AiRound and CV-BrCT: Novel Multiview Datasets for Scene Classification

Gabriel Machado, Edemir Ferreira, Keiller Nogueira, Hugo Oliveira, *Member, IEEE*, Matheus Brito, Pedro Henrique Targino Gama, and Jefersson Alex dos Santos, *Member, IEEE*

***Abstract*—It is undeniable that aerial/satellite images can provide useful information for a large variety of tasks. But, since these images are always taken from above, some applications can benefit from complementary information provided by other perspective views of the scene, such as ground-level images. Despite a large number of public repositories for both georeferenced photographs and aerial images, there is a lack of benchmark datasets that allow the development of approaches that exploit the benefits and complementarity of aerial/ground imagery. In this article, we present two new publicly available datasets named AiRound and CV-BrCT. The first one contains triplets of images from the same geographic coordinate with different perspectives of view extracted from various places around the world. Each triplet is composed of an aerial RGB image, a ground-level perspective image, and a Sentinel-2 sample. The second dataset contains pairs of aerial and street-level images extracted from southeast Brazil. We design an extensive set of experiments concerning multiview scene classification, using early and late fusion. Such experiments were conducted to show that image classification can be enhanced using multiview data.**

***Index Terms*—Data fusion, dataset, deep learning, feature fusion, multimodal machine learning, remote sensing.**

I. Introduction

SATELLITE images become more accessible to civilian applications each year. New technologies are enabling the wide usage of better and cheaper images in comparison with the past few decades. Nowadays, it is also possible to access many free remote sensing image repositories with a variety of spatial, spectral, and temporal resolutions [1]. Images with aerial perspective give us a unique view of the world, allowing the capture of relevant information (not provided by any other type of image) that may assist in several applications, such as automatic geographic mapping and urban planning.

Despite the clear benefits of optical aerial imagery, the fact that they are always taken from above may make their use limited. Precisely, the presence of vegetation cover, clouds, or simply the need of more detailed on-the-ground information can decrease the effectiveness of such images in some applications. In multiview scenarios, it would be crucial to combine the complementary information of aerial and ground images in order to efficiently tackle a problem. Such combination of multiple sources images can benefit many applications in different fields, such as 3-D human pose estimation [2], places geolocalization [3], and urban land use [4]. Motivated by these benefits, several approaches [5]–[9] have been proposed to exploit multiview datasets to face distinct tasks. Although important, it is not easy to find multiview datasets for image-related tasks, given the difficulty in creating and labeling such data. In fact, as far as we know, there is no other publicly available multiview (aerial and ground) dataset for image classification tasks in the literature.

In this article, we present two novel multiview images datasets. The main purpose of creating these datasets is to make them publicly available so that the scientific community can carry out image classification experiments in multiview scenarios. One of the datasets is composed of 11 753 triplets of images, each one of those consisting of a ground scene, a high-resolution aerial image, and a multispectral aerial data. The images are unevenly divided into 11 classes, including airport, bridge, church, forest, lake, park, river, skyscraper, stadium, statue, and tower. An interesting property of our dataset is that it was designed to contain a high interclass variety; so, these places were selected from all around the world to compose the samples. The other dataset is composed of 24k pairs of images, each one containing a street-level scene and a high-resolution aerial image. Those samples are labeled in nine different classes, which include apartment, hospital, house, industrial, parking lot, religious, school, store, and vacant lot. Both datasets were evaluated for image classification, using early and late fusion strategies. Although we assessed the performance of both datasets for image classification, it is important to emphasize that they were proposed to be used in distinct image-related tasks, varying from image classification to cross-view matching and multimodal learning.

In summary, the main contributions of this work are the *two* novel multiview scene classification datasets, named AiRound and CV-BrCT, as well as a full evaluation of the proposed datasets in image classification tasks using several deep learning state-of-the-art methods and fusion approaches.

The remainder of this article is organized as follows. Section II presents related work. The proposed datasets are presented in Section III, whereas Section IV introduces the methods and

Manuscript received July 30, 2020; revised September 30, 2020; accepted October 17, 2020. Date of publication October 23, 2020; date of current version January 6, 2021. This work was supported in part by CAPES, in part by CNPq under Grant 424700/2018-2 and Grant 311395/2018-0, and in part by FAPEMIG under Grant APQ-00449-17. *(Corresponding author: Gabriel Machado.)*

Gabriel Machado, Edemir Ferreira, Hugo Oliveira, Matheus Brito, Pedro Henrique Targino Gama, and Jefersson Alex dos Santos are with the Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte 31270-901, Brazil (e-mail: gabriel.lucas@dcc.ufmg.br; edemirm@dcc.ufmg.br; oliveirahugo@dcc.ufmg.br; matheus.brito000@gmail.com; pehtg13@gmail.com; jefersson@dcc.ufmg.br).

Keiller Nogueira is with the Division of Computing Science and Mathematics, University of Stirling, FK9 4LA Stirling, U.K. (e-mail: keiller.nogueira@dcc.ufmg.br).

Digital Object Identifier 10.1109/JSTARS.2020.3033424

TABLE I
PROPERTIES OF OTHER DATASETS FOUND IN THE LITERATURE THAT ARE SIMILAR TO AIROUND AND CV-BRCT

Dataset	Image Type			Publicly Available	Paired Aerial/Ground Images	Total of Samples	Number of Classes	Task	Year
	Aerial RGB	Ground	Multispectral						
CV-USA [14]	✓	✓	✗	✓	✓	$\sim 44k$	-	Cross-View Matching	2015
Cities [15]	✓	✓	✗	✗	✓	$\sim 156k$	-	Cross-View Matching	2015
Pasadena Urban Trees [7]	✓	✓	✗	✓	✓	$\sim 100k$	18	Object Detection	2016
Brooklyn and Queens [17]	✓	✓	✗	✓	✓	$\sim 213k$	-	Instance Segmentation	2017
Urban Environments [16]	✓	✓	✗	✗	✓	$\sim 18k$	-	Cross-View Matching	2017
CV-ACT [9]	✓	✓	✗	✓	✓	$\sim 128k$	-	Cross-View Matching	2019
Buildings [5]	✓	✓	✗	✗	✓	$\sim 261k$	4	Classification	2019
Île-de-France land use [4]	✓	✓	✗	✗	✓	∼50k	16	Classification	2019
AiRound (ours)	✓	✓	✓	✓	✓	∼35.4k	11	Classification	2020
CV-BrCT (ours)	✓	✓	✗	✓	✓	∼48k	9	Classification	2020

tasks evaluated using these datasets. The experimental setup is introduced in Section V, whereas Section VI presents the obtained results. Finally, Section VII concludes the article.

II. RELATED WORK

Considering recent advances in satellite data acquisition and cloud computing, access to high-resolution satellite images and other types of data was facilitated. Despite the great advantages that aerial images provide, some applications demand information that an aerial perspective may lack. In these cases, an alternative solution is to use complementary perspectives of the same view, i.e., ground-level view, to better seek these pieces of information [10]–[13]. Due to the high demand for images to be used by those kinds of tasks, a lot of multiview datasets were proposed in the literature. In Table I, we summarized some of the most similar datasets compared to the novel ones proposed for this work.

The CV-USA [14] and CV-ACT [9] datasets were proposed specifically for retrieval tasks, i.e., cross-view matching. The first one contains millions of pairs of aerial and ground images, which were taken from across the United States. Relating to its data collection, the aerial images were collected using Bing Maps (BM) API, and the ground images used Flickr and Google Street View (GSV) API. Another important aspect to mention is that even if CV-USA has millions of samples, most of the works use a subset of it, which has around 44k images. Relating to the CV-ACT dataset, it contains approximately 128k images, which were taken covering a dense area of the city Canberra. All its images were collected using Google Maps (GM) and GSV APIs. Similarly, Cities [15] and Urban Environments [16] datasets were designed to tackle cross-view matching problem, but both of them were not publicly released. The first one used Google APIs to collect pairs of images from cities around the world. The latter one collected pairs of images from Pittsburg, Orlando, and Manhattan using GSV and BM APIs.

The Pasadena Urban Trees [7] was designed for object detection. This dataset used OpenStreetMap's (OSM) bounding box annotations of trees in the city of Pasadena. It contains 18 different species of trees, whose ground samples were collected using GSV API, and the aerial ones were collected using GM API.

Another multiview dataset was named Brooklyn and Queens [17], and it was proposed for instance segmentation. It contains approximately 213k images of 206 different types of buildings, covering areas from the two boroughs of New York City. All the images from this dataset were collected using BM and GSV APIs, and OSM was used to define the labels of all samples.

Relating to the Buildings [5] and Île-de-France land use [4] datasets, both were designed for multiview scene classification. The first dataset contains 56 259 paired aerial/street-level images of 4 different types of buildings, covering Washington, D.C., Puerto Rico, and 49 different states across the United States. Relating to the first dataset, all of its building labels were defined using annotations contained in OSM. The data collection was made using two different APIs, being those, GM API for aerial samples, and GSV API for the ground perspective ones. The Île-de-France land use dataset contains approximately 25k pairs of aerial/ground images of 16 different land use classes, covering the metropolitan region of Paris and some nearby suburbs. This dataset also uses OSM to collect its labels, and the same APIs of the Buildings dataset to collect the samples.

Differentiating our datasets from the ones in Table I, some of the existing datasets were designed in a way that each image pair can be seen as a class. Such datasets do not contain groups of classes that share the same label, which ends up making its use for image classification unenviable. Other datasets are quite different from the ones proposed here, given that the main task for which they were proposed is different. That difference mainly comes because those problems require different types of labels as inputs and also generates distinct outputs (bounding boxes and segmentation). Finally, relating to multiview image classification datasets, two datasets [4], [5] are quite similar to both datasets proposed here. However, neither of these existing datasets is publicly available, whereas ours will be.

III. PROPOSED DATASETS

In this work, we proposed two novel multiview datasets. *It is important to mention that both datasets are publicly available for research purposes at the project's website*.[1] Since both datasets were designed in a different way, in the following sections, we will describe the relevant characteristics of each one and the methodologies used to collect the samples.

A. AiRound Dataset

The first dataset is named AiRound, and is composed of 11 753 images distributed among 11 classes, including: airport,

[1][Online]. Available: http://www.patreo.dcc.ufmg.br/multi-view-datasets/

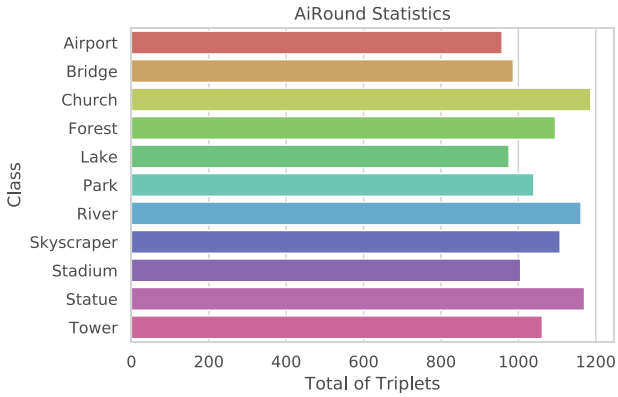


Fig. 1. Class distribution of the proposed AiRound dataset. Note that each image is represented by a triplet of ground, aerial, and multispectral data.

bridge, church, forest, lake, park, river, skyscraper, stadium, statue, and tower. Each sample is composed of a triplet, which contains following images in three distinct points of view:

1) a ground perspective image;
2) a high-resolution RGB aerial image; and
3) a multispectral image taken from the Sentinel-2 satellite.

All the images collected for this dataset corresponds to real places around the world. The distribution of the samples from AiRound can be checked in Fig. 1 and examples of instances can be seen in Fig. 2.

This dataset was created using two methodologies. In the first one, to download the samples, two types of metadata were required: 1) the name of the place; and 2) its correspondent geographical coordinates. These metadata were collected using web crawlers in diversified lists of Wikipedia web pages. As instance, a list of tallest buildings[2] was used to ensure that samples from building class have been extracted from different parts of the world. For more details about the web pages used and all the metadata collected from them, we recommend checking the project's website.[1]

Given the metadata, the RGB aerial images were collected using BM API.[3] The zoom level was empirically selected in order to adapt a proper vision for the samples of each class. Since there is a huge difference in areas occupied for some classes (river and skyscraper, for instance), this zoom level ended with large variance, specifically between 5 and 19, which corresponds to a spatial resolution that varies from 4891.97 to 0.30 m per pixel. Finally, it is important to mention that all aerial images downloaded have an image size of 500×500 pixels.

