Hyperspectral Image Classification via Discriminant Gabor Ensemble Filter

Ke-Kun Huang, *Member, IEEE*, Chuan-Xian Ren, *Member, IEEE*, Hui Liu, Zhao-Rong Lai, *Member, IEEE*, Yu-Feng Yu, *Member, IEEE*, and Dao-Qing Dai, *Senior Member, IEEE*

***Abstract*—For a broad range of applications, hyperspectral image (HSI) classification is a hot topic in remote sensing, and convolutional neural network (CNN)-based methods are drawing increasing attention. However, to train millions of parameters in CNN requires a large number of labeled training samples, which are difficult to collect. A conventional Gabor filter can effectively extract spatial information with different scales and orientations without training, but it may be missing some important discriminative information. In this article, we propose the Gabor ensemble filter (GEF), a new convolutional filter to extract deep features for HSI with fewer trainable parameters. GEF filters each input channel by some fixed Gabor filters and learnable filters simultaneously, then reduces the dimensions by some learnable 1 × 1 filters to generate the output channels. The fixed Gabor filters can extract common features with different scales and orientations, while the learnable filters can learn some complementary features that Gabor filters cannot extract. Based on GEF, we design a network architecture for HSI classification, which extracts deep features and can learn from limited training samples. In order to simultaneously learn more discriminative features and an end-to-end system, we propose to introduce the local discriminant structure for cross-entropy loss by combining the triplet hard loss. Results of experiments on three HSI datasets show that the proposed method has significantly higher classification accuracy than other state-of-the-art methods. Moreover, the proposed method is speedy for both training and testing.**

Manuscript received 19 August 2020; revised 18 November 2020; accepted 3 January 2021. Date of publication 5 February 2021; date of current version 19 July 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61976104, Grant 61976229, Grant 11631015, Grant U1611265, Grant 61703182, and Grant 62006056; in part by the Natural Science Foundation of Guangdong Province under Grant 2020A1515010702 and Grant 2019A1515011266; in part by the Science and Technology Program of Guangzhou under Grant 201804010248; and in part by the Science and Technology Program of Guangdong Province under Grant 2020B121201013: Guangdong Provincial Key Laboratory of Conservation and Precision Utilization of Characteristic Agricultural Resources in Mountainous Areas. This article was recommended by Associate Editor L. Jiao. *(Corresponding author: Dao-Qing Dai.)*

Ke-Kun Huang and Hui Liu are with the School of Mathematics, Jiaying University, Meizhou 514015, China (e-mail: kkcocoon@163.com; imlhxm@163.com).

Chuan-Xian Ren is with the School of Mathematics, Sun Yat-sen University, Guangzhou 510275, China, and also with the Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, Sun Yat-sen University, Guangzhou 510275, China (e-mail: rchuanx@mail.sysu.edu.cn).

Zhao-Rong Lai is with the Department of Mathematics, College of Information Science and Technology, Jinan University, Guangzhou 510632, China (e-mail: laizhr@jnu.edu.cn).

Yu-Feng Yu is with the Department of Computer and Information Science, University of Macau, Macau, China (e-mail: yuyufeng220@163.com).

Dao-Qing Dai is with the School of Mathematics, Sun Yat-sen University, Guangzhou 510275, China (e-mail: stsddq@mail.sysu.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TCYB.2021.3051141.

Digital Object Identifier 10.1109/TCYB.2021.3051141

***Index Terms*—Convolutional neural network (CNN), discriminant learning, Gabor filter, hyperspectral image (HSI) classification.**

I. Introduction

A HYPERSPECTRAL image (HSI) contains hundreds of continuous bands in the ultraviolet, visible, and infrared regions, which effectively combine spatial and spectral information. HSI classification, that is, classifying every pixel with a certain land-cover type, is the cornerstone of HSI analysis. It has a broad range of applications, including land-cover mapping, mineral exploration, water-pollution detection, natural disasters, and biological threats [1].

Many HSI classification algorithms have been proposed over the past decade, including subspace-based methods [2]–[7], support vector machine (SVM) [8], extreme learning machine (ELM) [9], sparse representation classifier (SRC) [10], low-rank representation [11], extended morphological attribute profiles (EMAPs) [12], invariant attribute profiles (IAPs) [13], etc. Deep-learning-based methods have drawn much attention recently in image classification [14]–[16]. Deep learning uses a neural network with multiple hidden layers to automatically learn features from the original image, layer by layer. Due to its excellent performance, deep learning has been applied to HSI classification, with better results compared to conventional shallow methods.

For HSI, the number of labeled samples is limited, because it takes effort to determine the class of each pixel. Some unsupervised deep-learning methods, including stacked automatic encoder (SAE) and deep brief network (DBN), have been proposed to extract features for HSI [17], [18]. However, since SAE- and DBN-based methods only have 1-D fully connected layers, they cannot automatically learn spatial features. Furthermore, they cannot provide end-to-end classification methods, because they need to use traditional classifiers.

Convolutional neural networks (CNNs) have excellent image-classification capabilities and provide end-to-end classification methods [19]–[22]. Many CNN-based HSI classification methods have been proposed recently [23]–[34]. In particular, Li *et al.* [26] increased the number of training samples by constructing sample pairs to overcome the problem of insufficient training samples and proposed a CNN with pixel-pair features (CNN-PPF) to classify HSI. Zhang *et al.* [27] proposed a diverse region-based CNN (CNN-DR) for HSI classification, where the diverse regions can simultaneously take advantage of spectral, spatial structure, and semantic context-aware information in each pixel. Mei *et al.* [28] proposed a sensor-specific CNN for HSI classification (CNN-C), which can fine-tune and transfer features to other images by the same sensor. Zhong *et al.* [29] proposed a spectral–spatial residual network (CNN-SSRN) for HSI classification, achieving good classification accuracy. Xu *et al.* [30] proposed a spectral–spatial unified network (CNN-SSUN) for HSI classification, which integrates spectral feature extraction, spatial feature extraction, and classifier training in a unified network. Xu *et al.* [31] proposed a random patches network based on CNN (CNN-RPNet) for HSI classification, which directly regards the random patches taken from the image as the convolution kernels without training. Zhu *et al.* [32] presented a convolutional capsule network (CNN-Capsule). Instead of using full connections, it uses local connections and shared transform matrices in the CNN-capsule network architecture. Tang *et al.* [34] proposed a 3-D octave convolution with the spatial–spectral attention network (3DOC-SSAN) to capture discriminative spatial–spectral features for the classification of HSIs.

The above CNN-based methods have shown good results, but still have some problems.

1) To learn good features for HSI, the depth and the number of parameters of CNN must be sufficiently large. However, because of the limited training samples for HSI, it is easy to overfit if the network is complex. CNNs normally fail to handle large and unknown object transformations when the training data are insufficient. Furthermore, a complex network requires a long training time.
2) Most CNN-based HSI classification methods only use traditional cross-entropy loss to train the network. However, as some samples have similar spectra but different labels, or vice versa, to solely use the cross-entropy loss is not good enough to learn discriminative features.

Recently, some methods are proposed to address the limited training samples for deep-learning-based HSI classification. Deng *et al.* [35] proposed a similarity-based deep metric model (S-DMM), which attained good results both for same- and cross-scene HSI classifications with limited training samples. Liu *et al.* [36] presented the morphological attribute profile cube and deep random forest (MAPC-DRF) for small sample classification of HSI. Zhang *et al.* [37] designed a deep quadruplet network for HSI with a small number of samples.

On the other hand, Gabor filtering has attracted attention due to its ability to extract edges and textures with different scales and orientations without training processing [38], [39], and its effectiveness for HSI classification has been shown [40]–[45]. In particular, Li and Du [42] exploited the benefits of using spatial features extracted from Gabor filters for the nearest regularized subspace classifier, and He *et al.* [44] designed a low-rank Gabor filtering that is computationally efficient and tailored for the special characteristics of HSI.

Because Gabor filters can extract good convolutional features without training processing, people are inspired to incorporate Gabor filters and CNN for HSI classification. Chen *et al.* [46] combined Gabor filters with CNN (CNN-Gabor) for HSI classification to mitigate the problem of overfitting. Kang *et al.* [47] proposed a method based on Gabor filtering and deep network (GFDN), which stacks the Gabor features and spectral features as the inputs for an SAE network. However, CNN-Gabor and GFDN only use Gabor features instead of the original pixel as input to the network. Luan *et al.* [48] developed a Gabor convolutional network (GCN), modulating the learnable filters via Gabor filters. However, GCN still has more than a few learnable parameters and is not specifically designed for HSI classification. Liu *et al.* [49] designed CNN kernels strictly in the form of Gabor filters to reduce the number of learnable parameters (GaborNet). However, GaborNet only learns the Gabor parameters, including the phase and standard deviation, which discards to learn the standard convolutional filter.

In this article, we propose a new filter, called the Gabor ensemble filter (GEF), to effectively combine the Gabor filters and the standard convolutional filters. Different from the existing methods that use Gabor filters merely for preprocessing [46], [47], and those discarding to train the standard convolutional filters [48], [49], the proposed method filters each input channel by fixed Gabor filters together with learnable filters, followed by some learnable 1×1 filters to obtain the output channels. Based on GEF, we designed a network architecture for HSI classification, which can not only extract deep enough features but can be learned by limited training samples. To learn more discriminative features and an end-to-end system at the same time, we propose to introduce the local discriminant structure for cross-entropy loss.

The main contributions of this article are as follows.

1) We propose a new convolutional operator by combining traditional Gabor filters and learnable filters, so that deep enough features can be extracted by the network with fewer trainable parameters that can be learned by a limited number of training samples. The Gabor filters can extract common features with different scales and orientations, while the learnable filters can learn some complementary features that Gabor filters cannot extract.
2) We propose to introduce the local discriminant structure for cross-entropy loss by combining the triplet hard loss, so that more discriminative features and an end-to-end system are learned at the same time.
3) With limited training samples, the proposed method performs significantly better than other state-of-the-art HSI classification methods. Moreover, the proposed method is fast for both training and testing.

The remainder of this article is organized as follows. Section II outlines the proposed method. Section III reports on experimental results, and Section IV provides the conclusion.

