ETH zürich

Project report on

"Cloud Segmentation with the Infrared All-sky Cloud Camera (IRCCAM) "

Data Science Lab

January 2021

Authors

Elrich Groenewald
Henry Valeyre
Luka Lodrant

Supervised by

Prof. Dr. Ce Zhang

In collaboration with

Dr. Julian Gröbner, PMOD/WRC

Abstract

Cloud coverage is an important metric in weather prediction but is still most commonly determined by human observation. Automatic measurement using an RGB all-sky camera is unfortunately limited to daytime. To alleviate this problem the team at PMOD/WRC developed a prototype thermal infrared camera (IRCCAM). Their previous work approached the cloud segmentation of these images through fixed thresholding which had problems with consistently detecting thin cirrus clouds. We used the RGB images taken at the same location to create a labelled dataset on which we trained a deep learning cloud segmentation model. The resulting algorithm matches the previous approach in detecting thicker clouds and qualitatively outperforms it in detecting thinner cloud. We believe that coupled with the IRCCAM our model is comparable to human observation and can be used for continuous cloud coverage monitoring anywhere.

Contents

1 Introduction

Monitoring fractional cloud coverage is useful for a few different applications. It is an important metric for weather prediction and it can be used to model the earth's radiation budget, which has applications in climate change research. Currently, the most common method for determining cloud fraction is by human observation. However, this method is not objective, is limited to daytime observations, and results in a limited observation frequency [1].

Various techniques have been used for automatic cloud segmentation using all-sky RGB cameras ([13], [1], [9], [4], [12]), the results of which can be used to calculate cloud fraction. However, using RGB cameras is also limited to daytime observations.

The Infrared all-sky Cloud Camera (IRCCAM) [1] was developed and installed at the Physikalisch-Meteorologisches Observatorium Davos, World Radiation Center (PMOD/WRC) in Davos, Switzerland, to improve automated measurement of cloud coverage. The IRCCAM consists of a thermal infrared camera that points down onto a spherical, gold-plated mirror, to get a view of the whole visible sky. It takes a 640×480 pixel image every minute, where each pixel value is a brightness temperature measurement.

The IRCCAM has the advantage of being able to measure brightness temperature during both daytime and nighttime, so it can be used to continuously monitor cloud fraction. The current approach used to segment clouds in IRCCAM images uses a fixed threshold temperature value. A clear-sky reference model is subtracted from the raw data to equalize temperature throughout the camera view, then pixels with a value greater than the fixed threshold are predicted as cloud, and the rest as sky.

The fixed threshold method can accurately detect thick clouds, as their temperature differs significantly from the surrounding sky. However, this approach struggles to accurately detect thin, high-altitude cirrus clouds. The temperature of these clouds is close to that of the surrounding sky, and adjusting the fixed threshold results in either missing many cloud pixels or falsely predicting many sky pixels as clouds.

The goal of the project was to develop an algorithm for semantic segmentation of the IRCCAM images which requires no additional inputs other than the image and the clear-sky model and is efficient enough to detect clouds in real-time completely autonomously.

We investigated a deep learning approach for cloud segmentation using the IRCCAM. We used daytime RGB images taken from an RGB camera near the IRCCAM to create the labels for training our IRCCAM model. Through qualitative evaluation, we found that our model performs better than current approaches for IRCCAM cloud segmentation, and is considerably better at detecting high-altitude cirrus clouds. All the code required to create the dataset and to train our model is available on GitHub. [1]

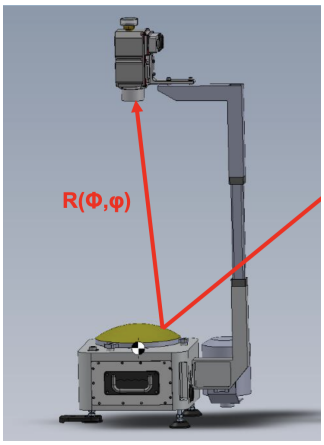

Figure 1: IRCCAM Model

Figure 2: IRCCAM in Davos

[1] https://github.com/elrichgro/irccam-pmodwrc

2 Models and Methods

2.1 Overview

The main problem of any deep learning-based approach was the lack of any ground-truth labels for training the model. We could manually create labels for a set of images, but manual labelling for image segmentation is extremely tedious and time-consuming.

