UNIVERSITY OF OSNABRÜCK

BACHELOR THESIS

Detection of Melt Ponds on Arctic Sea Ice from Infrared Images using U-net

---

Author:
Marlena REIL

Supervisors:
Dr. Ulf KRUMNACK,
Dr. Gunnar SPREEN

A thesis submitted in fulfillment of the requirements
for the degree of Cognitive Science, B.Sc.

Submission: August 4, 2023

Abstract

Melt ponds are pools of water on Arctic sea ice that play a crucial role in the summer heat budget. Accurate quantitative analysis of melt pond distribution is essential for improving climate model predictions. High-resolution thermal infrared imagery acquired from aircraft can be used to validate low-resolution satellite products and extend existing tools to derive melt pond characteristics. This requires a method to classify the images into distinct surface types. The aim of this thesis is to develop a segmentation model to separate helicopter-borne thermal infrared images into melt pond, sea ice, and ocean classes. To address this task, we explored the application of a pre-trained U-net, a deep convolutional neural network. Time-consuming annotation limited the availability of labeled training data, which affected the generalization ability of our model and challenged robust evaluation. Our final model cannot be readily employed as a standalone solution. Nevertheless, it serves as a foundation for future improvements and can be used as a tool to annotate more data. In addition, we were able to gain insight into the effectiveness of different regularization techniques that can inform future studies in this domain. The code of this work can be accessed in our GitHub repository (`https://github.com/marlens123/ponds`).

Contents

List of Abbreviations

MPF	**M**elt **P**ond **F**raction
VIS	**VIS**ible light
TIR	**T**hermal **I**nfra**R**ed
CNN	**C**onvolutional **N**eural **N**etwork
CCE	**C**ategorical **C**ross **E**ntropy
CFDL	**C**ategorical **F**ocal **D**ice **L**oss
mIoU	**m**ean **I**ntersection **o**ver **U**nion

Chapter 1

Introduction and Context

Studies show that the Arctic is warming up to four times faster than the rest of the Earth (Rantanen et al., 2022). Sea ice decline has increased in recent years (Meredith et al., 2019), and a seasonally ice-free Arctic is predicted by the middle of this century (Notz and Stroeve, 2018). Impacts on our global climate system, such as changes in jet streams or increased cooling of the mid-latitudes, are being discussed (Francis and Vavrus, 2015; Cohen et al., 2020). Consequences for Arctic people and the shipping industry are already evident (Hovelsrud et al., 2011; Eguíluz et al., 2016).

1.1 Melt Ponds and the Arctic Heat Budget

Strong seasonal changes of Arctic surface structures occur in summer, when temperatures rise close to or above zero degrees. The melting of sea ice and snow leads to the formation of melt ponds, pools of water that collect on areas of lower topography. They begin to appear around the end of May, can cover up to about 60-80% of the sea ice area at the peak of melt, and refreeze in August and September (Eicken et al., 2004; Eicken et al., 2002; Perovich et al., 2002). Melt ponds have a strong influence on the Arctic energy budget. Due to their darker appearance, they absorb significantly more sunlight than reflective sea ice and snow (Fetterer and Untersteiner, 1998; Grenfell and Maykut, 1977; Grenfell and Perovich, 1984; Grenfell and Perovich, 2004; Nicolaus et al., 2010). This causes the surrounding areas to warm up, leading to further melt and rapidly changing surface properties (Polashenski, Perovich, and Courville, 2012a).

The impact of melt ponds on the Arctic climate system depends on their spatial extent (Eicken et al., 2004). Melt pond fraction (MPF), the fraction of sea ice surface covered by ponds, is a crucial parameter for models that predict Arctic climate evolution, but has not

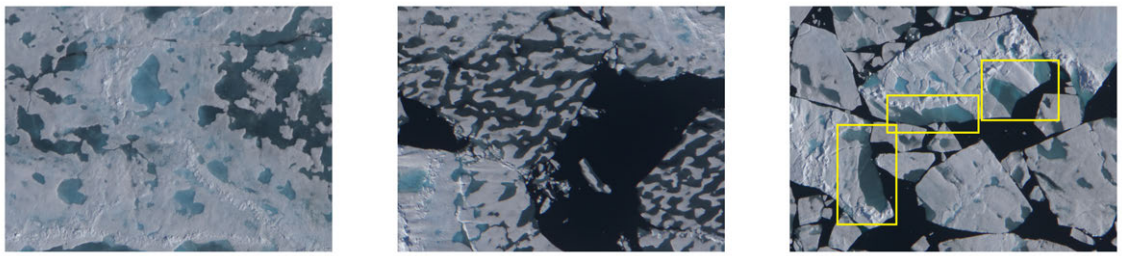

FIGURE 1.1: Various melt pond shapes depicted in VIS imagery. Yellow rectangles are examples of submerged ice. Location: Fram Strait region. Date: July 18, 2022 (Kanzow, 2023, AWI PS131 02). VIS Images are provided by Lena Buth, Alfred Wegener Institute.

been sufficiently integrated in recent models (Flocco, Feltham, and Turner, 2010; Flocco et al., 2012; Hunke, Hebert, and Lecomte, 2013; Polashenski, Perovich, and Courville, 2012b; Schröder et al., 2014). The development of methods to accurately retrieve and analyze MPF is an ongoing research area.

1.2 Methodology to Observe Melt Ponds

Accessing MPF is hampered by the remoteness of the Arctic Ocean. For a comprehensive analysis, a combination of different data sources is needed. Existing methods observe the Arctic from different platforms and with different sensors.

1.2.1 Spatial Scale

In situ measurements collect melt pond data directly from the location of interest (Landy et al., 2014; Polashenski, Perovich, and Courville, 2012b; Eicken et al., 2004; Polashenski et al., 2017). These methods are needed to understand the underlying processes, but are rare, resource intensive, and not suited to represent spatially and temporally varying MPF for the entire Arctic.

Remote sensing techniques provide data from a distance, making it possible to observe a larger area [1]. Satellites are a promising method in the long term because they can cover major parts of the Arctic on a regular basis. To date, most MPF retrievals have used low- and medium-resolution satellite imagery, e. g., MODIS with a spatial resolution of 250m to 1km (Fuchs, 2023a; Rösel and Kaleschke, 2011). This does not resolve individual ponds, making accurate acquisition a challenge (Stroeve et al., 2021; Rösel, Kaleschke,

[1] For a comprehensive introduction to remote sensing, see Campbell and Wynne, 2011

and Birnbaum, 2012). High-resolution satellite products are available (Niehaus et al., 2023), but still limited in spatial coverage (Wright and Polashenski, 2020).

In conjunction with satellite observations and ground-based measurements, airborne campaigns can obtain MPF for specific areas with high resolution. Airborne measurements have the advantage of flexibility, as they can be targeted to areas of interest and avoid cloud cover. They can be used for the validation of satellite retrievals, see, e. g., Niehaus et al., 2023 and Istomina et al., 2015.

1.2.2 Sensors

Depending on the sensor used, remote sensing techniques record radiation at different frequencies.

Visual imagery (VIS) is often used to derive MPF (e. g., Lu et al., 2010; Perovich, Tucker III, and Ligett, 2002; Niehaus et al., 2023; Wright and Polashenski, 2018). VIS captures the reflected amount of sunlight in three channels (usually RGB) and indicates melt ponds by their blueish to dark gray color (Perovich et al., 1996, Figure 1.1). A drawback of VIS imagery is its dependence on sunlight and sensitivity to different lighting conditions. Also, VIS is affected by cloud cover, which can be high in summer when MPF is at its peak (Perovich, Tucker III, and Ligett, 2002).

Microwave measurements are independent of sunlight and clouds. Scharien et al., 2007 and Kim et al., 2013 used high-resolution synthetic aperture radar (SAR) data for MPF retrieval. However, difficulties in detecting small melt ponds were reported (Kim et al., 2013). Apart from SAR, microwave technology is hampered by relatively low resolution.

Thermal infrared (TIR) imagery measures the emissivity of thermal radiation with wavelengths around 10μm. Like VIS, TIR is affected by cloud cover, but can be used in the absence of daylight. Thielke et al., 2023 was able to show that infrared imagery is capable of detecting winter sea ice surface features. Figure 1.2 shows the potential to identify summer melt ponds. By capturing the surface temperature, TIR allows to address new research questions about the thermal properties of melt ponds and can contribute to a better understanding of the summer heat budget.

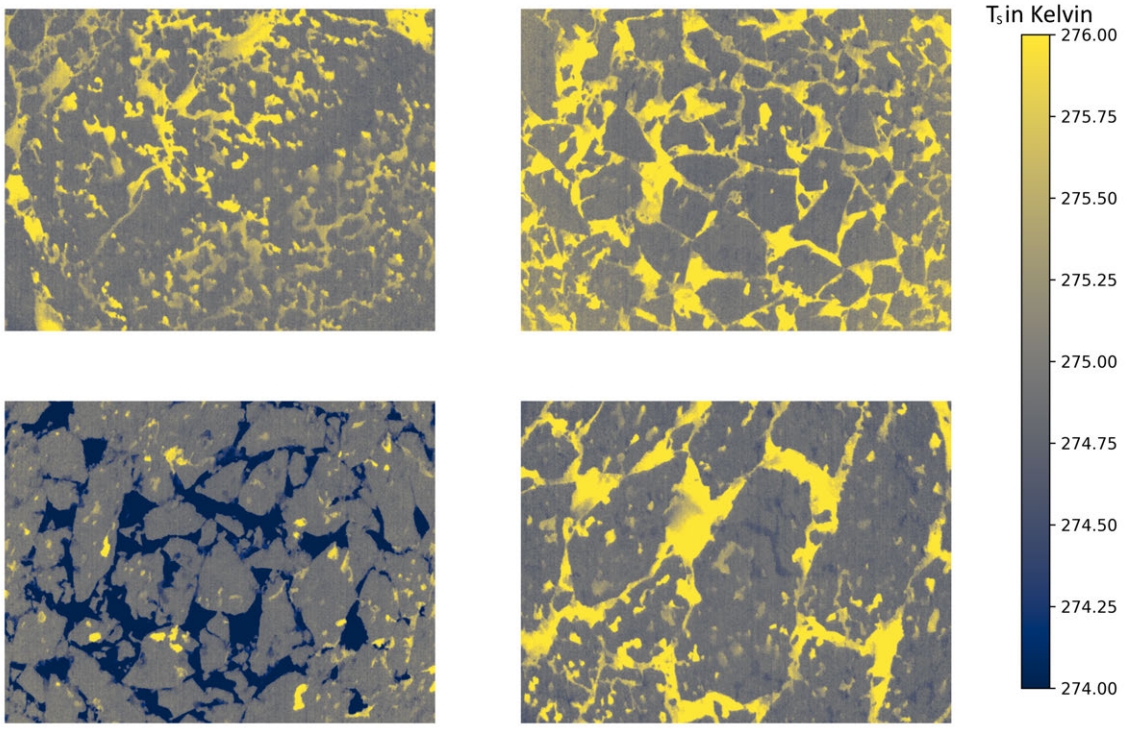

FIGURE 1.2: Ice floes with melt ponds depicted in TIR imagery. Images are pseudo-colored for visibility reasons. Yellow corresponds to warm, blue to cold temperatures, as shown on the right (surface temperatures (T_S) in Kelvin). Location: Fram Strait region. Date: July 18, 2022.

So far, TIR imagery is rarely used in MPF retrieval. At present, only low-resolution TIR satellite products are available, which need to be validated by helicopter measurements (Thielke et al., 2022). Methods are needed to separate these data into different surface types.

This thesis investigates the segmentation of helicopter-borne TIR images into different surface classes. We follow the surface type definition of Wright and Polashenski, 2018: (1) melt ponds - surfaces where liquid water covers ice. This includes ponds that are completely embedded in ice floes and submerged ice at the edges of floes (Figure 1.1), (2) sea ice and snow, and (3) ocean - open bodies of water that are not covered by ice.

The remainder of this thesis is organized as follows. In Chapter 2 we review related work and present the method that we intend to use. In Chapter 3 we specify our experimental setting. In Chapter 4 we provide and interpret our results. In Chapter 5 we summarize our findings and give directions for future work.

Chapter 2

Background

2.1 Related Work and Requirements for TIR Segmentation

Semantic segmentation is the task of assigning a class value to each pixel of an image. Existing methods in the field of melt pond retrieval often use traditional image processing techniques and are mainly divided into two categories: (1) pixel-based and (2) object-based segmentation.

Pixel-based methods rely on single pixel values. Many works on VIS imagery are based on thresholding approaches (Lu et al., 2010; Perovich, Tucker III, and Ligett, 2002; Tschudi, Curry, and Maslanik, 2001; Krumpen et al., 2011; Inoue, Curry, and Maslanik, 2008; Huang et al., 2016), where a threshold is set to separate instances below and above that value. Other approaches have used pixel-wise classification with supervised methods such as random forests (Fuchs, 2023b).

Object-based segmentation aims to group multiple pixels into meaningful objects and can take into account both spatial and spectral features. For example, Miao et al., 2015 and Wright and Polashenski, 2018 developed object-based algorithms that first separate objects from images by edge detection and classify them according to manually defined features such as texture and color.

For TIR imagery, spatially and temporally varying temperatures place special demands on the segmentation algorithm that cannot be met by traditional methods. Due to its salinity, ocean water has a freezing temperature of -1.8 degrees Celsius. This means that in summer, sea ice can be warmer or colder than the surrounding ocean (Figure 1.2). Surface types cannot be predicted from pixel brightness alone, and single pixel-based

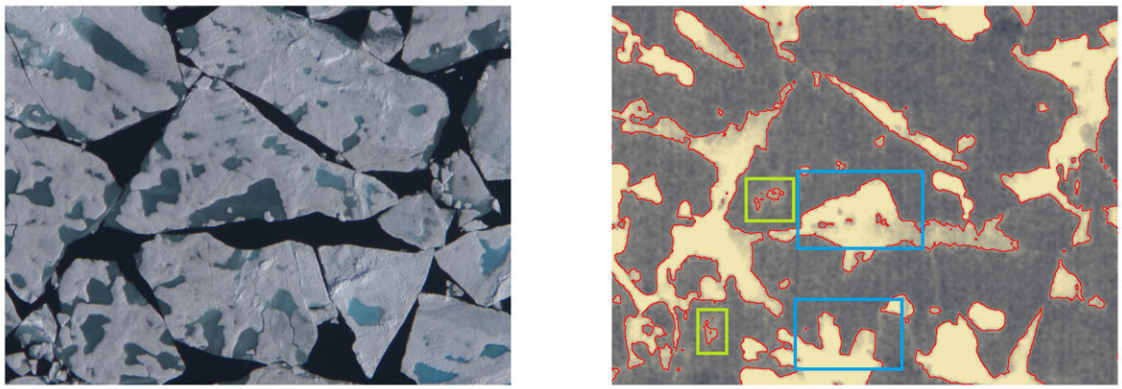

FIGURE 2.1: Edge-based segmentation has difficulties to distinguish submerged ice from ocean (blue rectangles) and cannot detect melt ponds with small temperature changes (green rectangles). For this example, a Scharr filter was used (red outlines, Scharr, 2000). Yellow corresponds to warm, gray/blue to cold temperatures (right image). We did not plot a colorbar because the image was edited for visibility reasons. Location: Fram Strait region. Date: July 18, 2022 (Kanzow, 2023, AWI PS131 02).

methods are not applicable. Considering object-based segmentation, we observed that object boundaries in TIR are challenged by smooth temperature transitions or no transitions at all, as in the case of submerged ice and ocean (Figure 2.1).

What is needed for TIR segmentation is a more general method that relies on various attributes such as shape and size simultaneously. Melt pond shapes can range from single circular shapes to complex interconnected networks (Hohenegger et al., 2012; Polashenski, Perovich, and Courville, 2012a) and sizes can range from square centimeters to square kilometers (Perovich, Tucker III, and Ligett, 2002). We considered to use a deep learning-based approach that is capable of efficiently extracting multiple features.

Only a few studies have applied deep learning to melt pond segmentation. This may be because in VIS imagery, surface types are relatively easy to distinguish by simple features, and thus complex methods are not needed. Lee et al., 2020 trained a multilayer perceptron for pixel-wise classification of optical satellite imagery. Panchi, Kim, and Bhattacharyya, 2021 used a neural network ensemble to segment ship-based optical imagery into ice and surface features, and Sudakow et al., 2022 developed a modified U-net to retrieve melt ponds from airborne optical images.

2.2 Feature Extraction with Convolutional Neural Networks

In recent years, major improvements in image classification and segmentation have been achieved with convolutional neural networks (CNN; O'Shea and Nash, 2015; Krizhevsky, Sutskever, and Hinton, 2012). CNNs are supervised deep learning methods that automatically learn spatial features from images at different levels of abstraction. CNNs can capture local features and spatial context simultaneously, making them promising for the task at hand. For example, a CNN might recognize that a circular shape alone does not necessarily indicate a melt pond, but its embedding in sea ice suggests that it is a pond rather than an ice floe. In addition, CNNs take over the task of thorough manual feature engineering that was necessary for many of the works mentioned in Section 2.1.

The main components of CNNs are:

- convolutional layers that extract features by sliding a filter or kernel across the image dimensions. They are typically followed by a rectified linear unit (ReLU) activation function;
- pooling layers that are used to reduce the spatial dimensions of the input data by downsampling.