In order to collect the ground level samples, it was checked if the correspondent class exists in the Google Places' database. If the sample class exists, a query was built using this place's geographical coordinates as input. The outputs returned by this API were all manually checked, and if they do not correspond to the class, then a second protocol was performed. The second protocol was used for cases that the class did not exist in Google Places' database or the image retrieved did not correspond to the query requested. This protocol consists of crawling the top five images from Google Images using, as query, the place's name. Finally, it was manually selected to represent each sample on AiRound, the best instance between the five images downloaded. It should be pointed out that the resolution of each sample is not standardized because the methodology employed does not allow the selection of a specific resolution.

Relating to the Sentinel-2 images acquisition, we followed exactly the same protocol that was proposed by Ferreira *et al.* [18]. In this protocol, Google Earth Engine [1] was used to download the data using the place's geographical coordinates. After careful analysis, we decided to resize all images to 224×224 pixels, a resolution that could cover all the classes' areas.

After working with this methodology for a while, we noticed that it was not scalable because of limited metadata (per class) available in the Wikipedia lists. Due to this, we decided to move to another more scalable methodology. The second methodology is applied to build this dataset, the metadata were obtained from the publicly available data of the OpenStreetMap,[4] a community-based project where users annotate aerial images to create maps, and were collected using the Overpass API.[5] As the data are provided by users, not necessarily specialists, they can contain poorly annotated samples, which can lead to outliers in the dataset. These lists consist of only geographic coordinates, for most of the classes, with exception of the classes forest, lake, river, and park, which we collected from the only places that have a name assigned to it. The lists are then fed to scripts that utilize the Google StaticMap API,[6] to collect the aerial images, and the GSV API,[7] to collect the frontal images. Except for the zoom parameter, which was set to a proper value per class empirically, the default values of the Google APIs were used for the aerial images. Since we could not retrieve street-level images for the classes forest, lake, river, and park, we used its name as a query in a Google Images crawler. We followed the same protocol used in the first methodology to download images from these classes. To download the Sentinel-2 images, we also applied the same protocol used in the first methodology. Finally, since we gathered a large collection of locations, we ignored points where we could not retrieve an image from each view.

As a final step, an additional removal of outliers was applied after all the images were collected. This final filter consisted of a feature vector of the first obtained ground images. These feature vectors were produced by a ResNet pretrained on the ImageNet dataset, collected from the final layer of the architecture. Then, for each class, a k-means++ [19] clusterization was applied using these feature vectors. With the clusters, the distance of each data point, within a class, was calculated to its closest centroid as well as the mean and standard deviation distance of each cluster. Points that were more than three standard deviations away from a centroid cluster were removed from the dataset.

[2][Online]. Available: https://en.wikipedia.org/wiki/List_of_tallest_buildings
[3][Online]. Available: https://docs.microsoft.com/en-us/bingmaps/
[4][Online]. Available: www.openstreetmap.org/
[5][Online]. Available: https://overpass-turbo.eu/
[6][Online]. Available: https://developers.google.com/maps/documentation/maps-static/intro
[7][Online]. Available: https://developers.google.com/maps/documentation/streetview/intro

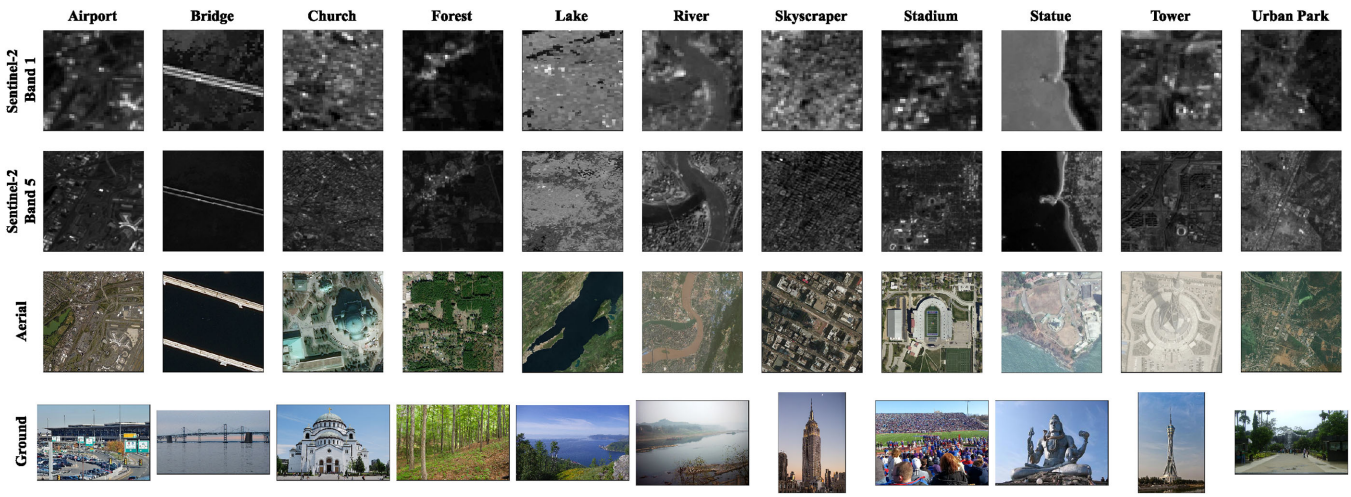


Fig. 2. Examples of instances taken from AiRound. The two top rows show channels of a Sentinel-2 sample, whereas the third and fourth rows show the high-resolution aerial perspective image and the ground view image, respectively.

Even with these removal operations, by the nature of the data collection and the simplicity of the filters applied, noise may be present in the dataset. However, we assume that the noise is minimal after all the process.

B. CV-BrCT

The CV-BrCT dataset, which stands for Cross-View *Brazilian Construction* Type, comprises approximate 24k pairs of images and split into 9 urban classes. The pairs are composed of images from two different views: an aerial view and a frontal view of a location. This dataset is focused on the urban environment and the nine classes are as follows.

1) Apartment: Buildings with at least two stories primarily for residential use.
2) Hospital: Health-related constructions, primarily hospitals but can include small particular clinics.
3) House: A single-family residence.
4) Industrial: Manufactured-related buildings, which include large-storage constructions.
5) Parking Lot: Includes both open and indoors parking lots.
6) Religious: Religious buildings; this include catholic churches and protestant churches.
7) School: Any school construction; from elementary schools to high school.
8) Store: Any commercial- or service-related building.
9) Vacant Lot: Demarked areas without construction. It can include abandoned open areas.

The class distribution is presented in Fig. 3, whereas examples can be seen in Fig. 4. Regarding the images, all of them are 500×500 RGB images. As implied by the name, this dataset contains only Brazilian locations. These are mainly in the Southeast region of Brazil, specifically the states of Minas Gerais and São Paulo, with some classes adding locations from states from other regions, i.e., Goiás in the Center-West region.

The lists of coordinates used to build this dataset were collected using the second methodology described in Section III-A,

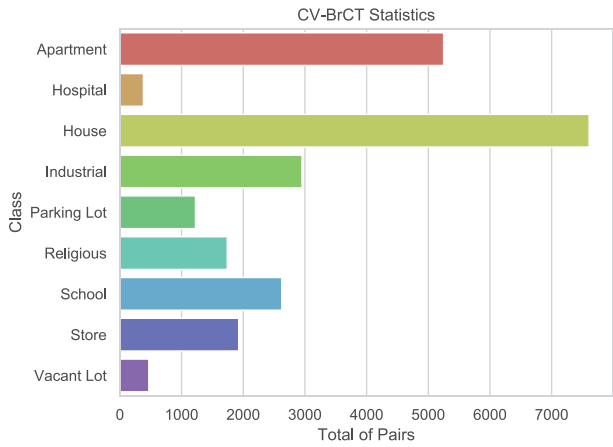


Fig. 3. Class distribution of the proposed CV-BrCT dataset.

which was applied to all classes, except Vacant Lot, that was manually annotated. To download the samples, we also use the same APIs of the second methodology previously described. The Google StaticMap API's zoom parameter was set to 19, which was defined empirically; the other parameters were used with their default values. Finally, it is important to mention that we decided not to collect samples from Sentinel-2 satellite for this dataset. This decision was made considering the nature of all the classes, which only includes objects that would not benefit from Sentinel-2's resolution.

IV. Benchmarked Methods

This section presents the evaluated methods. Different approaches were tested for multiview scene classification. In order to better assess the improvement provided by combining distinct sources of data, we first evaluate the use of distinct existing networks for single-view data. Then, we evaluate the use of early and late fusion to perform multiview classification. All evaluated techniques are described next.

Fig. 4. Examples of instances taken from CV-BrCT.

A. Deep Architectures

Convolutional neural networks (ConvNets) [20] have become the standard state-of-the-art technique for visual recognition over the past decade. Their capability to provide end-to-end feature learning and inference turns them into powerful statistical models for computer vision applications, including scene classification. Supported by this, we evaluated several ConvNet-based approaches for multiview image classification using the proposed datasets. All experimented techniques are described next.

AlexNet [20]: The first network evaluated is the AlexNet one. Originally proposed for and winner of the ILSVRC 2012 competition, this pioneer ConvNet is composed of five convolutional layers, some of which are followed by max-pooling layers, and three fully connected layers with a final softmax. The first convolutional layers use large convolutional filters in order to quickly reduce the spatial resolution of the input image. Fig. 5(a) presents the architecture of AlexNet network.

VGG [21]: This work was the first one to observe that smaller sequential convolutional filters had the representation capabilities of one single large trainable convolutional kernel. Supported by this, the authors deepened the network, which has eight 3×3 convolutional layers, five pooling ones, and four fully connected ones (considering the softmax). Fig. 5(b) illustrates the architecture of VGG-11 network.

Inception [22], [23]: Following the same guidelines of the VGG network [21], this architecture employed more convolutional layers in order to increase the feature extraction ability. Specifically, this network is based on the "Inception" modules that exploit feature diversity through parallel convolutions with different filter sizes. This module is replicated several times producing the final architecture that has 48 layers. Through Fig. 5(c), it is possible to see how an inception-v3 architecture and inception modules work.

ResNet [24]: This work was the first one to notice that adding even more layers to the architecture only worsened the vanishing gradient problem. So, to mitigate this problem, the authors employed shortcut connections to allow the efficient training of earlier layers in the ConvNet. Based on this concept, several networks were proposed, some of them with hundreds or even thousand convolutional layers. In this work, ResNet-18 [24], which has 18 convolutional layers with adding shortcuts, was evaluated. Fig. 5(d) shows how a ResNet-18 architecture is built.

DenseNet [25]: Following the same idea of the ResNets [21], this architecture employed shortcut connections in order to allow the gradients to easily flow and better optimize the initial layers. The difference between ResNets [21] and DenseNet [24] is that in the former one, the shortcuts add the inputs, whereas in the latter one, the input layers are concatenated in the shortcut connections. Again, due to this shortcut design, dense architectures, with hundreds or even thousand convolutional layers, were proposed and employed in several applications [24]. In this work, DenseNet-169 [25], which has 169 convolutional layers with shortcuts, was evaluated. The architecture of this model is presented in Fig. 5(e).

SqueezeNets [26]: This network uses a combination of pruning, compression techniques, and fire modules composed of squeeze and expand convolutions in order to create a lean and efficient architecture that can be incorporated into devices with limited memory (such as mobile). In fact, SqueezeNets are able to achieve visual recognition objective scores close to early ConvNet architectures (as AlexNet [20]) with between one or two orders of magnitude fewer parameters. Fig. 5(f) illustrates how a fire module works and how they are integrated with a SqueezeNet architecture.