Luckily, an all-sky RGB camera has been deployed at the same location. Thin, high-altitude clouds are easier to detect in RGB images than in thermal infrared images because although their temperature does not differ much from the surrounding sky, their colour does. These clouds are still significantly whiter than clear sky.

The idea was to first develop an algorithm to produce the labels from the RGB images and then use those labels to train our final model. We made two assumptions here. First, we assumed that the nighttime IRCCAM images do not differ from daytime so the model trained on daily images would still work for them and secondly we assumed that unsupervised cloud segmentation is easier on the RGB images.

Our approach consists of the following steps:

1. Align the RGB and IRCCAM images since they do not capture the same perspective.
2. Develop a cloud segmentation algorithm for the RGB images to produce the training labels.
3. Clean up the dataset for training. Remove the nighttime images and all images affected by bad weather conditions or camera malfunction.
4. Train a deep convolutional neural network model on the cleaned up and labelled dataset.

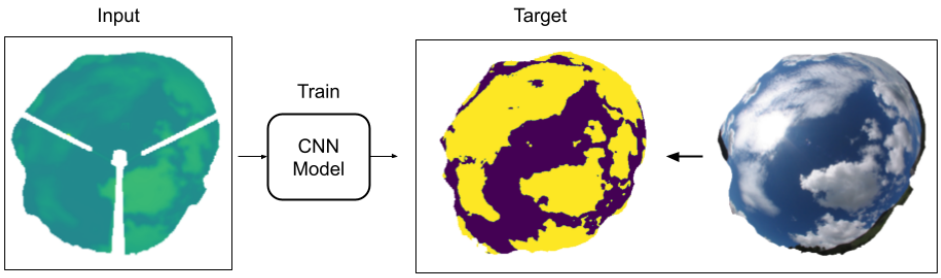

Figure 3: Overview of our approach

2.2 Data acquisition

We were given IRCCAM data taken in Davos between 1.1.2018 and 31.12.2018. This data included 640×480 thermal infrared images taken every minute for the whole year (with timestamps), as well as theoretical clear-sky model data computed for each image timestamp. The clear-sky model data is the theoretical temperature that the clear sky should be at any given time throughout the camera view. An example is shown in Figure 4.

Along with the IRCCAM images and clear-sky data, we were given two sets of RGB images at different exposures taken from an all-sky RGB camera mounted near the IRCCAM. The entire dataset was roughly 900GB in size. In the preprocessing step we took the IRCCAM image, single RGB image and the clear-sky reference and packed them into daily timestamped datasets which we saved as compressed HDF files to make them easier to work with.

2.3 Image alignment

The RGB camera and IRCCAM have slightly different views of the sky since they have different lens distortions and are not at the exact same location on the ground. So in order to use the RGB images to create labels for the IRCCAM, we had to align their views of the sky.

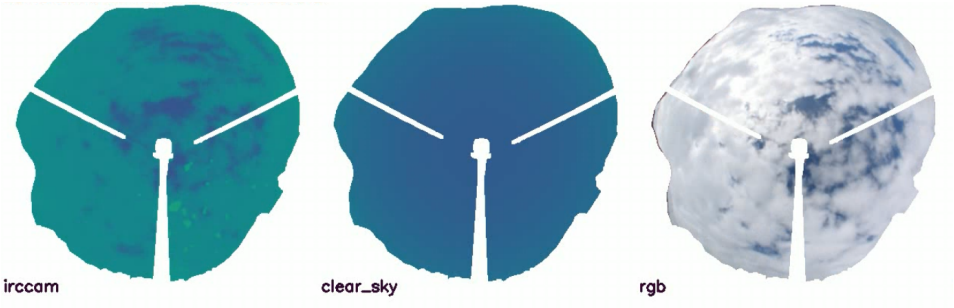

Figure 4: Example of all the input images

To do this we used the SIFT [5] algorithm to match corresponding points in RGB and IRCCAM images taken at the same time of day. The SIFT algorithm finds points of interest in each image and for each point, it computes a high-dimensional descriptor vector. To match points across two images, we used a brute force algorithm that matches each point in the first image to the closest point in the second image, based on the euclidean distance between the corresponding descriptor vectors. We filtered out matches that were clearly wrong based on pixel distance between the matched points, and the remaining outliers were removed using the RANSAC [3] algorithm. Figure 5 shows matching SIFT features on an RGB to IRCCAM image pair after outliers have been removed.

