Internship Report

—

Contribution of different artificial intelligence techniques for the classification of low-cloud spatial organizations

Submitted by: Samy Rayan RAMOUL

Issue date: 27/08/2021

Supervisors: BRIENT Florent, DENBY Leif, BONY Sandrine

Referent teacher : BEREZIAT Dominique

E-Mail: raysamram@gmail.com

Matriculation number: 28707807

Contents

List of Figures

Acknowledgements

I want to thank the EUR-IPSL and Sorbonne Center For Artificiel Intelligence (SCAI) for financing and support of this internship.

I want to express my deepest gratitude to my supervisor Florent BRIENT, for having accompanied and guided me throughout this project. Without whom this work would not have been successful. His help and supervision were precious throughout the internship. They helped me get resources, get everything available for me to work on, and organize weekly meetings, allowing us to keep in touch with the advancements of the work.

I want to thank Leif DENBY for his advice and support. And with whom I was able to discuss different solutions concerning models and data sets, but also better understand the connections between machine learning and cloud images.

Finally, I would like to express my special gratitude and thanks to Sandrine BONY for giving me such attention, time, and expertise, giving me insights into the climatology field, and helping me understand the research field in general.

Introduction

Studying the evolution and responses of low clouds to global warming is a big source of uncertainty in the climatology research area.

Researchers need to better understand the physical processes controlling the formation and life cycle of these clouds to understand existing correlations between visual features similarities and physical processes, represent them more realistically in their models, and thus improve their credibility.

This internship aims to take an interest in the structural organizations of the clouds, a signature of physical boundary layer processes. But also, to study the link between the supervised and unsupervised cloud classifications, and define the important parameters of the models and even testing it on new cloudy situations.

To tackle this problem, we will use multiple satellite images from different sensors onboard having spatial resolutions of the order of kilometer, allowing close observation of these structures. In the context of the measurement campaign EUREC4A (January and February 2020) that aim to study in more details the organization of shallow convection off the Barbados (Bony et al. 2017), a classification of these structures was made using a supervised model trained on collaborative labeling of observations of a satellite in orbit.

From this, four main classes of the spatial organization of trade winds clouds have been identified for the first time, which allow to deduce their climatology and to study their sensitivity to environmental conditions. By introducing this new automated model (Rasp et al. 2020), researchers open new ways to study cloud structures. However, there is still several problems such as labeling time, human cognitive biases for the visual labeling.

A second study was made (Denby 2020), tackling the problem of clouds classification in an unsupervised way that should reduce human biaises. This study has proven its ability to identify several cloud areas with different physical features and morphological characteristics, using a ResNet-34 pre-trained network and a triplet-based training method sample images.

Our aim will be

- To understand the way the clouds satellites data works and how the RGB images are created.
- Handle the different data sets and study their characteristics.
- Take the original model from Denby 2020 article and apply it to new data sets and see how to use it and improve it.
- Determine a comparison between the unsupervised predictions and the supervised ones, by developing a means to compare the discrete classes produced by the supervised network with the embedding vectors of the unsupervised network.
- Finally, determine the different strength and weaknesses of our methodology.

In this introductory section of the report, we will introduce the context in which this internship takes place, the basic concepts in meteorology and cloud fields we need, the algorithms that we will to use but also the different data sets and metrics we are interested in.

1 Context

1.1 Laboratoire de Météorologie Dynamique

This internship took part in the Laboratoire de Météorologie Dynamique (LMD). It was created in 1968 and has become a mixed research unit since 1998. It is located on three places in Ile-de-France: the Ecole Polytechnique in Palaiseau, the Ecole Normale Supérieure, and the Science Campus of Sorbonne University in Paris.

The laboratory is a member of the Institute Pierre Simon Laplace (IPSL), bringing nine public research centers in environmental sciences.

The LMD research is centered on studying meteorological phenomena and their physical mechanisms through theoretical approaches, observations, and climate models. There are five research teams within the laboratory, each one with its research theme :

- ABC (t) team studies of global climate and climatic processes from the study of radiation.
- DPAO team studies the fundamental mechanisms of the dynamics and physics of geophysical fluids applied to the atmosphere and the ocean, from the turbulent scale to the planetary scale.
- InTro team is dedicated to the study, on a regional scale, of physio-chemical processes in the troposphere.
- Planéto's team focuses its research around planetary atmospheres by building and applying digital climate models similar to the one developed for Earth but applied to climatic systems of other planets.
- The EMC3 (Etude et Modélisation du Climat et du Changement Climatique / Study and Modeling of Climate and Climate Change) team brings together researchers whose work revolves around the goal to improve our physical understanding of the climate system and to better anticipate future climate change by using models, which are essential tools to achieve these objectives, on condition of being able to trust the results of simulations. This internship is part of this teamwork.

1.2 Clouds

1.2.1 Composition and localization

Clouds are aerosols consisting of an apparent mass of minute liquid droplets, frozen crystals, or other particles suspended in the atmosphere. The clouds can be composed of crystals, droplets, or other chemicals. Their composition process is formed after air saturation, leading it to be cooled when it gains sufficient moisture. Clouds are seen in the Earth's troposphere (8 to 15 km), stratosphere (12 to 50 km), and mesosphere (50 to 80 km).

Clouds are seen in the different layers of the atmosphere. This study will focus on warm liquid clouds that form on the lowest part of the troposphere (below 4 km), which are usually name cumulus clouds [1].

The study of clouds is crucial because they strongly modify the Earth's radiative balancess, and contribute on different scales to heat and moisture transport, rain production and the dynamical circulation.

1.2.2 Mesoscale

Mesoscale in meteorology expresses a medium horizontal scale within the planetary circulation of synoptic-scale (depressions and anticyclones across a whole continent, ocean currents, etc.) and systems with tiny scales of smaller than 2 km in diameter (micro-scale).

In this internship, we are interested in mesoscale organizations of shallow clouds (extending 20 to 2,000 km). Comparison of typical patterns with radar imagery suggests that even this subjective and qualitative visual inspection of imagery appears to capture several significant physical differences between shallow cloud regimes, such as precipitation and radiative effects. And creating a pipeline to predict the dominant domain classes automatically is then really important to help the researcher's work and make it more easy to then define the physical features (Stevens et al. 2019).

1.2.3 Cloud Classes

Recently, Stevens et al. 2019 visually recognized four mesoscale patterns of shallow convection, attributed to as flowers, fish, gravel, and sugar (see figure 1) :

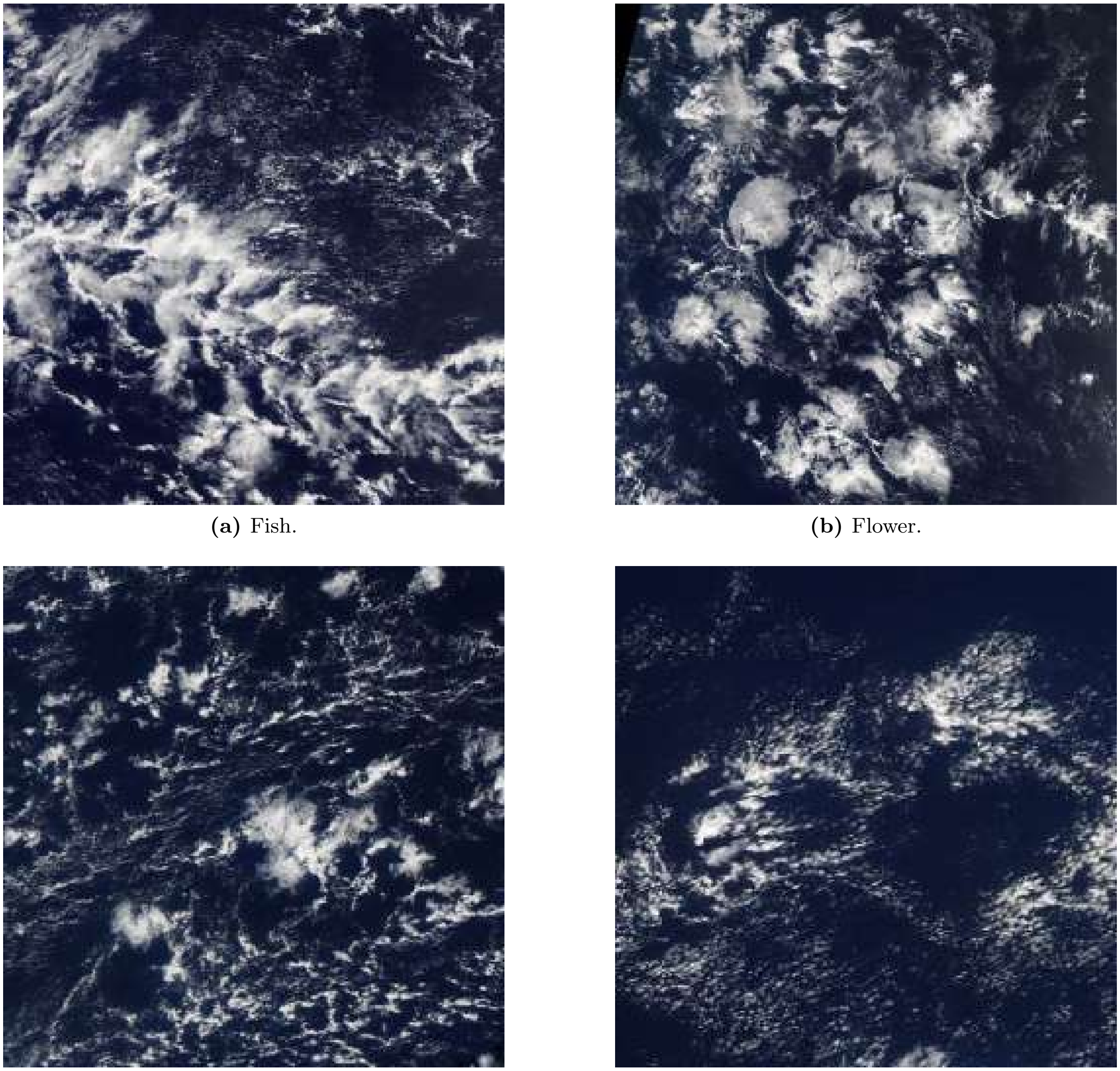
- Fish are elongated, skeletal arrangements that sometimes traverse up to 1 000 km, essentially longitudinally.These remarkably well-structured cloud forms are usually seen in the middle of all ocean basins.

- Flower depicts areas with isotropic cloud arrangements, each varying from 50 to 200 km diameter, with correspondingly vast cloud-free regions in between. Flowers are often less densely packed than typically closed cells (look comparable to a capped honeycomb from over, with thick cumulus clouds on the center of the cells), with narrow cloud-free areas at the edges.

- Gravel represents areas of granular features marked by arcs or circles. The standard scale of these arcs is approximately 20 km. We assume that these patterns are run by cold pools produced by raining cumulus clouds.

- Sugar reports widespread areas of really fine cumulus clouds. Overall these areas are not very reflective, do not have large pockets of cloud-free regions, and, ideally, present limited proof of mesoscale organization.

[1] https://cloudatlas.wmo.int/en/definition-cumulus-cu.html



(a) Fish.

(b) Flower.

(c) Gravel.

(d) Sugar.

Figure 1: Example image for each class. Each image is from the Aqua-MODIS satellite and covers the area from 60°W to 48°W and 10°N to 20°N © Stevens et al. 2019

Recently, supervised algorithms were applied on domain images of those different classes (see section 3.2.1).

1.3 The EUREC4A Project

The EUREC4A campaign (Bony et al. 2017) took place in January and February 2020 in the lower Atlantic trades, over the ocean east of Barbados (13 N; 59 W). This location was chosen because of the following reasons :

- Shallow cumuli are prominent in this area, especially during winter.

- The cloudiness in the region of Barbados is representative of clouds over the whole trade wind region of the tropical ocean, both in models and in observations.
- The possibility to use measuring instruments of Barbados Cloud Observatory, which have been observing clouds continuously since 2010, and also, the possibility to use the legacy of flight campaigns organized in the area in the last few years

It has created a lot of interest in the international community concerning the diversity of patterns of low-level cloudiness which can be found in the trade-wind regions and it also created questions about them.

One of the EUREC4A questions is how these different patterns form and what their role is in climate. With this in mind, an early classification has been made by humans as part of the preparation of the campaign. Later on, other initiatives aimed at recognizing and classifying the patterns more automatically, including using machine learning approaches. And the internship takes place in this context, and one of the goals is to tackle the satellite acquisition data in this time frame which we provide us a lot of meta-data for our analysis.

1.4 Acquisition Methods

To better understand how the data is used in this internship work, we will introduce the different acquisition methods used.

1.4.1 Satelittes

The GOES (which stands for Geostationary Operational Environmental Satellite) satellites are in geostationary orbit, at a range of 35,790 km from the Earth's surface. All of them are fixed in an appropriate point, so they can consistently cover a particular portion of the globe. The series had multiple variants and generations, and the first one was called the GOES-I satellite. In this work, we will focus on the more recent satellite, called GOES-16 and launched on December 2016.

Geostationary satellites are called Geostationary because they appear fixed as they move at the same angular velocity as the Earth and orbit along a path parallel to Earth's rotation, providing coverage to a specific area.

The GOES satellites are maintained and permanently face their devices towards the Earth using a three-axis gyroscopes system.

Because the goal of the satellites is meteorological tasks, they have two types of sensors:

- An imaging radiometer that can catch several wavelengths of the electromagnetic spectrum in the visible and infrared.
- To extract the temperature and humidity structure, it uses a sounder that remotely takes out an aerological survey of the Earth's atmosphere.

These sensors demand around 10 minutes to acquire the data from the domain they are staring at. Each part is probed according to the resolution of 1 $\mathbf{km^2}$ below the satellite point and lowers towards the edges of the terrestrial disc because of the angle of sight.

The acquisition frequency is of a series of images every 15 minutes, which can be increased by limiting the area covered, reaching by this a series every 7.5 minutes.

Then, using acquisition extracted from a tool *Advanced Baseline Imager* which provide 16 spectral bands, over which we extract three of them (radiances at $0.64\mu m$ which we use to create Red, $0.86\mu m$ for Green, and $0.47\mu m$ for Blue) to create the RGB image. Finally, we apply processes of true color computing (to get a natural green), gamma correction, and projection of coordinates (because earth is not flat) to get a final RGB Image (see Appendix).

1.4.2 Moderate Resolution Imaging Spectroradiometer (MODIS)

Low-orbit satellites orbit at an altitude within 160 to 2,000 kilometers. Thus, a constellation of low-orbit satellites can give continuous, global coverage as the satellite progress. Unlike geostationary satellites, low-orbit satellites also operate much faster because of their closeness to Earth.

MODIS (Moderate Resolution Imaging Spectroradiometer) is a payload imaging sensor launched into Earth orbit by NASA in 1999 onboard the Terra satellite and in 2002 onboard the Aqua satellite.

The tools catch data in 36 spectral bands ranging in wavelength from $0.4\mu m$ to $14.4\mu m$ and at differing spatial resolutions. Together they image the whole Earth each 1 to 2 days, and can provide measures in large-scale global dynamics including the Earth's cloud cover (see Figure 1 for some examples).

With its base spatial resolution but large temporal resolution, MODIS data are helpful to follow developments in the landscape across time. The most important difference with geostationary satellites is the fact that instruments embedded in low-orbit satellites provide only two images for a given location.

2 Data Sets

We used two different data sets in this internship. In this section, we will introduce each of them and their corresponding characteristics and features.

2.1 The ISSI Database

This data set arises from the activity of an International Space Science Institute (ISSI) International Team researching "The Role of Shallow Circulations in Organising Convection and Cloudiness in the Tropics" (Stevens et. al 19). It consists in a collection of cloud images produced by the MODIS instruments aboard on Terra and Aqua satellites.

This data set is hand-labeled by cloud specialists. Five different labels can define each cloud image representing a one of the four class listed on section 1.2.3 or expressing none.

Each image has been labeled with only one class. However, the large size of the image (around 2000x1000 km) increase the chance of them being labeled as a whole, but having a large domain increased the chances that different patterns of shallow-cloud organization would arise in other parts of the environment.

The domain used was located at 58°W and 10°N, upwind of Barbados. The group labeled within a season (December, January, and February) over five years of images, in a period of 10 seasons going from 2007/2008 to 2016/2017. These years were chosen as they were the only ones available on Worldview [2] during the labeling activity. And for the season, the choice was motivated by the predomination of trade winds (trade winds favor the extension of low-cloud organisation given the fact that most of the storm activity shifts in the southern hemisphere during the boreal winter season).

The classification was performed only on daytime MODIS-Aqua images using the "Corrected reflectance" product and have a resolution of 250 or 500 m.

2.2 Zooniverse

In 2019 Stephan Rasp uploaded a part of the crowdsourced data set created on Zooniverse [3], labeled by multiple users on Kaggle, a data science competition website. In the competition description, he mentions that by getting good results (in terms of overall intersection over union), the models created could help scientists understand how clouds will shape our future climate and reduce uncertainties in climate projections.

An image is created by stitching together data from two orbits, and the remaining areas (uncovered by the orbits) are marked black. Images from MODIS (1.4.2) were collected and then uploaded on a crowdsourcing platform (Zooniverse) so users can annotate parts of the image with cloud categories as labels. Sixty-seven participants screened 10,000 satellite images on a crowdsourcing program and classified around 50,000 mesoscale cloud groups.

The data set is in the form of an image folder and .csv file, which contains the pixels labeled by each user as a class of cloud. So, for each user, we have different labels (or none) of each image pixel.

[2] https://worldview.earthdata.nasa.gov

[3] https://www.zooniverse.org/projects/raspstephan/sugar-flower-fish-or-gravel

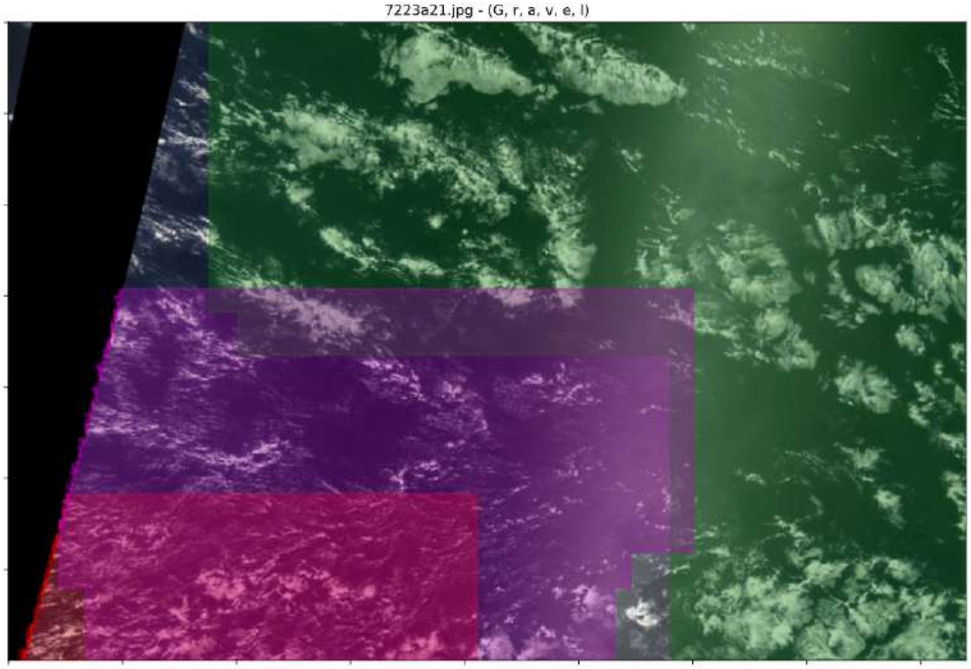


Figure 2: Example of an image labeled by multiple users / (Green : Flower, Red : Gravel and Purple : Sugar)

For example, in the image in figure 2 we can see that different users saw the presence of three different classes, each one of them having a different color.

Rasp downloaded images of this data set originates from the NASA Worldview and describes 3 different regions of the globe where low clouds prevail. The true-color pictures were taken from two polar-orbiting satellites, TERRA and AQUA.

3 Machine Learning

In this section we will introduce and describe the different techniques of artificial intelligence on clouds classification we will use.

3.1 Models

3.1.1 Hierarchical Clustering

Hierarchical clustering is an algorithm for the unsupervised clustering of data points. It aims to build a hierarchy of clusters where each cluster can be divided into children clusters.
The algorithm follows this process :

- Consider each observation as a complete cluster.
- Then executes repeatedly the two next steps.
  - Computer clusters distance to get the one that is the closest.
  - Merge them in a similar cluster.

- The two tasks are executed until only one cluster is remaining, as shown in the figure 3.

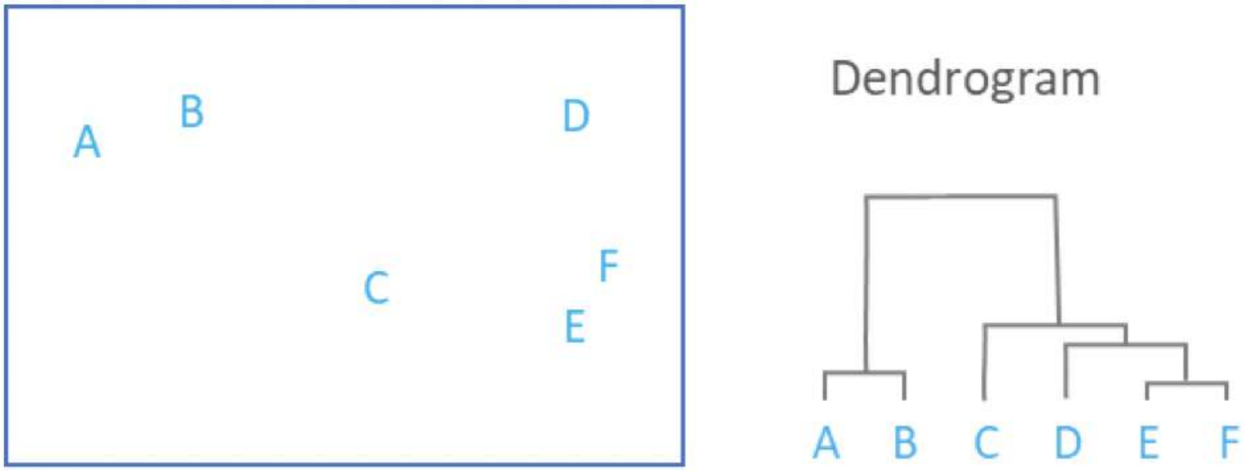


Figure 3: Example of hierarchical clustering © Displayr

The distance between two clusters can be calculated based on the euclidean distance between the two clusters (or points in the first iteration). But this metric can be adapted depending on the theoretical concerns of the domains we are working on.
In this algorithm, clusters can be constructed in two different ways :

- By merging similar clusters, as explained before, a process called agglomerative hierarchical clustering.
- By grouping all the data points into one cluster and then iteratively splitting these clusters, a process called divisive hierarchical clustering.

