ResBaGAN: A Residual Balancing GAN with Data Augmentation for Forest Mapping

Álvaro G. Dieste, Francisco Argüello, and Dora B. Heras, *Member, IEEE*

***Abstract*—Although deep learning techniques are known to achieve outstanding classification accuracies, remote sensing datasets often present limited labeled data and class imbalances, two challenges to attaining high levels of accuracy. In recent years, the GAN architecture has achieved great success as a data augmentation method, driving research toward further enhancements. This work presents ResBaGAN, a GAN-based method for the classification of remote sensing images, designed to overcome the challenges of data scarcity and class imbalances by constructing an advanced data augmentation framework. This framework builds upon a GAN architecture enhanced with an autoencoder initialization and class balancing properties, a superpixel-based sample extraction procedure with traditional augmentation techniques, and an improved residual network as classifier. Experiments were conducted on large, very high-resolution multispectral images of riparian forests in Galicia, Spain, with limited training data and strong class imbalances, comparing ResBaGAN to other machine learning methods such as simpler GANs. ResBaGAN achieved higher overall classification accuracies, particularly improving the accuracy of minority classes with F1-score enhancements reaching up to 22%.**

***Index Terms*—BAGAN, classification, data augmentation, multispectral, residual network, superpixels.**

Abbreviations

AA	Average accuracy.
ACGAN	Auxiliary classifier generative adversarial network.
Adam	Adaptive moment estimation.
BAGAN	Balancing generative adversarial network.
CGAN	Conditional generative adversarial network.
CNN	Convolutional neural network.
ELU	Exponential linear unit.
ETPS	Extended topology preserving segmentation.
F1	F1-score.
FID	Fréchet inception distance.
FV	Fisher vectors.
GAN	Generative adversarial network.
GCN	Graph convolutional network.
κ	Cohen's kappa.
KELM	Kernel extreme learning machine.
LeakyReLU	Leaky rectified linear unit.
OA	Overall accuracy.
PA	Producer's accuracy.
PReLU	Parametric rectified linear unit.
ReLU	Rectified linear unit.
ResBaGAN	Residual balancing generative adversarial network.
ResNet	Residual network.
RBM	Restricted Boltzmann machines.
SEEDS	Superpixels extracted via energy-driven sampling.
SLIC	Simple linear iterative clustering.
UA	User's accuracy.
UAV	Unmanned aerial vehicle.
VAE	Variational autoencoder.
WP	WaterPixels.

Manuscript received 16 December 2022; revised 17 April 2023 and 23 May 2023; accepted 25 May 2023. Date of publication 1 June 2023; date of current version 21 July 2023. This work was supported in part by the Agencia Estatal de Investigación, Government of Spain under Grant PID2019-104834GB-I00 and Grant TED2021-130367B-I00, in part by the Consellería de Cultura, Educación, Formación Profesional e Universidades, Xunta de Galicia under Grant ED431G 2019/04 and Reference Competitive Group accreditation under Grant ED431C 2022/16, and in part by the Junta de Castilla y León under Grant Project VA226P20 (PROPHET–II). All are cofounded by the European Regional Development Fund (ERDF). The work of Álvaro G. Dieste was also supported by the Consellería de Cultura, Educación, Formación Profesional e Universidades, Xunta de Galicia under Grant ED481A 2022/257. *(Corresponding author: Álvaro G. Dieste.)*

Álvaro G. Dieste and Dora B. Heras are with the Centro Singular de Investigación en Tecnoloxías Intelixentes, Universidade de Santiago de Compostela, 15782 Santiago de Compostela, Spain (e-mail: alvaro.goldar.dieste@usc.es; dora.blanco@usc.es).

Francisco Argüello is with the Departamento de Electrónica e Computación, Universidade de Santiago de Compostela, 15782 Santiago de Compostela, Spain (e-mail: francisco.arguello@usc.es).

The code developed for this work will be available at https://github.com/alvrogd/ResBaGAN.

Digital Object Identifier 10.1109/JSTARS.2023.3281892

I. Introduction

Remote sensing is essential for monitoring the Earth's surface, aiding in tasks, such as tracking human-made constructions [1], detecting cropland changes [2], and studying ecosystems [3]. Accurate image classification methods are crucial for many of these applications, such as identifying areas invaded by non-native plant species for environmental monitoring. Unfortunately, remote sensing datasets often present limited labeled data and class imbalances (i.e., certain classes have significantly more samples than others), making it challenging to develop accurate classification methods [4]. These challenges become even more pronounced when processing data from multispectral sensors, given their limited spectral resolution [5]. It is crucial to develop advanced methods that optimally use the available data to overcome these challenges.

Neural networks have become increasingly prevalent in remote sensing applications owing to their exceptional performance. For instance, Zhu et al. [6] and Yuan et al. [7]

reviewed the application of different deep learning techniques in agriculture and environmental monitoring. The CNN stands out for its particularly well-suited image processing capabilities, which has led to its widespread application. For example, in Morales et al. [8], DeeplabV3+ CNNs were used to monitor the deforestation of the *Mauritia flexuosa* palm, a dominant species in the Amazon rainforest ecosystem, through high-resolution aerial RGB images acquired by UAV. In Hamdi et al. [9], a modified U-Net CNN was implemented to automatically detect and map the damaged areas in a forested area in Bavaria, Germany, by using aerial photographs with four spectral bands. The U-Net architecture was also used in Isaienkov et al. [10] for deforestation detection in the forest-steppe zone in the Kharkiv region of Ukraine, by using multispectral images of Sentinel-2. Finally, a region-based Mask CNN was used in Chiang et al. [11] for an automated forest health diagnosis in RGB aerial images from the Wood of Cree in Scotland, aiding in the early detection of dead trees.

Over the past few years, remote sensing research has developed more sophisticated deep learning architectures to better utilize the available training data. The CNN design has progressed from shallow architectures with few layers [12] to deeper, more powerful ones with tens or hundreds of layers, commonly known as residual networks [13], [14], [15], [16], which can be recognized by their innovative integration of skip-connections between layers. Recent advances have also introduced new components to CNNs. For example, Li et al. [17] included attention mechanisms into a simple CNN to extract richer spatial and spectral information. Another example is Hong et al. [18], which combined a CNN with a GCN to model relationships between training samples, enhancing the classification performance. Researchers have also explored deep learning architectures beyond CNNs. For instance, He et al. [19] leveraged the transformer architecture [20] to assimilate high volumes of data, advancing the state-of-the-art in the multimodal semantic segmentation of images. Another promising research direction is using multimodal data from various sensors for complex scene analysis. For instance, Hong et al. [21], [22] studied the use of dual-branch CNN architectures to combine features from two different modalities (e.g., hyperspectral and LiDAR). Furthermore, designing self-supervised architectures that can learn from both unlabeled and labeled data is another promising area. For instance, Sun et al. [23] developed a general-purpose model by analyzing millions of unlabeled scenes with a transformer-backed autoencoder architecture, acquiring extensive remote sensing knowledge that can be fine-tuned for a specific task, outperforming various state-of-the-art architectures.

Data augmentation provides another promising solution to address data scarcity and class imbalance constraints. These techniques can assist advanced deep learning classification architectures by synthetically generating new training samples to enrich the learning data. Numerous data augmentation techniques exist [24] and have been applied to remote sensing. The traditional approaches involve transformations that convert a sample into a new one and techniques that combine different samples of the same class to create new ones. For instance, Haut et al. [25] showed the effectiveness of augmentation through the random deletion of input patch segments, while Acción et al. [26] subdivided each patch and applied independent transformations to each segment. In Nalepa et al. [27], new samples were generated from the first principal component of the dataset or by calculating the mean value of each band. More novel augmentation approaches use generative techniques that synthesize samples from scratch after estimating the data distribution, as opposed to transforming existing samples into new ones.

Currently, the most widely adopted data generative approach is the GAN [28]. This deep learning-based augmentation approach has shown great potential compared to traditional techniques [29], owing to its remarkable capacity to generate highly realistic and diverse data from scratch, outperforming earlier generative architectures, such as the RBM [30] and the VAE [31]. Further advancements in the GAN architecture have also enabled its use as a classifier network, establishing it as a valuable tool for remote sensing applications like environmental monitoring. For instance, in Shashank et al. [32], a GAN was used to identify the target epiphyte *Werauhia kupperiana* in RGB images acquired by UAV in Costa Rican forests. Other interesting scenarios of GANs for environmental monitoring are described in [6] and [33]. Unfortunately, successfully training these networks is challenging unless large datasets are available [34]. Second, class imbalances hinder the learning of minority classes, even preventing it entirely. In such cases, GANs tend to synthesize identical samples for these classes or even fail to capture their data distribution, resulting in pure noise [35]. However, a recent development in GANs, the BAGAN [35], provides a promising solution to enable the successful application of GANs when facing these two challenges.

This work presents ResBaGAN, a novel GAN-based classification method for remote sensing images applied to environmental monitoring. The ResBaGAN architecture overcomes the challenges of data scarcity and class imbalances by constructing an advanced data augmentation framework to support the classifier. The framework builds upon the cutting-edge innovations of BAGAN to provide effective GAN-based data augmentation. In addition, the framework leverages a superpixel segmentation-based sample extraction process with traditional augmentation techniques, and a ResNet-based classifier [13] improved to further enhance its capabilities. Experiments were conducted to demonstrate how the combined synergistic effects of all components significantly improved classification performance, particularly in the minority classes, compared to simpler methods like a stand-alone BAGAN. More specifically:

1) The proposed method integrates a GAN architecture inspired by BAGAN to enable effective deep learning-based data augmentation for scarce datasets and their minority classes.
2) The sample extraction process is guided by a superpixel segmentation and includes traditional augmentation techniques, thereby facilitating the more accurate modeling of the different classes. The segmentation ensures that each

sample predominantly contains pixels from a single class, while the additional augmentation techniques increase the diversity of learning data.

3) The classifier features a ResNet-based design improved to integrate data from different levels of abstraction for better generalization. This architecture enhances both the quality of the GAN-based augmentation and the final classification accuracy of ResBaGAN.
4) In terms of evaluation, the FID score [36], which is the current standard metric for GAN-based architectures on RGB data, was adapted to multispectral images. This modification allowed for an accurate assessment of the proposed deep learning-based data augmentation.

The rest of this article is organized as follows. Section II outlines the GAN architecture and its evolutions. Then, Section III describes ResBaGAN, detailing its different components. Thereafter, Section IV presents the experiments for evaluating ResBaGAN in terms of classification performance and robustness. Next, Section V carries out the discussion of this work. Finally, Section VI concludes this article.

II. Related Work

A GAN [28] is a generative deep learning architecture that consists of the two networks represented in Fig. 1(a): a generator G and a discriminator D. These networks are trained in an adversarial fashion: G learns to synthesize samples as realistically as possible to deceive D into believing that they are real, while D is trained to distinguish between real and fake samples. As a result of this opposition of objectives, the improvement of one network encourages the other to perform better.

Both D and G use CNN architectures. D is a standard CNN, while G replaces conventional convolutions with transposed convolutions, which function inversely. D needs to transform a sample into a unidimensional feature vector of length Z for the final classification. In contrast, G aims to generate data akin to the input of D. To achieve this, it starts with a random vector of Z elements, often referred to as a latent vector, reshaped into a low-resolution sample. The size of this sample is gradually increased by applying transposed convolutions to produce a fake sample.

Building upon the initial GAN architecture in Fig. 1(a), numerous optimizations have been proposed in recent years to further extend its capabilities [37]. The following are particularly relevant to this work:

1) **CGAN [38].** In the original GAN, G cannot synthesize samples for a specific class on demand. The CGAN architecture, shown in Fig. 1(b), addressed this limitation by incorporating information corresponding to the desired class in the latent vector.
2) **ACGAN [39].** A natural extension of CGAN is to enable D to assign each sample to the best-fitting class, in addition to discerning between real and fake samples. To this end, D in ACGAN has two outputs as shown in Fig. 1(c): a binary output for detecting fake samples and an output with as many elements as classes.
3) **BAGAN [35].** This architecture introduced several modifications to stabilize training with small datasets and effectively synthesize the minority classes. First, the networks' parameters are initialized by an autoencoder that processes

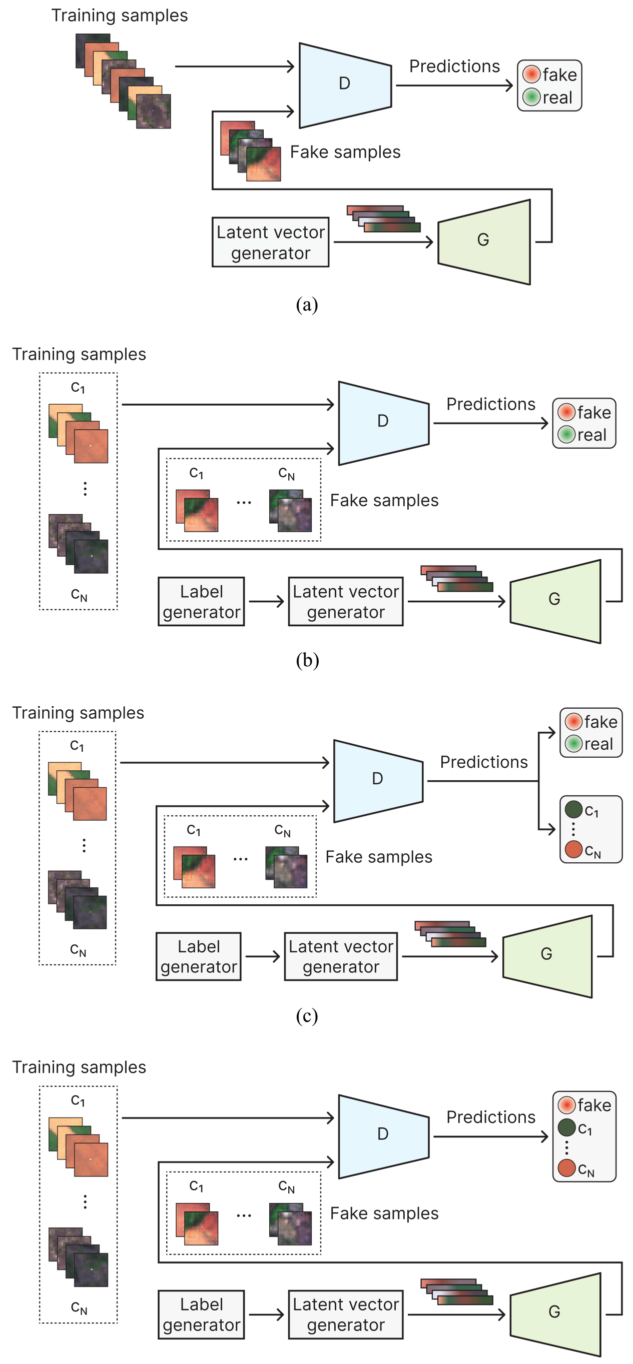


Fig. 1. Comparison between the initial GAN architecture and various evolutions. Note that BAGAN also introduced an autoencoder module to assist the main GAN module in parameter initialization; this autoencoder is not represented for brevity but can be seen in Fig. 2. (a) GAN. (b) CGAN. (c) ACGAN. (d) BAGAN.

the dataset in advance, instead of from scratch. As a result, both D and G start learning from an approximation of the data distribution, significantly stabilizing their training on small datasets. In addition, a more sophisticated sampling strategy for the latent vectors is introduced by modeling class-specific distributions in the latent space provided by the autoencoder, which further stabilizes the training of G. Moreover, the dual outputs of D are combined into a single output, as shown in Fig. 1(d). By extension, the loss function used in the training process becomes simpler, making it easier for G to learn to synthesize the minority classes.

