Deep Cross-Domain Few-Shot Learning for Hyperspectral Image Classification

Zhaokui Li, Ming Liu, Yushi Chen, *Member, IEEE*, Yimin Xu, Wei Li, *Senior Member, IEEE*, and Qian Du, *Fellow, IEEE*

***Abstract*— One of the challenges in hyperspectral image (HSI) classification is that there are limited labeled samples to train a classifier for very high-dimensional data. In practical applications, we often encounter an HSI domain (called target domain) with very few labeled data, while another HSI domain (called source domain) may have enough labeled data. Classes between the two domains may not be the same. This article attempts to use source class data to help classify the target classes, including the same and new unseen classes. To address this classification paradigm, a meta-learning paradigm for few-shot learning (FSL) is usually adopted. However, existing FSL methods do not account for domain shift between source and target domain. To solve the FSL problem under domain shift, a novel deep cross-domain few-shot learning (DCFSL) method is proposed. For the first time, DCFSL tackles FSL and domain adaptation issues in a unified framework. Specifically, a conditional adversarial domain adaptation strategy is utilized to overcome domain shift, which can achieve domain distribution alignment. In addition, FSL is executed in source and target classes at the same time, which can not only discover transferable knowledge in the source classes but also learn a discriminative embedding model to the target classes. Experiments conducted on four public HSI data sets demonstrate that DCFSL outperforms the existing FSL methods and deep learning methods for HSI classification. Our source code is available at https://github.com/Li-ZK/DCFSL-2021.**

***Index Terms*— Domain adaptation, few-shot learning (FSL), hyperspectral image (HSI), meta-learning.**

I. Introduction

A HYPERSPECTRAL image (HSI) contains many consecutive bands. And there is a lot of spectral and spatial information in it [1]. Due to such abundant information, it has been applied in many fields [2]–[6], such as environmental monitoring, disaster prevention, and control. Hence, HSI analysis, such as classification, has gained more and more attention over the past few decades [7].

In early research, most traditional HSI classification methods focused on handcraft features extraction [8]–[11] and traditional classifiers, including k-nearest neighbor (KNN) [12], support vector machines (SVM) [13], logistic regression [14], and so on. These traditional classification methods can obtain good results, but the process of feature extraction and classification does not fully consider the adaptability between them.

For the past few years, deep learning methods have been applied to classify HSI and shown its effectiveness [15], [16]. Chen *et al.* [17] presented a deep feature extraction method based on a stacked autoencoder (SAE), which used a logistic regression method to perform the final classification. Then, Chen *et al.* [18] introduced a deep belief network (DBN) to extract features from HSI data in a hierarchical fashion and completed the final classification task by means of logistic regression. However, these two methods require many parameters for training in a network composed of fully connected (FC) layers.

In order to decrease the number of parameters, a convolutional neural network (CNN) with local connections and weight sharing is adopted [19], and many CNN-based methods have been designed for HSI classification [20]. Hu *et al.* [21] proposed a classification method based on 1-D CNN to obtain deep spectral features in HSI. Chen *et al.* [22] and Li *et al.* [23] designed a classification framework based on 3-D CNN to classify HSI accurately, respectively. In [24], a deep pixel-pair feature was proposed to enhance the original CNN, and a voting strategy was adopted to achieve the final classification.

Although the CNN model has achieved good classification results, a very deep CNN model inevitably faces the overfitting problem. The residual block introduced by the ResNet network [25] has replaced the traditional convolution block to alleviate the overfitting problem. Zhong *et al.* [26] proposed a supervised spectral–spatial residuals network (SSRN), in which the identity mapping in the residual block reduced the loss of accuracy. Inspired by ResNet, DenseNets [27] followed and extended the idea of ResNet. Wang *et al.* [28] proposed a fast dense spectral–spatial convolution network for HSI classification. Li *et al.* [29] proposed a novel deep multilayer fusion dense network for HSI classification.

Although great success has been made with deep learning in HSI classification, training a deep learning model usually requires sufficient labeled data. It is difficult to obtain sufficient labeled data because the collection of labeled data is

Manuscript received November 5, 2020; revised January 26, 2021; accepted January 31, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61971164 and Grant 61922013; in part by the Liaoning Provincial Natural Science Foundation under Grant 20180550337 and 2019-MS-254; and in part by the Beijing Natural Science Foundation under Grant JQ20021. *(Corresponding author: Zhaokui Li.)*

Zhaokui Li, Ming Liu, and Yimin Xu are with the School of Computer Science, Shenyang Aerospace University, Shenyang 110136, China (e-mail: lzk@sau.edu.cn; liuming18340837829@163.com; yimin.xu@duke.edu).

Yushi Chen is with the School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150001, China (e-mail: chenyushi@hit.edu.cn).

Wei Li is with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China (e-mail: liwei089@ieee.org).

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

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TGRS.2021.3057066.

Digital Object Identifier 10.1109/TGRS.2021.3057066

time-consuming and expensive. Some unsupervised learning methods [30], [31] have been applied to address the problem of limited labeled data. Liu *et al.* [30] proposed an unsupervised feature extraction method using a graph for HSI. Mei *et al.* [31] proposed an unsupervised spatial–spectral feature learning strategy for HSI using a 3-D convolutional autoencoder (3-D-CAE). Although an unsupervised mode does not need label information for feature learning, it still uses label information for classifier training.

In order to make better use of labeled and unlabeled data, semisupervised methods [32], [33] have been proposed. Liu *et al.* [33] proposed a new semisupervised CNN for HSI classification, which was trained to simultaneously minimize the sum of supervised and unsupervised cost functions. In addition, some active learning methods [34]–[36] have been applied to address the problem of limited labeled samples. Although unsupervised and semisupervised approaches can alleviate the problem of insufficient labeling data, these methods are all based on the assumption that data of the same class are sampled from the same data distribution.

In practical remote sensing applications, we often encounter an HSI (called target domain) with very few labeled data, while the other HSI (called source domain) has enough labeled data. Typically, the source and target domains have different data distributions. To tackle the cross-domain learning task, a variety of methods have been proposed, which can be roughly divided into two types, i.e., domain adaptation-based [37]–[41] and fine-tuning-based [42]–[44].

The former type mainly focuses on the feature-level domain adaptation, which intends to directly adjust the feature space of the source and target domains to reduce the discrepancy of their distribution. Although the domain adaptation method can reduce the data shift between two domains, it generally assumes the target domain classes to be consistent with the source domain classes, so the new unseen classes cannot be classified.

The latter type transfers the weights of the first layers of the pretrained network to the weights of the target model, and then fine-tunes the top layers of the pretrained network by the backpropagation algorithm. Mei *et al.* [43] first trained a five-layer CNN for classification (C-CNN) in the source domain, and then constructed a companion feature-learning CNN (FL-CNN) by extracting FC feature layers in this C-CNN. Yang *et al.* [44] proposed a two-branch convolutional neural network (Two-CNN) to extract the joint spectral–spatial features from HSI. They also adopted a fine-tuning strategy to improve the performance of limited labeled samples. This type of method does not require the target domain classes to be consistent with the source domain classes, so it can classify the unseen classes in the target domain. However, when data shift between the two domains is severe, fine-tuning cannot handle it. In addition, in order to obtain better classification performance, more target domain labeled data are needed.

The high cost of manual annotation greatly limits the applicability of current learning model to effectively learn new knowledge. In contrast, the human visual system can recognize new classes through rarely labeled examples. Learning how to generalize to unseen classes during training, the so-called few-shot classification has recently attracted attention [45], [46].

In the few-shot classification, classes between the source and target domains are not required to be the same, so few-shot learning (FSL) can classify new unseen classes in the target domain. An FSL approach is essentially a meta-learning approach [47], where general information (meta-knowledge) learned from the source class data can help to make predictions on the target class data when only one or few labeled target class data are available. Specifically, given a K-shot C-way classification task, its purpose is to use K-labeled training data of each class to learn a C-class classification model [48]. The K here is usually a small number (such as 1 or 5).

To acquire meta-knowledge that is tailored for performing few-shot tasks, this method uses labeled source class data to simulate few-shot tasks by episodes [48], [49]. After training by executing the episodic paradigm, the meta-knowledge is stored in a model. Then, the model is used to perform a new few-shot classification task. The few labeled target class data and acquired meta-knowledge are intended to predict the labels of all unlabeled target class data.

Recently, FSL has been used for HSI classification. A deep few-shot learning (DFSL) method is proposed by Liu *et al.* [50] to classify HSI with few labeled data. DFSL learns a metric space via a deep residual network based on 3-D CNN from four source class data sets. In the test phase, DFSL adopts SVM and nearest neighbor (NN) method to classify unlabeled data of test data set. A relation network [51] model for HSI few-shot classification (RN-FSC) is designed by Gao *et al.* [52] to classify HSI, where few labeled target class data are used to fine-tune the model.

Though the above-mentioned FSL methods can improve classification accuracy with few labeled test data, they generally assume that both source class and target class data come from domains with the same distribution. However, in practice, both source class and target class data could come from domains with severe domain shift. This means that the model trained from the source domain needs to deal with new classes and new domain (target domain), with few data from the target domain. Moreover, the information about the labeled target class data is not used efficiently. In this article, a new deep cross-domain few-shot learning (DCFSL) method is proposed to solve the above issues. First, two mapping layers and a spectral–spatial embedded feature extraction network are used to learn the spatial–spectral metric space. In this metric space, inner-class distance is close and interclass distance is far away. Then FSL is executed by comparing the similarity or distance between labeled and unlabeled data. Both mapping layer and embedded feature extraction network are trained with the idea of meta-learning in a unified framework. In order to overcome domain shift, a conditional adversarial domain adaptation strategy is adopted to achieve domain distribution alignment. In addition, FSL is executed in the source classes and target classes at the same time, which can not only discover transferable meta-knowledge in the source classes but also learn a discriminative embedding space to the target classes. To demonstrate the effectiveness of DCFSL, we con-

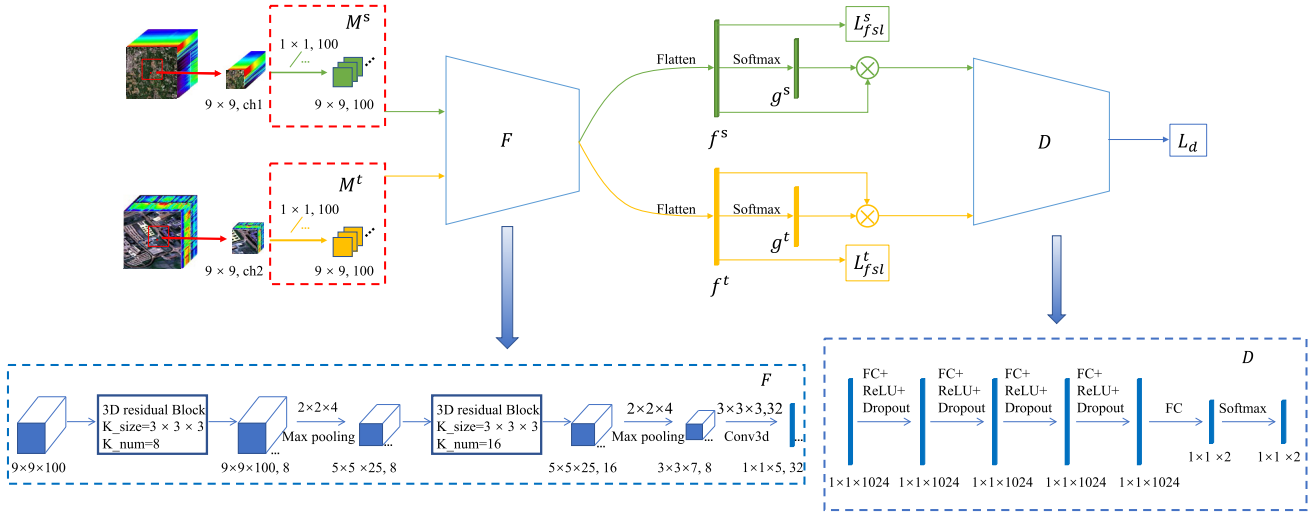


Fig. 1. Framework of proposed DCFSL in the training phase.

duct sufficient experiments on four public HSI data sets. The results demonstrate that the DCFSL method outperforms the existing FSL methods and deep learning methods.

The main contributions of this article can be summarized as follows.

1) As far as we know, FSL and domain adaptation issues are addressed in a unified framework for the first time to realize HSI cross-domain few-shot classification, which can learn a classification model that is less likely to be affected by domain shift.
2) A conditional adversarial domain adaptation strategy is adopted to learn a domain-adaptive feature embedded space, which can achieve domain distribution alignment.
3) By employing FSL in both source classes and target classes at the same time, DCFSL can not only discover transferable knowledge in the source classes but also learn the discriminative embedding model to the target classes.