The number of clusters is then chosen by either a specific preference (to be compared to existing classes) or using metrics like silhouette score, which measures the maximization of variance inter-classes and minimization of variances intra-classes.

3.1.2 GradCAM

GradCam is an algorithm aimed to help in the understanding and interpretation of deep neural networks. It is used to see which parts of a given image made a convnet to its final classification decision. This technique comes from a more general category of processes called class activation map (CAM) visualization.

It helps to understand why a convolutional neural network made its predictions and, for example, debugging when it is doing the wrong one. It is also helpful to transform a classification task into a segmentation task because we can see the single activated pixels and consider it a segmentation map. However it lacks precision in terms of resolution.

The process produces heat maps representing the pixels activated for classes on the input images, so a heat map is associated with a pair of pictures and class.

These classes are calculated for each pixel of an input image, indicating the importance of each pixel relative to the considered class. We can visualize this process in the figure 4.

The algorithm is as follows :
Step 1 : Gradient computation y^c is the raw output of the neural network for class c before the softmax is applied to transform the raw score into a probability. Compute the gradient of y^c relatively to the feature map activation A^k of the last convolution layer. We get from a 2D input image a 3D gradient with the same shape as feature maps. K feature maps with each one a height of v and width u, and final shape of [k, v, u], and so the gradient has the same shape.

Step 2: Compute Alphas by mean of Gradients Compute mean gradient on height and width axes, which gives a value of gradient per neuron; then we multiply it times the gradient computed in step 1, which gives us this equation :

$$\alpha_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\delta y^c}{\delta A_{ij}^k} \tag{1}$$

Gradient Shape : [k, v, u], pooling on width and height, and we finish with a shape of [k, 1, 1] or [k] alpha values.

Step 3: Compute Final Grad-CAM Heatmap We process this final equation :

$$L_{\text{GradCAM}}^c = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right) \tag{2}$$

The size of the final heatmap is [u, v], the same size as the final convolutional feature map, but we can upsample it to fit the size of the original image.

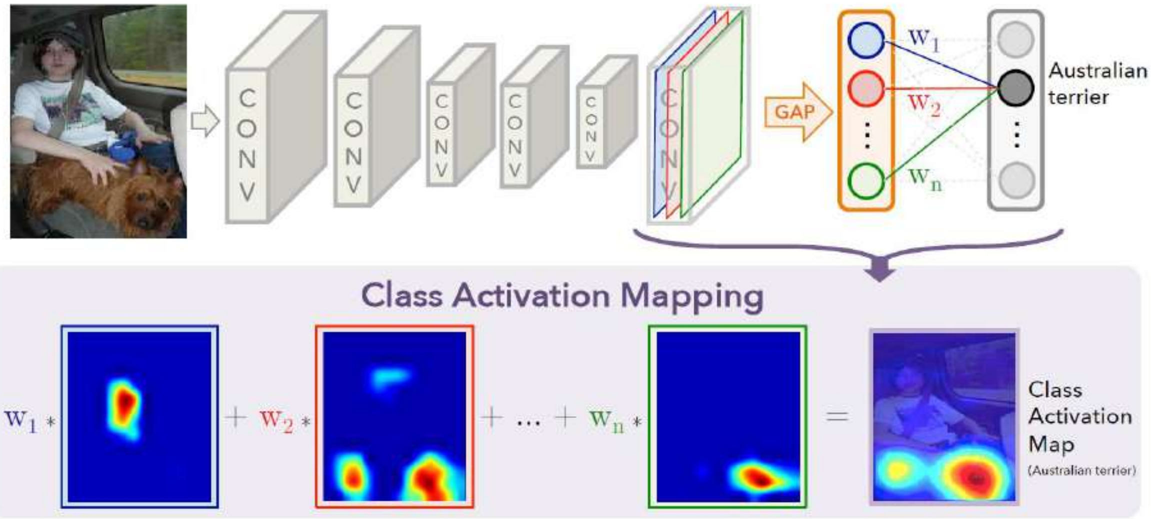


Figure 4: GradCAM algorithm © Zhou et al. 2016

3.1.3 Image Embedding

Image embedding is the process of creating a 1D vector representation of associated images.

The image embedding methodology used in the unsupervised learning follows the approach developed by Jean et al. 2019, which is similar to word embeddings in nature language processing (e.g., Mikolov et al. 2013) but with images instead of words.

The neighborhoods are defined by a spatial distance in the domain image within some radius. And, to learn a mapping from image tiles to low-dimensional embeddings, we use a convolutional neural network trained on triplets of tiles, where each triplet consists of an anchor tile, a neighbor tile, and a distant tile (as we can see in figure 5).

This method assumes that tiles within a certain radius from an anchor tile are more likely to be similar than distant ones. For a typical land-cover classification task, we see that anchor and neighbor tiles often come from the same class, while remote tiles are dissimilar.

For each triplet, we then train the model to (as shown in equation 3) :

- Minimize the distance between the anchor and neighbor spatial embeddings.
- Maximize the distance between the anchor and distant spatial embeddings.

In the figure 6 we can see the different tiles of multiples triplets, their respective distances, and extraction method.

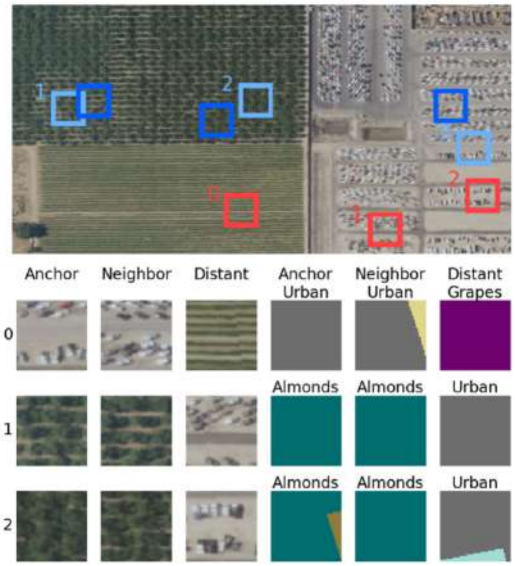


Figure 5: Different tiles types extraction © Jean et al. 2019

The equation 3 represents the associated expression of Loss function $L\left(t_a, t_n, t_d\right)$

$$L\left(t_a, t_n, t_d\right) = \left[\left\|f_\theta\left(t_a\right) - f_\theta\left(t_n\right)\right\|_2 - \left\|f_\theta\left(t_a\right) - f_\theta\left(t_d\right)\right\|_2 + m\right]_+ \tag{3}$$

Where :

- t_a is the anchor tile.
- t_n the neighbor tile.
- t_d the distant tile.
- f_θ is the CNN with weights θ that maps from the domain of image tiles X to d-dimensional real-valued vector representations.
- m is a rectifier margin that restricts the network from pushing the distant tile farther without limitation.

The figure 6 shows the final architecture of a model in the method :

- In input, we have the different corresponding matrices for each tile of the triplet.
- It is then given to a model, a convolutional neural network, to extract visual features.
- The output is the associated embedding for each of the triplet tiles.

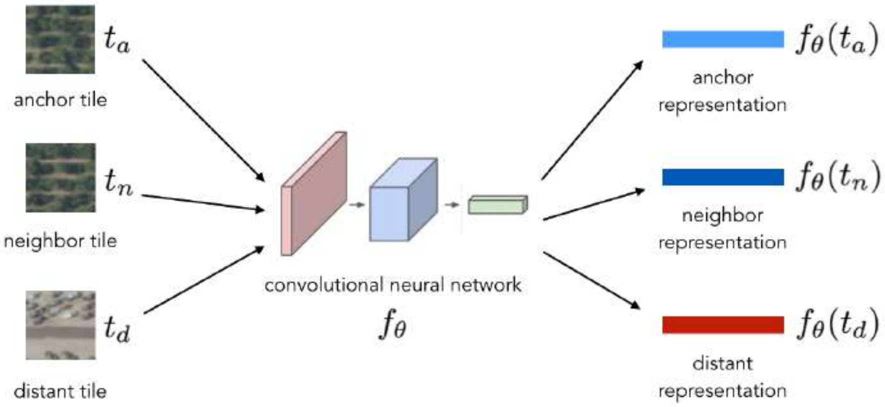


Figure 6: Learning scheme for the embedding © Jean et al. 2019

3.2 State of the art cloud classification

3.2.1 Supervised Clouds Classification

In Rasp et al. 2020, the Zooniverse data set is used as a training data set for deep learning algorithms, making it feasible to automate pattern detection in satellite images and build the global climatology of the four patterns.

From the data, they train a segmentation and an object detection algorithms. Both algorithms accurately identify the most noticeable patterns in the image and rival well with human labels. However, the object detection algorithm sometimes misses features. On the other hand, the segmentation algorithm manages to produce small patches because, other than humans and the object detection algorithm (which the range of possible box sizes is an adjustable parameter). There is no specific guidance to label larger patches.

An interesting and helpful feature of the segmentation algorithm is that, despite every training labels being rectangular, it seems to concentrate on the specific underlying shape of the patterns. This implies that despite the ambiguity in the human data set, the segmentation algorithm appears to clarify the image predictions from their noise to obtain essential shapes for the predictions.

The model gave good Intersection over Union results, better than the Intersection over Union scores between the different uses, which showed an efficient model, with less noisy model predictions.

3.2.2 Unsupervised Clouds Images Clustering

The second paper from one of my supervisors (Denby 2020) handles the problem in an unsupervised way and approaches the issue of cloud classification by understanding that the manner in which we classify clouds is based on their relative difference (or visual difference/distance) to other clouds.

From that, he developed [4] a model that learns to create a dimensional space (that we would call embedding space) for this visual difference metric; the objective of the model is to learn to assign coordinates in a multi-dimensional space for an image.

The way the model learns this new dimensional space is by taking a first tile (where a tile is a 256 by 256 pixels shaped image extracted from a domain) called anchor tile and a neighbor tile (within a random direction and a distance of 125 pixels and a 50% overlap to guarantee a minimum visual similarity) and learns to assign them a small distance, and then it takes a distant tile extracted from a random different day and try to maximize the learned distance to it (see section 3.1.3).

The three images are passed through an architecture composed of a pre-trained neural network (a ResNet) using a technique called Transfer Learning (which consists of using a model trained on big data set of other domains and which its convolutional learned to extract features and apply it to another context with fixed parameters) and then to a multi-layer fully connected neural network with a 100 sized vector which corresponds to the coordinates on a 100-dimensional embedding space.

Figure 7 shows an example of a cloud domain. We see the extraction of the tiles of the triplet, the neighbors one with an overlapping, and the distant tile coming from another field. We also see the architecture of the model consisting of multiple layers of convolution extracting edges then textures and object parts to finalize with a fully connected model which learns a mapping from the CNN features to the final embedding.

[4] https://github.com/convml/convml_tt

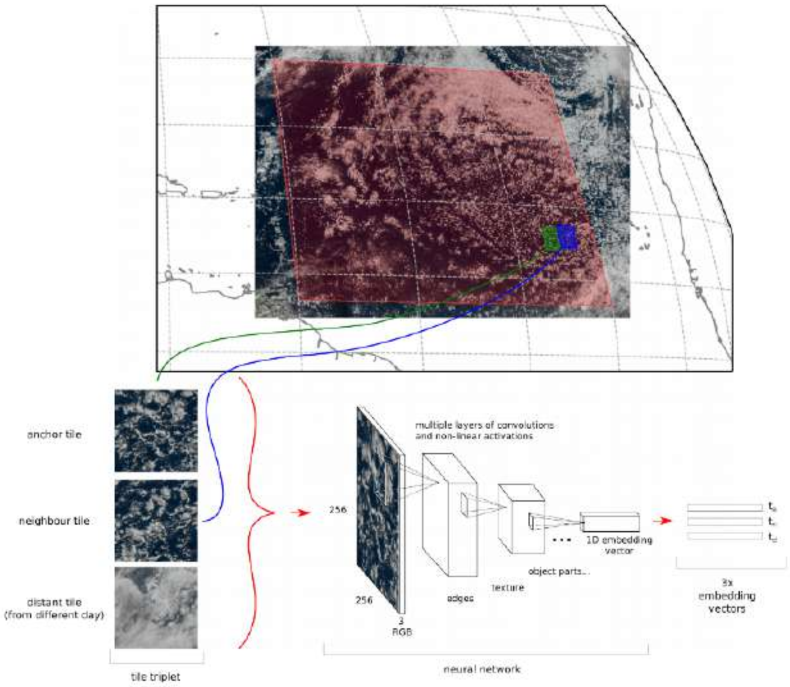


Figure 7: Cloud images tiles generation © Denby 2020

The figure 8 shows the result of applying hierarchical clustering on the associate embedding of cloud tiles (see section 3.1.1), and it obtained a good separation in terms of colors and cloud shapes. This first result suggests that the unsupervised method is able to separate different classes of low-cloud organisation.

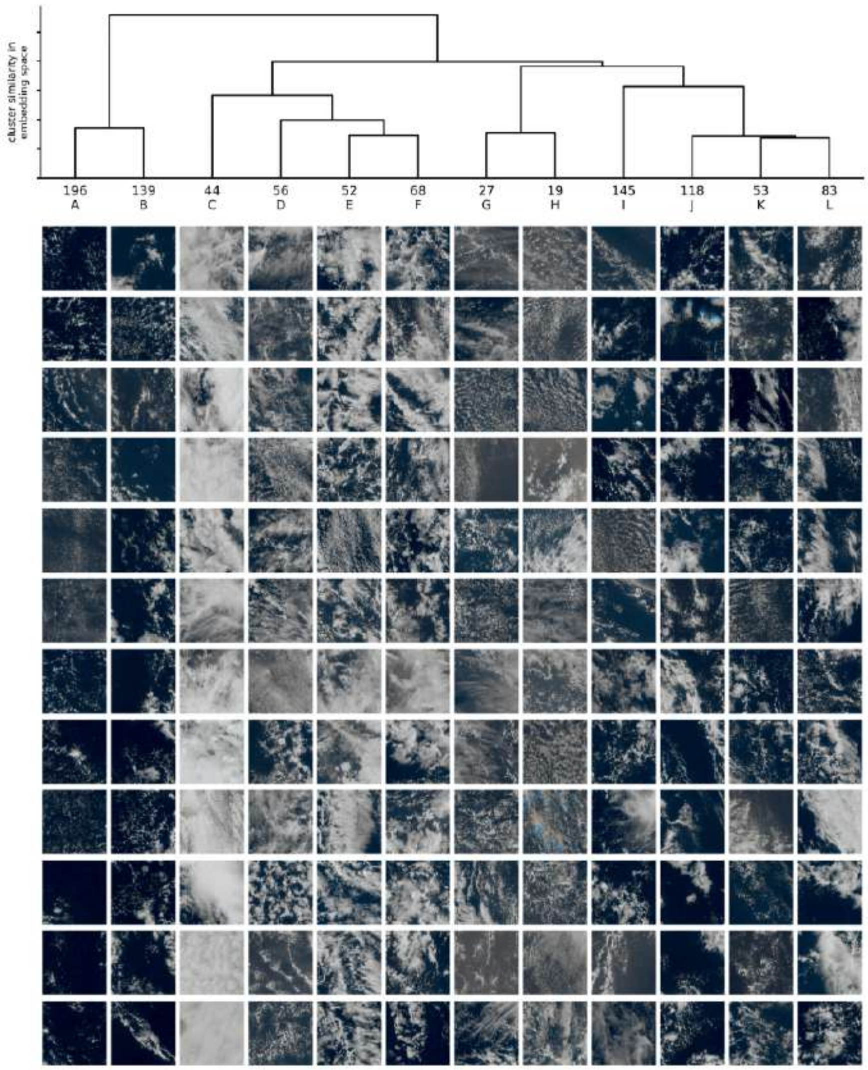


Figure 8: Clustering of clouds tiles © Denby 2020

4 Evaluation Metrics

In this section, we will describe the different ways we had to evaluate the efficiency of the models either in terms of metrics or visual and physical representation.

4.1 Normalized Mutual Info Score

Normalized mutual info score is a metric used to determine the quality of a clustering, but it needs a ground-truth class association for the data points. And since it is normalized we can measure and compare through those metrics different clusterings having different number of clusters.
It is expressed by the following equation :

$$NMI(Y, C) = \frac{2 \times I(Y; C)}{[H(Y) + H(C)]} \tag{4}$$

Where C and Y are the cluster and class labels respectively, H(.) the entropy of labels (either of classes or clusters) and I(Y ; C) the mutual Information between Y and C. This latter shows the reduction in the entropy of class labels that we get if we know the cluster labels

4.2 Representation

When applying clustering on embeddings of cloud images, ideally, we want to find similarities in shape in both macro or micro aspects of the details of the images. Also, a good result would be to find a perfect matching between one of the clusters and a known class.

Another important aspect is to observe a good explainability of the data (in other terms our algorithms have to provide a reasonable variance between the different cloud tiles).

4.3 Cloud Physics

There are many features that we can compute using RGB images. Those features are usually used in many atmospheric science studies and provide a quantitative description of the different spatial organization of clouds that exist. However, one metric characterizes only a part of the specificity of the spatial organization. Thus, a multi-metric approach might help better discriminate cloud classes.

Following are the main metrics (where apply consider clouds as individual objects and apply object detection algorithm) :

- Cloud fraction (The percentage of each pixel in the image that is covered with clouds. A cloud fraction of one indicates the pixel is fully covered with clouds).
- Max length scale of scene's largest object.
- Total perimeter of all scene's cloud objects.
- Exponent of cloud size distribution (power law fit).
- Mean length of cloud object in scene.
- Spectral length scale (Jonker, Elferink-Gemser, and Visscher 2010).
- Convective Organisation Potential White et al. (2018).
- Simple Convective Aggregation Index Tobin et al. (2012).
- Number of clouds in scene.
- Organisation index as used in Tompkins and Semie 2017.
- Minkowski-Bouligand dimension.
- Organisation index as modified by Benner, Curry, and Pinto 2001.
- Image raw moment covariance-based orientation metric.

5 Tools and Hardware

In this internship, we were provided with a set of cloud platforms, each of them having one feature and one of them featuring GPU capabilities. For this, we needed to create a pipeline to deploy our code, which is based on a combination of ssh and Conda environments.

The main libraries we used are :

- Pytorch-Lightning : It is a fork of the deep learning library PyTorch. It provides tools to make it easier to modularize and develop complex models but also to train them using the capabilities of multiple GPUs at a time.
- Luigi : Python package that helps you build complex pipelines of batch jobs. By using it, you define each task as a class that needs an input (the previous task) and gives output as a file representing a file which is then used as an input for the following tasks but also at the next execution to verify if this task needs to be executed again or not. This library also provides an easy way to multi-thread workflows and provide a visualization tool for all of them.
- Weight and Biases Website (W & B): This website [5] permits to monitor online the different training that is running but also associates each training with its corresponding metadata and pieces of information that you want it to be associated with and finally also to store each model you train to stick with the best one of them.

Around those, we used libraries like *sci-kit* learn for clustering models and PCA, *OpenCV* for image processing and object detections (to count different clouds, their areas, distances ..etc.), *pandas*, and *NumPy* for data management, and finally *seaborn* for the different plots.

6 Crowdsourced Data Set

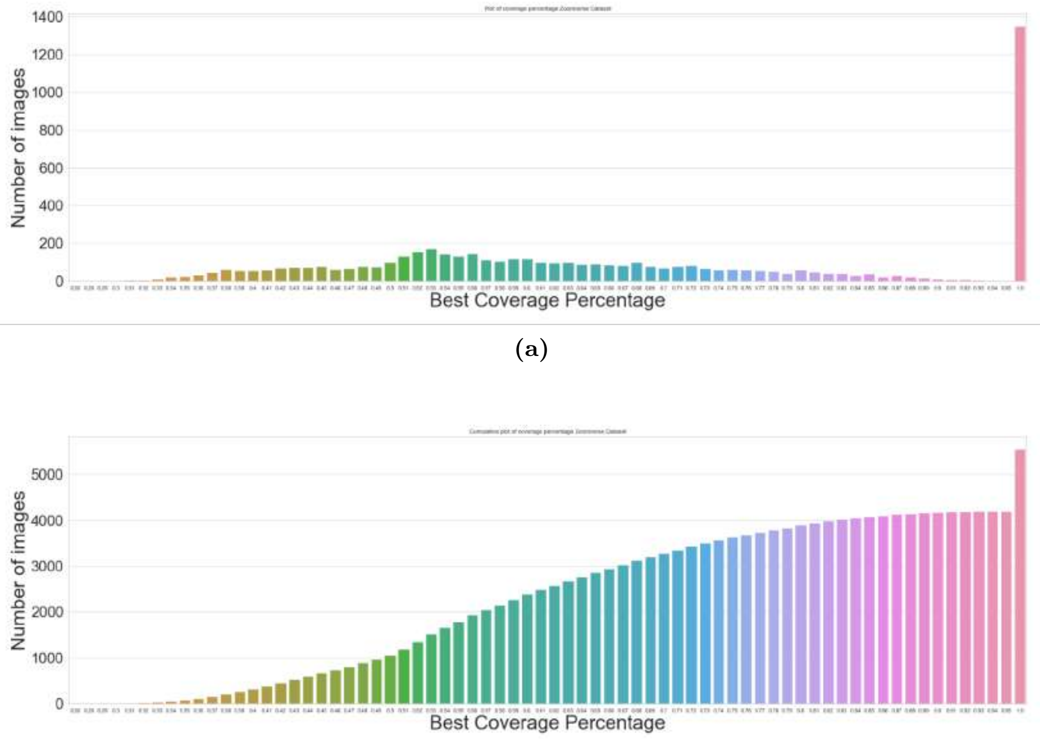
We first focused our work [6] on the Zooniverse data set (section 2.2) because it provided a significant quantity of labeled data, which permitted an excellent comparison with the point of comparison or "truth" for our future models.

