

# AiRound and CV-BrCT: Novel Multi-View Datasets for Scene Classification

Gabriel Machado, Edemir Ferreira, Keiller Nogueira, Hugo Oliveira, Pedro Gama and Jefersson A. dos Santos

**Abstract**—It is undeniable that aerial/satellite images can provide useful information for a large variety of tasks. But, since these images are always looking 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 paper, 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 multi-view scene classification, using early and late fusion. Such experiments were conducted to show that image classification can be enhanced using multi-view data.

**Index Terms**—Remote Sensing, Deep Learning, Data Fusion, Multi-Modal Machine Learning, Dataset, Feature Fusion.

## 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 looking 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 multi-view 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

Authors would like to thank NVIDIA for the donation of the GPUs that allowed the execution of all experiments in this paper. We also thank CAPES, CNPq (424700/2018-2), and FAPEMIG (APQ-00449-17) for the financial support provided for this research.

Gabriel Machado, Edemir Ferreira, Hugo Oliveira, Pedro Gama and Jefersson A. dos Santos are with the Department of Computer Science, Universidade Federal de Minas Gerais, Brazil; gabriel.lucas@dcc.ufmg.br

Keiller Nogueira is with Computing Science and Mathematics, University of Stirling, Stirling, FK9 4LA, Scotland, UK; kno@cs.stir.ac.uk

many applications in different fields, like 3D human pose estimation [2], places geo-localization [3], and urban land use [4]. Motivated by these benefits, several approaches [5], [6], [7], [8], [9] have been proposed to exploit multi-view datasets to face distinct tasks. Although important, it is not easy to find multi-view 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 multi-view (aerial and ground) dataset for image classification tasks in the literature.

In this paper, we present two novel multi-view 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 multi-view scenarios. One of the datasets is composed of 1,165 triplets of images, each one of those consisting of a ground scene, a high-resolution aerial image, and a multi-spectral aerial data. The images are unevenly divided into 11 classes, including airport, bridge, church, forest, lake, river, skyscraper, stadium, statue, tower, and urban park. An interesting property of our dataset is that it was designed to contain a high inter-class variety, so it was selected places 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 8 different classes, which includes 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. It is important to emphasize that, although we assessed the performance of both dataset for image classification, they were proposed to be used in distinct image-related tasks, varying from image classification to cross-view matching and multi modal learning.

In summary, the contributions of this work are: (i) two novel multi-view scene classification datasets, named AiRound and CV-BrCT, (ii) a full evaluation of the proposed datasets in image classification tasks using several deep learning state-of-the-art methods and late fusion techniques, (iii) a novel methodology of performing early feature fusion using state-of-the-art deep architectures as a base.

The remainder of this paper is organized as follows. Section II presents related work. The proposed datasets are presented in Section III, while Section IV introduces the methods and tasks evaluated using these datasets. The experimental setup is introduced in Section V while Section VI presents the obtained results. Finally, Section VII concludes the paper.

## 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 information [10], [11], [12], [13]. Due to the high demand for images to be used by those kinds of tasks, a lot of multi-view 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, that were taken from across the United States. Relating to its data collection, the aerial images were collected using Bing Maps API (BM), and the ground images used Flickr and Google Street View API (GSV). 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 have around  $44k$  images. Relating to the CV-ACT dataset, it contains approximately  $128k$  images, that 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 Pittsburgh, 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, which ground samples were collected using GSV API, and the aerial ones used Google Maps (GM) API.

Another multi-view 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 it was used OSM to define the labels of all samples.

Relating to the Buildings [5] and le-de-France land use [4] datasets, both were designed for multi-view scene classification. The first dataset contains 56,259 paired aerial/street-level images of 4 different types of buildings, covering Washington DC, Puerto Rico, and 49 different states across the US. 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 le-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). Lastly, relating to multi-view image classification datasets, two datasets [5], [4] are quite similar to both datasets proposed here. However, neither of these existing datasets are publicly available, while ours will be.

## III. PROPOSED DATASETS

In this work, we proposed two novel multi-view datasets. **It is important to mention that both datasets are publicly available for research purposes at the project's website<sup>1</sup>.** Since both datasets were designed using a different methodology, in the following sections we will describe the relevant characteristics of each one and the methodology used to collect the samples.

### A. The AiRound Dataset

The first dataset is named AiRound, and is composed of 3,495 images distributed among 11 classes, including: airport, bridge, church, forest, lake, river, skyscraper, stadium, statue, tower, and urban park. Each sample is composed by a triplet, that contains images in 3 distinct points of view: (i) a ground perspective image; (ii) a high resolution RGB aerial image; and (iii) a multi-spectral image taken from the Sentinel-2 satellite. All images are paired and were manually checked to guarantee their correctness. The distribution of samples from AiRound can be checked in Figure 1 and examples of instances can be seen in Figure 2.

The data collected in this dataset are directly linked to real places around the world. To download the samples, two types of metadata were required: (i) the name of the place; and (ii) its correspondent geographical coordinates. Those information were collected using web crawlers in diversified lists of Wikipedia web pages. As instance, a list of tallest buildings<sup>2</sup> 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 required to create AiRound dataset, we recommend checking the project's website<sup>3</sup>.

Given the metadata, the RGB aerial images were collected using Bing Maps API<sup>4</sup>. 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

<sup>1</sup><http://www.patreo.dcc.ufmg.br/multi-view-datasets/>

<sup>2</sup>[https://en.wikipedia.org/wiki/List\\_of\\_tallest\\_buildings](https://en.wikipedia.org/wiki/List_of_tallest_buildings)

<sup>3</sup><http://www.patreo.dcc.ufmg.br/multi-view-datasets/>

<sup>4</sup><https://docs.microsoft.com/en-us/bingmaps/>

Dataset	Image Type			Publicly Available	Paired Aerial/Ground Images	Total of Samples	Number of Classes	Task	Year
	Aerial RGB	Ground	Multispectral						
CV-USA [14]	✓	✓	✗	✓	✓	~ 44k	-	Cross-View Matching	2015
Cities [15]	✓	✓	✗	✗	✓	~ 156k	-	Cross-View Matching	2015
Pasadena Urban Trees [7]	✓	✓	✗	✓	✓	~ 100k	18	Object Detection	2016
Brooklyn and Queens [17]	✓	✓	✗	✓	✓	~ 213k	-	Instance Segmentation	2017
Urban Environments [16]	✓	✓	✗	✗	✓	~ 18k	-	Cross-View Matching	2017
CV-ACT [9]	✓	✓	✗	✓	✓	~ 128k	-	Cross-View Matching	2019
Buildings [5]	✓	✓	✗	✗	✓	~ 261k	4	Classification	2019
le-de-France land use [4]	✓	✓	✗	✗	✓	~ 50k	16	Classification	2019
<b>AiRound (ours)</b>	✓	✓	✓	✓	✓	~ 3.5k	11	Classification	2020
<b>CV-BrCT (ours)</b>	✓	✓	✗	✓	✓	~ 48k	9	Classification	2020

TABLE I: Properties of other datasets found in the literature that are similar to AiRound and CV-BrCT.