Squeeze and Excitation Networks [27]: Instead of focusing on spatial components to enhance feature extraction results, this work focuses on the relationship between channels. For this task, the authors propose a new block named "Squeeze and Excitation block." This block operates recalibrating channelwise feature impacts by modeling interdependencies between those channels in an explicit way. In this work, the authors also show that using these blocks, networks can outperform previously state-of-the-art results on the ImageNet dataset [29] and that the use of this block can be easily adapted to existing architectures. Fig. 5(g) shows how a "Squeeze and Excitation block" works and how it can be implemented in a ResNet-50 architecture.

Selective Kernels Networks [28]: Most of the designed ConvNets use receptive fields of the same size in each one of its layers. In this work, the authors propose an attention block named "Selective Kernel unit." The main objective of this block is to allow each neuron to adaptively adjust the size of its receptive field, looking at different scales of input information. Relating to the functioning of this block, it is based on a fusion of kernels that have different sizes using a softmax attention, guided by the input information that enters the block. In this

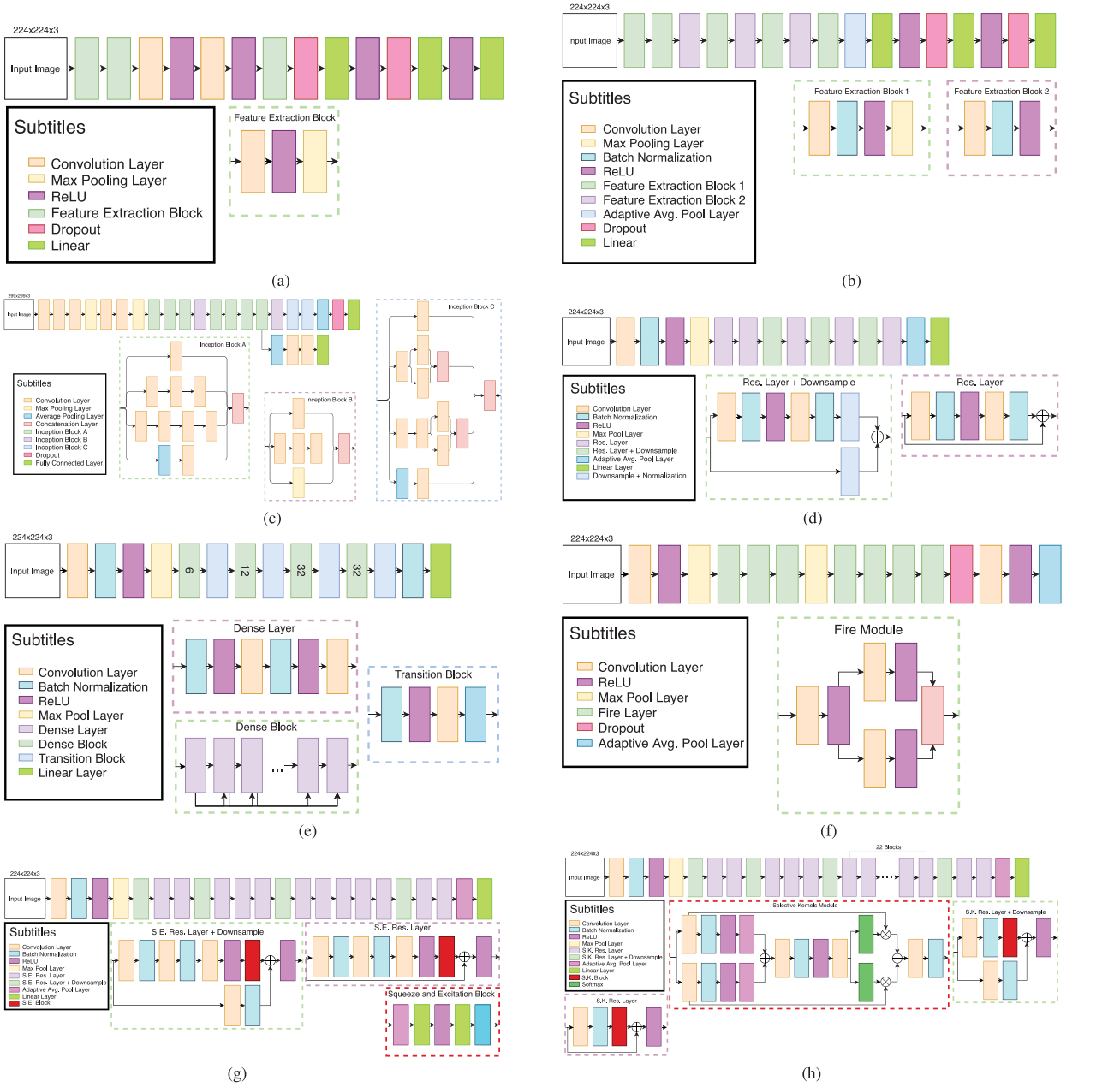


Fig. 5. Benchmarked architectures. (a) AlexNet [20] architecture. (b) VGG-11 [21] architecture. (c) Inception-V3 [23] architecture. (d) ResNet-18 [24] architecture. (e) DenseNet-169 [25] architecture. (f) SqueezeNet [26] architecture. (g) Squeeze and Excitation ResNet-50 [27] architecture. (h) Selective Kernels ResNet-101 [28] architecture.

work, the authors also show that using these blocks on a ResNet [24] can outperform previously state-of-the-art results on the ImageNet dataset [29]. Through Fig. 5(g), it is possible to see how a "Selective Kernel unit" operates and how it was integrated into a ResNet-101 architecture.

B. Fusion Methods

To enhance scene classification results, we evaluated several models for early and late fusion in order compare both approaches. In this work, those techniques were applied to fuse aerial/ground/satellite features, acquiring new information, and using them to enhance the final predictions. In the sections ahead, we will describe all such techniques.

1) Early Fusion Methods: In order to exploit the correlations and interactions between low-level features from different modalities [30], we propose an early fusion approach based on the deep architectures used for the experiments. A great advantage of early fusion approaches is that they require the training of a single model, which usually results in compacted models compared to the late fusion ones.

The early fusion strategy performed in this work consists of using the first feature extraction layers of the target network as a backbone. This backbone is replicated to aerial and ground

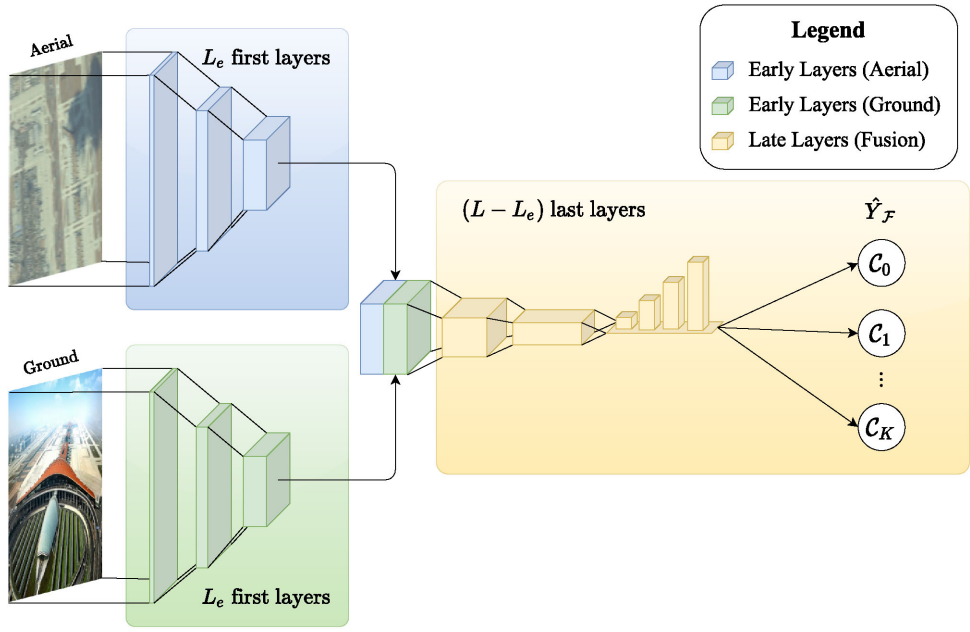


Fig. 6. Example of the proposed early fusion methodology.

images. The fusion of the features is made by applying a concatenation layer on the low-level features, which results in a tensor that contains the double amounts of kernels than the original ones. The choice of where those concatenations were performed is based on the total number of kernels that each convolution layer have. So, in order, to be possible, to fully explore pretrained models, we decided to concatenate those feature vectors before the first convolution layer that doubles its amount of kernels in the target network. In this way, we ignore the convolution that duplicates this amount of kernels and substitute it to a fusion that also duplicates the amount of feature vectors. Fig. 6 represents the early fusion methodology proposed for this work. The first L_e layers (blue and green blocks in the figure) represent the early feature extraction process, which is made individually for each view. After a few amount of layers, the features are concatenated and transported to the remaining of the architecture, which was used as a base (yellow block in the figure). Finally, after the high-level features are extracted, the classification process is performed.

2) Late Fusion Methods: Late fusion or decision-based algorithms perform integration of results after each of the modalities has made a prediction [30]. Those algorithms use unimodal decision values and combine them using different types of fusion mechanisms, such as averaging, voting schemes, or weighting based. Fig. 7 presents a typical late fusion procedure exploited in this work.

To formally define all the fusion operations used for this work, for all definitions of this section, we will use the following notation. Let σ_i be the softmax scores returned by the network i, α_i be the accuracy score that the network i achieved on the validation set of each dataset, and m be the number of networks used to perform a fusion operation.

Sum: The sum fusion is a well-known late fusion algorithm. The main idea of it is to sum all the vectors (softmax scores) and select the index, which contains the maximum value of this sum as the prediction. This procedure is formally defined by

$$\text{Sum}_{\text{Prediction}} = \arg\max \sum_{i=1}^{m} \sigma_i. \tag{1}$$

Majority Voting: The majority voting fusion is another well-known late fusion method in the literature. This method is based on the concept of a democratic election, i.e., each model act as a voter and provides its output as a vote, the final prediction is selected as the returned value with more votes. To mathematically express this procedure, it was used as a mode operation, that is, a statistic that indicates the most common element contained in a vector. The majority voting procedure is properly defined by

$$\text{MV}_{\text{Prediction}} = \text{mode}\ \arg\max\sigma_i \quad \forall i \in [1, m]. \tag{2}$$

In this work, majority voting was used to perform late fusion between two or three models, so ties constantly happen. To solve this issue, it was used the confidence (probability value) that each model has on its answer. In this way, when a vote ties, we select the output with the biggest confidence between all models' outputs. The same process was used for fusions using only two models since there is no point in checking which is the most common vote between two voters. The procedure used for tiebreaker and voting using only two models is formally defined by

$$\begin{aligned} &\text{MV}_{\text{Prediction}} = \beta_\theta, \text{where} \\ &\beta_i = \arg\max\sigma_i \\ &\theta = \arg\max \max \sigma_i \quad \forall i \in [1, m]. \end{aligned} \tag{3}$$