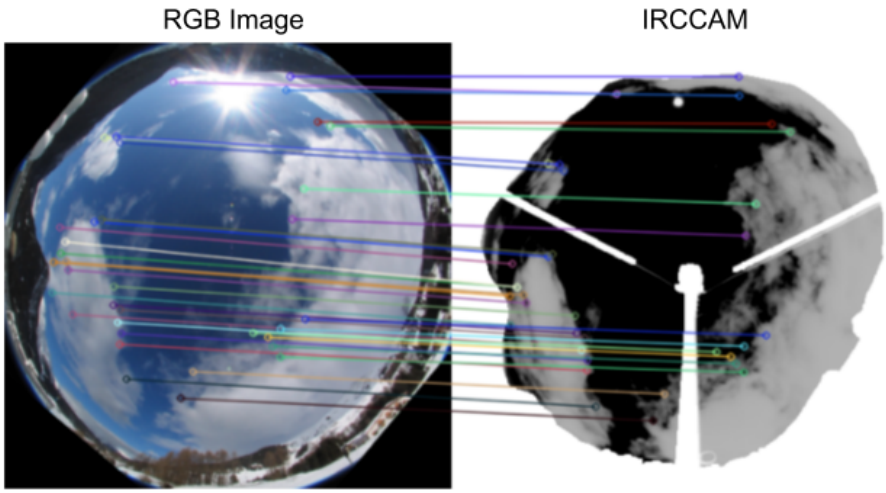

Figure 5: SIFT feature matches (after removing outliers)

For a given RGB to IRCCAM pair we do the following.

1. Compute the SIFT features for both images.
2. Use a brute force matching algorithm that matches SIFT features in the RGB image to the closest corresponding SIFT feature in the IRCCAM image.
3. Filter out matches with a pixel distance greater than 140 pixels between the two matched points.
4. Use the RANSAC algorithm to remove outlier matches.

Using the matches over multiple pairs of RGB and IRCCAM images, we calculated a transformation matrix that transforms the RGB view to match up with the IRCCAM view. This transformation matrix was computed once and used for all RGB to IRCCAM pairs, which could be done because both cameras stay at a fixed location on the ground.

Figure 6 shows an example of an RGB to IRRCAM image pair, before and after the alignment transformation has been applied. The IRCCAM image is shown with a red tint on top of the RGB image.

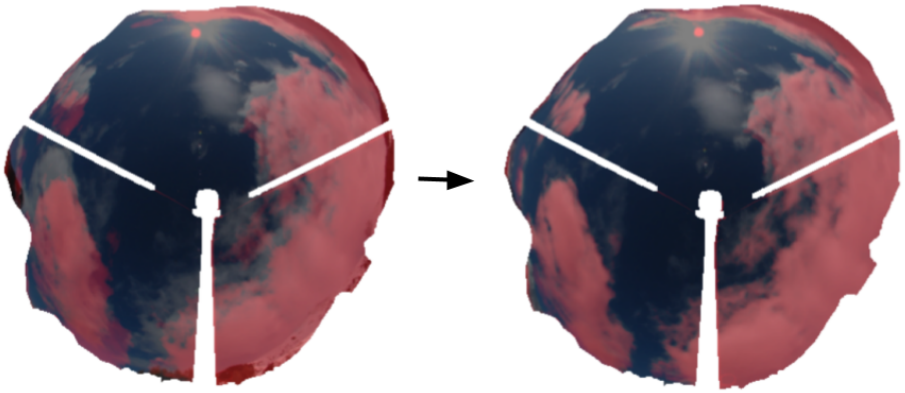

Figure 6: Before (left) and after (right) image alignment

2.4 RGB image segmentation

There are a few common approaches for cloud segmentation in all-sky RGB cameras. The simplest is fixed thresholding. This approach takes the ratio of the red and blue channels of an RGB image and applies a fixed threshold over all pixels. Pixels with a ratio greater than the threshold are classified as "cloud", and those with a ratio less than the threshold are classified as "sky". Due to varying conditions of the images (contrast, brightness, etc.) and varying cloud formations, no single threshold works well for all the images. Our threshold selection is presented later together with dataset filtering.

