Copernicus Masters in Digital Earth

Geodata Science Workshop

On Cloud N: Cloud Cover Detection Challenge

January–February 2022

Participants:

Team Gradient Descendants

Adebayo Adebowale Daniel

Aybar Camacho Cesar

Balogun Rufai

Barbosa-Ferreira Vitória

Donike Simon

Sharma Pratichhya

Singh Tanya

Zafar Hira

Table of Contents

1. Overview

The Goal of the Cloud cover detection challenge was to label clouds in satellite imagery. Clouds obscure important ground-level features in satellite images, complicating their use in downstream applications. Cloud detection consists of labeling each pixel of a scene by a binary variable indicating whether this pixel corresponds to a cloud or not. The labeling map is referred to as the cloud mask.

The dataset consists of Sentinel-2 satellite imagery stored as GeoTiffs. There were almost 12,000 chips in the training data collected between 2018 and 2020. Each chip is the imagery of a specific area captured at a specific point in time.

The main idea of the cloud detection method investigated for this project is to use a convolutional network operating on an input window to produce a binary segmentation map with label probability for each segment.

2. Literature Review

This section of the report presents an overview of semantic segmentation and classical cloud segmentation techniques encountered during the task. An understanding of different deep learning architectures for semantic segmentation of clouds.

2.1. Semantic segmentation

The CNN network can be trained to automatically segment image pixels into the same object classes in semantic segmentation. Also, it is considered one of the most popular tasks in computer vision and remote sensing applications for scene understanding [1]. Compared to general computer-vision based semantic segmentation tasks, the challenges in remote sensing semantic image segmentation of high-resolution sub-meter satellite images are to produce more refined predictions for every pixel in the large-scale image [2]. Convolution Neural Network (CNN)-based models have demonstrated excellent capabilities, making it a state-of-the-art method in remote sensing for semantic segmentation or classification of dwellings using satellite images.

The recent emergence of deep learning techniques has highly promoted semantic segmentation implementation in further research. Compared to the traditional algorithm for segmentation, deep learning-based methods have shown remarkable improvement in the model's performance [3]. The first review on semantic segmentation using the deep learning approach was presented by Garcia-Garcia et al. [4]. The authors provided a broad and organized review of the effective methods that used deep learning for semantic segmentation and highlighted the performance based on different evaluation metrics. Later, Wu et al. also presented a review of the segmentation approach and summarized their strengths, weaknesses, and major challenges in the paper [5]. Implementation of FCN for semantic segmentation, initially proposed by Zhang et al. in [6], paved the way for deep learning-based semantic image segmentation.

Since knowledge of cloud coverage is a must for efficiently handling remote sensing imagery, several research works have been done for cloud detection to generate binary masks of the predicted foreground (cloud) and background regions. Some commonly used methods include the band grouping/thresholding methods and the semantic segmentation based methods [7]. In this project, we proposed to use multiple state-of-art models for the semantic segmentation of clouds.

Modified Cloud-Net+ was proposed by Mohajerani & Saeedi via Filtered Jaccard Loss Function for and Parametric Augmentation [8]. But in the comparative analysis done by Yim et al. using methods based on semantic segmentation for cloud detection in remote sensing imagery, they presented that DeepLabV3 performed significantly better compared to Cloud-Net+ [9]. Also, research was done to solve cloud and cloud shadow segmentation with a global attention fusion residual network by Xia et al. [10]. Their purpose was to use ResNet as a backbone to extract semantic information at different feature levels to deal with boundaries using the atrous spatial pyramid pooling method. Based on the research done in this paper, the use of DeepLabV3 and PSPNet was carried on in this project. Further, Jiao et al. used UNet-based refinement networks for cloud and shadow precise segmentation [11]. They used a UNet with a conditional random field (Dense CRF) applied as a post-processing step to detect clouds precisely. Eventually, their experiment illustrated that a refined UNet model with dense CRF could provide a better solution.

2.2. Cloud Detection

Classical cloud detectors based on machine learning techniques need a set of handcrafted features computed for each pixel in order to predict if the pixel belongs to a cloud or not. The main factor of detection performance is the choice of the handcrafted features [12]. On the other hand, the choice of features (e.g. bands) is critical to achieving good classification performance [13].

When dealing with multispectral images, clouds and backgrounds exhibit different spectral characteristics. For instance, clouds generally scatter various wavelengths evenly, hence have high reflectivity in the visible and near-infrared bands. However, this spectral reflectance decreases slowly with increasing wavelength. Accordingly, the reflectance values of blue, green, red, and near-infrared bands are used to obtain spectral features of clouds [14]. This widespread use of the RGB, NIR and additionally SWIR band in the scientific literature informed our decision to use the band combinations as model features.

SWIR[15] was selected, in particular, due to the physical characteristics of cloud spatial structure, which is more visible as a spectral signature in shortwave irradiance through the physical mechanism of molecular scattering [16]. Consequently, adding shortwave-infrared bands can improve the accuracy of the semantic segmentation task of cloud and cloud shadow.

Cloud Detection Models

Numerous semantic segmentation models have been tested for domain-specific tasks of cloud segmentation. The most widely accepted cloud semantic segmentation models include DeepLab, UNet, and UNet++. DeepLab [17] is a state-of-the-art deep segmentation network for natural images, and it has been tested by [6] together with a different deep convolutional network, designed for cloud and snow detection tasks in satellite imagery[18]. On the other hand, UNet is a very effective image segmentation model with a remarkable performance in many image segmentation tasks, especially medical image segmentation tasks [19]. Many studies have found that models based on the U-Net architecture also show excellent performance in satellite remote sensing image segmentation. For instance, [20] proposes a cloud detection algorithm (RS-Net) based on the U-net architecture and detects clouds in satellite images. From the experimental results, it was found that the RS-Net performed better than the conventional method. [21] presented an improved U-Net convolutional neural network for the task of multi-sensor cloud and cloud shadow segmentation.

3. Methodology

3.1. Exploratory Data Analysis

Based on two well-known cloud detection datasets (biome-8 and cloud catalog), a couple of DL models were trained to filter noisy image patches. Our filter rule takes into account the accuracy and Jaccard index:

$$bad\_image\_patch = ((IoU\_M1 \mid IoU\_M2) < 0.3) \mid (acc\_M1 \mid acc\_M2) < 0.3$$

Additionally, we generate a benchmark test dataset using a block split.

