Cross-Domain Self-Taught Network for Few-Shot Hyperspectral Image Classification

Mingyang Zhang, *Member, IEEE*, Hao Liu, Maoguo Gong, *Senior Member, IEEE*, Hao Li, *Member, IEEE*, Yue Wu, *Member, IEEE*, and Xiangming Jiang, *Member, IEEE*

***Abstract*— In recent years, deep learning models, which possess powerful feature extraction abilities, have achieved remarkable success in the classification of hyperspectral images (HSIs). Nevertheless, a common challenge faced by most deep learning models, including few-shot learning (FSL) models, is the scarcity of valid labeled samples. To address this issue, we propose a cross-domain self-taught network (CDSTN) for few-shot HSI classification. The proposed CDSTN merges domain adaptation (DA) and semisupervised self-taught strategy to implement the FSL, which utilizes adequate labeled and unlabeled samples from source as well as target domains, respectively. For the feature information extraction of HSI, we propose a deep spatial–spectral feature embedded extractor composed of four residual blocks and a channel attention module (CAM). Additionally, a set of domain classifiers are introduced behind each residual block for the purpose of domain alignment by extracting more domain information at different depths of the network. Finally, plenty of unlabeled samples are assigned with pseudo labels through the trained network, and a pseudo label refinement (PLR) module is designed to select the most confident pseudo label sample for each class to further enrich the labeled database of target domain. Experiments conducted on four widely used benchmark HSI datasets demonstrate that CDSTN can obtain superior and stable performance with limited labeled samples compared with some state of the arts.**

***Index Terms*— Domain adaptation (DA), few-shot learning (FSL), hyperspectral image (HSI) classification, self-taught learning.**

I. Introduction

HYPERSPECTRAL imagery (HSI) is a 3-D data cube composed of a wide range of electromagnetic wave, which contains abundant geospatial and spectral information. It is one of the most significant applications using HSI to recognize and classify ground objects in remote sensing field. HSI classification has been invested in precision land cover [1], resource investigation [2], [3], ecological environmental monitoring [4], and so on.

There are two aspects in terms of traditional HSI classification, shallow feature engineering and simple classifier, both of which have been widely studied and generated lots of research. Shallow feature engineering consists of many feature extraction algorithms, such as principal component analysis (PCA) [5], extended morphological profile (EMP) [6], and simple linear iterative cluster (SLIC) [7]. Traditional HSI classifiers mainly include support vector machine (SVM) [8], random forest (RF) [2], k-nearest neighbor (KNN) [9], and so on. However, these methods only focus on a single pixel without considering the spatial and spectral correlation among pixels. Besides, the lack of high-level feature also leads to the poor performance.

With the development of deep learning and the improvement of computing power, HSI classification combined with deep learning method has achieved greater success. Compared with the previous research, these methods make full use of high-level features extracted from raw data through artificial neural network. Chen et al. [10], [11] first proposed stacked autoencoder (SAE) and deep belief network (DBN) combined with HSI classification. Then, Mou et al. [12] proposed a recurrent neural network (RNN) for image classification. Ansari et al. [13] proposed a convolutional kernel classifier (CKC) and Boulila et al. [14] proposed a distributed convolutional neural networks (DCNNs) approach for the efficient classification of large remote sensing images. Besides, there also are fully connected deep neural network (DNN), convolution neural network (CNN) [15], 2-D CNN [16], 3-D CNN [17], a hybrid spectral convolutional neural network (HybridSN) [18] comprising 3-D CNN and 2-D CNN, and generative adversarial network (GAN) [19] applied to HSI classification task. Moreover, some other methods to optimize the classification effect of basic neural network structure have also been proposed, such as residual structure [20], dense connection structure [21], and attention mechanism [22]. These methods have been introduced into the basic structure and achieved exciting performance. However, these deep-learning-based methods are easily fall into underfitting causing by low model complexity or overfitting due to insufficient training samples. In particular, obtaining high-quality manual labels for HSI is time-consuming and laborious. How to use a small number of labeled samples to better the classification is still an open challenge.

Manuscript received 6 January 2023; revised 2 April 2023; accepted 8 April 2023. Date of publication 12 April 2023; date of current version 26 April 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62036006, in part by the China Postdoctoral Science Foundation under Grant 2021T140528, and in part by the Fundamental Research Funds for the Central Universities under Grant ZYTS23147 and Grant ZYTS23136. *(Corresponding author: Maoguo Gong.)*

Mingyang Zhang, Hao Liu, Maoguo Gong, Hao Li, and Xiangming Jiang are with the School of Electronic Engineering and the Key Laboratory of Collaborative Intelligent Systems of Ministry of Education, Xidian University, Xi'an 710071, China (e-mail: omegazhangmy@gmail.com; hliu_7@stu.xidian.edu.cn; gong@ieee.org; omegalihao@gmail.com; omegajiangxm@gmail.com).

Yue Wu is with the School of Computer Science and Technology, Xidian University, Xi'an 710071, China (e-mail: ywu@xidian.edu.cn).

Digital Object Identifier 10.1109/TGRS.2023.3266565

At present, three effective solutions, including transfer learning, few-shot learning (FSL), and semisupervised learning, have been developed to settle the problem of lacking valuable samples. Besides, data augmentation by implementing rotation and flipping [23] has also contributed to this issue, which improves sample diversity and enhances the robustness of the model. However, for data enhancement using neural networks, higher quality and valuable samples need to be generated by a larger scale neural network, which is more expensive to deploy for a network with complex tasks [24].

In real scenes, only a minority of HSI datasets used for research have a large number of labels, while most HSI datasets have few available labels. Therefore, the domain adaptation (DA) method in transfer learning can be used to learn prior knowledge from source domains with a large number of labeled and then apply it to tasks in the target domain. Long et al. [25] proposed deep adaptation network (DAN) applied the multikernel maximum mean discrepancy (MK-MMD) metric to three adaptive layers. Zhu et al. [26] proposed the deep subdomain adaption network (DSAN) and used local MMD (LMMD) to align the relevant subdomains, respectively. Besides, Deng et al. [27] proposed multikernel learning with active learning (MKL-AL) and Sun et al. [28] proposed domain transfer multiple-kernel learning (DTMKL), which combined the method of DA with HSI classification. In the realm of remote sensing, Zhang et al. [29] proposed the discriminative cooperative alignment (DCA) method, which addresses both geometric and statistical shifts by aligning subspaces and distributions. Zhao et al. [30] introduced a method that leverages fractional fusion and spatial–spectral DA to mitigate disparities between scenes. Zhang et al. [31] also proposed the topological structure and semantic information transfer netw-k (TSTnet), which effectively tackles spectral shifts by aligning distributions and graphs. However, most DA frameworks require the same classes of source and target domains, and the performance will crash when new categories appear in the target domain.

As for the few-shot classification, one of the most important training methods of the model is meta-learning [32], [33], which requires that the datasets are decomposed into different meta-tasks for meta-knowledge, and therefore the common categories of source domain and target domain are not necessary. Existing FSL approaches can be roughly categorized into three groups: 1) data-based methods; 2) model-based methods; and 3) algorithm-based methods. Data-based methods pay attention to the strategy of data augment by transforming each sample into several sample with some variation. The transformation is learned from the training set [34], [35], other similar datasets [36], weakly labeled [37] or unlabeled datasets [38]. Model-based methods design a model based on prior knowledge to constrain the complexity of the hypothesis space and reduce its sample complexity. The strategies include embedding learning [32], [39], generative modeling [40], and multitask learning [41], [42]. Algorithm-based methods attempt to search in the hypothesis space for the best hypothesis, such as fine-tuning [43], aggregating a set of parameters [35], and refining the meta-learned parameters [44], [45]. The reduced dependence on training samples makes FSL a popular approach in the fields of image classification [46], [47], object detection [48], [49], and image segmentation [50], [51]. In the context of HSI classification, numerous studies have been conducted to address the FSL problem, which are mainly the model-based methods such as prototypical network [52], [53], relation network [54], and siamese network [55], [56]. In addition, some other related studies focused on FSL for HSI DA [57], [58], [59], [60] also have been proposed and made great progress. These methods require to obtain minimizing experience risk, which means that the data distribution of the entire samples can be learned from few labeled samples. However, it is troublesome to obtain reliable minimizing experience risk due to the insufficient samples and complex models with few training samples can easily fall into overfitting.

Besides, another class of effective methods to solve the problem of few labeled samples is semisupervised learning method, which is generally categorized into three types: self-training models [61], cotraining models [62], [63], and generative models [64]. Self-training models train a classifier first, and then classify the unlabeled data until all the data are labeled, which is one of the most straightforward ways in utilizing unlabeled data information. Cotraining models use several classifiers trained by the labeled data to assign pseudo to unlabeled samples. The new data expand the training set and the procedure repeats until some stopping criterion reaches. Generative models estimate conditional density to predict the labels of unlabeled samples. All of these methods have achieved certain results, but the inferred pseudo labels may not be trustworthy. It is essential to investigate the confidence of each unlabeled samples.

To address the limitations discussed above, in this article, a new cross-domain self-taught network (CDSTN) for few-shot HSI classification is designed. First, four 3-D residual blocks and a channel attention module (CAM) construct an embedded feature extraction network, which is trained using meta-learning to map the data to a low-dimensional spatial–spectral measurement space. Second, four domain classifiers are, respectively, connected behind each residual block to perform domain classification on the processed and flattened features of each block, which can combine common and unique domain information to achieve faster and more accurate domain alignment. Furthermore, when the optimal value of FSL classification is reached, for each category, the Top-k samples and one sample with the highest probability are screened out through the FSL module and the linear regression model, respectively, which can minimize the damage of false pseudo labels to the FSL model. Finally, add selected samples to the labeled sample set and the process repeats until the iteration is complete. Specifically, the contributions of this article are as follows.

1) We proposed a self-taught learning method including assigning pseudo labels and pseudo label refinement (PLR) strategies, which could select the most confident instances and minimize noise from pseudo labels.
2) A multilevel adversarial DA strategy is proposed to effectively leverages the informative and uninformative

representatives from different levels of the embedded feature network.

3) A novel embedded feature extractor is designed based on the architecture of deep residual CNN, which consists of a convolutional CAM. The embedded feature extractor can extract deep spatial–spectral joint features with concise convolution structure and CDSTN is an end-to-end training method.

The remainder of this article is organized as follows. In Section II, we give some background knowledge and the motivation of CDSTN. In Section III, we describe the proposed method in detail. Section IV validates the proposed method on four real datasets and analyzes the hyperparameters in the proposed method. Finally, Section V gives the concluding remarks of this article.

II. Preliminaries and Motivation

A. Cross-Domain Few-Shot Learning

In terms of FSL, one of the most typical training methods is meta-learning. The model is trained every time by constructing a meta-learning task T containing support set S and query set Q, where S is a labeled sample set including C classes with K samples in each class. This FSL task T is named as C-way K-shot due to S, while Q is an unlabeled sample set for evaluating the performance of the model. We define a domain as a joint distribution P over input space X and label space Y. The marginal distribution of X is denoted as P_X.

Compared with the traditional pretraining and fine-tuning, when a major domain shift occurs between a basic category and a new category, some FSL algorithms based on meta-learning perform poorly. For the sake of settling this issue, a cross-domain FSL method has been proposed, which requires that base and novel classes are both drawn from different domains, and the class label sets are disjoint [65]. Recently, several researches have also appeared with regard to cross-domain FSL [66], [67].

To the specific, the HSI datasets are divided into source datasets with numerous labeled samples and target datasets with only a few labeled samples. We have a source domain D_s and a target domain D_t with joint distribution P_s and P_t, respectively, $P_s \neq P_t$, and Y_s is disjoint from Y_t. The number of categories and labeled samples in the source domain C_s is larger than that in the target domain C_t [68], so that knowledge can be migrated from the source domain to the target domain. Thereby, the cross-domain FSL tasks are constituted: source domain FSL task T_s and target domain FSL task T_t. T_s and T_t are executed alternately to enable the embedded feature extractor to learn to map D_s and D_t to a common feature space. In the field of HSI classification, Li et al. [59] proposed a deep cross-domain FSL (DCFSL) method. Zhang et al. [69] proposed a graph information aggregation cross-domain FSL (Gia-CFSL) framework. Xi et al. [70] proposed an FSL framework with a class-covariance metric (CMFSL). These works have made significant progress in HSI classification.

Due to the scarcity of labeled samples in the target domain, our goal is to align the source domain D_s and target domain D_t as closely as possible and leverage the embedded feature information extracted from each sample. Concerning a deep network with several blocks, some distinctive information at high blocks from target domain could get ignored. It is essential to explore the low-level features at low blocks such as corners and edges. Also, probability information from classifier can be used as an invariant feature and transferred between domains. Thus, we propose to extract the domain informative and uninformative features from different depths of the network and execute an adaptive domain alignment strategy by adding a probability information contained in the unlabeled samples to optimize training process.

B. Semisupervised Few-Shot Learning

Unlike the mentioned FSL methods, semisupervised FSL (SSFSL) aims to improve the classification performance by resorting to a certain amount of unlabeled data. Ren et al. [71] proposed three semisupervised variants of ProtoNets [32], mainly using soft k-Means method to tune clustering centers with unlabeled data. TransMatch [72] employs a transfer learning framework for SSFSL by learning a cosine similarity-based recognition model without episodic training. Jia et al. [73] proposed a semisupervised siamese network (3DAES) which use the unlabeled samples to train the encoder and decoder. Besides, some exploration [74], [75] of SSFSL attempt to use label propagation to construct the relationship between unlabeled and labeled data. Specially, it is a direct and valid way for SSFSL by assigning pseudo labels of unlabeled samples and selecting the data with high degree of confidence for iterative training [76].

However, there are few labeled data and a much larger amount of unlabeled data for training classifiers. With respect to FSL, a wrongly labeled instance may make the performance of the model collapse sharply, which drops faster than supervised learning. It is essential that the pseudo label samples must be carefully selected. Besides, due to the fact that a prototype composed of small samples cannot represent the average feature vector of all samples in the category, additional samples are incorporated into the training dataset to enhance the realism of the prototype as the category center. Thus, we proposed a two-stage strategy to select the samples. For each category, first, choose Top-k samples with the highest classification probability of FSL classifier from all unlabeled samples, and then select one sample with the highest confidence for these Top-k samples to add to the labeled sample set. After these operations, the accuracy rate of the screened samples with pseudo labels is much higher than the accuracy rate of the directly assigned pseudo labels.

III. Methodology

The goal of this work is to develop the learning ability of FSL through using pseudo labeled samples. The overall framework of the proposed CDSTN is shown in Fig. 1. It can be observed that the proposed architecture mainly consists of three portions: embedded feature extractor, DA module, and PLR module. In this section, we elaborate on the three parts in detail.

