CS577 Project Proposal

Understanding Clouds from Satellite Images

Jorge Gonzalez - A20474413

Luis Cavanillas - A20474430

Department of Computer Science

Illinois Institute of Technology

April 25, 2021

Abstract

In this project, an automated cloud detector has been created in order to improve the physical understanding of cloud formations, which in turn will help build better climate models. To do so, two different networks have been compared, with and without transfer learning, and a comparison between them has been established. As a two-member team project, the code was implemented together with a Colaboratory Notebook. To that purpose, both team members have carried on a prior analysis of the data to determine the useful information and then an appropriate pre-processing for the semantic segmentation approach.

1. Problem statement

Shallow clouds play a huge role in determining the Earth's climate, although they are difficult to understand and to represent in climate models. There are many ways in which clouds can organize, however, even though the human eye is really good at detecting their features, the boundaries between different forms of organization are not usually well defined. This makes it challenging to build traditional rule-based algorithms to separate cloud features.

Therefore, as it is stated in the paper *Combining crowdsourcing and deep learning to explore the mesoscale organization of shallow convection* [1], four subjective patterns of organization have been defined: Sugar, Flower, Fish and Gravel in a total of 10,000 satellite images on a crowd-sourcing platform and deep learning networks have been trained with this dataset to achieve an automated pattern detection and create global climatologies of the four formations, suggesting promising results.

Hence, by classifying different types of cloud organizations, researchers at Max Planck hope to improve the physical understanding of these types of clouds, which in turn will help build better climate models.

2. Background Information

The aim of the model is to classify different cloud formations in a satellite image. For that reason, the pattern recognition task can be framed with two different deep learning approaches: object detection and semantic segmentation [1], see Figure 2.1.

Object detection algorithms draw boxes around features of interest. Here, [1] proposes the model keras-retinanet, an implementation of a modern detection network in Keras. This is basically a ResNet model that matches the speed of one-stage detectors while surpassing the accuracy of all existing state-of-the-art two-stage detectors [2]. It uses a Resnet50 backbone. The original images had a resolution of 2100 by 1400 pixels, hence the images have been downscaled to 1050 by 700 pixels to fit the batch (batch size = 4) into GPU RAM.

In contrast, semantic segmentation algorithms classify every pixel of the image, assigning a category to every pixel from the original image that form the features of interest. For that purpose, a Unet structure [3] with a ResNet50 backbone that uses the fastai Python library v17 is used [1]. A mask will be created by converting the boxes of the cloud formations images to an array with their corresponding categories. In case the bounding boxes are overlapped, the mask will be chosen to represent the value of the smaller box. At the same time, the images will be downscaled to 700 by 466 pixels (batch size = 6).

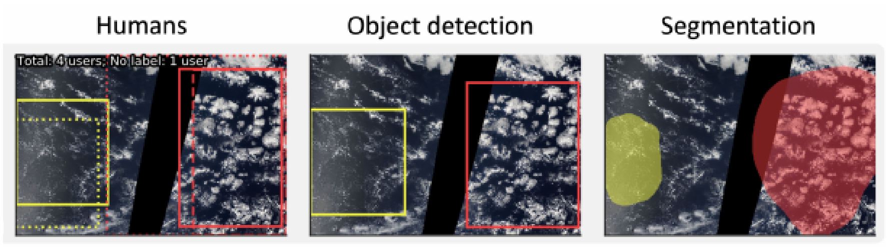


Figure 2.1: Different approaches to the detection of cloud formations.

3. Proposed Solution

A total of 5546 satellite images will be used as part of the dataset of the network, and they will be downloaded from the NASA Worldview website [4]. Labels will be retrieved from [5]. These images contain different cloud formations, with label names: Fish, Flower, Gravel, and Sugar. Each image has at least one cloud formation, and can possibly contain up to all four. The labels have been created in a crowd-sourcing activity at the Max-Planck-Institute for Meteorology in Hamburg, Germany, and the Laboratoire de météorologie dynamique in Paris, France. A team of 68 scientists identified areas of cloud patterns in each image, and each image was labeled by approximately 3 different scientists.

Three regions, spanning 21 degrees longitude and 14 degrees latitude, will be chosen. 4991 (90%) will be used for training purposes (*train_images*) and 555 (10%) will be used for validation of the results. A CSV file (*train.csv*) will also be used for each image-label pair in the training dataset, and will contain their run length encoded segmentations.