Adaptive thresholding alleviates some of these problems. It can adapt per image (where a threshold is based on image properties), or per-pixel (where the threshold is computed based on the neighbourhood around each pixel). For example, Otsu's method [6] determines a threshold per-image by maximising the inter-class variance in pixel intensities between the two classes it is separating.

We also experimented with a few other approaches for RGB cloud segmentation, including superpixel segmentation using a min-cut max-flow algorithm, and optical flow algorithms which use multiple consecutive images to segment based on which pixels move together over time.

However, our best results were achieved using a combination of per-image adaptive thresholding and per-pixel adaptive thresholding. The algorithm we used is as follows. For each image:

1. Determine threshold A for the entire image using Otsu's algorithm.
2. For each pixel i:
   (a) Compute the mean intensity, μ_i, of the 111×111 neighbourhood around the pixel and set threshold $B_i = \mu_i - 1$.
   (b) If the pixel intensity is greater than A or greater than B_i, classify as "cloud", otherwise, classify as "sky".

We found that the Otsu thresholding worked better for mixed cloud/sky conditions, and the per-pixel adaptive thresholding worked better for detecting cirrus clouds. By combining the two approaches, the algorithm worked well in a majority of cloud conditions. An example of our RGB labelling algorithm is shown in Figure 7.

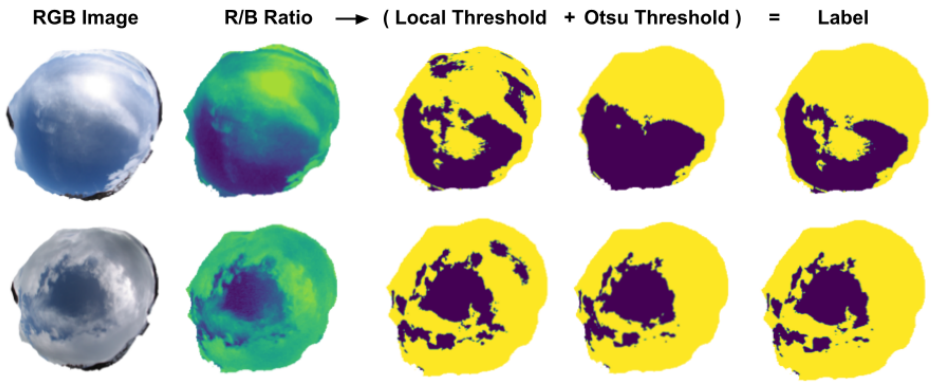

Figure 7: Process for cloud segmentation in RGB images

2.5 Dataset grooming

After aligning the images and creating RGB labels, we filtered out "bad" data from the dataset to prepare it for training. First, we filtered out nighttime data, which is too dark to obtain accurate RGB labels (this also includes early morning and late evening).

From the remaining data, we kept only every 10^{th} image. Images taken one minute apart are highly correlated since clouds do not move much in one minute. We kept only every 10^{th} image to reduce the correlation between training data points.

On some days, the IRCCAM would produce faulty (completely blank) data, so we filtered out these days. We also filtered out images taken during bad weather conditions. Some examples include heavy rain, heavy snowfall, iced-over camera (on winter mornings), and exceptional sun glare. Figures 8 and 9 show examples of bad weather conditions.

Figure 8: Morning ice over camera lens

Figure 9: Snow covering camera lens

To facilitate this data-filtering process, we created a video clip for each day of data which iterates through all the timestamps at which images were taken throughout the day and for each timestamp compiled a frame including the timestamp, the IRCCAM image, the RGB image, as well as the semantic segmentation masks produced by the adaptive thresholding method and fixed thresholding method with three different fixed thresholds. We then watched the videos and noted down the non-problematic sections of the day to use in the final dataset. Figure 10 shows an example of one of these frames.

