Structure Aware Generative Adversarial Networks for Hyperspectral Image Classification

Tayeb Alipour-Fard and Hossein Arefi

***Abstract*—Generative adversarial networks (GANs) have shown striking performances in computer vision applications to augment virtual training samples (VTS). However, the VTS generating by GANs in the context of hyperspectral image classification suffer from structural inconsistency due to the insufficient number of training samples in order to learn high-order features from the discriminator. This work addresses the scarcity of training samples by designing a GAN, in which the performance of discriminator is improved to produce more structurally coherent VTS. In the proposed method, by splitting the discriminator into two parts, GAN undertakes two tasks: the main task is to learn to distinguish between real and fake samples, and the auxiliary task is to learn to distinguish structurally corrupted and real samples. With this setup, GAN will produce real-like VTS with a higher variation than conventional GAN. Furthermore, in order to reduce the computational cost, subspace-based dimension reduction was performed to obtain the dominant features around the training samples to generate meaningful patterns from the original ones to be used in the learning phase. Based on the experimental results on real, and well-known hyperspectral benchmark images, the proposed method improves the performance compared with GANs-related, and conventional data augmentation strategies.[1]**

***Index Terms*—Deep learning (DL), hyperspectral images (HSIs), convolutional neural network (CNN), generative adversarial networks (GANs), remote sensing.**

I. Introduction

HYPERSPECTRAL sensors provide valuable information of the surface of the earth including hundreds of image bands in visible and infrared regions of the electromagnetic spectrum at a certain spatial resolution. This rich cube of data provides an opportunity to detect and recognize different objects and land-cover types in hyperspectral images (HSIs). Image classification is one of the major tasks in which HSIs are used for information extraction purposes [1]. Some general challenges are still to be addressed in the area of HSI classification, such as the high dimensionality of the data, the problem related to the relatively small sample size (SSS), the correlation of spectral signatures among different objects in the desired scene, and the use of spatial information in the classification process [1]–[3]. Deep learning (DL), as a novel machine learning paradigm, has achieved state-of-the-art performance in many applications such as object recognition [4], handwritten digit recognition [5], natural language processing [6], as well as image classification and segmentation [7], [8].

Among different neural network architectures, convolutional neural networks (CNNs) have shown notable efficiency in image processing, due to both their powerful learning strategy and their capabilities to models contextual information [9]–[11]. In HSI processing, CNNs are able to extract meaningful features automatically by training data and performing an end-to-end classification task. Accordingly, users are not required to manually select the relevant features for the addressed problem [12]. However, despite the many advantages associated with CNNs, there are still several critical problems in their use with hyperspectral data, especially with respect to the parameters to be estimated in the learning phase of the CNN architecture. A large number of training data is needed to achieve appropriate estimations and thus, robust classification results in models such as GoogleNet and VGGNet [13].

In recent years, several methods have been proposed to overcome the scarcity problem of training data for training CNNs in the context of HSI classification [14]–[18]. The first category includes semisupervised methods. These methods seek to increase the number of training samples by using clustering techniques, segmentation, and calculating spatial–spectral similarity to unlabeled data. Ma *et al.* in [17] presented a two-step approach to extend the number and the representativeness of training samples. First, the method takes two types of decisions to prelabel each unlabeled sample: local decisions based on weighted neighborhood information are made by the surrounding samples, and global decisions based on DL are taken by the most similar training samples. Next, some unlabeled samples having high labeling confidence are selected to extend the training set. Wu and Prasad in [18] trained a deep convolutional recurrent neural network (CRNN) with training data generated from the clustering of HSIs and then performed fine-tuning with limited labeled training data to obtain the final classification map. These methods are computationally expensive and the design of suitable criteria for sharing of unlabeled data for classification is very complex. Cotraining is another important semisupervised method [19]. In the cotraining methods, two classifiers are trained separately based on two feature subsets (views), and then unlabeled data are selected with the most credible predicted labels as the new labeled data. Ju *et al.* in [19] applied two classifiers (CNN and CRNN) individually to HSIs. Next, credible unlabeled data from each classifier were selected and a deeper architecture

Manuscript received June 21, 2020; revised August 12, 2020 and September 1, 2020; accepted September 4, 2020. Date of publication September 2, 2020; date of current version September 24, 2020. *(Corresponding author: Hossein Arefi.)*

The authors are with the School of Surveying and Geospatial Engineering, University of Tehran, Tehran 1417466191, Iran (e-mail: tayebalipour@ut.ac.ir; hossein.arefi@ut.ac.ir).

Digital Object Identifier 10.1109/JSTARS.2020.3022781

[1]Source code: https://github.com/TayebAlipour/SAGAN-HSI

than the two primary networks was created and trained with the new set of training samples for the classification of HSIs. The main challenge of this method is the creation of two views of features that should be independent and provide classification accuracy acceptable. In order to address this challenge, Zhang *et al.* [20] created two views based on spectral and spatial characteristics of HSIs and were able to take thus advantage of the cotraining. However, this method did not exploit all the benefits of spectral-spatial classification. The second category of methods for addressing the issue of the number of training samples is data augmentation. The most popular method that has been shown as an effective practice for data augmentation is to perform traditional affine and elastic transformations. Here, the new training samples are generated from original training data by performing rotation or reflection, zooming in and out, shifting, applying distortions as well as changing the radiometric palette [15], [21]. However, this method does not provide any significant improvement in HSI classification, as it does not add any new information on the classes. It mainly works when it is required to generate a more robust statistics and in cases we need to balance training examples in different classes. The third category of methods is to perform pretrained models. A pretrained model is a model that is created to solve a different yet similar problem. Instead of building a model from scratch, a pretraining model can be used and its parameters can be fine-tuned by using a limited number of training data to solve the specific problem considered [22]. The size of pretraining datasets used in HSI classification is still very small and largely insufficient in comparison with those used in the computer vision community to train image classifiers. With these conditions, other solutions are the use of networks such as VGGNet and ResNet that are taught on other source images (non-HSIs) [23]. However, this strongly limits the capability to exploit the crucial spectral measurements of hyperspectral sensors. Moreover, in order to achieve acceptable accuracies, it is necessary to start the fine-tuning from lower layers that are still requiring a large number of training samples. The fourth and last category of methods is based on the use of ensemble learning [24]. Due to the high variance of CNN, boosting methods have been developed to increase the overall accuracy (OA) using an implementation of an ensemble of light CNNs working on band subset instead of implementing a deep CNN on all bands [24]. In addition to the above methods, techniques such as regularization, dropout, batch normalization, and weight decay have also been used to improve the conditions for dealing with SSS, which are now considered as part of a standard CNN implementation [25].

Recently, to overcome the SSS problem, generative adversarial networks (GANs) provide a unique way to train DL algorithms by creating training data from the existing training samples [14], [26]–[29]. Zhu *et al.* [14] pioneered to use GANs for hyperspectral data processing by proposing 1-D and 3-D GANs. However, they used large amounts of training data (e.g., about 50% of the whole image in the Indian Pines dataset as a training sample), and they ignored the discriminator's component in the cost function and made significant simplifications. They also did not provide any criteria for quality assessment of generated training samples. Zhong and Li [28] proposed an integration of GANs and probabilistic graphical models and adopted a conditional random field to refine the results of GANs for HSI classification. He *et al.* [29] presented a semisupervised GAN method by adding a supervised term to the standard cost function of GANs and using unlabeled pixels.

As introduced by Goodfellow *et al.* [30] for the first time, GANs consist of two competing models named as generator and discriminator. Generator takes noise as the input and generates samples. Discriminator receives the samples from both generator and training data, and distinguishes between the two sources. The discriminator is the leader in the development of the GAN. The main problem of GAN is the limited performance of discriminator during the training process. The function space from which the discriminators are selected significantly affects GANs' performance, as the discriminator contributes to the creation of quality virtual training samples (VTS) by passing a quality signal to the generator. The regular architectural choice for the discriminator is a CNN [31], [32]. Because of SSS problem in HSI, the discriminator is not readily amenable to learning higher-order features to preserve structural consistency in training samples. In other words, the considering of deep CNNs networks (more than four convolution layers) for the discriminator only increases the computational volume and does not have the required efficiency. This shortcoming led to generation of spatially corrupted VTS, reducing the accuracy in modeling the HSI classification problems. The inclusion of poor-quality VTS into the classification process significantly impacts on the effectiveness of the learning process.

In this article, in order to produce more realistic training samples with structurally coherent content, an auxiliary task is assigned to the discriminator, which is to identify real samples whose structure has been manually disassembled. To this end, the two regions that are randomly selected in the intermediate convolutional feature maps of sample are swapped, then the resulting modified feature map is transferred through the next consecutive layers of discriminator network. This process, along with the main objective of discriminator, which is the distinguishing of real and fake samples, acts as an auxiliary task. As a result, the discriminator has two losses: the normal loss and the auxiliary loss. By combining the two losses, the signal provided to the generator by discriminator is enhanced and the generator is directed to generate VTS with coherent structure. We term our method "structurally aware sample generation by generative adversarial network (SA-GAN)." It is worthy of mention that the SA-GAN is not a three-player minimax optimization game, but the two-player minimax assumption persists and the intermediate feature maps of discriminator are manipulated and modified. In general, the borders between the different classes in HSI are very little (in terms of labeled samples), so finding an optimal Nash equilibrium in original HSI feature space (original number of bands) is practically impossible. For this reason, the SA-GAN considers the use of subspace feature extraction to obtain optimal feature space and also dominant features in input patch for each class and accordingly produce VTS for each class. To the best of our knowledge, generation of VTS by GANs with a coherent structure in the HSI classification has not been addressed so far. The main contributions of this article are as follows: 1) generating structurally coherent VTS by GANs to overcome the lack of training data in HSIs by developing an auxiliary task for the conventional GANs; 2) implementing subspace dimension reduction (DR) methods to reduce the dimensionality of HSIs and generating high-quality real training sample, which are more compatible with the nature of SA-GAN. The proposed approach has been compared with some of the most recent and widely used classification methods on standard hyperspectral benchmark datasets. The final results demonstrate the effectiveness of the developed approach in terms of quantitative classification accuracy metrics and quality of generated maps.

The rest of this article is structured as follows. Related research works are discussed in Section II. Section III presents the detail of the proposed method. In Section IV, the experimental results are illustrated and discussed. Finally, Section V draws the conclusion of this article.