As discussed in Section I, the evolving capabilities of GANs have established them as valuable tools for data augmentation in remote sensing applications. In particular, BAGAN has shown promise in different computer vision tasks to address the challenges of scarce and imbalanced datasets, two common limitations in remote sensing. However, to the best of our knowledge, the use of BAGAN in remote sensing has been limited to object detection, as demonstrated in Zhang et al. [40].

The method proposed in this article, ResBaGAN, is a GAN-based classification approach inspired by innovations in BAGAN, among other features. We not only aim to expand its usage in remote sensing, particularly for classification applied to environmental monitoring, but also enhance its performance further. This is achieved by incorporating traditional augmentation techniques, superpixel segmentation-based sample extraction, and an improved ResNet-based classifier.

III. Proposed Method

ResBaGAN, the proposed method for remote sensing classification applied to environmental monitoring, is illustrated in Figs. 2 and 3. This section describes its different components as follows. First, the GAN architecture designed for deep learning-based data augmentation is discussed in Section III-A. Next, the sample extraction procedure that combines superpixel segmentation and traditional augmentation techniques is introduced in Section III-B. Then, following these explanations, a complete step-by-step explanation of how ResBaGAN operates is provided. Finally, the design of the neural architectures in ResBaGAN is detailed in Section III-C.

A. Network Architecture

The core of ResBaGAN is a BAGAN-based architecture that includes an improved ResNet-based classifier [13], resulting in a combination that enhances both the quality of the GAN-based augmentation and the final classification accuracy. The integration of shortcuts in residual networks allows building much deeper architectures compared to stacking convolutional layers alone, circumventing training issues like gradient vanishing. This makes ResNet more suitable than shallow CNNs for classification with limited data, as it facilitates the extraction of more detailed features from the available samples. Moreover, the generalization capabilities of the designed ResNet architecture have been improved by fusing features from every convolutional stage into the final classification layers, instead of using features just from the last stage. Furthermore, the designed GAN module leverages two BAGAN features to assist the generator network in learning: autoencoder initialization and an improved loss function. This approach leads to the synthesis of more realistic samples when dealing with scarce and imbalanced datasets, thus providing an effective data augmentation for such situations.

As shown in Fig. 2, three main elements can then be identified in the network architecture of ResBaGAN:

1) The input data are patches from the different classes, as displayed on the left side of the figure. The specialized sample extraction procedure provides this augmented training data and will be detailed in Section III-B.
2) ResBaGAN leverages an autoencoder module [41] to stabilize the subsequent training of the GAN on limited data. The autoencoder is depicted in the upper part of the figure. Autoencoders are easier to train than GANs under data scarcity, but they do not assign classes to the acquired knowledge, making them unsuitable for classification. However, they can help in initializing the weights of the GAN.
   The autoencoder must learn to compress and reconstruct samples as accurately as possible, encouraging it to acquire an initial understanding of the dataset. The autoencoder consists of an encoder (E in the figure) and a decoder (Δ). As the encoding and decoding steps resemble the tasks of the discriminator (D) and the generator (G), respectively, weight-sharing by using the same network topologies allows transferring the knowledge of the trained autoencoder to the uninitialized GAN, thus complementing each other's strengths and weaknesses.
3) The main networks of ResBaGAN are D and G, collectively referred to as the GAN module in the figure. Once initialized from the autoencoder, these networks are trained in an adversarial manner, so that D learns on both the real samples and the samples that G synthesizes.
   To facilitate G the learning of minority classes, D combines its real/fake prediction and class prediction in a single output. This output contains $N+1$ elements, where N denotes the number of possible classes, and samples suspected as fake are assigned the additional class. The specific network topologies of the D and G networks (and thus E and Δ) in ResBaGAN will be detailed in Section III-C.

Therefore, the ResBaGAN training process involves three steps:

1) *Learning of the autoencoder module:* The autoencoder trains on all of the available samples without considering their labels, to produce the initial understanding of the data distribution. L2 loss, shown as the dotted line at the top of the autoencoder, guides the training process. This loss is calculated as the mean squared difference between the input and the reconstructed samples, which the autoencoder aims to minimize. Given a reconstructed sample $\hat{y}$ and its target y, the L2 loss is as follows:

$$L_{L2} = (\hat{y}_i - y_i)^2. \tag{1}$$

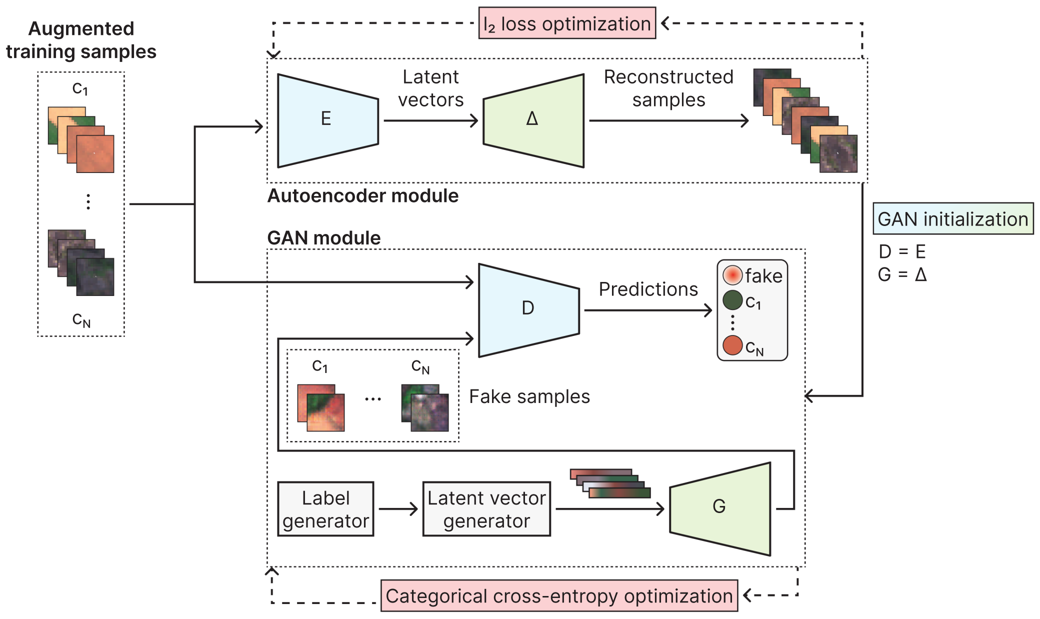


Fig. 2. Neural network architecture of ResBaGAN. Its main features are a BAGAN-based design that incorporates an improved ResNet-based classifier (D). The input augmented training samples are obtained by using the extraction procedure presented in Fig. 3.

2) *GAN initialization:* All of the learned parameters are transferred from the autoencoder to the GAN networks, enabling them to acquire the knowledge of the autoencoder. This is shown in the rightmost line of the figure.

3) *Learning of the GAN module:* Owing to the use of the transferred weights as values, the training starts from a more stable point than randomly initialized parameters, as the GAN refines the initial understanding of data. Categorical cross-entropy loss, represented by the dotted line at the bottom of the Fig. 2, guides this learning process. This loss is calculated as the difference between the predicted and the expected class-probability distributions for a sample. Given a predicted distribution $\hat{y}$, obtained by applying the softmax function to D's raw output, and a one-hot encoded class target y, the cross-entropy loss is as follows:

$$L_{CE} = -\sum_{c=1}^{N+1} y_c \log(\hat{y}_c) \tag{2}$$

where y_c denotes the target probability of the sample belonging to class c. Both D and G strive to minimize their respective losses, L_D and L_G, which are based on the cross-entropy loss. As D wants to accurately classify the real samples and assign the fake label to the samples from G, its loss can be split into a real part and a fake part:

$$L_D = L_D^{\text{real}} + L_D^{\text{fake}}. \tag{3}$$

The real part is obtained by applying the cross-entropy loss between the predicted class probabilities of the real samples and their reference labels, and the fake part of the loss is obtained by applying the cross-entropy loss between the predicted probabilities of synthetic samples and the fake label. In contrast, G's objective is to ensure that D assigns its generated fake samples to the class that they intend to represent. To do so, the cross-entropy loss is applied between the predicted probabilities of the synthetic samples and their intended labels. Training proceeds by alternately optimizing the discriminator and the generator, which can be summarized in a minimax manner as follows:

$$\begin{aligned}&\min_G \max_D L(D,G)\\&= \mathbb{E}x \sim p_{\text{data}}(x), y \sim p_{\text{data}}(y)[\log D(y|x)]\\&+\mathbb{E}z \sim p_z(z), y \sim p_{\text{data}}(y)[\log(1-D(N+1|G(z,y)))].\end{aligned} \tag{4}$$

The first term corresponds to the probability of all real samples x, drawn from the $p_{\text{data}}(x)$ data distribution, being correctly classified according to their labels y. The second term represents the probability of all synthetic samples being classified as the fake class $N+1$; z is a latent vector drawn from the latent space distribution $p_z(z)$, which is conditioned by the intended class y in $G(z,y)$ to produce a synthetic sample.

Once all of the neural networks in ResBaGAN have been trained, the fake output of D is disabled, yielding the final classifier.

B. *Sample Extraction via Superpixel Segmentation and Traditional Augmentation*

To further improve ResBaGAN in handling scarce and imbalanced datasets, a specialized procedure for extracting samples from the dataset is introduced. This procedure leverages two extensively studied techniques in remote sensing: superpixel segmentation and traditional data augmentation. Fig. 3 illustrates

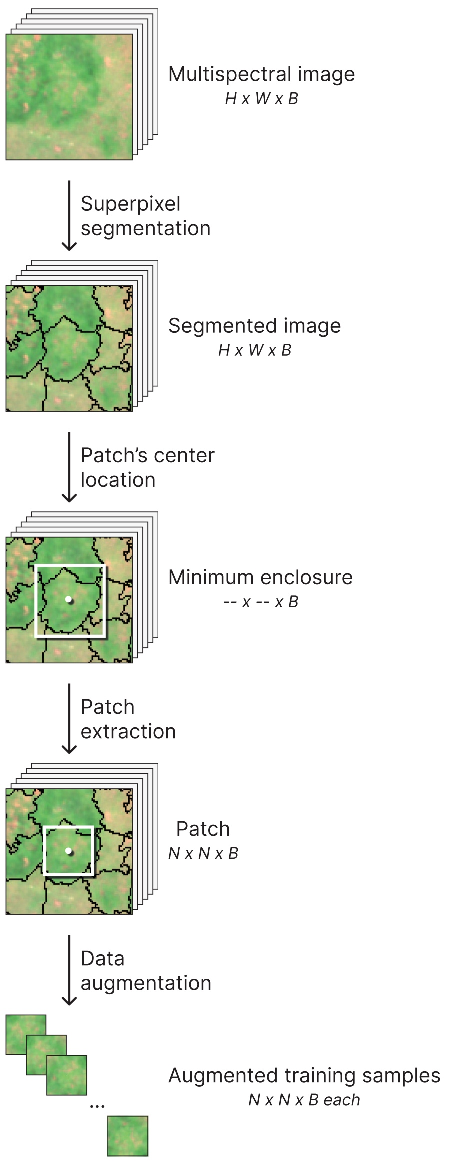


Fig. 3. Procedure for extracting training samples from the dataset in ResBaGAN. A superpixel segmentation guides the process, and traditional augmentation techniques enrich the learning data that enters the network architecture in Fig. 2. The augmentation is not performed on samples destined for the validation or test sets.

