



Extraction of Surface Water Bodies from Polarimetrically Decomposed Sentinel-1 Images with Deep Learning

Master Thesis

for the attainment of the Master's degree "Master of Science",
abbreviated "MSc"

submitted within the University Master Program for Further Education
"Geographical Information Science & Systems – (UNIGIS MSc)"
at the Department of Geoinformatics - Z_GIS,
Faculty of Digital and Analytical Sciences,
University of Salzburg

submitted by

MSc Lucie Patzwahl

Supervisor:
Dr. Zahra Dabiri

Germering, August 2026

Declaration

"I hereby declare that I have written this thesis on my own and that I have not used any sources or other tools and resources other than those indicated. All passages that have been taken literally or in substance from the cited sources are clearly identified as such.

I confirm that I have not used any illustrations from textbooks or other sources unless they are subject to a license permitting free use.

I further confirm that I have used artificial intelligence (AI)-based applications exclusively for those purposes explicitly permitted in the guidelines on the use of generative AI in the UNIGIS program, in its current version. Any deviations from this policy must be explicitly approved in writing by the supervisor. I bear full responsibility for all content of this thesis.

This thesis has not previously been submitted in the same or a similar form as a Bachelor's/Master's/Diploma thesis/Dissertation/Master Thesis."

Lucie Patzwahl

1 Preface

This thesis is written as a manuscript-style master thesis with the paper (Chapter 2) to be potentially published in the *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* [1]. The journal is an open access journal for the Geoscience and Remote Sensing Society (GRSS) and is published monthly. It addresses the growing field of applications in Earth observations and remote sensing. The range of topics that are appropriate for the journal include current issues and techniques in applied remote sensing for understanding the Earth. The technical content of the paper must be both new and significant. The journal is chosen as it is open access and peer reviewed. Previous papers that this thesis builds on are also published in this paper. The paper contributes to the journal's scope by presenting a new method for water body segmentation from Sentinel-1 images. The manuscript meets the journal's requirements [2]:

The paper

1. should be in double column, single-spaced format.
2. is written in English.
3. should be accompanied by brief biographies and clear photographs of the authors (will be added for publication).
4. must have references in IEEE style.
5. must have an abstract of 150-250 words.
6. requires an Open Researcher and Contributor ID (ORCID) for all authors (will be added for publication).

A technical report (Chapter 3) complements the paper. It provides further details on the used dataset, deep learning model, and some results that could not be included in the paper.

2 Manuscript

The following pages contain the final version of the paper in formatting that is ready for publication. The manuscript was prepared in $\text{\LaTeX}$ using a template provided by the IEEE¹. As the first author, I was responsible for the data acquisition and preparation, the experiment design, the implementation, and the analysis and visualization of results. Furthermore, I prepared the manuscript. During my time at DLR, Sandro Groth gave me feedback on my work, while Marc Wieland provided the most recent version of the convolutional neural network used in this thesis.

The rest of this page is intentionally left blank.

¹<https://template-selector.ieee.org/secure/templateSelector/downloadTemplate?publicationTypeId=1&titleId=23&articleId=3&fileId=28>

Extraction of Surface Water Bodies from Polarimetrically Decomposed Sentinel-1 Images with Deep Learning

Lucie Patzwahl, Sandro Groth, Marc Wieland

Abstract—Surface water maps derived from Sentinel-1 synthetic aperture radar (SAR) imagery are commonly used for flood mapping and water resource monitoring. Convolutional neural networks (CNNs) have improved SAR-based water mapping, yet the potential of dual-polarimetric decomposition parameters as additional input features has not been thoroughly researched. This study investigates whether polarimetric decomposition products improve CNN-based surface water segmentation. A pre-processing workflow is developed to generate the covariance matrix (C_2) elements and H-A- α parameters. These products are integrated into an existing dataset for the benchmarking of CNN for surface water bodies using Sentinel-1 and Sentinel-2 satellite images (*SIS2-Water*). A U-Net-based CNN is trained and tested using different combinations of polarimetric decomposition products as input. Additionally, a pixel-wise analysis based on the ESA WorldCover 2020 dataset is carried out to investigate the influence of land cover types on segmentation errors. The results show that adding polarimetric information produces modest improvements in segmentation performance. The best model (VV-VH- C_2) achieves an Intersection over Union (IoU) of 0.920, Precision of 0.977 and Recall of 0.941. The integration of the decomposition products mainly improves Precision at the expense of reducing Recall. The largest reductions in false positive detections are observed for bare or sparse vegetation and cropland. However, narrow rivers, wetlands and areas of emergent vegetation remain challenging for all tested models.

Index Terms—water body segmentation, Sentinel-1, polarimetric decomposition, convolutional neural networks (CNN).

I. INTRODUCTION

Surface water bodies, like lakes and rivers, are essential for life on Earth. They help maintain ecosystem functions and supply resources for humans, such as drinking water, water for irrigation agriculture, transportation routes, and resources for energy production. Monitoring the geographic extent and temporal dynamics of surface water bodies is, therefore, important, particularly as climate change intensifies hydrological extremes, like droughts and floods, worldwide. Consequently, accurate surface water maps for monitoring water extent and level [1] or identifying flooded areas [2] have grown increasingly important. These maps are also used to validate hydrological models and simulations [1], [3] that support flood risk assessment and help reduce flood-related damage to infrastructure, ecosystems, and communities. During flood events, the time- and cost-efficient preparation and high accuracy of surface water maps is essential since they

form the basis for flood extent maps [4]. To distinguish temporarily flooded areas from permanent water bodies, the detected flood extent is commonly compared with a static reference water mask [5], [6].

To produce the reference water masks, flood mapping services and researchers often use satellite remote sensing. Satellites provide information on surface water extent over large areas and in regions without hydrological monitoring stations. Optical Earth observation sensors are widely used for water extent mapping because high resolution optical images are widely available and straightforward to interpret. Methods using optical imagery are well established and obtain good results [7]–[9]. Huang et al. [8] and Nagaraj et al. [9] give comprehensive reviews of surface water mapping methods based on optical sensors. However, optical sensors are affected by cloud cover along with shadows, which limit their ability to observe the Earth's surface. This limitation is most severe in regions with persistent cloud cover [10] or during flooding events associated with heavy rain and extensive cloud cover. Another limitation is their inability to see through vegetation, which hinders the detection of water bodies beneath vegetation cover. Many flood mapping services, therefore, rely on space-borne Synthetic Aperture Radar (SAR) systems as an active sensor that can acquire data regardless of weather conditions or daylight availability.

SAR data from the Sentinel-1 satellites is frequently employed in mapping applications as the data is free and openly available. Part of the ESA Copernicus program [11], the Sentinel-1 mission comprises a constellation of two polar-orbiting satellites carrying a C-band SAR instrument [12]. The Sentinel-1 satellites systematically acquire data over the entire global landmass with high spatial and temporal resolution enabling consistent and timely monitoring of surface water bodies. They have dual-polarization capability: they can transmit in either horizontal (H) or vertical (V) polarization, and receive in both H and V polarization. The primary mode over land is the interferometric wide swath (IW) mode with VV+VH polarization where signals are transmitted in vertical polarization and received in vertical and horizontal polarization. The suitability of Sentinel-1 SAR data for surface water mapping has been widely evaluated in the literature [7], [13], [14]. Zeng et al. [14] present an in-depth review of the use of the Sentinel satellite constellation (Sentinel-1, Sentinel-2, and Sentinel-3) for inland water monitoring. They conclude that Sentinel-1 is one of the most important sensors for operational surface water mapping because of its all-weather imaging capability and reliable detection of open water. However, accurately distinguishing water from other land cover types with similar radar backscatter characteristics

Manuscript received TBD, 2026; revised TBD, 2026. Date of publication TBD

Lucie Patzwahl is with the University of Salzburg, Salzburg, Austria (email: lucie.patzwahl@online.de). Sandro Groth and Marc Wieland are with the German Remote Sensing Data Center (DFD), German Aerospace Center (DLR), D-82234 Oberpfaffenhofen, Germany (email: sandro.groth@dlr.de; sandro.martinis@dlr.de)

continues to be difficult.

Due to their low surface roughness open water surfaces act as specular reflectors, causing most incident radar energy to be reflected away from the sensor and resulting in low backscatter intensity. In contrast, rougher land surfaces return more energy to the sensor and, therefore, appear brighter. Most SAR-based water mapping methods exploit this contrast using thresholding techniques applied to radar backscatter [15]. However, sediments, strong wind and infrastructure interact with the water surface and change its appearance in the image [5]. These variations in the water's reflected signal, as well as the water bodies' size, shape, and surrounding land cover complicate the development of a universally applicable threshold. Surfaces with similar backscatter reflectance, like sand and concrete, create false positives [16]. Finally, radar shadows and layover can hide water surfaces or be confused for water [1]. Consequently, threshold-based methods often require user intervention or scene-specific tuning to determine an appropriate threshold, limiting their transferability across different regions [15].

More sophisticated approaches use complex sets of rules to calculate the threshold [6], [13] or use supplementary data like the Height above Nearest Drainage (HAND) index to reduce false positives [6], [13]. Landuyt et al. [16] compare some established pixel-based flood mapping approaches, including global and enhanced thresholding, active contour modeling and change detection. Machine learning methods are also widely employed, most commonly Random Forests (RF) and Support Vector Machines (SVM), which learn decision boundaries from labelled training data and generally provide greater robustness than rule-based thresholding approaches. Kirby et al. [7] compare different input combinations for Sentinel-1 and Sentinel-2 images with some commonly used algorithms: Support Vector Machine (SVM), Random Trees (RT), Maximum Likelihood Classifier (MLC), and K-Nearest Neighbor (KNN). Guo et al. [17] provide an overview of common machine learning methods for water body segmentation. They find that these methods have all been successfully applied, but require manual feature engineering and are sensitive to speckle noise. Over the last decade, surface water mapping evolved from manual thresholding techniques to automatic and semi-automatic methods [5], [6], [13], [18].

Deep learning has seen an increased interest by researchers due to more available (high resolution optical and multi-spectral) data and increased processing power [19]. In particular, convolutional neural networks (CNN) are capable of outperforming conventional image processing techniques for image segmentation [20] as they preserve spatial relationships within an image [21]. They are widely used for satellite image analysis, e.g. for land cover classification or the extraction of water bodies [17], [22]. Guo et al. [17] review some deep learning methods for water body segmentation. They conclude that deep learning has become the state-of-the-art approach for SAR water-body segmentation, performing better than traditional machine learning methods because it can automatically learn complex spatial features. However, there are several unsolved challenges, including speckle noise inherent to SAR imagery, poor performance on narrow rivers and complex

water boundaries, limited availability of accurately labelled training data, and models that generalize across different sensors, seasons, and geographic regions [17]. Bentivoglio et al. [21] give a comprehensive overview of the application of deep learning to flood mapping. They conclude that deep learning, particularly CNN-based architectures, has become the state of the art for flood mapping, outperforming both traditional image processing techniques and machine learning methods. However, future progress is expected to depend less on increasingly complex network architectures and more on improving training datasets and integrating multiple data sources [20], [21].

U-Net has emerged as the dominant deep learning architecture for SAR water segmentation because its encoder-decoder design and skip connections preserve fine spatial details while performing well with relatively small labelled datasets, a common limitation in SAR applications [17]. Katiyar et al. [23] use the *Sen1Flood11* dataset [24] to train and test a SegNet-like and a UNet-like network, finding that the UNet-like networks outperform the SegNet-like networks. Berczky et al. [20] test several possible network architectures of CNN for surface water segmentation. All tested CNN architectures achieve higher segmentation accuracy than an operational threshold-based processor when dual-polarization (VV-VH) data is used. But increasing network depth or using more sophisticated architectures does not result in significant improvements. Specifically, U-Net achieves a performance comparable to that of all other tested architectures. A standard U-Net, therefore, offers a good trade-off between segmentation accuracy and model complexity.

Research in surface water mapping has focused primarily on methods operating on SAR intensity data derived from a one or two polarization channels. As an example, Tuele et al. [6] use automatic intensity thresholding to derive flood extent maps from Sentinel-1 intensity data. They find that certain polarizations (VV) yield slightly better results and that rough water surfaces induced by strong winds, sand and emergent vegetation lead to misclassifications. Although reducing these misclassifications is the aim of current research, the phase information contained in the SAR signal is rarely used for water mapping [17].

One approach for making use of the phase information is polarimetric decomposition, a technique used in polarimetric Synthetic Aperture Radar (PolSAR) to characterize the physical scattering mechanisms of observed targets. When a transmitted radar pulse interacts with the surface, its polarization is modified depending on the shape, orientation, size, and dielectric properties of the involved scatterers. PolSAR exploits these polarization changes by analyzing both the amplitude and phase information contained in the returned signal. Depending on the type of scattering and the number of signal bounces, different scattering mechanisms can be distinguished: single-bounce, double-bounce, and volume scattering. These scattering mechanisms provide useful information for the classification of land cover types and other surface targets.

One technique involves the interpretation of the scattering matrix that is calculated from the recorded amplitude and phase data. Due to the complexity of the scattering mech-

anisms, the interpretation of the scattering matrix or other derived matrices like the coherency or covariance matrix is not always straightforward. For this reason, researchers apply polarimetric decomposition methods that reduce the complex scattering information into a set of physically meaningful parameters representing the different scattering mechanisms. These decomposition techniques can be classified into two categories: model-based methods [25]–[27], which assume predefined scattering models, and eigenvalue- or eigenvector-based methods [28], [29], which derive scattering characteristics directly from the scattering or coherency matrix.

Most studies use fully-polarimetric (quad-pol) SAR data and model-based decompositions because the complete polarimetric information enables a detailed description of scattering processes. In contrast, comparatively less research has explored decomposition methods for dual-polarimetric SAR data, where the reduced polarization information poses some challenges. Sentinel-1 data is dual-polarized and does not contain the full polarimetric information needed for most decomposition methods.

This study aims to contribute to the advancements of using dual-polarimetric decomposition of Sentinel-1 data in the mapping of surface water bodies by answering the following questions:

- 1) For Sentinel-1, does the integration of a polarimetric decomposition method into deep learning algorithms improve the accuracy of water surface segmentation compared to purely amplitude-based inputs?
- 2) Which Sentinel-1 polarimetric decomposition product (e.g., covariance matrix elements or $H-A-\alpha$ parameters) provides the highest segmentation accuracy for water surfaces when used as input for deep learning algorithms?
- 3) Does the application of polarimetric decomposition methods improve segmentation results for specific land cover types (e.g., sand, bare soil, flooded vegetation) compared to the amplitude-based method?

To answer the research questions, an existing training dataset of Sentinel-1 SAR scenes is enriched with polarimetric decomposition products. Section II introduces the used datasets, describes the polarimetric decomposition methods used to derive additional input features, and details the preprocessing workflow to prepare the data for model training. The usefulness of different combinations of Sentinel-1 input features is investigated by adapting a convolutional neural network created for water body segmentation. Section III presents the network architecture, the experimental design, and the performance metrics used to assess the different input configurations. The results of these experiments are presented in Section IV and discussed in Section V, where additionally the limitations of this work and directions for future research are discussed. Finally, Section VI summarizes the main findings of this work.

II. DATA

The training, validation and testing of the neural network requires a suitable dataset with a sufficient amount of training samples. In order for the algorithm to learn the essential characteristics, the images in the dataset should contain accurate

surface water bodies of varying appearance, size, shape and orientation [30]. To enable generalization, the dataset should also contain all other possible land cover types sampled from around the world and at different times of the year [31]. To answer the research question, the dataset must at least contain Sentinel-1 intensity data, polarimetric decomposition products and corresponding binary water masks. Additional information like the elevation or slope is optional.

A. SIS2-Water Dataset

For the training, validation and testing of CNNs for semantic segmentation of surface water bodies using Sentinel data, Wieland et al. [31] introduce a global reference dataset called *SIS2-Water* that contains 65 sets of Sentinel-1 ground range detected (GRD) interferometric wide swath data and Sentinel-2 L1C images. The images were acquired between May 2018 and November 2020, with the majority of images from 2019. The Sentinel-1 images each contain a band for VV and for VH polarization. The dataset covers permanent water bodies across 29 countries and 18 predominant land cover types. The scenes are 100 km by 100 km and are based on the Sentinel-2 tile-grid. The dataset provides binary water masks which are derived with the help of a NDWI threshold and manually checked and refined. The dataset also includes slope and elevation data derived from the Copernicus 30 m Digital Elevation Model (DEM) [32]. A valid pixel mask excludes areas where data is not available due to cropping. A spatio-temporal asset catalog (STAC) [33] complements the dataset with metadata. The dataset has a fixed division into training (50 %), validation (25 %) and test (25 %) splits to ensure reproducibility. The *SIS2-Water* dataset [34] as well as preparation scripts and SNAP graphs [35] are available for download.

B. Polarimetrically Decomposed Sentinel-1 Images

To test the usefulness of using decomposed data for semantic segmentation of water bodies, the *SIS2-Water* dataset needs to be extended with polarimetrically decomposed Sentinel-1 images. Sentinel-1 data is dual-polarized and thus does not contain the full polarimetric information needed for most decomposition methods. One possible decomposition method is the widely used Cloude-Pottier decomposition which is based on the eigenvalues of the covariance matrix [28], [29]. The scattering matrix $[S_2]$ for dual-polarimetric Sentinel-1 is defined as

$$[S_2] = \begin{bmatrix} 0 & 0 \\ S_{VH} & S_{VV} \end{bmatrix} \quad (1)$$

$[S_2]$ describes the transformation of the incident wave to the backscattered wave.

The covariance matrix $[C_2]$ for the dual-polarimetric case is

$$[C_2] = \begin{bmatrix} \langle S_{VV} S_{VV}^* \rangle & \langle S_{VV} S_{VH}^* \rangle \\ \langle S_{VH} S_{VV}^* \rangle & \langle S_{VH} S_{VH}^* \rangle \end{bmatrix} \quad (2)$$

where S stands for the entries in the scattering matrix, $*$ denotes the complex conjugate and $\langle \rangle$ the local mean value.

For Sentinel-1 data, the decomposition adapted for the dual-polarization case [29] is used. The decomposition results

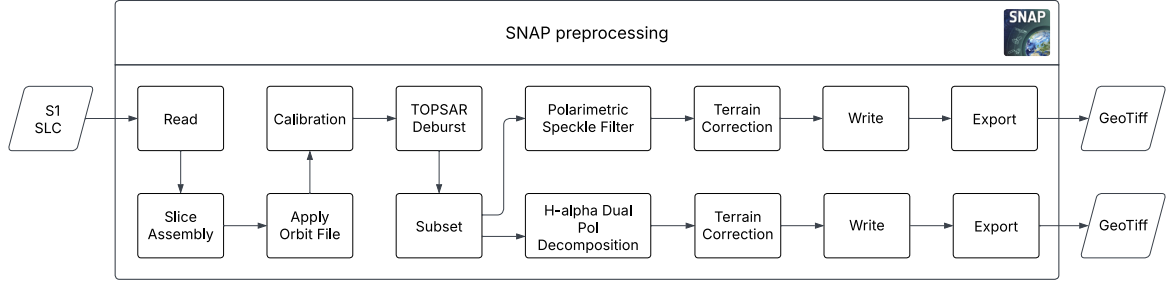


Fig. 1. Preprocessing workflow in SNAP

in three variables: the polarimetric entropy H , the mean scattering angle α and the anisotropy A . With p_i defined as

$$p_i = \frac{\lambda_i}{\lambda_1 + \lambda_2}, \quad i = 1, 2 \quad (3)$$

where λ stands for the eigenvalue, the entropy H is defined as [29]

$$H = -\sum_{i=1}^2 p_i \log_2 p_i, \quad H \in [0, 1] \quad (4)$$

The entropy H describes the degree of randomness of the polarizations or the purity of the scattering mechanisms. A low entropy indicates a target with little variation in polarization, such as bare and smooth soil. A high entropy indicates a complex scattering behavior with several contributing scattering mechanisms and varying polarizations, such as forests.