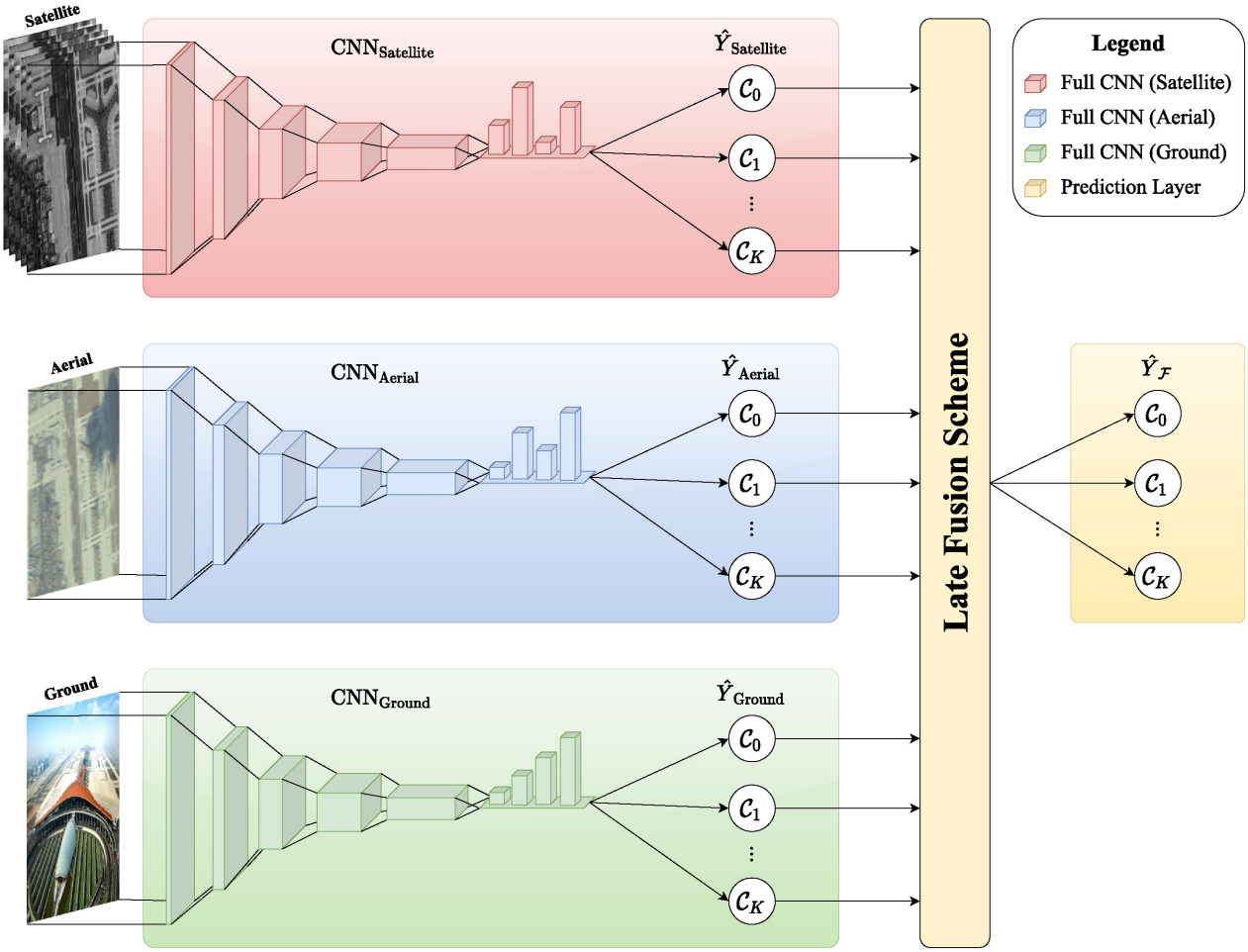


Fig. 7. Typical late fusion pipeline. As can be seen, each ConvNet is trained individually and their outputs are combined using a late fusion algorithm, resulting in the final prediction.

Weighted Sum: As can be noted by its name, this method operates in a similar way that sum fusion does. The main difference between them is that the weighted sum multiplies values (weights) while it is performing a sum operation. This procedure is useful in situations that different classifiers have very distinct results. So, in this case, the weighted sum can use values for trying to calibrate this huge variance between the models' results. The formal definition of weighted sum can be checked in the following equation, in which the weights used for the experiments on this work were taken from the individual performance score (accuracy) of each model on the validation set of each dataset:

$$\text{WSum}_{\text{Prediction}} = \arg\max \sum_{i=1}^{i=m} \alpha_i \sigma_i. \tag{4}$$

Minimum: The main advantage of the minimum fusion method is that the algorithm can eliminate possible overfitting that may have occurred during the training phase. The first step of the method is to select the individual prediction of each model (the index that contains the maximum value on softmax scores vector). After that, the method looks for the scores associated to each one of the returned indexes and returns as the final prediction the index that has the smallest score associated with it. Following equation formally defines the described procedure:

$$\begin{aligned} &\text{Min}_{\text{Prediction}} = \beta_\theta, \text{where} \\ &\beta_i = \arg\max \sigma_i \\ &\theta = \arg\min \max \sigma_i \quad \forall i \in [1, m]. \end{aligned} \tag{5}$$

Product: The product fusion is a very used late fusion algorithm. The main idea of it is to perform an elementwise multiplication between softmax scores, and after that return the index that contains the biggest value. This procedure is defined by

$$\text{Prod}_{\text{Prediction}} = \arg\max \prod_{i=1}^{m} \sigma_i. \tag{6}$$

V. Experimental Setup

In this section, we describe the experimental setup used for the experiments using both datasets. It is important to mention that all the methods, previously described in Section IV, were used for the experiments, and in all of those experiments, a fivefold

cross-validation protocol was used to properly evaluate each technique. We reported the mean of balanced accuracy and/or F1-score, taken from all fivefold experiments with its correspondent standard deviation. Finally, in Section V-A, we present the protocol used to train the models using AiRound dataset, whereas in Section V-B, we detail the methodology used for the CV-BrCT dataset models.

A. AiRound

Since all the networks used for this work are well known in the literature, it is possible to find pretrained models of them. So, in order to allow a better comparison and understanding of the most suitable training strategy for AiRound, we trained all the models from scratch and fine-tuned. For all the experiments made on AiRound, each model was trained for 200 epochs, using early stop technique with 20 epochs checking for improvements in validation. Relating to the other hyperparameters, it was used a batch size of 32, stochastic gradient descent as optimizer, a learning rate of 0.001, and a momentum of 0.9. Finally, about the data augmentation techniques applied, it was performed the randomized crop and random horizon flip.

Relating to the models evaluated, for late fusion, we trained an individual network for each kind of data, as was shown in Fig. 7. For a better comparison, we evaluated the combinations of the models using all the late fusion algorithms previously described. In order to test all possible combinations of fusions between different views, each late fusion algorithm fuses outputs of two or three networks, trained in different views, by combining alternated models' outputs, e.g., three-view perspective, aerial with ground, etc. All the combinations were made using models with the same architecture and trained using data from only one-view perspective.

For the early fusion models, an end-to-end training was performed, using aerial and ground paired data as inputs. All the training processes were also made using the same combination of hyperparameters used to train the one-view individual models.

B. CV-BrCT

For the second dataset, we used a very similar protocol than the one previously described. The main differences between them are that we trained each model for 100 epochs, instead of 200, and we used an early stop with 10 epochs. This decision was made because the CV-BrCT has way more samples than AiRound and the models tended to converge faster. Naturally, for this dataset, we could not perform the same set of late fusions, because it does not have Sentinel-2 data, so we performed the late fusions only using aerial and ground data.

VI. Results and Discussion

In this section, we present and discuss the obtained results. The results for AiRound dataset are presented in Section VI-A. Sections VI-A1 and VI-A2 present the results achieved for training networks using only one-view type and applying fusion techniques, respectively. Relating to CV-BrCT dataset, the results can be found in Section VI-B. Following the same organization used for AiRound, we present the results of the models trained using one-view in Section VI-B1, whereas the results of the models using fusion techniques can be found in Section VI-B2.

TABLE II
Results in Terms of F1 Score of the Evaluated Models for AiRound Dataset

Training Strategy	Network	Input Data		
		Aerial	Ground	Sentinel-2
Training from scratch	AlexNet [20]	0.70 ± 0.02	0.63 ± 0.02	0.60 ± 0.02
	VGG [21]	0.69 ± 0.10	$\mathbf{0.66 \pm 0.03}$	$\mathbf{0.64 \pm 0.02}$
	Inception [23]	0.70 ± 0.02	0.63 ± 0.02	0.62 ± 0.01
	ResNet [24]	$\mathbf{0.72 \pm 0.01}$	0.64 ± 0.02	0.54 ± 0.02
	DenseNet [25]	0.68 ± 0.02	0.61 ± 0.02	0.62 ± 0.01
	SqueezeNet [26]	0.62 ± 0.04	0.61 ± 0.02	0.55 ± 0.03
	SENet [27]	0.69 ± 0.01	0.62 ± 0.02	0.54 ± 0.02
	SKNet [28]	0.67 ± 0.03	0.61 ± 0.01	0.54 ± 0.02
Fine tuning from ImageNet	AlexNet [20]	0.76 ± 0.01	0.70 ± 0.01	-
	VGG [21]	0.81 ± 0.01	0.76 ± 0.01	-
	Inception [23]	$\mathbf{0.84 \pm 0.01}$	$\mathbf{0.78 \pm 0.01}$	-
	ResNet [24]	0.80 ± 0.00	0.75 ± 0.01	-
	DenseNet [25]	$\mathbf{0.84 \pm 0.01}$	0.77 ± 0.02	-
	SqueezeNet [26]	0.83 ± 0.01	0.76 ± 0.01	-
	SENet [27]	$\mathbf{0.84 \pm 0.01}$	0.76 ± 0.01	-
	SKNet [28]	$\mathbf{0.84 \pm 0.01}$	0.77 ± 0.01	-

Bold values indicates the best results achieved by each type of data and/or training strategy.

A. Experiments on the AiRound

1) Networks Architectures Comparison: Here, we present the results obtained from the deep-learning-based models trained individually, for each view, from scratch and fine-tuned (from the ImageNet dataset [29]). As introduced, the objective is to analyze and define the most suitable network and training strategy for AiRound dataset. All obtained results are presented in Table II. It is important to highlight that we did not fine-tune the networks for Sentinel-2 images, given the incompatibility between the number of bands of these data and the number of input channels expected by the networks, i.e., given that Sentinel-2 data have 13 channels, and the first convolution layer of the evaluated architectures receive only 3 input bands (RGB).

Analyzing the results, it is possible to observe that, as has been seen in the literature, fine-tuned networks produced better outcomes than their counterpart models trained from scratch [31]. Another interesting aspect is that the models trained with the same data (mainly aerial and ground) yielded very similar results, without one model outperforming others, except for SqueezeNet [26] trained from scratch using aerial data and fine-tuned AlexNet [20] using ground data. For the Sentinel-2 models, AlexNet [20], VGG [21], Inception [23], and DenseNet [25] achieved slightly better results comparing to the other networks. Furthermore, considering the distinct input data, one may note from the experiments that aerial images tend to produce better outcomes, whereas ground and Sentinel-2 data tend to yield worse results. The differences between aerial and Sentinel-2 results may be justified by the difference in the spatial resolution of the images since Sentinel-2 data have a resolution in meters per pixel whereas the aerial can have a spatial resolution in centimeters per pixel. Relating to the ground images, we believe that it achieved slightly worse results comparing to aerial, due

TABLE III
RESULTS OF THE EVALUATED EARLY FUSION NETWORKS FOR AIROUND DATASET

Early Fusion Networks	Training Strategy			
	From Scratch		Fine Tuning	
	B. Acc.	F1 Score	B. Acc.	F1 Score
AlexNet [20]	0.80 ± 0.02	0.79 ± 0.02	0.81 ± 0.02	0.80 ± 0.02
VGG [21]	$\mathbf{0.82 \pm 0.02}$	$\mathbf{0.82 \pm 0.02}$	0.84 ± 0.02	0.84 ± 0.02
Inception [23]	0.77 ± 0.01	0.76 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
ResNet [24]	0.76 ± 0.01	0.76 ± 0.01	0.83 ± 0.02	0.83 ± 0.02
DenseNet [25]	0.74 ± 0.01	0.74 ± 0.01	0.83 ± 0.01	0.83 ± 0.01
SqueezeNet [26]	0.66 ± 0.03	0.65 ± 0.03	0.78 ± 0.02	0.77 ± 0.02
SENet [27]	0.73 ± 0.02	0.73 ± 0.02	0.84 ± 0.02	0.83 ± 0.01
SKNet [28]	0.73 ± 0.01	0.72 ± 0.01	$\mathbf{0.86 \pm 0.02}$	$\mathbf{0.86 \pm 0.02}$

Bold values indicates the best results achieved by each type of data and/or training strategy.