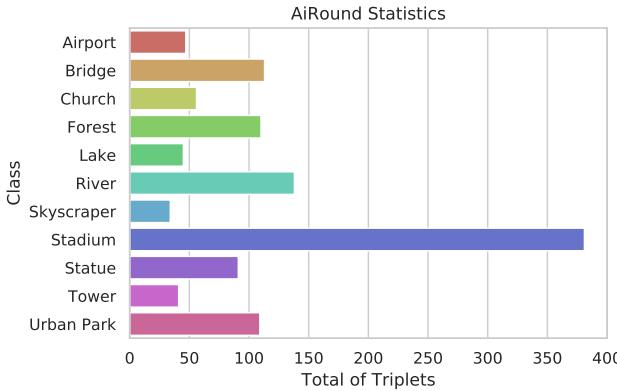


Fig. 1: Class distribution of the proposed AiRound dataset. Note that each image is represented by a triplet of ground, aerial, and multispectral data.

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 meters per pixel. Finally, it is important to mention that all aerial images downloaded have a image size of  $500 \times 500$  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, another 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 5 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 5 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. Lastly, it is important to mention that, besides most of the papers use Google Street-View API to download street-level data, we could not use such API for this dataset, because it is not capable of downloading images for some classes, like river or lake, for instance.

Finally, concerning 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  $300 \times 300$  pixels, a resolution that could cover all the classes' areas.

### B. CV-BrCT

The CV-BrCT dataset, which stands for Cross-View Brazilian Construction Type, comprises of approximate  $24k$  pairs of images 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 9 classes are:

- Apartment: Buildings with at least two stories primarily for residential use.
- Hospital: Health-related constructions, primarily hospitals but can include small particular clinics.
- House: a single-family residence.
- Industrial: Manufactured related buildings. Includes large storage constructions.
- Parking Lot: Includes, both open and indoors parking lots.
- Religious: Religious buildings; this includes catholic churches, protestant churches.
- School: Any school construction. From elementary schools to high school.
- Store: Any commercial or service related building.
- Vacant Lot: Demarcated areas without construction. It can include abandoned open areas.

Examples can be seen in Figure 4, whereas the class distribution is presented in Figure 3. Regarding the images, all of them are  $500 \times 500$  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 Sao Paulo, with some classes adding locations from states from other regions, i.e Goias in the Center-West region.

The data was collected following a simple protocol. For all the classes, a list of geographical coordinates was generated where each coordinate represents a location. Except for the Vacant Lot class, that was manually annotated, these lists were obtained from the publicly available data of the OpenStreetMap<sup>5</sup>, a community-based project were users annotate

<sup>5</sup>[www.openstreetmap.org/](http://www.openstreetmap.org/)

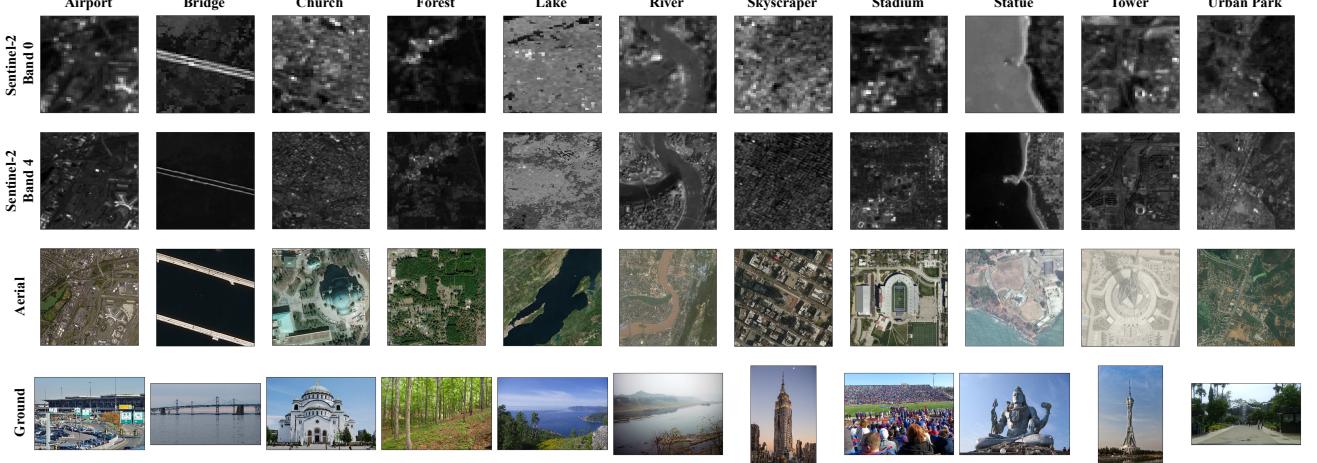


Fig. 2: Examples of instances taken from AiRound. The two top rows show channels of a sentinel-2 sample, while the third and fourth rows show the high resolution aerial perspective image and the ground view image, respectively.

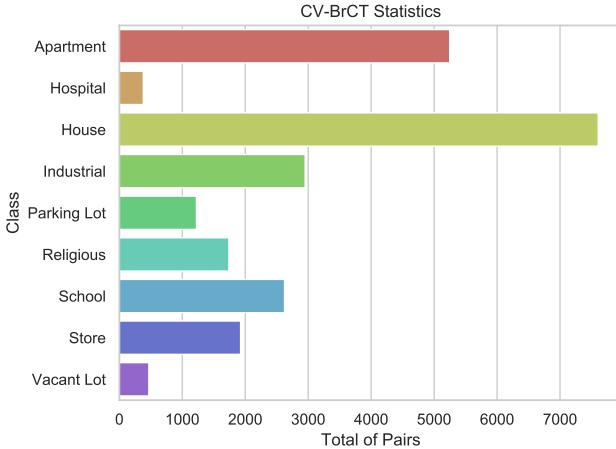


Fig. 3: Class distribution of the proposed CV-BrCT dataset.

aerial images to create maps, and collected using the Overpass API<sup>6</sup>. As the data is provided by users, not necessarily specialists, they can contain they can contain poorly annotated samples which can lead to outliers in the dataset. The lists are then fed to scripts that utilize the Google StaticMap API<sup>7</sup>, to collect the aerial images, and the Google StreetView API<sup>8</sup>, to collect the frontal images. With the exception of the zoom parameter, which was set to 19 empirically, the default values of the the Google APIs were used for the aerial images. As we gathered a large collection of locations, we ignored points where the StreetView API could not retrieve an image.

As a final step, an additional removal of outliers was applied after all the images were collected. This final filter consisted of firstly obtain a feature vector of the frontal images. These feature vector were produced by a ResNet pre-trained 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 3 s.d. away from a centroid cluster were removed from the dataset.

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

#### IV. BENCHMARKED METHODS

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

##### A. Deep Architectures

Convolutional Neural Networks (ConvNets) [20] have become the standard state-of-the-art technique for visual recognition over the last 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 multi-view image classification using the proposed datasets. All experimented techniques are described next.

**AlexNet** [21]. 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

<sup>6</sup><https://overpass-turbo.eu/>

<sup>7</sup><https://developers.google.com/maps/documentation/maps-static/intro>

<sup>8</sup><https://developers.google.com/maps/documentation/streetview/intro>

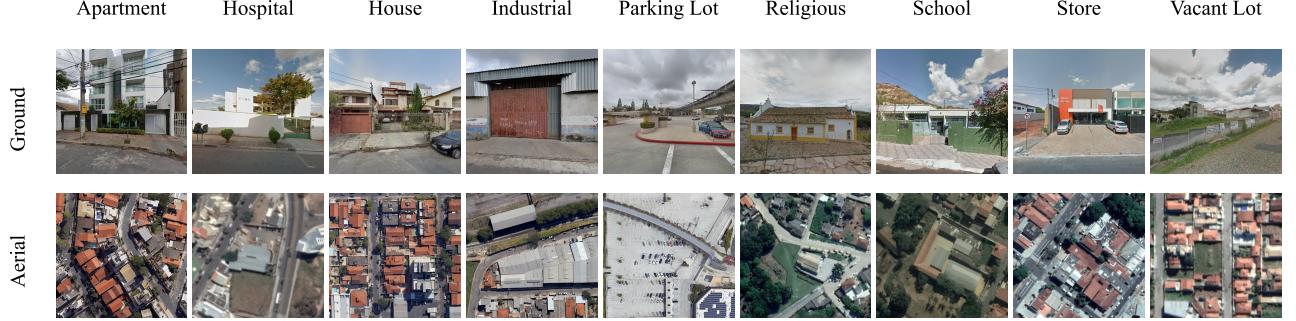


Fig. 4: Examples of instances taken from CV-BrCT.

input image. Figure 5a presents the architecture of AlexNet network.

