remote sensing

MDPI

Article

A New Multispectral Data Augmentation Technique Based on Data Imputation

Álvaro Acción [1,*], Francisco Argüello [2] and Dora B. Heras [1]

[1] Centro Singular de Investigación en Tecnologías Inteligentes, Universidade de Santiago de Compostela, 15782 Santiago de Compostela, Spain; dora.blanco@usc.es
[2] Departamento de Electrónica y Computación, Universidade de Santiago de Compostela, 15782 Santiago de Compostela, Spain; francisco.arguello@usc.es
* Correspondence: alvaro.accion.montes@usc.es

Abstract: Deep Learning (DL) has been recently introduced into the hyperspectral and multispectral image classification landscape. Despite the success of DL in the remote sensing field, DL models are computationally intensive due to the large number of parameters they need to learn. The high density of information present in remote sensing imagery with high spectral resolution can make the application of DL models to large scenes challenging. Methods such as patch-based classification require large amounts of data to be processed during the training and prediction stages, which translates into long processing times and high energy consumption. One of the solutions to decrease the computational cost of these models is to perform segment-based classification. Segment-based classification schemes can significantly decrease training and prediction times, and also offer advantages over simply reducing the size of the training datasets by randomly sampling training data. The lack of a large enough number of samples can, however, pose an additional challenge, causing these models to not generalize properly. Data augmentation methods are used to generate new synthetic samples based on existing data to increase the classification performance. In this work, we propose a new data augmentation scheme using data imputation and matrix completion methods for segment-based classification. The proposal has been validated using two high-resolution multispectral datasets from the literature. The results obtained show that the proposed approach successfully increases the classification performance across all the scenes tested and that data imputation methods applied to multispectral imagery are a valid means to perform data augmentation. A comparison of classification accuracy between different imputation methods applied to the proposed scheme was also carried out.

Keywords: multispectral; classification; deep learning; CNN; segmentation; data augmentation; imputation

Citation: Acción, Á.; Argüello, F.; Heras, D.B. A New Multispectral Data Augmentation Technique Based on Data Imputation. *Remote Sens.* **2021**, *13*, 4875. https://doi.org/10.3390/rs13234875

Academic Editors: Benoit Vozel, Vladimir Lukin and Yakoub Bazi

Received: 23 October 2021
Accepted: 26 November 2021
Published: 30 November 2021

1. Introduction

Advances in hyperspectral and multispectral image capture technology have improved the data acquisition capabilities of sensors mounted in satellites and, more recently, in unmanned aerial vehicles (UAVs). The greater availability of the technology and continuous increases in scene resolution [1], as well as the commoditization of data processing power and data storage, have made these images more common as a source of information to extract knowledge from. Hyperspectral and multispectral image processing [2] have been successfully used to perform a variety of semi-automated tasks as a way to increase the efficiency of productive processes in diverse fields [3–7].

Multispectral images (MSIs) consist of a vector of pixels that represent the intensity for a point in the image at a given wavelength. The number of components of the vector varies depending on the sensor, and normally ranges from three to ten. This is also referred to as spectral resolution and is given by the minimum distance between two wavelengths required to distinguish two different lines of the spectrum [8]. Sensor characteristics also

determine what regions of the electromagnetic spectrum can be effectively observed, often given as a continuous range. The rich amount of spectral information present in these images allows for improved separability of the materials present in a scene [9].

Several methods and classifiers have been developed or adapted for hyperspectral and multispectral image classification. The application of Deep Learning (DL) to this task has shown successful results in the literature [10–17]. Convolutional neural networks (CNNs) [18] in particular are commonly used to solve multi-class classification problems involving feature extraction from high dimensionality images. 2D-CNNs operate on small cubes of data that are centered around a pixel, called patches, combining both spatial and spectral information. Some of the challenges of the application of DL models to hyperspectral image classification are related to the complex training processes, which result in a high computational cost to process the large amount of data HSIs contain, and also to the limited training data available [19]. Data augmentation [20] is a method that attempts to increase the generalization capabilities of the resulting models by generating new observations to be used during training and prediction. Data augmentation strategies are commonly used in the image classification [21] or object recognition [22] fields and have been successfully applied to hyperspectral image classification in combination with DL models [23–28].

In this paper, a new approach to data augmentation based on the application of data imputation methods for the generation of synthetic samples during the training process is proposed. Data imputation [29], defined as the substitution of missing data by estimated values, is one of the most common methods used to mitigate the deficiencies caused by missing or low-quality data. Missing data [30] is a recurring problem in real-life datasets due to a variety of circumstances such as sensor limitations, system malfunction, inadequate acquisition conditions, sampling bias, etc. Incomplete datasets can cause undesirable problems during the analysis of the data, such as producing biased estimates, lowering the statistical power [31], or leading to incorrect inferences [32]. Multiple imputation [33], where each missing value estimate is generated multiple times and then combined, can restore the variability of the original dataset and account for the uncertainty in the missing data, yielding valid inferences. MICE [34] is possibly the most widely known multiple imputation method. Matrix completion [35] algorithms, which attempt to predict missing entries from a partially observed matrix by learning a low-rank model, are also capable of performing the imputation of highly dimensional numeric data. Matrix completion has garnered increased interest from researchers over the past decade due to its potential application to recommender systems, computer vision, and so forth. Algorithms, such as SoftImpute [36], belong to this category.

The proposed data augmentation approach makes use of superpixel segmentation as a preprocessing step in order to determine regions to impute in the input patches. Segmentation [37] is a preprocessing method that involves partitioning an image into similar regions or subgroups of pixels, which are also called segments, based on some common characteristics. One of the main purposes of this method is the simplification of images into more meaningful components, decreasing the complexity of subsequent analysis steps. Segmentation has been used in hyperspectral image classification in combination with traditional models in [38–40], and as part of DL classification schemes in [41,42].

In this work, we propose a novel approach to data augmentation for MSIs in which superpixel segmentation [43] is used both as a means of reducing the computational cost of the model, and as part of the data augmentation stage. In contrast with previous works [38–42], where segmentation is used exclusively to simplify the image or in a post-processing stage to take advantage of its spatial regularization properties, our work also uses this information to generate new samples to train a 2D-CNN. An initial set of samples corresponds to the patches extracted from the segment centers. This set is augmented with a second synthetic set created using the resulting segmentation to determine regions to erase inside the extracted patches. Data imputation is then used at the patch level in order

to generate new pixel vectors inside the erased regions. Lastly, both sets are fed to the CNN model for training. The main contributions of this paper are:

1. A new data augmentation scheme based on leveraging the properties of segment-based classification in combination with data imputation methods is proposed. This scheme increases classification performance and reduces the computational complexity of the resulting CNN model. To the best of our knowledge, this is the first time data imputation algorithms are used to perform data augmentation for either hyperspectral or multispectral images;
2. A study of the classification performance of the proposed scheme in combination with several imputation methods over high-resolution multispectral images is carried out.

The remainder of this paper is organized into the following sections. Section 2 explains and describes the different stages of the proposed classification scheme. Section 3 presents an explanation of how the imputation process works, and a description of the different imputation methods that were considered for the scheme. Section 4.3 contains the experimental results for the evaluation of the classification performance of the proposal on a set of high-resolution multispectral datasets. Section 5 contains the discussion about the features of the proposal. Finally, Section 6 summarizes the main conclusions drawn from this work.

2. Classification Scheme

This section describes in detail the data augmentation approach developed as part of this work, as well as the associated classification scheme.

The proposal, which can be seen in Figure 1, comprises of four main stages: segmentation, where a segmentation algorithm is applied to the image to obtain a segmentation map; patch extraction, consisting of the extraction of the samples to be used for the classification process; patch erasure, that discards the portions of the patches considered less relevant, and patch imputation, where the missing parts of the patch are filled with new values using an imputation algorithm. Out of the four stages, the first stage, segmentation, is performed offline; the results are stored and loaded directly during the next stages. The segmentation map is also reused during the prediction phase. Stages two to four are performed online, and only during the training phase. Data augmentation is performed in stages three and four. Algorithm 1 displays the pseudocode for the data augmentation process.

A detailed explanation for each of the stages can be found below.

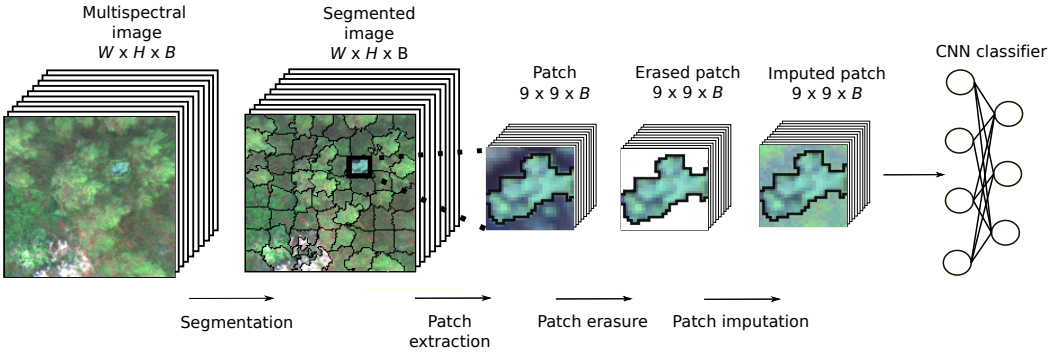


Figure 1. Proposed classification scheme.

Algorithm 1 Pseudocode for the patch augmentation process.

```
for each segment in segmentation do
    extract patch centered in segment
    mask = segment ∩ patch
    for each pixel in patch do
        if pixel inside mask then
            copy pixel from dataset into patch
        else
            mark pixel in patch as missing
        end if
    end for

  // Imputation
    reshape patch as 2D table
    impute table
    reshape table as 3D patch
end for
```

2.1. Segmentation

Superpixel segmentation has been used frequently in the literature for hyperspectral classification tasks. In [38], superpixel segmentation is used as a postprocessing step in order to refine the classification results obtained by a support-vector machine (SVM) classifier. Superpixels are also used in joint spectral–spatial classification proposals [44]. Formally, segmentation can be defined as follows [45]. Given a set of pixels belonging to a region $\mathcal{R}$, if $\mathrm{P}(\mathrm{o})$ is a homogeneity predicate defined on groups of connected pixels, then the segmentation corresponds to a partition of the set $\mathcal{R}$ into connected components or subsets $\{\mathcal{S}_1, \ldots, \mathcal{S}_n\}$ such that:

$$\bigcup_{i=1}^{n} \mathcal{S}_i = \mathcal{R} \text{ with } \mathcal{S}_i \cap \mathcal{S}_j = \emptyset, \forall i \neq j. \tag{1}$$

We have that $\mathrm{P}(\mathcal{S}_i) = true, \forall i = 0, \ldots, n$ and $\mathrm{P}(\mathcal{S}_i \cup \mathcal{S}_j) = false$ when $i \neq j$ and subsets $\mathcal{S}_i$ and $\mathcal{S}_j$ are neighbors.

The proposed superpixel-based classification scheme benefits from the segmentation information in different ways. The segmentation stage begins with the application of a segmentation algorithm to the original MSI to obtain a segmentation map. The main advantage associated with the use of a segmentation algorithm before further processing of the data is the reduction of the computational cost during the patch selection stage described in Section 2.2, where only a subset of the pixel vectors in each segment is selected for the training and prediction stages.

The use of a superpixel-based scheme also impacts the prediction stage, where all the members of a segment are assigned the same class label. This proposal makes extensive use of the segment map to remove regions of the patches and later generate new pixel vectors through the application of data augmentation to create new samples, as described in Sections 2.3 and 2.4.

2.2. Patch Extraction

A multispectral image can be defined as a 3D matrix of dimensions $W \times H \times B$, where W represents the width, H represents the height, and B represents the number of spectral bands of the image. In traditional pixel-based classification, a common approach is to divide the MSI into 3D blocks of data, or patches, representing portions of the image. Let us denote $P_{x,y}$ as the patch centered around a pixel vector $\boldsymbol{p}_{x,y} \mid \boldsymbol{p} \in R^n$, extracted using a sliding window of dimensions $s \times s$, where s is the size of the window. The window size s determines the size of the spatial neighborhood extracted, with larger sizes leading to a larger amount of information the CNN can exploit. However, it also increases the

computational cost associated with the processing of the data. Generally, larger sizes yield better classification performance at the cost of higher resource consumption. Let us denote the class label for a pixel $\boldsymbol{p}_{x,y}$ as $l_{x,y} \mid l \in L$, where L is the set class labels of size c. These blocks of data containing a mixture of spectral and spatial information, along with the corresponding class label, constitute the inputs for a 2D-CNN.