II. Proposed Method

A. Motivation

Gabor filters can encode shape and texture with different scales and orientations for HSIs. Fig. 1 shows a band of an HSI filtered by a Gabor filter. We can find that the

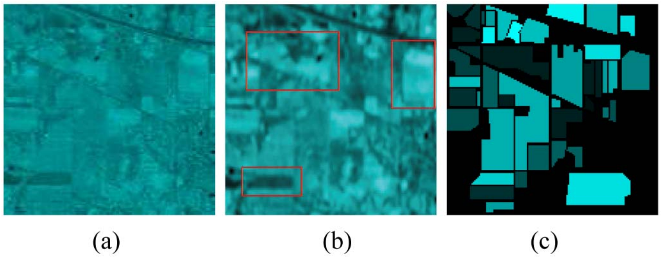


Fig. 1. Band of an HSI filtered by a Gabor filter. A Gabor filter can encode the shape and texture with a certain scale and orientation. The rectangular areas of the Gabor filter result contain more useful discriminative features than those of the original HSI. (a) Band of an HSI. (b) Band filtered by a Gabor filter. (c) Ground truth.

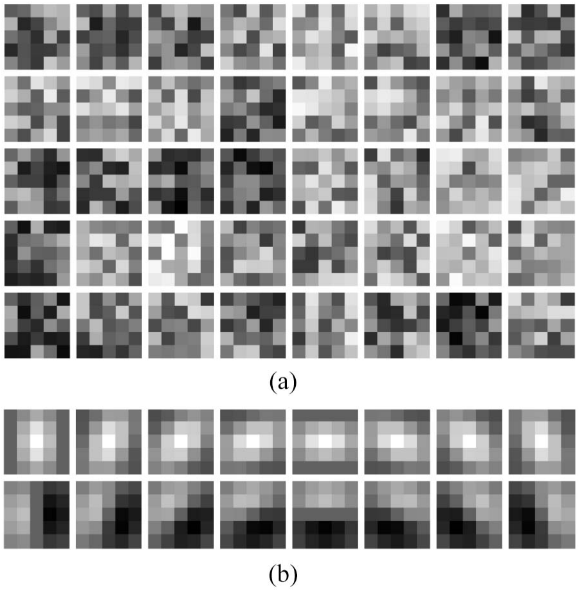


Fig. 2. Comparison of the learnable filters trained by CNN and the Gabor filters. The learnable filters and the Gabor filters are different and have their own characteristics. It is possible to combine them for better representation for HSI. (a) Filters trained by CNN on a hyperspectral dataset. (b) Gabor filters at eight orientations. Top: Real part. Bottom: Imaginary part.

rectangular areas of the Gabor filter result contain more useful discriminative features than those of the original HSI. On the other hand, CNNs have achieved impressive results for HSI classification compared with traditional methods. However, it requires a large number of labeled training samples, which are difficult to collect. Because Gabor filters can extract good convolutional features without training processing, and it is hard to obtain Gabor-like filters automatically when training CNN, we are inspired to incorporate Gabor filters and learnable convolutional filters for HSI classification. However, how to effectively combine them is a meaningful issue. Existing methods either used Gabor features as the preprocess step [46], [47], or discarded to train the standard convolutional operator [48], [49].

Fig. 2 shows the comparison of the learnable filters trained by CNN in the first layer and the Gabor filters. The learnable filters are trained by 100 samples per class on a hyperspectral dataset. It can be found that the learnable filters and Gabor filters are different and have their own characteristics. On the one hand, though Gabor filters can effectively extract features with different scales and orientations, they may be missing some important discriminative information. On the other hand, because there are limited training samples, it is hard to train a well CNN for HSI classification. Inspired by the aspects, we proposed a new method to effectively combine the standard convolutional operator and the Gabor filters, so that deep enough features can be extracted by the network with fewer trainable parameters. Specifically, we first empirically set different parameter values of scales and orientations to generate some fixed Gabor filters. Note that the fixed Gabor filters do not participate any training procedure. Then, we combine the fixed Gabor filters and the learnable filters to extract deep features of HSI. The Gabor filters can extract common features with different scales and orientations, while the learnable filters can capture some complementary features that Gabor filters cannot extract.

Most existing CNN-based HSI classification methods only used cross-entropy loss to train the network [23]–[33]. However, cross-entropy loss is not the best criterion for HSI classification. There are some other loss functions that can learn discriminative features that suppress intraclass variations and maximize the gap between the samples from different classes [50]–[52]. In particular, the triplet hard loss [52] can characterize the local discriminative structure, which suppresses the distance between the selected positive pairs and maximizes the gap between the selected negative pairs. However, only using the triplet loss cannot learn an end-to-end system. In this article, we propose to combine the triplet hard loss and cross-entropy loss to learn more discriminative features and an end-to-end system at the same time.

B. Proposed Gabor Ensemble Filter

As discussed above, we are motivated to propose the GEF, which combines fixed Gabor filters with learnable filters, so that the learnable parameters are reduced and deep features are learned at the same time.

Specifically, each channel of input data is convoluted by some fixed Gabor filters and some learnable filters. Note that here we use depthwise convolution instead of normal convolution. The filtering result contains many channels, so we apply some learnable 1×1 convolutions to reduce the dimensions and learn more discriminative features.

Suppose $\mathbf{x}_s^{(l)}$ denotes the sth channel of layer l in the network, and $\mathbf{g}_k$ denotes the kth fixed Gabor filter. We convolute each channel of input data by some fixed Gabor filters as follows:

$$\mathbf{y}_{s,k}^{(l,1)} = \mathbf{x}_s^{(l)} * \mathbf{g}_k \tag{1}$$

where

$$\begin{aligned}\mathbf{g}_k(x, y) = \mathbf{g}_{\lambda_k,\theta_k}(x, y) &= e^{-\frac{x'^2+\gamma^2 y'^2}{2\sigma_k^2}} \times e^{i\left(\frac{2\pi x'}{\lambda_k}+\psi\right)}\\ x' &= x\cos\theta_k + y\sin\theta_k\\ y' &= -x\sin\theta_k + y\cos\theta_k.\end{aligned} \tag{2}$$

θ_k is the orientation angle of Gabor kernels, λ_k is the wavelength of the sinusoidal factor, γ is the spatial aspects ratio,

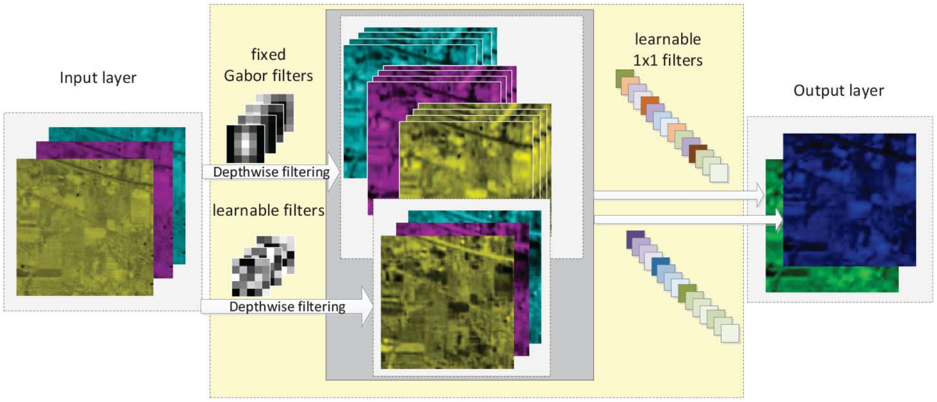


Fig. 3. Proposed GEF. In GEF, the input channels are convoluted by the fixed Gabor filters and the learnable filters, followed by some learnable 1×1 filters to obtain the output channels. The Gabor filters can extract common features with different scales and orientations, while the learnable filters can capture some complementary features that Gabor filters cannot extract.

σ_k is the standard derivation of the Gaussian envelope, and $\psi = 0$ and $\psi = (\pi)/(2)$ return the real and imaginary parts, respectively, of the Gabor filter. The parameter σ_k is determined by λ_k and the spatial frequency bandwidth b as $\sigma_k = (\lambda_k)/(\pi)\sqrt{(\ln 2)/(2)}(2^b+1)/(2^b-1)$.

For example, we can set different scales $\lambda_k \in \{8, 16\}$ and different orientations $\theta_k \in \{0, (\pi)/(8), (2\pi)/(8), (3\pi)/(8), (4\pi)/(8), (5\pi)/(8), (6\pi)/(8), (7\pi)/(8)\}$, with $b = 5$ and $\gamma = 1$.

Suppose $\mathbf{w}_{s,i}^{(l)}$ denotes the ith learnable filter for $\mathbf{x}_s^{(l)}$. We convolute each channel of input data by some learnable filters as follows:

$$\mathbf{y}_{s,i}^{(l,2)} = \mathbf{x}_s^{(l)} * \mathbf{w}_{s,i}. \tag{3}$$

Then, we merge the convolutional results by fixed Gabor filters and learnable filters as follows:

$$\mathbf{Y}^{(l)} = \left\{ \bigcup_{s,k} \mathbf{y}_{s,k}^{(l,1)}, \bigcup_{s,i} \mathbf{y}_{s,i}^{(l,2)} \right\}. \tag{4}$$

There would be many channels in $\mathbf{y}^{(l)}$, so we further apply 1×1 convolution to reduce the dimension. The t-th channel of the next layer $\mathbf{x}_t^{(l+1)}$ can be obtained as

$$\mathbf{x}_t^{(l+1)} = f\left(\sum_s \mathbf{y}_s^{(l)} * \mathbf{w}_{s,t}^{(l)} + \mathbf{b}_t \right) \tag{5}$$

where $\mathbf{y}_s^{(l)}$ is the sth channel of $\mathbf{Y}^{(l)}$, $\mathbf{w}_{s,t}^{(l)}$ is the learnable 1×1 filter for $\mathbf{y}_s^{(l)}$ to create output channel $\mathbf{x}_t^{(l+1)}$, $\mathbf{b}_t$ is the bias, and f is the activation function, such as the rectified linear unit $f(x) = \max(x, 0)$.

For example, as shown in Fig. 3, it is assumed that there are three channels in the input layer l. Each input channel is convoluted by four fixed Gabor filters and one learnable filter, for a total of 15 channels in the middle of Fig. 3. The 15 channels are further convoluted by two learnable 1×1 filters, resulting in two output channels of the next layer.

We mix the fixed Gabor filters and the learnable filters to extract deep features for HSI with fewer trainable parameters, so we called it a GEF. Gabor filters can extract good features with different scales and orientations and have no learnable parameters. In the proposed GEF, we only need to train a few learnable filters to learn some complementary features that Gabor filters cannot extract. Without Gabor filters, we have to train more learnable filters and need more training samples to learn good features. According to (4), the convolutional results contain two parts, that is: 1) the output for Gabor filters $\mathbf{y}_{s,i}^{(l,1)}$ and 2) the output for learnable filters $\mathbf{y}_{s,i}^{(l,2)}$, where only the learnable filters need to train. So, the proposed method can extract deep features for HSI with fewer trainable parameters. The experimental results will demonstrate the effectiveness of the proposed method.