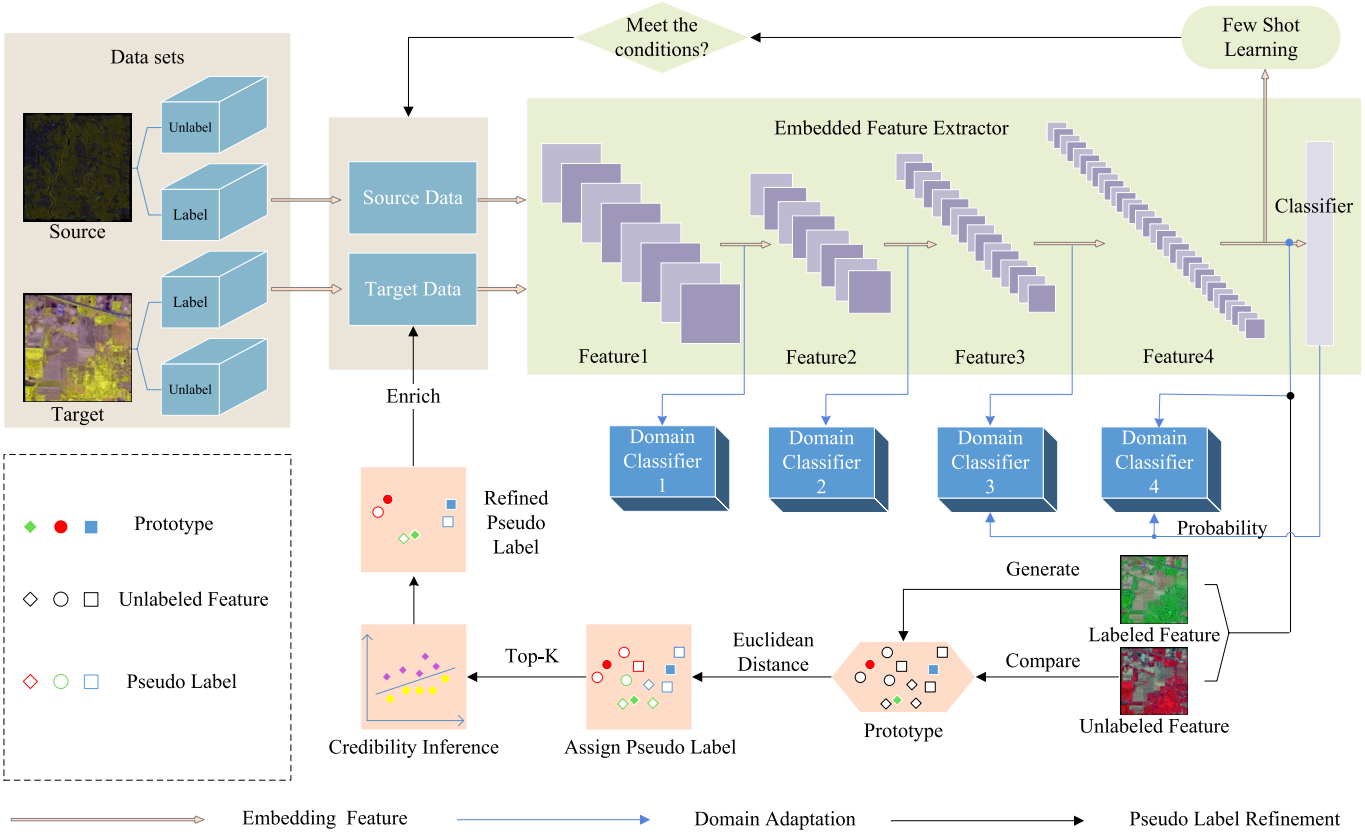


Fig. 1. Flowchart of the proposed CDSTN approach. CDSTN is composed of an embedded feature extractor, four domain classifiers, and the PLR module. CDSTN is an end-to-end training method.

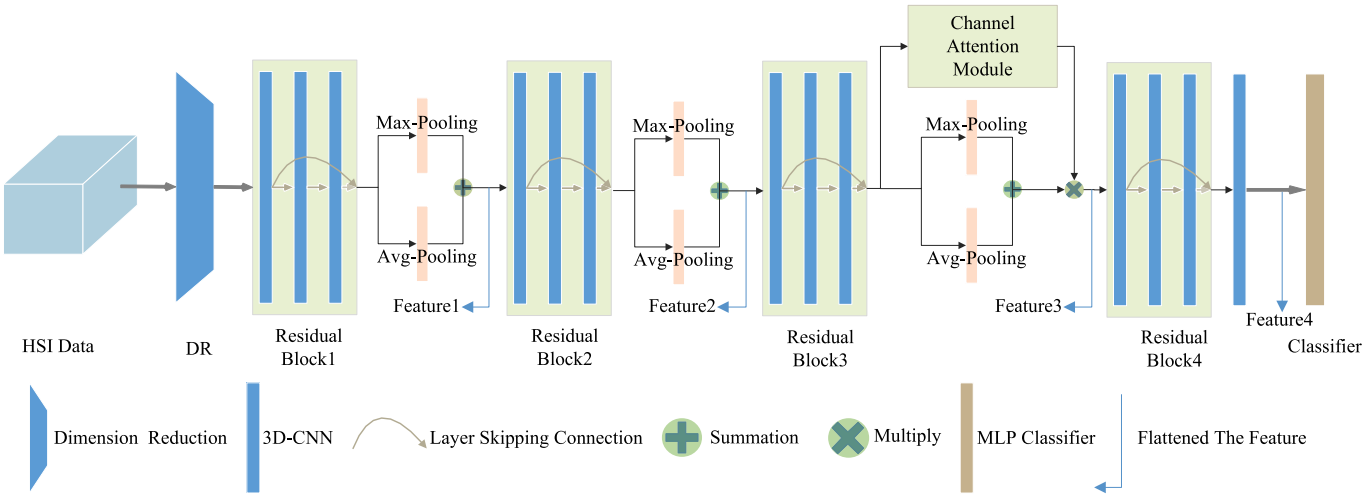


Fig. 2. Illustration framework of the proposed embedded feature extractor.

A. Embedded Feature Extractor

The spectral dimensions, spatial resolution, and other parameters of HSIs captured by different sensors are generally different, so it is necessary to first reduce the spectral dimensions of HSI in the source domain and target domain. The detailed structure of embedded feature extractor is shown in Fig. 2. In this part, a dimension reduction (DR) with 2-D convolution is used to reduce the spectral dimension to a lower dimension, which is convenient to input the data from source domain and target domain into the same embedded feature extractor. For instance, Chikusei dataset with 128 bands and Salinas dataset with 204 bands both are reduced to 100 dimensions after DR module.

Given the spectral redundancy of HSIs, we propose to establish a deep neural network, and combine the DA strategy mentioned below to carry out the network full-level information mining. The main part of the extractor is composed of four residual blocks, each of which is composed of three 3-D

convolutions. Based on residual structure, we introduce layer skipping connection for each block, which means the output of the first 3-D convolution is added to the output of the third 3-D convolution and then enter the activation function together. Besides, we propose to build a dual-channel pooling layer including max-pooling as well as average-pooling between each residual block, which could not only retain the extensive background information of HSIs but also extract edges and texture information of specific categories. A feature is derived from a residual block and a pooling layer, consequently the embedded feature network generate four features with different levels. These features from different blocks are flattened to input the domain classifiers for DA module, which is detailed in Section III-B.

Even with the feature DR module in place, the spectral dimension of HSI remains substantial, resulting in redundant information. Considering the convention modeling ideas for FSL, the complexity of the model cannot be increased excessively. Therefore, a CAM, which is mainly composed of pooling and convolution, is applied in this structure for the sake of making better use of HSI information. In particular, the feature map first inputs global average-pooling and global max-pooling, respectively, and the two outputs are concatenate for performing a 2×1 convolution. Next, the output is fed to a multilayer perception (MLP) layer and the corresponding weight values of different channels are obtained through activation function. The specific formula can be expressed as follows:

$$\mathrm{CAM}(\boldsymbol{f}) = \mathrm{ReLU}\big(\mathrm{MLP}\big(\mathrm{Conv2d}\big(\boldsymbol{f}_{\max}, \boldsymbol{f}_{\mathrm{avg}}\big)\big)\big) \tag{1}$$

where $\boldsymbol{f}_{\max}$ and $\boldsymbol{f}_{\mathrm{avg}}$ are the outputs of global max-pooling and global average-pooling. $\boldsymbol{f}$ is the input feature. Rectified linear unit (ReLU), Conv2d, and CAM represent ReLU activation function, and 2-D convolution with a filter size of 2×1 and the CAM.

B. Domain Adaptation

There exists a huge domain gap between HSI datasets from different sensors. In light of reducing the shift between the source and target data domain, a collaborative and adversarial strategy is designed in this work. So as to balance the huge amount of redundant information in HSI and the loss of key domain information in deep network learning, we also need to seek domain information at different depths of the network. Inspired by the collaborative and adversarial network (CAN) [77], we introduce a set of domain discriminators into multiple blocks.

In the field of DA, one of the most commonly used strategy is based on adversarial learning. The common space characteristics of source domain and target domain are learned through domain adversarial strategy to confuse domain discriminator. After the samples are embedded as feature vectors by building a domain discriminator, which is unable to distinguish whether the samples come from the source domain or the target domain, domain alignment could be achieved. Here, the gradient reversal layer [78] is used to update parameters of the domain discriminator. Considering the depth of the feature embedded extraction, we add a domain discriminator after the fourth block, and use domain adversarial strategy to realize the learning of common domain features. Given sample $\boldsymbol{x_i}$, the feature vector after processing by the embedded feature extractor F with parameter $\boldsymbol{\theta}$ is expressed as $\boldsymbol{f_i} = F(\boldsymbol{x_i}, \boldsymbol{\theta})$. The purpose of domain discriminator D is to judge whether $\boldsymbol{f}$ belongs to source domain with label 0 or target domain with label 1. And to quickly confuse the domain discriminator, we add the posterior probability information of the classifier to the feature vector, and its output is shown as follows:

$$\max_{\boldsymbol{\theta}} \min_{\boldsymbol{w}} \frac{1}{N} \sum_{i=1}^{N} L_D(D((F(\boldsymbol{x_i}, \boldsymbol{\theta}), \boldsymbol{g_i}); \boldsymbol{w}), d_i) \tag{2}$$

where L_D is the classification loss and N is the total number of the samples. $\boldsymbol{w}$ and $\boldsymbol{\theta}$ are the parameters of the domain discriminator and feature embedded extractor, respectively. $\boldsymbol{g_i}$ and d_i are the posterior probability and the label of domain corresponding to the sample $\boldsymbol{x_i}$, respectively. It is worth noting that since the self-taught network often introduces new data, it needs to achieve domain alignment more quickly. In addition to using the information provided by the shallow network, $\boldsymbol{g_i}$ could accelerate the alignment of domains, which often happens in the high level of a network. Consequently, we combine $\boldsymbol{f}$ and $\boldsymbol{g_i}$ after the two deepest blocks into a new feature vector to feed the discriminators.

In addition, the low layer of the neural network can usually extract representative features of different domains, such as edges and corners. which can help the domain discriminator better distinguish between source domain and target domain. Therefore, several domain discriminators are also established in the low blocks. Then the objective for domain discriminators can be written as follows:

$$\min_{\boldsymbol{\theta}, \boldsymbol{w}} \frac{1}{N} \sum_{i=1}^{N} L_D(D((F(\boldsymbol{x}_i, \boldsymbol{\theta}), \boldsymbol{g_i}); \boldsymbol{w}), d_i) \tag{3}$$

where the meaning of these symbol are the same as (2), but the optimization objectives of the two formulas are different.

With respect to our work, there exist four blocks in the embedded feature extractor, which means four domain discriminators are after each block. However, different levels usually have different emphasis for feature extraction. We set different weights $\boldsymbol{\lambda_k}(k = 1, \ldots, m)$ for these discriminators with different parameters $\boldsymbol{w}_k(k = 1, \ldots, m)$. Considering that the feature received by the discriminator following the last block is input into the FSL module as the final feature vector, then this feature should already achieve domain alignment. Therefore, the fourth domain discriminator must perform DA based on the adversarial strategy, while the other three domain discriminators cannot determine whether it is the adversarial strategy of (2) or the optimization strategy of (3) since they are in different depths of the network. Finally, weight $\boldsymbol{\lambda_m}$ and initial weight sum $\boldsymbol{\lambda}_0$ are set for the fourth domain discriminator and the rest discriminators, respectively, and each weight is added to the loss function and automatically adjusted with the network iteration process. The specific parameter setting experiment is in Section IV-E. Then the overall optimization

objectives of the network are as follows:

$$\min_{\Theta,\boldsymbol{\lambda}} L_{\mathrm{DA}} = \sum_{k=1}^{m-1} \boldsymbol{\lambda}_k \min_{\boldsymbol{w}_k} L_D(\boldsymbol{\theta}_k, \boldsymbol{w}_k) + \boldsymbol{\lambda}_m \min_{\boldsymbol{w}_m} L_D(\boldsymbol{\theta}_m, \boldsymbol{w}_m)$$
$$\text{s.t.} \sum_{k=1}^{m-1} \boldsymbol{\lambda}_k = \boldsymbol{\lambda}_0, \quad |\boldsymbol{\lambda}_k| \geq \boldsymbol{\lambda}_0 \tag{4}$$

where $\Theta = (\boldsymbol{\theta_1}, \ldots, \boldsymbol{\theta_m})$ and $\boldsymbol{\theta}_k (k = 1, \ldots, m)$ is the parameter of the kth block. $\boldsymbol{\lambda} = (\boldsymbol{\lambda_1}, \ldots, \boldsymbol{\lambda_m})$ and L_D is the loss of the domain discriminator mentioned in (2).

In the optimization process, the parameters $\boldsymbol{\lambda}_k$ change automatically. When $\boldsymbol{\lambda}_k \leq 0$, the object is as described in (2) and the features trained from different domains are aligned into the same space. And when $\boldsymbol{\lambda}_k > 0$, the object is as similar as (3) to extract more domain specific information.

C. Few-Shot Learning Module

After four blocks and attention modules, the samples have become feature vectors with very low dimensions. Input the features vector into the FSL module, and use the labeled samples to train the embedded feature extractor with parameter $\boldsymbol{\theta}$. We choose the method of meta-learning to train the model, that is carrying out FSL tasks in the source domain and target domain successively. In each iteration, the prototype of each category of samples is generated by the support set. The classification results are obtained according to the Euclidean distance between the samples in query set and the prototype, while the network parameters are updated iteratively. The probability result of an instance is shown in the following formula:

$$\boldsymbol{P}\left(\hat{y}_i | \boldsymbol{x_i} \in Q_s\right) = \mathrm{Softmax}\left(-\mathrm{ED}\left(f(\boldsymbol{x_i}), \boldsymbol{c^k}\right)\right) \tag{5}$$

where $\boldsymbol{x_i}$ is a sample from query set and $\hat{y}_i$ means the predicted sample label. Softmax$(\cdot)$ is the softmax function, ED$(\cdot)$ is the Euclidean distance, $f(\cdot)$ is the feature embedded extractor, and $\boldsymbol{c^k}$ is the prototype of the kth category.