According to Ma et al. [36], the mean scattering angle α can be calculated as

$$\alpha = \sum_{i=1}^2 p_i \alpha_i = \sum_{i=1}^2 p_i \cos^{-1}(|u_i(1)|), \quad \alpha \in [0^\circ, 90^\circ] \quad (5)$$

where $u_i(1)$ is the first entry in the corresponding eigenvector. α identifies the dominant scattering mechanism.

The anisotropy A is calculated from the eigenvalues as follows:

$$A = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}, \quad A \in [0, 1] \quad (6)$$

As can be seen from Equation (4) and Equation (6), entropy and anisotropy are calculated using the same eigenvalues. Therefore, in the dual-polarization case, these two parameters contain the same information. Since dual-polarized data does not provide a complete scattering matrix, the resulting decomposition parameters do not have the same physical interpretation as in the full-polarimetric case. This limitation must be taken into account when interpreting the resulting values.

C. Data Preprocessing

Polarimetric decomposition requires the complex-valued scattering information contained in Sentinel-1 Single Look Complex (SLC) products, because the phase information is not preserved in Ground Range Detected (GRD) products. Therefore, for each sample in the *SIS2-Water* dataset, the corresponding SLC acquisition to the existing GRD product

is retrieved from the Copernicus Data Space Ecosystem [37]. We process the SLC products with the help of the Graph Builder and Batch Processing framework of the Sentinel Application Platform (SNAP) (Version 11) [38]. The complete preprocessing workflow in SNAP is illustrated in Figure 1. We first apply standard Sentinel-1 preprocessing operations, including orbit file application, radiometric calibration, and TOPSAR debursting. Subsequently, a subset of the scene is created to match the extent of the original dataset, reducing the needed time and computational cost for the following processing steps.

To derive the covariance matrix, we apply a polarimetric speckle filter. Unlike conventional speckle filters, polarimetric filters preserve the phase information while reducing speckle noise. In this work, the Refined Lee filter [39], [40] is used with a 5×5 moving window, following recommendations for Sentinel-1 polarimetric processing [41]. The output of the filter consists of the four elements of the dual-polarimetric covariance matrix, C_2 , namely C_{11} , $C_{12,\text{real}}$, $C_{12,\text{imag}}$, and C_{22} , each represented as a separate image band. SNAP provides different implementations of polarimetric decomposition algorithms, including the dual-polarimetric H - A - α decomposition used in this study. This decomposition transforms the covariance matrix into the entropy (H), the anisotropy (A), and the mean scattering angle (α), storing each parameter in an individual image band. As last step, we apply terrain correction to the images before they are exported as GeoTIFF.

Using a Python script, we subsequently co-register the resulting decomposition layers with the *SIS2-Water* Sentinel-1 intensity data to ensure spatial alignment. The co-registered layers are then combined into a multi-band image stack comprising the two Sentinel-1 backscatter channels (VV and VH), the four covariance matrix elements, the three dual-polarimetric decomposition parameters. During training and inference, we select different combinations of these bands through the CNN configuration file to perform a systematic evaluation of different input feature combinations. To be consistent with the existing *SIS2-Water* dataset, the final products are exported as Cloud Optimized GeoTIFFs (COGs) [42]. The corresponding STAC [33] metadata files are updated to include the polarimetric decomposition products. Before training, the created image stacks are divided into smaller tiles for ingestion into the model. During this step the slope can optionally be appended to the image stack.

D. Land Cover Map

To analyze the influence of different land cover types on the segmentation results, we use ESA WorldCover 2020 [43] to extract the land cover classes for each test scene. WorldCover is a global land cover map with a spatial resolution of 10m based on the Sentinel-2 UTM grid [44]. The classification follows the Land Cover Classification System (LCCS) as developed by the Food and Agriculture Organization of the United Nations [45] and contains 11 land cover classes. The classes are *Tree cover*, *Shrubland*, *Grassland*, *Cropland*, *Built-up*, *Bare / Sparse Vegetation*, *Snow and Ice*, *Permanent Water Bodies*, *Herbaceous Wetland*, *Mangroves*, and *Moss and Lichen*. For classification, WorldCover uses features extracted from Sentinel-1 and Sentinel-2 as well as auxiliary data like the elevation and slope derived from a digital elevation model. Its algorithm is based on the algorithm producing the yearly Copernicus Global Land Service Land Cover (CGLS-LC). An independent validation reports a global overall accuracy of 74.4% [46]. In terms of class-specific accuracy, the *Tree Cover*, *Snow and Ice*, *Cropland*, *Permanent Water Bodies*, and *Bare / Sparse Vegetation* classes achieve the highest accuracies while *Shrubland*, *Herbaceous Wetland*, and *Moss and Lichen* achieve lower accuracies. The tiles can be downloaded from the ESA WorldCover Viewer Website [47]. After downloading, we cut the land cover tiles to size and reproject them to match the resolution, projection (UTM), and extent of the image stack.

III. METHOD

In the following section, we describe the used convolutional neural network architecture and its hyperparameters. The experiments we conduct to test the influence of different input feature combinations on the performance are described. We also define the evaluation metrics used to assess the performance.

A. Deep Learning Model

We use a convolutional neural network (CNN) for the segmentation of the surface water bodies. The CNN model tested in this study is based on the U-Net architecture [48] as it has shown good results in the area of water segmentation [20], [49]. For the encoder, we use EfficientNet-B0 [50], as it offers a good compromise between accuracy and through-put [31]. We use an equally weighted combination of cross-entropy and Lovász loss ($\alpha = 0.5$) for training, as this combination achieved the best results in previous studies [20]. The training data shows a class imbalance, with land being considerably more common than water. A standard cross-entropy loss would make the model biased towards predicting the dominant class (land), leading to poor performance on the less frequent class (water). To account for this, the loss function assigns higher weights to the underrepresented class and lower weights to the dominant one. The cross-entropy is weighted according to the positive class distribution in the training data. The lowest validation loss determines the best model state. The neural network is optimized using the AdamW optimizer ($\beta_1 = 0.9$ & $\beta_2 = 0.999$). The initial learning rate was set to 0.001 and

TABLE I
DEEP LEARNING MODEL & HYPERPARAMETERS

Hyperparameter	Value
Encoder	U-Net [48]
Decoder	EfficientNet-B0 [50]
Loss function	Cross-Entropy + Lovász with $\alpha = 0.5$ [20]
Loss weight (cross-entropy)	7.797 (water class) [31]
Optimizer	AdamW with $\beta_1 = 0.9$ & $\beta_2 = 0.999$ [51]
Initial Learning Rate	0.001
Weight Decay	0.01
Batch Size	32
Max number of epochs	100
Early Stopping Patience	9

scheduled using cosine annealing warm restarts. Stochastic Weight Averaging (SWA) replaces the scheduler during the final 20% of the configured maximum number of epochs. A weight decay of 0.01 is applied, and all model weights are initialized randomly. Training was performed with a batch size of 32 for a maximum of 100 epochs, using an early stopping patience of 9 epochs. To guarantee reproducibility, we perform training in deterministic mode and set a random seed of 99. We do not apply any data augmentations in order to isolate the influence of the different input feature combinations. All experiments are executed on a workstation running Windows 10, equipped with an NVIDIA RTX A4000 GPU and an Intel Xeon W-2223 CPU @ 3.60 GHz.

B. Evaluation Metrics

Accuracy is not a suitable metric for the evaluation of semantic segmentation results when classes are imbalanced [21], [31]. The inaccuracy of minority classes (water) are overshadowed by the accuracy of majority classes (non-water). Therefore, we calculate Intersection over Union (IoU) instead. It is defined as the ratio between the intersection and the union of the predicted and the reference segmentation mask and is calculated as follows:

$$IoU = \frac{tp}{tp + fp + fn}, \quad (7)$$

with number of true positives tp (water pixels correctly identified as water), false positives fp (land pixels wrongly identified as water) and false negatives fn (water pixels wrongly identified as land).

The Precision measures the proportion of predicted water pixels that are correctly identified. It is a helpful measure when the cost of false positives (non-water pixels marked as water) is high.

$$Precision = \frac{tp}{tp + fp} \quad (8)$$

Recall measures the proportion of actual water pixels that are correctly identified by the model. It is useful when the cost of false negatives (missed water pixels) is high.

$$Recall = \frac{tp}{tp + fn} \quad (9)$$

The F1-score (F1) is the harmonic mean of Precision and Recall, representing both in one metric.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (10)$$

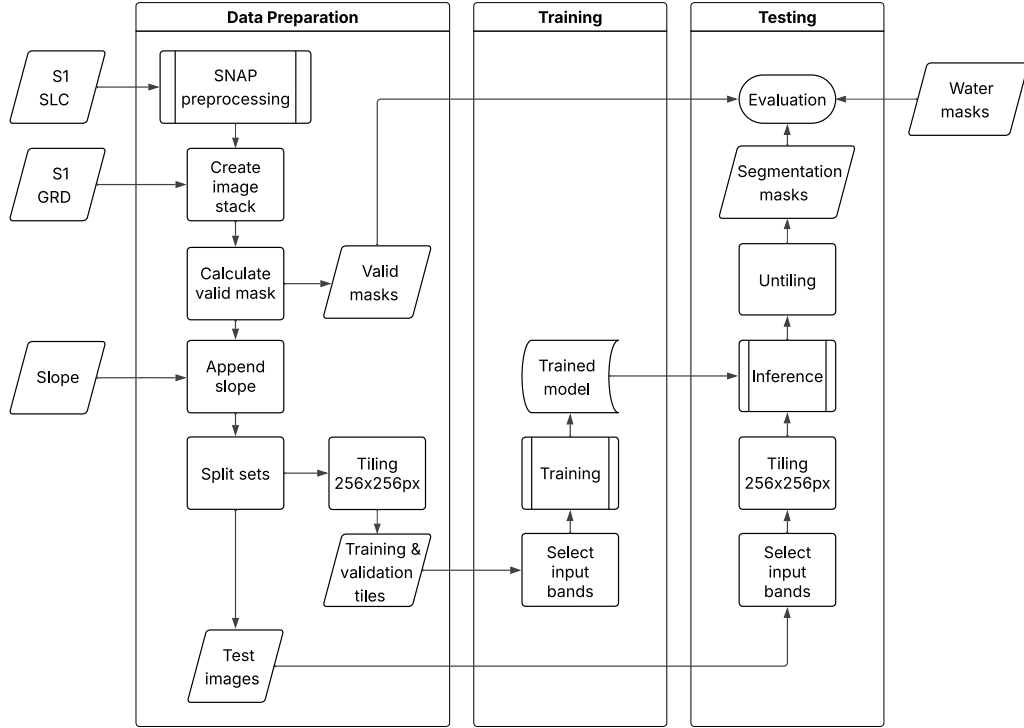


Fig. 2. Workflow scheme for the preparation of the polarimetric products, the training and testing of the convolutional neural network.

TABLE II
BANDS USED FOR EXPERIMENTS

ID	VV	VH	C_{11}	$\text{Re}(C_{12})$	$\text{Im}(C_{12})$	C_{22}	H	A	α	Slope
VV	×	×								×
C2			×	×	×	×				×
HA			×	×	×	×	×	×	×	×
HAC2			×	×	×	×	×	×	×	×
VVC2	×	×	×	×	×	×				×
VVHA	×	×					×	×	×	×
VVC2HA	×	×	×	×	×	×	×	×	×	×

TABLE III
AVERAGE INTERSECTION OVER UNION (IoU), F1-SCORE, PRECISION, AND RECALL FOR DIFFERENT EXPERIMENTS ACROSS ALL TEST SCENES.

ID	IoU	F1	Precision	Recall
VV	0.911	0.954	0.956	0.951
C2	0.833	0.909	0.985	0.844
HA	0.848	0.918	0.959	0.880
HAC2	0.804	0.891	0.997	0.806
VVC2	0.920	0.958	0.977	0.941
VVHA	0.900	0.947	0.968	0.928
VVC2HA	0.911	0.953	0.982	0.927

We use an open-source Python package [52] to calculate these metrics.

C. Experiments

To test the impact of using polarimetrically decomposed images, we conduct experiments with different band combinations. All experiments are listed in Table II. The initial experiment uses only the VV and VH bands to create a baseline and to enable a comparison with the results reported by Wieland et al. [31]. We then evaluate the influence of the covariance matrix C_2 features, H - A - α decomposition parameters, and their combinations. Terrain slope derived from the Copernicus DEM is included in all experiments as Wieland et al. [31] show its inclusion improves segmentation results by reducing false negatives.

IV. RESULTS

Table III shows the overall segmentation performance over all scenes for the different CNN input band combinations. Most experiments have relatively high median IoU values pointing to generally good segmentation performances. The baseline band combination of VV has the lowest Precision and the highest Recall value indicating that this band combination is good at detecting water at the expense of introducing false positives. All other band combinations improve the Precision at the cost of lowering the Recall value, i.e., less land is classified as water but also less water is identified correctly. The band combination VVC2 has the highest IoU and F1-score.

Figure 3 presents box plots of the IoU, F1-score, Precision, and Recall obtained for the different experiments over all test scenes. The experiments which include VV-VH show better and more consistent performance. They have higher median IoU values and lower variability across scenes than

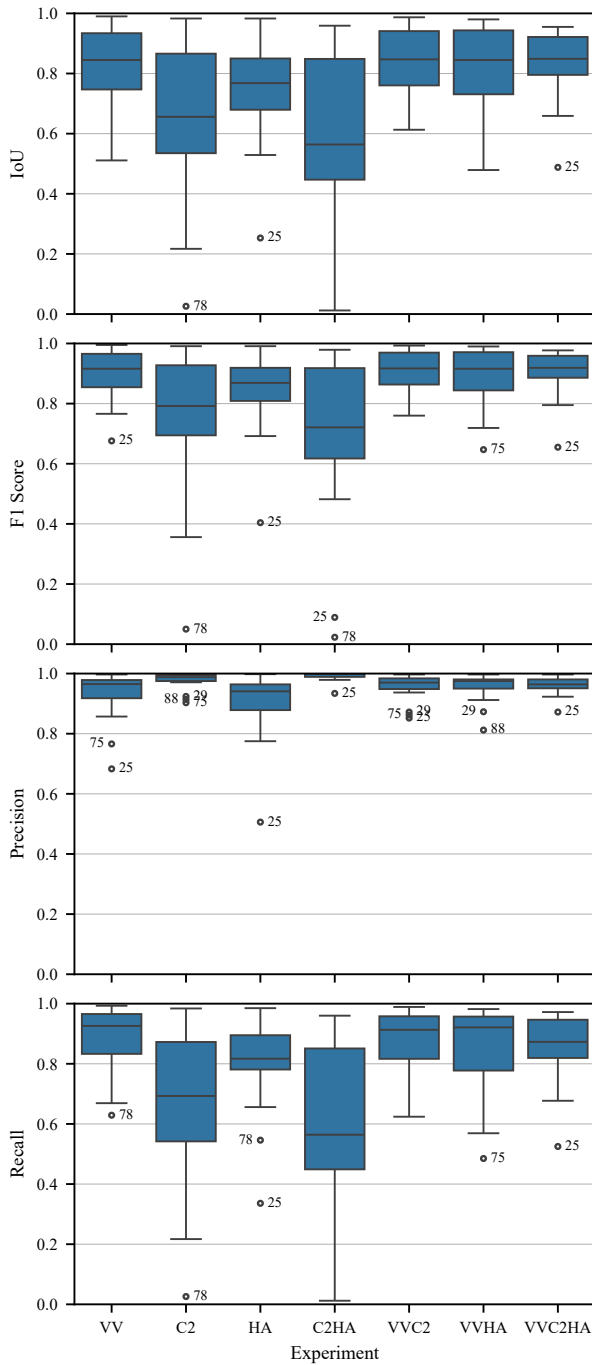


Fig. 3. Intersection over Union (IoU), F1-score, Precision, and Recall for different experiments across all test scenes. Horizontal lines show medians, boxes represent the interquartile range, and whiskers represent 1.5 times the interquartile range. Outliers are labelled with the corresponding scene ID.

experiments without those bands. The combination *C2HA* has the lowest performance of all band combinations. It has the lowest median IoU, the largest variability and values close to zero. The F1-score distributions largely follow the same pattern as the IoU, which is expected given the close relationship between the two metrics.

Apart from some outlier scenes, Precision values are con-

sistently high and the variability low across nearly all experiments indicating that most band combinations produce relatively few false positives. Adding the polarimetric information reduces the variability between scenes while keeping high median values. The band combinations with only the decomposition products have the highest Precision values but also the lowest Recall values. Of all the metrics, Recall shows the lowest median values with broad distributions and very low values for several band combinations. This indicates that these band combinations are less effective at detecting all water pixels.

The experiments using *VV* generally achieve higher Recall medians and less variable performance across scenes. Experiments without *VV* input display larger variability and more frequent low-performing scenes. This suggests that the *VV* band contributes strongly to lowering the number of false negatives. Therefore, only experiments using the intensity bands are considered for the remainder of the analysis.

Several scenes consistently show up as outliers for all experiments and evaluation metrics, particularly scenes 25, 75, and 78. These scenes are associated with lower model performance regardless of the input band combination. To analyze the scene dependence further, Figure 4 shows scene-wise IoU, Precision, and Recall values for the band combinations containing the intensity bands. Scenes 33, 35, 53, 57, 80, 82, and 89 achieve the strongest performance among all metrics and experiments. These scenes appear to be comparatively easy to segment, as all band combinations produce accurate predictions with minimal differences between experiments.

In contrast, the remaining scenes are more challenging to classify. Scenes 25, 75, and 88 have substantially lower Precision values than the other test scenes, particularly for the *VV-VH*-only experiment. The inclusion of additional polarimetric features, especially the *C2* data, consistently improves Precision for these scenes by reducing false-positive detections. In addition to the scenes with low Precision, several other scenes stand out because of their comparatively low Recall values for all experiments: scenes 23, 25, 29, 47, 77, and 78. For these scenes, including the polarimetric information (*C2* and *HA*) slightly increases the Recall without a significant loss in Precision.