The remainder of this article is organized as follows. Section II introduces relevant concepts of the cross-domain FSL. Section III introduces the DCFSL method in detail. The experimental results on four public HSI data sets are shown in Section IV. Finally, a conclusion is drawn in Section V.

II. Definition of Cross-Domain FSL

To better explain the proposed DCFSL for HSI classification, several relevant concepts are introduced in the rest of this section. Under the cross-domain FSL setting, two domain data sets are given: source domain data set D_S with C_s classes and target domain data set D_T with C_t classes, where C_s and C_t represent the number of classes in the source and target domain, respectively. In general, in order to ensure the diversity of training samples, C_s should be larger than C_t, which is constructive to meta-learning [50], [52]. According to whether the data are labeled, target domain data set can further split into two parts: few-shot data set D_f with labeled data and test data set D_t with unlabeled data, and $D_f \cup D_t = D_T$. The few-shot data set gets its name since it is relatively small compared with the testing data set.

In our DCFSL method, the source domain data set and the few-shot data set are utilized to train an embedded feature extractor, and then its generalization ability is evaluated on the testing data set. To learn an appropriate embedded space for an HSI FSL task, we cast sampled minibatches called episodes to mimic the few-shot task [48], [49]. The episodic training procedure is based on a simple machine learning principle: testing and training conditions must match [49]. To simulate the FSL task, samples from both D_S and D_f are selected to form source episodic training tasks and target episodic training tasks. Actually, one task is a training iteration. During each iteration, take source episodic training tasks as example, first, C classes are randomly selected from the source domain data set, and then K labeled samples per class are sampled to form a support set. Thus, the support set is denoted as $S = \{(x_i, y_i)\}_{i=1}^{C\times K}$. Similarly, N unlabeled samples, which are different from samples in the support set, are randomly sampled from the same C classes to form a query set. Thus, the query set can be denoted as $Q = \{(x_j, y_j)\}_{j=1}^{C\times N}$. Target episodic training tasks are like source episodic training tasks. Note that the samples in D_f are scarce and cannot generate a training episode, so a data augment is employed by adding random Gaussian noise to the current known samples, which is widely used in [22] and [53] to increase training samples.

III. Proposed DCFSL Framework

As shown in Fig. 1, DCFSL is equipped with two mapping layers, one embedded feature extractor, and one domain discriminator. Two mapping layers (i.e., M^s for source domain and M^t for target domain) are employed to ensure the input of the embedded feature extractor has the same dimension. The embedded feature extractor, denoted as F, attempts to map each sample from either domain into a domain-invariant metric space. The domain discriminator, denoted as D, achieves domain distribution alignment.

A. DCFSL Framework

The framework of the proposed DCFSL in the training phase is shown in Fig. 1. Specifically, the DCFSL mainly consists of *two kinds of* FSL. One is the FSL for the source domain. The other is the FSL for few-shot data set in the target domain. These two kinds of FSL are carried out alternately. For each

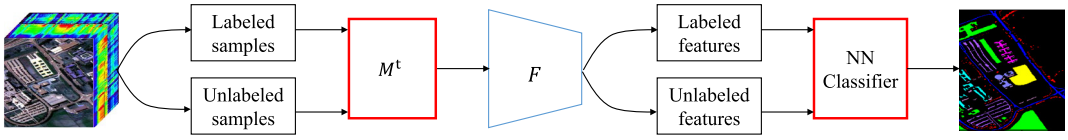


Fig. 2. Flowchart of classification in the testing phase.

kind of FSL, there are mainly four parts. First, two mapping layers are used to ensure the same input dimension between the source and target domain. Second, an embedded feature extractor is implemented to embed both source and target HSI cubes into the spatial–spectral embedded space, where samples with the same classes are mapped as close as possible and samples with different classes are mapped as far as possible. Third, we can obtain the spatial–spectral embedded features of source domain data and perform source domain FSL by computing distances between unlabeled and labeled samples of each class in the spatial–spectral embedded space. Similarly, we can obtain embedded features of target domain and carry out target domain FSL. Finally, a conditional domain discriminator is utilized to mitigate domain shift between two domains to make extracted spatial–spectral embedded features domain invariant and achieve domain distribution alignment.

Therefore, the total loss function of source domain FSL with domain adaptation is given by

$$L^s_{\text{total}} = L^s_{\text{fsl}} + L_d. \tag{1}$$

Similarly, the total loss function of target domain FSL with domain adaptation is given by

$$L^t_{\text{total}} = L^t_{\text{fsl}} + L_d. \tag{2}$$

Finally, the total loss function for the DCFSL is given by

$$L_{\text{total}} = L^s_{\text{total}} + L^t_{\text{total}}. \tag{3}$$

Several modules in the framework are designed for FSL and domain adaptation. They will be introduced in detail in the next several sections.

In addition, in the testing phase, the classification process of the unlabeled samples is shown in Fig. 2. It includes four steps. First, the target mapping layer is used to reduce the dimensions of input samples. Second, spectral–spatial embedded features are extracted via trained deep residual 3-D CNN. Third, unlabeled samples are classified via a NN classifier. It is noteworthy that the entire few-shot data set is considered as labeled samples to fit a simple NN classifier in the testing phase. After training, the deep 3-D embedded feature extractor can produce discriminative features with low intraclass and high interclass variability. Therefore, the unlabeled samples of the target domain can be classified by the NN classifier. Finally, the classification maps of all unlabeled samples are produced to assess the performance of DCFSL.

B. Mapping Layer and Embedded Feature Extractor

The spatial–spectral embedded feature extractor is to extract discriminative spatial–spectral embedded features of the input samples. As DFSL [50] method utilizes a deep residual 3-D CNN network for embedded feature learning module, the same architecture setting is used as spatial–spectral embedded feature extractor network for fair comparison. In addition, the mapping layer, first proposed in [54], is adopted to ensure the same dimension of input samples before the embedded feature extractor network. The detailed network of mapping layer and embedded feature extractor is introduced as follows.

First, two mapping layers, M^s and M^t, are used to make the input dimension between the source domain (e.g., 128 bands of Chikusei data set) and the target domain (e.g., 103 bands of University of Pavia data set) equal before feature extraction. The mapping layer is implemented via 2-D CNN. Let $I \in R^{9\times9\times ch}$ be the input HSI cube, where 9×9 is the spatial dimensions and ch is the number of bands. The feature output of mapping layer is

$$I^{'} = I \times T \tag{4}$$

where $I^{'} \in R^{9\times9\times100}$ is the transformed data set, and $T \in R^{ch\times100}$. There are $ch\times100$ learnable parameters in the transformation. For M^s, there are 128×100 parameters. For M^t, there are 103×100 parameters.

Then a deep residual network based on 3-D CNN is employed to extract embedded features. The output of mapping layers is the input of the embedded feature extractor. The network of embedded feature extractor is shown in the lower left part of Fig. 1. The embedded features extractor network consists of two residual blocks, two max pooling layers, and a convolutional layer. After that, a metric learning method is used to perform the FSL for HSI classification.

C. Source and Target FSL

In the proposed DCFSL method, FSL is executed in source and target classes simultaneously, which can not only discover transferable knowledge in the source classes but also learn a discriminative embedding model to the target classes. After mapping layers and the embedded feature extraction network, the spatial–spectral embedded features of source domain samples can be obtained in the spatial–spectral metric space, and then source domain FSL is executed by computing distances between unlabeled and labeled samples of each class. Similarly, the embedded features of the target domain can be obtained and target domain FSL can be executed. Specifically, the FSL method is shown in Fig. 3. Take the source domain FSL as an example. First, C classes from C_s source classes are randomly selected to form an episode, namely a classification task. Then K samples from selected classes are selected as support set $S_s = \{(x_i, y_i)\}_{i=1}^{C\times K}$ and N samples from the same classes are selected as query set $Q_s = \{(x_j, y_j)\}_{j=1}^{C\times N}$. It is

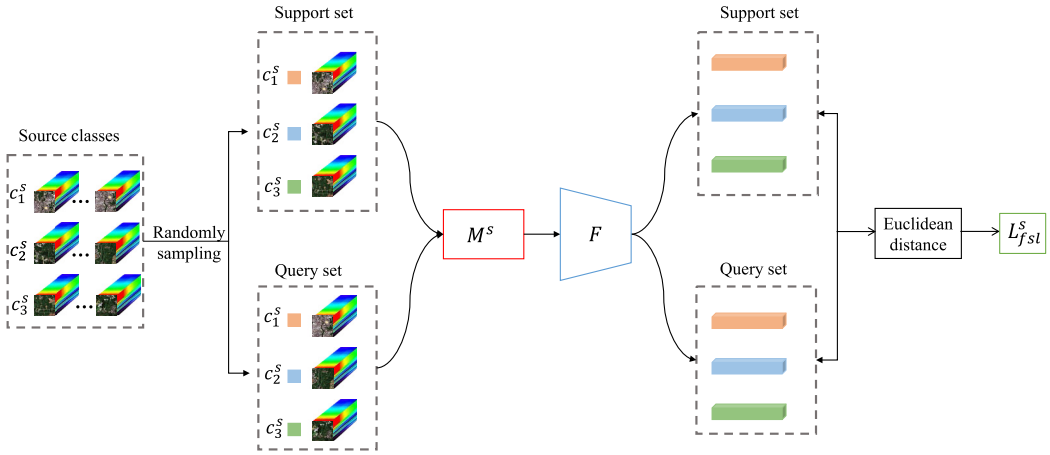


Fig. 3. FSL for source domain samples. 3-way, 1-shot as an example.

noteworthy that the samples in the support set are different from the samples in the query set. Samples from support and query set are first fed through the mapping layer to reduce dimensions and then fed through an embedded feature extractor network to extract the embedded features. During the training phase, a nonparametric softmax based on the distance metric results is used to update network parameters. The class distribution for a query sample x_j in the query set Q_s is

$$P(y_j = k|x_j \in Q_s) = \frac{\exp(-d(f_\varphi(x_j), c_k))}{\sum_{k=1}^{C} \exp(-d(f_\varphi(x_j), c_k))} \tag{5}$$

where $d(,)$ denotes a Euclidean distance function, c_k is the embedded features of the kth class in the support set, C is the number of unique classes in each episode, f_φ denotes the mapping layer and embedded feature extractor network with parameter φ, x_j is the sample of the query set, and y_j is the label of x_j.

According to the negative log-probability of the query sample x and its true class label k, the classification loss for all query samples in a source episode can be formulated as

$$L^s_{\text{fsl}} = E_{S_s, Q_s}\left[-\sum_{(x,y)\in Q_s} \log p_\varphi(y = k|x)\right] \tag{6}$$

where S_s and Q_s are the support and query set from source domain data set, respectively.

Similarly, the classification loss over each target episode can be formulated as

$$L^t_{\text{fsl}} = E_{S_t, Q_t}\left[-\sum_{(x,y)\in Q_t} \log p_\varphi(y = k|x)\right] \tag{7}$$

where S_t and Q_t are the support and query set of few-shot data set from the target domain, respectively.

D. Conditional Domain Discriminator

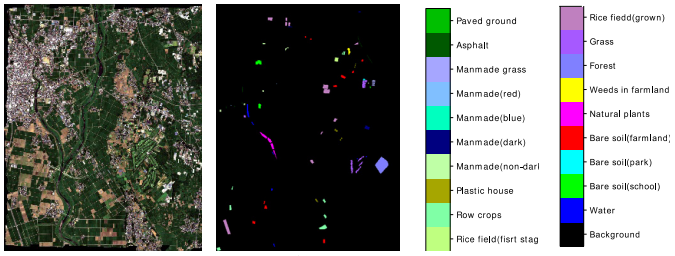
In order to mitigate domain shift, a conditional adversarial domain adaptation strategy is employed to align the global data distribution for source and target domain. Inspired by the Conditional Domain Adversarial Network (CDAN) [55], a conditional domain discriminator D on source distribution $P_s(x)$ and target distribution $P_t(x)$ is explored to reduce domain shift. The classifier prediction g contains discriminative information, which can be conditioned on adversarial adaptation of feature representation f. By conditioning, domain variances in both f and g can be modeled at the same time. The domain adversarial loss function L is defined on domain discriminator D, the feature representation $f = F(x)$ and the classifier prediction $g = G(x)$.

$$\min_D \max_{F,G} L = -E_{x_i^s \sim P_s(x)} \log\left[D\left(f_i^s, g_i^s\right)\right] - E_{x_j^t \sim P_t(x)} \log\left[1 - D\left(f_j^t, g_j^t\right)\right] \tag{8}$$

where $D(,)$ is the probability that the discriminator predicts that x is the source domain sample, $1 - D(,)$ is the probability that the discriminator predicts that x is the target domain sample. Discriminator D minimizes the above function, while feature extractor F and classifier G maximize it.

