UNIVERSITAT DE BARCELONA

FUNDAMENTALS OF DATA SCIENCE MASTER'S THESIS

# Man-made Structures Detection from Space

*Author:*
Peter WEBER

*Supervisor:*
Dr. Jordi VITRIA

*A thesis submitted in partial fulfillment of the requirements for the degree of MSc in Fundamentals of Data Science*

*in the*

Facultat de Matemàtiques i Informàtica

June 30, 2019

UNIVERSITAT DE BARCELONA

# *Abstract*

Facultat de Matemàtiques i Informàtica

**Man-made Structures Detection from Space**

by Peter WEBER

With the development of affordable and recurrent remote sensing technology, we can now access frequent geospatial information in different levels of detail, ranging from 100m to 0.01m. The task of detecting various types of man-made structure and man-induced change has become a key problem in remote sensing image analysis.

In this work we focus on providing an answer to the question: what is the optimal tradeoff between resolution and cost when aiming at determining the existence of man-made structures in remote sensing images? Obtaining this value is important not only for designing optimal satellite sensors but also to use optimal data sources when developing data-based remote sensing products. At a global level, this knowledge contributes to understand the impact of our species on the planet.

Our approach is based on developing a deep learning detector to classify human impact on aerial images. In particular, we exploit recent advances of Convolutional Neural Networks (CNN) that were successfully used for object detection and scene classification. We apply transfer learning by integrating a ResNet pre-trained on ImageNet to perform image classification on datasets of few thousand aerial images that we have manually collected and annotated. Using this classification pipeline we are able to determine the existence of man-made structure with an accuracy of 95% at the best resolution.

We study the performance of our detector for resolutions ranging from 0.3m to 16m. We observe a linear decrease of the classification accuracy down to about 81% at the lowest resolution. Furthermore, we estimate the cost associated to capturing and processing satellite images. This includes the entire pipeline from building and launching a satellite to predicting human impact on images. We estimate that monitoring the entire land surface of the earth at 1m resolution amounts for about $15 million. This cost increases by about two orders of magnitude at the best resolution studied here, and decreases by about one order of magnitude at a resolution of 10m per pixel. These results could be further improved by training a CNN on a labeled large scale remote sensing dataset. Nevertheless, our results suffice for studying the expansion of human kind using satellite imagery and provide valuable information for designing optimal satellite sensors.

# *Acknowledgements*

First, we want to thank **Santi Seguí**, **Lluis Garido**, and **Eloi Puertas** to serve as experts in our thesis committee. We are very grateful to our thesis advisor **Jordi Vitrià** for close guidance and exceptionally constructive and creative ideas on how to afront the problems we encountered. His scientific instinct was indispensable in critical moments of the project. We also want to thank **Marco Bressan** from Satellogic for fruitful discussions, support with material, and advice about tools and data sources. Further, we want to thank **Javier Marin** and **Aitor Lucas** (both from Satellogic) for help with the datasets.

ADD Edu

# Contents

# Chapter 1

# Image classification pipeline

TODO: Figure captions, citations, last two paragraphs.

In the previous chapters we have introduced the key components of the approach we followed in our study: on the one hand, we have described existing satellite image datasets and introduced the actual data we will consider, and on the other, we have discussed Deep Learning, how it works and how we can use it for our problem. Now, we are ready to describe our approach: the image feature extraction, the model architecture and the training scenario.

## 1.1 Image features and transfer learning

In order to train a model based on images, some sort of features need to be extracted. Traditionally, this image feature extraction was based on a set of hand-crafted detectors aimed to detect edges, corners, blobs and other feature descriptors. Some of these detectors are the Sobel filter, Laplacian of Gaussian (LoG), Difference of Gaussians (DoG), Determinant of Hessian (DoH), SIFT [1, 2], SURF [3], Histograms of Oriented Gradients (HOG) [4] and Gabor filters.

More recent approaches to image classification using Neural Networks have benefited from the existing and increasing computational power, and deep Convolutional Neural Networks have been able to achieve higher performance than traditional models.

Yet, training a deep CNN from scratch for a particular problem requires a large and exhaustive dataset along with a huge amount of computational power. However, it has been shown that the architectures of pre-trained NN can be reused for other purposes and achieve equally great performance. This is known as **Transfer Learning**. Figure 1.1 schematizes this idea. These pre-trained architectures can be re-purposed by reusing the learned weights and either replacing the final layers of the net by some other classifier, or even fine-tuning all the layers for the specific problem. In any case, the initial layers of the Neural Network provide a great image feature extractor.

In the next section we describe our approach using transfer learning from a ResNet architecture.

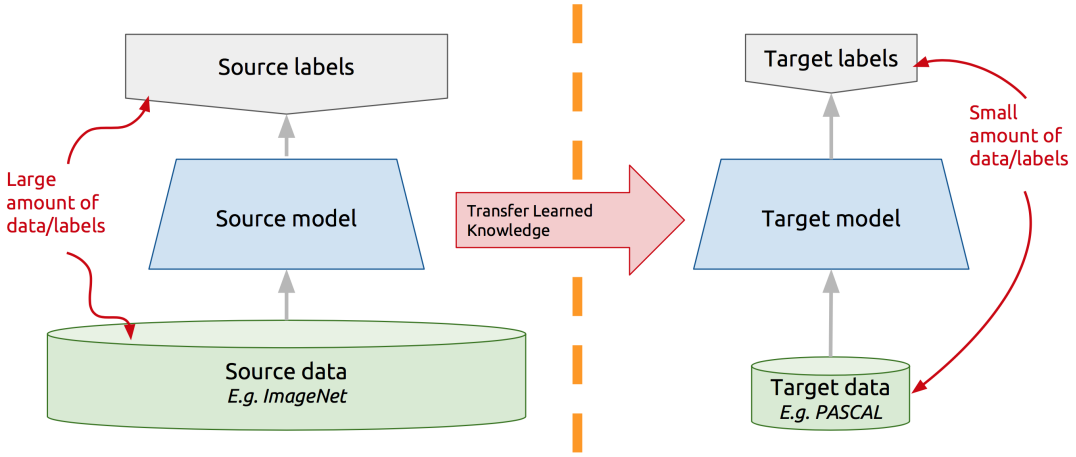


FIGURE 1.1: **Transfer Learning**: a model learned from a large dataset can be transferred and reused for another purpose. [**McGuinness2017**]

## 1.2 Proposed architecture

As described before (Sections **??** and 1.1), we can use for our problem a pre-trained ResNet with our own final classification layers. Hence, the architecture we propose for our problem consists of the activation layers of a ResNet, which act as the feature extractors of our images, followed by a shallow classifier made of a single dense (fully connected) layer. Figure 1.2 gives a schema of this approach.

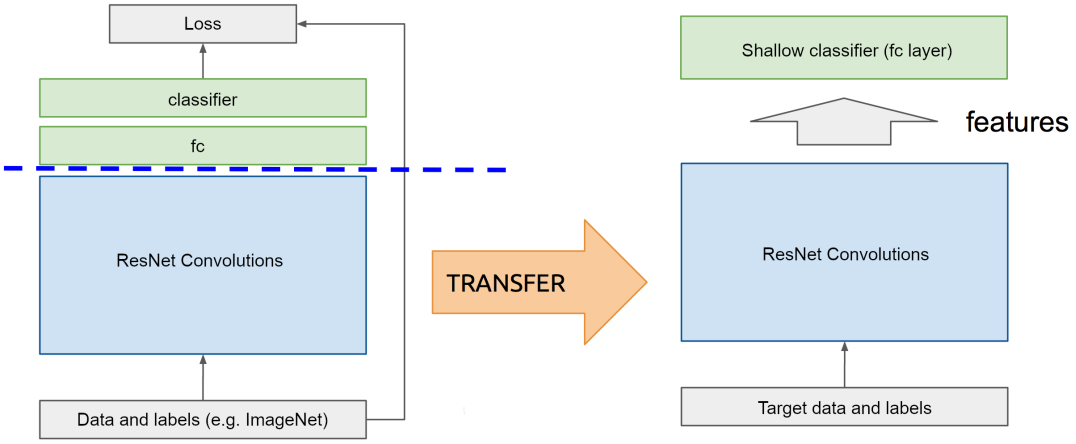


FIGURE 1.2: **Transfer Learning from a ResNet** (figure adapted from [**McGuinness2017**])

### 1.2.1 ResNet activations

The ResNet we consider (ResNet50) has a total of 49 activation layers, so the output at each of them is different. Initial layers are able to recognize edges, textures and patterns while keeping an image size similar to the input. On the other hand, deeper activation layers show convolutions of higher order hierarchical structures. These structures are more complex and therefore the ResNet authors use much more channels in deeper layers. At the same time they decrease the spatial image size by

applying stride 2 whenever they increase the number of filters. The purpose of the latter is to keep the number of parameters manageable.

For instance, for an input image of (tensor) size $512 \times 512 \times 3$ (a $512x512$ image with 3 RGB channels), the output of the first activation layer is of size $256 \times 256 \times 64$, the $10^{th}$ gives a $128 \times 128 \times 256$ tensor, and the last $49^{th}$ activation layer outputs $16 \times 16 \times 2048$. For our purpose, we will consider the final output of the ResNet ($49^{th}$ activation layer), although this could be further investigated and discussed.

Figures 1.3 - 1.6 shows 8 activation maps both for the $10^{th}$ and the $49^{th}$ layer for samples of different categories in the dataset. Some of the $10^{th}$ activations are particularly sensitive to edges, shadows, or textures, which later translate into more abstract outputs at the $49^{th}$ layer. Here one can observe that only a very selected number of neurons have fired, namely when a very specific feature was found on the corresponding position in the input image.