In this superpixel-based segmentation scheme, once the segments have been generated during stage one of Figure 1, the next step is to extract patches from them. There is no restriction in the number of patches that can be extracted other than the number of pixels inside a given segment.

In the proposed scheme, a single patch is extracted per segment during the training phase. This is shown in Figure 2. This choice implies that the size of the dataset will depend, thus, on the parameters used for the application of the segmentation algorithm. The computational complexity of the backward and forward passes of the network is directly correlated to the number of samples being fed to it, which this method reduces by a similar factor as the chosen segment size used in the segmentation algorithm.

The central pixel of each of the segments is chosen as the center of a patch, $P_{x,y}$. This pixel is determined by obtaining the center of the bounding box containing the segment of interest (SOI). The patch is extracted by defining a window of dimensions $s \times s$ around it. The resulting patch is assigned the class label $l_{x,y}$, which corresponds to the label of its central pixel. The left side of Figure 3 displays the resulting segment map for a crop of the River Oitaven scene, and, on the right, the resulting reference data generated from the segmentation after the labels were assigned to the segments. More information about the specific window dimensions used can be found at the end of Section 4.2.

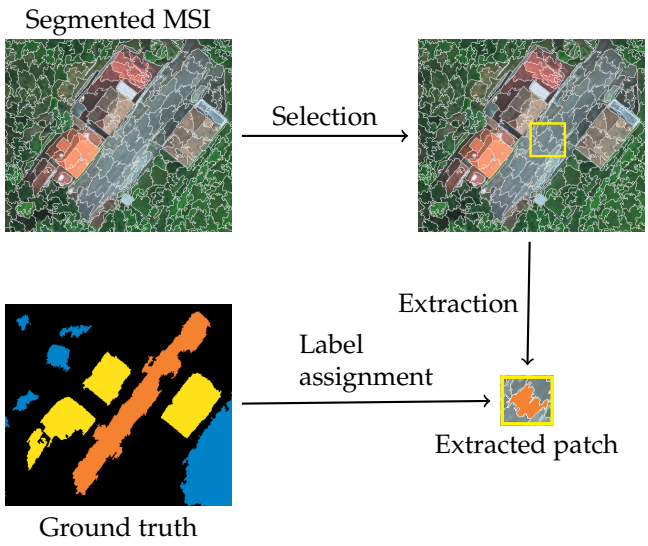


Figure 2. Stage 2: Patch extraction.

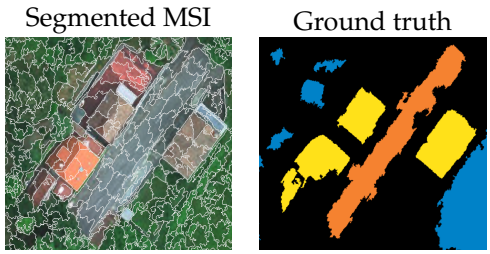


Figure 3. Cropped section of the River Oitaven scene after superpixel segmentation along with the corresponding reference data.

2.3. Patch Erasure

Although the advantages of segment-based schemes are significant, they present some drawbacks. The greatly reduced number of resulting patches can negatively affect classification performance due to a lack of training data. Patches belonging to the already-diminished training dataset may also exhibit some undesirable properties, such as containing information from several different segments.

By reducing the heterogeneity of this information and replacing part of it with information belonging to the SOI the patch was centered on, a new sample can be generated. In this stage, the pixel vectors inside the patch that do not belong to the SOI are erased as a prior step to the generation of that new sample.

The patch erasure process is depicted in Figure 4. The upper left figure represents the different segments present in a patch. The original segment map is used to obtain a list of coordinates belonging to the segment of interest. These coordinates are transformed into the patch space and are then clipped to form what we call the patch mask, which corresponds to the pixel vectors that will be preserved, shown in white in the figure. The remaining pixel vectors are erased from the patch. The result of this process is a cleaned patch that should contain mostly homogeneous information.

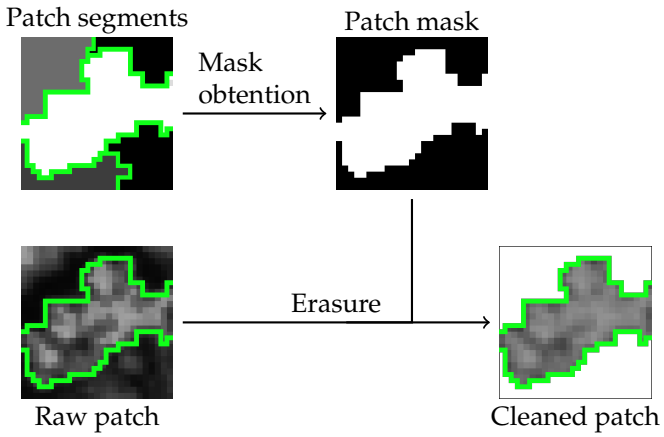


Figure 4. Stage 3: Patch erasure.

2.4. Patch Imputation

Once the cleaned patch is obtained, the missing values are replaced by numeric values, which are generated using imputation algorithms in order to serve as inputs of the CNN.

This patch imputation process is depicted in Figure 5. In order to apply data imputation, the patch has to be transformed into a suitable representation. This is achieved by flattening the 3D array representing the patch into a 2D matrix. Each of the rows of the matrix will correspond to a pixel vector in the image, with each component or band stored in a column. This matrix is augmented with information about the row and column coordinates for each pixel. The columns x, y represent the coordinates of the pixel in the patch. The remaining columns $b1$, and $b2$ would be the values of the pixel for the different bands of the HSI. Erased data are denoted by NaN values in the matrix. All the bands are present in the actual matrix used by the algorithm, that is, the imputation is performed in all bands simultaneously. For the application of matrix completion algorithms, a copy of the bands containing the original data is also included in the matrix as columns. Finally, the imputation method of choice is applied to the matrix. The results are reshaped into a 3D array once again and are sent to the CNN to be processed. More information about each of the imputation methods used in the scheme and their parameters can be found in Sections 3 and 4.1.

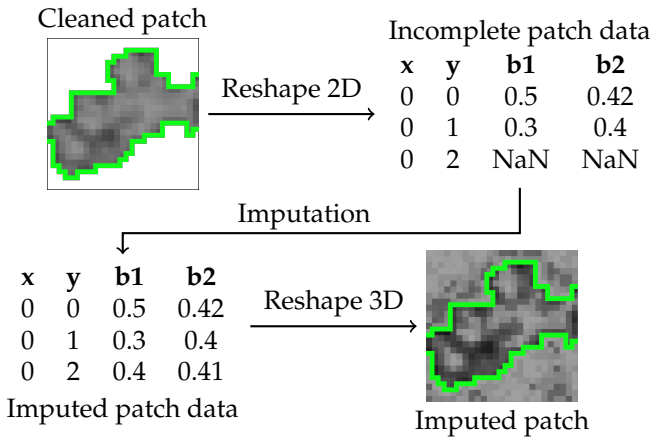


Figure 5. Stage 4: Patch imputation.

3. Data Imputation Methods

Five imputation methods were considered in this work: Constant, KNNimpute [46], MICE [34], SoftImpute [36] and SVDimpute [46]. These methods were selected due to their popularity in the literature [47–55] and moderate computational cost.

Multispectral data are usually represented as a 3D cube of information. This representation, however intuitive, is not suitable for the application of data imputation methods. Imputation algorithms often require the data to be in tabular form, in a representation where each row corresponds to an observation and each column corresponds to one of the different variables being observed. To apply this representation to our data cubes, each of the pixels will correspond to an observation in the dataset, with each of the columns representing the intensity value for the corresponding band.

3.1. Constant Imputation

Constant imputation is a method where the missing values are replaced by a constant. In our case, this value is applied to the pixels in the patch that fall outside the segment. This method has the drawback of introducing bias in the data since the correlations between the different variables are not preserved. The computational complexity of the constant imputation method is $O(mn)$.

3.2. KNNimpute

KNNimpute [46] is based on the estimation of the missing values of an observation from a subset of its k nearest observations. In order to achieve this, the Euclidean distance between all the pixel vector pairs in the patch is computed and then ranked. The missing values are replaced by the weighted mean of the subset of k nearest observations based on their distance to the observation to be imputed. KNNimpute is a non-parametric imputation method and supports continuous, discrete, ordinal, and categorical data. The algorithm does not require a model to be built as it uses the raw elements in the dataset to perform the imputation (instance-based learning). Some of the known shortcomings of k-NN-based imputation methods include the sensitivity to noise, the high computational cost when the number of elements in the dataset is high, and its low performance for datasets with a high number of dimensions. KNNimpute has a computational complexity of approximately $O(m^2 n)$ for a matrix of m columns and n assuming $m \gg k$.

3.3. MICE

MICE [34], or multivariate imputation by chained equations, is one of the most popular methods in the literature to perform data imputation. MICE has high flexibility and supports many different kinds of data. The multiple imputations produced by this method take into account the imputation uncertainty and present accurate standard errors. MICE performs what is called a fully conditional specification (FCS), requiring an imputation

model for each missing variable. Each missing observation in the dataset is modeled against all the other observations. This is done for all the missing observations, one at a time, for a certain number of iterations. Despite its advantages, the theoretical basis of FCS is a known weakness since fitting several conditional distributions may not be consistent with a joint model [56].

MICE performs an initialization step, replacing all the missing values with placeholders. For each iteration, the missing values in the observation are replaced by the values obtained from the chosen model. Several models are viable depending on the type of data and the characteristics of the dataset. After all the missing values have been updated, a cycle is complete, and newly obtained values replace the initial ones. The application of several cycles until convergence is reached represents an imputation. Each imputation will generate a new complete dataset. Lastly, the imputations are pooled together to obtain the final dataset. The time complexity of MICE can be defined as [57] $O(T \cdot n^3 \cdot (m + n^{0.807355}))$, with T being the number of iterations performed by the algorithm.

3.4. SoftImpute

SoftImpute [36] finds a low-rank matrix approximation to a matrix with missing values via nuclear-norm regularization. It iteratively replaces the missing elements with those obtained from a soft-thresholded singular value decomposition (SVD). The algorithm works like expectation–maximization (EM), filling in the missing values with the current guess, and then solving the optimization problem on the complete matrix using a soft-thresholded SVD. The algorithm scales well to very large matrices with order linear, achieving low training and test errors. The computational complexity of the method can be defined as follows. For a matrix $X_{m \times n}$ let $\Omega \subset \{1, \ldots, m\} \times \{1, \ldots, n\}$ denote the indices of observed entries. The total cost to compute the regularization path on a grid of λ values is given by:

$$\sum_{i=1}^{K} O\left(\left(|\Omega| \bar{r}_{\lambda_i} + (m+n) \bar{r}_{\lambda_i}^2 \right) \frac{2}{\delta} \left\| \hat{Z}_{\lambda_{i-1}} - Z_{\lambda_i}^{\infty} \right\|_F^2 \right), \tag{2}$$

where $\lambda \geq 0$ is a regularization parameter controlling the nuclear norm of the minimizer $\hat{Z}_\lambda$, $\bar{r}_\lambda$ denotes the rank (on an average) of the iterates Z_λ^k generated for a fixed λ, and $\|\cdot\|_F$ denotes the Frobenius norm.

3.5. SVDimpute

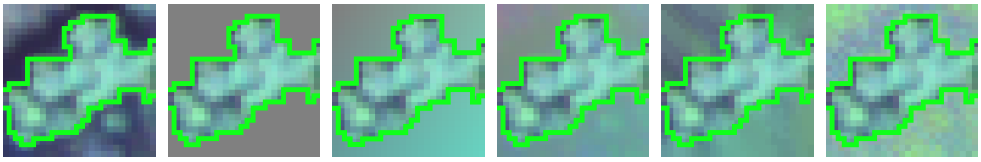
Originally introduced in [46], SVDimpute works by iteratively applying SVD decomposition to the matrix representing all the observations in the dataset. The resulting eigenvectors are then used to obtain the missing values for each of the pixels. The eigenvectors are first sorted by their corresponding eigenvalues. For each pixel, the imputation is performed band by band. The value of the pixel on the band is regressed against the k most relevant eigenvalues to obtain the coefficients. The missing value is replaced by a linear combination of the coefficients. The newly obtained values replace those at the beginning of the iteration.