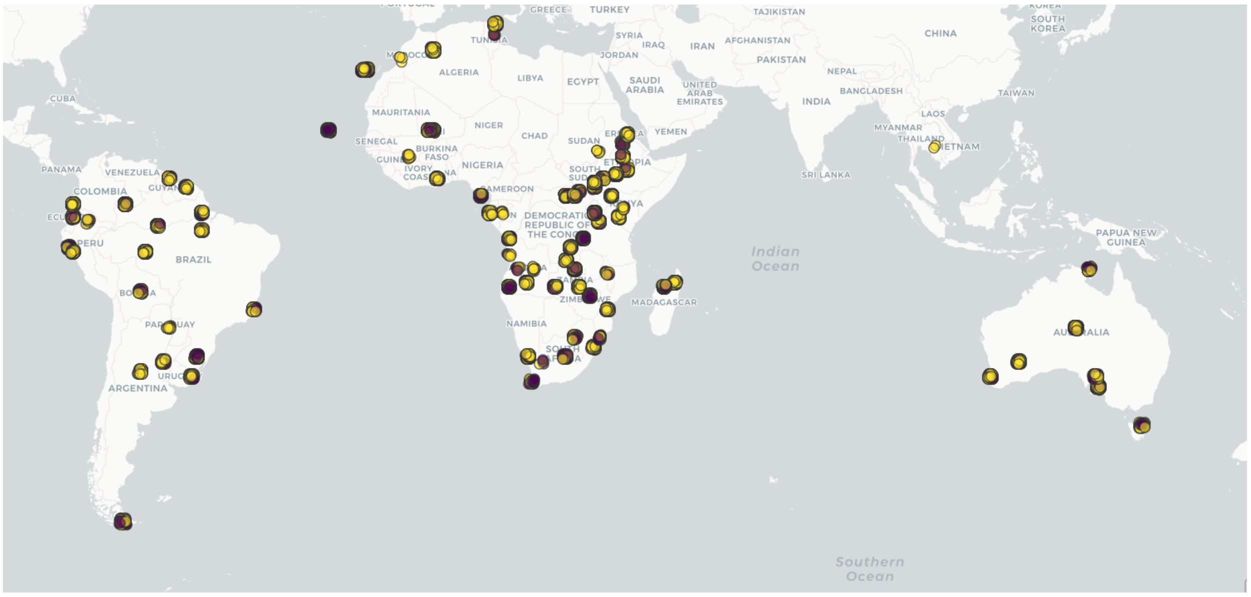


Figure 1: Spatial distribution of the training dataset

3.2. Experiments

Deep learning based cloud detection models were compared both visually and quantitatively. Models selected for experiments were UNet++ and DeepLabV3 because of their strong recommendation in the related literature. Details of the model experiments are given below:

3.2.1. SegNet

SegNet is a deep fully convolutional neural network for semantic segmentation tasks. It consists of an encoder, decoder and a final pixel-wise classification layer. Each encoder performs batch normalized convolution operations passed through a ReLU, followed by a 2X2 max-pooling layer with stride 2. With the max-pooling indices memorized, the 13 decoders up-sample the input feature maps. The output of the final decoder is sent into a multiclass soft-max classifier, which generates class probabilities for each pixel separately. [22]

The choice of SegNet was justified by previous experience with the network and proven good performance in earlier works of the team, especially considering boundary delineation. We used the SegNet implementation proposed by [23].

The code was implemented in colab, adapting the competition's benchmark. Due to data storage constraints in Google Drive, only 10% of the data were used. The achieved overall accuracy was 0.394, lower than all other networks tested within a Mobilenet_v2 encoder, Adam optimizer and a Binary Cross-Entropy Loss. The minimum overall accuracy achieved for other models was 0.43. After that, we had a meeting to discuss the best models based on the tests on the benchmark and the literature review. The final decision filtered out SegNet, since other models, such as Unet++ and DeepLabV3, have shown more promising results.

3.2.2. PSPNet

The Pyramid Scene Parsing Network (PSPNet)[24] is a well-adapted semantic segmentation model that was built on the architecture of a typical Fully Convolutional Network, with an encoder – which extracts the features from the image, and a decoder – which predicts the class of the pixel at the end. The PSPNet Encoder uses a CNN backbone with dilated convolutions which are passed through a pyramid pooling module. This architectural design allows the model to capture image objects at different scales and extract image contexts. Through the pyramid pooling module, the model extends its receptive field to learn global context information to capture local context features used for the scene segmentation. In the case of binary cloud segmentation, the PSPNet architecture was tested since it has achieved competitive performance like the DeepLabv3 and consequently fits into several model designs for Fusion [25] [26].

To speed up the experiment time, the PSPNet model was trained on 50% of the training data with an *efficient b0* backbone, Adam optimizer and a Binary Cross-Entropy Loss. This experiment was tested with and without the Early Stopping Callback. In the first case, the model yields a performance on the validation set of 86.31% accuracy and a corresponding 76.98 IoU. In the latter, the model accuracy increased to 91.70% and 83.87% IoU.

3.2.3. DeepLabV3

DeepLab is a semantic segmentation architecture that has used dilated convolution technique since its first version. It helps to get the larger field of view of the image while keeping the computation cost almost constant [17]. Rozenhaimer et al. have used combined cloud and cloud shadow detection using DeepLab that showed a performance of 90% correct rate values while 8% false rate values. However, the prediction was highly dependent on different types of clouds [27].

DeeplabV3 is a modified version of DeepLabV2 that no longer uses Dense CRF in its architecture. DeepLabv3 outperforms DeepLabv1 and DeepLabv2, even with the post-processing step Conditional Random Field (CRF) removed, which is originally used in DeepLabv1 and DeepLabv2 [17]. DeepLabV3 has Encoder-Decoder with atrous spatial pyramid pooling (ASPP) architecture [28]. In other words, the encoder will have atrous convolution with different rates that will provide encoded information; then, the decoder will take the output from atrous convolution, which will make the dimensional reduction, which is then concatenated with encoded data. Though ASPP was introduced in DeepLabV2, in DeepLabV3 batch normalization from inception-v2 was used that showed improvement in its performance.