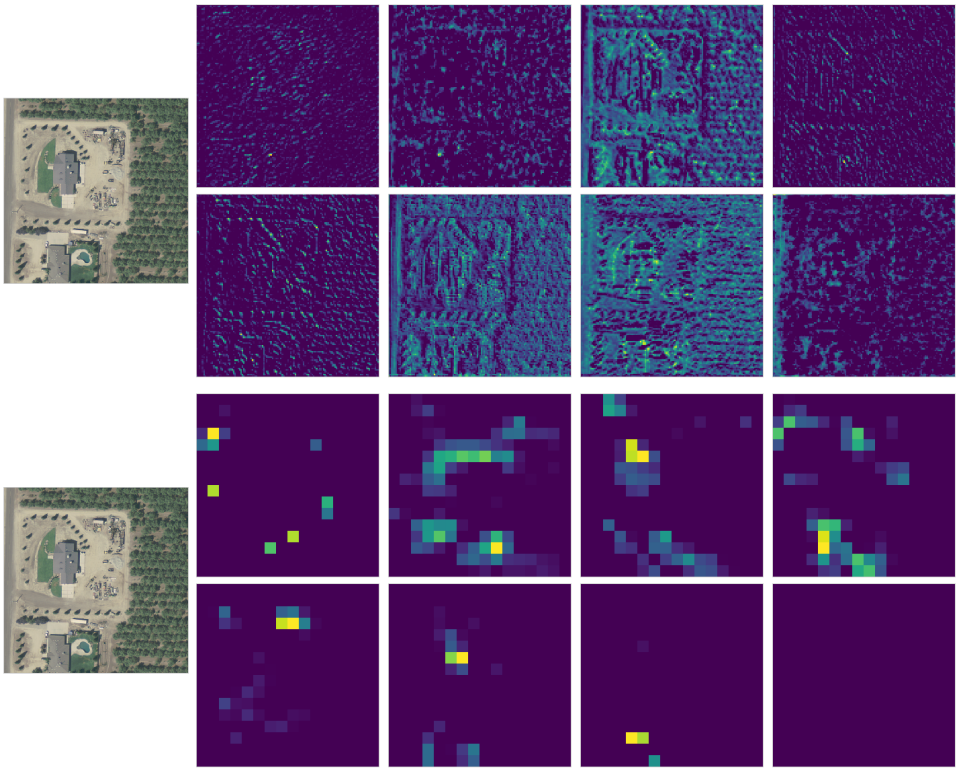

FIGURE 1.3: **ResNet activations of an Agriculture image: $10^{th}$ layer (top) and final layer, $49^{th}$ (bottom).**

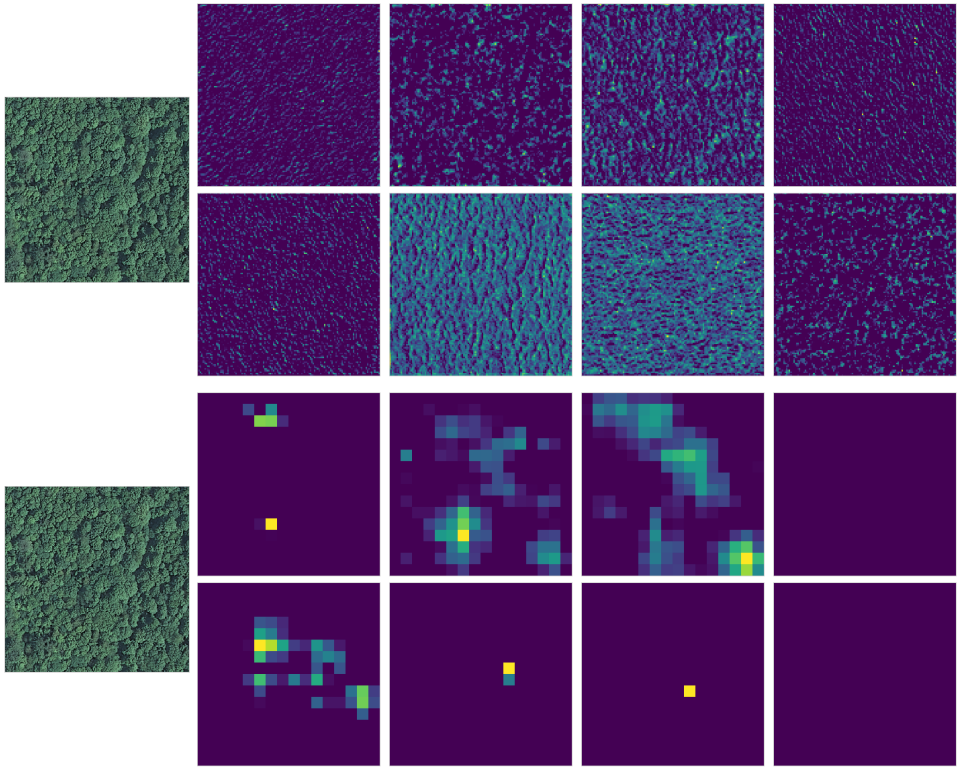

FIGURE 1.4: **ResNet activations of a Forest-woodland image: $10^{th}$ layer (top) and final layer, $49^{th}$ (bottom).**

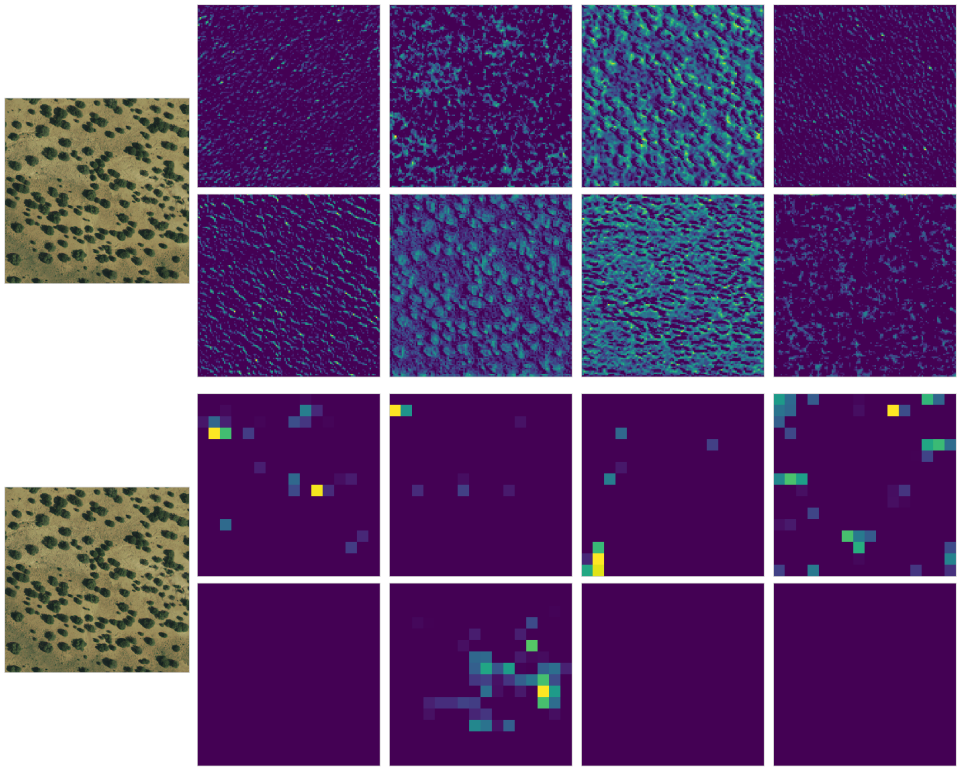

FIGURE 1.5: **ResNet activations of a Semi-desert image: $10^{th}$ layer (top) and final layer, $49^{th}$ (bottom).**

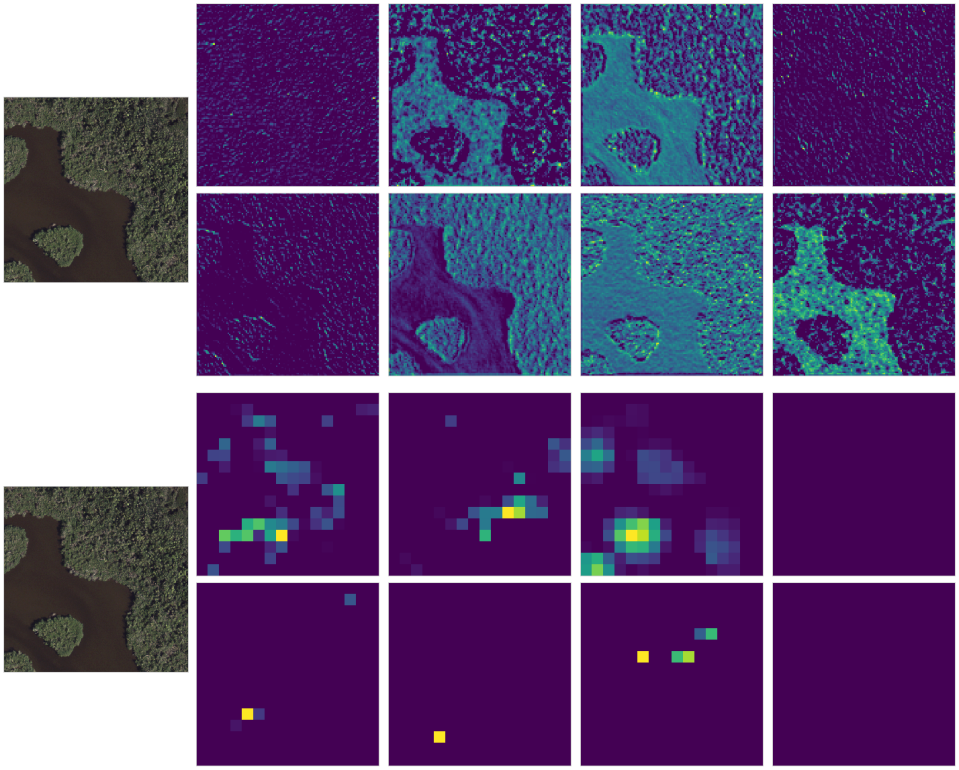

FIGURE 1.6: **ResNet activations of a Shrubland-grassland image: $10^{th}$ layer (top) and final layer, $49^{th}$ (bottom).**

### 1.2.2 Complete architecture

For our purpose we considered the last ($49^{th}$) activation layer of the ResNet as the features of our images. These features can be extracted and saved on disk in order to speed up the process (as we did), or computed each time, and then passed through a simple fully connected Neural Network.

We terminated the ResNet architecture with a NN that consists of a single dense layer of 100 or 200 neurons with ReLU activation, followed by a single dense node with a Sigmoid activation acting as the classifier. This model is trained on the dataset with RMSprop optimizer [5] and a binary cross-entropy loss function. The same architecture is used and trained separately for each of the resolutions considered.