The proposed approach to determine the cloud formations in each image will be semantic segmentation, which consists of a form of pixel-level prediction that clusters the parts of an image which belong to the same object class together. Therefore, to be able to train a segmentation model, its outputs will be also an image with each specific class highlighted with a different value (‘mask’).

Initially a solution consisting in a single mask overlaying all the clouds in each image was designed. However, as it is described in Section 5, the poor results led to further research on segmentation models and the development of a mask with a depth of 4 channels, one for each type of cloud.

The models implemented have been the following:

a) Original U-Net from scratch: As described in the original paper, the network architecture consists of a contracting path and an expansive path. The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions, each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling, Figure 3.1. At each downsampling step the number of feature channels is doubled.

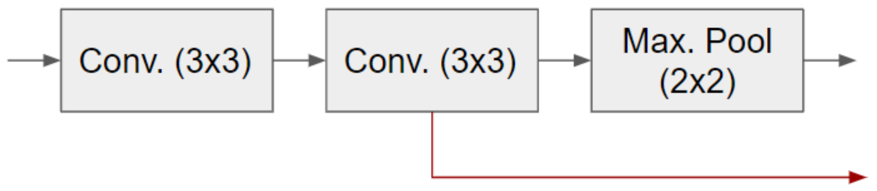


Figure 3.1: Contracting block.

On the other hand, every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU, Figure 3.2.

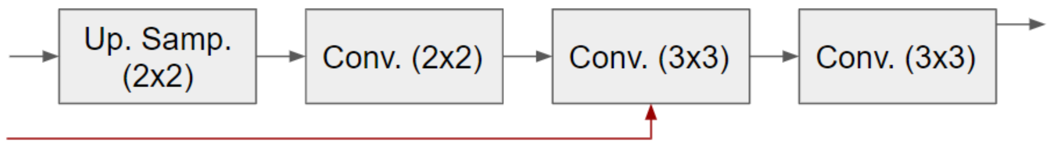
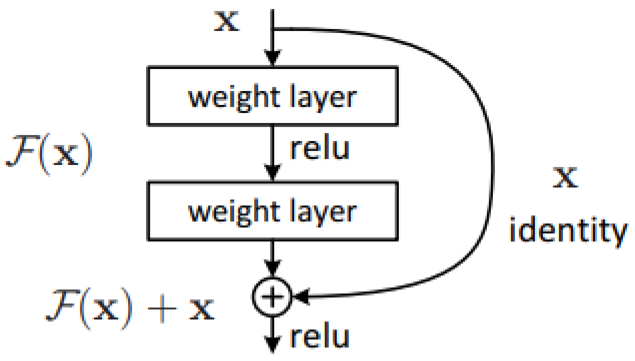


Figure 3.2: Expanding block.

At the final layer a 1x1 convolution is used to map each feature vector to the desired number of classes. In total the network has 23 convolutional layers.

b) Pretrained U-Net with a Resnet50 backbone: It consists of a U-Net type model with a ResNet50 network structure for the contracting and expanding paths. The ResNet50 [6] is formed by 16 residual blocks, each of them with the structure in Figure 3.2.



x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3.3: Residual block.

At the same time, just for this model, the initial weights will be initialized with the weights obtained from training the ResNet over the imagenet dataset to reduce the training time and take advantage of transfer learning.

4. Implementation Details

Before training the models, an initial analysis of the data has been carried out to understand the data before training is one of the most important aspects of every deep learning solution. Then, an appropriate pre-processing of the data has been done to generate the masks of the input images and the network models have been trained.

As it can be seen in Figure 4.1, there are two and three types of cloud formations in a single image most of the time and it is really rare for four types of cloud formations in a single image. At the same time, Figure 4.2, the data looks very evenly distributed for all four types of cloud formation. This is a good indicative of a dataset with a balanced class distribution.