2.3 Semantic Segmentation with U-net

U-net is a segmentation architecture that was originally introduced by Ronneberger et al. in 2015 (Ronneberger, Fischer, and Brox, 2015). U-net uses a CNN as an encoder for feature extraction. This is combined with a decoder path that maps the features learned at different levels to the pixel space (Figure 2.2). This way, the compressed feature representation is restored to a larger image size. In addition to CNN components, U-net uses:

- upsampling, which, in contrast to pooling, is used to increase the spatial resolution of the feature maps. This can be done through upsampling layers or transposed convolutions, where in this work, we used upsampling layers;

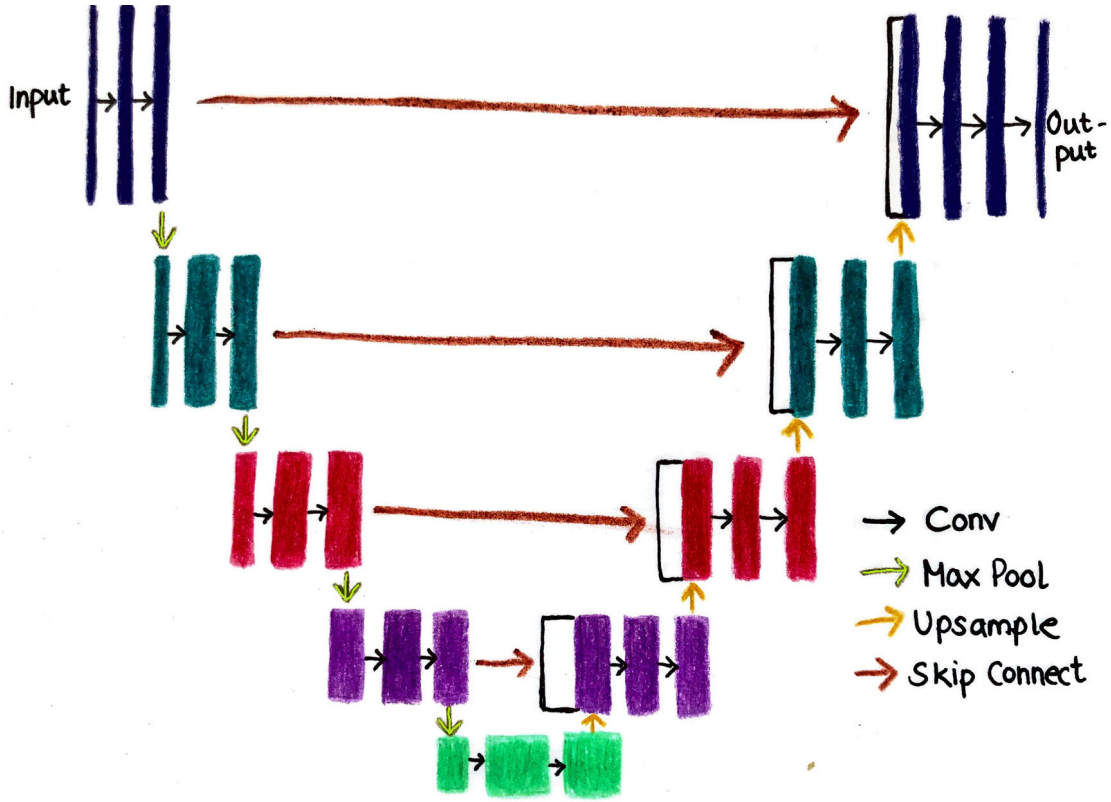

FIGURE 2.2: Basic U-net architecture. Each box represents a multi-channel feature map. The white boxes correspond to copied feature maps. Contracting path (encoder) and expanding path (decoder) form a U-shaped network. Skip connections provide localization information.

- concatenations, which are used to combine the upsampled feature maps with corresponding features from the encoder path at different stages. This allows the decoder to access high-resolution information that would otherwise be lost during downsampling.

For more information on the original architecture, see Ronneberger, Fischer, and Brox, 2015.

Although initially developed for the purpose of biomedical image segmentation, U-net has been the primary architecture used for remote sensing tasks (Hoeser, Bachofer, and Kuenzer, 2020). It won the DSTL Satellite Imagery Feature Detection challenge on Kaggle (Benjamin, 2016; Iglovikov, Mushinskiy, and Osin, 2017). Skip connections allow for precise location restoration, which is useful for small melt pond features. Its simple structure is easily modifiable, providing potential for task-specific refinements in the future.

2.4 Overfitting

A drawback of CNN-based architectures is their dependence on big data due to a large number of trainable parameters. Small data leads to the problem of overfitting: The network learns a function with high variance, so that it is not able to generalize on unseen samples.

This is a challenge for Arctic remote sensing tasks where data acquisition is limited. Even more demanding for the segmentation task at hand is the difficult and time-consuming annotation process (Section 3.2).

Still, training U-net with small data can be successful (Iglovikov, Mushinskiy, and Osin, 2017; Ronneberger, Fischer, and Brox, 2015). Regularization methods can help to mitigate overfitting. In the following, we present some of the techniques investigated in this thesis.

2.4.1 Augmentation

Augmentation has been proven to be one of the most powerful techniques to combat overfitting. It refers to synthetically modifying training data prior to model training by applying image transformations. In this way, the size of the dataset can be increased and more information added to the input. For example, variations in altitude, atmospheric effects, or seasonal and regional changes can be mimicked. For a comprehensive overview of the most popular augmentation techniques, see Shorten and Khoshgoftaar, 2019.

So far, there is no consensus on the best increase in dataset size through augmentation, nor on the ideal number of transformations to be applied to an image (Shorten and Khoshgoftaar, 2019; Wagner, Eltner, and Maas, 2023). Which augmentation techniques work best is highly task dependent.

2.4.2 Transfer Learning

Transfer learning (Weiss, Khoshgoftaar, and Wang, 2016; Shao, Zhu, and Li, 2014) addresses overfitting by using a pre-trained model that is usually trained on a large-scale classification task. The idea is that the features learned from a large dataset are general enough to be transferable to different tasks. For very small datasets, a pre-trained

network can improve the generalization ability of the model, as it has seen different examples. The pre-trained model can be used as a fixed feature extractor, or fine-tuned for the task at hand.

Often, ImageNet is used as pre-training dataset. ImageNet (Deng et al., 2009) is a benchmark computer vision dataset that comprises over 14+ million labeled images with 1000 classes. It contains RGB images of complex everyday objects such as tables, chairs, cars, and cats. This makes it very different from Arctic TIR images, which consist of homogeneous surface structures contained in a single color channel and captured from an overhead perspective. However, previous work has shown that pre-training on ImageNet is beneficial even in cases where the secondary task is significantly different from the primary task (Hu et al., 2015; Yosinski et al., 2014; Lima and Marfurt, 2019; Esteva et al., 2017).

2.4.3 Dropout

Dropout is a regularization technique applied during model training. It randomly sets a fraction of network units to zero during learning, which prevents the model from relying too much on specific features (Srivastava et al., 2014).

In this thesis, we aim to use U-net to semantically segment melt ponds, sea ice, and ocean classes from TIR imagery.

We divide our experimental work into two parts. (1) Model selection, where we compare different hyperparameters settings in order to optimize the model configuration, and (2) final model construction, where we build and evaluate our final model. We refer to hyperparameters as external choices that can be made for the training configuration, namely: patch size, loss function, whether to use dropout layers, the pre-training strategy, augmentation techniques (the transformations applied to our images), and augmentation design (whether to use on-the-fly or offline augmentation). More information on the hyperparameters follows in the next chapter.

Chapter 3

Methodology

In this chapter, we exhibit our experimental setup for constructing a U-net that segments TIR imagery into different surface classes.

It contains a description of the dataset (Section 3.1), the annotation procedure (Section 3.2), and our model architecture (Section 3.3). We present preprocessing steps (Section 3.4), the hyperparameters under optimization (Section 3.5), the training and prediction procedure (Section 3.6), and evaluation techniques (Section 3.7).

3.1 Data

We used helicopter-borne TIR imagery acquired with an Infratec Vario-CAM HD head 680 camera during the PS131 **AT**lantic **WA**ter pathways to the **ICE** campaign (ATWAICE; Kanzow, 2023). A total of 16 flights were conducted at an altitude of about 300m in July and early August 2022, corresponding to the season of peak pond coverage (Perovich, Tucker III, and Ligett, 2002). The geographic area of study was the marginal ice zone of the Fram Strait region (Figure 3.1).

We selected training images from Flight 9, which was performed on July 18, 2022. This was the only choice where no clouds or mist were reported. Out of 4989 images, we selected 10 for labeling. We tried to include images with features of different sizes and shapes, and with good visibility to ensure labeling accuracy.

For the small amount of training data that we were able to annotate, Flight 9 could provide a sufficient variety of images. For testing purposes, we included Flight 16 with mostly sunny conditions.

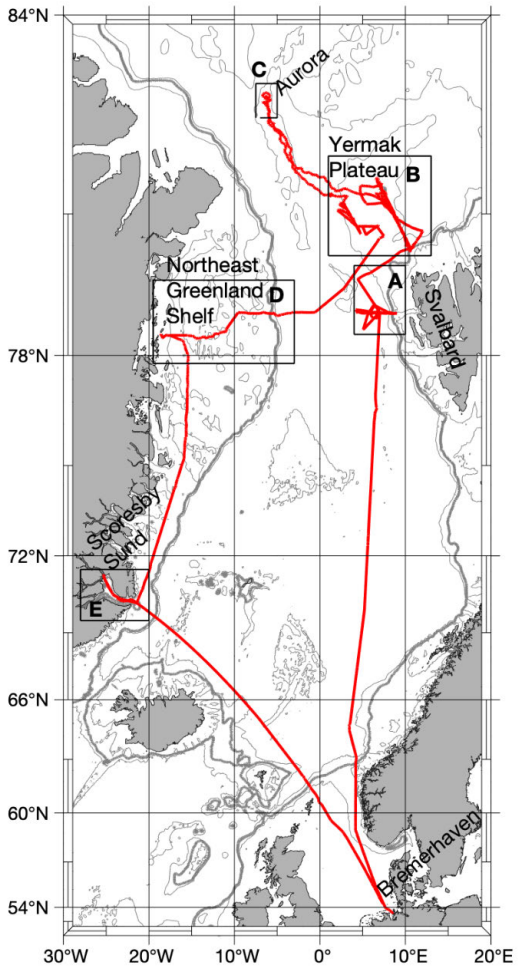

FIGURE 3.1: Route of PS131. Image Source: Kanzow, 2023.

Data was obtained in NetCDF4 file format with 640 x 480 pixels per image. Each image records the broadband infrared radiation (7.5μm to 14μm) at roughly 1m resolution. It is likely that image gradient and drift correction was applied prior to this study (Thielke et al., 2022), although it was not possible to fully trace these steps. The images were not georeferenced in advance, which introduces geometric distortions, especially at the image boundaries. We colormapped the images and saved them in PNG format using matplotlib (Hunter, 2007) with colormap 'cividis' for inspection and 'gray' for further processing.

For more information on the infrared camera and the cruise, see Thielke et al., 2022 and Kanzow, 2023.

3.2 Annotation

We labeled each selected image pixel-wise into melt pond, sea ice, and ocean. Small features and smooth boundaries challenged our ability to discriminate during annotation. We approached this problem using a variety of methods, which we present in this section. A detailed description of the annotation process can be found in Appendix A.

Prior to annotation, we created a binary edge map to provide a starting point for manual correction (Figure 3.2). This was done using a Scharr operator (Scharr, 2000). We chose GIMP 2.10 (The GIMP Development Team, 2019) as annotation software because it allowed us to work with both the image and the edge map, and provided the tools needed for fine-grained labeling. Other popular software such as LabelMe were not considered because they only allow polygon-based annotation, which is not suitable for small and irregularly shaped ponds.

For annotation purposes, we resized the image to 3210 x 2345 pixels per image. We found this to be more convenient for the labeling process, as small features were found to be hard to distinguish at low resolution. However, some accuracy may be lost in the final annotation masks due to interpolation.

In parallel with annotation, we inspected VIS images recorded at the same second to reduce annotation uncertainty. This was particularly important for submerged ice, which was difficult to identify from TIR imagery alone. The VIS images were taken with a Canon EOS-1D Mark III 14mm camera with 3908 x 2600 pixels per image and were

FIGURE 3.2: Edge map (middle) used as a starting point for labeling with corresponding TIR (left, clipped to temperature values between 273 and 276 Kelvin for visibility reasons) and VIS image (right). Location: Fram Strait region. Date: July 18, 2022 (Kanzow, 2023, AWI PS131 02).

provided by Lena Buth from the Alfred Wegener Institute. They show a slightly larger surface area than the corresponding TIR images (Figure 3.2). To further speed up the process, we tried to align the corresponding images. Due to distortions, this was not possible with simple means and we postponed this task to future work.

We used the following criteria to annotate an object as melt pond:

- features that were characterized by a temperature change in TIR, detected by the edge detection algorithm, and identified as melt ponds in VIS;
- features at the edges of ice floes that were identified as submerged ice in VIS.

We observed that the VIS images show more ponds than can be detected for the same area in TIR. Some ponds in TIR are characterized by only small temperature changes with smooth boundaries. Their shape often did not match that shown in VIS, and we decided not to annotate these cases. Thus, an underrepresentation of melt ponds in the training set compared to ponds in the VIS region is likely. We discuss this further in Section 4.2.2.

For each image, the annotation process took several hours. Thus, 10 images was the maximum number of images that could be labeled for this work.

3.3 Model Architecture

For our U-net architecture, we used the publicly available Segmentation Models library (Iakubovskii, 2019) based on Keras (Chollet et al., 2015) and TensorFlow (Abadi et al., 2016). We selected this implementation because of its high-level functionality and the ease of using pre-trained models. Changes that we implemented are reported in the

`README.md` of our GitHub repository [1]. Below we list some main characteristics of our U-net implementation.

- Backbone: To be able to use pre-trained weights from the ImageNet classification task, we constructed a backboned U-net with a pre-existing CNN architecture as the encoder. We chose ResNet34 as our backbone because it is commonly used for remote sensing tasks (Hoeser, Bachofer, and Kuenzer, 2020). ResNet34 is a popular classification architecture that uses residual connections to make training more efficient (He et al., 2015a). It is composed of 34 layers organized in residual blocks, which allows to learn only the differences (residuals) between the input and the desired output. For incorporation into U-net, the fully connected layers of the ResNet are removed and connected to a decoder.

- Batch normalization: Batch normalization layers (Ioffe and Szegedy, 2015) are embedded in the encoder and decoder after each convolution and before activation. This is to normalize batch activations for convergence acceleration.

- Zero padding is used to be able to predict border pixels and restore the original image size. Inaccuracies in the border regions due to missing context are addressed in Section 4.3.

Our final model contains over 24 million trainable parameters. We adjusted the input size of the model to fit our specific image size. The final layer uses a softmax activation function to obtain a probability distribution over classes. By adding an argmax operation, each pixel is assigned the most likely class value.

Figure 3.3 shows our final network architecture. For more information on the model used, see our GitHub repository (`summary/model_dropout.png`) and Iakubovskii, 2019.

3.4 Data Preprocessing

Initial attempts to train with temperature pixel values resulted in poor performance. For our final setting, we loaded the colormapped PNG images with pixel values ranging from 0-255.