The process starts by replacing all missing values by zero and repeatedly applying the previous process until the difference between the newly obtained matrix and the previous one falls below a threshold.

SVDimpute is known to be able to handle large amounts of missing data and can be applied to very large matrices. A known drawback of this method is the fact that multiple possible solutions to the imputation problem exist depending on the values used for the initialization, which may make it unsuitable for tasks that require a deterministic output. The computational complexity of SVDimpute for a matrix of m columns and n is defined as $O(n^2 mi)$, where i is the number of iterations performed before the threshold value is reached.

Figure 6 shows an example patch after being imputed with all of the methods described in this section. The different bands were then scaled to an 8-bit unsigned integer data type for their visual representation. The patches from the imputation methods are

sorted based on the number of structures they generate in the image, in increasing order. Constant imputation has the highest impact on the data of the synthetic patch as it strips away all the existing information in the imputed pixels. SoftImpute creates gradients around the image that alternate very progressively between dark and light areas. It can be seen that SVDimpute produces the closest result to the original data, with the same pixel patterns but different grayscale values for the generated information. KNNimpute replicates the outermost pixel of the patch continuously based on the distance. MICE displays the highest variability for the external pixels, generating the highest number of structures.



(**a**) (**b**) (**c**) (**d**) (**e**) (**f**)

Figure 6. Example of augmented patches for each of the studied imputation methods, for a segment of the River Mestas scene with a segment size of 800, and a patch size of 25 × 25. (**a**) None, (**b**) Constant, (**c**) SoftImpute, (**d**) SVDimpute, (**e**) KNNimpute, (**f**) MICE.

4. Results

This section contains the description of the experimental conditions and experimental results obtained to evaluate the data augmentation approach presented in this work.

4.1. Experimental Conditions

All the experiments follow the network architecture and classification scheme presented in Section 2. The CNN, shown in Figure 7, is formed by two 2-dimensional convolutional layers with no padding, each of them followed by a 2D max-pooling layer with 2 × 2 stride, with ELU activation functions. ELU activations were preferred over the ReLU family due to their robustness and faster learning rates [58]. Dropout layers with a ratio of 0.5 are placed after the last convolutional step and the first dense layer. The network ends in a dense layer of 256 neurons, followed by an output layer of c neurons. More network configuration details are included in Table 1. This architecture, proposed in [59], was chosen because of its low computational complexity and memory footprint compared to other alternatives such as residual neural networks [60], and was deemed adequate for the comparison of the different data augmentation methods. The proposed methods would, however, apply to other types of CNNs.

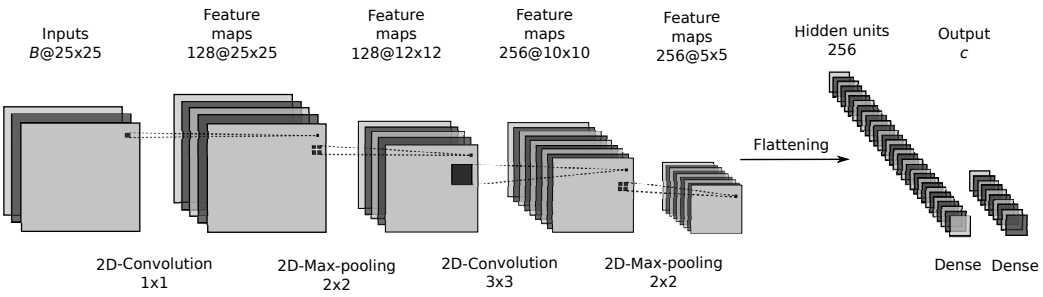


Figure 7. Proposed CNN architecture.

Table 1. Neural network architecture.

#	Type	Output Shape	Activation	Filter Size	Stride
1	2D-Convolutional	$25 \times 25 \times 128$	ELU	1×1	1×1
2	2D-MaxPooling	$12 \times 12 \times 128$	-	2×2	2×2
3	2D-Convolutional	$10 \times 10 \times 256$	ELU	3×3	1×1
4	2D-MaxPooling	$5 \times 5 \times 256$	-	2×2	2×2
5	Dropout	$5 \times 5 \times 256$	-	-	-
6	Flatten	1×6400	-	-	-
7	Dense	1×256	ELU	-	-
8	Dropout	1×256	-	-	-
9	Dense	$1 \times c$	Softmax	-	-

The training consisted of 77 epochs and used a NADAM [61] optimizer with learning_rate $= 0.0001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-7}$. Data augmentation is performed only during the first epoch and then cached during the rest of the training. Data is shuffled between epochs. The training data uses single-precision arithmetic.

A total of five imputation methods developed as part of this work were tested with the scenes described in Section 4.2. Constant imputation was performed using NumPy's [62] *nan_to_num* method. MICE imputation was performed using the reference R language [63] implementation through rpy2 [64] via *mice's* package [65] *mice* method with parameters $m = 3$ (number of imputations), and $method =$ "*pmm*". The KNNimpute, SoftImpute, and SVDimpute imputation methods were performed in Python [66], through the fancyimpute [67] package. KNNimpute imputation was performed using the KNN solver initialized with $k = 5$. SoftImpute imputation was performed using the SoftImpute solver initialized with parameters $convergence\_threshold = 0.001$, $max\_iters = 100$, and $init\_fill\_method =$ "*zero*". SVDimpute was performed through the IterativeSVD solver initialized with parameters $rank = 8$, $convergence\_threshold = 0.00001$, $max\_iters = 200$, and $init\_fill\_method =$ "*zero*". Additionally, the Dual-Flip data augmentation method described in [59] was also included in the comparison, using an augmentation factor of $2\times$, the same value used by the current proposal.

Three accuracy metrics [68] were collected during the experimental phase: the overall number of correctly classified pixels, or overall accuracy (OA), the mean of the number of correctly classified pixels for all classes, or average accuracy (AA), and Kappa (κ) [69]. The experimental data were collected running the classification scheme ten times per scenario, with a fixed number of randomly selected samples per run. The results in the tables correspond to the mean over the ten experiments. All the experiments were performed on a test system equipped with a 6-core Intel i5 8400 CPU at 2.80 GHz and 48 GB of RAM, and an NVIDIA GeForce GTX 3070ti with 8 GB of VRAM. The test system ran Ubuntu Linux 20.04 64-bits, Docker 20.10.7 with GPU support enabled, Python 3.6.9, Tensorflow 2.5.0 with GPU support enabled, and CUDA toolkit 11.3.

4.2. Datasets

In this section, the characteristics of the datasets used in the evaluation of the performance of the proposed approach are described. This includes training and test dataset composition, class counts, and so forth. Four vegetation multispectral scenes belonging to the Galicia dataset [70] and seventeen multispectral scenes from the large-scale, land cover Gaofen Image Dataset (GID) [71] were selected.

From the Galicia dataset, four scenes from river basins were selected, namely, River Oitavén, Creek Ermidas, Eiras Dam, and River Mestas, were considered. These images were captured at an altitude of 120 m by a UAV mounting a MicaSense RedEdge multispectral camera [72]. The sensor resolution is 8.2 cm/pixel and it covers a spectral range from 475 to 840 nm. Figure 8 displays the false-color composite and reference data for the four scenes.

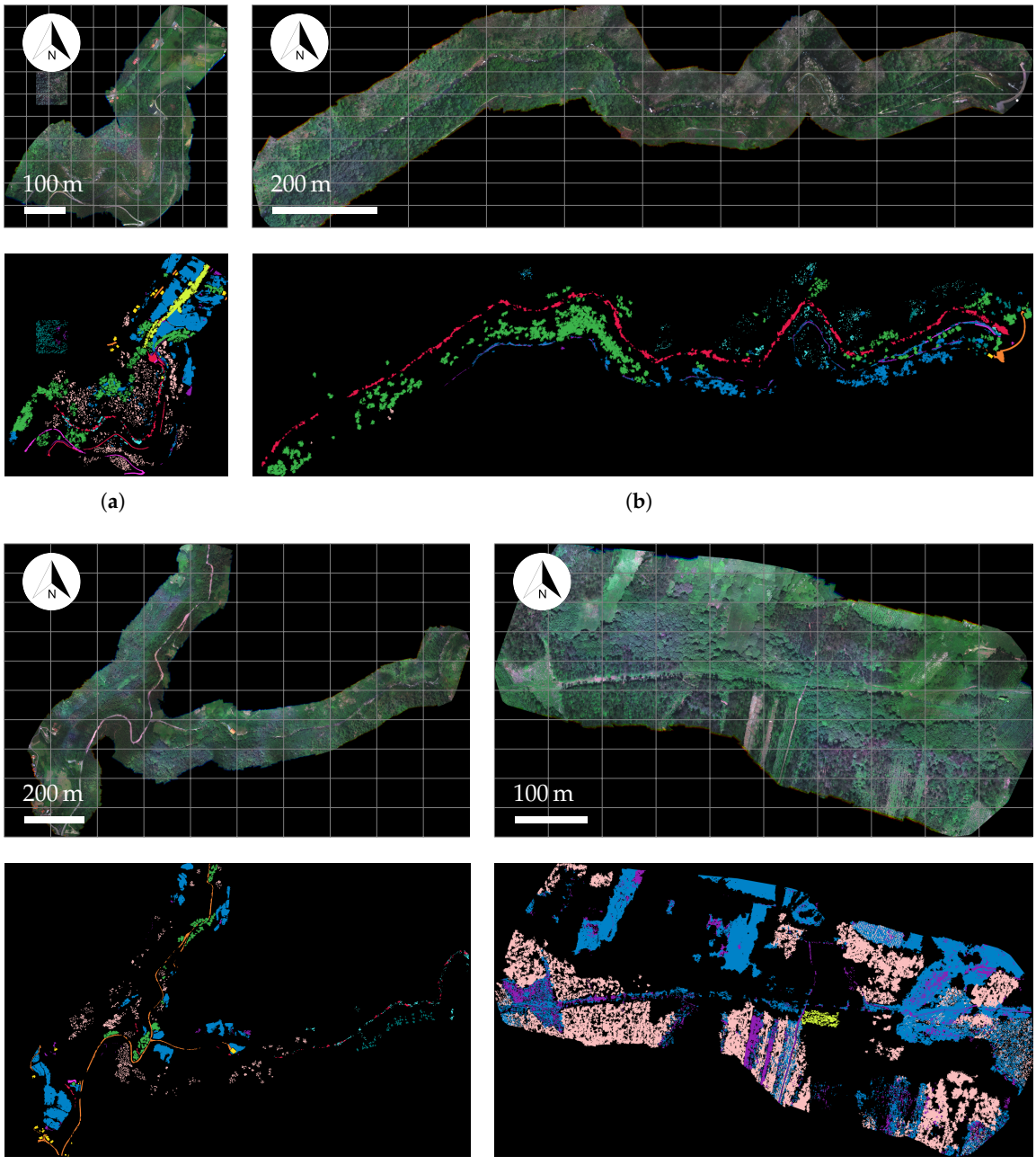


Figure 8. False color composite and ground truth for scenes from the Galicia dataset: (**a**) Oitaven, (**b**) Eiras, (**c**) Ermidas and (**d**) Mestas. Color codes are the same as those introduced in Table 2.

1. River Oitavén (Oitaven): Multispectral vegetation scene of the Oitaven river from Pontevedra, Spain. The image is 6689 × 6722 pixels and has 5 spectral bands. The scene is located at 42°22′15.3″N 8°25′47.4″W.
2. Creek Ermidas (Ermidas): Multispectral vegetation scene showing the point where Creek Ermidas and River Oitavén meet, from Pontevedra, Spain. The image is 11,924 × 18,972 pixels and has 5 spectral bands. The scene is located at 42°22′51.9″N 8°24′53.5″W.
3. Eiras Dam (Eiras): Multispectral vegetation scene showing the reservoir that supplies running water to the town of Vigo from Pontevedra, Spain. The scene is 5176 × 18,224 pixels and has 5 spectral bands. The scene is located at 42°20′46.5″N 8°30′10.5″W.