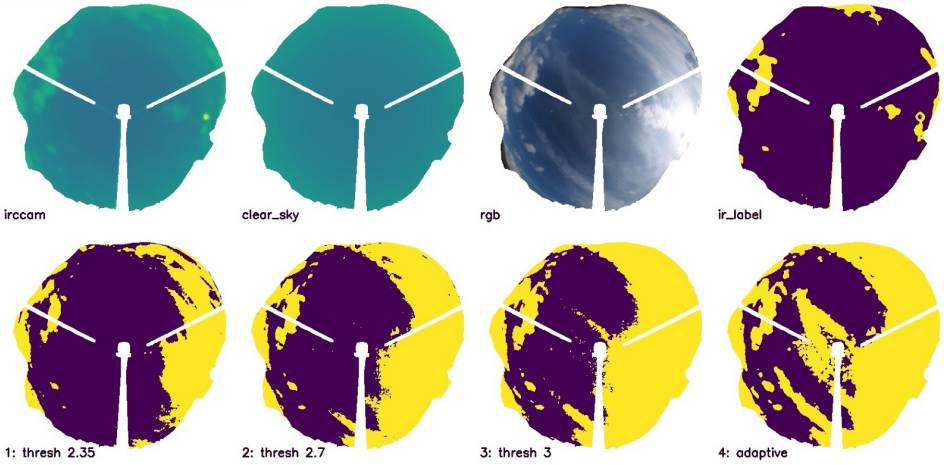

Figure 10: Example frame from video for filtering out "bad" data

While filtering out the "bad" data, we also selected the best performing RGB labelling algorithm for the given day. Different options can be seen in the lower row of Figure 10, first three different fixed thresholds and then the adaptive thresholding. The choices in many cases were not easy because different RGB algorithms produced very different predictions (some predicted too much sky and some predicted too little). The selection was also quite subjective since in case of some very thin clouds we were not completely certain whether to even classify them as clouds.

The final, filtered dataset is 6GB large and contains just over 8000 images, which makes it easy enough to work with and still large enough for training a deep neural network.

2.6 Sun masking

On sunny days, the sun causes bright lens flare in RGB images. As a result, the sun and the area around the sun are predicted as clouds in these images.

To mitigate this problem we chose the simple solution of masking out the sun and area around it with a circular mask. For each image, we used the `pysolar` library to calculate the azimuth and zenith angle of the sun using the timestamp of the image, and the latitude and longitude of the IRCCAM. We used the following equation (based on a similar computation in [12]) to map this to an (x, y) pixel location in the image:

$$x = c_x - \frac{2r\ \theta_z \sin\theta_a}{\pi}, \quad y = c_y + \frac{2r\ \theta_z \cos\theta_a}{\pi},$$

where (c_x, c_y) is the center of the IRCCAM view in the image, r is the radius of the sky view (in pixels), and θ_z and θ_a are the zenith and azimuth angles of the sun.

Given the pixel location of the sun, we masked out a circle with a radius of 40 pixels. Figure 11 shows an example of this. The radius was chosen manually, with a trade-off between masking out enough of the lens flare around the sun and not removing too much useful information from the image. For training, the value of the masked pixels was set to -80 °C and the masked pixels in the corresponding training label were set to the "sky" label.