6.1 Data Analysis

Before trying new approaches and ways to improve the results, we computed some statistical distribution for the Kaggle data set to see if it would confirm previous knowledge on clouds classes stipulating usual sizes of each class of clouds, and by doing those plots, we could compare users labeling of clouds to their known ones. First, the number of images in the data set and their best class percentage coverage (divided by the sum of all rectangles surfaces) are shown in figure 9.

[5] https://wandb.ai/

[6] https://github.com/rayansamy/convml_tt

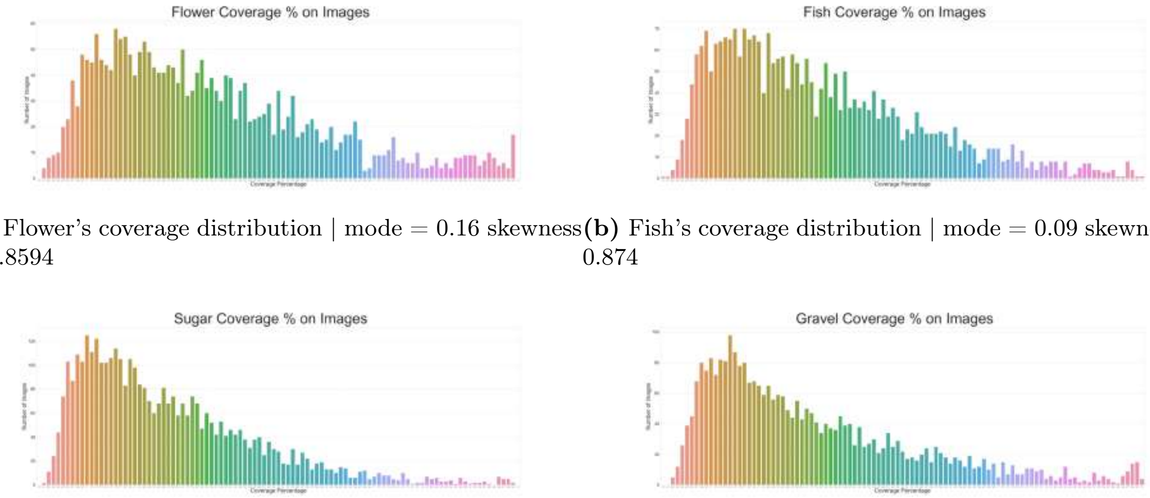


(a)

(b)

Figure 9: Distribution of number of images representing the biggest class-labeled rectangles percentage of image's coverage (a) for each individual bin and (b) represented as a cumulative distribution function. (X-axis going from 26% to 100%.

And then, each class percentage coverage on images of the Zooniverse data set and as we can see in the figure 10.



(a) Flower's coverage distribution | mode = 0.16 skewness = 0.8594

(b) Fish's coverage distribution | mode = 0.09 skewness = 0.874

(c) Gravel's coverage distribution | mode = 0.14 skewness = 1.099

(d) Sugar's coverage distribution | mode = 0.12 skewness = 1.178

Figure 10: Distribution of percentage coverage of the labeled rectangle on the images related to each class. / coverage percentage going on X-axis from 01% to 99%.

Those two distributions showed that users usually label big-sized rectangles on images, and also that the classes which are more used to be labeled with big coverage percentage on an image are Flower and Fish, which confirms previous research on the average size of clouds classes.

6.2 Results with domains divided in tiles

The initial goal was to try to make a comparison between unsupervised and human visual classifications of clouds.

To make this comparison and because predictions on the data set were in the form of user-selected rectangles, we decided to follow this process :

- Reading images of the data set (which have sizes of 1500 by 2000 pixels).
- Extract the content of bounding boxes.
- Cropping Images (from boxes) to 256 by 256 pixels tiles (going from 5546 images to more than 100.000 tiles).
- Associate each tile to the bounding box class.

We used the original model used in the paper Denby 2020 (the one trained on the GOES data set available on Amazon's website) for the first clustering of those tiles. As we can see in the figure 11, we get clusters (that we fixed for 12 to get a good grasp of the data distribution) with no striking visual similarities.

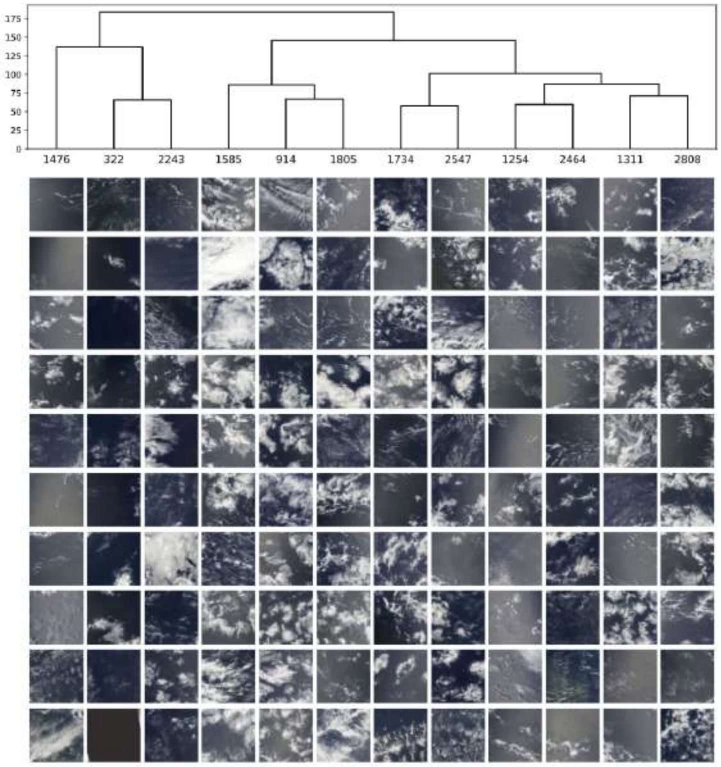
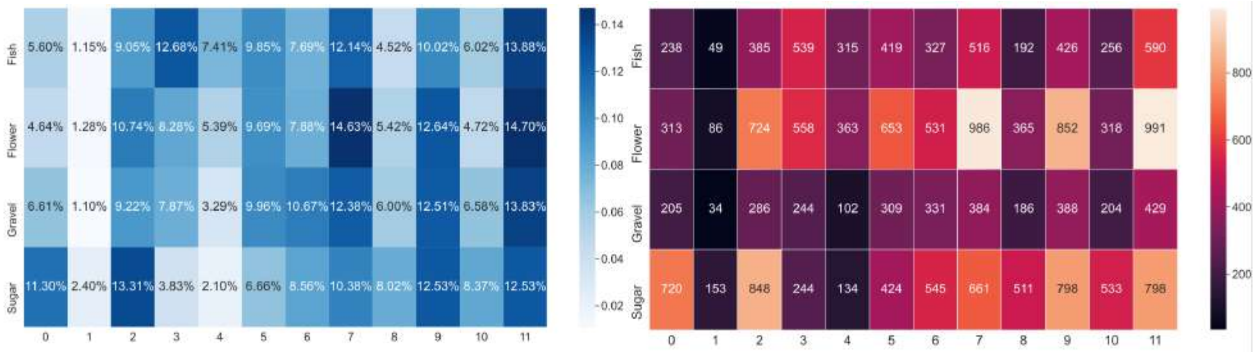


Figure 11: Clustering using the trained model from Denby 2020 on and applied on Zooniverse data set true-colour RGB-composites triplets generated from GOES-16 observations

Because each tile was initially contained in a rectangle which was associated with a class that had a cluster labeling made by the model, we have the ability to construct a confusion matrix.

By considering the rows as the original classes of the tiles and the columns as the different clusters created by the model, figure 12 shows that there is no significant correlation in any pair of class and we conclude that using this methodology didn't get us any conclusive result except that a network pre-trained on true-colour RGB images generated from GOES-16 observations doesn't directly work on MODIS color images.

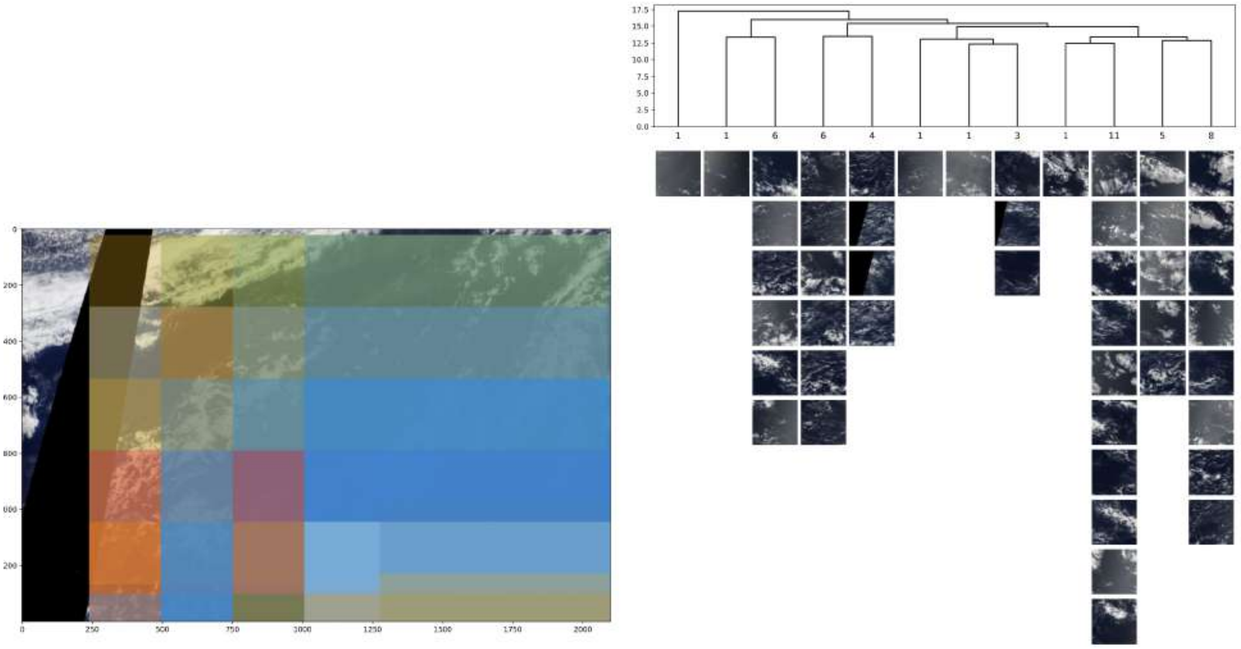


(a) The percentages correspond to the distribution of the number of images for each of the original labeled classes (Fish, Flower, Gravel, Sugar) among the 12 clusters generated by the unsupervised learning model.

(b) confusion matrix.

Figure 12: Confusion matrices produced by the resulting cluster labeling.

To help visualizing the model results, we take each of the user-predicted rectangles and divided the entire domain into separate 256 by 256 pixels tiles, and passed it through the model. As an example, figure 13 shows the model results for an image labeled as 'Fish'. Each color color correspond to one cluster, and we can see that there is no continuity in the clouds in the domain with the clouds clusters prediction even with what we could label behind as a fish cloud.



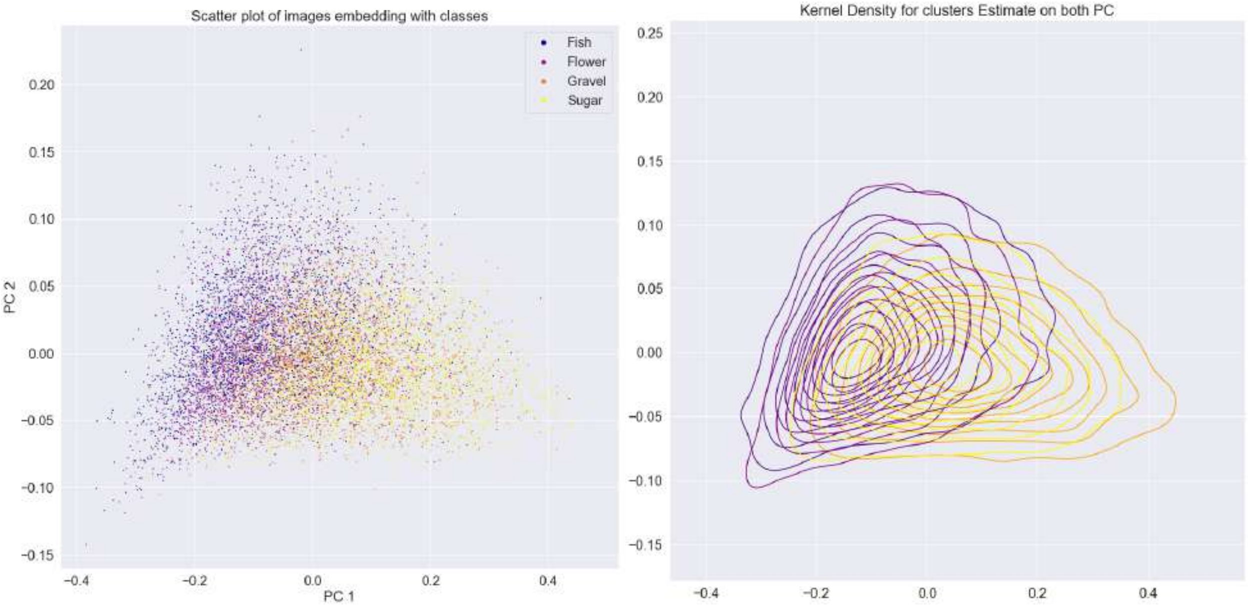
(a) Visualization of the clusters on the domain.

(b) Hierarchical clusters.

Figure 13: Domain's tile prediction.

This lack of continuity led us to think that there was a problem with the tiles' embedding, so we applied a principal components analysis (PCA), and then after extracting the two axes with the best-explained variance, we then plotted each image by its first components and associated colors depending on the original class of the rectangle. As a result in the figure 14 we can observe the following things :

- The entire data set tiles do cover only the center portion of the space.
- The principal components axes do not maximize the variance between the different classes.
- The center of gravity of each class is dissociated with the other ones.
- Weak separation between Fish/Flower and Sugar/Gravel.



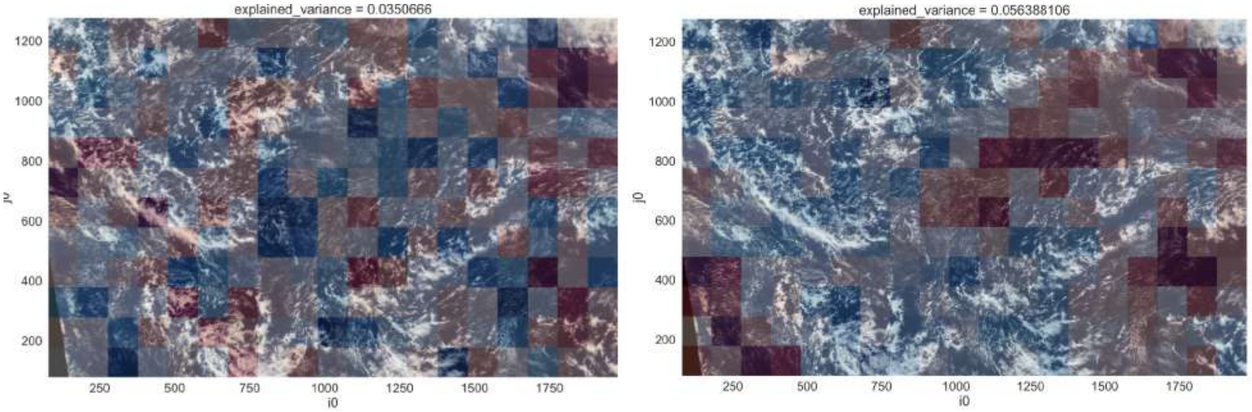
(a) Images embedding plotted.

(b) Kernel Distribution Estimate of each class.

Figure 14: A scatterplot and kernel distribution of the two first principal components calculated from the Zoouniverse data set (5546 images) using the pre-trained model of Denby 2020. Images are colored by the human labellization provided by the supervised learning.

This spatial representation we see led us to understand that the unsupervised network training on GOES-16 triplets needs to be trained on MODIS imagery to be able to separate the four cloud classes.

To confirm our intuition, we took each of the principal components as heat value to plot it on a domain depending on the tile it covers. In the figure 15 we see, same as previously, no continuity on the value of the principal component in the tiles (except on the second component where there is some but not covering the entire sea pixels). We can see that the explained variance is really low, going below 0.1.



(a) First principal component heat map.

(b) Second principal component heat map.

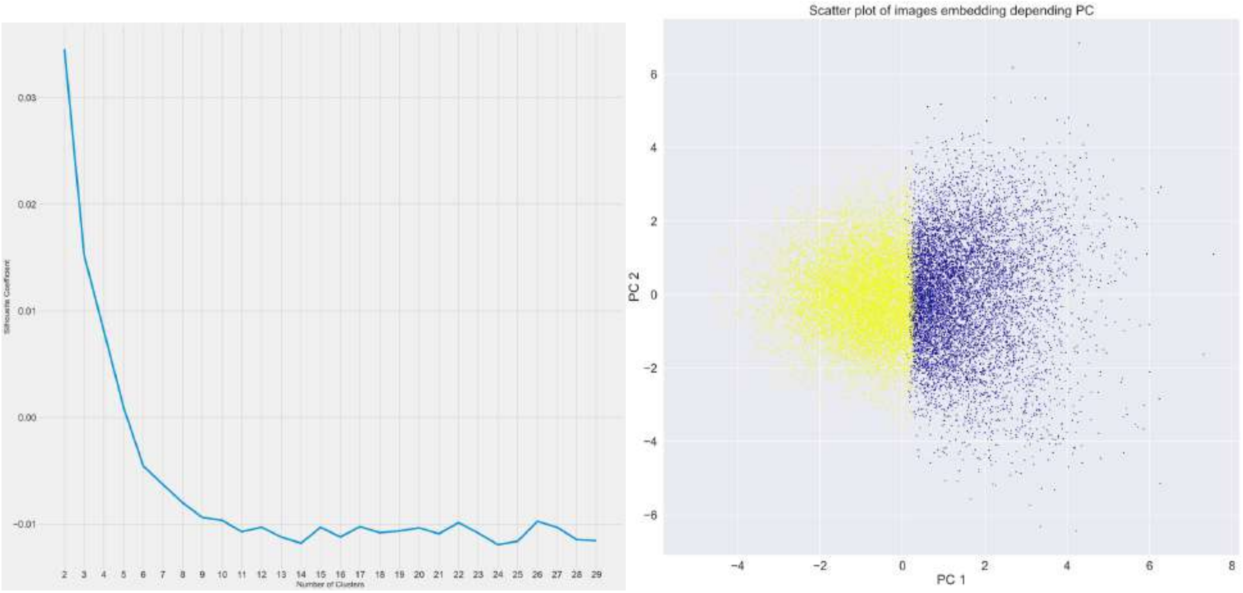
Figure 15: Heat map of the first two principal components.

6.3 K-Means testing

In the meantime, of the representativity tests, we experimented with another clustering0 algorithm (but the same original embedding model), K-Means, to see how it would behave with the embedding.

In the first test, we trained and applied the algorithm on the same data set (Kaggle data set), and we tried optimizing the number of clusters by using the silhouette score. His score fits a number of clusters equals to 2 (figures 16 and 17), which is explainable with the spatial representation where the points are stacked in a block. Therefore the number minimum of k is 2 in the silhouette algorithm, the number of clusters minimizing the variance between the clusters in a block is 2.

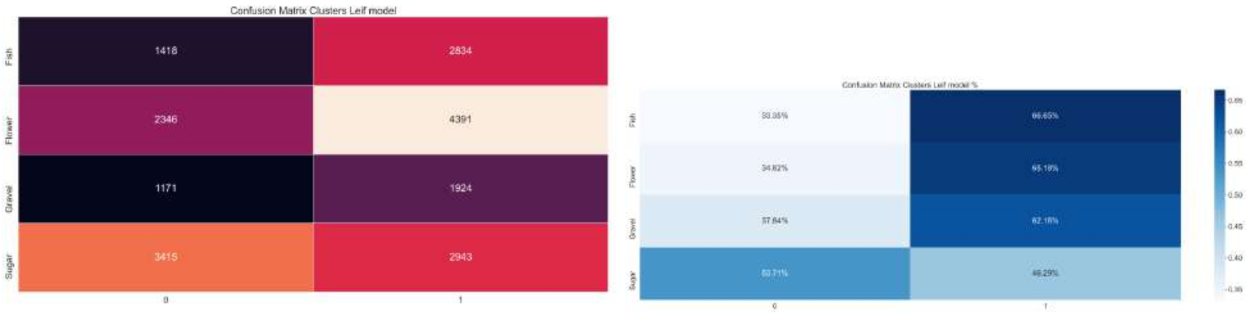
The corresponding results in confusion matrices (using, as explained before, the tiles' original rectangles classes) give us also no significant correlation as we can see in figure 17.



(a) Evolution of silhouette score by the number of clusters.

(b) Corresponding spatial representation.

Figure 16: Optimized K-Means clustering.