[1] `https://github.com/marlens123/ponds`

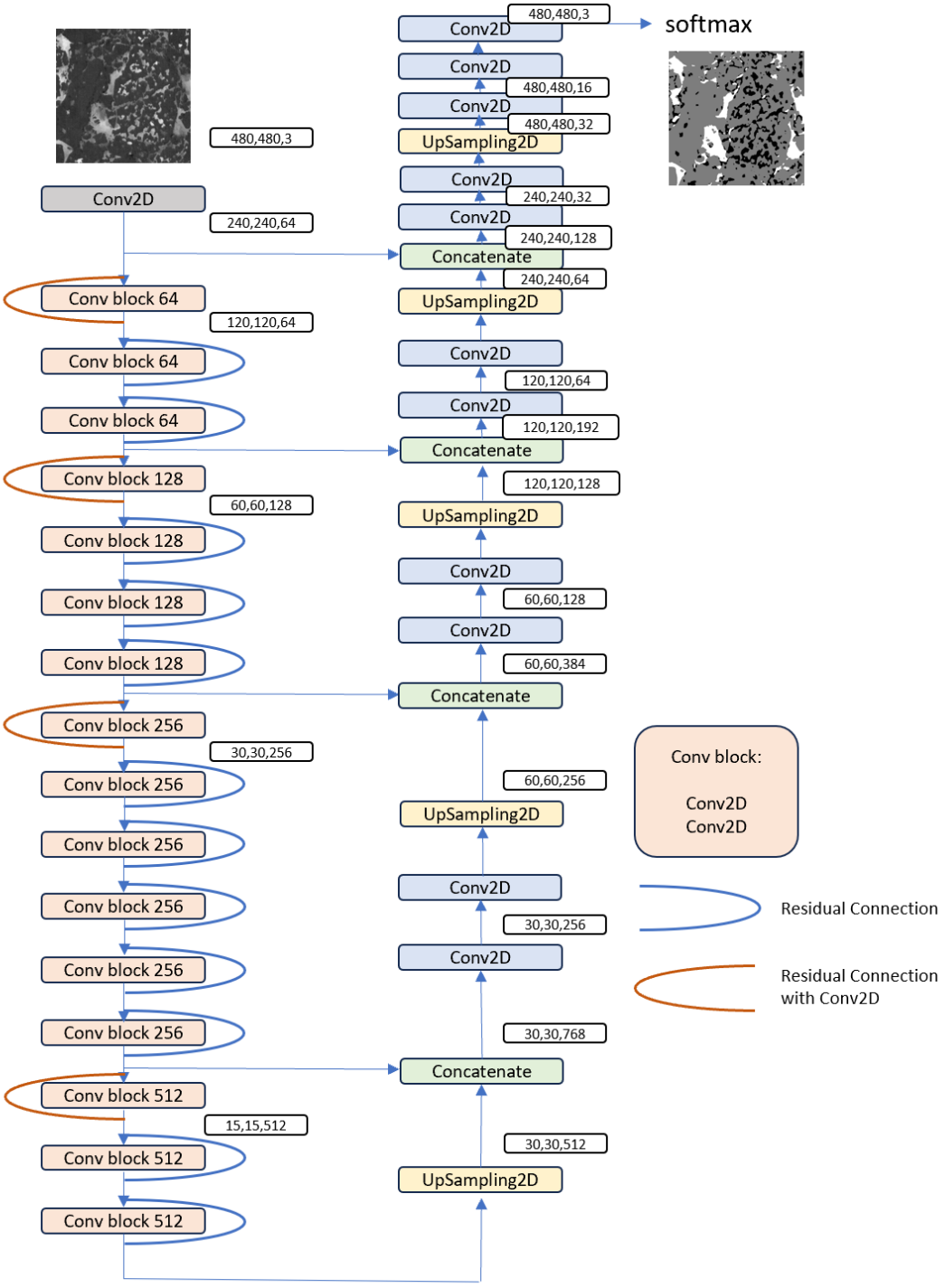

FIGURE 3.3: Model architecture used in our work. The encoder (left) consists of a ResNet34 architecture. Skip connections transfer information from different layers in the encoder to the decoder (right). Batch normalization, zero padding and max pooling layers are not shown.

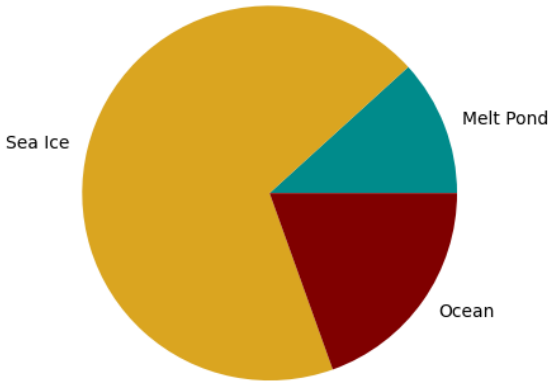

FIGURE 3.4: Class-wise pixel distribution in the dataset used for training.

We center-cropped the images to a size of 480 x 480 pixels to eliminate major distortions at the image boundaries. Since our backbone architecture is designed for 3-channel input, we repeated the last dimension 3 times. The data type was converted from int32 to float32 to increase precision for training. We followed the implementation used and did not apply any further preprocessing steps.

We loaded the annotated masks as grayscale images and converted pixel values to 0, 1 and 2, where 0 encodes the melt pond class, 1 encodes the sea ice class, and 2 encodes the ocean class. We resized the masks back from 3210 x 2345 to 640 x 480 and center-cropped them to a size of 480 x 480.

Figure 3.4 shows the class-wise percentage of pixels in the training dataset. The dataset is very unbalanced, which we account for in Section 3.5.2.

Our preprocessing steps are located in `prepare_data.py` and `utils/data.py`. The entire dataset can be inspected in Appendix A.

3.5 Hyperparameters

3.5.1 Patch Size

Prior work has found that for small datasets, performance can be improved by cropping the images into smaller patches, which allows for a larger batch size (Iglovikov, Mushinskiy, and Osin, 2017). In addition, a smaller patch size may help the model to focus on the

spatial context in the immediate vicinity of melt ponds, which may be more important for detection rather than the larger surrounding.

On the other hand, a patch size that is too small may not capture enough context for correct classification, e. g., embedding in pond networks may be important for separating melt pond and ocean classes.

We investigated this trade-off by testing five different patch sizes. We started with a patch size of 32 x 32, as this was large enough to fully capture individual ponds. The model input size was limited by the implementation used to be divisible by 32 (Iglovikov, Mushinskiy, and Osin, 2017). For smaller patch sizes, we used larger batch sizes. Summaries of the respective dataset sizes and image contents can be found in Table 3.1. Example patches are shown in Appendix A.

We created the patches with a sliding window approach. For the 256 x 256 patch size, overlapping was inevitable when extracting multiple patches. For the other sizes, we increased the step size to avoid overlap. We performed patch extraction after the dataset split to prevent information leakage between the train and test sets in the case of overlap. We applied patch extraction to both the training and test sets in the same way.

3.5.2 Loss Function

For the initial training runs, we applied the categorical cross-entropy (CCE) loss function, which is widely used for multi-class classification tasks. It is defined as follows:

$$\mathrm{CCE} = -\sum_{i=1}^{C} y_i \log(p_i), \tag{3.1}$$

where C is the number of classes, y_i the ground truth probability, and p_i the predicted probability of class i.

We additionally tested a dual loss function including a multi-class version of focal loss (CFL; Lin et al., 2017) and dice loss (DL), as implemented by Iakubovskii, 2019. The dual categorical focal dice loss function (CFDL) used in this work is defined as:

$$\mathrm{CFDL} = \mathrm{CFL} + \mathrm{DL}, \tag{3.2}$$

where

$$\text{CFL} = -y_i \cdot \alpha \cdot ((1 - p_i)^\gamma) \cdot \log(p_i), \tag{3.3}$$

and

$$\text{DL} = L(tp, fp, fn) = \frac{(1 + \beta^2) \cdot tp}{(1 + \beta^2) \cdot fp + \beta^2 \cdot fn + fp}. \tag{3.4}$$

For CFL, α is a weighting factor and γ the focusing parameter for the modulating factor $(1 - p_i)$. Both values are kept by default. For DL, tp are the true positives, fp the false positives, fn the false negatives and β a parameter that balances the weight of precision and recall.

For image segmentation tasks, the loss is computed for each pixel individually and then averaged over all pixels in a training batch. This is problematic in the presence of class imbalance, as all pixels are treated equally and the training loss is biased towards the majority class.

To overcome this problem in our highly unbalanced dataset, we incorporated class weights into the DL part of the CFDL function by assigning higher weights to the contributions of minority classes.

We computed class weights for the training set after the dataset split to avoid data leakage using the scikit-learn library (Pedregosa et al., 2011). These weights are multiplied by the corresponding class score obtained by the loss function before averaging over all pixels in a batch.

3.5.3 Dropout

We chose to incorporate dropout layers after each upsampling operation in the decoder with a dropout probability of 0.5. The dropout rate was chosen based on literature (Ren et al., 2021; Rajaraman et al., 2020).

3.5.4 Pre-training Strategy

We tested different pre-training strategies: (1) In fine-tuning, we used pre-trained encoder weights from ImageNet and set all network layers to trainable for the entire training. This allowed us to tune the learned features to our specific task. (2) In encoder freeze, we used a fixed feature extractor pre-trained on ImageNet and updated only the decoder

weights during training. (3) In addition, we trained from scratch, i. e., we randomly initialized the weights for both the encoder and the decoder.

3.5.5 Augmentation Techniques

We considered the following augmentation techniques:

- horizontal and vertical flipping, to simulate changes in flight orientation;
- rotation, for the same reason;
- cropping, to simulate instability in flight altitude and to introduce variations in object scale, as ice floes and ponds can vary greatly in size;
- brightness contrast, to account for varying surface temperatures;
- sharpen blur, blurring simulates the impacts of noise or atmospheric effects, while sharpening reduces these effects;
- Gaussian noise injection, to increase the model's robustness to noise.

Inappropriate augmentation techniques can introduce unrealistic transformations into the dataset and lead to a decrease in model performance. Therefore, we subsequently added the candidate methods in different training runs.

We excluded augmentation techniques like color transformations and perspective changes from the beginning. Color transformations such as color jitter were not applied due to the single-channel nature of the images, and perspective transformations are unrealistic since all images are taken from overhead and approximately the same angle.

We used the publicly available Albumentations library for implementation (Buslaev et al., 2018). For methods using interpolation, we changed the interpolation to nearest to preserve categorical mask labels. For rotation, boundary values were extrapolated by reflection, as this was the most natural choice. Cropping was done by cutting an area with a minimum of 0.5 and a maximum of 0.8 of the image size and then resizing the cut to the original image size. If augmentation was sharpen blur, either the operation 'Sharpen', 'Blur', or 'MotionBlur' was chosen with equal probability. Other parameters were left at their default values.

We added augmentation after patch extraction, i. e., transformations are applied to patches and not to the entire images. The implementation can be found in `utils/augmentation.py`. Example applications of the augmentation techniques are shown in Appendix A.

3.5.6 Augmentation Design

After selecting the best augmentation configuration, we tested the effect of on-the-fly versus offline augmentation.

On-the-fly augmentation transforms images in real time as they are fed into the model (Shorten and Khoshgoftaar, 2019). This typically provides a high degree of flexibility, as augmentations are applied randomly to each image in a training iteration. On-the-fly augmentation is memory efficient because no additional storage is required. We implemented on-the-fly augmentation by applying each considered method to the training image with a probability of 0.5.

Offline augmentation refers to applying transformations to the dataset before training begins, generating a fixed augmented dataset (Shorten and Khoshgoftaar, 2019). This allows more control and transparency over the applied augmentations. Offline augmentation effectively increases the size of the dataset, but at the cost of storage requirements. Wagner, Eltner, and Maas, 2023 reported that offline augmentation can be beneficial for very small datasets.

3.6 Training and Prediction Procedure

Our training was performed on an Intel i7-9700K CPU@3.60GHz with 8 cores, coupled with 32GB of RAM. We monitored and plotted our training and validation curves with respect to mIoU and per-class IoU using the Weights & Biases dashboard (Biewald, 2020). The implementation can be found in `train.py`.

We followed Iakubovskii, 2019, and for weights that are randomly initialized, we used He uniform (He et al., 2015b) initialization for each convolutional block and Glorot uniform (Glorot and Bengio, 2010) initialization for the final convolutional layer. To optimize the model parameters, we employed the commonly used Adam optimizer (Kingma and Ba, 2017) with a default learning rate of 0.001. Batch sizes were dependent on the patch size used (Table 3.1). By default, we trained with the categorical cross-entropy loss

function, without augmentation, and used the fine-tuning pre-training strategy to speed up model convergence.

For model selection, we trained most configurations for 100 epochs using gradient descent. A number of epochs of 100 seemed to be a good compromise between the required computational resources and model convergence. At the end of the training phase, we saved the best weights of the model in HDF5 format with respect to the minimum validation loss. This ensures that even if performance drops in later epochs, the best state of the model is saved.

For prediction (`predict_image.py`), we used the same image preprocessing as described in Section 3.4. For visualization purposes, we optionally converted the resulting class values to gray levels. Prediction on smaller patch sizes required additional steps, which we describe in Section 4.3.

3.7 Evaluation

3.7.1 Evaluation Metric

Intersection over Union (IoU), also known as Jaccard index (J), is the standard metric for segmentation tasks. It measures the similarity between the predicted region and the ground truth region for a given class. IoU considers both false alarms and missed values for each class. It is defined as follows:

$$J(G, P) = \frac{|G \cup P|}{|G \cap P|}, \tag{3.5}$$

where G is the ground truth and P the predicted segmentation map.

We used this metric for the melt pond, sea ice, and ocean classes, respectively. To obtain a single performance metric, we averaged over all classes. This is hereafter referred to as mean IoU (mIoU).

3.7.2 Dataset Split

For model selection, we needed a technique that properly described the relative performance of the model between different configurations. Using a simple train-test split for

our very small dataset would likely have suffered from high variation between sets, resulting in different estimates depending on the split. For example, the test set might have contained only images that were easy to predict, resulting in an optimistically biased estimate of model performance.

To provide a more robust technique, we used k-crossfold validation. In k-crossfold validation, the dataset is divided into k subsets of equal size. For each hyperparameter configuration under observation, we trained k times. In each of the k runs, we used a different subset for validation and merged the remaining subsets for training. We averaged the performance outputs of all runs to have a single overall measure. In addition, we monitored the standard deviation of the runs to obtain an estimate of reliability. To compare different hyperparameter configurations, we considered the best averaged validation mIoU after 20 training steps.

We used a k of 4 to have an equal distribution of 2 images per subset and a manageable computational cost. For implementation, we used the scikit-learn library (Pedregosa et al., 2011). For comparison, we ensured that each hyperparameter configuration received the same splits by setting a random state. The class proportions of each subset can be found in Appendix A.

To construct our final model, we trained on the entire dataset to ensure that we were not withholding valuable information (Raschka, 2018). We then used an unseen test set with 2 images as the first component for the final model evaluation. The test set contained the same class distribution as the training set (see Appendix A). To obtain a more robust final performance estimate, we also inspected some prediction results and evaluated the final MPF product (Section 3.7.3).

An overview of the evaluation procedure is shown in Figure 3.5 and the implementation is part of `train.py`.

3.7.3 Evaluation of Resulting Melt Pond Fraction

For additional evaluation, we computed the MPF using segmentation maps predicted by our final model and compared the result with an estimate obtained from VIS imagery (`mpf.ipynb`). We used the VIS images introduced in Section 3.2. We randomly selected

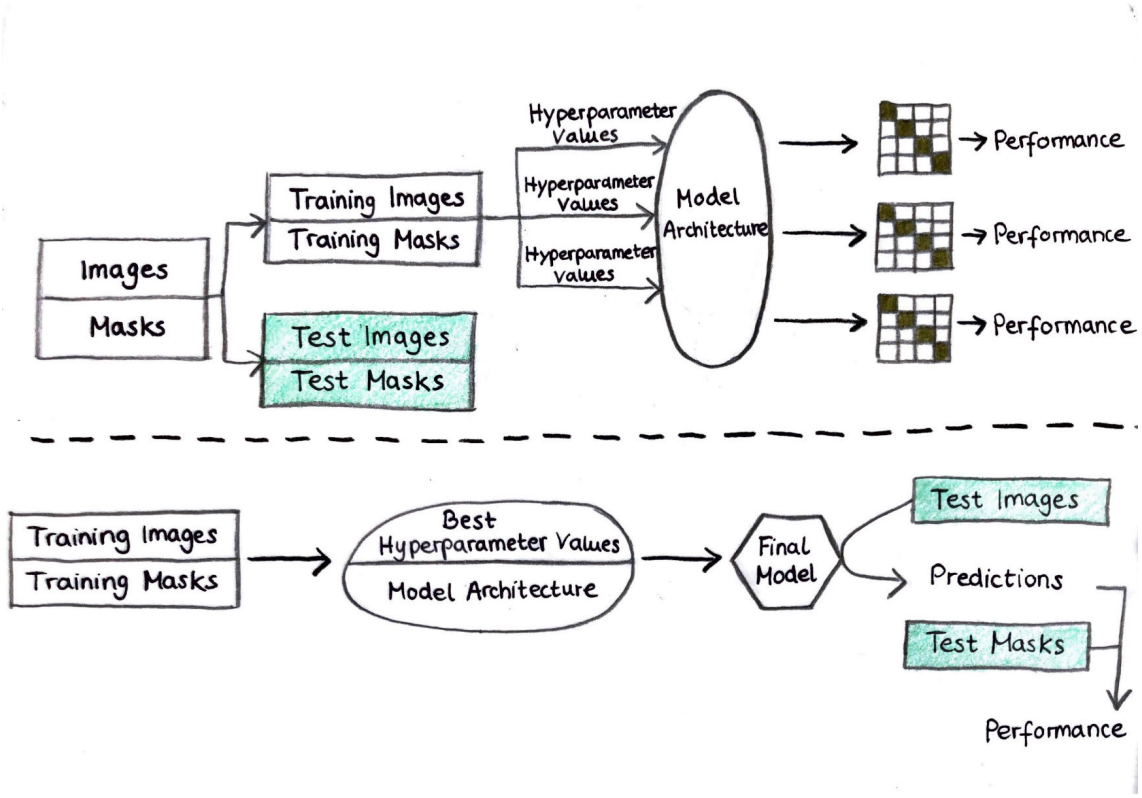

FIGURE 3.5: The process of model selection (top) and final model construction (bottom) in our experiment. For visibility reasons, only a part of the final model evaluation process is shown. The figure is inspired by Raschka, 2018.

50 images from Flight 9 for which we had TIR and VIS records taken at the same second. To segment the VIS images, we used the publicly available OSSP algorithm from Wright and Polashenski, 2018, which was developed for this task. The OSSP algorithm provided images in TIFF format with four classes: gray ice, white ice, melt pond, and open water.