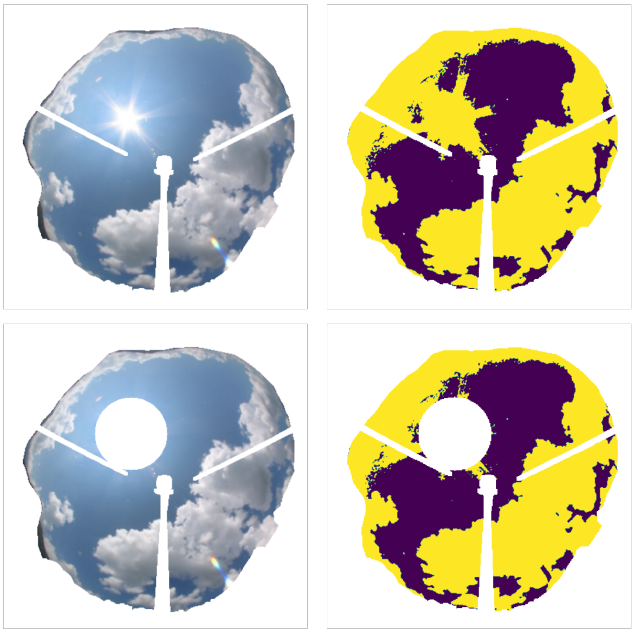

Figure 11: Example of sun masking

2.7 Data augmentation

Before feeding our filtered data to the models for training we also performed some slight augmentation to it. We normalized the brightness temperature by setting all the pixels with values outside the interval $[-80, 60]$ to the boundary values and then rescaling it to the interval $[0, 1]$. We also masked out the IRCCAM structure in the images, as well as the ground and buildings around the IRCCAM. This mask was the same for all images. The masked input values were set to 0 and the masked label values were set to an "ignore" class, which was ignored when calculating the loss and validation metrics.

This represented our base dataset on which we could train the models, but we also performed two additional modifications to the dataset and compared the results of all options.

The first modification was to perform histogram equalization on the IRCCAM images. This process enhances an image with low contrast by spreading out the most frequent intensity values in an image [10].

Secondly, instead of training the model directly on the brightness temperature measurements, we tried training it on the differences between the measurements and the theoretical clear-sky model. The idea was to reduce the effect of the absolute temperature on the model since the relevant information is mostly embedded in the differences.

An example of the same base image with both of the operations applied is shown in Figure 12. To the human eye, the difference that histogram equalization makes is quite large, but the models mostly preferred the more balanced images that the clear-sky differences produce.

2.8 Models and Training

We compared the performance of two model architectures. The first is the U-Net model [8] which was originally introduced for medical image segmentation. The second is the DeepLab v3 model introduced by Chen et al. [2]. Both models follow the typical encoder-decoder architecture, where the encoder consists of a series of convolutions and max-pooling that map the inputs to a high-dimensional latent representation. The decoder uses this feature representation to predict a segmentation mask.

For training, we split our dataset into training, validation, and test splits of sizes 60%, 20%, and 20%, respectively. For each split, we randomly selected the required number of days and assigned

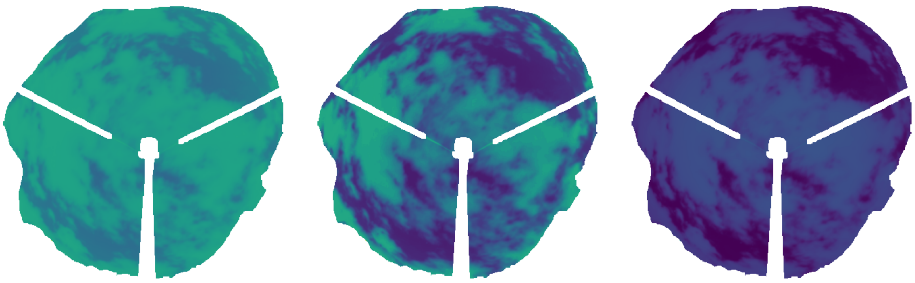

Figure 12: From left to right: raw IRCCAM image, histogram equalized image, difference between the raw image and the clear-sky model

	Data Augmentation		
Model	None	Clear-sky	Equalization
U-Net	0.74	0.76	0.74
DeepLab v3	0.76	**0.77**	0.72

Table 1: Validation IOU comparison

all images from those days to the corresponding split. The split was done by day to minimize data leakage between dataset splits.

We implemented the training infrastructure and data augmentation in `PyTorch` [7] and used slightly adapted pre-implemented versions of U-Net and DeepLab v3 as our models. The pre-implemented version had to be adapted to support the single-channel IRCCAM data, as they were originally implemented for RGB images.

3 Results

We evaluated the performance of our models on the validation dataset using the Intersection over Union (IOU) metric. This metric takes the intersection of the labeled cloud area and predicted cloud area and divides it by the union of them.

$$\text{IOU} = \frac{\text{labeled area} \cap \text{predicted area}}{\text{labeled area} \cup \text{predicted area}} \tag{1}$$

The range of possible values is $[0, 1]$ with 1 meaning perfect accuracy.

The validation IOU metrics for different model and input combinations are shown in Table 1. We found that the best performance was achieved with the DeepLab v3 model using the IRCCAM data adjusted with the clear-sky model.