Let $h = (f, g)$ be the joint variable of f and g. The multilinear map $f \otimes g$ (defined as the outer product of multiple random vectors) is chosen to condition D on g. Compared with the concatenation strategy, the multilinear map $f \otimes g$ can fully capture the multimodal structures behind complex data distributions. However, a drawback of multilinear map is dimension explosion. Let d_f and d_g be the dimensions of f and g, respectively. The output dimension of multilinear map is $d_f \times d_g$. The dimensional is usually too high for deep embedded space. To solve this dimension explosion issue, DCFSL replaces multilinear map with randomized multilinear map. The multilinear map $T_\otimes(f, g)$ can be approximated by the dot-product $T_\odot(f, g) = (1/\sqrt{d})(R_f f) \odot (R_g g)$, where $\odot$ denotes the element-wise product, $R_f \in R^{d \times d_f}$ and $R_g \in R^{d \times d_g}$ denote two random matrices, which are sampled only once and fixed in the training phase, and $d \ll d_f \times d_g$. Each element in R_f or R_g follows a symmetric distribution with univariance (e.g., the uniform distribution and Gaussian distribution). Finally, the following conditioning strategy is adopted:

$$T(h) = \begin{cases} T_\otimes(f, g), & d_f \times d_g \le 1024 \\ T_\odot(f, g), & \text{otherwise} \end{cases} \tag{9}$$



(a) (b)

Fig. 4. Chikusei data set. (a) False-color image. (b) Ground-truth map.

where 1024 is the largest number of units in the deep network, and if the dimension of the multilinear map $T_{\otimes}$ is larger than 1024, we will use randomized multilinear map $T_{\odot}$. The loss for learning domain-invariant embedded features is formulated as

$$L_d = \min_D \max_T L = -E_{x_i^s \sim P_s(x)} \log\left[D\left(T(h_i^s)\right)\right] - E_{x_j^t \sim P_t(x)} \log\left[1 - D\left(T(h_j^t)\right)\right]. \quad (10)$$

The network of domain discriminator is shown in the lower right part of Fig. 1. In this work, we consider a multilayer perceptron stacked after the multilinear map. A multilayer perceptron consists of five fully connected layers. A rectified linear unit (ReLU) nonlinearity activation function and dropout layers are added after each fully connected layer except the last layer. The final activation function is softmax, which is adopted to predict whether the input is from source or target domain data.

IV. Experiments

Experiments are conducted on a workstation with an Intel Xeon processor (2.67 GHz), 64 GB of memory, and Nvidia TITAN RTX 24 GB graphics card. The training and testing experiments were implemented by using the open-source software framework Pytorch.

A. Experimental Data Set

To assess the performance of DCFSL, five public HSI data sets were used, including the Chikusei, the University of Pavia (UP), the Pavia Center (PC), the Salinas, and the Indian Pines (IP). Because the proposed DCFSL is a type of transfer learning, the Chikuse is selected as the source domain data set, and the UP, the PC, the Salinas, and the IP data set are selected as target domain data sets.

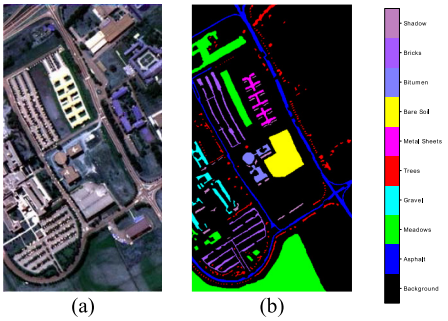
1) Source Domain Data Set: The Chikusei data set, provided in [56], was collected by Hyperspectral Visible/Near-Infrared Cameras (Hyperspec-VNIR-C) in Chikusei, Ibaraki, Japan, on July 29, 2014. It contains 19 classes and has 2517 × 2335 pixels. Its spatial resolution is 2.5 m per pixel. It consists of 128 spectral bands, which range from 363 to 1018 nm. The false-color image and corresponding ground-truth map can be seen in Fig. 4. Table I shows the samples of the Chikusei data set.

2) Target Domain Data Set: The UP data set was gathered by the Reflective Optics Spectrographic Image System (ROSIS). It contains 610 × 340 pixels. And its spatial resolution is 1.3 m per pixel. It has 103 spectral bands, which range from 430 to 860 nm. There are nine categories representing different types of land cover. The false-color image and corresponding ground-truth map can be seen in Fig. 5. Table II shows the samples of the UP data set.

The PC data set was captured by the ROSIS sensor during a flight campaign over Pavia, Northern Italy. It consists of 1906 × 715 pixels. Its spatial resolution is 1.3 m per pixel. It has 102 bands and ranges from 430 to 860 nm. The ground truth contains nine different classes representing

TABLE I
Land Cover Classes and Numbers of Samples in the Chikusei Data Set

Class	Name	Pixels
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	77,592



(a) (b)

Fig. 5. UP data set. (a) False-color image. (b) Ground-truth map.

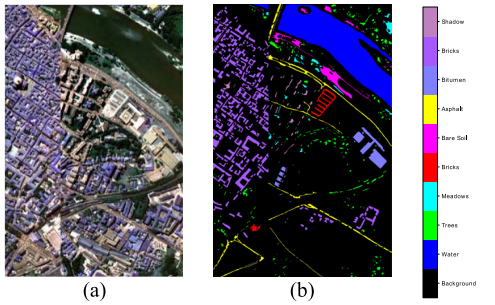
TABLE II
Land Cover Classes and the Numbers of Samples in the UP Data Set

Class	Name	Pixels
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Sheets	1345
6	Bare soil	5029
7	Bitumen	1330
8	Bricks	3682
9	Shadow	947
	Total	42,776

TABLE III

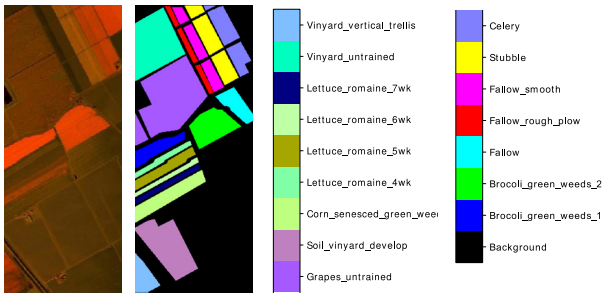
LAND COVER CLASSES AND THE NUMBERS OF SAMPLES IN THE PC DATA SET

Class	NAME	Pixels
1	Water	65971
2	Trees	7598
3	Meadows	3090
4	Bricks	2685
5	Bare Soil	6584
6	Asphalt	9248
7	Bitumen	7287
8	Bricks	42826
9	Shadow	2863
	Total	148,152



(a) (b)

Fig. 6. PC data set. (a) False-color image. (b) Ground-truth map.



(a) (b)

Fig. 7. Salinas data set. (a) False-color image. (b) Ground-truth map.

a typical urban site. The false-color image and corresponding ground-truth map can be seen in Fig. 6. Table III shows the samples of the PC data set.

The Salinas data set contains 512 × 217 pixels. Its spatial resolution is 3.7 m per pixel. It has 204 spectral bands that range from 400 to 2500 nm. It was collected by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor over Salinas Valley, CA, USA. It contains 16 classes. The false-color image and corresponding ground-truth map can be seen in Fig. 7. Table IV shows the samples of the Salinas data set.

The IP data set was also gathered by AVIRIS in 1992 from Northwest Indiana. It consists of 200 bands and ranges from 400 to 2500 nm. It has 145 × 145 pixels. Its spatial resolution is 20 m per pixel. It includes 16 vegetation classes. The false-color image and corresponding ground-truth map can be seen in Fig. 8. Table V shows the samples of the IP data set.

B. Experimental Setting

To verify the effectiveness of our method, we compared the DCFSL method with six supervised methods

TABLE IV

LAND COVER CLASSES AND THE NUMBERS OF SAMPLES IN THE SALINAS DATA SET

Class	NAME	Pixels
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_develop	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	54,129

TABLE V

LAND COVER CLASSES AND THE NUMBERS OF SAMPLES IN THE IP DATA SET

Class	NAME	Pixels
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-tree	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	10,249

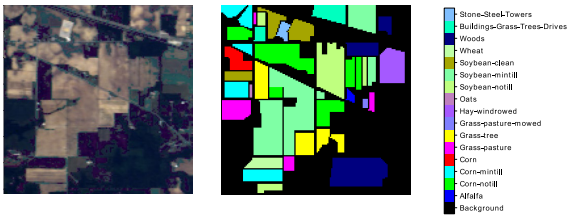


Fig. 8. IP data set. (a) False-color image. (b) Ground-truth map.

(SVM, 3-D-CNN [23], SSRN [26], DFSL+NN [50], DFSL+SVM [50], and RN-FSC [52]), four semisupervised methods (Laplacian SVM (LapSVM) [32], semi-supervised CNN (SS-CNN) [33], sFL-CNN [43], and Two-CNN-transfer [44]), and three unsupervised methods (collaboration-competition preserving graph embedding (CCPGE) [30], 3-D-CAE [31], and uFL-CNN [43]).

In order to ensure the fairness of the experiment, in all comparing experiments, we select 1–5 labeled target domain samples per class for training. For some cross-domain methods, we also randomly select 200 labeled source domain

TABLE VI
PARAMETERS OF MAPPING LAYER AND EMBEDDED FEATURE EXTRACTOR

Layer Name	OUTPUT SHAPE	Filter Size	Padding
Input	$9 \times 9 \times ch$	N/A	N
Mapping	$9 \times 9 \times 100$	$1 \times 1 \times 100$	N
Convolution 1	$9 \times 9 \times 100 \times 8$	$3 \times 3 \times 3 \times 8$	Y
Convolution 2	$9 \times 9 \times 100 \times 8$	$3 \times 3 \times 3 \times 8$	Y
Convolution 3	$9 \times 9 \times 100 \times 8$	$3 \times 3 \times 3 \times 8$	Y
Shortcut	Convolution 1 + Convolution 3		
Max Polling 1	$5 \times 5 \times 25 \times 16$	$2 \times 2 \times 4$	N
Convolution 4	$5 \times 5 \times 25 \times 16$	$3 \times 3 \times 3 \times 16$	Y
Convolution 5	$5 \times 5 \times 25 \times 16$	$3 \times 3 \times 3 \times 16$	Y
Convolution 6	$5 \times 5 \times 25 \times 16$	$3 \times 3 \times 3 \times 16$	Y
Shortcut	Convolution 4 + Convolution 6		
Max Pooling 2	$3 \times 3 \times 7 \times 16$	$2 \times 2 \times 4$	N
Convolution 7	$1 \times 1 \times 5 \times 32$	$3 \times 3 \times 3 \times 32$	N
Flatten (Output)	160	N/A	N

TABLE VII
CLASSIFICATION RESULTS (%) ON THE UP DATA SET (FIVE LABELED SAMPLES PER CLASS)

Class	SVM	3-D-CNN	SSRN	DFSL+NN	DFSL+SVM	RN-FSC	DCFSL
1	88.98	59.82	**91.84**	69.19	73.43	68.55	82.20
2	83.91	63.05	**95.13**	84.63	89.25	93.44	87.74
3	39.98	**68.91**	55.23	57.47	48.09	49.81	67.46
4	60.22	77.31	78.02	89.99	84.72	92.15	**93.16**
5	95.44	90.77	98.34	**100**	99.65	99.43	99.49
6	37.12	63.40	53.56	71.23	67.81	57.99	**77.32**
7	40.62	**87.64**	60.07	70.62	64.48	70.04	81.18
8	68.17	57.27	**85.34**	58.13	67.37	63.48	66.73
9	99.13	95.57	98.08	96.92	92.92	**99.19**	98.66
OA	64.12 ±4.55	65.74 ±1.77	76.26 ±5.78	77.75 ±1.16	79.63 ±1.09	80.19 ±2.18	**83.65 ±1.77**
AA	68.18 ±2.27	73.72 ±1.01	79.51 ±3.21	77.57 ±0.31	76.41 ±1.39	77.12 ±0.84	**83.77 ±1.74**
Kappa	55.59 ±4.64	57.37 ±1.97	70.56 ±6.69	71.11 ±1.22	73.05 ±1.60	73.73 ±2.79	**78.70 ±2.01**

samples per class to learn transferable knowledge, such as DFSL+NN [50], DFSL+SVM [50], RN-FSC [52], Two-CNN-transfer [44], sFL-CNN [43], and uFL-CNN [43]. The rest samples of the target domain data set are reserved as the testing set.