For selected scenes with low Recall, Figure 5 presents the Sentinel-2 images and corresponding water masks, showing areas that are consistently false negatives across all band combinations. False negatives occur predominantly in narrow rivers and water channels (Scenes 23 and 77), wetland areas (Scene 47), lake boundaries and shorelines (Scenes 29 and 78), and small water bodies (Scene 78). Scenes 23 and 77 are both characterized by false negatives along the rivers. The enlarged regions show that the models completely fail to detect the narrow river channels. They also fail to delineate the shore of the larger river, resulting in false negatives along the riverbanks. However, the two scenes represent different landscape settings: Scene 23 depicts a canyon system with steep banks in a sparsely vegetated area, whereas Scene 77 shows a narrow river flowing through a hilly, forested landscape. The false negatives in both scenes suggest that accurately delineating narrow rivers remains challenging regardless of the

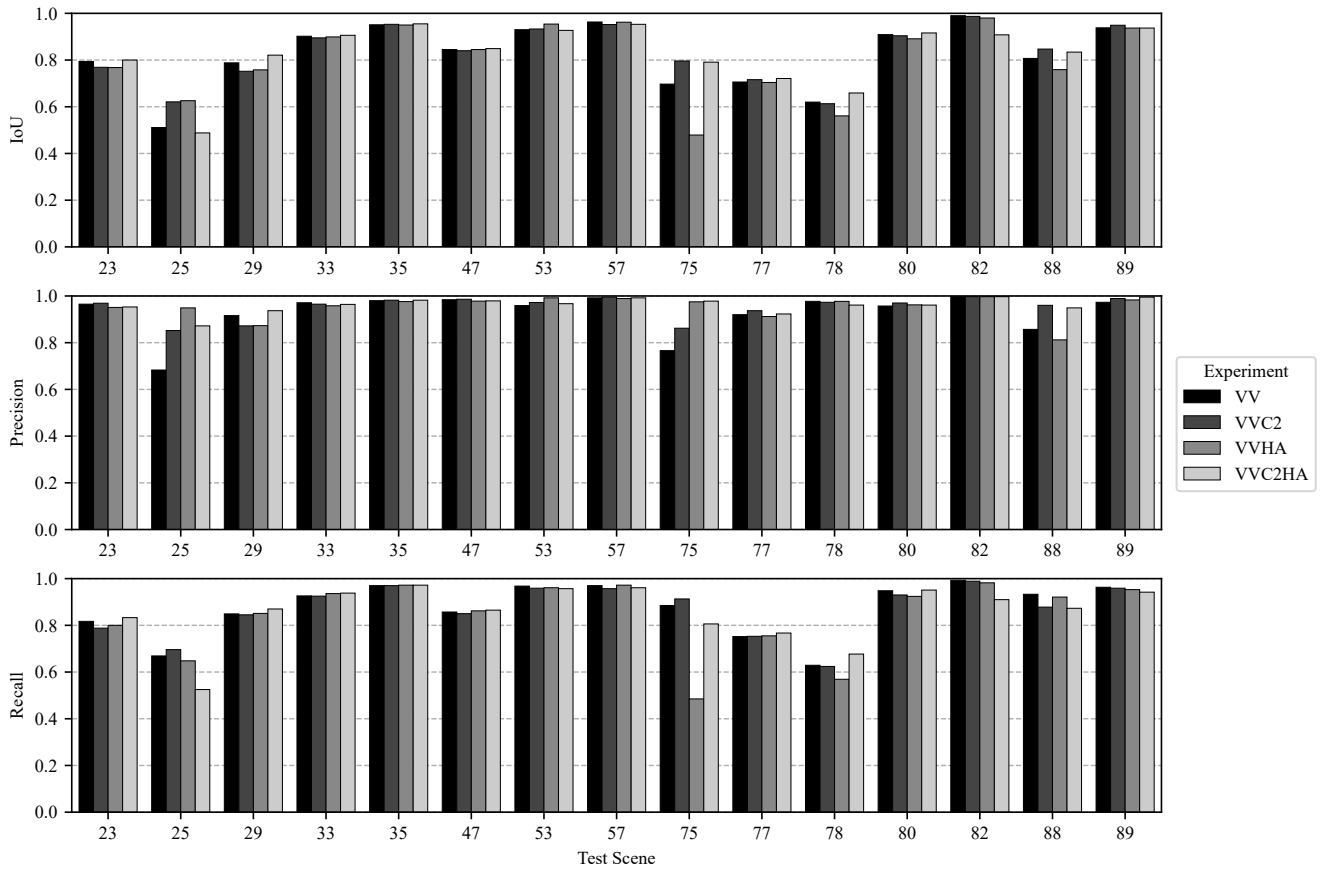


Fig. 4. Intersection over Union (IoU), Precision, and Recall for each test scene across chosen input band combinations.

band combination.

In Scene 29, false negatives are largely concentrated along the shorelines of the oxbow lakes. While the models successfully detect the main water bodies, they have difficulties to accurately delineate their boundaries. The enlarged Sentinel-2 image reveals that many of the areas classified as false negatives correspond to shallow vegetated areas rather than open water. The northern part of the lake in the western section of the scene is another example of an area of emergent vegetation.

Scene 47 has the most extensive false negatives among the selected scenes. Large clusters of false negatives are visible throughout the wetland complex shown in the inset. This scene again shows that all band combinations have difficulties delineating wetland areas, which have more complex backscatter characteristics than open water. This suggests that wetlands and transition zones between land and water remain particularly difficult.

Scene 78 shows an example of small scattered water bodies in a hilly landscape dominated by grassland. Recall is reduced by many small false negatives at the borders of the lakes. Again, false negatives arise from the difficulty of distinguishing open water from surrounding emergent vegetation.

Figure 6 shows the land cover and the segmentation results for chosen experiments with low Precision values. Among all

scenes, scene 25 shows the worst results in VV with both low Precision and Recall values. The Precision improves by including polarimetric information, at the expense of lower Recall. Among the evaluated combinations, the inclusion of $H-A-\alpha$ produces the highest IoU, mostly through improved Precision with only a slight reduction in Recall. The models struggle to map the extent of the narrow river in the central part of the scene, resulting in false negatives (purple) along river and lake boundaries. Among the experiments, the VV model has the largest number of false positives (red). False positives occur in areas that are classified as wetland in the ESA WorldCover map. Note that while the lake is marked as water in the *SIS2-Water* dataset it is classified as wetland in the ESA WorldCover map. Incorporating the polarimetric information reduces the false positives, although false negatives along the narrow rivers and lake remain.

For scene 75, VV correctly identifies the major water bodies in the northwestern part of the scene. However, it produces numerous false-positive detections across the scene, specifically in areas that are classified as cropland. The addition of the C_2 matrix (VVC2) decreases the amount of false positives, although a number of spurious water detections remain. The VVHA and VVC2HA models also reduce false positives but cause a noticeable increase in false negatives, particularly in the water body and wetland area in the northwest.

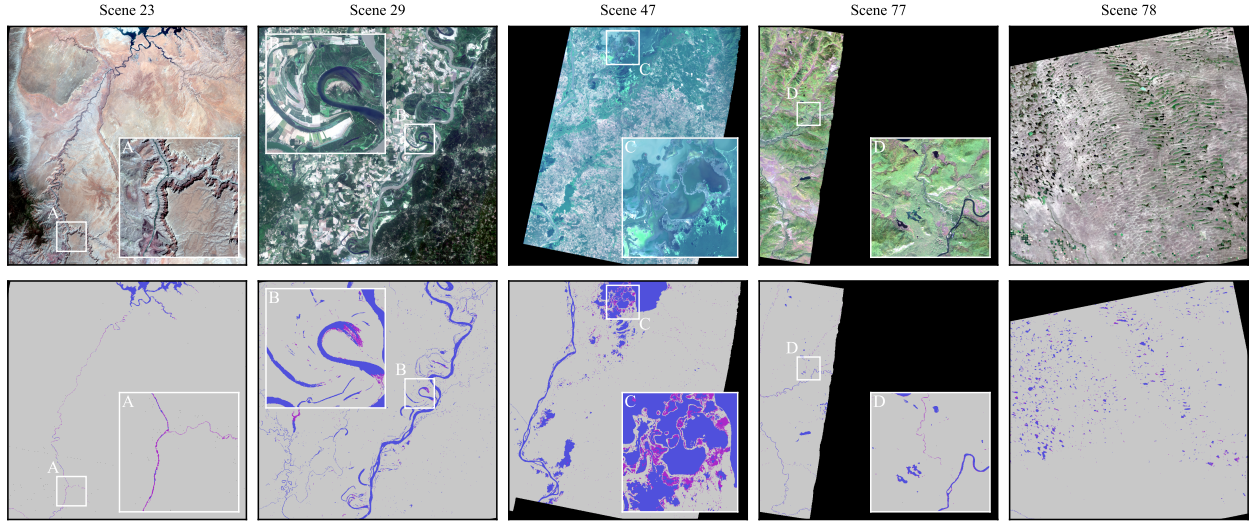


Fig. 5. Land cover and water masks for selected test scenes with low Recall value. Blue: true positives (water), gray: true negatives (non-water), purple: false negatives (missed water) by all band combinations. Land cover types: Bare / Sparse Vegetation (gray), Built-up (red), Cropland (pink), Grassland (Yellow), Herbaceous Wetland (teal), Mangroves (green), Moss and Lichen (beige), Permanent Water Bodies (blue), Shrubland (orange), Snow and Ice (white), Tree Cover (dark green).

TABLE IV

PERCENTAGE OF FALSE-POSITIVE PIXELS FOR EACH LAND COVER TYPE AND EXPERIMENT; PERCENTAGE OF FALSE-NEGATIVE PIXELS FOR EACH EXPERIMENT.

Land Cover	VV	VVC2	VVHA	VVC2HA
Bare/Sparse Vegetation	2.4 %	0.7 %	2.7 %	0.6 %
Built-up	0.2 %	0.2 %	0.2 %	0.1 %
Cropland	1.6 %	1.0 %	0.2 %	0.3 %
Grassland	0.1 %	0.1 %	0.1 %	0.1 %
Herbaceous Wetland	1.0 %	0.6 %	0.7 %	0.8 %
Mangroves	0.2 %	0.2 %	0.3 %	0.4 %
Moss and Lichen	0.0 %	0.0 %	0.0 %	0.0 %
Shrubland	0.2 %	0.0 %	0.1 %	0.1 %
Snow and Ice	0.5 %	0.5 %	0.6 %	0.3 %
Tree Cover	0.0 %	0.0 %	0.0 %	0.0 %
False Negatives	4.9 %	5.9 %	7.2 %	7.3 %

For scene 88, the dominant issue is the large number of false positives in the bare and sparsely vegetated areas of the scene. VV and VVHA produce the largest amount of false positives. The inclusion of the C_2 matrix reduces these errors, although scattered false positives remain. All models miss portions of the water body's boundaries, particularly in the northeastern section of the reservoir and along its narrow branches.

To assess whether false-positive detections are associated with specific land cover types, we analyze the distribution of false-positive pixels depending on land cover class. Table IV lists the percentage of false-positive pixels for each land cover type and experiment. *Bare/Sparse Vegetation* is the dominant source of false positives for all models, accounting for the largest percentage of falsely detected water pixels. *Cropland* and *Herbaceous Wetland* contribute the second most, although their influence varies between models. *Built-up*, *Grassland*, *Mangroves*, *Shrubland*, and *Snow and Ice* add relatively little to the false-positive rate. *Tree Cover* and *Moss and Lichen*

account for barely any false-positive detections.

Figure 7 shows false positives FP (%) across land cover types for each experiment and the change relative to the baseline experiment (VV). False-positive detections are primarily associated with bare/sparse vegetation, cropland, and herbaceous wetlands, whereas the contribution of the remaining land cover classes is only minor. The VV experiment alone struggles to distinguish water from bare or sparsely vegetated areas, agricultural lands, and wetland environments. Adding C_2 features reduces false positives suggesting that the additional covariance information improves discrimination between water and non-water surfaces for these land cover types. The inclusion of $H-A-\alpha$ decomposition features shows the strongest reduction for the land cover *Cropland*. However, false positives remain relatively high for *Bare/Sparse Vegetation* and *Herbaceous Wetland*. Thus, $H-A-\alpha$ appear beneficial for agricultural areas but less effective for sparsely vegetated surfaces or emergent vegetation. The VVC2HA combination has the lowest overall false-positive rate, although *Herbaceous Wetland* and *Bare/Sparse Vegetation* remain notable sources of false positives. However, the improvement in false-positive rate comes at the expense of an increased false-negative rate. Among all tested combinations, VVC2HA has the highest proportion of false negatives.

V. DISCUSSION

Since both the training data and the CNN architecture are based on the work of Wieland et al. [31], we first compare the results with those reported in their study. The VV band combination shows a performance similar to those achieved by their EfficientNet-B0 model using VV-VH-Slope as input features. Scenes with high IoU values in their study generally also perform well in the present study, while the same challenging scenes achieve lower IoU values. The main exception is that

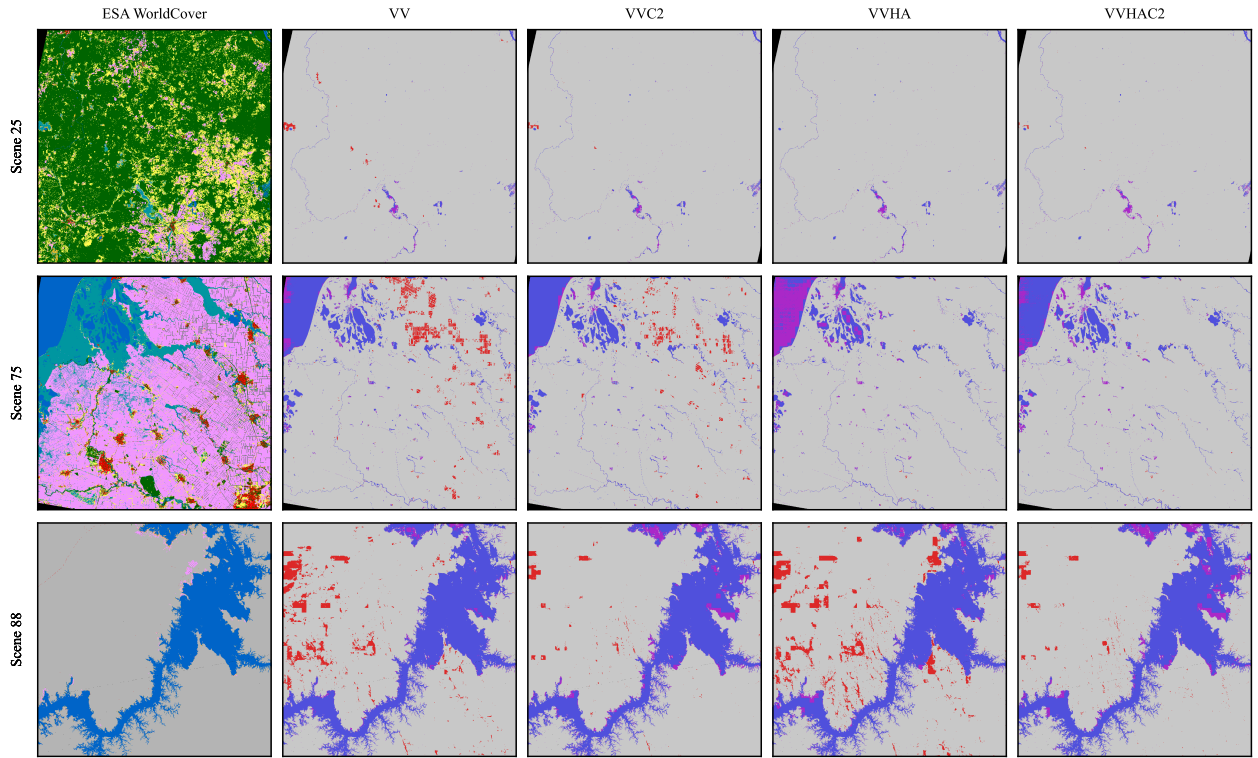


Fig. 6. Comparison of predictions for selected test scenes with low Precision values. Blue: true positives (water), gray: true negatives (non-water), red: false positives (wrongly detected water), purple: false negatives (missed water). Land cover types: Bare / Sparse Vegetation (gray), Built-up (red), Cropland (pink), Grassland (Yellow), Herbaceous Wetland (teal), Mangroves (green), Moss and Lichen (beige), Permanent Water Bodies (blue), Shrubland (orange), Snow and Ice (white), Tree Cover (dark green).

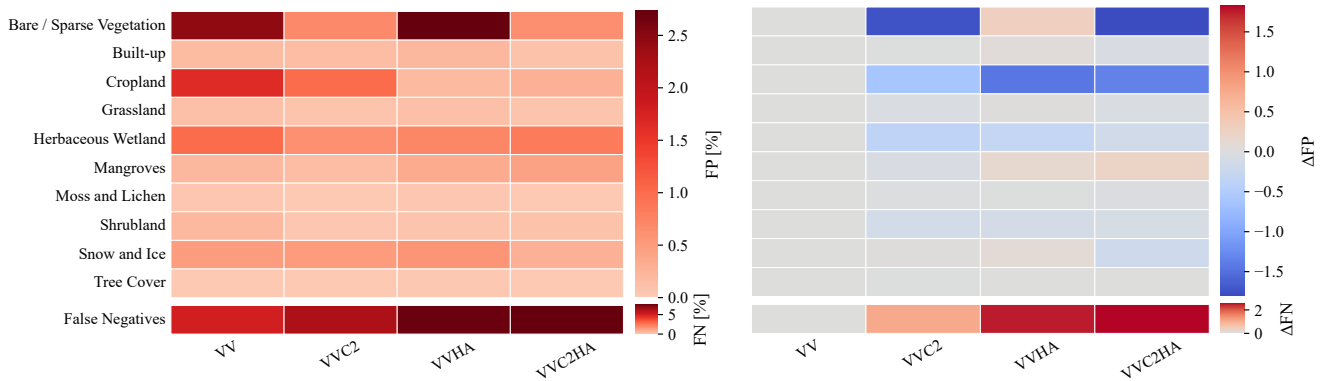


Fig. 7. Left: false positives FP (%) across land cover types and false negatives FN (%) for each experiment. Right: change relative to baseline experiment (VV). Positive values (red): increase in FP or FN compared to the baseline; negative values (blue): reduction in FP.

in this study scene 75 performs noticeably worse, whereas scene 77 achieves better results. Scene 75 suffers mainly from false positives in cropland areas while scene 77 has false negatives on the narrow rivers. The observed differences could be due to differing hyperparameters and the stochastic nature of deep neural network training. Small differences in model initialization or optimization can have an impact on the final segmentation results.

This study investigates whether polarimetric decomposition parameters improve Sentinel-1 surface water mapping when

used as additional inputs to a convolutional neural network. The overall high Precision for all feature combinations and the high variability in Recall suggest that the different combinations mainly differ in how well they detect water bodies. VV-based configurations have the most robust overall performance, particularly in terms of IoU and Recall. They produce higher median IoU and Recall values and lower variance between scenes, showing a better performance compared to the other configurations. The results suggest that SAR backscatter intensity bands should be included as input features for CNN-based

surface water segmentation as they reduce the number of false negatives. Adding polarimetric information mostly improves Precision at the cost of lowering Recall. This illustrates the trade-off between Precision and Recall: as additional features (HA and C_2) are incorporated, the models are less prone to falsely classifying land as water, but they are also more likely to miss true water pixels.

The performance is strongly dependent on the scene, with some scenes consistently segmented well by all configurations while others remain challenging regardless of the selected feature combination. While some scenes show slight improvement in segmentation performance through the inclusion of additional polarimetric features, for most scenes, the inclusion does not lead to significant improvements. The magnitude of the improvement depends on a number of factors, including the land cover composition and topography of the individual scene.

False negatives predominantly occur along narrow rivers and vegetated littoral zones. The inclusion of additional polarimetric features does not substantially reduce false negatives in any of these environments. Narrow river channels are particularly challenging. They are often only partially identified or not detected at all. Similar limitations for narrow rivers and complex water boundaries have also been reported by Guo et al. [17]. Kirby et al. [7] argue that terrain correction can artificially narrow rivers or remove shoreline pixels, reducing mapping accuracy. The limited effect of polarimetric information for narrow rivers is likely due to the spatial averaging inherent in decomposition products, which can blur small water bodies.

False-positive detections in SAR-based water mapping are often related to the backscatter characteristics of the surrounding land cover, i.e., false positives arise in land cover types with radar backscatter characteristics similar to those of open water. Under certain conditions, urban surfaces such as roofs and asphalt, bare soil, and agricultural fields can all produce low backscatter similar to open water. In mountainous terrain, radar shadow caused by steep slopes may also be misclassified as water because it has similarly low backscatter.

In this study, we investigate whether using polarimetric decomposition products as input features improves segmentation results for certain land cover types compared to amplitude-based input. The influence of land cover observed in this study is largely consistent with the findings of Wieland et al. [31]. Their Sentinel-1 model performed worst in sparsely vegetated areas, rain-fed cropland, and grassland. In this study, scenes dominated by sparse vegetation have low IoU values, while scenes with cropland show the greatest variability in performance. However, unlike Wieland et al., the VV baseline model in the present study does not perform poorly for grassland.

Bare/Sparse Vegetation is the biggest contributor of false positives for all models, accounting for the largest number of falsely detected water pixels. For this land cover class, adding polarimetric information, especially the C_2 matrix, can be beneficial. The percentage of false positives is significantly lower for VVC_2 while the number of false negatives increases only slightly. The VVC_2HA combination shows the highest reduction in false positives, although at the cost of the highest

rate of false negatives. Interestingly, the addition of the HA parameters introduces more false positives for this land cover type. A possible explanation is that the derived decomposition parameters may provide little additional information beyond what is already contained in the covariance matrix.

