning with graph neural networks," 2017, *arXiv:1711.04043*.

[24] A. A. Rusu *et al.*, "Meta-learning with latent embedding optimization," 2018, *arXiv:1807.05960*.

[25] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele, "Meta-transfer learning for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 403–412.

[26] R. Hou, H. Chang, B. Ma, S. Shan, and X. Chen, "Cross attention network for few-shot classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4005–4016.

[27] V. Verma *et al.*, "Manifold mixup: Better representations by interpolating hidden states," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 6438–6447.

[28] D. Rezende *et al.*, "One-shot generalization in deep generative models," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 1521–1529.

[29] H.-Y. Tseng *et al.*, "Few-shot viewpoint estimation," 2019, *arXiv:1905.04957*.

[30] R. Vuorio, S.-H. Sun, H. Hu, and J. J. Lim, "Multimodal model-agnostic meta-learning via task-aware modulation," 2019, *arXiv:1910.13616*.

[31] Y. Lifchitz, Y. Avrithis, S. Picard, and A. Bursuc, "Dense classification and implanting for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9258–9267.

[32] B. N. Oreshkin, P. Rodriguez, and A. Lacoste, "TADAM: Task dependent adaptive metric for improved few-shot learning," 2018, *arXiv:1805.10123*.

[33] B. Liu, X. Yu, A. Yu, P. Zhang, G. Wan, and R. Wang, "Deep few-shot learning for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2290–2304, Apr. 2018.

[34] K. Gao, B. Liu, X. Yu, J. Qin, P. Zhang, and X. Tan, "Deep relation network for hyperspectral image few-shot classification," *Remote Sens.*, vol. 12, no. 6, p. 923, Mar. 2020.

[35] J. Kim, T. Kim, S. Kim, and C. D. Yoo, "Edge-labeling graph neural network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11–20.

[36] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Dec. 2009.

[37] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*. Sacramento, CA, USA: Curran Associates, 2017, pp. 1025–1035.

[38] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi, "Graph neural networks with convolutional ARMA filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 99, pp. 1–14, Jul. 2021.

[39] L. Chen, Z. Gan, Y. Cheng, L. Li, L. Carin, and J. J. Liu, "Graph optimal transport for cross-domain alignment," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, Jul. 2020, pp. 1542–1553.

[40] Z. Li, M. Liu, Y. Chen, Y. Xu, W. Li, and Q. Du, "Deep cross-domain few-shot learning for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–18, 2021.

[41] L. Ying, H. Zhang, and S. Qiang, "Spectral-spatial classification of hyperspectral imagery with 3D convolutional neural network," *Remote Sens.*, vol. 9, no. 1, p. 67, Jan. 2017.

[42] C. Yu, J. Wang, Y. Chen, and M. Huang, "Transfer learning with dynamic adversarial adaptation network," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2019, pp. 778–786.

[43] L. Yang, L. Li, Z. Zhang, X. Zhou, E. Zhou, and Y. Liu, "DPGN: Distribution propagation graph network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13390–13399.

[44] K. Sun *et al.*, "A robust and efficient band selection method using graph representation for hyperspectral imagery," *Int. J. Remote Sens.*, vol. 37, no. 20, pp. 4874–4889, 2016.