4. River Mestas (Mestas): Multispectral vegetation scene showing the River Mestas from Pontevedra, Spain. The image is 4915 × 9040 pixels and has 5 spectral bands. The scene is located at 43°38′29.8″N 7°58′42.2″W.

Two different groups of images were used from the GID. The first group contains the first seven images from the 5-class large-scale classification set. The second group contains the ten images from the 15-class large-scale classification set. These images were obtained by the Gaofen-2 (GF-2) satellite of the China National Space Administration. The GF-2 multispectral sensor has a resolution of 4 m per pixel and covers a spectral range from 450 to 890 nm. For simplicity's sake, we will refer to the images in the first group as GF2-5A to GF2-5G, and the images of the second group as GF2-15A to GF2-15J. Figures 9 and 10 display the false-color composite and reference data for the scenes of the first and second groups respectively.

Table 2 contains information on the number of superpixels of each class contained in the training and test datasets for all the scenes of the Galicia dataset. The number of superpixels remained constant during all the experiments across all the scenarios that were tested. To perform the experiments, each of the scenes was split into three disjoint subsets of superpixels. 60% of the samples were used for training, 20% of the samples were used for validation, and the remaining 20% were used for testing. Settings for the scenes from the GID are identical to those used for the Galicia dataset. Tables 3 and 4 contain information about the full name, internal code used in this paper (GF2-5x and GF2-15x), and the number of superpixels in the training and test datasets.

Table 2. Galicia dataset. Oitaven, Ermidas, Eiras and Mestas scenes. Color codes for the classes and number of superpixels in the training and testing sets. The "-" symbol denotes the absence of samples for the specified class.

		Oitaven			Ermidas	
#	**Classes**	**Train**	**Test**	**Classes**	**Train**	**Test**
1.	Water	367	120	Water	183	70
2.	Oak	2227	739	Oak	1201	390
3.	Tiles	112	48	Tiles	137	33
4.	Meadows	2709	863	Meadows	3550	1142
5.	Asphalt	52	14	Asphalt	976	331
6.	Bare Soil	256	92	Bare Soil	253	83
7.	Rock	192	72	Rock	522	172
8.	Concrete	206	79	Concrete	45	8
9.	Endemic veg.	592	197	Endemic veg.	-	-
10.	Eucaliptus	2553	838	Eucaliptus	2704	926
11.	Pines	430	151	Pines	41 3	149
		Eiras			**Mestas**	
#	**Classes**	**Train**	**Test**	**Classes**	**Train**	**Test**
1.	Water	823	278	Water	-	-
2.	Oak	3332	1066	Oak	-	-
3.	Tiles	12	2	Tiles	-	-
4.	Meadows	1476	485	Meadows	5373	1780
5.	Asphalt	86	32	Asphalt	-	-
6.	Bare Soil	247	86	Bare Soil	1653	487
7.	Rock	830	286	Rock	-	-
8.	Concrete	56	16	Concrete	-	-
9.	Endemic veg.	-	-	Endemic veg.	138	40
10.	Eucaliptus	23	5	Eucaliptus	5540	1887
11.	Pines	187	70	Pines	-	-

All the scenes were segmented using the simple linear iterative clustering (SLIC) [73] algorithm with an initial superpixel size of 800 and a compactness parameter of 40. The compactness parameter determines the balance between space and spectral proximity. Higher compactness favors spatial proximity and causes superpixels to be more square, whereas lower compactness usually produces more irregular shapes. The selected window dimensions used for all the datasets were 25 × 25, as a compromise between accuracy and performance.

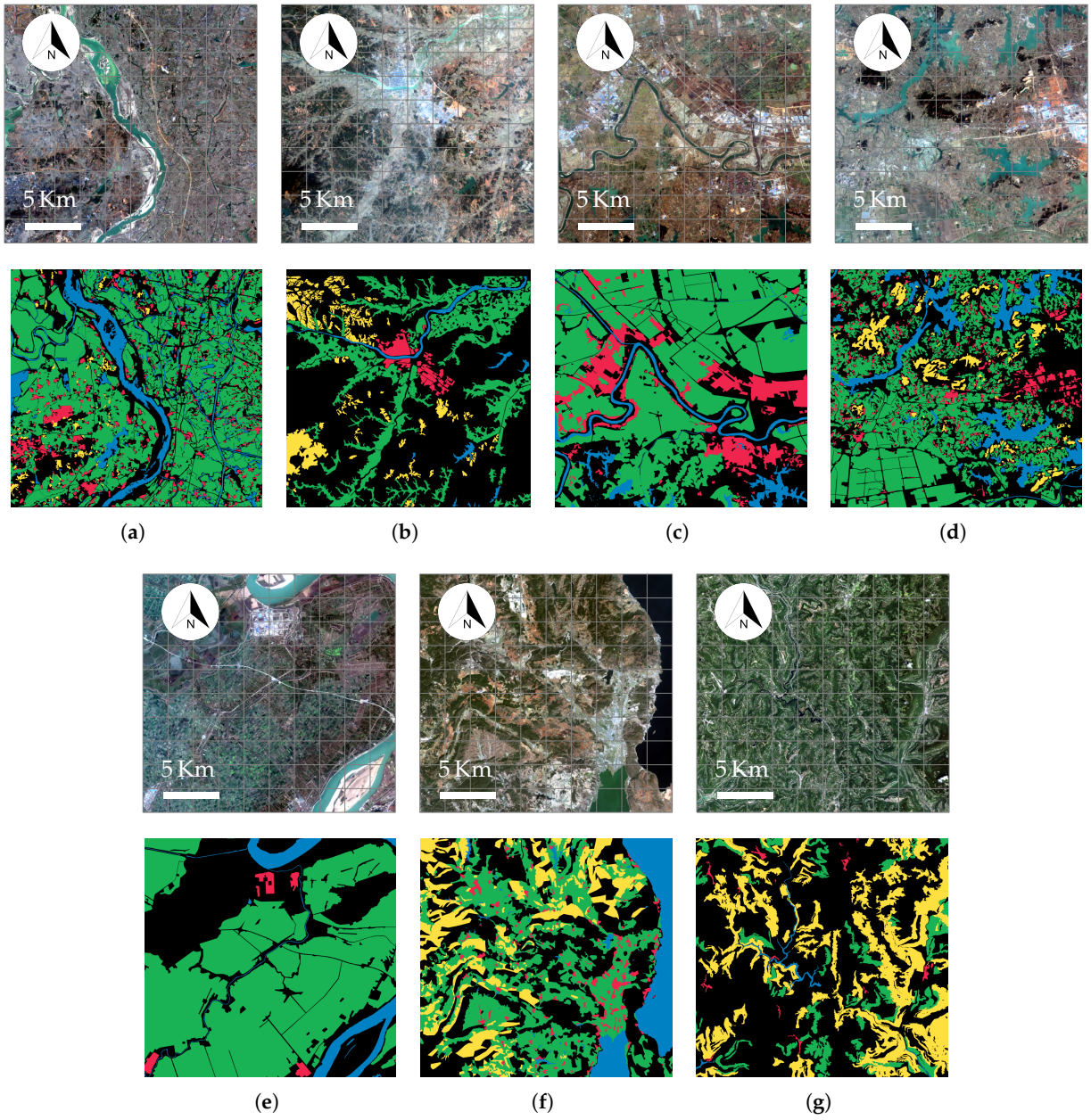


Figure 9. False color composite and ground truth for scenes from the GID: (**a**) GF2-5A, (**b**) GF2-5B, (**c**) GF2-5C, (**d**) GF2-5D, (**e**) GF2-5E, (**f**) GF2-5F, and (**g**) GF2-5G. Color codes are the same as those introduced in Table 3.

Table 3. Scenes from the 5-class large scene classification set of the GID. Color codes for the classes and number of superpixels in the training and testing sets. The "-" symbol denotes the absence of samples for the specified class. Classes are, in ascending order, built-up, farmland, forest, meadow, and water.

Set	Classes 1	2	3	4	5	Set	Classes 1	2	3	4	5
	GF2_PMS1__L1A0000564539-MSS1 - GF2-5A						GF2_PMS1__L1A0000647770-MSS1 - GF2-5E				
Train	3547	27,699	755	5151	-	Train	591	32,054	-	2139	-
Test	1109	9244	246	1737	-	Test	204	10,595	-	729	-
	GF2_PMS1__L1A0000575925-MSS1 - GF2-5B						GF2_PMS1__L1A0000708367-MSS1 - GF2-5F				
Train	1295	12,351	3759	1027	-	Train	1663	19,207	6795	4143	-
Test	477	4102	1219	345	-	Test	543	6382	2237	1395	-
	GF2_PMS1__L1A0000647767-MSS1 - GF2-5C						GF2_PMS1__L1A0000962382-MSS1 - GF2-5G				
Train	6432	27,704	-	2920	-	Train	476	5225	12,942	397	-
Test	2153	9281	-	911	-	Test	170	1738	4279	119	-
	GF2_PMS1__L1A0000647768-MSS1 - GF2-5D										
Train	3482	21,976	2316	3682	-						
Test	1247	7224	799	1204	-						

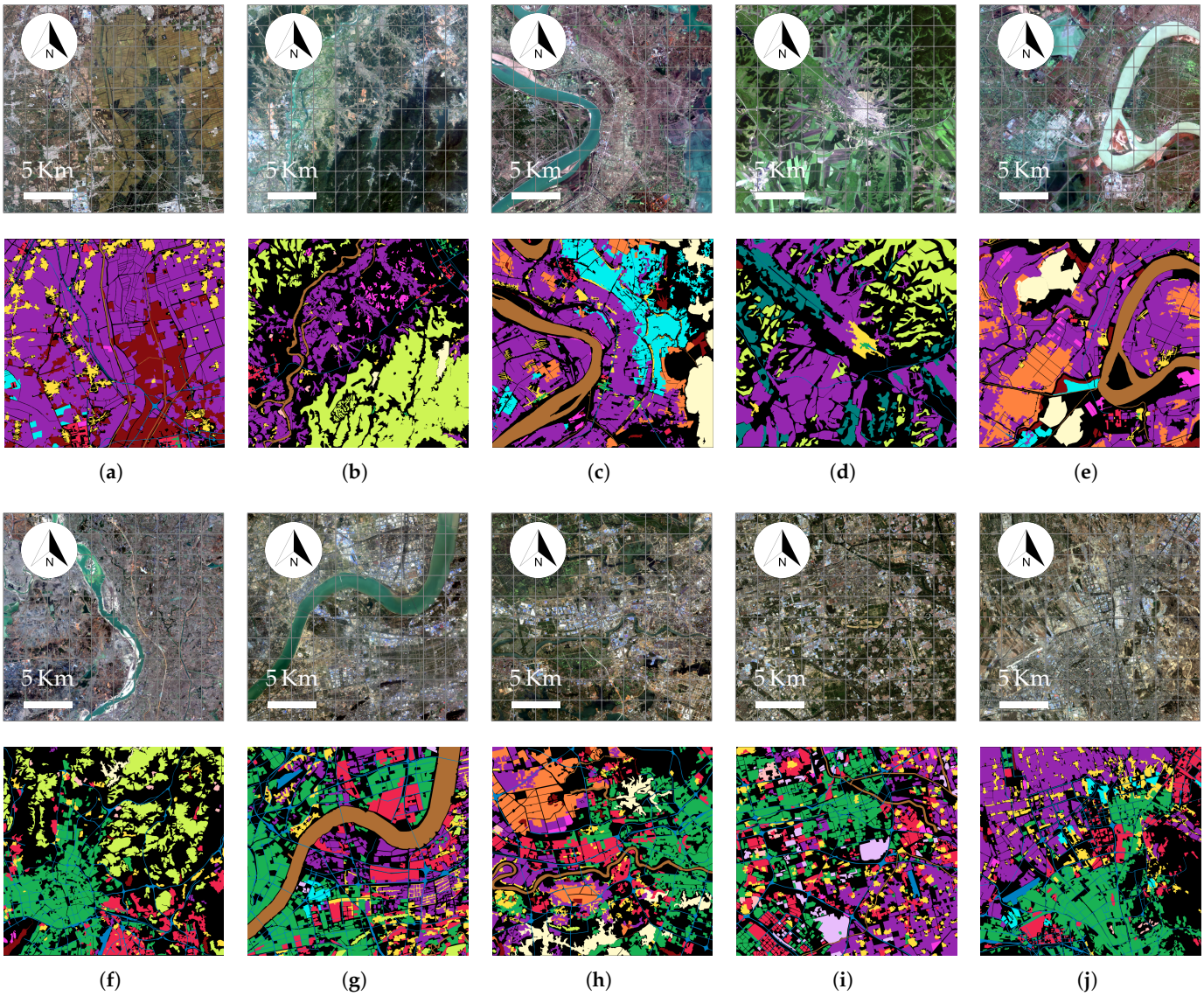


Figure 10. False color composite and ground truth for scenes from the GID: (**a**) GF2-15A, (**b**) GF2-15B, (**c**) GF2-15C, (**d**) GF2-15D, (**e**) GF2-15E, (**f**) GF2-15F, (**g**) GF2-15G, (**h**) GF2-15H, (**i**) GF2-15I, and (**j**) GF2-15J. Color codes are the same as those introduced in Table 4.