**VGG** [22]. 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, that has 8  $3 \times 3$  convolutional layers, 5 pooling ones and 4 fully-connected ones (considering the softmax). Figure 5b illustrates the architecture of VGG-11 network.

**Inception** [23], [24]. Following the same guidelines of the VGG network [22], 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 the Figure 5c, it is possible to see how an inception-v3 architecture and inception modules work.

**ResNet** [25]. 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 [25], which has 18 convolutional layers with adding shortcuts, was evaluated. Figure 5d shows how a ResNet-18 architecture is built.

**DenseNet** [26]. Following the same idea of the ResNets [22], this architecture employed shortcut connections in order to allow the gradients to easily flow and better optimize the initial layers. The difference between ResNets [22] and DenseNet [26] is that in the former one, the shortcuts add the inputs, while 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 [25]. In this work, DenseNet-169 [26], which has 169 convolutional layers with shortcuts, was evaluated. The architecture of this model is presented in Figure 5e.

**SqueezeNets** [27]. 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 [21]) with between one or two orders of magnitude fewer parameters. Figure 5f illustrates how a fire module works and how they are integrated with a SqueezeNet architecture.

**Squeeze and Excitation Networks** [28]. Instead of focusing on spatial components to enhance feature extraction results, this work focus on the relationship between channels. For this task, the authors propose a new block named “Squeeze and Excitation block”. This block operates recalibrating channel-wise 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 [30] and that the use of this block can be easily adapted to existing architectures. Figure 5g shows how a “Squeeze and Excitation block” works and how it can be implemented in a ResNet-50 architecture.

**Selective Kernels Networks** [29]. 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 work, the authors also show that using these blocks on a ResNet [25] can outperform previously state-of-the-art results on the ImageNet dataset [30]. Through Figure 5h 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 late fusion algorithms and a novel early fusion methodology. In this work, those techniques were applied to fuse aerial/ground/satellite features, acquiring new information, and enhancing the final predictions. In the sections bellow, we will describe all such techniques.

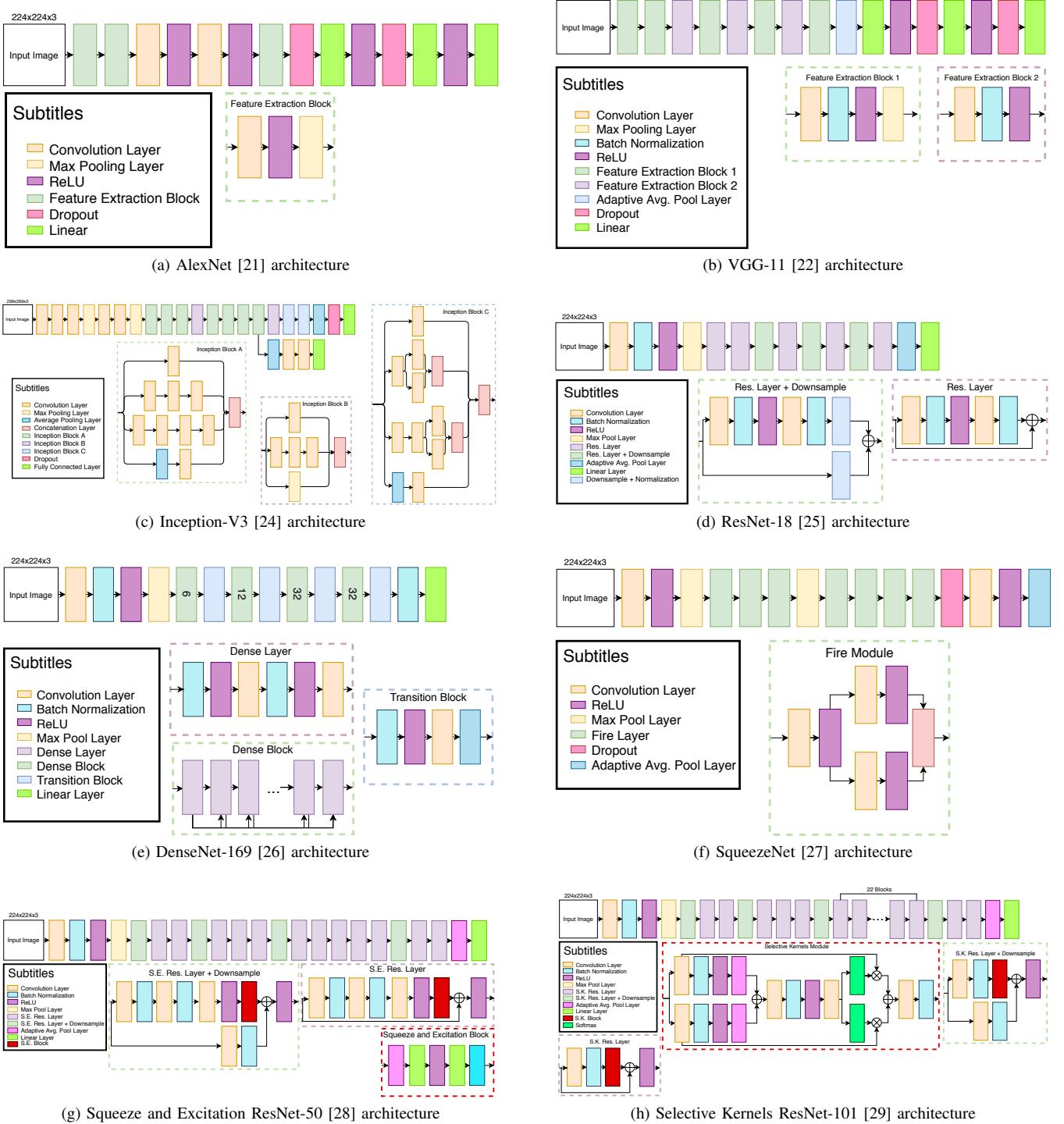


Fig. 5: Benchmarked architectures.

1) *Early Fusion Methods*: In order to exploit the correlations and interactions between low level features from different modalities [31], we propose early fusion approaches 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 proposed for 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 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 of kernels that each convolution layer have. So, to be possible of fully explore pre-trained

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, as can be seen in Figure 6.

