RöjSat

AI Classification of Need for Clearing from Sen2 Satellite Images

Örebro University

Emma Svensson
emma@masvab.se

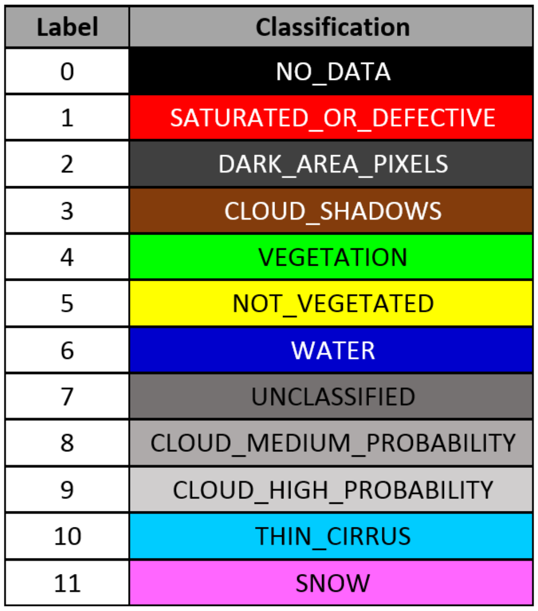
July 22, 2020

Introduction

In this report two approaches to AI classification of the need for clearing a young forest region, based on Sen2 satellite images, are explored and evaluated. The first approach uses image classification to classifying images as either containing an area in need of clearing or containing no areas in need of clearing. As the ultimate goal of the project is to achieve more detailed information the second approach attempts to use semantic segmentation to label each pixel with a degree of need for clearing. Both approaches are performed through transfer learning of pre-trained convolutional networks. Two datasets extracted from the satellite images with resolution 10×10 meters are used to trained and evaluate the models, the true color images with RGB-data for each pixel (TCI) and a set of 10 image indices calculated per pixel (S2). The raw wavelength data for top-of-atmosphere (TOP) and bottom-of-atmosphere (BOA) are also available but the methods have not yet been evaluated on them.

Satellite Data

Sen2 satellite data with a resolution of 10×10 meters over an area of roughly 68×53 km around Linköping have been used in this project. This data includes three different sub-areas; 065 (full area), 022 (south, eastern corner) and 108 (full area apart from area covered by 022). Each sample comes with the raw wavelengths for bottom-of-atmosphere (BOA) and top-of-atmosphere (TOA), a scene classification map (SCL) and the true color image (TCI). Also included in the data is a set of 10 indices calculated from the raw data, here referred to as S2. These indices are BNDVI, Chlred-edge, EVI, MIRBI, MSAVI2, MTVI2, NBR, NDRE, NDSI, SAVI and their calculations can be found here. The SCL data comes from a rule-based classification model, where for example clouds and snow has been detection through various filters on different bands in the BOA data. The classes, and their respective values and colors, for the scene classification can be seen in Figure 1, source. Throughout the project the input samples were split into a training set (80%), validation set (10%) and test set (10%). Different ways of splitting the data has been explored.



Label	Classification
0	NO_DATA
1	SATURATED_OR_DEFECTIVE
2	DARK_AREA_PIXELS
3	CLOUD_SHADOWS
4	VEGETATION
5	NOT_VEGETATED
6	WATER
7	UNCLASSIFIED
8	CLOUD_MEDIUM_PROBABILITY
9	CLOUD_HIGH_PROBABILITY
10	THIN_CIRRUS
11	SNOW

Figure 1: Scene Classification Value from built-in SCL of SENTINEL-2 Level-1C.

Image Classification

As an initial step in applying deep learning to the given problem, image classification was performed on cropped out images of the bounding boxes surrounding labeled regions for the two classes; in need of clearing and not in need of clearing.

Background

As part of the previous method for identifying forest regions with a need for clearing, namely manually going around and mapping each region, shapefiles of such regions are available specifying both regions in need of clearing and not in need of clearing. This data was used as ground truth for the task of image classification.

The convolutional network ResNet, first introduced in the paper *Deep Residual Learning for Image Recognition* [1], have been suggested as a good image classifier and is included in the image classification part of the PyTorch package torchvision.models. The network is based on the idea of a residual learning framework for better handling of very deep networks. This is done by introducing identity shortcuts between layers, which does not add any parameters to the network and thus do not increasing complexity but lets some inputs skip layers which speeds-up the process significantly. In PyTorch ResNet comes in different depths, spanning from 18 to 152 layers, and can be pre-trained on ImageNet, which is an image dataset of over 14 million samples for more than 20 000 categories. A good summery and explanation of the idea, usage and coding for ResNet can be found here. The same source has served as an inspiration for much of the code related to transfer learning.

Method

The method of image classification includes a preprocessing step of preparing the data and creating ground truth targets and also a training process of applying transfer learning on a pre-trained convolutional network. Two IPython Notebooks have been used, one for the preprocessing, `RojSat_ImageClassification_Preprocessing.ipynb`, and one for the training process, `RojSat_ImageClassification.ipynb`.

Preprocessing The shapefiles of labeled regions first have to be transformed to coordinates of pixels in the satellite images, Figure 2 shows the transformed shapes for the two classes. These shapefiles also give easy access to bounding boxes around each polygon area, which were used during the preprocessing step to crop the original satellite images. This had a number of benefits including increasing the number of samples and decreasing the size of the data significantly. The most important motivation for this step is that each image can now be labeled as one of the two classes; "includes region in need of clearing", "does not include region in need of clearing". When the data mode 'S2' is used the different features of the data are also stacked on top of each other during the preprocessing step. Lastly, the images

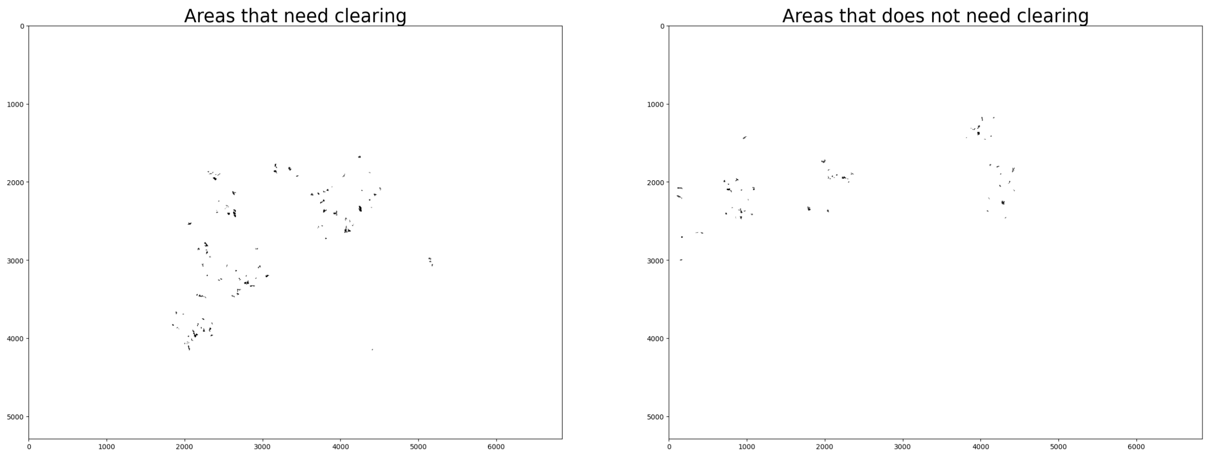


Figure 2: Heatmap of labels after they have been transformed to pixels in the Sen2 satellite TIFF images. Left panel shows the areas that need clearing and the one that doesn't need clearing are shown in the right panel.

had to be resized to a common size before they could be used as input to the network. The largest dimensions of all bounding boxes was found. Scaling the images to this size was deemed potentially harmful and instead padding with zeros was used. The actual images were placed in the upper left corner of the padded images. A visualization of the cropping and padding process is shown in Figure 3.