In DCFSL, 9×9 neighborhoods are selected as the spatial size of the input cubes with reference to [33], [50]. The detailed parameters of the mapping layer and the embedded features extractor network are shown in Table VI. The DCFSL is trained via Adam optimizer and all convolution kernels are initialized with Xavier normalization. The number of the training iterations is set to 10 000, which is sufficient to train the network. The learning rate is 0.001. We use C to denote the class number of each episode, and K to denote the sample number per class in the support set. Each episode is a K-shot, C-way classification task. C is set to the class number of the target data set (e.g., the UP and PC data sets are 9, and the Salinas and IP data sets are 16). For the source FSL and target FSL, K is set to 1, that is, only one labeled sample is randomly selected to participate in the model training in each episode. The sample number per class in the query set is denoted N_Q. In theory, the higher the N_Q value, the better it will be to simulate a small sample of classes in the test data set, so N_Q is set to 19 [50], [52].

The classification performance of all methods is assessed with overall accuracy (OA), average accuracy (AA), and kappa coefficients.

C. Compared With Supervised Methods

To further demonstrate the effectiveness of the DCFSL method in HSI classification with few labeled samples, the DCFSL method is compared with several different supervised methods. These methods include standard SVM, 3-D-CNN [23], SSRN [26], and existing FSL methods DFSL+NN [50], DFSL+SVM [50], and deep relation network RN-FSC [52].

For supervised methods (SVM, 3-D-CNN, and SSRN method), only the few-shot data set from the target domain can be utilized to train a classifier. Because training classes need to be the same as testing classes in these methods. Specifically, SVM utilizes the kernel method to map nonlinear data to high-dimensional feature space, which is linearly separable. It is noteworthy that the standard SVM method only explores spectral information in HSI and ignores spatial information. The 3-D-CNN method can extract the deep spectral–spatial-combined features effectively to classify HSI accurately. SSRN consists of sequential spectral and spatial residual blocks. Spectral–spatial embedded features in HSI can be extracted by these residual blocks. It should be noted that for SVM, 3-D-CNN, and SSRN method, 5 labeled samples for each class in the target domain were randomly selected to build the training set and the rest of target domain samples were selected as the testing set.

However, for FSL methods (DFSL+NN, DFSL+SVM, and RN-FSC), source domain samples can be utilized to learn transferable knowledge. In these methods, the classes between the two domains may be different. Specifically, DFSL+NN and DFSL+SVM methods learn a metric space via deep residual 3-D CNN. In this metric space, classification with few labeled samples can be performed by a NN or SVM classifier. RN-FSC method can utilize the spatial–spectral features in HSI to learn relation by comparing the distance between samples. For DFSL+NN, DFSL+SVM, and RN-FSC method, we also collected four available HSI data sets, Chikusei, Houston, Botswana, and Kennedy Space Center (KSC) to form source classes. There are 61 classes in total. The classes that are fewer than 200 samples were discarded. Thus, 40 classes were available to build the source classes. 100 bands in each HSI data set were selected by the graph representation based band selection (GRBS) [57] method to ensure the same uniformity of input dimension, like [50], [52].

Tables VII–X report the OA, AA, kappa coefficient, and the classification accuracy of each class for HSI classification. For all classification methods, 5 labeled samples for each class are

TABLE VIII
CLASSIFICATION RESULTS (%) ON THE PC DATA SET
(FIVE LABELED SAMPLES PER CLASS)

Class	SVM	3-D-CNN	SSRN	DFSL+NN	DFSL+SVM	RN-FSC	DCFSL
1	99.17	98.31	**99.98**	99.92	99.10	99.64	99.77
2	80.08	82.29	**94.81**	91.02	89.93	94.48	90.55
3	80.11	86.63	85.64	87.20	93.03	**94.10**	94.06
4	76.09	78.86	68.50	84.71	83.73	**93.58**	91.42
5	78.87	87.01	**96.04**	82.37	80.83	84.25	91.13
6	92.16	83.60	82.61	95.53	90.83	89.83	**96.70**
7	68.72	82.16	**91.48**	83.40	83.25	81.44	85.01
8	91.06	96.12	**99.91**	99.08	97.71	97.43	98.89
9	**99.92**	99.56	98.73	98.64	95.73	98.53	98.90
OA	92.21 ±2.37	94.07 ±1.15	96.62 ±0.70	96.79 ±0.86	95.65 ±1.04	96.30 ±0.68	**97.45 ±0.41**
AA	85.13 ±4.86	88.28 ±3.63	90.86 ±1.29	91.32 ±2.79	90.46 ±2.52	92.59 ±1.11	**94.05 ±0.78**
Kappa	89.09 ±3.27	91.66 ±1.60	95.24 ±0.98	95.46 ±1.22	93.87 ±1.46	94.78 ±0.95	**96.40 ±0.58**

TABLE IX
CLASSIFICATION RESULTS (%) ON THE SALINAS DATA SET
(FIVE LABELED SAMPLES PER CLASS)

Class	SVM	3-D-CNN	SSRN	DFSL+NN	DFSL+SVM	RN-FSC	DCFSL
1	97.57	95.29	97.55	95.63	73.92	96.47	**99.40**
2	87.43	97.20	98.97	99.09	96.85	99.47	**99.76**
3	82.95	91.45	92.47	94.01	**96.28**	85.05	91.96
4	99.11	97.31	96.50	99.54	99.11	98.75	**99.55**
5	**94.29**	91.24	94.20	90.58	80.72	83.45	92.70
6	98.36	98.8	99.28	98.47	91.63	96.73	**99.52**
7	94.39	99.69	**99.98**	99.81	97.73	99.61	98.88
8	59.99	66.40	86.90	77.74	**82.33**	72.11	74.57
9	96.09	96.25	**99.64**	91.13	94.44	88.35	99.59
10	71.45	70.72	**92.01**	60.98	80.96	70.53	86.42
11	91.25	93.15	95.86	95.99	93.38	90.03	**96.61**
12	97.22	99.65	99.15	93.13	97.94	93.15	**99.93**
13	97.30	92.63	89.24	**99.34**	95.79	98.54	99.30
14	91.84	93.56	95.15	98.06	**98.87**	96.43	98.85
15	60.52	68.02	55.97	**77.54**	71.13	70.18	75.38
16	81.45	81.41	**98.91**	85.05	90.57	82.39	92.22
OA	80.71 ±2.75	84.20 ±2.62	86.39 ±2.68	87.05 ±0.83	86.95 ±1.30	84.11 ±1.36	**89.34 ±2.19**
AA	87.58 ±1.84	89.56 ±1.79	93.24 ±1.29	91.01 ±0.66	90.08 ±1.44	88.83 ±2.07	**94.04 ±1.14**
Kappa	78.61 ±3.0	82.46 ±2.90	84.95 ±2.90	85.63 ±0.91	85.51 ±1.42	82.38 ±1.53	**88.17 ±2.40**

randomly selected as supervised samples in the target domain. All experiments were carried out ten times to eliminate the influence of random sampling. From Tables VII–X, we can make four observations.

TABLE X
CLASSIFICATION RESULTS (%) ON THE IP DATA SET
(FIVE LABELED SAMPLES PER CLASS)

Class	SVM	3-D-CNN	SSRN	DFSL+NN	DFSL+SVM	RN-FSC	DCFSL
1	72.20	95.12	18.38	**96.75**	**96.75**	96.34	95.37
2	34.27	37.70	**64.79**	38.65	36.38	46.13	43.26
3	39.18	19.77	27.65	42.79	38.34	40.61	**57.95**
4	50.34	32.51	26.97	68.10	77.16	58.62	**80.60**
5	69.75	**88.45**	80.76	71.20	73.92	64.96	72.91
6	66.36	73.65	86.87	76.18	86.25	69.45	**87.96**
7	89.13	81.82	32.24	**100**	97.10	**100**	99.57
8	68.73	53.35	**100**	74.84	81.82	77.70	86.26
9	86.67	**100**	57.69	**100**	75.56	**100**	99.33
10	37.49	41.35	59.69	47.98	52.22	25.49	**62.44**
11	33.96	66.71	**70.87**	57.95	59.96	65.51	62.75
12	31.43	37.40	45.00	38.21	36.56	27.13	**48.72**
13	86.50	85.71	88.29	97.50	98.00	**99.75**	99.35
14	62.93	62.57	**97.18**	83.44	84.63	76.35	85.40
15	28.08	56.42	36.64	62.29	**74.10**	70.34	66.69
16	90.91	90.36	60.98	**100**	**100**	**100**	97.61
OA	45.85 ±2.44	54.76 ±0.03	61.36 ±0.49	59.65 ±0.63	61.69 ±1.85	58.17 ±0.02	**66.81 ±2.37**
AA	59.24 ±1.36	63.93 ±0.02	59.75 ±0.20	72.24 ±0.42	73.05 ±0.84	69.90 ±0.40	**77.89 ±0.86**
Kappa	39.68 ±2.48	48.72 ±0.03	56.91 ±0.48	54.55 ±0.52	56.78 ±1.90	52.52 ±0.14	**62.64 ±0.86**

1) The classification accuracy of the deep learning methods (3-D-CNN and SSRN method) is always superior to the standard SVM method. Deep learning methods can obtain discriminative features by building a hierarchical structure network, so they have better classification performance. For instance, the OA of SSRN is 12.14% higher than that of SVM on the UP data set.
2) By comparing FSL methods and deep learning methods, we can observe that the classification results of the FSL methods are better than deep learning methods with few labeled samples. Because a meta-learning strategy used in these FSL methods can discover transferable meta-knowledge from the source class data, which can help to make predictions on the target class data.
3) Compared with SVM, deep learning, and FSL methods, DCFSL generally has the best classification results in all target domain data sets. DCFSL can extract discriminative and domain invariance spatial–spectral features for the target domain data set, which can effectively address the problem of few labeled samples.
4) For the classes misclassified by other methods, DCFSL can obtain more accurate results, such as class 4 (Trees), class 6 (Bare soil) in the UP data set, class 3 (Meadows) in the PC data set, class 1 (Brocoli_green_weeds_1), class 11 (Lettuce_romaine_4wk) in the Salinas data set,

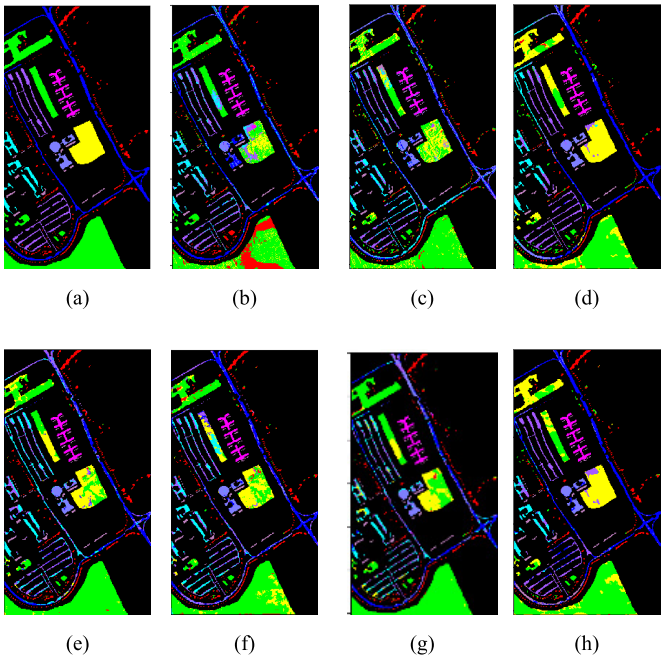


Fig. 9. UP. (a) Ground-truth. (b)–(h) Classification maps for different classifiers. (b) SVM. (c) 3-D-CNN. (d) SSRN. (e) DFSL+NN. (f) DFSL+SVM. (g) RN-FSC. (h) DCFSL.

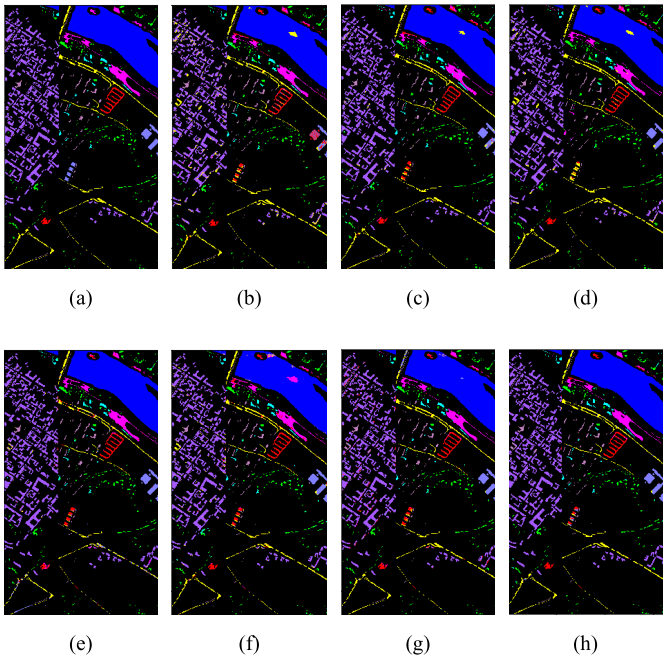


Fig. 10. PC. (a) Ground-truth. (b)–(h) Classification maps for different classifiers. (b) SVM. (c) 3-D-CNN. (d) SSRN. (e) DFSL+NN. (f) DFSL+SVM. (g) RN-FSC. (h) DCFSL.

and class 3 (Corn-mintill), class 10 (Soybean-notill) in the IP data set.