Based on the probability of query set from source domain, we could calculate the loss of source domain by cross entropy loss, as shown in the following formula:

$$L_{\mathrm{CE}}^s = -\frac{1}{N_s} \sum_{i=1}^{N_s} y_i^s \log \boldsymbol{P}\left(\hat{y}_i^s | \boldsymbol{x}_i^s\right) \tag{6}$$

where N_s is the number of the query set sample from source domain, y_i^s and $\hat{y}_i^s$ are the corresponding true label and predicted label of the sample $\boldsymbol{x}_i^s$, respectively.

For the same reason, we could figure out the loss of target domain, which is shown in the following formula:

$$L_{\mathrm{CE}}^t = -\frac{1}{N_t} \sum_{i=1}^{N_t} y_i^t \log \boldsymbol{P}\left(\hat{y}_i^t | \boldsymbol{x}_i^t\right) \tag{7}$$

where the symbolic meaning is similar to (6) except that the samples are from target domain. In addition, we could conclude the loss of source and target domain iteration

$$\begin{aligned} L_{\mathrm{total}}^s &= L_{\mathrm{CE}}^s + L_{\mathrm{DA}} \\ L_{\mathrm{total}}^t &= L_{\mathrm{CE}}^t + L_{\mathrm{DA}}. \end{aligned} \tag{8}$$

Finally, the final loss for our CDSTN can be written as

$$L_{\mathrm{total}} = L_{\mathrm{total}}^s + L_{\mathrm{total}}^t. \tag{9}$$

D. Pseudo Label Refinement

As shown in Fig. 3, the process of assigning labels to unlabeled samples is divided into two stages. In the first stage, convert the unlabeled samples into feature vectors according to the trained feature embedded extractors, use FSL module to classify the samples, and select the Top-k samples with the highest classification probability to assign pseudo labels. In the second stage, we select the linear regression model as another standard to evaluate the confidence of these pseudo labeled samples, then select the highest one sample for each class as the selected labeled data. Finally, add them to the labeled dataset and continue the training.

1) Classification Probability Confidence: When the feature embedded extractor is trained, the data from both source and target domains are mapped into the spatial–spectral embedded space. FSL module could be used to classify the unlabeled data and tag the unlabeled samples.

Given labeled samples, feature vectors are generated by the trained feature embedded extractor. Take the average value for each type of feature vectors to get the prototype

$$\boldsymbol{c}_k = \frac{1}{N_t^k} \sum_{i=1}^{N_t^k} f\left(\boldsymbol{x}_i^k\right) \tag{10}$$

where N_t^k is the labeled samples of kth category from target domain, $f(\cdot)$ is feature embedded extractor, and $\boldsymbol{c}_k$ is the calculated prototype of the kth class. Significantly, the addition of qualified pseudo label samples to the training set can update the prototypes, N_t^k could increase with each pseudo label filtering stage. However, the generation rules of the prototypes remain unchanged.

Then, we input unlabeled samples into the trained feature embedded extractor and FSL module to generate feature vectors. As (5) expressed, by measuring the distance between a vector and each prototype, we could estimate the probability that the corresponding sample belongs to each category.

According to the probability of FSL module classification, we select Top-k samples with the highest probability from each category and Top-k samples from unlabeled samples are picked. We add kurtosis and the lowest classification probability as thresholds in the first-stage screening process, to ensure the probability distribution closer to a normal distribution than a uniform distribution. This helps avoid pseudo labels with low confidence and reduces interference of spectral-category problems on the prototypes, such as cases where the same object exhibits different spectral characteristics or where different objects display the same spectral features. The definition of kurtosis is as follows:

$$\mathrm{Kurtosis} = E\left[\left(\frac{\boldsymbol{g} - \boldsymbol{\mu}}{\boldsymbol{\sigma}}\right)^4\right] \tag{11}$$

where $\boldsymbol{g}$ is the probability distribution of all categories for a sample, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the mean value and variance of these probability. $E(\cdot)$ is the mean operation.

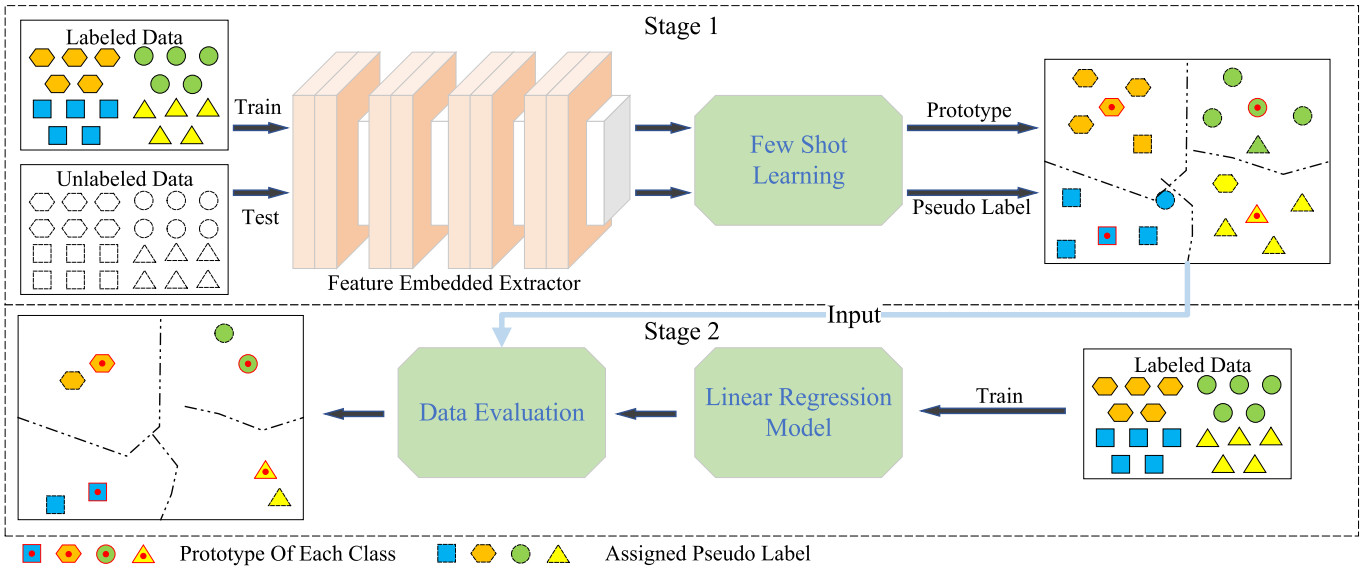


Fig. 3. Flowchart of PLR module.

2) Linear Regression Model Confidence: The method of directly classifying and marking pseudo labels through FSL module may bring some errors. Since the model is very sensitive to data as for FSL. Once error labels appear, continuing to train the model with these samples will not only bring noise to the feature embedded extractor, but also do great harm to the prototype and classification result of FSL. Inspired by the instance credibility inference (ICI) [79], we use an additional model to select some samples with the highest confidence from the first stage, based on the idea of ensemble learning, which addresses the sensitivity of FSL to noise labels. Since there are a large number of unlabeled samples, we optimize the multi-model voting to a multistage screening process to improve the speed of model voting selection. As the training parameters of the network are fixed at this stage, screening for all unlabeled samples will not increase complexity significantly.

A linear regression model is introduced to map the feature vector extracted from the feature embedded extractor to the label space, that is $f(\boldsymbol{x}) \rightarrow y$. Let $\boldsymbol{x_i}$ be a sample, the model can be expressed as follows:

$$y_i = f(\boldsymbol{x_i})^T \boldsymbol{\beta} + \boldsymbol{\gamma}_i + \boldsymbol{\epsilon}_i \tag{12}$$

where $\boldsymbol{\beta}$ is coefficient matrix, $\boldsymbol{\epsilon}_i$ is the Gaussian noise, and $\boldsymbol{\gamma}_i$ is a correction which measure the confidence of samples along the regularization path. The larger $\boldsymbol{\gamma}_i$, the smaller its probability belonging to y. Therefore, we could estimate the probability by the sparsity of correction matrix in terms of a sample.

Let $\boldsymbol{X} \in \mathbb{R}^{N\times d}$ and $\boldsymbol{Y} \in \mathbb{R}^{C\times N}$ are feature vectors and labels of all samples. $\boldsymbol{\gamma} \in \mathbb{R}^{C\times N}$ denotes the correction matrix corresponding to all correction $\boldsymbol{\gamma}_i$. Then, (12) can be expressed as follows:

$$(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}}) = \underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\arg\min} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_F^2 + \alpha \boldsymbol{R}(\gamma) \tag{13}$$

where $\|\cdot\|_F^2$ is the Frobenius norm and $\boldsymbol{\alpha}$ is the coefficient of penalty. Solve and simplify (13) to get

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{\dagger}\boldsymbol{X}^T(\boldsymbol{Y} - \boldsymbol{\gamma}) \tag{14}$$

where $(\cdot)^{\dagger}$ denotes the Moore–Penrose pseudo-inverse. Let us suppose that $\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{\dagger}$, $\tilde{\boldsymbol{X}} = (\boldsymbol{I} - \boldsymbol{H})$, and $\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{X}}\boldsymbol{Y}$. C and N and are number of classes and samples. d is the dimension of feature vector. Then the last object can be simplified as

$$\underset{\boldsymbol{\gamma}\in\mathbb{R}^{C\times N}}{\arg\min} \left\|\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma}\right\|_F^2 + \alpha \boldsymbol{R}(\gamma). \tag{15}$$

The above multiple-response regression problem is solved by blockwise descent algorithm implemented in Glmnet [80]. At least the theoretical maximum $\boldsymbol{\alpha}_{\max} = \max_i \|\tilde{\boldsymbol{X}}_{\cdot i}^T\tilde{\boldsymbol{Y}}\}/n$ makes all solutions zero. Then a series of $\boldsymbol{\alpha}$ values from 0 to $\boldsymbol{\alpha}_{\max}$ are used to solve and the sparsity gradually increase. In the case of different $\boldsymbol{\alpha}$, all samples correspond to $\boldsymbol{\gamma}$ with different sparsity, considering $\boldsymbol{\gamma}$ can be treated as a function of $\boldsymbol{\alpha}$. The earlier $\boldsymbol{\gamma}$ disappears, the higher the confidence of the solution.

Specifically, use labeled samples to train the above linear regression model, then input the samples and the corresponding pseudo labels selected in the first stage into the linear regression model for data evaluation, then select the sample with the highest confidence of each type, and add them to the labeled dataset as the correct samples for final screening to continue training.

In fact, the linear regression model from the second stage can have multiple variations, and the selection stage can also be expanded to three or four stages, etc. However, increasing the number of models and screening stages will further burden the model screening process at each stage. Moreover, in the HSI data, the category label is assigned to the central pixel and the data is partitioned into patch-sized blocks. Excessive pseudo-label screening may filter out other pixels except for

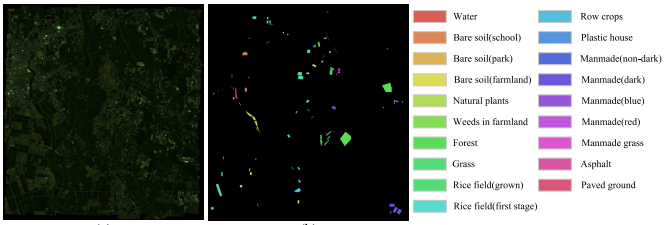


Fig. 4. Chikusei. (a) Pseudo color image (bands 12, 41, and 55). (b) Ground-truth map.

the patch block of the training samples and make the pseudo label samples too dependent on the training set. This can cause the prototypes to update its direction toward the training samples rather than the overall data center for the class. Taking into account the aforementioned issues, we choose the two-stage pseudo-label screening method, which has demonstrated the best performance.

The implementation procedure of CDSTN is summarized in Algorithm 1.

Algorithm 1 Implementation Procedure of CDSTN

Input : HSI data patch D^S and $D^T = D^T_{label} \cup D^T_{unlabel}$, number of pseudo label samples selected in stage one Top-k, total epochs N_{epo}

Output : Classification Result

for epoch in range $N_{epo}/2$ **do**
  Initialize : FSL task T_s from D^S, FSL task T_t from D^T_{label}
  Perform source domain tasks T_s;
  Calculate the total loss L^s_{total};
  Upgrade the parameter of network;
  Perform target domain tasks T_t;
  Calculate the total loss L^t_{total};
  Upgrade the parameter of network;
  if The accuracy is the best when testing **then**
    Perform pseudo label refinement stage 1:select Top-k samples per class from $D^T_{unlabel}$;
    Perform pseudo label refinement stage 2: select one samples per class from stage 1;
    upgrade D^T_{label};
return result.

IV. Experimental Analysis

A. Datasets Description and Evaluation Criteria

CDSTN is a cross-domain learning model, which need to learn prior knowledge from the source domain and then apply it to the target domain. Therefore, we select four mainstream hyperspectral remote sensing datasets as target domain, including Indian Pines (IP), Salinas, Pavia University (PU), and the Pavia Center (PC). And Chikusei is chosen as the source domain owing to its great variety.

1) Source Domain: The airborne hyperspectral dataset Chikusei was collected by Headwall Hyperspec-VNIR-C imaging sensor over agricultural and urban areas in Chikusei.

TABLE I
Numbers of Pixels and Land Cover Classes in the Chikusei

ID	Class	Numbers
1	Water	2845
2	Bare soil(school)	2859
3	Bare soil(park)	286
4	Bare soil(farmland)	48525
5	Natural plants	4297
6	Weeds in farmland	1108
7	Forest	20516
8	Grass	6515
9	Rice field(grown)	13369
10	Rice field(first stage)	1268
11	Row crops	5961
12	Plastic house	2193
13	Manmade(non-dark)	1220
14	Manmade(dark)	7664
15	Manmade(blue)	431
16	Manmade(red)	222
17	Manmade grass	1040
18	Asphalt	801
19	Paved ground	145
	Total	77592

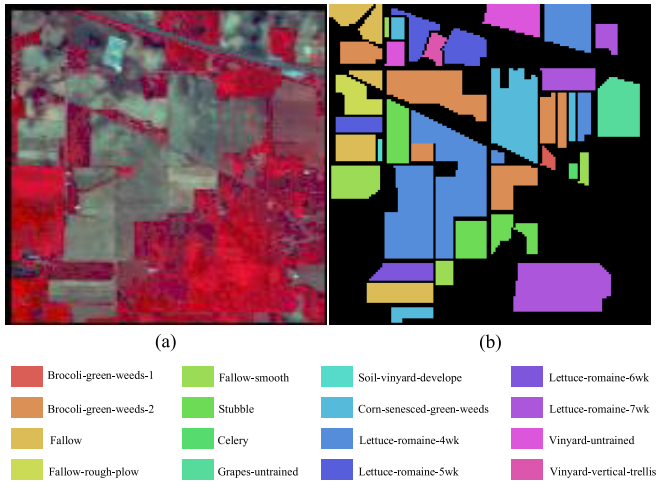


Fig. 5. IP. (a) Pseudo color image (bands 11, 21, and 43). (b) Ground-truth map.