C. Proposed Network Architecture Based on GEF

In this section, we designed a proper network architecture based on GEF for HSI classification, in order to extract deep enough features by limited training samples, as shown in Fig. 4.

Because the original HSI has hundreds of channels, we usually use some dimensionality reduction methods to compress the data before feeding it for the network. HSI reduction methods have two categories. The first one is feature exaction [6], [7], which combines some bands to generate new features. The other is feature selection [53]–[57], which just drops some redundant bands. Although there are many dimensionality reduction methods, we found that using principal component analysis (PCA) as the initialization method can obtain better performance for the proposed method. So, we first apply PCA to reduce the dimensions to 20 principal components. Different dimensionality reduction methods will be verified in the experiments.

After PCA transformation, we extract a patch of size $p \times p$ centered on each pixel to create an input sample for the network. So, the input cube has size $p \times p \times 20$. We can set $p = 27$ as the patch size. Different patch sizes will be verified in the experiments.

In the first convolution layer, a GEF with 16 fixed Gabor filters generated by two scales $\lambda \in \{8, 16\}$ and eight orientations is used to convolute each channel, obtaining data with size $17 \times 17 \times 320$, which is downsampled by max pooling with step 2×2, followed by 128 1×1 learnable filters and ReLu

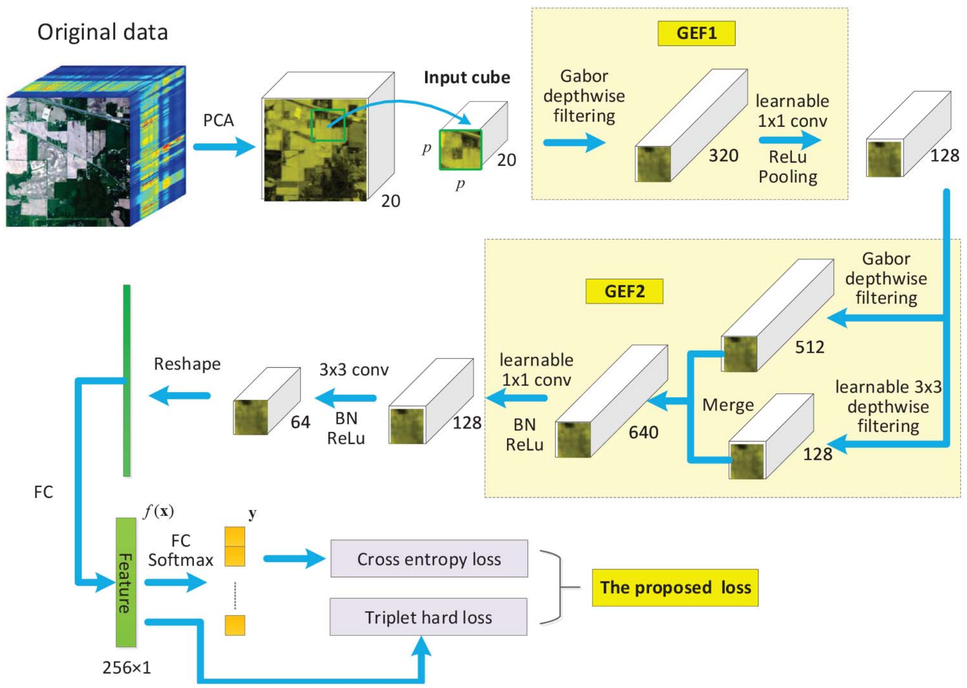


Fig. 4. Proposed network architecture. We first apply PCA to the original HSI and extract a patch centered on each pixel to create an input cube for the network. Then, two different GEFs are applied to convolute the data, including the Relu activation function, max-pooling, and batch normalization layers. After that, a standard convolution of spatial size 3 × 3 and two full connections are applied to predict the class label. Note that we use the feature of size 256 × 1 to calculate the triplet hard loss, and use the output of the softmax layer to calculate the cross-entropy loss, then combine them to formulate the proposed loss.

activation function, resulting in data with size $9 \times 9 \times 128$ for the next layer. The GEF is called GEF1.

In the second convolution layer, we use another GEF, called GEF2, with four fixed Gabor filters and one learnable 3×3 filter, to depthwise convolute the data, obtaining data with size $9 \times 9 \times 512$ and $9 \times 9 \times 128$, respectively. Then, we merge them into data with size $9 \times 9 \times 640$, and convolute it by 128 1×1 learnable filters. After applying batch normalization and a ReLu activation function, we obtain data with size $9 \times 9 \times 128$ for the next layer. In GEF2, the fixed Gabor filters are generated by $\lambda = 8$ and $\theta \in \{0, (\pi)/(4), (\pi)/(2), (3\pi)/(4)\}$.

In the third convolution layer, a convolution with 64 learnable filters of spatial size 3×3 is applied, followed by batch normalization and a ReLu activation function, resulting in data of size $7 \times 7 \times 64$. We reshape it to 3136×1 and apply a full connection to a feature of size 256×1.

Finally, a full connection of size $C \times 1$ and a softmax activation function are applied to predict the class label, where C is the number of classes.

D. Discriminative Learning

Most CNN-based HSI classification methods only use traditional cross-entropy loss to train the network. However, because some samples have similar spectra but different labels, and vice versa, to only use the cross-entropy loss is not sufficient to learn discriminative features.

Some loss functions have been proposed to train more discriminative features for CNN [50]–[52]. In particular, triplet hard loss [52] focuses on hard samples for classification. For each sample, it picks the most dissimilar sample with the same identity and the most similar sample with a different identity to obtain a triplet, and it suppresses the distance between the selected positive pairs and maximizes the gap between the selected negative pairs

$$\mathcal{L}_d = \frac{1}{N}\sum_{a=1}^{N} h\left(\alpha + \max_{p} D_{a,p} - \min_{n} D_{a,n}\right) \tag{6}$$

where $D_{a,p} = \|f(\mathbf{x}_a) - f(\mathbf{x}_p)\|_2$ is the Euclidean distance for the feature-embedding output from the network, $(\mathbf{x}_a, \mathbf{x}_p)$ is a positive pair, $(\mathbf{x}_a, \mathbf{x}_n)$ is a negative pair, $h(x) = \max(0, x)$ is the hinge loss function, and α is a margin to filter trivial pairs.

After the discriminative features attained by the loss functions [50]–[52], it is necessary to use a classifier, such as a nearest neighbor classifier or SVM, to classify testing samples. In other words, the loss functions [50]–[52] cannot output the class labels directly. To address the problem, we propose a new loss that combines the cross-entropy loss and triplet hard loss, that is, we introduce the local discriminant structure for cross-entropy loss, which can be formulated as

$$\mathcal{L}_{\text{proposed}} = -\frac{1}{N}\sum_{i=1}^{N} < \mathbf{y}_i, \log \hat{\mathbf{y}}_i > +\gamma \mathcal{L}_d \tag{7}$$

where $\mathbf{y}_i$ is a one-hot vector indicating the true label for training sample $\mathbf{x}_i$, $\hat{\mathbf{y}}_i$ is the output of the softmax layer connected to the resulting feature for $\mathbf{x}_i$, and γ is a given weight to balance the cross-entropy loss and triplet hard loss. We will verify in experiments that it is easy to select an appropriate value of γ to obtain good performance. When $\gamma = 0$ or $\gamma \to \infty$, the performance will be diminished.

In Fig. 4, we use the feature of size 256×1 to calculate the triplet hard loss, and use the output of the softmax layer to calculate the cross-entropy loss, then combine them to formulate the proposed loss.

E. Complexity Analysis

The training time for CNN is mainly decided by the number of trainable parameters and the convergence speed. In the proposed network architecture, GEF1 has $1 \times 1 \times 320 \times 128 = 41\,088$ trainable parameters. In GEF2, the learnable depthwise filter has $3 \times 3 \times 128 + 128 = 1280$ trainable parameters, and the learnable 1×1 filter has $1\times1\times(128{+}128\times4)\times128{+}128 = 82\,048$ trainable parameters. The following learnable 3×3 filter has $3 \times 3 \times 128 \times 64 + 64 = 73\,792$ trainable parameters. The batch normalization layer has $2 \times 64 = 256$ trainable parameters. Suppose the input patch size is 17×17, then the full connection layer has 803 072 trainable parameters, and the softmax layer has 2313 trainable parameters. The total number of trainable parameters of the network is 1 005 648, which is fewer than that of other state-of-the-art CNNs. We will show in our experiments that the training process converges only after using hundreds of training batches, which only takes several minutes.

III. Experimental Results

We implemented the proposed method by Keras based on the Python language. Keras is a high-level neural networks API, which is capable of running on top of TensorFlow. The proposed method is a discriminant CNN based on GEF, denoted by DGEF.

A. Data Description and Experimental Setting

We adopted three HSI datasets, that is: 1) Salinas; 2) Houston; and 3) Indian Pines, for experiments to evaluate the performance of the proposed method.

The first dataset was gathered by the Airborne Visible Infrared Imaging Spectrometer (AVIRIS) sensor over Salinas Valley, CA, USA. The data composed of 512×217 pixels with a spatial resolution of 3.7 m and 204 bands after 20 water-absorption bands were removed. It contained 16 classes, including vegetables, bare soil, and vineyards. The false-color image, ground truth, and corresponding class names of the Salinas data are shown in Fig. 5.

The second dataset was acquired over the University of Houston campus and the neighboring area, had 144 spectral bands in the 380–1050 nm region, and consisted of 349×1905 pixels with a spatial resolution of 2.5 m. The Houston dataset contained 15 classes, including grass, trees, water, roads, and highways. The false-color image, ground truth, and corresponding class names of the Houston data are shown in Fig. 6.

The third dataset was acquired by the AVIRIS sensor over the Indian Pines test site in Northwest Indiana. After removing the water-absorption bands, the image consisted of 200 spectral bands with 145×145 pixels. Its spectral covering range was from 0.4 to 2.5 μm with a spatial resolution of 20 m. It contained 16 classes, including alfalfa, corn, oats, wheat, and woods. The false-color image, ground truth, and corresponding class names of the Indian Pines data are shown in Fig. 7.