Table VII shows the detailed classification accuracies with different methods on the UP data set. We can find that FSL methods (DFSL+NN, DFSL+SVM, and RN-FSC) are better than deep learning methods (3-D-CNN and SSRN) in the case of limited labeled samples. These results show that FSL methods trained with the idea of meta-learning can better handle the problem of few labeled samples. Compared with DFSL+NN and DFSL+SVM methods (without domain adaptation), DCFSL increases OA by 5.9% and 4.02%, AA by 6.2% and 7.36%, kappa by 7.59% and 5.65%, respectively, which shows that the domain adaptation is important. Compared with RN-FSC (fine-tuned with very limited labeled samples in the target domain), our proposed DCFSL increases OA by 3.46%, AA by 6.65%, Kappa by 4.97%, respectively, which shows that DCFSL not only has a stronger generalization ability but also can learn a discriminative embedded model to the target classes. Similar results can also be drawn from other testing data sets from Table VIII–X, which demonstrates the robustness and superiority of DCFSL.

To further compare the classification performances of the above-mentioned methods, Figs. 9–12 show corresponding classification maps on the different target domain with five labeled samples per class. From the classification maps in Figs. 9–12, it can be visually concluded that compared with the proposed DCFSL, much more samples are assigned to incorrect classes by other methods. In contrast, the maps produced by the DCFSL are most similar to the ground-truth maps.

To verify the effect of different numbers of labeled samples on the DCFSL method, we also randomly select 1, 2, 3, 4, and 5 labeled samples for each class from the target domain to build few-shot data set. Meanwhile, deep learning methods randomly selected 1, 2, 3, 4, and 5 labeled samples for each class to train the model. The results are shown in Fig. 13.

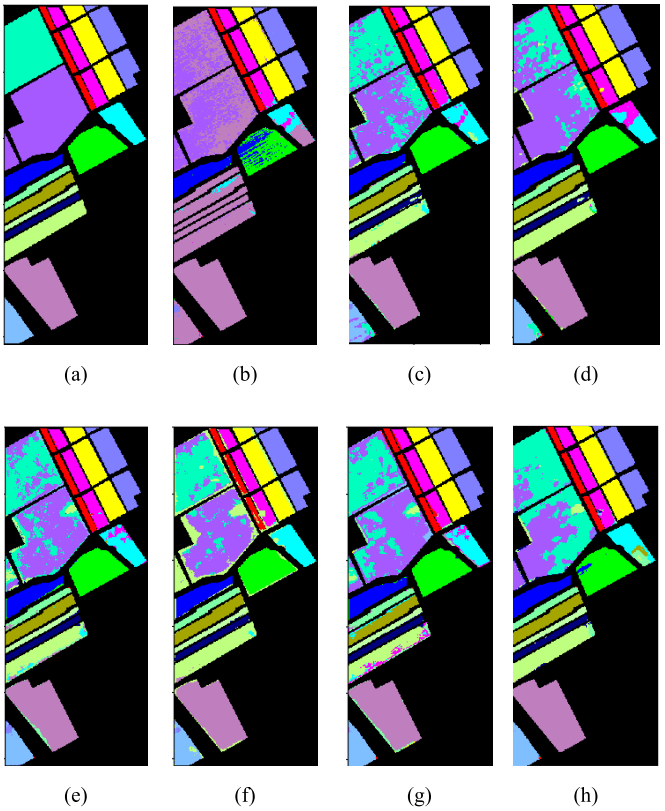


Fig. 11. Salinas. (a) Ground-truth. (b)–(h) Classification maps for different classifiers. (b) SVM. (c) 3-D-CNN. (d) SSRN. (e) DFSL+NN. (f) DFSL+SVM. (g) RN-FSC. (h) DCFSL.

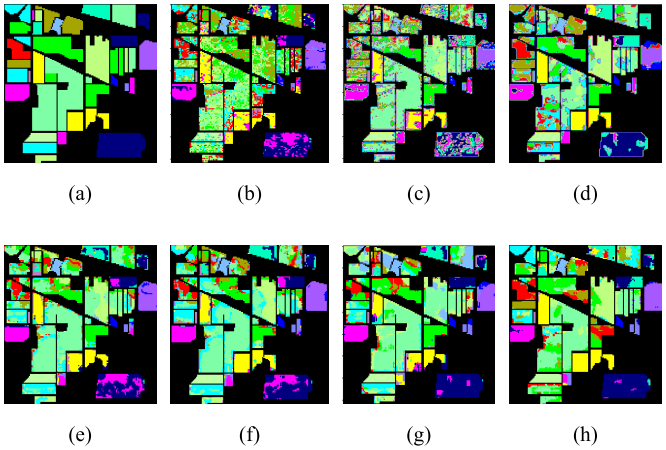


Fig. 12. IP. (a) Ground-truth. (b)–(h) Classification maps for different classifiers. (b) SVM. (c) 3-D-CNN. (d) SSRN. (e) DFSL+NN. (f) DFSL+SVM. (g) RN-FSC. (h) DCFSL.

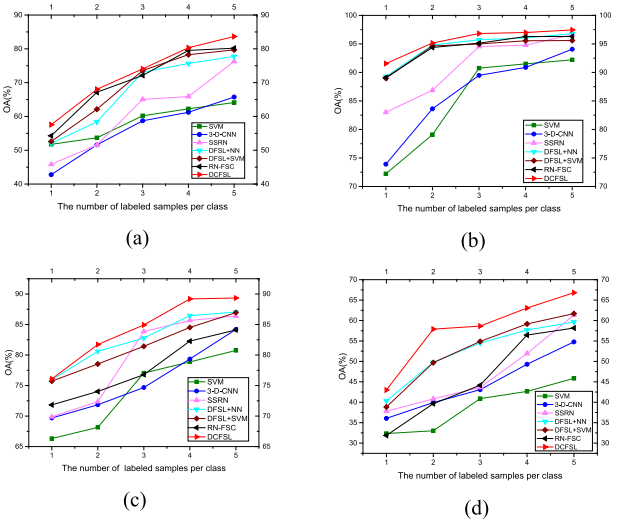


Fig. 13. Classification accuracies with different number of labeled samples on four target data sets.

From Fig. 13, we can find that the OA of the classification methods generally rises as the number of labeled samples increases on four target domain data sets. For all conducted experiments, the proposed DCFSL outperforms, demonstrating its adaptability to the change in the number of labeled samples.

D. Compared With Semisupervised Methods

To further demonstrate the effectiveness of the DCFSL method in HSI classification with few labeled data, the DCFSL method is compared with several semisupervised methods. These semisupervised methods include two semisupervised without cross-domain methods (i.e., LapSVM [32], SS-CNN [33]), and two semisupervised with cross-domain methods (i.e., sFL-CNN [43], Two-CNN-transfer [44]).

For semisupervised methods without cross-domain, few labeled data and many unlabeled data from the target domain are utilized to train a classifier. Specifically, the LapSVM is a classical semisupervised SVM, and the SS-CNN is a novel semisupervised CNN.

TABLE XI

CLASSIFICATION RESULTS (%) ON THE UP DATA SET WITH DIFFERENT SEMISUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	LapSVM	SS-CNN	Two-CNN-transfer	sFL-CNN	DCFSL (PC)	DCFSL (Chikusei)
L=1	45.73 ±4.08	34.72 ±6.79	53.72 ±4.61	55.83 ±6.47	57.34 ±0.83	**57.55 ±6.41**
L=2	47.98 ±8.36	41.43 ±4.81	59.15 ±5.23	64.25 ±5.66	63.20 ±3.45	**68.03 ±5.29**
L=3	50.11 ±3.89	48.44 ±5.68	65.78 ±5.41	65.84 ±3.58	71.48 ±2.15	**74.07 ±4.58**
L=4	50.64 ±4.56	51.20 ±5.13	69.41 ±3.93	69.46 ±4.94	80.25 ±0.66	**80.32 ±3.40**
L=5	52.06 ±10.57	54.48 ±4.65	72.46 ±0.80	70.12 ±3.77	80.60 ±1.02	**83.65 ±1.77**

DCFSL(PC) method selects the PC data set as source domain.
DCFSL(Chikusei) method selects the Chikusei data set as source domain.

For semisupervised methods with cross-domain, adequate source domain labeled data can be utilized to learn transferable features and few target domain labeled data can be utilized to further learn discriminant features of the target domain. Among them, the Two-CNN-transfer uses a two-branch deep CNN to extract joint spectral–spatial features from source HSIs, and transfers the bottom and middle layer weights of the source domain pretraining network to the target domain model. Then the top layers are fine-tuned with limited target domain training samples. In the Two-CNN-transfer method, the same number of spectral bands remained by discarding the absorption bands or noisy bands [44]. The sFL-CNN learns sensor-specific features by constructing a five-layer CNN for classification (FL-CNN). The proposed FL-CNN is further fine-tuned with target domain labeled data collected by the same sensor. In the sFL-CNN method, the data of the source and target domains are collected by the same sensor, and the same number of spectral bands are obtained by setting the missing bands to 0 [43].

Since the Two-CNN-transfer and sFL-CNN methods emphasize that the source and target data are collected from the same sensor, the Chikusei data set is not suitable as the source domain data of these methods. Therefore, for the two-CNN-transfer and sFL-CNN methods, the model trained on the IP data set was used for feature extraction of the Salinas data set, while the model trained on the Salinas data set was used for feature extraction of the IP data set [43], [44]. Similar experiments were carried out on the PC and UP data sets. For fair comparison, we also chose the corresponding HSI data set obtained by the same sensor as the source data set of our proposed method, and expressed the method as DCFSL(.). For example, when the PC data set is used as the source data set, it is recorded as DCFSL(PC).

Tables XI–XIV show the detailed classification accuracies with different methods under the different number of labeled samples (1, 2, 3, 4, and 5) on the UP, PC, Salinas, and

TABLE XII

CLASSIFICATION RESULTS (%) ON THE PC DATA SET WITH DIFFERENT SEMISUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	LapSVM	SS-CNN	Two-CNN-transfer	sFL-CNN	DCFSL (UP)	DCFSL (Chikusei)
L=1	79.13 ±16.73	47.41 ±20.69	79.48 ±2.35	80.84 ±4.76	87.74 ±7.50	**91.55 ±2.67**
L=2	81.40 ±1.30	73.44 ±4.85	80.43 ±3.53	89.94 ±3.80	93.09 ±1.55	**95.15 ±1.37**
L=3	86.17 ±6.36	76.38 ±6.18	83.56 ±1.06	93.53 ±0.83	95.65 ±0.50	**96.82 ±0.43**
L=4	89.61 ±2.31	77.12 ± 4.22	83.97 ±2.74	94.14 ±1.06	96.13 ±0.45	**97.00 ±0.55**
L=5	90.03 ±1.00	80.83 ±2.89	86.90 ±1.49	94.36 ±0.91	96.63 ±0.40	**97.45 ±0.41**

DCFSL(UP) method selects the UP data set as source domain.
DCFSL (Chikusei) method selects the Chikusei data set as source domain.

TABLE XIII

CLASSIFICATION RESULTS (%) ON THE SALINAS DATA SET WITH DIFFERENT SEMISUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	LapSVM	SS-CNN	Two-CNN-transfer	sFL-CNN	DCFSL (IP)	DCFSL (Chikusei)
L=1	38.28 ±4.33	55.46 ±4.66	56.03 ±5.97	48.15 ±7.94	72.33 ±3.44	**72.70 ±3.43**
L=2	38.86 ±1.96	56.85 ±5.83	65.82 ±3.91	61.35 ±7.81	80.39 ±3.40	**80.88 ±2.77**
L=3	42.41 ±1.91	65.86 ±2.41	70.32 ±0.64	69.37 ±6.50	85.70 ±1.55	**85.50 ±1.69**
L=4	45.01 ±1.24	70.95 ±3.96	74.09 ±1.58	76.09 ±0.88	87.88 ±1.09	**88.49 ±0.66**
L=5	47.23 ±8.59	71.76 ±1.68	77.48 ±0.84	81.64 ±2.49	88.25 ±1.03	**89.14 ±0.95**

DCFSL(IP) method selects the IP data set as source domain.
DCFSL (Chikusei) method selects the Chikusei data set as source domain.