TABLE IV
BENCHMARKED METHODS PROPERTIES FOR AIROUND DATASET

Fusion Type	Network	GPU train time (in seconds)	Total Parameters (in milions)
Early Fusion	**AlexNet [20]**	139.75	56.72
	VGG [21]	271.30	128.60
	Inception [23]	2295.35	24.37
	ResNet [24]	333.52	10.81
	DenseNet [25]	2868.82	12.42
	SqueezeNet [26]	517.65	0.73
	SENet [27]	2412.17	24.90
	SKNet [28]	9459.28	42.60
Late Fusion	**AlexNet [20]**	155.89	114.09
	VGG [21]	453.69	268.62
	Inception [23]	4711.90	48.84
	ResNet [24]	513.53	22.36
	DenseNet [25]	5132.37	27.20
	SqueezeNet [26]	732.51	1.48
	SENet [27]	4259.64	52.08
	SKNet [28]	15111.72	87.30

Note: It is important to mention that all the times were calculated using RTX2080TI and it was accounted only forward and backward time during the training phase using the strategy of training from scratch.

to the lack of information that help discriminate between some classes, for instance, the classes lake and river or forest and park. Finally, as can be noted in Table II, there was not a single-network architecture that yielded to the best results using any training strategy/data. Besides that, we can highlight between the models trained from scratch that VGG [21] yielded the best results using ground and sentinel data. We also highlight that between the fine-tuned models, the inception [23] yielded the best results using both data types.

2) Multiview Fusion Strategies: This section presents and discusses the results obtained applying early and late fusion techniques.

AiRound Early Fusion: For early fusion experiments, we followed the scheme described in Section IV-B1 using all the eight architectures. In Table III, comparing the same architecture models, it is notable that, for both training strategies, most of the results tended to achieve a superior mark compared to the one-view results reported in Table II. We also highlight that some of the results are statistically equivalent, for instance, the fine-tuned inception models for aerial and early fusion.

Analyzing the main fusion gains, for the models trained from scratch, we can highlight the VGG [21] model, which achieve the best results, and the biggest gain in F1 score (0.16) comparing to the one-view experiments, previously reported. For the fine-tuned models, we can highlight the AlexNet [28] adaption for early fusion, which obtained gains of 0.04 and 0.10 in F1 Score, comparing to the same networks trained using only aerial and ground data, respectively. It is also notable by Table IV that the early fusion adaptation of AlexNet [20] also converged faster than the other network models. The selective kernels networks [28] was the model that took the most time to converge, but it also achieved the best overall result between all early fusion networks, using the fine-tuning strategy.

Finally, for some results, it is notable that a downgrade occurred, if compared to the one-view results presented in Table II. For those results, we hypothesize that the same feature degradation phenomenon,[8] which was reported in [5], occurred.

AiRound Late Fusion: For all models trained from scratch, we evaluated all 4 possible combinations of views for all 8 networks. Since we performed 5 different types of fusions and it was trained models from scratch using 3 different types of data (aerial, ground, and Sentinel-2), all the combinations would result in 184 experiment results. Given that high number of experiments and that, as previously discussed, all the network architectures produced similar results and anyone could be selected for further experiments, we reported the obtained results for only the VGG [21].

As can be seen in Fig. 8, most of the late fusion techniques outperformed the models trained with only one view, with a special highlight for the three-view and aerial–ground fusions. This result can be explained due to the amount of complementary information that exists between aerial and ground images. On the other hand, based on all experiments, it is possible to observe that the combination of aerial and Sentinel-2 images tends to not statistically improve the results. This is probably due to the amount of similar information that both types of images have in common, since both are from the same view perspective.

Finally, considering the ground-Sentinel-2 fusions, it is possible to notice a little improvement, which can be justified by the same reason of aerial-Sentinel-2 fusions. We conclude that the gain in this fusion was not as good as aerial–ground one, because of the limited spatial-resolution that Sentinel-2 satellite offers (10×10 m or 20×20 m or 60×60 m per pixel, depending on the channel).

For the fine-tuned models, the late fusion results are presented in Table V. In this case, all results are reported, given that Sentinel-2 images could not be exploited and only one combination could be performed (aerial and ground). As expected, comparing the VGG results reported in Fig. 8, there was a significant gain, due to the fine tuning process. Those gains occurred for all the architectures that were used for the experiments. We also highlight that the product fusion were the technique that yielded to the best results for both training strategies.

Comparing the results using only one type of data (see Table II) with the fusion outcomes, it is possible to notice that the

[8] A destructive effect that occurs in training phase due to the misalignment of the geometry of the bottleneck features of the two image types.

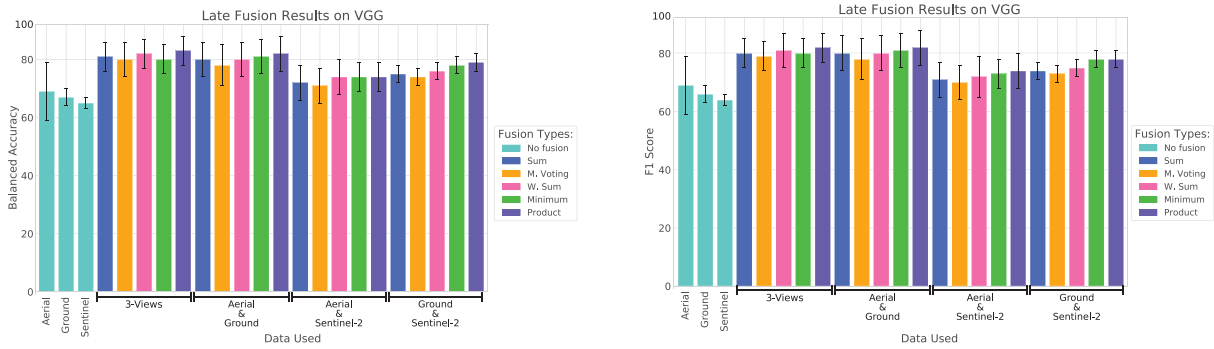


Fig. 8. Results comparison of all fusion types using VGG trained from scratch. (a) Results comparison (in terms of balanced accuracy) of all fusion types using VGG trained from scratch. (b) Results comparison (in terms of f1-score) of all fusion types using VGG trained from scratch.

TABLE V
RESULTS OF THE EVALUATED LATE FUSION TECHNIQUES FOR AIROUND DATASET USING FINE-TUNED MODELS

Network	Fusion Strategy									
	Sum		M. Voting		W. Sum		Minimum		Product†	
	B. Acc.	F1 Score	B. Acc.	F1 Score	B. Acc.	F1 Score	B. Acc.	F1 Score	B. Acc.	F1 Score
AlexNet [20]	0.84 ± 0.00	0.84 ± 0.00	0.83 ± 0.01	0.83 ± 0.01	0.83 ± 0.01	0.83 ± 0.01	0.85 ± 0.01	0.85 ± 0.01	0.86 ± 0.01	0.86 ± 0.01
VGG [21]	0.88 ± 0.01	0.88 ± 0.01	0.88 ± 0.01	0.87 ± 0.01	$\mathbf{0.88 \pm 0.01}$	0.87 ± 0.01	0.89 ± 0.01	0.89 ± 0.01	0.90 ± 0.00	0.90 ± 0.00
Inception [23]	0.88 ± 0.01	0.88 ± 0.01	0.88 ± 0.01	0.87 ± 0.01	$\mathbf{0.88 \pm 0.01}$	0.87 ± 0.01	0.89 ± 0.01	0.89 ± 0.01	0.90 ± 0.00	0.90 ± 0.00
ResNet [24]	0.88 ± 0.01	0.87 ± 0.01	0.87 ± 0.01	0.87 ± 0.01	0.86 ± 0.02	0.86 ± 0.02	0.88 ± 0.01	0.88 ± 0.01	0.89 ± 0.01	0.89 ± 0.01
DenseNet† [25]	$\mathbf{0.90 \pm 0.01}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.90 \pm 0.01}$	$\mathbf{0.90 \pm 0.01}$	$\mathbf{0.91 \pm 0.01}$	$\mathbf{0.91 \pm 0.01}$
SqueezeNet [26]	0.85 ± 0.01	0.85 ± 0.01	0.85 ± 0.01	0.85 ± 0.01	0.85 ± 0.01	0.84 ± 0.01	0.86 ± 0.01	0.86 ± 0.01	0.87 ± 0.01	0.86 ± 0.01
SENet [27]	0.89 ± 0.02	$\mathbf{0.89 \pm 0.02}$	0.88 ± 0.02	0.88 ± 0.02	0.87 ± 0.01	0.87 ± 0.01	$\mathbf{0.90 \pm 0.01}$	0.89 ± 0.01	0.90 ± 0.01	0.90 ± 0.01
SKNet [28]	$\mathbf{0.90 \pm 0.01}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.90 \pm 0.01}$	$\mathbf{0.90 \pm 0.00}$	0.90 ± 0.01	0.90 ± 0.01

The † Symbols mark the best overall network and fusion strategy.
Bold values indicates the best results achieved by each type of data and/or training strategy.

late fusion outperformed all approaches using only one view. This corroborates with our initial analysis that the combination of multisource data could improve the results for the scene classification task. In Table V, the best overall results were achieved by the DenseNet [25] architecture.

In order to better understand how the fusion methods are able to improve the results, we performed an analysis, per class, of such techniques for all eight architectures. However, again, because of the same aforementioned reason, only results for the VGG architecture [21] were reported. We also compared the results using only aerial and ground data, since the fusions using Sentinel-2 data did not lead to significant improvements.

Fig. 10(a) reports the fusion improvements per class. As can be seen in the figure, this process was performed individually (per view and class) using models trained from scratch and fine-tuned. This is because the purpose of this heat map is to see which classes benefit the most from each fusion in each type of view. Again, we also did not use the Sentinel-2 results in the figure, for the same aforementioned reasons. Through Fig. 10(a), it is possible to observe that, for the model trained from scratch using aerial data, the classes tower, bridge, and statue were the ones that most benefited from the aerial/ground fusion. This happened because most of them are hard to classify using only aerial images since those structures naturally have high heights and occupy a restricted area, which are characteristics that are not well explored in an aerial perspective. For the class bridge, we believe that this happened because the class also contemplates viaducts, which are harder to classify using only aerial perspective, since in some examples it is hard to note the height of the viaduct. In those cases, it is understandable to misclassify between this class and airport, due to the similarity between roads and airstrips. For the fine-tuned models, a similar phenomenon repeated, but mostly for the class tower, which reinforces that the information that came from the ground perspective helps in discriminating this class.

Concerning the ground images, for models trained with both strategies, the classes lake, river, forest, and park were the ones that most improved from the aerial/ground fusion. The main reason for this is that the context around those classes may help a lot in discriminating them. Specifically, parks are naturally located in cities, and therefore, the information about the existence of a city nearby, which comes from aerial images, may help in its distinction between forests. Furthermore, classes river and lake are quite similar, since both represent water bodies. Thus, both classes may benefit from the information that aerial data provides about the shape and the area (such as the surrounding vegetation), which may help in these two classes' discrimination.

In Fig. 9(a), there are some examples of predictions made by each classifier. For most of the cases in which there was a misclassification, it was understandable, due to the similar characteristics between the prediction and the labeled classes, as can be seen for the bridge, stadium, and river samples. The image also shows a sample that is hard to classify (statue). This sample has characteristics that differ from most of the statues from the dataset, and besides that, there is a vegetation area nearby, which caused the wrong predictions to the classes park and forest.