We used the following formula for both VIS and TIR:

$$\text{MPF} = \frac{\text{nr\_melt\_pond\_pixels}}{\text{nr\_sea\_ice\_pixels} + \text{nr\_melt\_pond\_pixels}}, \tag{3.6}$$

where we have merged gray ice and white ice pixels for the VIS images.

TABLE 3.1: Comparison of different patch sizes. Trainset and valset refer to the training and validation subsets during k-crossfold validation, testset to the test set used for final model evaluation. Step size corresponds to the stride at which the sliding window moves across the input image to extract patches. Note that the content descriptions are approximate observations and depend on the size of the features examined. Patch sizes, step sizes and overlap are given in pixels, the other numbers in images.

patch size	trainset size	valset size	testset size	content	step size	overlap	batch size
32 x 32	1350	450	450	individual melt ponds are entirely covered, most of the context cut out.	32	no	32
64 x 64	294	98	98	can cover multiple melt ponds, smaller floes can be detected but are not fully captured.	68	no	16
128 x 128	54	18	18	can fully cover smaller floes and parts of pond networks.	160	no	8
256 x 256	24	8	8	can cover multiple smaller floes and shapes of larger floes, although not fully captured. Networks of ponds are covered.	224	32	4
480 x 480	6	2	2	covers large parts of larger floes and ocean areas.	no	no	2

Chapter 4

Results and Discussion

In this chapter, we present our results. First, we discuss our observations from experimenting with different hyperparameters to find the optimal network configuration (Section 4.1). Then we try to evaluate the final performance of the model (Section 4.2). We do this through quantitative evaluation on an independent test set, qualitative inspection of prediction results, and evaluation of the final melt pond product.

We provide additional training and validation curves in Appendix B.

4.1 Model Selection

As introduced in Section 3.7.2, we used 4-crossfold validation for model selection with the validation mIoU of the best average epoch as selection criterion. The model converged in each training run unless otherwise noted.

4.1.1 Patch Size

We first investigated to choose the optimum patch size for model training. The results can be found in Table 4.1 and Figure 4.1.

We found a positive correlation between larger patch size and segmentation performance, with the exception of the patch size of 256 x 256, which may have suffered from redundant information due to overlapping patches. Especially the performance of the ocean class increased with a larger receptive field (Δ=0.207, comparing the patch size of 480 x 480 to 32 x 32). This is probably due to the fact that the ocean usually covers larger areas, which can only be partially captured by a small patch size.

Patch sizes 128 x 128 and 480 x 480 had similarly high performance with stable validation curves. The validation curve of 480 x 480 converged more slowly, which may be due to the smaller batch size. We selected 480 x 480 for further experiments because of a higher best average mIoU. In addition, 128 x 128 was disfavored because less information was used due to patch extraction with a large step size (Table 3.1).

Our results indicate the importance of context to correctly identify the surface features. In contrast to Iglovikov, Mushinskiy, and Osin, 2017, we achieved superior performance even with a very small batch size of 2 in the case of 480 x 480. One possible reason for this is that much of the variance of the features is already contained in individual images, and thus enough diversity is seen before a weight update is performed.

TABLE 4.1: Comparison of different patch sizes with rounded values. We colored the column that served as selection criterion. Best results are in bold. 'epoch 99' refers to the end of training. 'std' = standard deviation, where lower is better. All values except for those in column 'epoch 99' are obtained from the best average epoch in terms of validation mIoU.

PATCH SIZE	train mIoU	val mIoU			val melt pond IoU	val sea ice IoU	val ocean IoU
	best average	best average (epoch)	epoch 99	std best average	best average	best average	best average
32	0.955	0.555 (65)	0.528	0.083	0.373	0.873	0.419
64	0.907	0.662 (23)	0.607	0.116	0.489	0.919	0.579
128	0.939	0.729 (34)	0.707	0.123	**0.57**	**0.935**	0.682
256	0.874	0.659 (14)	0.579	0.140	0.479	0.846	0.651
480	0.927	**0.762** (59)	**0.737**	0.083	0.558	0.929	**0.797**

4.1.2 Loss Function

Next, we tested the weighted focal dice loss function introduced in Section 3.5.2. Previous runs with categorical cross-entropy loss showed a large disparity in model performance across classes, indicating sensitivity to class imbalance (Table 4.1). In previous runs, the sea ice class, which has the largest number of pixels, performed best, while the melt pond class performed poorly.

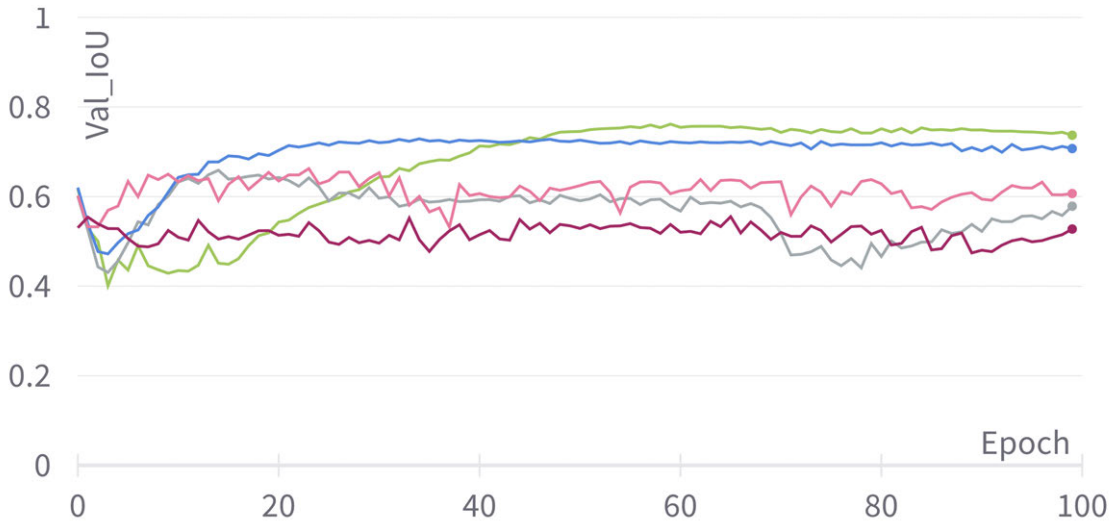

FIGURE 4.1: Comparison of the validation curves of different patch sizes, averaged over all crossfold subsets. Light green = 480, gray = 256, light blue = 128, pink/rose = 64, dark red = 32.

Using the weighted focal dice loss, we observed an overall performance improvement (Δ=0.035), which was particularly present for melt pond IoU (Δ=0.072, Table 4.2). Sea ice performance was slightly worse due to less emphasis on this class during training. We continued to use the weighted focal dice loss function for further experiments.

TABLE 4.2: Comparison of different loss functions with rounded values. 'CCE' = categorical cross-entropy loss, 'CFDL' = categorical focal dice loss. For more information refer to the caption of 4.1.

LOSS	train mIoU	val mIoU			val melt pond IoU	val sea ice IoU	val ocean IoU
	best average	best average (epoch)	epoch 99	std best average	best average	best average	best average
CCE	0.927	0.762 (59)	0.737	0.083	0.558	**0.929**	0.797
CFDL	0.945	**0.797** (37)	**0.751**	0.109	**0.630**	0.916	**0.825**

4.1.3 Dropout

We considered dropout layers because overfitting was present in all previous training runs. The results (Table 4.3) show that dropout was successful: Training performance decreased slightly due to information loss during training (Δ=-0.015), while validation performance increased (Δ=0.025). Therefore, we kept dropout for further experiments.

TABLE 4.3: Comparison of training with and without dropout. For more information refer to the caption of 4.1.

DROP-OUT	train mIoU	val mIoU			val melt pond IoU	val sea ice IoU	val ocean IoU
	best average	best average (epoch)	epoch 99	std best average	best average	best average	best average
no dropout	0.945	0.797 (37)	0.751	0.109	0.630	0.916	0.825
dropout	0.93	**0.822** (54)	**0.807**	0.106	**0.679**	**0.941**	**0.846**

4.1.4 Pre-training Strategy

Table 4.4 shows our results when experimenting with the pre-training strategy. When we froze the encoder, performance decreased compared to the fine-tuning strategy (Δ=-0.035). This suggests that ImageNet and melt pond images are too dissimilar to use the same feature extractor, and fine-tuning is needed. However, both pre-training strategies on ImageNet yielded much better results than random initialization. When we trained the model from scratch, mean performance decreased significantly (Δ=-0.11, when comparing to fine-tuning) and melt pond IoU showed an even higher drop (Δ=-0.212). We kept the fine-tuning strategy for further experiments.

TABLE 4.4: Comparison of different pre-training settings. 'fine-tuning' = all layers are set trainable during training. 'encoder freeze' = only decoder is set trainable. 'from scratch' = all weights are randomly initialized. For more information refer to the caption of 4.1.

PRE-TRAINING	train mIoU	val mIoU			val melt pond IoU	val sea ice IoU	val ocean IoU
	best average	best average (epoch)	epoch 99	std best average	best average	best average	best average
fine-tuning	0.93	**0.822** (54)	**0.807**	0.106	**0.679**	**0.941**	**0.846**
encoder freeze	0.904	0.787 (32)	0.769	0.115	0.63	0.924	0.809
from scratch	0.939	0.712 (95)	0.709	0.178	0.467	0.909	0.760

4.1.5 Augmentation Techniques

Next, we explored the effectiveness of different augmentation techniques in on-the-fly mode (Table 4.5).

After adding brightness contrast augmentation, we observed that the training curve remained at a very low level and was highly fluctuating (Figure 4.2). The same was true for Gaussian noise transformation. This may indicate that the added variations were too strong, preventing the model from learning meaningful patterns from the data.

For cropping and rotation changes, the model converged but still produced an unstable validation curve at a lower performance level (Figure 4.2). The cropping factor may have been too large to successfully predict validation data, because all of the images considered were taken from a similar flight altitude and variations in feature size may not have been significant. Extrapolation may have affected rotation performance.

Only sharpen blur could slightly improve the validation performance compared to the setting without augmentation (Δ=0.008) and was kept for the final model evaluation. The training performance became slightly worse (Δ=-0.034), indicating the method's success in overfitting. However, the improvements remain small, making augmentation relatively unimportant for this work.

TABLE 4.5: Comparison of different augmentation techniques. 'flip' comprises horizontal and vertical flipping. For more information refer to the caption of 4.1.

AUGMENTATION TECHNIQUE	train mIoU	val mIoU			val melt pond IoU	val sea ice IoU	val ocean IoU
	best average	best average (epoch)	epoch 99	std best average	best average	best average	best average
flip	0.873	0.778 (71)	0.739	0.054	0.586	0.920	0.827
rotate	0.858	0.784 (58)	0.7	0.064	0.594	0.932	0.826
crop	0.867	0.758 (56)	0.578	0.095	0.576	0.917	0.781
brightness contrast	0.538	0.657 (53)	0.413	0.084	0.458	0.842	0.670
sharpen blur	0.896	**0.83** (50)	**0.811**	0.09	**0.685**	**0.94**	**0.864**
Gaussian noise	0.523	0.547 (50)	0.457	0.09	0.377	0.75	0.515

4.1.6 Augmentation Design

We compared on-the-fly and offline augmentation using only sharpen blur, as it was the only technique that showed performance improvement in the previous experiment.

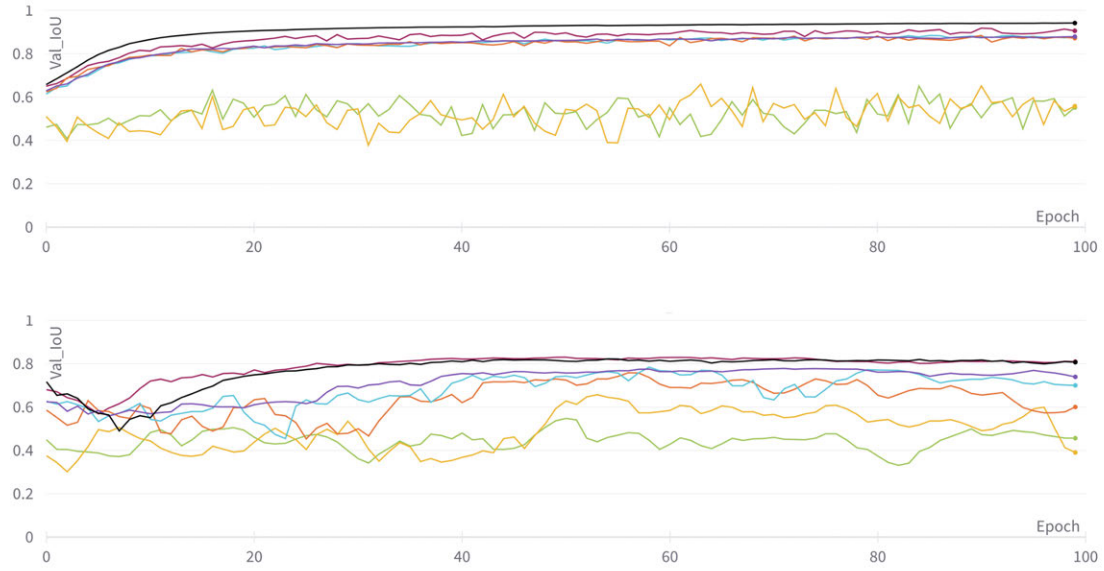

FIGURE 4.2: The top part shows a comparison of the training curves of different augmentation techniques, averaged over all crossfold subsets. The bottom part shows corresponding validation curves. Black = no augmentation, purple = horizontal and vertical flip, dark red = sharpen blur, light blue = crop, orange = rotate, yellow = brightness contrast, light green = Gaussian noise.

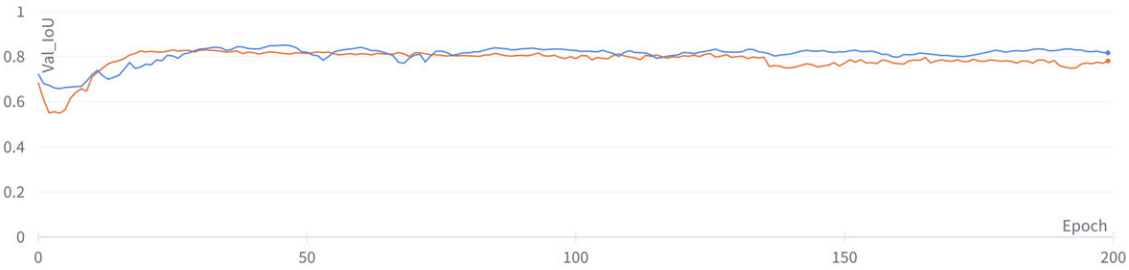

FIGURE 4.3: Comparison of the validation curves of on-the-fly augmentation (blue) and offline augmentation (orange), averaged over all crossfold subsets. Augmentation technique applied was sharpen blur.

For offline augmentation, we set the size of the dataset increase to a factor of 2 to avoid duplicates in the training set. The final offline training set included an equal number of original and augmented images. This maintained a similar proportion as in on-the-fly mode, where each image had a 0.5 probability to be augmented. Additionally, we increased the number of epochs to 200 to observe further variations beyond our previous examination.

The results are shown in Table 4.6 and Figure 4.3. In terms of performance, no significant difference could be observed. We decided to keep on-the-fly augmentation because of its slightly higher validation mIoU (Δ=0.022) and its better ability to mitigate overfitting. No noticeable changes occured after 100 epochs.

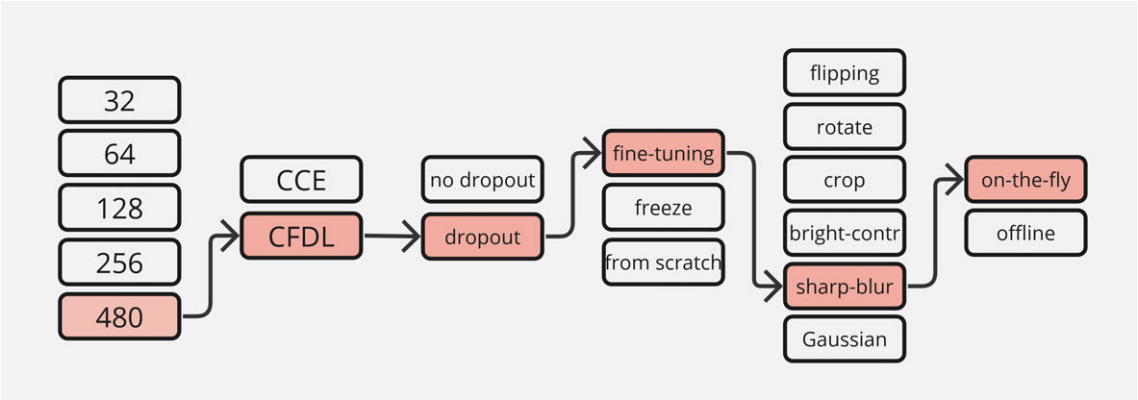

FIGURE 4.4: The hyperparameter optimization process with selected values in red. Hyperparameters from left to right: patch size, loss function, dropout, pre-training strategy, augmentation technique, augmentation design. 'CCE' = categorical cross-entropy loss, 'CFDL' = categorical focal dice loss, 'freeze' = encoder freeze, 'bright-contr' = brightness contrast, 'sharp-blur' = sharpen blur.