II. Related Works

The number of parameters to be estimated in CNNs is much higher than the number of training samples. For this reason, a successful implementation of CNNs for HSI requires a large number of training samples. Given the fact that the collection of real training samples is very time consuming and costly, research studies have led to new ways of generating VTS. These methods attempt to generate VTS and increase the proportion of number of training samples compared to the number of CNN unknown parameters. Also, the enrichment of training samples increases the CNN generalization [33]. GAN is one of the approaches that has gained a lot of attention in recent years [14], [26]–[29]. Discriminator and generator play a continuous game, where the generator is learned to produce more and more realistic samples, and the discriminator is learned to get better and better at distinguishing the generated data from real data. The most critical challenge of developing the GAN framework is to produce samples that are realistic as possible [31]. In the recent study, the authors suggested to use collaborating learning and attention mechanism to assists the generator to provide real VTS [34]. Gao *et al.* [35] proposed using multidiscriminator with scoring mechanism rather than single discriminator to overcome the problem of insufficient diversity of generated samples. A training sample generated by the generator might partially be real and partially be fake. However, in the conventional GANs, this generated sample can only take one (real) or zero (fake) label, thus reducing the accuracy in modeling the HSI classification problems. This is critical, because in a training sample (a window around a labeled pixel) in HSI there may be several different classes. Feng *et al.* in [36] proposed the multiclass spatial–spectral GAN (MSGAN) to overcome this problem. In MSGAN two generators are proposed to produce spectral vector (by 1-D transpose convolution operator) and spatial patches (by 2-D transpose convolution operator) of HSIs, and a discriminator is designed to extract joint spatial–spectral features and output multiclass probabilities. Tao *et al.* [37] proposed a semisupervised variational GAN by taking advantages of decoder–encoder network to build a collaborative relationship between generator and classification task. Their proposed method led to the generation of real-like and meaningful training samples.

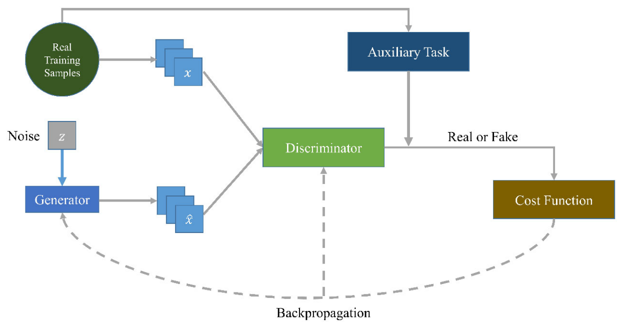


Fig. 1. General idea of multitask GANs.

The multitask GAN is proposed to force GAN to simultaneously learn different but related tasks, thus increasing the generalization power of their hidden representations [38]–[40]. The general idea is that if there is a certain relation between the tasks, the quality signals could be transferred to the generator from the discriminator (see Fig. 1). Auxiliary classifier GAN proposed by Mirza and Osindero [38] includes the class label as an auxiliary task in the first layer of the generator and the last layer of the discriminator. Chen *et al.* [39] added an auxiliary task to the GAN to predict correct rotation of input images by exploiting global structure in real data. The multitask cGAN is proposed to refine the digital surface model and, at the same time, produced roof-type classification maps from the high-resolution satellite images [40].

Outside of the GANs, VTS are generated by modifying the variance of the real training samples by applying random coefficients and linear and nonlinear combinations of them [41]. Adjusting the parameters and choosing the appropriate linear or nonlinear combination to control the uncertainty caused by the injection of random parameters is one of the challenges of this approach. Indeed, a slight change in parameters results in sharp changes in the classification accuracy. In addition, VTS are propagated by applying geometric operators such as rotation in different directions and changing the scale on real training samples [21]. One of the important parameters of this method is choosing the optimal input patch size. If the input patch size is too small, the accuracy of the CNN algorithm is reduced, whereas if the size of the input patch increases, the spatial quality of the final classification map decreases and the classes with small structure (such as buildings in urban areas) are merged into other classes.

Another relevant challenge to design CNNs and GANs is to adopt an appropriate DR approach. Although it has been claimed that CNNs are being utilized end to end, when considering HSI there is consensus in the literature on the fact that DR is necessary [12], [41]–[44], also taking into account the impossibility of have millions of labeled data for the training phase. Moreover, the high dimensionality of HSI gives rise to computational challenges in spite of the advantages in distinguishing similar spectral signatures. Chen *et al.* in [41] used the principal component analysis (PCA) method to reduce the dimension of

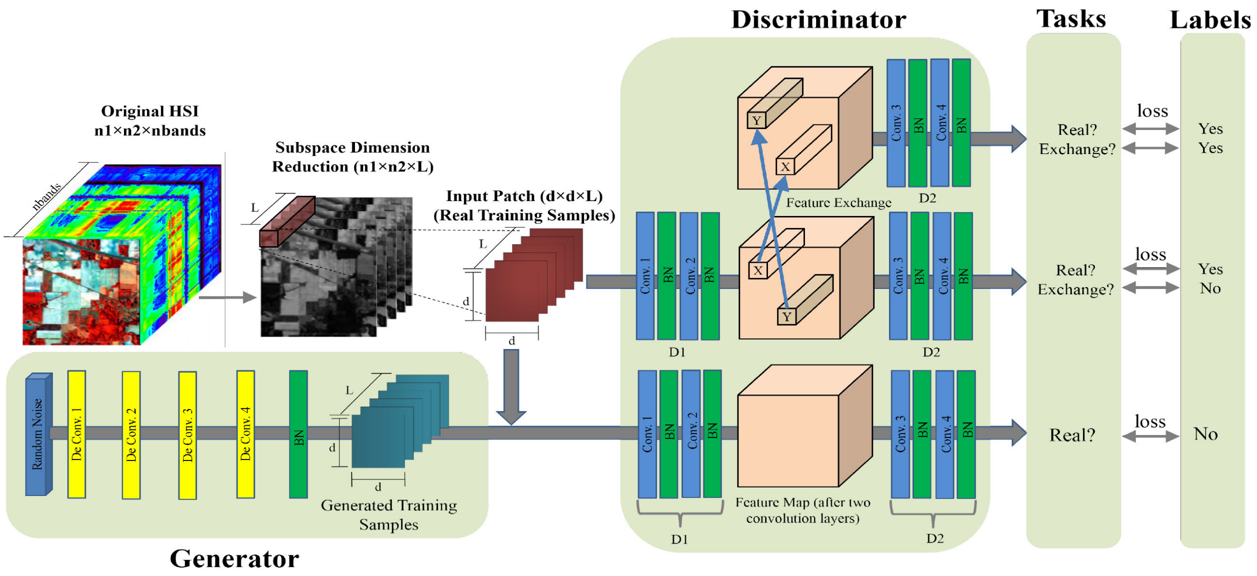


Fig. 2. General flow of the proposed approach (SA-GAN) for VTS generation for HSIs (L: number of subspaces in DR method, d: input patch size, DeConv.: deconvolution (upsampling) operator, BN: batch normalization, Conv.: convolution operator).

HSI. The authors evaluated the effectiveness of their proposed CNN method with the extracted features in three scenarios using only spectral information, only spatial information, as well as spectral–spatial information. Kernelized PCA method was used to reduce the dimension of HSI in the framework of unsupervised CNNs [44]. In a similar work, Zhao and Du [42] provided the PCA method with local marginal information classes to increase the capability to model for separation among classes. Moreover, this method requires a large amount of training samples and does not have the required performance against other supervised DR methods such as subspace-based DR [2]. Evolutionary-based DR methods like particle swarm optimization were also presented for DR [12]. However, they did not achieve significant accuracy improvement in HSI classification by CNNs. There are also some studies in which the CNNs have been implemented on the original images without using DR [45], [46]. The advantages of this naive approach are to maintain the end to end of the implementation process with CNNs. The problem is that a huge amount of training samples is needed.

III. Methodology

The structure of our proposed method (SA-GAN) is shown in Fig. 2. As we can see, the SA-GAN requires two main steps, including DR and designing the SA-GAN components (discriminator, generator, and loss function). The output of the DR method is an input to the SA-GAN step. Then, the set of training samples (i.e., the combination of real and virtual training samples) are the input to the CNN. Definition of the CNN architecture to produce final classification map is discussed in Section III-C. The details of the proposed method are given in the following sections.

A. Dimensionality Reduction

Hyperspectral sensors acquire images in hundreds of spectral bands. This results in the curse of dimensionality problem that

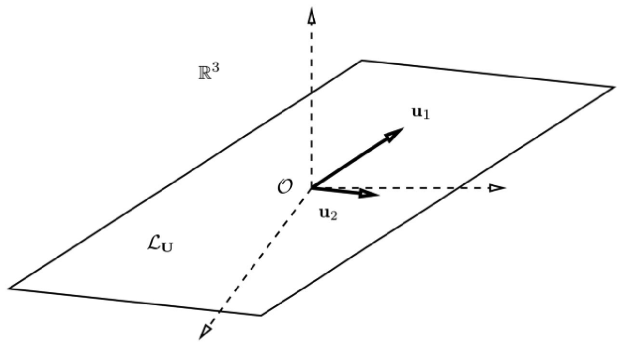


Fig. 3. 2-D subspace L defined from the 3-D space. L is defined by its basis vectors u_1 and u_2 [2].

is one of the main obstacles for automatic classification. In this context, the main contribution of this article is getting good generalization properties in the generated virtual samples to overcome this problem. As mentioned in Section II, the between classes distance in original HSI bands is very small and causes to mode collapse of GAN. For fighting mode collapse and finding an optimal Nash equilibrium, we need to have the discriminative subspace of original HSI and the high-quality real training sets made up of reliable samples as an input to the SA-GAN. To take advantage from an adaptive feature subspace for each class, we apply subspace DR to HSI. The use of subspaces to represent class models is based on the assumption that the vector distribution in each class lies approximately on a lower dimensional subspace of the feature space [2]. The most common way to define an l-dimensional subspace is to utilize a set of linearly independent basis vectors, $\{u_1, ..., u_l\}$, which can be combined into a $d \times l$ matrix U having rank l. Fig. 3 illustrates this concept with a 2-D subspace in a 3-D space. Let $U(c)$ be a set of $r^{(c)}$-dimensional orthonormal basis vectors for the subspace associated with classes $c = 1, 2, ...l$.

$U(c)$ is computed as $U(c) = \{e_1^{(c)}, ..., e_{r(c)}^{(c)}\}$, whereas $E(c) = \{e_1^{(c)}, ..., e_d^{(c)}\}$ is the eigenvector matrix computed from the correlation matrix $R(c) = E^{(c)} \Lambda E^{(c)'}$. Here, Λ is the eigenvalue matrix defined with decreasing magnitude and e is an eigenvector corresponding to the eigenvalue of correlation matrix $R(c)$. We use a subspace projection accounting for 99.9% of the original spectral information in order to determine the size of $U(c)$. The feature space is defined by the nonlinear functions $h(x_i) = [\|X_i\|^2, \|X_i' U^1\|^2, \ldots, \|X_i' U^k\|^2]'$, i.e., the feature vectors containing as components the energy of the projections on all class subspaces plus the energy of the original vector.