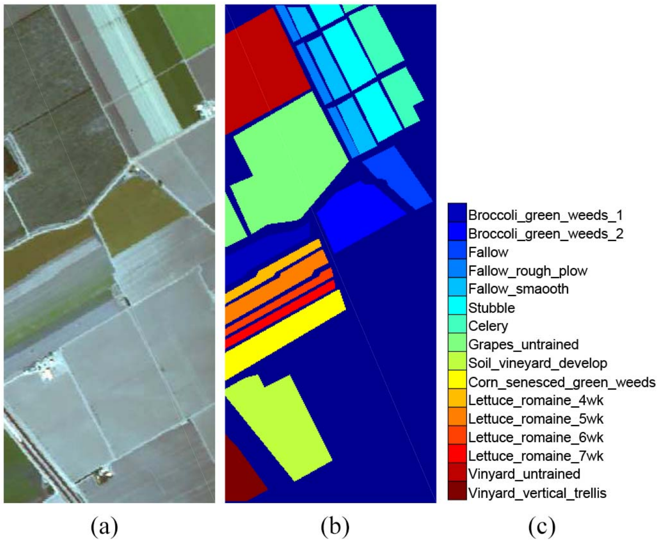


Fig. 5. Salinas dataset. (a) False-color image. (b) Ground truth. (c) Class names.

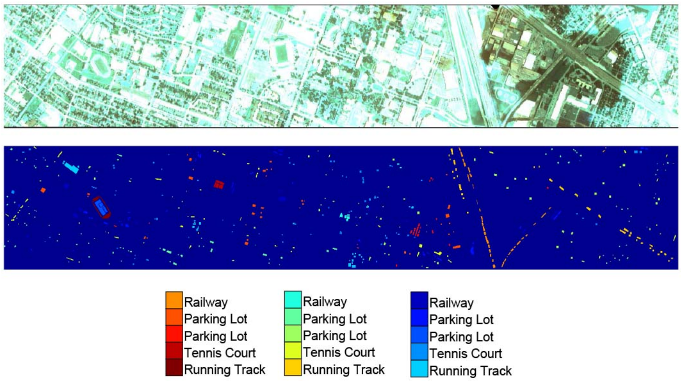


Fig. 6. Houston dataset. Top: False-color image. Middle: Ground truth. Bottom: Class names.

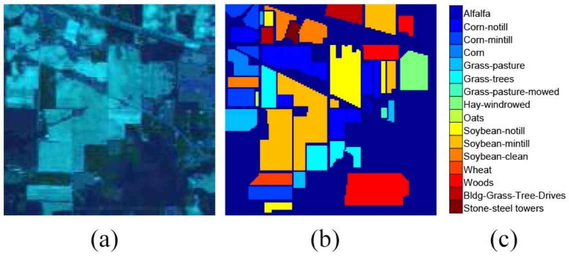


Fig. 7. Indian Pines dataset. (a) False-color image. (b) Ground truth. (c) Class names.

The experiments were conducted in two parts. The first part compared the proposed method and other state-of-the-art methods. The second part analyzed the impact of different parameters of the proposed method. If not specified, we repeated the experiments five times to report the mean performance. The overall accuracy (OA), average accuracy (AA) (the average of the accuracies for each class), and Kappa coefficient were utilized to quantitatively estimate different methods.

TABLE I
CLASSIFICATION RESULTS OF STATE-OF-THE-ART METHODS ON THE SALINAS DATASET

Class	Train(#)	Test(#)	CNN–	CNN–	CNN–	CNN–	CNN–	CNN–	CNN–	CNN–	S-DMM	MAPC-	**DGEF**
			PPF	DR	C	SSRN	SSUN	RPNet	Gabor	Capsule		DRF	
1	30	1979	97.02	100.0	98.48	71.30	99.95	99.49	100.0	100.0	100.0	100.0	100.0
2	30	3696	99.27	98.84	98.13	99.97	95.89	99.19	100.0	98.93	100.0	100.0	100.0
3	30	1946	98.00	99.69	98.82	99.54	100.0	99.79	99.90	99.38	100.0	100.0	100.0
4	30	1364	99.85	95.67	97.43	99.85	100.0	99.71	100.0	97.93	99.49	100.0	96.70
5	30	2648	94.11	98.79	97.92	98.53	98.68	98.49	99.22	100.0	100.0	96.56	99.54
6	30	3929	97.00	99.95	98.24	100.0	99.92	99.72	99.85	100.0	100.0	99.34	100.0
7	30	3549	99.55	96.42	99.69	99.89	99.63	99.69	97.04	93.86	100.0	100.0	100.0
8	30	11241	64.13	79.23	79.73	80.62	84.75	76.82	83.80	100.0	88.28	99.07	97.29
9	30	6173	94.86	99.74	99.17	99.98	98.98	99.66	98.60	96.69	99.68	100.0	100.0
10	30	3248	88.05	95.54	96.12	98.28	97.04	98.86	99.91	100.0	98.74	99.97	99.88
11	30	1038	98.75	98.17	98.84	99.81	98.94	99.61	100.0	100.0	99.71	100.0	100.0
12	30	1897	100.0	100.0	99.53	100.0	100.0	99.95	100.0	97.55	100.0	100.0	100.0
13	30	886	98.31	99.55	97.40	100.0	99.44	99.77	100.0	100.0	100.0	99.32	100.0
14	30	1040	94.62	98.56	98.94	98.37	96.63	95.87	99.81	74.14	100.0	99.52	100.0
15	30	7238	74.97	85.24	77.07	85.02	67.44	86.65	83.02	100.0	86.03	93.62	99.25
16	30	1777	98.48	99.49	99.66	95.95	99.44	99.10	99.50	99.50	99.10	99.94	100.0
Overall Accuracy (%)			86.81	92.77	91.67	92.48	91.62	92.91	93.91	95.11	95.50	98.70	**99.21**
Average Accuracy (%)			93.56	96.56	95.95	95.44	96.05	97.02	97.54	97.37	98.19	99.21	**99.54**
Kappa Coefficient			0.8593	0.9229	0.9111	0.9197	0.9106	0.9214	0.9324	0.9478	0.9499	0.9861	**0.9916**

TABLE II
CLASSIFICATION RESULTS OF STATE-OF-THE-ART METHODS ON THE HOUSTON DATASET

Class	Train(#)	Test(#)	CNN–	CNN–	CNN–	CNN–	CNN–	CNN–	CNN–	CNN–	S-DMM	MAPC-	**DGEF**
			PPF	DR	C	SSRN	SSUN	RPNet	Gabor	Capsule		DRF	
1	30	1221	97.71	89.93	97.72	98.94	95.25	89.52	87.93	95.89	99.67	83.77	97.95
2	30	1224	98.61	99.84	97.88	87.01	97.39	94.12	83.97	99.86	99.10	92.98	99.43
3	30	667	100.0	98.05	100.0	100.0	99.70	100.0	97.42	92.21	100.0	98.42	100.0
4	30	1214	99.84	93.74	93.03	99.75	99.01	92.09	86.90	100.0	93.74	83.44	84.10
5	30	1212	99.92	99.09	100.0	100.0	97.69	98.35	95.65	96.62	98.51	100.0	100.0
6	30	295	95.25	82.37	92.67	97.97	98.98	91.53	98.15	87.17	100.0	100.0	95.59
7	30	1238	94.67	89.42	92.92	88.61	81.26	91.68	87.22	78.92	87.24	90.30	92.49
8	30	1214	72.90	71.42	57.75	39.29	74.63	84.02	60.61	92.54	78.34	89.55	84.60
9	30	1222	82.98	84.04	89.32	90.75	83.88	87.64	79.63	87.59	91.65	85.38	82.08
10	30	1197	83.79	88.14	94.51	92.98	90.48	90.81	100.0	90.40	92.65	98.04	93.57
11	30	1205	94.11	91.20	89.01	85.73	97.51	87.22	88.42	84.75	94.94	82.19	99.67
12	30	1203	79.63	85.29	68.96	71.65	92.93	82.13	88.16	87.19	86.95	90.43	91.52
13	30	439	78.82	89.07	90.77	95.22	77.22	80.64	94.88	100.0	95.44	100.0	98.86
14	30	398	100.0	100.0	100.0	100.0	98.99	100.0	100.0	100.0	100.0	100.0	100.0
15	30	630	99.05	100.0	100.0	100.0	98.41	98.10	100.0	90.23	100.0	99.85	100.0
Overall Accuracy (%)			91.26	90.24	89.70	87.72	91.67	90.64	87.94	91.17	93.43	91.30	**93.66**
Average Accuracy (%)			91.82	90.77	90.97	89.86	92.22	91.19	89.93	92.22	94.55	92.96	**94.66**
Kappa Coefficient			0.9064	0.8954	0.8887	0.8681	0.9107	0.8988	0.8697	0.9054	0.9290	0.9068	**0.9320**

To compute the discriminative loss, we randomly selected 400 samples as a training batch. If there were not enough training samples, all of the samples were selected as a training batch. The parameter α was fixed at 5.0 for all of the datasets. Then, stochastic gradient descent (SGD) was used with 500 iterations, momentum of 0.99, and weight decay of 0.0001. We initially set a base learning rate of 0.001. All of the convolutional layers were initialized using zero-mean Gaussian random variables with a standard deviation of $(2)/(N_{\text{in}} + N_{\text{out}})$, where N_{in} is the number of input units and N_{out} is the number of output units in the weight tensor.

B. Comparison of State-of-the-Art Methods Based on Deep Learning

We compared our method to other methods based on deep learning, including CNN-PPF [26], CNN-DR [27], CNN-C [28], CNN-SSRN [29], CNN-SSUN [30], CNN-RPNet [31], Gabor-CNN [46], CNN-Capsule [32], S-DMM [35], and MAPC-DRF [36].

We downloaded the source code written by the original authors for each compared method, and set the parameters optimally to generate the results.

