Classifying Greenland Landscape From Satellite Imagery

Machine Learning Project 2

John Hausberg Anfindsen, Rasmus Moorits Veski, Andrea Giacobbi
School of Computer and Communication Sciences, EPFL, Switzerland

***Abstract*—Landscapes change between seasons and throughout years. To not have to manually track the development of all plots of land, there is a need for automatic detection of landscapes from satellite images. In this project we explore the capabilities of segmentation models on the task of pixel-level classification of satellite images. Out of our tested models we find that U-Net with Hypercolumn has the best accuracy and IoU score.**

I. Introduction

Greenland has experienced significant transformations in land cover over recent decades, driven by climate change. These shifts include the retreat of ice cover, expansion of vegetation, and alterations in wetlands, meltwater, and sediment coverage. These modifications are influenced by climate variations and associated geomorphological activity, with implications for surface energy balance, greenhouse gas dynamics, and landscape stability [1]. This project aims to explore deep learning models for semantic segmentation to create segmented maps of Greenland's land cover. These maps can help analyze and understand the ongoing transformations in land cover using specific classification categories.

II. Data and Goal

For our task, our data is 7-channel images of 128x128 pixels. Our goal is to classify each pixel in that image into one of 7 classes: Bad data, Snow and Ice, Wet ice and meltwater, Freshwater, Sediment, Bedrock, and Vegetation. The images and their corresponding labels can be seen in Figure 1.

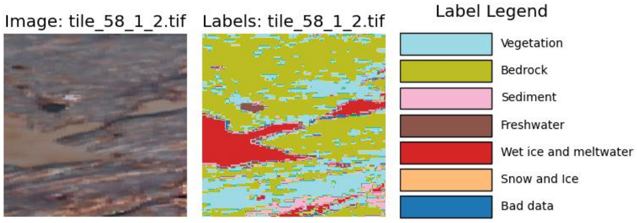

Fig. 1: Example of an image and its corresponding labels (only RGB channels plotted for image)

We need to predict what landscape each pixel in 500 images taken in 2023 would be. To achieve this we have 1500 images each from the years 2014, 2015 and 2016. For this data we have a joint 1500 labels, meaning that the images correspond to the same locations from year to year, sharing the labels. This means the labels cannot be 100% accurate for all years, as naturally conditions change.

It is important to note that the labels we use for training were also extracted with a model. Hence we are not truly modelling ground truth as much as we are modelling the predictions of this model, which for us functions as the black-box ground truth.

A challenge in this task is the time difference of the images taken. As the climate changes so do the conditions in Greenland. Naturally, this means a different distribution of landscapes in 2023 than what we have in our training data. The landscape distributions of the train and testing set are displayed in Table I.

Label	Train set	Test set
Bad data	10.08%	11.6%
Snow and Ice	**25.45**%	18.86%
Wet ice and meltwater	4.68%	4.74%
Freshwater	5.29%	7.51%
Sediment	5.08%	2.8%
Bedrock	**29.66**%	8.35%
Vegetation	19.75%	**46.14**%

TABLE I: Proportion of labels in training and test data

As seen in Table I, the proportion of area held by a certain landscape type varies between training and testing data. Notably, there is much more snow and bedrock in the training data, which is largely replaced by the vegetation making up nearly half of the landscape in 2023. Note that this information was **not** used in our training process, but can be used to explain the performance of our models on certain labels.

III. Models and Methods

For this task, we employed three machine learning models: **U-Net**, **DeepLab**, and **U-Net with Hypercolumn**. Each model was trained on three different datasets: data from the year closest to the classification target year (2016 for 2023), the same data augmented with transformations and filters, and a combined dataset containing data from all available years. To evaluate the performance of these models, we used **accuracy** and **Intersection over Union (IoU)** as metrics. Accuracy measures the overall percentage of correctly classified pixels, while IoU provides a more detailed assessment by calculating the overlap between predicted and ground-truth segments, making it particularly valuable for segmentation tasks.

Data Augmentation

Data augmentation is widely used to improve the performance of deep learning models [2]. We utilized two data augmentation techniques for our training data: Using a Gaussian filter, and using simple rotations of the images and labels.

For our training, we combined these two techniques to produce one of our three training datasets, consisting of images from 2016, the same images with Gaussian filter applied, and the same images rotated 90, 180, and 270 degrees. For the Gaussian-filtered data we simply copied over the labels as they were unchanged. For the rotated images we also rotated the labels. This produced a much larger dataset, intended to produce a model which generalizes better.