Algorithm 1: Superpixel-Guided Sample Extraction.

```
function EXTRACTPATCHES($dataset, N$)
    Segment $dataset$ into superpixels
    Initialize $patches \leftarrow \emptyset$
    for all superpixels with reference data
        Measure minimum enclosure
        Locate central point
        Extract a $patch$ of $N \times N$ pixels
        Assign class via majority voting
        Add $patch$ to $patches$
    end for
    return $patches$
end function
```

Algorithm 2: Traditional Augmentation on Training Samples.

```
function AUGMENT($sample$)
    Select random rotation from $\{0°, 90°, 180°, 270°\}$
    Rotate $sample$
    if random() $< 0.5$ then
        Flip $sample$ horizontally
    end if
    if random() $< 0.5$ then
        Flip $sample$ vertically
    end if
    return $sample$
end function
function GETTRAININGBATCH($patches, batch\_size$)
    Initialize $batch \leftarrow \emptyset$
    while $|batch| < batch\_size$ do
        Select random $patch$ from $patches$
        $augmented\_sample \leftarrow AUGMENT(patch)$
        Add $augmented\_sample$ to $batch$
    end while
    Initialize $fake\_samples \leftarrow \emptyset$
    Initialize $samples\_per\_class \leftarrow \frac{batch\_size}{classes}$
    while $|fake\_samples| < samples\_per\_class$ do
        Generate $synthetic\_sample$ with $G$
        Add $synthetic\_sample$ to $fake\_samples$
    end while
    Add $fake\_samples$ to $batch$
    return $batch$
end function
```

this procedure step by step, yielding the training samples fed to networks described in Section III-A.

A superpixel segmentation groups similar pixels in an image into homogeneous and contiguous regions called superpixels. These regions have a relatively constant size but not a specific shape, as they adapt to the objects in the scene [42]. Thus, superpixels can be used as larger pixels for image processing and have been widely used in remote sensing classification [3], [26], [43], [44]. Instead of extracting a patch for each pixel in a sliding-window fashion [45], one patch per superpixel can be extracted, assigning the predicted class to all of its pixels. If the patch size is adjusted to fit within the average size of the superpixels, patches centered on the superpixels predominantly contain pixels of a single class. This approach helps avoid the noise introduced in learning data by patches containing multiple classes, particularly at the object edges, thus facilitating the modeling of the different classes for neural networks.

On this basis, ResBaGAN extracts samples from the dataset by using a superpixel segmentation as guidance, as shown in Fig. 3 and detailed in Algorithm 1. Moreover, traditional data augmentation techniques are applied to all of the training samples, as described in Algorithm 2.

Regarding the superpixel segmentation algorithm, numerous options exist in the literature, such as SLIC [46], ETPS [47], and

Algorithm 3: ResBaGAN Classification Method.

```
function ResBaGAN($target\_dataset$)
    $patches \leftarrow$ SampleExtraction($target\_dataset$)
    Training($patches$)
    Classification($patches$)
end function
function SampleExtraction($target\_dataset$)
    Segment $target\_dataset$ using WP
    Extract a sample per superpixel with reference data
    Split samples into train, validation, and test
end function
function Training($patches$)
    Train autoencoder on augmented train samples to learn
     data distribution
    Transfer autoencoder parameters to GAN module
    Train GAN on augmented train samples and synthetic
     samples from $G$
end function
function Classification($patches$)
    Disable additional fake class in $D$'s output
    Map target dataset using $D$ as the classifier
end function
```

SEEDS [48]. Typically, these algorithms are based on gradient descent or graphs [42], with the latter being more computationally expensive and lacking control of the segment size or regularity [46]. WP [49] is a popular gradient descent algorithm that provides good-quality segmentations with moderate computational consumption [42], [46], and allows for the customization of the segment size and regularity. Therefore, WP is chosen as the superpixel segmentation algorithm for ResBaGAN.

As all ResBaGAN components are explained, its step-by-step operation can be summarized as Algorithm 3.

C. Network Topologies

As explained in Section III-A, the autoencoder and the GAN modules share network topologies. First, the topology of the residual discriminator (D in Fig. 2) will be discussed, which is also applicable to the encoder (E). Then, the design of the generator (G) will follow, which is also applicable to the decoder (Δ).

The ResBaGAN classifier (D) features a ResNet-based topology, with the novelty of integrating data from different levels of abstraction to enhance generalization, as it will be detailed in the following paragraph. This architecture is depicted in Fig. 4. When compared to shallow CNNs, a residual approach allows for more efficient use of the available learning data and pushes the capabilities of G further owing to the adversarial nature of GANs.

More specifically, the classifier consists of three residual stages (colored differently in the figure), each containing three residual blocks that use the same number of convolutional filters.

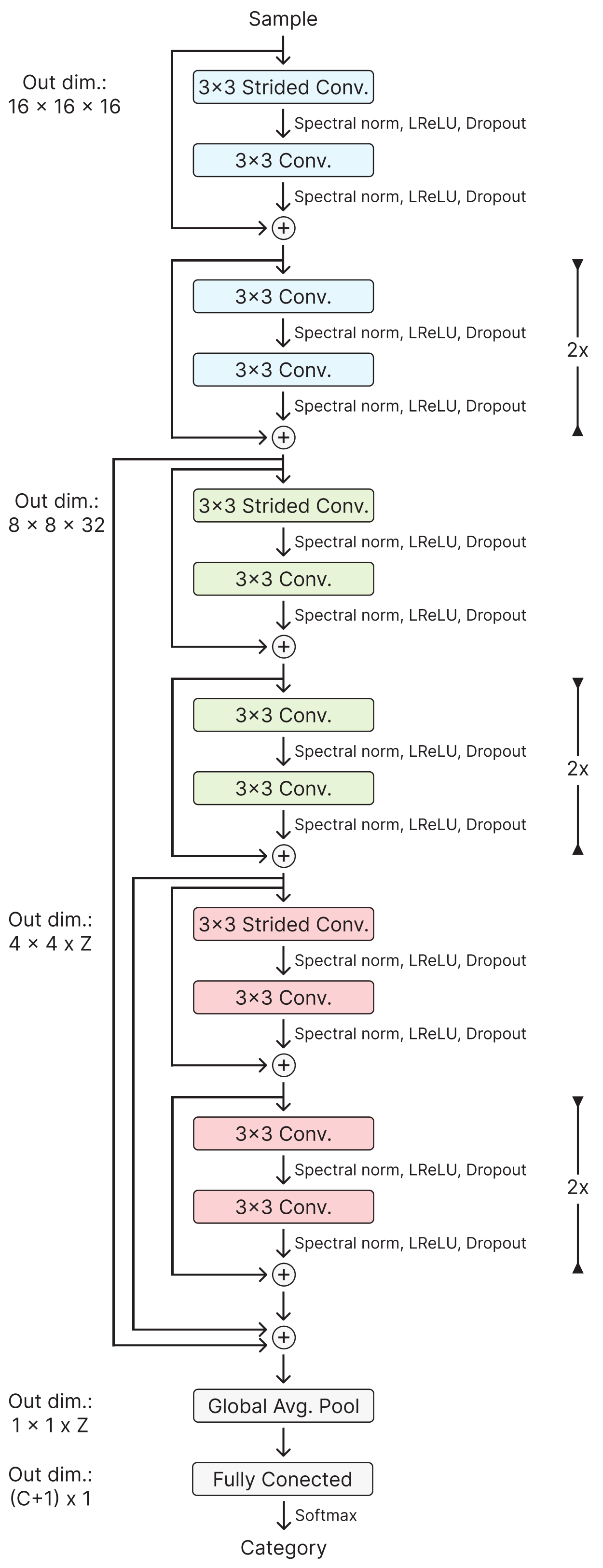


Fig. 4. Diagram of the classifier (D) for ResBaGAN. The output dimensions correspond to an input sample with dimensions $H \times W \times B = 32 \times 32 \times B$. A different color is assigned to each residual stage. For compactness, the figure does not elaborate either on the convolutional layers that adapt the input of each stage to its dimensionality, or on the layers that adapt the output of the first two stages to the final feature fusion.

TABLE I
DETAILS OF THE LAYERS IN THE CLASSIFIER (D) FOR RESBAGAN

# stage	Component	Output dimensions	Activation	Filter size	Stride
1	Residual Block	$16 \times 16 \times 16$	(see Section IV-A4)	3×3	2×2
	Residual Block	$16 \times 16 \times 16$	(see Section IV-A4)	3×3	1×1
	Residual Block	$16 \times 16 \times 16$	(see Section IV-A4)	3×3	1×1
2	Residual Block	$8 \times 8 \times 32$	(see Section IV-A4)	3×3	2×2
	Residual Block	$8 \times 8 \times 32$	(see Section IV-A4)	3×3	1×1
	Residual Block	$8 \times 8 \times 32$	(see Section IV-A4)	3×3	1×1
3	Residual Block	$4 \times 4 \times Z$	(see Section IV-A4)	3×3	2×2
	Residual Block	$4 \times 4 \times Z$	(see Section IV-A4)	3×3	1×1
	Residual Block	$4 \times 4 \times Z$	(see Section IV-A4)	3×3	1×1
–	2D Average Pooling	$1 \times 1 \times Z$	–	4×4	4×4
–	Flatten	$Z \times 1$	–	–	–
–	Dense Layer	$(C + 1) \times 1$	Softmax	–	–

The output dimensions correspond to an input sample with dimensions $H \times W \times B = 32 \times 32 \times B$.

The growing filters allow for extracting increasingly complex visual features. Additional convolutional layers precede each stage to ensure appropriate input dimensionality, and the first convolutional layer in each stage has a stride of 2 for learned dimensionality reduction. In contrast to the original ResNet architecture, this improved design fuses the features extracted from the last stage with those derived from all preceding stages, by adding the corresponding feature maps. This approach merges data from the low-, mid-, and high-level abstractions into the input for the final layers, leading to better generalization. Two additional convolutional layers map the dimensionality of the outputs from Stage 1 and Stage 2 to Stage 3 to perform the addition. A global average-pooling layer transforms the fused feature maps into a vector of features processed by a fully connected layer for the final classification.

The details of the classifier's layers are presented in Table I. For brevity, the table omits the convolutional layers adapting the input dimensionality between stages and those adapting the output from the first two stages for feature fusion; they have 3×3 filters and use the same activation function as the other layers. As usual in residual networks, a dropout layer follows each convolutional layer to stabilize training and address problems such as overfitting [50]. The selected dropout probability and activation function will be explained in Section IV-A4. Spectral normalization [51] is also applied to all convolutional layers, as it commonly improves GAN networks.

In the design of ResBaGAN, achieving an equilibrium between G and D is crucial for stable learning and thus, sustained adversarial behavior. This balance enables both networks to progress together, rather than allowing one to outperform the other significantly, disrupting the adversarial learning and impeding their ability to compete. Such disruption would prevent G from learning to perform a deep learning-based augmentation that is useful to D. Thus, a standard GAN generator has been found to be an ideal counterpart for the residual classifier in ResBaGAN.

As depicted in Fig. 5, the generator consists of a series of transposed convolutional layers. The initial embedding layer [52] transforms the desired class into a Z-element vector, combined

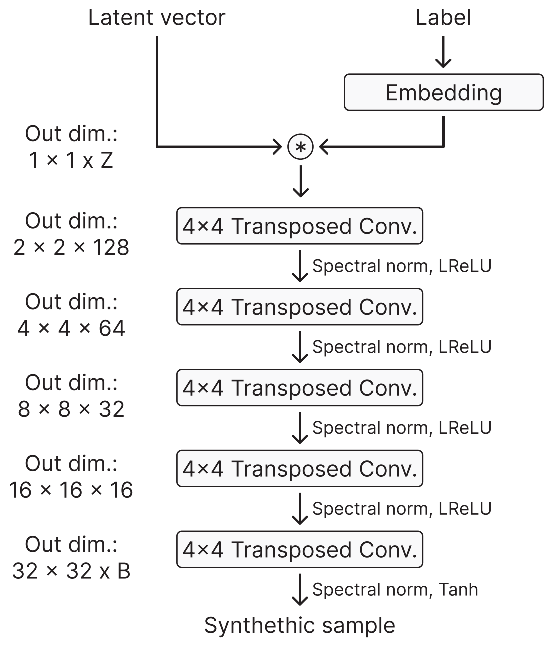


Fig. 5. Diagram of the generator (G) for ResBaGAN. The output dimensions correspond to an input latent vector with Z elements.

with the latent vector via a dot product operation. The result, interpreted as a patch with the dimensions $N \times N \times B = 1 \times 1 \times Z$, is supplied to the transposed convolutions to generate the synthetic sample by gradually increasing its resolution.

The details of the generator's layers are presented in Table II. Notably, the choice of Z can significantly impact the synthesis performance. Short latent vectors may struggle to create realistic samples, whereas large vectors could be too difficult to handle effectively. Hence, the impact of Z will be examined in Section IV-A4. The activation function also follows the rationale in Section IV-A4. The only exception is the activation of the last layer, which uses a tangent function, as all the dataset is scaled to the $[-1, 1]$ range before being input to ResBaGAN. Spectral normalization is also applied to all of the convolutional layers.

TABLE II
DETAILS OF THE LAYERS IN THE GENERATOR (G) FOR RESBAGAN

Component	**Output dimensions**	**Activation**	**Filter size**	**Stride**
Embedding	$Z \times 1$	–	C	–
2D Transposed Convolution	$2 \times 2 \times 128$	(see Section IV-A4)	4×4	2×2
2D Transposed Convolution	$4 \times 4 \times 64$	(see Section IV-A4)	4×4	2×2
2D Transposed Convolution	$8 \times 8 \times 32$	(see Section IV-A4)	4×4	2×2
2D Transposed Convolution	$16 \times 16 \times 16$	(see Section IV-A4)	4×4	2×2
2D Transposed Convolution	$32 \times 32 \times B$	Tangent	4×4	2×2

The output dimensions correspond to an input latent vector with Z elements.