B. Structure Aware GAN

1) Review of GAN: GANs are inspired from the game between two players in which one player, the Generator (*G*), is responsible for producing training samples and the other one, the Discriminator (*D*), is to determine whether they are real or fake [30]. The GAN framework should define the discriminator, the generator, the architecture, and the training process. The definition of the architecture is the first and most important step for designing a GAN. The discriminator should be trained on real training samples of each class and be locked in advance. Then, the generator generates fake data from a random variable and this VTS feed into the discriminator to predict them. The training phases of these networks occur simultaneously and can be described as a minimax game. The discriminator tries to maximize its own performance in distinguishing between real and generated samples, whereas the generator maximizes its ability to generate samples that manage to fool the discriminator. What generator does is to estimate the density, from the noise to real data, and to provide it to the discriminator. Goodfellow *et al.* [30] suggest (1) as standard objective function for GAN

$$V(G, D) \coloneqq E_{x \sim P_{(x)}}[\log(D(x))] + E_{z \sim P_{(z)}}[\log(1 - D(G(z)))] \quad (1)$$

where $p_{(x)}$ is the data distribution, $z \in \mathbb{R}^{d_z}$ is a latent variable, $p_{(z)}$ is the standard normal distribution, and $D_{(x)}$ denotes the probability of x being sampled from $p_{(x)}$. After enough iterations, $p_{(z)}$ will converge to $p_{(x)}$, which indicates that *G* is capable of learning the true distribution of $p_{(x)}$. The solution to this problem is an equilibrium point of the game, which is a saddle point of the discriminator loss. Theoretically, Goodfellow *et al.* [30] proved the convergence of GAN training by assuming that the generator is always updated according to the temporarily optimal discriminator at each training step. Practically, this assumption is too difficult to satisfy and GANs remain difficult to train. A general solution is to meet the Lipschitz-continuity condition [47]. Weight clipping [48], gradient penalty [49], and spectral normalization [50] are the most important methods that are recently proposed. Another group of researchers believe that the main solution is to change the cost function [47]. Methods such NS-GAN, LS-GAN, and W-GAN are based on the change of the cost function and the minimization of statistical divergences between the model and target distributions.

TABLE I
NETWORK ARCHITECTURE FOR THE SA-GAN

Network	Dimension	Input Size	L_1	L_2	L_3	L_4
Discriminator	$k1$,$k2$	32	32	16	8	4
	Channels	L	64	128	256	512
Generator	$k1$,$k2$	4	8	16	32	32
	Channels	256	256	128	64	L

The number represents the tensor shapes after convolution (for the discriminator) and upsampling (for the generator).

2) Architecture of SA-GAN: Our proposed SA-GAN model attempts to incorporate an auxiliary task learning system into the GAN, which will allow the discriminator to identify inconsistency in input real training samples (by exchanging regions in the feature map) to enhance the quality of the VTS. As demonstrated in Fig. 2, the proposed discriminator is divided into two consecutive parts: $D1$ and $D2$. The feature map produced by $D1$ in the middle row of discriminators box is duplicated and the two regions (randomly selected) are exchanged in the feature map. To this end, we define the exchange operator. Suppose $\mathbf{X} \in \mathbb{R}^{W \times W \times d}$ denotes a feature map cube where W and d represent the spatial size of the cube and the number of features, respectively, and let $X_{(p,q)} \in \mathbb{R}^d$ represent the feature vector at spatial location (p, q). Then, the exchange operator $\xi(X, \delta)$, which inputs X and outputs a feature map cube that is identical to X except that its feature vector at some spatial location (i, j) is exchanged with that at location $\delta(i, j)$, where δ denotes a random permutation of the location index set

$$\xi(X, \delta)_{(p,q)} = \begin{cases} X_{(i,j)} & \text{if } (p, q) = \delta(i, j) \\ X_{\delta(i,j)} & \text{if } (p, q) = (i, j). \\ X_{(p,q)} & \text{otherwise} \end{cases} \quad (2)$$

Discriminator is expected to distinguish the real cubes from the structurally corrupted ones by exchange operator. Table I lists the architecture of the discriminator network, which is comprised of four convolution layers by giving the size of the cube in the spatial and channel dimensions. The input size of discriminator is $32 \times 32 \times L$, (L: number of subspaces from DR method, which vary according to each dataset). In Section IV-B2, the effect of input patch size has been investigated and we find the 32×32 is optimal input size.

For the discriminator, $D1$ is the primary two layers and $D2$ is the last two layers (see Fig. 2). The input to the D goes through $D1$ to produce a feature cube, on which an exchange operator may be used if the input was a real training sample. The cube is then passed through $D2$ to produce the discriminator output. The kernel size of four 2-D convolution layers is the same and padding operator is used to make the output spatial size unchanged. After each convolution operator, a max-pooling operator is added to reduce the spatial size by two times. Assume $X^{[L]}$, where $L \in \{1, 2, 3, 4\}$, is the output of the $[L]$th convolution layer, the computational process can be written as

$$X^{[L]} = \beta(W^{[L]} * X^{[L-1]} + b^{[L-1]}) \quad (3)$$

where $X^{[0]} = X$, $W^{[L]}$ is the $[L]$th convolution kernel, $*$ is the convolution operator, $b^{[L]}$ is the bias, and β is the batch normalization operator that normalizes the output cube between $[-1, 1]$.

For the generator, the input noise is first converted into a cube of $4 \times 4 \times 256$ elements, then passed through a sequence of deconvolution/upsampling filters to increase the spatial size. The objective of the generator module is to generate X with four upsampling layers. For the first upsampling layer, $X^{[4]}$ is first upsampled by two times in the spatial domain and then a 2-D convolutional operator is applied to it. The process of the convolutional operator is the same as (3) and the kernel size is set to 3×3 (see Table I).

The design of suitable loss functions is essential for GAN models. The idea of SA-GAN can be combined with a variety of existing GAN algorithm. In the context of HSI classification, due to the spectral mixing and very little border between different classes, finding the Nash equilibrium is very hard. Therefore, we use a margin [value "1" in (4) and (5)] to make the training more robust, as proposed in [50]

$$V_D(\hat{G}, D) = \tag{4}$$

$$E_{x\sim P_{(x)}}[\{\min(0, 1 + D(x))] + E_{z\sim P_{(z)}}[\min(0, 1 - D(\hat{G}(z)))]$$

$$V_G(G, \hat{D}) = -E_{z\sim P_{(z)}}[\hat{D}(G(z))]. \tag{5}$$

The discriminator's target is to produce a score less than -1 for the fake images and producing a score greater than 1 for the real images. The proposed method is designed to produce samples with consistent structure. To this end, for discriminator network, instead of one scalar output, we have two outputs. One is $D_{r/f}$, that is the standard discriminator output of recognizing whether its input is real or fake and the other is D_{sc} to recognize whether or not its inputs have been structurally changed. Based on two outputs, we have a new discriminator ($D^{'}$) and we can state the final objective of SA-GAN learning problem as

$$V_{D^{'}}(\hat{G}, D^{'}) = E_{x\sim P_{(x)}, z\sim P_{(z)}}[L_{r/f}(x, z) + \lambda_1 L_{sc}(x)] \tag{6}$$

$$V_G(G, \hat{D^{'}}) = -E_{z\sim P_{(z)}}[D_{r/f}(G(z))] \tag{7}$$

where λ_1 is a regularization parameter used to tradeoff two losses, i.e., $L_{r/f}$ and L_{sc}, which are defined as follows:

$$L_{sc}(x) = E_{x\sim P_{(x)}}[\min(0, 1 + D_{sc}(x))] \tag{8}$$
$$+ E_{x\sim P_{(x)}}[\min(0, 1 - D(\xi(x))]$$

$$L_{r/f}(x, z) = E_{z\sim P_{(z)}}[\min(0, 1 + D_{r/f}(G(z)))]$$
$$+ E_{x\sim P_{(x)}}[\min(0, 1 - D_{r/f}(x))]$$
$$+ \lambda_2 E_{x\sim P_{(x)}}[\min(0, 1 - D_{r/f}\xi(x))]. \tag{9}$$

The loss defined in (8) identifies whether the real input images are structurally corrupted. The loss $L_{r/f}$ in (9) is an extension of the loss in (4), but it differs in that we also include a term for the structurally corrupted real images (which are considered real). Also, empirically we found it beneficial to use weighting λ_2 for this new term. To take advantages of labeled training data, conditional GAN is utilized. Ideally, conditional GAN results in a generator that generates convincingly realistic data that corresponds to the input labels and a discriminator that has learned strong feature representations for each label.

C. Classification by CNN

The use of CNNs on HSI comprises two major challenges. The first challenge is to adopt an appropriate DR approach. Indeed, the curse of dimensionality results in decreasing the accuracy of the classification by increasing the number of bands. The second challenge belongs to the lack of training samples. The two above-mentioned challenges are addressed in Sections III-A and III-B2. Afterward, to obtain the classification map, the set of training samples (i.e., the combination of real and virtual training samples) are given to the CNN (see Fig. 4). A CNN is a sequence of layers, where each different layer plays a different role in the processing of input data by imposing a different function [4], [7]. In the present study, the main types of layers were employed to build the CNN architectures including convolution layer, rectified linear unit (ReLU) layer, pooling layer, dropout layer, and fully connected layer. The convolution layer aims to learn feature representations of the inputs. Convolution layer is composed of several convolution kernels (receptive fields), which are used to compute different feature maps. ReLU performs a $\max(\cdot)$ function between 0 and the input data. In comparison to the conventional activation functions, ReLU prunes the negative values to zero and retains the positive value in order to reduce the effect of the vanishing gradient. Robustness to noise and distortions are the main motivation of pooling the feature maps obtained by previous layers. Max pooling is used to get faster convergence during training with respect to other strategy like average pooling. Dropout is used for addressing the overfitting problem by randomly skipping some units or connections with a certain probability. This random drop generates different thin network architectures, and ultimately one network with small weights is chosen [51]. Following the dropout layer, a traditional fully connected layer is used to map the feature domain to the class domain. After max pooling (also dropout), the feature map should be unfolded and fully connected to each class. In HSI classification, an increase in the number of fully connected layers leads to a significant increase in the number of parameters [25]. Regarding a classification problem with more than two classes, the output unit activation function is the softmax function. The softmax output layer provides an estimation of the probabilities that the considered sample belongs to each class. Then a metric is necessary to measure the similarity between the output of the network (highest probability in the previous layer) and the label of each pixel to perform the backpropagation procedure. The loss function is the cross entropy function for 1-of-L classes. The classification layer is the last layer in CNNs and Adam optimization algorithm is used to update the parameters of the objective function in an iterative manner. Therefore, these layers were stacked up to form a complete architecture of CNNs in the present analysis. Furthermore, ReLU manipulates the transformation between the layers as an activation and dropout layer to reduces the effect of gradient vanishing and overfitting issues, respectively.