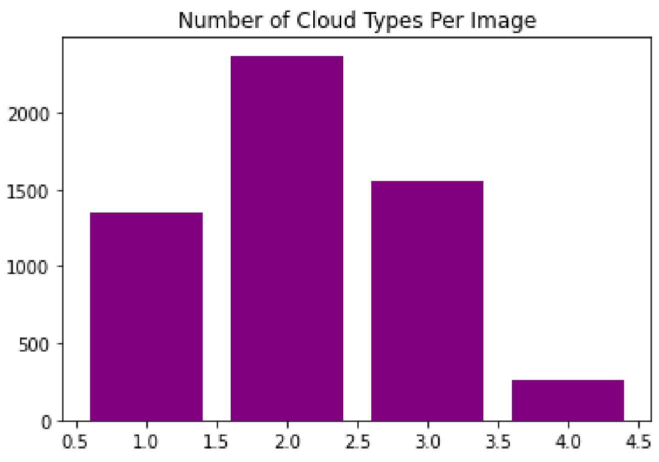


Figure 4.1: Number of Cloud Types per image..

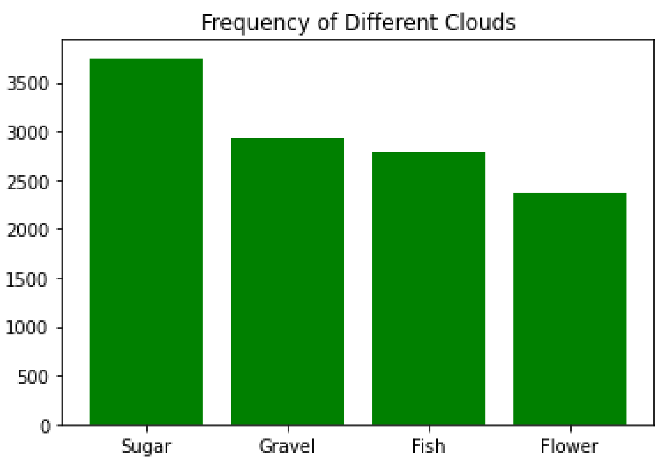


Figure 4.2. Frequency of different clouds

Furthermore, as it is shown in Figure 4.3, all combinations of cloud formations appearing together is a possibility, and the combinations between Sugar, Fish, and Gravel are more likely than with Flower cloud formation.

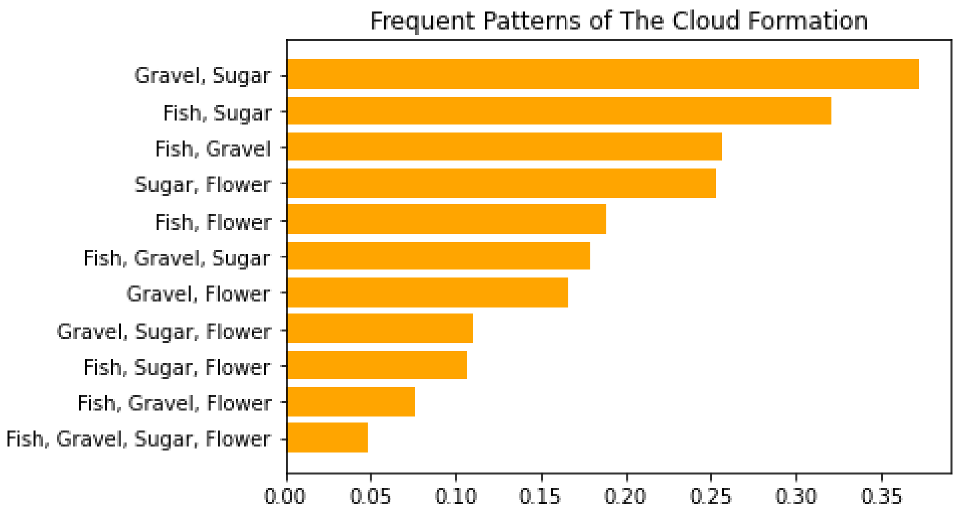


Figure 4.3. Frequent patterns of the cloud formations.

Hablar de dice loss y toda la vaina

Then, after having an general idea of the data that is going to be used, a preprocessing will be carried out. This preprocessing will consist of a resizing of the images to 256 x 256 and the generation of the masks. As it was mentioned in Section 3, two different masks are going to be created an tested:

1. SIngle Mask with all types of cloud formations overlayed: the value of the pixels of the mask will be 0 if there is not any type of cloud formation or either 1, 2, 3 or 4 depending on the type of cloud. These values will be normalized to the range [0,1].
2. Masks consisting of 4 channels or '4 masks': Each channel will identify a different cloud formation and the values of their pixels will be either 0 if there is not that specific type of cloud formation or 1 if there is that type of cloud.