Table 4. Scenes from the 15-class large scene classification set of the GID. Color codes for the classes and number of superpixels in the training and testing sets. The "-" symbol denotes the absence of samples for the specified class. Classes are, in ascending order, industrial land, urban residential, rural residential, traffic land, paddy field, irrigated land, dry cropland, garden plot, arbor woodland, shrub land, natural grassland, artificial grassland, river, lake, and pond.

	Classes														
Set	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
GF2_PMS1__L1A0001064454-MSS1 - GF2-15A															
Train	1001	24	3627	1302	-	29,047	491	65	-	-	-	-	658	-	8169
Test	294	8	1238	442	-	9630	168	30	-	-	-	-	206	-	2754
GF2_PMS1__L1A0001118839-MSS1 - GF2-15B															
Train	632	165	564	774	-	9009	-	871	13,310	126	-	-	968	291	10
Test	191	49	182	272	-	2958	-	296	4440	45	-	-	337	90	6
GF2_PMS1__L1A0001395956-MSS1 - GF2-15C															
Train	338	133	1277	343	2596	17,872	7035	294	-	-	-	-	3728	2404	492
Test	130	36	423	104	851	5968	2296	89	-	-	-	-	1221	832	170
GF2_PMS1__L1A0001680858-MSS1 - GF2-15D															
Train	-	28	565	714	-	12,686	-	-	5815	-	5122	-	-	-	30
Test	-	13	203	213	-	4225	-	-	1990	-	1634	-	-	-	8
GF2_PMS2__L1A0000718813-MSS2 - GF2-15E															
Train	275	11	952	213	7462	20,709	547	798	-	-	-	-	3441	2608	872
Test	84	3	331	72	2470	6857	206	273	-	-	-	-	1164	856	277
GF2_PMS2__L1A0001378501-MSS2 - GF2-15F															
Train	3614	7748	1102	3815	-	175	-	63	8054	249	-	-	-	290	330
Test	1166	2563	351	1229	-	56	-	24	2817	81	-	-	-	92	87
GF2_PMS2__L1A0001471436-MSS2 - GF2-15G															
Train	6365	11,858	3001	5266	-	6557	382	-	473	-	-	156	4342	-	-
Test	2050	4019	1006	1788	-	2205	110	-	157	-	-	40	1395	-	-
GF2_PMS2__L1A0001517494-MSS2 - GF2-15H															
Train	4727	9354	1621	3983	5383	4931	-	107	115	44	-	-	1061	2459	423
Test	1554	3094	599	1281	1824	1644	-	40	32	18	-	-	371	784	132
GF2_PMS2__L1A0001787564-MSS2 - GF2-15I															
Train	7427	10,006	3054	3436	-	7511	-	307	-	448	-	2588	743	-	-
Test	2432	3362	1072	1131	-	2438	-	97	-	162	-	882	244	-	-
GF2_PMS2__L1A0001821754-MSS2 - GF2-15A															
Train	4401	9498	3585	5299	-	11,754	711	92	-	122	-	-	-	-	58
Test	1519	3138	1182	1780	-	3899	237	25	-	29	-	-	-	-	28

4.3. Experimental Results

Table 5 shows the experimental results for the scenes of the Galicia dataset. Figure 11 displays this information in a graphical way. KNNimpute obtains the highest accuracy for Mestas, and Ermidas, with an OA of 78.84% and 78.86%, respectively. Dual-Flip 2× obtained the best results for the Oitaven and Eiras scenes, which may indicate that augmentation methods based on geometric transforms offer better results for high-resolution datasets. Constant imputation obtains worse results than other imputation strategies in terms of accuracy. This can be attributed to the reduction in available information and the amount of bias this imputation strategy introduces.

Table 6 shows the experimental results for the scenes from the 5-class large-scale classification set of the GID. Figure 12 displays this information in a graphical way. It can be seen that all the proposed imputation methods increase the classification metrics for all the scenes. MICE is the method that obtained the highest classification accuracy in most of the scenes, yielding an increase in OA of 2.57 for the GF2-5G scene. MICE introduces high variability in the imputed data, as explained in Section 2.4, which further helps to generalize the classification model, and this variability will be higher the lower the resolution of the image. Pixels from lower resolution images are more prone to being uniform as they contain a mix of spectral information from several different materials. For these images, imputation by MICE generates more information the classifier can learn from.

In our particular case, the GID dataset scenes present lower spatial resolution and therefore better results. The results for the remaining methods are similar, with Dual-Flip 2× and Constant obtaining the worst overall results.

Table 5. Classification performance (in percent) for the scenes of the Galicia dataset. None denotes the baseline, or training with no data augmentation. Bold font indicates the best result for a scene.

		Oitaven	Eiras	Mestas	Ermidas
None	OA	78.88 ± 0.35	86.45 ± 0.20	78.30 ± 0.29	77.82 ± 0.57
	AA	71.93 ± 1.01	80.25 ± 1.70	71.36 ± 1.02	70.61 ± 0.81
	K	72.92 ± 0.55	81.26 ± 0.26	66.24 ± 0.45	68.96 ± 0.92
Constant	OA	79.62 ± 0.46	86.83 ± 0.13	78.45 ± 0.15	78.35 ± 0.16
	AA	73.39 ± 0.66	82.14 ± 0.40	70.63 ± 1.79	70.57 ± 1.28
	K	73.89 ± 0.64	81.81 ± 0.15	66.52 ± 0.24	69.72 ± 0.28
SoftImpute	OA	80.32 ± 0.17	87.02 ± 0.26	78.65 ± 0.05	78.73 ± 0.26
	AA	75.40 ± 0.81	82.05 ± 0.41	71.47 ± 1.68	72.62 ± 1.48
	K	74.84 ± 0.22	82.02 ± 0.39	66.85 ± 0.09	70.46 ± 0.48
SVDimpute	OA	80.61 ± 0.27	87.14 ± 0.28	78.62 ± 0.08	78.84 ± 0.22
	AA	75.56 ± 0.63	82.46 ± 0.49	71.46 ± 1.79	**73.29 ± 1.43**
	K	75.22 ± 0.33	82.24 ± 0.40	66.78 ± 0.14	70.47 ± 0.37
KNNimpute	OA	80.26 ± 0.24	87.31 ± 0.23	**78.84 ± 0.03**	**78.86 ± 0.09**
	AA	75.08 ± 0.80	**82.90 ± 0.51**	**72.20 ± 0.74**	72.52 ± 1.07
	K	74.75 ± 0.33	82.45 ± 0.31	**67.10 ± 0.08**	**70.48 ± 0.15**
MICE	OA	80.01 ± 0.28	86.98 ± 0.05	78.49 ± 0.09	78.17 ± 0.28
	AA	74.97 ± 0.62	80.95 ± 1.14	72.02 ± 0.75	69.89 ± 1.17
	K	74.44 ± 0.40	82.00 ± 0.10	66.63 ± 0.10	69.57 ± 0.54
Dual-Flip 2× [59]	OA	**81.02 ± 0.18**	**87.43 ± 0.06**	78.46 ± 0.21	78.73 ± 0.18
	AA	**75.61 ± 0.50**	82.77 ± 0.43	71.49 ± 1.10	73.30 ± 1.57
	K	**75.73 ± 0.23**	**82.60 ± 0.08**	66.54 ± 0.34	70.44 ± 0.18

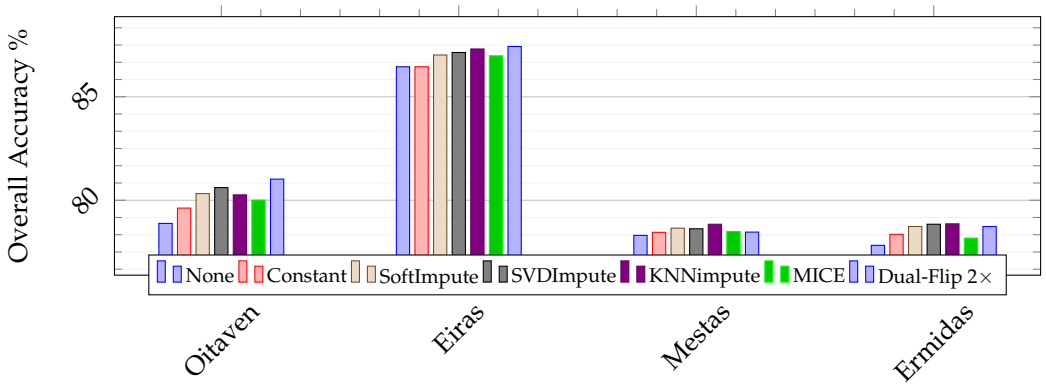


Figure 11. Classification performance (in percent) for the scenes from the for the scenes of the Galicia dataset. None denotes the baseline, or training with no data augmentation.

Table 6. Classification performance (in percent) for the scenes from the 5-class large scene classification set of the GID. None denotes the baseline, or training with no data augmentation. Bold font indicates the best result for a scene.

		GF2-5A	GF2-5B	GF2-5C	GF2-5D	GF2-5E	GF2-5F	GF2-5G
None	OA	95.86 ± 0.04	98.26 ± 0.34	94.57 ± 0.13	96.53 ± 0.04	99.53 ± 0.05	91.51 ± 0.23	90.87 ± 0.69
	AA	90.84 ± 0.66	95.00 ± 1.38	89.11 ± 0.37	92.69 ± 0.24	97.80 ± 0.35	85.37 ± 0.53	80.96 ± 2.57
	K	89.38 ± 0.10	96.40 ± 0.73	85.81 ± 0.37	92.83 ± 0.09	97.20 ± 0.26	86.21 ± 0.41	77.61 ± 2.07
Constant	OA	96.03 ± 0.06	98.88 ± 0.09	95.04 ± 0.46	96.66 ± 0.29	99.51 ± 0.04	92.09 ± 0.19	91.96 ± 0.42
	AA	91.33 ± 0.46	97.44 ± 0.38	90.25 ± 2.21	93.37 ± 1.19	**97.90 ± 0.40**	86.13 ± 0.67	84.03 ± 3.00
	K	89.80 ± 0.17	97.71 ± 0.18	87.19 ± 1.31	93.12 ± 0.64	97.08 ± 0.22	87.12 ± 0.33	80.46 ± 1.28
SoftImpute	OA	96.03 ± 0.16	98.73 ± 0.15	95.45 ± 0.11	96.78 ± 0.10	99.57 ± 0.03	92.14 ± 0.31	91.90 ± 0.29
	AA	90.44 ± 1.74	96.73 ± 0.80	91.84 ± 0.51	93.80 ± 0.36	97.80 ± 0.18	86.74 ± 1.27	85.87 ± 0.92
	K	89.80 ± 0.47	97.39 ± 0.31	88.39 ± 0.28	93.39 ± 0.20	97.47 ± 0.19	87.27 ± 0.58	80.17 ± 0.80
SVDImpute	OA	96.09 ± 0.06	98.89 ± 0.06	95.55 ± 0.05	96.82 ± 0.08	99.55 ± 0.03	92.33 ± 0.08	92.16 ± 0.40
	AA	91.42 ± 0.67	97.35 ± 0.30	91.18 ± 0.44	93.73 ± 0.45	97.77 ± 0.70	87.05 ± 0.48	86.69 ± 0.89
	K	90.00 ± 0.16	97.73 ± 0.12	88.54 ± 0.14	93.46 ± 0.20	97.31 ± 0.19	87.53 ± 0.13	80.87 ± 1.13
KNNimpute	OA	96.13 ± 0.10	98.75 ± 0.22	95.29 ± 0.31	96.80 ± 0.05	**99.59 ± 0.04**	91.78 ± 0.51	92.62 ± 0.19
	AA	91.51 ± 0.43	97.47 ± 0.24	90.88 ± 0.52	93.82 ± 0.47	97.62 ± 0.55	86.47 ± 0.83	87.14 ± 0.23
	K	90.10 ± 0.24	97.45 ± 0.44	87.83 ± 0.84	93.43 ± 0.12	**97.57 ± 0.23**	86.58 ± 0.90	82.07 ± 0.56
MICE	OA	**96.49 ± 0.05**	**98.99 ± 0.21**	**96.10 ± 0.08**	**97.07 ± 0.07**	99.58 ± 0.06	**92.86 ± 0.10**	**93.44 ± 0.24**
	AA	**91.96 ± 0.32**	**97.86 ± 0.30**	**93.34 ± 0.51**	**93.92 ± 0.30**	97.36 ± 0.84	**88.07 ± 0.40**	**88.39 ± 0.75**
	K	**91.03 ± 0.14**	**97.93 ± 0.42**	**90.08 ± 0.15**	**93.96 ± 0.16**	97.51 ± 0.37	**88.45 ± 0.18**	**84.24 ± 0.67**
Dual-Flip 2× [59]	OA	96.01 ± 0.06	98.78 ± 0.17	95.37 ± 0.18	96.84 ± 0.08	99.57 ± 0.07	91.69 ± 0.36	91.84 ± 0.39
	AA	91.30 ± 0.38	97.07 ± 0.28	91.52 ± 0.91	93.59 ± 0.41	97.44 ± 0.89	85.60 ± 0.85	84.82 ± 0.90
	K	89.72 ± 0.19	97.50 ± 0.35	88.12 ± 0.44	93.49 ± 0.18	97.43 ± 0.42	86.46 ± 0.65	80.08 ± 1.13