Advantages of using DeepLabV3:

- It has adjustable atrous convolution, with which we can adapt to modify the filter's field of view. We can control the resolution at which feature responses are computed within a neural network using atrous convolution without having to learn additional parameters [29].

Disadvantages of using DeepLabV3:

- Computation cost is relatively high.
- Though it has interesting architecture, it was complex to understand.

In order to maintain consistency in parameters, DeepLabV3 was trained in a similar environment as the other 3 models except that once it was trained with MobileNetV2 as backbone later tested with EfficientNet-b0 and EfficientNet-b1. The highest overall accuracy of 0.92 was achieved with efficient net-b1 compared to the other two when using Binary Cross-Entropy loss. The value of overall accuracy for validation reached 0.96 and IoU of 0.93 when DeepLabV3 was trained using MobileNetv2 with IoU as loss function.

3.2.4. UNet++ model

UNet++'s [30] architecture uses the Dense block ideas from DenseNet to give better performance than U-Net. The architecture is basically a deeply-supervised encoder-decoder network where their sub-networks are connected through a series of nested, dense skip pathways. The re-designed skip pathways aim at reducing the semantic gap between the feature maps

Advantages of using UNet++

- The re-designed skip pathways bridges the semantic gap between encoder and decoder feature maps.
- The dense skip connections on skip pathways improve the gradient flow
- Deep supervision adopted from the dense net enables model pruning which improves performance.

For the final implementation of the UNet++ model, we adapted the Tversky Binary Cross-Entropy Loss to deal with the overestimation and underestimation of the cloud mask boundary. The Tversky loss is a loss function that is implemented when dealing with unbalanced datasets, as in our case – where the foreground(clouds) pixels cover a lesser area than the background. This kind of training set typically leads to predictions that are biased towards precision but low recall (sensitivity). The Tversky index penalizes the false negatives and false positives to achieve a better trade-off between precision and sensitivity. It is then added to the BCE loss to form the final loss function.

Before this newly adapted architecture was trained, the images were passed through a pre-processing workflow using both the CLAHE and sharpening transformations to make the clouds more distinct compared to the background as shown in figure 2. The Contrast Limited Adaptive Histogram Equalization (**CLAHE**) is a variant of the Adaptive Histogram Equalization (AHE) typically used in improving image contrast. This transformation operates on small regions in the image (tiles) and the neighboring tiles are combined using bilinear interpolation to remove artificial boundaries. The implementation of this algorithm was adapted from the ***albumentation*** transformation class using the clip limit of 3.5 and a probability of applying the transform to 1, since we want the transformation to be applied for all images in the training sets. Next, we sharpened the image and overlayed it on the original image. These transformed images were passed to the U-Net++ with the adapted loss function leading to an IoU of 0.81 in the inference mode.

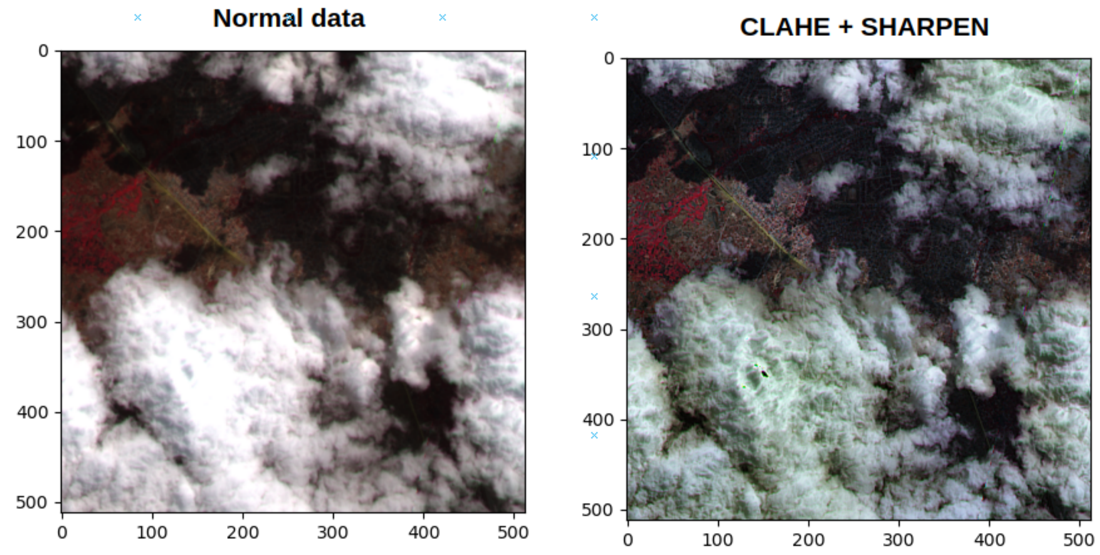


Figure 2: Outcome of the adapted pre-processing with CLAHE and SHARPEN algorithms

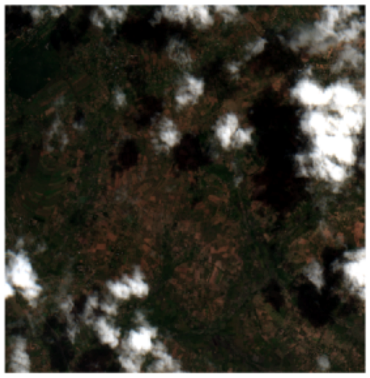
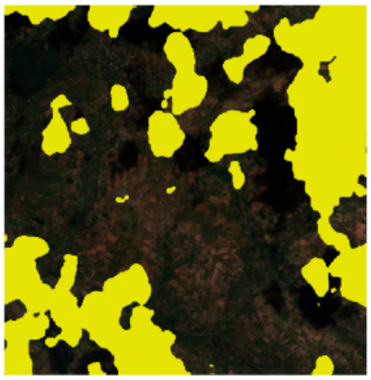
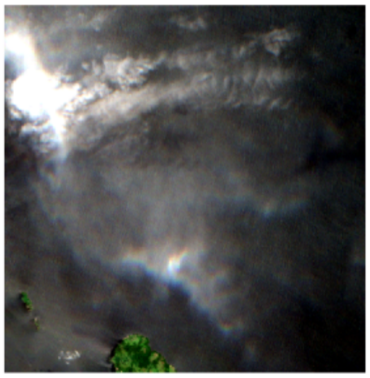
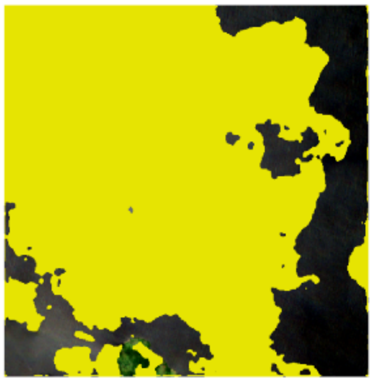
	INPUT	OUTPUT
AEJO		
AEGB		

Figure 3: Final prediction with improved boundary information

3.3. Models tested and corresponding performances

The table in this section summarizes the best performance of all the models tested during the workshop.