TABLE 4.6: Comparison of different augmentation designs. For more information refer to the caption of 4.1.

AUGMENTATION DESIGN	train mIoU	val mIoU			val melt pond IoU	val sea ice IoU	val ocean IoU
	best average	best average (epoch)	epoch 199	std best average	best average	best average	best average
on-the-fly	0.907	**0.851** (46)	**0.816**	0.108	**0.727**	**0.946**	**0.879**
offline	0.935	0.829 (25)	0.781	0.201	0.676	0.951	0.860

A summarizing overview of the entire selection process is shown in Figure 4.4.

4.1.7 Training Stability

We attempted to assess the robustness of our evaluation method by examining the variance across different crossfold subsets. For all hyperparameter configurations, validation IoU had a relatively high standard deviation with respect to different validation sets (Tables 4.1, 4.2, 4.3, 4.4, 4.5, 4.6). This suggests that the performance was dependent on the particular dataset split. Figure 4.5 shows that this is mainly due to the first subset: When this was chosen for validation, performance was particularly poor. This suggests a shift in the distribution of images in this set compared to the others, which is also indicated by a slight difference in class proportions (Appendix A).

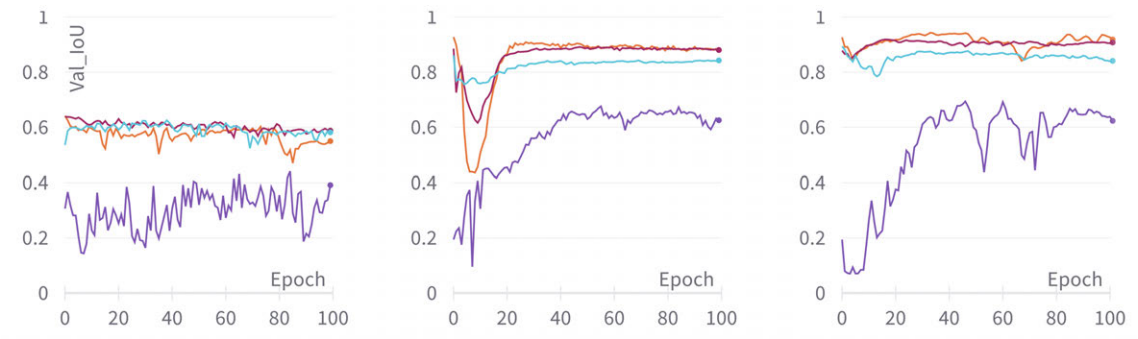

FIGURE 4.5: Validation performance of different dataset splits in crossfold validation. Each plot shows the performance of a model configuration, where each line was produced by evaluating with a different validation subset, with the same splits across different configurations. Purple = subset 1, light blue = subset 2, dark red = subset 3, orange = subset 4. The model configurations shown are randomly chosen. Left: patch size 32, middle: patch size 480 with categorical focal dice loss and dropout layers, right: same configuration as in the middle plot with on-the-fly sharpen blur augmentation added. Note that the x-axis was squeezed compared to other plots for visibility reasons.

4.2 Final Model Construction

To obtain a final model from the optimized hyperparameter configuration, we trained our network on the entire crossfold dataset and evaluated it on the unseen test images. We trained for 500 epochs. In previous experiments, extended training did not result in noteworthy changes. However, for the final run, we were able to accommodate the required computational resources and still saved the model weights from the epoch with the minimum validation loss. The results are shown in Table 4.7 and Figure 4.6. The final test mIoU is much lower than the validation mIoU from previous experiments (Δ=-0.152). This may indicate that we overfitted our validation sets during hyperparameter optimization.

TABLE 4.7: Final model performance after training the best configuration for 500 epochs and evaluation on an independent test set. Values refer to epoch 85, as this was the epoch with minimal test loss, from which the final weights are stored.

FINAL MODEL	train mIoU	test mIoU	test melt pond IoU	test sea ice IoU	test ocean IoU
	0.896	0.699	0.477	0.932	0.688

4.2.1 Qualitative Results

We present some prediction results in Figure 4.7.

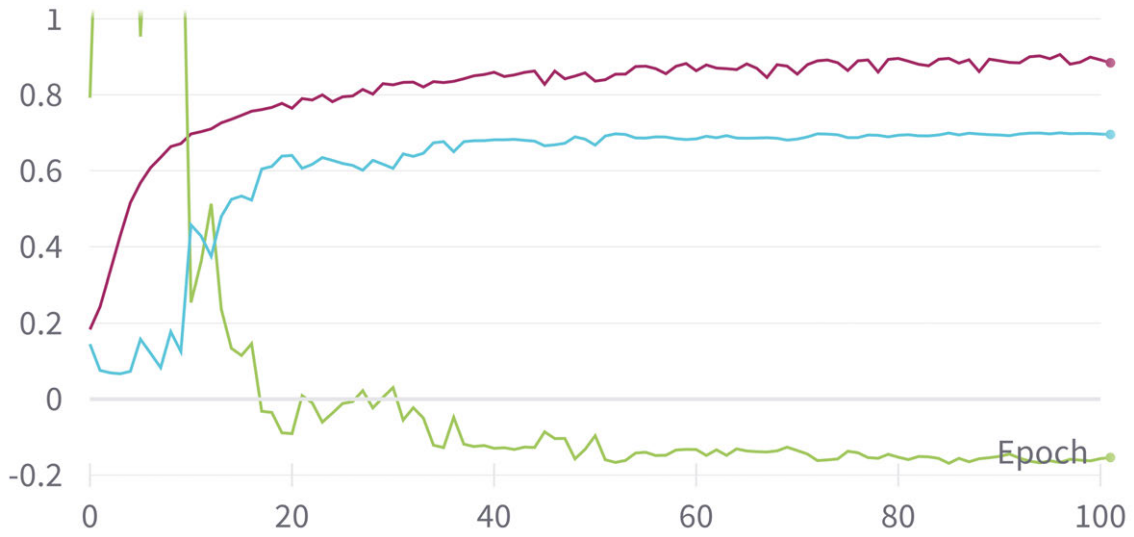

FIGURE 4.6: Test loss (light green), training mIoU (dark red), and test mIoU (blue) for the final model configuration. Note that the loss extends the plot scale in the early epochs.

Qualitative evaluation reveals the weaknesses of our model. Poor results are obtained for images with ice floes that are warmer than the ocean area (fourth row, left). Here, the model seems to have learned from the temperature ordering and predicted warmer areas (lighter) as ocean and colder areas (darker) as ponds and ice. From Flight 16, samples with small temperature contrasts between ice and ocean could not be predicted correctly.

We attribute the poor prediction performance to the small training dataset with limited feature diversity, as images from Flight 16 and samples with reversed temperatures were not seen during training. Images similar to those in the training dataset were already predicted relatively well, with most errors occurring in the confusion of submerged ice and small areas of ocean between floes (bottom row, left; second row, left).

4.2.2 Melt Pond Fraction Results

The results of computing melt pond fraction from predicted TIR and corresponding VIS images are shown in Table 4.8, top part. The following must be considered for the interpretation of the values: (1) The VIS images show a larger area than the TIR segmentation maps (Figure 3.2), where the latter are additionally center-cropped. TIR image distortions affect comparability. (2) The VIS classification algorithm itself is subject to uncertainties. (3) Differences between the sensors in the ability to detect melt ponds (Section 4.4).

These uncertainties, together with the observation of large prediction errors in the

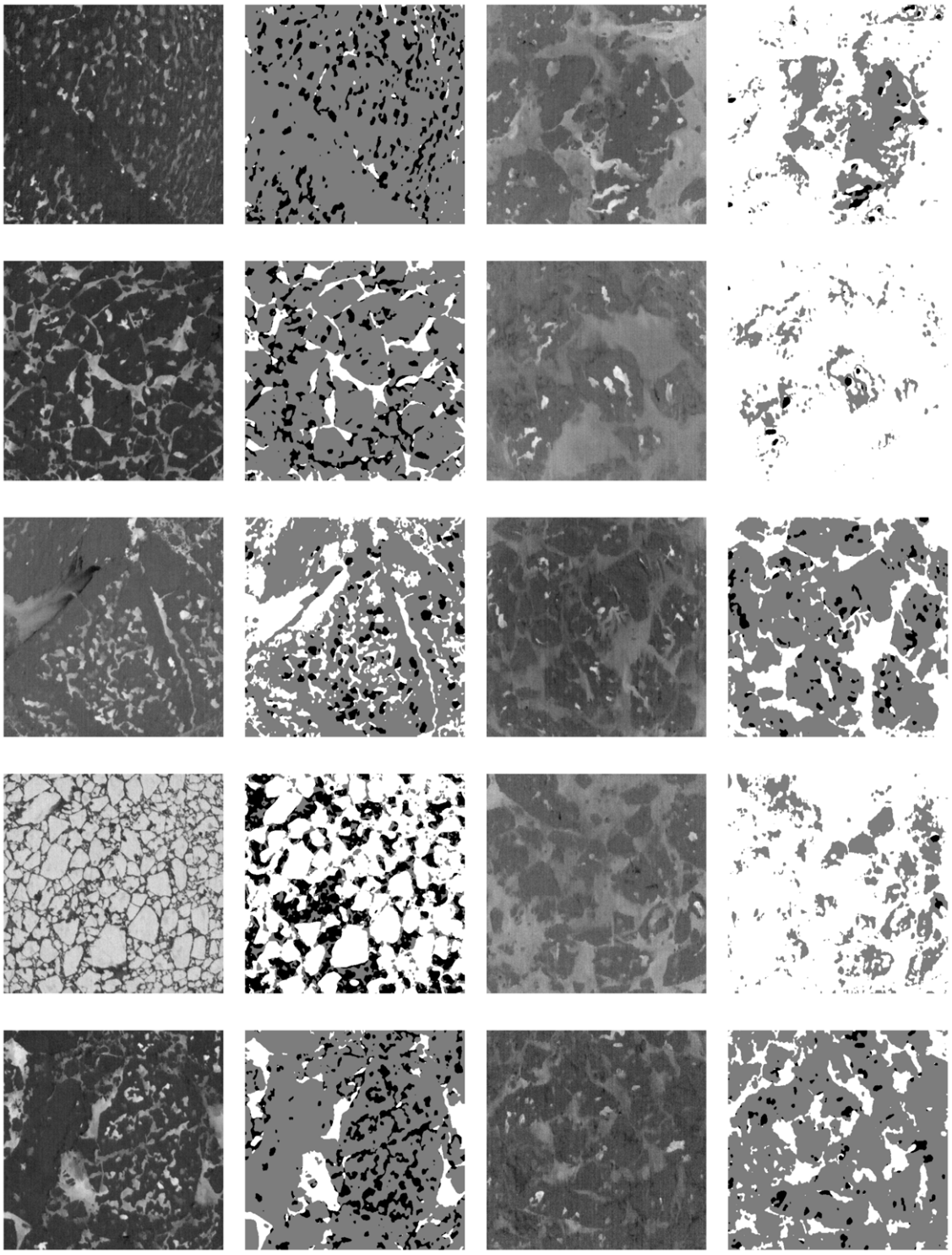

FIGURE 4.7: Qualitative results of the final segmentation model for selected images. Black = melt pond, gray = sea ice, white = ocean. Columns from left to right: model input Flight 9, prediction Flight 9, model input Flight 16, predictions Flight 16. In input imagery, lighter regions are warmer and darker regions colder. Location: Fram Strait region. Date: July 18, 2022 (Flight 9) and July 30, 2022 (Flight 16).

TABLE 4.8: The top part shows MPF computed from 50 randomly selected VIS images (left) and corresponding TIR images predicted by our final model (right). The bottom part shows MPF computed from 10 cropped VIS images (left) and corresponding annotated TIR images (right). The classification of the VIS images was performed using the OSSP algorithm of Wright and Polashenski, 2018.

MPF VIS (classified)	MPF TIR (classified)
0.1754	0.1451

MPF VIS (cropped, classified)	MPF TIR (annotated)
0.1992	0.1445

previous evaluation, make it difficult to correctly interpret the MPF result in terms of new insights into model performance. Nevertheless, we have included it in our work so that it can be refined and used in the future as an additional evaluation method and, in the final use case, to compute MPF from TIR.

4.3 Ablation Study: Border Effects

When examining the predictions of smaller patch sizes, we found that additional processing was required to produce more accurate results. In this section, we discuss the problems we observed and how we accounted for them.

Since we fixed the model input size before training, segmentation for smaller patch size configurations was done by using crops of the image to be predicted. We initially used a simple procedure of extracting patches, feeding them into the model, and concatenating predictions to reach the original image size. This led to problems: (1) Patch sizes of 64, 128, and 256 could not be used for full image predictions because they are not divisors of the original image size, and (2) border effects resulted in square structures in the final image (Figure 4.8, middle). This is due to missing context in the border regions, which leads to inaccurate predictions in these areas (Iglovikov, Mushinskiy, and Osin, 2017).

To address both problems, we integrated a function that smoothly blends segmentation masks by predicting class values for overlapping squares and using 2D interpolation to determine the final prediction. The original code with more details can be found at `https://github.com/Vooban/Smoothly-Blend-Image-Patches`. We continue to refer to this function as patch stitching function.

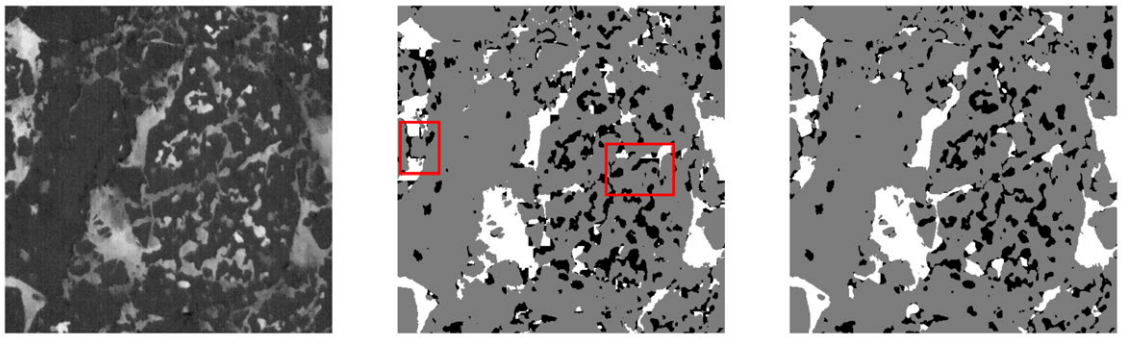

FIGURE 4.8: Example of border effects for patch size 32 (middle), corresponding model input (left), and prediction postprocessed by the patch stitching function (right). Red rectangles show examples of square structures. Prediction was performed using the 32 x 32 patch size configuration with weights from crossfold run 3 due to relatively high performance.

We observed improved results (Figure 4.8, right). Since a smaller patch size has more borders in total, and we computed validation performance without incorporating postprocessing, evaluation in Section 4.1.1 may have been pessimistically biased with respect to smaller patch sizes.

4.4 Ablation Study: Comparison of Melt Pond Detection between TIR and VIS

In Section 3.2 we observed that VIS images show more ponds than TIR. This discrepancy influenced our evaluation method in Section 4.2.2 and is important to be aware of when using TIR for MPF retrieval.

We compared MPF in classified VIS with all annotated TIR images to estimate the difference in melt pond detection due to the sensor. This time, we manually cropped the VIS images to match the approximate region of the annotated TIR images. We could afford to do this because the number of images was relatively small. Otherwise, we followed the same procedure as in Section 3.7.3. The results are shown in Table 4.8, bottom part, and provide a numerical evidence base for underrepresentation of melt ponds in TIR that needs to be considered in future work.

Chapter 5

Summary and Conclusion

In this final chapter, we summarize our experimental findings and give directions for the future.

The goal of this thesis was to develop a segmentation method that separates helicopter-borne TIR images into melt pond, sea ice, and ocean classes. We decided to use a U-net model, which can automatically extract multiple features and is known to be able to learn from limited training data. We tested different hyperparameters, namely patch size, loss function, dropout, transfer learning, and augmentation. Then we constructed a final model with the best configuration.

The primary limitation that we faced was the small number of annotated images in our dataset, resulting from the time-consuming process of manual labeling. As a consequence, our model experienced overfitting. We observed limited generalization ability on images dissimilar to those seen during training, such as cases when sea ice was warmer than the surrounding ocean area or images from a different flight. In addition, small data affected the reliability of our evaluation method. During hyperparameter optimization, we found divergent performance outcomes depending on the specific validation set used, indicating that different sets could not represent the real data distribution. Our final model achieved a test mIoU of 0.7, although this value is likely to vary when different data is used for testing.

Despite the challenges, we remain hopeful for using U-net as segmentation tool for the task at hand. Successful predictions in some cases demonstrate the potential of the method when annotation proceeds.

For reference in the future, we summarize our findings on hyperparameters in the following.

- Larger patch sizes were more successful than smaller ones, suggesting that context helps to accurately detect melt ponds. Another possible reason for this pattern is that border effects may have pessimistically biased performance for smaller patch sizes. We found that border effects can be improved using an overlapping patch stitching function as a postprocessing step.

- Regarding the patch extraction method, retrieving overlapping patches for training set creation seemed to be detrimental to performance, possibly due to redundant information.