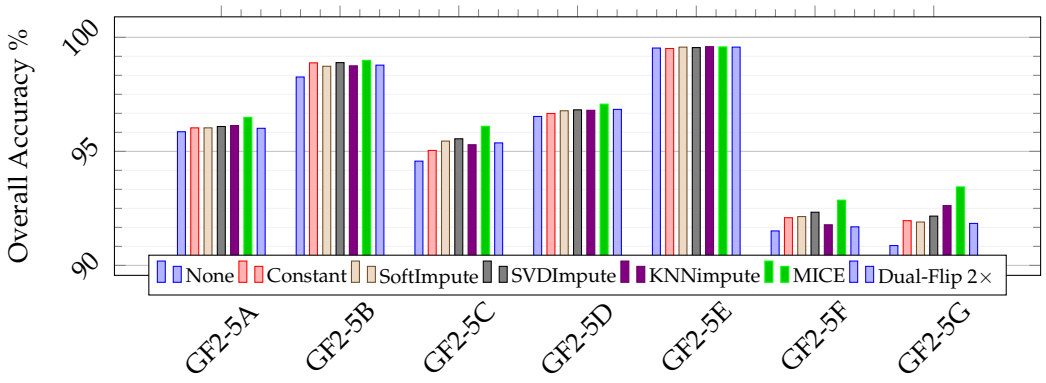


Figure 12. Classification performance (in percent) for the scenes from the 5-class large scale classification set of the GID. None denotes the baseline, or training with no data augmentation.

Table 7 shows the experimental results for the scenes from the 15-class large-scale classification set of the GID. Figure 13 displays this information in a graphical way. The situation is similar to the previous case, with MICE obtaining the best accuracy results for all the scenes. These scenes are more complex and have the different classes mixed in more irregular patterns than the GID scenes from the 5-class large-scale classification set. The more irregular the structures, and the higher the number of frontiers between groups of pixels labeled with different classes, the higher chances of producing better classification results. Constant imputation and Dual-Flip 2× obtains again the lowest accuracy of all the methods studied.

Table 8 displays the execution times of our proposal for all the imputation methods previously considered, excluding the initial segmentation time, for the scenes of the 15-class large scene classification set of the GID. At a glance, it can be seen that the prediction times, denoted by Pred. in the table, are very similar across all the methods for each of the images.

This is due to imputation not being applied during prediction. When data augmentation is performed, the application of imputation methods causes an increase in execution time during training. This is especially the case with KNNimpute, which is markedly slower than the average of the other methods tested. KNNimpute is a computationally intensive algorithm that needs to calculate the distances for every pair of vectors present in the dataset that has to be completed. Constant imputation offers the lowest execution time of all the imputation methods due to its linear complexity. While the asymptotic complexity of SoftImpute is lower than that of SVDimpute, the small size of the dataset that has to be imputed causes both methods to have very similar execution times. Dual-Flip 2×, being based on geometric transformations, has a lower training time than any of the data imputation methods developed as part of this work.

Table 7. Classification performance (in percent) for the scenes from the 15-class large scale classification set of the GID. None denotes the baseline, or training with no data augmentation. Bold font indicates the best result for a scene.

		GF2-15A	GF2-15B	GF2-15C	GF2-15D	GF2-15E	GF2-15F	GF2-15G	GF2-15H	GF2-15I	GF2-15J
None	OA	96.08 ± 0.05	96.47 ± 0.11	89.58 ± 0.11	97.24 ± 0.15	92.04 ± 0.16	86.6 ± 0.20	82.25 ± 0.13	79.96 ± 0.23	76.91 ± 0.41	81.38 ± 0.18
	AA	73.07 ± 0.36	74.42 ± 1.59	76.38 ± 1.57	87.98 ± 1.19	75.24 ± 1.89	75.00 ± 0.71	78.91 ± 1.05	60.49 ± 1.18	66.06 ± 1.87	68.11 ± 2.00
	K	92.24 ± 0.10	93.97 ± 0.21	85.69 ± 0.13	95.59 ± 0.25	87.94 ± 0.28	81.75 ± 0.22	78.07 ± 0.19	76.00 ± 0.28	71.31 ± 0.55	75.08 ± 0.27
Constant	OA	96.41 ± 0.06	96.76 ± 0.19	90.71 ± 0.20	97.5 ± 0.08	92.69 ± 0.22	88.95 ± 0.12	85.08 ± 0.37	82.18 ± 0.29	79.69 ± 0.45	84.8 ± 0.22
	AA	79.49 ± 0.67	74.54 ± 1.19	77.80 ± 1.82	87.22 ± 5.79	79.27 ± 0.74	78.98 ± 1.28	80.67 ± 1.56	66.94 ± 1.27	69.75 ± 2.62	70.3 ± 2.10
	K	92.93 ± 0.11	94.48 ± 0.32	87.25 ± 0.31	96.03 ± 0.12	88.99 ± 0.33	84.97 ± 0.18	81.52 ± 0.47	78.62 ± 0.33	74.78 ± 0.58	79.62 ± 0.27
SoftImpute	OA	96.38 ± 0.07	97.03 ± 0.20	90.98 ± 0.18	97.56 ± 0.16	92.77 ± 0.23	88.93 ± 0.22	85.29 ± 0.17	82.98 ± 0.11	80.63 ± 0.31	84.84 ± 0.28
	AA	78.65 ± 1.14	80.83 ± 3.17	79.29 ± 0.97	**89.56 ± 0.68**	77.64 ± 1.18	78.76 ± 1.42	81.75 ± 1.18	69.71 ± 1.43	70.99 ± 2.10	70.75 ± 2.01
	K	92.87 ± 0.13	94.93 ± 0.35	87.63 ± 0.28	96.10 ± 0.27	89.07 ± 0.38	84.89 ± 0.31	81.8 ± 0.19	79.59 ± 0.16	75.87 ± 0.43	79.72 ± 0.36
SVDImpute	OA	96.43 ± 0.12	96.98 ± 0.34	91.03 ± 0.12	97.70 ± 0.08	93.10 ± 0.04	88.73 ± 0.24	85.77 ± 0.19	82.82 ± 0.60	81.11 ± 0.47	85.70 ± 0.26
	AA	**80.14 ± 2.94**	81.24 ± 3.62	79.50 ± 0.50	88.90 ± 1.17	78.95 ± 0.41	79.00 ± 0.97	82.88 ± 0.57	66.81 ± 3.14	71.15 ± 1.90	72.75 ± 1.62
	K	92.97 ± 0.23	94.86 ± 0.60	87.71 ± 0.15	96.33 ± 0.12	89.64 ± 0.09	84.60 ± 0.37	82.37 ± 0.23	79.37 ± 0.78	76.48 ± 0.67	80.82 ± 0.34
KNNimpute	OA	96.41 ± 0.07	96.93 ± 0.33	91.09 ± 0.20	97.66 ± 0.16	93.15 ± 0.12	89.11 ± 0.21	85.41 ± 0.15	82.64 ± 0.24	80.63 ± 0.28	84.66 ± 0.42
	AA	79.14 ± 0.84	78.02 ± 2.04	78.84 ± 2.22	89.30 ± 0.41	79.17 ± 0.79	79.67 ± 1.56	81.75 ± 0.42	69.27 ± 1.90	71.24 ± 0.77	70.93 ± 2.20
	K	92.93 ± 0.15	94.78 ± 0.53	87.79 ± 0.30	96.27 ± 0.27	89.7 ± 0.18	85.14 ± 0.32	81.93 ± 0.22	79.18 ± 0.31	75.91 ± 0.34	79.38 ± 0.62
MICE	OA	**96.68 ± 0.05**	**97.23 ± 0.32**	**91.73 ± 0.36**	**97.93 ± 0.07**	**93.86 ± 0.11**	**90.39 ± 0.28**	**87.31 ± 0.18**	**84.63 ± 0.28**	**82.52 ± 0.18**	**86.74 ± 0.22**
	AA	78.32 ± 1.76	**82.21 ± 2.34**	**80.22 ± 1.76**	87.33 ± 0.43	**80.37 ± 0.95**	**82.28 ± 2.51**	**84.62 ± 0.47**	**72.92 ± 1.11**	**73.78 ± 0.89**	**73.58 ± 1.55**
	K	**93.46 ± 0.10**	**95.28 ± 0.55**	**88.68 ± 0.48**	**96.69 ± 0.12**	**90.76 ± 0.15**	**86.94 ± 0.37**	**84.31 ± 0.23**	**81.59 ± 0.35**	**78.25 ± 0.22**	**82.24 ± 0.29**
Dual-Flip 2× [59]	OA	96.36 ± 0.06	96.90 ± 0.12	90.81 ± 0.07	97.62 ± 0.08	93.17 ± 0.19	88.35 ± 0.10	84.25 ± 0.22	82.44 ± 0.08	79.41 ± 0.41	83.68 ± 0.10
	AA	78.69 ± 0.78	80.47 ± 3.96	77.73 ± 1.03	86.39 ± 0.87	76.76 ± 1.81	77.90 ± 0.96	80.59 ± 0.94	66.71 ± 0.95	68.64 ± 1.42	68.80 ± 2.70
	K	92.83 ± 0.12	94.72 ± 0.21	87.38 ± 0.12	96.20 ± 0.12	89.69 ± 0.31	84.15 ± 0.17	80.52 ± 0.27	78.95 ± 0.10	74.37 ± 0.55	78.08 ± 0.19

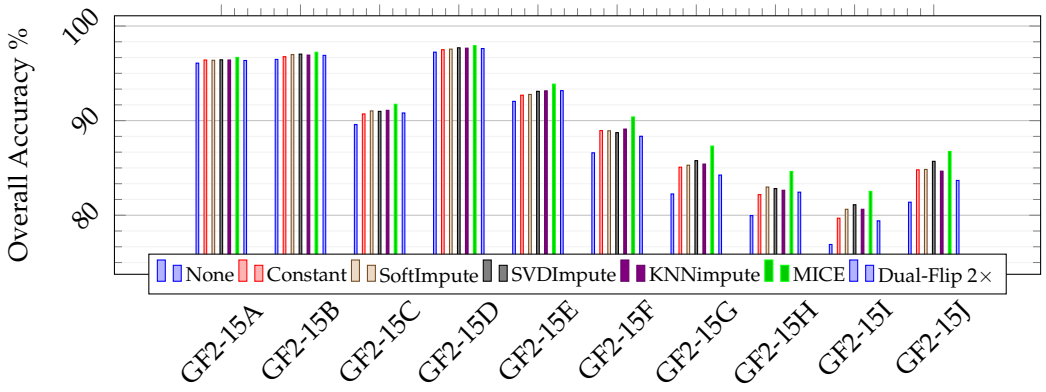


Figure 13. Classification performance (in percent) for the scenes from the 15-class large scale classification set of the GID. None denotes the baseline, or training with no data augmentation.