In the VV experiment, *Cropland* is the second-largest contributor to false-positive detections. In the test split of the S1S2-Water dataset, two scenes are dominated by agricultural land: scene 53 and 75. While scene 53 was captured in autumn, scene 75 was captured in early spring. Both scenes contain extensive areas of bare agricultural fields which have not yet been sown, have already been harvested, or may be left fallow at this point in time. These conditions may explain the high number of false positives observed in these scenes and why incorporating polarimetric data reduces the number of false positives. Recently plowed or fallow fields can have low surface roughness and, therefore, low radar backscatter. After rainfall or irrigation, agricultural fields may contain shallow standing water, which can reduce radar backscatter and make them resemble open water. Over the growing season, cropland changes considerably, leading to different backscatter characteristics depending on the acquisition date which may explain the high variability in performance. Including agricultural lands at all stages of vegetation growth in the training dataset is important for capturing the seasonal variability of cropland.

Models including decomposition products continue to struggle with vegetated littoral zones and wetlands. Both land cover types are inherently heterogeneous, containing a mixture of open water, saturated soil and emergent vegetation, sometimes within one pixel. Instead of specular reflection characteristic of open water, there is scattering from stems, leaves, and branches. Depending on the vegetation density, this can produce volume or double-bounce scattering, making these areas look more like terrestrial vegetation than open water. These scattering mechanisms are difficult to characterize using the limited information available from Sentinel-1 dual-polarization data. As Sentinel-1 operates at C-band, its radar signal has limited penetration capability into dense vegetation canopies, making it difficult to detect underlying water. Consequently, the Sentinel-1 decomposition parameters may provide limited additional information for identifying emergent vegetation.

Moreover, both the reference water masks and the ESA WorldCover dataset have limited accuracy in wetland areas. The reference water masks show inconsistencies in the way wetland areas are annotated. The *Herbaceous Wetland* class in ESA WorldCover is one of the least accurate of all land cover classes [46]. The results for the *Herbaceous Wetland* class should, therefore, be interpreted with caution. The transition from open water to emergent vegetation and wetland is often gradual, making it challenging to define a clear class boundary. Special care should be taken when mapping open water extent in wetland areas. We recommend developing guidelines that explicitly define where to draw boundaries between open water and emergent vegetation in herbaceous wetlands. Following these guidelines while creating a benchmarking dataset would ensure that the model learns the intended classification rules and that the performance evaluation is more accurate as well.

For the challenging land cover types investigated in this study, incorporating the C_2 matrix elements offers the best trade-off between reducing false-positive detections and only slightly increasing false negatives. However, the overall performance improvements are small and are primarily observed for bare or sparsely vegetated areas and cropland. Whether these improvements justify the additional preprocessing effort depends on the application. Generating polarimetric features requires processing Sentinel-1 SLC data to derive the covariance matrix and decomposition products, which is more computational effort than using backscatter intensities. Besides the preprocessing effort, these additional rasters increase the size of the dataset and the dimensionality of the CNN input tensor, resulting in higher memory consumption and computational requirements during both training and inference. If the additional channels provide only limited additional information, they may increase the risk of overfitting while providing only small improvements in segmentation performance. In this study, parts of the preprocessing workflow were performed manually. Automating this workflow by integrating ESA SNAP into a Python-based processing pipeline would reduce the required effort and processing time, making the inclusion of polarimetric features more practical for general application.

Despite the improvements, all models still have some false-positive as well as considerable false-negative detections, suggesting that some land cover types remain difficult to distinguish from water based on their radar backscatter characteristics alone. The additional polarimetric information may be incapable of resolving ambiguities between water and certain land cover types. One possible explanation is that this study is limited to Sentinel-1 dual-polarization (dual-pol) data, which only permits the computation of a C_2 matrix and H - α decomposition. Unlike decomposition methods derived from fully polarimetric quad-pol data, dual-pol decompositions provide only limited information on the underlying scattering mechanisms [29]. Consequently, the information available from the decomposition products is restricted, which may limit their ability to improve the discrimination of challenging land cover types. Other polarimetric decomposition methods could be evaluated to determine whether they provide more valuable information for surface water mapping than the decomposition used in this study. The model-based decomposition for dual-polarization SAR proposed by Mascolo et al. [53], which has recently been implemented in ESA SNAP, could be promising.

The used dataset consists of a relatively small number of scenes with a limited range of geographic regions and land cover classes. Some regions are underrepresented which may result in lower prediction performance in these regions. Moreover, land cover classes have variable characteristics across time and regions of the world. All this makes it difficult to ensure that a model trained on the dataset generalizes well to unseen regions and different times of the year. Expanding the dataset with additional scenes, particularly ones containing challenging land cover types, is likely to improve the performance of the model and its generalization ability. As the dataset is build upon freely available Sentinel data, a future expansion of the dataset should be relatively straightforward.

The quality of the reference water masks is another limitation of the used dataset. Although the annotations were manually refined, they still contain mapping inaccuracies, like mountain shadows mapped as water. As shown in this study, some lakes and rivers are only partially mapped, while wetlands and vegetated littoral zones are not always mapped consistently. Furthermore, the *SIS2-Water* dataset focuses on permanent water bodies, while floodwater and some important background classes, including urban areas, bare soil, and snow or ice, are comparatively underrepresented [31]. Annotation errors and class underrepresentations affect both model training and performance evaluation. Reference masks produced by expert using high-resolution optical imagery would be preferable. However, acquiring this data contemporaneously with Sentinel-1 acquisitions is often costly and the manual annotation of this data time-consuming. As a result, creating high-quality reference data still is a major challenge, even when supported by automated workflows. Wieland et al. [31] emphasize that rigorous quality control involving multiple independent experts, together with clear annotation guidelines, is key for creating reliable remote sensing benchmark datasets. Likewise, Bereczky et al. [20] argue that future improvements are more likely to come from expanding training datasets and incorporating auxiliary information rather than from increasingly complex CNN architectures. Overall, the results of this study support the conclusion that further progress in SAR-based surface water mapping depends not only on improved models but also on larger and more diverse high-quality datasets.

The land cover analysis is also subject to the limitations of the ESA WorldCover dataset. Since the WorldCover map is generated automatically from Sentinel-1 and Sentinel-2 imagery [44], it is affected by classification errors. The Sentinel-1 scenes used in ESA WorldCover and this study were not necessarily acquired in the same year or season, meaning that land cover may differ between the used datasets. Seasonal variations, for example fluctuating water levels in reservoirs or changes in land cover can lead to discrepancies between the mapped land cover class and the land cover at the time of the Sentinel-1 acquisition. Although the overall accuracy of ESA WorldCover is high, classification performance varies considerably between land cover classes and geographic regions. *Water Bodies*, *Tree Cover*, *Cropland*, *Bare / Sparse Vegetation*, and *Snow or Ice* are mapped with relatively high accuracy, whereas *Grassland*, *Built-up* areas, *Shrubland*, *Herbaceous Wetland*, and *Moss or Lichen* have lower accuracies [46]. Common confusion between classes include irrigated cropland with herbaceous wetlands, grassland with shrubland, and grassland with sparse vegetation. Accuracies for different continents range from 67.5% in Oceania to 80.7% in Asia [46]. The relatively low accuracy in Oceania is mainly attributed to confusion between the *Shrubland* and *Grassland* classes. The varying accuracy should be taken into account when interpreting the distribution of false-positive detections across land cover classes and regions, particularly for wetlands and grasslands. In addition, the WorldCover maps contain known artifacts, including classification errors for clouds and cloud shadows, and visible class boundaries

resulting from differences in acquisition [44].

Further research into using polarimetric decomposition products for improving results in areas of bare soil, sparse vegetation, and cropland is needed. Furthermore, the individual contribution of each decomposition parameter could be investigated by evaluating them separately to identify which parameters (e.g., entropy H or certain elements of the matrix C_2) provide the greatest benefit. Similarly, different weighting of the input features could be explored to assess whether emphasizing some polarimetric features allows the CNN to make better use of the information. The evaluation of deep learning architectures specifically designed for polarimetric SAR data could be a promising area of research. Since the covariance matrix contains complex-valued information, models capable of directly processing complex-valued inputs may perform better than conventional CNN [54].

Finally, the operational applicability of the proposed workflow could be improved by fully automating the preprocessing pipeline. The ESA SNAP Python interface (snappy) enables automated generation of decomposition products with minimal manual intervention. Such automation would reduce the effort of processing big datasets and enable more comprehensive studies into the best preparation and use of polarimetric products. One such study could systematically evaluate the effect of different preprocessing steps, like speckle filtering or decomposition method.

VI. CONCLUSION

In this study, we investigated whether dual-polarimetric decomposition parameters can improve CNN-based surface water mapping with Sentinel-1 SAR imagery. We developed a preprocessing workflow to generate dual-polarimetric decomposition products (C_2 and $H-A-\alpha$), which were combined with the *SIS2-Water* dataset and evaluated using different input feature combinations. In addition, we conducted an analysis based on the ESA WorldCover 2020 dataset to investigate the influence of land cover on segmentation performance and the potential of polarimetric information to reduce segmentation errors for specific land cover classes. We generated covariance matrix elements and dual-polarimetric $H-\alpha$ decomposition parameters for the *SIS2-Water* dataset. Using the newly created dataset, we trained a U-Net-based CNN with different combinations of intensity and polarimetric data.

Overall, the inclusion of additional polarimetric features resulted in only minor improvements in segmentation performance. The improvements were also strongly scene-dependent. For many scenes the additional polarimetric information produced only small improvements compared to using only the *VV-VH* intensity bands. To reduce false-negative detections, we recommend including the backscatter intensity bands as input features when using polarimetrically decomposed features as inputs.

The comparison with the ESA WorldCover map showed that the benefit of polarimetric information depends on the land cover type. Incorporating polarimetric information can improve segmentation performance by reducing false-positive detections in specific land cover types. False-positive detec-

tions were reduced for bare or sparse vegetation and cropland. In contrast, narrow rivers, vegetated littoral zones and wetlands remained challenging independent of input feature combination. These environments were most affected by false negatives that could not be reduced by integrating decomposition parameters.

However, the improvements have to be weighed against the increased preprocessing effort and computational complexity, since the overall improvement across all scenes remains limited. Future work should focus on improving both the quality and diversity of the training and evaluation datasets, particularly by improving the reference water masks and including additional scenes from different geographic regions. Further research should also investigate the effect of individual decomposition parameters, evaluate alternative decomposition methods, and explore deep learning architectures specifically designed for complex-valued polarimetric SAR data. Overall, this research has the potential to improve the accuracy and operational applicability of surface water mapping using Sentinel-1 imagery and deep learning.

REFERENCES

- [1] S. Grimaldi, Y. Li, V. R. N. Pauwels, and J. P. Walker, "Remote Sensing-Derived Water Extent and Level to Constrain Hydraulic Flood Forecasting Models: Opportunities and Challenges," *Surv. Geophys.*, vol. 37, no. 5, pp. 977–1034, Sep. 1, 2016. DOI: 10.1007/s10712-016-9378-y.
- [2] G. Schumann, L. Giustarini, A. Tarpanelli, B. Jarihani, and S. Martinis, "Flood Modeling and Prediction Using Earth Observation Data," *Surv. Geophys.*, vol. 44, no. 5, pp. 1553–1578, Oct. 1, 2023. DOI: 10.1007/s10712-022-09751-y.
- [3] P. Matgen, M. Montanari, R. Hostache, *et al.*, "Towards the sequential assimilation of SAR-derived water stages into hydraulic models using the Particle Filter: Proof of concept," *Hydrol. Earth Syst. Sci.*, vol. 14, no. 9, pp. 1773–1785, Sep. 27, 2010. DOI: 10.5194/hess-14-1773-2010.
- [4] S. Martinis, S. Groth, M. Wieland, L. Knopp, and M. Rättich, "Towards a global seasonal and permanent reference water product from Sentinel-1/2 data for improved flood mapping," *Remote Sens. Environ.*, vol. 278, p. 113077, Sep. 2022. DOI: 10.1016/j.rse.2022.113077.
- [5] S. Martinis, J. Kersten, and A. Twele, "A fully automated TerraSAR-X based flood service," *ISPRS J. Photogramm. Remote Sens.*, vol. 104, pp. 203–212, Jun. 1, 2015. DOI: 10.1016/j.isprsjprs.2014.07.014.
- [6] A. Twele, W. Cao, S. Plank, and S. Martinis, "Sentinel-1-based flood mapping: A fully automated processing chain," *Int. J. Remote Sens.*, vol. 37, no. 13, pp. 2990–3004, Jul. 2, 2016. DOI: 10.1080/01431161.2016.1192304.
- [7] K. Kirby, and I. and Nistor, "Accuracy of Surface Water Maps Derived from Radar Satellite Imagery Compared to Multispectral Satellite Imagery," *Can. J. Remote Sens.*, vol. 50, no. 1, p. 2433591, Dec. 31, 2024. DOI: 10.1080/07038992.2024.2433591.

- [8] C. Huang, Y. Chen, S. Zhang, and J. Wu, "Detecting, Extracting, and Monitoring Surface Water From Space Using Optical Sensors: A Review," *Rev. Geophys.*, vol. 56, no. 2, pp. 333–360, 2018. DOI: 10.1029/2018RG000598.
- [9] R. Nagaraj and L. S. Kumar, "Extraction of Surface Water Bodies using Optical Remote Sensing Images: A Review," *Earth Sci. Inform.*, vol. 17, no. 2, pp. 893–956, Apr. 1, 2024. DOI: 10.1007/s12145-023-01196-0.
- [10] J. M. F. d. S. Diniz, F. F. Gama, and M. Adami, "Evaluation of polarimetry and interferometry of sentinel-1A SAR data for land use and land cover of the Brazilian Amazon Region," *Geocarto Int.*, vol. 37, no. 5, pp. 1482–1500, 2020. DOI: 10.1080/10106049.2020.1773544.
- [11] "Copernicus Programme." (), [Online]. Available: <https://sentiwiki.copernicus.eu/web/copernicus-programme> (visited on 04/28/2025).
- [12] "Sentinel-1." (), [Online]. Available: <https://sentiwiki.copernicus.eu/web/sentinel-1> (visited on 04/28/2025).
- [13] F. Bioresita, A. Puissant, A. Stumpf, and J.-P. Malet, "A Method for Automatic and Rapid Mapping of Water Surfaces from Sentinel-1 Imagery," *Remote Sens.*, vol. 10, no. 2, p. 217, 2 Feb. 2018. DOI: 10.3390/rs10020217.
- [14] F. Zeng, C. Song, Z. Cao, *et al.*, "Monitoring inland water via Sentinel satellite constellation: A review and perspective," *ISPRS J. Photogramm. Remote Sens.*, vol. 204, pp. 340–361, Oct. 1, 2023. DOI: 10.1016/j.isprsjprs.2023.09.011.
- [15] F. Isikdogan, A. C. Bovik, and P. Passalacqua, "Surface Water Mapping by Deep Learning," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 10, no. 11, pp. 4909–4918, Nov. 2017. DOI: 10.1109/JSTARS.2017.2735443.
- [16] L. Landuyt, A. Van Wesemael, G. J.-P. Schumann, R. Hostache, N. E. C. Verhoest, and F. M. B. Van Coillie, "Flood Mapping Based on Synthetic Aperture Radar: An Assessment of Established Approaches," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 722–739, Feb. 2019. DOI: 10.1109/TGRS.2018.2860054.
- [17] Z. Guo, L. Wu, Y. Huang, Z. Guo, J. Zhao, and N. Li, "Water-Body Segmentation for SAR Images: Past, Current, and Future," *Remote Sens.*, vol. 14, no. 7, p. 1752, 7 Jan. 2022. DOI: 10.3390/rs14071752.
- [18] L. Pulvirenti, M. Chini, N. Pierdicca, and G. Boni, "Use of SAR Data for Detecting Floodwater in Urban and Agricultural Areas: The Role of the Interferometric Coherence," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1532–1544, Mar. 2016. DOI: 10.1109/TGRS.2015.2482001.
- [19] T. Hoeser and C. Kuenzer, "Object Detection and Image Segmentation with Deep Learning on Earth Observation Data: A Review-Part I: Evolution and Recent Trends," *Remote Sens.*, vol. 12, no. 10, p. 1667, 10 Jan. 2020. DOI: 10.3390/rs12101667.
- [20] M. Bereczky, M. Wieland, C. Krullikowski, S. Martinis, and S. Plank, "Sentinel-1-Based Water and Flood Mapping: Benchmarking Convolutional Neural Networks Against an Operational Rule-Based Processing Chain," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 15, pp. 2023–2036, 2022. DOI: 10.1109/JSTARS.2022.3152127.
- [21] R. Bentivoglio, E. Isufi, S. N. Jonkman, and R. Taormina, "Deep learning methods for flood mapping: A review of existing applications and future research directions," *Hydrol. Earth Syst. Sci.*, vol. 26, no. 16, pp. 4345–4378, Aug. 25, 2022. DOI: 10.5194/hess-26-4345-2022.
- [22] F. Pech-May, R. Aquino-Santos, and J. Delgadillo-Partida, "Sentinel-1 SAR Images and Deep Learning for Water Body Mapping," *Remote Sens.*, vol. 15, no. 12, p. 3009, 12 Jan. 2023. DOI: 10.3390/rs15123009.
- [23] V. Katiyar, N. Tamkuan, and M. Nagai, "Near-Real-Time Flood Mapping Using Off-the-Shelf Models with SAR Imagery and Deep Learning," *Remote Sens.*, vol. 13, no. 12, p. 2334, 12 Jan. 2021. DOI: 10.3390/rs13122334.
- [24] D. Bonafilia, B. Tellman, T. Anderson, and E. Issenberg, "Sen1Floods11: A georeferenced dataset to train and test deep learning flood algorithms for Sentinel-1," in *2020 IEEE CVF Conf. Comput. Vis. Pattern Recognit. Workshop CVPRW*, Jun. 2020, pp. 835–845. DOI: 10.1109/CVPRW50498.2020.00113.
- [25] A. Freeman and S. Durden, "A three-component scattering model for polarimetric SAR data," *IEEE Trans. Geosci. Remote Sens.*, vol. 36, no. 3, pp. 963–973, May 1998. DOI: 10.1109/36.673687.
- [26] Y. Yamaguchi, T. Moriyama, M. Ishido, and H. Yamada, "Four-component scattering model for polarimetric SAR image decomposition," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 8, pp. 1699–1706, Aug. 2005. DOI: 10.1109/TGRS.2005.852084.
- [27] Y. Yamaguchi, A. Sato, W.-M. Boerner, R. Sato, and H. Yamada, "Four-Component Scattering Power Decomposition With Rotation of Coherency Matrix," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 6, pp. 2251–2258, Jun. 2011. DOI: 10.1109/TGRS.2010.2099124.
- [28] S. Cloude and E. Pottier, "An entropy based classification scheme for land applications of polarimetric SAR," *IEEE Trans. Geosci. Remote Sens.*, vol. 35, no. 1, pp. 68–78, Jan. 1997. DOI: 10.1109/36.551935.
- [29] S. Cloude, "The Dual Polarization Entropy/Alpha Decomposition: A PALSAR Case Study," *Proc. 3rd Int. Workshop Sci. Appl. SAR Polarim. Polarim. Interferom.*, ESA Special Publication, vol. 644, p. 2, 2007.
- [30] Y. Long, G.-S. Xia, S. Li, *et al.*, "On Creating Benchmark Dataset for Aerial Image Interpretation: Reviews, Guidances, and Million-AID," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 14, pp. 4205–4230, 2021. DOI: 10.1109/JSTARS.2021.3070368.
- [31] M. Wieland, F. Fichtner, S. Martinis, *et al.*, "S1S2-Water: A Global Dataset for Semantic Segmentation of Water Bodies From Sentinel- 1 and Sentinel-2 Satellite