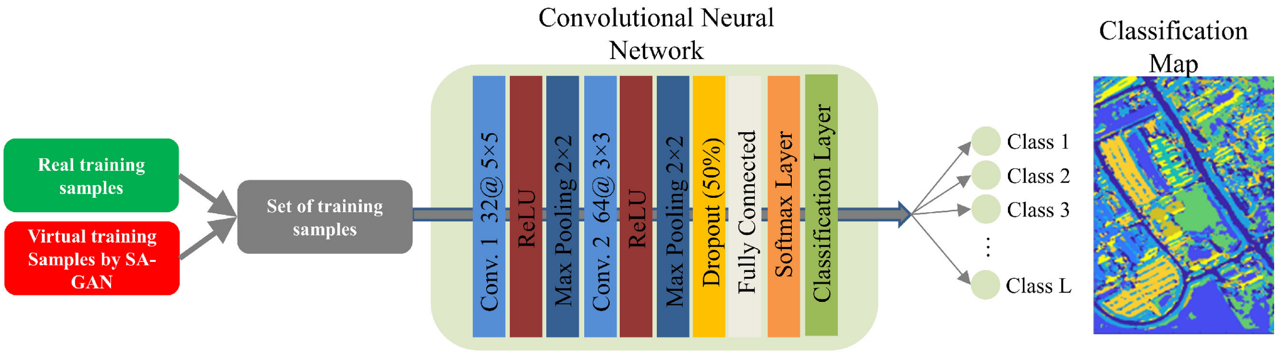


Fig. 4. Architecture of CNN for HSI classification after VTS generation by the proposed SA-GAN model. (Same convolution, stride=1, L: number of classes).

IV. Experimental Results and Discussion

In order to study the performance of the proposed model for HSI classification, an implementation has been developed and tested on a hardware environment with a Sixth Generation Intel Core i7-6800K processor with 6M of Cache, installed over an ASUS motherboard, 64 GB of DDR4 RAM. A graphic processing unit NVIDIA GeForce GTX 1080Ti with 8 GB RAM is also used. The proposed method was implemented over the Keras framework, with Ubuntu 18.04.2 x64 as operating system.

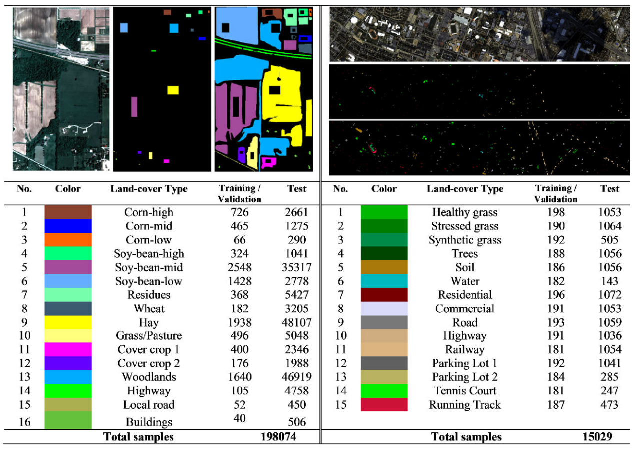
A. *Dataset Description*

The proposed method was applied to four well-known benchmark HSIs: 1) the Indian Pines 2010, 2) the Houston University 2012, 3) the Houston University 2017, and 4) the Pavia University datasets. Details of each dataset are provided in the following sections.

1) Indian Pines 2010 Dataset (IP-10): The first dataset is an image of an agricultural area obtained by the ProSpecTIR system over near Purdue University, Indiana, between 24th and 25th of May 2010. Its spatial resolution is 2 m and the spectral width of 5 nm. The spatial size of data is 445 × 750 × 360 and contains 16 classes. In total, 20 bands have been removed due to water absorption. Fig. 5(a) shows the RGB-composite and the available sample distribution in each class for training and testing.

2) Houston University 2012 Dataset (HU-12): Houston 2012 scene was collected by CASI on 2012 over the neighboring urban area of University of Houston campus. The dimension of this scene is 349 × 1905 × 144, containing 15 ground-truth classes and its spatial resolution is 2.5 m. In this sense, HU scene provides an interesting benchmark dataset. This scene was first presented by the IEEE Geoscience and Remote Sensing Society Image Analysis and Data Fusion Technical Committee during the 2013 data fusion contest [52]. Fig. 5(b) shows the RGB-composite and the available sample distribution in each class for training and testing.

3) Houston University 2017 Dataset (HU-17): This dataset was acquired on 2017 by the hyperspectral imager CASI 1500 over the area of the University of Houston [53]. It has a larger spatial size (1 m) but fewer spectral bands compared with the data from Houston 2012 dataset. The dimension of this scene is



No.	Color	Land-cover Type	Training / Validation	Test
1	🟫	Corn-high	726	2661
2	🟦	Corn-mid	465	1275
3	🟧	Corn-low	66	290
4	🟩	Soy-bean-high	324	1041
5	🟪	Soy-bean-mid	2548	35317
6	🟦	Soy-bean-low	1428	2778
7	🟩	Residues	368	5427
8	⬛	Wheat	182	3205
9	🟨	Hay	1938	48107
10	🟨	Grass/Pasture	496	5048
11	🟪	Cover crop 1	400	2346
12	🟪	Cover crop 2	176	1988
13	🟦	Woodlands	1640	46919
14	🟩	Highway	105	4758
15	🟫	Local road	52	450
16	🟩	Buildings	40	506
		Total samples	198074	

(a)

No.	Color	Land-cover Type	Training / Validation	Test
1	🟩	Healthy grass	198	1053
2	🟩	Stressed grass	190	1064
3	🟩	Synthetic grass	192	505
4	🟩	Trees	188	1056
5	🟫	Soil	186	1056
6	🟦	Water	182	143
7	🟥	Residential	196	1072
8	⬜	Commercial	191	1053
9	⬛	Road	193	1059
10	🟫	Highway	191	1036
11	🟫	Railway	181	1054
12	⬛	Parking Lot 1	192	1041
13	🟫	Parking Lot 2	184	285
14	🟩	Tennis Court	181	247
15	🟥	Running Track	187	473
		Total samples		15029

(b)

Fig. 5. Visualization and distribution of training/validation and test samples in the (a) Indian Pines 2010 and the (b) Houston University 2012 HSI datasets. The number of training/validation (top middle) and test samples (top right for IP-10 and top (bottom row) for HS-12) used for each class is shown in the training/validation and test column, respectively.

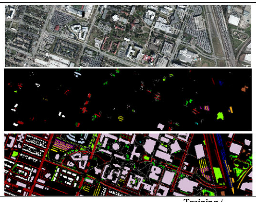
601 × 2384 × 48, containing 20 ground truth classes. Fig. 6(a) shows the RGB-composite and the available sample distribution in each class for training and testing.

4) Pavia University Dataset (PU): The Pavia University image was collected by reflective optics system imaging spectrometer (ROSIS) over the city of Pavia in Italy [54]. The ROSIS sensor captured an image with a spatial resolution of 1.3 m and acquired 115 spectral bands in the spectral range between 430 and 860 nm. The size of the image in pixels is 610 × 340, containing nine ground truth classes. Fig. 6(b) shows the RGB-composite and the available sample distribution in each class for training and testing.

B. *Analysis on the SA-GAN Model*

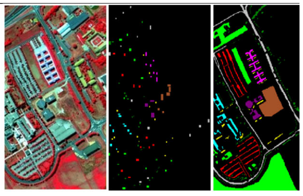
In order to study the analysis of the proposed method, several different experiments were carried out.

1) The first experiment focuses on the effect of two main hyperparameters of the SA-GAN (i.e., λ_1 and λ_2). In this experiment, the optimal value of hyperparameters for each dataset was determined.



No.	Color	Land-cover Type	Training / Validation	Test
1	🟩	Healthy grass	1458	8341
2	🟩	Stressed grass	4316	28186
3	🟩	Synthetic grass	331	353
4	🟩	Evergreen Trees	2005	11583
5	🟩	Deciduous Trees	676	4372
6	🟫	Soil	1757	2759
7	🟦	Water	147	119
8	⬜	Residential	3809	35953
9	🟪	Commercial	2789	220895
10	🟥	Road	3188	42622
11	▨	Sidewalk	2699	31303
12	▨	Crosswalk	225	1291
13	🟥	Major Thoroughfares	5193	41165
14	🟫	Highway	700	9149
15	🟧	Railway	1224	5713
16	🟨	Paved Parking Lot	1179	10296
17	🟧	Gravel Parking Lot	127	22
18	🟪	Cars	848	5730
19	🟦	Trains	493	4872
20	🟦	Seats	1313	5511
		Total samples	**504712**	

(a)



No.	Color	Land-cover Type	Training / Validation	Test
1	▨	Asphalt	548	6631
2	🟩	Meadows	540	18649
3	🟦	Gravel	392	2099
4	🟩	Trees	524	3064
5	🟪	Painted metal sheets	265	1345
6	🟫	Bare soil	532	5029
7	🟪	Bitumen	375	1330
8	🟥	Self-Blocking Bricks	514	3682
9	🟨	Shadows	231	947
		Total samples	**42776**	

(b)

Fig. 6. Visualization and distribution of training/validation and test samples in the (a) Houston University 2017 and the (b) Pavia University HSI datasets. The number of training/validation (top middle) and test samples (top (bottom row) for HS-17 and top right for PU) used for each class is shown in the training/validation and test column, respectively.

2) The second experiment focuses on the effect of input patch size on classification accuracy.
3) The third experiment focuses on the size of the region for the exchange operator. It is another hyperparameter of the SA-GAN, which is essential for generating virtual samples.
4) The fourth experiment investigates the visual quality of the VTS produced by the proposed method.
5) The fifth experiment analyzes the contribution of the DR and SA-GAN models to classification accuracy.
6) The sixth and last experiment focuses on classification accuracy obtained by SA-GAN compared to the cutting-edge methods related to VTS expansion and enrichment. Six CNN-based methods were selected from the literature to assess the performance of the proposed method: a) standard CNN framework to HSI classification (S-CNN) [55]; b) changing radiation-based virtual samples generation for HSI classification by CNN (CRVS-CNN) [41]; c) meta-heuristic dimension reduction method (MDR-CNN) [12]; d) standard GAN for producing virtual training samples (S-GAN) [14]; e) the Wasserstein GAN for HSI classification (W-GAN) [56]; and f) self-attention GAN for HSI classification (Self-GAN) the same as [57]. For all the comparison models, the best settings indicated in the relevant paper are used, except for the number of real training samples considered to be equal for a fair comparison. The performance of each classification procedure was measured by the OA, the average accuracy (AA), as well as the Kappa coefficient (K).

Fig. 2 and Table I visualize and report the proposed architecture for the generator and discriminator (which are the building blocks of SA-GAN). Table I lists the layers of G and D with the specific dimensions of each layer. The input of G is a noise vector of dimension drawn from a Gaussian distribution. The generator basically transforms this input vector into a $32 \times 32 \times L$ training image by passing through each layer in G. In the IP-10, HU-12, HU-17, and PU datasets, the L (the number of subspaces and also the number of classes in each dataset) is 16, 15, 20, and 9, respectively. The discriminator takes an image (an image from real training samples and an image from the generator) and calculates the outputs of (8) and (9) to recognize whether its input is real or fake and to recognize whether or not its inputs have been structurally changed. Prior to the aforementioned studies of GANs, the discriminator (D) and the generator (G) are optimized by Adam optimizer [58], [59]. The hyperparameters associated with the Adam optimizer are learning rate (lr), β_1, β_2, and ϵ, which are set to 0.001, 0.5, 0.9, and 0.1×10^{-7}, respectively. The number of the epochs to run the whole network for learning phase and the minibatch size are 400 and 32, respectively, and the input patches for SA-GAN are normalized into $[-1\ 1]$. These empirical settings are the same for all datasets, except for the HU-17 dataset, where the number of the epochs increased to 500 to meet the condition of the Adam optimizer.