TABLE III
DESCRIPTIONS OF THE DATASETS USED IN THIS WORK

Dataset	**Location**	**Size**	**Labeled pixels**	**Labeled superpixels**
Das Mestas River	43°38′30.24″ N 7°58′44.08″ W	9040 × 4915 px (920 × 510 m^2)	27.17 Mpx	57 058
Eiras Dam	42°20′45.26″ N 8°30′10.81″ W	18 221 × 5176 px (2260 × 660 m^2)	38.35 Mpx	27 865
Ermidas Creek	42°22′48.43″ N 8°24′53.36″ W	18 972 × 11 924 px (2190 × 1390 m^2)	65.56 Mpx	36 063
Ferreiras River	43°32′46.96″ N 7°57′16.66″ W	9219 × 9335 px (740 × 750 m^2)	40.19 Mpx	82 856
Mera River	43°34′31.15″ N 7°52′34.81″ W	22 116 × 10 718 px (2770 × 1370 m^2)	99.24 Mpx	125 142
Oitavén River	42°22′15.48″ N 8°25′47.07″ W	6722 × 6689 px (760 × 760 m^2)	22.04 Mpx	35 838
Ulla River	42°49′14.32″ N 7°54′5.29″ W	16 555 × 4220 px (1420 × 380 m^2)	46.01 Mpx	3548
Xesta Basin	42°23′34.95″ N 8°21′21.23″ W	17 202 × 3848 px (1945 × 435 m^2)	40.00 Mpx	82 218

The column "Labeled superpixels" is obtained after applying the WP segmentation.

IV. EXPERIMENTS

In this section, ResBaGAN is evaluated in terms of classification performance to assess both its general effectiveness and the contributions of its individual components, such as the synthesis quality of the GAN-based augmentation. ResBaGAN's capabilities are compared with simpler classification approaches, such as standalone CNNs and other GAN-based approaches like ACGAN.

The section is organized as follows. First, the datasets, metrics, and experimental environment used for the assessment are outlined in Section IV-A. A range of design choices that were left open in Section III are also addressed, such as the selection of activation functions, the optimizer, and the number of training epochs, among other factors. Then, the experimental results are presented and analyzed in Section IV-B.

A. Experimental Setup

1) Datasets: Eight large, very high-resolution multispectral images of natural regions with dense vegetation were used [3]. These images were captured in 2018, 2019, and 2020, by flying a UAV at a 120 m altitude over various river basins in Galicia, Spain, resulting in a spatial resolution of 10 cm/px. The UAV carried a MicaSense RedEdge-MX multispectral camera, capturing five spectral bands corresponding to wavelengths of 475 nm (blue), 560 nm (green), 668 nm (red), 717 nm (red edge), and 842 nm (near-infrared). Table III details the specific locations and dimensions of the scenes.

The composite color image and the reference data for each dataset can be found in Fig. 6. Table IV enumerates the ten identifiable classes in the reference data, detailing the number of samples in each dataset. These classes range from native vegetation to human-made structures such as roads or buildings. It is important to highlight the strong imbalances between classes in all datasets, with minority classes having multiple orders of magnitude fewer samples at times. This imbalance introduces a bias toward majority classes, potentially preventing a balanced classification accuracy across the different classes.

All of the datasets were segmented using the WP algorithm by choosing an average size of 400 px/superpixel, allowing a minimum size of 100 px/superpixel, and employing a compactness factor of 0.5 points, following the approach in [3]. The extracted patches had spatial dimensions of $N \times N = 32 \times 32$ px. In addition, all the data were normalized to the $[-1, 1]$ range. For a given dataset, all associated experiments used the same subset of 15% training samples, and 5% validation samples to monitor the training progress by identifying potential issues like overfitting. This limited availability of learning data is enforced

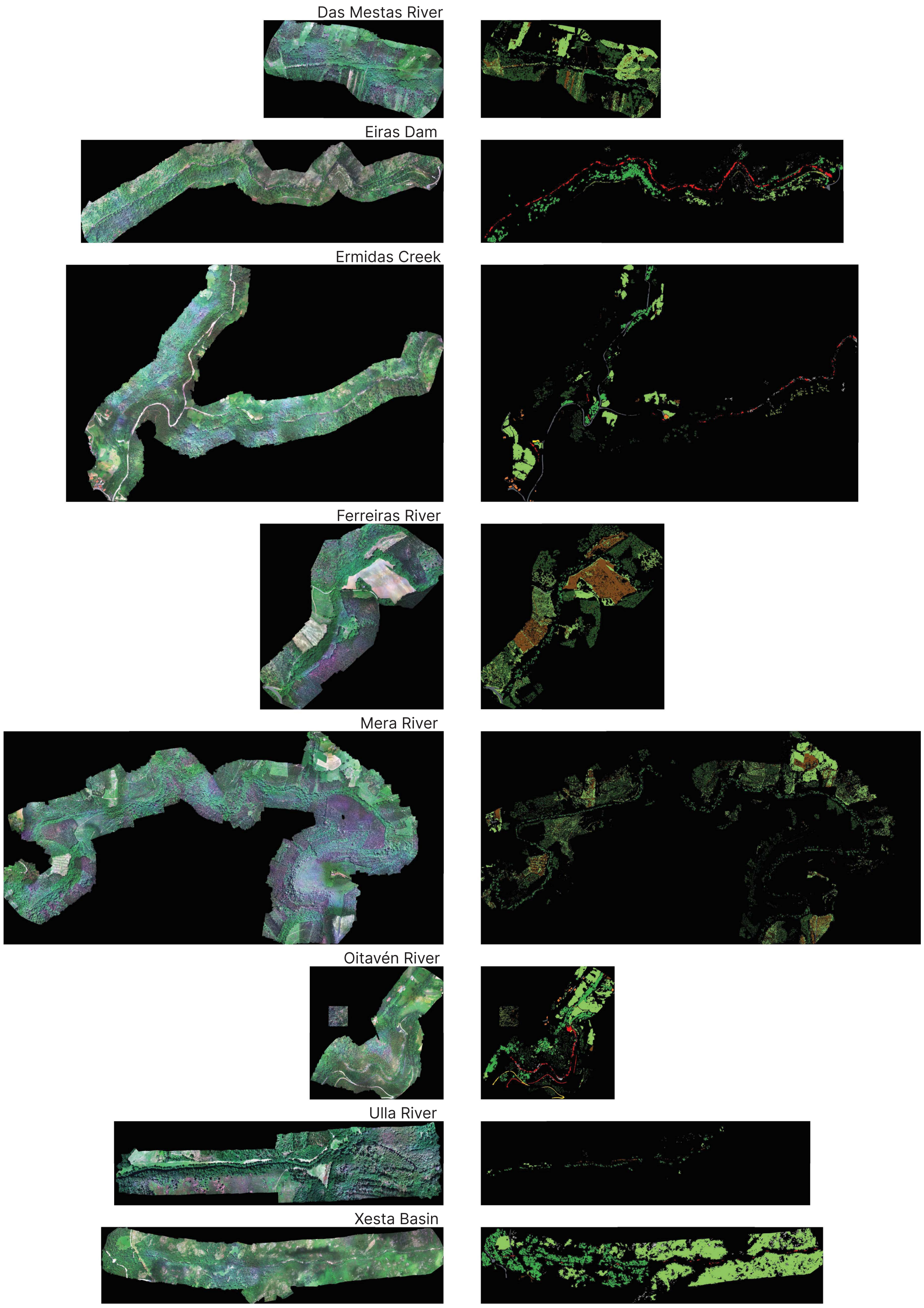


Fig. 6. Composite color images (left) and reference data (right) for the datasets used in this work [3]. All representations follow the same size scale. The class corresponding to each color in the reference data is described in Table IV; black means no reference data are available.

TABLE IV
IDENTIFIABLE CLASSES IN THE DATASETS USED IN THIS WORK

#	Color	Class	Das Mestas River	Eiras Dam	Ermidas Creek	Ferreiras River	Mera River	Oitavén River	Ulla River	Xesta Basin
1.	🟥	Water	0	3557	890	0	0	1515	0	488
2.	🟫	Bare soil	7076	867	874	24 377	18 535	978	495	411
3.	⬜	Rocks	0	2923	1846	0	0	749	0	4632
4.	⬛	Asphalt	0	340	3597	685	395	202	32	700
5.	🟨	Concrete	0	182	126	0	10	774	0	0
6.	🟧	Tiles	0	47	608	0	18	448	0	0
7.	🟩	Meadows	23 417	5722	13 052	22 364	43 191	9376	494	52 276
8.	🟩	Native trees	2973	13 466	4559	4830	13 241	11 300	2243	23 711
9.	🟩	Pines	0	687	1527	14	0	2109	284	0
10.	🟩	Eucalyptus	23 592	74	8984	30 586	49 752	8387	0	0
		Total samples:	57 058	27 865	36 063	82 856	125 142	35 838	3548	82 218

The number of samples is reported in superpixels after applying the WP segmentation.

to make training difficult for neural networks, as allocating 15% of training often yields less than 100 reference samples for many classes, particularly impacting the minority ones.

2) Metrics: The classification performance of ResBaGAN was determined by predicting the class of each available labeled sample in the dataset and comparing the results to the reference data. The following standard pixel-level metrics in remote sensing classification [53] were computed, excluding only the central pixels from the training samples as described in [3]: OA, AA, and Cohen's Kappa (κ).

To understand the performance of ResBaGAN across majority and minority classes, the classification performance was further examined with the following classwise metrics [53]: PA, UA, and F1. PA indicates the probability that a pixel belonging to a specific class was correctly classified, whereas UA denotes the probability that a pixel classified as a certain class indeed belonged to that class. F1 is the harmonic mean of PA and UA, providing a summary of classification accuracy for individual classes.

To also assess the synthesis quality of ResBaGAN, the FID score [36] was adapted to use the full spectral resolution of the datasets. This score is a widely accepted metric for determining the quality of GAN-generated samples in RGB data. However, we have not found a standard procedure for its application in the remote sensing field.

The FID score uses a classifier network to determine the similarity between the data distribution of the synthesized samples and that of real samples. This metric leverages a pretrained Inception v3 architecture to summarize the samples into the visual features that are compared. However, as this RGB network cannot fully use the spectral resolution of remote sensing data for an accurate assessment [54], [55], it was replaced by the improved ResNet from Table I to use the full spectral resolution of the samples.

To calculate the FID score of ResBaGAN on a particular dataset, all available real samples for each class were gathered and compared with an equal number of randomly synthesized samples from G for the corresponding class. The FID score comparison was run on the features extracted by the average pooling layer of the improved ResNet. This network had been pretrained for the corresponding dataset in a standalone manner, to prevent introducing biases from the ResBaGAN architecture.

Finally, to account for the inherent randomness in the experiments due to factors such as the random parameter initialization in neural architectures, all of the experiments were repeated ten times under identical conditions, reporting metrics summaries such as mean values and 95% confidence intervals.

3) Execution Environment: All of the experiments were conducted on a computer equipped with an Intel Core i7-11700 K CPU [56], 128 GB of RAM, and an NVIDIA RTX 3080 Ti GPU with 12 GB of VRAM [57]. The system ran Ubuntu 20.04 [58], with most of the source developed using Python 3.8.13 [59] and PyTorch 1.12.0 [60] for neural architectures. CUDA 11.6.2 [61] and cuDNN 8.4.0.27 [62] were leveraged to speed up execution by using single precision arithmetic. Some auxiliary codes, such as WP segmentation, were developed using C [63] and C++ [64], compiled with GCC 9.4.0 [65]. The tool GuildAI [66] was also used for automating the hyperparameter optimization step.

4) Hyperparameter Optimization: Several design aspects of ResBaGAN were left open in Section III, such as the activation function and learning rate. To determine these factors, a hyperparameter optimization procedure was conducted to evaluate ResBaGAN's performance under many scenarios and identify the optimal design choices. In particular, the following configurations were examined:

- *Activation function:* ELU [67], LeakyReLU [68], and PReLU [69] were tested, as they generally outperform traditional options like ReLU [70].
- *Size of latent vectors (Z):* Guided by the dimensionalities of the convolutional layers in ResBaGAN, the values $Z = \{32, 64, 128\}$ were tried.
- *Dropout probability (p_{dropout}):* To cover a wide range of possibilities, the values $p_{\text{dropout}} = \{0.05, 0.20, 0.35, 0.50\}$ were tested.
- *Learning rate (α):* All of the values in the range $\alpha = \{0.0005 \ldots 0.0030\}$ with a step of 0.0005 were used.
- *Batch size ($batch_{\text{size}}$):* Larger batches speed up the training but may result in reduced generalization capabilities. To prioritize classification accuracy, the batch size was limited to moderate values such as $batch_{\text{size}} = \{32, 64, 128\}$.

All of the possible combinations resulted in 648 configurations to evaluate. To keep the cost of this phase reasonable, the number of training epochs was limited to 20 iterations, as the behavior of ResBaGAN in its initial epochs was observed to be a reliable indicator of future performance.

Once all of these tests were run, the results were filtered based on the best classification accuracy and synthesis quality of the GAN-based augmentation. Specifically, the aim was the highest classification accuracies, as determined by OA, AA, and κ, and the lowest training losses in G. Since D and G were trained in an adversarial manner, the lowest loss for G and around 50% error rate for D indicate that G synthesized samples that were so realistic that D relied on chance to tell them apart from real ones.

Considering these factors, the design choices that maximized the performance of ResBaGAN were the LeakyReLU activation function, latent vectors of size $Z = 128$, 5% dropout probability, learning rates of around 0.001, and a batch size of 32 samples.

5) On Training Neural Architectures: The ADAM optimizer [71] was employed to train any neural architecture, given its strong performance with deep and complex architectures. Its parameters were set to $\beta_1 = 0.5$ and $\beta_2 = 0.999$, as usual for GAN-based architectures [72]. All of the parameters in the neural architectures were initialized using Xavier's (or Glorot's) Initialization [73]. Regarding the number of training epochs, different options were tested while monitoring the loss of networks in ResBaGAN throughout the process. The conclusion was that 600 training epochs were sufficient for both D and G to reach peak performance in the datasets. To ensure a fair comparison, 600 training epochs were also provided to the remaining architectures, such as the autoencoder inside ResBaGAN or standalone CNNs. Validation accuracy monitoring was performed as needed to ensure that all classification approaches included in these experiments, in addition to ResBaGAN, experienced stable training without facing problems such as overfitting. Moreover, when needed, the same hyperparameter values as in ResBaGAN were applied to the other classification methods.