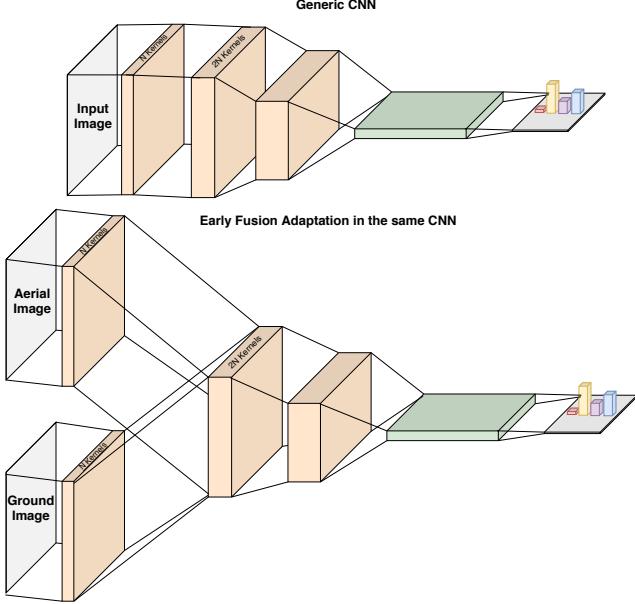


Fig. 6: Example of the proposed early fusion methodology.

2) *Late Fusion Methods:* Late fusion or decision-based algorithms performs integration of results after each of the modalities has made a prediction [31]. Those algorithms use uni-modal decision values and combine them using different types of fusion mechanisms, such as averaging, voting schemes, or weighting based. Figure 7 presents a typical late fusion procedure exploited in this work.

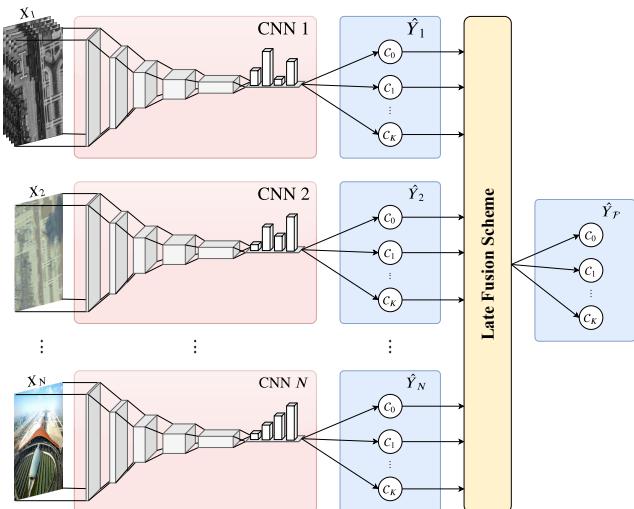


Fig. 7: A typical late fusion pipeline. As can be seen, each ConvNet is trained individually and their outputs are combined using a late fusion algorithm, resulting in the final prediction.

To formally define all the fusion operations used for this work, for all definitions of this section we will use the following notation. Let  $\sigma_i$  be the softmax scores returned by the network  $i$ ,  $\alpha_i$  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 equation 1.

$$\text{Sum}_{\text{Prediction}} = \arg \max \sum_{i=1}^m \sigma_i \quad (1)$$

**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 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 equation 2.

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

In the case of this work, majority voting was used to perform late fusion between 2 or 3 models, so ties constantly happen. To solve this issue, as a tiebreaker 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 2 models since there is no point in checking what is the most common vote between two voters. The procedure used for tiebreaker and voting using only 2 models is formally defined by equation 3.

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

**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 equation 4, 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.

$$\text{WSum}_{\text{Prediction}} = \arg \max \sum_{i=1}^{i=m} \alpha_i \sigma_i \quad (4)$$

**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 which 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. Equation 5 formally defines the described procedure.

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

**Product.** The product fusion is a very used late fusion algorithm. The main idea of it is to perform an element-wise multiplication between softmax scores and after that return the index which contains the biggest value. This procedure is defined by equation 6.

$$Prod_{Prediction} = \arg \max \prod_{i=1}^m \sigma_i \quad (6)$$

## 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 5-fold cross-validation protocol was used to properly evaluate each technique. We reported the mean of balanced accuracy and/or F1-score, taken from all 5 folds experiments with its correspondent standard deviation. Finally, in Section V-A we present the protocol used to train the models using AiRound dataset, while 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 pre-trained 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 300 epochs, using early stop technique with 30 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 a individual network for each kind of data, as was shown in Figure 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 2 or 3 networks, trained in different views,

by combining alternated models' outputs, e.g., 3-views, aerial with ground, etc. All the combinations were made using models with the same architecture and trained using data from only 1-view perspective.

For the early fusion models, an end-to-end training was performed, using aerial + ground paired data as inputs. All the training process was also made using the same combination of hyperparameters used to train the 1-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 300, 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 1-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 1-view in Section VI-B1, while the results of the models using fusion techniques can be found in Section VI-B2.

### 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 [30]). 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 we did not fine-tune the networks for sentinel-2 images, given the incompatibility between the number of bands of this data and the number of input channels expected by the networks, i.e., given that Sentinel-2 data has 13 channels, and the first convolution layer of the evaluated architectures receive only 3 input bands (RGB). We also decided not to use only Sentinel-2's RGB channels as input to pre-trained models, because the average spatial resolution of the aerial-RGB images is much better than Sentinel-2, providing much more details and information.

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 [32]. Comparing each training strategy separately, we can notice that, in both cases, the networks yielded very similar results (mainly for the aerial and ground data) without one

model clearly outperforming others. Furthermore, considering the distinct input data, one may note, from the experiments with networks trained from scratch, that aerial and ground images tend to produce comparable outcomes, while sentinel-2 data tend to yield worse results. This may be justified by the difference in the spatial resolution of the images since Sentinel-2 data has a resolution in meters per pixel whereas the aerial and ground have pixel resolution in centimeters per pixel.

Training Strategy	Network	Input Data		
		Aerial	Ground	Sentinel-2
Training from scratch	AlexNet [21]	0.75 ± 0.06	0.75 ± 0.03	0.51 ± 0.07
	VGG [22]	<b>0.79 ± 0.07</b>	<b>0.78 ± 0.03</b>	<b>0.71 ± 0.06</b>
	Inception [24]	0.69 ± 0.07	0.70 ± 0.05	0.69 ± 0.07
	ResNet [25]	0.76 ± 0.05	0.75 ± 0.04	0.70 ± 0.05
	DenseNet [26]	0.73 ± 0.04	0.74 ± 0.03	<b>0.71 ± 0.05</b>
	SqueezeNet [27]	0.70 ± 0.07	0.73 ± 0.03	0.59 ± 0.04
	SENet [28]	0.69 ± 0.04	0.69 ± 0.07	0.68 ± 0.04
	SKNet [29]	0.67 ± 0.05	0.65 ± 0.10	0.60 ± 0.04
Fine tuning from ImageNet	AlexNet [21]	0.82 ± 0.05	0.82 ± 0.02	-
	VGG [22]	0.88 ± 0.03	0.86 ± 0.03	-
	Inception [24]	0.88 ± 0.04	<b>0.88 ± 0.03</b>	-
	ResNet [25]	0.87 ± 0.02	0.86 ± 0.05	-
	DenseNet [26]	0.88 ± 0.05	0.87 ± 0.02	-
	SqueezeNet [27]	0.85 ± 0.05	0.83 ± 0.02	-
	SENet [28]	0.87 ± 0.02	0.87 ± 0.03	-
	SKNet [29]	<b>0.90 ± 0.02</b>	0.87 ± 0.01	-

TABLE II: Results in terms of F1 Score of the evaluated models for AiRound dataset.

2) *Multi-View Fusion Strategies*: This section presents and discusses the results obtained applying early and late fusion techniques.