Table 8. Execution times (in seconds) for the scenes from the 15-class large scale classification set of the GID. None denotes the baseline, or training with no data augmentation. Bold font indicates the best result for a scene.

		GF2-15A	GF2-15B	GF2-15C	GF2-15D	GF2-15E	GF2-15F	GF2-15G	GF2-15H	GF2-15I	GF2-15J
None	Train	**769.7**	**472.3**	**644.0**	**439.5**	**641.6**	**450.8**	**681.2**	**612.1**	**640.2**	**638.0**
	Pred.	**35.7**	**33.8**	**35.8**	**29.8**	**35.3**	**33.6**	**36.1**	**35.9**	**35.7**	**35.1**
Constant	Train	1267.8	769.9	1051.3	723.8	1084.3	639.2	1174.7	948.9	874.2	974.2
	Pred.	48.8	45.5	49.4	41.5	44.3	48.3	48.9	50.9	50.7	49.9
SoftImpute	Train	1829.9	1120.6	1530.2	1052.2	1577.1	1080.2	1616.2	1445.5	1501.6	1501.6
	Pred.	45.7	41.9	45.5	38.5	44.4	42.7	46.2	46.0	45.2	45.4
SVDImpute	Train	1763.1	1231.6	1697.6	1161.0	1741.1	1231.3	1828.6	1659.6	1779.9	1758.2
	Pred.	45.3	42.2	45.3	38.4	44.3	42.5	46.0	45.8	45.6	45.4
KNNimpute	Train	3625.8	2379.2	3049.1	2181.3	3152	2250.8	3346.7	2920.0	3087.0	3040.0
	Pred.	45.6	42.1	45.4	38.5	44.2	42.7	46.2	45.8	45.1	45.2
MICE	Train	1848.4	1126.2	1436.2	985.8	1495.7	1603.3	2051.1	2070.4	2994.7	2361.8
	Pred.	45.3	42.5	45.5	38.5	44.3	42.8	46.1	4	45.42	45.46
Dual-Flip 2× [59]	Train	1081.5	674.3	922.0	623.40	960.6	635.3	976.1	733.7	857.2	878.6
	Pred.	60.9	51.8	56.7	46.7	54.8	49.8	56.9	57.3	54.0	55.4

5. Discussion

Several data augmentation methods to be used in combination with multispectral and hyperspectral images exist in the literature. These data augmentation methods, however, were mostly developed for pixel-based classification schemes. Among those, methods based on geometric transformations of the data can be found in [74–76]. Ref. [59] introduces a data augmentation framework for both pixel and superpixel-based classification that uses geometric transforms after subdividing the input patches into two independent regions to generate new samples. In [77], new samples are synthesized by drawing data from a multivariate normal distribution initialized with the standard deviations of the bands of an HSI. Generative Adversarial Networks are used in [24,27] to create new hyperspectral samples. A recent [28] proposes two data augmentation methods based on generating new samples by selecting similar samples with different window sizes. We propose a data augmentation scheme targeting segment-based classification built on the application of data imputation methods to the original patches serving as samples. Data imputation is applied to these patches after performing data erasure of the pixels that are not considered to be relevant. These pixels are selected based on the segmentation information obtained in prior steps.

The use of segmentation is not a new concept in the hyperspectral classification field. In [41], information from segmentation is used as a means to enhance the classification results by exploiting the spatial consistency of neighboring pixels. A similar approach can be seen in [42], where superpixel segmentation is applied to the prediction maps generated by a CNN. Then the information obtained is used as a robust estimation of local spatial features in the sense of Bayesian inference. The work from [59] already introduced the use of superpixel segmentation as a way to perform sample selection and reduce the computational performance of the classification process. The data augmentation method proposed in this work makes use of the information obtained after the segmentation of the multispectral images not only to perform sample selection but also as part of the data augmentation step. The data augmentation approach followed is based on the application of data imputation methods to generate synthetic pixel vectors. It constitutes the first instance of such use in the literature and shows that these methods are valid for generating new synthetic information by damaging and repairing input patches up to an arbitrary degree using data imputation. This is true for both hyperspectral and multispectral images. Several data imputation methods have been proposed to be used as part of this newly developed data augmentation approach and an extensive comparison using 21 multispectral datasets has been carried out. The classification scheme can also be applied to hyperspectral images without any changes.

The experimental results show that the proposal is capable of increasing the classification performance of all the multispectral datasets studied regardless of the particular data imputation method chosen. KNNimpute, in particular, obtains the best classification results for the multispectral scenes from the Galicia dataset. MICE obtains the highest OA across 16 out of the 17 multispectral scenes from the GID. The OA increase achieved reaches up to 5.61% for the GF2-15I scene. The execution times obtained for the 15-class large-scale classification set of the GID show the computational penalty associated with the prediction stage is negligible, and the training stage shows an increase of 2.15× to 5.3×.

It is important to note that the resulting performance is influenced by the size of the segments and the dimensions of the sliding window used to perform the patch extraction. If the segment size is too big or, conversely, the patch is too small in relation to the segment size, the proposed data augmentation method will not generate new synthetic samples.

6. Conclusions

In this work, a new data augmentation framework for CNN-based classification built on the combination of superpixel segmentation and data imputation is introduced. The new framework is deployed in a classification scheme consisting of four steps: segmentation, patch extraction, patch erasure, and patch imputation. The original HSI is first segmented and a patch is extracted from each segment. Patches then undergo a pixel erasure stage, leaving only the data belonging to the SOI. Data augmentation is then used to replace the missing pixels, and the new patch is fed to a CNN classifier.

The proposal was evaluated using two multispectral land-cover classification datasets; a very high spatial resolution dataset captured from a drone (Galicia dataset), and a lower resolution, large scene dataset captured by the Gaofen-2 satellite (GID). Five common imputation methods, constant imputation, KNNimpute, MICE, SoftImpute, and SVDimpute were evaluated. The results obtained show that the proposed scheme is capable of increasing the performance across all scenes independently of the imputation method being used. KNNimpute obtains the best accuracy for two of the four scenes from the high-resolution Galicia dataset. MICE obtains the highest accuracy for 16 out of the 17 scenes from the lower resolution GID dataset.

In future work, we plan to focus our efforts on studying the impact of selectively deleting information from certain bands to perform the imputation, without making use of the segment information. This would allow us to extend this data augmentation method to traditional pixel-based classification schemes and simplify parameter tuning.

Author Contributions: Conceptualization, F.A.; data curation, Á.A.; formal analysis, F.A. and D.B.H.; investigation, Á.A.; methodology, F.A. and D.B.H.; resources, Á.A., F.A. and D.B.H.; software, Á.A.; supervision, F.A. and D.B.H.; validation, Á.A., F.A. and D.B.H.; visualization, Á.A.; writing—original draft, Á.A.; writing—review and editing, Á.A., F.A. and D.B.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Ministerio de Ciencia e Innovación, Government of Spain (grant numbers PID2019-104834GB-I00 and BES-2017-080920), the Consellería de Educación, Universidade e Formación Profesional (grant number ED431C 2018/19, and accreditation 2019–2022 ED431G-2019/04), and by the Junta de Castilla y León (project VA226P20 (PROPHET II Project)). All are co-funded by the European Regional Development Fund (ERDF).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AA	average accuracy
CNN	convolutional neural networks
DL	deep learning
DWS	dual-window superpixel
HSI	hyperspectral image
MSI	multispectral image
OA	overall accuracy
UAV	unmanned aerial vehicle

References

1. Shaw, G.A. Spectral imaging for remote sensing. *Linc. Lab. J.* **2003**, *14*, 3–28.
2. Ghamisi, P.; Maggiori, E.; Li, S.; Souza, R.; Tarabalka, Y.; Moser, G.; De Giorgi, A.; Fang, L.; Chen, Y.; Chi, M.; et al. New frontiers in spectral-spatial hyperspectral image classification: The latest advances based on mathematical morphology, Markov random fields, segmentation, sparse representation, and deep learning. *IEEE Geosci. Remote Sens. Mag.* **2018**, *6*, 10–43. [CrossRef]
3. Lanthier, Y.; Bannari, A.; Haboudane, D.; Miller, J.R.; Tremblay, N. Hyperspectral data segmentation and classification in precision agriculture: A multi-scale analysis. In Proceedings of the IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium, Boston, MA, USA, 7–11 July 2008; Volume 2, p. II-585.
4. Mahlein, A.K. Plant disease detection by imaging sensors–parallels and specific demands for precision agriculture and plant phenotyping. *Plant Dis.* **2016**, *100*, 241–251. [CrossRef] [PubMed]
5. Moroni, M.; Mei, A.; Leonardi, A.; Lupo, E.; Marca, F.L. PET and PVC separation with hyperspectral imagery. *Sensors* **2015**, *15*, 2205–2227. [CrossRef] [PubMed]
6. Entezari, I.; Rivard, B.; Vajihinejad, V.; Wilson, W.G.; Soares, J.B.; Fisseha, B.; Beier, N. Monitoring tailings flocculation performance using hyperspectral imagery. *Can. J. Chem. Eng.* **2019**, *97*, 2465–2471. [CrossRef]
7. Yan, Y.; Ren, J.; Tschannerl, J.; Zhao, H.; Harrison, B.; Jack, F. Nondestructive Phenolic Compounds Measurement and Origin Discrimination of Peated Barley Malt Using Near-Infrared Hyperspectral Imagery and Machine Learning. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–15. [CrossRef]
8. McNaught, A.D.; Wilkinson, A. *Compendium of Chemical Terminology*; Blackwell Science: Oxford, UK, 1997; Volume 1669.
9. Ponce, J.; Karahoca, A. *Data Mining and Knowledge Discovery in Real Life Applications*; BoD—Books on Demand: Norderstedt, Germany, 2009.
10. Chen, Y.; Lin, Z.; Zhao, X.; Wang, G.; Gu, Y. Deep learning-based classification of hyperspectral data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 2094–2107. [CrossRef]
11. Makantasis, K.; Karantzalos, K.; Doulamis, A.; Doulamis, N. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, Italy, 26–31 July 2015; pp. 4959–4962.
12. Chen, Y.; Zhao, X.; Jia, X. Spectral–Spatial classification of hyperspectral data based on deep belief network. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 2381–2392. [CrossRef]
13. Chen, Y.; Jiang, H.; Li, C.; Jia, X.; Ghamisi, P. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 6232–6251. [CrossRef]
14. Lee, H.; Kwon, H. Going deeper with contextual CNN for hyperspectral image classification. *IEEE Trans. Image Process.* **2017**, *26*, 4843–4855. [CrossRef] [PubMed]
15. Audebert, N.; Le Saux, B.; Lefèvre, S. Deep learning for classification of hyperspectral data: A comparative review. *IEEE Geosci. Remote Sens. Mag.* **2019**, *7*, 159–173. [CrossRef]
16. Paoletti, M.; Haut, J.; Plaza, J.; Plaza, A. Deep learning classifiers for hyperspectral imaging: A review. *ISPRS J. Photogramm. Remote Sens.* **2019**, *158*, 279–317. [CrossRef]
17. Hong, D.; Gao, L.; Yao, J.; Zhang, B.; Plaza, A.; Chanussot, J. Graph convolutional networks for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 5966–5978. [CrossRef]
18. Lawrence, S.; Giles, C.L.; Tsoi, A.C.; Back, A.D. Face recognition: A convolutional neural-network approach. *IEEE Trans. Neural Netw.* **1997**, *8*, 98–113. [CrossRef] [PubMed]
19. Shabbir, S.; Ahmad, M. Hyperspectral Image Classification–Traditional to Deep Models: A Survey for Future Prospects. *arXiv* **2021**, arXiv:2101.06116.
20. Taylor, L.; Nitschke, G. Improving deep learning using generic data augmentation. *arXiv* **2017**, arXiv:1708.06020.
21. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 60. [CrossRef]
22. Montserrat, D.M.; Lin, Q.; Allebach, J.; Delp, E.J. Training object detection and recognition CNN models using data augmentation. *Electron. Imaging* **2017**, *2017*, 27–36. [CrossRef]
23. Li, W.; Wu, G.; Zhang, F.; Du, Q. Hyperspectral image classification using deep pixel-pair features. *IEEE Trans. Geosci. Remote Sens.* **2016**, *55*, 844–853. [CrossRef]