B. Experimental Results

1) Stability of Adversarial Training: To ensure sustained adversarial learning, the evolution of various training metrics for both D and G is analyzed in an experiment with ResBaGAN on the Ermidas Creek dataset, as depicted in Fig. 7(a) to (c).

Fig. 7(a) shows the evolution of G over all 600 training epochs. Its performance increased until around 200 iterations before stabilizing. The remaining iterations still contributed to the final performance of ResBaGAN given that D persistently lowered its training loss, as Fig. 7(b) shows. However, the continuous loss reduction in D could also be a symptom of overfitting. In this case, Fig. 7(c) reveals that the generalization consistently improved until approximately iteration 500 before stabilizing for the remaining 100 iterations. Overall, these findings demonstrate that executing 600 training epochs enabled ResBaGAN to optimize its performance for the experimental datasets.

2) Synthesis Performance: The synthesizing capabilities of ResBaGAN are compared with two simpler approaches: an ACGAN and a BAGAN-based architectures that assist the shallow

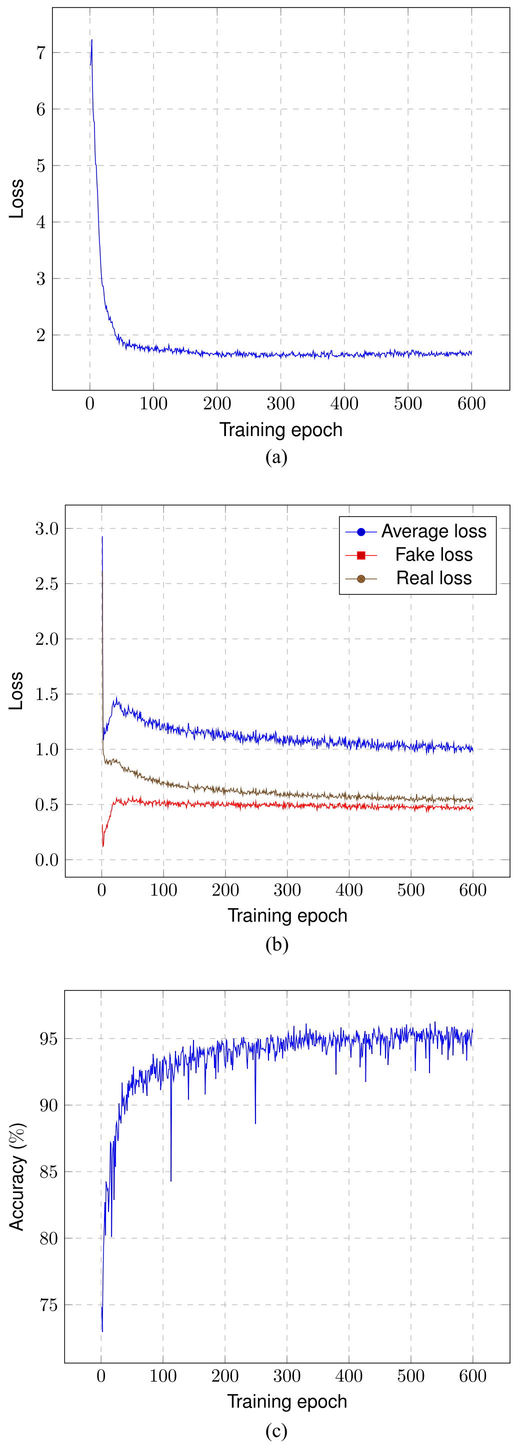


Fig. 7. Evolution of various training metrics of ResBaGAN in an experiment on the Ermidas Creek dataset. Lower values are better in the first two figures, while higher values are better in the third one. (a) G's loss. (b) D's losses. (c) D's validation accuracy.

CNN described in Table V, while using the same G network as in ResBaGAN. This BAGAN is a stripped-down version of ResBaGAN, lacking the residual classifier and support from traditional data augmentation techniques. ACGAN can be seen as an even simpler version of BAGAN, missing the autoencoder and the improved loss function for training.

TABLE V
DETAILS OF THE LAYERS IN THE SHALLOW CNN

Component	Output dimensions	Activation	Filter size	Stride
2D Convolution	$32 \times 32 \times 16$	LeakyReLU	3×3	2×2
2D Convolution	$16 \times 16 \times 32$	LeakyReLU	3×3	2×2
2D Convolution	$8 \times 8 \times 64$	LeakyReLU	3×3	2×2
2D Convolution	$4 \times 4 \times Z$	LeakyReLU	3×3	2×2
Flatten	$(Z \cdot 4 \cdot 4) \times 1$	–	–	–
Dense Layer	$Z \times 1$	LeakyReLU	–	–
Dense Layer	$C \times 1$	Softmax	–	–

The output dimensions correspond to an input sample with dimensions $H \times W \times B = 32 \times 32 \times B$. A dropout layer is also applied to each convolutional layer and the first fully connected layer.

TABLE VI
SUMMARIZED FID SCORES FOR THE THREE TESTED GAN-BASED APPROACHES ON THE OITAVEN RIVER DATASET

	ACGAN	BAGAN	ResBaGAN
Averaged–FID	124.81	47.08	**40.13**
Weighted–FID	49.73	28.75	**27.77**

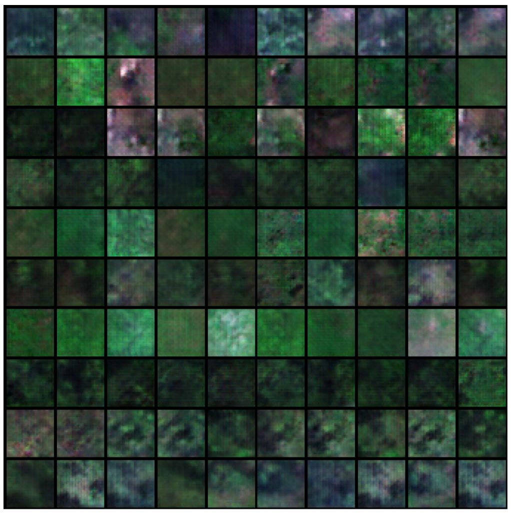
Lower values are better, and results in bold are the best within the metric.

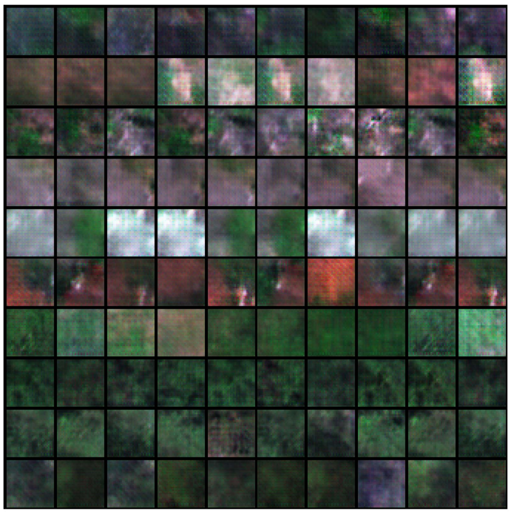
Fig. 8 displays synthesized samples for the Ermidas Creek dataset using these three GAN-based approaches. It is evident that the G network of ACGAN struggled to model the different classes. This resulted in fake samples that primarily resemble different types of vegetation, which are the majority classes in the dataset, and exhibit limited variation. In contrast, class-balancing enhancements allowed BAGAN-based approaches to perform proper data augmentation. As illustrated in Fig. 8(b) and (c), they could adequately generate fake samples for all classes, including minority ones. Furthermore, increased visual variations are present between fake samples within the same class, particularly in ResBaGAN.

The synthesis performances can also be numerically assessed using the FID score. Following the approach described in Section IV-A2, the FID scores for experiments on the Oitavén River dataset were measured and are displayed in Fig. 9 and Table VI. Fig. 9 displays the score for each class, while Table VI summarizes the FID into an averaged-FID and a weighted-FID for each GAN. The former metric averages the score among classes, whereas the latter assigns greater importance to classes with more elements.

Compared to the other approaches, the high FID scores of ACGAN reflect its limited ability to synthesize varied samples. In contrast, BAGAN-based approaches achieved significantly better FID scores, particularly in classes 1 to 6 and 9, which contain fewer samples in the Oitavén River dataset. ResBaGAN's FID scores were either comparable to or better than BAGAN's, as summarized in Table VI. The particularly enhanced averaged-FID for ResBaGAN suggests it mainly outperformed BAGAN in classes with fewer samples. We attribute this superior synthesis performance to the more complex D network in ResBaGAN compared to the shallow CNN in BAGAN; a more robust D enables G to outperform itself even further, owing to the adversarial nature of the GAN training.

3) Overall Classification Performance: Table VII presents the resulting OA, AA, and κ accuracy metrics for ResBaGAN


(a)


(b)

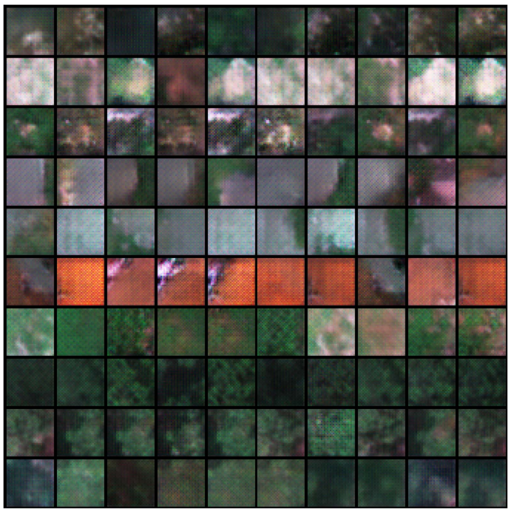

(c)

Fig. 8. Comparison of fake sample synthesis for the Ermidas Creek dataset using the three tested GAN-based approaches. Within each grid of fake samples, the different rows correspond from top to bottom to the classes listed in Table IV. Ten random fake samples are shown within each class. (a) ACGAN. (b) BAGAN. (c) ResBaGAN.

TABLE VII
RECORDED CLASSIFICATION ACCURACIES FOR ALL THE DATASETS USING ALL THE CLASSIFIERS

Dataset		FV/KELM	CNN	Improved ResNet	ACGAN	BAGAN	ResBaGAN
Das Mestas River	OA	89.27 ± 0.09	89.36 ± 0.35	90.69 ± 0.19	87.80 ± 0.48	89.74 ± 0.23	**91.50** ± 0.12
	AA	82.35 ± 0.33	79.15 ± 2.02	85.81 ± 0.56	82.67 ± 0.48	81.12 ± 2.05	**87.03** ± 0.52
	κ	82.27 ± 0.15	81.59 ± 0.55	83.88 ± 0.33	79.37 ± 0.73	82.21 ± 0.38	**85.30** ± 0.19
Eiras Dam	OA	94.45 ± 0.22	91.01 ± 0.40	95.44 ± 0.49	90.78 ± 0.34	95.97 ± 0.21	**97.69** ± 0.12
	AA	77.58 ± 2.06	63.95 ± 1.61	78.62 ± 1.01	73.82 ± 1.14	79.05 ± 1.52	**90.90** ± 0.93
	κ	91.55 ± 0.34	84.47 ± 0.66	92.20 ± 0.80	84.77 ± 0.57	93.07 ± 0.36	**96.07** ± 0.20
Ermidas Creek	OA	94.76 ± 0.21	93.16 ± 0.40	96.98 ± 0.38	93.72 ± 0.48	97.85 ± 0.24	**98.66** ± 0.42
	AA	87.42 ± 0.92	79.18 ± 1.95	88.05 ± 1.79	88.17 ± 1.02	92.44 ± 0.94	**95.96** ± 0.83
	κ	92.49 ± 0.31	87.64 ± 0.68	94.47 ± 0.70	88.84 ± 0.84	96.04 ± 0.44	**97.57** ± 0.74
Ferreiras River	OA	90.27 ± 0.04	90.25 ± 0.08	91.50 ± 0.16	89.06 ± 0.24	89.72 ± 1.40	**92.46** ± 0.24
	AA	74.69 ± 0.37	71.39 ± 0.76	76.31 ± 0.32	73.99 ± 0.38	73.32 ± 0.64	**76.99** ± 0.45
	κ	85.59 ± 0.07	84.98 ± 0.13	86.92 ± 0.23	83.36 ± 0.33	84.23 ± 2.01	**88.39** ± 0.36
Mera River	OA	87.65 ± 0.06	89.11 ± 0.49	91.26 ± 0.19	89.27 ± 0.31	89.31 ± 0.69	**92.05** ± 0.11
	AA	63.98 ± 0.93	60.17 ± 1.44	71.69 ± 3.16	65.22 ± 0.47	67.33 ± 2.87	**76.89** ± 2.77
	κ	81.34 ± 0.10	83.31 ± 0.72	86.62 ± 0.30	83.63 ± 0.46	83.68 ± 1.00	**87.85** ± 0.15
Oitavén River	OA	93.03 ± 0.20	89.87 ± 0.81	93.76 ± 0.38	90.23 ± 0.63	95.57 ± 0.36	**96.81** ± 0.71
	AA	87.18 ± 0.49	77.62 ± 0.90	84.83 ± 0.54	84.10 ± 0.98	88.42 ± 1.07	**94.40** ± 0.75
	κ	90.38 ± 0.28	84.10 ± 1.22	90.09 ± 0.59	84.87 ± 0.93	92.95 ± 0.58	**94.93** ± 1.13
Ulla River	OA	95.56 ± 0.50	98.23 ± 0.32	**99.06** ± 0.18	77.42 ± 4.75	97.66 ± 0.65	97.31 ± 2.70
	AA	88.04 ± 1.42	91.87 ± 3.05	**96.99** ± 0.90	71.93 ± 3.45	91.73 ± 2.40	95.98 ± 2.08
	κ	92.65 ± 0.86	97.27 ± 0.48	**98.55** ± 0.28	69.42 ± 5.53	96.37 ± 1.01	95.82 ± 4.23
Xesta Basin	OA	96.18 ± 0.04	96.35 ± 0.34	98.18 ± 0.11	93.07 ± 4.18	97.26 ± 0.24	**98.24** ± 0.20
	AA	81.88 ± 1.02	67.30 ± 1.18	81.68 ± 1.75	77.20 ± 0.95	71.94 ± 2.37	**88.23** ± 2.11
	κ	90.75 ± 0.12	88.38 ± 0.90	94.16 ± 0.31	81.60 ± 7.75	91.12 ± 0.69	**94.32** ± 0.69
Average	OA	92.65 ± 0.17	92.17 ± 0.40	94.61 ± 0.26	88.92 ± 1.43	94.13 ± 0.50	**95.59** ± 0.58
	AA	80.39 ± 0.94	73.83 ± 1.62	83.00 ± 1.25	77.14 ± 1.11	80.67 ± 1.73	**88.30** ± 1.30
	κ	88.38 ± 0.28	86.47 ± 0.67	90.86 ± 0.44	81.98 ± 2.14	89.96 ± 0.81	**92.53** ± 0.96

All metrics are in the range [0, 100], with higher values being better. Results in bold are the best for the corresponding dataset.