**AiRound Early Fusion.** Like was specified before, for early fusion experiments we followed the scheme described at Section IV-B1 using all the 8 architectures that were also previously described. In Table III, it is notable that all the results achieved a superior mark compared to the 1-view results reported in Table II.

Analyzing the main fusion gains, for the models trained from scratch, we can highlight two models: the VGG [22], that achieve the best results, and the inception-v3 [24], which achieved the biggest gain in F1 score (0.1) comparing to the 1-view experiments, previously reported. For the fine-tuned models, we can highlight the DenseNet [26] adaption for early fusion, that obtained a gain of 0.05 in F1 Score, comparing to the same network trained on single-view. It is also notable that early fusion DenseNet achieved the best overall result between all early fusion networks.

Finally, for some results, it is notable that a downgrade occurred, if compared to the 1-view baseline presented in Table II. For those results, we hypothesize that the same feature degradation phenomenon<sup>9</sup>, that 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 3 different models from scratch, all the

<sup>9</sup>A destructive effect that occurs in training phase due to the misalignment of the geometry of the bottleneck features of the two image types.

Early Fusion Networks	Training Strategy			
	From Scratch		Fine Tuning	
	B. Acc.	F1 Score	B. Acc.	F1 Score
AlexNet [21]	0.73 ± 0.06	0.80 ± 0.06	0.85 ± 0.05	0.89 ± 0.03
VGG [22]	<b>0.76 ± 0.03</b>	<b>0.83 ± 0.03</b>	0.83 ± 0.04	0.88 ± 0.03
Inception [24]	0.71 ± 0.06	0.79 ± 0.03	0.87 ± 0.02	0.91 ± 0.01
ResNet [25]	0.69 ± 0.04	0.77 ± 0.02	0.85 ± 0.03	0.89 ± 0.03
DenseNet [26]	0.68 ± 0.08	0.77 ± 0.05	<b>0.89 ± 0.04</b>	<b>0.94 ± 0.01</b>
SqueezeNet [27]	0.64 ± 0.07	0.75 ± 0.06	0.78 ± 0.05	0.85 ± 0.03
SENet [28]	0.69 ± 0.09	0.77 ± 0.06	0.85 ± 0.04	0.89 ± 0.02
SKNet [29]	0.66 ± 0.07	0.74 ± 0.04	0.87 ± 0.06	0.91 ± 0.04

TABLE III: Results of the evaluated early fusion networks for AiRound dataset.

combinations would result in 184 experiments. Given that high number of experiments and that, as previously discussed, all networks produced similar results and any one could be selected for further experiments, we reported the obtained results for only the VGG [22].

As can be seen in Figure 8, most of the late fusion techniques outperformed the models trained with only one view, with a special highlight for the 3-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.

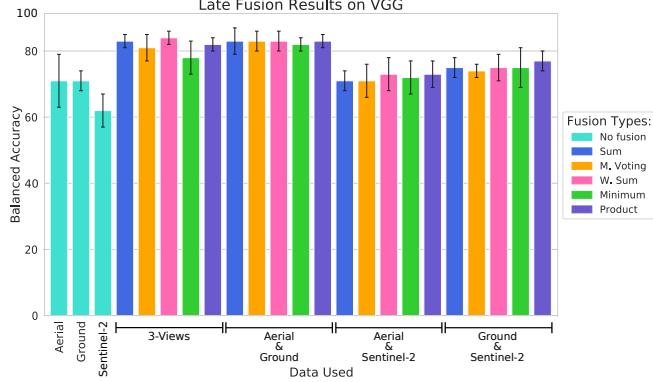
Finally, considering the ground-Sentinel-2 fusions, it is possible to notice a little improvement, that can be justified by the same reason of aerial-ground fusions. We conclude that the gain in this fusion was not as good as aerial-ground one, because of the limited spatial-resolution that Sentinel-2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

For the fine-tuned models, the late fusion results are presented in Table IV. 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).

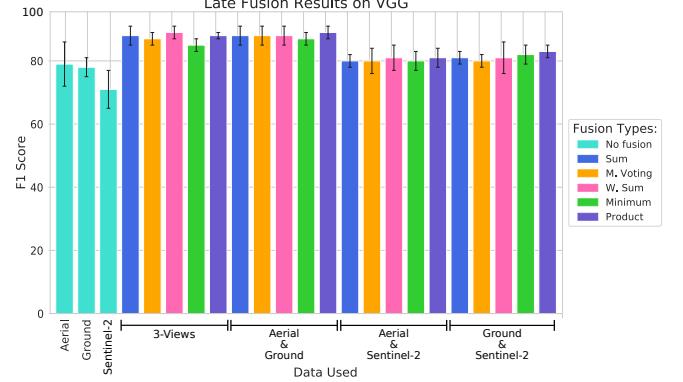
Comparing the results using only one type of data (Table II) with the fusion outcomes, it is possible to notice that the late fusion outperformed all approaches using only one view. This corroborates with our initial analysis that the combination of multi-source data could improve the results for the scene classification task. Furthermore, the late fusion results of the fine-tuned models yielded better results than the fusion results of the networks trained from scratch, an expected outcome.

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 [22] were reported.

Figure 9a 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



(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.

Fig. 8: Results comparison of all fusion types using VGG trained from scratch..

Network	Fusion Strategy									
	Sum		M. Voting		W. Sum		Minimum		Product	
	B. Acc.	FI Score								
AlexNet [21]	0.87 ± 0.06	0.90 ± 0.05	0.86 ± 0.08	0.89 ± 0.06	0.87 ± 0.08	0.90 ± 0.05	0.87 ± 0.04	0.91 ± 0.05	0.89 ± 0.06	0.92 ± 0.04
VGG [22]	<b>0.93 ± 0.02</b>	<b>0.95 ± 0.02</b>	<b>0.93 ± 0.02</b>	0.94 ± 0.01	0.92 ± 0.02	0.94 ± 0.01	0.92 ± 0.06	0.94 ± 0.04	0.93 ± 0.04	0.95 ± 0.02
Inception [24]	<b>0.93 ± 0.03</b>	<b>0.95 ± 0.02</b>	<b>0.93 ± 0.03</b>	<b>0.95 ± 0.02</b>	<b>0.93 ± 0.04</b>	<b>0.95 ± 0.02</b>	<b>0.93 ± 0.03</b>	<b>0.95 ± 0.01</b>	<b>0.94 ± 0.03</b>	0.95 ± 0.01
ResNet [25]	0.92 ± 0.02	0.94 ± 0.01	0.91 ± 0.03	0.93 ± 0.02	0.92 ± 0.03	0.94 ± 0.01	0.91 ± 0.02	0.93 ± 0.01	0.92 ± 0.04	0.94 ± 0.02
DenseNet [26]	<b>0.93 ± 0.03</b>	<b>0.95 ± 0.02</b>	0.92 ± 0.03	0.94 ± 0.02	0.92 ± 0.03	0.94 ± 0.02	0.92 ± 0.03	0.94 ± 0.02	0.93 ± 0.04	0.95 ± 0.02
SqueezeNet [27]	0.91 ± 0.04	0.93 ± 0.03	0.91 ± 0.04	0.92 ± 0.03	0.90 ± 0.05	0.92 ± 0.02	0.88 ± 0.04	0.91 ± 0.02	0.91 ± 0.03	0.93 ± 0.02
SENet [28]	0.92 ± 0.03	0.93 ± 0.02	0.92 ± 0.04	0.93 ± 0.02	0.92 ± 0.03	0.93 ± 0.02	0.92 ± 0.04	0.94 ± 0.02	0.92 ± 0.04	0.94 ± 0.02
SKNet [29]	<b>0.93 ± 0.04</b>	<b>0.95 ± 0.02</b>	<b>0.93 ± 0.04</b>	<b>0.95 ± 0.03</b>	<b>0.93 ± 0.03</b>	<b>0.95 ± 0.02</b>	0.92 ± 0.04	<b>0.95 ± 0.02</b>	<b>0.94 ± 0.04</b>	<b>0.96 ± 0.02</b>

TABLE IV: Results of the evaluated late fusion techniques for AiRound dataset using fine-tuned models.

see which classes benefit the most from each fusion in each type of view.