1) Experiment 1. The Effects of Hyperparameters λ_1 and λ_2: There are two hyperparameters (regularization parameters) in the objective function of SA-GAN [(6) and (9)]. To investigate their effects on the classification performance, the parameter values are chosen from a grid of candidate set {0.0001, 0.001, 0.01, 0.1, 1} for both λ_1 and λ_2, then the optimal values of both parameters were obtained simultaneously. Fig. 7 shows the OAs of classification versus the change in the two parameters.

To evaluate the effect of each parameter separately, Fig. 8 demonstrates the OAs acquired by SA-GAN with different HSI datasets. When $\lambda_1 = 0.1$, SA-GAN can achieve the highest OA on the all datasets or the values close to the highest ones on the Indian Pines 2010. Consequently, it is reasonable to select 0.1 for λ_1. Similarly, OA follows the same trend in terms of λ_2. It is obvious to find that $\lambda_2 = 0.001$ is the best choice for the all datasets, if we disregard the small variations. One of the noteworthy findings is that sensitivity of IP-10 dataset [see Fig. 8(a)] to changing λ_1 is lower than the other three datasets. It is because the IP-10 dataset was collected from an agricultural area, whereas the other three datasets were collected from an urban area with features at various scales. If we focus on the results on IP-10 [Fig. 8(a)] and HU-17 [Fig. 8(c)] datasets, we can see that the effect of the λ_1 on the HU-17 dataset is significant because there are classes in different scales (small and big structures), so the producing structurally consistent VTS is essential and improves the efficiency of the model. However, the resulting form the IP-10 dataset, which is the scene in the agricultural area, is more like a hill (with soft changes) because there are fewer classes with different structures.

2) Experiment 2. The Effects of Input Patch Size: One of the parameters in the generation of training samples is the choice of the optimal dimension of the input patch (spatial size). This parameter must be determined in a way that it is large enough to have the information to make variation to the generated training samples. However, choosing relatively large dimensions increases the computational cost and causes to merge small structures such as buildings area in final classification map. We tested spatial sizes in the set $\{8 \times 8, 16 \times 16, 32 \times 32, 64 \times 64\}$

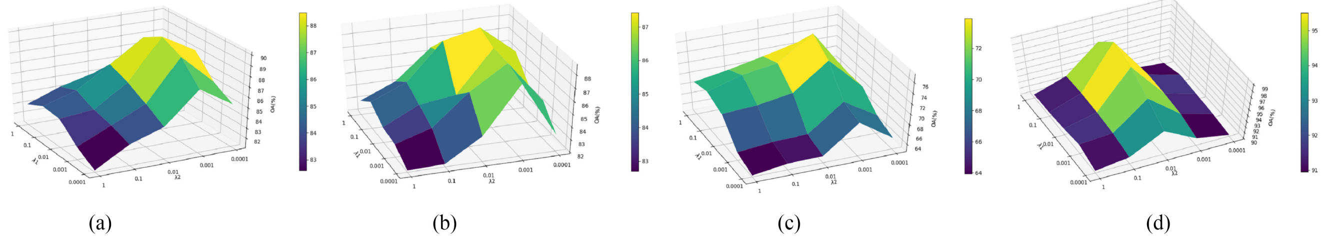


Fig. 7. Effects of λ_1 and λ_2 on the classification performance. (a) Indian Pines 2010, (b) Houston University 2012, (c) Houston University 2017, and (d) Pavia University datasets.

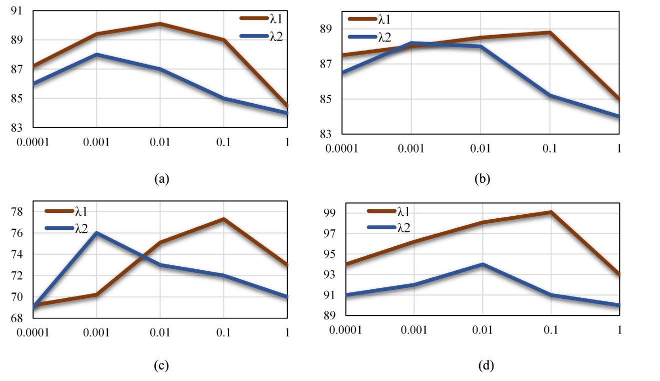


Fig. 8. Investigate the effects of λ_1 and λ_2, separately, on classification performance. (a) Indian Pines 2010, (b) Houston University 2012, (c) Houston University 2017, and (d) Pavia University datasets. The vertical axis and the horizontal axis indicate the OA (%) and candidate set for λ_1 and λ_2, respectively.

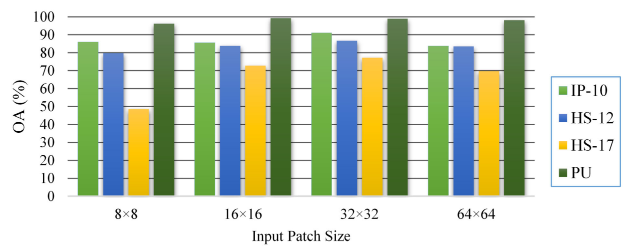


Fig. 9. Effects of input patch size on classification performance.

TABLE II
EFFECTS OF VARYING THE REGION SIZE TO EXCHANGE

Dataset	1 × 1	3 × 3	5 × 5	7 × 7	9 × 9
IP-10	87.39±1.95	88.19±1.08	**90.21**±1.02	85.22±1.15	84.21±0.96
HU-12	81.21±0.95	84.32±0.85	**85.10**±0.95	79.02±0.80	73.31±0.85
HU-17	73.12±2.12	**76.01**±1.85	72.15±2.01	60.41±1.93	58.99±1.90
PU	98.25±0.07	**99.12**±0.05	97.25±0.05	97.25±0.06	96.16±0.04

For each region size, the table shows the average OA and standard deviation our proposed method on four datasets.

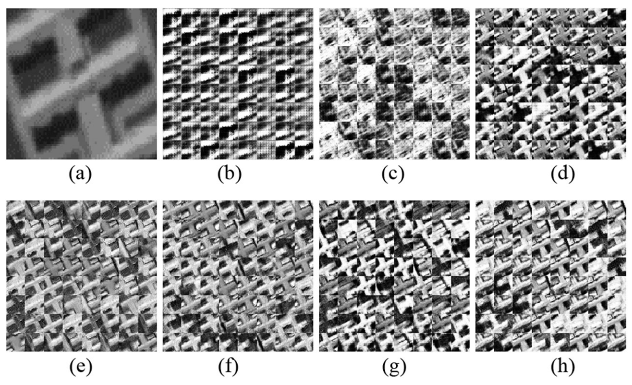


Fig. 10. Example of generating process of 64 VTS by GAN in class metal-sheet in the Pavia University dataset—Each epoch has 64 VTS that are arranged in eight rows and eight columns in the given figures. (a) Zoomed real training sample. (b)–(h) VTS generated by SA-GAN.

to capture sufficient spatial information avoiding possible over-smoothing. For evaluation, real training samples were extracted according to each size from HSI datasets (IP-10, HU-12, HU-17, and PU datasets) and the architecture of the discriminator and generator was adjusted based on the input patch size and then the process of production of VTS was performed. The quality of virtual samples produced with different size was evaluated using the OA. Fig. 9 presents the results for each dataset based on OA for all VTS. The output of the sensitivity analysis and the test with different sizes is that the optimal input size for the IP-2010, HU-12, and HU-17 datasets is 32 × 32. The best input patch size for PU dataset is 16 × 16.

3) Experiment 3. The Effects of Region Size: Another important parameter of the proposed method that needs to be adjusted is the selection of the appropriate region size of feature vectors that are exchanged within that feature map. In order to investigate the effect of region size, we trained a SA-GAN with different region size on all datasets with input patch size 32 × 32 and measured the OA. Table II lists the average OA and standard deviation results when the feature exchange operator is utilized with different region sizes after the algorithm is tested 50 times because of random permutation parameter (δ) in the exchange operator (2). It shows that when the region size is increased, OA reduces. When the region size is large, it is easy to see whether the feature map is being exchanged or not and the efficiency of the proposed method is reduced. The maximum standard deviations were $1.95, 0.95, 2.12$, and 0.07 for IP-10, HU-12, HU-17, and PU datasets, respectively. Approximately with increasing region size, the standard deviation decreases but the classification accuracy decreases (for dimensions larger than 5 × 5).

4) Experiment 4. The Visual Quality of VTS: Fig. 10 illustrates the process of creating VTS for the class of metal sheets from the Pavia University dataset in different training epochs. As one can see, the initial epochs of the generated virtual sample

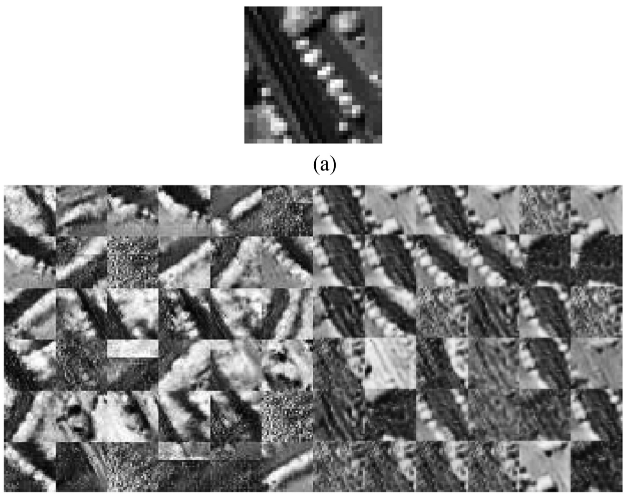


Fig. 11. Examples of interpolated VTS between two latent vectors by the proposed SA-GAN for PU dataset class ♯4, "Trees." (a) Real training sample. (b) Images contain some VTS with different variety compare to real training sample.

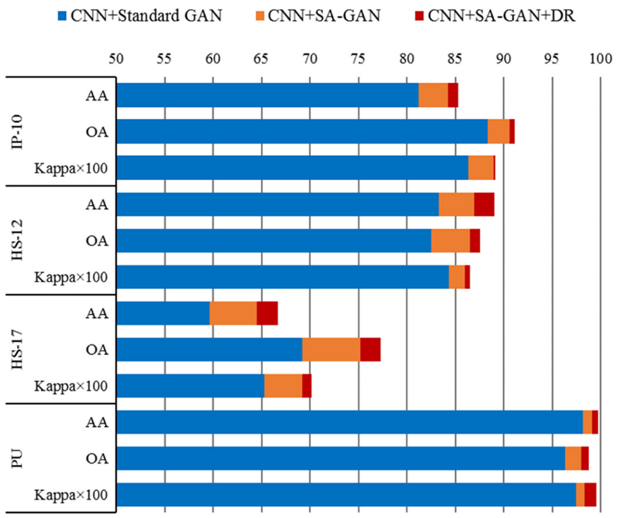


Fig. 12. Effects of the proposed SA-GAN model and dimensionality reduction on classification performance.

are free from any kind of information, but the generated samples in the final epochs are similar to real training samples.