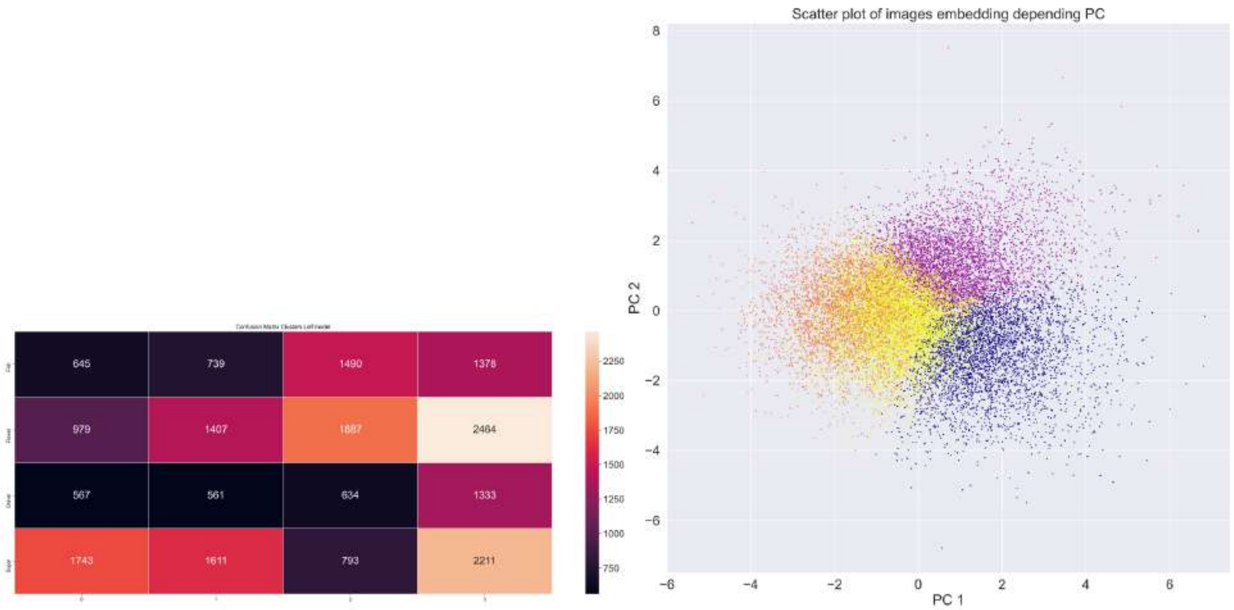


(a) Confusion matrix.

(b) Percentage confusion matrix.

Figure 17: Corresponding confusion matrices.

And because the human labels suggested 4 classes; we also forced a number of clusters of 4 in the K-means algorithm to see how it would compare to, and as we can see in the figure 18 the results were no better.



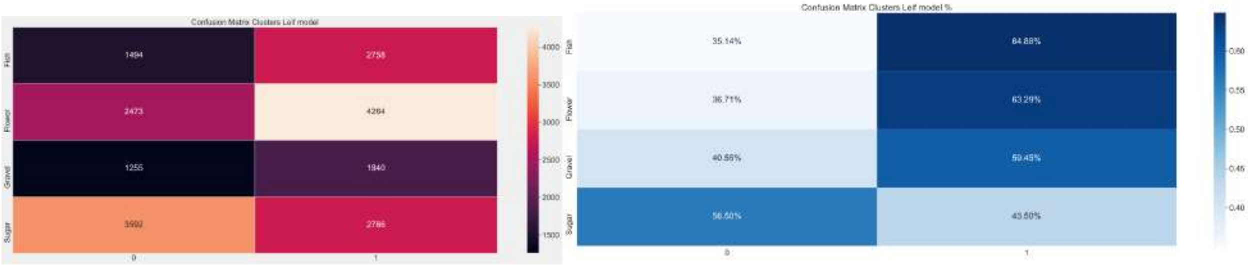
(a) Confusion matrix.

(b) Percentage confusion matrix.

Figure 18: K-means with K=4.

The final test was by training K-means on the data set used for the embedding model training (GOES images) and then simply use it on the new data and fixing the clusters' centroids coordinates on the Kaggle's data set.

As before and as we can see in the figure 19 the results were still no conclusive with this model, which, added to the first tests, led us to think that there was work to do on the embedding model itself.



(a) Confusion matrix.

(b) Percentage confusion matrix.

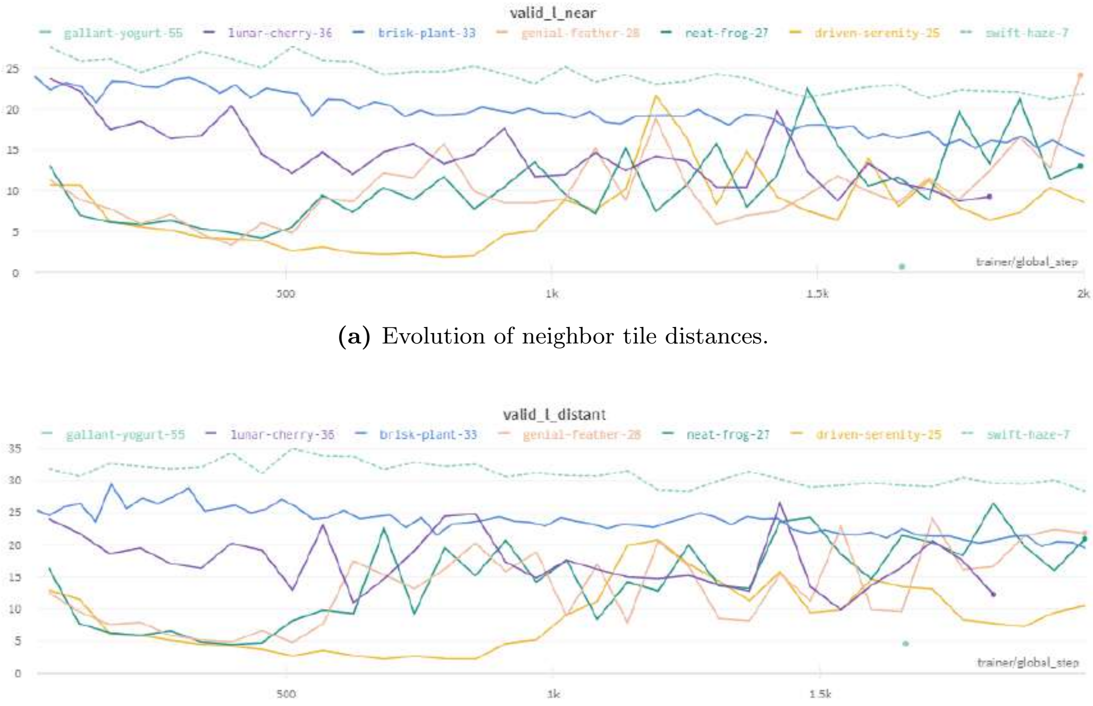
Figure 19: K-means trained on GOES images embedding.

7 Training a new model for Zooniverse data

In order to improve the ability to detect the supervised cloud classes with the unsupervised method, we trained a new model by using the Kaggle data set images and by following this methodology for the embedding training :

- Divide each image of the data set in equal 256 by 256 pixels sized tiles.
- Loop in each tile, consider it as an anchor, the neighbor tile would then be a tile in a distance of 128 pixels (so we can get a 50% overlap) in a random angle, and the distant tile would be extracted randomly from another random image.

We then train several models (with different backbones going from ResNet18 to ResNet101 with or without fixed parameters) using different architectures for the feature extraction part and different batch sizes and learning rates, and we took the one which minimized the neighbor tiles' distance and maximized the distant tiles' distance the most, as we can see in the figure 20. But the best interpretable metrics are the following one : explained variance and comparison with existing classes; that we will observe in further steps.

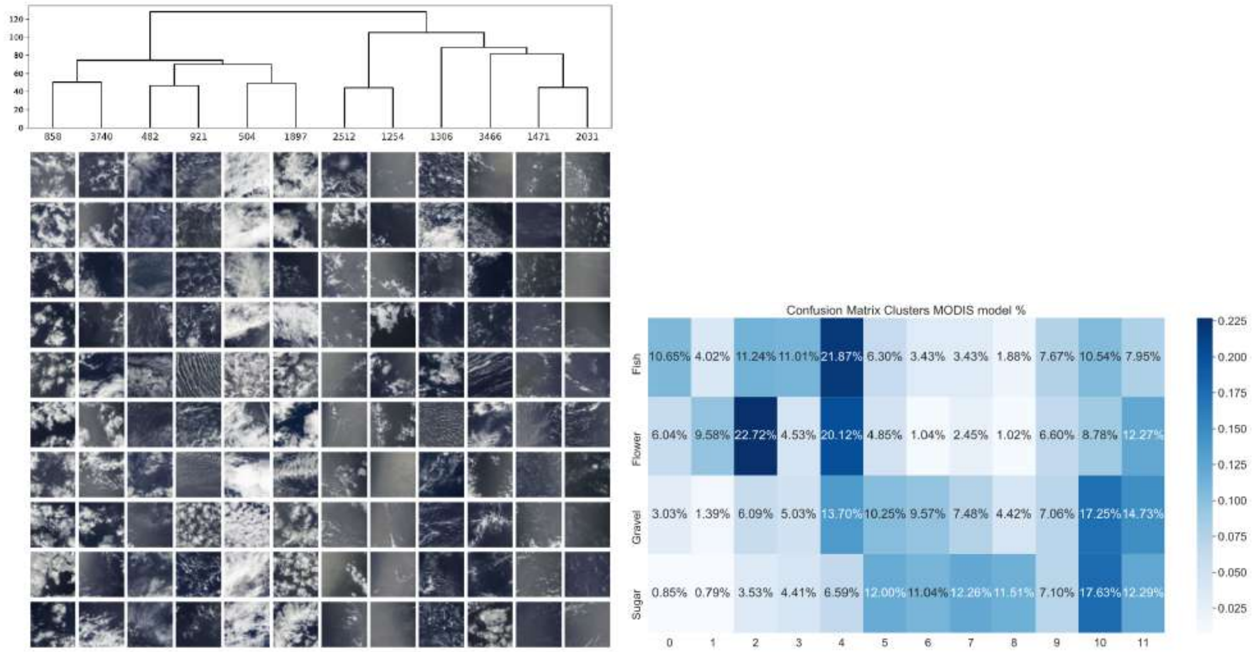


(a) Evolution of neighbor tile distances.

(b) Evolution of distant tile distances.

Figure 20: Training on Kaggle's data set.

Then, for the testing and clustering part, we took each of the rectangles labeled by the users, and independently of their original size was we resized them to 256 by 256 pixels (without trying to preserve aspect ratio) tiles and fed them to the embedding model. This new methodology gave us different results, and as we can see in figure 21, the different clusters have really visually additional images in terms of brightness level (cloud coverage) or forms. On the other hand, the clusters, as shown by the confusion matrix, seem to really discriminate nicely between the two groups of classes: Fish/Flower (clusters 0 to 4) and Gravel/Sugar (clusters 5 to 8).



(a) Hierarchical clustering.

(b) Percentage confusion matrix.

Figure 21: Clustering with the new model on resized users rectangles.

Then, by applying the same model on 256 by 256 pixels tiles contained in rectangles, we get, as shown in figure 22 clusters separation well more focused on brightness and gray levels than on forms as we can see in the clusters images examples. Within a label image which can have a large scale, different types can actually occur (especially for the Fish classification).

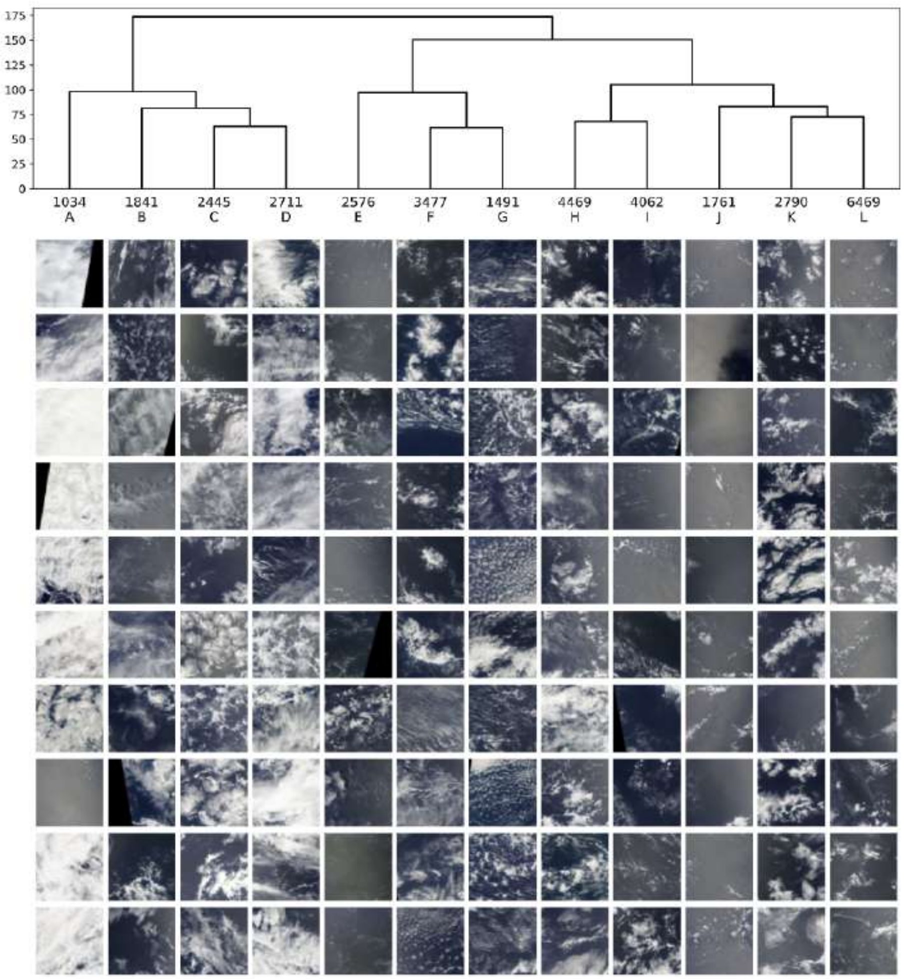
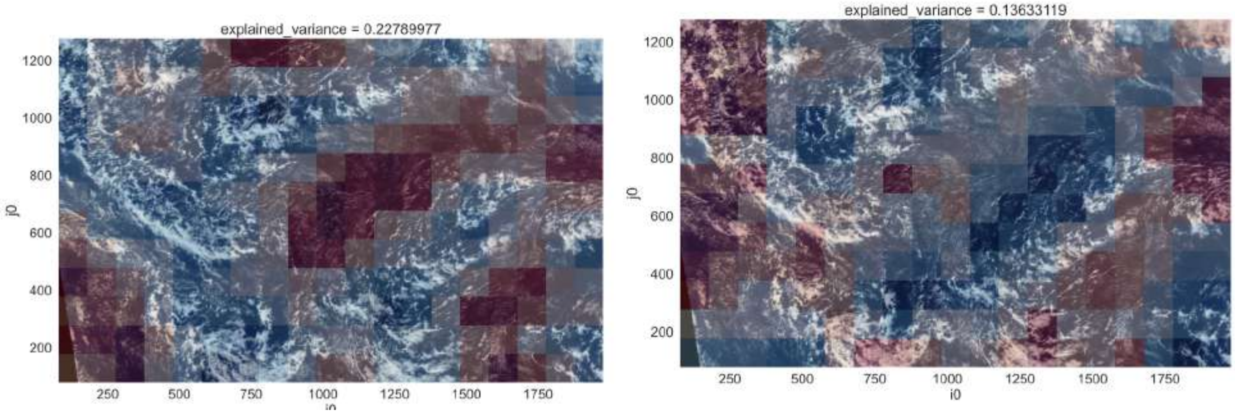


Figure 22: New model applied on divided tiles.

We again use the principal component analysis and take each component as a heat value on a domain. Figure 23 shows that we obtain (1) higher explained variance values, (2) continuity in the heat levels and (3) more interpretable results, for example, the first component seems to be really strongly associated with clear skies.

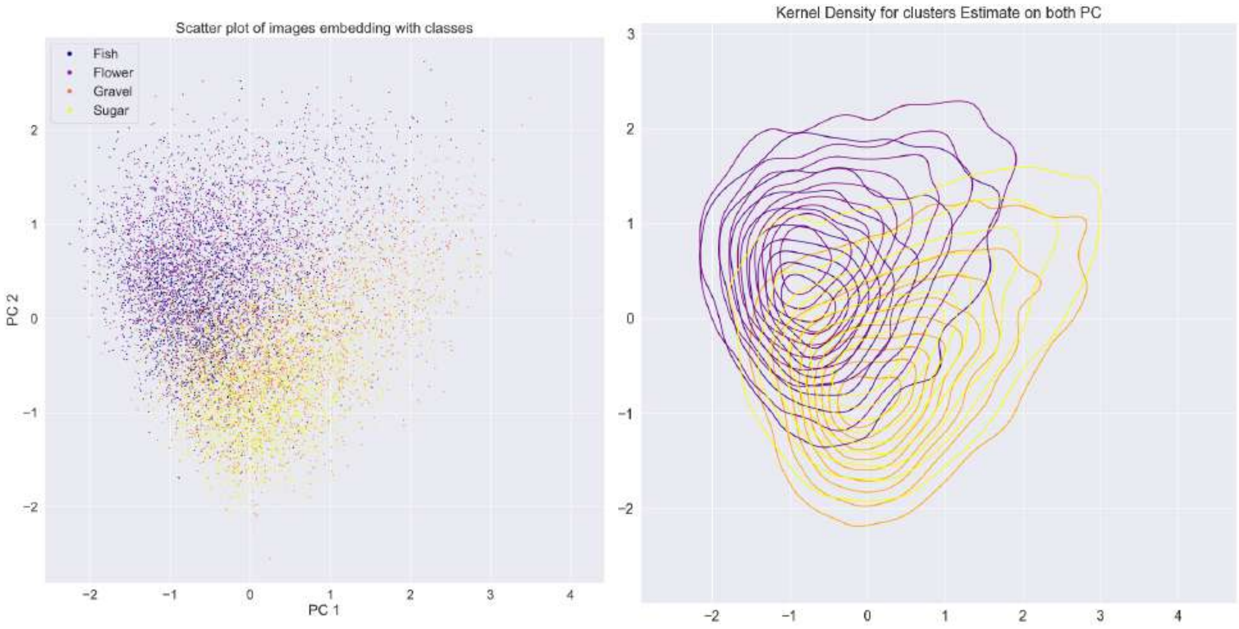


(a) First principal component heat map.

(b) Second principal component heat map.

Figure 23: Principal components heat maps using new model.

As a final comparison we plot the images by their first principal components (figure 24). We can see that this new model distribute way more equally the images, and also that the two groups of classes seem really well separated by those components (especially figure 24 (b)).



(a) Distribution Estimate for both labeled and unlabeled clouds.

(b) Kernel Distribution Estimate for both labeled and unlabeled clouds

Figure 24: Visualization of the distributions.

From this work analysis, we concluded that we could apply this methodology to various cloud images. Still, it was necessary to dedicate and train a specific model for the acquisition method associated with the data set.

We also observed the efficiency of the unsupervised algorithm to separate groups of cloud classes with the smallest possible deviation from the existing classes even it lacks some accuracy to distinguish in-between the two groups: Fish/Flower and Gravel/Sugar.

In the following part, we tried other algorithms and approaches to study the models' interactions with cloud images.

8 Classification and GradCAM algorithm

In this session, we describe the multiple methods we used to better understand the way the models work and interact with the cloud image.

As a first step we try [7] the GradCAM algorithm (see section 3.1.2). For that, we need a working classification algorithm, and that is why we train a new separate model for this classification task. We train multiple architectures on different data sets :

- The hand labeled data set from experts.
- The entire image of Kaggle's data set.
- Maximum sized square in the rectangles labeled from users to keep the aspect ratio when resizing for the model (this option gave us the best results by reaching 76% in precision).

We also trained different models in the first pass on ISSI data set but without getting good accuracy (never reaching more than 65% precision) which was explainable by many factors like the small size of the data set (245 images) and also the fact that the labels were associated to an entire domain for a dominant pattern even if there was a multiple of them.

The best models we got were when training (using a grid search and precision and recall as our metrics to compensate the unbalanced classes) on the Zooniverse data set with the content of the labeled rectangles as images and resized to a fixed shape (512x512 px). From there, we tried multiple architectures (different pre-trained networks and also different fully connected perceptron sizes). The best one was a ResNet-18 Backbone concatenated with four layers of the fully connected perceptron (100-30-10-5) with a SoftMax, which performed really well on the test set but did poorly on ISSI data set.

In figure 25 we can see that each color correspond to different architectures used for training, the "*earnest-yogurt-50*" being our ResNet-18 with the four layers multi layers perceptron.

[7] https://github.com/rayansamy/clouds-segmentation

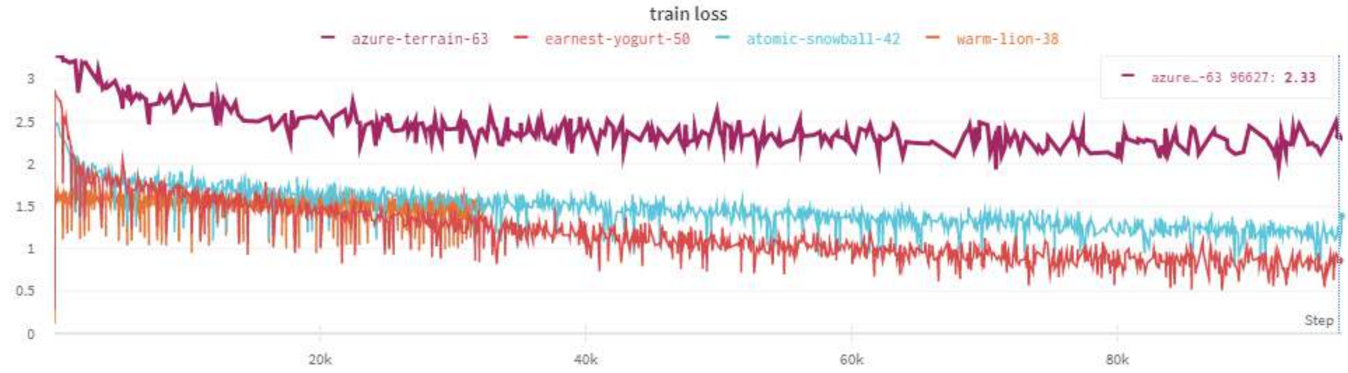
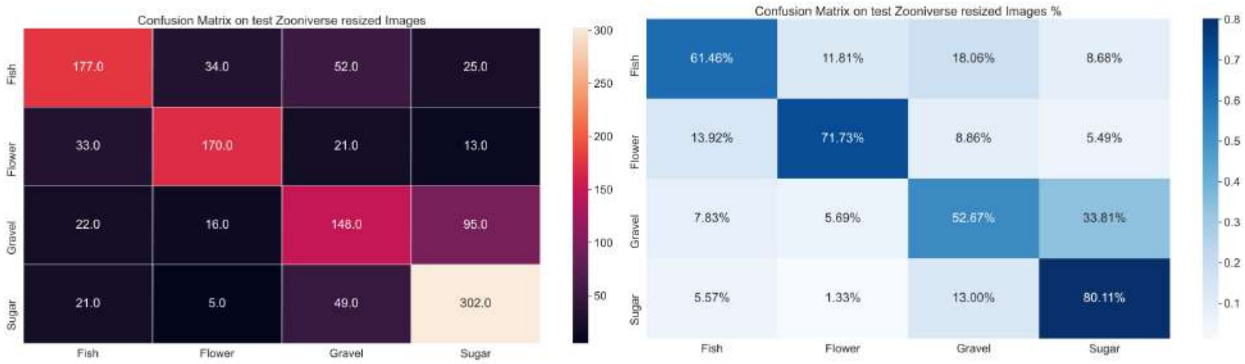


Figure 25: Training loss evolution.