U-Net

Originally proposed by Ronneberger et al. [3], the **U-Net** structure is a symmetric encoder-decoder architecture designed for segmentation tasks. The encoder extracts hierarchical features using convolutional layers and downsampling operations, while the decoder reconstructs the spatial resolution using transposed convolutions and skip connections. Skip connections link corresponding layers in the encoder and decoder to preserve spatial information, enabling precise segmentation. For this implementation, we used **ResNet-34** as the backbone for the encoder, as it is tailored for image classification tasks.

DeepLab

Originally proposed by Chen et al. [4], the **DeepLab** model is a deep learning architecture designed for semantic segmentation, notable for its use of atrous (dilated) convolutions. Atrous convolutions allow DeepLab to capture features at multiple scales by controlling the receptive field, enhancing the model's ability to segment objects of varying sizes without increasing computational costs.

For this implementation, we used **DeepLabv3** with **ResNet-50** as the backbone for the encoder.

U-Net with Hypercolumn

The **U-Net with Hypercolumn** builds upon the standard U-Net architecture by incorporating the **hypercolumn technique**, a method originally proposed by Hariharan et al.[5] for segmentation tasks. This technique combines multi-scale features from different depths of the decoder, aggregating both low-level spatial details and high-level semantic information to create a richer pixel-wise representation. For this work, we implemented the model as proposed by Karchevskiy et al.[6], adapting it to handle our input data. The multi-scale outputs are concatenated and processed through a final convolutional layer to generate pixel-level predictions, offering a significant refinement over the standard U-Net, which uses only the last decoder layer without explicitly combining features across scales.

In addition to the hypercolumn technique, the model integrates **deep supervision**, where intermediate outputs from multiple decoder stages are combined to enhance gradient flow during training. These multi-scale outputs are concatenated and passed through a final convolutional layer to produce pixel-level predictions. This approach contrasts with the standard U-Net, which directly uses the final decoder layer for predictions without explicitly combining features from multiple scales.

For this implementation, we used **SE-ResNeXt50** as the backbone for the encoder, leveraging its advanced **squeeze-and-excitation (SE)** blocks to enhance channel-wise feature representation.

Table II summarizes the parameters used for each model.

Parameter	**U-Net**	**DeepLab**	**Hypercolumn**
Loss function	Cross-Entropy	Cross-Entropy	Cross-Entropy
Optimizer	Adam	Adam	Adam
Activation function	softmax	softmax	softmax
Learning rate	10^{-4}	10^{-4}	10^{-3}
Batch size	20	16	16
Epochs	40	30	20
Library	TensorFlow	PyTorch	PyTorch

TABLE II: Model Parameters

For all models, the activation function for the final layer was **softmax**, enabling the models to output class probabilities for each pixel, which is essential for pixel-wise classification tasks. The **cross-entropy loss function** was employed, as it is well-suited for multi-class segmentation problems.

Additionally, with U-Net, we experimented with using a loss inspired by IoU-based loss functions. In rough terms, our loss function divided the number of true positives in each class by the number of total positives (both predicted and in true values) in that class. Then it took the average of all classes and subtracted it from 1. We hoped this would lead to higher mean IoU, as it gives equal weight to the classification accuracy of all classes.

The **batch size** and **epochs** were determined based on the performance capabilities of our hardware, ensuring the models could run efficiently while achieving optimal performance. Additionally, the backbones for all models were modified to accept **7-channel input data** instead of the default 3-channel RGB, making them compatible with our multi-spectral dataset. None of the models used pre-trained weights.

IV. Results

The accuracies and mean IoU scores for all our trained models can be found in Table III.

Model architecture	Training data	Accuracy	Mean IoU
U-Net	2016 data	0.798	0.447
	2016 data augmented	0.793	0.439
	all years data	0.81	0.492
U-Net soft IoU loss	all years data	0.796	0.455
DeepLab	2016 data	0.798	0.349
	2016 data augmented	0.792	0.330
	all years data	0.803	0.353
Hypercolumn	2016 data	0.803	0.466
	2016 data augmented	0.804	0.462
	all years data	**0.829**	**0.526**

TABLE III: Model performances on test set

While most models had a comparable accuracy, the mean IoU metric varied more, with the Hypercolumn architecture having the best performance. This model achieved an accuracy of 83% and a mean IoU of 0.526.

It can also be seen in Table III, that using different data splits and augmentation had little effect on the final performance. Data augmentation sometimes even worsened the performance. This means that a similarly-performing model could likely be reached with using even less data.