Finally, the images will be introduced to the model with image generators and yield functions to reduce the RAM usage and the results of the hyper-parameter tuning have been the following:

1. Optimizer: Adam.
2. Output function: Sigmoid.
3. Activation functions (hidden layers): ReLu
4. Batch size: 16
5. Loss: binary cross entropy + dice coefficient

The dice loss is a popular loss function for segmentation models that is based on the Dice coefficient, which is essentially a measure of overlap between two samples. This measure ranges from 0 to 1 where a Dice coefficient of 1 denotes perfect and complete overlap. It can be calculated as follows:

$$Dice\ Loss = \frac{2|A \cap B|}{|A|+|B|}$$

where |A∩B| represents the common elements between sets A and B (the sum of the element-wise multiplication between the prediction and target mask), |A| represents the number of elements in set A and |B| represents the number of elements in set B.

Finally, the number of epochs for which the model has been trained are 100 epochs for the models trained from scratch and 20 epochs for the pretrained model.

5. Results and discussion

During this section, the cloud formation *e6388bb* (from the validation dataset) will be used as an example for the sake of simplicity. This input image is shown in Figure 5.1.

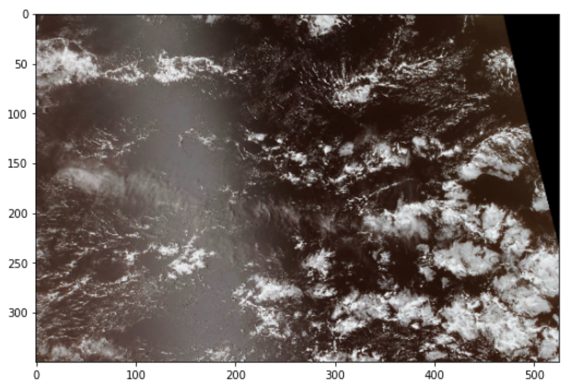


Figure 5.1.Cloud formation image *e6388bb* from the validation dataset.

5.1. Model 1: Custom U-Net & 1 mask

The first model is based on a single mask approach. Therefore, the four different types of cloud formations will appear on a single mask as shown in Figure 5.1.1. The validation image *e6388bb* will be used as example, where two *Sugar* clouds (yellow), one *Flower* (green) and one *Fish* (blue) are present.

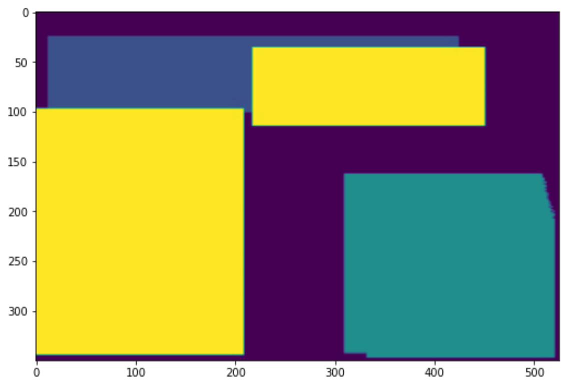


Figure 5.1.1.Generated mask for the validation image *e6388bb.*

For obvious reasons, accuracy will not be considered as the Semantic segmentation approach takes all the pixels present in the image for the later mask creation (binary map). Metrics used will be the BCE-Dice loss as the dice coefficient represents a measure of overlap between images.

After training for 100 epochs, the results are not as expected and the obtained BCE-Dice loss is 1.2450. The output for the validation image *e6388bb* is shown in Figure 5.1.2. Here, the predicted mask does not show clearly which cloud formations are present in the input image nor their correct position.