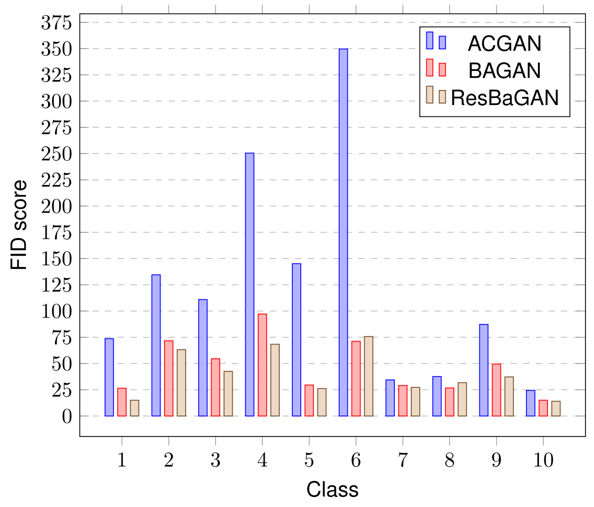


Fig. 9. Recorded FID scores for the Oitavén River dataset using the three tested GAN-based approaches. Lower values are better.

and other classification approaches on all experimental datasets. To facilitate visual comparison of the classifiers, graphs for each metric are also provided in Figs. 10–12. These graphs display the mean values and confidence intervals from the table for the top four classifiers.

The additional classification approaches compared include a traditional machine learning approach, two standalone CNNs, and the ACGAN and BAGAN described in Section IV-B2. The texture-based traditional method, proposed in [3] for the classification of the same datasets, served as a baseline for non-deep learning approaches. It employs the FV algorithm [74] for texture extraction and KELM [75] for the final classification. The standalone CNNs included the shallow CNN described in Table V, and the improved ResNet described in Table I, which also served as the classifier for FID score assessment.

First, the results reveal that the shallow CNN failed to achieve better classification accuracy than the traditional approach in most cases. These experiments demonstrate that carefully crafted traditional machine learning approaches can still outperform certain modern techniques. That said, the standalone improved ResNet significantly outperformed the traditional method. This is due to the increased potential of using a very deep convolutional architecture rather than a shallow one, to better utilize the limited available learning data.

Examining the results for ACGAN, its classification accuracies were not always better than those of the shallow CNN, even though ACGAN used it as D. These results are consistent with the analysis in Section IV-B2, as ACGAN struggled to properly synthesize fake samples, leading to confusion for the classifier and ultimately damaging its performance. In contrast, BAGAN's enhancements enabled effective data augmentation.

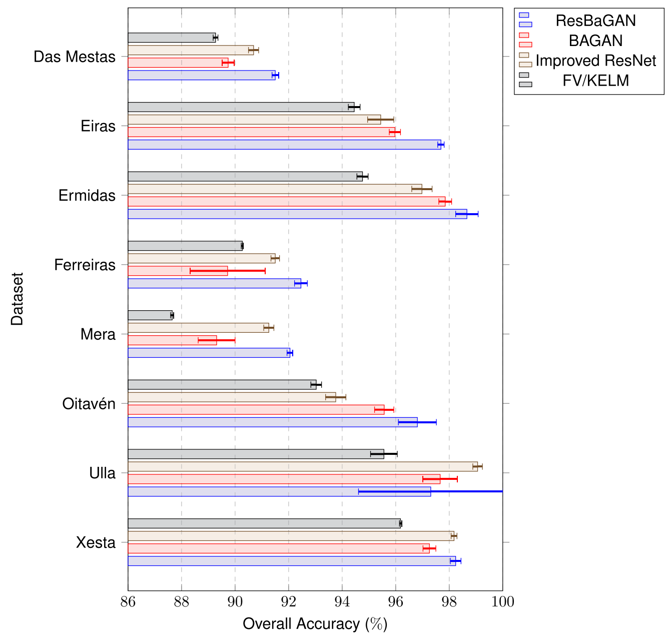


Fig. 10. Recorded OA metrics for all the datasets using the top four classifiers, including ResBaGAN. The bars represent the mean values along their confidence intervals. Higher values are better.

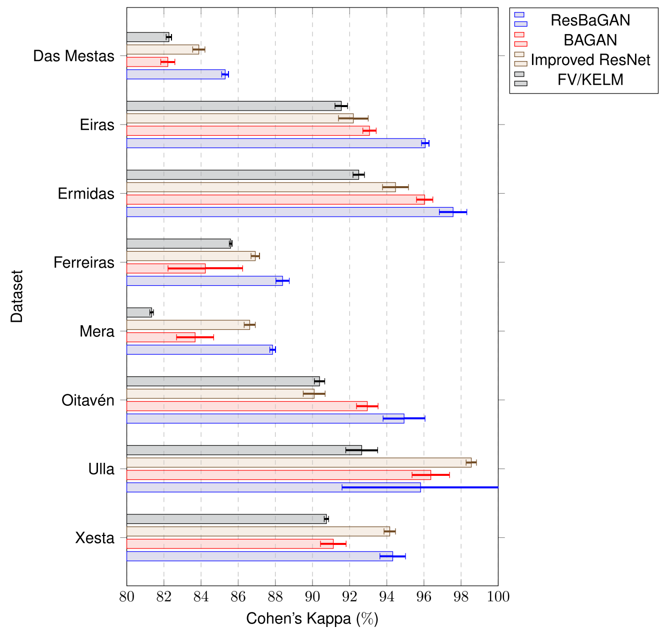


Fig. 12. Recorded κ metrics for all the datasets using the top four classifiers, including ResBaGAN. The bars represent the mean values along their confidence intervals. Higher values are better.

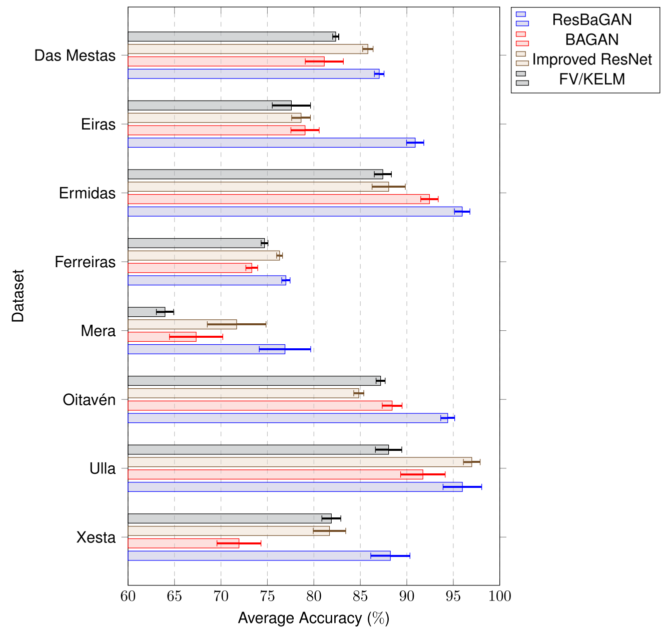


Fig. 11. Recorded AA metrics for all the datasets using the top four classifiers, including ResBaGAN. The bars represent the mean values along their confidence intervals. Higher values are better.

Despite both BAGAN and ACGAN sharing the topology of D, the classification performance of BAGAN was significantly higher than the shallow CNN in almost every case. More specifically, the most significant gains were in AA, while OA and κ had more subtle improvements. This suggests that GAN-based data augmentation primarily enhanced the accuracy in the minority classes, which should be more challenging to learn for the classifier.

Although both the improved ResNet and the BAGAN provided impressive results, ResBaGAN, the proposed classification method in this work, managed to outperform them. ResBaGAN achieved the highest and more consistent values for all the classification metrics in seven out of eight datasets, ranking among the best in the remaining one. Moreover, the confidence intervals reveal the substantial significance of ResBaGAN's enhancements in nearly all cases, as its potential minimum values consistently match or exceed the possible maximum values achieved by the other classification approaches.

4) Classwise Classification Performance: To better understand the contributions of ResBaGAN in dealing with scarce and imbalanced datasets, Table VIII presents the mean PA, UA, and F1 for ResBaGAN and the other approaches developed in this work on all the classes of elements, averaging results across datasets. A graphical depiction of F1 score is also provided in Fig. 13 for easy comparison.

All class-specific findings align with the observations in Section IV-B3. Among the standalone CNNs, the improved ResNet significantly outperformed the shallow CNN in all cases, which is expected due to its more complex network topology.

Using the F1 score to summarize class-specific accuracies, ACGAN deteriorated the overall performance of CNN. This occurred due to a slight increase in PA but a considerable decrease in UA, particularly for non-vegetation classes. This indicates that ACGAN produced numerous false positives, particularly for minority classes. This observation aligns with the fact that ACGAN's G generated synthetic samples resembling vegetation regardless of the target class, leading to a confused D. Consequently, this network classified many more samples as minority classes, leading to occasional additional hits, but also a significant increase in overconfidence and false positives.

TABLE VIII
Recorded Per-Class Classification Accuracies for All the Classes Using the Classifiers Developed in This Work

Class				CNN	Improved ResNet	ACGAN	BAGAN	ResBaGAN
1.	🟥	Water	PA	77.86 ± 8.47	92.06 ± 2.32	88.44 ± 3.61	83.73 ± 6.93	**92.89 ± 3.06**
			UA	82.14 ± 4.91	91.08 ± 2.21	72.56 ± 7.92	93.89 ± 1.48	**96.11 ± 1.36**
			F1	78.55 ± 7.07	91.40 ± 2.02	77.69 ± 6.85	86.83 ± 5.29	**94.24 ± 2.19**
2.	🟫	Bare soil	PA	69.79 ± 5.66	74.19 ± 4.58	75.88 ± 4.59	75.99 ± 5.13	**85.13 ± 2.72**
			UA	73.91 ± 4.16	82.19 ± 2.66	61.35 ± 5.83	84.59 ± 2.76	**89.53 ± 1.79**
			F1	70.46 ± 5.15	77.02 ± 3.87	66.60 ± 5.28	78.25 ± 4.29	**86.90 ± 2.10**
3.	⬜	Rocks	PA	67.87 ± 6.42	77.52 ± 4.80	79.37 ± 4.28	80.68 ± 3.32	**89.77 ± 1.92**
			UA	66.09 ± 3.74	80.72 ± 2.52	63.06 ± 5.42	79.59 ± 2.44	**91.04 ± 1.83**
			F1	65.08 ± 4.05	78.33 ± 3.16	69.63 ± 4.93	79.32 ± 1.65	**90.18 ± 1.42**
4.	⬛	Asphalt	PA	82.84 ± 4.27	96.04 ± 0.89	86.79 ± 4.57	91.91 ± 1.40	**97.22 ± 0.63**
			UA	92.32 ± 1.81	92.46 ± 1.84	65.83 ± 5.03	96.21 ± 1.13	**96.86 ± 1.32**
			F1	85.91 ± 3.02	94.08 ± 1.28	73.49 ± 4.94	93.89 ± 1.04	**96.95 ± 0.87**
5.	🟨	Concrete	PA	39.71 ± 8.58	55.45 ± 9.83	53.65 ± 9.36	58.37 ± 10.46	**84.25 ± 7.59**
			UA	85.84 ± 3.08	91.64 ± 2.44	37.27 ± 7.57	90.54 ± 5.08	**95.49 ± 1.54**
			F1	46.97 ± 8.58	61.90 ± 9.71	43.09 ± 8.16	64.72 ± 10.18	**86.60 ± 6.98**
6.	🟧	Tiles	PA	51.29 ± 10.56	71.48 ± 10.19	65.26 ± 11.17	71.42 ± 10.48	**80.87 ± 8.70**
			UA	87.46 ± 5.10	91.35 ± 4.52	42.00 ± 10.03	**98.10 ± 0.82**	94.15 ± 5.93
			F1	56.88 ± 9.66	74.60 ± 9.26	49.07 ± 10.63	76.66 ± 10.07	**83.36 ± 8.35**
7.	🟩	Meadows	PA	90.47 ± 1.99	93.69 ± 1.51	89.37 ± 2.01	92.75 ± 1.83	**94.16 ± 1.60**
			UA	91.10 ± 1.77	93.52 ± 1.37	92.75 ± 1.60	92.91 ± 1.56	**94.55 ± 1.26**
			F1	90.72 ± 1.83	93.58 ± 1.40	90.84 ± 1.74	92.78 ± 1.66	**94.27 ± 1.36**
8.	🟩	Native trees	PA	85.63 ± 2.77	94.33 ± 1.02	84.93 ± 2.42	91.46 ± 2.03	**95.86 ± 0.81**
			UA	84.48 ± 3.18	90.25 ± 2.18	84.83 ± 3.76	88.63 ± 3.34	**93.61 ± 1.74**
			F1	84.32 ± 2.76	92.00 ± 1.51	83.34 ± 2.64	89.12 ± 2.69	**94.53 ± 1.14**
9.	🟩	Pines	PA	55.63 ± 9.79	65.34 ± 10.34	61.47 ± 9.72	61.96 ± 9.59	**79.17 ± 9.04**
			UA	83.39 ± 3.32	91.04 ± 2.69	61.18 ± 9.54	**93.71 ± 1.56**	88.03 ± 6.15
			F1	60.01 ± 9.12	69.28 ± 9.76	60.52 ± 9.62	68.29 ± 9.20	**80.75 ± 8.59**
10.	🟩	Eucalyptus	PA	77.43 ± 7.03	85.04 ± 5.03	78.17 ± 5.64	89.14 ± 2.47	**93.77 ± 1.08**
			UA	78.93 ± 4.53	88.20 ± 2.61	75.92 ± 7.96	91.97 ± 1.00	**94.43 ± 1.18**
			F1	75.50 ± 6.18	85.08 ± 4.09	75.80 ± 7.36	90.12 ± 1.41	**93.95 ± 0.70**

The results are averaged across datasets. All metrics are in the range [0, 100], with higher values being better. Results in bold are the best for the corresponding class.