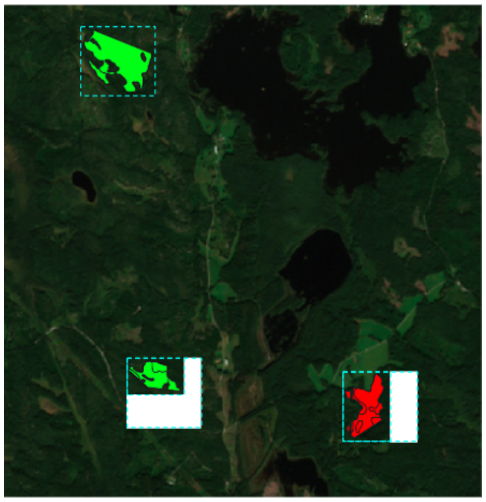


Figure 3: Three example regions, visualizing how the satellite images were cropped to each regions bounding box (dashed blue) and then padded with zeros to reach a common size (white). The green region are not in need of clearing and the red one is in need of clearing.

Transfer Learning Two different layer depths of the ResNet network, namely ResNet18 and Resnet152, were adapted to the TCI and S2 data separately by applying transfer learning, i.e. continuing to train the pre-trained models on the new data. The first layer of each network was changed to take the correct number of channels from the data, 3 for TCI and 10 for S2. The pre-trained weights for the first layer was still used but the new weights, when needed, were initialized randomly. See the setup below.

```
first_layer_weights = model.conv1.weight.clone()
model.conv1 = torch.nn.Conv2d(n_channels, 64, kernel_size=7,
                              stride=2, padding=3, bias=False)
model.conv1 = model.conv1.cuda() if use_cuda else model.conv1
with torch.no_grad():
    model.conv1.weight[:, :3] = first_layer_weights
```

The last layer was also altered to fit the new number of classes, 2, see below.

```
num_ftrs = model.fc.in_features
model.fc = torch.nn.Linear(num_ftrs, n_class)
model.fc = model.fc.cuda() if use_cuda else model.fc
```

Training was then preformed with cross entropy loss and an SGD optimizer with varying learning rates according to the following learning rate scheduler.

```
optimizer = torch.optim.SGD(list(filter(lambda p: p.requires_grad, model.parameters())),
                            lr=0.001, momentum=0.9)
exp_lr_scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.1)
```

The training was performed on the training set and evaluated on the validation set after each epoch, whilst the test set was kept separate until after the training process was completed. The last few dates of samples for all regions were used for the validation and test set.

Results

Below cross entropy losses, classification reports and confusion matrices are presented for the transfer learning and results of ResNet18 and ResNet152 on the two data sets TCI and S2.

TCI Data Figure 4 shows the training and validation cross entropy losses for a transfer learning process of ResNet adapted to the true color indices, TCI, data, for a depth of 18 layers to the left and 152 layers to the right. A classification report from scikit-learn, sklearn.metrics.classification_report, for the best models on the test set can be found in Table 1 and confusion matrices from the same evaluation are shown in Figure 5. The confusion matrix have been normalized to the number of true labels in each class.

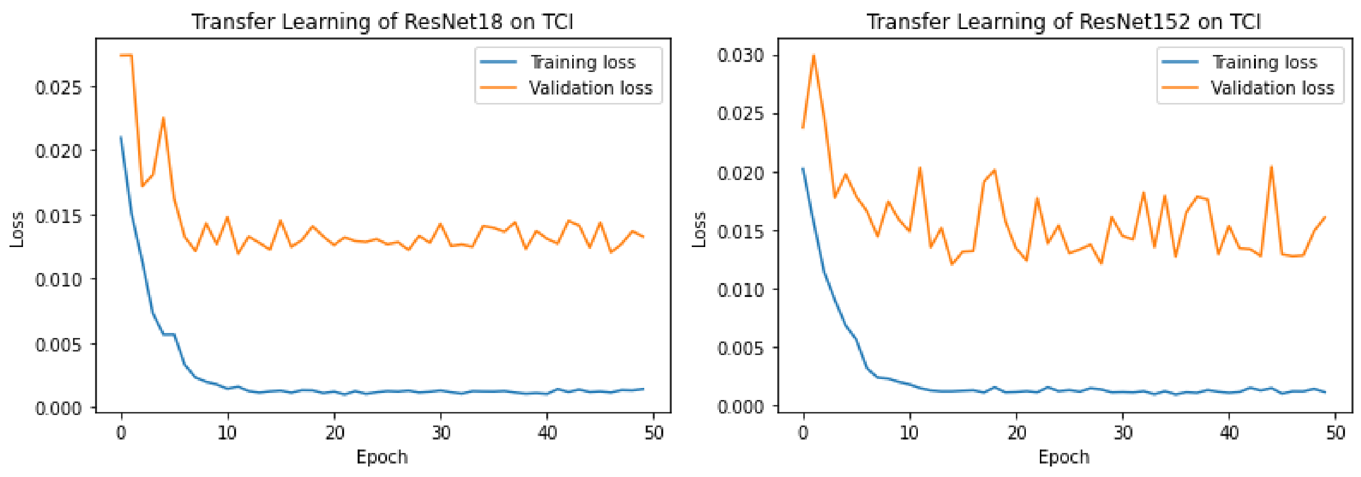


Figure 4: Transfer learning on TCI data cropped and padded to bounding boxes of regions of interest, on ResNet18 in left panel and ResNet152 in right panel.

	ResNet18				ResNet152			
	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Clear	0.83	0.83	0.83	84	0.78	0.85	0.81	84
Need Clearing	0.92	0.92	0.92	170	0.91	0.87	0.89	158
Accuracy			0.89	254			0.86	242
Macro avg	0.88	0.88	0.88	254	0.85	0.86	0.85	242
Weighted avg	0.89	0.89	0.89	254	0.87	0.86	0.86	242

Table 1: Classification report for ResNet18 and ResNet152 models on the test set of the TCI data. The support is the number of samples that the result is based on.