This architecture (see Fig. 1.2) has been implemented with Python and Keras. Figure 1.7 shows the model build, while in the following section we describe the complete training pipeline in more detail.

### 1.2.3 Training pipeline and experiments

As introduced in previous chapters, our goal with this model is to study how feasible it is (technically and costly speaking) to detect different kind of human impact on satellite images, and how this detection behaves for different image resolutions. To do so, we build two datasets of annotated images at base resolutions of $0.3m$ and $1m$

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 100)               52428900
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 101
=================================================================
Total params: 52,429,001
Trainable params: 52,429,001
Non-trainable params: 0
_________________________________________________________________
```

FIGURE 1.7: **Model build with Keras**

(see Chapter **??**), which we later downgrade to a range of resolutions suitable for our study.

Starting from an image at its base resolution and size ($512 \times 512$ pixels), the downgrade process consists of downsampling (removing) pixels, so the image becomes smaller. Therefore, this imposes a limit on how far a given dataset can be downgraded, as the CNN ResNet model requires a minimum input size of $32 \times 32$ pixels due to the application of stride 2 at multiple layers [6]. Note that during the downsampling process the physical area displayed by the image is not changed. Tables 1.8 and 1.9 show the resolutions and sizes considered for the two datasets.

| **resolution (m)** | 0.3 | 0.6 | 0.9 | 1.2 | 1.5 | 1.8 | 2.1 | 2.4 | 2.7 | 3.0 | 3.3 | 3.6 | 3.9 | 4.2 | 4.5 | 4.8 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **size (pixels)** | 512 | 256 | 171 | 128 | 102 | 85 | 73 | 64 | 57 | 51 | 47 | 43 | 39 | 37 | 34 | 32 |

FIGURE 1.8: **Relation between resolution and size for the** $0.3m$ **dataset**. Size is the width (and height) of a square image.

| **resolution (m)** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **size (pixels)** | 512 | 256 | 171 | 128 | 102 | 85 | 73 | 64 | 57 | 51 | 47 | 43 | 39 | 37 | 34 | 32 |

FIGURE 1.9: **Relation between resolution and size for the** $1m$ **dataset**. Size is the width (and height) of a square image.

Next, for each of the datasets and downgraded resolutions we want to train a separate model and assess its performance. To this end, we consider the following pipeline for each of the datasets:

1. Load the original images (at the original resolution) from disk along with the human impact labels and categories.

2. Downsample the images to the desired resolution (from the lists in 1.8 and 1.9)

3. Compute the ResNet activations (at the $49^{th}$ activation layer) of the resulting downgraded images. These activations can be saved to disk for later use if needed.

4. Consider a stratified KFold split of the dataset (with 8 splits) for cross-validation. That means, the dataset is split into 8 sets with labels 0-1 equally distributed. Note that label 1 here represents the class with clear human impact, in contrast to the convention in chapter **??** (where it was label 2). Each cross-validation fold consists of around 300 images for the $0.3m$ dataset, and around 200 samples for the $1m$ dataset.

5. Train the model (Fig. 1.7) separately for each combination of the 7 training sets considering 30 epochs. The remaining set is used as a validation set to assess the accuracy. After training on all folds, the results of the 8 experiments are averaged in order to obtain more consistent measures.

6. Repeat the process for all downgraded resolutions.

Note that the splitting parameters of the cross-validation, the model complexity and the training epochs could be further analyzed in order to find the best combination for each of the resolutions tested. Actually, all the experiments consist of training NN models for two datasets, each with 16 resolutions and 8 folds per resolution, so every fine-tuning (like changing the size of the NN) implies several executions with some variability on each stage.

Nonetheless, as already mentioned in the introduction, the final goal of the experiments is to have a statistical reference of how well the models can be trained, and not to achieve the highest accuracy for each scenario. In order to do so, a larger dataset would be needed, with more well defined categories and objects, and a clear goal of what needs to be modeled.

Modify these two paragraphs a bit! The plots in Figures 1.10 and 1.11 (on the next pages) show the convergence of some of the models trained, for the different datasets and resolutions (base and lowest resolutions), and considering one of the cross-validation folds only. Also, two architectures for the dense layer have been tested: 100 neurons and 200 neurons.

In general, the NN is able to converge and achieve a good accuracy (as shown in the plots), although for some particular splits of the data, it fails to converge and stays in a low accuracy point. This is probably due to the fact of having a relative small dataset. Hence, these particular folds are not going to be considered when computing the final averaged results.

In the next chapter we discuss in much more detail the results obtained from all these experiments.

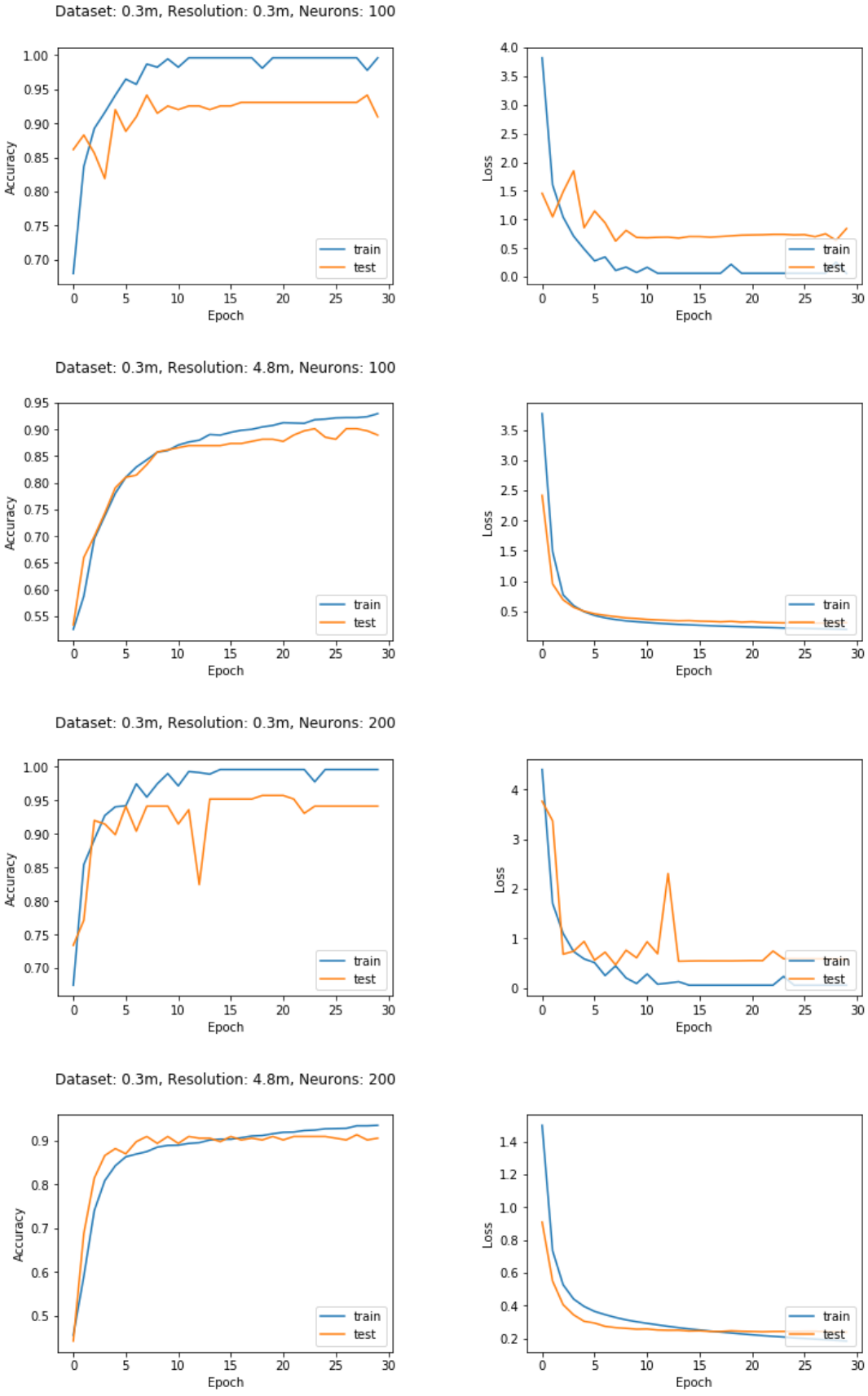