By applying this classification model on the test set and computing the confusion matrices as shown in figure 26 we get consistent results, as in the clustering part, the two groups (Fish/Flower and Gravel/Sugar) are really split with good proportions, even if the Gravel class seems to underperform.



(a) Confusion matrix.

(b) Percentage confusion matrix.

Figure 26: Confusion matrices on Kaggle's test set.

But when applying the same model on different images from ISSI that were hand labeled by experts we get bad results as we can see in figure 27.

This result and the one we got by training a new embedding model specifically for Kaggle's data set which gave us better results, might lead us to understand that there are differences between the data sets in values or acquisitions and that would explain the fact that models seems to work only in their own data set images.

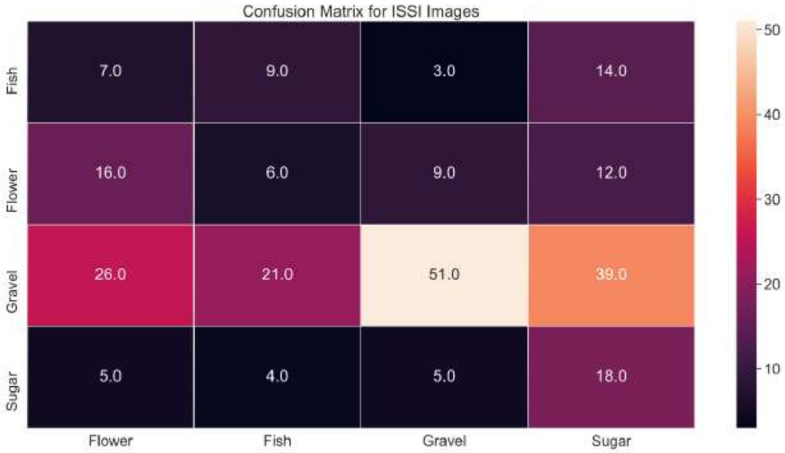


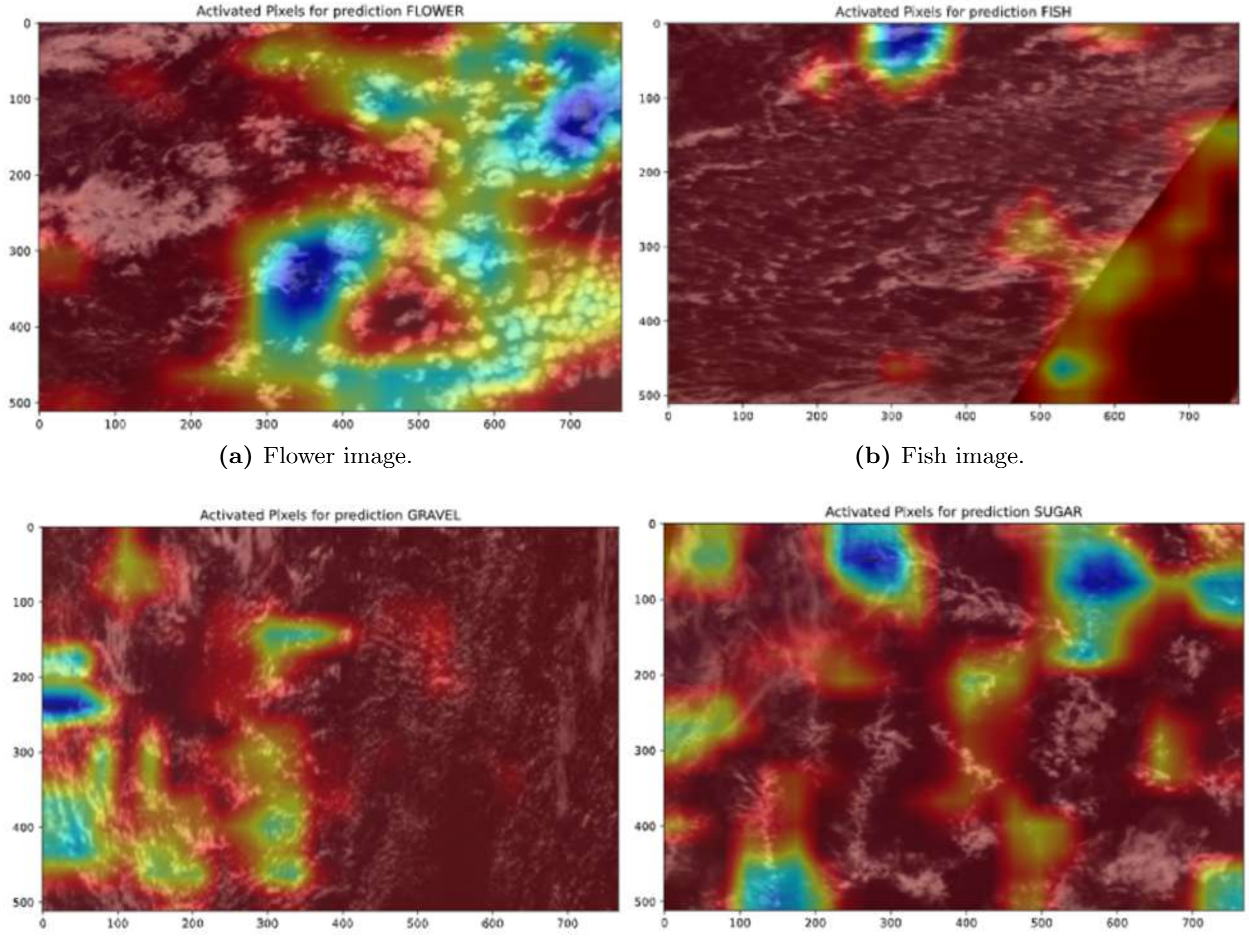
Figure 27: Confusion matrix on the expert labeled data set.

8.1 GRADCAM on Zooniverse predictions's rectangles resized

We then apply the GradCAM algorithm using this new classification model on images from the test set of Kaggle. As shown in the figure 28, we can see in orders in each quatuor of images, pixels activating the output for: fish, flower, gravel, and sugar.

We can see how the flower output, for example, is activated on the parts that would also help a human to label its flower as well, and that is the same case with the fish class, but the other two categories do not follow the same path.

Note that because we are using images with a lot of cloud free regions (water regions), we inverted the usual heat-map colors to get a better visualization.



(a) Flower image.

(b) Fish image.

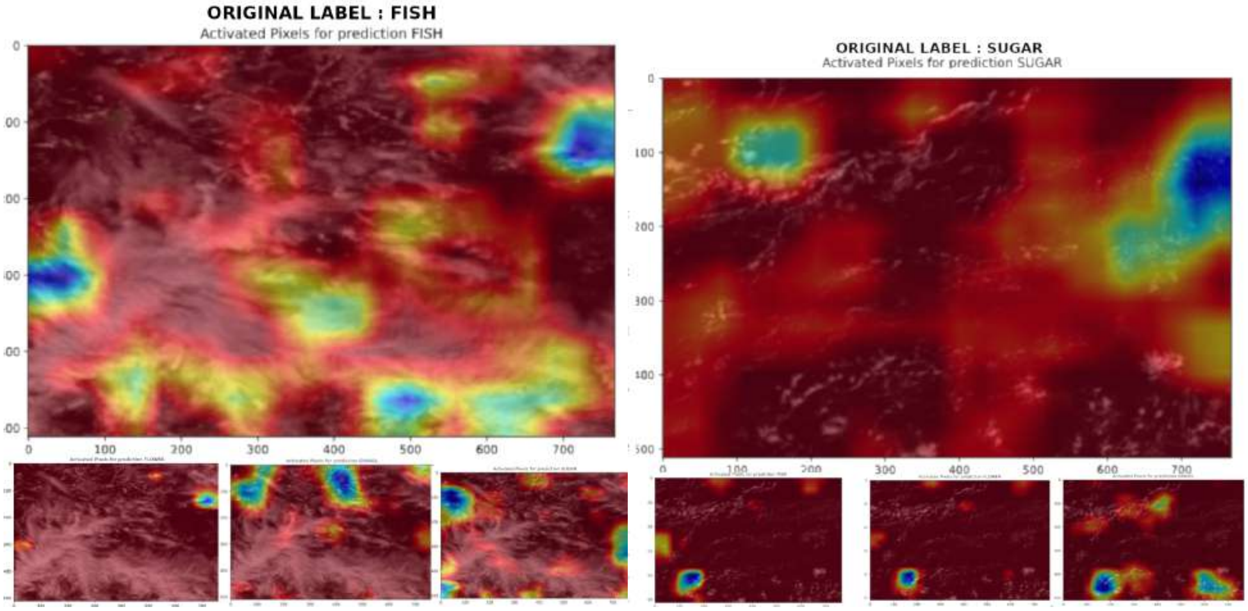
(c) Gravel image.

(d) Sugar image.

Figure 28: GradCAM of each class.

8.2 GRADCAM on others data sets

We then apply GradCAM also on the same model, but using images from other data sets, and the resulting heat maps do not make any interpretable sense (as we can see in figure 29), which can be correlated with the bad confusion matrix we got on figure 27.



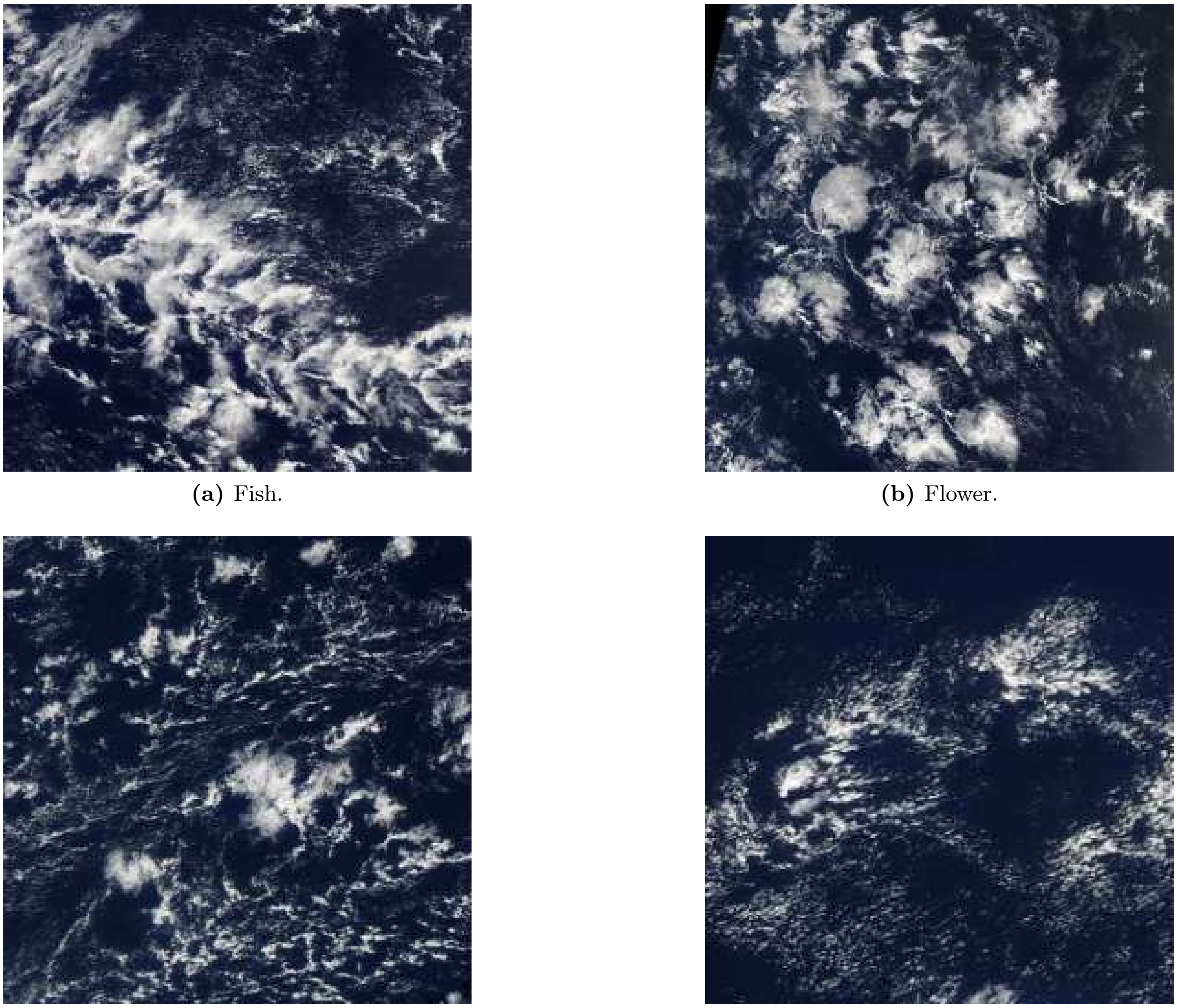
(a) Image 1.

(b) Image 2.

Figure 29: EUREC4A images.

8.3 Feature Maps

In addition to GradCAM, we extract the feature maps from an image of each class to visualize at multiple layers. We can see in figure 30 the pictures we used for this process.



(a) Fish.

(b) Flower.

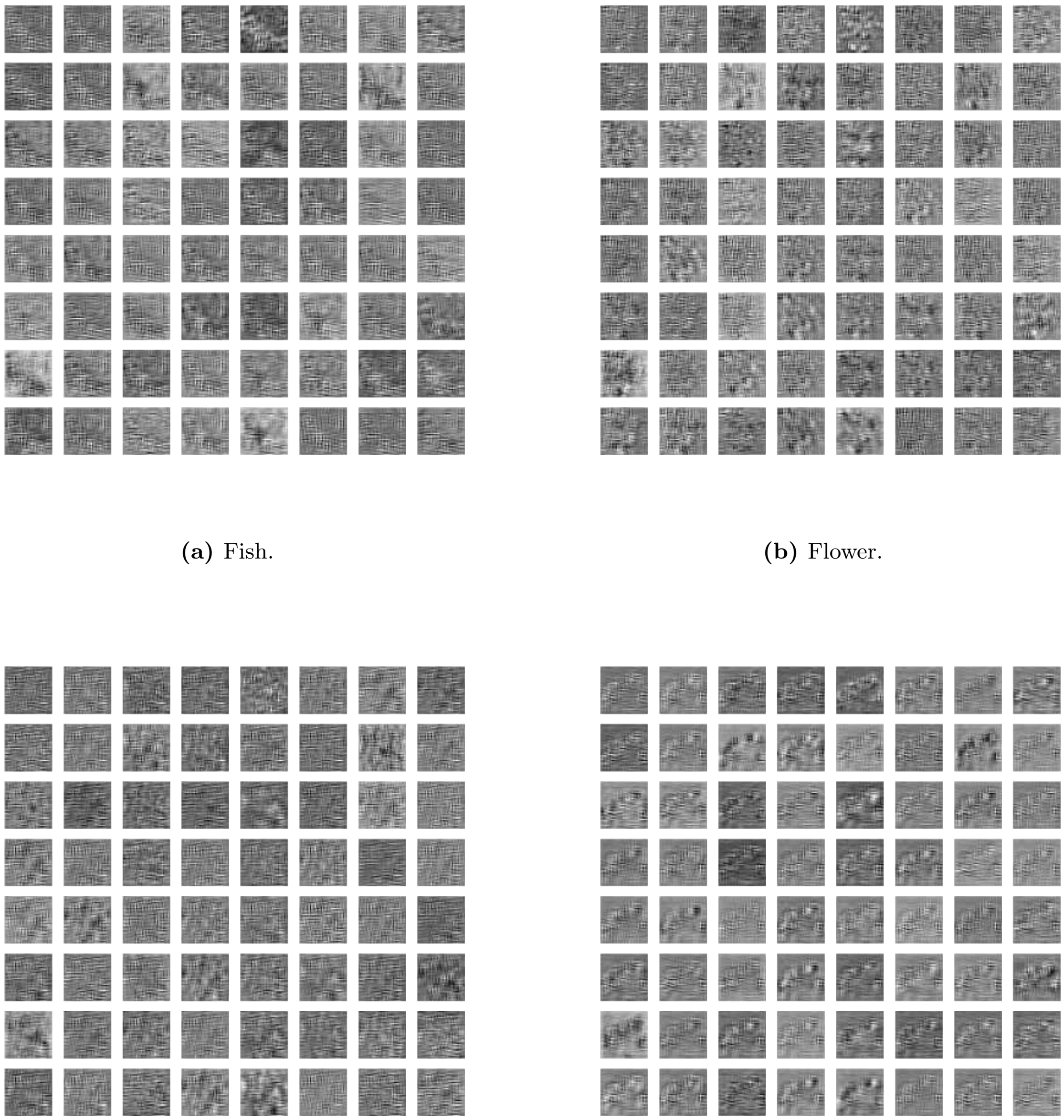
(c) Gravel.

(d) Sugar.

Figure 30: Original images.

By looking at the original images (figure 30 and the convolution filters of the last convolution layer (figure 31), we can see a consequent level of compression in the information, which can still be effective for the detection of macro forms in the clouds but might cause problems in the detection of more detailed microelements in the clouds (which could help distinguish between gravel and sugar for example), this observation might lead to two different options then :

- Using a pre-trained architecture might be interesting to try different cuts in the convolution layers.
- It might be interesting to train a complete model (convolution layers and multi-layer perceptron) and analyze if a fresh model would learn a more efficient feature extraction and compression.
- We could use a model with stacking feature maps like DenseNet (Huang et al. 2017).



(a) Fish.

(b) Flower.

(c) Gravel.

(d) Sugar.

Figure 31: Feature maps of the last convolution layer.

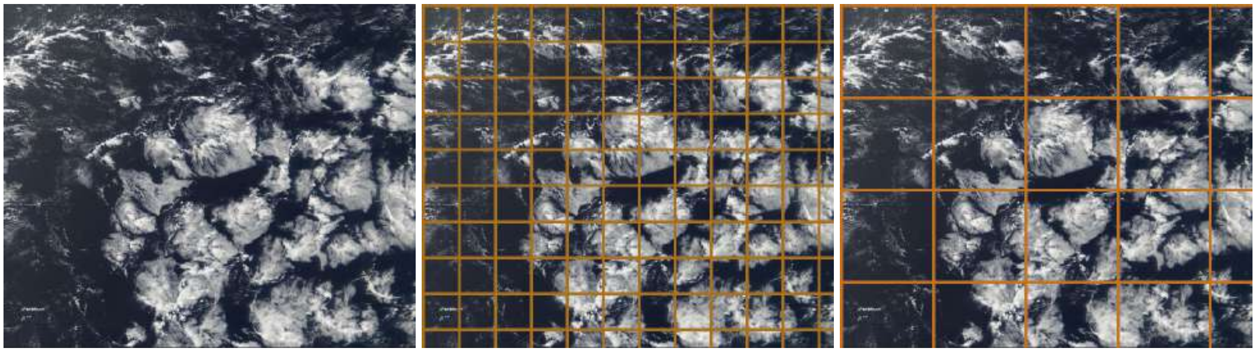
9 Multi-scale analysis

Scale is a prominent factor in clouds studies, they vary on very different length scales, and some classes appear to be visually dominated by specific length scales.

9.1 Scale Study

. We analyzed the impact of the spatial scale of an organization on the efficiency of our model trained on the Kaggle's data set.

We extracted for multiple sizes (in terms of pixels and therefore kilometers) all the tiles of the defined length in the user rectangles having this size as we can see in the figure 32. We limited the maximum number of total tiles for each scale to 10 000 to limit the computation time



(a) Original Rectangle labeled as Flower.

(b) 50x50 division

(c) 128x128 division

Figure 32: Examples of dividing a labeled rectangle.

We then compute the tiles' principal components, plot them on the axes, and plot one image example for each cluster.

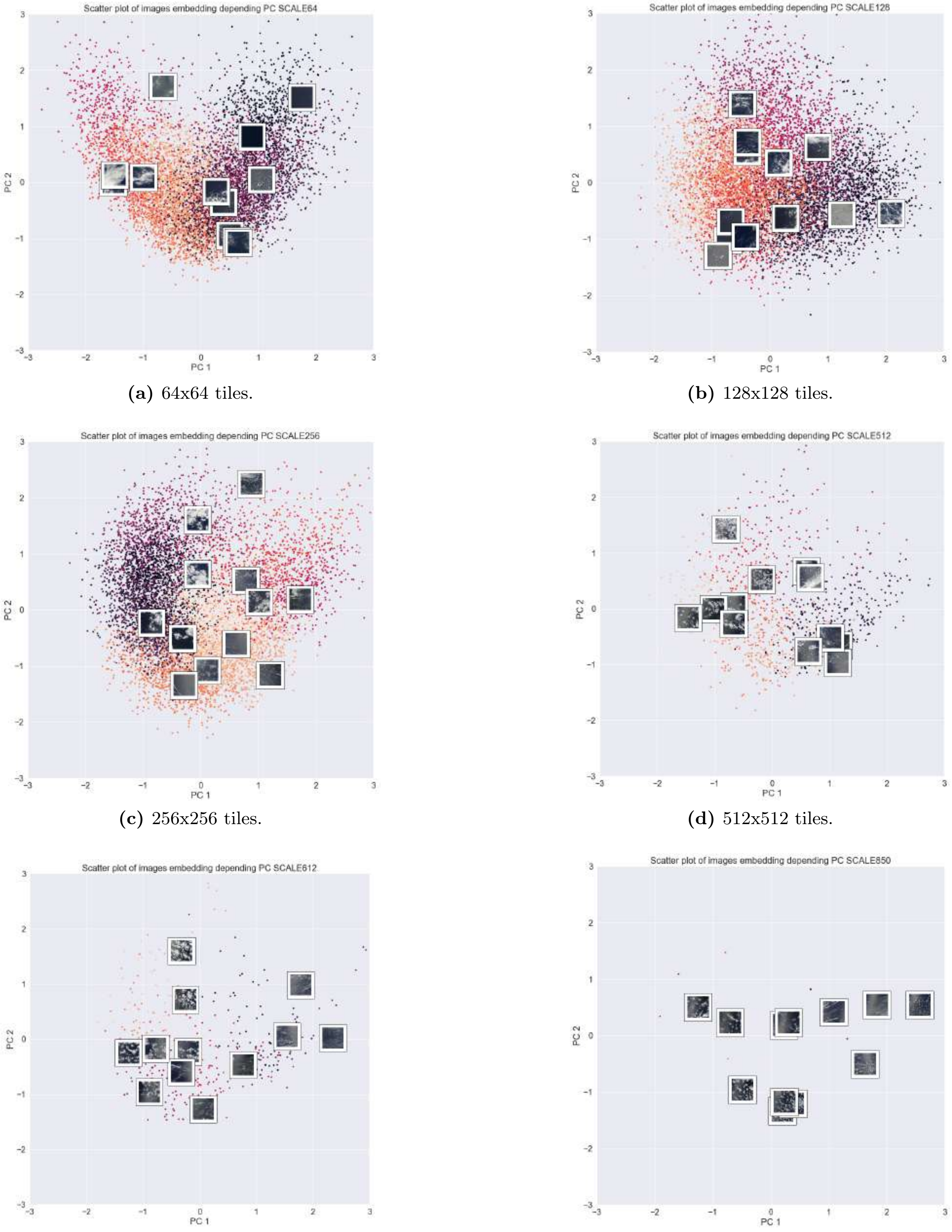
We see in the figure 33 (where each color corresponds to a different cluster) that by increasing the pixel size of tiles, there is less and less concentration of images on the upright part of the axes. Also, this upright part seems to be associated with homogeneous images (either the clear sky or covered), and the down left part appears to be related to big blocks of clouds separated by significant margins.