TABLE IX
Aggregation of All Recorded Confusion Matrices for ResBaGAN Across All Ten Experiments for Each Dataset

	Prediction									
Target	**Water**	**Bare soil**	**Rocks**	**Asphalt**	**Concrete**	**Tiles**	**Meadows**	**Native trees**	**Pines**	**Eucalyptus**
Water	4 949 931	4027	26 089	29 877	33 416	59	4229	10 208	367	1957
Bare soil	3394	39 827 912	18 154	12 762	6194	14 746	3 245 940	5540	1325	409 383
Rocks	33 911	9978	2 231 260	22 595	6942	187	230 460	962	249	5026
Asphalt	22 948	24 722	10 007	5 351 937	7689	1006	15 495	5911	16	2639
Concrete	9743	18 478	9230	22 058	874 745	398	4738	1163	10	67
Tiles	5	41 719	3094	3081	12	1 338 723	10 165	1873	66	1752
Meadows	1116	2 883 351	169 030	40 452	4146	3033	195 858 563	1 533 906	41 286	2 829 437
Native trees	5558	16 477	2388	10 998	549	2784	1 237 633	47 212 401	80 922	320 210
Pines	549	986	3784	48	12	8	22 764	88 367	1 914 330	11 742
Eucalyptus	430	609 194	623	180	18	0	2 771 576	139 884	11 244	60 907 361

In contrast, the enhancements in BAGAN provided an effective GAN-based augmentation for the CNN, significantly increasing all metrics to match the performance of the improved ResNet. BAGAN and the ResNet showed varied performance, with one method slightly outperforming the other at times, as summarized by the F1 score. Interestingly, the improved ResNet tended to improve PA the most, achieving better values than BAGAN in six out of ten classes, while BAGAN was more effective in increasing UA, with better values in six out of ten classes.

Finally, ResBaGAN further improved performance across all classes, achieving the best PA, UA, and F1 scores in almost every case. These enhancements were particularly noticeable in the minority classes. For example, *concrete* is consistently one of the scarcest classes, and ResBaGAN boosted its F1 by 22%. The solid performance of ResBaGAN when facing scarce and imbalanced data is further exemplified by the confidence intervals for F1 scores. Even at its lowest potential, ResBaGAN consistently obtains competitive results compared to the improved ResNet and BAGAN. This behavior is further reiterated in the aggregated confusion matrix for ResBaGAN across all ten experiments for each dataset, depicted in Table IX. The strong diagonal dominance indicates high classification accuracy across all classes.

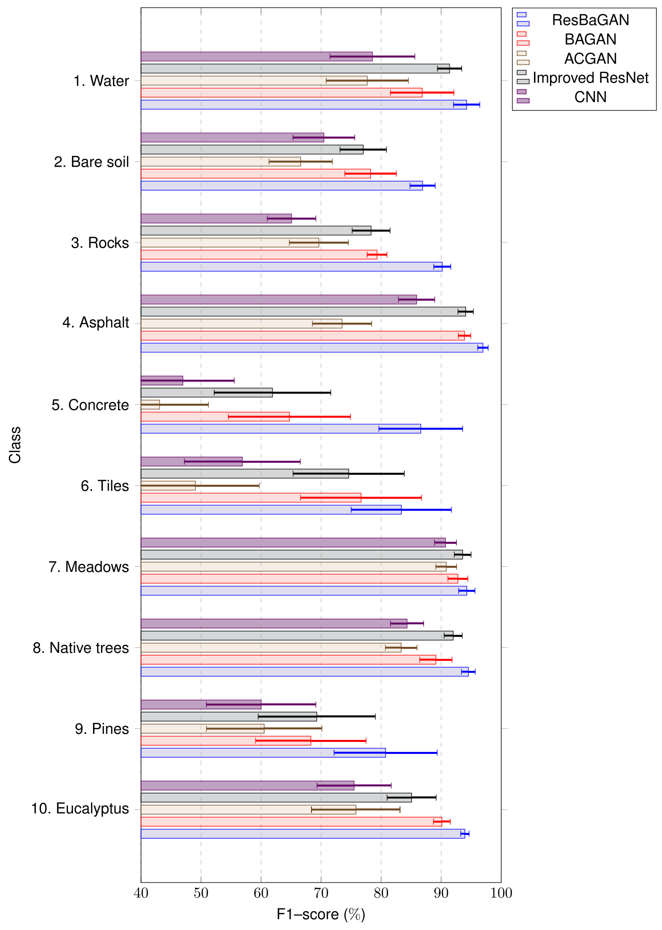


Fig. 13. Recorded F1 scores for all the classes using the classification approaches developed in this work, averaging results across datasets. The bars represent the mean values along their confidence intervals. Higher values are better.

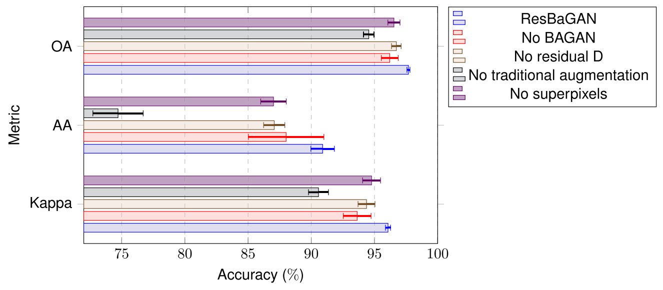


Fig. 14. Recorded OA, AA, and κ metrics for the Eiras Dam dataset in the ablation study of ResBaGAN. The bars represent the mean values along their confidence intervals. Higher values are better.

TABLE X
COMPUTATIONAL COST OF THE DEEP LEARNING METHODS USED IN THIS WORK. THE RESULTS ARE PRESENTED AS AVERAGED SPEEDUP VALUES FOR TRAINING AND CLASSIFICATION ACROSS ALL EXPERIMENTS, RELATIVE TO THE SHALLOW CNN. HIGHER VALUES ARE BETTER

	CNN	Improved ResNet	ACGAN	BAGAN	ResBaGAN
Training	1.00×	0.50×	0.52×	0.34×	0.16×
Classification	1.00×	0.59×	0.87×	0.85×	0.49×

5) Ablation Study: To assess the importance of the various components of ResBaGAN to its final performance, further experiments were conducted by removing one component at a time and comparing the resulting classification accuracy to the baseline ResBaGAN. These additional experiments were run on the Eiras Dam dataset, with reported OA, AA, and κ. Specifically, the following four alterations were examined:

- Removing the BAGAN innovations, namely the autoencoder and the improved loss function, transforming ResBaGAN into an ACGAN-derived architecture.
- Replacing the residual topology in D with the shallow CNN described in Table V.
- Removing the traditional augmentation techniques applied to training samples.
- Removing the superpixel segmentation-guidance from the sample extraction procedure, to instead extract one patch per pixel in a sliding window manner. The training used the same number of samples as with superpixels.

The results of the ablation experiments are represented in Fig. 14. It is evident that removing any of ResBaGAN's components significantly impacted its performance, as the potential maximum accuracies are always lower than, or comparable at most, to the minimum expected accuracies in the baseline ResBaGAN.

Interestingly, AA was the most affected accuracy metric. Given that it assigns equal weights to minority and majority classes, this observation highlights the particular importance of all ResBaGAN components in developing an effective data augmentation framework to adequately assist the classifier in learning minority classes with limited data.

6) Computational Cost: Finally, the computational cost of the various deep learning methods tested in this work is compared. The cost is presented in Table X as averaged speedup values for the training and classification phases across all experiments, using the shallow CNN as a baseline.

First, the improved ResNet required approximately twice as much computation time as the shallow CNN during both phases. This is expected due to the much larger neural topology. In contrast, ACGAN also required twice as much training time compared to the CNN, as it uses this network as D and a companion G network of similar complexity. However, the classification time for ACGAN was only slightly higher than the CNN, as G did not play a role in this phase. We attribute the minor slowdown to the PyTorch overhead of having both networks on the GPU.

BAGAN further increased the training time by requiring the learning of the autoencoder before the GAN networks themselves. However, as the autoencoder did not play a role during classification, the inference time remained roughly the same as that of ACGAN.

Finally, ResBaGAN exhibits increased complexity due to combining features from both the improved ResNet and the BAGAN, in addition to the traditional augmentation techniques, resulting in the longest training times, and slightly slower classification compared to the improved ResNet. We attribute this slight difference again to the PyTorch overhead of having multiple networks on the GPU.

V. Discussion

Enhancing the accuracy of remote sensing image classification is crucial for numerous monitoring tasks of the Earth's surface. Although deep learning techniques are known for their excellent classification performance, they usually require large amounts of data to unleash their full potential. This can be especially challenging for remote sensing applications, as datasets often have limited labeled data and class imbalances [4]. Data augmentation techniques, particularly those based on GAN architectures [28], present a promising and powerful solution for generating rich additional learning data [29], as demonstrated in this work. BAGAN [35] is a particularly interesting GAN architecture, as it is specifically designed to address the aforementioned limitations. Despite its potential, to the best of our knowledge, the application of BAGANs in remote sensing has been limited to object detection [40] until now.

This study shows that BAGAN-inspired architectures outperform more common GAN approaches like ACGAN [39] in providing effective deep learning-based data augmentation, particularly when working with limited and imbalanced multispectral datasets. Consequently, BAGAN-inspired architectures are proposed as a new baseline for researchers working with GANs in remote sensing. Another crucial finding of this study is that integrating additional techniques can further improve the quality of the GAN-based data augmentation and the final classification accuracy, as demonstrated by the proposed method ResBaGAN, which improves upon BAGAN.

ResBaGAN's adaptability presents a significant advantage for the future, as it can be easily integrated with classifiers from other works to increase their accuracy. To achieve this, the classifier only needs to be integrated as the D network within ResBaGAN, with adjustments made to the design of the G network to match the capabilities of both modules, as illustrated throughout this work.

The primary limitation of ResBaGAN is its long training time compared to other deep learning approaches. However, it presents a good tradeoff between cost and performance. As shown in the experiments in this work, none of the other classification methods achieved 80% for the F1 score across all classes. It is also worth noting that the classification cost of ResBaGAN mainly depends on the computational cost of the chosen classifier, so it could be reduced if the classifier is replaced by one with lower complexity. One additional advantage of ResBaGAN is that applying the superpixel segmentation to the image greatly reduces the computational cost of classification compared to a pixel-focused approach. This features makes ResBaGAN well-suited for rapidly mapping extensive terrain areas in real applications involving remote sensing classification.

Another significant contribution of this work is the successful adaptation of the FID score metric to assess the quality of GAN-based synthesis while using the full spectral resolution of remote sensing data. This modification is necessary as there are currently no standard models for evaluating FID on remote sensing data, similar to the Inception v3 network used for the RGB data. This adaption allows for more accurate comparisons among GAN-based networks and can be easily included in other research works. Nonetheless, it would be valuable to explore the development of standard models for evaluating the FID score on remote sensing data.

Various aspects of ResBaGAN open up future work directions to improve its performance. For instance, the BAGAN-based designs in this work do not feature the sophisticated latent vector sampling strategy, which could further ease working with limited data. Another possibility for improvement could involve integrating state-of-the-art deep learning architectures like GCN [18] and transformers [19] as classifiers in ResBaGAN. Moreover, other traditional augmentation techniques not considered in this analysis, such as those described in [25] or [26], could be integrated into the proposed solution.

Regarding the field of application, this study focused on applying ResBaGAN to mapping images of forest areas. It would also be interesting to evaluate ResBaGAN on new datasets to further explore its strengths and limitations, whether in vegetation terrains or in other types of images. Moreover, to evaluate the robustness of ResBaGAN, it would be valuable to analyze its performance using datasets captured under variable illumination and atmospheric conditions, or including noise produced by the sensor. As ResBaGAN extracts patches centered on superpixels and works with the whole spectral dimensionality of the image, it is expected that it could handle the intraclass spectral variability produced by these varying conditions to some extent.

The solution proposed in this work operates over very high-resolution images. However, ResBaGAN could be adapted to operate over remote sensing images with different spectral and spatial resolutions by adapting its stages. For instance, modifying patch and superpixel size parameters would be necessary for dealing with datasets with different spatial resolutions. For coarser grained datasets, such as satellite images, employing spectral unmixing techniques [76] might be necessary for situations where multiple, different elements are captured within the same pixel.