Through Figure 9a, it is possible to observe that, for aerial data, the classes tower, skyscraper, and statue were the ones that most benefited from the aerial/ground fusion. This is due to the fact that all 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. Concerning the ground images, the classes lake, river, and urban 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, urban parks are naturally located in cities and, therefore, the information about the existence of a city nearby, that come from aerial images, may help in its classification. 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 area (such as the surrounding vegetation) which may help in these two classes' discrimination.

**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 competitive results, as can be seen comparing the results for fine-tuned DenseNets [26], for instance. Lastly, we evaluated a series of experiments, measuring the spent time to train each one of the models in both datasets. Table V contain those results, demonstrating clearly the aforementioned advantages of early fusion for AiRound dataset.

Fusion Type	Network	GPU Train Time AiRound (in seconds)	GPU Train Time CV-BrCT (in seconds)	Total Parameters (in millions)
Early Fusion	AlexNet [21]	34.98	294.00	56.72
	VGG [22]	41.95	600.00	128.60
	Inception [24]	636.92	5226.10	24.37
	ResNet [25]	59.16	736.40	10.81
	DenseNet [26]	527.04	6215.60	12.42
	SqueezeNet [27]	89.47	1107.60	0.73
	SENet [28]	461.42	5276.00	24.90
	SKNet [29]	1705.60	20856.60	42.60
Late Fusion	AlexNet [21]	129.92	361.95	114.09
	VGG [22]	144.54	994.24	268.62
	Inception [24]	1231.75	10284.88	48.84
	ResNet [25]	207.37	1164.06	22.36
	DenseNet [26]	984.54	11197.90	27.20
	SqueezeNet [27]	264.29	1604.75	1.48
	SENet [28]	1166.22	9511.95	52.08
	SKNet [29]	4942.72	33189.20	87.30

TABLE V: Benchmarked methods properties. It is important to mention that all the times were calculated using a RTX2080TI and it was accounted only forward and backward time during the training phase.

## B. Experiments on the CV-BrCT

1) *Networks Architectures 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 the table VI.

As seen in the 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, while 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 a similar facades, prevalence of stores in first floors of buildings, etc. Thus the discrepancy between results of different image types.

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

Training Strategy	Network	Input Data			
		Aerial		Ground	
		B. Acc.	F1 Score	B. Acc.	F1 Score
Training from scratch	AlexNet [21]	0.68 ± 0.03	0.79 ± 0.02	0.50 ± 0.03	0.62 ± 0.01
	VGG [22]	0.70 ± 0.04	<b>0.81 ± 0.03</b>	<b>0.54 ± 0.02</b>	<b>0.66 ± 0.01</b>
	Inception [24]	0.69 ± 0.03	0.80 ± 0.02	0.49 ± 0.03	0.62 ± 0.02
	ResNet [25]	0.68 ± 0.07	0.79 ± 0.03	0.50 ± 0.05	0.63 ± 0.03
	DenseNet [26]	<b>0.71 ± 0.02</b>	<b>0.81 ± 0.01</b>	0.49 ± 0.01	0.62 ± 0.01
	SqueezeNet [27]	0.55 ± 0.07	0.70 ± 0.05	0.41 ± 0.08	0.56 ± 0.06
	SENet [28]	0.69 ± 0.04	0.80 ± 0.02	0.49 ± 0.02	0.62 ± 0.02
	SKNet [29]	0.68 ± 0.06	0.79 ± 0.04	0.47 ± 0.03	0.61 ± 0.02
	AlexNet [21]	0.75 ± 0.02	0.84 ± 0.01	0.54 ± 0.01	0.66 ± 0.01
Fine Tuning from ImageNet	VGG [22]	0.79 ± 0.03	0.87 ± 0.01	<b>0.60 ± 0.02</b>	<b>0.71 ± 0.01</b>
	Inception [24]	<b>0.80 ± 0.02</b>	0.87 ± 0.00	<b>0.60 ± 0.03</b>	<b>0.71 ± 0.01</b>
	ResNet [25]	0.78 ± 0.02	0.86 ± 0.01	0.58 ± 0.04	0.69 ± 0.02
	DenseNet [26]	<b>0.80 ± 0.02</b>	0.87 ± 0.01	<b>0.60 ± 0.01</b>	<b>0.71 ± 0.01</b>
	SqueezeNet [27]	0.70 ± 0.02	0.80 ± 0.01	0.56 ± 0.02	0.68 ± 0.01
	SENet [28]	<b>0.80 ± 0.02</b>	0.87 ± 0.01	<b>0.60 ± 0.01</b>	<b>0.71 ± 0.01</b>
	SKNet [29]	0.80 ± 0.03	<b>0.88 ± 0.01</b>	<b>0.60 ± 0.02</b>	<b>0.71 ± 0.01</b>

TABLE VI: Results of the evaluated models for CV-BrCT dataset.

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 the table VII. As in the experiments, the fine-tuned models outperform the non-pretrained ones. In 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.

**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 VIII.

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.

Early Fusion Networks	Training Strategy			
	From Scratch		Fine Tuning	
	B. Acc.	F1 Score	B. Acc.	F1 Score
AlexNet [21]	0.69 ± 0.03	0.8 ± 0.01	0.72 ± 0.02	0.82 ± 0.01
VGG [22]	<b>0.73 ± 0.03</b>	0.82 ± 0.01	0.76 ± 0.02	0.84 ± 0.02
Inception [24]	<b>0.73 ± 0.04</b>	<b>0.83 ± 0.02</b>	0.79 ± 0.03	<b>0.87 ± 0.01</b>
ResNet [25]	0.68 ± 0.02	0.79 ± 0.01	0.74 ± 0.02	0.83 ± 0.01
DenseNet [26]	0.71 ± 0.04	0.80 ± 0.02	0.72 ± 0.03	0.81 ± 0.01
SqueezeNet [27]	0.60 ± 0.01	0.73 ± 0.02	0.67 ± 0.04	0.79 ± 0.02
SENet [28]	0.67 ± 0.04	0.78 ± 0.02	0.78 ± 0.02	0.86 ± 0.01
SKNet [29]	0.70 ± 0.04	0.80 ± 0.03	<b>0.80 ± 0.02</b>	<b>0.87 ± 0.01</b>

TABLE VII: Results of the evaluated early fusion networks for CV-BrCT dataset.

As the networks trained with only ground images are less reliable classifiers, i.e., 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, are less significant, thus the improvement exists but are relatively small.

Similar to Figure 9a, we also produced a figure to the CV-BrCT dataset. In Figure 9b, 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 an 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.