- Melt pond and ocean classes were poorly detected compared to sea ice, likely due to the class imbalance in the training dataset. Some improvement was achieved by using a weighted loss function.

- Using dropout layers in the decoder could help against overfitting.

- Consistent with the findings of Hu et al., 2015, Yosinski et al., 2014, and Lima and Marfurt, 2019, pre-training on ImageNet proved to be successful, even though our dataset is very dissimilar. This shows the generalization power of ImageNet even for very different tasks. Fine-tuning resulted in better performance than using a fixed feature extractor.

- Sharpen blur was the only augmentation technique that could lead to a slight performance improvement, while Gaussian noise and brightness contrast changes were likely too unrealistic, resulting in unsuccessful training. We observed that the choice between on-the-fly and offline augmentation did not make a large difference for our dataset.

We also found a discrepancy between the detection of melt ponds using TIR and VIS sensors, with VIS showing more melt ponds. It is important to consider this difference when using TIR data for accurate MPF retrieval in the future.

Future directions are discussed in the next, final section.

5.1 Future Directions

To improve model performance, the first and most crucial step is to create a larger and more diverse annotated dataset. This includes images from different flights and, if available, from different seasons and areas. To speed up the labeling process, we propose to consider semi-supervised approaches such as self-training (Amini et al., 2023). This could be done by using our model to predict labels for unlabeled images, adding high confidence predictions to the original labeled training data, and retraining the model. By iterating this approach several times, we may be able to gradually improve performance.

Furthermore, it is important to include georeferencing either before or after model training to eliminate image distortions.

To enhance the model's generalizability, further investigation into appropriate augmentation techniques could be done. If simple methods continue to fail, more advanced techniques may be encountered. One option would be elastic deformations, which introduce local transformations by applying random displacements to image pixels (Simard, Steinkraus, Platt, et al., 2003). This may be able to mimic differences in melt pond shapes. Elastic deformations are used extensively in the original U-net.

In addition, we suggest a number of modifications that could be tested as part of our experimental pipeline. Currently, the data is prepared by mapping the temperature values to matplotlib's 'gray' colormap and loading them as grayscale for model training. This may have adversely affected the pixel values. An alternative option would be to convert the temperatures to 256-level grayscale values and directly train on the resulting images instead of saving and colormapping them as an intermediate step.

As a further preprocessing step, normalization can increase model performance. It can also help to encounter scale and distribution shifts across different flights.

During hyperparameter optimization, we chose a relatively high dropout probability of 0.5, which could be tested for smaller values.

Furthermore, to better estimate the dependence on image context, configurations for smaller patch sizes should be evaluated after processing masks with the patch stitching function.

While this work exclusively considered U-net, future work could explore other architectures. Attention mechanisms may be an option to help focus on more relevant features

(Fu et al., 2018; Ren et al., 2022).

Ultimately, the goal of achieving accurate segmentation of TIR images remains an ongoing research area and opens avenues for better understanding the characteristics of Arctic sea ice during the summer.

5.2 Acknowledgements

I would like to thank my supervisors Ulf and Gunnar for their support and guidance throughout this work. I would like to extend my appreciation to Peter, Miguel and my brother Linus for technical support, and to my father for supplying his computer for model training. I am grateful to Lena Buth from the Alfred Wegener Institute for providing the VIS images.

A special thanks goes to the whole Polar Remote Sensing group at the University of Bremen for welcoming me into their group and giving me insight into a fascinating field of research.

Appendix A

Additional Material

Annotation Procedure

Image preparation (`data_preparation/extract.ipynb`)[1]:

- Clip TIR temperature range to 273 - 276 Kelvin to increase contrast
- Use matplotlib's 'cividis' colormap for better visibility

Edge map preparation (`data_preparation/edge_detection_annotation.ipynb`):

- Resize image to 2345 x 3210 for better visibility
- Show different thresholding methods from skimage applied to the image and select the best by manual inspection (in our case mostly 'mean' and 'otsu')
- Binarize the image using the selected filter
- Apply Scharr operator from skimage and convert all non-zero values in the image to 255 to create a clearer mask

Annotation (The GIMP Development Team, 2019):

- Load image into GIMP software
- Add mask as extra layer
- Make edge map background transparent with Colors > Color to Alpha
- Manually correct edges with pencil and eraser tool and fill object with class-specific color (black - melt pond, gray - sea ice, white - ocean)
- Export mask as PNG file

Make sure that mask only contains three color values in the end.

[1] `https://github.com/marlens123/ponds`

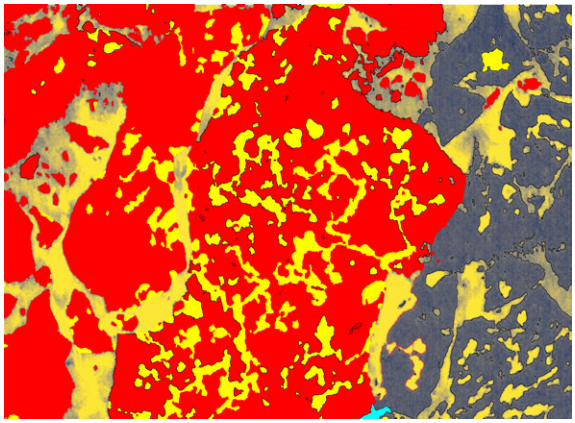

FIGURE A.1: Intermediate mask while editing in GIMP. Note that red pixels are converted to gray, turquoise to white and yellow to black in the end.

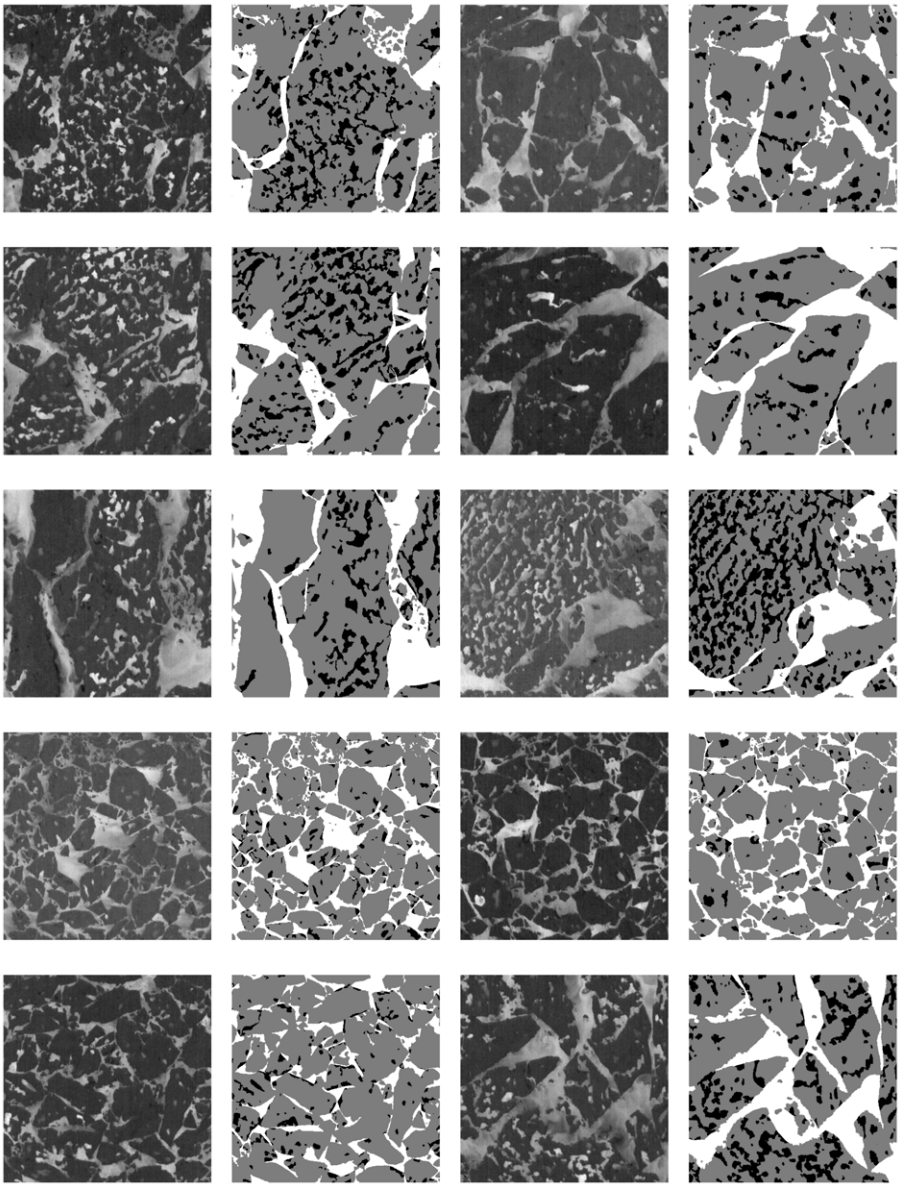

FIGURE A.2: The entire dataset with training data (including all crossfold subsets) in the upper 4 rows and test data (used for final model evaluation) in the lowest row. Images and masks are already center-cropped.

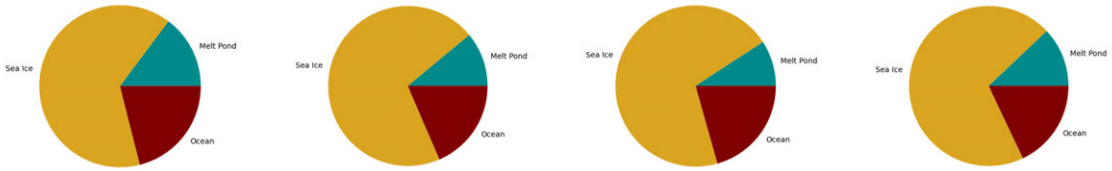

FIGURE A.3: Class distributions for different crossfold subsets. From left to right: fold 1 - fold 2 - fold 3 - fold 4.

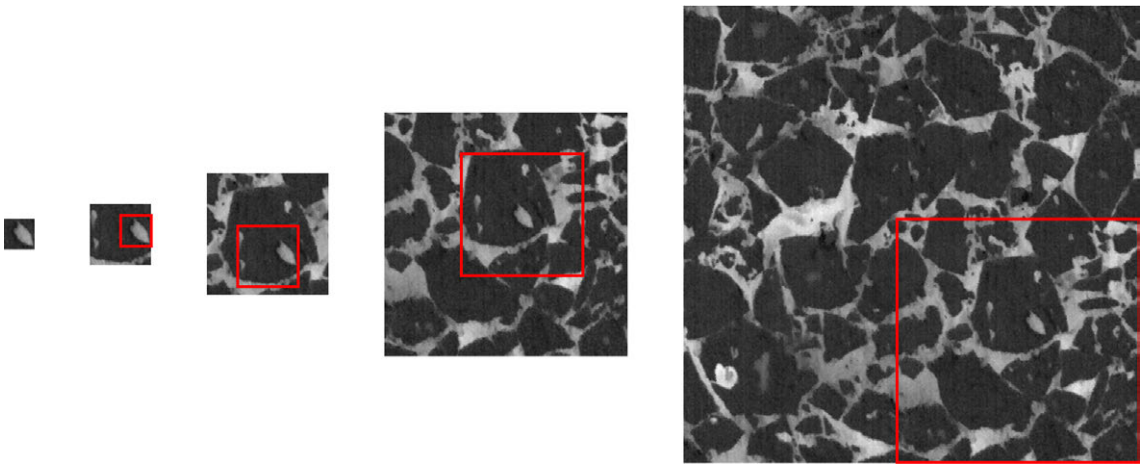

FIGURE A.4: Comparison of different patch sizes. From left to right: 32 x 32, 64 x 64, 128 x 128, 256 x 256, 480 x 480. The red rectangles mark the next smaller patch size. Note that the image sections are chosen for visualization purposes and do not necessarily correspond to the real training data.

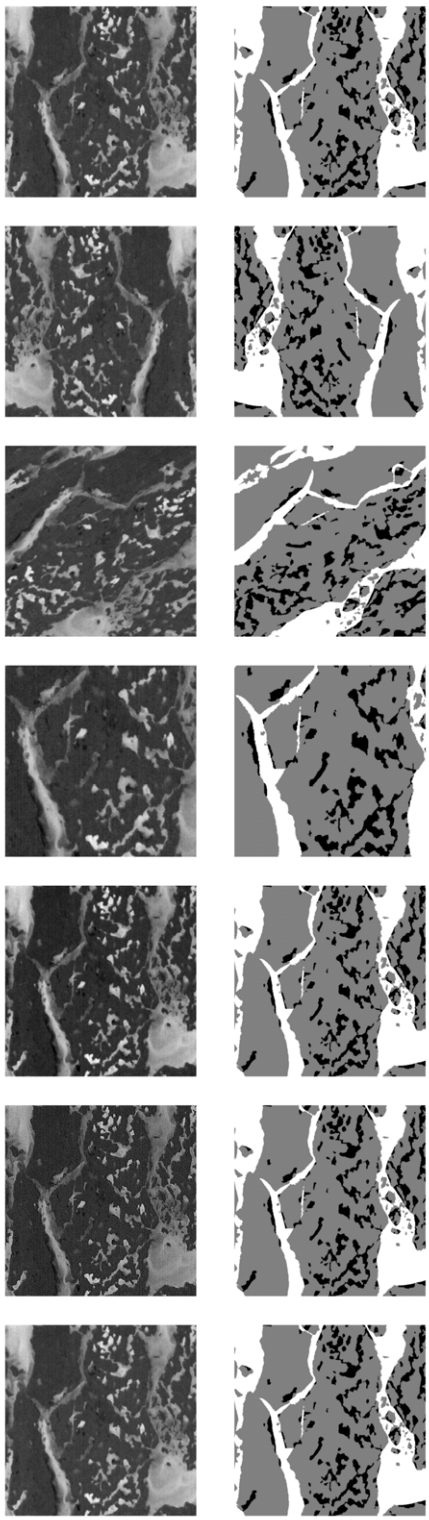

FIGURE A.5: Comparison of different augmentation techniques. From top to bottom: original image, flipping, rotation, cropping, brightness contrast, sharpen blur, Gaussian noise.

Appendix B

Evaluation Plots

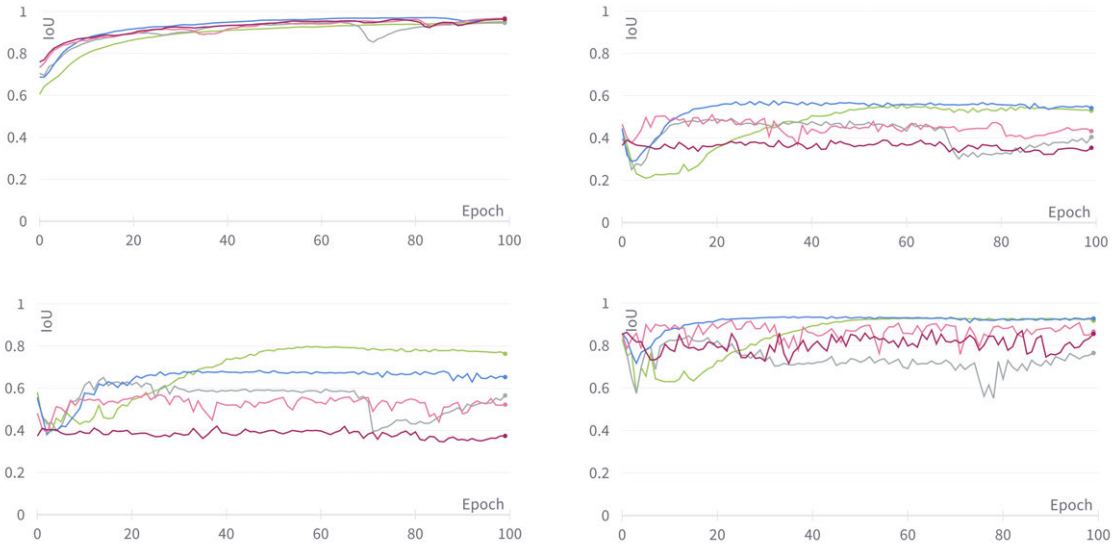

FIGURE B.1: Comparison of different patch sizes. From top left to bottom right: mean IoU, validation melt pond IoU, validation ocean IoU, validation sea ice IoU. Light green = 480, gray = 256, blue = 128, pink/rose = 64, dark red = 32.