VI. Conclusion

This work proposes ResBaGAN, a deep-learning method for the classification of remote sensing images, designed to overcome the prevalent challenges of data scarcity and class imbalances. This is achieved by constructing an advanced data augmentation framework that combines BAGAN-inspired augmentation, sample extraction on top of traditional augmentation techniques and superpixel segmentation, and an improved ResNet-based classifier. This integrated approach maximizes the usage of spectral and spatial information from the available training data to effectively overcome the aforementioned issues. Moreover, the superpixel segmentation also makes ResBaGAN suitable for recognizing large terrain areas. To the best of the authors' knowledge, this is the first study that applies a BAGAN-based architecture to remote sensing classification. In addition, this work also proposes an adaptation of the FID score metric for evaluating the synthesis quality of GANs in multi- and hyperspectral remote sensing images.

ResBaGAN's performance was evaluated in the context of environmental monitoring by using eight large, very high-resolution multispectral images of riparian forests, with limited learning data and strong class imbalances. A comparison to other state-of-the-art classification methods in the remote sensing field, such as ACGAN, was carried out. The results revealed the effectiveness of ResBaGAN in achieving high classification accuracies in constrained datasets with improved performance across all classes of elements, particularly in minority ones.

Finally, numerous interesting research directions have been identified for future work. For instance, it would be valuable to assess the performance of ResBaGAN on new datasets featuring diverse spatial and spectral resolutions. Additionally, exploring the integration of more advanced components into ResBaGAN could yield further improvements, such as substituting the traditional augmentation techniques with more sophisticated methods or replacing the residual classifier with cutting-edge architectures like transformers.

References

[1] Z. Shao, P. Tang, Z. Wang, N. Saleem, S. Yam, and C. Sommai, "BRRNet: A fully convolutional neural network for automatic building extraction from high-resolution remote sensing images," *Remote Sens.*, vol. 12, no. 6, 2020, Art. no. 1050.

[2] B. Lu, P. D. Dao, J. Liu, Y. He, and J. Shang, "Recent advances of hyperspectral imaging technology and applications in agriculture," *Remote Sens.*, vol. 12, no. 16, 2020, Art. no. 2659.

[3] F. Argüello, D. B. Heras, A. S. Garea, and P. Quesada-Barriuso, "Watershed monitoring in Galicia from UAV multispectral imagery using advanced texture methods," *Remote Sens.*, vol. 13, no. 14, 2021, Art. no. 2687.

[4] A. E. Maxwell, T. A. Warner, and F. Fang, "Implementation of machine-learning classification in remote sensing: An applied review," *Int. J. Remote Sens.*, vol. 39, no. 9, pp. 2784–2817, 2018.

[5] B. A. Bradley, "Remote detection of invasive plants: A review of spectral, textural and phenological approaches," *Biol. Invasions*, vol. 16, pp. 1411–1425, Jul. 2014.

[6] N. Y. Zhu et al., "Deep learning for smart agriculture: Concepts, tools, applications, and opportunities," *Int. J. Agricultural Biol. Eng.*, vol. 11, no. 4, pp. 32–44, 2018.

[7] Q. Yuan et al., "Deep learning in environmental remote sensing: Achievements and challenges," *Remote Sens. Environ.*, vol. 241, 2020, Art. no. 111716.

[8] G. Morales, G. Kemper, G. Sevillano, D. Arteaga, I. Ortega, and J. Telles, "Automatic segmentation of mauritia flexuosa in unmanned aerial vehicle (UAV) imagery using deep learning," *Forests*, vol. 9, no. 12, 2018, Art. no. 736.

[9] Z. M. Hamdi, M. Brandmeier, and C. Straub, "Forest damage assessment using deep learning on high resolution remote sensing data," *Remote Sens.*, vol. 11, no. 17, 2019, Art. no. 1976.

[10] K. Isaienkov, M. Yushchuk, V. Khramtsov, and O. Seliverstov, "Deep learning for regular change detection in Ukrainian forest ecosystem with Sentinel-2," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 364–376, 2021.

[11] C.-Y. Chiang, C. Barnes, P. Angelov, and R. Jiang, "Deep learning-based automated forest health diagnosis from aerial images," *IEEE Access*, vol. 8, pp. 144064–144076, 2020.

[12] 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.

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

[14] 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.

[15] 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.

[16] 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.

[17] R. Li, S. Zheng, C. Duan, Y. Yang, and X. Wang, "Classification of hyperspectral image based on double-branch dual-attention mechanism network," *Remote Sens.*, vol. 12, no. 3, 2020, Art. no. 582.

[18] D. Hong, L. Gao, J. Yao, B. Zhang, A. Plaza, and J. Chanussot, "Graph convolutional networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 7, pp. 5966–5978, Jul. 2021.

[19] Q. He, X. Sun, W. Diao, Z. Yan, D. Yin, and K. Fu, "Transformer-induced graph reasoning for multimodal semantic segmentation in remote sensing," *ISPRS J. Photogrammetry Remote Sens.*, vol. 193, pp. 90–103, 2022.

[20] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[21] D. Hong et al., "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 5, pp. 4340–4354, May 2021.

[22] X. Wu, D. Hong, and J. Chanussot, "Convolutional neural networks for multimodal remote sensing data classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5517010.

[23] X. Sun et al., "Ringmo: A remote sensing foundation model with masked image modeling," *IEEE Trans. Geosci. Remote Sens.*, 2022, to be published, doi: 10.1109/TGRS.2022.3194732.

[24] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, Jul. 2019, Art. no. 60.

[25] J. M. Haut, M. E. Paoletti, J. Plaza, A. Plaza, and J. Li, "Hyperspectral image classification using random occlusion data augmentation," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 11, pp. 1751–1755, Nov. 2019.

[26] A. Acción, F. Argüello, and D. B. Heras, "Dual-window superpixel data augmentation for hyperspectral image classification," *Appl. Sci.*, vol. 10, no. 24, 2020, Art. no. 8833.

[27] J. Nalepa, M. Myller, and M. Kawulok, "Training- and test-time data augmentation for hyperspectral image segmentation," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 2, pp. 292–296, Feb. 2020.

[28] I. J. Goodfellow et al., "Generative adversarial networks," *CoRR*, vol. abs/1406.2661, 2014. [Online]. Available: http://arxiv.org/abs/1406.2661

[29] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification," *Neurocomputing*, vol. 321, pp. 321–331, 2018.

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

[31] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. 2nd Int. Conf. Learn. Representations*, Y. Bengio and Y. LeCun, Eds. Banff, AB, Canada, Apr. 14-16, 2014. [Online]. Available: http://arxiv.org/abs/1312.6114

[32] A. Shashank, V. V. Sajithvariyar, V. Sowmya, K. P. Soman, R. Sivanpillai, and G. K. Brown, "Identifying epiphytes in drones photos with a conditional generative adversarial network (C-GAN)," *Int. Arch. Photogrammetry, Remote Sens. Spatial Inf. Sci.*, vol. 99, pp. 99–104, 2020.

[33] 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.

[34] M. Lucic, M. Tschannen, M. Ritter, X. Zhai, O. Bachem, and S. Gelly, "High-fidelity image generation with fewer labels," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4183–4192.

[35] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi, "BAGAN: Data augmentation with Balancing GAN," 2018, *arXiv:1803.09655*.

[36] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6629–6640.

[37] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65, Jan. 2018.

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

[39] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2642–2651.
[40] Y. Zhang, X. Liu, S. Wa, S. Chen, and Q. Ma, "GANsformer: A detection network for aerial images with high performance combining convolutional network and transformer," *Remote Sens.*, vol. 14, no. 4, 2022, Art. no. 923.
[41] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE J.*, vol. 37, no. 2, pp. 233–243, 1991.
[42] D. Stutz, A. Hermans, and B. Leibe, "Superpixels: An evaluation of the state-of-the-art," *Comput. Vis. Image Understanding*, vol. 166, pp. 1–27, 2018.
[43] P. G. Bascoy, A. S. Garea, D. B. Heras, F. Argüello, and A. Ordóñez, "Texture-based analysis of hydrographical basins with multispectral imagery," in *Proc. SPIE*, 2019, vol. 11149, pp. 225–234.
[44] S. R. Blanco, D. B. Heras, and F. Argüello, "Texture extraction techniques for the classification of vegetation species in hyperspectral imagery: Bag of words approach based on superpixels," *Remote Sens.*, vol. 12, no. 16, 2020, Art. no. 2633.
[45] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 22–40, Jun. 2016.
[46] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.
[47] J. Yao, M. Boben, S. Fidler, and R. Urtasun, "Real-time coarse-to-fine topologically preserving segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2947–2955.
[48] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 13–26.
[49] V. Machairas, M. Faessel, D. Cárdenas-Peña, T. Chabardes, T. Walter, and E. Decencière, "Waterpixels," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3707–3716, Nov. 2015.
[50] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.
[51] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," 2018, *arXiv:1802.05957*.
[52] D. Jurafsky and J. H. Martin, *Speech and Language Processing–An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 2nd Edition*, Prentice Hall, Pearson Education International, 2009. [Online]. Available: https://www.worldcat.org/oclc/315913020
[53] R. G. Congalton, "A review of assessing the accuracy of classifications of remotely sensed data," *Remote Sens. Environ.*, vol. 37, no. 1, pp. 35–46, 1991.
[54] Y. Xu, W. Yu, P. Ghamisi, M. Kopp, and S. Hochreiter, "Txt2Img–MHN: Remote sensing image generation from text using modern Hopfield networks," 2022, *arXiv:2208.04441*.
[55] P. Shamsolmoali, M. Zareapoor, H. Zhou, R. Wang, and J. Yang, "Road segmentation for remote sensing images using adversarial spatial pyramid networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 6, pp. 4673–4688, Jun. 2021.
[56] AnandTech, "Intel Core i7–11700 K review: Blasting off with Rocket Lake," Accessed on: Jun. 11, 2022. [Online]. Available: https://www.anandtech.com/show/16535/intel-core-i7-11700k-review-blasting-off-with-rocket-lake
[57] TechPowerUp, "NVIDIA GeForce RTX 3080 Ti specs—TechPowerUp GPU database," Accessed on: Jun. 11, 2022. [Online]. Available: https://www.techpowerup.com/gpu-specs/geforce-rtx-3080-ti.c3735
[58] Canonical Ltd., "Enterprise open source and Linux—ubuntu," Accessed: Jun. 11, 2022. [Online]. Available: https://ubuntu.com/
[59] Python software foundation, "Welcome to Python.org," Accessed on: Jun. 11, 2022. [Online]. Available: https://www.python.org/
[60] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, pp. 8024–8035.
[61] NVIDIA Corporation, "CUDA zone–library of resources—NVIDIA developer," Accessed on: Jun. 11, 2022. [Online]. Available: https://developer.nvidia.com/cuda-zone
[62] S. Chetlur et al., "cuDNN: Efficient primitives for deep learning," *CoRR*, vol. abs/1410.0759, 2014. [Online]. Available: http://arxiv.org/abs/1410.0759
[63] JTC1/SC22/WG14, "ISO/IEC 9899," Standard, ISO/IEC, 2018. Accessed: Jun. 11, 2022. [Online]. Available: https://www.open-std.org/jtc1/sc22/wg14/
[64] Standard C++ Foundation, "Standard C++," Accessed on: Jun. 11, 2022. [Online]. Available: https://isocpp.org/
[65] Free Software Foundation, Inc., "GCC, the GNU compiler collection–GNU project," Accessed on: Jun. 11, 2022. [Online]. Available: https://gcc.gnu.org/
[66] TensorHub, Inc., "GuildAI–Experiment tracking, ML developer tools," Accessed: Jun. 13, 2022. [Online]. Available: https://guild.ai/
[67] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," 2015, *arXiv:1511.07289*.
[68] A. L. Maas, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. 30th Int. Conf. Mach. Learn.*, vol. 28, no. 1, p.3, 2013.
[69] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1026–1034.
[70] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 807–814.
[71] D. P. Kingma and J. Ba, "ADAM: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, San Diego, CA, USA, May 7-9, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980
[72] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*.
[73] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, vol. 9, pp. 249–256.
[74] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *Int. J. Comput. Vis.*, vol. 105, pp. 222–245, Dec. 2013.
[75] G.-B. Huang, "An insight into extreme learning machines: Random neurons, random features and kernels," *Cogn. Comput.*, vol. 6, pp. 376–390, Sep. 2014.
[76] D. Hong, N. Yokoya, J. Chanussot, and X. X. Zhu, "An augmented linear mixing model to address spectral variability for hyperspectral unmixing," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1923–1938, Apr. 2019.

Álvaro G. Dieste received the B.S. degree in computer engineering and the M.S. degree in high performance computing in 2021 and 2022, respectively, from the University of Santiago de Compostela, Santiago, Spain, where he is currently working toward the Ph.D. degree in computer science.

He is currently an Assistant Researcher with the Centro Singular de Investigación en Tecnoloxías Intelixentes, University of Santiago de Compostela. His research interests focus on computer vision tasks, while bringing together knowledge from diverse computing areas, such as artificial intelligence, high performance computing, and big data.

Francisco Argüello received the B.S. and Ph.D. degrees in physics from the University of Santiago de Compostela, Santiago, Spain, in 1988 and 1992, respectively.

He is currently an Associate Professor with the Department of Electronics and Computer Engineering, University of Santiago de Compostela. His research interests include signal and image processing, computer graphics, parallel and distributed computing, and quantum computing.

Dora B. Heras (Member, IEEE) received the M.S. and Ph.D. degrees in physics from the University of Santiago de Compostela, Santiago, Spain, in 1994 and 2000, respectively.

She is currently an Associate Professor with the Department of Electronics and Computer Engineering, University of Santiago de Compostela. Her research interests cover a range of topics in the combined fields of image processing, remote sensing, machine learning, and high performance computing. In particular, she has published papers on high performance computing, registration, classification, domain adaptation, and change detection applied to remotely sensed images.