1) Comparison of Classification Accuracy: Tables I–III show the class-specific accuracy, OA, OA standard deviation (OA std), AA, and Kappa coefficient of different methods on the Salinas, Houston, and Indian Pines datasets, respectively. Here, we randomly selected 30 samples per class for training, and the remaining for testing. If there are not enough samples for a certain class, we randomly selected 75% samples for training. From the results, it can be observed that:

1) CNN-PPF attains lower performance, for it only takes 1-D spectral information as the input to the CNN

TABLE III
CLASSIFICATION RESULTS OF STATE-OF-THE-ART METHODS ON THE INDIAN PINES DATASET

Class	Train(#)	Test(#)	CNN–PPF	CNN–DR	CNN–C	CNN–SSRN	CNN–SSUN	CNN–RPNet	CNN–Gabor	CNN–Capsule	S-DMM	MAPC-DRF	**DGEF**
1	30	16	96.15	96.15	100.0	100.0	100.0	87.50	100.0	100.0	100.0	100.0	100.0
2	30	1398	60.09	56.96	81.90	69.24	77.04	79.33	77.00	67.23	91.56	77.68	88.41
3	30	800	68.23	62.72	93.75	81.75	92.00	80.25	89.53	92.05	91.25	95.62	87.62
4	30	207	74.19	76.04	98.07	100.0	93.24	97.10	100.0	99.58	99.03	100.0	100.0
5	30	453	88.12	88.55	97.57	95.58	98.68	90.95	99.65	98.96	93.60	96.03	93.82
6	30	700	97.89	92.11	95.86	98.86	89.14	96.57	98.22	97.95	100.0	94.29	99.57
7	21	7	87.50	100.0	100.0	100.0	100.0	85.71	100.0	100.0	100.0	100.0	100.0
8	30	448	82.31	97.59	98.21	99.33	100.0	100.0	100.0	100.0	99.33	100.0	100.0
9	15	5	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
10	30	942	84.74	90.22	79.51	79.72	82.70	81.95	90.30	77.06	92.46	79.41	95.54
11	30	2425	72.76	73.93	71.42	56.04	81.61	71.75	76.40	88.72	81.57	85.86	92.12
12	30	563	63.70	63.53	77.80	98.76	98.58	86.68	91.04	80.94	94.85	94.85	88.45
13	30	175	98.38	98.38	99.43	100.0	100.0	100.0	99.51	100.0	100.0	99.43	100.0
14	30	1235	94.45	89.24	93.52	97.17	95.63	95.06	91.73	97.87	97.41	97.98	96.36
15	30	356	92.08	88.80	80.90	98.88	99.16	95.51	98.99	98.70	94.38	100.0	98.03
16	30	63	100.0	100.0	100.0	100.0	100.0	100.0	98.92	100.0	100.0	93.65	100.0
Overall Accuracy (%)			78.40	77.89	84.53	80.50	88.21	84.40	86.52	88.29	91.67	89.78	**93.44**
Average Accuracy (%)			85.04	85.89	91.75	92.21	94.24	90.52	94.46	93.69	95.97	94.67	**96.25**
Kappa Coefficient			0.7557	0.7497	0.8349	0.7916	0.8742	0.8245	0.8458	0.875	0.9053	0.8909	**0.9301**

and cannot automatically learn spatial–spectral features. CNN-DR cannot improve the performance compared with CNN-PPF when using 30 training samples per class on the Houston and Indian Pines datasets. CNN-C achieves better accuracy on the Salinas and Indian Pines datasets, but it cannot improve the performance on the Houston dataset;

2) CNN-SSRN obtains better performance on the Salinas dataset, but the accuracy is not good on the Houston and Indian Pines datasets. CNN-SSUN improves the accuracy on the Houston and Indian Pines datasets except on the Salinas dataset. CNN-RPNet attains good performance accuracy on the Salinas and Houston datasets, but the accuracy is not good on the Indian Pines dataset;
3) CNN-Gabor achieves good performance on the Salinas dataset but not good on the Houston and Indian Pines datasets. CNN-Capsule improves the accuracy on the Salinas datasets compared with the above methods, but it the accuracy is not good on the Indian Pines dataset;
4) S-DMM attains high accuracy on the Houston dataset, but not good enough on the Salinas and Indian Pines datasets. MAPC-DRF achieves better performance on the Salinas dataset but not good on the Houston and Indian Pines datasets;
5) the proposed method obtains the highest OA and smallest standard deviation for all of the datasets, which demonstrates its effectiveness. It improves a mean OA of 8.5%, 6.0%, 3.9%, and 2.2% for three datasets, respectively, compared with CNN-DR, CNN-Gabor, CNN-Capsule, and MAPC-DRF, respectively. The AA and Kappa coefficient of the proposed method are also significantly better than those of other methods.

TABLE IV
COMPARISON OF DIFFERENT NUMBERS OF TRAINING SAMPLES PER CLASS ON THE SALINAS DATASET

Train(#)		CNN–DR	CNN–Gabor	CNN-Capsule	MAPC-DRF	Proposed **DGEF**
30	OA	92.77	93.91	95.11	98.70	**99.21**
	OA std	±0.82	±0.72	±0.51	±0.22	±0.12
	AA	96.56	97.54	97.37	99.21	**99.54**
	Kappa	0.9229	0.9324	0.9478	0.9861	**0.9916**
50	OA	93.46	96.08	95.69	99.03	**99.51**
	OA std	±0.50	±0.52	±0.58	±0.10	±0.09
	AA	97.68	98.48	98.00	99.48	**99.78**
	Kappa	0.9302	0.9562	0.9540	0.9897	**0.9945**
100	OA	95.94	98.07	99.29	99.81	**99.84**
	OA std	±0.43	±0.37	±0.21	±0.08	±0.07
	AA	98.12	99.41	99.76	99.86	**99.91**
	Kappa	0.9567	0.9784	0.9924	0.9979	**0.9983**

2) Comparison of Different Numbers of Training Samples Per Class: Tables IV–VI show the comparisons of different numbers of training samples per class on the Salinas, Houston, and Indian Pines datasets, respectively. According to the results, we can find that the fewer the training samples the more improvement the proposed method obtains. With fewer training samples, the accuracy of the proposed method only decreases slightly, while that of the other methods dramatically declines, especially on the Salinas dataset.

In the above experiments, we randomly selected training samples over the whole HSI image. However, for the Houston dataset, IEEE Geoscience and Remote Sensing Society (GRSS) defined a spatially disjoint training and testing set, where the number of training samples for each class is equal or close to 200. In order to simulate a more realistic situation, here we use the spatially disjoint training and testing sets on the Houston dataset for the experiment. Table VII shows the results under this scenario. As we can observe that there is a significant performance gap between the obtained results considering randomly selected training samples in Table V and spatially disjoint training and testing samples in Table VII, but

TABLE V
COMPARISON OF DIFFERENT NUMBERS OF TRAINING SAMPLES PER CLASS ON THE HOUSTON DATASET

Train(#)		CNN–DR	CNN–Gabor	CNN-Capsule	MAPC-DRF	Proposed **DGEF**
30	OA	90.24	87.94	91.17	91.30	**93.66**
	OA std	±1.12	±1.18	±0.92	±0.95	±0.61
	AA	90.77	89.93	92.22	92.96	**94.66**
	Kappa	0.8954	0.8697	0.9054	0.9068	**0.9320**
50	OA	96.20	92.15	96.18	94.89	**96.90**
	OA std	±0.41	±0.82	±0.47	±0.61	±0.42
	AA	96.55	93.52	96.76	95.55	**97.42**
	Kappa	0.9593	0.9152	0.9591	0.9452	**0.9668**
100	OA	98.00	97.25	98.29	97.64	**98.82**
	OA std	±0.37	±0.37	±0.36	±0.39	±0.32
	AA	98.27	97.65	98.57	98.12	**98.94**
	Kappa	0.9785	0.9702	0.9817	0.9747	**0.9874**

TABLE VI
COMPARISON OF DIFFERENT NUMBERS OF TRAINING SAMPLES PER CLASS ON THE INDIAN PINES DATASET

Train(#)		CNN–DR	CNN–Gabor	CNN-Capsule	MAPC-DRF	Proposed **DGEF**
30	OA	77.89	86.52	88.88	89.78	**93.44**
	OA std	±1.94	±0.86	±1.21	±1.32	±0.69
	AA	85.89	91.46	94.69	94.67	**96.25**
	Kappa	0.7497	0.8458	0.8814	0.8909	**0.9301**
50	OA	83.90	92.28	93.26	94.56	**95.41**
	OA std	±1.65	±0.67	±0.78	±0.52	±0.35
	AA	86.73	95.98	96.22	97.28	**97.82**
	Kappa	0.8188	0.9087	0.9281	0.9421	**0.9511**
100	OA	94.05	96.86	95.19	95.48	**98.32**
	OA std	±0.65	±0.63	±0.82	±0.83	±0.21
	AA	97.94	98.16	97.36	97.98	**98.25**
	Kappa	0.9365	0.9678	0.9487	0.9518	**0.9801**

TABLE VII
COMPARISON OF SPATIALLY DISJOINT TRAINING AND TESTING SET GIVEN BY GRSS ON THE HOUSTON DATASET

	CNN–DR	CNN–Gabor	CNN-Capsule	MAPC-DRF	Proposed **DGEF**
OA	73.12	76.58	83.07	79.70	**84.89**
OA std	±1.73	±1.32	±1.12	±1.34	±0.92
AA	76.25	80.16	84.59	80.26	**87.62**
Kappa	0.7121	0.7505	0.8174	0.7811	**0.8305**

the proposed method still achieves better performance than other methods under this scenario.

3) Comparison of Classification Maps: Figs. 8 and 9 show the classification maps of different methods using 30 training samples per class on the Salinas dataset and the Indian Pines dataset, respectively. We can easily find that many regions of the classification maps achieved by the proposed method are obviously more accurate than those of CNN-DR, CNN-Gabor, and CNN-Capsule. The classification map of different methods on the Houston dataset can be found in the Appendix.

Fig. 10 shows the full classification maps of different methods using 30 training samples per class on the Indian Pines dataset. As illustrated by the map, the assignments by the proposed method are more accurate and smoother than other compared methods. The full classification maps of different methods on the Salinas dataset and the Houston dataset can be found in the supplementary material.