Fig. 11 shows the example of samples generated from the proposed SA-GAN. Samples are created from vectors (latent space corresponds to specific class from generator network), whose values were linearly interpolated [60]. To display the samples, three vectors were sampled from the latent space belonging to class ♯4 of Pavia University dataset (see Fig. 11), and then to have more diversity, samples were produced by averaging the distance between the three vectors and passing those vectors through the generator network (see Table I). The virtual samples extracted by the proposed method are shown in Fig. 11(b). Considering the real training sample [see Fig. 11(a)], we can see the variety of VTS and the preservation of the structure in Fig. 11(b). By a qualitative analysis of the generated images, the following observations can be made.

1) The SA-GAN is able to capture the global structure of the training samples.
2) The SA-GAN is able to generate the sharper training samples compared to the real training samples.
3) The SA-GAN is capable to generate diversity in training samples including changing shade, rotation and intensity.
4) The SA-GAN is able to approximate the distribution of real training samples to produce virtual samples, which are structurally consistent.

5) Experiment 5. The Effect of DR and SA-GAN: DR and proposed new formulation of GAN (6)–(9) are essential components of SA-GAN. For evaluating their effects on the classification performance, we perform the classification procedure by CNN (see Fig. 4) and standard GAN on original bands of four datasets (without DR preprocessing) as a basis for comparison. Then, we set the hyperparameters to their optimal values (obtained by previous experiments) on different dataset to achieve another result with the help of SA-GAN. Also, the experiment was performed with DR. Fig. 12 compares their performance in terms of AA, OA, and K. In this figure, blue bars are the results with CNN+standard GAN, whereas red bars represent the performance improvements by proposed SA-GAN and finally the green bars represent the performance improvement with SA-GAN+DR. On the basis of these findings, it is reasonable to infer that the proposed approach indeed have a significant contribution to the HSI classification.

6) Experiment 6. Accuracy Performance and Visual Analysis of Classification Maps: For comparison, six methods have been selected, which are divided into two categories. Three methods based on the CNN that generate the VTS using methods other than the GAN idea (i.e., S-CNN [55], CRVS-CNN [41], and MDR-CNN [12]) and three methods based on the GAN idea to generate VTS (i.e., S-GAN [14], W-GAN [56], and Self-GAN [57]). In this experiment, we used training and test data shown in Figs. 5 and 6 for each dataset. To fairly compare the methods, the number of considered training and test samples are the same. To obtain the classification results by the proposed method, we use three main types of layers to build CNNs architectures: convolution layer, pooling layer, and fully connected layer. We stacked these layers to form a full CNNs architecture (see Fig. 4). During the training phase, virtual and real training samples were concatenated and then the estimation of learning parameters of the CNN was conducted by the Adam optimization strategy in order to determine the parameters of weights and bias of the convolution layers. The basic CNN architecture for classification was considered the same for all methods. Of course, for S-CNN, CRVS-CNN, and MDR-CNN, the architecture presented in their respective articles was also examined and the best results for each method were evaluated. The size of input patches is 32×32. All the real and virtual samples (after VTS generation) were normalized into $[-1\ 1]$ and the number of epochs was set to 500 for the proposed CNN in Fig. 4. Other settings are explained in Section IV-B. During the

TABLE III
CLASSIFICATION ACCURACIES(%) OBTAINED BY DIFFERENT TECHNIQUES FOR THE INDIAN PINES 2010 DATASET

Class	Classification Methods						
	S-CNN [55]	**CRVS-CNN [41]**	**MDR-CNN [12]**	**S-GAN [14]**	**W-GAN [56]**	**Self-GAN [57]**	**SA-GAN (Proposed)**
OA (%)	82.03	86.05	84.67	88.12	86.18	89.53	**91.95**
AA (%)	78.03	76.86	80.02	83.34	80.42	81.35	**88.18**
K(x100)	77.27	83.05	82.38	85.66	83.29	87.28	**90.23**
1	95.88	87.39	87.79	88.91	88.63	92.56	**98.61**
2	90.86	90.94	90.62	98.50	97.27	99.16	**99.50**
3	91.85	93.63	93.83	98.55	96.26	96.98	**99.08**
4	72.01	75.92	**83.35**	82.79	73.65	83.21	81.09
5	83.52	84.21	82.18	83.84	83.80	86.08	**90.46**
6	84.61	91.80	87.70	89.72	85.70	95.67	**96.46**
7	55.09	57.19	60.87	59.96	65.14	68.26	**74.22**
8	45.89	27.53	83.18	75.27	44.21	29.42	**86.44**
9	68.46	84.44	78.44	84.19	84.50	87.76	**87.92**
10	84.98	80.83	79.15	83.21	86.71	81.96	**93.70**
11	69.32	73.58	69.40	76.24	76.96	78.58	**92.40**
12	**100.00**	97.54	95.24	98.00	99.58	98.67	99.83
13	97.33	94.18	94.35	98.24	95.16	97.91	**99.03**
14	90.35	89.79	94.73	**99.85**	91.05	97.57	99.84
15	96.16	80.84	94.78	98.95	91.69	98.36	**99.79**
16	22.12	20.02	4.67	17.18	**26.46**	9.46	12.54
Training (s)	**412.38**	647.14	896.65	1136.31	1145.75	1236.12	1212.01
Test (s)	**15.85**	15.96	16.32	20.15	20.3	19.3	19.8

TABLE IV
CLASSIFICATION ACCURACIES(%) OBTAINED BY DIFFERENT TECHNIQUES FOR THE HOUSTON UNIVERSITY 2012 DATASET

Class	Classification Methods						
	S-CNN [55]	**CRVS-CNN [41]**	**MDR-CNN [12]**	**S-GAN [14]**	**W-GAN [56]**	**Self-GAN [57]**	**SA-GAN (Proposed)**
OA (%)	82.74	77.95	80.86	86.03	83.24	85.98	**88.18**
AA (%)	83.44	79.50	82.90	86.57	84.62	86.77	**88.42**
K(x100)	81.96	76.09	79.28	83.35	81.81	84.77	**87.25**
1	80.45	82.74	82.43	82.48	**84.52**	83.12	81.56
2	97.63	96.37	93.92	85.43	94.13	**99.28**	97.26
3	84.19	97.50	98.48	91.77	**100.00**	85.27	89.63
4	96.42	97.26	96.79	88.12	98.61	**98.87**	98.19
5	97.61	94.49	94.46	99.95	96.88	99.59	**100.00**
6	81.86	**97.24**	96.65	86.80	95.98	91.22	83.08
7	86.80	78.82	80.15	79.39	87.65	87.00	**93.84**
8	48.17	48.89	49.95	**81.32**	66.91	74.92	76.39
9	79.40	67.56	72.63	79.63	74.07	84.14	**85.00**
10	76.83	63.53	70.31	73.90	61.50	66.68	**92.27**
11	84.95	77.60	82.79	83.62	82.47	85.45	**89.49**
12	85.68	57.23	68.59	87.57	72.47	77.50	**89.74**
13	80.85	39.27	61.60	84.41	58.16	**88.21**	85.44
14	88.58	96.39	97.66	95.13	**98.89**	93.71	92.99
15	82.19	97.68	97.07	**99.02**	97.12	86.66	71.37
Training (s)	**65.27**	69.13	75.01	191.20	195.34	198.74	196.29
Test (s)	**1.25**	2.10	2.56	3.23	3.88	3.76	4.05

testing phase, pixels were classified with the parameters obtained during the training phase.

Tables III–VI report the results of the proposed SA-GAN methods and the other techniques used for comparisons. In these tables, the first three rows are corresponding to OA, AA, and K of the whole data, whereas the next rows report the individual class accuracies and for each row, the bold fonts indicate the best performance. Several conclusions can be drawn on the basis of these results. Generally, from Tables III–VI, the GAN-based models (i.e., S-GAN, W-GAN, Self-GAN, and SA-GAN) achieve higher performance than the other models (i.e., S-CNN, CRVS-CNN, and MDR-CNN), in terms of OA, AA, and K, due to the advantage of adversarial networks for VTS generation. The MDR-CNN method has the lowest performance in HU-12, HU-17, and PU datasets. This is due to metaheuristic DR method, which has caused the loss of input information to the CNN classifier. The CRVS-CNN method, which uses PCA for DR, is more efficient than the MDR-CNN method in IP-10, HU-17, and PU datasets.

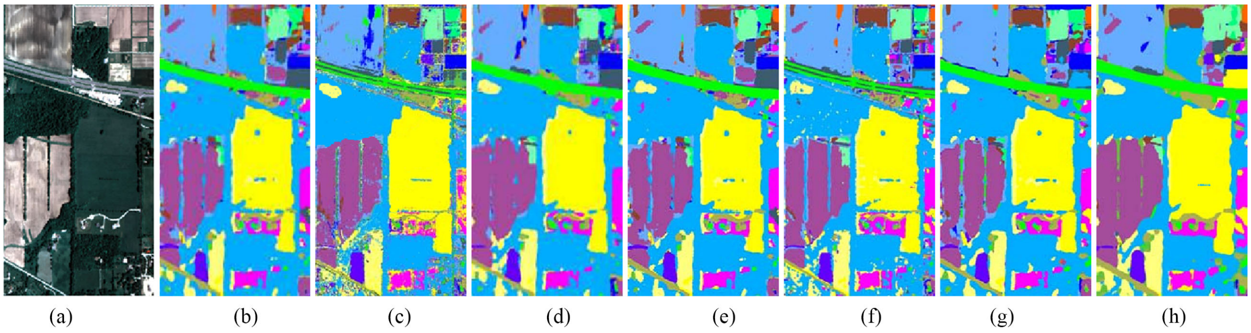
In all four datasets, the performance of the proposed method in term of OA, AA, and K is higher than that of GAN-based methods, because the structure of the training samples is considered and the appropriate DR method is utilized in our SA-GAN model. One can see that the SA-GAN has been able to increase the OA by 2.42%, 2.15%, 2.78%, and 0.02% on IP-10, HU-12, HU-17, and PU datasets, respectively. Regarding the IP-10 dataset (see Table III), for each-class accuracies, SA-GAN model can obtain the highest values for 12 out of the total 16 classes, which validates the performance of our proposed model. Focusing on Table V, one can see that for the challenging HU-17 dataset, the SA-GAN shows the highest OA, AA, and K, with an improvement of 2.78%, 1.66%, and 0.0314 over the S-GAN method, respectively. Another observation is that GAN-based model provides very good performance compare to other models