FIGURE 1.10: **Convergence plots of some experiments on the** $0.3m$ **dataset, for base (**$0.3m$**) and lowest (**$4.8m$**) resolutions, and considering** 100 **and** 200 **neurons.** The models converge more smoothly for lower resolutions, were there are fewer parameters to be fitted.

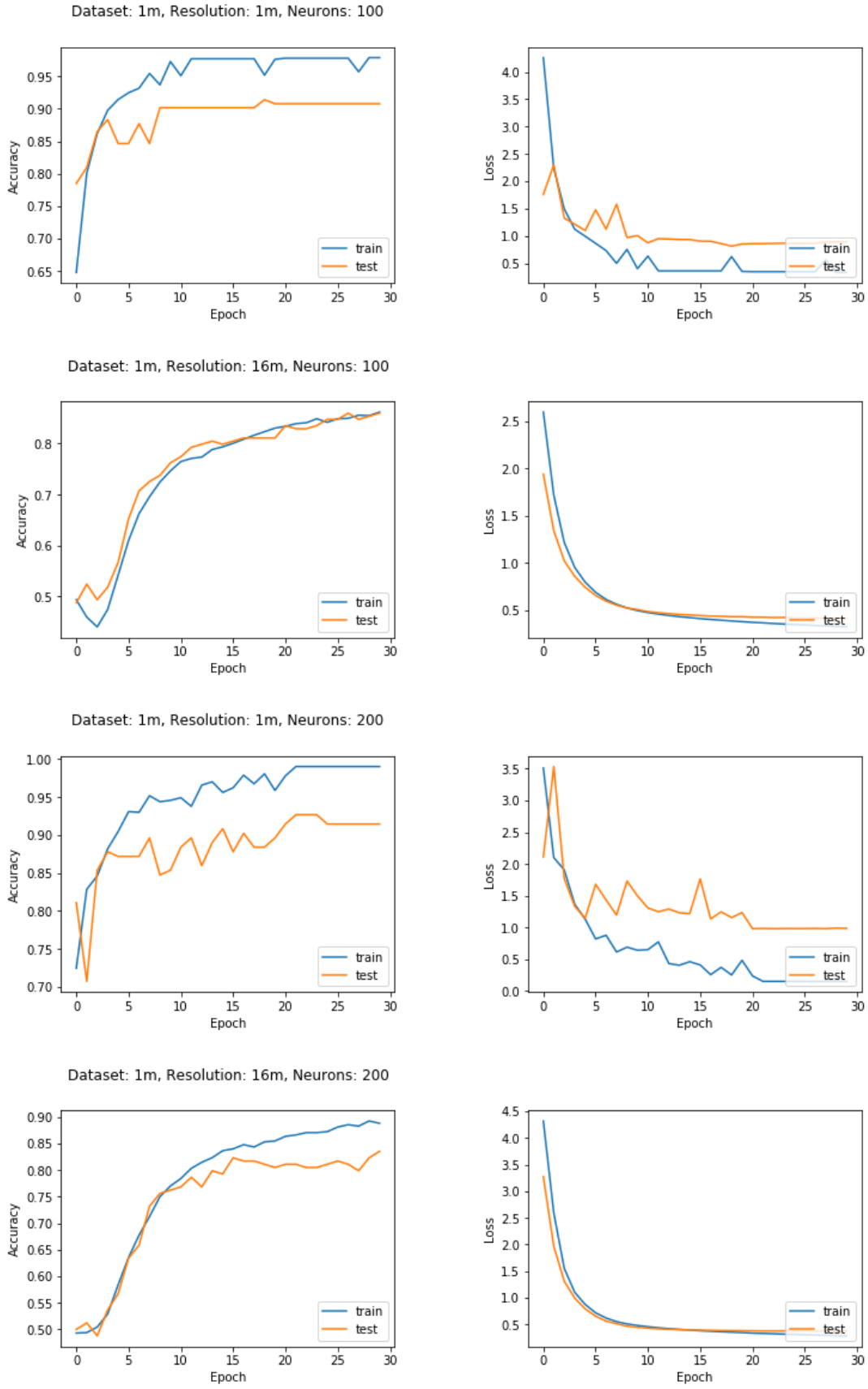


FIGURE 1.11: **Convergence plots of some experiments on the $1m$ dataset, for base ($1m$) and lowest ($16m$) resolutions, and considering 100 and 200 neurons.** The models converge more smoothly for lower resolutions, were there are fewer parameters to be fitted.

# Chapter 2

# Detection of man-made structures with satellites

In this chapter we discuss the results obtained in this thesis. We first evaluate for different resolutions the performance of the Deep Learning pipeline that was described in the previous chapter. Subsequently we discuss several examples of correctly and wrongly classified images. In the last part of the chapter, we estimate the cost associated with providing earth observation data involving satellites.

## 2.1 Man-made structures detection at different scale

Having indicated in the last chapter that the developed transfer learning approach achieves remarkable accuracies, we now turn to a more detailed and explicit study. We evaluate how the trained model, that is based on a pre-trained ResNet as feature extractor for aerial images, allows to properly discriminate the existence of human impact for different resolutions, different datasets, different categories, different ResNet terminations, and different cross-validation folds. The results for all these experiments have been aggregated and are shown in Fig. 2.1 as well as summarized in tables A.1-A.4 in the appendix.

The highest accuracy, 95.7%, is achieved at a resolution of 0.6m for the dataset with 0.3m base resolution and a fully connected layer with 200 neurons. The lowest accuracy, 78.1%, is obtained at 14m resolution for the architecture with the fully connected layer with 100 neurons. As expected, the accuracy decreases roughly linearly as the resolution becomes worse. However, between 0.3m and 8m the accuracies are consistently above 88%, and only drop below 85% at resolution values higher than 8m. At these high values for the resolution the model is not able anymore to detect subtle elements of man-made structures, which will be discussed in more detail in section 2.2.1.

We further note that the accuracy on the base resolution (0.3m and 1m) is always slightly worse than the accuracy at the next resolution. We suspect that this anomaly is related to the fact that the input image size is quite large ($512 \times 512$), which makes the dense layer more complex to train. Indeed, Figs. 1.10 and 1.11 in chapter 1 suggest that the models struggle to be optimized. More sample images would be

required in order to compensate for the complexity and hence to achieve a higher accuracy.

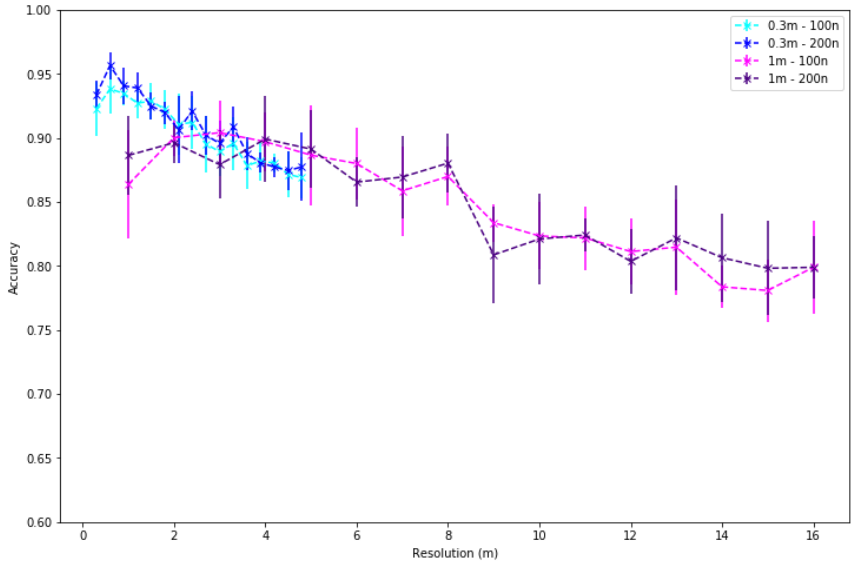


FIGURE 2.1: **Accuracy at each resolution for all datasets and architectures.** The blue (purple) lines correspond to the 0.3m (1m) dataset and lighter (darker) color belongs to a 100 (200) neurons architecture. The 0.3m (1m) dataset was trained and evaluated on a total of ∼2000 (∼1300) images. Note that, however the base resolution of the 0.3m dataset was trained and evaluated with only 1500 images, because Colab's memory requirements did not allow to process more images with size $512 \times 512$. The vertical lines represent the variability (standard deviation) for the different folds of the 8-fold cross-validation.

The architecture with the fully connected layer with 200 neurons performs slightly better than the one with 100 neurons, but accuracy improvements are generally in the range of 1% - 2%. Hence, we will focus on the 200 neurons architecture from now on. Another important observation is that at the region where the resolutions overlap between the two datasets also the accuracies agree i.e. they are within the error bars. This indicates that both datasets are comparable and can be considered together. Note that the error bars of the 1m dataset (∼1300 images) are about twice as large as the error bars for the 0.3m dataset (∼2000 images). The larger variation for the 1m dataset is attributed to the different sizes of the datasets, and hence significantly smaller out-of-fold datasets.

## 2.2 Man-made structures detection for different categories

Let us consider how the accuracies behave for each of the USGS land use categories. As discussed in section **??**, these categories are rough approximations of the kind of terrain and human impact, but we can not assure with absolute certainty that every image is assigned to the correct category. Fig. 2.2 shows a comparison of the accuracy of each category with respect to the aggregated accuracy. For the 0.3m dataset (and 200 neurons model) we observe that there are substantial differences between categories.

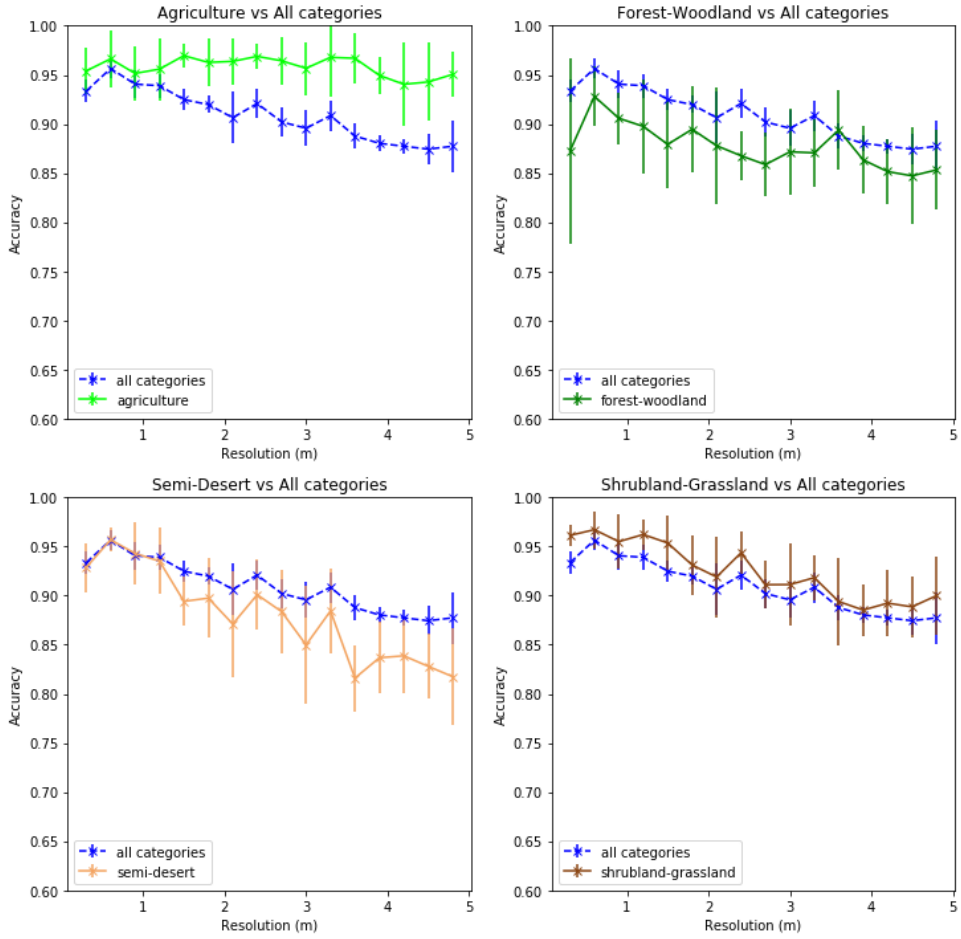


FIGURE 2.2: **Comparison of accuracies between each category and all categories aggregated.** Here the model is trained over all categories, on the 0.3m dataset, and tested on each category separately (in the case of a single category).