TABLE XIV

CLASSIFICATION RESULTS (%) ON THE IP DATA SET WITH DIFFERENT SEMISUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	LapSVM	SS-CNN	Two-CNN-transfer	sFL-CNN	DCFSL (Salinas)	DCFSL (Chikusei)
L=1	20.95 ±6.49	32.18 ±6.46	26.50 ±2.31	23.29 ±1.83	39.88 ±2.99	**43.00 ±3.87**
L=2	27.91 ±3.12	43.43 ±5.66	31.96 ±2.49	32.24 ±3.30	53.36 ±0.68	**57.89 ±3.49**
L=3	30.73 ±7.81	43.67 ±3.26	35.11 ± 4.53	34.58 ±4.39	56.10 ±1.57	**58.64 ±1.94**
L=4	33.77 ±11.90	48.03 ±1.11	39.46 ±3.61	45.16 ±2.28	60.05 ±1.38	**63.06 ±2.03**
L=5	36.17 ±1.64	51.39 ±1.46	42.90 ±2.12	49.57 ±1.30	63.22 ±0.61	**66.81 ±2.37**

DCFSL(Salinas) method selects the Salinas data set as source domain.
DCFSL(Chikusei) method selects the Chikusei data set as source domain.

IP data sets, respectively. It can be found that the semisupervised methods with cross-domain (Two-CNN-Transfer and sFL-CNN) are superior to those without cross-domain

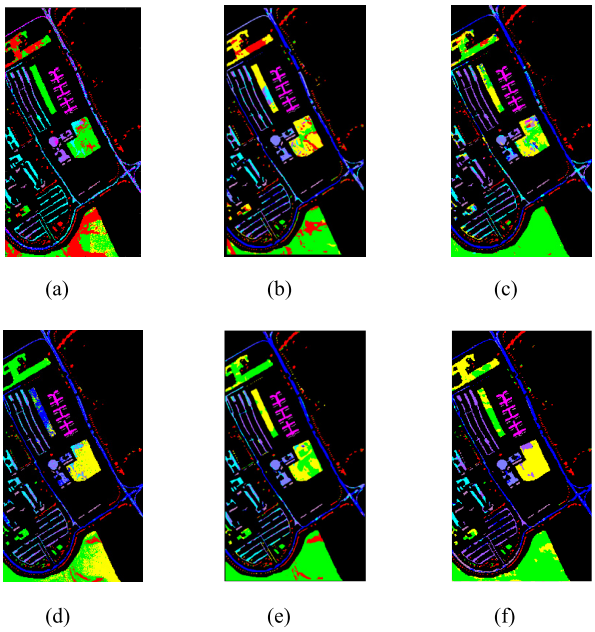


Fig. 14. UP. (a)–(f) Classification maps for different classifiers. (a) LapSVM. (b) SS-CNN. (c) Two-CNN-transfer. (d) sFL-CNN. (e) DCFSL(PC). (f) DCFSL(Chikusei).

(LapSVM and SS-CNN). Compared with these semisupervised methods, all DCFSL(.) methods proposed in this article offer better results. Compared with the Two-CNN-Transfer and sFL-CNN methods, the corresponding DCFSL method with same source domain data set achieves better classification performance on all the four target domain data sets, which indicates the superiority of DCFSL. On each target domain data set, DCFSL(Chikusei) is usually superior to the DCFSL method that takes another data set as the source domain. For example, Table XI shows that the DCFSL(Chikusei) is better than the DCFSL(PC). The main reason is that the categories of Chikusie data set are more diverse, which is constructive to meta-learning [52]. Compared with all these semisupervised methods, the DCFSL(Chikusie) method proposed in this article obtains the best results. For example, as can be seen from Table XI, the OA of DCFSL(Chikusie) is 11.19% higher than that of Two-CNN-transfer on the UP data set with 5 labeled samples per class. The OA of DCFSL(Chikusie) is 7.50% higher than that of sFL-CNN on the UP data set with 5 labeled samples per class, which further indicates the superiority of DCFSL.

To better compare the classification results of the above-mentioned semisupervised methods, Figs. 14–17 show corresponding classification maps, where the classification result of DCFSL is the best match with the ground truth, which further demonstrates the superiority of DCFSL.

E. Compared With Unsupervised Methods

The DCFSL method is also compared with several unsupervised methods to verify the effectiveness of the proposed method, including two without cross-domain methods.

For those with cross-domain, only unlabeled data from the target domain data set can be utilized to train the feature

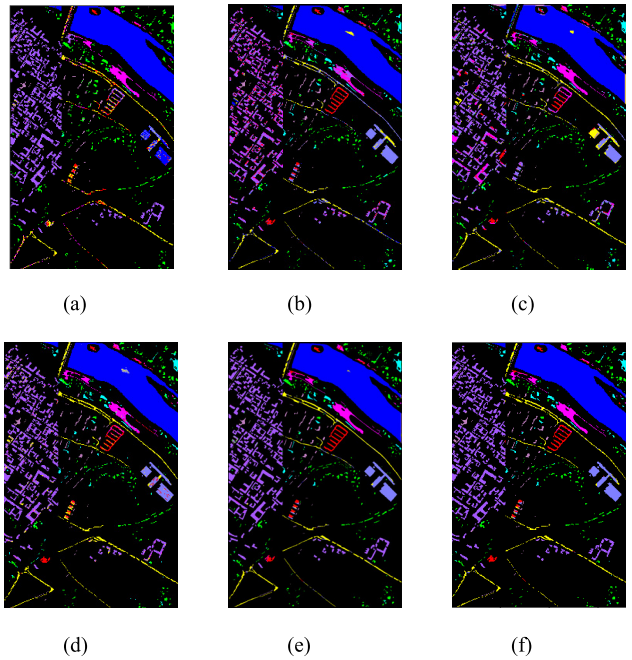


Fig. 15. PC. (a)–(f) Classification maps for different classifiers. (a) LapSVM. (b) SS-CNN. (c) Two-CNN-transfer. (d) sFL-CNN. (e) DCFSL(UP). (f) DCFSL(Chikusei).

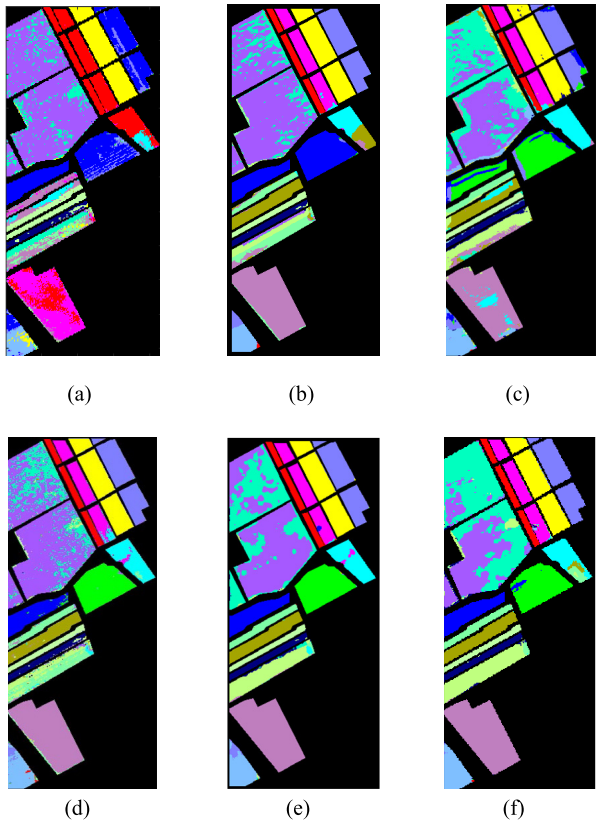


Fig. 16. Salinas. (a)–(f) Classification maps for different classifiers. (a) LapSVM. (b) SS-CNN. (c) Two-CNN-transfer. (d) sFL-CNN. (e) DCFSL(IP). (f) DCFSL(Chikusei).

extraction network. Specifically, the CCPGE is a graph-based feature extraction method, which incorporates collaborative representation using l_2-norm regularization with locality constrained property into graph construction, named collaboration

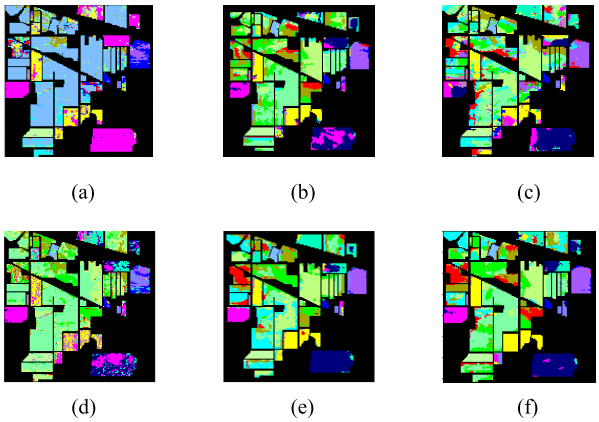


Fig. 17. IP. (a)–(f) Classification maps for different classifiers. (a) LapSVM. (b) SS-CNN. (c) Two-CNN-transfer. (d) sFL-CNN. (e) DCFSL(Salinas). (f) DCFSL(Chikusei).

TABLE XV

CLASSIFICATION RESULTS (%) ON THE UP DATA SET WITH DIFFERENT UNSUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	CCPGE	3D-CAE	uFL-CNN	DCFSL (PC)	DCFSL (Chikusei)
L=1	3.6 ±5.08	49.76 ±3.18	48.11 ±6.71	57.34 ±0.83	**57.55 ±6.41**
L=2	40.29 ±12.17	54.00 ±0.78	59.14 ±6.72	63.20 ±3.45	**68.03 ±5.29**
L=3	46.62 ±6.27	56.15 ±6.61	63.04 ±0.70	71.48 ±2.15	**74.07 ±4.58**
L=4	48.24 ±12.65	58.87 ±0.35	67.19 ±2.78	80.25 ±0.66	**80.32 ±3.40**
L=5	57.93 ±8.34	66.40 ±2.42	67.70 ±1.36	80.60 ±1.02	**83.65 ±1.77**

DCFSL(PC) method selects the PC data set as source domain.
DCFSL(Chikusei) method selects the Chikusei data set as source domain.

competition preserving graph embedding [30]. The 3-D-CAE uses a 3-D convolutional autoencoder (3-D-CAE) to extract spatial–spectral features. Furthermore, for the convenience of comparison with the proposed method, SVM is adopted to obtain the classification results [31]. As for the uFL-CNN method, it is a five-layer CNN (FL-CNN) trained with source domain data set to extract sensor-specific features from the same sensor. After training, the only fully connection layer for classification in the FL-CNN is fine-tuned by few (1, 2, 3, 4, and 5) labeled samples from the target data set.

Since the sFL-CNN emphasizes that the source and target data are collected from the same sensor, the Chikusei data set is not suitable as the source domain data of this method. For fair comparison, we also select the corresponding HSI data set obtained by the same sensor as the source data set of our proposed method, and expressed the method as DCFSL(.).

The classification results with different methods under the different number of labeled samples on the UP, PC, Salinas, and IP data sets are shown in Tables XV–XVIII. Compared with these unsupervised methods, all DCFSL(.)

TABLE XVI

CLASSIFICATION RESULTS (%) ON THE PC DATA SET WITH DIFFERENT UNSUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	CCPGE	3D-CAE	uFL-CNN	DCFSL (UP)	DCFSL (Chikusei)
L=1	2.05 ±0.71	81.51 ±3.45	71.18 ±0.19	87.74 ±7.50	**91.55 ±2.67**
L=2	85.57 ±5.12	83.05 ±1.06	81.48 ±2.56	93.09 ±1.55	**95.15 ±1.37**
L=3	90.03 ±0.49	83.66 ±1.13	83.22 ±5.11	95.65 ±0.50	**96.82 ±0.43**
L=4	90.89 ±5.25	86.43 ±2.70	87.75 ±1.89	96.13 ±0.45	**97.00 ±0.55**
L=5	91.76 ±0.70	89.68 ±4.35	90.56 ±0.80	96.63 ±0.40	**97.45 ±0.41**

DCFSL(UP) method selects the UP data set as source domain.
DCFSL (Chikusei) method selects the Chikusei data set as source domain.