Model	**Backbone**	**Loss function**	**Validation Accuracy**	**Jaccard Index**	**Percentage of data used**
Baseline (U-Net)	ResNet34	Cross-Entropy Loss	–	0.8151	100%
SegNet	MobileNetv2	Dice Loss	0.394	–	10%
PSPNet	EfficientNet-b0	Binary Cross-Entropy Loss	0.91703	0.83869	50%
UNet++	MobileNetv2	Binary Cross-Entropy Loss	0.93335	0.867	50%
DeepLabV3	MobileNetv2	Binary Cross-Entropy Loss	0.927	0.862	50%
DeepLabV3	EfficientNet-b0	IoU Loss	0.929	0.862	50%

Table 1: Summary of best model experiments and corresponding performances (Validation accuracy and Jaccard Index). **Note:** All models used the Adam optimizer, a batch size of 16 and regularized with the L2 penalty, available through the PyTorch Weight Decay module.

3.4. Post-processing

Here, we appraised the challenges encountered with the output of binary segmentation models for remote sensing tasks. Fully Convolutional Networks like UNet++, DeepLabv3 and PSPNet that were tested in this exercise are typically associated with the production of blurry, noisy and overly regularized boundary objects masks that are less sharp, lack convexity and connectivity as expected [31]. This same challenge was encountered in the development of our model and the inferences the model produced as shown in Figure 1. These excessively smoothed boundaries are false positives that have been recognized as a typical challenge in the binary cloud segmentation research community, where certain techniques have been developed to tackle it. Our first approach was to test a couple of morphological operators to refine the prediction masks to get as close as possible to the ground truth using conditions for filtering out excessive small masks that might be other objects rather than clouds and enhancing the edge information in the final output.

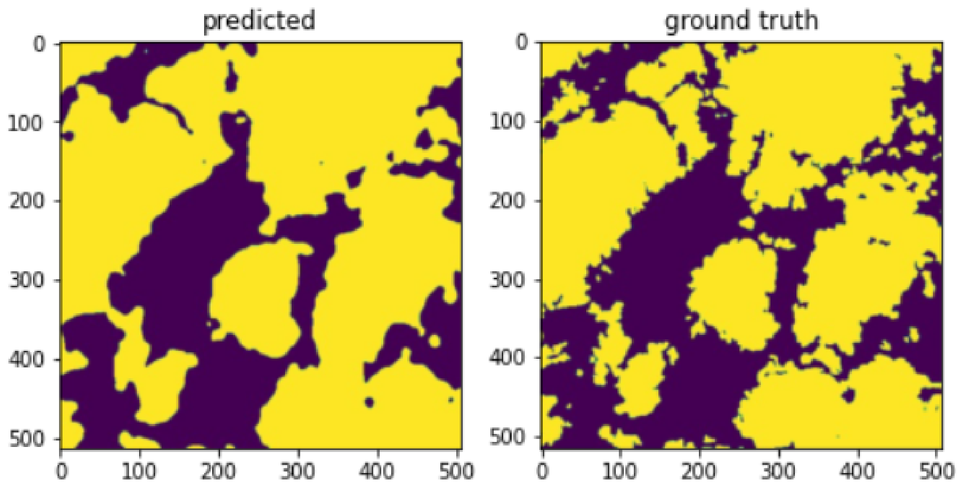


Figure 4: Predicted versus ground truth cloud masks

The morphological operators that were tested include: 1.) the removal of noisy holes in the predicted output using a combination of binary closing and binary opening by setting a threshold for the typical width of small and tiny cirrus clouds [32] [33]. 2.) dealing with the edge information by a combination of morphological opening, closing, erosion and dilation. Apart from these basic morphological operations, an attempt to distance transform function available in skimage[1] for addressing boundary detection issues. The next subsection provides further details on the outcome of these experiments and the corresponding inferences and observations.

3.4.1. Morphological Operators

Regardless of the model, the visual perception of the predicted models is quite different from the ground-truth labels. The edges of the cloudy areas seem a lot rougher in some of the images. Most likely depending on the annotator of the labels or the mood that he/she was in, the edges can be very detailed, ergo rough, or very smooth because the annotator did not take too much time to accurately depict the cloud borders. In general, though, the predictions seem smoother. In order to change these edges and potentially gain some accuracy improvement, a workflow[2] implementing Mathematical Morphology[3] was created.

Since it is impractical to visually investigate all labels to find the best method, the workflow was created to perform the prediction on only 200 samples and visualize those. A before-and-after ‘Intersection over Union’ calculation to be able to quantify the improvement or lack thereof is calculated as well. Several

[1] https://colab.research.google.com/drive/1AeXJCLyUXk6uEvucr_NUvx0YurhEXraI?usp=sharing
[2] https://github.com/gradient-descendant/compare_labels_pred/blob/main/morphology_.ipynb
[3] https://scikit-image.org/docs/stable/api/skimage.morphology.html

morphological operators with different structuring elements were examined in this workflow to see which ones perform best (See table 1). A total of 48 runs were completed for the different operators, structuring elements and sizes of the structuring elements. The best performances of a given size of the structural elements are visualized in table 1.

The performance was judged by subtracting the IoU of the normal predicted labels from the IoU of the manipulated labels; positive values, therefore, indicate an increase in accuracy while negative numbers show a decrease in accuracy.