- Images,” *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 17, pp. 1084–1099, 2024. DOI: 10.1109/JSTARS.2023.3333969.
- [32] C. D. S. Ecosystem. “Copernicus DEM - Global and European Digital Elevation Model — Copernicus Data Space Ecosystem.” (), [Online]. Available: <https://dataspace.copernicus.eu/explore-data/data-collections/copernicus-contributing-missions/collections-description/COP-DEM> (visited on 07/12/2026).
- [33] “STAC: SpatioTemporal Asset Catalogs.” (), [Online]. Available: <https://stacspec.org/en/> (visited on 06/14/2024).
- [34] M. Wieland, F. Fichtner, S. Martinis, *et al.*, *SIS2-Water: A global dataset for semantic segmentation of water bodies from Sentinel-1 and Sentinel-2 satellite images*, version v1.0.1, Zenodo, May 24, 2024. DOI: 10.5281/zenodo.8314174.
- [35] M. Wieland, *SIS2-Water GitHub Repository*, Jul. 31, 2024.
- [36] X. Ma, J. Xu, P. Wu, and P. Kong, “Oil Spill Detection Based on Deep Convolutional Neural Networks Using Polarimetric Scattering Information From Sentinel-1 SAR Images,” *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–13, 2022. DOI: 10.1109/TGRS.2021.3126175.
- [37] “Copernicus Data Space Ecosystem APIs.” (), [Online]. Available: <https://dataspace.copernicus.eu/analyse/apis> (visited on 06/19/2024).
- [38] “STEP – Science Toolbox Exploitation Platform.” (), [Online]. Available: <https://step.esa.int/main/> (visited on 06/14/2024).
- [39] J.-S. Lee, “Refined filtering of image noise using local statistics,” *Comput. Graph. Image Process.*, vol. 15, no. 4, pp. 380–389, Apr. 1, 1981. DOI: 10.1016/S0146-664X(81)80018-4.
- [40] J. S. Lee, L. Jurkevich, P. Dewaele, P. Wambacq, and A. Oosterlinck, “Speckle filtering of synthetic aperture radar images: A review,” *Remote Sens. Rev.*, vol. 8, no. 4, pp. 313–340, Feb. 1, 1994. DOI: 10.1080/02757259409532206.
- [41] S. Plank, M. Jüssi, S. Martinis, and A. Twele, “Mapping of flooded vegetation by means of polarimetric Sentinel-1 and ALOS-2/PALSAR-2 imagery,” *Int. J. Remote Sens.*, vol. 38, no. 13, pp. 3831–3850, Jul. 3, 2017. DOI: 10.1080/01431161.2017.1306143.
- [42] “Cloud Optimized GeoTIFF.” (), [Online]. Available: <https://www.cogeotiff.com/> (visited on 06/14/2024).
- [43] D. Zanaga, R. Van De Kerchove, W. De Keersmaecker, *et al.*, *ESA WorldCover 10m 2020 v100*, version v100, 2021. DOI: 10.5281/zenodo.5571936.
- [44] R. Van De Kerchove, D. Zanaga, L. Li, N. Tsendbazar, and M. Lesiv, *ESA WorldCover Product User Manual v1.0*, Oct. 19, 2021.
- [45] F. a. A. O. of the United Nations (FAO). “Land Cover Classification System (LCCS) — Land & Water — Food and Agriculture Organization of the United Nations — Land & Water — Food and Agriculture Organization of the United Nations.” (2026), [Online]. Available: <https://www.fao.org/land-water/land/land-governance/land-resources-planning-toolbox/category/details/en/c/1036361/> (visited on 03/27/2026).
- [46] N. Tsendbazar, L. Li, M. Koopman, *et al.*, *ESA WorldCover Product Validation Report v1.1*, Oct. 22, 2021.
- [47] E. W. Project. “WorldCover Viewer,” ESA WorldCover Viewer. (2021), [Online]. Available: <https://viewer.esa-worldcover.org/worldcover> (visited on 03/31/2026).
- [48] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” in *Med. Image Comput. Comput.-Assist. Interv. – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., Cham: Springer International Publishing, 2015, pp. 234–241. DOI: 10.1007/978-3-319-24574-4_28.
- [49] M. Wieland and S. Martinis, “A Modular Processing Chain for Automated Flood Monitoring from Multi-Spectral Satellite Data,” *Remote Sens.*, vol. 11, no. 19, p. 2330, Oct. 8, 2019. DOI: 10.3390/rs11192330.
- [50] M. Tan and Q. V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.” arXiv: 1905.11946. (Sep. 11, 2020), [Online]. Available: <http://arxiv.org/abs/1905.11946> (visited on 11/20/2024), pre-published.
- [51] I. Loshchilov and F. Hutter. “Decoupled Weight Decay Regularization.” arXiv: 1711.05101 [cs]. (Jan. 4, 2019), [Online]. Available: <http://arxiv.org/abs/1711.05101> (visited on 10/17/2025), pre-published.
- [52] E. O. C. (A. Center). “ukis_metrics v0.1.4.” (Nov. 20, 2024), [Online]. Available: <https://github.com/dlr-eoc/ukis-metrics> (visited on 11/20/2024).
- [53] L. Mascolo, S. R. Cloude, and J. M. Lopez-Sanchez, “Model-Based Decomposition of Dual-Pol SAR Data: Application to Sentinel-1,” *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–19, 2022. DOI: 10.1109/TGRS.2021.3137588.
- [54] R. Hänsch, “Complex-Valued Multi-Layer Perceptrons – An Application to Polarimetric SAR Data,” *Photogramm. Eng. Remote Sens.*, vol. 76, no. 9, pp. 1081–1088, Sep. 1, 2010. DOI: 10.14358/PERS.76.9.1081.

AUTHORS



Lucie Patzwahl received the B.Sc. degree in Aerospace Engineering from University of Stuttgart, Germany, in 2014, and the M.Sc. degree in Aerospace Engineering from Technical University of Munich, Germany, in 2019. She is currently studying Geographical Information Science & Systems at Paris Lodron University of Salzburg.

3 Technical Report

This thesis aims to contribute to the research into using polarimetric data for surface water mapping by investigating the use of dual-polarimetric decomposition products derived from Sentinel-1 data as input features to a convolutional neural network. For this purpose, I want to answer following research questions:

1. For Sentinel-1, does the integration of a polarimetric decomposition method into deep learning algorithms improve the accuracy of water surface segmentation compared to purely amplitude-based inputs?
2. Which Sentinel-1 polarimetric decomposition product (e.g., covariance matrix elements or $H-A-\alpha$ parameters) provides the highest segmentation accuracy for water surfaces when used as input for deep learning algorithms?
3. Does the application of polarimetric decomposition methods improve segmentation results for specific land cover types (e.g., sand, bare soil, flooded vegetation) compared to the amplitude-based method?

To answer the research questions, I enrich an existing training dataset of Sentinel-1 SAR scenes with polarimetric decomposition products. Section 3.1 introduces the used datasets in more detail compared to the manuscript and describes the workflow to prepare the polarimetric data. An existing convolutional neural network designed by researchers at the DLR for flood segmentation is used in this thesis. Section 3.2 presents the network architecture and the experimental design choices in more detail.

3.1 Data

This thesis builds upon the S1S2-Water dataset, an existing benchmark dataset for deep learning-based surface water segmentation, which is described in detail in Section 3.1.1. To investigate the influence of additional polarimetric information, I extended the dataset with dual-polarimetric decomposition products. The preprocessing workflow used to create these products, the challenges associated with processing Sentinel-1 SLC data, and the resulting dataset are described in Section 3.1.2. To assess whether the effectiveness of the decomposition products depends on land cover type, I analyzed the segmentation results using the *ESA WorldCover 2020* land-cover map, which is described in Section 3.1.3.

3.1.1 S1S2-Water Dataset

For this thesis, I use the *S1S2-Water* dataset, a global benchmarking dataset for semantic segmentation of surface water bodies introduced by Wieland et al. [3]. The dataset comprises 65 Sentinel-1 Ground Range Detected (GRD) Interferometric Wide (IW) swath scenes, each covering an area of $100 \text{ km} \times 100 \text{ km}$ and aligned with the Sentinel-2 tile grid. Each Sentinel-1 image contains the VV and VH backscatter intensity bands, which are scaled by a factor of 100 and stored as integer values to reduce file size. In addition, the dataset includes Sentinel-2 Level-1C imagery, which is an orthorectified top-of-atmosphere (TOA) reflectance product. For the same reason as for the Sentinel-1 data, the pixel values are scaled by a factor of 10000. The dataset includes slope and elevation derived from the Copernicus 30 m Digital Elevation Model (DEM) [4]. To ensure that only valid areas are considered during testing, valid pixel masks exclude areas without Sentinel-1 coverage as well as regions affected by clouds in the Sentinel-2 imagery. Each file follows the naming convention `sentinel12_<SENSOR>_<ID>_<LAYER>.tif`, e.g., `sentinel12_s1_75_img.tif`. An overview of the dataset layers and naming convention is provided in Table 3-1.

The images used in the dataset were acquired between May 2018 and November 2020, with the majority of the scenes from 2019. The dataset comprises permanent water bodies in 29 countries. The global distribution of

Table 3-1: S1S2-Water dataset components (adapted from [5])

Sensor	Layer	Description	Values	Format	Bands
s1	img	Sentinel-1 GRD	dB (scaled by factor 100)	GeoTIFF Int16	0: VV 1: VH
s2	img	Sentinel-2 L1C	TOA reflectance (scaled by factor 10000)	GeoTIFF UInt16	0: Blue 1: Green 2: Red 3: NIR 4: SWIR1 5: SWIR2
s1/s2	msk	water mask	0: no water 1: water	GeoTIFF UInt8	0: water mask
s1/s2	valid	valid pixel mask	0: invalid 1: valid	GeoTIFF UInt8	0: valid mask
copdem30	elevation	Copernicus DEM elevation	meters	GeoTIFF Int16	0: elevation
copdem30	slope	Copernicus DEM slope	degrees	GeoTIFF Int16	0: slope
–	meta	metadata	STAC metadata item	JSON	–

the dataset is shown in Figure 3-1. The majority of the scenes are located in the Northern Hemisphere, with a particular concentration in Europe and North America. In contrast, the Southern Hemisphere is represented by fewer samples. This distribution is expected, as the Southern Hemisphere contains less land mass with suitable permanent water bodies. Equatorial regions are noticeably absent from the dataset. The frequent occurrence of clouds and cloud shadows in regions near the equator hampers the acquisition of suitable Sentinel-2 imagery. Wieland et al. [3] identify the limited availability of samples affected by challenging atmospheric conditions, including dense cloud cover, haze, and smoke, as one of the main limitations of their dataset.

Figure 3-2 illustrates the distribution of land cover types within the dataset based on the ESA WorldCover classification. According to this classification, *Permanent Water Bodies* account for slightly more than one tenth of the pixels in the dataset. The most frequent land cover type is *Tree Cover*, which takes up more than one third of the observed scenes. Other land cover classes, including *Grassland*, *Cropland*, and *Bare / Sparse Vegetation*, are also well represented. *Built-up*, *Herbaceous Wetland*, *Shrubland*, and *Moss and Lichen* areas take up a small proportion of the dataset. *Snow and Ice* and *Mangroves* only have a very limited number of samples in the dataset.

The dataset includes binary water masks which were derived with the help of an NDWI threshold and subsequently manually refined. Sentinel-1 and Sentinel-2 acquisitions cannot always be obtained contemporaneously. Therefore, the dataset provides separate water masks for each sensor to ensure that the masks accurately represent the water extent visible in the corresponding image. This is particularly important for regions where water extent can change drastically within a few days. The average temporal difference between Sentinel-1 and Sentinel-2 acquisitions in the *S1S2-Water* dataset is one day, with a standard deviation of four

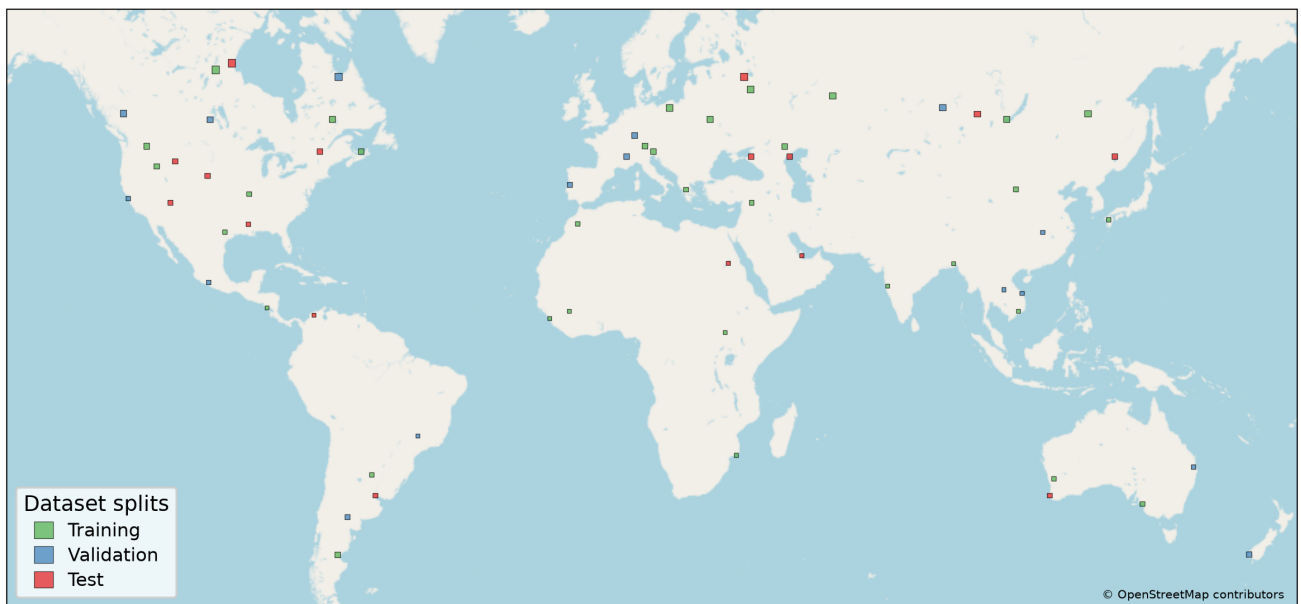


Figure 3-1: Distribution of S1S2-Water scenes

days. The water masks are directly derived from the Sentinel satellite images rather than from independent sources, e.g., high-resolution optical imagery. This approach is unsuitable for datasets used for model benchmarking as it is important to maintain accurate spatio-temporal alignment between input imagery and reference masks. Wieland et al. [3] argue that spatial and temporal discrepancies between the source of the reference masks and the corresponding input image can introduce label noise into a dataset.

A *SpatioTemporal Asset Catalog* (STAC) [6] complements the dataset by providing standardized metadata. The metadata is both machine-readable and human-readable and thus can be automatically processed while remaining accessible for manual interpretation and adjustment by the user. The STAC specification 'provides a common structure for describing and cataloging spatiotemporal assets' [6]. For this dataset, the used STAC specifications are STAC Item and STAC Catalog. A STAC Item is the smallest unit, representing a single spatio-temporal item as a GeoJSON including geometry, date time and links to files. A STAC Catalog is a JSON file of links to single STAC Items that provides a structure to organize and browse the assets. Both are flexible and can be easily extended and adapted when the dataset is changed. I use the STAC metadata in conjunction with *PySTAC* [7], a Python library for working with STAC.

All layers are stored as *Cloud Optimized GeoTIFF* (COG), which in combination with the use of STAC, makes the dataset suitable for cloud processing. The dataset has a fixed division into training (50%), validation (25%) and test (25%) splits to ensure reproducibility. The *S1S2-Water* dataset [8] as well as preparation scripts and a SNAP graph for preprocessing Sentinel-1 [5] are available for download. Figure 3-3 shows a few examples of the scenes contained in the dataset.

Limitations of the S1S2-Water Dataset

The dataset consists of a relatively small number of scenes, resulting in a limited diversity of geographic regions and land cover types. As a result, some important land cover classes are represented by only a limited number of samples, which may affect the generalization capability of the trained models. Challenging land cover types, such as bare or sparsely vegetated landscapes and certain wetland types, are also underrepresented. Land cover classes can have different characteristics in different parts of the world, making it difficult to ensure that a

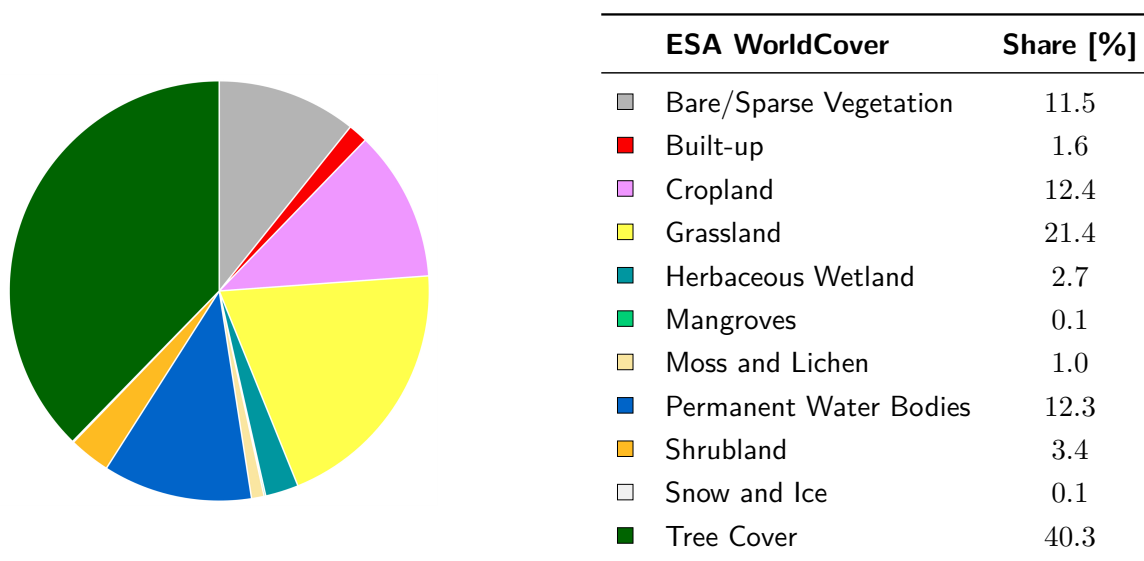


Figure 3-2: Share of ESA WorldCover land cover types in S1S2-Water dataset

model trained on the available scenes generalizes well to other geographic regions. The *S1S2-Water* dataset also includes limited samples containing challenging atmospheric conditions, such as dense cloud cover, haze, or smoke [3]. Expanding the dataset with additional scenes, particularly those with underrepresented land cover types and geographic areas, is therefore likely to improve the performance and generalization performance of the models.

The quality of the reference water masks represents another limitation of the dataset. Although the annotations were manually refined, they are derived from the same Sentinel-1 and Sentinel-2 imagery that is used for model development and still contain mapping inaccuracies. As shown in this study, some lakes and rivers are only partially mapped, while wetlands, vegetated littoral zones, and mountain shadows are not always mapped consistently.

Figure 3-4 presents two scenes that contain apparent annotation errors. The detail of scene 35 shows a mountainous area containing numerous small lakes. A comparison of the prediction mask with the Sentinel-1 and Sentinel-2 images reveals that several lakes, which are clearly visible in both datasets, are missing from the reference water mask and are therefore marked as false positive predictions (red). On the other hand, an area near the top of the image is labeled as water in the reference water mask (purple), although it is shadows cast by the surrounding mountains.

The detail of scene 53 shows a river that is almost completely missing from the reference water mask. However, the river is clearly visible in both the Sentinel-1 and Sentinel-2 imagery and is also mapped in the ESA WorldCover dataset. These examples demonstrate that, even though the reference water masks were manually refined, some water bodies remain unmapped while other areas are incorrectly annotated as water.