These plots have been obtained once the model was trained for all the categories. Then, the accuracy on the validation set was computed for each individual category and over all images in the set. The validation set in each iteration of the cross-validation was small, consisting of few hundred images, and a homogeneous representation of the categories was not imposed (validation samples were picked randomly, only preserving proportion between man-made vs. natural structures), which explains the large variability for each experiment (large error bars).

Although the category is not taken into account when training, we can see that the models are capable of detecting agriculture-related human impact with an accuracy consistently above 95%, without being affected by the drop in resolution. This is not surprising since this category contains only images with man-made structures, and therefore the algorithm might indeed have learned image textures instead of features related to human impact. The shrubland-grassland category has about 2% higher accuracy than all categories, while the resolution dependence is similar. The forest-woodland category is about 4% worse than the overall also showing a similar resolution dependence. Finally, the semi-desert category starts

off at the same accuracy as all categories at high resolutions but drops siginificantly faster down to about 82% (all categories  88%). Overall we conclude, that each of the three non-agriculture categories encode slightly different image features, and therefore the algorithm behaves slightly different.

Similar results are obtained on the 1m dataset, which are shown in Fig. 2.3. The models are able to achieve a remarkable accuracy when detecting agriculture-related human impact (consistently above 95%), and the three non-agriculture categories behave similar to the aggregated view. However, we note a drop of accuracy at 8m resolution by about 5% across all non-agriculture categories, which is an indicator that many man-made structures in the dataset have a characteristic size of about 8m.

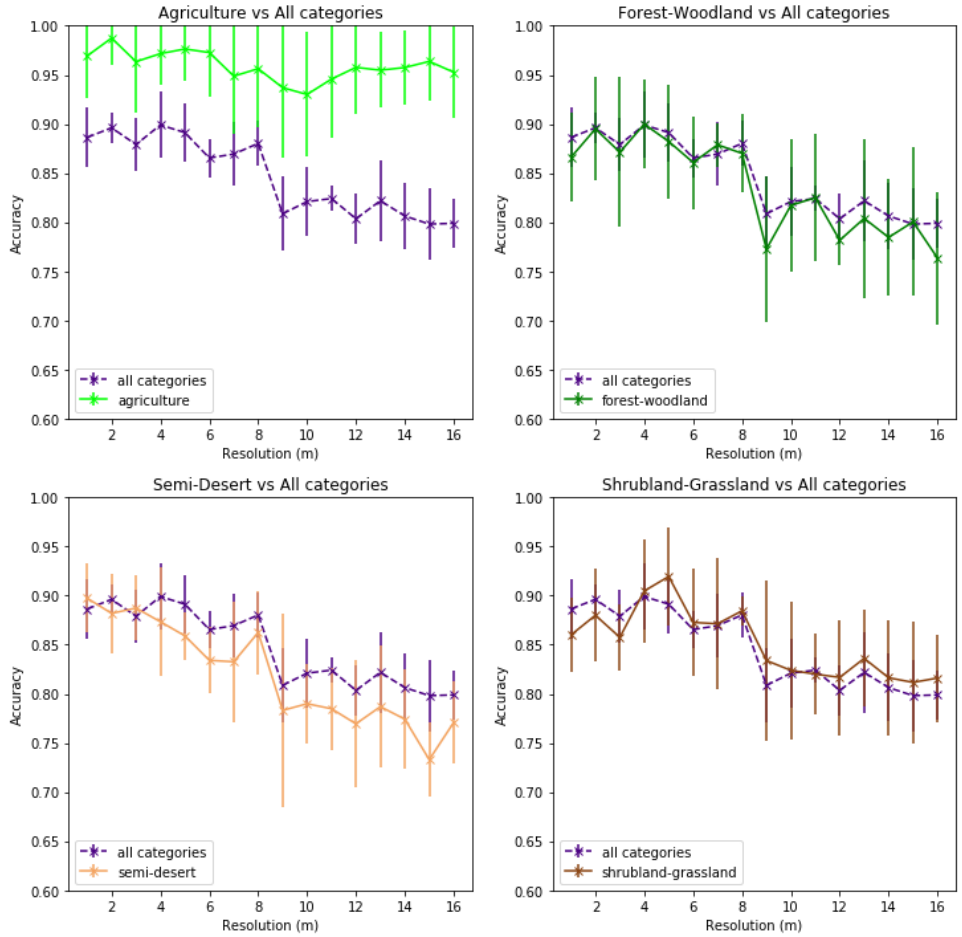


FIGURE 2.3: **Comparison of accuracies between each category and all categories aggregated.** Here the model is trained over all categories, on the 1m dataset, and tested on each category separately (in the case of a single category).

### 2.2.1 Performance on selected images

In this subsection we will illustrate the algorithm behaviour by considering concrete examples of correctly and wrongly classified images. In Figs. 2.4 and 2.5 we show examples for the 0.3m dataset at base resolution (for one of the cross-validation folds), respectively. The first set of samples (Fig. 2.4) shows that the model accurately

detects clear human impact related to agriculture (2nd picture in the second row) and paths. On the other hand, the second set (Fig. 2.5) indicates that it might fail to detect it when the impact is subtle, covering a small region of the image, or when it can even be confused with natural structures (or vice versa). Note that in this subsection we refere to images with clear human impact as images with label 1 (in contrast to chapter **??** where they were defined as label 2).

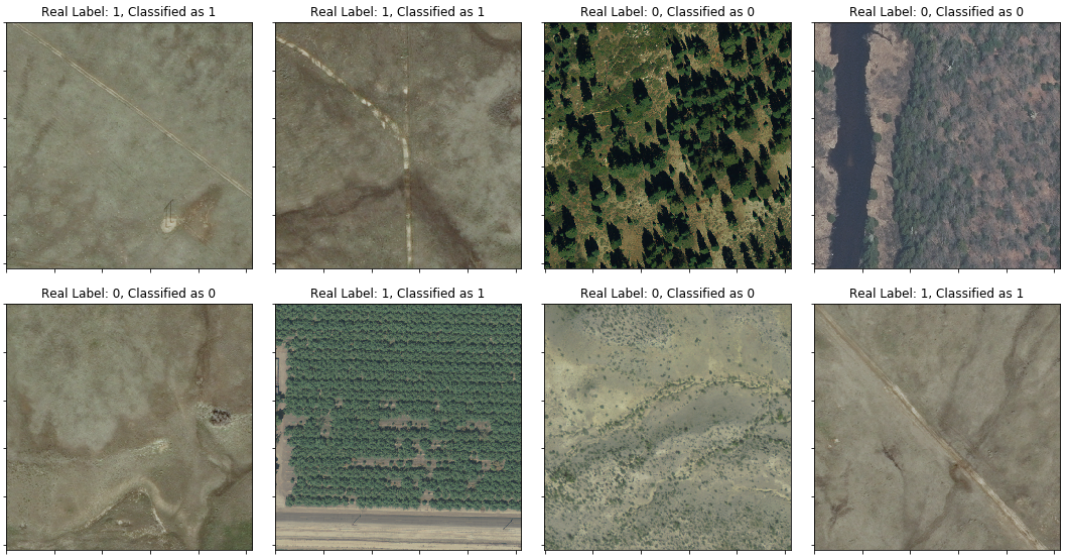


FIGURE 2.4: **Examples of correctly classified images at base resolution of 0.3m dataset.** The title of each image marks whether an image shows man-made structures, and indicates the output of the classifier.

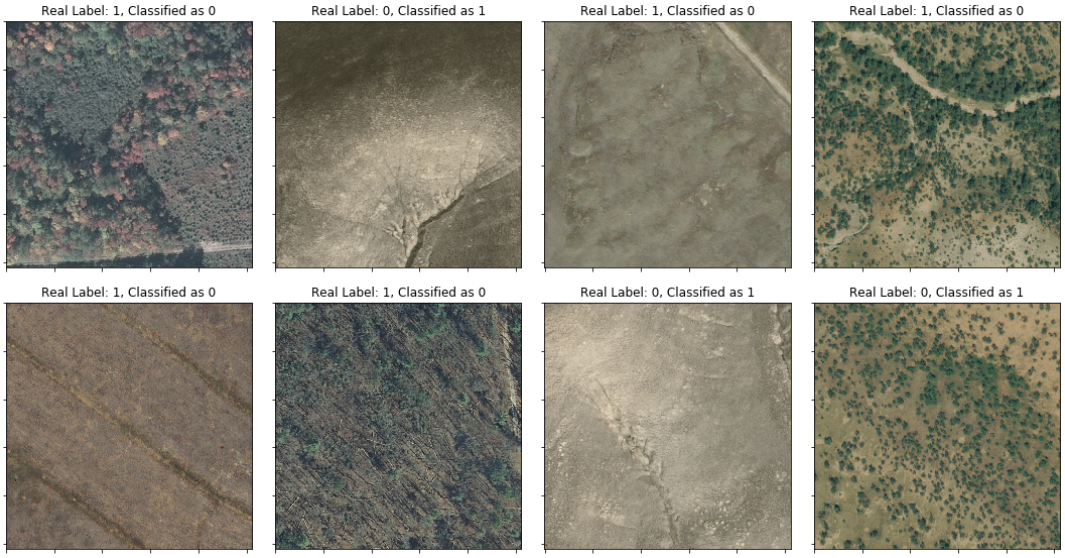


FIGURE 2.5: **Examples of wrongly classified images at base resolution of the 0.3m dataset.**

A similar analysis can be done for the last resolution, 4.8m, of the 0.3m dataset, which is shown in Figs. 2.6 and 2.7. The first set of images (Fig. 2.6) indicates that the model detects human impact when it is still evident, even with the low resolution.

However, the second set of images (Fig. 2.7) implies that it commits errors when the evidence associated to man-made structures is lost with the downgrade process. Similarly, it might classify as man-made structures patterns that are indeed natural.

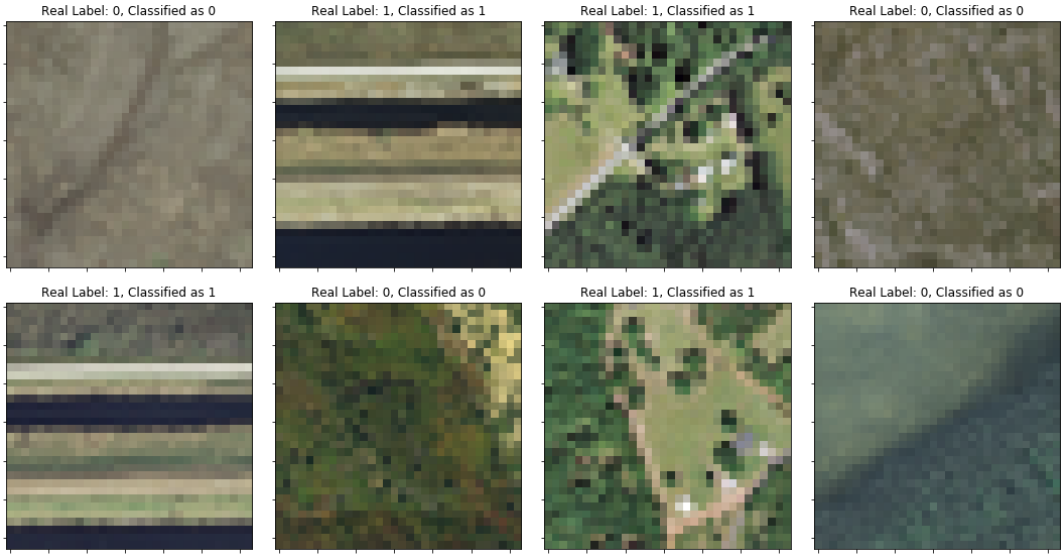


FIGURE 2.6: **Examples of correctly classified images at last resolution, 4.8m, of 0.3m dataset.**

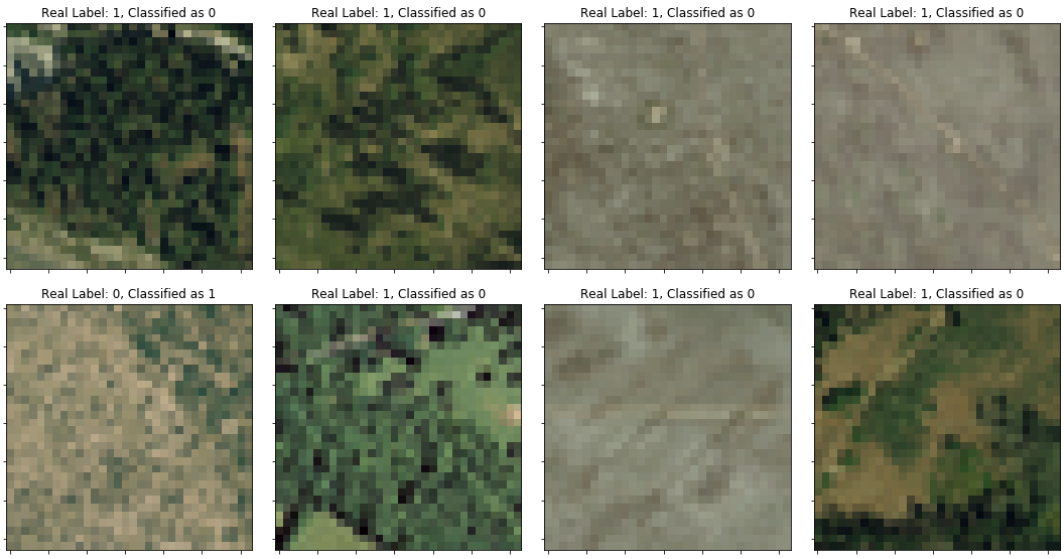


FIGURE 2.7: **Examples of wrongly classified images at the last resolution, 4.8m, of 0.3m dataset.**

These observations are demonstrated in Fig. 2.8, in which we show images that are correctly classified at 0.3m resolution (top row) but wrongly classified at 4.8m. The first and third pair of images demonstrate that, when the human impact is subtle, the model missed it in the downgraded resolution. Conversely, non human activity can also be misclassified at lower resolutions (second and fourth images).

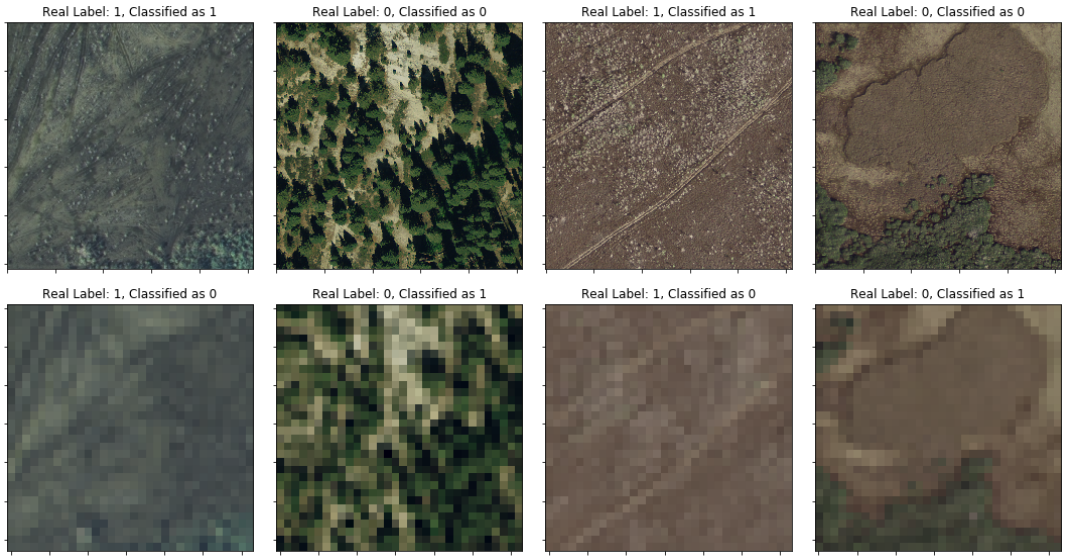


FIGURE 2.8: **Correctly and wrongly classified images at different resolutions.** Here the first row displays images in their base resolution, $0.3m$. These images were correctly classified by the model. The same images at a resolution of $4.8m$ (second row) were wrongly classified by the model.

Finally, we investigate how the model behaves with images where human impact is very minimal. For this purpose, we consider the images with the intermediate label (label 1 in Chapter **??**, Figure **??**). The model has never been faced with these images, so this can give a good perception of whether the model has been able to learn relevant features of human impact. Figure 2.9 shows several images with the label, that the model predicted, in the title. Even if man-made structures in these pictures are small, the model is able to detect straight lines and shapes as human activity.

