* Check if offline augmentation läuft
* Check normalization amax und amin
* Dropout research

TO-DO:

Dropout

Normalization

Test vgg

Ulf Meeting:

* Overlapping patches in validation set
* Cross validation and final model selection

VGG UND INCEPTIONNET PREPROCESSING IN PREDICTION

K-FOLD NOW DOES RANDOM SPLITS

AUGMENTATION IS DONE AFTER PATCH EXTRACTION

Figures

* Varying surface temperatures, shapes, sizes in IR data
* Simple edge detection does not work
* Feature extraction idea CNN
* Basic U-Net architecture: Encoder 🡪 Decoder
* (Transfer Learning idea)
* (Supplementary Materials: Intermediate Annotation)
* All training images
* Different Patch Sizes
* Augmentation Example
* Model Architecture

Tables

* Dataset sizes and description of what to see in different patch sizes
* Results different backbones

Today

* U-Net comparison
* Test annotation
* Pretraining implementation und train function
* Prepare mp fraction calculation
* Backbone checking
* Check keras implementation

Questions to Gunnar:

* Dataset description: Where is the region, every flight the same region, kann ich das manual verlinken, availability of the data (can be obtained through…), Linda zitieren?, link atwaice campaign?, spatial resolution, preprocessing steps, which spectral band
* Pond stage July
* Paper characteristic melt pond optical properties
* Welche arbeit von hannah zitieren?
* Helicopter observations are providing ground-truth at a higher spatial resolution than achieved by satellite remote sensing to assess the impact of sub-footprint scale variability of the ice surface on the satellite retrievals. Melt pond fraction and size distribution, melt pond optical and IR temperature properties, surface temperature of different ice types and the ocean. Parameters are needed to assess albedo changes, surface energy budget and moment transfer in atmosphere-ice-ocean system and are relevant for corresponding process studies. Improved melt pond, lead and ridge distributions as well as ice type information can be retrieved by combining measurements from different sensors.
* The second overarching objective is to improve and develop new satellite retrievals for melt pond fraction and ice albedo.
* In these two examples melt ponds are the warmest class, the ice surface the second warmest, and the ocean the coldest class. But this is not always the case and for example the ice can be colder than the ocean and the melt pond etc. Thus, no constant IR temperature thresholds can be used.
* However, due to the changing surface conditions, different sections of the helicopter flights have to be manually pre-classified in the primary order of surface temperatures by class, e.g., Tocean < Tice < TMeltPond.

This thesis does this and that. A IR dataset is manually annotated to train, test, and refine the method. A U-Net is evaluated. The objectives / contributions of this thesis are the following:

* To annotate and

TO-DO: Describe images more (homogenouos shapes and colors, melt pond properties 🡪 patch size required), research on too small patch size

**2. Background**

- Related Work in the field

- optical properties of Melt Ponds (?): What does a method need?

In recent years, major improvements in image classification could be achieved with deep convolutional neural network (CNN) architectures. Feature extractors (effective in capturing spatial relationships and extracting hierarchical features) and why they are useful, application in sea ice / RS domain. Can be used for Classification and segmentation, respectively.

**CNN**

CNN architectures mainly comprise three types of layers:

Convolutions

Pooling

**Overfitting**

One disadvantage connected to CNN is their need of big data to learn parameters. Seeing only small amounts leads to the problem of overfitting: The network learns the training data too well such that it cannot generalize to unseen samples. Training data needs to be annotated exhaustively such that with limited resources, only small amounts are available. To combat the problem of overfitting, different regularization techniques are introduced, such as in the following.

**Augmentation**

Augmentation refers to synthetically modifying data before model training. By applying image transformations, dataset size can be increased and more diversity added to the training data (Shorten et al). This way, generalization capability is increased and overfitting combatted. With augmentation methods such as geometric or brightness changes, varying conditions can be simulated that are not captured by the flight at hand, accounting for variations in flight height, atmospheric effects or seasonal and region-based changes, thus improving the generalization of the model especially for future work.

Transformations can happen on pixel-level (such as brightness change or noise injection (57 and 58 in Shorten) or applied on the whole image (flip, rotate, shift) and also targeted towards masks.

(Divide augmentation methods and list some: Basic augmentation techniques are geometric transformations, color space transformations and kernel filters).

(explicitly state that augmentation is applied on training data only).