24. Audebert, N.; Le Saux, B.; Lefèvre, S. Generative adversarial networks for realistic synthesis of hyperspectral samples. In Proceedings of the IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, Valencia, Spain, 22–27 July 2018; pp. 4359–4362.
25. Li, W.; Chen, C.; Zhang, M.; Li, H.; Du, Q. Data augmentation for hyperspectral image classification with deep CNN. *IEEE Geosci. Remote Sens. Lett.* **2018**, *16*, 593–597. [CrossRef]
26. Nalepa, J.; Myller, M.; Kawulok, M. Training-and test-time data augmentation for hyperspectral image segmentation. *IEEE Geosci. Remote Sens. Lett.* **2019**, *17*, 292–296. [CrossRef]
27. Alipourfard, T.; Arefi, H. Virtual Training Sample Generation by Generative Adversarial Networks for Hyperspectral Images Classification. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *42*, 63–69. [CrossRef]
28. Wang, W.; Liu, X.; Mou, X. Data Augmentation and Spectral Structure Features for Limited Samples Hyperspectral Classification. *Remote Sens.* **2021**, *13*, 547. [CrossRef]
29. van Buuren, S. *Flexible Imputation of Missing Data*; Chapman & Hall/CRC Interdisciplinary Statistics, CRC Press LLC: Boca Raton, FL, USA, 2018.
30. Graham, J.W.; Cumsille, P.E.; Shevock, A.E. Methods for handling missing data. In *Handbook of Psychology*, 2nd ed.; American Psychological Association: Washington, DC, USA, 2012; Volume 2.
31. Graham, J.W. Missing data analysis: Making it work in the real world. *Annu. Rev. Psychol.* **2009**, *60*, 549–576. [CrossRef] [PubMed]
32. Rubin, D.B. Inference and missing data. *Biometrika* **1976**, *63*, 581–592. [CrossRef]
33. Kang, H. The prevention and handling of the missing data. *Korean J. Anesthesiol.* **2013**, *64*, 402. [CrossRef] [PubMed]
34. Buuren, S.v.; Groothuis-Oudshoorn, K. mice: Multivariate imputation by chained equations in R. *J. Stat. Softw.* **2010**, 45, 1–67. [CrossRef]
35. Candès, E.J.; Recht, B. Exact matrix completion via convex optimization. *Found. Comput. Math.* **2009**, *9*, 717–772. [CrossRef]
36. Mazumder, R.; Hastie, T.; Tibshirani, R. Spectral regularization algorithms for learning large incomplete matrices. *J. Mach. Learn. Res.* **2010**, *11*, 2287–2322.
37. Shapiro, L. *Computer Vision and Image Processing*; Academic Press: Cambridge, MA, USA, 1992.
38. He, Z.; Shen, Y.; Zhang, M.; Wang, Q.; Wang, Y.; Yu, R. Spectral-spatial hyperspectral image classification via SVM and superpixel segmentation. In Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings, Montevideo, Uruguay, 12–15 May 2014; pp. 422–427.
39. Fang, L.; Li, S.; Duan, W.; Ren, J.; Benediktsson, J.A. Classification of hyperspectral images by exploiting spectral–spatial information of superpixel via multiple kernels. *IEEE Trans. Geosci. Remote Sens.* **2015**, *53*, 6663–6674. [CrossRef]
40. Priya, T.; Prasad, S.; Wu, H. Superpixels for spatially reinforced Bayesian classification of hyperspectral images. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 1071–1075. [CrossRef]
41. Liu, Y.; Cao, G.; Sun, Q.; Siegel, M. Hyperspectral classification via deep networks and superpixel segmentation. *Int. J. Remote Sens.* **2015**, *36*, 3459–3482. [CrossRef]
42. Cao, J.; Chen, Z.; Wang, B. Deep convolutional networks with superpixel segmentation for hyperspectral image classification. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 10–15 July 2016; pp. 3310–3313.
43. Wang, M.; Liu, X.; Gao, Y.; Ma, X.; Soomro, N.Q. Superpixel segmentation: A benchmark. *Signal Process. Image Commun.* **2017**, *56*, 28–39. [CrossRef]
44. Alam, F.I.; Zhou, J.; Liew, A.W.C.; Jia, X. CRF learning with CNN features for hyperspectral image segmentation. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 10–15 July 2016; pp. 6890–6893.
45. Pal, N.R.; Pal, S.K. A review on image segmentation techniques. *Pattern Recognit.* **1993**, *26*, 1277–1294. [CrossRef]
46. Troyanskaya, O.; Cantor, M.; Sherlock, G.; Brown, P.; Hastie, T.; Tibshirani, R.; Botstein, D.; Altman, R.B. Missing value estimation methods for DNA microarrays. *Bioinformatics* **2001**, *17*, 520–525. [CrossRef] [PubMed]
47. Somasundaram, R.; Nedunchezhian, R. Evaluation of three simple imputation methods for enhancing preprocessing of data with missing values. *Int. J. Comput. Appl.* **2011**, *21*, 14–19. [CrossRef]
48. Tutz, G.; Ramzan, S. Improved methods for the imputation of missing data by nearest neighbor methods. *Comput. Stat. Data Anal.* **2015**, *90*, 84–99. [CrossRef]
49. De Silva, H.; Perera, A.S. Missing data imputation using Evolutionary k-Nearest neighbor algorithm for gene expression data. In Proceedings of the Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer), Negombo, Sri Lanka, 1–3 September 2016; pp. 141–146.
50. Shah, A.D.; Bartlett, J.W.; Carpenter, J.; Nicholas, O.; Hemingway, H. Comparison of random forest and parametric imputation models for imputing missing data using MICE: A CALIBER study. *Am. J. Epidemiol.* **2014**, *179*, 764–774. [CrossRef] [PubMed]
51. Jolani, S.; Debray, T.P.; Koffijberg, H.; van Buuren, S.; Moons, K.G. Imputation of systematically missing predictors in an individual participant data meta-analysis: A generalized approach using MICE. *Stat. Med.* **2015**, *34*, 1841–1863. [CrossRef] [PubMed]
52. Yao, Q.; Kwok, J.T. Accelerated and inexact soft-impute for large-scale matrix and tensor completion. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 1665–1679. [CrossRef]

53. Sakaori, F.; Kurosawa, H. Fully Bayesian Soft Impute for Matrix Completion. Available online: https://oparu.uni-ulm.de/xmlui/bitstream/handle/123456789/4563/GerJap2016.pdf?sequence=1&isAllowed=y#page=17 (accessed on 23 October 2021).
54. Oba, S.; Sato, M.A.; Takemasa, I.; Monden, M.; Matsubara, K.i.; Ishii, S. A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics* **2003**, *19*, 2088–2096. [CrossRef] [PubMed]
55. Cai, Z.; Heydari, M.; Lin, G. Iterated local least squares microarray missing value imputation. *J. Bioinform. Comput. Biol.* **2006**, *4*, 935–957. [CrossRef] [PubMed]
56. Brand, J. Development, Implementation and Evaluation of Multiple Imputation Strategies for the Statistical Analysis of Incomplete Data Sets. Ph.D. Thesis, Erasmus University Rotterdam, Rotterdam, The Netherlands, 1999.
57. Martín Ballesteros, X. Comparative Study of Missing Data Treatment Methods in Radial Basis Function Neural Networks: Is It Necessary to Impute? Bachelor's Thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, 2020.
58. Clevert, D.A.; Unterthiner, T.; Hochreiter, S. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *arXiv* **2015**, arXiv:1511.07289.
59. Acción, Á.; Argüello, F.; Heras, D.B. Dual-Window Superpixel Data Augmentation for Hyperspectral Image Classification. *Appl. Sci.* **2020**, *10*, 8833. [CrossRef]
60. Shao, Y.; Lan, J.; Liang, Y.; Hu, J. Residual networks with multi-attention mechanism for hyperspectral image classification. *Arab. J. Geosci.* **2021**, *14*, 252. [CrossRef]
61. Dozat, T. Incorporating Nesterov Momentum into Adam. In Proceedings of the 4th International Conference on Learning Representations (ICLR), Workshop Track, ICLR, San Juan, Puerto Rico, 2–4 May 2016.
62. Harris, C.R.; Millman, K.J.; van der Walt, S.J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N.J.; et al. Array programming with NumPy. *Nature* **2020**, *585*, 357–362. [CrossRef] [PubMed]
63. R Core Team. *R: A Language and Environment for Statistical Computing*; R Foundation for Statistical Computing: Vienna, Austria, 2021.
64. Gautier, L. Python Interface to the R Language, Python Package Version 3.4.5. Available online: https://rpy2.github.io/doc/v3.4.x/html/index.html (accessed on 23 October 2021).
65. van Buuren, S.; Groothuis-Oudshoorn, K. mice: Multivariate Imputation by Chained Equations, R Package Version 3.13.0. Available online: https://github.com/amices/mice (accessed on 23 October 2021).
66. Van Rossum, G.; Drake, F.L., Jr. *Python Reference Manual*; Centrum voor Wiskunde en Informatica Amsterdam: Amsterdam, The Netherlands, 1995.
67. Rubinsteyn, A.; Feldman, S. Fancyimpute: An Imputation Library for Python. Available online: https://github.com/iskandr/fancyimpute (accessed on 23 October 2021).
68. He, L.; Li, J.; Liu, C.; Li, S. Recent advances on spectral–spatial hyperspectral image classification: An overview and new guidelines. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 1579–1597. [CrossRef]
69. Richards, J.A.; Richards, J. *Remote Sensing Digital Image Analysis*; Springer: Berlin/Heidelberg, Germany, 1999; Volume 3.
70. Bascoy, P.G.; Garea, A.S.; Heras, D.B.; Argüello, F.; Ordóñez, A. Texture-based analysis of hydrographical basins with multispectral imagery. In *Remote Sensing for Agriculture, Ecosystems, and Hydrology XXI*; International Society for Optics and Photonics: Bellingham, WA, USA, 2019; Volume 11149, p. 111490Q.
71. Tong, X.Y.; Xia, G.S.; Lu, Q.; Shen, H.; Li, S.; You, S.; Zhang, L. Land-Cover Classification with High-Resolution Remote Sensing Images Using Transferable Deep Models. *Remote Sens. Environ.* **2020**, *237*, 111322. [CrossRef]
72. MicaSense RedEdge MX Multispectral Camera. Available online: https://micasense.com/rededge-mx/ (accessed on 13 October 2020).
73. Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; Süsstrunk, S. *SLIC Superpixels*; Technical Report; EPFL: Lausanne, Switzerland, 2010.
74. Zhang, H.; Li, Y.; Zhang, Y.; Shen, Q. Spectral-spatial classification of hyperspectral imagery using a dual-channel convolutional neural network. *Remote Sens. Lett.* **2017**, *8*, 438–447. [CrossRef]
75. Yu, X.; Wu, X.; Luo, C.; Ren, P. Deep learning in remote sensing scene classification: A data augmentation enhanced convolutional neural network framework. *Gisci. Remote Sens.* **2017**, *54*, 741–758. [CrossRef]
76. Haut, J.M.; Paoletti, M.E.; Plaza, J.; Plaza, A.; Li, J. Hyperspectral image classification using random occlusion data augmentation. *IEEE Geosci. Remote Sens. Lett.* **2019**, *16*, 1751–1755. [CrossRef]
77. Slavkovikj, V.; Verstockt, S.; De Neve, W.; Van Hoecke, S.; Van de Walle, R. Hyperspectral image classification with convolutional neural networks. In Proceedings of the 23rd ACM international conference on Multimedia, Brisbane, Australia, 26–30 October 2015; pp. 1159–1162.