Structuring Element / / / / / / / **Method**	Disc (3,5,10)	Ball (3,5,10)	Star(3,5,10)	Diamond(3,5,10)
Erosion	-0.02	-0.01	-0.02	-0.03
Dilation	0.03	0.01	-0.01	0.03
Opening	0.01	-0.02	0.00	-0.02
Closing	0.04	0.02	0.02	0.01

Table 2: Morphology operators and their structuring elements, the number displayed is the best result out of the listed SEs averaged over 200 predictions. Positive values: improvement; negative values: decrease. Accuracy difference in the same units as the accuracy (0<n<1).

Clearly, improvements are marginal at best. Neither operator has a clear advantage over the others and the differences are very slim. No clear trend that some operators are better than others could be established. In addition, since the results of the methods were only quantified for 200 samples, the results might not be accurately extrapolatable to the whole dataset.

When assessing the performance change for singular predictions, the differences are quite high. For some styles of labels improvements up to 0.45 were seen, while accuracy decreases of a similar magnitude could be observed for other predictions. This intrinsic variance shows that the performance of the morphology is highly dependent on the labeling style. Those different styles tend to average out over the dataset, leading to the small average values observed in table 1.

Because of this uncertainty, it was decided not to include any morphological operators in the submissions.

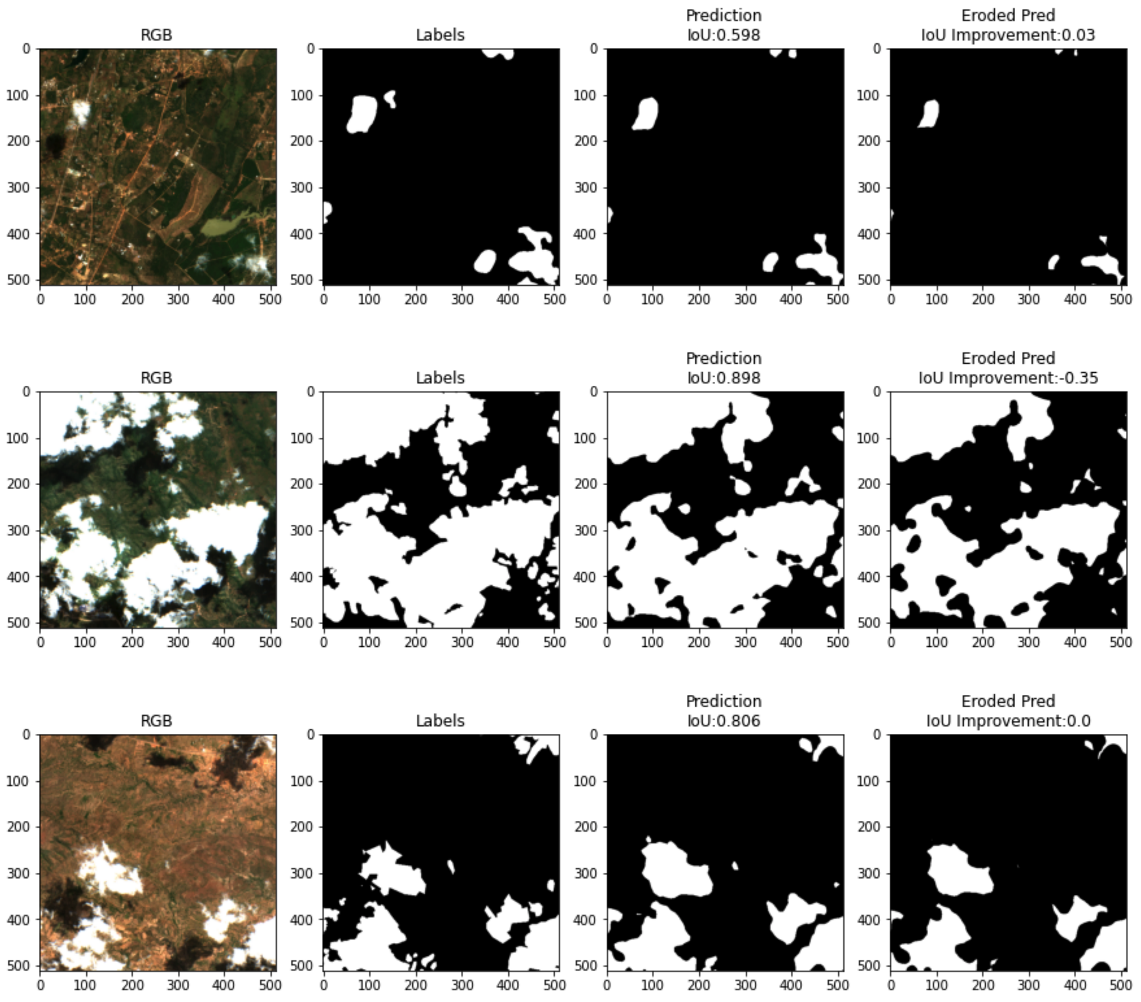


Figure 5: Visualization of some RGB input images, their ground truth labels, the predictions and the predictions after mathematical morphology, in this case, an erosion.

3.4.2. Boundary-Aware Segmentation

We then pivoted to test other techniques from within the model, leveraging the research outputs of previous researchers in the development of boundary-aware segmentation models. In this process, we tested two established techniques for dealing with blurry boundaries in the segmentation models: 1.) A Multi-task learning approach that regresses the distance transform of the high-level features from the background pixels [31], and 2.) Adaptation of the Tversky loss function with the binary cross-entropy loss function for improving precise cloud boundary predictions. Other methods considered but not tested included the Graphical method of Conditional Random Fields for merging model outputs.

I. Boundary-aware segmentation model using Distance Transform

Unlike other existing methods for improving the prediction of precise boundary information in segmentation models, Audebert et al, 2019 proposed the use of the distance transform regression. The distance transform regression informs the network, through a multi-task learning approach, about the proximity of objects and their corresponding shapes along edges and further provides cues about the spatial structure of the network. The adapted architecture of the boundary-aware model using distance transform is shown in Figure 3.