Annotation errors and the underrepresentation of certain land cover classes affect both model training and the evaluation of the prediction performance. Reference water masks produced by experienced annotators using high-resolution optical imagery are usually preferable. However, acquiring such data contemporaneously with Sentinel acquisitions is often difficult and costly. As a result, creating high-quality reference data remains a major challenge, as the annotation process is both time- and labor-intensive, even when supported by automated workflows.

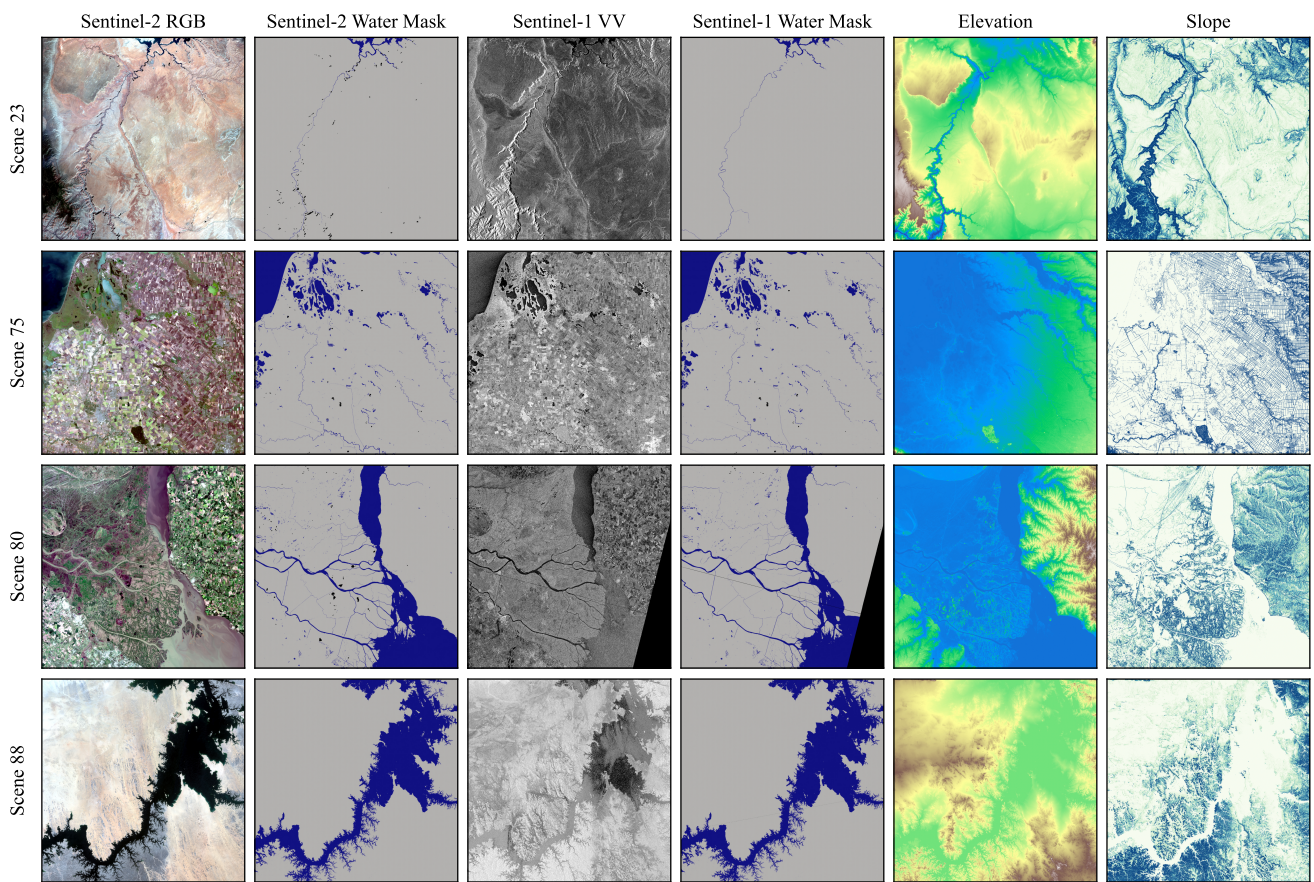


Figure 3-3: Examples of S1S2-Water samples: Sentinel-2 (RGB), Sentinel-1 (VV polarization), corresponding water masks, elevation and slope information.

3.1.2 Polarimetrically Decomposed Sentinel-1 Images

For each sample in the *S1S2-Water* dataset, I create a set of polarimetric decomposition products. The generation of these products requires Sentinel-1 Single Look Complex (SLC) data, as these data preserve the phase information required for polarimetric decomposition. I perform the following processing steps to derive the decomposition products:

1. Downloading Sentinel-1 SLC for every scene (Section 4.2).
2. Processing SLC in SNAP to obtain polarimetric decomposition products (Section 4.3).
3. Processing SLC in Python for CNN input generation (Section 4.4).

The first step of the processing workflow is the acquisition of the corresponding Sentinel-1 SLC products for each sample. For the retrieval of the matching SLC products I use the Copernicus Data Space Ecosystem (CDSE) API [9]. The script used for downloading the products is provided in Section 4.2. I use the provided STAC Catalog to browse the individual scenes and obtain their acquisition dates and geometries. This information is used to identify the Sentinel-1 SLC product ID for each scene. The ID is necessary to download the SLC products by issuing a GET request to the CDSE API. This requires a valid username and password, which are stored separately and accessed by the download script. The downloaded products are provided as compressed ZIP archives and must be extracted before further processing.

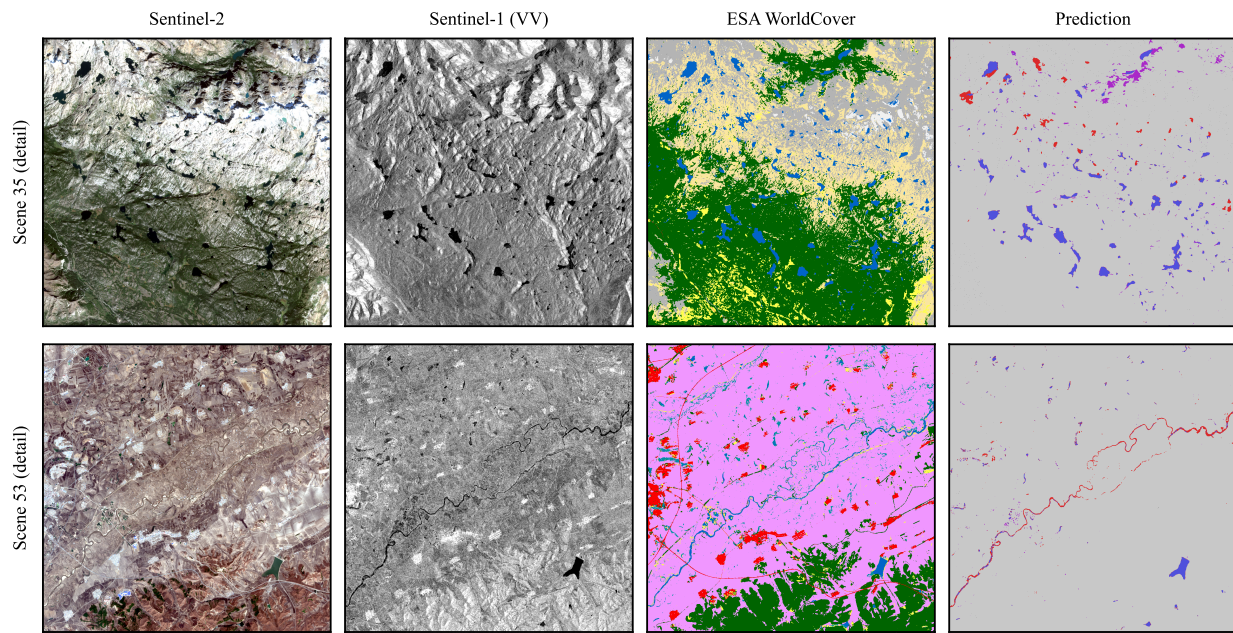


Figure 3-4: S1S2-Water samples with annotations errors: Sentinel-2 (RGB), Sentinel-1 (VV polarization), ESA WorldCover, Prediction (VV)

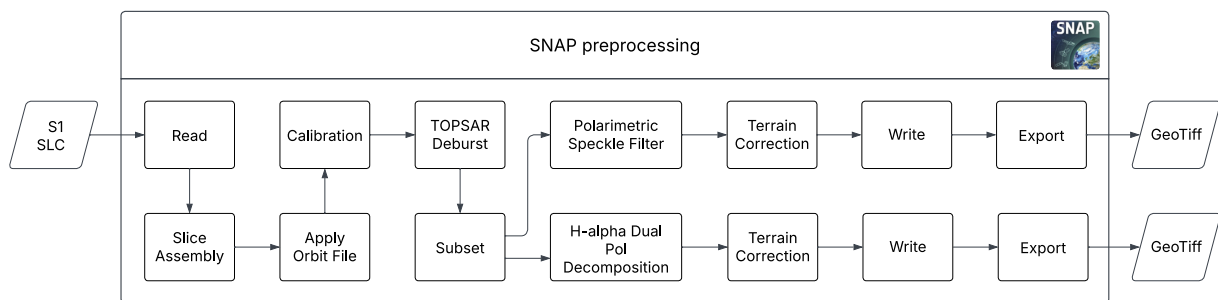


Figure 3-5: SNAP processing graph.

After downloading, I preprocess the Sentinel-1 SLC products with the help of the *Graph Processing Tool* (GPT) of the ESA *SeNtinel Application Platform* (SNAP) [10]. The complete processing chain is illustrated in Figure 3-5. Executing the entire workflow as a single GPT graph exceeds the available memory of SNAP and frequently results in program crashes. Therefore, the workflow is divided into six smaller processing graphs with intermediate results saved as BEAM-DIMAP (the standard I/O product format in SNAP) after each stage. The individual SNAP graphs are provided in Section 4.3. The SNAP workflow is divided into the following stages:

1. Slice Assembly (Code 4-4),
2. Apply Orbit File, Calibration and TOPSAR Deburst (Code 4-5),
3. Subset (Code 4-6),
4. Polarimetric Speckle Filter for C_2 matrix (Code 4-7),
5. Polarimetric Decomposition for H , A and α (Code 4-8),

6. Terrain Correction (Code 4-9).

Dividing the workflow into multiple processing graphs requires manual intervention to initiate each subsequent processing step. To reduce the effort, I use a batch processing script for SNAP GPT [11] to execute each processing stage for all scenes automatically. The script requires an input directory containing the files to be processed, an output directory for the generated results, the path to the SNAP processing graph, and the file extensions of the input and output products. Based on these parameters, the script generates a batch file that can be executed from the command prompt to sequentially process all files in the input directory. Even though this automates the execution of each processing stage, it still needs manual intervention to initiate the next step in the overall workflow. This workflow could be improved by fully automating the workflow with the help of the SNAP API from Python. Unfortunately, I was not able to make the SNAP Python interface operational within the scope of this thesis.

After the SNAP processing, I perform additional post-processing in Python to generate the input data for the convolutional neural network (CNN). The corresponding scripts are provided in Section 4.4. The processing steps in Python are:

1. Merging of neighboring slices if necessary (Code 4-10),
2. Resampling and reprojection of polarimetric data to the reference band (Code 4-11),
3. Creation of multi-band input stacks for CNN processing (Code 4-11),
4. Replacement of NaN values in selected layers (Code 4-12),
5. Creation of valid pixel masks (Code 4-13).

All scripts expect an input directory containing subdirectories with the GeoTIFF files exported from SNAP. Some scenes span adjacent Sentinel-1 orbits and must be merged to provide complete coverage of an *Area of Interest* (Aoi). I merge the images using `merge.py`, which merges all files with the `.tif` extension in a folder into a single image.

For use in the deep learning segmentation model, the processed layers are combined into a single multi-band raster. Specifically, the two Sentinel-1 intensity bands, the four bands of the C_2 matrix, and the three decomposition parameter bands are stacked into a single file. Optionally, the slope can be appended during this step. For the *SIS2-Water* dataset, however, the slope layer is added during the subsequent dataset tiling and splitting. During the stacking, the VV band serves as the reference layer for resampling to ensure that the raster pixels are spatially aligned. The resulting image stack is in the Universal Transverse Mercator (UTM) coordinate system. The stacking operation is implemented in `stack.py`.

The processing workflow introduces *Not a Number* (NaN) values in pixels for which no valid data is available. Before the data can be used as input to the CNN, these values are replaced with zeros using `replace_nan.py`. The resulting multi-band raster is then saved as *Cloud Optimized GeoTIFF* (COG) [12] to maintain consistency with the file format of the *SIS2-Water* dataset.

Some scenes contain regions without valid data. Therefore, a validity mask is generated which assigns a value of 1 to valid pixels and 0 to invalid pixels. The bands used to derive the mask can be specified by the user. The mask is generated using `calc_valid_mask.py` and is subsequently stored as COG. The resulting layers are listed in Table 3-2.

To avoid conflicts with the existing dataset and reduce data duplication, I create a new version of the dataset that only contains the Sentinel-1 assets and other relevant data. The STAC Catalog and Items are updated to reflect the changes made to the dataset. An example of an Item and an abridged version of the Catalog are provided in Section 4.1. In the Catalog file, the `id`, `title`, and `description` have to be modified. For the Item files, all assets associated with Sentinel-2 data are removed. Additionally, the `name` and `href` of the items are changed to reference the renamed files. The `s1_srcids` field is updated to include the ID of the

SLC product.

Table 3-2: S1S2-Water-Pol dataset components

Sensor	Layer	Description	Values	Format	Bands
s1	img_pol	Sentinel-1 GRD C_2 elements H , A , α	0-1: dB (scaled by factor 100) 2-5: $[-1..1]$ 6: $[0..1]$ 7: $[0..1]$ 8: degrees $[0..90]$	GeoTIFF Float32	0: VV 1: VH 2: C_{11} 3: $C_{12,real}$ 4: $C_{12,img}$ 5: C_{22} 6: H 7: A 8: α
s1	msk	water mask	0: no water 1: water	GeoTIFF UInt8	0: water mask
s1	valid	valid pixel mask	0: invalid 1: valid	GeoTIFF UInt8	0: valid mask
copdem30	elevation	Copernicus DEM elevation	meters	GeoTIFF Int16	0: elevation
copdem30	slope	Copernicus DEM slope	degrees	GeoTIFF Int16	0: slope
–	meta	metadata	STAC metadata item	JSON	–

Figure 3-6 presents examples of the resulting polarimetric decomposition products for several test scenes. Due to the dual-polarization acquisition mode of Sentinel-1, the derived parameters do not have a direct physical interpretation. As a result, interpreting the individual parameters shown in the figure is not straightforward. Nevertheless, several observations can be made regarding the appearance of water surfaces in the different decomposition products.

Water surfaces generally appear dark in the C_{11} and C_{22} parameters. However, open soil and bare landscapes can also have low values, as shown in scenes 23 and 88. In the real and imaginary parts of the C_{12} parameter, water surfaces appear relatively smooth, whereas the surrounding areas have a more textured appearance. This is particularly evident for vegetated areas, which generally appear grainier.

The appearance of water in the entropy H images is less consistent. Depending on the scene, water can either appear bright, as in scenes 23 and 88, or relatively dark, as in scenes 75 and 80. The angle α shows a similar behavior, as water also appears either bright or dark depending on the scene. In contrast, the opposite is true for the anisotropy A , with water appearing dark in scenes 23 and 88 and relatively bright in scenes 75 and 80.

Challenges during the Creation of the Polarimetric Products

Preparing and processing the data was the most time-consuming part of this work. Sentinel-1 data has large file sizes, meaning that even the initial download requires a considerable amount of time. Handling data of this size also requires additional memory and storage capacity. As already mentioned, executing the complete

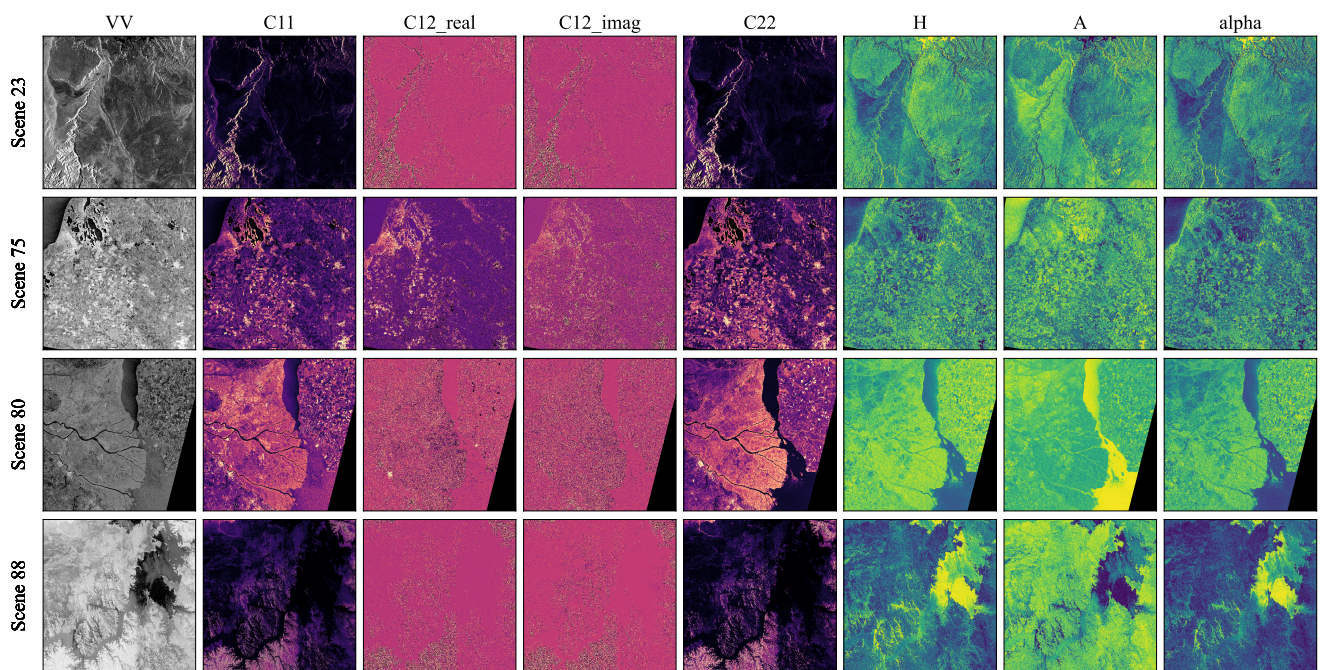


Figure 3-6: Examples of polarimetrically decomposed Sentinel-1 scenes (false color).

workflow as a single GPT graph exceeded the available memory and frequently resulted in program crashes. Dividing the workflow into smaller processing graphs reduced the memory requirements, but required frequent manual intervention. Processing all files using the batch processing tool in SNAP was not successful, which is why the described approach with a batch file was used instead. Some processing steps, such as terrain correction, required manual adjustment of parameters because they were not automatically updated during batch processing. Integrating the SNAP processing steps into a Python-based processing pipeline and thereby automating the complete workflow for creating the image stack of CNN input features would significantly reduce the required effort and make the use of polarimetric data in applications more feasible. However, I was not able to implement further automation of the workflow during my time at DLR. During the time of processing the data, colleagues had already attempted to deploy a SNAP processing pipeline on the cloud processing cluster, but these attempts were unsuccessful.

For one of the testing scenes (scene 31), there is no corresponding Sentinel-1 SLC data at the required date. I, therefore, excluded this scene from the testing split. The training and validation splits included all scenes from the original dataset.

3.1.3 ESA WorldCover Map

Wieland et al. [3] analyze the segmentation performance of their models with respect to the most common land cover type in each scene. They label each sample with the land cover class that occupies the largest share of the scene. During my preliminary analyses, however, it became apparent that the majority of false-positive predictions frequently did not occur in the predominant land cover class. For example, scenes labelled as *Tree Cover* often have the highest number of false positives in *Herbaceous Wetlands* or *Cropland*. Comparing the segmentation performance solely based on the scenes' land cover label may hide the actual sources of misclassified pixels. To obtain a more detailed picture of where misclassifications appear, I therefore carry out a pixel-based analysis, evaluating the performance separately for each land cover class contained in each scene.