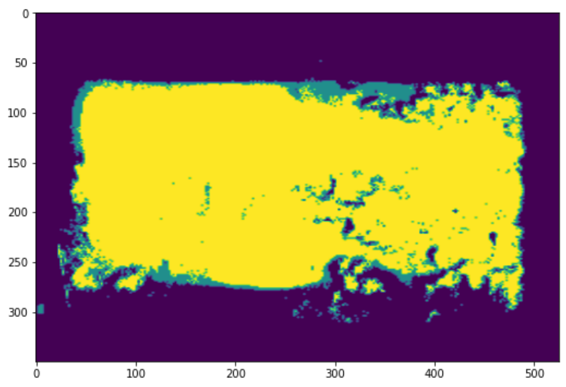


Figure 5.1.2. Model 1 predicted mask for the validation image *e6388bb.*

The reason behind these results can be deducted: little training has been performed due to the lack of resources and the mask is too complex for the network to learn. As shown in Figure 5.1.3, the training loss decreases over time, but at a very slow rate and in very small steps.

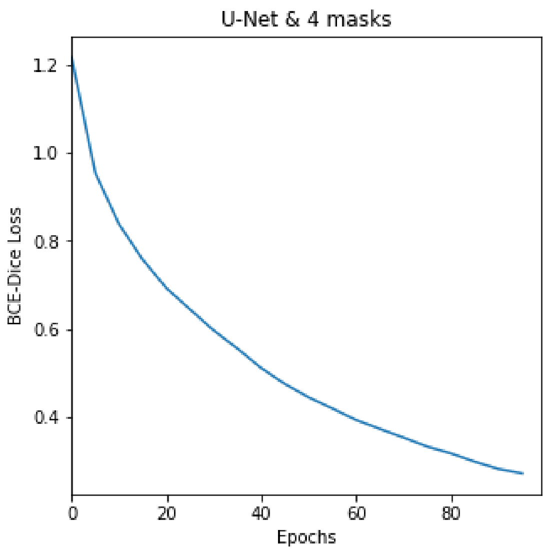


Figure 5.1.3. BCE-Dice loss during 100 epochs in Model 1.

A different approach will be taken on Model 2, where multiple masks will be used for cloud formations individually, instead of all in one. This is done because overlapping between classes occurs inevitably.

5.2. Model 2: Custom U-Net & 4 masks

For this model, the U-Net structure will remain untouched and untrained. One of the goals of this implementation is to confirm whether the new mask approach is correct and, based on that, further modifications will be carried out.

The four different masks from the validation image *e6388bb* are shown in Figures 5.2.1, 5.2.2, 5.2.3, and 5.2.4.

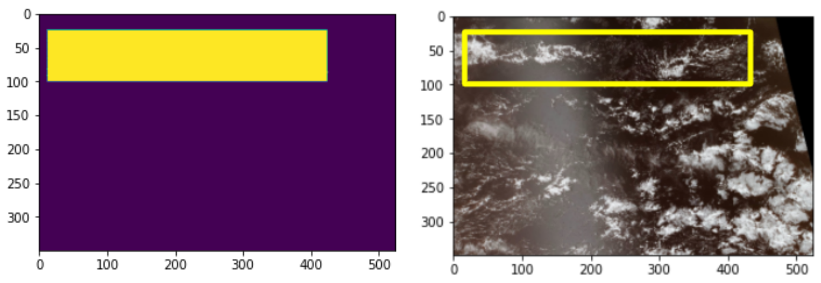


Figure 5.2.1. (A) On the left, generated *Fish* mask for the validation image *e6388bb.* (B) On the right, visualization of the mask on the cloud image.

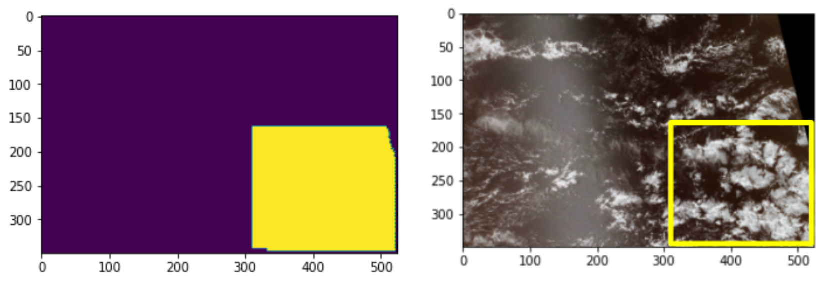


Figure 5.2.2. (A) On the left, generated *Flower* mask for the validation image *e6388bb.* (B) On the right, visualization of the mask on the cloud image.