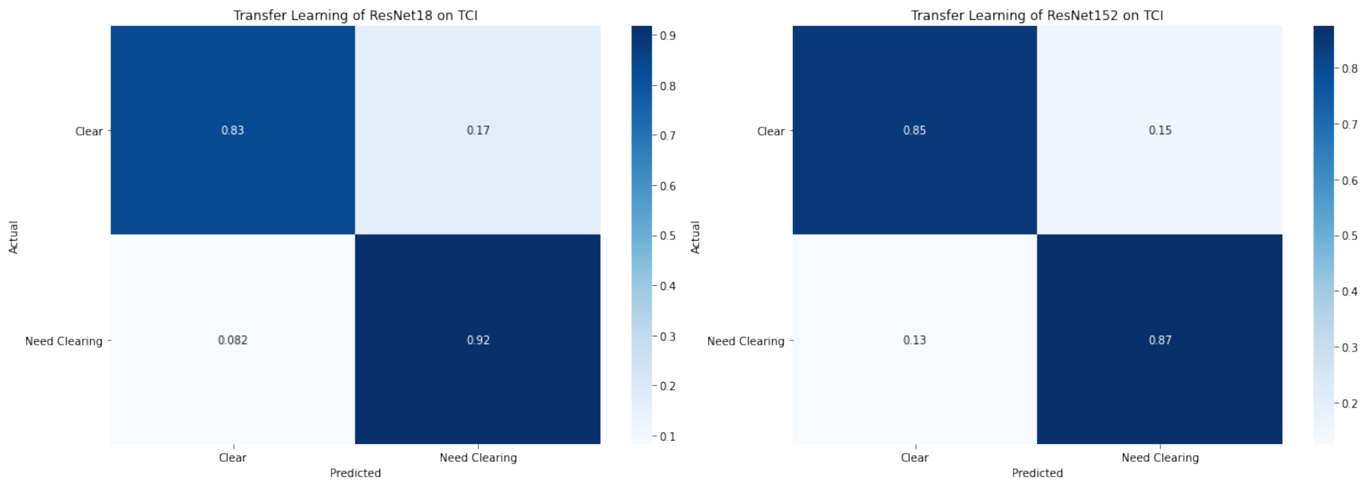


Figure 5: Confusion matrix for ResNet18 (left) and ResNet152 (right) evaluated on the test set of the TCI data. The presented values have been normalized to the total number of true labels, i.e. by each row in the matrix.

S2 data Figure 6 shows the same result as above but for when the transfer learning was instead adapted to the image indices in the S2 data. Table 2 presents the classifications reports and Figure 7 shows the confusion matrices for the same evaluations on test sets of the S2 data.

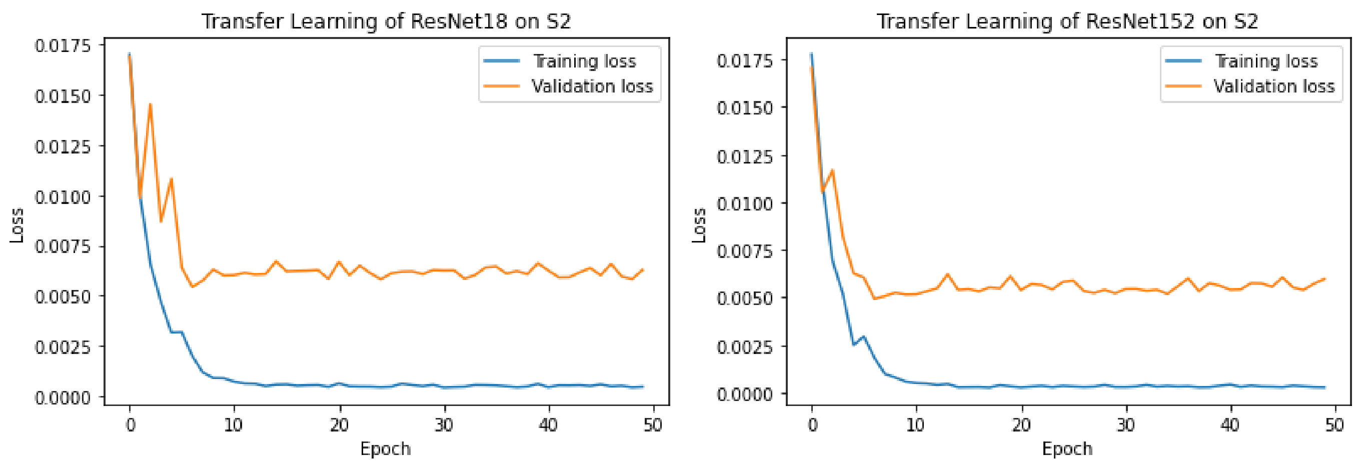


Figure 6: Transfer learning on image indices in S2 data cropped and padded to bounding boxes of regions of interest, on ResNet18 in left panel and ResNet152 in right panel.

	ResNet18				ResNet152			
	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Clear	0.98	0.90	0.94	156	0.95	0.90	0.92	129
Need Clearing	0.93	0.99	0.96	216	0.94	0.97	0.96	210
Accuracy			0.95	372			0.94	339
Macro avg	0.95	0.94	0.95	372	0.95	0.94	0.94	339
Weighted avg	0.95	0.95	0.95	372	0.94	0.94	0.94	339

Table 2: Classification report for ResNet18 and ResNet152 model transfer trained on the test set of the S2 data. The support is the number of samples that the result is based on.

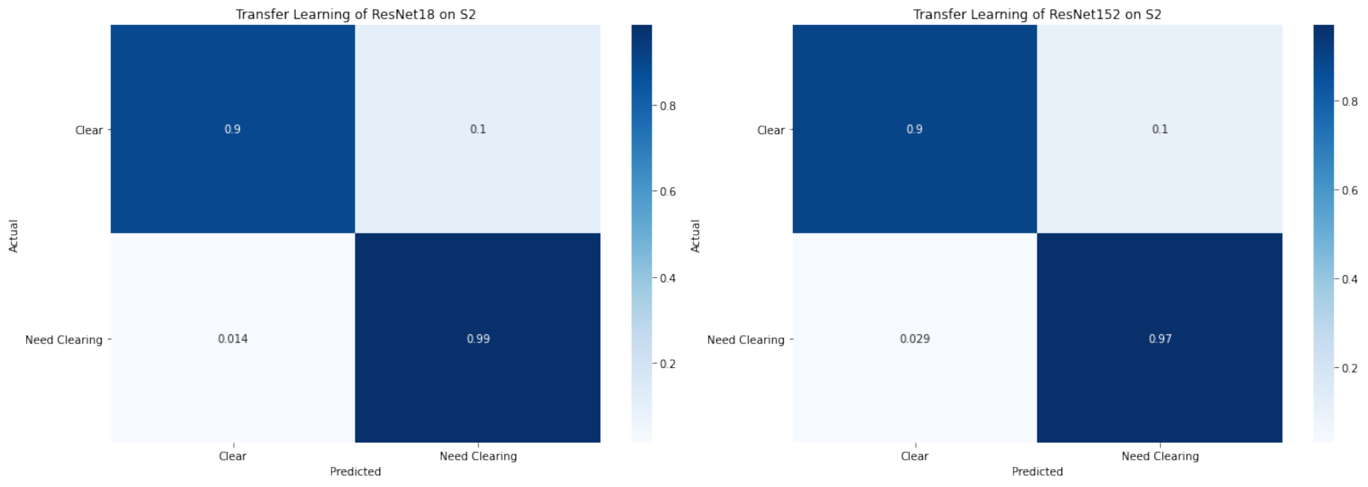


Figure 7: Confusion matrix for ResNet18 (left) and ResNet152 (right) evaluated on the test set of the S2 data. The presented values have been normalized to the total number of true labels, i.e. by each row in the matrix.

Discussion

These results of this approach are promising in terms of the high accuracy of the models. It should be noted however that the accuracy measure can be misleading because of the class imbalance. There is also the possibility that the similarity between test samples and training/validation samples is to high when splitting the data by dates, as was done here. This could also be an explanation for the high accuracy, namely that the models can simply find a very similar image of a given region in the training data when predicting on a test sample and therefore make a good prediction. In the second approach to the given task another way of splitting the data is explored. However this method was mainly used as an initial test of the data and will not ultimately be used since more detailed classification is sought for. The main problem with simple image classification for this task is that regions in need of clearing and not in need of clearing can be intertwined, which this method cannot handle.