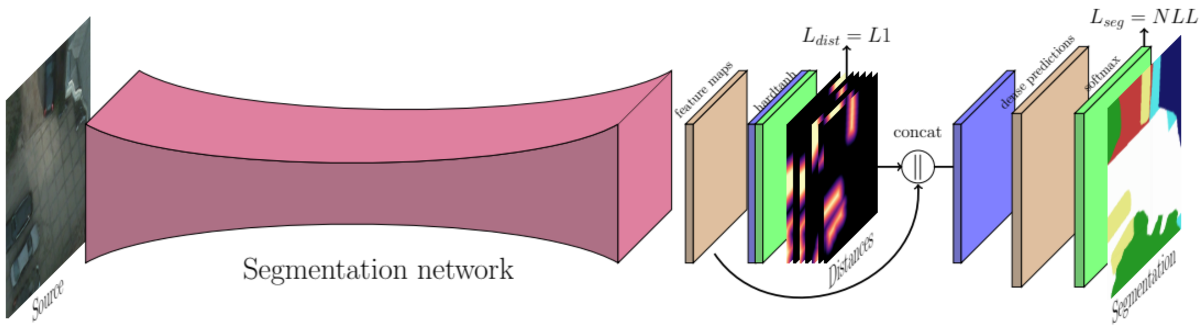


Figure 6: Adapted Segmentation Network with a Distance Transform Regression [31]

We adapted this architecture with the PSPNet model, and experimented with the data in the cluster. This resulted in surprisingly low model performance with a validation accuracy of 0.3995 and an "Intersection over Union" accuracy of 0.2127. Due to this low performance, we explored other alternative methods for improving the segmentation mask boundary.

II. Boundary-aware segmentation model using Tversky Binary Cross-Entropy Loss function

As already explained in the model experiments session 3.2.2, we adapted a new loss function adapting both the Tversky loss function and the binary cross-entropy loss function and pre-processing the data using the CLAHE and SHARPEN algorithms to deal with two main issues in the modeling process: 1.) the data imbalance between the cloudy pixels and the background, 2.) the inaccurate prediction of the cloud masks. This yielded a much better accurate mode performance, reaching 81% of the IoU metric. This improvement on the modeling process was finally adapted for the final submission since its inference time was lesser than others and was adapted using the UNet++ model, see section … for information on the model inference time.

3.4.3. Domain knowledge - Cloud detection using multi-temporal image

There are existing methods [34] [35] [36] for using multi-temporal/time-series images in cloud detection. The idea we proposed with respect to this contest is to pull from the sentinel2 image collections extra images for each chip(*t*) at *t-1* and *t+1* with the least scene percentage cloud cover, where t represents the image sensing time filtered from the metadata.

Comparing pixel-to-pixel differences in the multi-temporal images based on high reflection variation as suggested by [34] between the cloudy image(t) and the less cloudy image(t-1 & t+1), a “change mask(Ĺ)” is generated for each chip. Therefore, the change mask (Ĺ) is used in training the model with the provided label mask. A strength of this approach can be that it can make our model robust. The artificially generated masks won't have a very high confidence level for clouds which can put them into the category of noisy labels. Incorporating these noisy labels in training can make the model robust.

The shortcoming of this approach would be that the mask we will use for training will be very different from the masks used for testing; this can strongly negatively impact the accuracy.

3.5. Evaluation Metrics

The performance of each network is evaluated by the pixel difference between the generated cloud mask and its estimation referred to as pixel accuracy. Along with accuracy, our main criterion for the comparison of different approaches was a metric called the Jaccard index, also known as Generalized Intersection over Union (IoU). Jaccard index is a similarity measure between two label sets. In this case, it is defined as the size of the intersection divided by the size of the union of non-missing pixels. In this competition, there should be no missing data. Because it is an accuracy metric, **a higher value is better**.

Given the imperfection around boundaries of clouds in the operator cloud mask, our attention was focused on precision: the error on the boundaries are less important than the presence or absence of a cloud.

.

4. Submissions

There were a total of 14 model submissions (and 12 others canceled by the submitter). From which only one was accepted. The errors emerged mostly by missing packages in the organizers' environment (especially Omega Conf [37], which was used to facilitate the loading of the parameters in the training phase); the wrong assignment of directory paths; excessive number of workers; exceeding the running time limit; rasterio error, among others.

We changed the final submission so that Omegaconf was no longer required and that all directory mistakes were corrected. One submission went through 3 days before the deadline, a Unet++, trained using the Tversky loss and 40% of the data. The achieved accuracy was 0.8128, which was lower than the result in the benchmark best score (approximately 0.82).

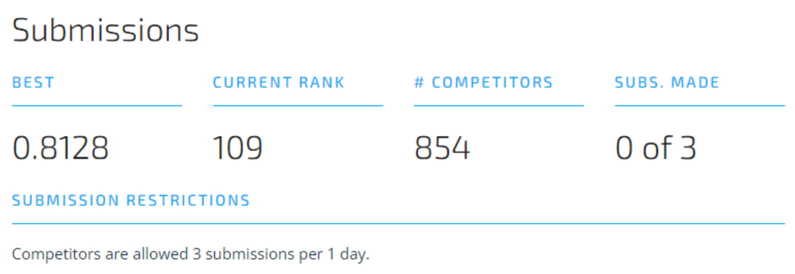
The inaccuracies in the number of workers happened because only 6 workers were available in the submission platform, and 12 were used in the cluster.

Few other submissions were done using DeepLabV3, since it achieved the best scores during our experiments[4]. However, it exceeded the allowed running time each time, leading to failed attempts. The model is evaluated on the organizer's side with around 10.000 images, which must run within 4 hours, plus around an extra hour to set up the environment. These limitations resulted in a backlog of over 30 contestants in the last days of the contest, resulting in a wait period of around 20 hours per submission (only one submission could enter the queue at each time).

Despite not surpassing the benchmark score, the submitted model ranked 109 out 854 competitors (Figure 7).

4

https://docs.google.com/spreadsheets/d/1e_vuTPy2IM44miqocFz1EUp7u990rXUohT4vMuJAyt8/edit?usp=sharing


Submissions

BEST	CURRENT RANK	# COMPETITORS	SUBS. MADE
0.8128	109	854	0 of 3

SUBMISSION RESTRICTIONS

Competitors are allowed 3 submissions per 1 day.

Figure 7: Final submission outcome, ranked 109 out of 854 competitors.