Augmentation can be applied at to stages in model training: ‘Offline’ before training (increases dataset size, requires memory and training time, preferred for smaller datasets (source)) or ‘on the fly’, generating augmented samples during each training epoch, just before feeding into the model.

Augmentation has been proven as one of the most powerful techniques to combat overfitting, especially in cases with small data. It has been successfully applied in Remote Sensing tasks, an overview can be found in (reference).

**Transfer Learning**

Transfer Learning tackles the problem of overfitting by using pre-initialized weights as starting point for model training.

**U-Net**

U-Net architecture was originally introduced by Ronneberger et al. for biomedical image segmentation. U-Net consists of a U-shaped architecture with an encoder that samples the image down to a condensed representation, and a decoder that restores locational information to pixel-wise classification. Via skip connections, information from the feature extracting path can be used for localization. For more information about the architecture refer to [Ronneberger et al].

Similar to a classical CNN, the encoder serves as feature extractor / U-Net is based on the principles of CNNs / The encoder part resembles the traditional CNN structure.

U-Net is a relatively simple yet powerful architecture that has been applied successfully in many remote sensing segmentation tasks with small datasets.

- CNN architecture and short intro into layers

- CNN and big data

- Techniques to tackle overfitting:

Augmentation

Transfer Learning. Frozen layers means parameters are not updated during training (fixed feature extractor).

Additionally to the CNN layers, decoder uses Upsampling blocks that…

* Dataset description: Where is the region, every flight the same region, kann ich das manual verlinken, availability of the data (can be obtained through…), Linda zitieren?, link atwaice campaign?, spatial resolution, preprocessing steps, which spectral band
* Pond stage July
* Paper characteristic melt pond optical properties
* Welche arbeit von hannah
* Probleme von daten und annotierung in methods oder discussion?

**3. Methodology**

In this section, we exhibit the dataset and implementation details. Then we elaborate on experimental pipeline for model selection. The objectives of this section are the following:

- To annotate and preprocess the IR dataset to be suitable as an input to the segmentation model

- To test various methods that help small data training: patch sizes, pretraining method, augmentation methods

- To build a segmentation pipeline using a trained segmentation model

- To select a winning model based on previous experiments

**3.1 Data**

We used helicopter-borne infrared (IR) imagery acquired during the PS131 ATWAICE campaign with an Infratec Vario-CAM HD head 680 camera. In total, 16 flights are available taken in July and early August 2022 which refers to extended melt pond stage according to … et. al. The geographical area under study is the marginal ice zone of the Fram Strait.

We picked training images from flight 9 due to good weather conditions as fog or clouds can have blurring effects and hinder accurate annotation. Flight 9 covers floes and ponds of various distinct sizes, shapes and temperatures and was therefore sufficient for the scope of our experiments. Out of 4989 available images in flight 9, 8 were selected as training data. The selection process was governed to capture feature diversity (Figure X).

Data was obtained in NetCDF4 file format with 640 x 480 pixels per image. Each image contains one spectral band of 4-17nm with spatial resolution of 1m per pixel. All data had been atmospherically corrected and georeferenced primarily to this study. Images were not ground projected in advance of this study which implicates geometrical distortions at the image borders.

More information about the data can be found in the Supplementary Materials (refer to manual; preprocessing steps as in lindas winter data).

**3.2 Annotation**

Each selected image was pixel-wise manually labelled into one of three classes: (a) melt pond, (b) sea ice and (c) ocean.

In the pixel-level annotations, we roughly used the following criteria to annotate an object as melt pond:

* The pond was visible as pond on the VIS image and detected by the Scharr filter
* Ponds visible on VIS but no or only slight temperature changes on IR were not labelled as ponds
* Ponds at floe edges: In several cases, ponds appeared at the floe edges, ranging into th e floe but no temperature gradient. These cases were labelled as ponds as stated by … et al.
* Temperature change
* (what to do with ponds in contact with water and light temperature changes)

We corrected scharr filter outlines manually to match the shape of the ponds.

Annotation through visual inspection of the imagery.

A more elaborated pipeline of the annotation steps used can be found in the supplementary materials.