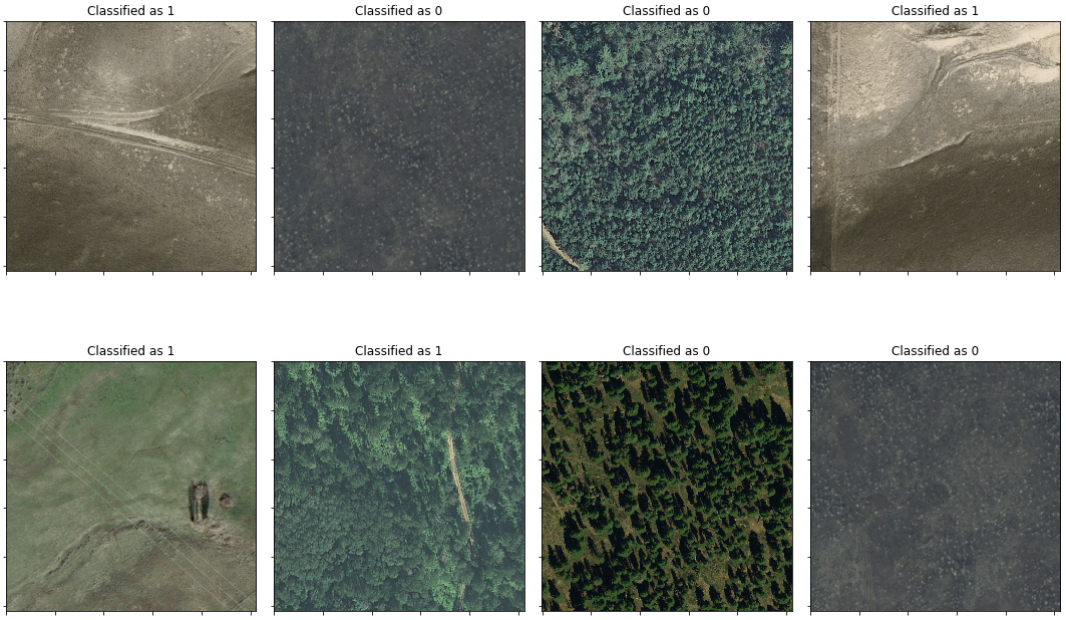


FIGURE 2.9: **Examples of predictions made for images with subtle human impact**. Images showing minimal human impact were referred to as label 1 images in Chapter **??**. Note that these images were not used when training the model.

## 2.3 Cost estimation

We discuss the financial cost associated to building and launching a satellite, and to renting infrastructure for performing the entire image processing pipeline. We further study the cost as a function of pixel resolution. However, our estimates are very rough approximations because many factors are involved and large variations occur between them. To give an example, choosing one material over the other might change the cost of manufacturing and launching a satellite by one order of magnitude. It is also completely different to have a satellite for 3 years in space, or to target a lifespan of 20 years.

Having this in mind, we follow laws from physics to estimate the dependency of the satellite cost on resolution. First, the cost of launching a satellite into the orbit scales linearly with its mass [7], which is given by the amount of fuel needed. Second, the mass of the satellite scales quadratic with resolution so that overall we obtain a cubic dependency for launching a satellite into space. The latter increase in cost is associated with the optical instruments used. As a reference for the satellite cost we use a Skysat satellite from Planet [8] that has a resolution of about 1m and a value of $30 million. This amount was provided to us by Satellogic and includes construction, launch and maintenance during the satellite's lifespan.

Our final goal is to give an estimation of the expenditure to monitor once the entire surface of the earth (about 149 million $km^2$). To this end, we multiply the satellite cost by the ratio: time needed to scan the earth over the satellite's lifespan. Further, a satellite can map 1 million $km^2$ at 1m resolution in 4.2 days [9]. We hence

can calculate the satellite cost per km$^2$. Assuming a lifespane of 10 years we have $area = 10^6 \times \frac{10 \cdot 365}{4.2}\ km^2$ so we obtain $cost\ satellite\ per\ km^2 = cost\ satellite/area \approx 0.035\ \$/km^2$.

| description | cost | unit | cost (\$/km$^2$) | cost (\$/pixel) |
|---|---|---|---|---|
| process raw data | | | 0.004 | $4 \times 10^{-9}$ |
| hot storage | $72 \times 10^{-6}$ | \$/(km$^2$/month) | 0.000864 | $8.64 \times 10^{-10}$ |
| cold storage | $36 \times 10^{-6}$ | \$/(km$^2$/month) | 0.000432 | $4.32 \times 10^{-10}$ |
| archive storage | $9 \times 10^{-6}$ | \$/(km$^2$/month) | 0.000108 | $1.08 \times 10^{-10}$ |
| download data | 8 | \$/Gb | 0.021 | $2.1 \times 10^{-8}$ |
| serving to final client | 0.09 | \$/Gb | 0.000236 | $4.7232 \times 10^{-10}$ |
| prediction (AWS) | 0.05 & $\sim$6 | \$/h & s/km$^2$ | 0.00145 | $1.45 \times 10^{-9}$ |

TABLE 2.1: **Costs for image data processing.** All costs except the prediction are provided by Satellogic.

Another cost intensive block when capturing satellite imagery involves image data processing for which the cost scales quadratic with resolution. For example, an operation that costs 100\$/km2 at 1m resolution will cost only 1\$/km$^2$ at 10m resolution. The data processing step consists of multiple parts: transformation of raw data into image pixels, storing data in a hot, cold, and archive storage, downloading data from the satellite, serving it to the final client, and in our case predicting human impact. These costs are summarized for 1m resolution in table 2.1. Note that we used the conversion factor 0.002624 for an image to convert from Gigabytes to km$^2$ (2$\times$ compressed) and we assume 12 months of data storage.

The prediction step is estimated by loading 4 images that each have an area of about $500 \times 500 m^2$, calculating the ResNet activations of the final layer, and predicting the class using the models trained in chapter 1 in an ensemble fashion. This part amounts to a processing time of about 6s for an area of 1km$^2$, which can be converted into costs per km$^2$ assuming 0.05\$/h of AWS EC2 compute [10].

To finally obtain the dependence of the resolution on the total financial cost we sum the data cost per km$^2$ and the satellite cost per km$^2$ at 1m resolution, and convert to cost per pixel ($\times 10^{-6}$). We then multiply with the number of pixels necessary to cover the entire surface of the earth. Here the satellite cost per km$^2$ is a cubic function and the earth surface in pixel is a quadratic function in resolution. The result is shown in Fig. 2.10. We obtain a cost of about \$15 million dollars at 1m resolution with a very steep slope towards better resolutions. At 0.3m resolution the cost is two orders of magnitude higher than at 1m while for worse resolutions the cost decreases by two orders of magninute when the resolution is about 10m. We conclude that for worse resolutions the data processing cost is the dominating cost whereas for very good resolutions the satellite cost dominates.

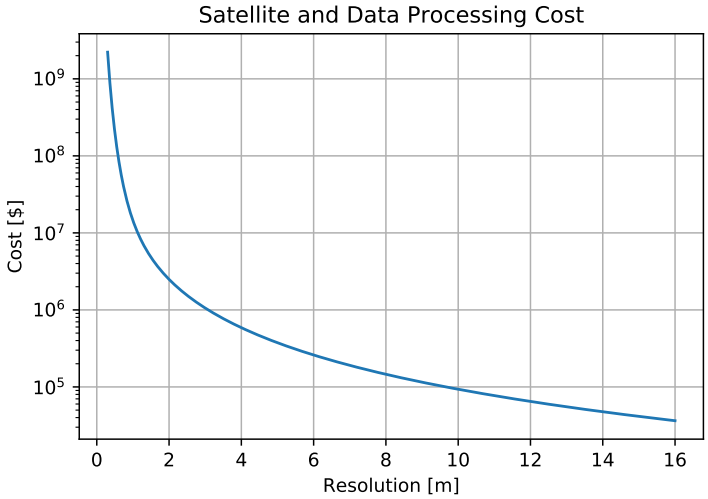


FIGURE 2.10: **Satellite and data processing cost.** The total financial cost to capture images with a satellite and process the data as function of resolution.