Semantic Segmentation

For the purpose of achieving a finer granulation of classification, including the degree to which clearing is needed, semantic segmentation was attempted next. In this approach transfer learning of a convolutional network is applied to cropped out parts of the same satellite images as used before, but fitted to ground truth masks created from GPS data points collected during previous clearing work.

Background

Foran has produced GPS data points from phones carried by workers carrying out clearing of a number of clearing object, i.e. regions of interest. The time steps of these data points varies but in general the density of the points give an approximation to the degree of need for clearing in a certain spot of the clearing object.

A model which can be used for semantic segmentation of images is the convolutional network DeepLabV3, introduced in the paper *Rethinking Atrous Convolution for Semantic Image Segmentation* [2]. This model is available in the semantic segmentation part of the PyTorch package torchvision.models. In the implementation a pre-trained version can be loaded, which has been trained on a subset of the COCO train2017 dataset. Two versions of DeepLabV3 are also available one with a ResNet50 backend and one with a ResNet101 backend. In the paper it's stated that context and global features are beneficial when classifying individual pixels in semantic segmentation. Pyramid image processing is a typical way to deal with this, namely to repeatedly use smoothing and subsampling to decrease the size of the input. This can also be done multiple times in parallel, downsampling to various sizes and then combining the results. Atrous convolution, also known as dilated convolutions, allows for extracting denser feature maps by removing the downsampling operations from the last few layers and upsampling the corresponding filter kernels, equivalent to inserting holes between filter weights. With Atrous convolution, it's possible to control the resolution at which feature responses are computed within deep convolutional neural networks without having to learn extra parameters. Atrous convolution can be done with varying rates and running parallel ones with different rates is called Atrous Spatial Pyramid Pooling (ASPP) which is used as a building block in the DeepLabV3 network working as a context module.

Method

Similar to the image classification method, this approach consists of a preprocessing step and a transfer learning process.Again, two IPython Notebooks have been used, one for the preprocessing, `RojSat_SemanticSegmentation_Preprocessing.ipynb`, and one for the training process, `RojSat_SemanticSegmentation.ipynb`. The final models were also tested on some additional inputs, which where prepared using a third Notebook, `RojSat_SemanticSegmentation_ExtraTestPrep.ipynb`.

Preprocessing GPS data, as described above, was available for nine clearing objects inside the area of the satellite data used for the image classification task. Bounding boxes for these nine regions were manually found and used to crop and pad input images in the same way as for the image classification task. Again the images were first cropped to their own bounding box and then padded with zeros to reach the largest size of all bounding boxes in each dimension. For this method the TCI and S2 data was combined such that the first three channels of the input samples were the RGB features from TCI and the next 10 channels were the S2 image index features. Samples of missing data or almost entirely filled with clouds were filtered and never used as input to the model.

Ground truth masks were first generated for each region of interest by creating a heatmap of the number of GPS points in each 10×10 meter area corresponding to a pixel in the images. The 97th percentile of all these heatmap values for all regions were then found according to the histogram in Figure 8. It was found to be 45 and this was used as an upper threshold for the heatmaps since too many points in one region can likely be due to the worker taking a break or having some issues. Lastly, the values in the heatmaps were normalized to $[0, 1]$, where 0 means no clearing need and 1 is the highest need for clearing. The resulting heatmaps are visualized in Figures 9 and 10. Note that ROI 4 is missing because it was found to not be a clearing object.

As a last step to fine tune the ground truth masks, the cloud related classes in the SCL data for each input sample was used. Pixels classified as “CLOUD_MEDIUM_PROBABILITY”, “CLOUD_HIGH_PROBABILITY” or “THIN_CIRRUS” were marked with a special index, -1, referred to as the “ignore index label”, used by the loss function in the training process, see below. The ignore index label was also used to pad the ground truth images to the common size used for the inputs.

This preprocessing step resulted in 126 input samples, with individual ground truth masks, from the period July to October 2019 over the nine regions of interest. The common size of each sample image came to be $(67, 70)$.

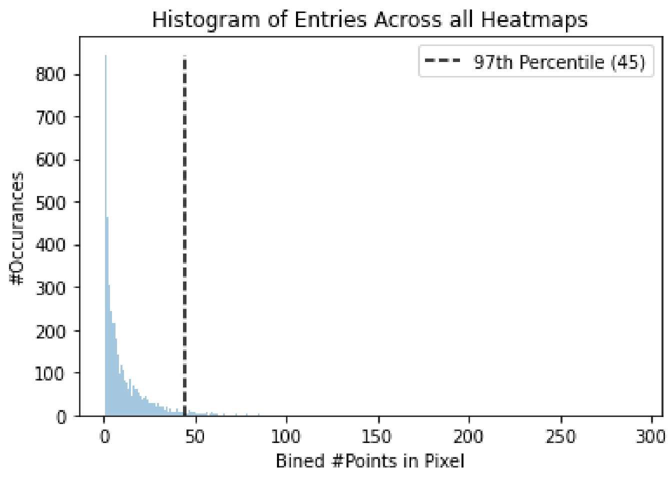


Figure 8: Histogram of heatmap entries, used to find the 97th percentile, 45, used to threshold the values before normalization.