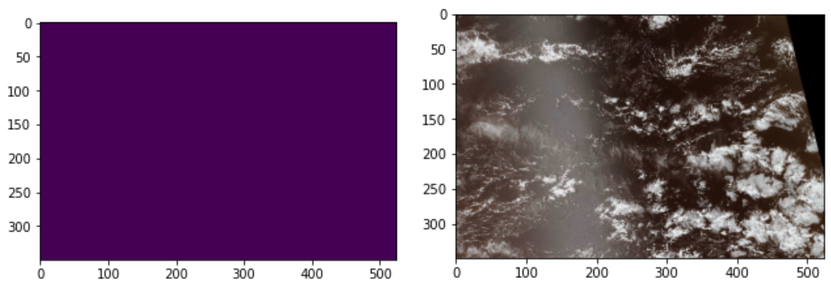


Figure 5.2.3. (A) On the left, generated *Gravel* mask for the validation image *e6388bb.* (B) On the right, visualization of the mask on the cloud image.

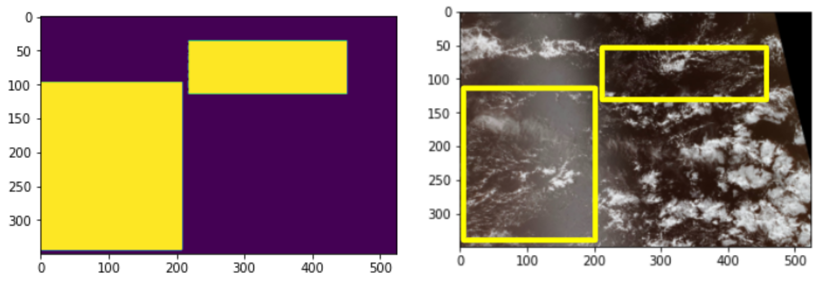


Figure 5.2.4. (A) On the left, generated *Sugar* mask for the validation image *e6388bb.* (B) On the right, visualization of the mask on the cloud image.

After training for 100 epochs, the results are still very poor. The obtained BCE-Dice loss is 1.2595 and the output for the validation image *e6388bb* is shown in Figure 5.2.5. As with the previous result, the predicted masks do not show the cloud formation position. However, in this case, masks in yellow indicate whether the cloud formation is present in the input image or not.

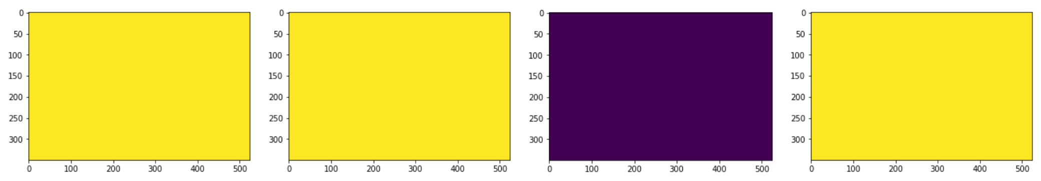


Figure 5.2.5. Model 2 predicted masks for the validation image *e6388bb*. Cloud types from left to right: *Fish*, *Flower*, *Gravel*, and *Sugar*.

One clear indicator for the great loss occurring in this model is the lack of time and resources to further train the model. The BCE-Dice loss decreases over each epoch, but at a very slow rate (see Figure 5.2.6). Under the current conditions, it would take weeks to achieve the desired outputs.

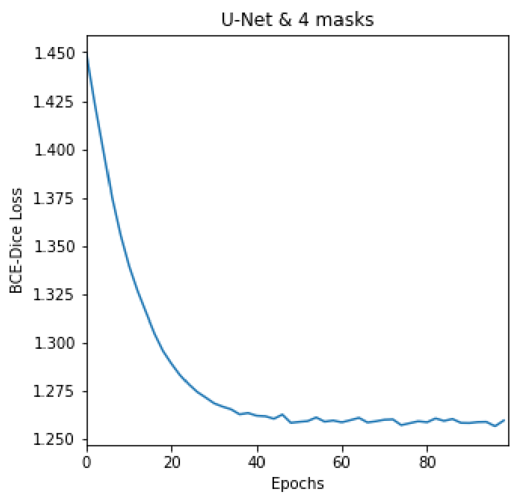


Figure 5.2.6. BCE-Dice loss during 100 epochs in Model 2.