**Remark.** As can be noted in Tables VII, and VIII, 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 [24] matches to the late fusion approach for the same network, and for fine-tuned models, the same happened to Selective Kernels Network [29]. Lastly, we would like to highlight the times spent to train each model in the CV-BrCT dataset. As can be noted in Table V, 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 multi-view image tasks, which were named AiRound and CV-BrCT. We conducted extensive experiments in which results can be summarized as: (1) early and late fusion-based aerial + ground feature combination yielded very relevant results, but there is still room for improvements, specially in CV-BrCT dataset; (2) fine-tuned models with feature fusion are quite effective; (3) some classes in the dataset were

Training Strategy	Network	Fusion Strategy									
		Sum		M. Voting		W. Sum		Minimum		Product	
		B. Acc.	F1 Score								
Training from Scratch	AlexNet [21]	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 [22]	<b>0.72 ± 0.03</b>	<b>0.82 ± 0.02</b>	<b>0.71 ± 0.03</b>	<b>0.81 ± 0.02</b>	<b>0.72 ± 0.03</b>	0.82 ± 0.02	0.71 ± 0.02	<b>0.82 ± 0.01</b>	0.72 ± 0.02	<b>0.83 ± 0.02</b>
	Inception [24]	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 [25]	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 [26]	<b>0.72 ± 0.03</b>	<b>0.82 ± 0.02</b>	<b>0.71 ± 0.02</b>	<b>0.81 ± 0.02</b>	<b>0.72 ± 0.02</b>	<b>0.83 ± 0.02</b>	<b>0.71 ± 0.02</b>	<b>0.82 ± 0.02</b>	<b>0.73 ± 0.03</b>	<b>0.83 ± 0.02</b>
	SqueezeNet [27]	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
Fine Tuning	SENet [28]	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 [29]	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
	AlexNet [21]	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 [22]	0.80 ± 0.03	0.88 ± 0.01	0.80 ± 0.03	<b>0.88 ± 0.01</b>	0.80 ± 0.03	0.88 ± 0.01	0.79 ± 0.02	<b>0.88 ± 0.01</b>	0.80 ± 0.02	0.88 ± 0.01
	Inception [24]	<b>0.81 ± 0.01</b>	0.88 ± 0.01	0.80 ± 0.02	<b>0.88 ± 0.01</b>	<b>0.81 ± 0.02</b>	0.88 ± 0.00	<b>0.80 ± 0.02</b>	<b>0.88 ± 0.01</b>	<b>0.81 ± 0.01</b>	0.89 ± 0.01
	ResNet [25]	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
Training from Scratch	DenseNet [26]	0.81 ± 0.02	0.88 ± 0.01	0.80 ± 0.02	<b>0.88 ± 0.01</b>	<b>0.81 ± 0.03</b>	0.88 ± 0.01	<b>0.80 ± 0.02</b>	<b>0.88 ± 0.01</b>	<b>0.81 ± 0.02</b>	0.89 ± 0.01
	SqueezeNet [27]	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 [28]	<b>0.81 ± 0.02</b>	0.88 ± 0.00	<b>0.81 ± 0.02</b>	<b>0.88 ± 0.00</b>	<b>0.81 ± 0.02</b>	0.88 ± 0.00	<b>0.80 ± 0.02</b>	<b>0.88 ± 0.01</b>	<b>0.81 ± 0.02</b>	0.89 ± 0.01
	SKNet [29]	<b>0.81 ± 0.04</b>	<b>0.89 ± 0.01</b>	0.81 ± 0.04	<b>0.88 ± 0.01</b>	0.81 ± 0.03	<b>0.89 ± 0.01</b>	0.80 ± 0.04	<b>0.88 ± 0.02</b>	0.81 ± 0.04	<b>0.89 ± 0.02</b>

TABLE VIII: Results of the evaluated late fusion techniques for CV-BrCT dataset.

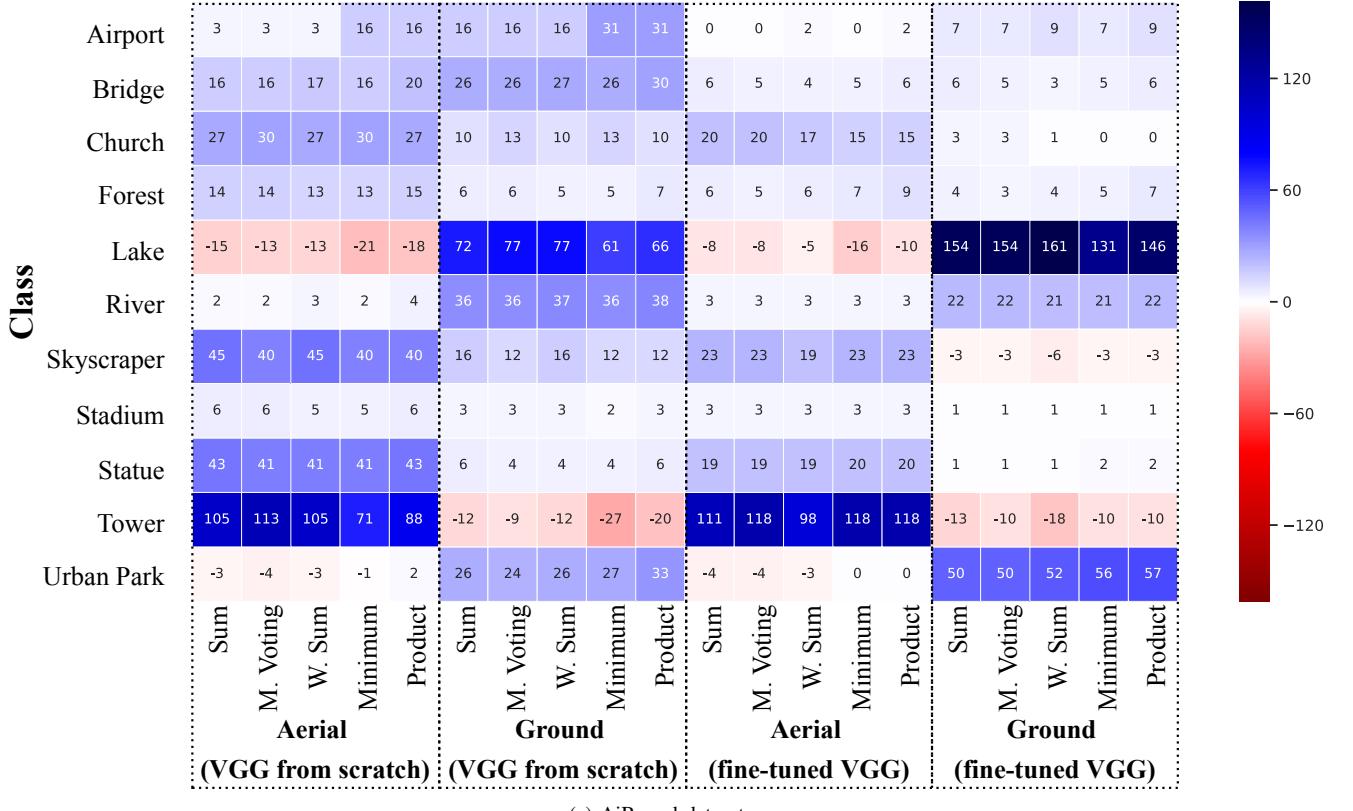
unable to benefit from the multispectral information present in the sentinel-2 images from AiRound.

As future work, we foresee to expand both datasets, adding more classes and instances to the existing ones. We also point out the need for deeper studies about cross-view matching algorithms, multi-view domain adaptation 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.