TABLE VIII
COMPARISON OF TRAINING TIME AND TESTING TIME

	Method	Indian Pines	Salinas	Houston
Training (minutes)	CNN-PPF	151	312	278
	CNN-DR	31	52	45
	CNN-Gabor	5	7	6
	CNN-Capsule	5	10	7
	DGEF	**4**	**6**	**5**
Testing (seconds)	CNN-PPF	**3**	**14**	**4**
	CNN-DR	15	62	18
	CNN-Gabor	5	21	6
	CNN-Capsule	24	125	60
	DGEF	5	16	5

TABLE IX
COMPARISON OF RELATED METHODS

Methods	Salinas	Houston	Indian pines
Baseline	86.55±0.83	87.52±0.66	67.05±1.32
Gabor [42]	94.84±0.48	92.83±0.45	91.16±0.76
Gabor-NRS [42]	96.64±0.37	96.91±0.38	96.40±0.48
Lowrank-Gabor [44]	97.35±0.32	97.10±0.33	96.58±0.50
Gabor-CNN [46]	98.07±0.30	97.25±0.40	96.92±0.47
GFDN [47]	98.36±0.31	96.39±0.44	98.48±0.38
Proposed **DGEF**	**99.84**±0.07	**98.82**±0.32	**99.28**±0.25

4) Comparison of Training and Testing Time: Table VIII shows the comparison of training time in minutes and testing time in seconds. The experiments were conducted on a single GPU of an NVIDIA Titan X in the Python language. The testing time is the whole time for all of the testing samples. We can find that the training process of the proposed method only takes a few minutes, while CNN-PPF takes a few hours and CNN-DR takes more than half an hour. CNN-Capsule is fast for training, but takes much more testing time for its complex network. The testing process of the proposed method is the fastest of all of the compared methods.

Fig. 11 shows the training loss versus iterations for the proposed method. Here, an iteration denotes the use of a batch of 400 samples to train the network. We can find that only after 200 iterations on the Indian Pines and Salinas datasets, and 400 iterations on the Houston dataset, the loss will converge. So, we fix the iterations to 500 for all of the datasets for training.

C. Comparison of Related Methods

We compared our method to some related methods based on Gabor features, including original Gabor [42], Gabor-NRS [42], Lowrank-Gabor [44], Gabor-CNN [46], and GFDN [47]. Here, we randomly selected 100 samples per class for training and the remaining for testing, and discarded seven small classes for the Indian Pines dataset. The baseline method is the nearest neighbor classifier based on the 1-D spectral feature. Table IX compares the OA and its standard deviation of the related methods. We can find that

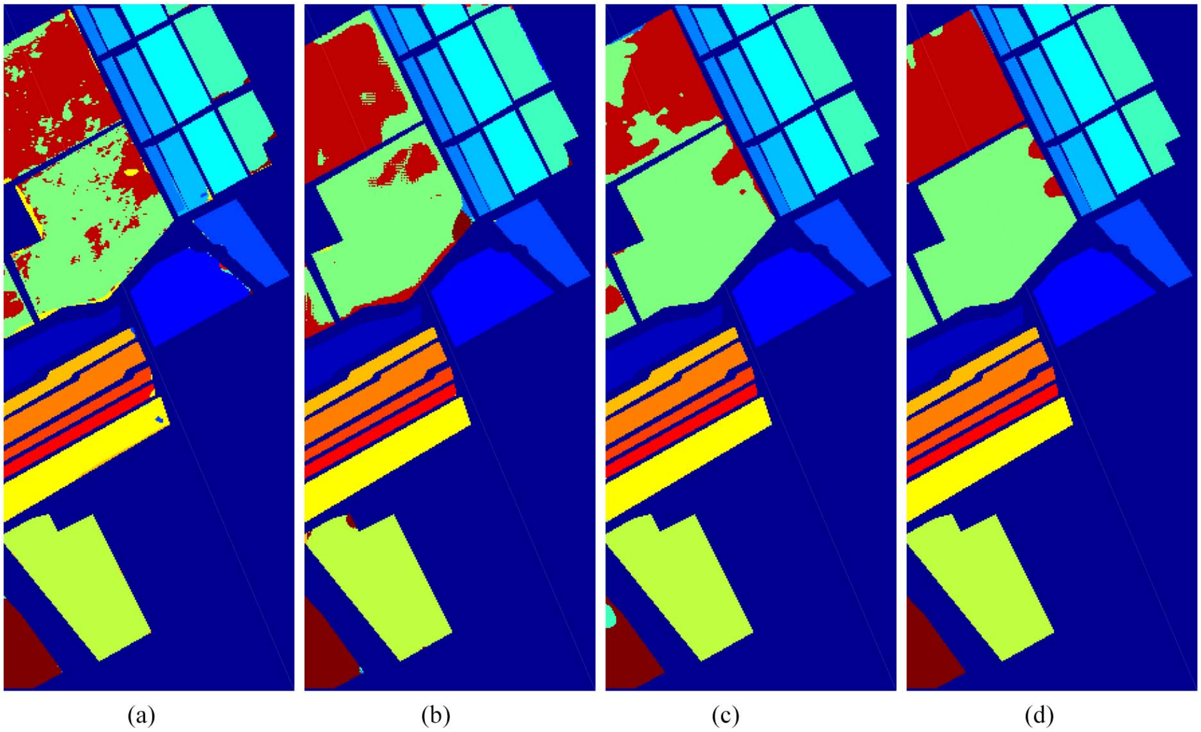


Fig. 8. Classification maps of different methods using 30 training samples per class on the Salinas dataset. (a) CNN-DR (92.77%). (b) CNN-Gabor (93.91%). (c) CNN-Capsule (95.11%). (d) Proposed DGEF (99.21%).

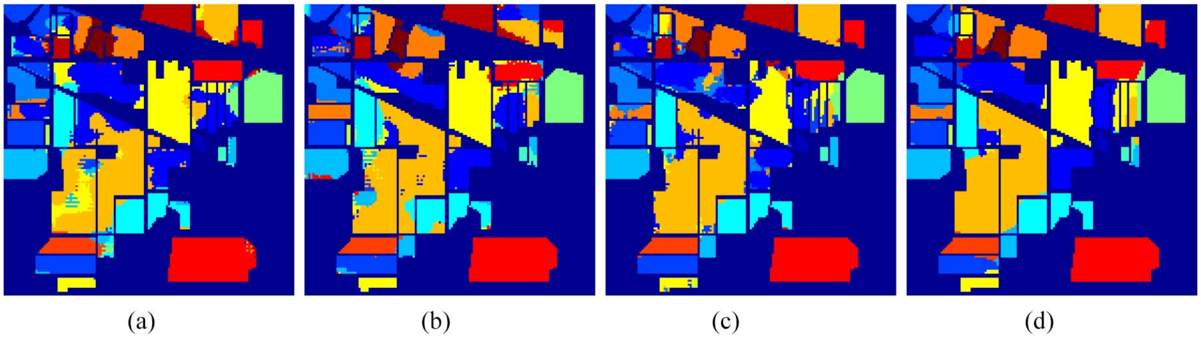


Fig. 9. Classification maps of different methods using 30 training samples per class on the Indian Pines dataset. (a) CNN-DR (77.89%). (b) CNN-Gabor (86.52%). (c) CNN-Capsule (88.29%). (d) Proposed DGEF (93.44%).

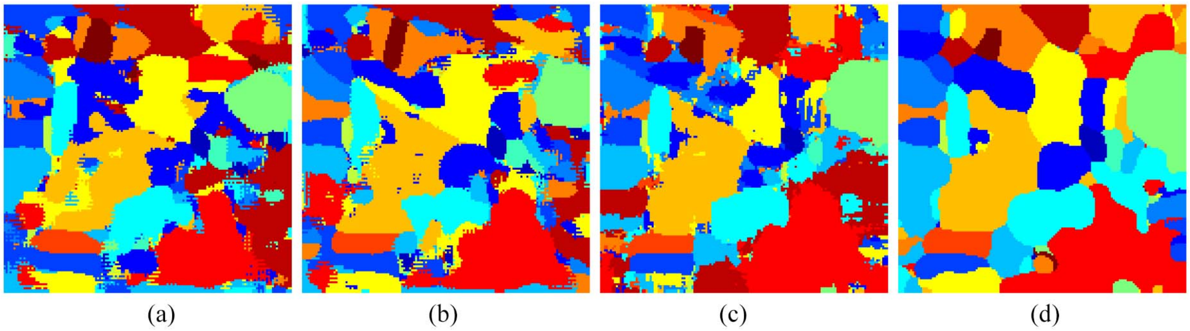
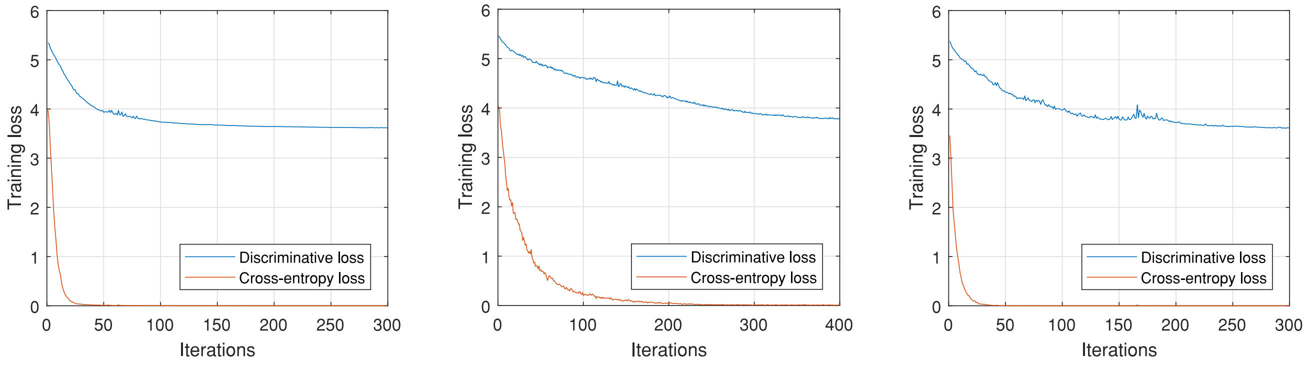


Fig. 10. Full classification maps (with background) of different methods using 30 training samples per class on the Indian Pines dataset. (a) CNN-DR. (b) CNN-Gabor. (c) CNN-Capsule. (d) Proposed DGEF.

the Gabor feature can obtain much higher accuracy than the baseline method, which shows the superiority of the Gabor feature for HSI classification. Gabor-NRS and lowrank-Gabor significantly improve the accuracy compared with the Gabor feature. Gabor-CNN obtains better performance, but not good enough. GFDN achieves good results on the Indian Pines and



(a) (b) (c)

Fig. 11. Training loss versus iterations for the proposed method on three datasets. (a) Salinas. (b) Houston. (c) Indian Pines.

TABLE X
CONTRIBUTION OF EACH COMPONENT OF THE PROPOSED METHOD

Component	Salinas	Houston	Indian Pines
Baseline	86.55±0.83	87.52±0.66	67.05±1.32
GEF1+1NN	96.12±0.43	96.48±0.42	95.20±0.52
Without learnable filters in GEF2	99.37±0.17	98.59±0.30	99.01±0.27
DGEF	**99.84**±0.07	**98.82**±0.32	**99.28**±0.25

Salinas datasets, but is not good on the Houston dataset. The proposed DGEF method attains the best results for all of the datasets, which demonstrates its effectiveness.