Annotation was done with Gimp 2.10 to allow for fine-grained control. Other common labelling tools like LabelMe were disregarded due capturing the shape not well enough (melt ponds often have round shapes that cannot be captured with polygon shape of LabelMe; trace the contours of irregular-shaped objects using a series of connected vertices). For annotation purposes only, image contrast was increased by clipping temperature values to the range of 273 and 276 Kelvin and ‘cividis’ colormap was used to allow for better visibility. As a starting point, outline has been created using mean thresholding and Scharr operator (edge detection filter; manually selected). This outline was blended with the corresponding IR image and turned into a final mask by manual correction and filling with pencil and filling tool. However, these preliminary masks were subject to a lot of noise and could only serve as a rough starting point. Optical images were used as ground-truth during labelling. The labelled masks have been optimized and fine-tuned several times. Despite the careful annotation process, there are labelling uncertainties discussed in Section X.

Outlines could only serve as rough estimate and were often overestimating ponds, noise added or couldn’t detect unclear borders.

The labelling process took several hours for each image, with individual differences depending on the image complexity and accuracy of initial outlines and was limiting factor for the small dataset size in this study. Limiting factor in terms of training set size was the time-consuming labelling process, not data availability.

The resulting masks contained three colors, one for each masks. These were later stacked to 1,2,3 and one-hot encoded for model training, resulting in masks of 3 color channels.

Figure: Proportional distribution of each class in the dataset. (Tortendiagramm)

Figure: Temperature distribution in the dataset and in the entire data of flight 9 (???)

Figure: Labelled images

( Figure: Scharr mask on image; intermediate labelling result )

Supplementary: Annotation Pipeline

**3.3 Preprocessing**

Temperature values in the annotated dataset ranged from 273,15 to 276 Kelvin with a mean of X and a standard deviation of Y. Initial experiments on training with raw temperature values resulted in bad performance which is why images converted to pixel values 0 – 255 using matplotlib.

Data has been been center cropped to 480 x 480 pixels as square sized images are required by the implementation. Additionally, this cuts out larger distortions at the image borders.

Pixel values have been normalized using z-score.

ResNet34 architecture is designed for 3-channel input. As our image dataset consists of one band only, the last channel was copied to reach the shape of three (as often done in …). Data has been converted to float32 as this is best performing for tf.

Data was not normalized otherwise as this is standard for ResNet34 (reference).

Figure: Pixel value distribution

**3.4 Patch Extraction**

Individual melt ponds cover only small parts of the images. By extracting multiple smaller patches, they can still be covered entirely. (Might not be detected when receptive field too large; embedded in too large surface pattern). This way, training set size is increased for the same amount of parameters which might reduce the risk of overfitting (???). However, a patch size that is too small might not capture enough context that is needed to correctly distinguish the surface features under investigation. “By extracting patches, models can focus on learning the specific features and patterns associated with objects of interest.” Might help focussing on the specific spatial context of ponds in close proximity of the pond. Similar to augmentation.

Randomly extracting patches results in additional dataset size increase similar to the effect of shift augmentation, studied in Section X and thus decrease the risk of overfitting.

Melt ponds are characterized by their shape and interconnection channels. This must be covered by the image size. Additionally, ice floes and ocean need context.

Patch sizes too small might not be able to encapsulate characteristic channels or interconnections. Patch sizes too large might capture pattern changes resulting in too large context needed to detect single ponds, not focusing on features.

By training on different patch sizes, the trade-off on dataset size and contextual information is investigated that is needed for accurately detecting surface features with small data.

Patch sizes of 32, 64, 128 and 256 have been investigated, complemented by the entire image 480.

From the 480 x 480 image crops, overlapping patches have been extracted by using a sliding window algorithm with a stride of 128 pixels. Patch extraction was performed on train and test sets individually to avoid information leakage between the sets due to overlapping patches. (random patch extraction?).

Note that the following statements are rough estimation, depend on the feature sizes in individual images.

32: Individual melt ponds are entirely covered, most of the context is cut out.

64: Cover shapes of smaller floes, though not entirely.

128:

256: Covers multiple smaller floes and border shapes of larger floes. Networks of floes are covered.

480: Covers large parts of larger floes, ocean areas.

Figure: Table with patch sizes and resulting dataset sizes.

Figure: Comparison of two images in different patch sizes.