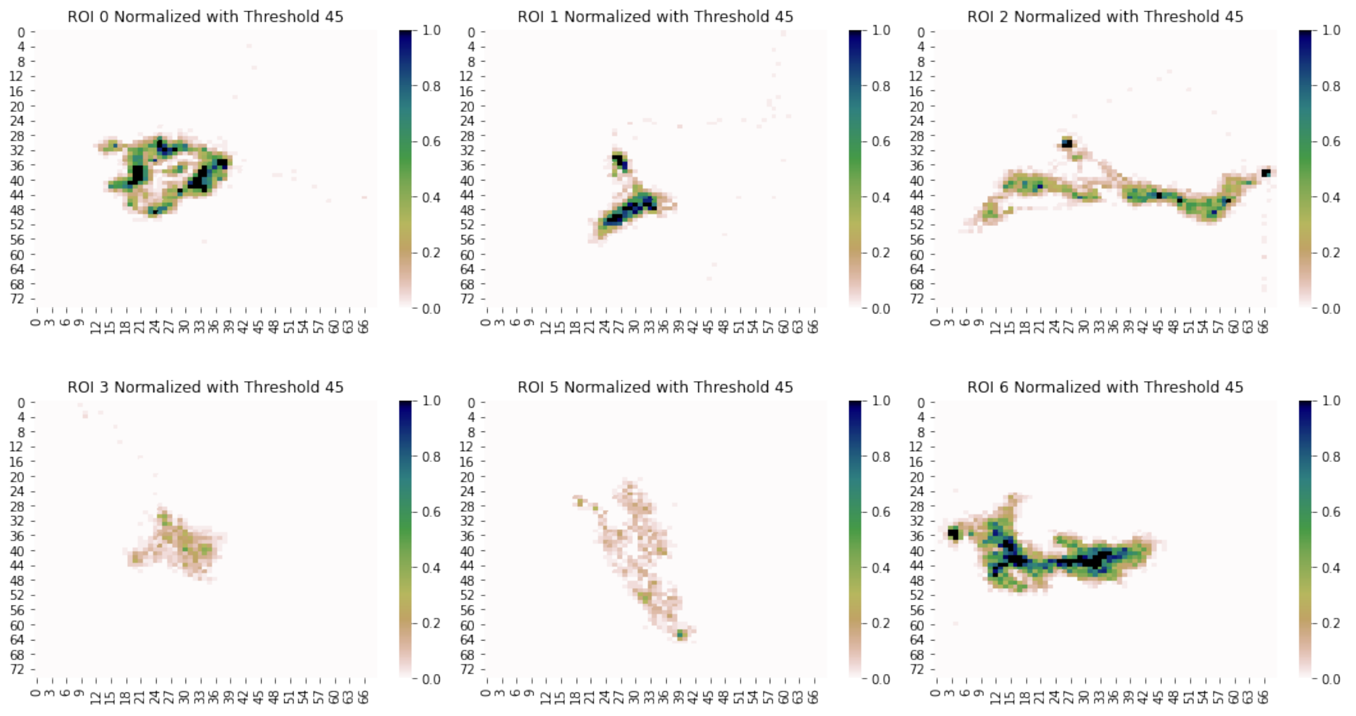


Figure 9: Heatmap over densities of GPS data points from clearing work in each of the regions of interest. All values above 45, the 97th percentile, have been set to 45 after which all values have been normalized to $[0, 1]$. ROI 4 was found to not be a clearing object and has been omitted.

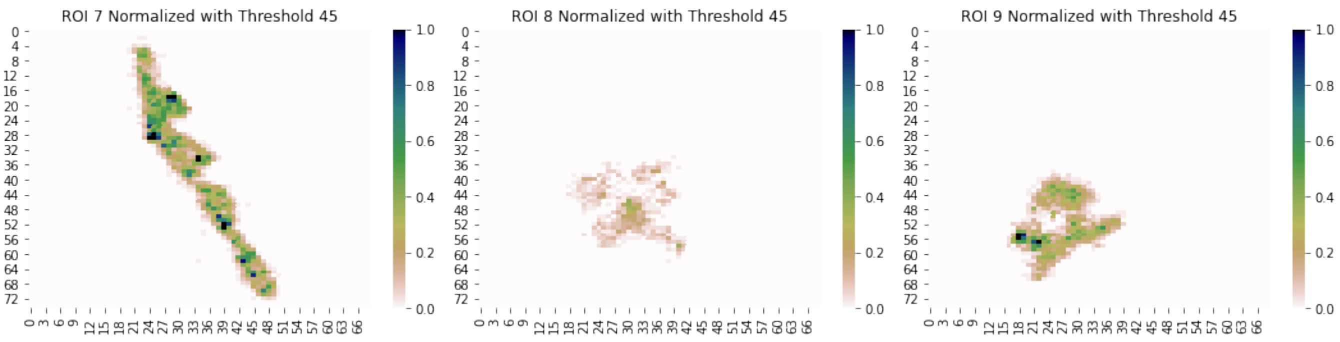


Figure 10: Continuation of Figure 9.

Transfer Learning The 126 input samples generated by the preprocessing step were split into a training set (around 80%), validation set (around 10%) and test set (around 10%) in two different ways. First, the method used in the image classification task was used, where the last few dates were used for validation and testing for all regions. The second method used all samples of ROI 5 as validation data and all samples of one of the regions ROI 6, ROI 7 and ROI 8 for testing. These particular regions were used because they had the best amount of samples to approximately match 10% of the full dataset.

For each of the two data split approaches transfer learning was applied to the DeepLabV3 convolutional network with a ResNet101 backend adapted to the correct number of inputs, 13, and a single output channel. The prior was done in the same way as for the image classification task, but the later was instead done using the line below.

```
model.classifier[4] = torch.nn.Conv2d(in_channels=256, out_channels=n_classes,
                                      kernel_size=1, stride=1)
```

A dataloader was implemented such that training samples were randomly flipped horizontally as data augmentation, for the purpose of increasing the variety of the training set. In case the augmentation was done, it was done on both the image and the ground truth mask a like. Mean square error (MSE) loss was used to train the model but an extension of the built-in function had to be implemented in order for the training to be performed on only the pixels which had not been labeled as "ignore index". This was done by setting the indexes with the ignore label in the mask to zero in both the mask and the input image and then calculating the MSE with the "sum" option and dividing by the number of elements which should not be ignored. Finally, the training process was optimized using torch.optim.Adam with learning rate 0.0001.

Apart from testing the resulting models on the respective test sets, they were tested on a number of other input images as well. This included images of regions known to be either in need of clearing or not but also some random, non-forest regions such as a residential area, a sports arena and part of a cropland.

Results

In this section the loss for the transfer learning of DeepLabV3 adapted to the combined set of TCI and S2 data, is presented for the two data split approaches together with some sample outputs side by side with their respective inputs and masks. Whenever sample images are shown they have been cropped back down to their original size, what was passed through the networks was still the full sized padded images.

Data Split by Dates Figure 11 shows the MSE loss, adapted to the ignore index label, for the transfer learning process of the DeepLabV3 network with ResNet101 backend on the combined dataset of TCI and S2. In the side-by-side view of Figures 12 and 13 the models predicting segmentation on few different regions can be compare between when they are done on a sample from the test set and on a sample from the training set, i.e. on the same region but from different dates. It is clear that the samples are to similar between the test and training sets and thus that the model has simply used what it learned on the training set and duplicated that on the test set.

Additional sample predictions are shown in Figures 15, 14 and 16. For these samples no ground truth degree of need for clearing is available but the first two Figures include samples of regions know to be in need of clearing in general and not in need of clearing at all. The last Figure shows some random, non-forest regions which of course are not in need of clearing. The predictions are in general not very accurate further corroborating the theory that the model has fitted to the training data but not learned any general trends that apply to general inputs.

Throughout the results some spots in each predicted segmentation, often close to the edges of the images, get completely misclassified with high values (dark colors) supposedly indicating a great need for clearing. It is unclear why these spots occur and what they mean.

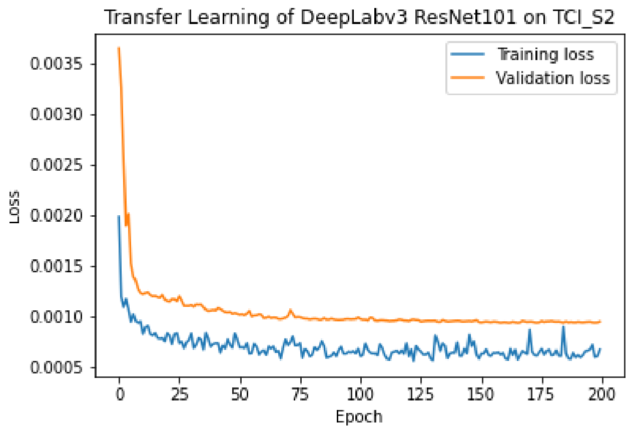


Figure 11: MSE loss with ignore index from transfer learning of DeepLabV3 on the combined set of TCI and S2 data.