D. Analysis of the Proposed Method

In this section, we analyze the influence of the parameters for the proposed method, and then select the proper parameters as above experiments. The training and testing samples are the same as the previous section.

1) Contribution of Each Component of the Proposed Method: We first analyze the contribution of each component of the proposed method, Table X shows the results. We can find that GEF1 attains significantly better performance than the baseline method, which shows that the Gabor-based method can well extract the spatial features for HSI, but the performance is not good enough only using GEF1. When we remove the learnable filters in GEF2 to retrain the network where the others are the same as the proposed method, that is, without the learnable filters in GEF2, the performance will decrease, which shows the learnable filters in GEF can learn some complementary features that Gabor filters cannot extract. When we use all components of the proposed method, that is, the proposed GEF, network architecture, and the loss function, the accuracy is the best, which proves that each component of the proposed method makes a contribution for the good results.

2) Comparison of Different Dimensionality Reduction Methods: Here, we give a comparison different dimensionality reduction methods, including without any dimensionality reduction method (None), scalable one-pass self-representation learning (SOPSRF) [55], adaptive subspace partition strategy (ASPS) [56], optimal neighborhood reconstruction (ONR) [57], and PCA. All methods reduce the dimensions to 20 before feeding to the network. Fig. 12 shows

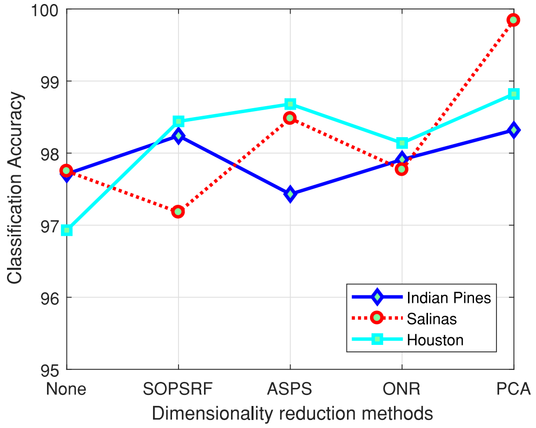


Fig. 12. Comparison of different dimensionality reduction methods. Using PCA as the initialization method can obtain better performance for the proposed method.

the results. When there is no any dimensionality reduction method (None), the performance is worse than PCA. It may be because when there are hundreds of channels as the input for the network, there are too many learnable parameters that are hard to train with limited samples. SOPSRF improves the accuracy on the Indian Pines and Houston datasets, except on the Salinas dataset. ASPS attains better accuracy on the Indian Pines and Salinas datasets, except on the Houston dataset. ONR raises the performance on three datasets, but is not good enough. PCA can obtain the best performance, especially on the Salinas dataset. So, we use PCA to reduce the dimension of HSIs.

3) Comparison of Different Patch Sizes: Fig. 13 shows the classification accuracies versus patch sizes for input cubes to CNN. We can find that with the increase of the patch size, the accuracy tends to grow on the Salinas dataset, while it tends to decrease on the Houston dataset. So, we select 17 × 17 as the patch size on the Indian Pines and Salinas datasets, and select 9 × 9 on the Houston dataset. When the input patch is 9 × 9, it does not need to apply the max-pooling layer for the proposed method.

4) Comparison of Different Weights for Discriminative Loss: Fig. 14 shows the classification accuracies versus weights for discriminative loss, where the weight for the cross-entropy loss is set to 1. We can find that the accuracy is

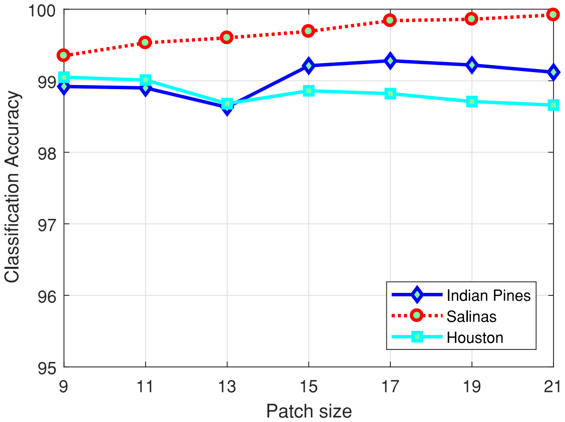


Fig. 13. Comparison of different patch sizes for input cube.

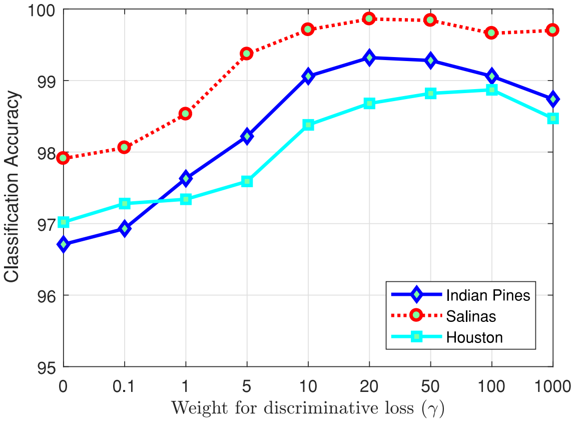


Fig. 14. Comparison of different weights for discriminative loss.

not good when we do not use the discriminative loss, that is, the weight for a discriminative loss γ is 0. When the weight tends to infinity, the performance declines. We can select an appropriate weight between 20 and 50 to obtain good enough accuracy. This shows that the proposed loss combining the cross-entropy and discriminative loss can improve the accuracy.

5) Comparison of Different Thresholds in $\mathcal{L}_d$: Fig. 15 shows the classification accuracies versus thresholds α in $\mathcal{L}_d$. α is a margin to filter trivial. If $\alpha = 0$, that is, as long as the distance of the hardest positive pair is less than that of the hardest negative pair, we will not use the sample to calculate the loss. In this situation, the accuracy is not good, because the distance between positive and negative pairs is not large enough. On the contrary, when α is too big, the performance cannot increase. We can select an appropriate weight between 2 and 8 to obtain good enough accuracy.

Comparison of different numbers of principal components and comparison of different numbers of Gabor filters can be found in the supplementary material.

IV. Conclusion

In this article, we have proposed the GEF, which filters each input channel by fixed Gabor filters together with learnable filters, followed by some learnable 1×1 filters to generate the output channels. Based on the proposed GEF, we designed a network architecture for HSI classification. To learn more discriminative features and an end-to-end system at the same time, we proposed to introduce the local discriminant structure for cross-entropy loss by combining the triplet hard loss. With limited training samples, the proposed method performs significantly better than other state-of-the-art HSI classification methods. Moreover, the proposed method is fast for both training and testing.

However, the proposed method cannot obtain higher accuracy for normal image classification, because the input for the proposed network is a small image patch with many channels, which is only designed for HSI classification with limited training samples. For normal image classification, an image is a sample. When there are millions of training samples, the network should be much more complex and deeper to obtain better performance. In our future work, we will investigate how to improve the proposed method for the normal image classification task, and how to combine other conventional features with CNN to obtain better performance.

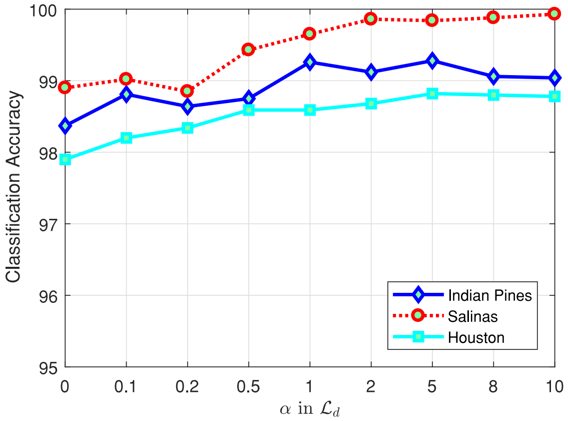


Fig. 15. Comparison of different thresholds in $\mathcal{L}_d$.

Acknowledgment

The authors thank Prof. Jun Li for preprocessing the experimental data and her valuable comments. They thank the authors of the compared methods for providing source codes. They also thank the Computational Intelligence Group from the Basque University for providing the Indian Pines and Salinas datasets, and the HSI Analysis Group and the NSF Funded Center for Airborne Laser Mapping at the University of Houston for providing the Houston dataset.

References

[1] M. Paoletti, J. Haut, J. Plaza, and A. Plaza, “Deep learning classifiers for hyperspectral imaging: A review,” *ISPRS J. Photogrammetry Remote Sens.*, vol. 158, pp. 279–317, Dec. 2019.

[2] W. Li and Q. Du, “Laplacian regularized collaborative graph for discriminant analysis of hyperspectral imagery,” *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 12, pp. 7066–7076, Dec. 2016.

[3] F. Feng, W. Li, Q. Du, and B. Zhang, “Dimensionality reduction of hyperspectral image with graph-based discriminant analysis considering spectral similarity,” *Remote Sens.*, vol. 9, no. 4, p. 323, 2017.

[4] L. Pan, H. C. Li, Y. J. Deng, F. Zhang, X. D. Chen, and Q. Du, “Hyperspectral dimensionality reduction by tensor sparse and low-rank graph-based discriminant analysis,” *Remote Sens.*, vol. 9, no. 5, p. 452, 2017.

[5] K.-K. Huang, D.-Q. Dai, and C.-X. Ren, "Regularized coplanar discriminant analysis for dimensionality reduction," *Pattern Recognit.*, vol. 62, no. 2, pp. 87–98, 2017.

[6] F. Luo, B. Du, L. Zhang, L. Zhang, and D. Tao, "Feature learning using spatial–spectral hypergraph discriminant analysis for hyperspectral image," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2406–2419, Jul. 2019.

[7] H. Huang, G. Shi, H. He, Y. Duan, and F. Luo, "Dimensionality reduction of hyperspectral imagery based on spatial–spectral manifold learning," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2604–2616, Jun. 2020.

[8] M. Wang, Y. Wan, Z. Ye, and X. Lai, "Remote sensing image classification based on the optimal support vector machine and modified binary coded ant colony optimization algorithm," *Inf. Sci.*, vol. 402, pp. 50–68, Sep. 2017.