TABLE V
CLASSIFICATION ACCURACIES(%) OBTAINED BY DIFFERENT TECHNIQUES FOR THE HOUSTON UNIVERSITY 2017 DATASET

Class	Classification Methods						
	S-CNN [55]	**CRVS-CNN [41]**	**MDR-CNN [12]**	**S-GAN [14]**	**W-GAN [56]**	**Self-GAN [57]**	**SA-GAN (Proposed)**
OA (%)	61.78	59.38	48.51	72.84	68.96	69.75	**75.62**
AA (%)	57.20	57.71	54.54	62.88	60.91	61.41	**64.48**
K(x100)	50.14	49.48	39.36	64.85	60.14	60.73	**67.99**
1	70.07	79.32	56.96	34.60	78.92	**80.63**	50.38
2	76.68	78.85	69.69	**89.38**	82.66	83.99	87.34
3	72.32	**99.91**	99.64	81.21	92.99	93.29	98.42
4	89.34	92.18	86.07	94.83	95.07	**95.98**	95.16
5	38.26	49.79	40.34	**55.14**	54.93	53.27	50.34
6	25.43	25.79	25.36	31.30	25.14	26.74	**32.28**
7	24.06	30.15	30.24	28.33	**36.96**	36.58	17.90
8	80.36	72.11	73.56	**89.87**	83.47	84.67	86.79
9	74.69	63.00	41.85	79.07	76.87	75.96	**84.40**
10	54.01	38.11	34.58	**56.58**	42.82	44.14	45.14
11	49.54	41.37	46.91	58.12	63.87	62.11	**64.26**
12	5.06	1.44	**6.87**	4.21	4.68	5.78	6.25
13	46.45	45.22	47.86	52.09	54.68	54.55	**66.00**
14	47.86	37.83	33.03	**60.54**	33.20	35.33	43.03
15	57.72	51.95	52.85	**61.73**	52.61	54.53	58.18
16	79.91	74.53	70.85	78.75	68.61	69.68	**83.63**
17	90.35	99.91	**99.93**	80.91	81.25	79.95	96.33
18	58.58	53.40	60.74	69.48	55.00	55.23	**78.53**
19	86.02	65.61	65.41	93.06	72.73	73.81	**88.87**
20	57.37	53.80	47.97	58.40	61.81	**62.04**	56.38
Training (s)	**832.14**	965.07	1002.46	2150.87	2220.87	2350.47	2298.10
Test (s)	**8.21**	9.14	9.30	15.09	16.83	18.20	18.90

TABLE VI
CLASSIFICATION ACCURACIES(%) OBTAINED BY DIFFERENT TECHNIQUES FOR THE PAVIA UNIVERSITY DATASET

Class	Classification Methods						
	S-CNN [55]	**CRVS-CNN [41]**	**MDR-CNN [12]**	**S-GAN [14]**	**W-GAN [56]**	**Self-GAN [57]**	**SA-GAN (Proposed)**
OA (%)	99.54	96.48	95.64	99.80	99.78	96.74	**99.82**
AA (%)	99.66	97.03	97.20	99.69	**99.72**	97.68	99.71
K(x100)	99.41	96.01	95.53	99.73	99.53	96.65	**99.76**
1	99.36	97.42	94.67	99.87	99.65	94.88	**99.94**
2	99.36	95.76	93.65	**99.99**	99.55	96.07	99.98
3	**99.69**	94.05	98.24	99.21	99.47	96.13	99.05
4	**99.63**	97.52	99.52	99.17	99.52	98.93	99.39
5	99.95	100.00	99.91	99.97	100.00	99.91	100.00
6	99.96	99.13	99.78	99.97	99.64	99.07	**99.99**
7	100.00	96.19	96.13	99.94	100.00	98.01	99.95
8	99.65	93.62	96.81	99.45	**99.68**	97.54	99.25
9	99.38	99.60	96.09	99.66	**100.00**	98.60	99.81
Training (s)	**71.14**	71.56	73.21	201.63	202.06	208.44	203.58
Test (s)	**2.67**	2.98	3.02	5.69	6.17	7.88	7.39



🟫	🟦	🟧	🟩	🟪	🟦	🟩	⬛
Corn-high	Corn-mid	Corn-low	Soy-bean-high	Soy-bean-mid	Soy-bean-low	Residues	Wheat
🟨	🟨	🟪	🟪	🟦	🟩	🟫	🟩
Hay	Grass/Pasture	Cover crop 1	Cover crop 2	Wood-lands	Highway	Local road	Building

Fig. 13. Classification maps obtained by the different methods on the Indian Pines 2010 dataset. (a) RGB image. (b) Classification map obtained by the S-CNN method. (c) Classification map obtained by the CRVS-CNN method. (d) Classification map obtained by the MDR-CNN method. (e) Classification map obtained by the S-GAN method. (f) Classification map obtained by the W-GAN method. (g) Classification map obtained by the Self-GAN method. (h) Classification map obtained by the SA-GAN method (proposed).

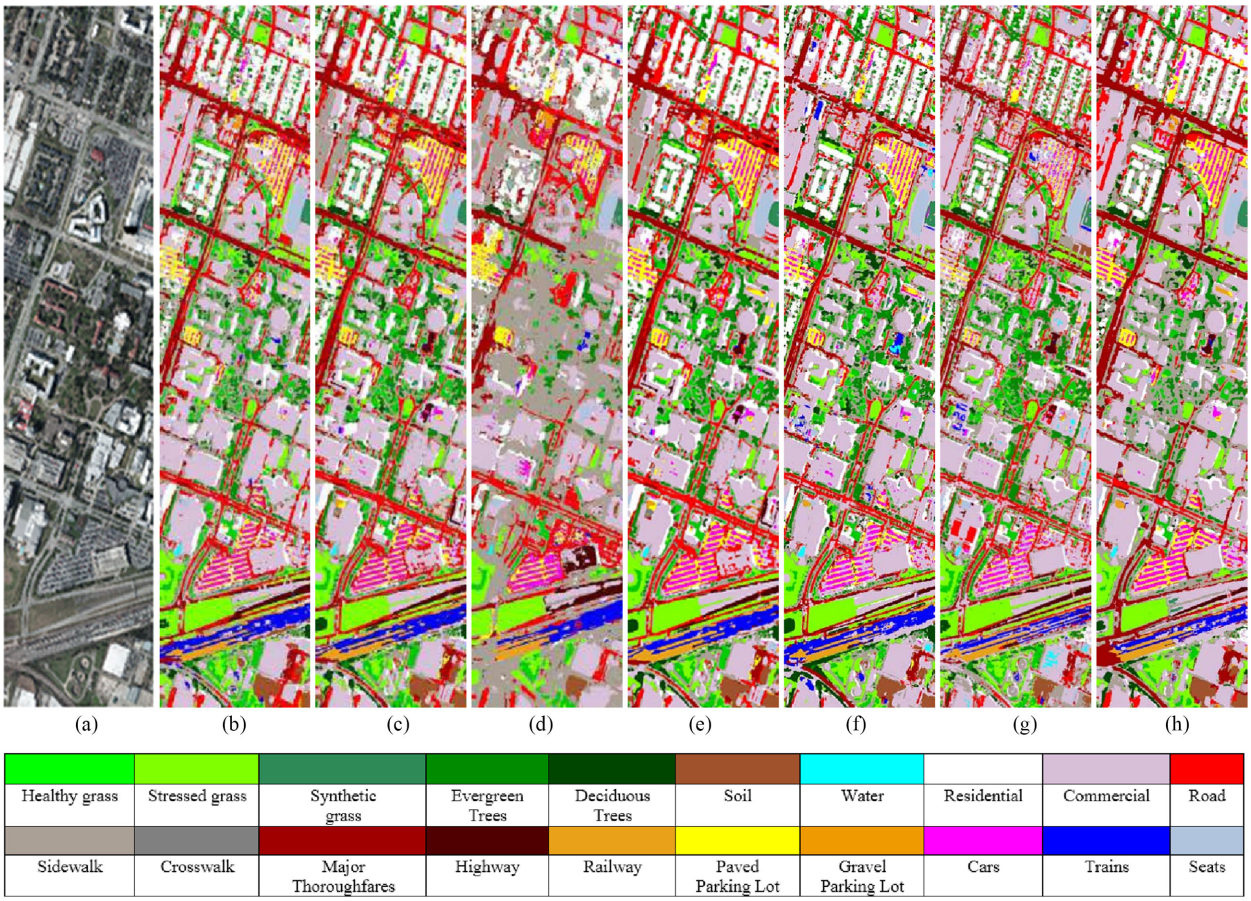


Fig. 14. Classification maps obtained by the different methods on the Houston University 2017 dataset. (a) RGB image. (b) Classification map obtained by the S-CNN method. (c) Classification map obtained by the CRVS-CNN method. (d) Classification map obtained by the MDR-CNN method. (e) Classification map obtained by the S-GAN method. (f) Classification map obtained by the W-GAN method. (g) Classification map obtained by the Self-GAN method. (h) Classification map obtained by the SA-GAN method (proposed).

(i.e., S-CNN, CRVS-CNN, and MDR-CNN) that the OA of the SA-GAN compared to the S-CNN has increased by about 14%. As can be seen from Table VI, the OA of the proposed method reaches 99.82% for PU dataset. All of the above findings can confirm the effectiveness of the proposed model.

The last two rows in Tables III–VI report the computation time of various models in four datasets, including the training time and the test time. As expected, for each model the training process requires much more time than the test process. By comparing the computation time of two groups (CNN-based models and GAN-related models), it can be concluded that the GAN-related models need more time for training, because the GAN process takes time for convergence. In terms of the training time and test time, S-CNN takes less time than the other models on the four datasets. Although the proposed method is slightly more time consuming than the S-GAN and the W-GAN, the classification results show the higher efficiency of the SA-GAN in term of OA, AA, and *K*.

In terms of visual analysis, the quality of the classification maps of the proposed method is highly remarkable. Figs. 13 and 14 show the classification maps provided by the proposed method on Indian Pines 2010 and Houston University 2017 dataset, respectively. From the figures, one can see the impact of different methods on the classification results. By comparing the results with the reference data, we can see that the classification map obtained by the proposed method is more accurate, which confirms that the SA-GAN is a suitable method for classifying HSIs. This is due to the inclusion in the same framework the generation of structure aware virtual samples by GAN and the subspace base reduction.

V. Conclusion and Future Lines

In this article, we proposed an SA-GAN for HSI classification. We extend the discriminator ability in such a way that it can simultaneously indicate the real and fake samples and recognize structural inconsistency. A new loss function is proposed to train the multitask discriminator, which leads to a regularized feature representation for the discriminator and thus a better generator. The discriminator architecture was divided into two parts to produce structure-aware virtual samples with high diversity and quality. In order to evaluate the performance of the proposed model, we conducted experiments on four HSI datasets, and compared it with six models including GAN-based and conventional models for generation of virtual samples. We analyzed the effects of different hyperparameters of SA-GAN on the classification performance, including the regularization parameters, the input patch size, and the region size. Moreover, the visual quality of virtual samples and the adversarial effect of SA-GAN were investigated. Experiments and analysis confirmed the efficiency and effectiveness of the proposed approach.

Future studies can be conducted for further improving the generalization capability of the CNN-based methods. In this regard, we plan to study a new architecture for the generator and the discriminator to generate more realistic training samples. Also, the development of quantitative evaluation methods for assessing the quality of VTS used in GAN training is a topic for future research.