5. After submission

Towards the end of the competition and as exceeding the running time continues to prevent submissions, we ran a few tests on the cluster, noting the difference in running time-based on batch size. Our test data included 1784 chips and 6 workers, which was the maximum number of workers recommended by the competition organizers. Although during the competition, our test data consisted of only 289 chips, we decided to use a higher number of chips during this experiment in order to better access the required running time. Only UNet++ and DeepLabV3 were accessed since they were the models that performed the best and were thus entered into the competition. The table below illustrates our findings.

Model	**Batch Size**	**Running Time on the Cluster (seconds)**
UNet ++	32	283.760
UNet ++	16	305.369
UNet ++	8	311.701
UNet ++	4	332.433
DeepLabV3	32	376.354
DeepLabV3	16	391.593
DeepLabV3	8	402.536
DeepLabV3	4	402.813

Table 4: Runtime comparison for predictions

Overall, Unet++ runs faster than DeepLabV3 with the same hyperparameters, which may be explained by the simpler architecture of Unet++. With the increase in the data size, this difference may have been crucial to run within the time constraint. The accepted model was a

Unet++ with a batch size of 32 and 6 workers. Unfortunately, in later submissions, we opted for a batch size of 4 or 8 using DeepLabV3 and 6 or 12 workers, which led to exceeding running time errors.

6. Challenges and Limitations

During the contest, many challenges were faced and had to be overcome. Starting from the reproduction of the benchmark notebook provided by the organizers of the contest, where it was not possible to reproduce their environment and run it on the UBS servers. The environments could not be installed correctly, mostly due to dependency issues and errors in conda when trying to install the provided yaml files. Also, while tie downloading of other bands was successful on our servers, reproducing the same in the submission docker proved to be quite complex. With differences in writing/reading permissions and the docker's nature of being a "black box" where it is not quite clear what is happening, in combination with the long waiting times, led to us using only the pre-downloaded three bands.

A whole variety of issues had to be overcome when adapting the binary segmentation template[5]. The structure of this template was quite complex and required some trial-and-error to get it to work, which was successful in the end and each group could implement their model based on the template.

The submission procedure proved to be quite difficult as well. As outlined in the 'submissions chapter, the main problems were getting the prediction module to work on their docker, working around the environment, path and runtime limitation issues. During the time when the prediction did not work yet, the model's quality was assessed only through our own train/test split. Not having the submission on the server forced us to rely only on these results, optimizing the models to perform well on our own evaluation method. While the performance measured by us was quite high, it was shocking to see the lower results on the competition platform. Having more chances to submit predictions and using these results to optimize our models would have most likely resulted in higher performances, enabling us to generalize the models better.

[5] https://github.com/gradient-descendant/binary_segmentation_template

7. References

[1] O. Ghorbanzadeh, D. Tiede, Z. Dabiri, M. Sudmanns, and S. Lang, “DWELLING EXTRACTION IN REFUGEE CAMPS USING CNN – FIRST EXPERIENCES AND LESSONS LEARNT,” in *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Sep. 2018, vol. XLII–1, pp. 161–166. doi: 10.5194/isprs-archives-XLII-1-161-2018.

[2] X. Liu, F. Han, K. H. Ghazali, I. I. Mohamed, and Y. Zhao, “A review of Convolutional Neural Networks in Remote Sensing Image,” in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, New York, NY, USA, Feb. 2019, pp. 263–267. doi: 10.1145/3316615.3316712.

[3] S. Hao, Y. Zhou, and Y. Guo, “A Brief Survey on Semantic Segmentation with Deep Learning,” *Neurocomputing*, vol. 406, pp. 302–321, Sep. 2020, doi: 10.1016/j.neucom.2019.11.118.

[4] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, “A Review on Deep Learning Techniques Applied to Semantic Segmentation,” *ArXiv170406857 Cs*, Apr. 2017, Accessed: Feb. 10, 2022. [Online]. Available: http://arxiv.org/abs/1704.06857

[5] G. Wu *et al.*, “Automatic Building Segmentation of Aerial Imagery Using Multi-Constraint Fully Convolutional Networks,” *Remote Sens.*, vol. 10, no. 3, Art. no. 3, Mar. 2018, doi: 10.3390/rs10030407.

[6] L. Zhang, J. Wu, Y. Fan, H. Gao, and Y. Shao, “An Efficient Building Extraction Method from High Spatial Resolution Remote Sensing Images Based on Improved Mask R-CNN,” *Sensors*, vol. 20, no. 5, Art. no. 5, Jan. 2020, doi: 10.3390/s20051465.

[7] Z. Zou, W. Li, T. Shi, Z. Shi, and J. Ye, “Generative Adversarial Training for Weakly Supervised Cloud Matting,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 201–210. doi: 10.1109/ICCV.2019.00029.

[8] S. Mohajerani and P. Saeedi, “Cloud and Cloud Shadow Segmentation for Remote Sensing Imagery via Filtered Jaccard Loss Function and Parametric Augmentation,” *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 14, pp. 4254–4266, 2021, doi: 10.1109/JSTARS.2021.3070786.

[9] J. H. Yim *et al.*, “Comparative Analysis of Methods Based on Semantic Segmentation for Cloud Detection in Remote Sensing Imagery,” in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*, Oct. 2021, pp. 1156–1159. doi: 10.1109/ICTC52510.2021.9620222.

[10] M. Xia, T. Wang, Y. Zhang, J. Liu, and Y. Xu, “Cloud/shadow segmentation based on global attention feature fusion residual network for remote sensing imagery,” *Int. J. Remote Sens.*, vol. 42, no. 6, pp. 2022–2045, Mar. 2021, doi: 10.1080/01431161.2020.1849852.

[11] L. Jiao, L. Huo, C. Hu, and P. Tang, “Refined UNet: UNet-Based Refinement Network for Cloud and Shadow Precise Segmentation,” *Remote Sens.*, vol. 12, no. 12, Art. no. 12, Jan. 2020, doi: 10.3390/rs12122001.

[12] A. Geethu Chandran and J. Christy, “A survey of cloud detection techniques for satellite images,” *Int. Res. J. Eng. Technol. IRJET*, 2015.

[13] D. Patel and S. Rami, “Deep Learning Technique for Cloud Detection using Satellite Data,” vol. 7, no. 7, pp. 33–39, Jul. 2019.