[9] W. Li, Q. Du, F. Zhang, and W. Hu, "Collaborative-representation-based nearest neighbor classifier for hyperspectral imagery," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 2, pp. 389–393, Feb. 2015.

[10] Y. Yuan, J. Lin, and Q. Wang, "Hyperspectral image classification via multitask joint sparse representation and stepwise mrf optimization," *IEEE Trans. Cybern.*, vol. 46, no. 12, p. 2966, Dec. 2016.

[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] M. D. Mura, A. Villa, J. A. Benediktsson, J. Chanussot, and L. Bruzzone, "Classification of hyperspectral images by using extended morphological attribute profiles and independent component analysis," *IEEE Geosci. Remote Sens. Lett.*, vol. 8, no. 3, pp. 542–546, May 2011.

[13] D. Hong, X. Wu, P. Ghamisi, J. Chanussot, N. Yokoya, and X. X. Zhu, "Invariant attribute profiles: A spatial-frequency joint feature extractor for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 6, pp. 3791–3808, Jun. 2020.

[14] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1995, pp. 255–258.

[15] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[17] 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.

[18] X. Ma, H. Wang, and J. Geng, "Spectral–spatial classification of hyperspectral image based on deep auto-encoder," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens*, vol. 9, no. 9, pp. 4073–4085, Sep. 2016.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Rep.*, 2015, pp. 1–13.

[21] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 1–9.

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

[23] 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.

[24] 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.

[25] M. E. Paoletti, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. J. Plaza, and F. Pla, "Deep pyramidal residual networks for spectral-spatial hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 740–754, Feb. 2019.

[26] 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.

[27] M. Zhang, W. Li, and Q. Du, "Diverse region-based CNN for hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 2623–2634, Jun. 2018.

[28] 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.

[29] 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.

[30] Y. Xu, L. Zhang, B. Du, and F. Zhang, "Spectral–spatial unified networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 10, pp. 5893–5909, Oct. 2018.

[31] Y. Xu, B. Du, F. Zhang, and L. Zhang, "Hyperspectral image classification via a random patches network," *ISPRS J. Photogrammetry Remote Sens.*, vol. 142, pp. 344–357, Aug. 2018.

[32] K. Zhu, Y. Chen, X. Ghamisi, P. Jia, and J. A. Benediktsson, "Deep convolutional capsule network for hyperspectral image spectral and spectral-spatial classification," *Remote Sens.*, vol. 11, no. 3, p. 233, 2019.

[33] M. Zhang, W. Li, Q. Du, L. Gao, and B. Zhang, "Feature extraction for classification of hyperspectral and lidar data using patch-to-patch CNN," *IEEE Trans. Cybern.*, vol. 50, no. 1, pp. 100–111, Jan. 2020.

[34] X. Tang *et al.*, "Hyperspectral image classification based on 3-D octave convolution with spatial–spectral attention network," *IEEE Trans. Geosci. Remote Sens.*, early access, Jul. 14, 2020, doi: 10.1109/TGRS.2020.3005431.

[35] 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.

[36] B. Liu *et al.*, "Morphological attribute profile cube and deep random forest for small sample classification of hyperspectral image," *IEEE Access*, vol. 8, pp. 117096–117108, 2020.

[37] 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, 2020.

[38] C.-X. Ren, D.-Q. Dai, X.-X. Li, and Z.-R. Lai, "Band-reweighed Gabor kernel embedding for face image representation and recognition," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 725–740, Feb. 2014.

[39] C.-X. Ren, D.-Q. Dai, K.-K. Huang, and Z. Lai, "Transfer learning of structured representation for face recognition," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5440–5454, Dec. 2014.

[40] T.-C. Bau, S. Sarkar, and G. Healey, "Hyperspectral region classification using a three-dimensional Gabor filterbank," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 9, pp. 3457–3464, Sep. 2010.

[41] L. Shen and S. Jia, "Three-dimensional Gabor wavelets for pixel-based hyperspectral imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 12, pp. 5039–5046, Dec. 2011.

[42] W. Li and Q. Du, "Gabor-filtering-based nearest regularized subspace for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 4, pp. 1012–1022, Apr. 2014.

[43] Y. T. Yuan, L. Yang, and H. Yuan, "Hyperspectral image classification based on three-dimensional scattering wavelet transform," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 5, pp. 2467–2480, May 2015.

[44] L. He, J. Li, A. Plaza, and Y. Li, "Discriminative low-rank Gabor filtering for spectral–spatial hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 3, pp. 1381–1395, Mar. 2017.

[45] S. Jia, L. Shen, J. Zhu, and Q. Li, "A 3-D Gabor phase-based coding and matching framework for hyperspectral imagery classification," *IEEE Trans. Cybern.*, vol. 48, no. 4, pp. 1176–1188, Apr. 2018.

[46] Y. Chen, L. Zhu, P. Ghamisi, X. Jia, G. Li, and L. Tang, "Hyperspectral images classification with Gabor filtering and convolutional neural network," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 12, pp. 2355–2359, Dec. 2017.

[47] X. Kang, C. Li, S. Li, and H. Lin, "Classification of hyperspectral images by Gabor filtering based deep network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens*, vol. 11, no. 4, pp. 1166–1178, Apr. 2018.

[48] S. Luan, C. Chen, B. Zhang, J. Han, and J. Liu, "Gabor convolutional networks," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4357–4366, Sep. 2018.

[49] C. Liu, J. Li, L. He, A. Plaza, S. Li, and B. Li, "Naive Gabor networks for hyperspectral image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 376–390, Jan. 2021.

[50] R. Hadsell, S. Chopra, and Y. Lecun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2006, pp. 1735–1742.

[51] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," 2015. [Online]. Available: arXiv:1503.03832.

[52] A. Hermans, L. Beyer, and B. Leibe. (2017). *In Defense of the Triplet Loss for Person Re-Identification*. [Online]. Available: http://arxiv.org/abs/1703.07737

[53] Q. Wang, J. Lin, and Y. Yuan, "Salient band selection for hyperspectral image classification via manifold ranking," *IEEE Trans. Neural Netw.*, vol. 27, no. 6, pp. 1279–1289, Jun. 2016.

[54] J. Feng *et al.*, "Convolutional neural network based on bandwise-independent convolution and hard thresholding for hyperspectral band selection," *IEEE Trans. Cybern.*, early access, Jun. 29, 2020, doi: 10.1109/TCYB.2020.3000725.

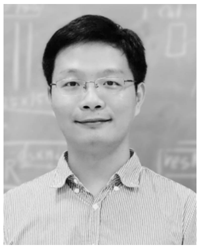
[55] X. Wei, W. Zhu, B. Liao, and L. Cai, "Scalable one-pass self-representation learning for hyperspectral band selection," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4360–4374, Jul. 2019.

[56] Q. Wang, Q. Li, and X. Li, "Hyperspectral band selection via adaptive subspace partition strategy," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 12, pp. 4940–4950, Dec. 2019.

[57] Q. Wang, F. Zhang, and X. Li, "Hyperspectral band selection via optimal neighborhood reconstruction," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 12, pp. 8465–8476, Dec. 2020.

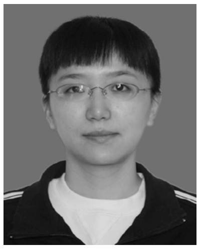
Ke-Kun Huang (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in applied mathematics from Sun Yat-sen University, Guangzhou, China, in 2002, 2005, and 2016, respectively.

He is currently a Professor with the Department of Mathematics, Jiaying University, Meizhou, China. He has published over 20 papers in top journals, including IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON CYBERNETICS, and *Pattern Recognition*. He was elected as a Nanyue Excellent Teacher of Guangdong province. His research interests include image processing and pattern recognition.



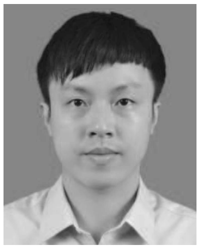
Chuan-Xian Ren (Member, IEEE) received the Ph.D. degree from Sun Yat-sen University, Guangzhou, China, in 2010.

From 2010 to 2011, he was with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, as a Senior Research Associate. He is currently an Associate Professor with the School of Mathematics, Sun Yat-sen University. He was elected a candidate for the "Thousand-Hundred-Ten" Talents Program of Guangdong Province in 2014. His research interests include pattern recognition, computer vision, and statistical learning.



Hui Liu received the B.Sc. and M.Sc. degrees in applied mathematics from South China Normal University, Guangzhou, China, in 2000 and 2007, respectively.

She is currently an Associate Professor with the Department of Mathematics, Jiaying University, Meizhou, China. Her research interests include fractal geometry and image processing.



Zhao-Rong Lai (Member, IEEE) received the B.Sc. degree in applied mathematics, the M.Sc. degree in information and computational science, and the Ph.D. degree in statistics from the School of Mathematics, Sun Yat-sen University, Guangzhou, China, in 2010, 2012, and 2015, respectively.

He is currently an Associate Professor with the Department of Mathematics, College of Information Science and Technology, Jinan University, Guangzhou. His research interests include machine learning, mathematical programming, and mathematical finance.

Dr. Lai is an Invited Senior Program Committee Member for IJCAI-20 and IJCAI-21, and a Program Committee Member for IJCAI-18, IJCAI-19, AAAI-19, and AAAI-20. He is also an invited reviewer for IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING and IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.

Yu-Feng Yu (Member, IEEE) received the B.Sc. degree in mathematics from Shangrao Normal University, Shangrao, China, in 2011, the M.Sc. degree in mathematics from Shenzhen University, Shenzhen, China, in 2014, and the Ph.D. degree in statistics from Sun Yat-sen University, Guangzhou, China, in 2017.

He is currently a Postdoctoral Fellow with the Department of Computer and Information Science, University of Macau, Macau, China. His research interests include statistical optimization and pattern recognition.

Dao-Qing Dai (Senior Member, IEEE) received the B.Sc. degree in mathematics from Hunan Normal University, Changsha, China, in 1983, the M.Sc. degree in mathematics from Sun Yat-sen University, Guangzhou, China, in 1986, and the Ph.D. degree in mathematics from Wuhan University, Wuhan, China, in 1990.

From 1998 to 1999, he was an Alexander von Humboldt Research Fellow with Free University, Berlin, Germany. He is currently a Professor with the School of Mathematics, Sun Yat-sen University. He is an author or coauthor of over 100 refereed technical papers. His current research interests include image processing, wavelet analysis, face recognition, and bioinformatics.