Acknowledgment

The authors would like to thank Prof. M. Crawford, the National Center for Airborne Laser Mapping, Hyperspectral Image Analysis Laboratory at the University of Houston, IEEE GRSS Image Analysis and Data Fusion Technical Committee, and Prof. P. Gamba for providing the Indian Pines 2010, Houston University 2012 and 2017, and Pavia University datasets, respectively. They would also like to thank the researchers who published their source code and they made it possible to compare the proposed method of this study with their methods.

References

[1] A. Plaza *et al.*, "Recent advances in techniques for hyperspectral image processing," *Remote Sens. Environ.*, vol. 113, pp. S110–S122, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0034425709000807

[2] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and Markov random fields," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 3, pp. 809–823, Mar. 2012.

[3] 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.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[5] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[6] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "Joint learning of words and meaning representations for open-text semantic parsing," in *Proc. 15th Int. Conf. Artif. Intell. Statist.*, Apr. 21–23, 2012, vol. 22, pp. 127–135. [Online]. Available: http://proceedings.mlr.press/v22/bordes12.html

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances Neural Inf. Process. Syst.*, vol. 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-conv% olutional-neural-networks.pdf

[8] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems*, vol. 26, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, Inc., 2013, pp. 2553–2561. [Online]. Available: http://papers.nips.cc/paper/5207-deep-neural-networks-for-object-detect% ion.pdf

[9] B. Rasti *et al.*, "Feature extraction for hyperspectral imagery: The evolution from shallow to deep (overview and toolbox)," *IEEE Geosci. Remote Sens. Mag.*, to be published, doi: 10.1109/MGRS.2020.2979764.

[10] T. Alipour-Fard, M. E. Paoletti, J. M. Haut, H. Arefi, J. Plaza, and A. Plaza, "Multibranch selective kernel networks for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, to be published, doi: 10.1109/LGRS.2020.2990971.

[11] Y. Chen, K. Zhu, L. Zhu, X. He, P. Ghamisi, and J. A. Benediktsson, "Automatic design of convolutional neural network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 7048–7066, Sep. 2019.

[12] P. Ghamisi, Y. Chen, and X. X. Zhu, "A self-improving convolution neural network for the classification of hyperspectral data," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 10, pp. 1537–1541, Oct. 2016.

[13] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," 2016, *arXiv:1605.07678*.

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

[15] W. Li, C. Chen, M. Zhang, H. Li, and Q. Du, "Data augmentation for hyperspectral image classification with deep CNN," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 4, pp. 593–597, Apr. 2019.

[16] A. Signoroni, M. Savardi, A. Baronio, and S. Benini, "Deep learning meets hyperspectral image analysis: A multidisciplinary review," *J. Imag.*, vol. 5, no. 5, 2019. [Online]. Available: https://www.mdpi.com/2313-433X/5/5/52

[17] X. Ma, H. Wang, and J. Wang, "Semisupervised classification for hyperspectral image based on multi-decision labeling and deep feature learning," *ISPRS J. Photogramm. Remote Sens.*, vol. 120, pp. 99–107, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0924271616303124

[18] H. Wu and S. Prasad, "Semi-supervised deep learning using pseudo labels for hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1259–1270, Mar. 2018.

[19] Y. Ju, L. Li, L. Jiao, Z. Ren, B. Hou, and S. Yang, "Modified diversity of class probability estimation co-training for hyperspectral image classification," 2018, *arXiv:1809.01436*.

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

[21] S. Yu, S. Jia, and C. Xu, "Convolutional neural networks for hyperspectral image classification," *Neurocomputing*, vol. 219, pp. 88–98, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231216310104

[22] L. Windrim, A. Melkumyan, R. J. Murphy, A. Chlingaryan, and R. Ramakrishnan, "Pretraining for hyperspectral convolutional neural network classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 5, pp. 2798–2810, May 2018.

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

[24] E. Basaeed, H. Bhaskar, and M. Al-Mualla, "Supervised remote sensing image segmentation using boosted convolutional neural networks," *Knowl.-Based Syst.*, vol. 99, pp. 19–27, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705116000484

[25] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of tricks for image classification with convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 558–567.

[26] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*.

[27] T. Alipourfard and H. Arefi, "Virtual training sample generation by generative adversarial networks for hyperspectral images classification," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 42, pp. 63–69, 2019.

[28] Z. Zhong and J. Li, "Generative adversarial networks and probabilistic graph models for hyperspectral image classification," 2018, *arXiv:1802.03495*.

[29] Z. He, H. Liu, Y. Wang, and J. Hu, "Generative adversarial networks-based semi-supervised learning for hyperspectral image classification," *Remote Sens.*, vol. 9, no. 10, 2017. [Online]. Available: https://www.mdpi.com/2072-4292/9/10/1042

[30] I. J. Goodfellow *et al.*, "Generative adversarial networks," 2014, *arXiv:1406.2661*.

[31] R. Huang, W. Xu, T. Lee, A. Cherian, Y. Wang, and T. K. Marks, "FX-GAN: Self-supervised GAN learning via feature exchange," in *Proc. IEEE Winter Conf. Appl. Comput. Vision*, 2020, pp. 3183–3191.

[32] W. Wang, H. Li, Y. Deng, L. Shao, X. Lu, and Q. Du, "Generative adversarial capsule network with ConvLSTM for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, to be published, doi: 10.1109/LGRS.2020.2976482.

[33] N. He *et al.*, "Feature extraction with multiscale covariance maps for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 755–769, Feb. 2019.

[34] J. Feng *et al.*, "Generative adversarial networks based on collaborative learning and attention mechanism for hyperspectral image classification," *Remote Sens.*, vol. 12, no. 7, 2020. [Online]. Available: https://www.mdpi.com/2072-4292/12/7/1149

[35] H. Gao *et al.*, "A hyperspectral image classification method based on multi-discriminator generative adversarial networks," *Sensors*, vol. 19, no. 15, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/15/3269

[36] J. Feng, H. Yu, L. Wang, X. Cao, X. Zhang, and L. Jiao, "Classification of hyperspectral images based on multiclass spatial–spectral generative adversarial networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 5329–5343, Aug. 2019.

[37] C. Tao, H. Wang, J. Qi, and H. Li, "Semisupervised variational generative adversarial networks for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 914–927, Feb. 2020.

[38] M. Mirza and S. Osindero, "Conditional generative adversarial Nets," 2014, *arXiv:1411.1784*.

[39] T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby, "Self-supervised GANs via auxiliary rotation loss," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 12154–12163.

[40] K. Bittner, M. Korner, F. Fraundorfer, and P. Reinartz, "Multi-task cGAN for simultaneous spaceborne DSM refinement and roof-type classification," *Remote Sens.*, vol. 11, no. 11, 2019. [Online]. Available: https://www.mdpi.com/2072-4292/11/11/1262

[41] 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.

[42] W. Zhao and S. Du, "Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4544–4554, Aug. 2016.

[43] B. Pan, Z. Shi, and X. Xu, "MugNet: Deep learning for hyperspectral image classification using limited samples," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 108–119, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0924271617303416

[44] A. Romero, C. Gatta, and G. Camps-Valls, "Unsupervised deep feature extraction for remote sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1349–1362, Mar. 2016.

[45] 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.

[46] Y. Li, H. Zhang, and Q. Shen, "Spectral-spatial classification of hyperspectral imagery with 3D convolutional neural network," *Remote Sens.*, vol. 9, no. 1, 2017. [Online]. Available: https://www.mdpi.com/2072-4292/9/1/67

[47] Y. Qin, N. Mitra, and P. Wonka, "How does Lipschitz regularization influence GAN training?" 2018, *arXiv:1811.09567*.

[48] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. 34th Int. Conf. Mach. Learning*, Aug. 6–11, 2017, vol. 70, pp. 214–223. [Online]. Available: http://proceedings.mlr.press/v70/arjovsky17a.html

[49] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," 2017, *arXiv:1704.00028*.

[50] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," 2018, *arXiv:1802.05957*.

[51] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artif. Intell. Rev.*, pp. 1–62, 2020.

[52] C. Debes *et al.*, "Hyperspectral and LiDAR data fusion: Outcome of the 2013 GRSS Data Fusion Contest," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2405–2418, Jun. 2014.

[53] Y. Xu *et al.*, "Advanced multi-sensor optical remote sensing for urban land use and land cover classification: Outcome of the 2018 IEEE GRSS Data Fusion Contest," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 6, pp. 1709–1724, Jun. 2019.

[54] B. Kunkel, F. Blechinger, R. Lutz, R. Doerffer, H. van der Piepen, and M. Schroder, "ROSIS (reflective optics system imaging spectrometer)—A candidate instrument for polar platform missions," *Proc. SPIE*, vol. 0868, p. 8, 1988.

[55] M. Paoletti, J. Haut, J. Plaza, and A. Plaza, "A new deep convolutional neural network for fast hyperspectral image classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 120–147, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0924271617303660

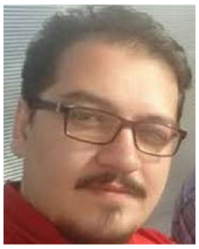
[56] M. Zhang, M. Gong, Y. Mao, J. Li, and Y. Wu, "Unsupervised feature extraction in hyperspectral images based on Wasserstein generative adversarial network," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 5, pp. 2669–2688, May 2019.

[57] W. Zhao, X. Chen, J. Chen, and Y. Qu, "Sample generation with self-attention generative adversarial adaptation network (SaGAAN) for hyperspectral image classification," *Remote Sens.*, vol. 12, no. 5, 2020. [Online]. Available: https://www.mdpi.com/2072-4292/12/5/843

[58] M. Paoletti, J. Haut, J. Plaza, and A. Plaza, "Deep learning classifiers for hyperspectral imaging: A review," *ISPRS J. Photogramm. Remote Sens.*, vol. 158, pp. 279–317, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0924271619302187

[59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[60] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," 2017, *arXiv:1710.10196*.



Tayeb Alipour-Fard received the M.S. degree in remote sensing from Kerman Graduate University of Technology, Kerman, Iran, in 2013. He is currently working toward the Ph.D. degree in remote sensing and photogrammetry with the School of Surveying and Geospatial Engineering, University of Tehran, Tehran, Iran.

Since 2017, he was a Visiting Researcher with the RSLab with the Department of Information Engineering and Computer Science, University of Trento, Trento, Italy. His research interests include machine learning, pattern recognition, deep learning, feature extraction, and classification of hyperspectral imagery.

Mr. Alipour-Fard has been a Reviewer for the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS and IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING journals, since 2018.

Hossein Arefi received the M.Sc. degree from HFT University of Stuttgart, Stuttgart, Germany, in 2003 and the Ph.D. degree from UniBW Munich University, Munich, Germany, in 2009.

He is currently an Assistant Professor with the School of Surveying and Geospatial Engineering, University of Tehran, Tehran, Iran. From September 2008 to August 2013, he was with the Institute of Remote Sensing Technology, German Aerospace Center (DLR), Cologne, Germany. His research interests include 3-D point cloud analysis, 3-D object and surface modeling, machine learning, and digital elevation modeling.