For both models, subtracting the clear-sky reference from the raw inputs improved performance. This is most likely due to the fact that the most useful information for detecting a cloud edge is the difference between the cloud temperature and the sky temperature. Subtracting the clear-sky model from the raw inputs equalizes the sky temperature throughout the image, reducing the effect of the increased sky temperature for pixels further away from the zenith.

Histogram equalization did not yield any significant improvements, in the case of DeepLab v3 it even reduced the final validation scores. In case of a completely clear sky, the equalization resulted in artificial contrast around the edges of the image which could be difficult to discern from a cloud.

The best performing model was qualitatively compared to the current IRCCAM segmentation approach on a test dataset, as well as on nighttime IRCCAM images. Based on our qualitative assessment, we found that our model detects regular clouds with a similar level of accuracy as the fixed threshold method and outperforms it in the case of cirrus clouds. Some examples of the segmentation results of our model and the current fixed thresholding approach are shown in figures 13 and 14.

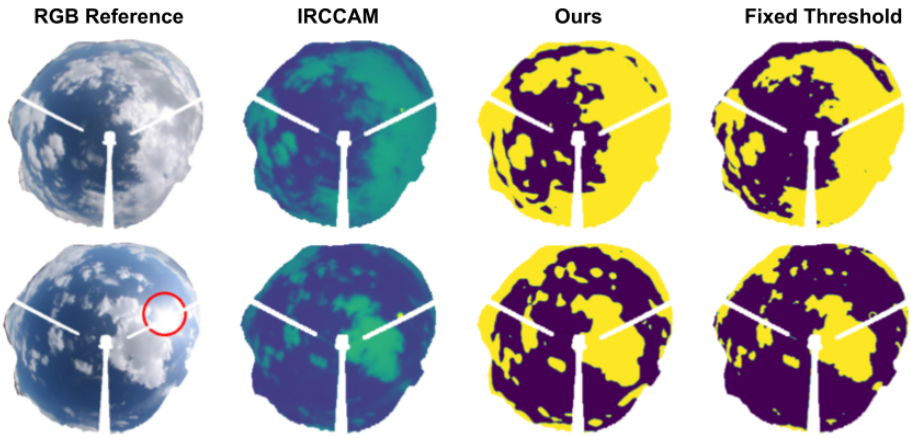

Figure 13: Our model vs fixed threshold method (regular clouds)

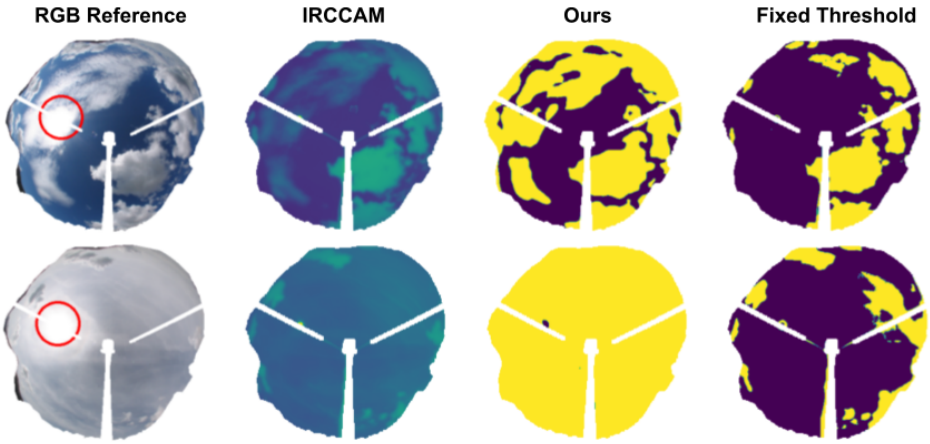

Figure 14: Our model vs fixed threshold method (cirrus clouds)

4 Conclusion and future work

While the quantitative comparison between our model and any previous attempts is impossible due to the lack of ground truth labels, we believe that qualitatively our model outperforms them in most circumstances. Even so, there are still a lot of possible research avenues to improve its performance.