Remark: Comparing the results between early and late fusion, we can see a clear advantage of late fusion in most of the cases. However, in some situations, early fusion achieved similar results, as can be seen comparing the results for fine-tuned

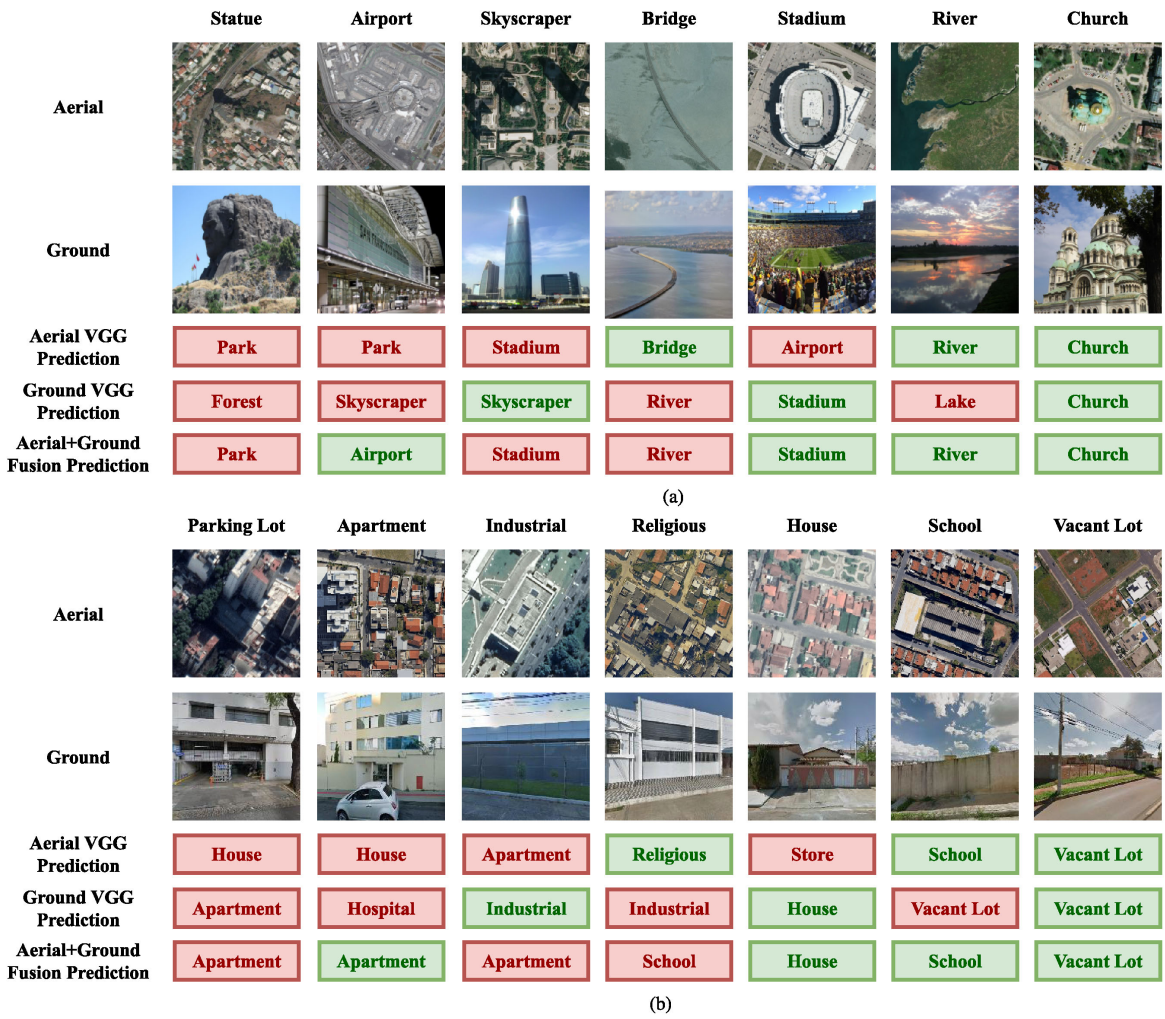


Fig. 9. Visual examples of classifications made in the proposed datasets. For this image, we used the predictions made by VGGs [21] trained from scratch, and it was selected one of each possible case that can happen on the fusion process. The samples in the figure are sorted from the pair with most wrong predictions to the one with most correct classifications. Since it is a decision-based fusion, naturally, there was not any case in which both aerial and ground VGGs have a correct prediction and its fusion does not. Finally, for the fusion prediction, it was used the product fusion. (a) AiRound dataset. (b) CV-BrCT dataset.

selective kernels networks [28], for instance. Finally, in Table VIII, we demonstrated the aforementioned advantages of early fusion. For all cases, this technique converged faster than late fusion, and it also used less parameters.

B. Experiments on the CV-BrCT

1) Network Architecture Comparison: We replicate the same experiments realized in the AiRound dataset in the CV-BrCT dataset in regards to its two image types—aerial and ground (frontal). The results for the experiments using a single type of image are presented in Table VI.

As seen in Table VI, the best training protocol is again to fine-tune the networks. For all architectures, the fine-tuned models have better performance for both types of images, a result similar to the AiRound dataset experiments. Different from the other experiment, in the CV-BrCT dataset the networks tend to perform better with the aerial images than with the ground images, whereas in the AiRound dataset, these results were closer. We argue that some classes have a visual similarity in the ground images, e.g., hospitals and schools can have similar facades, prevalence of stores in first floors of buildings, etc. Thus, the discrepancy occurs between results of different image types.

With a few exceptions, the networks have comparable results and four achieved practically the same metric values.

TABLE VI
RESULTS OF THE EVALUATED MODELS FOR CV-BRCT DATASET

Training Strategy	Network	Input Data			
		Aerial		Ground	
		B. Acc.	F1 Score	B. Acc.	F1 Score
Training from scratch	AlexNet [20]	0.68 ± 0.03	0.79 ± 0.02	0.50 ± 0.03	0.62 ± 0.01
	VGG [21]	0.70 ± 0.04	$\mathbf{0.81 \pm 0.03}$	$\mathbf{0.54 \pm 0.02}$	$\mathbf{0.66 \pm 0.01}$
	Inception [23]	0.69 ± 0.03	0.80 ± 0.02	0.49 ± 0.03	0.62 ± 0.02
	ResNet [24]	0.68 ± 0.07	0.79 ± 0.03	0.50 ± 0.05	0.63 ± 0.03
	DenseNet [25]	$\mathbf{0.71 \pm 0.02}$	$\mathbf{0.81 \pm 0.01}$	0.49 ± 0.01	0.62 ± 0.01
	SqueezeNet [26]	0.55 ± 0.07	0.70 ± 0.05	0.41 ± 0.08	0.56 ± 0.06
	SENet [27]	0.69 ± 0.04	0.80 ± 0.02	0.49 ± 0.02	0.62 ± 0.02
	SKNet [28]	0.68 ± 0.06	0.79 ± 0.04	0.47 ± 0.03	0.61 ± 0.02
Fine Tuning from ImageNet	AlexNet [20]	0.75 ± 0.02	0.84 ± 0.01	0.54 ± 0.01	0.66 ± 0.01
	VGG [21]	0.79 ± 0.03	0.87 ± 0.01	$\mathbf{0.60 \pm 0.02}$	$\mathbf{0.71 \pm 0.01}$
	Inception [23]	$\mathbf{0.80 \pm 0.02}$	0.87 ± 0.00	$\mathbf{0.60 \pm 0.03}$	$\mathbf{0.71 \pm 0.01}$
	ResNet [24]	0.78 ± 0.02	0.86 ± 0.01	0.58 ± 0.04	0.69 ± 0.02
	DenseNet [25]	$\mathbf{0.80 \pm 0.02}$	0.87 ± 0.01	$\mathbf{0.60 \pm 0.01}$	$\mathbf{0.71 \pm 0.01}$
	SqueezeNet [26]	0.70 ± 0.02	0.80 ± 0.01	0.56 ± 0.02	0.68 ± 0.01
	SENet [27]	$\mathbf{0.80 \pm 0.02}$	0.87 ± 0.01	$\mathbf{0.60 \pm 0.01}$	$\mathbf{0.71 \pm 0.01}$
	SKNet [28]	$\mathbf{0.80 \pm 0.03}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.60 \pm 0.02}$	$\mathbf{0.71 \pm 0.01}$

Bold values indicates the best results achieved by each type of data and/or training strategy.

TABLE VII
RESULTS OF THE EVALUATED EARLY FUSION NETWORKS FOR CV-BRCT DATASET

Early Fusion Networks	Training Strategy			
	From Scratch		Fine Tuning	
	B. Acc.	F1 Score	B. Acc.	F1 Score
AlexNet [20]	0.69 ± 0.03	0.8 ± 0.01	0.72 ± 0.02	0.82 ± 0.01
VGG [21]	$\mathbf{0.73 \pm 0.03}$	0.82 ± 0.01	0.76 ± 0.02	0.84 ± 0.02
Inception [23]	$\mathbf{0.73 \pm 0.04}$	$\mathbf{0.83 \pm 0.02}$	0.79 ± 0.03	$\mathbf{0.87 \pm 0.01}$
ResNet [24]	0.68 ± 0.02	0.79 ± 0.01	0.74 ± 0.02	0.83 ± 0.01
DenseNet [25]	0.71 ± 0.04	0.80 ± 0.02	0.72 ± 0.03	0.81 ± 0.01
SqueezeNet [26]	0.60 ± 0.01	0.73 ± 0.02	0.67 ± 0.04	0.79 ± 0.02
SENet [27]	0.67 ± 0.04	0.78 ± 0.02	0.78 ± 0.02	0.86 ± 0.01
SKNet [28]	0.70 ± 0.04	0.80 ± 0.03	$\mathbf{0.80 \pm 0.02}$	$\mathbf{0.87 \pm 0.01}$

Bold values indicates the best results achieved by each type of data and/or training strategy.

2) Fusions: In this next section, we present the results for the fusion methods in the CV-BrCT dataset.

CV-BrCT Early Fusion: The early fusion architectures proposed were evaluated with pretrained weights and initially randomized weights. The results are presented in Table VII. As in the experiments, the fine-tuned models outperform the non-pretrained ones. With respect to the single-type networks, these early fusion architectures seem to perform slightly better than the trained from scratch with one type of image, while performing the same, or slightly worse than the fine-tuned models using aerial images.

As noted in the previous experiment, the results of networks using only ground images were worse than the aerial image models. Through the experiments, we noted that using a network that merges and combines features of both images from the start leads to no improvements, in the fine-tune scenario. This issue can also be justified by the same feature degradation phenomenon, aforementioned. However, when the networks are trained from scratch, it seems to learn how to better extract and combine features of both images, which yields a slightly superior performance of these models, comparing to the results previously reported in Table VI. For these experiments, we also measured the estimated training time of each network model. Those times can be checked in Table VIII. As can be noted, again, the AlexNet [20] was the model that converged faster, and the selective kernels network [28] was the one that took the most time to train.

CV-BrCT Late Fusion: We tested the fusion of the two image types in all the five methods discussed in Section IV-B2. All the results are show in Table IX.

Overall, all fusion methods improved the results of the networks trained with a single type, in both the initially randomized and fine-tuned cases. The results across fusion methods are similar, although some techniques show a consistent improvement, e.g., weighted sum, and other do not appears to have a noticeable effect, e.g., minimum.

As the networks trained with only ground images are less reliable classifiers, i.e., they have achieved worse results than the aerial models, the score each one assign to a sample is smaller than the aerial model. Henceforth, the impact these classifiers have in the final prediction, regardless of the fusion method, is less significant; thus, the improvement exists but is relatively small. Besides that, the best overall results from the models trained from scratch were achieved by DenseNet [25], but it was also one of the models that took the longest to converge, losing only to selective kernels networks [28], as can be seen in Table VIII. Regarding the best results using pre-trained models, this was achieved by selective kernels networks. Finally, comparing the late fusion algorithms, it is notable that the product fusion usually resulted in the best enhancements.