TABLE XVII

CLASSIFICATION RESULTS (%) ON THE SALINAS DATA SET WITH DIFFERENT UNSUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	CCPGE	3D-CAE	uFL-CNN	DCFSL (IP)	DCFSL (Chikusei)
L=1	0.37 ±0.13	68.39 ±1.86	50.67 ±1.36	72.33 ±3.44	**72.70 ±3.43**
L=2	56.01 ±8.16	71.07 ±5.62	52.28 ±1.70	80.39 ±3.40	**80.88 ±2.77**
L=3	77.49 ±4.7	75.22 ±2.03	52.92 ±4.17	85.70 ±1.55	**85.50 ±1.69**
L=4	84.73 ±2.67	80.65 ±3.84	54.96 ±6.66	87.88 ±1.09	**88.49 ±0.66**
L=5	85.78 ±0.09	82.38 ±0.57	55.20 ±1.89	88.25 ±1.03	**89.14 ±0.95**

DCFSL(IP) method selects the IP data set as source domain.
DCFSL (Chikusei) method selects the Chikusei data set as source domain.

TABLE XVIII

CLASSIFICATION RESULTS (%) ON THE IP DATA SET WITH DIFFERENT UNSUPERVISED METHODS (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

	CCPGE	3D-CAE	uFL-CNN	DCFSL (Salinas)	DCFSL (Chikusei)
L=1	1.38 ±1.89	33.34 ±2.92	33.18 ±1.91	39.88 ±2.99	**43.00 ±3.87**
L=2	28.68 ±7.92	36.64 ±1.57	38.03 ±0.61	53.36 ±0.68	**57.89 ±3.49**
L=3	34.28 ±0.37	37.37 ±2.68	41.68 ±4.20	56.10 ±1.57	**58.64 ±1.94**
L=4	39.62 ±6.50	42.41 ±4.73	42.93 ±1.04	60.05 ±1.38	**63.06 ±2.03**
L=5	43.80 ±5.43	45.47 ±4.17	43.63 ±1.92	63.22 ±0.61	**66.81 ±2.37**

DCFSL(Salinas) method selects the Salinas data set as source domain.
DCFSL(Chikusei) method selects the Chikusei data set as source domain

methods proposed in this article obtain better results, and the DCFSL(Chikusei) can produce the best classification results. For example, as can be seen from Table XV, compared with the 3-D-CAE and uFL-CNN methods, the OA of DCFSL method increases by 12.75% and 15.95%, respectively, on the

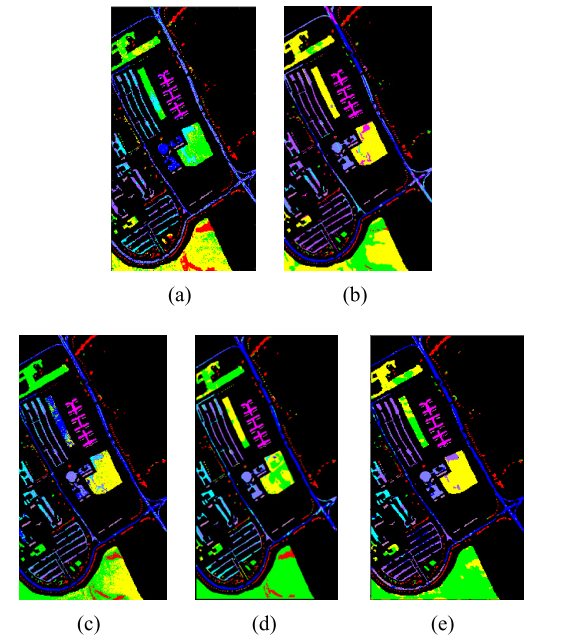


Fig. 18. UP. (a)–(f) Classification maps for different classifiers. (a) CCPGE. (b) SVM. (c) uFL-CNN. (d) DCFSL(PC). (e) DCFSL(Chikusei).

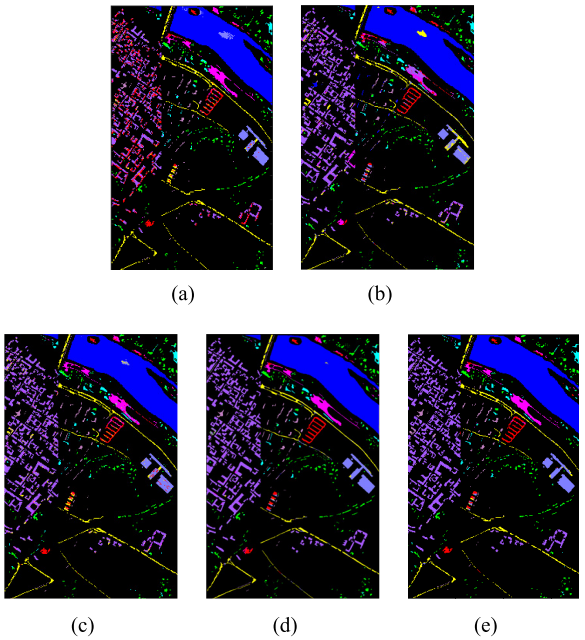


Fig. 19. PC. (a)–(e) Classification maps for different classifiers. (a) CCPGE. (b) SVM. (c) uFL-CNN. (d) DCFSL(UP). (e) DCFSL(Chikusei).

UP data set with five labeled samples per class, which fully demonstrates the superiority of DCFSL when there are fewer labeled samples.

Figs. 18–21 show classification maps on different target domains with five labeled samples per class. It can be seen from Figs. 18–21 that the classification result of DCFSL matches reasonably well with the ground truth, which further illustrates its superiority.

F. Computational Complexity

In order to show the computational efficiency of different methods, the training, testing time, and the number of para-

TABLE XIX
COMPUTATIONAL TIME (SECONDS), FLOPS, AND PARAMETERS ON EACH DATA SET WITH DIFFERENT METHODS

		Supervised Methods						
	Methods	SVM	3DCNN	SSRN	DFSL+NN	DFSL+SVM	RN-FSC	**DCFSL**
UP	Training time	0.31	53.86	66.56	435.03	650.86	310.28	1240.94
	Testing time	0.65	6.23	30.32	5.00	5.02	46.29	5.11
	FLOPs	-	200548	992309	41135920	41135920	6246433	46191390
	#params	-	50277	199153	34920	34920	186433	4259294
PC	Training time	0.29	54.39	66.49	594.99	687.72	391.50	560.80
	Testing time	2.04	21.50	106.37	17.43	16.59	72.97	17.84
	FLOPs	-	198500	976949	41135920	41135920	6246433	46183290
	#params	-	49765	196081	34920	34920	186433	4259194
Salinas	Training time	0.25	63.03	83.58	946.63	485.02	495.20	1267.57
	Testing time	1.74	8.36	41.43	6.54	14.05	55.53	7.45
	FLOPs	-	205490	1760645	41135920	41135920	6246433	47009490
	#params	-	102892	352,28	34920	34920	186433	4269394
IP	Training time	0.25	62.35	145.28	706.71	475.33	450.34	1255.45
	Testing time	0.43	1.5	7.6	1.28	4.77	42.68	1.39
	FLOPs	-	402788	1729925	41135920	41135920	6246433	46977090
	#params	-	100844	346784	34920	34920	186433	4268994

		Semi-supervised Methods				Unsupervised Methods		
	Methods	LapSVM	SS-CNN	Two-CNN-transfer	sFL-CNN	CCPGE	3D-CAE	uFL-CNN
UP	Training time	0.13	1212.41	790.85	100.76	250.07	1169.08	107.26
	Testing time	0.18	3.85	37.93	2.80	6.68	1.61	6.29
	FLOPs	-	7934433	13470554	410974	-	23222295	410974
	#params	-	4360753	1743999	175429	-	256757	175429
PC	Training time	0.95	7037.19	796.23	100.97	322.40	2239.05	104.91
	Testing time	0.23	14.66	243.25	10.03	22.02	5.60	21.64
	FLOPs	-	7895232	13470554	410974	-	23222295	410974
	#params	-	4356112	1743999	175429	-	256757	175429
Salinas	Training time	0.25	674.55	1272.75	260.32	2138.89	1161.58	273.52
	Testing time	0.17	2.86	65.78	41.94	22.82	2.89	12.55
	FLOPs	-	11898781	15127076	873016	-	108020394	873016
	#params	-	4834541	1906806	372136	-	448373	372136
IP	Training time	0.04	109.78	1571.88	251.76	237.09	1159.90	269.31
	Testing time	0.17	0.50	17.66	2.80	4.77	0.68	2.38
	FLOPs	-	11741977	15127076	873016	-	108020394	873016
	#params	-	4815977	1906806	372136	-	448373	372136

TABLE XX
CLASSIFICATION RESULTS (%) ON THE DIFFERENT SOURCE AND TARGET DATA SET (L IS THE NUMBER OF LABELED SAMPLES PER CLASS IN TARGET DOMAIN DATA SET)

Target data set	UP		PC		Salinas		IP	
Method	DCFSL (chikusei)	DCFSL (Houston)	DCFSL (chikusei)	DCFSL (Houston)	DCFSL (chikusei)	DCFSL (Houston)	DCFSL (chikusei)	DCFSL (Houston)
L=1	57.55±6.41	56.52±9.87	91.55±2.67	87.76±7.90	76.09±3.73	72.70±3.43	43.00±3.87	38.65±3.47
L=2	68.03±5.29	65.26±5.51	95.15±1.37	94.75±0.62	81.72±2.69	80.88±2.77	57.89±3.49	53.81±3.49
L=3	74.07±4.58	72.58±2.00	96.82±0.43	96.01±0.44	84.93±2.69	85.50±1.69	58.64±1.94	55.24±1.54
L=4	80.32±3.40	80.09±2.47	97.00±0.55	96.34±0.45	89.19±1.43	88.49±0.66	63.06±2.03	59.04±3.18
L=5	83.65±1.77	82.38±1.53	97.45±0.41	96.86±0.41	89.34±2.19	89.14±0.95	66.81±2.37	60.88±3.06

DCFSL(Chikusei) method selects the Chikusei data set as source domain. DCFSL(Houston) method selects the Houston data set as source domain.

meters and floating-point operations (FLOPs) are analyzed to evaluate the computational complexity of the proposed method. Table XIX lists the training time, testing time, FLOPs, and parameters for all aforementioned methods on four target domain data sets (five labeled samples per class in target domain data set). For methods with cross-domain (DFSL+NN, DFSL+SVM, RN-FSC, Two-CNN-transfer, sFL-CNN, uFL-CNN, and our DCFSL method), the training time is consumed by transfer learning and the testing time contains the time of fine-tuning the model or fitting labeled samples via NN or SVM classifier with five labeled target domain samples and the time of predicting unlabeled samples for each HSI data set. For methods without cross-domain (SVM, 3-D-CNN, SSRN, LapSVM, SS-CNN, CCPGE, and 3-D-CAE), the training time is spent for the training model with target domain samples, and the testing time is also the prediction time of all unlabeled

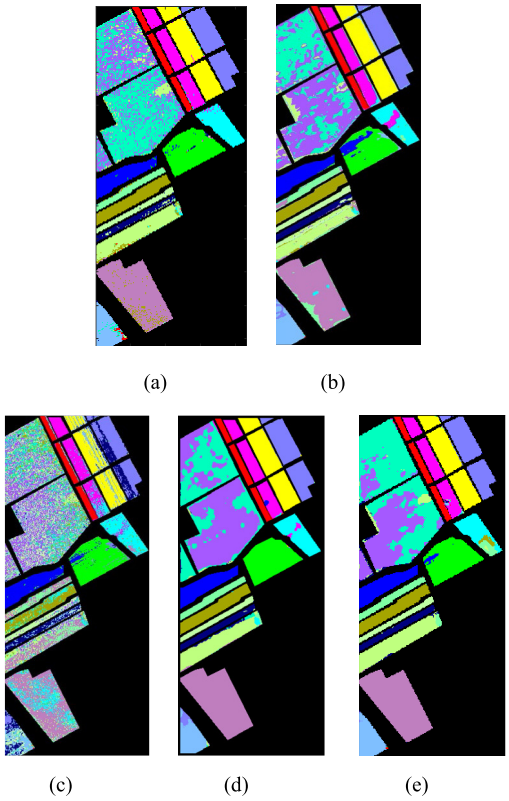


Fig. 20. Salinas. (a)–(e) Classification maps for different classifiers. (a) CCPGE. (b) SVM. (c) uFL-CNN. (d) DCFSL(IP). (e) DCFSL(Chikusei).