**3.5 Augmentation**

We considered data augmentation to increase the dataset size and variety of training information. However, inappropriate augmentation can introduce unrealistic transformations to the dataset and result in decreasing model performance. To test the effectiveness of different techniques, we created a pool of preselected methods. We incrementally increased the number of augmentation methods applied. If a method resulted in decreasing validation performance, it was disregarded for further configurations. To safe training time, methods have been tested in ‘on fly’ mode. We used the Albumentations library for implementation.

The methods considered were the following:

* Rotation, horizontal and vertical flipping: Simulate changes of orientation. As the images are in overhead perspective, labels are preserved.
* Cropping: Simulate instabilities in flight height and introduce variations in object position and scale.
* Brightness and Contrast: Brightness changes simulate varying surface temperatures while contrast could capture seasonal changes, when temperature differences across ice and water get greater.
* Sharpening and blurring: Blurring (and motion blurring) simulates effects of noise or atmospheric influences such as water vapour. Sharpening could reduce these effects. Sharpening and blurring have been applied exclusively to one another.
* Noise Injection: Gaussian noise has been added to increase robustness to noise.

Each method in the current configuration has been applied with a probability of 0.5 to each training image. For methods that apply interpolation, interpolation has been changed to nearest to preserve categorical mask labels. Other parameters were kept default. Note that for rotation, reflection is used as border interpolation / background embedding method which was decided to be the most natural but however might result in artificial elongations of floe and pond features at the image border.

Color augmentations such as jittering have been disregarded to preserve labels. RGB-based methods were not applicable due to one-channel character. Perspective changing transformations have been disregarded as images will be usually taken of the same angle (overhead perpective).

To test the effectiveness of on-fly versus offline augmentation, the best methods have been selected and applied for both trainings. On-fly augmentation is directly applied when retrieving data and will result in more efficient model training, however, no additional datasize increase. In offline augmentation, one can choose the magnitude by which dataset size is increased. As there is no consensus as to which ratio of original and augmented data is best (Shorten), three different factors have been tested: 2, 5 and 20. Note that a larger factor results in more memory requirements and longer training time for the same number of epochs.

**3.6 Model Architecture**

We used a slightly different U-Net version as introduced in Section X. The modifications are listed below. A detailed description of the model architecture can be found in Figure X. Information about network components is given in Subsections Y.

* The encoder is replaced by the feature extractor of a classification model. This was done to be able to pretrain on the ImageNet dataset *(shortly introduced above)*. ResNet34 was used as backbone because it performed best in preliminary experiments compared to VGG19 and Inception V3 (Table X; maybe refer to thesis for further justification). ResNet34 uses residual connections that make training more efficient and help against the vanishing gradients problem where networks stop to learn (for more information about ResNet refer to …). The finally fully connected layer is replaced by the decoding pathway of the U-Net. Skip connections transfer information from different stages of the ResNet to the decoder.
* The average pooling and fully connected layer is replaced with decoder

“The key difference between a U-Net and a backboned U-Net being that the two convolutional layers and a 2x2 maxpooling operation for each level in the downsampling path are replaced with the different convolutional blocks of the backbone architecture”

Batch normalization, zero padding, check kernel sizes, check network beginning (adjusted to input size)

* The decoder consists of 3x3 upsampling convolutions. This was done following the default setting of the implementation used.
* The decoder uses Batch Normalization layers. These normalize batch activations and have shown to combat the problem of overfitting.

The final layer uses softmax activation function to obtain probability scores for the classes.

Model has been implemented using the implementation of Quebvel. Local modifications are listed in README.md of the Github repository used.

**3.7 Pretraining**

- Imagenet. Two different pretraining modes were tested: Pretraining the feature extractor and fine-tune all layers

- epoch – epoch

- to be able to pretrain on Imagenet, backboned UNet is used. The difference being…

**3.6 Model Training**

Class Imbalance:

- class weights have been computed after data splitting to avoid data leakage

- show class distribution for one example set

The experiments were performed on a Windows platform with an Intel…

**3.6.1 Baseline Training**

**3.6.2 Transfer Learning**

Freeze training for 50 epochs and then unfreeze.

**Experiments and Results**

*Metrics*

To assess the quantitative model performance, we used the following metrics:

IoU: Intersection over Union or Jaccard Index is the ratio of the area of intersection between ground truth and predicted segmentation maps to the area of their union. It is a standard metric for segmentation tasks.

(X.X) *Formula of IoU*

, where G and P are the ground truth and predicted segmentation maps.