It contains 19 classes and has 2517 × 2335 pixels with 2.5 m per pixel spatial resolution. It consists of 128 spectral bands, ranging from 363 to 1018 nm. The pseudo color image by selecting the 12th, 41st, and 55th bands and corresponding ground-truth map can be seen in Fig. 4. And Table I shows the samples of the Chikusei dataset.

2) Target Domain: The IP dataset was obtained by Air-Borne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor over the IP region in 1992. It contains 145 × 145 pixels and 220 spectral bands covering the range from 0.4 to 2.5 μm. For the sake of model classification, 20 bands absorbed by water vapor are removed and the remaining 200 spectral bands are retained. This dataset contains 16 types of landscapes. The number of pixels in each landscape class are listed in Table II. The pseudo color image with bands the 11th, 21st, and 43th as well as ground-truth map are shown in Fig. 5.

TABLE II
NUMBERS OF PIXELS AND LAND COVER CLASSES IN THE IP

ID	Class	Numbers
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-grass-trees-drives	386
16	Stone-steel-towers	93
	Total	10249

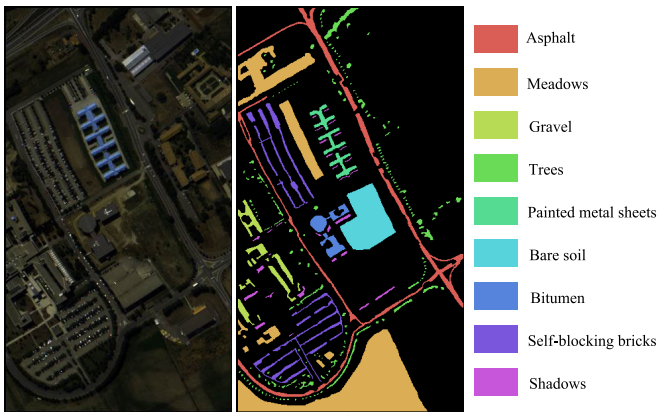


Fig. 6. PU. (a) Pseudo color image (bands 12, 41, and 55). (b) Ground-truth map.

TABLE III
NUMBERS OF PIXELS AND LAND COVER CLASSES IN THE PU

ID	Class	Numbers
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare soil	5029
7	Bitumen	1330
8	Self-blocking bricks	3682
9	Shadows	947
	Total	42776

The PU dataset was acquired by the Reflective Optics System Imaging Spectrometer (ROSIS) over the University of Pavia in Italy in 2001. It contains 610 × 340 pixels in 103 spectral bands and has a wavelength range from 0.43 to 0.86 μm after removing the noisy bands. Table III lists the nine classes and their amounts. Fig. 6 shows the pseudo color image with the 12th, 41st, and 55th bands and the ground-truth map of the dataset.

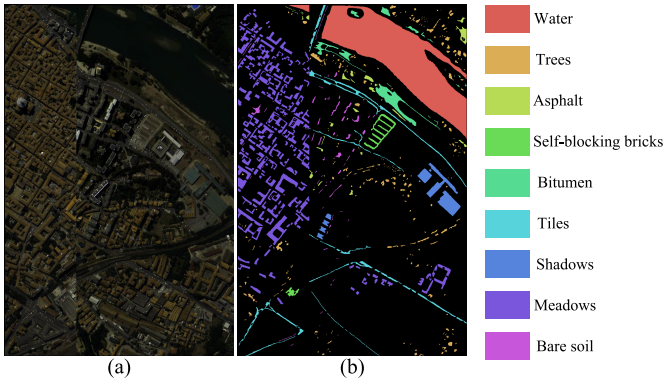


Fig. 7. PC. (a) Pseudo color image (bands 12, 41, and 55). (b) Ground-truth map.

TABLE IV
NUMBERS OF PIXELS AND LAND COVER CLASSES IN THE PC

ID	Class	Numbers
1	Water	65971
2	Trees	7598
3	Asphalt	3090
4	Self-blocking bricks	2685
5	Bitumen	6584
6	Tiles	9248
7	Shadows	7287
8	Meadows	42826
9	Bare soils	2863
	Total	148152

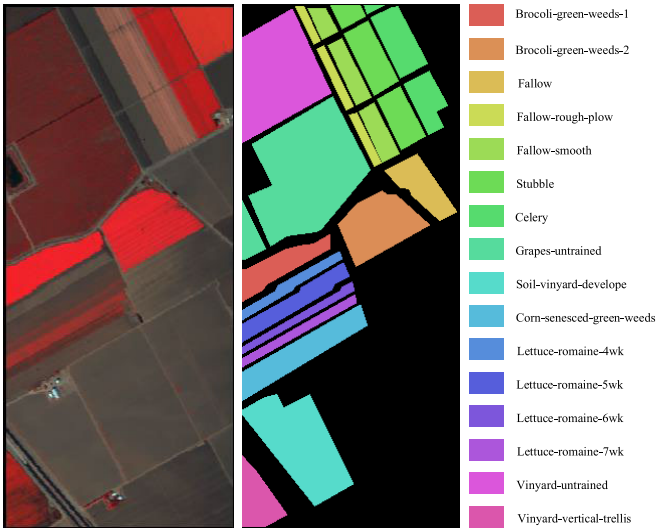
The PC dataset was also acquired by ROSIS over Pavia. It contains 1906 × 715 pixels in 102 spectral bands and has a wavelength range from 0.43 to 0.86 μm after removing the noisy bands. Table IV represents nine categories and corresponding quantities. Fig. 7 shows the pseudo color image with the 12th, 41st, and 55th bands and the ground-truth map of the dataset.

The Salinas dataset captured the Salinas Valley in California by AVIRIS sensor. This dataset contains 512 × 127 pixels in 224 bands. Twenty bands absorbed by water vapor are removed and the remaining 204 spectral bands range from 0.4 to 2.5 μm. Its ground-truth map and pseudo color image with the 11st, 21st, and 43th bands are shown in Fig. 8 and the details of pixels and types are listed in Table V.

In our experiment, four metrics are adopted to evaluate the classification performance quantitatively which are class-specific accuracy, overall accuracy (OA), average accuracy (AA), and kappa coefficient (Kappa).

B. Experimental Settings

All the experiments are conducted by the PyTorch framework and our evaluation is performed using a NVIDIA GeForce RTX 3090 24-GB graphics card. The input is set as a patch size of 9 × 9 × Bands. Adaptive moment estimation (Adam) is used as the optimization scheme and the learning rate is 1.5 × 10^{-3}. We use Xavier normalization to initialize



(a) (b)

Fig. 8. Salinas. (a) Pseudo color image (bands 11, 21, and 43). (b) Ground-truth map.

TABLE V
NUMBERS OF PIXELS AND LAND COVER CLASSES IN THE SALINAS

ID	Class	Numbers
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	Grapes-untrained	11271
9	Soil-vinyard-develope	6203
10	Corn-senesced-green-weeds	3278
11	Lettuce-romaine-4wk	1068
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

the convolution kernel as well as linear layers. The general network parameters of embedded feature extractor is shown in Table VI. As for DA, each domain discriminator consists of four linear layers, whose input is the size of flatten feature. Then the outputs of the linear layers are 1024, and an ReLU nonlinearity activation function and the dropout strategy with 0.5 rate are added after every linear layer. Besides, initial weight sum $\boldsymbol{\lambda}_0$, weight $\boldsymbol{\lambda}_{m-1}$, and weight $\boldsymbol{\lambda}_m$ are 0.1, 0.1, and 0.8, respectively. Each iteration process is to execute the task of N-way K-shot. In light of meta-learning, we set K to 1 and the number of query set is 19 for each category. Finally, the number of training iterations is 10 000 and the Top-k is 25.

For the sake of evaluating the performance of the model, we choose seven supervised and three semisupervised HSI classification methods for comparison. These supervised methods only use one to five labeled samples each type from target

TABLE VI
PARAMETERS OF EMBEDDED FEATURE EXTRACTION

Layer	Operation	Output Size	Filter Size
Input	N/A	$9 \times 9 \times$ Bands	N/A
DR	Conv2d	$9 \times 9 \times 100$	$1 \times 1 \times 100$
Block 1	Conv3d+BN	$9 \times 9 \times 100 \times 8$	$3 \times 3 \times 3 \times 8$
	Conv3d+BN	$9 \times 9 \times 100 \times 8$	$3 \times 3 \times 3 \times 8$
	Conv3d+BN	$9 \times 9 \times 100 \times 8$	$3 \times 3 \times 3 \times 8$
Summation	MaxPooling	$7 \times 7 \times 25 \times 8$	$3 \times 3 \times 4$
	AvgPooling	$7 \times 7 \times 25 \times 8$	$3 \times 3 \times 4$
Block 2	Conv3d+BN	$7 \times 7 \times 25 \times 8$	$3 \times 3 \times 3 \times 8$
	Conv3d+BN	$7 \times 7 \times 25 \times 8$	$3 \times 3 \times 3 \times 8$
	Conv3d+BN	$7 \times 7 \times 25 \times 8$	$3 \times 3 \times 3 \times 8$
Summation	MaxPooling	$5 \times 5 \times 12 \times 8$	$3 \times 3 \times 2$
	AvgPooling	$5 \times 5 \times 12 \times 8$	$3 \times 3 \times 2$
Block 3	Conv3d+BN	$5 \times 5 \times 12 \times 16$	$3 \times 3 \times 3 \times 16$
	Conv3d+BN	$5 \times 5 \times 12 \times 16$	$3 \times 3 \times 3 \times 16$
	Conv3d+BN	$5 \times 5 \times 12 \times 16$	$3 \times 3 \times 3 \times 16$
Summation	MaxPooling	$3 \times 3 \times 7 \times 16$	$3 \times 3 \times 2$
	AvgPooling	$3 \times 3 \times 7 \times 16$	$3 \times 3 \times 2$
CAM	MaxPooling	$1 \times 1 \times 1 \times 16$	$3 \times 3 \times 3 \times 7$
	AvgPooling	$1 \times 1 \times 1 \times 16$	$3 \times 3 \times 3 \times 7$
	Concatenate	$2 \times 1 \times 1 \times 16$	N/A
	Conv2d	$1 \times 1 \times 1 \times 16$	2×1
	Multiply By Channel	$3 \times 3 \times 7 \times 16$	N/A
Block 4	Conv3d+BN	$3 \times 3 \times 7 \times 32$	$3 \times 3 \times 3 \times 32$
	Conv3d+BN	$3 \times 3 \times 7 \times 32$	$3 \times 3 \times 3 \times 32$
	Conv3d+BN	$3 \times 3 \times 7 \times 32$	$3 \times 3 \times 3 \times 32$
Conv3d	Conv3d	$1 \times 1 \times 5 \times 32$	$3 \times 3 \times 3$
Flatten	Flatten	160	N/A

domain and semisupervised methods exploit unlabeled data on this basis. There are five supervised methods including SVM [8], 3-D-CNN [17], deep metric model (DMM) [60], relation network for few-shot classification (RN-FSC) [54], DCFSL [59] Gia-CFSL [69], and CMFSL [70]. Semisupervised methods involve Semi-CNN [63], Two-CNN [81], and 3DAES [73].

C. Classification Results

To demonstrate the superiority of CDSTN in the case of few samples, we selected some classic HSI classification models, including a classic machine learning algorithm, i.e., SVM [8], two kinds of deep learning algorithms, i.e., semi-CNN [63] and 3-D-CNN [17], a semisupervised transfer learning method, i.e., two-CNN [81], two kinds of FSL algorithm supervised, i.e., DMM [60] and semisupervised 3DAES [73], and four kinds of cross-domain FSL algorithm, i.e., RN-FSC [54], DCFSL [59], Gia-CFSL [69], and supervised CMFSL [70].

Considering that supervised methods do not need to obtain prior knowledge from the source domain, the only data available is the labeled data in the target domain. Therefore, for SVM, 3-D-CNN, and DMM, we randomly select the same number of labeled samples from the target domain while we select all the unlabeled samples for Semi-CNN and 3DAES on this basis for the experiment. For RN-FSC and DCFSL, we select Chikusei, Botswana and Kennedy Space Center to form the source domain with a total of 46 categories.

Specifically, SVM is used to solve the maximum-margin hyperplane in linear separable problems by mapping data

TABLE VII
CLASSIFICATION RESULTS (%) ON SALINAS DATASET WITH FIVE LABELED SAMPLE EACH CLASS

Class	SVM	Semi-CNN	3D-CNN	Two-CNN	RN-FSC	DMM	3DAES	DCFSL	Gia-CFSL	CMFSL	CDSTN
1	95.95±3.25	93.72±0.08	74.93±32.27	94.27±2.61	**99.95±0.10**	99.54±0.77	99.05±5.28	93.30±6.09	98.02±1.62	99.83±0.27	97.80±1.43
2	69.68±14.74	26.76±27.64	70.74±38.05	86.89±7.01	95.60±3.46	97.78±0.34	99.71±0.68	98.30±1.07	98.93±1.27	**99.80±0.22**	98.15±1.63
3	30.08±9.33	84.06±8.44	63.99±31.53	53.70±13.00	**98.99±0.84**	96.46±3.48	86.27±12.07	93.68±9.79	96.06±0.39	92.24±1.57	88.91±3.89
4	94.96±1.46	98.35±0.25	91.82±49.90	90.11±5.91	99.55±0.97	87.42±3.37	99.45±0.67	93.92±5.57	94.82±0.78	**99.75±0.21**	99.60±3.09
5	96.38±0.35	92.32±10.40	**96.59±6.10**	92.30±4.53	94.96±4.37	95.35±2.38	90.39±2.92	90.91±6.73	90.44±2.07	91.22±2.18	94.23±3.19
6	94.44±1.30	96.91±0.64	99.41±0.27	96.29±0.39	99.36±0.57	91.97±3.44	99.27±0.87	97.97±1.83	**99.58±0.56**	99.08±0.78	97.22±2.05
7	96.47±1.27	97.66±0.29	74.96±35.31	85.01±15.99	87.35±1.97	99.80±0.17	99.04±2.30	94.52±4.42	99.76±0.15	**99.86±0.17**	94.93±3.02
8	51.30±18.95	32.92±28.79	60.60±27.07	55.11±20.16	86.94±3.26	66.73±12.93	72.61±8.36	85.12±9.34	81.65±5.55	78.94±0.65	**87.50±6.82**
9	90.33±11.67	97.83±1.33	91.59±20.35	92.14±3.81	88.31±3.82	98.70±1.54	94.10±2.94	95.39±3.07	99.65±0.36	**99.88±0.11**	97.09±1.45
10	9.64±19.25	48.87±18.86	50.69±20.86	50.08±16.85	**88.93±8.06**	87.64±4.62	84.51±11.81	87.68±5.82	81.92±5.36	83.37±2.58	85.08±4.64
11	74.67±23.22	68.26±15.85	84.19±17.26	76.09±16.07	**99.13±1.41**	96.11±4.12	98.17±11.00	80.60±13.28	91.93±1.59	91.21±0.78	91.38±8.27
12	90.14±5.56	86.86±12.04	78.69±29.97	66.96±17.80	77.76±10.95	99.77±0.45	99.04±6.89	90.05±8.35	99.48±0.49	**99.78±0.23**	96.45±4.00
13	95.17±5.14	96.18±0.71	93.09±17.41	93.77±7.64	73.63±12.17	96.93±2.09	96.56±1.76	94.15±4.47	92.83±0.53	**99.08±0.48**	96.35±6.72
14	86.32±1.55	94.49±1.73	85.89±16.67	83.07±7.98	94.59±3.42	98.67±0.99	97.97±0.73	87.96±10.25	98.75±0.35	91.14±0.92	**98.71±2.86**
15	57.17±17.39	80.10±22.14	45.22±26.32	44.56±19.01	80.03±3.50	**81.34±10.20**	74.12±10.85	78.44±8.00	71.80±2.53	77.59±4.99	77.60±10.16
16	50.11±17.88	67.02±14.78	64.52±8.46	68.75±8.26	91.00±9.28	**92.74±8.09**	90.62±7.18	87.52±11.01	89.94±8.00	84.12±3.18	87.19±11.05
OA	68.92±3.08	69.68±5.56	69.73±13.59	71.45±3.55	87.84±1.81	89.02±2.75	88.53±1.50	89.10±2.44	89.66 ±0.53	89.86 ±0.62	**90.89±1.45**
AA	73.93±2.30	78.89±3.20	74.62±14.26	76.82±2.91	89.73±1.30	92.93±1.10	92.56±1.33	90.11±2.09	92.85±0.96	92.93±0.29	**93.01±0.95**
Kappa	65.80±3.55	66.46±5.91	66.65±13.87	68.65±3.79	87.43±2.00	87.83±3.03	87.27±1.65	87.88±2.72	88.84±0.58	89.14±0.69	**89.83±1.61**