U-Net

U-Net was our second-best performing model. When training with categorical crossentropy on all data as seen in Figure 2, the training process only increased its accuracy, while changes in the IoU were so minimal that they are not visible on the plot. However, the mean IoU on test data was significantly higher than on training and validation data.

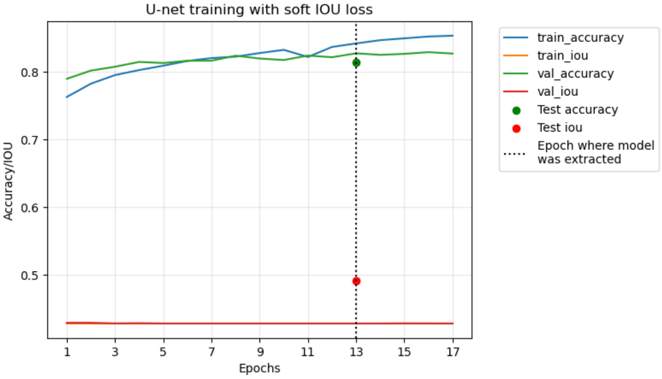

Fig. 2: Training of U-Net on all Data

This is bizarre, as when comparing it to the training process of U-Net using the aforementioned loss that optimises for IoU as seen in Figure 3, the IoU makes major strides in training, but when it comes to evaluation, it drops. This is possibly indicative of the model overfitting to the training distribution as in Table I and/or categorical crossentropy being a better loss function for generalisability.

DeepLab

DeepLab had a similar performance in accuracy to U-Net. However, its mean IoU was worse than other models. Figure 4 depicts the individual IoU scores of the classes through the training of DeepLab on all data.

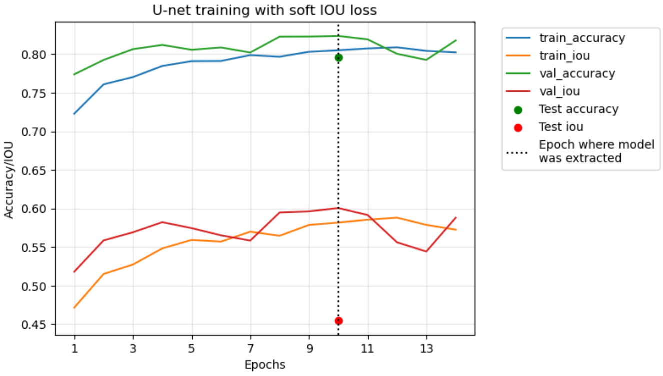

Fig. 3: Training of U-Net on all data with IoU loss

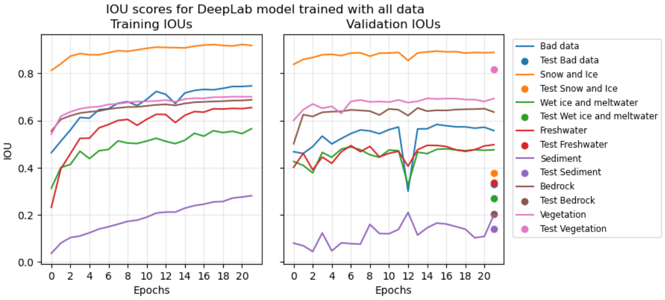

Fig. 4: Deeplab class training and validation IoU scores through epochs (lines) and the trained model testing IoU (dots).

As seen in Figure 4, all the IoU scores for the training data and validation data increase or stay at a high level throughout training. However, on the testing set all IoU scores except vegetation fall drastically down. This is likely due to the training and validation data coming from the same distribution and the model not being able to generalise to the test set.

U-Net with HyperColumn

U-Net with Hypercolumn was our best-performing model, both in terms of accuracy and mean IoU. It was better than the other models for every subset of data we trained on. As seen on Figure 5, the testing accuracy is almost on-par with the training accuracy, indicating good generalisation on unseen data, even from different distributions.

We still, however, see a large difference in IoU scores between different classes, as with Deeplab, as seen in Table IV.