The training dataset we have produced as a part of this project is quite extensive but the final model could still be improved by having more data, especially if the images could be more diverse. A big part of the problem is that all the data was taken at the same location and is therefore hard to judge how the model would perform with images taken in another location with different weather patterns. By utilizing more IRCCAM and RGB images from different locations we believe the model could be improved to be even more general.

Any improvement to the RGB image segmentation would also directly convert into final model performance. SegCloud ([11] is a novel cloud image segmentation model using similar neural network structure to ours but is focused on RGB images. By using a more advanced model like this to produce our training labels the effectiveness our IRCCAM model could certainly improve.

Most of the research in this project was focused on the training dataset preparation so the deep learning model is not so finely tuned. This is due to both a lack of time and our observation that any improvement to the training dataset and the labels improved the final model much more than the marginal improvements we achieved through hyperparameter tuning. Spending more time on hyperparameter tuning could improve the final model performance.

References

[1] C. Aebi, J. Gröbner, and N. Kämpfer. "Cloud fraction determined by thermal infrared and visible all-sky cameras". In: *Atmospheric Measurement Techniques* 11.10 (2018), pp. 5549–5563. DOI: 10.5194/amt-11-5549-2018. URL: https://amt.copernicus.org/articles/11/5549/2018/.

[2] Liang-Chieh Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. arXiv: 1706.05587 [cs.CV].

[3] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". In: *Commun. ACM* 24.6 (June 1981), 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692. URL: https://doi.org/10.1145/358669.358692.

[4] Qingyong Li et al. "Thin Cloud Detection of All-Sky Images Using Markov Random Fields". en. In: *IEEE Geoscience and Remote Sensing Letters* 9.3 (May 2012), pp. 417–421. ISSN: 1545-598X, 1558-0571. DOI: 10.1109/LGRS.2011.2170953. URL: http://ieeexplore.ieee.org/document/6071030/ (visited on 01/27/2021).

[5] D.G. Lowe. "Object recognition from local scale-invariant features". en. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Kerkyra, Greece: IEEE, 1999, 1150–1157 vol.2. ISBN: 978-0-7695-0164-2. DOI: 10.1109/ICCV.1999.790410. URL: http://ieeexplore.ieee.org/document/790410/ (visited on 01/27/2021).

[6] N. Otsu. "A Threshold Selection Method from Gray-Level Histograms". In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), pp. 62–66. DOI: 10.1109/TSMC.1979.4310076.

[7] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV].

[9] Cunzhao Shi et al. "Ground-Based Cloud Detection Using Graph Model Built Upon Superpixels". en. In: *IEEE Geoscience and Remote Sensing Letters* 14.5 (May 2017), pp. 719–723. ISSN: 1545-598X, 1558-0571. DOI: 10.1109/LGRS.2017.2676007. URL: http://ieeexplore.ieee.org/document/7885057/ (visited on 01/27/2021).

[10] Wikipedia contributors. *Histogram equalization*. [Online; accessed 06-January-2020]. 2006. URL: https://en.wikipedia.org/wiki/Histogram_equalization.

[11] W. Xie et al. "SegCloud: a novel cloud image segmentation model using a deep convolutional neural network for ground-based all-sky-view camera observation". In: *Atmospheric Measurement Techniques* 13.4 (2020), pp. 1953–1961. DOI: 10.5194/amt-13-1953-2020. URL: https://amt.copernicus.org/articles/13/1953/2020/.

[12] J. Yang et al. "An automated cloud detection method based on the green channel of total-sky visible images". en. In: *Atmospheric Measurement Techniques* 8.11 (Nov. 2015), pp. 4671–4679. ISSN: 1867-8548. DOI: 10.5194/amt-8-4671-2015. URL: https://amt.copernicus.org/articles/8/4671/2015/ (visited on 01/27/2021).

[13] Jun Yang et al. "An Automated Cirrus Cloud Detection Method for a Ground-Based Cloud Image". en. In: *Journal of Atmospheric and Oceanic Technology* 29.4 (Apr. 2012), pp. 527–537. ISSN: 0739-0572, 1520-0426. DOI: 10.1175/JTECH-D-11-00002.1. URL: http://journals.ametsoc.org/doi/10.1175/JTECH-D-11-00002.1 (visited on 01/26/2021).