to high-dimensional space. 3-D-CNN uses 3-D convolution kernel to process his data, which can effectively extract the combined spatial–spectral features, and achieve better classification effect. Semi-CNN adds the loss of the unlabeled sample part to the model loss, and the network is trained to simultaneously minimize the sum of supervised and unsupervised cost functions. Two-CNN learns the joint spectral-spatial features by a two-branch architecture. The method use plenty of samples from source domain to train the bottom and middle layers, then transfer these weights and fine-tuning the top layers with target domain training samples. In addition, the source domain and target domain data are supposed to be from the same sensor, as a result, PC and PU, are the source domain and target domain of each other, which is the same as IP and Salinas.

Furthermore, DMM and RN-FSC can make full use of a small number of labeled samples as two supervised FSL methods, and 3DAES uses unlabeled samples to train an autoencoder, and then uses the coded features of labeled samples to train a Siamese network. All these methods map the data to the metric space, and then select pairs of samples for similarity learning. However, RN-FSC and 3DAES apply cross-domain knowledge and unlabeled samples, respectively, based on FSL. Moreover, DCFSL trains a common feature space of source domain and target domain, so as to apply the prior knowledge of source domain data to target domain. Additionally, Gia-CFSL introduces DA strategy based on graph information and CMFSL transforms samples into a class-covariance metric embedded space. The source domain selection and parameter settings such as learning rate of DCFSL, Gia-CFSL, and CMFSL remain the same as those used for CDSTN.

For the above methods, we adopt a training strategy of few samples, choosing five labeled samples each class as the training set and the rest of the target domain data as the testing set. The semisupervised methods also include all unlabeled samples in the training set. Ten experiments were conducted for each method, and the results of OA, AA, Kappa, and classification accuracy of each category are shown in Tables VII–X. For visual comparison, the classification maps corresponding to each methods are shown in Figs. 9–12. The pseudo color image of each dataset are exhibited from Figs. 5–8. From these classification results, we can draw some conclusions as listed below.

1) Since the deep learning model has stronger feature extraction capabilities, the classic machine learning model SVM algorithm is less effective than other deep learning models in the case of few labeled samples. For example, measured by the OA index, the 3-D-CNN model is 0.81%, 6.58%, 1.68%, and 7.38% higher than the SVM on the four datasets.
2) As for semisupervised methods, compared with the Semi-CNN, Two-CNN uses the data of the source domain to train an encoder, so that with the help of prior knowledge, it achieves a better classification effect than above single-domain methods. In addition to using unlabeled samples to train an encoder, 3DAES also uses iamese network for FSL, which greatly improves the efficiency of data utilization. It is better than the previous semisupervised learning models.
3) FSL models perform better for classification than deep learning methods. RN-FSC, DMM, and 3DAES methods can learn to classify data faster under the condition of few labeled sample by measuring the similarity of pairs of samples. From the classification results of the four datasets, it can be seen that the worst classification results of these three methods are still 16.39%, 0.18%, 11.40%, and 1.75% higher than the previous best model on the OA index.
4) The methods based on the combination of FSL and DA have better performance than the FSL models. On account of the utilization of source domain data information, cross-domain FSL methods perform better on target domain tasks. DCFSL classification results are 0.08%–4.38% higher than the previous best methods in terms of OA. On this basis, the Gia-CFSL and CMFSL

TABLE VIII
CLASSIFICATION RESULTS (%) ON IP DATASET WITH FIVE LABELED SAMPLE EACH CLASS

Class	SVM	Semi-CNN	3D-CNN	Two-CNN	RN-FSC	DMM	3DAES	DCFSL	Gia-CFSL	CMFSL	CDSTN
1	84.86±10.33	73.91±11.17	76.30±15.72	62.17±22.41	22.12±7.86	83.48±14.59	88.60±5.45	92.20±8.64	**92.44±11.82**	88.54±1.62	82.45±11.64
2	24.93±15.20	37.65±10.01	35.64±12.29	26.19±15.40	42.26±0.65	39.03±11.84	47.46±9.28	46.20±8.34	43.90±7.58	44.33±10.05	**48.59±10.67**
3	22.86±18.16	21.59±7.41	24.80±12.26	16.95±9.82	26.84±4.23	45.54±6.62	41.88±10.05	52.73±8.47	45.81±9.41	56.27±4.88	**56.31±12.88**
4	22.45±14.31	35.61±7.84	37.48±9.65	29.09±15.97	48.67±12.00	51.31±7.63	78.61±6.66	84.14±7.36	76.21±11.67	81.13±4.94	**85.34±11.84**
5	14.68±21.44	49.65±13.49	53.79±16.79	33.82±18.79	66.49±6.21	68.36±21.73	67.76±19.68	77.15±7.76	74.00±8.13	**79.90±8.31**	74.33±5.87
6	39.88±18.98	65.48±10.43	73.96±12.54	51.67±15.74	66.05±5.38	**84.85±4.24**	81.25±5.55	83.50±11.36	80.44±8.98	84.48±7.80	75.40±9.57
7	92.98±1.12	90.00±10.93	88.69±10.76	87.50±8.52	49.13±23.48	**100.00±0.00**	94.10±2.14	99.13±1.74	98.26±3.48	**100.00±0.00**	91.58±16.51
8	36.32±16.42	86.19±13.55	72.13±15.71	61.09±17.29	88.82±2.24	92.43±2.41	88.12±5.51	83.34±16.11	86.74±10.84	84.93±13.49	**94.05±3.04**
9	78.00±8.62	99.00±0.00	97.67±4.42	87.50±10.31	21.45±14.10	**100.00±0.00**	96.26±3.00	92.00±0.00	98.00±3.06	**100.00±0.00**	85.33±29.70
10	37.14±17.78	51.33±13.18	40.97±12.67	38.40±17.90	43.60±13.18	63.44±3.34	59.68±10.77	**67.18±5.98**	60.91±8.44	60.36±4.46	66.18±7.76
11	38.58±21.36	46.18±22.67	36.93±15.24	43.98±19.57	**69.53±3.73**	46.22±9.29	63.54±12.33	50.94±18.05	63.40±9.89	65.42±7.79	69.15±3.55
12	16.47±9.47	33.15±15.41	28.01±11.61	11.75±5.82	37.76±3.43	37.64±9.39	39.06±7.65	37.04±5.08	45.53±7.91	41.04±11.43	**62.38±13.85**
13	94.65±2.68	88.29±10.57	87.38±9.98	84.76±9.43	46.31±18.43	99.32±0.96	94.11±11.84	99.40±0.80	98.85±1.40	**99.70±0.33**	90.00±21.91
14	81.00±16.81	72.32±9.01	71.37±16.58	69.81±14.87	**92.13±1.34**	91.57±1.69	83.10±8.04	78.32±8.27	79.74±9.38	86.14±5.39	81.43±6.83
15	12.72±7.85	23.99±16.85	22.88±9.60	33.24±14.63	36.38±5.04	56.94±6.71	65.74±7.31	**77.85±7.67**	72.44±11.98	79.66±8.07	75.61±9.55
16	85.13±1.82	99.35±1.29	69.43±21.94	79.09±14.22	66.44±2.12	98.71±1.72	93.47±1.76	**99.55±0.56**	99.09 ±1.75	88.41±2.05	84.06±31.40
OA	38.72±4.34	49.86±4.70	45.30±5.46	41.64±3.46	50.04±2.64	60.33±3.21	64.42±3.47	66.10±4.46	64.68±2.74	67.32±1.43	**69.19±4.85**
AA	48.92±1.81	60.86±2.22	56.95±5.53	51.06±2.92	51.50±3.19	72.43±1.04	73.94±2.24	76.29±1.97	75.98±2.09	**77.52±0.90**	76.39±2.10
Kappa	31.36±4.03	43.98±4.61	39.21±5.71	35.08±3.30	44.50±2.66	55.75±3.35	60.10±3.61	61.54±1.57	60.11±2.97	63.36±1.44	**65.28±5.44**

TABLE IX
CLASSIFICATION RESULTS (%) ON PU DATASET WITH FIVE LABELED SAMPLE EACH CLASS

Class	SVM	Semi-CNN	3D-CNN	Two-CNN	RN-FSC	DMM	3DAES	DCFSL	Gia-CFSL	CMFSL	CDSTN
1	54.47±11.05	49.91±22.57	41.94±14.79	41.70±16.87	77.18±4.66	67.97±20.19	70.12±11.05	80.01±13.09	81.22±7.99	**81.54±7.65**	78.96±7.34
2	61.47±21.14	59.09±12.95	68.24±14.31	68.71±7.32	88.99±0.49	64.58±8.45	80.81±9.92	78.37±5.27	88.75±4.41	84.12±7.66	**91.54±6.47**
3	34.99±19.01	51.75±24.20	46.01±22.55	29.38±8.27	41.50±2.45	57.90±15.81	52.15±14.78	60.58±3.70	56.11±14.35	**65.12±9.77**	59.36±10.39
4	89.27±2.46	71.70±15.71	84.43±12.00	70.91±18.87	62.97±3.92	83.61±2.64	91.23±3.53	88.50±11.40	91.94±4.70	**93.89±3.23**	89.49±5.68
5	90.69±9.49	90.08±9.91	90.26±14.23	95.43±4.34	95.12±2.82	89.87±0.23	96.91±4.20	96.93±8.23	98.53 ±1.77	**99.30±1.08**	96.43±7.22
6	22.37±21.75	54.56±7.87	35.08±16.92	39.44±9.67	43.71±4.44	**85.74±6.39**	69.88±9.97	80.50±11.93	74.60±9.68	74.69±11.84	71.87±12.99
7	84.15±7.62	83.71±11.00	70.05±20.92	45.11±14.89	30.15±1.64	**87.42±6.12**	71.20±18.03	76.33±8.49	77.69±7.99	72.93±7.82	79.50±8.34
8	53.16±24.03	50.55±27.73	46.54±21.98	51.50±19.83	77.20±2.36	76.44±12.23	66.85±13.88	62.67±21.62	72.90±12.77	**78.92±9.43**	75.34±7.96
9	95.87±0.11	97.72±1.85	95.89±7.26	87.65±9.78	61.31±2.05	**99.81±0.38**	92.49±6.48	98.72±3.94	90.71±1.89	93.11±0.96	90.69±16.11
OA	58.15±6.78	59.09±3.03	59.83±6.94	58.36±6.32	71.23±0.54	73.28±4.43	71.81±4.99	77.66±6.55	83.19±1.98	82.20±2.97	**83.67±3.45**
AA	65.16±1.45	67.68±2.18	64.27±6.64	58.87±6.86	64.24±0.94	79.24±1.18	76.12±3.02	79.84±6.21	81.38±1.71	**82.62±1.75**	81.47±3.23
Kappa	48.51±6.04	49.20±2.34	49.49±7.35	47.50±7.34	62.54±0.63	66.95±4.98	62.86±5.42	73.17±5.68	77.22±2.34	76.90±3.45	**78.76±4.43**

TABLE X
CLASSIFICATION RESULTS (%) ON PC DATASET WITH FIVE LABELED SAMPLE EACH CLASS

Class	SVM	Semi-CNN	3D-CNN	Two-CNN	RN-FSC	DMM	3DAES	DCFSL	Gia-CFSL	CMFSL	CDSTN
1	99.30±0.10	97.80±0.25	98.01±0.43	98.73±0.73	99.02±0.31	99.41±0.27	99.02±0.20	98.83±0.47	99.57±0.19	99.54±0.21	**99.79±0.09**
2	57.47±16.41	78.80±14.75	81.10±17.13	87.65±13.93	78.38±11.67	71.33±36.03	87.88±5.10	88.42±5.30	89.28±2.28	91.08±2.90	**93.10±2.33**
3	76.94±18.35	79.79±20.56	82.05±11.31	89.73±6.17	78.34±13.50	91.18±5.76	92.20±3.49	88.25±3.70	**95.57±1.88**	87.27±0.95	87.86±6.24
4	70.60±19.82	94.47±3.85	61.86±25.15	71.47±21.75	72.71±12.17	88.05±7.22	94.23±5.46	87.28±6.20	92.30±2.44	**96.19±3.44**	93.51±5.42
5	36.54±19.24	82.31±11.21	74.48±15.29	78.01±17.99	68.51±15.70	82.66±6.76	86.48±11.07	89.03±5.21	89.14±3.15	87.88±2.76	**94.33±1.45**
6	93.01±5.62	64.84±8.41	84.76±17.90	77.53±34.64	88.04±7.85	**99.24±0.17**	98.34±7.38	92.70±5.03	95.90±1.16	97.18±1.26	96.50±2.86
7	64.09±13.75	81.04±3.51	78.22±8.27	73.36±15.85	74.48±14.15	**87.29±3.62**	67.83±3.81	81.24±2.87	84.33±1.68	86.48±2.08	83.87±0.76
8	72.15±12.83	65.93±10.77	89.36±11.95	72.19±29.07	95.79±2.23	97.67±0.63	97.29±6.24	97.64±1.41	97.28 ±1.02	98.36±0.99	**98.81±0.29**
9	98.88±1.93	96.01±2.93	96.71±13.28	78.50±30.87	**99.73±0.24**	97.04±2.13	99.27±5.09	96.14±2.17	97.77 ±2.03	95.94±0.53	97.78±1.59
OA	83.40±3.37	84.21±2.97	90.78±4.77	91.13±13.72	92.88±1.09	95.69±2.03	95.59±1.85	95.79±0.78	96.69±0.37	97.07±0.40	**97.53±0.16**
AA	74.33±2.56	82.33±1.62	82.95±6.80	82.80±17.65	83.89±2.66	90.43±4.43	91.39±1.84	91.06±0.92	93.46±0.38	93.32±0.34	**93.95±0.80**
Kappa	77.10±4.35	76.89±3.70	87.20±6.43	85.10±18.17	89.98±1.54	93.92±2.87	92.89±2.50	93.22±0.72	95.43±0.52	96.03±0.56	**96.51±0.23**