The two observations might be linked and understanding that by increasing the size of the tiles, there is less probability of having homogeneous images.

The table 1 shows different information about each scale, and their NMI score tends to confirm our suggestion.

Scale	Number of tiles	N°Flower	N°Fish	N° Gravel	N°Sugar	Explained Variance Ratio	NMI
64x64	10k	3075	2432	1763	2730	0.32224	0.007
128x128	10k	2403	2816	1722	3059	0.33503	0.019
256x256	10k	2022	3322	1442	3212	0.34923	0.06
512x512	1172	209	461	139	363	0.15878	0.158
612x612	424	64	183	51	126	0.3537	0.192
850x850	22	1	14	3	4	0.43878	0.424

Table 1: Evolution of Silouhette Score



(a) 64x64 tiles.

(b) 128x128 tiles.

(c) 256x256 tiles.

(d) 512x512 tiles.

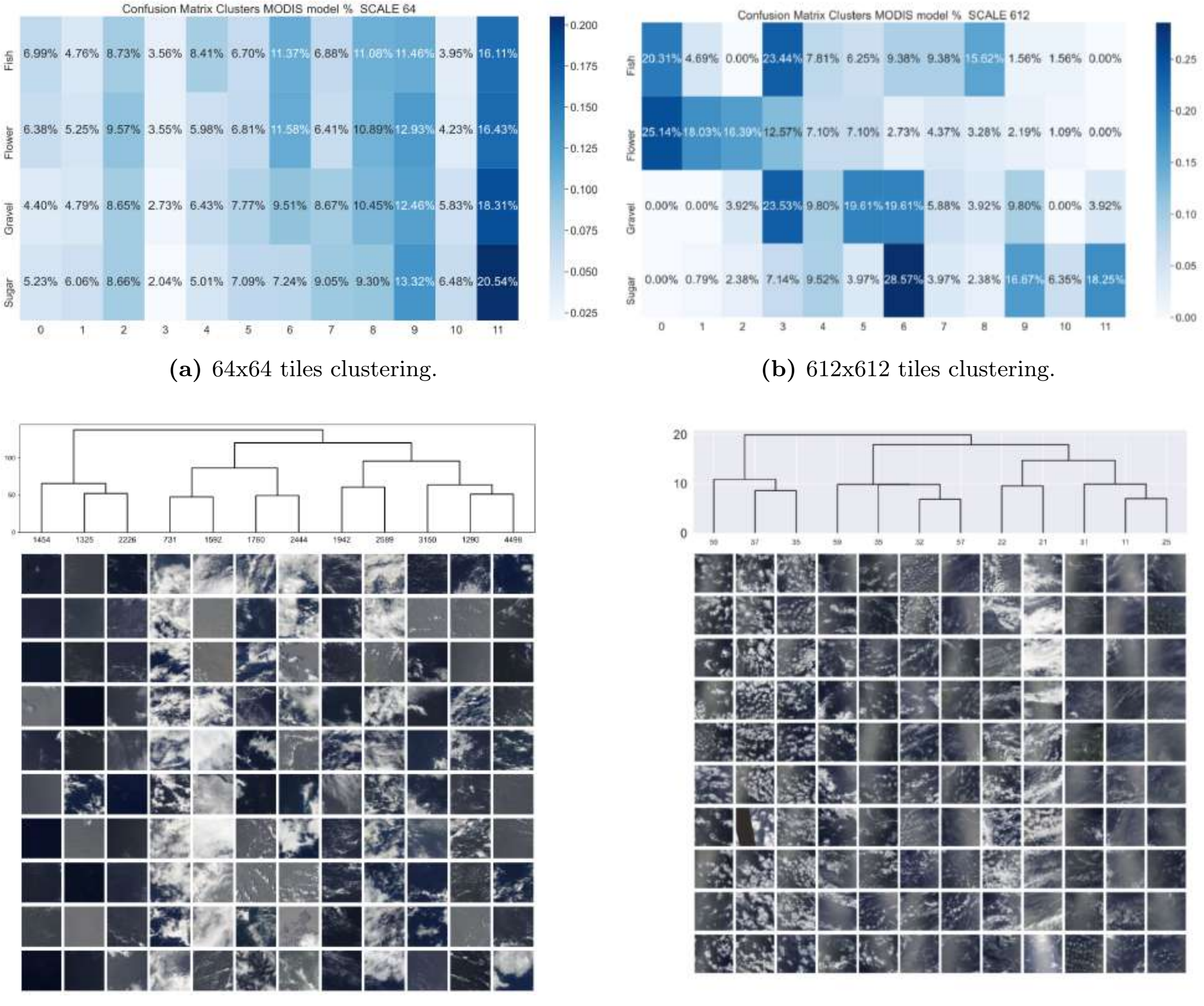
(e) 612x612 tiles.

(f) 850x850 tiles.

Figure 33: Spatial visualizations (clusters colors being independant from a scale to another).

In figure 34, by increasing the size of tiles, we can see that the clusters' differences based only on colors and brightness level changes to be more based on sparse cloud forms versus more homogeneous tiles. We can also see that increasing the tiles' sizes improves the prediction and confusion matrix results by better discriminating the two groups of classes: Fish/Flower and Gravel/Sugar.

For the 612x612 tile clustering; clusters 6, 9 and 11 are representative of sugar, cluster 0-2 are related to flower (as seen on the hierarchical clustering). Cluster 8 is related to fish, given the large cloud fraction but only the elongated shape of clouds as seen on the hierarchical clustering.



(a) 64x64 tiles clustering.

(b) 612x612 tiles clustering.

(c) 64x64 confusion matrix.

(d) 612x612 confusion matrix.

Figure 34: Multiple scales (of pixel by pixel size) clustering.

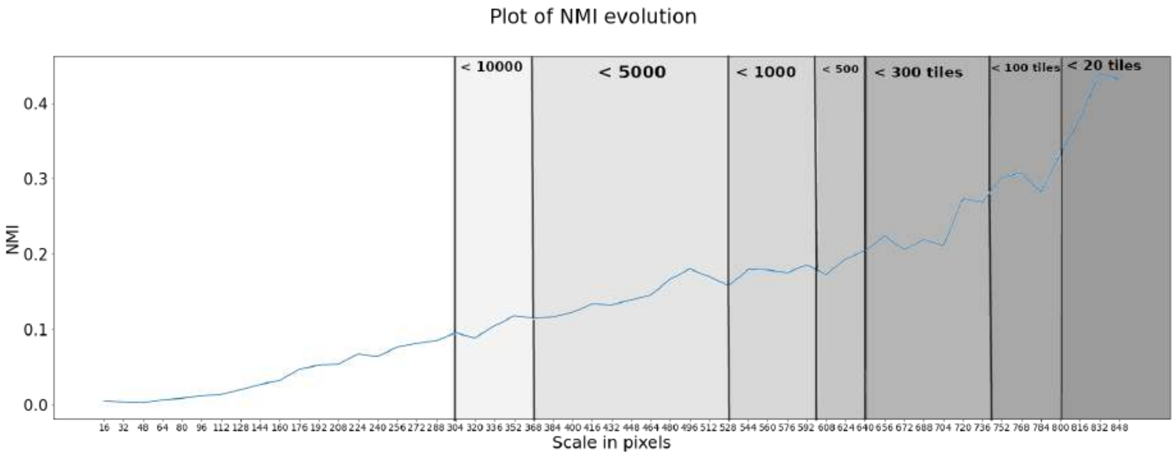


Figure 35: Evolution of the NMI score through the scales

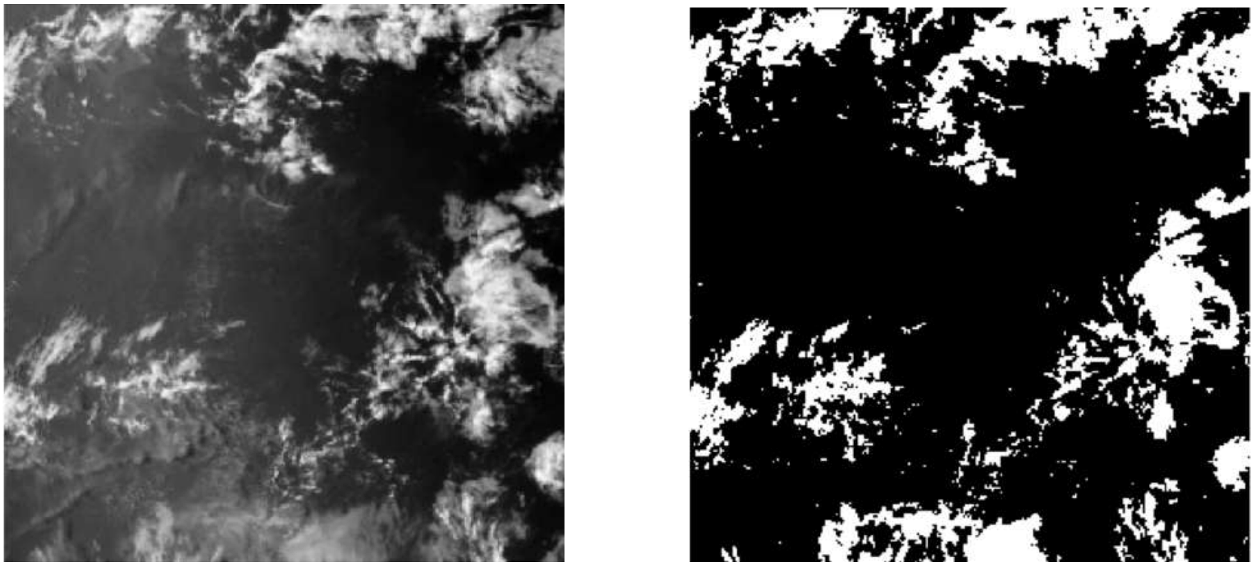
In figure 35 we can see the evolution of the NMI score through different scales (by testing each 16 px incrementation scale). We show a linear evolution in the score, with the exception of scales larger than 720x720 pixels which have a stronger evolution of the NMI score. However, the weak number of tiles available at these scales to perform the embedding clusters decreases the statistical significance of NMI. This linear progression finished to confirm our previous intuition that improving the scale will always have a good impact on the quality of the clustering using Tile2Vec.

9.2 Clouds Physical Aspects

We started studying the different correlations between the physical aspects of clouds and the other clusters at each scale.

To compute physical characteristics of clouds, they typically operate on a "cloud mask" rather than a scalar field. And that is why we need to produce a cloud mask by thresholding on the fields we have. We chose to greyscale the images then find a threshold value to produce the mask.

So first we needed to apply a thresholding on grey-scale version of the cloud tiles, we used a value for the threshold that extracted most of the cloud areas (a process that we can see in figure 36) , and from this binary images we then calculate the area of clouds in images, and apply object detection to count the number of "cloud objects". From those informations we can compute the needed metrics (mentioned in section 4).

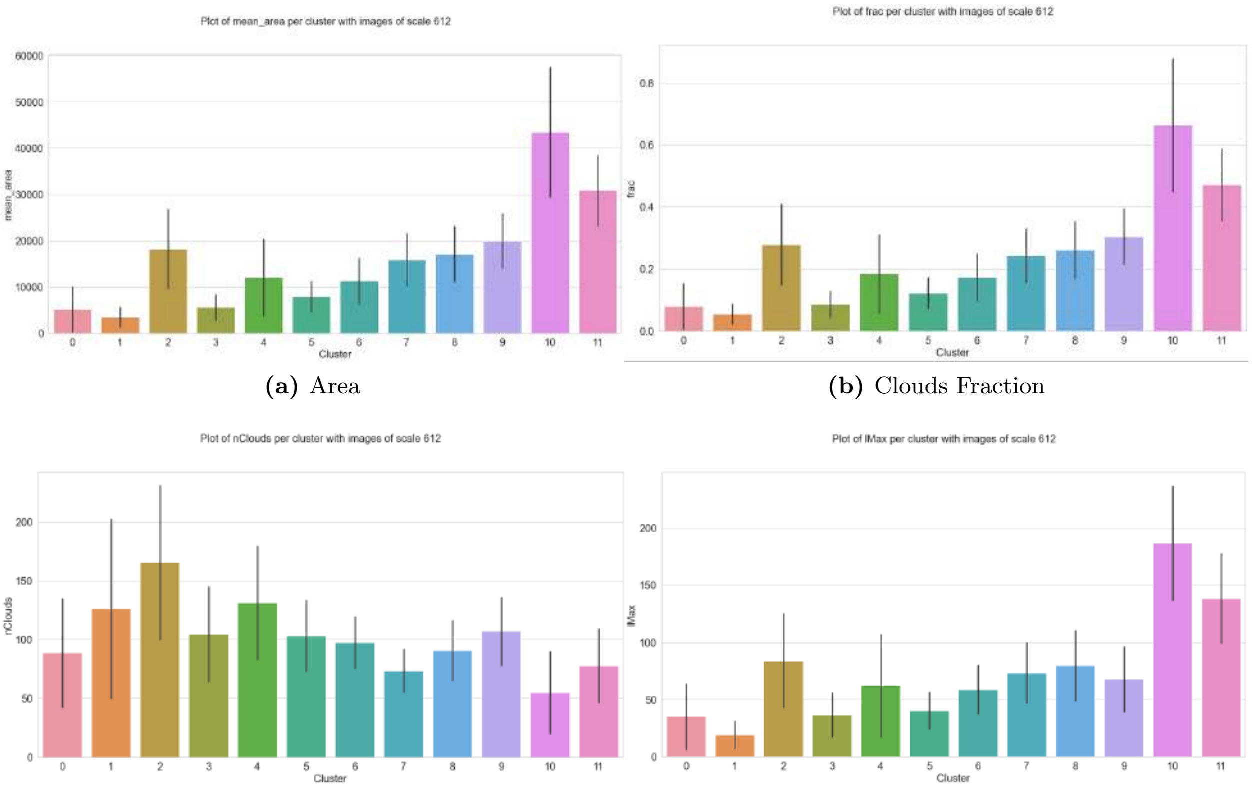


(a) Original greyscale version of a cloud tile.

(b) Thresholded version.

Figure 36: Example of our thresholding (binary thresh with a value of 127) of a cloud tile.

We took each tile corresponding to a cluster. Then we computed a variety of metrics (the one we mentioned in the first chapter). We can see the average and variance for each cluster and the metrics which differentiate the most between the clusters (at a scale of 612x612 px) on the figure 37. Those metrics has proven to be well correlated with the clusters we saw in figure 34 in terms of number of objects, their mean distances, .etc. Especially knowing the strong association with those clusters and the original classes.



(a) Area

(b) Clouds Fraction

(c) Number of Clouds

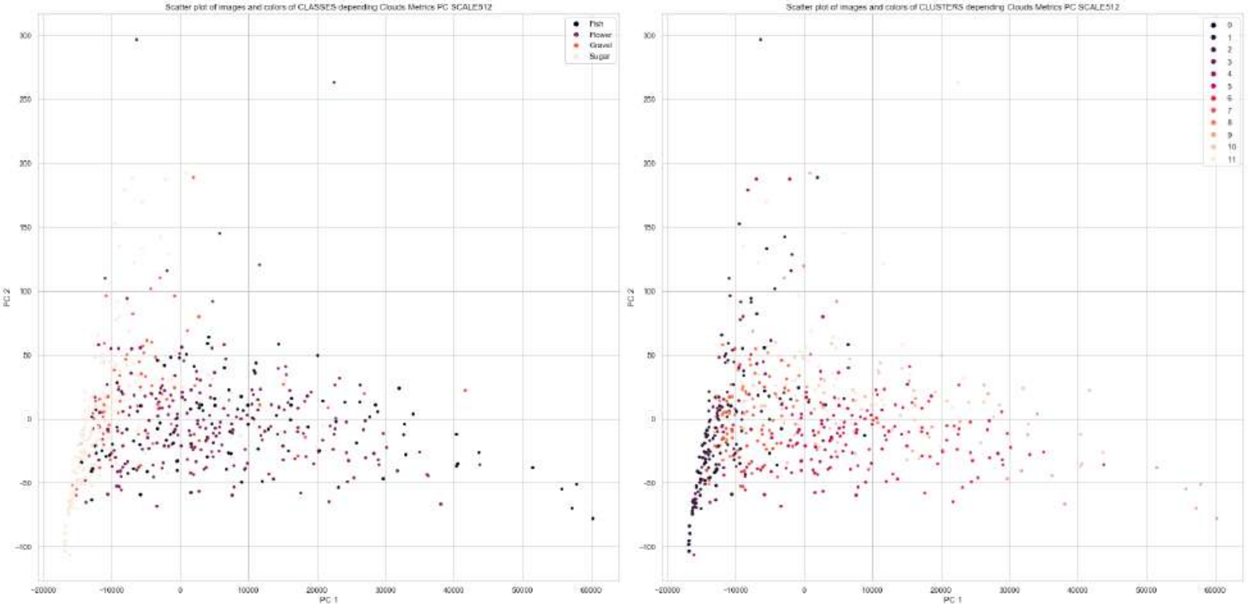
(d) Maximum length of cloud object in scene.

Figure 37: Cloud Metrics computed on a 612x612 scale and associated hierarchical clusters distributions.

Then, inspired by the work on Janssens, Vilà-Guerau De Arellano, et al. 2021 we applied principal component analysis on this set of metrics and represented each tile in the resulting main component axes. Finally, we associated on a hand to their initial classes (Figure 38 (a)) and, on the other hand, to their corresponding clusters (Figure 38 (b)).

From this computing, we noticed :

- A coherent association between clusters correlated with classes like flower or fish (figure 34 at a 612x612 px scale) and the standard metrics associated with those classes confirming our conclusions (figure 37).
- By analyzing those metrics in different increasing scales, the evolution of cloud coverage is decreasing, which confirms our conclusion of the previous subsection.



(a) Class Labeled

(b) Cluster Labeled

Figure 38: PCA on cloud metrics with colors associated to labels.

With this association, we proceeded to calculate the silhouette score for a clustering based on the clusters learned from the embedding and the one with the class labels associated to as mentioned in the table 2. Again, we can see a noticeable difference, the class labeling performing way better than the cluster one. Knowing that the PCA got an explained variance ratio on the first two components exceeding 99%, we could conclude that class separation remains better based on the physical and visual metrics of clouds.

We also calculated the contributions (or loadings) of each variable in the final components, and we noticed that the variables that maximized the variance were the cloud's mean areas and the number of clouds in a domain.

The conclusion is that the hand-labelled tiles are more distinct in terms of these physical characteristics than classes produced from hierarchical clustering of the embedding vectors.

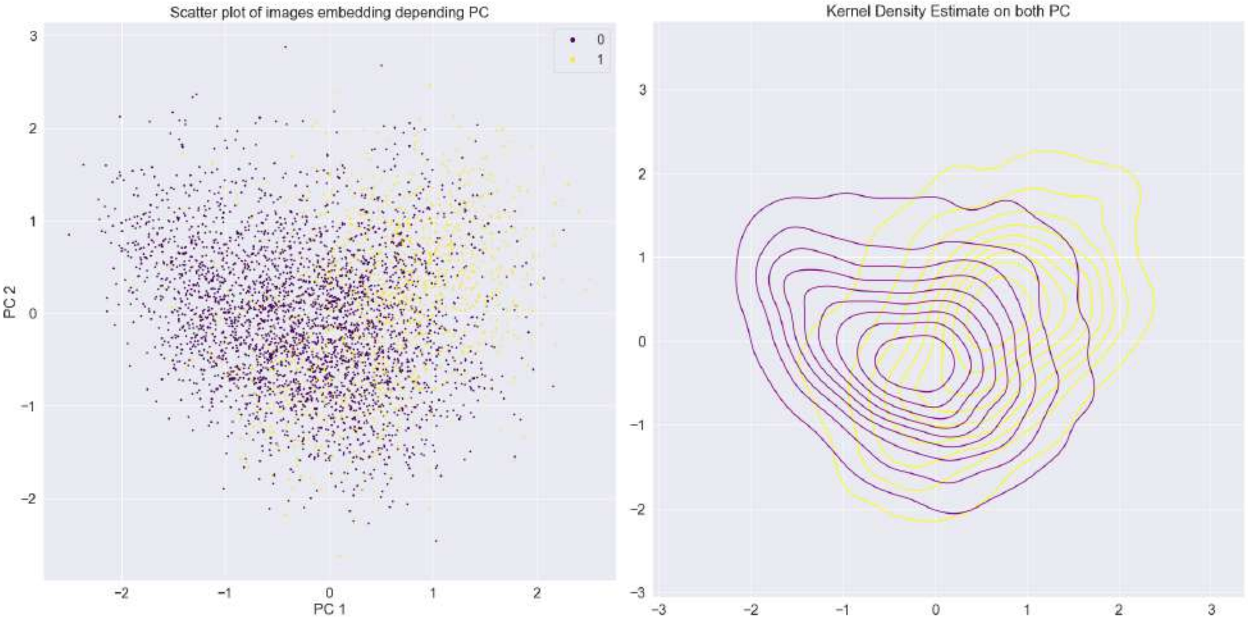
Scale/Label	Class	Cluster
256x256	- 0.0607115112553956666	- 0.27069403897696076
512x512	- 0.03543414516907521	- 0.208723318555586324
612x612	-0.06301196540787697	-0.17999969962399168

Table 2: Evolution of Silouhette Score

10 Unlabeled tiles

We are interested in finding relevant classes outside the 4 classes from the human labelling. Here the idea is to check whether the unlabeled parts of an image are related to one class more than others. In this purpose, we extracted tiles from images of Kaggle's data set and embedded them to pull their principal components and plot them as shown in figure 39. The labeled and unlabeled tiles seem to be mixed in the spatial representation, even if the unlabeled images (yellow points or class 1) seem to tend more to the left side of the space when the labeled tends more to the correct part. The tiles used here are the biggest square in prediction rectangles, and for the unlabeled part the biggest unlabeled square we can find on each image (with an inferior limit of size of 128x128 pixels).