Class	IoU Value
Bad data	0.574
Snow and Ice	**0.891**
Wet ice and meltwater	0.288
Freshwater	0.436
Sediment	0.209
Bedrock	0.404
Vegetation	**0.879**

TABLE IV: IoU Values of Hypercolumn for each class, on test data

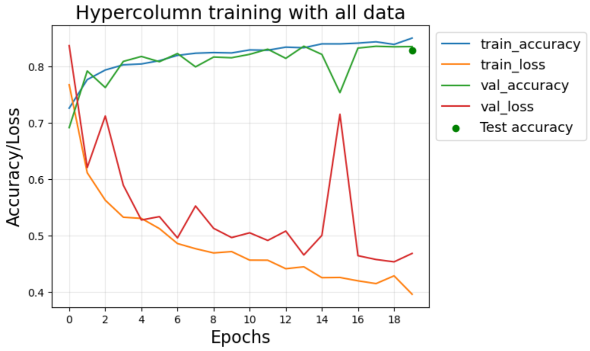

Fig. 5: Training of Hypercolumn on all data

V. Discussion

Our models achieved good performance metrics overall, with an 82.9% accuracy and 0.526 mean IoU for our best model. We consider this a good result, especially considering the difference between training data and test data distributions.

Unexpected IoU Score

There are notable aspects of the IoU scores worth discussing. Our best model instantiation, U-Net with Hypercolumn trained on the full training dataset, achieved a mean IoU of 0.526, with class IoU values (see table IV) ranging from 0.891 (Snow and Ice) to 0.209 (Sediment). In figures 6 and 7, we examine actual model outputs on test data compared with the target labels: We note that the model generally seems to find the class boundaries somewhat accurately, but struggles occasionally with class assignment for certain regions. We also see a preference for certain classes, which is expected due to the class imbalance and difference between train data and test data. As seen in Table I bedrock, for example, is far more common in the training data vs the test data, comprising 29.66% of the training data and 8.35% of the test data. This difference reflects a substantial change in the real-world data from 2016 to 2023, which is expected. We see this in effect in Figure 7, where the model predicts bedrock and snow and ice, where the 2023 target label is actually wet ice and meltwater.

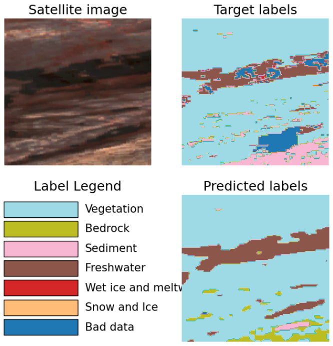

Fig. 6: Predicted labels vs. target labels, example 1

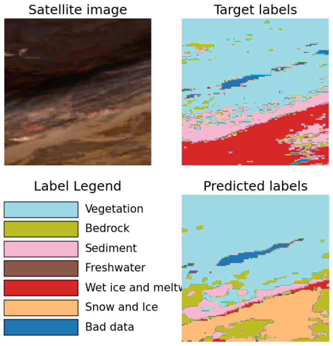

Fig. 7: Predicted labels vs. target labels, example 2

In the interesting case of Figure 6, our model predicts a consistent body of freshwater, where the target label is a seemingly uncertain mixture of bad data and freshwater. This may indicate that our model outperforms the model used to generate the target labels in this specific case, as it seems reasonable that this region is consistently water.

Data Augmentation

Interestingly, data augmentation on its own did not lead to improved results in our training. This outcome is difficult to fully explain, but it could indicate that there is in fact enough training data already, and that the current bottleneck for improved performance is at another pain point. One possible bottleneck is the uncertainty in the relationship between the training images and the labels. Our training data (as explored in Section II) consists of three sets of images from three different years, but only one set of labels. This inherently introduces uncertainty, as three slightly different image sets are mapped to identical labels.

Future research: Adapted Loss Functions

When training a neural network model, the loss function can be defined in many different ways. In this domain, it would be interesting to experiment with a weighted loss function based on the IoU scores of each class. In this manner one could prioritize high quality predictions for certain classes. For instance, this could be useful for high-precision prediction of change in snow and ice coverage in Greenland, a very interesting research topic in the context of climate change.

VI. Conclusion

In this project we assessed the capabilities of 3 semantic segmentation models on pixel-wise classification of Greenland satellite imagery. We found that U-Net with Hypercolumn has the best performance both in terms of accuracy and IoU.

VII. Ethical risk

Based on our digital ethics canvas analysis (VIII-B), we have identified two minor ethical risks. In this section, we describe the welfare risk, namely the potential for our model to be abused to maliciously target certain biomes or areas of Greenland, for example for industrial purposes that could damage the natural environment.