# Appendix A

# Tables

The following tables contain the aggregated results of the cross-validations performed for each dataset and downgraded resolution. Folds where the model was not able to converge have been removed.

| resolution (m) | All categories | | agriculture | | forest-woodland | | semi-desert | | shrubland-grassland | |
|---|---|---|---|---|---|---|---|---|---|---|
| | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std |
| 0.3 | 0.9226 | 0.0211 | 0.9850 | 0.0248 | 0.8630 | 0.0624 | 0.8882 | 0.0453 | 0.9561 | 0.0228 |
| 0.6 | 0.9383 | 0.0190 | 0.9699 | 0.0216 | 0.8779 | 0.0269 | 0.9231 | 0.0410 | 0.9748 | 0.0198 |
| 0.9 | 0.9347 | 0.0096 | 0.9439 | 0.0244 | 0.8944 | 0.0561 | 0.9327 | 0.0216 | 0.9578 | 0.0212 |
| 1.2 | 0.9271 | 0.0120 | 0.9488 | 0.0374 | 0.9061 | 0.0373 | 0.9150 | 0.0328 | 0.9399 | 0.0231 |
| 1.5 | 0.9288 | 0.0139 | 0.9622 | 0.0285 | 0.8679 | 0.0383 | 0.9144 | 0.0309 | 0.9604 | 0.0215 |
| 1.8 | 0.9224 | 0.0154 | 0.9678 | 0.0311 | 0.8864 | 0.0650 | 0.9014 | 0.0363 | 0.9371 | 0.0221 |
| 2.1 | 0.9108 | 0.0241 | 0.9618 | 0.0168 | 0.8749 | 0.0454 | 0.8885 | 0.0571 | 0.9216 | 0.0371 |
| 2.4 | 0.9120 | 0.0206 | 0.9573 | 0.0280 | 0.8686 | 0.0590 | 0.8867 | 0.0358 | 0.9352 | 0.0263 |
| 2.7 | 0.8952 | 0.0223 | 0.9620 | 0.0216 | 0.8631 | 0.0191 | 0.8510 | 0.0350 | 0.9117 | 0.0496 |
| 3.0 | 0.8893 | 0.0189 | 0.9571 | 0.0067 | 0.8717 | 0.0197 | 0.8340 | 0.0472 | 0.9055 | 0.0455 |
| 3.3 | 0.8957 | 0.0209 | 0.9539 | 0.0311 | 0.8723 | 0.0479 | 0.8529 | 0.0341 | 0.9121 | 0.0425 |
| 3.6 | 0.8784 | 0.0184 | 0.9415 | 0.0394 | 0.8727 | 0.0442 | 0.8199 | 0.0348 | 0.8935 | 0.0299 |
| 3.9 | 0.8819 | 0.0153 | 0.9456 | 0.0186 | 0.8738 | 0.0330 | 0.8397 | 0.0364 | 0.8796 | 0.0178 |
| 4.2 | 0.8804 | 0.0070 | 0.9389 | 0.0305 | 0.8525 | 0.0267 | 0.8444 | 0.0533 | 0.8961 | 0.0222 |
| 4.5 | 0.8715 | 0.0179 | 0.9383 | 0.0390 | 0.8445 | 0.0308 | 0.8301 | 0.0340 | 0.8821 | 0.0206 |
| 4.8 | 0.8690 | 0.0160 | 0.9508 | 0.0157 | 0.8569 | 0.0517 | 0.7798 | 0.0456 | 0.9035 | 0.0310 |

TABLE A.1: **Aggregated cross-validation results for** $0.3m$ **dataset,** 100 **neurons**

| resolution (m) | All categories | | agriculture | | forest-woodland | | semi-desert | | shrubland-grassland | |
|---|---|---|---|---|---|---|---|---|---|---|
| | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std |
| 0.3 | 0.9335 | 0.0113 | 0.9540 | 0.0234 | 0.8725 | 0.0946 | 0.9286 | 0.0253 | 0.9613 | 0.0105 |
| 0.6 | 0.9565 | 0.0104 | 0.9663 | 0.0296 | 0.9281 | 0.0304 | 0.9569 | 0.0120 | 0.9671 | 0.0189 |
| 0.9 | 0.9406 | 0.0139 | 0.9515 | 0.0273 | 0.9059 | 0.0264 | 0.9430 | 0.0314 | 0.9550 | 0.0273 |
| 1.2 | 0.9390 | 0.0122 | 0.9559 | 0.0317 | 0.8976 | 0.0483 | 0.9351 | 0.0336 | 0.9622 | 0.0153 |
| 1.5 | 0.9249 | 0.0106 | 0.9696 | 0.0116 | 0.8793 | 0.0443 | 0.8942 | 0.0250 | 0.9536 | 0.0280 |
| 1.8 | 0.9198 | 0.0087 | 0.9627 | 0.0243 | 0.8947 | 0.0439 | 0.8975 | 0.0404 | 0.9312 | 0.0304 |
| 2.1 | 0.9065 | 0.0263 | 0.9638 | 0.0238 | 0.8780 | 0.0597 | 0.8711 | 0.0541 | 0.9190 | 0.0415 |
| 2.4 | 0.9209 | 0.0152 | 0.9689 | 0.0129 | 0.8677 | 0.0247 | 0.9009 | 0.0354 | 0.9435 | 0.0211 |
| 2.7 | 0.9021 | 0.0150 | 0.9644 | 0.0248 | 0.8587 | 0.0327 | 0.8837 | 0.0426 | 0.9111 | 0.0247 |
| 3.0 | 0.8957 | 0.0181 | 0.9568 | 0.0269 | 0.8717 | 0.0440 | 0.8499 | 0.0597 | 0.9112 | 0.0424 |
| 3.3 | 0.9086 | 0.0156 | 0.9679 | 0.0397 | 0.8707 | 0.0345 | 0.8844 | 0.0426 | 0.9183 | 0.0230 |
| 3.6 | 0.8878 | 0.0125 | 0.9669 | 0.0250 | 0.8940 | 0.0402 | 0.8160 | 0.0339 | 0.8940 | 0.0442 |
| 3.9 | 0.8804 | 0.0078 | 0.9495 | 0.0191 | 0.8635 | 0.0345 | 0.8368 | 0.0365 | 0.8853 | 0.0264 |
| 4.2 | 0.8774 | 0.0078 | 0.9405 | 0.0422 | 0.8516 | 0.0330 | 0.8387 | 0.0374 | 0.8923 | 0.0343 |
| 4.5 | 0.8745 | 0.0150 | 0.9428 | 0.0397 | 0.8473 | 0.0493 | 0.8280 | 0.0328 | 0.8886 | 0.0309 |
| 4.8 | 0.8774 | 0.0265 | 0.9505 | 0.0228 | 0.8533 | 0.0404 | 0.8176 | 0.0497 | 0.9000 | 0.0396 |

TABLE A.2: **Aggregated cross-validation results for** $0.3m$ **dataset,** 200 **neurons**

| resolution (m) | All categories | | agriculture | | forest-woodland | | semi-desert | | shrubland-grassland | |
|---|---|---|---|---|---|---|---|---|---|---|
| | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std |
| 1 | 0.8636 | 0.0425 | 1.0000 | 0.0000 | 0.8387 | 0.0489 | 0.8561 | 0.0432 | 0.8541 | 0.0646 |
| 2 | 0.9001 | 0.0138 | 0.9803 | 0.0306 | 0.8991 | 0.0634 | 0.8843 | 0.0413 | 0.8881 | 0.0430 |
| 3 | 0.9041 | 0.0246 | 0.9846 | 0.0246 | 0.9245 | 0.0280 | 0.8742 | 0.0537 | 0.8805 | 0.0315 |
| 4 | 0.8970 | 0.0227 | 0.9846 | 0.0246 | 0.8743 | 0.0280 | 0.8786 | 0.0481 | 0.9129 | 0.0222 |
| 5 | 0.8865 | 0.0391 | 0.9809 | 0.0328 | 0.8692 | 0.0577 | 0.8821 | 0.0328 | 0.8797 | 0.0472 |
| 6 | 0.8800 | 0.0282 | 0.9673 | 0.0352 | 0.8685 | 0.0467 | 0.8419 | 0.0469 | 0.9093 | 0.0508 |
| 7 | 0.8587 | 0.0350 | 0.9584 | 0.0482 | 0.8360 | 0.0653 | 0.8468 | 0.0583 | 0.8682 | 0.0598 |
| 8 | 0.8699 | 0.0230 | 0.9831 | 0.0289 | 0.8788 | 0.0442 | 0.8432 | 0.0303 | 0.8498 | 0.0335 |
| 9 | 0.8338 | 0.0148 | 0.9341 | 0.0562 | 0.8330 | 0.0565 | 0.8086 | 0.0431 | 0.8245 | 0.0543 |
| 10 | 0.8234 | 0.0263 | 0.9911 | 0.0253 | 0.8159 | 0.0553 | 0.7815 | 0.0420 | 0.8199 | 0.0700 |
| 11 | 0.8219 | 0.0249 | 0.9703 | 0.0645 | 0.8052 | 0.0507 | 0.7901 | 0.0538 | 0.8193 | 0.0702 |
| 12 | 0.8112 | 0.0257 | 0.9490 | 0.0333 | 0.8178 | 0.0362 | 0.7606 | 0.0363 | 0.8118 | 0.0814 |
| 13 | 0.8147 | 0.0369 | 0.9666 | 0.0488 | 0.7988 | 0.0921 | 0.7719 | 0.0723 | 0.8244 | 0.0572 |