TABLE XI
ABLATION STUDY (%) ON THE OA INDEX FOR DATASETS WITH FIVE LABELED SAMPLES EACH CLASS

Data set	CAM	Without D_1	Without D_2	Without D_3	Without D_1 and D_2	Without D_1, D_2, D_3	Without PLR	CDSTN
Salinas	90.42±1.66	90.32±0.82	89.97±0.97	90.52±0.97	89.78±1.45	89.79±1.71	88.98±1.16	**90.89±1.45**
Indian Pines	68.24±3.51	68.41±3.88	67.00±3.27	68.26±4.03	67.67±3.16	65.96±4.96	64.69±1.26	**69.19±4.85**
Pavia University	83.40±2.53	83.14±2.75	83.43±3.49	84.30±2.02	82.79±1.99	82.50±2.62	80.86±2.20	**83.67±3.45**
Pavia Center	97.20±0.33	97.44±0.20	97.27±0.17	97.48±0.07	97.39±0.33	97.34±0.33	96.74±0.22	**97.53±0.16**

models, which prioritize intraclass similarity and interclass differences and employ superior cross-domain methods, enhance performance in this field, achieving optimal AA metrics for specific datasets. Furthermore, CDSTN with semisupervised learning achieves the highest OA and kappa scores, surpassing the previous model by 1.03%, 1.87%, 0.48%, and 0.46% on the OA index.

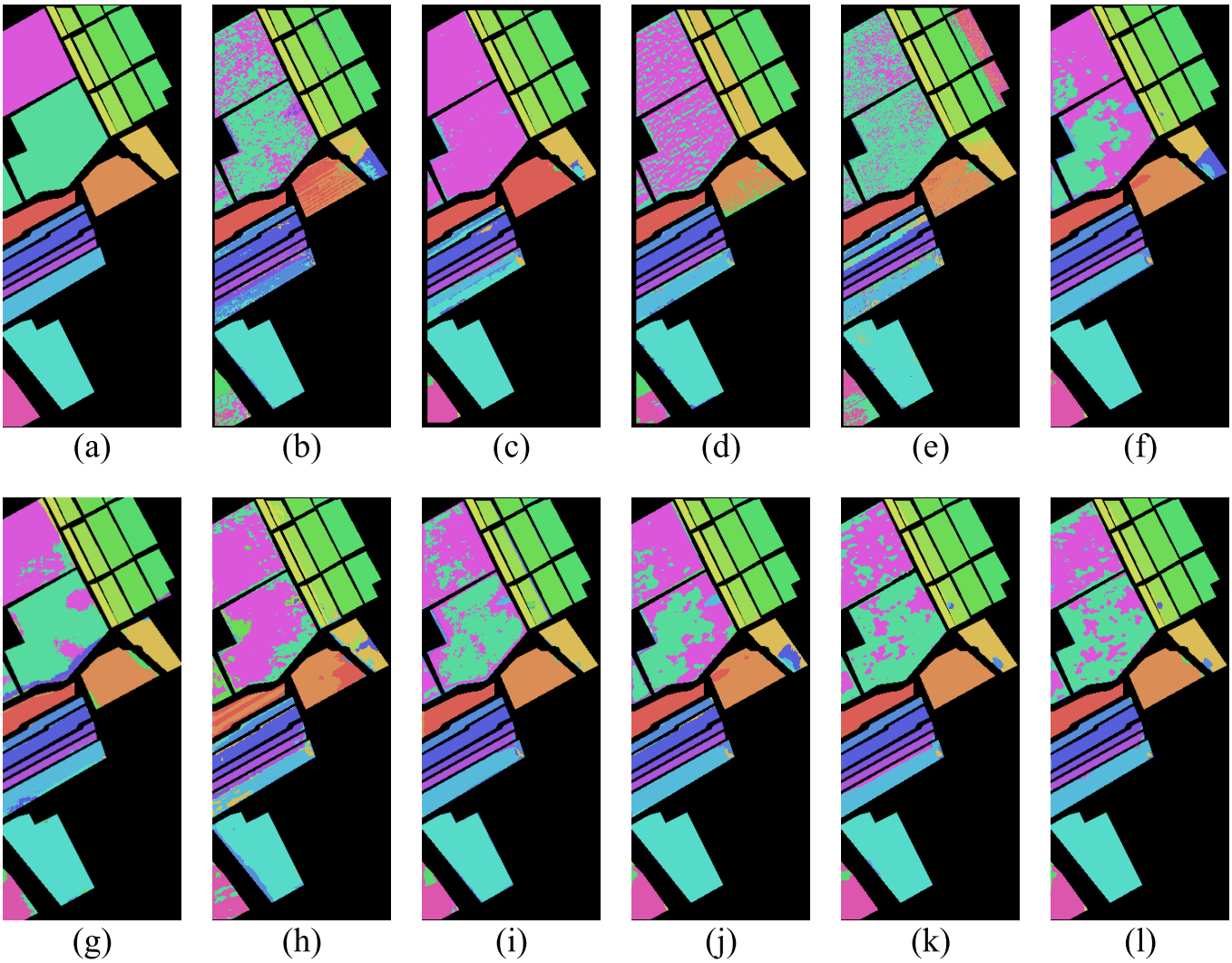


Fig. 9. Salinas. (a) Ground-truth map. (b) SVM (68.92%). (c) Semi-CNN (69.68%). (d) 3-D-CNN (69.73%). (e) Two-CNN (71.45%). (f) RN-FSC (87.84%). (g) DMM (89.02%). (h) 3DAES (88.53%). (i) DCFSL (89.10%). (j) Gia-CFSL (89.66%). (k) CMFSL (89.86%). (l) CDSTN (90.89%).

TABLE XII
RESULTS (%) WITH DIFFERENT PARAMETERS FOR DA

		λ_m/λ_{m-1}								
		0.7/0.1	0.7/0.2	0.7/0.3	0.8/0.1	0.8/0.2	0.8/0.3	0.9/0.1	0.9/0.2	0.9/0.3
λ_0	0.1	90.5	90.3	88.63	**90.89**	89.49	89.3	89.25	88.6	90.17
	0.2	90.88	89.55	89.36	88.96	88.18	90.04	89.44	89.67	89.11
	0.3	90.01	90.08	89.74	90.78	90.03	89.18	89.68	90.6	88.82

Considering that the number of labeled samples in the target domain has a certain impact on the model, we select 1–4 samples to repeat the above experiments to explore the importance of this factor. The mean OA of the classification results for each method is shown in Fig. 13. It can be seen from Fig. 13 that although the classification difference between different methods fluctuates under the same number of labels, more labels obviously contribute more to the model classification.

D. Ablation Study

The CAM, multilevel adversarial DA strategy and PLR are implemented in CDSTN. To investigate the necessity of each module, ablation studies are conducted on each module using the four datasets in this section.

1) Channel Attention Module: Due to max-pooling and average-pooling, we can fully extract the important information, aggregate these information through 2-D-CNN and linear layer, and further analyze the contribution of different channels to common embedded feature extraction. Therefore, we verify the necessity of the CAM by removing it. The comparison results are shown in Table XI. We can discover that after removing the CAM, the classification results are affected slightly.

2) Multilevel Adversarial Domain Adaptation Strategy: As described in Section III-B, the domain gap between source and target domains can be reduced by performing domain alignment at different levels of the network. Since a series of domain discriminator can identify both common and unique domain information through domain adversarial strategy leading great DA. Let the discriminators after 1–4 blocks be D_1, D_2, D_3, and D_4, respectively. We conduct comparative experiments by removing D_1, D_2, and D_3, respectively, while

TABLE XIII
TRAINING TIME, FLOPS (M, MILLION), AND NUMBER OF PARAMETERS (M, MILLION) OF DIFFERENT METHODS

		SVM	Semi-CNN	3D-CNN	Two-CNN	RN-FSC	DMM	3DAES	DCFSL	Gia-CFSL	CMFSL	CDSTN
Salinas	Training time(s)	0.02	544.79	51.24	1217.85	472.52	187.01	7428.31	2092.52	2490.59	1422.11	1134.45
	FLOPs(MMac)	-	11.94	45.74	15.13	6.25	10.33	3444.30	47.01	67.82	28.51	852.20
	#params(M)	-	4.83	0.63	1.91	0.19	0.14	7.16	4.26	0.31	0.65	2.19
Indian Pines	Training time(s)	0.02	105.58	27.21	1359.4	309.11	84.24	1483.50	1936.84	3916.49	1386.91	1072.45
	FLOPs(MMac)	-	11.74	45.03	15.13	6.25	10.29	3363.79	46.98	67.79	28.47	852.17
	#params(M)	-	4.82	0.62	1.91	0.19	0.14	7.16	4.26	0.31	0.65	2.19
PaviaU	Training time(s)	0.01	667.97	22.18	1553.53	275.86	265.81	12036.99	996.10	2673.92	763.49	704.17
	FLOPs(MMac)	-	7.89	21.57	13.47	6.25	9.29	1745.80	46.19	67.00	27.69	557.20
	#params(M)	-	4.36	0.18	1.74	0.19	0.12	3.55	4.26	0.31	0.64	2.18
Pavia	Training time(s)	0.05	1716.18	20.64	1654.45	180.67	619.06	35823.21	1469.07	1851.60	791.24	849.66
	FLOPs(MMac)	-	7.93	21.57	13.47	6.25	9.27	1759.00	46.18	67.01	27.68	557.20
	#params(M)	-	4.36	0.18	1.74	0.19	0.12	4.01	4.26	0.31	0.64	2.18

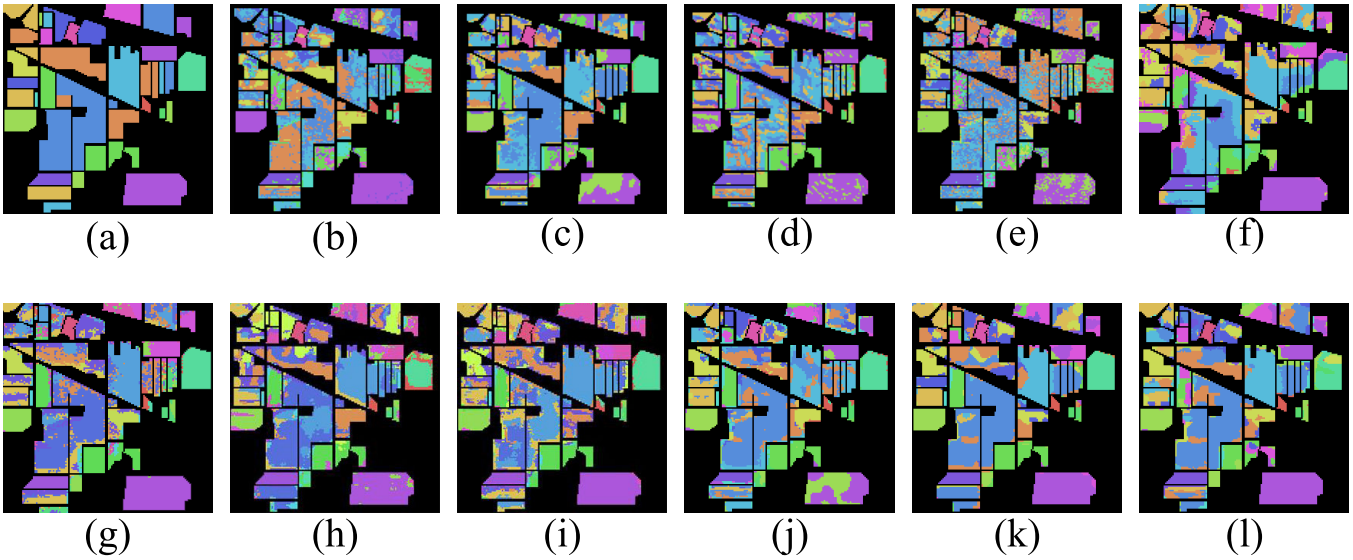


Fig. 10. IP. (a) Ground-truth map. (b) SVM (38.72%). (c) Semi-CNN (49.86%). (d) 3-D-CNN (45.30%). (e) Two-CNN (41.64%). (f) RN-FSC (50.04%). (g) DMM (60.33%). (h) 3DAES (64.42%). (i) DCFSL (66.10%). (j) Gia-CFSL (64.68%). (k) CMFSL (67.32%). (l) CDSTN (69.19%).

keeping D_4. The results are presented in Table XI. Table XI shows that a higher number of domain discriminators generally leads to better domain alignment. When one of D_1, D_2, and D_3 is missing, the model performance drops, the difference is not significant, but when removing 2 or 3 of them, the model deteriorates rapidly. Also, the experiment without D_1 and D_2 demonstrates discriminators from low blocks are very beneficial for DA.

3) Pseudo Label Refinement: PLR greatly improves the quality of pseudo labels obtained by introducing a linear regression model to assess the confidence level, which enables the network to use more data for training. To evaluate the function of this module, we designed a comparative experiment to remove and compare the classification result. Table XI lists the results of the comparative experiments, which demonstrate that the PLR module effectively enhances the accuracy of labeling unlabeled samples and improves the utilization of such samples.

E. Parameter Tuning

With regard to CDSTN, there are a certain number or amount hyperparameters that need to be adjusted, some of which have a great impact on the performance of the model. Therefore, we design a series of comparative experiments to explore the impact of parameters on the model. Specifically, the parameters weight $\boldsymbol{\lambda}_m$, weight $\boldsymbol{\lambda}_{m-1}$ and initial weight sum $\boldsymbol{\lambda}_0$ of the DA part and the number of pseudo labels selected Top-k from the PLR part.