If this risk were to be fully exploited, the main affected stakeholders would be the inhabitants in the area. Industrial development could potentially destroy their natural environment, which is essential with regards to both physical and mental health, and general well-being [7]. Additionally, the large indigenous population on Greenland (a large majority of the population [8]) could be even more affected by this, as indigenous peoples are known to have strong ties to nature [9].

It is difficult to evaluate such a risk, as it is hard to quantify. However, based on discussion within our team and the common knowledge that there are already easily accessible models that will perform as well as, if not better than, ours, we conclude that the risk is very minor. We consider the likelihood of our project being used for this purpose to be extremely low.

We have not taken this risk into account in our project, as we are submitting the code using a GitHub repository. Considering the very minor nature of the risk, we deem this appropriate.

VIII. Appendix

A. LLM usage

LLMs were utilized to generate code to get started with different methods, debug, improve, adapt, refactor, generate a baseline requirements.txt, and general code generation. It was also used for getting ideas and suggestions, as well as general improvement for this report.

B. Digital ethics canvas

DIGITAL ETHICS CANVAS

CONTEXT

SOLUTION

BENEFITS

Automating landscape detection

WELFARE

RISK

- Can the solution be used in harmful ways, in particular with regards to vulnerable populations?
- What kind of impacts can errors from the solution have?
- What type of protection does the solution have against attacks or misuse?

-Could conceivably be used to maliciously target certain biomes, eg. by harmful industry.

-Not a major risk.

MITIGATION

-Our goal is to make an accurate model, for academic and positive purposes. We would therefore, under normal circumstances, consider not to publish the code online. However, considering the very minor nature of this risk, and the requirements for the course, we do publish the code on Github.

FAIRNESS

RISK

- How accessible is the solution?
- What kinds of biases may affect the results?
- Can the outcomes of the solution be different for different users or groups?
- Could the solution contribute to discrimination against people or groups?

-Accessible for anyone with a decent computer

-Otherwise not relevant, our project deals with satelite imagery and not directly people.

MITIGATION

-No risk.

AUTONOMY

RISK

- Can users understand how the solution works and what its limits are?
- Are users able to make choices (e.g. consent, settings) in their use of the solution and how?
- How does the solution affect user autonomy and agency?

-Not relevant.

MITIGATION

-No risk.

PRIVACY

RISK

- What data does the solution collect
- Is it collecting personal or sensitive data
- Who has access to the data?
- How is the data protected?
- Could the solution disclose / be used to disclose private information?

-Solution does not collect data.

-However, we process data.

-Possible privacy concern in the training satelite imagery, eg. for indigenous population of Greenland.

MITIGATION

-The data set is already public, we have no say in this

-No risk.

SUSTAINABILITY

RISK

- What is the carbon footprint of the solution?
- What types of resources does it consume (e.g. water) and produce (e.g. waste)?
- What type of human labor is involved?

-Consumes electricity, compute.

-Not a big concern for this project.

-No labor involved except our own

MITIGATION

-Write efficient code, avoid wasting time and resources.

References

[1] M. Grimes, J. L. Carrivick, M. W. Smith, and A. J. Comber, "Land cover changes across greenland dominated by a doubling of vegetation in three decades," *Scientific Reports*, vol. 14, p. 3120, 2024. [Online]. Available: https://doi.org/10.1038/s41598-024-52124-1

[2] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image data augmentation for deep learning: A survey," 2023. [Online]. Available: https://arxiv.org/abs/2204.08610

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[4] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," in *arXiv preprint arXiv:1706.05587*, 2017. [Online]. Available: https://arxiv.org/abs/1706.05587

[5] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," 2015. [Online]. Available: https://arxiv.org/abs/1411.5752

[6] L. K. Mikhail Karchevskiy, Insaf Ashrapov, "Automatic salt deposits segmentation: A deep learning approach," *ArXiv e-prints*, 2018. [Online]. Available: https://arxiv.org/abs/1812.01429

[7] R. M. Nejade, D. Grace, and L. R. Bowman, "What is the impact of nature on human health? a scoping review of the literature," *Journal of global health*, 2022. [Online]. Available: https://doi.org/10.7189/jogh.12.04099

[8] I. W. G. for Indigenous Affairs. (2023) The indigenous world 2023: Kalaallit nunaat (greenland). [Online]. Available: https://www.iwgia.org/en/kalaallit-nunaat-greenland/5071-iw-2023-kalaallit-nunaat.html

[9] U. o. L. Arctic Centre. Arctic indigenous peoples. [Online]. Available: https://www.arcticcentre.org/EN/arcticregion/Arctic-Indigenous-Peoples