For this purpose, I use ESA WorldCover 2020 [13]. WorldCover is a global land cover map with a spatial resolution of 10 m based on the Sentinel-2 UTM grid [14]. The map follows the Land Cover Classification System (LCCS) developed by the Food and Agriculture Organization of the United Nations (FAO) [15] and distinguishes 11 land cover classes: *Tree Cover*, *Shrubland*, *Grassland*, *Cropland*, *Built-up*, *Bare / Sparse Vegetation*, *Snow and Ice*, *Permanent Water Bodies*, *Herbaceous Wetland*, *Mangroves*, and *Moss and Lichen*.

Unlike many previous global land cover datasets that rely exclusively on optical imagery, WorldCover combines Sentinel-1 C-band SAR and Sentinel-2 multispectral data, improving the robustness of the classification in regions affected by frequent cloud cover. In addition to the Sentinel-1 and Sentinel-2 imagery, auxiliary data such as elevation and slope derived from a digital elevation model are also used. During data preprocessing, Sentinel-2 Level-2A images are filtered to remove clouds, cloud shadows, cirrus, and saturated pixels. The Sentinel-1 GRD images are terrain corrected and converted to Gamma0 backscatter. Both datasets are then geocoded to the Sentinel-2 grid. Finally, temporal composites are generated to reduce noise and compensate for missing observations.

During feature extraction and classification, statistical features are computed from the Sentinel-1 and Sentinel-2 time series. Additional information, including terrain elevation, terrain slope, and geographical location, is also incorporated into the dataset. In total, 131 input features are used for the classification. The classification algorithm is based on the algorithm developed for the dynamic yearly Copernicus Global Land Service Land Cover (CGLS-LC) product. A gradient boosting decision tree classifier is trained using globally distributed reference samples and subsequently applied on a global scale to produce the final land cover map. The resulting map tiles can be downloaded from the ESA WorldCover Viewer website [16].

WorldCover was independently validated using a global validation dataset [17]. Although the overall accuracy of ESA WorldCover is relatively high, the classification performance varies considerably between land cover classes and geographic regions. The independent validation reported a global overall accuracy of 74,4%. Accuracies range from approximately 67,5% in Oceania to 80,7% in Asia. The relatively low accuracy in Oceania is mainly attributed to confusion between the *Shrubland* and *Grassland* classes.

Water bodies, tree cover, cropland, bare or sparse vegetation, and snow or ice are mapped with relatively high accuracy, whereas grasslands, built-up areas, shrublands, wetlands, and moss or lichen have lower accuracies [17]. These differences should be considered when interpreting the distribution of false positives across the different land cover classes, particularly for wetlands and grasslands. In the experiments I conduct, the majority of false-positive predictions occur in the *Bare / Sparse Vegetation* and *Cropland* classes, both of which are mapped with relatively high accuracy. The *Permanent Water Bodies* class also has one of the highest classification accuracies. The validation report further notes a slight underestimation of the *Grassland* and *Moss and Lichen* classes, as well as a slight overestimation of *Tree Cover*, compared to the validation data. In addition, the WorldCover product documentation reports artifacts, including cloud-related classification errors, confusion between irrigated cropland and herbaceous wetlands, and visible block boundaries resulting from differences in the available acquisitions [14].

Figure 3-7 presents several examples of the ESA WorldCover map together with the corresponding Sentinel-2 and Sentinel-1 imagery, as well as the reference water masks from the *SIS2-Water* dataset. The Sentinel-2 and Sentinel-1 images are included as a visual reference to help with the interpretation of the scenes. The following analysis focuses on the agreement between the water surfaces in the ESA WorldCover map and the reference water masks. Overall, the two datasets show good agreement, with most large water bodies being mapped consistently. However, several differences can be observed.

Scene 23 shows Lake Powell, a reservoir located on the border between Utah and Arizona in the United States. The reference water mask shows a higher water level than the ESA WorldCover map. Differences in the extent of the reservoir can be attributed to the different acquisition dates of the Sentinel imagery used for the datasets. The Sentinel imagery used for the *SIS2-Water* dataset was acquired in August 2019, whereas

the ESA WorldCover map represents conditions in 2020. Higher water levels in 2019 can be attributed to late snowfall in the surrounding mountains, which increased the water level in the reservoir due to snowmelt [18] [19].

Scene 25 shows a forested area north of Moscow, Russia, containing wetlands, cropland, and grassland. The main differences occur for the lake and river in the southern part of the scene, where the reference water mask identifies open water, while the ESA WorldCover map classifies these areas as *Herbaceous Wetland*. Scene 47 shows a river and a bay on the Atlantic coast of Colombia, south of Barranquilla. The largest differences occur in the bay in the northern part of the scene. The ESA WorldCover map classifies these areas as *Mangroves* (light green) and *Herbaceous Wetland* (teal), whereas the reference water mask identifies a larger share of the area as open water. Wetlands are difficult to classify because they often contain open water, emergent vegetation, and moist soil. Additionally, their characteristics vary throughout the year. During my experiments I observed many false positives and false negatives around wetlands.

Scene 53 shows a lake and mountainous terrain in Heilongjiang Province, China. The ESA WorldCover map and the reference water mask agree well for the larger lakes. Similar to scene 25, the main differences occur in the mapping of areas where wetlands and open water intermingle. In addition, the river shown in the ESA WorldCover map is almost completely absent from the reference water mask. As discussed previously, this missing river is caused by an annotation error in the *S1S2-Water* reference water mask.

Limitations of using ESA WorldCover

The use of ESA WorldCover for the land cover analysis has several limitations. First, WorldCover is a land cover product rather than a reference dataset. Although its overall accuracy is relatively high, individual land cover classes still contain classification errors. As a result, some false positives in the analysis can be attributed to an incorrect land cover class because of errors in the WorldCover map rather than the segmentation model failing for this particular land cover.

Another limitation is the temporal difference between the ESA WorldCover and the *S1S2-Water* datasets. The ESA WorldCover map represents land cover conditions during 2020, whereas the *S1S2-Water* dataset is based on Sentinel imagery mostly acquired in 2019. Land cover could have changed between these acquisition dates for example due to seasonal flooding, agricultural activities, or urban development. This effect is illustrated by the example of Lake Powell shown in Figure 3-7, where the extent of the reservoir differs between the two datasets. Consequently, some disagreements between the reference water masks and the WorldCover map are caused by changes over time.

Both the *S1S2-Water* dataset and the ESA WorldCover map are derived from Sentinel observations. As a result, the land cover information is not fully independent of the input data used by the model. Areas that are generally difficult to interpret in Sentinel imagery may cause errors in both datasets. Since both products rely on the same satellite data, they also inherit similar limitations, like speckle noise and geometric distortions inherent to Sentinel-1 SAR data.

Despite these limitations, ESA WorldCover provides several important advantages for my analysis. It aligns well with the *S1S2-Water* dataset. The dataset is globally available, has the same spatial resolution of 10 m as the Sentinel imagery used in this study, and is provided on the Sentinel-2 grid.

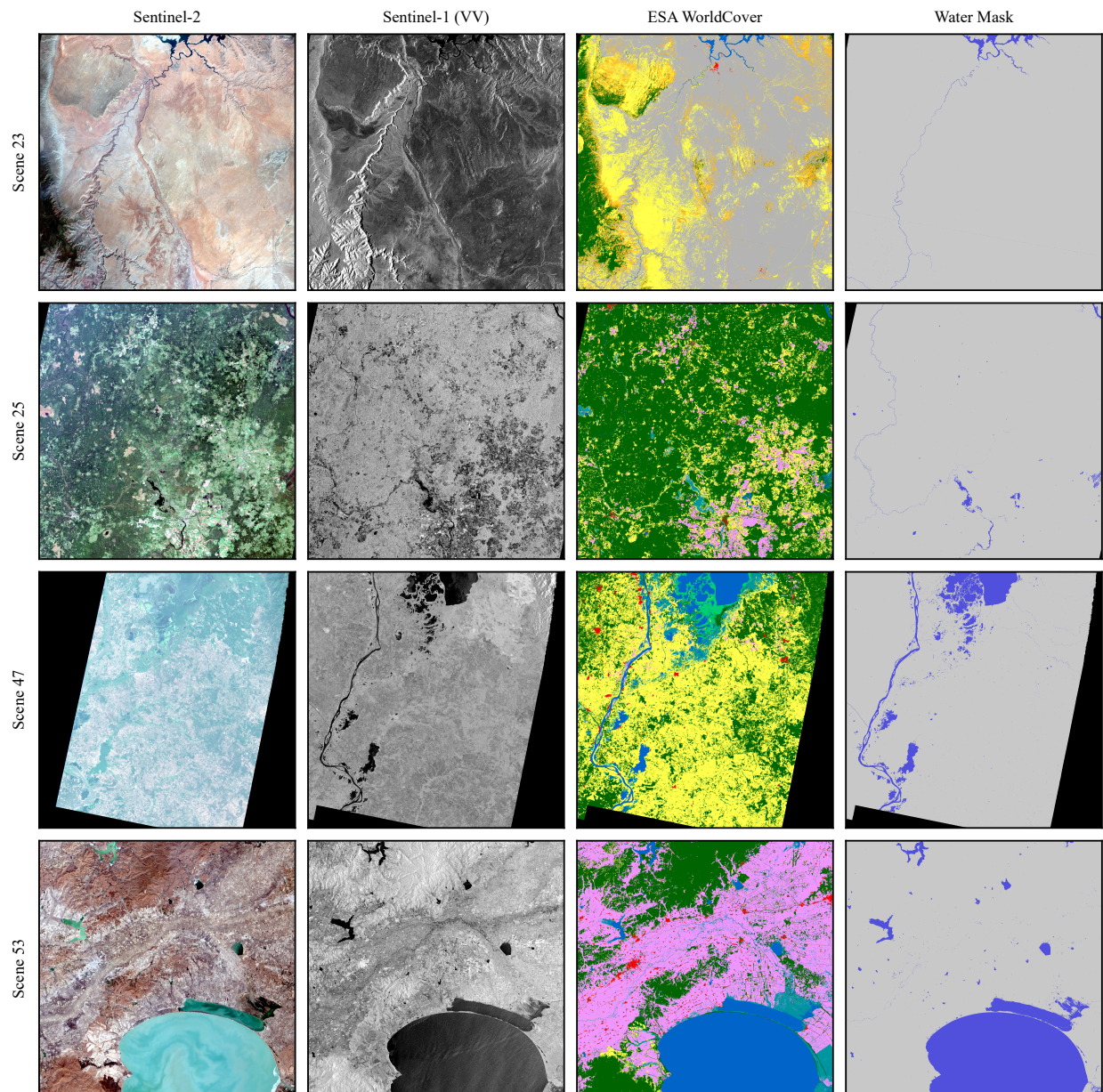


Figure 3-7: Sentinel-2 RGB, Sentinel-1 VV, ESA WorldCover and S1S2-Water reference mask. Land cover types: Bare / Sparse Vegetation (gray), Built-up (red), Cropland (pink), Grassland (Yellow), Herbaceous Wetland (teal), Mangroves (green), Moss and Lichen (beige), Permanent Water Bodies (blue), Shrubland (orange), Snow and Ice (white), Tree Cover (dark green).

3.2 Method

Figure 3-8 provides an overview of the used workflow, which consists of three main steps: data preparation, model training, and model testing. The workflow begins with the preparation of the input data, where Sentinel-1 Single Look Complex (SLC) data are processed in the Sentinel Application Platform (SNAP) to derive the polarimetric decomposition products, as described in Section 3.1.2. These products are subsequently combined with the Sentinel-1 Ground Range Detected (GRD) amplitude data to create a multi-band image stack (see Section 3.2.1). A validity mask is then calculated to identify pixels containing valid observations, after which the terrain slope is appended as an additional input layer. The prepared data are split into training, validation, and test sets. The training and validation images are further divided into non-overlapping tiles, while the test images are kept separately for later inference.

Training and evaluation of the convolutional neural network are performed in three consecutive stages: model training, inference on the test dataset, and performance evaluation. The desired input bands are selected from the prepared image stacks before the convolutional neural network is trained using the generated training and validation tiles. During training, the model learns to distinguish between water and non-water pixels using the training and validation datasets. For the inference stage, the images are divided into tiles and processed by the trained model to generate water segmentation predictions. The prediction tiles are subsequently merged to reconstruct complete segmentation masks for the original scenes. Finally, the predictions are evaluated by comparing them with the corresponding reference water masks.

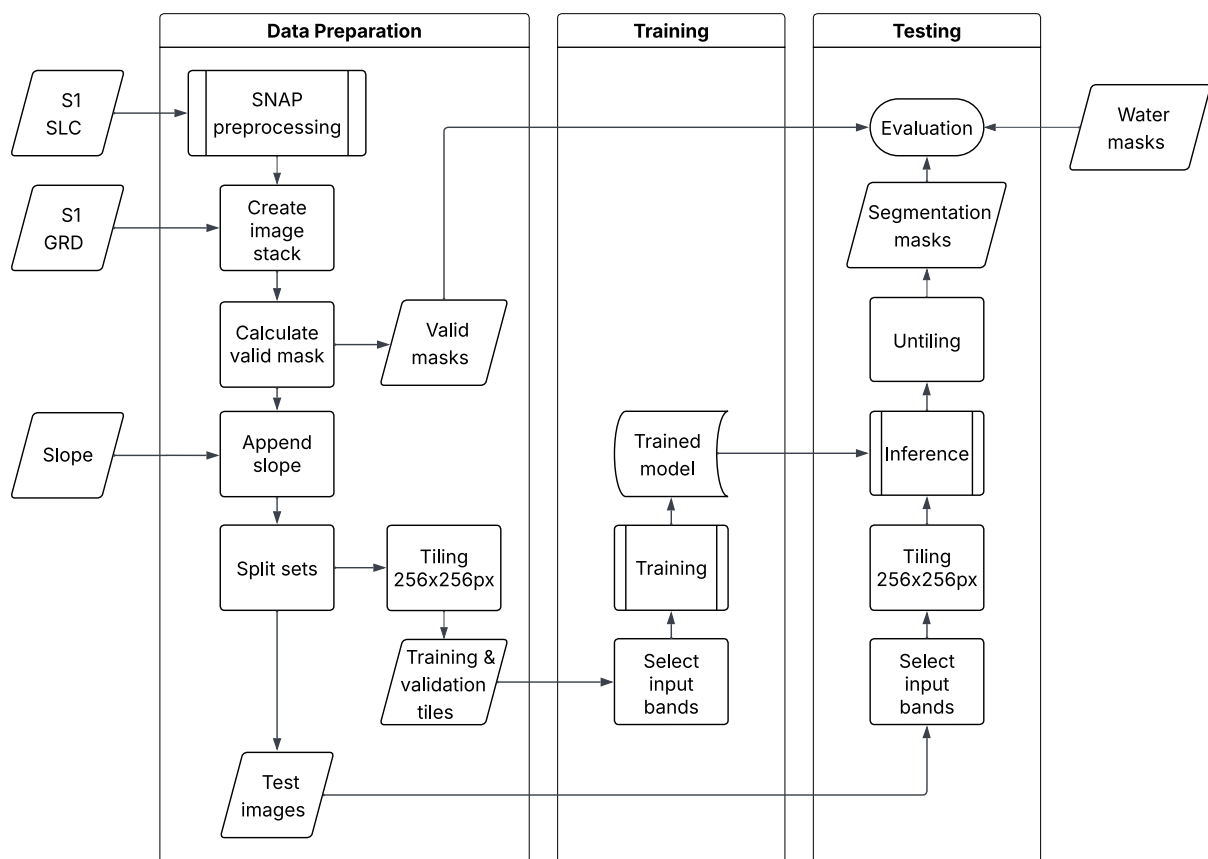


Figure 3-8: Methodology

3.2.1 Preparation of Dataset for Training and Testing

The *S1S2-Water* dataset provides a Python script for preparing the dataset for model training and evaluation [8]. This script splits the dataset into different sets for training, validation and testing. A TOML file allows the user to specify all required processing parameters for the script. The TOML file listed in Code 3-1. The first parameter `SENSOR` is for the selection of the input sensor, which can be either Sentinel-1 GRD (`s1`), Sentinel-1 polarimetric data (`s1pol`), Sentinel-2 Level-1C (`s211c`), or Sentinel-2 Level-2A (`s212a`). In addition, the user must specify the paths to the input and output directories. The input directory is expected to contain the complete dataset with all images and the corresponding STAC metadata files.

The convolutional neural network operates on smaller image tiles rather than the complete scenes. Consequently, the image stacks and reference water masks of the training and validation sets are divided into smaller tiles during dataset preparation. The tile size can be configured by the user through `TILE_SHAPE`. In this thesis, I use a tile size of 256×256 pixels as it is a standard tile size. However, the images in the test split are kept as complete scene images and masks. These scenes are divided into tiles during inference and are subsequently reassembled to their original extent after the predictions on the individual tiles have been generated.

The user can select input bands by specifying their band indices in the configuration file (`IMG_BANDS_IDX`). I include all available bands during dataset preparation and specify the different input band combinations later during the CNN configuration. If the terrain slope is not already contained in the image stack, it can be appended during this processing step (`SLOPE`). Tiles containing only *NoData* values are removed automatically. Optionally, tiles containing only some *NoData* values can also be excluded, which also removes tiles affected by cloud cover (`EXCLUDE_NODATA`).

Code 3-1: Settings for data preparation: settings.toml

```
SENSOR = "s1pol" # prepare Sentinel-1 or Sentinel-2 data ["s1", "s1pol", "s211c",  
↪ "s212a"]  
DATA_DIR = "path/to/images" # data directory that holds the original images  
OUT_DIR = "path/to/tiles" # output directory that stores the prepared train, val and  
↪ test tiles  
TILE_SHAPE = [256, 256] # desired tile shape  
IMG_BANDS_IDX = [0, 1, 2, 3, 4, 5, 6, 7, 8] # desired image band combination  
SLOPE = true # append slope band to image bands  
EXCLUDE_NODATA = false # exclude tiles that contain nodata values  
OVERWRITE = false # overwrite existing files  
  
# Sentinel-1 polarimetric image bands  
# {"vv": 0, "vh": 1, "c11": 2, "c12_real": 3, "c12_imag": 4, "c22": 5, "entropy": 6,  
↪ "anisotropy": 7, "alpha": 8}
```

The script creates separate folders for the individual splits in the output directory. The dataset preparation script uses the STAC metadata provided with the *S1S2-Water* dataset to identify the files that belong to each dataset split. Each STAC Item specifies in `['properties']['split']` whether the corresponding scene belongs to the training, validation, or test split. The script browses the STAC Catalog to create a list of all

the contained items. Iterating through this list, it retrieves the required assets from the input directory and performs the splitting. For the training and validation splits, the script generates new validity masks instead of using the masks provided with the dataset. This ensures that pixels affected by clouds and cloud shadows remain part of the training data. If required, the terrain slope is appended to the image stack before further processing. Subsequently, the scenes belonging to the training and validation splits are divided into tiles of the predefined size and stored in the output directories. In contrast, the scenes in the test split keep their original size and are copied directly to the output directory without tiling. To support the polarimetric decomposition products, I modify the dataset preparation script to be able to process Sentinel-1 polarimetric input data. This modification only required the addition of a new product type (`s1po1` and `_po1`) for the script to recognize and process the image stacks.

3.2.2 Deep Learning Model

The convolutional neural network is based on an existing deep learning framework under active development at the German Aerospace Center (DLR). The framework builds upon the network architecture presented by Wieland et al. [3] and has since been extended with additional functionality. The implementation uses *PyTorch Lightning* [20] and comprises dedicated modules for model training, inference, and evaluation. These modules are executed from the command line with the desired processing stage selected via command-line arguments. The configuration of each stage is defined by a TOML file, allowing the training, inference, and evaluation to be configured without modifying the source code.