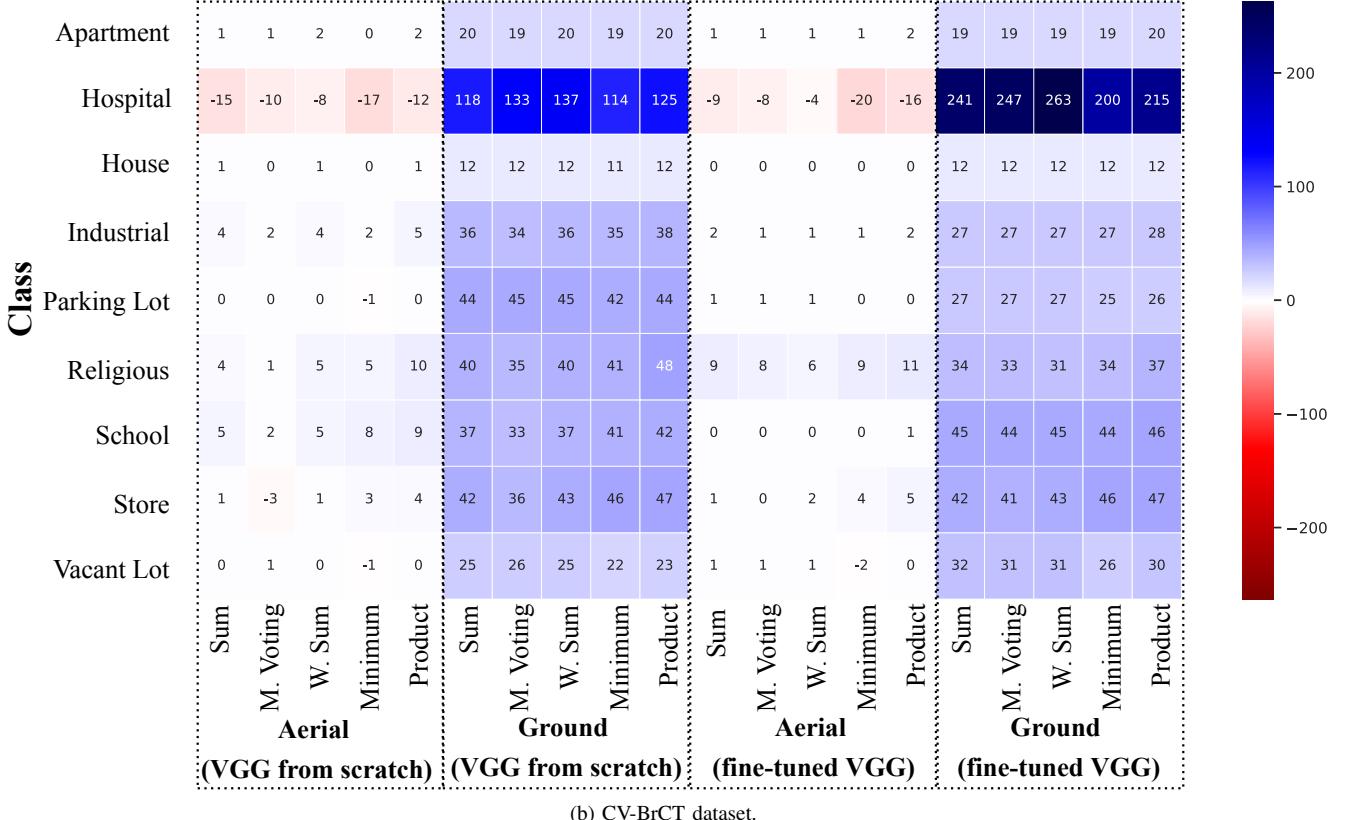
## REFERENCES

- [1] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau, and R. Moore, “Google earth engine: Planetary-scale geospatial analysis for everyone,” *Remote Sensing of Environment*, 2017.
- [2] H. Qiu, C. Wang, J. Wang, N. Wang, and W. Zeng, “Cross view fusion for 3d human pose estimation,” in *IEEE/CVF Computer Vision and Pattern Recognition*, 2019.
- [3] S. Hu, M. Feng, R. M. Nguyen, and G. Hee Lee, “Cvm-net: Cross-view matching network for image-based ground-to-aerial geo-localization,” in *IEEE/CVF Computer Vision and Pattern Recognition*, 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 Sensing of Environment*, 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 Sensing*, 2019.
- [6] N. Ghouaiel and S. Lefèvre, “Coupling ground-level panoramas and aerial imagery for change detection,” *Geo-spatial Information Science*, 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 *IEEE/CVF Computer Vision and Pattern Recognition*, 2016.
- [8] R. Cao, J. Zhu, W. Tu, Q. Li, J. Cao, B. Liu, Q. Zhang, and G. Qiu, “Integrating aerial and street view images for urban land use classification,” *Remote Sensing*, 2018.
- [9] L. Liu and H. Li, “Lending orientation to neural networks for cross-view geo-localization,” in *International Conference on Pattern Recognition*, 2019.
- [10] A. L. Majdik, Y. Albers-Schoenberg, and D. Scaramuzza, “Mav urban localization from google street view data,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 10–18.
- [12] M. Rumpler, A. Tscharf, C. Mostegel, S. Daftary, C. Hoppe, R. Prettenhofer, F. Fraundorfer, G. Mayer, and H. Bischof, “Evaluations on multi-scale camera networks for precise and geo-accurate reconstructions from aerial and terrestrial images with user guidance,” *Computer vision and 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 *IEEE/CVF Computer Vision and Pattern Recognition*, 2017.
- [14] S. Workman, R. Souvenir, and N. Jacobs, “Wide-area image geolocation with aerial reference imagery,” in , 2015.
- [15] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, “Learning deep representations for ground-to-aerial geolocation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5007–5015.
- [16] Y. Tian, C. Chen, and M. Shah, “Cross-view image matching for geo-localization in urban environments,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3608–3616.
- [17] S. Workman, M. Zhai, D. J. Crandall, and N. Jacobs, “A unified model for near and remote sensing,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2688–2697.
- [18] E. Ferreira, M. Brito, R. Balanuk, M. S. Alvim, and J. A. d. Santos, “Brazildam: A benchmark dataset for tailings dam detection,” *arXiv preprint arXiv:2003.07948*, 2020.
- [19] D. Arthur and S. Vassilvitskii, “k-means++: The advantages of careful seeding,” Stanford, Tech. Rep., 2006.
- [20] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.
- [21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.
- [22] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *IEEE/CVF Computer Vision and Pattern Recognition*, 2015.
- [24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *IEEE/CVF Computer Vision and Pattern Recognition*, 2016.
- [25] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *IEEE/CVF Computer Vision and Pattern Recognition*, 2016.
- [26] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *IEEE/CVF Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.
- [27] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size,” *arXiv preprint arXiv:1602.07360*, 2016.
- [28] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.
- [29] X. Li, W. Wang, X. Hu, and J. Yang, “Selective kernel networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 510–519.
- [30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database,” in *IEEE/CVF Computer Vision and Pattern Recognition*, 2009.
- [31] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, “Multimodal machine learning: A survey and taxonomy,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

- [32] K. Nogueira, O. A. Penatti, and J. A. Dos Santos, “Towards better exploiting convolutional neural networks for remote sensing scene classification,” *Pattern Recognition*, vol. 61, pp. 539–556, 2017.



(a) AiRound dataset.



(b) CV-BrCT dataset.

Fig. 9: Values represent the ratio of the accuracy per class between a single VGG [22], trained/fine-tuned in one specific domain, and a fusion of two VGGs [22], trained/fine-tuned on both aerial and ground views. In the numerator of this ratio, we calculated the difference between the accuracy pos-fusion 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.