Those results seem to indicate the existence of unlabeled data distributions which do not intersect with labeled distributions. This might be an indication of the presence of an unknown class recognized by the embedding model.



(a) Distribution for the tiles for both labeled and unlabeled clouds

(b) Kernel Distribution Estimate for both labeled and unlabeled clouds

Figure 39: Distributions of unlabeled data (class 1) and labeled data (class 0).

11 Other Embedding Methods

11.1 Deep Clustering

Deep Clustering (Caron et al. 2018) is one of the latest state-of-the-art method for unsupervised image clustering. It is based on an approach to iteratively group the features with a simple clustering algorithm like k-means. Then, it uses the resulting assignments as the ground-truth labels to compute the gradient and update the network's weights.

Overall, this method is based on alternating between steps of clustering the features from a pre-trained neural network, using the K-Means equation (5)

$$\min_{C \in \mathbb{R}^{d \times k}} \frac{1}{N} \sum_{n=1}^{N} \min_{y_n \in \{0,1\}^k} \| f_\theta (x_n) - C y_n \|_2^2 \quad \text{such that} \quad y_n^\top 1_k = 1 \tag{5}$$

Where : N is the number of images, θ are the parameters of a convnet f_θ, x_n is the image number n of the data set, associated with a label y_n in $\{0,1\}^k$ (the k possible clusters).

K-means learns a $d \times x$ centroid matrix C and the assignments y_n to clusters of each image n by solving equation (5).

The parameters W of this classifier and the parameters θ of the convnet are jointly learned by optimizing the function (6)

And then updating the parameters of the convnet (Pre-trained model + fully-connected layers with an output size which equals the number of clusters and a softmax) by using the label of the clustering as the ground-truth and minimizing the function (6).

$$\min_{\theta, W} \frac{1}{N} \sum_{n=1}^{N} \ell \left(g_W \left(f_\theta \left(x_n \right) \right), y_n \right) \tag{6}$$

g_W is a parametrized classifier which predicts the correct cluster using the top features of $f_\theta(x_n)$.

This approach has a substantial advantage over the previous image embedding method. First, it does not assume the fact that spatial distance is correlated with visual and semantic distance (two tiles being geographically distant does not necessarily mean a comparable visual difference). And secondly, it proves to have better efficiency in terms of performance and stability over the previous ones, which is justified by the fact that the training process of the neural net and the one of the clustering are combined, wherein previous methods were done separately

The performance of random convolutional networks is intimately tied to their convolutional structure, which gives a strong prior to the input signal. The idea of DeepCluster is to exploit this weak signal to bootstrap the discriminative power of a convolutional neural network.

In the paper Caron et al. 2018, they used a small AlexNet architecture which proves to learn more efficiently because of the data set size and the fact that they didn't freeze the pre-trained model weights. They observed that Deeper layers in the network seem to capture larger textural structures. However, the latest filters in the last convolutional layers merely replicate the texture already captured in the previous layers.

The figure 40 shows an overview of this method pipeline.

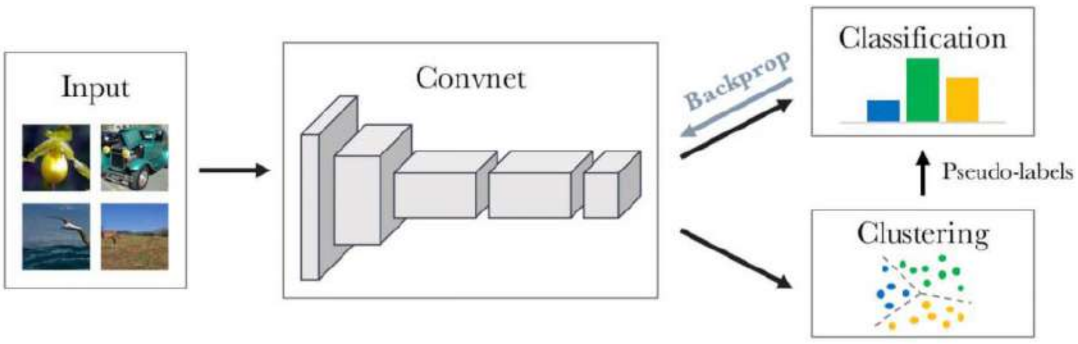
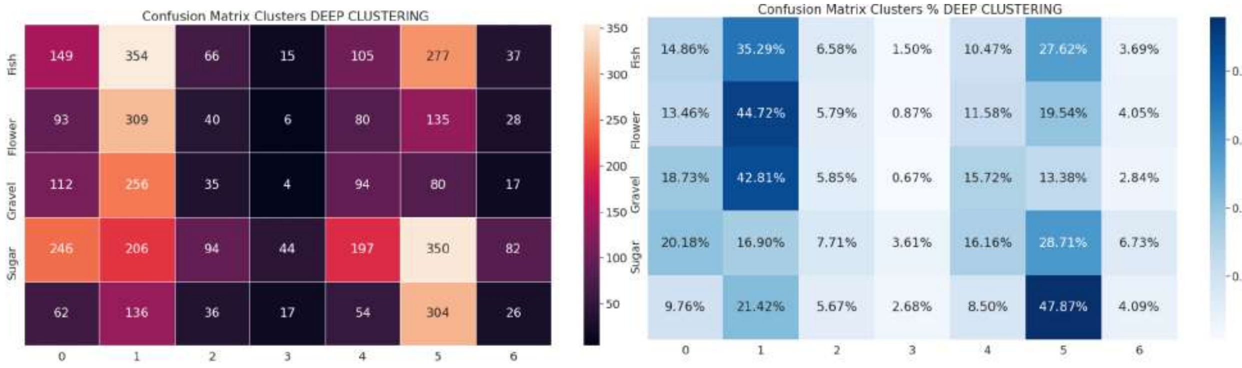


Figure 40: Deep Clustering overview © Caron et al. 2018

We implemented [8] the method and trained multiple models following this method (and on data set of biggest square rectangles extracted from labeled rectangles and unlabeled zones of images), using especially a VGG-16 which gave us the best results in terms of loss decrease. But as we can see in figure 41 the results of the clustering were not good enough in comparison to the previous method. We used 7 as number of clusters because it was the number which gave us the best efficiency : adding more clusters giving us each time less dense ones (with less than 6% of tiles associated to it).



(a) Confusion Matrix

(b) Percentage Confusion matrix

Figure 41: Deep Clustering confusion matrices for 4 known classes and unlabeled images.

The figure 42 shows random examples of images for each cluster, as we can see, there a in multiple clusters different classes of clouds. That confirms the confusion matrix results. The efficiency of this model might be improved in the future by, as mentioned in the original paper, working to use the best convolutional layer of the pre-trained model. For instance, they used a computation of max activation of each cluster associated output neuron with the different kernels of the layers. With this methdology it was possible to track, for each layer the outstanding images (best candidate of each cluster corresponding to the specific convolutional layer) and adapt the model to this information. This methodology might help trace the best convolutional layer for clouds and improve our results.

[8] https://github.com/rayansamy/Deep-Clustering-Cloud-Images

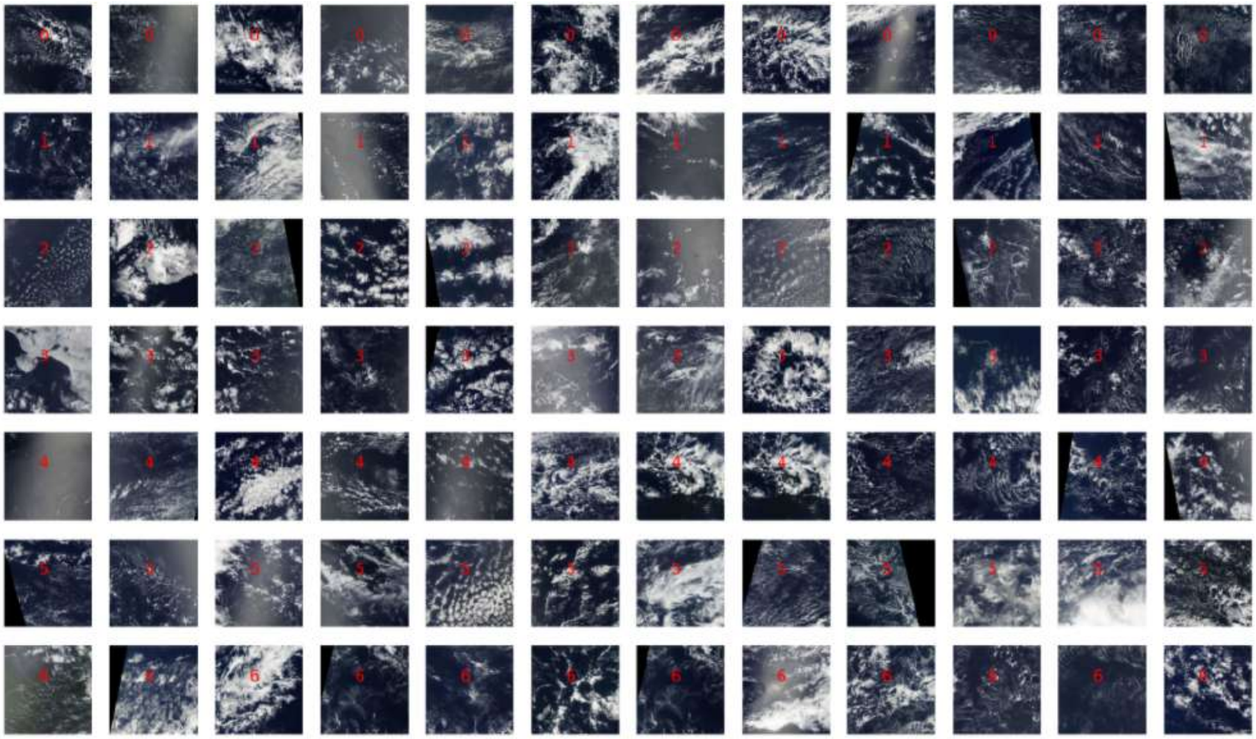


Figure 42: Example of images for each cluster

11.2 Deep Convolutional AutoEncoder

Deep Convolutional AutoEncoders (Alqahtani et al. 2018) or (DCAE) are unsupervised models for representation learning. They learns a mapping from images as inputs to a new representation space, from this learning they can learn a compressed representation of the original images. In difference of Deep Auto Encoders (DAE), DCAE uses in encoding part convolutional layers, to learn features of the compressed space (or called latent representation). This latent representation of the n^{th} feature map of existing layer is given by the equation (7)

$$h_n = \sigma\left(x * W_n + b_n\right) \tag{7}$$

Where W are the filters, b the corresponding bias of the n^{th} feature map and σ is the activation function and $*$ being the convolution operation.
In the decompression part of the architecture, DCAE uses deconvolutional layers, inverting and reconstructing the image from the latent representation.

$$y_n = \sigma\left(\sum_{n\in H} h_n * \tilde{W}_n + c\right) \tag{8}$$

Where $\tilde{W}$ is the flip operation over both dimensions of the weights, H the group of latent feature maps and c the corresponding bias.

This methodology allows to obtain useful features through the encoding procedure. The data is projected into a set of feature spaces, using the encoding part, from which the decoding part reconstructs the original data. The training is conducted in an unsupervised manner by minimizing the differences between original data and reconstructed data with distance metrics.

The major difference between a DAE and a DCAE is that the former adopts fully-connected layers to reconstruct the signal globally while the latter utilizes local information to achieve the same objective.

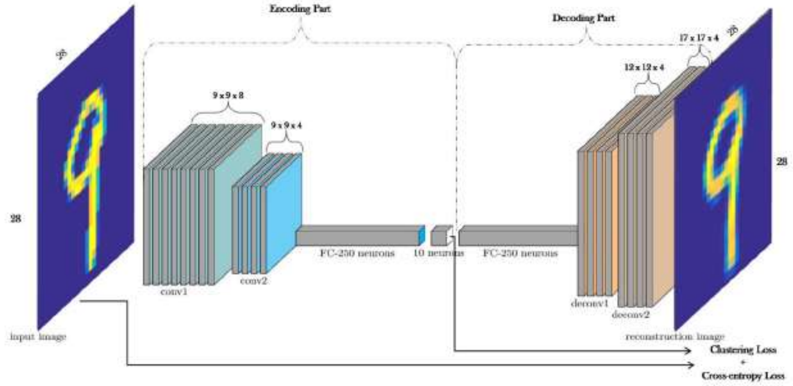


Figure 43: Example of architecture for a deep convolutional autoencoder © Alqahtani et al. 2018

This method that we implementend [9] is interesting because in addition to give a model for the compression/decompression of cloud images and having an embedding created for images, this also provides a way to visualize the information of our embedding (which then helps to adapt the size of the latent vector for example). This was not the case in the Tile2Vec method.

We used an architecture as shown in table 3

Encoder		Decoder
Conv2d(3,128,3,padding=1)		ConvTranspose2d(8, 16, 2, stride=2)
Conv2d(128,64,3,padding=1)	MaxPool2d(2, 2)	ConvTranspose2d(16,32,2,stride=2)
Conv2d(64,32,3,padding=1)	Flatten	ConvTranspose2d(32, 64, 2, stride=2)
Conv2d(32,16,3,padding=1)		ConvTranspose2d(64, 128, 2, stride=2)
Conv2d(16,8,3,padding=1)		ConvTranspose2d(128, 3, 2, stride=2)

Table 3: Architecture of our Deep Convolutional AutoEncoder

In figure 44 and 45 we can see examples of reconstruction using this architecture. As we can see, the global shapes seems to be respected, and that is also the case for the colors. But it lacks precision (which might cause problems for classes like sugar or gravel), and by using the deconvolution layers we can see their effect in terms of 2x2 blocks insides the reconstructed image.

[9] https://github.com/rayansamy/AutoEncoder-Image-Embedding

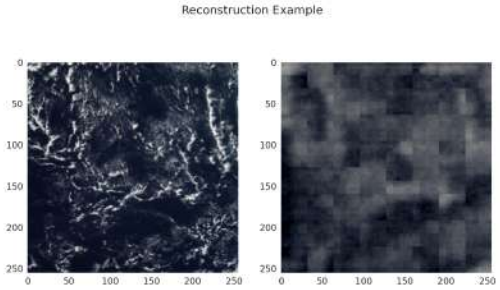


Figure 44: Reconstruction example

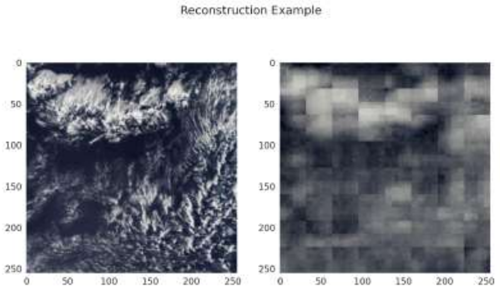


Figure 45: Reconstruction example 2

We can see in figure 46 and 47 that the quality of the clustering is not really efficient to separate the classes (4 known classes and the fifth one being unlabeled tiles).

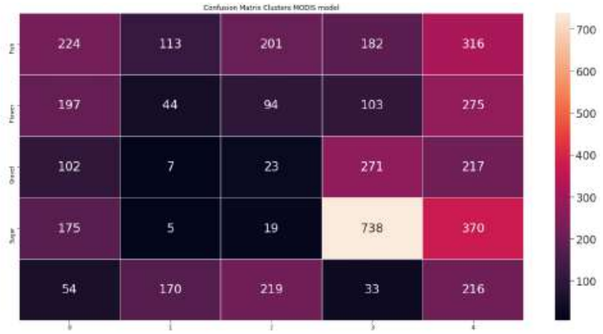


Figure 46: Confusion Matrix

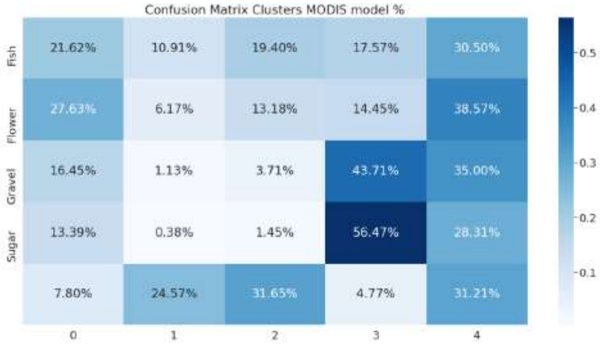


Figure 47: Percentage Confusion matrix

But, looking at the figure 48 we can observe that the clustering, even if it does not operate in terms of the classes that we know, seems to separate clouds in terms of cloud coverage first.

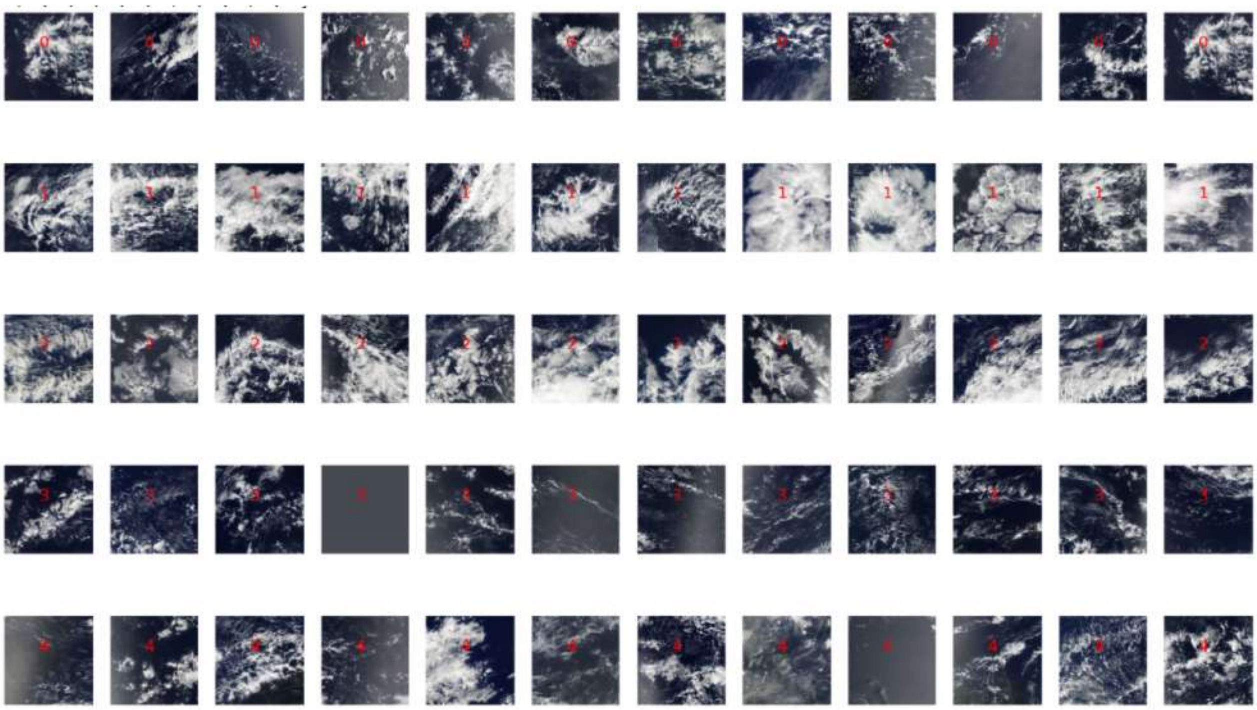


Figure 48: Example of images for each cluster

12 Conclusion and Outlook

In this internship, we applied several techniques of artificial intelligence (machine learning, deep learning, etc.) in the context of clouds classification. We knew the existence of 4 classes of spatial organisation of cloudiness, based on human labels and supervised learning. These 4 classes were the base of this internship.

The goal was to (1) understand : the way the different sets of clouds satellites data works, how to create the RGB images, and therefore study their characteristics, (2) apply the original model to our data sets and interpret results, and (3), establish a comparison between the unlabeled predictions and the labeled ones. and determine the different strengths and weaknesses of our methodology.

We studied the applications of Tile2Vec a methodology using spatial correlations of tiles in images to learn visual distances, in this context and arrived at the necessity of using a specific model for each acquisition method.

It was also determined that unsupervised methodology like Tile2Vec using rectangles of initial users labels could fit goodly enough the known classes and separate quite well the two main groups of classes: Fish/Flower and Gravel/Sugar. This work has proven that we are able to recover the four classes of clouds revealed by humans, and thus promises a deeper and less biased analysis of cloud spatial organizations, and their links to weather conditions.

This study also highlighted the essential impact of the scale factor. By improving the size of the tiles used for producing the embedding vectors, we improve confusion matrices, scores, and separations either in terms of original classes or of shape (intra-class similarities and inter-class dissimilarities). An objective metric for score measurement (NMI) allows us to quantify this improvement, but also the domain size when the separation becomes meaningful.

The models could create a good variance in data in terms of embedding space, but applying a clustering didn't maximize the variances of the clouds metrics as much as the known classes.