[14] T. Bai, D. Li, K. Sun, Y. Chen, and W. Li, “Cloud Detection for High-Resolution Satellite Imagery Using Machine Learning and Multi-Feature Fusion,” *Remote Sens.*, vol. 8, no. 9, Art. no. 9, Sep. 2016, doi: 10.3390/rs8090715.

[15] T. Maestri *et al.*, “Analysis of cirrus cloud spectral signatures in the far infrared,” *J. Quant. Spectrosc. Radiat. Transf.*, vol. 141, pp. 49–64, Jul. 2014, doi: 10.1016/j.jqsrt.2014.02.030.

[16] S. Song *et al.*, “The spectral signature of cloud spatial structure in shortwave irradiance,” *Atmospheric Chem. Phys.*, vol. 16, no. 21, pp. 13791–13806, Nov. 2016, doi: 10.5194/acp-16-13791-2016.

[17] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs,”

ArXiv160600915 Cs, May 2017, Accessed: Feb. 10, 2022. [Online]. Available: http://arxiv.org/abs/1606.00915

[18] Z. Li, H. Shen, Q. Cheng, Y. Liu, S. You, and Z. He, "Deep learning based cloud detection for medium and high resolution remote sensing images of different sensors," *ISPRS J. Photogramm. Remote Sens.*, vol. 150, pp. 197–212, Apr. 2019, doi: 10.1016/j.isprsjprs.2019.02.017.

[19] Y. Guo, X. Cao, B. Liu, and M. Gao, "Cloud Detection for Satellite Imagery Using Attention-Based U-Net Convolutional Neural Network," *Symmetry*, vol. 12, no. 6, Art. no. 6, Jun. 2020, doi: 10.3390/sym12061056.

[20] J. H. Jeppesen, R. H. Jacobsen, F. Inceoglu, and T. S. Toftegaard, "A cloud detection algorithm for satellite imagery based on deep learning," *Remote Sens. Environ.*, vol. 229, pp. 247–259, Aug. 2019, doi: 10.1016/j.rse.2019.03.039.

[21] M. Wieland, Y. Li, and S. Martinis, "Multi-sensor cloud and cloud shadow segmentation with a convolutional neural network," *Remote Sens. Environ.*, vol. 230, p. 111203, Sep. 2019, doi: 10.1016/j.rse.2019.05.022.

[22] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *ArXiv151100561 Cs*, Oct. 2016, Accessed: Feb. 10, 2022. [Online]. Available: http://arxiv.org/abs/1511.00561

[23] N. Audebert, B. Le Saux, and S. Lefèvre, "Beyond RGB: Very High Resolution Urban Remote Sensing With Multimodal Deep Networks," *ISPRS J. Photogramm. Remote Sens.*, vol. 140, pp. 20–32, 2018, doi: 10.1016/j.isprsjprs.2017.11.011.

[24] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," *ArXiv161201105 Cs*, Apr. 2017, Accessed: Feb. 10, 2022. [Online]. Available: http://arxiv.org/abs/1612.01105

[25] L. Chen, X. Dou, J. Peng, W. Li, B. Sun, and H. Li, "EFCNet: Ensemble Full Convolutional Network for Semantic Segmentation of High-Resolution Remote Sensing Images," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022, doi: 10.1109/LGRS.2021.3076093.

[26] M. Luotamo, S. Metsämäki, and A. Klami, "Multiscale Cloud Detection in Remote Sensing Images Using a Dual Convolutional Neural Network," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 6, pp. 4972–4983, Jun. 2021, doi: 10.1109/TGRS.2020.3015272.

[27] M. Segal-Rozenhaimer, A. Li, K. Das, and V. Chirayath, "Cloud detection algorithm for multi-modal satellite imagery using convolutional neural-networks (CNN)," *Remote Sens. Environ.*, vol. 237, p. 111446, Feb. 2020, doi: 10.1016/j.rse.2019.111446.

[28] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," *ArXiv170605587 Cs*, Dec. 2017, Accessed: Feb. 10, 2022. [Online]. Available: http://arxiv.org/abs/1706.05587

[29] "Summary of DeepLabv3 paper · Swetha's Blog." https://swethatanamala.github.io/2018/08/15/summary-of-DeepLabv3-paper/ (accessed Feb. 10, 2022).

[30] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: A Nested U-Net Architecture for Medical Image Segmentation," *ArXiv180710165 Cs Eess Stat*, Jul. 2018, Accessed: Feb. 10, 2022. [Online]. Available: http://arxiv.org/abs/1807.10165

[31] "Distance transform regression for spatially-aware deep semantic segmentation," *DeepAI*, Sep. 04, 2019. https://deepai.org/publication/distance-transform-regression-for-spatially-aware-deep-semantic-segmentation (accessed Feb. 10, 2022).

[32] "Cirrus Cloud - Description | Technology Trends." https://www.primidi.com/cirrus_cloud/description (accessed Feb. 10, 2022).

[33] "2.6.8.20. Cleaning segmentation with mathematical morphology — Scipy lecture notes." https://scipy-lectures.org/advanced/image_processing/auto_examples/plot_clean_morpho.html (accessed Feb. 10, 2022).

[34] N. Champion, "Automatic Cloud Detection from Multi-Temporal Satellite Images: Towards the Use of PLÉIADES Time Series," *ISPRS - Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 39B3,

pp. 559–564, Aug. 2012, doi: 10.5194/isprsarchives-XXXIX-B3-559-2012.
[35] H. Tang, K. Yu, O. Hagolle, K. Jiang, X. Geng, and Y. Zhao, "A cloud detection method based on a time series of MODIS surface reflectance images," *Int. J. Digit. Earth*, vol. 6, pp. 157–171, Dec. 2013, doi: 10.1080/17538947.2013.833313.
[36] H. Zhang, Q. Huang, H. Zhai, and L. Zhang, "Multi-temporal cloud detection based on robust PCA for optical remote sensing imagery," *Comput. Electron. Agric.*, vol. 188, p. 106342, Sep. 2021, doi: 10.1016/j.compag.2021.106342.
[37] "OmegaConf — OmegaConf 2.1.1 documentation." https://omegaconf.readthedocs.io/en/2.1_branch/ (accessed Feb. 10, 2022).