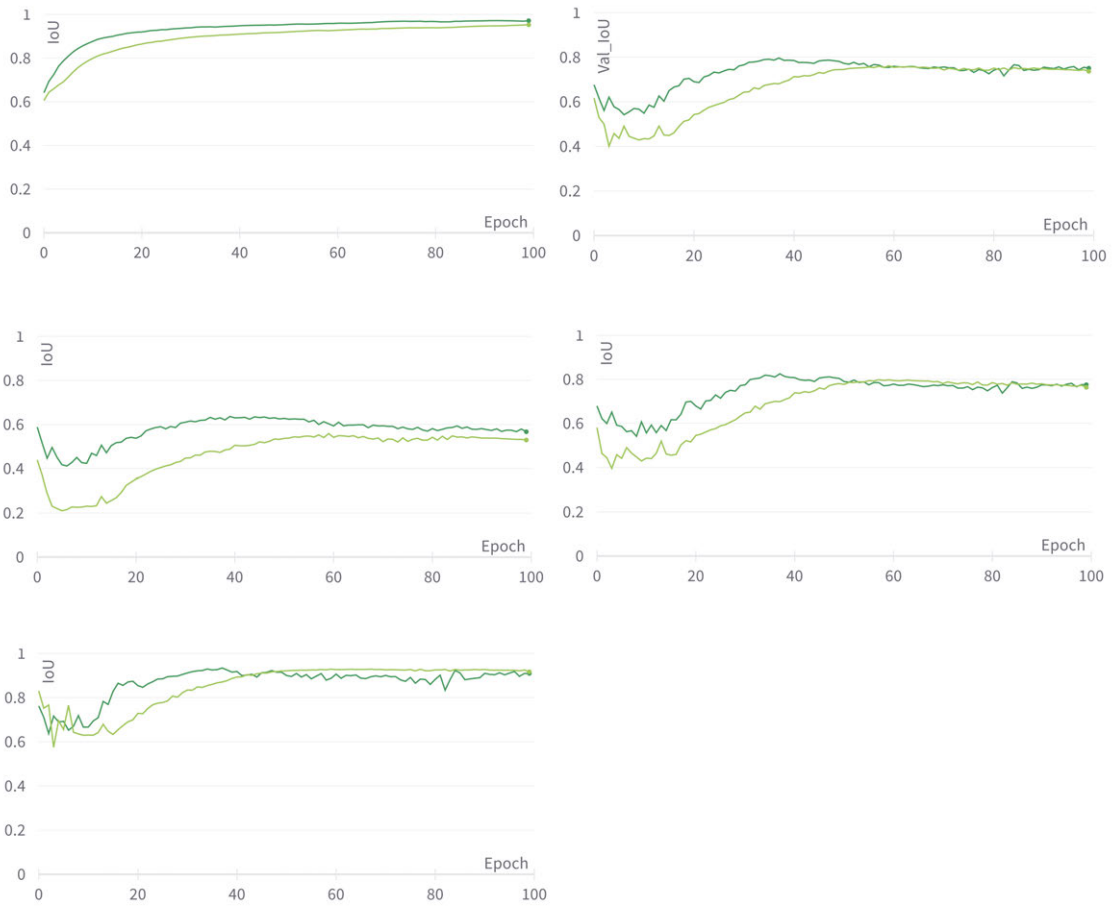

FIGURE B.2: Comparison of different loss function. From top left to bottom left: mean IoU, validation mean IoU, validation melt pond IoU, validation ocean IoU, validation sea ice IoU. Light green = CCE, dark green = CFDL.

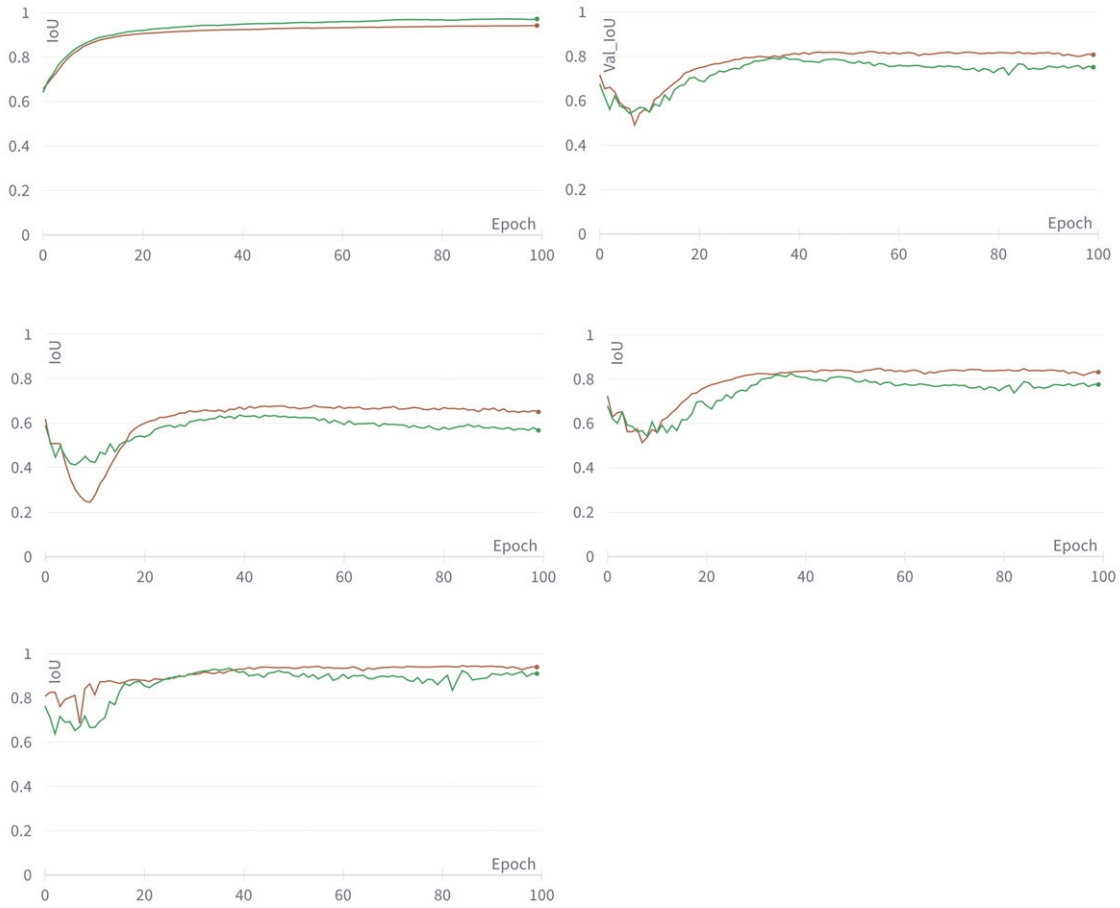

FIGURE B.3: Comparison of dropout versus no dropout. From top left to bottom left: mean IoU, validation mean IoU, validation melt pond IoU, validation ocean IoU, validation sea ice IoU. Brown = dropout, green = no dropout.

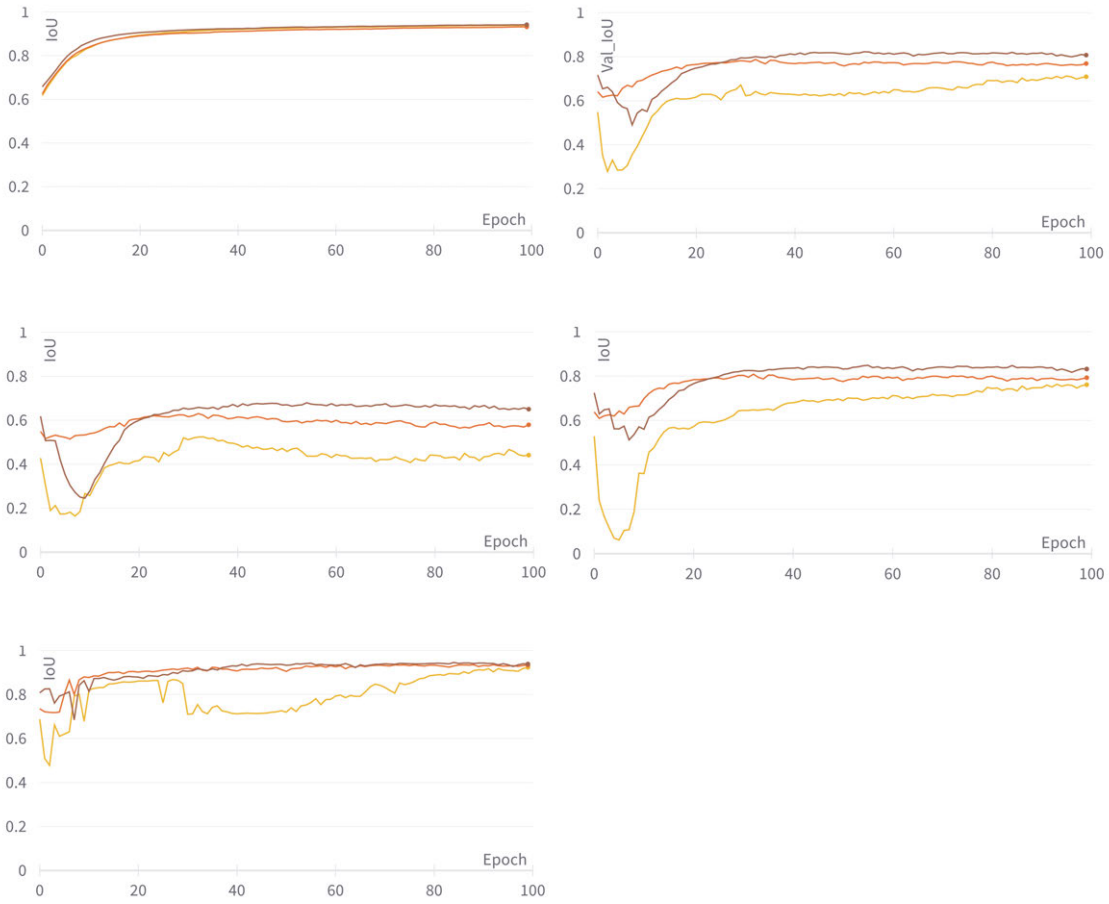

FIGURE B.4: Comparison of different pre-training strategies. From top left to bottom left: mean IoU, validation mean IoU, validation melt pond IoU, validation ocean IoU, validation sea ice IoU. Yellow = from scratch, brown = fine-tuning, orange = encoder freeze.

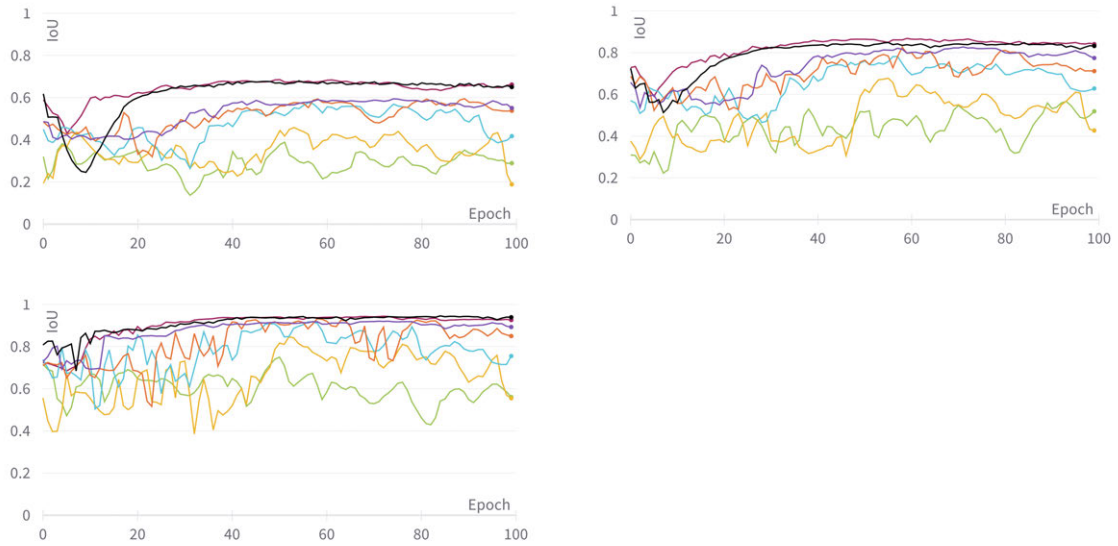

FIGURE B.5: Comparison of different augmentation techniques. From top left to bottom left: validation melt pond IoU, validation ocean IoU, validation sea ice IoU. Black = no augmentation, dark red = sharpen blur, purple = flip, orange = rotate, blue = crop, light green = Gaussian noise, yellow = brightness contrast.

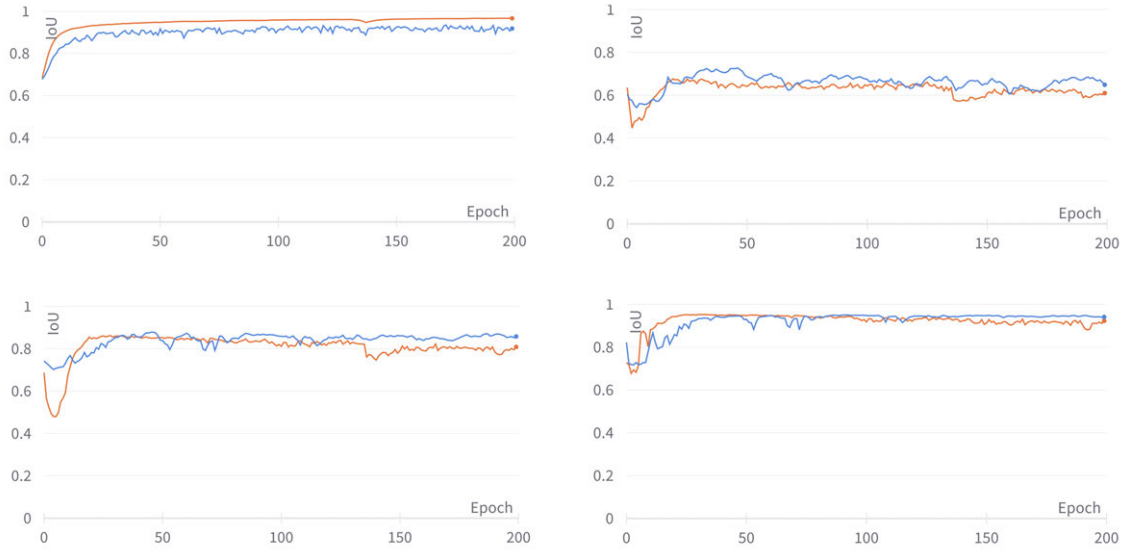

FIGURE B.6: Comparison of different augmentation designs. From top left to bottom right: mean IoU, validation melt pond IoU, validation ocean IoU, validation sea ice IoU. Blue = on-the-fly, orange = offline.

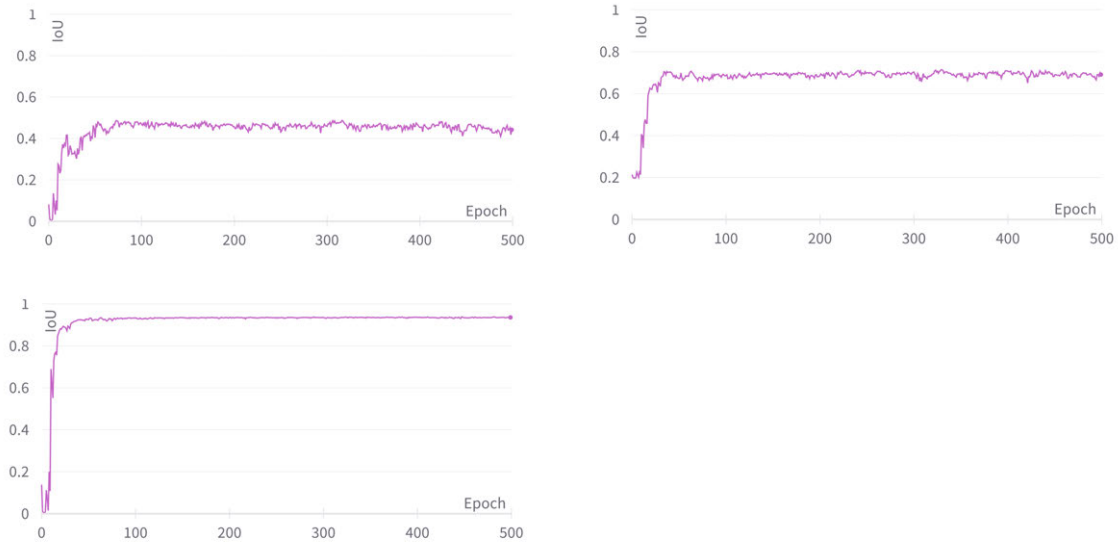

FIGURE B.7: Final run evaluation. From top left to bottom left: test melt pond IoU, test ocean IoU, test sea ice IoU.

Bibliography

Abadi, Martín et al. (2016). "Tensorflow: A system for large-scale machine learning". In: *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283.

Amini, Massih-Reza et al. (2023). *Self-Training: A Survey*. arXiv: 2202.12040 [cs.LG].

Benjamin MatvL, midaha PGibson RMcKinlay Wendy Kan (2016). *Dstl Satellite Imagery Feature Detection*. URL: https://kaggle.com/competitions/dstl-satellite-imagery-feature-detection.

Biewald, Lukas (2020). *Experiment Tracking with Weights and Biases*. Software available from wandb.com. URL: https://www.wandb.com/.

Buslaev, Alexander V. et al. (2018). "Albumentations: fast and flexible image augmentations". In: *CoRR* abs/1809.06839. arXiv: 1809.06839. URL: http://arxiv.org/abs/1809.06839.

Campbell, James B and Randolph H Wynne (2011). *Introduction to remote sensing*. Guilford Press.

Chollet, Francois et al. (2015). *Keras*. URL: https://github.com/fchollet/keras.

Cohen, Judah et al. (2020). "Divergent consensuses on Arctic amplification influence on midlatitude severe winter weather". In: *Nature Climate Change* 10.1, pp. 20–29.