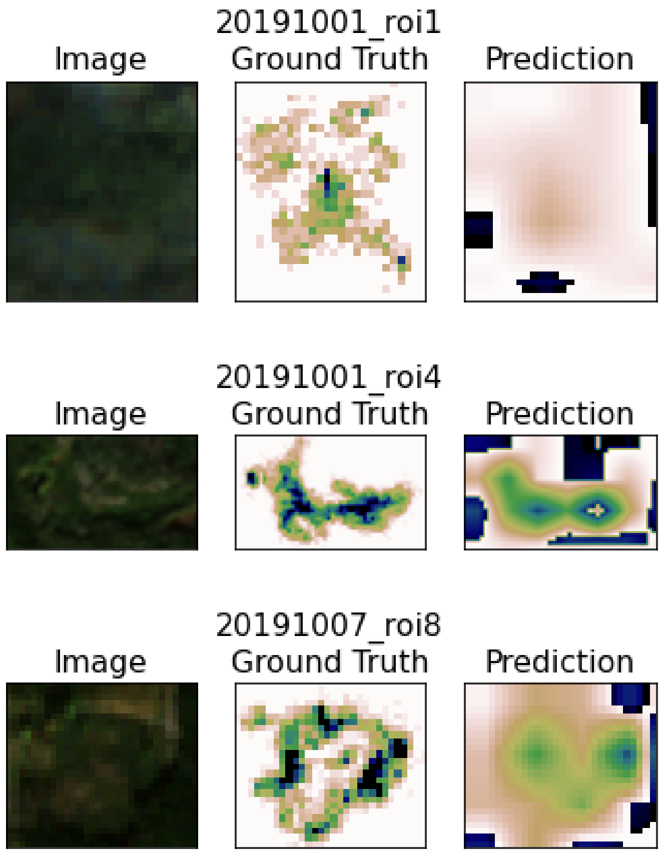


Figure 12: Sample of results, including predicted segmentation images, by the DeepLabV3 network on samples of three different regions of interest, ROI, from the test set. Thus, the model did not see these exact samples during the training process.

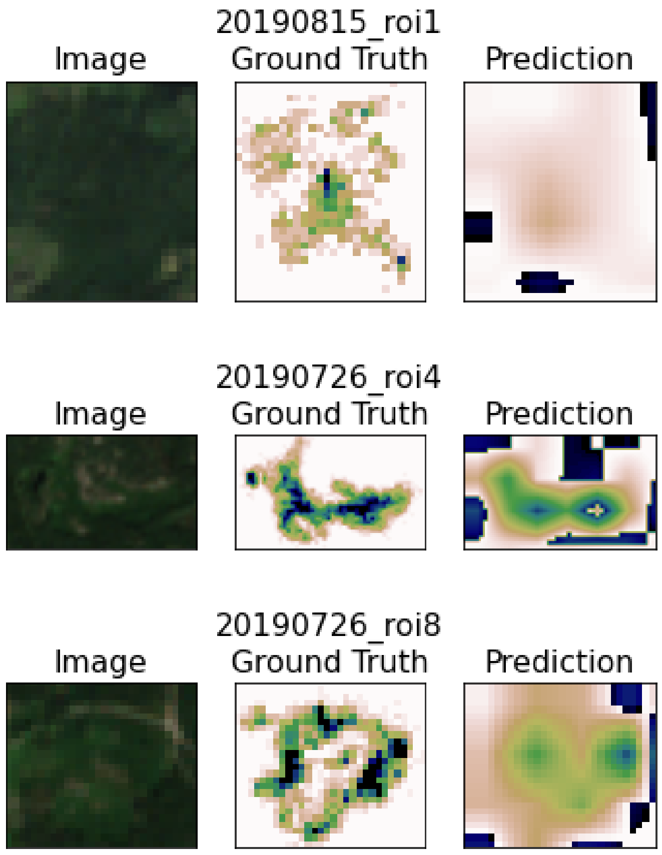


Figure 13: Results including predicted segmentation images by the DeepLabV3 network on samples of the same three regions of interest, ROI, as in Figure 12 but here from the training set. Meaning, samples from different dates than the ones shown to the right. The model was trained on these exact samples.

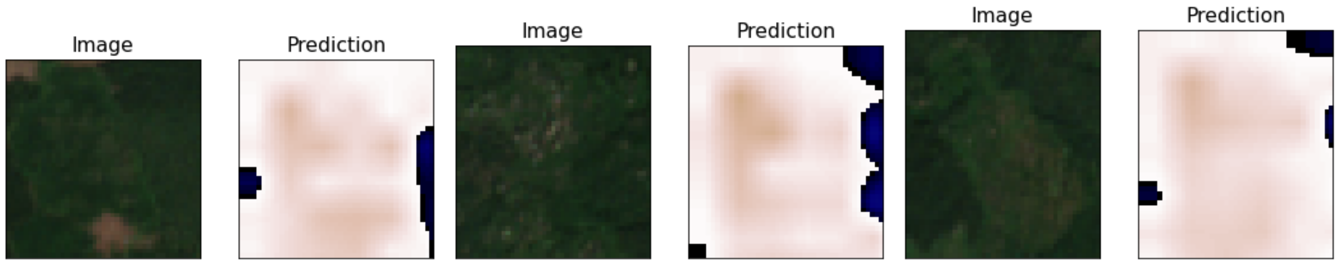


Figure 14: Sample of results, including predicted segmentation images, of areas known to be in need of clearing, by the DeepLabV3 network trained on the combined set of TCI and S2 data.

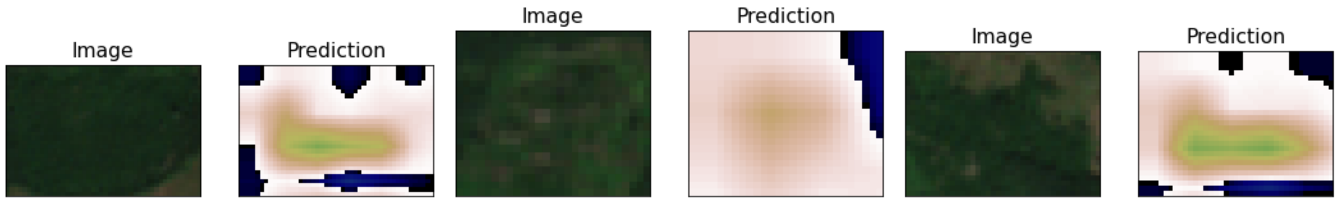


Figure 15: Sample of results, including predicted segmentation images, of areas known to not be in need of clearing, by the DeepLabV3 network trained on the combined set of TCI and S2 data.

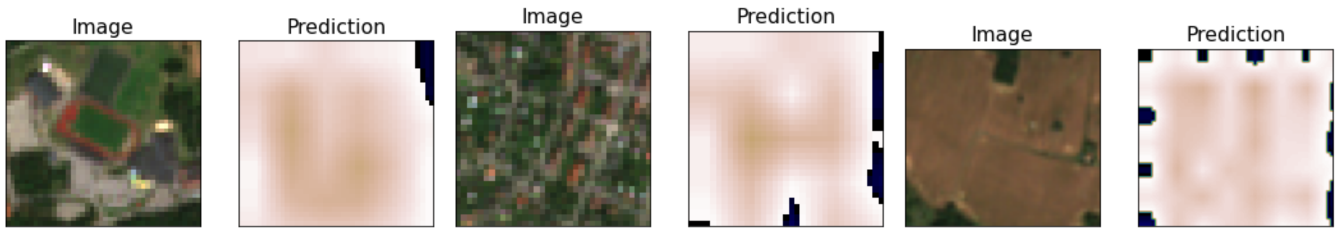


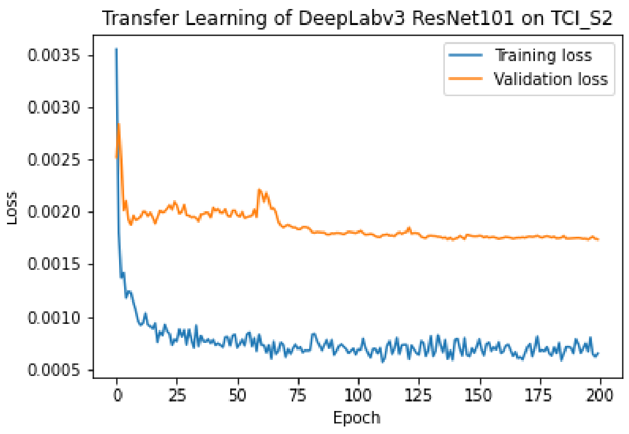
Figure 16: Sample of results, including predicted segmentation images, of some random, non-forest regions not in need of clearing, by the DeepLabV3 network trained on the combined set of TCI and S2 data.