TABLE VIII
BENCHMARKED METHODS PROPERTIES FOR CV-BRCT DATASET

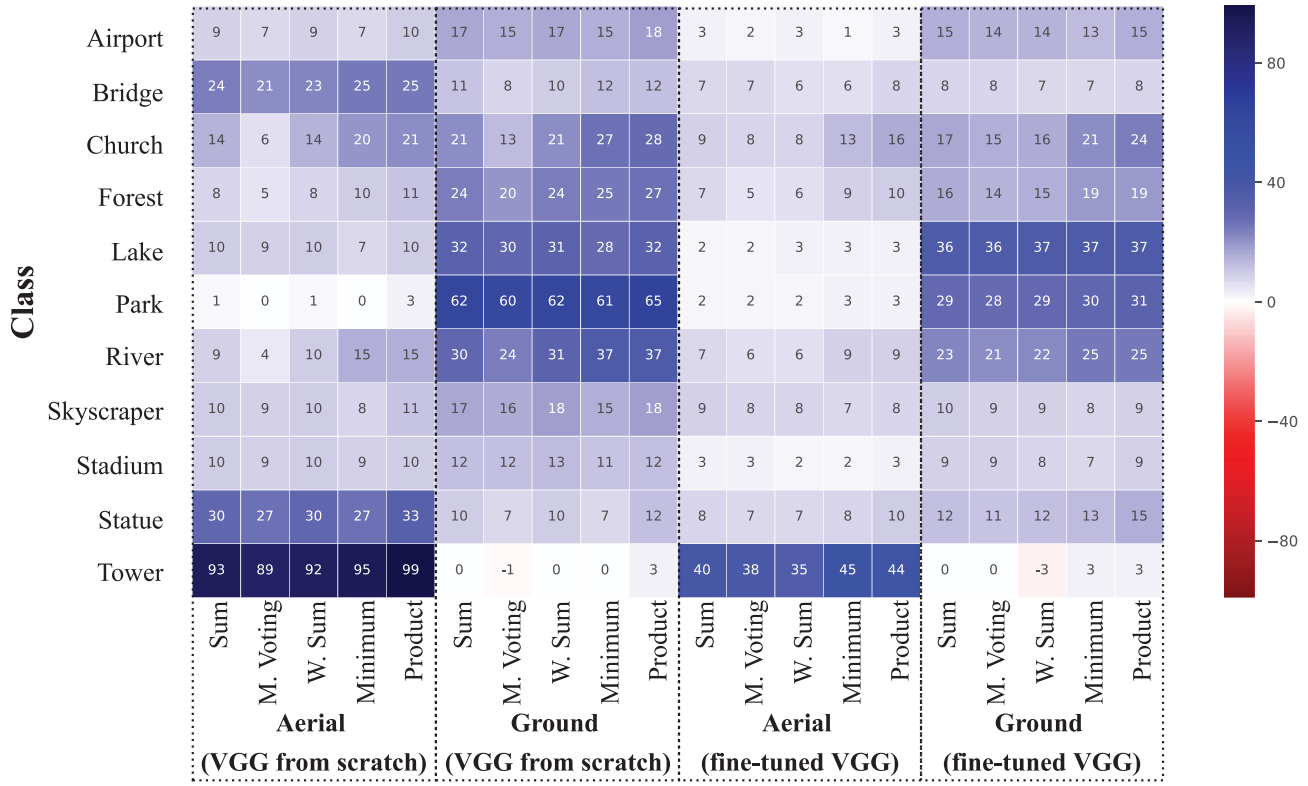
Fusion Type	Network	GPU train time (in seconds)	Total Parameters (in milions)
Early Fusion	**AlexNet [20]**	294.00	56.72
	VGG [21]	600.00	128.60
	Inception [23]	5226.10	24.37
	ResNet [24]	736.40	10.81
	DenseNet [25]	6215.60	12.42
	SqueezeNet [26]	1107.60	0.73
	SENet [27]	5276.00	24.90
	SKNet [28]	20856.60	42.60
Late Fusion	**AlexNet [20]**	361.95	114.09
	VGG [21]	994.24	268.62
	Inception [23]	10284.88	48.84
	ResNet [24]	1164.06	22.36
	DenseNet [25]	11197.90	27.20
	SqueezeNet [26]	1604.75	1.48
	SENet [27]	9511.95	52.08
	SKNet [28]	33189.20	87.30

Note: It is important to mention that all the times were calculated using RTX2080TI and it was accounted only forward and backward time during the training phase using the strategy of training from scratch.

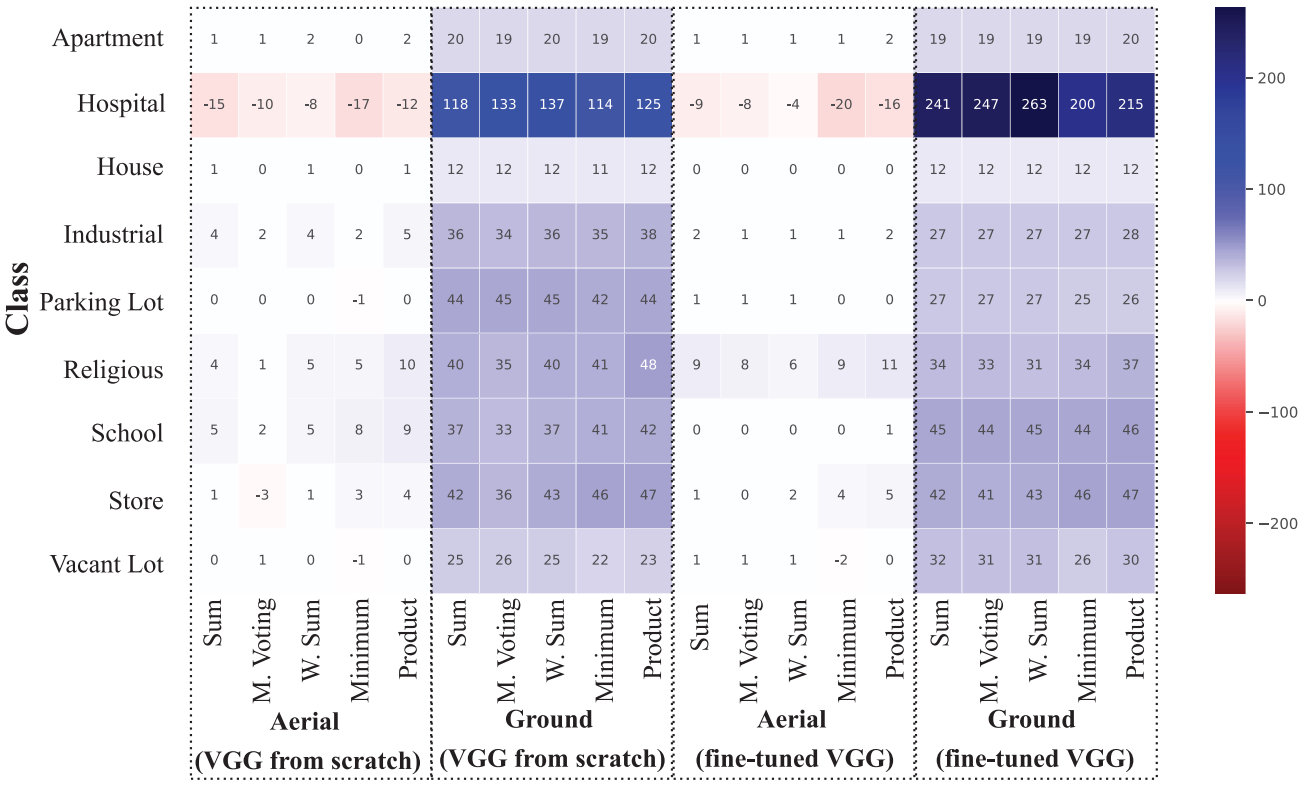
Similar to Fig. 10(a), we also produced a figure to the CV-BrCT dataset. In Fig. 10(b), we can see the impact of the different fusion methods in each class of the dataset, for each single-image-type network model (in this case, the VGG model). As we can see, all classes have an improvement in relation to the single-type networks of ground images. Furthermore, the Hospital class is the one mostly impacted by the addition of the aerial data. As hospital, usually, have large footprints, a single image from a frontal perspective can capture a facade easily confoundable with other classes facades. Consequently, the addition of an aerial view can distinguish an ambiguous hospital sample. In contrast, the aerial models display few improvements—probably a few ambiguous samples were corrected by frontal images—to all classes but Hospitals.

In Fig. 9(b), we see different examples of predictions in the CV-BrCT dataset. In three cases, where the final prediction is incorrect, it is arguably a plausible mistake, given the similarity between the images and the predicted classes. In general, the predictions of each individual model are befitting to the ambiguous visual of the images in the inputs. The cases with incorrect predictions, usually, have multiple classes' features, i.e., the first column where the parking lot is below an apartment building.

Remark: As can be noted in Tables VII and IX, for CV-BrCT, the late fusion models tended to achieve slightly better results than the early fusion networks. Again, some early fusion models achieved competitive results compared to the late fusion ones. For instance, analyzing models trained from scratch, the early fusion adaptation of Inception [23] matches to the late fusion approach for the same network, and for fine-tuned models, the



(a)



(b)

Fig. 10. Values represent the ratio of the accuracy per class between a single VGG [21], trained/fine-tuned in one specific domain, and a fusion of two VGGs [21], trained/fine-tuned on both aerial and ground views. In the numerator of this ratio, we calculated the difference between the late fusion accuracy and the accuracy using only one view, for each class. Therefore, positive/blue values indicate that the classification of that class was improved when comparing the network trained on a specific view and the fusion method, while the negative/red values indicate that the classification of that class worsened.

TABLE IX
RESULTS OF THE EVALUATED LATE FUSION TECHNIQUES FOR CV-BRCT DATASET

Training Strategy	Network	Fusion Strategy									
		Sum		M. Voting		W. Sum		Minimum		Product†	
		B. Acc.	F1 Score	B. Acc.	F1 Score	B. Acc.	F1 Score	B. Acc.	F1 Score	B. Acc.	F1 Score
Training from Scratch	AlexNet [20]	0.69 ± 0.03	0.80 ± 0.02	0.68 ± 0.03	0.80 ± 0.02	0.69 ± 0.03	0.81 ± 0.02	0.68 ± 0.03	0.80 ± 0.02	0.70 ± 0.03	0.82 ± 0.01
	VGG [21]	$\mathbf{0.72 \pm 0.03}$	$\mathbf{0.82 \pm 0.02}$	$\mathbf{0.71 \pm 0.03}$	$\mathbf{0.81 \pm 0.02}$	$\mathbf{0.72 \pm 0.03}$	0.82 ± 0.02	0.71 ± 0.02	$\mathbf{0.82 \pm 0.01}$	0.72 ± 0.02	$\mathbf{0.83 \pm 0.02}$
	Inception [23]	0.69 ± 0.02	0.81 ± 0.02	0.69 ± 0.03	0.80 ± 0.02	0.70 ± 0.02	0.81 ± 0.02	0.69 ± 0.03	0.81 ± 0.02	0.70 ± 0.02	0.82 ± 0.02
	ResNet [24]	0.69 ± 0.06	0.81 ± 0.03	0.68 ± 0.06	0.80 ± 0.03	0.70 ± 0.06	0.81 ± 0.03	0.68 ± 0.06	0.81 ± 0.03	0.70 ± 0.06	0.82 ± 0.03
	DenseNet† [25]	$\mathbf{0.72 \pm 0.03}$	$\mathbf{0.82 \pm 0.02}$	$\mathbf{0.71 \pm 0.02}$	$\mathbf{0.81 \pm 0.02}$	$\mathbf{0.72 \pm 0.02}$	$\mathbf{0.83 \pm 0.02}$	$\mathbf{0.71 \pm 0.02}$	$\mathbf{0.82 \pm 0.02}$	$\mathbf{0.73 \pm 0.03}$	$\mathbf{0.83 \pm 0.02}$
	SqueezeNet [26]	0.57 ± 0.05	0.72 ± 0.04	0.56 ± 0.05	0.70 ± 0.04	0.57 ± 0.05	0.72 ± 0.04	0.56 ± 0.04	0.72 ± 0.03	0.57 ± 0.05	0.73 ± 0.04
	SENet [27]	0.70 ± 0.04	0.81 ± 0.02	0.69 ± 0.04	0.80 ± 0.02	0.70 ± 0.04	0.81 ± 0.02	0.69 ± 0.03	0.80 ± 0.02	0.70 ± 0.04	0.82 ± 0.02
	SKNet [28]	0.69 ± 0.06	0.80 ± 0.04	0.68 ± 0.06	0.79 ± 0.04	0.69 ± 0.05	0.80 ± 0.03	0.67 ± 0.04	0.79 ± 0.03	0.69 ± 0.05	0.81 ± 0.03
Fine Tuning	AlexNet [20]	0.76 ± 0.03	0.85 ± 0.02	0.75 ± 0.03	0.84 ± 0.02	0.76 ± 0.03	0.85 ± 0.01	0.75 ± 0.02	0.85 ± 0.01	0.76 ± 0.02	0.86 ± 0.01
	VGG [21]	0.80 ± 0.03	0.88 ± 0.01	0.80 ± 0.03	$\mathbf{0.88 \pm 0.01}$	0.80 ± 0.03	0.88 ± 0.01	0.79 ± 0.02	$\mathbf{0.88 \pm 0.01}$	0.80 ± 0.02	0.88 ± 0.01
	Inception [23]	$\mathbf{0.81 \pm 0.01}$	0.88 ± 0.01	0.80 ± 0.02	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.81 \pm 0.02}$	0.88 ± 0.00	$\mathbf{0.80 \pm 0.02}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.81 \pm 0.01}$	0.89 ± 0.01
	ResNet [24]	0.78 ± 0.02	0.87 ± 0.01	0.78 ± 0.02	0.86 ± 0.01	0.78 ± 0.02	0.87 ± 0.01	0.77 ± 0.03	0.87 ± 0.01	0.79 ± 0.02	0.87 ± 0.01
	DenseNet [25]	0.81 ± 0.02	0.88 ± 0.01	0.80 ± 0.02	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.81 \pm 0.03}$	0.88 ± 0.01	$\mathbf{0.80 \pm 0.02}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.81 \pm 0.02}$	0.89 ± 0.01
	SqueezeNet [26]	0.72 ± 0.02	0.83 ± 0.01	0.72 ± 0.02	0.82 ± 0.00	0.73 ± 0.02	0.83 ± 0.00	0.71 ± 0.02	0.83 ± 0.01	0.73 ± 0.02	0.84 ± 0.01
	SENet [27]	$\mathbf{0.81 \pm 0.02}$	0.88 ± 0.00	$\mathbf{0.81 \pm 0.02}$	$\mathbf{0.88 \pm 0.00}$	$\mathbf{0.81 \pm 0.02}$	0.88 ± 0.00	$\mathbf{0.80 \pm 0.02}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.81 \pm 0.02}$	0.89 ± 0.01
	SKNet† [28]	$\mathbf{0.81 \pm 0.04}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.81 \pm 0.04}$	$\mathbf{0.88 \pm 0.01}$	$\mathbf{0.81 \pm 0.03}$	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.80 \pm 0.04}$	$\mathbf{0.88 \pm 0.02}$	$\mathbf{0.81 \pm 0.04}$	$\mathbf{0.89 \pm 0.02}$