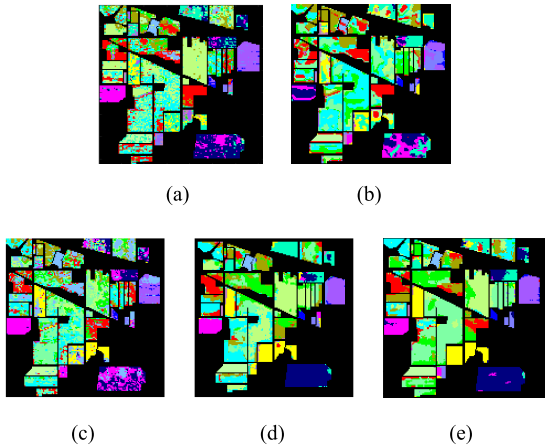


Fig. 21. IP. (a)–(e) Classification maps for different classifiers. (a) CCPGE. (b) SVM. (c) uFL-CNN. (d) DCFSL(Salinas). (e) DCFSL(Chikusei).

samples. The numbers of FLOPs and parameters are studied to evaluate the computational complexity of the deep learning methods.

We can observe from Table XIX that the training time of the methods with cross-domain is longer than that of the methods without cross-domain, which indicates that the computational cost of transfer learning is more expensive. Compared with other methods, we can also observe that our method has more FLOPs and parameters. Although DCFSL has a relatively long training time in most cases and has more FLOPs and parameters, its classification accuracy is the highest.

G. Influence of the Choice of Source Domain

To further explore the influence of the choice of source domain on DCFSL, comparative experiments on the Houston data set were conducted. The Houston data set, containing 15 classes and consisting of 349 $\times$ 1905 pixels and 144 bands, was captured by an airborne sensor over the area of University of Houston campus and neighbor area. Table XX shows the classification performance from the Chikusei or Houston to different target data sets. As can be seen from Table XX, DCFSL(Chikusei) method is slightly better than DCFSL(Houston). The Chikusei data set contains 19 classes, which is larger than the Houston data set. Moreover, the 19 classes comprise water, three types of bare soil, seven types of vegetation, and eight types of man-made objects. The diversity of training samples can be a guarantee, which is constructive to meta-learning. Therefore, a diverse and rich source domain is also important for HSI classification with few labeled samples.

V. Conclusion

In this article, a new method called DCFSL has been proposed to address the problem of cross-domain FSL for HSI classification. It attempts to simultaneously learn a classifier for target classes with a few labeled samples and reduce domain shift. Specifically, a conditional adversarial domain adaptation strategy is utilized to overcome domain shift, which can make extracted embedded features domain-invariant and achieve domain distribution alignment. In addition, FSL is executed in source classes to discover transferable knowledge, and it is also executed in target classes to learn a discriminative embedding model to target classes. The model is trained with the idea of meta-learning. The experimental results conducted on four different target domain data sets demonstrate that DCFSL not only has a better performance than other deep learning methods with few labeled samples but also outperforms existing HSI FSL methods.

References

[1] 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.
[2] L. Liang *et al.*, "Estimation of crop LAI using hyperspectral vegetation indices and a hybrid inversion method," *Remote Sens. Environ.*, vol. 165, pp. 123–134, Aug. 2015.
[3] X. Yang and Y. Yu, "Estimating soil salinity under various moisture conditions: An experimental study," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 5, pp. 2525–2533, May 2017.
[4] N. Yokoya, J. C.-W. Chan, and K. Segl, "Potential of resolution-enhanced hyperspectral data for mineral mapping using simulated EnMAP and sentinel-2 images," *Remote Sens.*, vol. 8, no. 3, pp. 172–189, Feb. 2016.
[5] S. Li, R. Dian, L. Fang, and J. M. Bioucas-Dias, "Fusing hyperspectral and multispectral images via coupled sparse tensor factorization," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 4118–4130, Aug. 2018.
[6] S. Zhang, J. Li, Z. Wu, and A. Plaza, "Spatial discontinuity-weighted sparse unmixing of hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 10, pp. 5767–5779, Oct. 2018.
[7] C.-I. Chang, *Hyperspectral Data Exploitation: Theory and Applications*. Hoboken, NJ, USA: Wiley, 2007.

[8] S. Jia, J. Hu, J. Zhu, X. Jia, and Q. Li, "Three-dimensional local binary patterns for hyperspectral imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 4, pp. 2399–2413, Apr. 2017.

[9] Y. Li, Q. Li, Y. Liu, and W. Xie, "A spatial-spectral SIFT for hyperspectral image matching and classification," *Pattern Recognit. Lett.*, vol. 127, pp. 18–26, Nov. 2019. [Online]. Available:https://www.sciencedirect.com/science/article/abs/pii/S0167865518305117

[10] Q. Wang, F. Zhang, and X. Li, "Optimal clustering framework for hyperspectral band selection," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 10, pp. 5910–5922, Oct. 2018.

[11] Q. Wang, X. He, and X. Li, "Locality and structure regularized low rank representation for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 911–923, Feb. 2019.

[12] L. Samaniego, A. Bardossy, 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.

[13] 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.

[14] Y. Ren, Y. Zhang, W. Wei, and L. Li, "A spectral-spatial hyperspectral data classification approach using random forest with label constraints," in *Proc. IEEE Workshop Electron., Comput. Appl.*, Ottawa, ON, Canada, May 2014, pp. 344–347.

[15] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Deep learning classifiers for hyperspectral imaging: A review," *ISPRS J. Photogramm. Remote Sens.*, vol. 158, pp. 279–317, Dec. 2019.

[16] A. Vali, S. Comai, and M. Matteucci, "Deep learning for land use and land cover classification based on hyperspectral and multispectral earth observation data: A review," *Remote Sens.*, vol. 12, no. 15, p. 2495, Jul. 2020.

[17] 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.

[18] 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.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, May 2017, vol. 60, no. 6, pp. 1097–1105.

[20] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6690–6709, Sep. 2019.

[21] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *J. Sensors*, vol. 2015, Jul. 2015, Art. no. 258619.

[22] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, Oct. 2016.

[23] Y. Li, H. Zhang, and Q. Shen, "Spectral–spatial classification of hyperspectral imagery with 3D convolutional neural network," *Remote Sens.*, vol. 9, no. 1, p. 67, 2017.

[24] W. Li, G. Wu, F. Zhang, and Q. Du, "Hyperspectral image classification using deep pixel-pair features," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 844–853, Feb. 2017.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[26] 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, Feb. 2018.

[27] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 1063–6919.

[28] 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, pp. 1068–1086, Jun. 2018.

[29] Z. Li *et al.*, "Deep multilayer fusion dense network for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 1258–1270, Mar. 2020.

[30] N. Liu, W. Li, and Q. Du, "Unsupervised feature extraction for hyperspectral imagery using collaboration-competition graph," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 6, pp. 1491–1503, Dec. 2018.

[31] S. Mei, J. Ji, Y. Geng, Z. Zhang, X. Li, and Q. Du, "Unsupervised spatial-spectral feature learning by 3D convolutional autoencoder for hyperspectral classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6808–6820, Apr. 2019.

[32] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Nov. 2006.

[33] 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.

[34] S. Rajan, J. Ghosh, and M. M. Crawford, "An active learning approach to hyperspectral data classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 4, pp. 1231–1242, Apr. 2008.

[35] D. Tuia, F. Ratle, F. Pacifici, M. F. Kanevski, and W. J. Emery, "Active learning methods for remote sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 7, pp. 2218–2232, Jul. 2009.

[36] M. M. Crawford, D. Tuia, and H. L. Yang, "Active learning: Any value for classification of remotely sensed data?" *Proc. IEEE*, vol. 101, no. 3, pp. 593–608, Mar. 2013.

[37] 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.

[38] 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.

[39] H. L. Yang and M. M. Crawford, "Domain adaptation with preservation of manifold geometry for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 2, pp. 543–555, Feb. 2016.

[40] E. Othman, Y. Bazi, F. Melgani, H. Alhichri, N. Alajlan, and M. Zuair, "Domain adaptation network for cross-scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 8, pp. 4441–4456, Aug. 2017.

[41] Z. Wang, B. Du, Q. Shi, and W. Tu, "Domain adaptation with discriminative distribution and manifold embedding for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 7, pp. 1155–1159, Jul. 2019.

[42] L. Jiao, M. Liang, H. Chen, S. Yang, H. Liu, and X. Cao, "Deep fully convolutional network-based spatial distribution prediction for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 10, pp. 5585–5599, Oct. 2017.

[43] S. Mei, J. Ji, J. Hou, X. Li, and Q. Du, "Learning sensor-specific spatial-spectral features of hyperspectral images via convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 8, pp. 4520–4533, Aug. 2017.

[44] J. Yang, Y. Zhao, and J. C. 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.

[45] W. Chen, Y. Liu, Z. Kira, Y. Wang, and J. Huang, "A closer look at few-shot classification," in *Proc. ICLR*, New Orleans, LA, USA, May 2019, pp. 1–16.

[46] Y. Wang, Q. Yao, J. Kwok, and L. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–34, 2020.

[47] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. ICLR*, Toulon, France, Mar. 2017, pp. 1–11.

[48] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.* Cambridge, MA, USA: MIT Press, 2017, pp. 4080–4090.

[49] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.* Cambridge, MA, USA: MIT Press, 2016, pp. 3630–3638.

[50] 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. 2019.

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

[52] 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.

[53] H. Lee and H. Kwon, "Going deeper with contextual CNN for hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 26, no. 10, pp. 4843–4855, Oct. 2017.
[54] X. He, Y. Chen, and P. Ghamisi, "Heterogeneous transfer learning for hyperspectral image classification based on convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 5, pp. 3246–3263, May 2020.
[55] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Proc. Adv. NIPS*, 2018, pp. 1640–1650.
[56] N. Yokoya and A. Iwasaki, "Airborne hyperspectral data over Chikusei," Space Appl. Lab., Univ. Tokyo, Tokyo, Japan, Tech. Rep. SAL-2016-05-27, May 2016.
[57] K. Sun *et al.*, "A robust and efficient band selection method using graph representation for hyperspectral imagery," *Int. J. Remote Sens.*, vol. 37, no. 20, pp. 4874–4889, Oct. 2016.

Zhaokui Li received the M.S. degree in computer application from Liaoning University, Shenyang, China, in 2003, and the Ph.D. degree in computer software and theory from Wuhan University, Wuhan, China, in 2014.

He is a Professor with the School of Computer, Shenyang Aerospace University, Shenyang. His research interests include hyperspectral image analysis, computer vision, and machine learning.

Ming Liu received the B.E. degree from Dalian Polytechnic University, Dalian, China, in 2018. She is pursuing the master's degree with the School of Computer, Shenyang Aerospace University, Shenyang, China.

Her research interests include hyperspectral image processing, computer vision, and deep learning.

Yushi Chen (Member, IEEE) received the Ph.D. degree from the Harbin Institute of Technology, Harbin, China, in 2008.

He is an Associate Professor with the School of Electronics and Information Engineering, Harbin Institute of Technology. His research interests include remote sensing data processing and machine learning.

Yimin Xu received the M.S. degree in electrical and computer engineering from Duke University, Durham, USA, in May 2020. He interns with the School of Computer Science, Shenyang Aerospace University, Shenyang, China.

His research interests include computer vision, deep learning, weak-supervised learning, and pattern recognition.

Wei Li (Senior Member, IEEE) received the B.E. degree in telecommunications engineering from Xidian University, Xi'an, China, in 2007, the M.S. degree in information science and technology from Sun Yat-sen University, Guangzhou, China, in 2009, and the Ph.D. degree in electrical and computer engineering from Mississippi State University, Starkville, MS, USA, in 2012.

Subsequently, he spent one year as a Postdoctoral Researcher with the University of California, Davis, CA, USA. He is a Professor with the School of Information and Electronics, Beijing Institute of Technology, Beijing, China. His research interests include hyperspectral image analysis, pattern recognition, and data compression.

Dr. Li is an Associate Editor of the IEEE SIGNAL PROCESSING LETTERS (SPL) and the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS).

Qian Du (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the University of Maryland, Baltimore, MD, USA, in 2000.

She is the Bobby Shackouls Professor with the Department of Electrical and Computer Engineering, Mississippi State University, Starkville, MS, USA. Her research interests include hyperspectral remote sensing image analysis and applications, pattern classification, data compression, and neural networks.

Dr. Du is a fellow of the Society of Photo-Optical Instrumentation Engineers (SPIE). She received the 2010 Best Reviewer Award from the IEEE Geoscience and Remote Sensing Society. She was a Co-Chair of the Data Fusion Technical Committee of the IEEE Geoscience and Remote Sensing Society from 2009 to 2013, and the Chair of the Remote Sensing and Mapping Technical Committee of the International Association for Pattern Recognition from 2010 to 2014. She has served as an Associate Editor for the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, the *Journal of Applied Remote Sensing*, and the IEEE SIGNAL PROCESSING LETTERS. Since 2016, she has been the Editor-in-Chief of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING. She is the General Chair of the 4th IEEE GRSS Workshop on *Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, Shanghai, in 2012.