Tile2Vec helps separating Fish/Flower and Gravel/Sugar, but our multi-scale approach goes a step further and permits to discriminate different classes (identification of Fish for instance). Further investigations would help highlighting realistic cloud classes that do not suffer from cognitive biases induced by human identification.

We also learned that in this newly created space, and in the hope of finding any indication of a potential fifth class, the unlabeled and labeled data in terms of distributions are rather intersected and have some dissociate and separate spaces. This result might indicate the existence of a potential visual organization in the unlabeled data.

Finally we tried new methods of applying deep learning to clouds unsupervised clustering, which gave us new unbiased comparison references to Tile2Vec. But also new, even less biased approaches to cluster tiles and visualize and understand models behaviors.

In perspective, we could try to :

- Use state-of-the-art multi-scale object detection methods to explore the efficiency to detect each class on an entire domain.
- Experiment with architecture like pyramidal ResNets to keep more details after the compressions of convolutional kernels.
- Testing with multi-scale unsupervised object detection algorithms to tackle the problem of scale in the clouds.
- Improve our existing algorithms like deep convolutional auto encoder and the deep clustering for this specific task to get another view on the cloud unsupervised classification.
- Study in precision the distributions of unlabeled clouds, especially the ones which do not intersect with the labeled distributions.
- Use heat-map extracted from the PC values as a segmentation method (within some threshold) and study its accuracy.

References

Alqahtani, Ali, X. Xie, J. Deng, and Mark Jones. 2018. "A Deep Convolutional Auto-Encoder with Embedded Clustering," 4058–4062. October.

Benner, T.C., Judith Curry, and James Pinto. 2001. "Radiative transfer in the summertime Arctic." *Journal of Geophysical Research: Atmospheres* 106 (July): 15173–15183.

Bony, Sandrine, Bjorn Stevens, Felix Ament, Sebastien Bigorre, Patrick Chazette, Susanne Crewell, Julien Delanoë, et al. 2017. "EUREC4A: A Field Campaign to Elucidate the Couplings Between Clouds, Convection and Circulation." *Surveys in Geophysics* 38 (November).

Caron, Mathilde, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. "Deep Clustering for Unsupervised Learning of Visual Features" (July).

Denby, Leif. 2020. "Discovering the Importance of Mesoscale Cloud Organization Through Unsupervised Classification." *Geophysical Research Letters* 47 (January).

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. "Imagenet: A large-scale hierarchical image database." In *2009 IEEE conference on computer vision and pattern recognition,* 248–255. Ieee.

Hoiem, Derek, Yodsawalai Chodpathumwan, and Qieyun Dai. 2012. "Diagnosing Error in Object Detectors" (October): 340–353.

Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. 2017. "Densely Connected Convolutional Networks." In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2261–2269.

Janssens, Martin, Jordi Vilà-Guerau de Arellano, Marten Scheffer, Coco Antonissen, A. Pier Siebesma, and Franziska Glassmeier. 2021. "Cloud Patterns in the Trades Have Four Interpretable Dimensions." E2020GL091001 2020GL091001, *Geophysical Research Letters* 48 (5): e2020GL091001.

Janssens, Martin, Jordi Vilà-Guerau De Arellano, Marten Scheffer, Coco Antonissen, A.P. Siebesma, and Franziska Glassmeier. 2021. "Cloud Patterns in the Trades Have Four Interpretable Dimensions" [in English]. *Geophysical Research Letters* 48, no. 5 (March).

Jean, Neal, Sherrie Wang, Anshul Samar, George Azzari, David Lobell, and Stefano Ermon. 2019. "Tile2Vec: Unsupervised Representation Learning for Spatially Distributed Data." *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (July): 3967–3974.

Jonker, Laura, Marije Elferink-Gemser, and Chris Visscher. 2010. "Differences in self-regulatory skills among talented athletes: The significance of competitive level and type of sport." *Journal of sports sciences* 28 (June): 901–8.

Lecun, Yann, Patrick Haffner, and Y. Bengio. 2000. "Object Recognition with Gradient-Based Learning" (August).

Mikolov, Tomas, Kai Chen, G.s Corrado, and Jeffrey Dean. 2013. "Efficient Estimation of Word Representations in Vector Space." *Proceedings of Workshop at ICLR* 2013 (January).

Rasp, Stephan, Hauke Schulz, Sandrine Bony, and Bjorn Stevens. 2020. "Combining Crowdsourcing and Deep Learning to Explore the Mesoscale Organization of Shallow Convection." *Bulletin of the American Meteorological Society* preprint (June).

Rs, Ramprasaath, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2020. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization.” *International Journal of Computer Vision* 128 (February).

Sermanet, Pierre, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann Lecun. 2013a. “OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.” *International Conference on Learning Representations (ICLR) (Banff)* (December).

———. 2013b. “OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.” *International Conference on Learning Representations (ICLR) (Banff)* (December).

Stevens, Bjorn, Sandrine Bony, Hélène Brogniez, Laureline Hentgen, Cathy Hohenegger, Christoph Kiemle, Tristan L’Ecuyer, et al. 2019. “Sugar, Gravel, Fish, and Flowers: Mesoscale cloud patterns in the Tradewinds.” *Quarterly Journal of the Royal Meteorological Society* 146 (September).

Tompkins, Adrian, and Addisu Semie. 2017. “Organization of tropical convection in low vertical wind shears: Role of updraft entrainment.” *Journal of Advances in Modeling Earth Systems* 9 (April).

Zeiler, Matthew D., and Rob Fergus. 2014. “Visualizing and Understanding Convolutional Networks.” In *Computer Vision – ECCV 2014,* edited by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, 818–833. Cham: Springer International Publishing.

Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. “Learning Deep Features for Discriminative Localization.” In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2921–2929.

Appendix

EUREC4A Data Set

As an extension of this internship work, we wanted to try processing and understand the data from the EUREC4A campaign (GOES data provided and then filtered to fit the campaign time period).

The data is initially in the form of NetCDF files, a set of software libraries, and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.

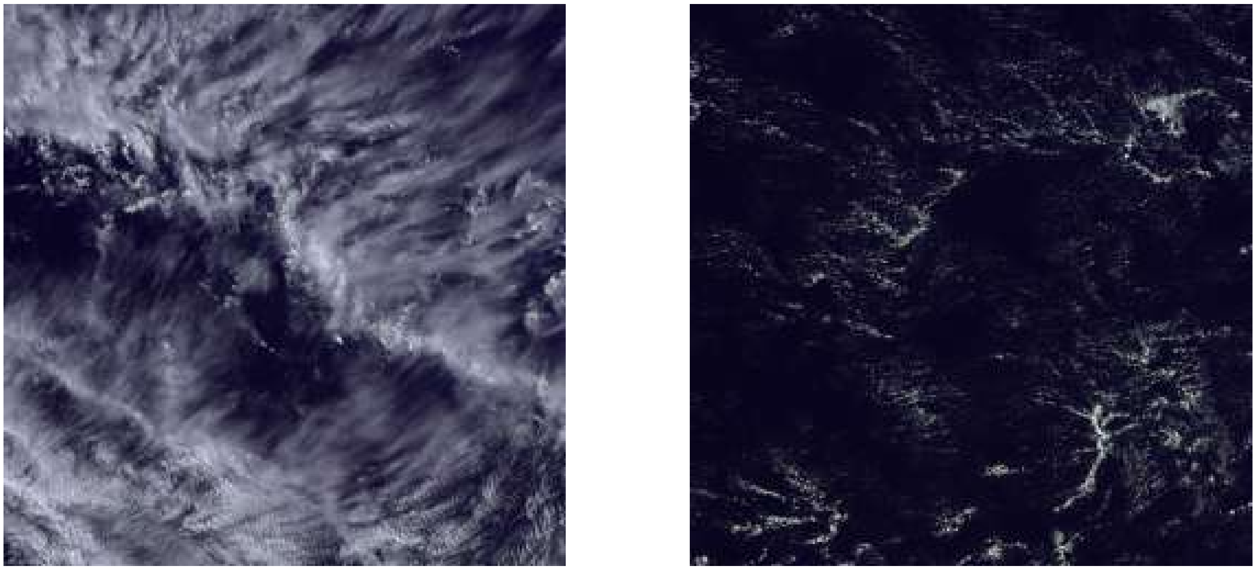
Some critical variables are the radiance levels. And the time analyzed :

- Date created: 2020-01-22T17:49:52.5Z
- Time coverage start: 2020-01-22T17:40:16.5Z
- Time coverage end: 2020-01-22T17:49:47.3Z

To read the files we used the library netCDF4. The processing algorithm is as follows :

- Reading .nc files.
- Combine 0.65μm, 0.86μm, 0.47μm to create RGB channels.
- Crop each image to multiples 256 by 256 pixels tiles.
- Delete tiles that contain black pixels.
- Delete tiles whose subset name (associated location) is not BARBADOS (campaign's location) .
- Delete tiles whose Hue level is between 80 and 100.

Some examples of images generated following this process are in figure 49.



(a) Example 1.

(b) Example 2.

Figure 49: Examples of images created.

Using this methodology of domain's images generation and after applying the original paper's model, we obtain clustering results shown in figure 50

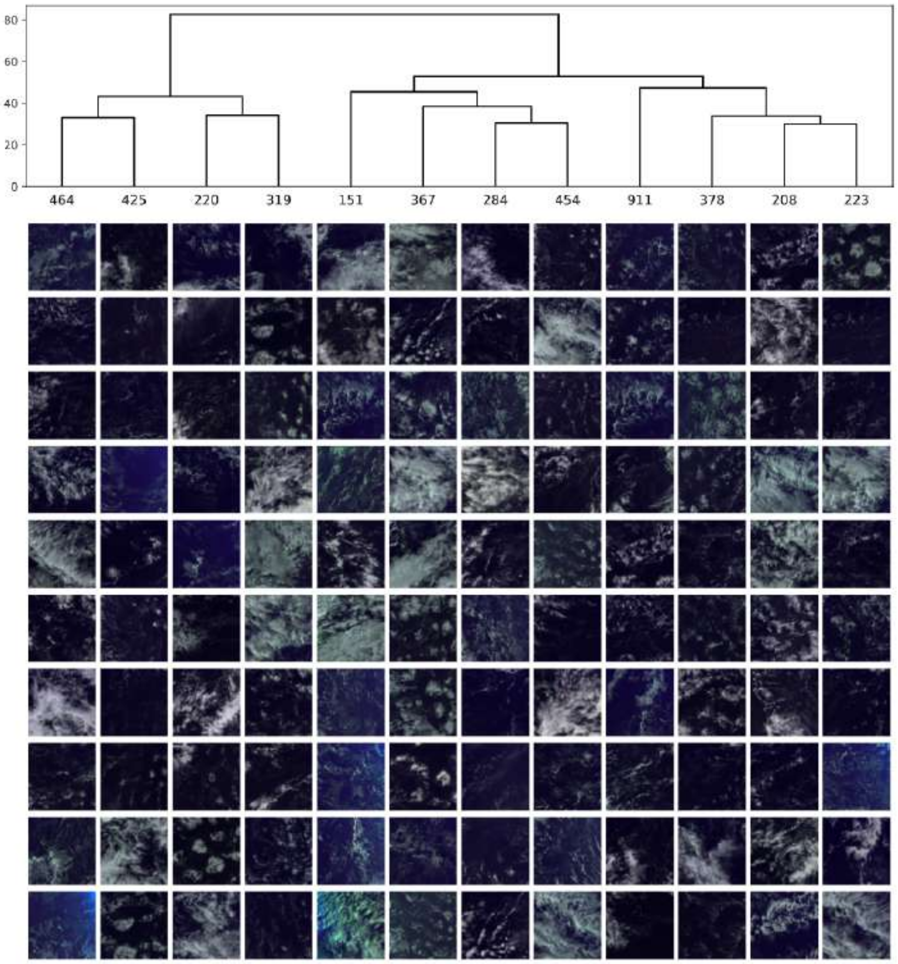


Figure 50: Clustering using original paper's model.

We can also see that there is no outstanding visual cluster. An explanation of this is that all images don't seem to give the same kind of coloration results (for example, some of them have way more green masks in them).

Amazon data set and algorithm for processing it

In Leif. 2020 paper, the data set used was from the amazon website. From there, we also found NetCDF files that contain different attributes from the EUREC4A ones.

From the same website, we can found a protocol to generate RGB images [10].

It consists in :

- Reading radiance.
- Convert them to reflectances.

[10] The OCC Environmental Data Commons: http://edc.occ-data.org/goes16/python/#reading-in-goes-16-netcdf

- Normalize them between (0, 1)
- Applying gamma correction.
- Compute true colors.

After the first four steps of the algorithm, we obtain a veggie image which looks as shown below :

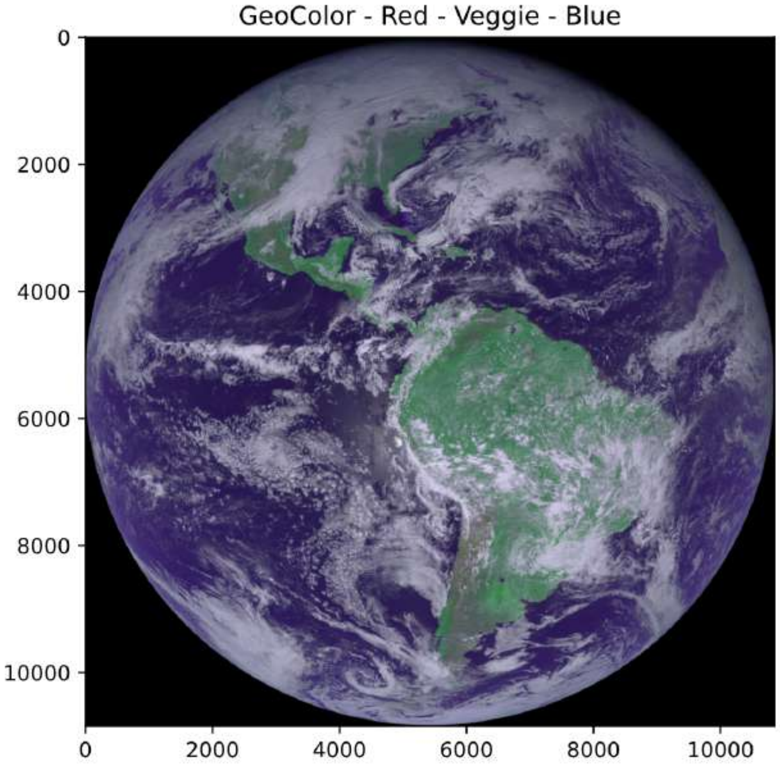


Figure 51: Image before true color computing.

True color operation :

$$Truegreen = 0.48358168 \times R + 0.45706946 \times B + 0.06038137 \times G \tag{9}$$

The same protocol on EUREC4A gives darker images. EURE4CA: netCDF file contains directly labels “reflectances,” but their range is between 0 and 100+, for applying the same protocol, i consider the variables in the file as radiance.

So we applied more steps :

- Gamma correction, with gamma = 0.25
- Contrast adjustment with contrast = 180
- We then get an image with way more brightness and contrast and close to what we need, it looks as in figure 52.

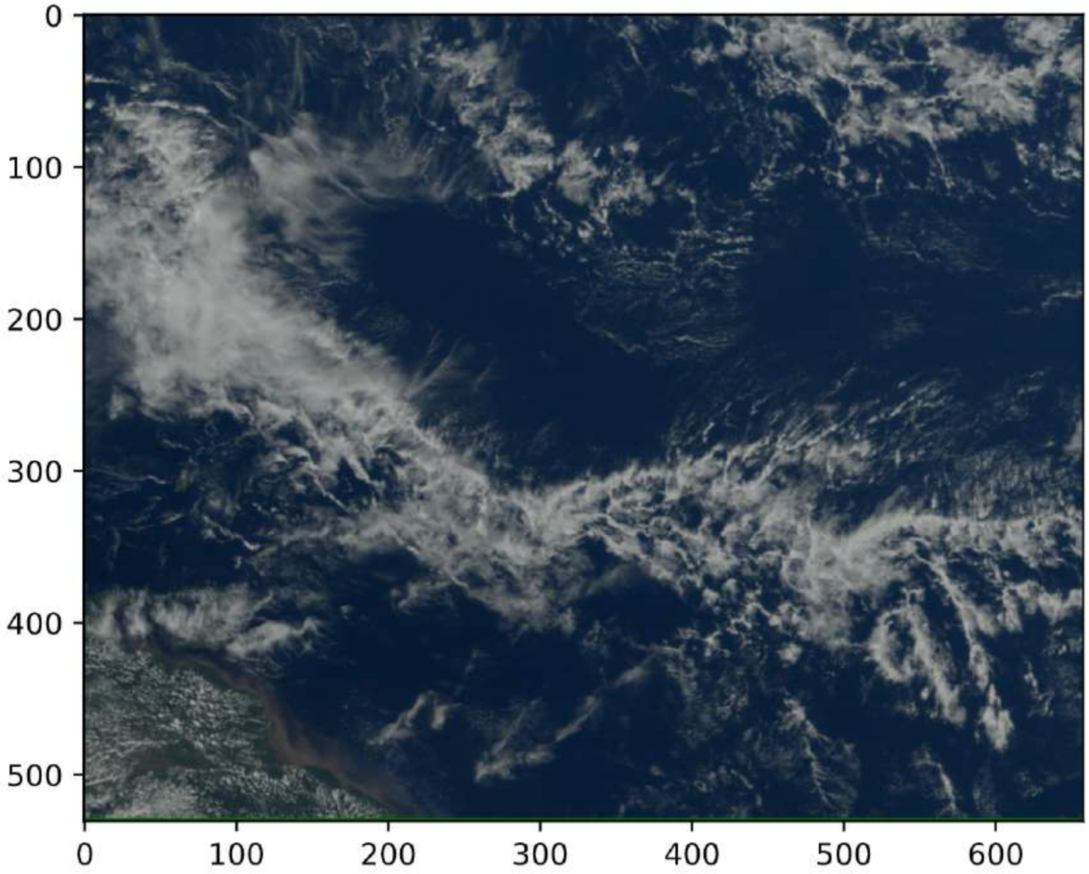


Figure 52: True color image.

We then use images generated from the entire EURE4CA Data set using the original model from Denby 2020 we divide each domain into 256x256 pixels tiles and take only photos between 12:55 and 13:10. We filter high clouds (by restricting our analysis to pictures for which the 25th percentile of T_b is higher than 285 K as mentioned in Stevens et al. 2019). Following are the clustering results using multiple different architectures trained on Amazon data set :

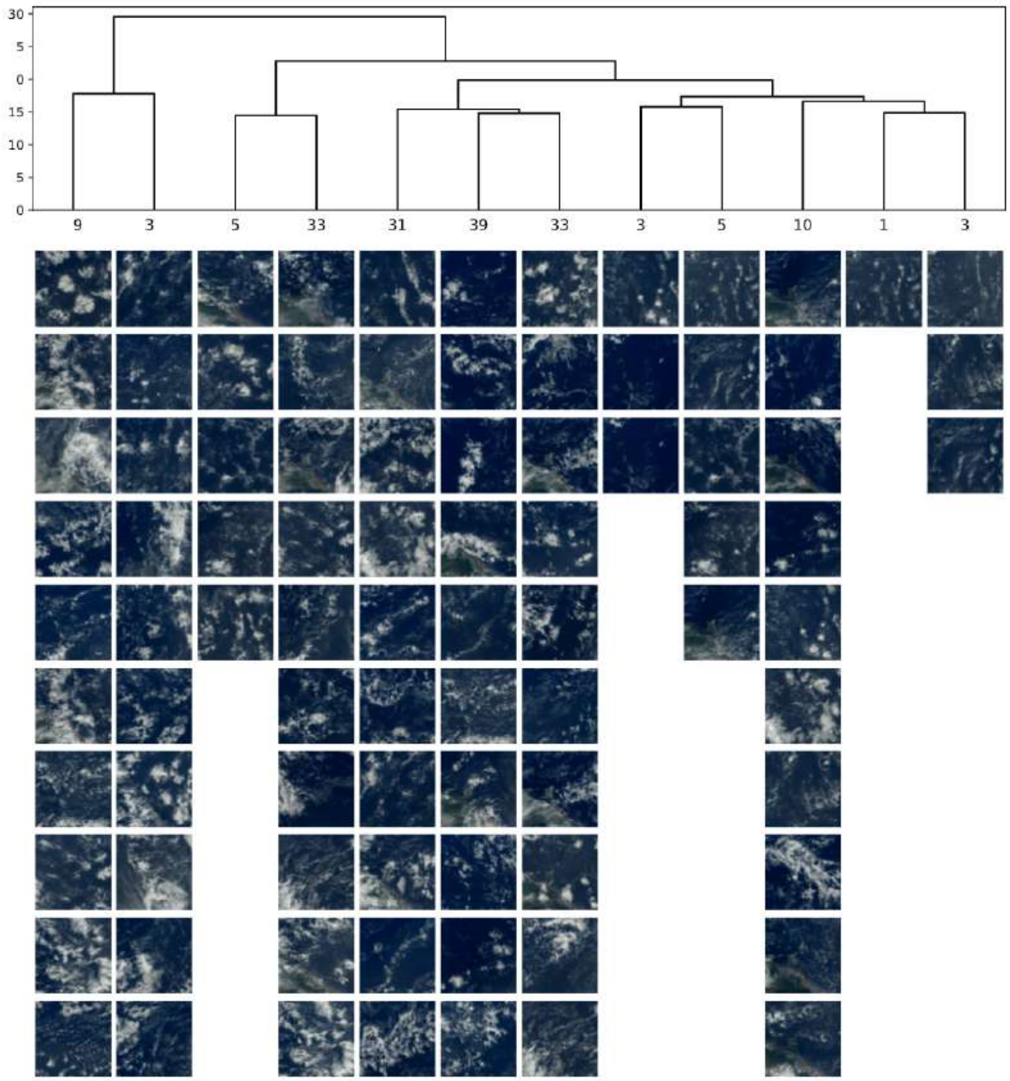


Figure 53: ResNet50 trained model.

From this process we understood the process to create real RGB images from netCDF files, but also create a pipeline from the original files to the embedding and clustering algorithms.

We then have chosen to focus on the Kaggle and ISSI data sets, which have labeled images, allowing us to investigate the realism of unsupervised methods to classify cloud organization.