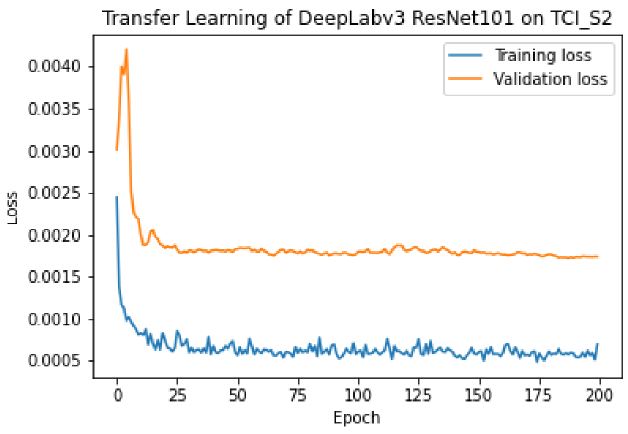
Data Split by Regions of Interest Figure 17 shows the MSE loss, adapted to the ignore index label, for three transfer learning processes of the DeepLabV3 network with ResNet101 backend on versions of the combined TCI and S2 data. The difference between the training processes is which of the regions 6, 7 and 8 was kept separated from the training process, i.e. which of the regions made up the test set.

In Figures 18 and 19 test and training results are compare, similarly as was done in Figures 12 and 13. However here the test results (left) show predictions done by a model trained on all other region apart from the one being shown and the training results (right) shows the same region and same date but from a model which was trained on that specific region.

Again, additional sample prediction are shown as well, in Figures 20, 21 and 22. The same regions used to test the previous data split method are used here, i.e. the first two Figures show areas known to be in need of clearing and not be in need of clearing respectively and the last figure shows some random non-forest regions not in need of clearing.

The results on training data have some accuracy to them but they are by no means perfect. The test results on the other hand are not very good at all. They indicate a need for clearing in somewhat the right spots of the images but by looking at the additional test predictions, where there should be no need for clearing at all, and seeing that the model still predicts spots of need for clearing it is clear that the model have not really learned the given task.





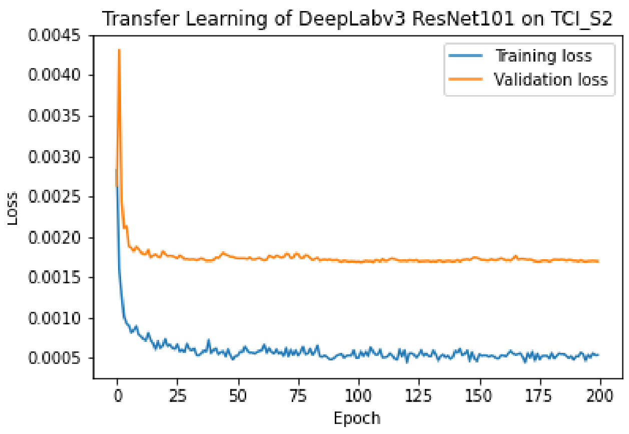


Figure 17: MSE loss without counting ignore indexes for transfer learning, where ROI 5 was used for validation and all other regions except one was used for training. Upper left panel shows results for when ROI 6 was kept separate for testing, upper right panel shows results for when ROI 7 was kept separate for testing and lower panel shows the results for when ROI 8 was kept for testing.

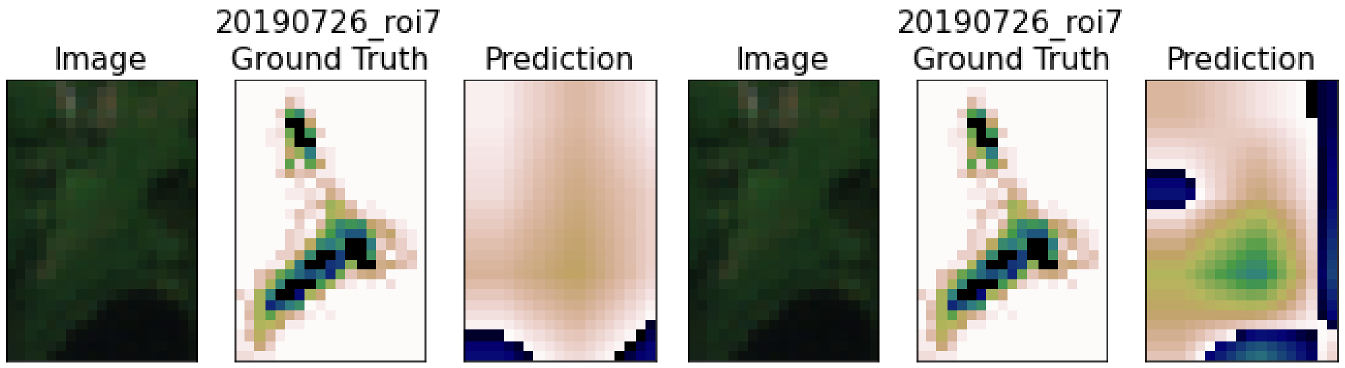


Figure 18: Sample of results, including predicted segmentation images, by the DeepLabV3 network on the three regions 6, 7 and 8. The prediction to the left is done by a network trained on all region apart from that region, i.e. the given region was part of the test set for that model. The prediction to the right on the other hand is done by a different network where the region was included in the training set.

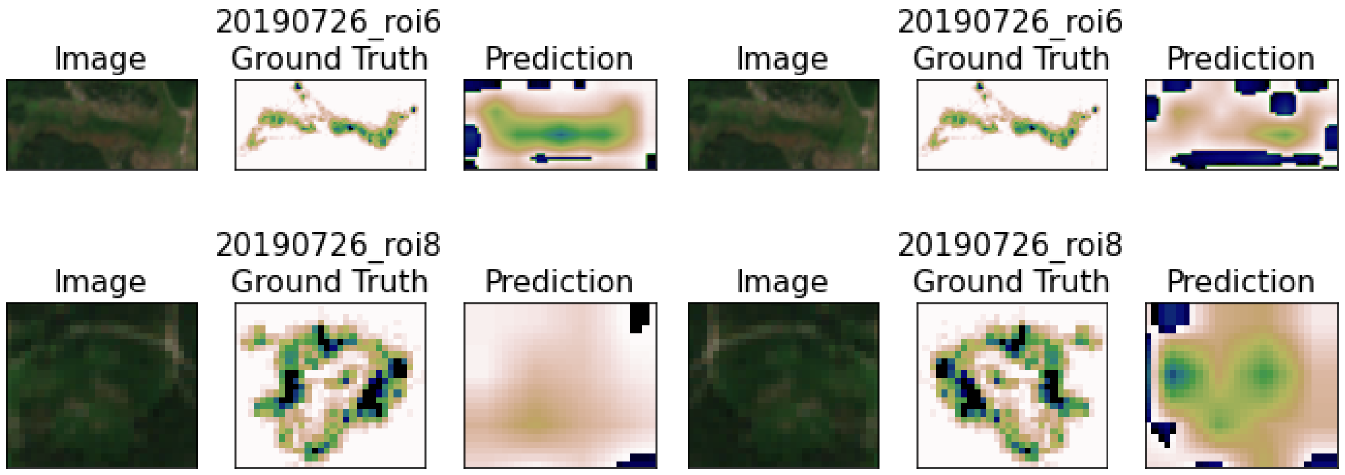


Figure 19: Continuation of Figure 18.