F1 Score: F1 Score measures the harmonic mean of precision and recall, where precision is the proportion of true positive predictions out of all positive predictions, while recall is the proportion of true positive predictions out of all actual positive instances.

(X.X) *Formula of F1*

, where TP is the number of true positives, FP the number of false positives and FN number of false negatives.

*Model Evaluation*

Two images were put aside as a test set for final model evaluation.

The remaining images were used to evaluate different training configurations via 4-fold cross-validation. In this technique, the dataset is divided into 4 subsets. Each training configuration is trained four times, where an individual run uses a different fold as validation set and the remaining for training. Validation performances from all folds are averaged to obtain a single measure and used as final performance measure for the training configuration under investigation.

This way, test bias can be avoided and more reliable measure provided, which is a common problem in small datasets. When train and test set don’t come from the same distribution, model is overevaluated. More robust assessment of the model’s generalization capability compared to a single train-test split. More about in Literature X.

4 was selected in order to result in equal splits of the images, 6 images used for training and 2 for testing. Patch Extraction was done after splitting in order to avoid information overlap between the sets due to overlapping image patches. Truly independent datasets will result in more realistic generalization evaluation.

The final split sizes for each patch size are noted in Table X.

*Experimental Setup*

We used implementation, which is based on Keras [1] with Tensorflow [2] as backend. For the main experiments, we used an Intel… machine with 2 cores. We used Weights & Biases to monitor results and metrics. Training was done for 100 epochs as this was sufficient for model convergence in preliminary experiments.

(Weighted) Categorical Cross Entropy Loss was used and Adam optimizer with default settings and learning rate of 10^-4. We used different batch sizes for each patch size, due to different dataset sizes and to obtain optimum memory utilization.

Definition of Categorical Cross Entropy or use other loss and define and why I chose

( Nagi Anmol Master Thesis most successful with Dual loss function )

Model Training and optimization is performed in a subsequent manner by using the metrics introduced in Section X. An illustration of the training pipeline can be found in Figure X.

For model weight initialization (those that are not loaded) uniform kernel initializer has been used.

Ablation Studies

* Ablation on weighted loss
* Ablation on backbone selection: To select the feature extractor architecture, we compared ResNet34 with VGG19 and InceptionV3. VGG19 and InceptionV3 resulted in very low generalization accuracy. This might be due to wrong implementation (maybe refer to master thesis).

Patch Size:

* We observed that there is a direct correlation between bigger patch size and the segmenation accuracy, with an exception to the patch size of 480 x 480. One possible reason for this reduction is a low batch size, or pattern to large. Overall, we observed that 256 x 256 resulted in the best performance.
* We observed that this model was able to identify small floes while it struggled to detect floes which appear partially in the patch

**Discussion**

**Annotation Uncertainties**

**Overfitting**

Overfitting is an issue. Augmentation could reduce but not much. To further reduce, more training data is needed or complexity of the model must be reduced.

Gridding: Ungridded data might lead to the model learning distorted shapes.

Generalization: In future, more flights should be included to the dataset. With extending dataset in future, other flights should taken into account to allow for a broader coverage of data diversity and allow for generalizatbility to future tasks.

Patch Stitching: Different segmentation masks are calculated independently for different patch sizes. These segmentation masks are stacked together and averaged with equal weights.

“To validate and test the model, patches are extracted serially from the images with an overlap of 50%”.

Patch Size – Batch Size (fluctuations might be due to single batch learning).

Although carefully tried after predefined criteria, annotation is subject to subjectiveness and hard to objectively done. In future, this process should be elaborated and done by several human annotators. Especially, the following cases were difficult: - slight temperature changes, - water vapour covering parts of the images

In total, this study can be seen as preliminary experiments to ease design choices and help for future exoerimental settings. 256 as optimal patch size, pretraining on ImageNet works. To create a powerful model that is able to generalize also on different seasons and data, more training data has to be annotated, VIS images or semi-supervised learning approaches could be fruitful.

[1] N. K. Manaswi, N. K. Manaswi and S. John, Deep Learning with Applications Using Python, Cham, Switzerland:Springer, 2018.

[2] M. Abadi et al., "Tensorflow: A system for large-scale machine learning", *Proc. 12th USENIX Symp. Oper. Syst. Design Implementation*, pp. 265-283, 2016.