These results are obviously not valid and have yet to be improved. Thus, the final and third model of this study will be implemented to fix this problem.

5.3. Model 3: Pre-Trained U-Net & 4 masks

Due to the fact that resources available are the main problem for these models, *Transfer Learning* has been applied. The aforementioned U-Net CNN with ResNet50 backbone (pre-trained) will be implemented.

Finally, after 20 epochs, the results are as expected and the output for the validation image *e6388bb* is shown in Figure 5.3.1.

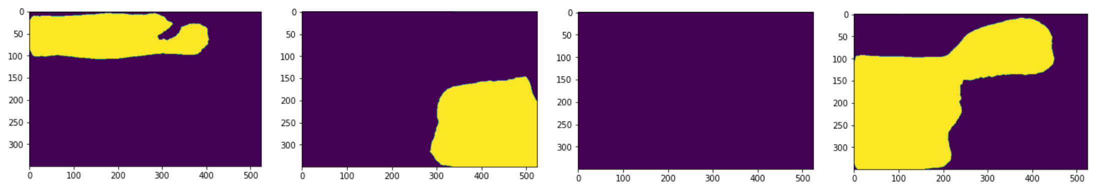


Figure 5.3.1. Model 3 predicted masks for the validation image *e6388bb*. Cloud types from left to right: *Fish*, *Flower*, *Gravel*, and *Sugar*.

The four different cloud formations are clearly present in each mask and their position is almost identical to the originally generated masks (Figures 5.2.1, 5.2.2, 5.2.3, 5.2.4). The obtained BCE-Dice loss has reduced to 0.2734, a 460% from the previous model (see Figure 5.3.2).

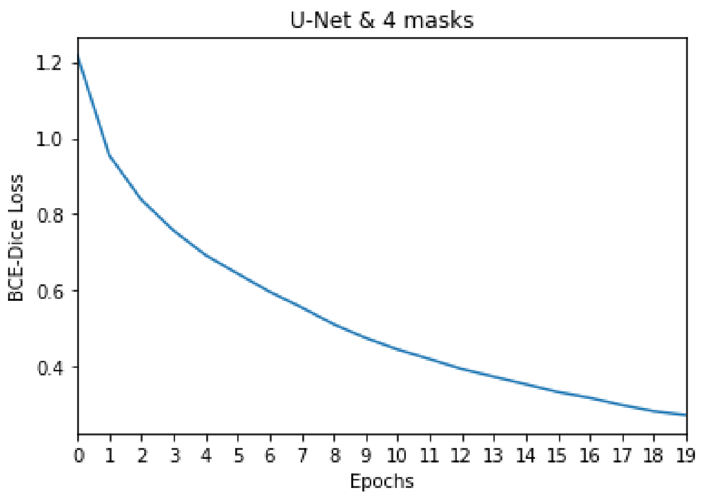


Figure 5.3.2. BCE-Dice loss during 20 epochs in Model 3.

6. References

[1] Rasp, S., Schulz, H., Bony, S., &amp; Stevens, B. (2020). Combining crowdsourcing and deep learning to explore the mesoscale organization of shallow convection. Bulletin of the American Meteorological Society, 101(11). doi:10.1175/bams-d-19-0324.1

[2] Lin, T., Goyal, P., Girshick, R., He, K., &amp; Dollar, P. (2017). Focal loss for dense object detection. 2017 IEEE International Conference on Computer Vision (ICCV). doi:10.1109/iccv.2017.324

[3] Ronneberger, O., Fischer, P., &amp; Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. Lecture Notes in Computer Science, 234-241. doi:10.1007/978-3-319-24574-4_28

[4] NASA Worldview Images. (n.d.). Retrieved from https://worldview.earthdata.nasa.gov/

[5] Zooniverse. (n.d.). Retrieved March 31, 2021, Retrieved from https://www.zooniverse.org/projects/raspstephan/sugar-flower-fish-or-gravel/about/research

[6] He, K., Zhang, X., Ren, S., &amp; Sun, J. (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2016.90

[7] Raspstephan. (n.d.). Raspstephan/sugar-flower-fish-or-gravel. Retrieved from https://github.com/raspstephan/sugar-flower-fish-or-gravel