| 14 | 0.7837 | 0.0167 | 0.9351 | 0.0690 | 0.7682 | 0.0403 | 0.7398 | 0.0543 | 0.7999 | 0.0397 |
| 15 | 0.7808 | 0.0245 | 0.9457 | 0.0289 | 0.7663 | 0.0673 | 0.7257 | 0.0472 | 0.8017 | 0.0439 |
| 16 | 0.7990 | 0.0361 | 0.9403 | 0.0635 | 0.7680 | 0.0700 | 0.7713 | 0.0857 | 0.8141 | 0.0764 |

TABLE A.3: **Aggregated cross-validation results for** $1m$ **dataset,** 100 **neurons**

| resolution (m) | All categories | | agriculture | | forest-woodland | | semi-desert | | shrubland-grassland | |
|---|---|---|---|---|---|---|---|---|---|---|
| | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std | mean acc. | std |
| 1 | 0.8863 | 0.0305 | 0.9693 | 0.0432 | 0.8661 | 0.0452 | 0.8974 | 0.0352 | 0.8599 | 0.0377 |
| 2 | 0.8962 | 0.0156 | 0.9875 | 0.0280 | 0.8951 | 0.0526 | 0.8819 | 0.0401 | 0.8800 | 0.0472 |
| 3 | 0.8791 | 0.0267 | 0.9635 | 0.0519 | 0.8713 | 0.0761 | 0.8873 | 0.0331 | 0.8576 | 0.0336 |
| 4 | 0.8991 | 0.0339 | 0.9717 | 0.0317 | 0.8999 | 0.0453 | 0.8734 | 0.0555 | 0.9047 | 0.0529 |
| 5 | 0.8913 | 0.0301 | 0.9764 | 0.0324 | 0.8823 | 0.0577 | 0.8591 | 0.0243 | 0.9195 | 0.0495 |
| 6 | 0.8655 | 0.0194 | 0.9728 | 0.0450 | 0.8609 | 0.0472 | 0.8341 | 0.0339 | 0.8728 | 0.0542 |
| 7 | 0.8695 | 0.0321 | 0.9487 | 0.0588 | 0.8789 | 0.0221 | 0.8327 | 0.0616 | 0.8713 | 0.0669 |
| 8 | 0.8803 | 0.0230 | 0.9563 | 0.0592 | 0.8702 | 0.0399 | 0.8625 | 0.0422 | 0.8845 | 0.0144 |
| 9 | 0.8087 | 0.0378 | 0.9370 | 0.0706 | 0.7727 | 0.0738 | 0.7832 | 0.0990 | 0.8339 | 0.0819 |
| 10 | 0.8211 | 0.0352 | 0.9301 | 0.0634 | 0.8172 | 0.0676 | 0.7900 | 0.0409 | 0.8239 | 0.0698 |
| 11 | 0.8242 | 0.0130 | 0.9458 | 0.0604 | 0.8250 | 0.0646 | 0.7849 | 0.0416 | 0.8201 | 0.0414 |
| 12 | 0.8036 | 0.0252 | 0.9577 | 0.0473 | 0.7817 | 0.0252 | 0.7695 | 0.0649 | 0.8168 | 0.0588 |
| 13 | 0.8219 | 0.0408 | 0.9548 | 0.0384 | 0.8040 | 0.0809 | 0.7870 | 0.0621 | 0.8361 | 0.0494 |
| 14 | 0.8064 | 0.0343 | 0.9575 | 0.0374 | 0.7846 | 0.0595 | 0.7743 | 0.0508 | 0.8164 | 0.0592 |
| 15 | 0.7982 | 0.0368 | 0.9638 | 0.0397 | 0.8004 | 0.0756 | 0.7330 | 0.0374 | 0.8117 | 0.0625 |
| 16 | 0.7989 | 0.0247 | 0.9528 | 0.0466 | 0.7636 | 0.0673 | 0.7708 | 0.0419 | 0.8161 | 0.0446 |

TABLE A.4: **Aggregated cross-validation results for** $1m$ **dataset,** 200 **neurons**

# Appendix B

# Files and Code

All the files used in this project, including the image datasets build and code generated, are available online:

- The images datasets are published in Google Drive (see link or reference [11]).
- All the code produced and used in these analysis is available in a GitHub repository (see link or reference [12]). It includes Python libraries generated, scripts and Jupyter Notebooks.

# Appendix C

# Author contributions

This thesis is a group project between Eduard Ribas Fernández and Peter Weber. Here we will describe the individual contributions of each author. Overall, both authors have contributed to all parts in this project with different weights in each part.

In the first major block, the generation of the dataset, the distribution is as follows. The image search and download of the raw images was performed by P. Weber, while programming the image processing pipeline was done by both authors with similar weight. The labeling of the processed images was also done by both authors.

In the second major block, the data analysis pipeline, P. Weber has higher contribution at the beginning of the pipeline i.e. prototyping first solutions using transfer learning. E. Ribas has higher contribution towards the end of the pipeline. This includes optimizing the code for Colab, tuning the hyperparamters, performing analysis per category and producing the final figures. Regarding estimation of the cost, both authors have equal contribution.

The same applies for preparing all documents related to this project (Github repository, high-level overview, thesis document), both authors have equally contributed.

# Bibliography


[1] D. G. Lowe. "Object recognition from local scale-invariant features". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2. 1999, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

[2] David G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "SURF: Speeded Up Robust Features". In: *Computer Vision – ECCV 2006*. Ed. by Aleš Leonardis, Horst Bischof, and Axel Pinz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 404–417. ISBN: 978-3-540-33833-8.

[4] N. Dalal and B. Triggs. "Histograms of oriented gradients for human detection". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. 2005, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

[5] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *CoRR* abs/1609.04747 (2016). arXiv: 1609.04747. URL: http://arxiv.org/abs/1609.04747.

[6] *ResNet - Applications - Keras Documentation*. URL: https://keras.io/applications/{\#}resnet (visited on 06/26/2019).

[7] *Tsiolkovsky Rocket Equation*. URL: https://en.wikipedia.org/wiki/Tsiolkovsky_rocket_equation.

[8] *Skysat Planet Labs*. URL: https://www.planet.com.

[9] *Youtube Channel Satellogic*. URL: https://www.youtube.com/watch?v=_kE7FC8yWGs.

[10] *AWS Elastic Cloud Compute Pricing*. URL: https://aws.amazon.com/de/ec2/pricing/on-demand/.

[11] *Image folder of published datasets*. URL: https://drive.google.com/open?id=1Hjod1ZTuSIW3VNO2IuGoq_iagI3imnJQ.

[12] *Github repository of this project*. URL: https://github.com/peterweber85/MasterThesis (visited on 06/26/2019).