Specially, $\boldsymbol{\lambda}_0$ determines the initial weight of D_1 and D_2, while their specific weights are automatically adjusted by the model. After the fourth block, the data is input into the FSL module or the PLR module, which means DA should have been achieved meanwhile. Therefore, the value of $\boldsymbol{\lambda}_m$ representing the effect of D_4 should be higher than other $\boldsymbol{\lambda}_k$. We design some experiments with different parameters, the detail can be seen from Table XII. From Table XII, we can figure out that overall, a range of intermediate values for $\boldsymbol{\lambda}_m$ with small values of $\boldsymbol{\lambda}_0$ and $\boldsymbol{\lambda}_{m-1}$ often yields better results. In fact, a larger weight of $\boldsymbol{\lambda}_m$ usually means that the domain adversarial strategy can be used to achieve DA faster after the fourth block, while a larger $\boldsymbol{\lambda}_0$ means that more auxiliary information from the shallow network can be used to help achieve DA. The results show that $\boldsymbol{\lambda}_m = 0.8$, $\boldsymbol{\lambda}_{m-1} = 0.1$, and $\boldsymbol{\lambda}_0 = 0.1$ are the best combination of parameters. The auxiliary

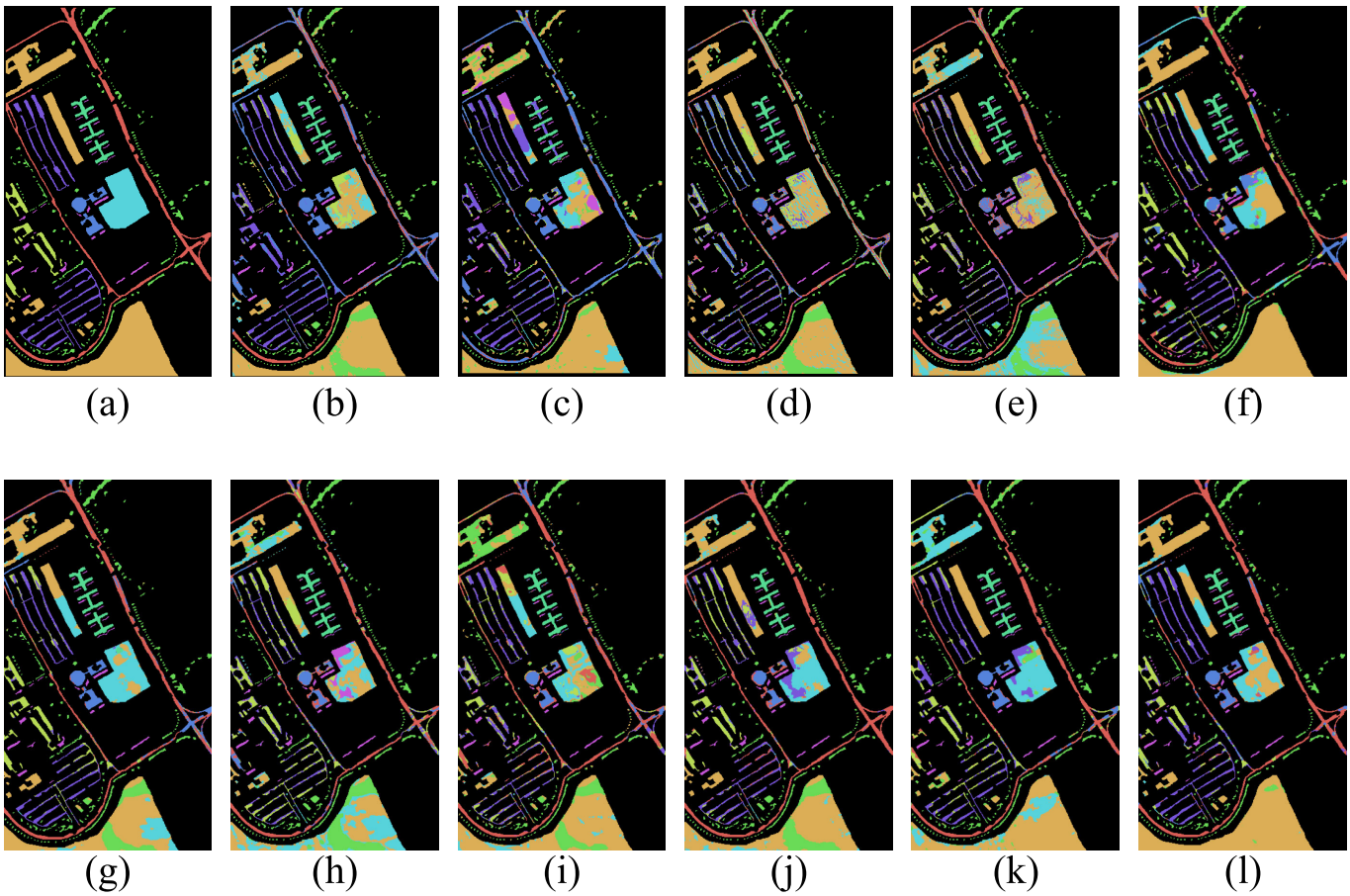


Fig. 11. PU. (a) Ground-truth map. (b) SVM (58.15%). (c) Semi-CNN (59.09%). (d) 3-D-CNN (59.83%). (e) Two-CNN (58.36%). (f) RN-FSC (71.23%). (g) DMM (73.28%). (h) 3DAES (71.81%). (i) DCFSL (77.66%). (j) Gia-CFSL (83.19%). (k) CMFSL (82.20%). (l) CDSTN (83.67%).

information used from the shallow network is moderate, which does not affect the domain alignment efficiency of the final feature vector, and also provides enough information to assist in achieving better domain alignment.

Besides, in the first stage of PLR, we input all unlabeled samples into the network for feature extraction and classification. Here, we select Top-k number of samples each class with the highest classification probability as dataset to be filtered, which are further screened in the second stage. The selected results in the second stage also vary depending on the parameter Top-k. For this part of the experiment, we call the pseudo labels selected in the first stage of the PLR module many times and compare them with the true labels. We find that the top five samples with the highest probability of each class selected in the first stage still have wrong pseudo labels, which also proves the necessity of refinement of pseudo labels. In fact, choosing a larger Top-k obviously lead to better results, but some datasets only have a small number of samples in a certain category, a too large Top-k is unnecessary. For example, the ninth category Oats in the IP dataset has only 20 samples. Considering the principle of assigning labels, the classification accuracy of most categories is between 50% and 95% when the PLR module is not used, then Top-$k = 25$ can guarantee that there are at least 12 correct labels to choose from among the pseudo-labels. This leaves sufficient screening space for the second stage. Finally, multiple experiments on the dataset also prove that Top-$k = 25$ can ensure that there are sufficient pseudo labels to choose for the second stage of PLR module.

F. Analysis of the Computational Complexity

Time complexity analysis is a fundamental approach used to assess the computational efficiency of an algorithm. In HSI classification, where large-scale HSI data are typically involved, it is critical to consider the time complexity when selecting and designing algorithms. This analysis helps us estimate the time and computational resources required for processing the input data. In this study, we evaluate the time complexity of the model using the number of floating-point operations per second (FLOPs), parameters, and training time.

Table XIII presents the training time, FLOPs and parameters of various methods on different datasets. As for supervised learning methods, the iterative training process of the network with labeled samples constitutes the training time. It is worth noting that for semisupervised learning methods, such as semi-CNN, two-CNN, and 3DAES, the training time includes both labeled and unlabeled samples.

As shown in Table XIII, semisupervised methods tend to require more training time than supervised methods with a similar network size, owing to the additional training of unlabeled samples. However, CDSTN only provides pseudo labels

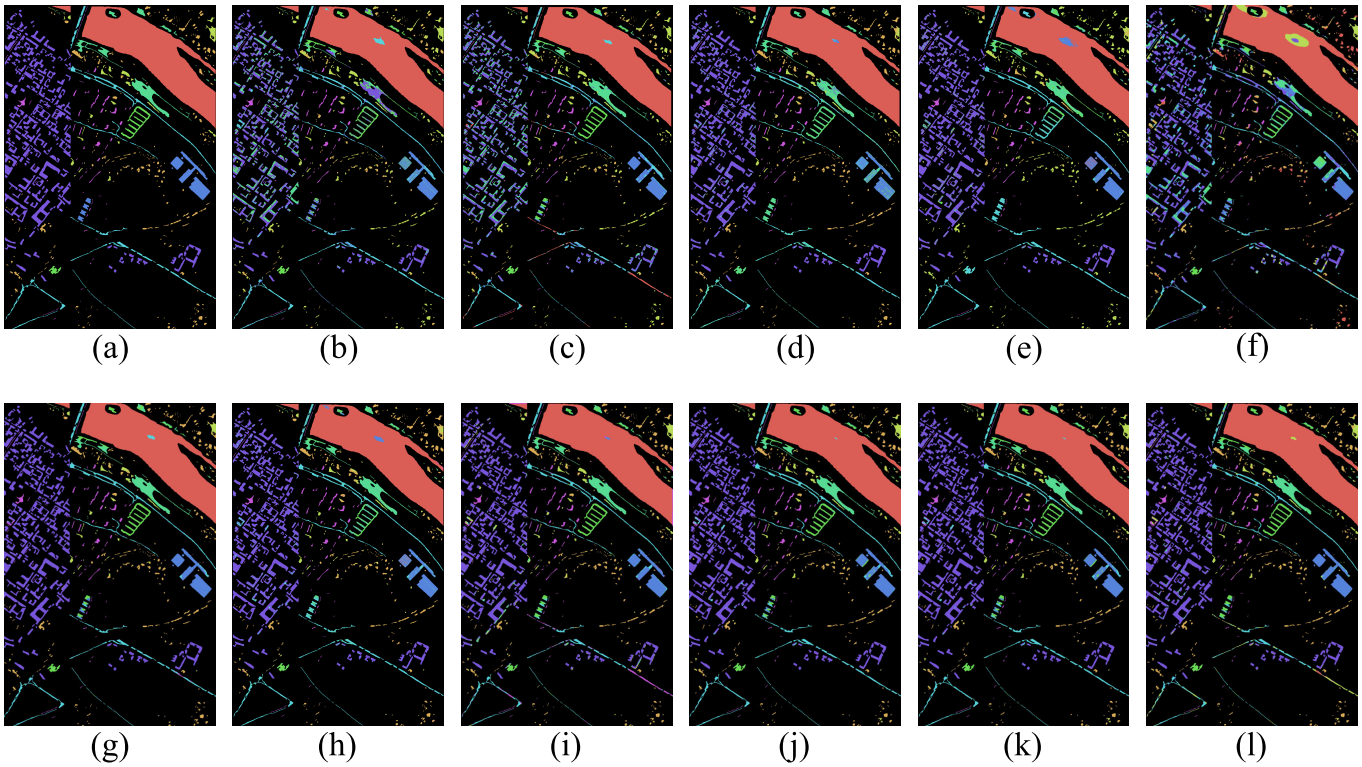


Fig. 12. PC. (a) Ground-truth map. (b) SVM (83.40%). (c) Semi-CNN (84.21%). (d) 3-D-CNN (90.78%). (e) Two-CNN (91.13%). (f) RN-FSC (92.88%). (g) DMM (95.69%). (h) 3DAES (95.59%). (i) DCFSL (95.79%). (j) Gia-CFSL (96.69%). (k) CMFSL (97.07%). (l) CDSTN (97.53%).

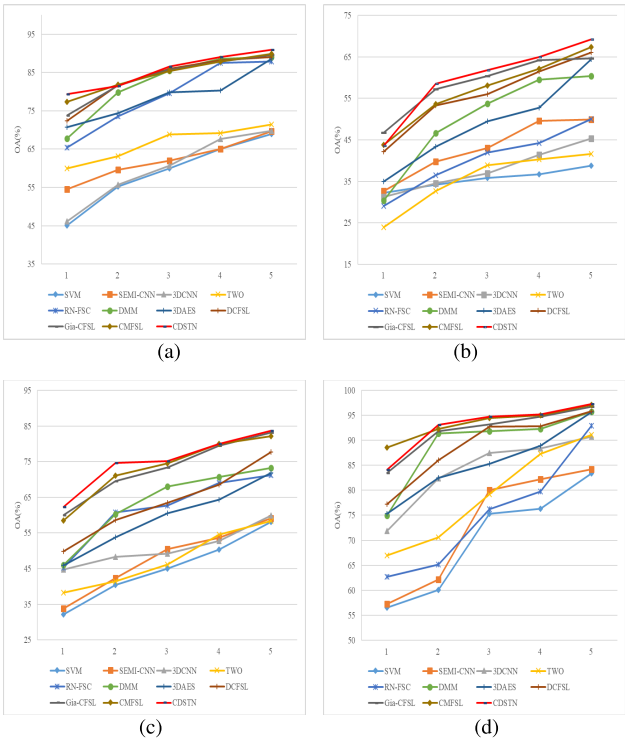


Fig. 13. OA with different number of labeled samples on the different methods. (a) Salinas. (b) IP. (c) PU. (d) PC.

to the unlabeled samples without gradient training, resulting in faster training time. Besides, cross-domain methods often require more training time but they have better performance than other methods. Due to the large size of the feature maps in the shallow layer of the network, the FLOPs value increases when performing linear layer operations in the corresponding domain classifier. However, the overall algorithm's simplicity ensures that this issue does not significantly impact the training time. Although CDSTN has higher parameters and FLOPs compared to some methods, it exhibits the best performance.

V. Conclusion

In this article, CDSTN has been proposed to solve the issues of lacking labeled samples for HSI classification. It makes full use of available data information to support target domain HSI classification by combining FSL, semisupervised learning and DA strategies. Specifically, we build an embedded feature extractor with a CAM to achieve efficient extraction of deeper spatial–spectral joint features. And based on the idea of adversarial and optimization, we introduce a series of domain classifiers to extract common and unique domain information from all levels of the network, which realize excellent DA and make better use of a large number of labeled samples from source domains. Furthermore, our PLR module selects samples with the highest confidence through two stages, alleviating the huge damage caused by wrongly pseudo labeled samples to FSL model. Extensive experiments on four benchmark datasets show that the proposed CDSTN presents superiority over the other classical models. In addition, several ablation studies also validate the effectiveness of different components.

In the future, further research will be implemented on FSL and DA, so as to inject more vitality into HSI classification

with more prior information. Moreover, we will also investigate the overfitting problem in FSL models and work toward improving the theoretical basis of DA.

REFERENCES

[1] S. Meng, X. Wang, X. Hu, C. Luo, and Y. Zhong, "Deep learning-based crop mapping in the cloudy season using one-shot hyperspectral satellite imagery," *Comput. Electron. Agricult.*, vol. 186, Jul. 2021, Art. no. 106188.

[2] P. Ghamisi et al., "Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 37–78, Dec. 2017.

[3] Y.-Q. Wan, Y.-H. Fan, and M.-S. Jin, "Application of hyperspectral remote sensing for supplementary investigation of polymetallic deposits in Huaniushan ore region, Northwestern China," *Sci. Rep.*, vol. 11, no. 1, pp. 1–12, Jan. 2021.

[4] C. Niu, K. Tan, X. Jia, and X. Wang, "Deep learning based regression for optically inactive inland water quality parameter estimation using airborne hyperspectral imagery," *Environ. Pollut.*, vol. 286, Oct. 2021, Art. no. 117534.

[5] F. Xue, F. Tan, Z. Ye, J. Chen, and Y. Wei, "Spectral–spatial classification of hyperspectral image using improved functional principal component analysis," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022.