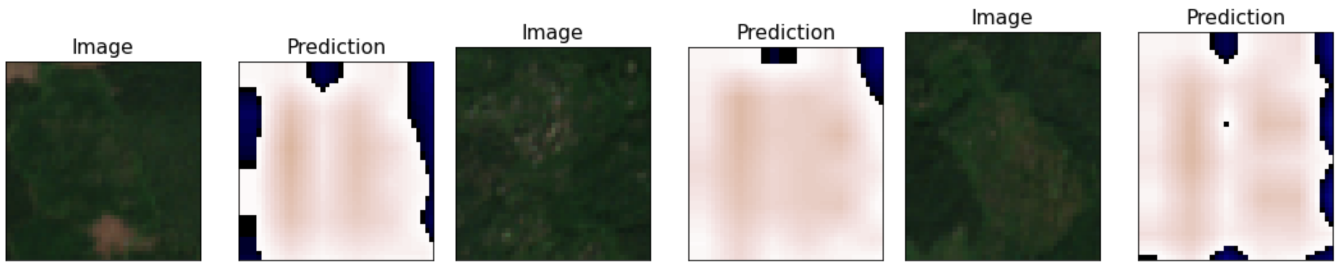


Figure 20: Sample of results, including predicted segmentation images, of areas known to be in need of clearing, by the DeepLabV3 network trained on the combined set of TCI and S2 data with ROI 8 in the test set.

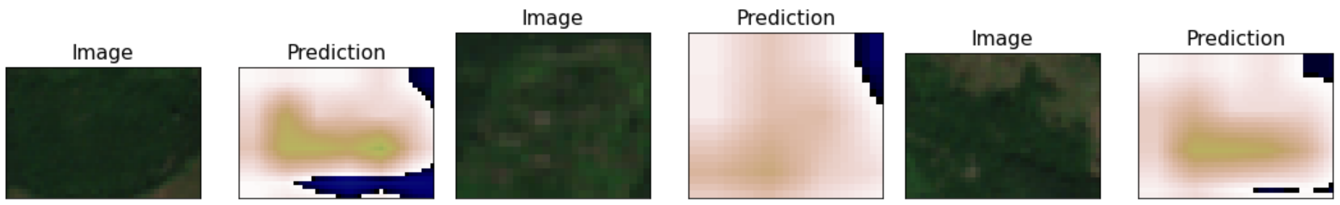


Figure 21: Sample of results, including predicted segmentation images, of areas known to not be in need of clearing, by the DeepLabV3 network trained on the combined set of TCI and S2 data with ROI 8 in the test set.

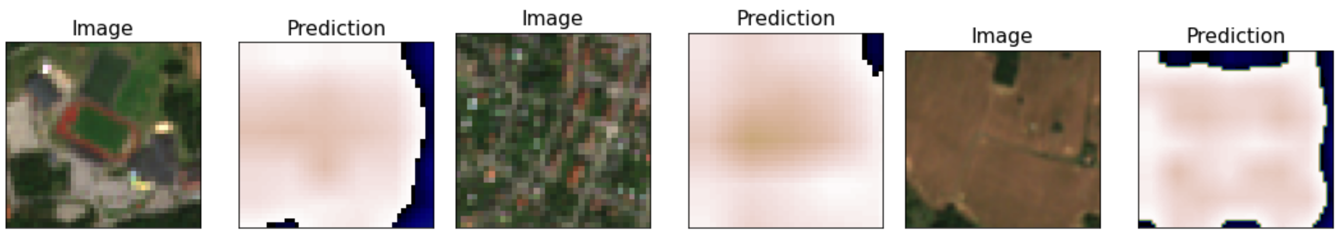


Figure 22: Sample of results, including predicted segmentation images, of some random, non-forest regions not in need of clearing, by the DeepLabV3 network trained on the combined set of TCI and S2 data with ROI 8 in the test set.

Discussion

The semantic segmentation results show some potential for being able to learn the task at hand, seen in the predictions on training data where the models to some extent picked out the spots with a need for clearing. These result were however not very precise, the non-zero regions were not a perfect match for the non-zero regions in the ground truth. Regarding the degree of need for clearing, i.e. how high the non-zero values were, they correlated even less with the ground truth. The models seemed to mostly increase the values the further into a given patch you went. Note also that these odd, misclassified, spots of very high values occurred in the training predictions as well.

When it came to all predictions on, by the model, unseen data there was a clear trend of not very accurate results. Regions predicted to be in need of clearing was mostly random and when predicting images with no need for clearing at all the models still found patches with a great need for clearing. Since the models seem to be able to learn something about the training data, the poor test results might simply be because of the overall small dataset used.

It should be noted that discrete ground truth masks were also explored during this project, by setting all non-zero values to class "in need of clearing" and all zeros to class "not in need of clearing". This was omitted from this report since the results achieved with this method were worse than the ones for continuous ground truth masks. For this method two output channels were used in the network, i.e. one hot encoded predictions, and cross entropy loss was used during the transfer learning process.

Future Work

Semantic segmentation should be explore further but with substantially larger datasets, possibly with data from all year around, not just from the summer months. If it is of interest that the model should be able to classify images with no need for clearing, perhaps some images like that should also be included in the training data. The GPS data used to generate the ground truth masks is available for a much larger area than the one used here, including many more clearing objects, and thus by downloading more satellite data the size of the training dataset can be increase easily. If a larger dataset still isn't enough perhaps some more augmentation could be added to the images.

Another possible next step to improve the segmentation predictions could be to append the raw satellite data in the bands of BOA and TOA as additional features to the inputs. This could give the models a better chance of learning something valuable about the given task. Other convolutional networks might also be explored and compared to the results of the DeepLabV3 network.

MSE is a quite general loss function and not necessarily the best choice for this task. The intercept over union (IoU) measure is know to handle the overlap of regions better. This measure could be used to evaluation the segmentation prediction by looking at the discrete case of zeros vs non-zero values after the fact. It is not trivial how the IoU measure could be used directly on continuous labels, but if a good way to do this can be found perhaps it would also be of interest to use this measure as loss function during the training process. This has been done for binary image segmentation, described in the article "Deep Neural Networks with Intersection over Union Loss for Binary Image Segmentation" [3].

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[2] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, vol. abs/1706.05587, 2017. [Online]. Available: http://arxiv.org/abs/1706.05587

[3] F. van Beers, A. Lindström, E. Okafor, and M. A. Wiering, "Deep neural networks with intersection over union loss for binary image segmentation." in *ICPRAM*, 2019, pp. 438–445.