The † symbols mark the best overall network and fusion strategy.
Bold values indicates the best results achieved by each type of data and/or training strategy.

same happened to selective kernels networks [28]. Finally, we would like to highlight the times spent to train each model in the CV-BrCT dataset. As can be noted in Table IV, it also demonstrates that early fusion models tend to converge faster than late fusion ones.

VII. CONCLUSION

In this work, we introduced two new publicly available datasets for multiview image tasks, which were named AiRound and CV-BrCT. We conducted extensive experiments in which results can be summarized as follows.

1) Early and late fusion-based aerial and ground feature combination yielded very relevant results, but there is still room for improvements, especially in the CV-BrCT dataset.
2) Fine-tuned models with feature fusion are quite effective.
3) Some classes in the dataset were unable to benefit from the multispectral information present in the Sentinel-2 images from AiRound.

Future works include the expansion of both AiRound and CV-BrCT to include more samples and classes to the proposed multiview benchmark. The addition of a multitude of classes containing few samples could allow for a standard evaluation of multiview few-shot learning algorithms. This would follow the steps of many other datasets in the computer vision literature [32]–[34] that propose standard evaluation protocols for zero-, one-, or few-shot scenarios.

We also point out the need for deeper studies about feature fusion involving multispectral data, multiview domain adaptation, the use of metalearning, i.e., few-shot learning and/or zero-shot learning, involving multiview fusion, and more sophisticated feature fusion techniques, such as hybrid fusion or techniques that can handle well with lack of data or the presence of noisy images.

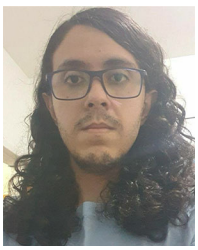
VII. ACKNOWLEDGMENT

The authors would like to thank NVIDIA for the donation of the GPUs that allowed the execution of all experiments in this article.

REFERENCES

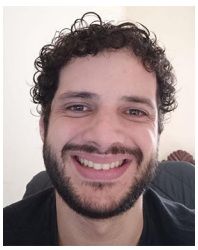
[1] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau, and R. Moore, "Google Earth Engine: Planetary-scale geospatial analysis for everyone," *Remote Sens. Environ.*, vol. 202, pp. 18–27, 2017.
[2] H. Qiu, C. Wang, J. Wang, N. Wang, and W. Zeng, "Cross view fusion for 3D human pose estimation," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2019, pp. 4341–4350.
[3] S. Hu, M. Feng, R. M. Nguyen, and G. H. Lee, "CVM-Net: Cross-view matching network for image-based ground-to-aerial geo-localization," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2018, pp. 7258–7267.
[4] S. Srivastava, J. E. Vargas-Muñoz, and D. Tuia, "Understanding urban landuse from the above and ground perspectives: A deep learning, multimodal solution," *Remote Sens. Environ.*, vol. 228, pp. 129–143, 2019.
[5] E. J. Hoffmann, Y. Wang, M. Werner, J. Kang, and X. X. Zhu, "Model fusion for building type classification from aerial and street view images," *Remote Sens.*, vol. 11, 2019, Art. no. 1259.
[6] N. Ghouaiel and S. Lefèvre, "Coupling ground-level panoramas and aerial imagery for change detection," *Geo-Spatial Inf. Sci.*, vol. 19, pp. 222–232, 2016.
[7] J. D. Wegner, S. Branson, D. Hall, K. Schindler, and P. Perona, "Cataloging public objects using aerial and street-level images-urban trees," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2016, pp. 6014–6023.
[8] R. Cao *et al.*, "Integrating aerial and street view images for urban land use classification," *Remote Sens.*, vol. 10, 2018, Art. no. 1553.
[9] L. Liu and H. Li, "Lending orientation to neural networks for cross-view geo-localization," in *Proc. Int. Conf. Pattern Recognit.*, 2019, pp. 5617–5626.
[10] A. L. Majdik, Y. Albers-Schoenberg, and D. Scaramuzza, "MAV urban localization from Google Street View data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 3979–3986.
[11] T. Koch, M. Korner, and F. Fraundorfer, "Automatic alignment of indoor and outdoor building models using 3D line segments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2016, pp. 10–18.
[12] M. Rumpler *et al.*, "Evaluations on multi-scale camera networks for precise and geo-accurate reconstructions from aerial and terrestrial images with user guidance," *Comput. Vis. Image Understanding*, vol. 157, pp. 255–273, 2017.
[13] M. Zhai, Z. Bessinger, S. Workman, and N. Jacobs, "Predicting ground-level scene layout from aerial imagery," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2017, pp. 4132–4140.
[14] S. Workman, R. Souvenir, and N. Jacobs, "Wide-area image geolocalization with aerial reference imagery," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 3961–3969.
[15] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *Proc. IEEE Conf. Comput. Vis.Pattern Recognit.*, 2015, pp. 5007–5015.
[16] Y. Tian, C. Chen, and M. Shah, "Cross-view image matching for geo-localization in urban environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3608–3616.
[17] S. Workman, M. Zhai, D. J. Crandall, and N. Jacobs, "A unified model for near and remote sensing," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2688–2697.

[18] E. Ferreira, M. Brito, R. Balaniuk, M. S. Alvim, and J. A. dos Santos, "BrazilDAM: A benchmark dataset for tailings dam detection," in *Proc. IEEE Latin Amer. GRSS ISPRS Remote Sens. Conf. (LAGIRS)*, 2020, pp. 339–344.
[19] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. Annu. ACM-SIAM Symp. Discrete Algo.*, vol. 8, Jan. 2007, doi: 10.1145/1283383.1283494.
[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.
[22] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.
[23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.
[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[25] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2017, pp. 4700–4708.
[26] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 1–5, doi: 10.1109/CVPR.2015.7298594.
[27] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.
[28] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 510–519.
[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
[30] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 423–443, Feb. 2019.
[31] K. Nogueira, O. A. Penatti, and J. A. dos Santos, "Towards better exploiting convolutional neural networks for remote sensing scene classification," *Pattern Recognit.*, vol. 61, pp. 539–556, 2017.
[32] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "The Omniglot challenge: A 3-year progress report," *Current Opinion Behavioral Sci.*, vol. 29, pp. 97–104, 2019.
[33] O. Vinyals *et al.*, "Matching networks for one shot learning," in *Proc. Advances Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.
[34] E. Triantafillou *et al.*, "Meta-dataset: A dataset of datasets for learning to learn from few examples," 2019, *arXiv:1903.03096*.



Gabriel Machado received the B.Sc. degree in computer science, in 2018 from Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, where he is currently working toward the M.Sc. degree in computer science.

His research interests include deep and machine learning, remote sensing, pattern recognition, and computer vision.



Edemir Ferreira received the B.Sc. degree in computational and applied mathematics and the M.Sc. degree in computer science, in 2014 and 2016, respectively, from Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, where he is currently working toward the doctoral degree in computer science.

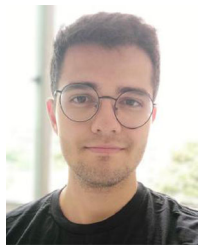
His research interests include deep and machine learning, pattern recognition, image processing, computer vision, and remote sensing.

Keiller Nogueira received the B.Sc. degree in computer science from Universidade Federal de Viçosa, Viçosa, Brazil, in 2012, and the M.Sc. and Ph.D. degrees in computer science from the Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, in 2015 and 2019, respectively.

He is currently a Lecturer with the Division of Computing Science and Mathematics, University of Stirling, Stirling, U.K. He has authored/coauthored several high-quality articles in leading journals and conferences. His research interests include deep and machine learning, pattern recognition, image processing, computer vision, and remote sensing.

Hugo Oliveira (Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science from Universidade Federal da Paraíba, João Pessoa, Brazil, in 2014 and 2016, respectively, and the Ph.D. degree in computer science from Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, in 2020.

He is currently a Postdoctoral Fellow with the Institute of Mathematics and Statistics (IME/USP), São Paulo, Brazil, and an Associate Researcher with the Pattern Recognition and Earth Observation (PATREO) Laboratory, Belo Horizonte. His research interests include machine learning, deep learning, domain adaptation, deep generative models, few-shot learning, metalearning, self-supervised learning, biomedical image analysis, remote sensing, image processing, pattern recognition, information theory, and data compression.



Matheus Brito is currently working toward the undergraduate degree in systems engineering with Universidade Federal de Minas Gerais, Belo Horizonte, Brazil.

His research interests include deep and machine learning, pattern recognition, image processing, computer vision, and remote sensing.

Pedro Henrique Targino Gama received the B.Sc. degree in computational and applied mathematics, in 2018 from Universidade Federal de Minas Gerais(UFMG), Belo Horizonte, Brazil, where he is currently working toward the M.Sc. degree with the Computer Science Department.

His research is primarily focused on Deep Learning and Computer Vision, specifically in Semantic Segmentation, although his interests include General Machine Learning, Pattern Recognition, and Remote Sensing.

Jefersson Alex dos Santos (Member, IEEE) received the Ph.D. degrees in computer science from Université de Cergy-Pontoise, Cergy, France, and also from the University of Campinas, Campinas, Brazil, in 2013.

He is currently an Associate Professor with the Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil. He has authored/coauthored several articles in journals with high impact factor and selective editorial policy. He has also authored/coauthored more than 30 articles in important conferences of remote sensing, image processing, and computer vision areas, in which he has experience in coordinating research with Brazilian funding agencies and R&D projects with companies. He is a Founder and a Coordinator of the Laboratory of Pattern Recognition and Earth Observation (PATREO—www.patreo.dcc.ufmg.br), one of Brazil's pioneer groups focused on the development of computer vision and machine learning for remote sensing applications. He has been a recipient of a Research Productivity Scholarship from the Brazilian Research Council (CNPq), since 2016.