[6] J. Huang, K. Liu, M. Xu, M. Perc, and X. Li, "Background purification framework with extended morphological attribute profile for hyperspectral anomaly detection," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 8113–8124, 2021.

[7] Y. Zhang, K. Liu, Y. Dong, K. Wu, and X. Hu, "Semisupervised classification based on SLIC segmentation for hyperspectral image," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 8, pp. 1440–1444, Aug. 2019.

[8] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778–1790, Aug. 2004.

[9] L. Samaniego, A. Bárdossy, and K. Schulz, "Supervised classification of remotely sensed imagery using a modified k-NN technique," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 7, pp. 2112–2125, Jul. 2008.

[10] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2094–2107, Jun. 2014.

[11] Y. Chen, X. Zhao, and X. Jia, "Spectral–spatial classification of hyperspectral data based on deep belief network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2381–2392, Jun. 2015.

[12] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3639–3655, Jul. 2017.

[13] M. Ansari, S. Homayouni, A. Safari, and S. Niazmardi, "A new convolutional kernel classifier for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 11240–11256, 2021.

[14] W. Boulila, M. Sellami, M. Driss, M. Al-Sarem, M. Safaei, and F. A. Ghaleb, "RS-DCNN: A novel distributed convolutional-neural-networks based-approach for big remote-sensing image classification," *Comput. Electron. Agricult.*, vol. 182, Mar. 2021, Art. no. 106014.

[15] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *J. Sensors*, vol. 2015, pp. 1–12, Jan. 2015.

[16] J. Yue, W. Zhao, S. Mao, and H. Liu, "Spectral–spatial classification of hyperspectral images using deep convolutional neural networks," *Remote Sens. Lett.*, vol. 6, no. 6, pp. 468–477, Jun. 2015.

[17] M. He, B. Li, and H. Chen, "Multi-scale 3D deep convolutional neural network for hyperspectral image classification," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 3904–3908.

[18] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri, "HybridSN: Exploring 3-D–2-D CNN feature hierarchy for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 2, pp. 277–281, Feb. 2019.

[19] L. Zhu, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Generative adversarial networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 9, pp. 5046–5063, Sep. 2018.

[20] Z. Zhong, J. Li, Z. Luo, and M. Chapman, "Spectral–spatial residual network for hyperspectral image classification: A 3-D deep learning framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 2, pp. 847–858, Aug. 2017.

[21] W. Wang, S. Dou, Z. Jiang, and L. Sun, "A fast dense spectral–spatial convolution network framework for hyperspectral images classification," *Remote Sens.*, vol. 10, no. 7, p. 1068, Jul. 2018.

[22] W. Ma, Q. Yang, Y. Wu, W. Zhao, and X. Zhang, "Double-branch multi-attention mechanism network for hyperspectral image classification," *Remote Sens.*, vol. 11, no. 11, p. 1307, Jun. 2019.

[23] W. Li et al., "Data augmentation for hyperspectral image classification with deep CNN," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 4, pp. 593–597, Apr. 2018.

[24] S. Yu, S. Jia, and C. Xu, "Convolutional neural networks for hyperspectral image classification," *Neurocomputing*, vol. 219, pp. 88–98, Jan. 2017.

[25] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, Jul. 2015, pp. 97–105.

[26] 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.

[27] C. Deng, X. Liu, C. Li, and D. Tao, "Active multi-kernel domain adaptation for hyperspectral image classification," *Pattern Recognit.*, vol. 77, pp. 306–315, May 2018.

[28] Z. Sun, C. Wang, H. Wang, and J. Li, "Learn multiple-kernel SVMs for domain adaptation in hyperspectral data," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 5, pp. 1224–1228, Sep. 2013.

[29] Y. Zhang, W. Li, R. Tao, J. Peng, Q. Du, and Z. Cai, "Cross-scene hyperspectral image classification with discriminative cooperative alignment," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 1, pp. 9646–9660, Nov. 2021.

[30] X. Zhao, M. Zhang, R. Tao, W. Li, W. Liao, and W. Philips, "Cross-domain classification of multisource remote sensing data using fractional fusion and spatial–spectral domain adaptation," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 5721–5733, 2022.

[31] 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.

[32] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural. Inf. Process. Syst.*, vol. 30, Dec. 2017, pp. 1–14.

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

[34] S. Benaim and L. Wolf, "One-shot unsupervised cross domain translation," in *Proc. Adv. Neural. Inf. Process. Syst.*, vol. 31, Dec. 2018, pp. 1–22.

[35] H. Qi, M. Brown, and D. G. Lowe, "Low-shot learning with imprinted weights," in *Proc. CVPR*, Jun. 2018, pp. 5822–5830.

[36] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[37] T. Pfister, J. Charles, and A. Zisserman, "Domain-adaptive discriminative one-shot learning of gestures," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2014, pp. 814–829.

[38] M. Douze, A. Szlam, B. Hariharan, and H. Jegou, "Low-shot learning with large-scale diffusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3349–3358.

[39] 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 Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 1199–1208.

[40] D. J. Rezende, S. Mohamed, I. Danihelka, K. Gregor, and D. Wierstra, "One-shot generalization in deep generative models," in *Proc. ICML*, Jun. 2016, pp. 1521–1529.

[41] Z. Luo, Y. Zou, J. Hoffman, and L. F. Fei-Fei, "Label efficient learning of transferable representations acrosss domains and tasks," in *Proc. Adv. Neural. Inf. Process. Syst.*, vol. 30, Dec. 2017, pp. 1–16.

[42] Y. Zhang, H. Tang, and K. Jia, "Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 233–248.

[43] J. Donahue et al., "DeCAF: A deep convolutional activation feature for generic visual recognition," in *Proc. 31st IEEE Int. Conf. Mach. Learn.*, Jun. 2014, pp. 647–655.

[44] Y. Lee and S. Choi, "Gradient-based meta-learning with learned layer-wise metric and subspace," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2018, pp. 2927–2936.

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

[46] B. Wang, L. Li, M. Verma, Y. Nakashima, R. Kawasaki, and H. Nagahara, "MTUNet: Few-shot image classification with visual explanations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 2294–2298.

[47] W.-H. Chu, Y.-J. Li, J.-C. Chang, and Y.-C. F. Wang, "Spot and learn: A maximum-entropy patch sampler for few-shot image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6251–6260.

[48] Y. Lu, X. Chen, Z. Wu, and J. Yu, "Decoupled metric network for single-stage few-shot object detection," *IEEE Trans. Cybern.*, vol. 53, no. 1, pp. 514–525, Jan. 2023.

[49] J. Wu, S. Liu, D. Huang, and Y. Wang, "Multi-scale positive sample refinement for few-shot object detection," in *Proc. Eur. Conf. Comput. Vis.*, Aug. 2020, pp. 456–472.

[50] G. Cheng, C. Lang, and J. Han, "Holistic prototype activation for few-shot segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 4, pp. 1–17, Apr. 2023.

[51] C. Lang, G. Cheng, B. Tu, and J. Han, "Learning what not to segment: A new perspective on few-shot segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8057–8067.

[52] C. Zhang, J. Yue, and Q. Qin, "Global prototypical network for few-shot hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4748–4759, 2020.

[53] D. Pal, V. Bundele, B. Banerjee, and Y. Jeppu, "SPN: Stable prototypical network for few-shot learning-based hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2021.

[54] 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.

[55] Z. Cao, X. Li, J. Jianfeng, and L. Zhao, "3D convolutional Siamese network for few-shot hyperspectral classification," *J. Appl. Remote Sens.*, vol. 14, no. 4, Nov. 2020, Art. no. 048504.

[56] C. Zhang, J. Yue, and Q. Qin, "Deep quadruplet network for hyperspectral image classification with a small number of samples," *Remote Sens.*, vol. 12, no. 4, p. 647, Feb. 2020.

[57] X. Liang, Y. Zhang, and J. Zhang, "Attention multisource fusion-based deep few-shot learning for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 8773–8788, 2021.

[58] Y. Shi, J. Li, Y. Li, and Q. Du, "Sensor-independent hyperspectral target detection with semisupervised domain adaptive few-shot learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 8, pp. 6894–6906, Aug. 2021.

[59] 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, 2022, Art. no. 5501618.

[60] B. Deng, S. Jia, and D. Shi, "Deep metric learning-based feature embedding for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 2, pp. 1422–1435, Feb. 2020.

[61] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Semisupervised hyperspectral image classification using soft sparse multinomial logistic regression," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 2, pp. 318–322, Mar. 2013.

[62] X. Zhang, Q. Song, R. Liu, W. Wang, and L. Jiao, "Modified co-training with spectral and spatial views for semisupervised hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2044–2055, Jun. 2014.

[63] B. Liu, X. Yu, P. Zhang, X. Tan, A. Yu, and Z. Xue, "A semi-supervised convolutional neural network for hyperspectral image classification," *Remote Sens. Lett.*, vol. 8, no. 9, pp. 839–848, Sep. 2017.

[64] J. Xia, J. Chanussot, P. Du, and X. He, "(Semi-) supervised probabilistic principal component analysis for hyperspectral remote sensing image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2224–2236, Jun. 2014.

[65] Y. Guo et al., "A broader study of cross-domain few-shot learning," in *Proc. Eur. Conf. Comput. Vis.*, Aug. 2020, pp. 124–141.

[66] Y. Hu and A. J. Ma, "Adversarial feature augmentation for cross-domain few-shot classification," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2022, pp. 20–37.

[67] F. Zhao, T. Huang, and D. Wang, "Graph few-shot learning via restructuring task graph," *IEEE Trans. Neural Netw. Learn. Syst.*, early acess, Jun. 8, 2022, doi: 10.1109/TNNLS.2022.3178849.

[68] 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.

[69] Y. Zhang, W. Li, M. Zhang, S. Wang, R. Tao, and Q. Du, "Graph information aggregation cross-domain few-shot learning for hyperspectral image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 30, 2022, doi: 10.1109/TNNLS.2022.3185795.

[70] B. Xi, J. Li, Y. Li, R. Song, D. Hong, and J. Chanussot, "Few-shot learning with class-covariance metric for hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 31, pp. 5079–5092, 2022.

[71] M. Ren et al., "Meta-learning for semi-supervised few-shot classification," 2018, *arXiv:1803.00676*.

[72] Z. Yu, L. Chen, Z. Cheng, and J. Luo, "TransMatch: A transfer-learning scheme for semi-supervised few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12856–12864.

[73] S. Jia et al., "A semisupervised Siamese network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5516417.

[74] L. Qiao, Y. Shi, J. Li, Y. Tian, T. Huang, and Y. Wang, "Transductive episodic-wise adaptive metric for few-shot learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3603–3612.

[75] P. Rodríguez, I. Laradji, A. Drouin, and A. Lacoste, "Embedding propagation: Smoother manifold for few-shot classification," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Aug. 2020, pp. 121–138.

[76] X. Li et al., "Learning to self-train for semi-supervised few-shot classification," in *Proc. Adv. Neural. Inf. Process. Syst.*, vol. 32, Dec. 2019, pp. 1–13.

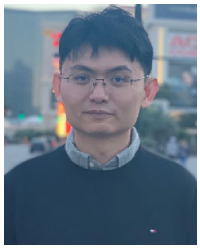
[77] W. Zhang, W. Ouyang, W. Li, and D. Xu, "Collaborative and adversarial network for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3801–3809.

[78] Y. Ganin et al., "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 59, pp. 1–35, 2016.

[79] Y. Wang, C. Xu, C. Liu, L. Zhang, and Y. Fu, "Instance credibility inference for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 12836–12845.

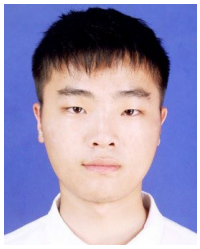
[80] N. Simon, J. Friedman, and T. Hastie, "A blockwise descent algorithm for group-penalized multiresponse and multinomial regression," 2013, *arXiv:1311.6529*.

[81] J. Yang, Y.-Q. Zhao, and J. C.-W. Chan, "Learning and transferring deep joint spectral–spatial features for hyperspectral classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 8, pp. 4729–4742, Aug. 2017.



Mingyang Zhang (Member, IEEE) received the B.S. degree in automation and the Ph.D. degree in pattern recognition and intelligent systems from Xidian University, Xi'an, China, in 2012 and 2018, respectively.

Since 2018, he has been a Lecturer with the School of Electronic Engineering, Xidian University. His research interests include computational intelligence and remote sensing image understanding.



Hao Liu received the B.S. degree in electronic information engineering from Xidian University, Xi'an, China, in 2021, where he is currently pursuing the M.S. degree in electronic information with the School of Electronic Engineering.

His research interests include domain adaptation, few-shot learning, and remote sensing image processing.

Maoguo Gong (Senior Member, IEEE) received the B.S. degree in electronic engineering and the Ph.D. degree in electronic science and technology from Xidian University, Xi'an, China, in 2003 and 2009, respectively.

Since 2006, he has been a Teacher with Xidian University, where he was promoted as an Associate Professor and a Full Professor, respectively, both with exceptive admission in 2008 and 2010. His research interests include computational intelligence with applications to optimization, learning, data mining, and image understanding.

Dr. Gong is also an Executive Committee Member of the Chinese Association for Artificial Intelligence and a Senior Member of the Chinese Computer Federation. He was a recipient of the prestigious National Program for the support of Top-Notch Young Professionals from the Central Organization Department of China, the Excellent Young Scientist Foundation from the National Natural Science Foundation of China, and the New Century Excellent Talent in University from the Ministry of Education of China. He is also the Vice-Chair for the IEEE Computational Intelligence Society Task Force on Memetic Computing.

Hao Li (Member, IEEE) received the B.S. degree in electronic engineering and the Ph.D. degree in pattern recognition and intelligent systems from Xidian University, Xi'an, China, in 2013 and 2018, respectively.

He is currently an Associate Professor with the School of Electronic Engineering, Xidian University. His research interests include computational intelligence and machine learning.

Dr. Li served as a member of the IEEE Neural Networks Technical Committee. He received the Excellent Doctoral Dissertation Award of Shaanxi Province in 2021.

Yue Wu (Member, IEEE) received the B.Eng. and Ph.D. degrees from Xidian University, Xi'an, China, in 2011 and 2016, respectively.

Since 2016, he has been a Teacher with Xidian University, where he is currently an Associate Professor. He has authored or coauthored more than 40 papers in refereed journals and proceedings. His research interests include computer vision and computational intelligence.

Xiangming Jiang (Member, IEEE) received the B.S. degree in mathematics and applied mathematics and the Ph.D. degree in pattern recognition and intelligent systems from Xidian University, Xi'an, China, in 2015 and 2020, respectively.

He is currently a Lecturer with the School of Electronic Engineering, Xidian University. His research interests include computational intelligence and ill-posed inverse problem in image understanding.

Dr. Jiang received the Excellent Doctoral Dissertation Award of Shaanxi Province in 2022.