Deng, Jia et al. (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Eguíluz, Victor M et al. (2016). "A quantitative assessment of Arctic shipping in 2010–2014". In: *Scientific reports* 6.1, pp. 1–6.

Eicken, H. et al. (2002). "Tracer studies of pathways and rates of meltwater transport through Arctic summer sea ice". In: *Journal of Geophysical Research: Oceans* 107.C10, SHE 22–1–SHE 22–20. DOI: https://doi.org/10.1029/2000JC000583. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2000JC000583. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2000JC000583.

Eicken, H. et al. (2004). "Hydraulic controls of summer Arctic pack ice albedo". In: *Journal of Geophysical Research: Oceans* 109.C8. DOI: https://doi.org/10.1029/2003JC001989. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2003JC001989. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2003JC001989.

Esteva, Andre et al. (2017). "Dermatologist-level classification of skin cancer with deep neural networks". In: *nature* 542.7639, pp. 115–118.

Fetterer, Florence and Norbert Untersteiner (1998). "Observations of melt ponds on Arctic sea ice". In: *Journal of Geophysical Research: Oceans* 103.C11, pp. 24821–24835. DOI: https://doi.org/10.1029/98JC02034. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/98JC02034. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/98JC02034.

Flocco, Daniela, Daniel L Feltham, and Adrian K Turner (2010). "Incorporation of a physically based melt pond scheme into the sea ice component of a climate model". In: *Journal of Geophysical Research: Oceans* 115.C8.

Flocco, Daniela et al. (2012). "Impact of melt ponds on Arctic sea ice simulations from 1990 to 2007". In: *Journal of Geophysical Research: Oceans* 117.C9.

Francis, Jennifer A and Stephen J Vavrus (2015). "Evidence for a wavier jet stream in response to rapid Arctic warming". In: *Environmental Research Letters* 10.1, p. 014005. DOI: 10.1088/1748-9326/10/1/014005. URL: https://dx.doi.org/10.1088/1748-9326/10/1/014005.

Fu, Jun et al. (2018). "Dual Attention Network for Scene Segmentation". In: *CoRR* abs/1809.02983. arXiv: 1809.02983. URL: http://arxiv.org/abs/1809.02983.

Fuchs, Niels (2023a). "A multidimensional analysis of sea ice melt pond properties from aerial images". dissertation. DOI: 10.26092/elib/2249.

— (Jan. 2023b). *PASTA-ice Github Repository https://github.com/nielsfuchs/pasta_ice*. Version v2023.01. DOI: 10.5281/zenodo.7548469. URL: https://doi.org/10.5281/zenodo.7548469.

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html.

Grenfell, Thomas C. and Gary A. Maykut (1977). "The Optical Properties of Ice and Snow in the Arctic Basin". In: *Journal of Glaciology* 18.80, 445–463. DOI: 10.3189/S0022143000021122.

Grenfell, Thomas C and Donald K Perovich (1984). "Spectral albedos of sea ice and incident solar irradiance in the southern Beaufort Sea". In: *Journal of Geophysical Research: Oceans* 89.C3, pp. 3573–3580.

— (2004). "Seasonal and spatial evolution of albedo in a snow-ice-land-ocean environment". In: *Journal of Geophysical Research: Oceans* 109.C1.

He, Kaiming et al. (2015a). "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385. arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

— (2015b). "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification". In: *CoRR* abs/1502.01852. arXiv: 1502.01852. URL: http://arxiv.org/abs/1502.01852.

Hoeser, Thorsten, Felix Bachofer, and Claudia Kuenzer (2020). "Object Detection and Image Segmentation with Deep Learning on Earth Observation Data: A Review—Part II: Applications". In: *Remote Sensing* 12.18. ISSN: 2072-4292. DOI: 10.3390/rs12183053. URL: https://www.mdpi.com/2072-4292/12/18/3053.

Hohenegger, C. et al. (2012). "Transition in the fractal geometry of Arctic melt ponds". In: *The Cryosphere* 6.5, pp. 1157–1162. DOI: 10.5194/tc-6-1157-2012. URL: https://tc.copernicus.org/articles/6/1157/2012/.

Hovelsrud, Grete K et al. (2011). "Arctic societies, cultures, and peoples in a changing cryosphere". In: *Ambio* 40, pp. 100–110.

Hu, Fan et al. (2015). "Transferring Deep Convolutional Neural Networks for the Scene Classification of High-Resolution Remote Sensing Imagery". In: *Remote Sensing* 7.11, pp. 14680–14707. ISSN: 2072-4292. DOI: 10.3390/rs71114680. URL: https://www.mdpi.com/2072-4292/7/11/14680.

Huang, W. et al. (2016). "Melt pond distribution and geometry in high Arctic sea ice derived from aerial investigations". In: *Annals of Glaciology* 57.73, 105–118. DOI: 10.1017/aog.2016.30.

Hunke, Elizabeth C, David A Hebert, and Olivier Lecomte (2013). "Level-ice melt ponds in the Los Alamos sea ice model, CICE". In: *Ocean Modelling* 71, pp. 26–42.

Hunter, John D (2007). "Matplotlib: A 2D graphics environment". In: *Computing in science & engineering* 9.3, pp. 90–95.

Iakubovskii, Pavel (2019). *Segmentation Models*. https://github.com/qubvel/segmentation_models.

Iglovikov, Vladimir, Sergey Mushinskiy, and Vladimir Osin (2017). "Satellite Imagery Feature Detection using Deep Convolutional Neural Network: A Kaggle Competition". In: *CoRR* abs/1706.06169. arXiv: 1706.06169. URL: http://arxiv.org/abs/1706.06169.

Inoue, Jun, Judith A Curry, and James A Maslanik (2008). "Application of Aerosondes to melt-pond observations over Arctic sea ice". In: *Journal of Atmospheric and Oceanic technology* 25.2, pp. 327–334.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *CoRR* abs/1502.03167. arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167.

Istomina, L. et al. (2015). "Melt pond fraction and spectral sea ice albedo retrieval from MERIS data – Part 1: Validation against in situ, aerial, and ship cruise data". In: *The Cryosphere* 9.4, pp. 1551–1566. DOI: 10.5194/tc-9-1551-2015. URL: https://tc.copernicus.org/articles/9/1551/2015/.

Kanzow, Thorsten (2023). *The Expedition PS131 of the Research Vessel POLARSTERN to the Fram Strait in 2022*. Ed. by Horst Bornemann and Susan Amir Sawadkuhi. Bremerhaven. DOI: 10.57738/BzPM_0770_2023.

Kim, Duk-jin et al. (2013). "Melt Pond Mapping With High-Resolution SAR: The First View". In: *Proceedings of the IEEE* 101.3, pp. 748–758. DOI: 10.1109/JPROC.2012.2226411.

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG].

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Krumpen, Thomas et al. (2011). "HELIOS, a nadir-looking sea ice monitoring camera". In: *Cold Regions Science and Technology* 65.3, pp. 308–313.

Landy, Jack et al. (2014). "Surface and melt pond evolution on landfast first-year sea ice in the Canadian Arctic Archipelago". In: *Journal of Geophysical Research: Oceans* 119.5, pp. 3054–3075. DOI: https://doi.org/10.1002/2013JC009617. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1002/2013JC009617. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2013JC009617.

Lee, Sanggyun et al. (2020). "Machine learning approaches to retrieve pan-Arctic melt ponds from visible satellite imagery". In: *Remote Sensing of Environment* 247, p. 111919.

Lima, Rafael Pires de and Kurt J. Marfurt (2019). "Convolutional Neural Network for Remote-Sensing Scene Classification: Transfer Learning Analysis". In: *Remote. Sens.* 12, p. 86.

Lin, Tsung-Yi et al. (2017). "Focal Loss for Dense Object Detection". In: *CoRR* abs/1708.02002. arXiv: 1708.02002. URL: http://arxiv.org/abs/1708.02002.

Lu, Peng et al. (2010). "Sea ice surface features in Arctic summer 2008: Aerial observations". In: *Remote Sensing of Environment* 114.4, pp. 693–699.

Meredith, Michael et al. (2019). "Polar Regions. Chapter 3, IPCC Special Report on the Ocean and Cryosphere in a Changing Climate". In.

Miao, Xin et al. (2015). "Object-based detection of Arctic sea ice and melt ponds using high spatial resolution aerial photographs". In: *Cold Regions Science and Technology* 119, pp. 211–222. ISSN: 0165-232X. DOI: https://doi.org/10.1016/j.coldregions.2015.06.014. URL: https://www.sciencedirect.com/science/article/pii/S0165232X15001433.

Nicolaus, Marcel et al. (2010). "Seasonality of spectral albedo and transmittance as observed in the Arctic Transpolar Drift in 2007". In: *Journal of Geophysical Research: Oceans* 115.C11. DOI: https://doi.org/10.1029/2009JC006074. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2009JC006074. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2009JC006074.

Niehaus, Hannah et al. (2023). "Sea Ice Melt Pond Fraction Derived From Sentinel-2 Data: Along the MOSAiC Drift and Arctic-Wide". In: *Geophysical Research Letters* 50.5. e2022GL102102 2022GL102102, e2022GL102102. DOI: https://doi.org/10.1029/2022GL102102. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.

1029/2022GL102102. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2022GL102102.

Notz, Dirk and Julienne Stroeve (2018). "The trajectory towards a seasonally ice-free Arctic Ocean". In: *Current Climate Change Reports* 4, pp. 407–416.

O'Shea, Keiron and Ryan Nash (2015). "An Introduction to Convolutional Neural Networks". In: *CoRR* abs/1511.08458. arXiv: 1511.08458. URL: http://arxiv.org/abs/1511.08458.

Panchi, Nabil, Ekaterina Kim, and Anirban Bhattacharyya (2021). "Supplementing Remote Sensing of Ice: Deep Learning-Based Image Segmentation System for Automatic Detection and Localization of Sea-ice Formations From Close-Range Optical Images". In: *IEEE Sensors Journal* 21.16, pp. 18004–18019. DOI: 10.1109/JSEN.2021.3084556.

Pedregosa, Fabian et al. (2011). "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct, pp. 2825–2830.

Perovich, D. K., W. B. Tucker III, and K. A. Ligett (2002). "Aerial observations of the evolution of ice surface conditions during summer". In: *Journal of Geophysical Research: Oceans* 107.C10, SHE 24–1–SHE 24–14. DOI: https://doi.org/10.1029/2000JC000449. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2000JC000449. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2000JC000449.

Perovich, DK et al. (2002). "Seasonal evolution of the albedo of multiyear Arctic sea ice". In: *Journal of Geophysical Research: Oceans* 107.C10, SHE–20.

Perovich, Donald K et al. (1996). "The optical properties of sea ice". In.

Polashenski, Chris, Donald Perovich, and Zoe Courville (2012a). "The mechanisms of sea ice melt pond formation and evolution". In: *Journal of Geophysical Research: Oceans* 117.C1.

— (2012b). "The mechanisms of sea ice melt pond formation and evolution". In: *Journal of Geophysical Research: Oceans* 117.C1. DOI: https://doi.org/10.1029/2011JC007231. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2011JC007231. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2011JC007231.

Polashenski, Chris et al. (2017). "Percolation blockage: A process that enables melt pond formation on first year Arctic sea ice". In: *Journal of Geophysical Research: Oceans* 122.1, pp. 413–440. DOI: https://doi.org/10.1002/2016JC011994. eprint: https://

agupubs.onlinelibrary.wiley.com/doi/pdf/10.1002/2016JC011994. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2016JC011994.

Rajaraman, Sivaramakrishnan et al. (Nov. 2020). "Analyzing inter-reader variability affecting deep ensemble learning for COVID-19 detection in chest radiographs". In: *PLoS ONE* 15. DOI: 10.1371/journal.pone.0242301.

Rantanen, Mika et al. (Aug. 2022). "The Arctic has warmed nearly four times faster than the globe since 1979". In: *Communications Earth Environment* 3, p. 168. DOI: 10.1038/s43247-022-00498-3.

Raschka, Sebastian (2018). "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning". In: *CoRR* abs/1811.12808. arXiv: 1811.12808. URL: http://arxiv.org/abs/1811.12808.

Ren, Junmei et al. (2021). "Improved Unet Combining Dropout and ACNET for Remote Sensing Image Change Detection". In: *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pp. 4380–4383. DOI: 10.1109/IGARSS47720.2021.9553666.

Ren, Yibin et al. (2022). "Development of a Dual-Attention U-Net Model for Sea Ice and Open Water Classification on SAR Images". In: *IEEE Geoscience and Remote Sensing Letters* 19, pp. 1–5. DOI: 10.1109/LGRS.2021.3058049.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597. arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

Rösel, A., L. Kaleschke, and G. Birnbaum (2012). "Melt ponds on Arctic sea ice determined from MODIS satellite data using an artificial neural network". In: *The Cryosphere* 6.2, pp. 431–446. DOI: 10.5194/tc-6-431-2012. URL: https://tc.copernicus.org/articles/6/431/2012/.

Rösel, Anja and Lars Kaleschke (2011). "Comparison of different retrieval techniques for melt ponds on Arctic sea ice from Landsat and MODIS satellite data". In: *Annals of Glaciology* 52.57, 185–191. DOI: 10.3189/172756411795931606.

Scharien, Randall K. et al. (2007). "Coincident high resolution optical-SAR image analysis for surface albedo estimation of first-year sea ice during summer melt". In: *Remote Sensing of Environment* 111.2. Remote Sensing of the Cryosphere Special Issue,

pp. 160–171. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2006.10.025. URL: https://www.sciencedirect.com/science/article/pii/S0034425707002866.

Scharr, Hanno (2000). "Optimale Operatoren in der digitalen Bildverarbeitung". PhD thesis.

Schröder, David et al. (2014). "September Arctic sea-ice minimum predicted by spring melt-pond fraction". In: *Nature Climate Change* 4.5, pp. 353–357.

Shao, Ling, Fan Zhu, and Xuelong Li (2014). "Transfer learning for visual categorization: A survey". In: *IEEE transactions on neural networks and learning systems* 26.5, pp. 1019–1034.

Shorten, Connor and Taghi M Khoshgoftaar (2019). "A survey on image data augmentation for deep learning". In: *Journal of big data* 6.1, pp. 1–48.

Simard, Patrice Y, David Steinkraus, John C Platt, et al. (2003). "Best practices for convolutional neural networks applied to visual document analysis." In: *Icdar*. Vol. 3. 2003. Edinburgh.

Srivastava, Nitish et al. (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56, pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

Stroeve, Julienne et al. (2021). "A multi-sensor and modeling approach for mapping light under sea ice during the ice-growth season". In: *Frontiers in Marine Science* 7, p. 592337.

Sudakow, Ivan et al. (2022). "MeltPondNet: A Swin Transformer U-Net for Detection of Melt Ponds on Arctic Sea Ice". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 15, pp. 8776–8784.

The GIMP Development Team (June 12, 2019). *GIMP*. Version 2.10.12. URL: https://www.gimp.org.

Thielke, Linda et al. (2022). "Sea ice surface temperatures from helicopter-borne thermal infrared imaging during the MOSAiC expedition". In: *Scientific Data* 9.1, p. 364.

Thielke, Linda et al. (2023). "Preconditioning of Summer Melt Ponds From Winter Sea Ice Surface Temperature". In: *Geophysical Research Letters* 50.4. e2022GL101493 2022GL101493, e2022GL101493. DOI: https://doi.org/10.1029/2022GL101493. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2022GL101493. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2022GL101493.

Tschudi, Mark A., J. A. Curry, and J. A. Maslanik (2001). "Airborne observations of summertime surface features and their effect on surface albedo during FIRE/SHEBA". In: *Journal of Geophysical Research: Atmospheres* 106.D14, pp. 15335–15344. DOI: https://doi.org/10.1029/2000JD900275. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2000JD900275. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2000JD900275.

Wagner, Franz, Anette Eltner, and Hans-Gerd Maas (2023). "River water segmentation in surveillance camera images: A comparative study of offline and online augmentation using 32 CNNs". In: *International Journal of Applied Earth Observation and Geoinformation* 119, p. 103305. ISSN: 1569-8432. DOI: https://doi.org/10.1016/j.jag.2023.103305. URL: https://www.sciencedirect.com/science/article/pii/S1569843223001279.

Weiss, Karl, Taghi M Khoshgoftaar, and DingDing Wang (2016). "A survey of transfer learning". In: *Journal of Big data* 3.1, pp. 1–40.

Wright, N. C. and C. M. Polashenski (2018). "Open-source algorithm for detecting sea ice surface features in high-resolution optical imagery". In: *The Cryosphere* 12.4, pp. 1307–1329. DOI: 10.5194/tc-12-1307-2018. URL: https://tc.copernicus.org/articles/12/1307/2018/.

Wright, Nicholas C. and Chris M. Polashenski (2020). "How Machine Learning and High-Resolution Imagery Can Improve Melt Pond Retrieval From MODIS Over Current Spectral Unmixing Techniques". In: *Journal of Geophysical Research: Oceans* 125.2. e2019JC015569 2019JC015569, e2019JC015569. DOI: https://doi.org/10.1029/2019JC015569. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2019JC015569. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019JC015569.

Yosinski, Jason et al. (2014). "How transferable are features in deep neural networks?" In: *CoRR* abs/1411.1792. arXiv: 1411.1792. URL: http://arxiv.org/abs/1411.1792.