Table 3-3: General settings for training the CNN, inference and testing

Setting	Used Value	Possible Values
DETERMINISTIC	true	boolean [true, false]
SEED	99	integer

The most relevant settings for training and testing the convolutional neural network are described in the following sections. Table 3-3 lists the general settings used to ensure reproducibility of the experiments. By setting the deterministic mode to `true` and specifying a fixed random seed, I ensure that training runs with identical settings produce consistent results.

Training

Table 3-4 provides an overview of all configuration parameters available for the CNN during training. The most relevant parameters are described in this section. The training and validation data directories must be specified before training. Each directory has to contain the subdirectories `img` and `msk`, which store the input image tiles and corresponding water mask tiles. To test the training configuration and input data, only a percentage of the available training and validation samples can be used. Only using a small subset like 10 % reduces the training time substantially and enables testing of the selected settings and a quick check of intermediate results.

The order of the input image bands can be configured using the parameter `INPUT_BANDS_IDX` with the band index starting at 0. The corresponding list of band names (`INPUT_BANDS_NAME`) must follow the same ordering to ensure that the input channels are correctly assigned during training. Optionally, the training data can be augmented to increase the effective dataset size (`AUGMENT`) with several available augmentation techniques. I do not use any data augmentation during my experiments, in order to better isolate the

Table 3-4: Settings for training the CNN

Setting	Used Value	Possible Values
TRAIN_DIRS	[".../train"]	list of strings
VAL_DIRS	[".../val"]	list of strings
OUT_DIR	""	string
TRAIN_SUBSAMPLE	1.0	float [0.0 .. 1.0]
VAL_SUBSAMPLE	1.0	float [0.0 .. 1.0]
INPUT_BANDS_IDX	<i>see Table 3-7</i>	list [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
INPUT_BANDS_NAME	<i>matching bands index</i>	list ["c11", "c12_real", "c12_imag", "c22", "entropy", "anisotropy", "alpha", "slope"]
IN_CHANNELS	<i>matching bands index</i>	integer
AUGMENT	false	boolean [true, false]
AUGMENT_TRANSFORMS	[]	["HorizontalFlip", "Contrast", "Rotate90", "Zoom", "Brightness"]
DECODER_NAME	"unet"	["unet", "unet++", "fpn", "linknet", "pspnet", "deeplabv3", "deeplabv3plus", "transformer"]
ENCODER_NAME	"efficientnet-b0"	[any of segmentation models]
ENCODER_WEIGHTS	""	["", "imagenet", "my-weights.pt"]
CLASSES	1	integer
CLASS_WEIGHTS	true	boolean [true, false]
LOSS_FN	"celovasz"	["celovasz", "focal"]
LEARNING_RATE	0.001	float or string ["auto"]
LEARNING_RATE_SCHEDULER	"CosineAnnealingWarmRestarts"	["ConstantLR", "ReduceLROnPlateau", "CosineAnnealingWarmRestarts"]
STOCHASTIC_WEIGHT_AVERAGING	true	boolean [true, false]
EARLY_STOP_PATIENCE	9	integer
MAX_EPOCH	100	integer
WEIGHT_DECAY	0.01	float
BATCH_SIZE	32	integer

effect of different input band combinations without introducing additional variability. Wieland et al. [3] reported that data augmentation does not improve the Intersection over Union (IoU) for Sentinel-1 imagery. Geometric transformations of SAR images must be applied with care, as they may introduce unrealistic image characteristics.

The training framework supports different encoder (`ENCODER_NAME`) and decoder (`DECODER_NAME`) architectures. I use a U-Net architecture with an EfficientNet-B0 encoder, as according to previous benchmarking tests [3], this combination provides the best compromise between segmentation performance and computational efficiency. The model supports pretrained encoder weights like *ImageNet*. Alternatively, the path to other pretrained weights (*myweights.pt*) can be specified. For my experiments, I use randomly initialized weights. Pretrained weights are typically obtained from optical image datasets and are therefore optimized for RGB imagery. My input data consist of Sentinel-1 SAR amplitude data and polarimetric decomposition products making *ImageNet* or similar weights unsuitable.

The number of output classes excluding the background class is specified with the `CLASSES` parameter. Since I work with binary water segmentation, I specify only a single foreground class (water). Owing to the strong class imbalance between water and background pixels, I use class weights (`CLASS_WEIGHTS`) in the loss function. A standard cross-entropy loss would make the model biased towards predicting the dominant class (land), leading to poor performance on the less frequent class (water). To account for this, the loss function assigns higher weights to underrepresented classes and lower weights to dominant ones. These weights are computed automatically from the dataset before training begins. For the *SIS2-Water* dataset, the class weight is calculated as 7.797.

The user can choose from several loss functions (`LOSS_FN`). I employ a combined cross-entropy and Lovász loss ("`ceLovasz`") with equal weighting, as this combination achieved the best results in previous studies [21]. The initial learning rate can either be specified manually or determined automatically using a learning rate finder (`LEARNING_RATE`). During training, the learning rate is controlled by the "`CosineAnnealingWarmRestarts`" scheduler [22], which periodically decreases the learning rate following a cosine function before resetting it to a higher value. Periodic warm restarts help the optimizer escape local minima and have been shown to improve the generalization performance of deep neural networks compared to fixed or stepwise learning rate schedules [22]. Stochastic Weight Averaging (SWA) can be enabled by setting `STOCHASTIC_WEIGHT_AVERAGING` to `true`. In this case, the cosine annealing scheduler is replaced during the final 20% of the configured `MAX_EPOCHS`, allowing the final model parameters to be averaged over multiple optimization steps to improve generalization [23]. I use both a `LEARNING_RATE_SCHEDULER` and `STOCHASTIC_WEIGHT_AVERAGING` as they were found to produce more reliable results during previous benchmarking tests.

Inference

During inference, I generate water segmentation predictions by applying the trained model to the images in the test split. The model outputs either binary water masks or probability maps containing the predicted likelihood of each pixel belonging to the water class. Table 3-5 gives an overview of all configuration parameters available. The paths to the test data and the trained model file (`model.onnx`) must be specified. The input band indices and corresponding band names must match the ones used during training to ensure that the input channels are interpreted correctly by the model.

If required, the terrain slope can be computed during inference by setting `GET_SLOPE` to `true`. This option is useful when the trained model expects a slope band as input, but the image stack does not contain one. In this work, however, the slope layer is appended during dataset splitting and is therefore already included in the input data.

The `OVERLAP` parameter defines the percentage of overlap between adjacent image tiles. The tiles overlap

Table 3-5: Settings for inference

Setting	Used Value	Possible Values
IMG_DIRS	[".../test/img"]	list of strings
OUT_DIR	".../test/pred"	string
MODEL_FILE	".../model.onnx"	string
INPUT_BANDS_IDX	<i>matching bands used for training</i>	list [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
INPUT_BANDS_NAME	<i>matching bands used for training</i>	list ["c11", "c12_real", "c12_imag", "c22", "entropy", "anisotropy", "alpha", "slope"]
GET_SLOPE	false	boolean [true, false]
OVERLAP	0.1	float [0..1]
SAVE_PROBAS	true	boolean [true, false]

with a tapered cosine window function. Overlapping tiles reduces artifacts at the edges of the tiles and results in smoother blending of the prediction results. Instead of directly creating binary prediction masks, predicted class probabilities are saved when `SAVE_PROBAS` is `true`. The final binary water masks are generated in a separate Python script by applying a threshold of 0.5 to the probability maps. Pixels with predicted probabilities greater than 0.5 are classified as water, otherwise, they are classified as background class. Figure 3-9 presents an example of the prediction probabilities and resulting water mask for one scene. As can be seen, the bare agricultural fields in the north-eastern part of the scene have relatively high probabilities and are therefore classified as water.

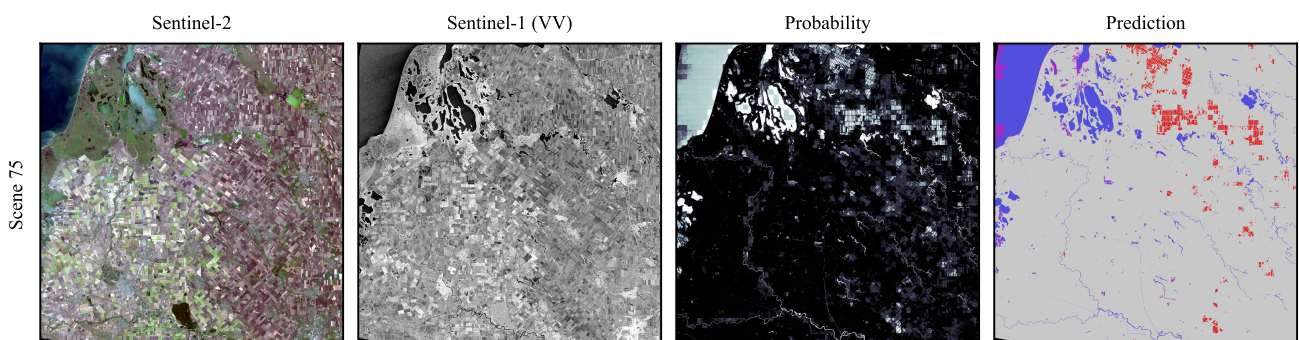


Figure 3-9: Sentinel-2 RGB, Sentinel-1 VV, probability map and prediction results for VV experiment. Lighter values: higher probability. Blue: true positives (water), gray: true negatives (non-water), purple: false negatives (missed water), red: false positives (falsely detected water)

Evaluation

During evaluation, the predicted water masks generated during inference are compared with the corresponding reference water masks from the *SIS2-Water* dataset. Table 3-6 gives an overview of all configuration parameters available for the evaluation. The user must specify the directories containing the test images (IMG_DIRS), the reference water masks (MSK_TRUE_DIRS), the validity masks (MSK_VALID_DIRS), and the predictions generated during inference (MSK_PRED_DIR). In addition, the number of foreground classes must be defined. The PROBAS parameter specifies whether the predicted masks are available as probabilities or binary predictions. When probability maps are available, additional evaluation metrics such as the precision-recall curve can be computed. The IGNORE_BACKGROUND parameter allows the background class to be excluded when calculating the aggregated performance metrics. Otherwise, the evaluation metrics is dominated by the background class which makes the evaluation of the model's performance on the water class less representative. Excluding the background class is especially advantageous for datasets where background pixels far outnumber water pixels, like in the *SIS2-Water* dataset.

Table 3-6: Settings for testing

Setting	Used Value	Possible Values
IMG_DIRS	[".../test/img"]	list of strings
MSK_TRUE_DIRS	[".../test/msk"]	list of strings
MSK_VALID_DIRS	[".../test/valid"]	list of strings
MSK_PRED_DIR	".../test/pred"	string
OUT_DIR	".../test/results"	string
CLASSES	1	integer
PROBAS	true	boolean [true, false]
IGNORE_BACKGROUND	true	boolean [true, false]

The testing module calculates following evaluation metrics: Intersection over Union (IoU), F1 Score, Precision, and Recall. It calculates the aggregated metrics for the complete set of images and individual metrics for each scene. In addition, the testing module calculates and plots the confusion matrix as well as the binary precision-recall curve, and determines the optimal threshold. As an example, Figure 3-10 presents the confusion matrix and Figure 3-11 the precision-recall curve for the baseline experiment VV experiment. To make the experiments comparable, I did not use the optimal threshold calculated for each experiment during evaluation but instead used the standard threshold of 0.5 for all experiments. For the VV experiment, Figure 3-12 presents the binary water masks for the used threshold (0.5) and the optimal threshold (0.6) for four different test scenes. The water masks obtained with the different thresholds do not show significant differences. There are small differences along the edges of falsely classified regions, where the optimal threshold slightly reduces the number of falsely classified pixels. There are, however, no major reductions in false positives (or false negatives) when using the optimal threshold. This is due to the probabilities in these regions being considerably higher (or lower) than the optimal threshold.

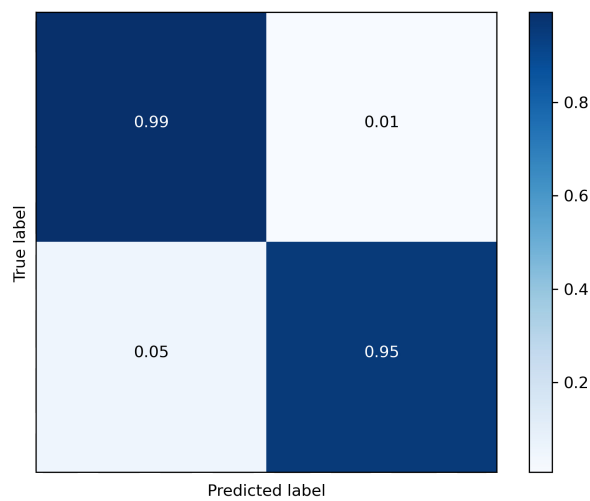


Figure 3-10: Confusion matrix for the VV experiment

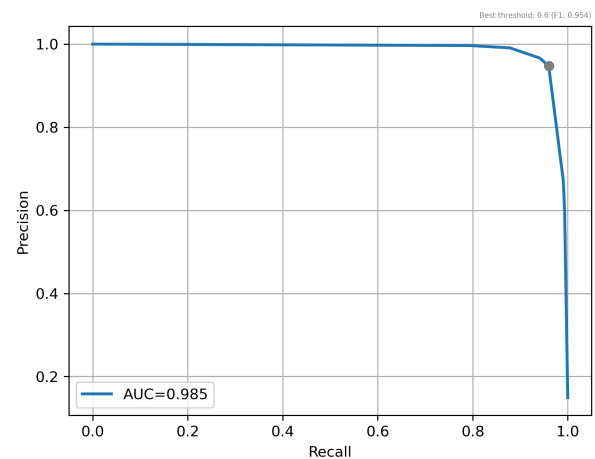


Figure 3-11: Binary precision-recall curve (blue) for the VV experiment

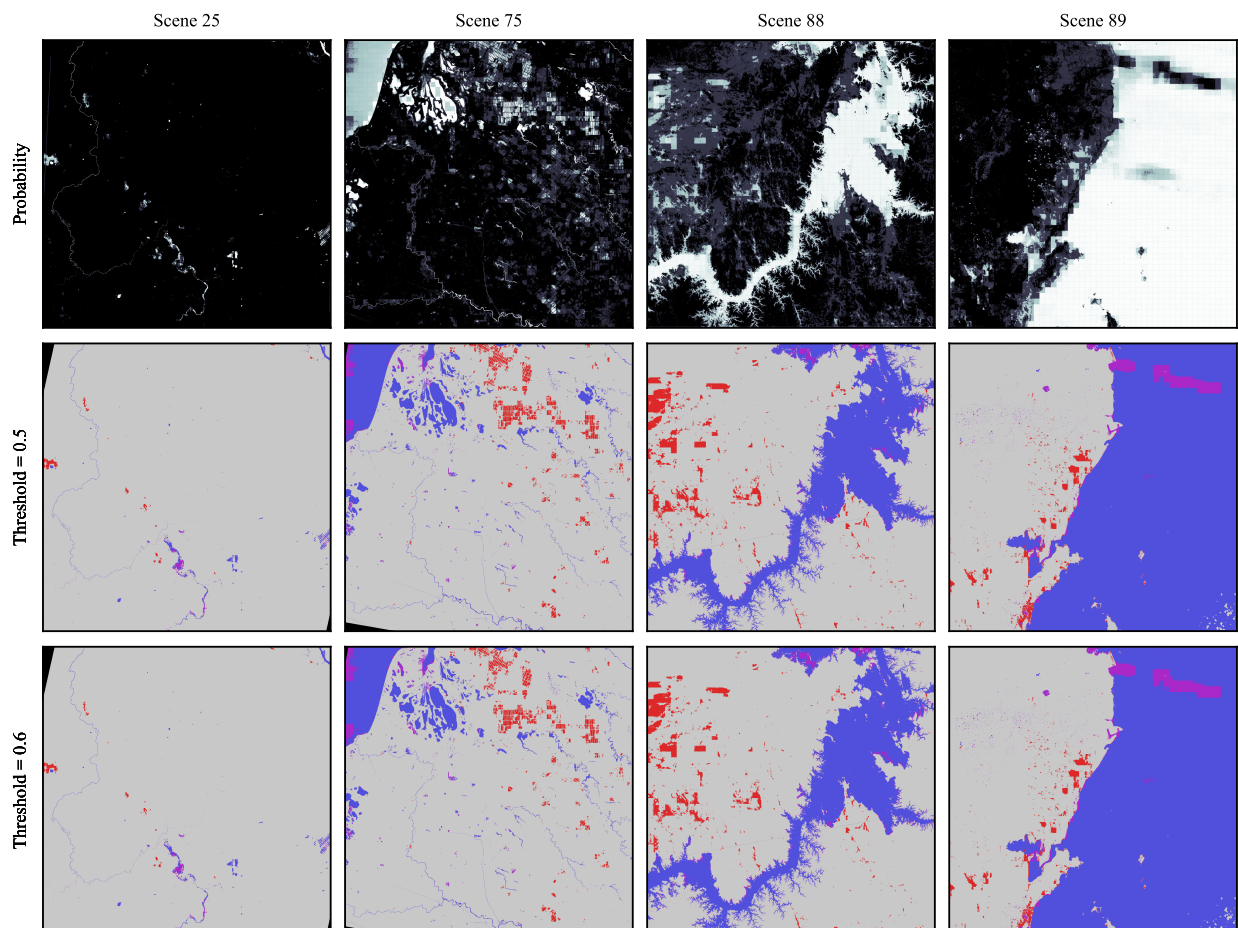


Figure 3-12: Comparison of the used threshold (0.5) and the optimal threshold (0.6) for the VV experiment. Blue: true positives (water), gray: true negatives (non-water), purple: false negatives (missed water), red: false positives (falsely detected water)

3.2.3 Experiments

To investigate the influence of polarimetric decomposition products on water segmentation performance, I conduct a series of experiments using different input band combinations. An overview of all experiments is provided in Table 3-7. The first experiment uses only the Sentinel-1 VV and VH amplitude bands to establish a baseline. This baseline also enables a comparison with the results reported by Wieland et al. [3]. Building upon the baseline experiment, I evaluate the contribution of the covariance matrix (C_2) features, the H - A - α decomposition parameters, and different combinations of these polarimetric features. In all experiments, terrain slope derived from the Copernicus Digital Elevation Model (DEM) is included as an additional input feature. Wieland et al. [3] showed that incorporating slope improves segmentation results by reducing the number of false negative predictions.

In this thesis, I evaluate only the complete sets of polarimetric decomposition parameters or covariance matrix elements. Consequently, the individual contribution the single parameters to the segmentation performance remains unclear. A more systematic analysis could investigate the influence of individual polarimetric features to identify those that provide the greatest benefit for water segmentation. Using all available polarimetric features may increase the risk of overfitting and result in the model failing to generalize to additional unseen data. Including more input features also increases the complexity of the model. Identifying the most useful polarimetric parameters could therefore also reduce model complexity. In particular, the simultaneous inclusion of both the entropy (H) and anisotropy (A) parameters may introduce redundant information. For dual-polarimetric SAR data, the H - A - α decomposition is derived from a reduced scattering matrix. This results in the parameters H and A using the same eigenvalues and therefore containing the same information. Investigating the effect of using only one of these parameters with or without the angle α , could help decrease the risk of overfitting.

Table 3-7: Bands used for Experiments

ID	VV	VH	C_{11}	$\text{Re}(C_{12})$	$\text{Im}(C_{12})$	C_{22}	H	A	α	Slope
VV	×	×								×
C2			×	×	×	×				×
HA							×	×	×	×
HAC2			×	×	×	×	×	×	×	×
VVC2	×	×	×	×	×	×				×
VVHA	×	×					×	×	×	×
VVC2HA	×	×	×	×	×	×	×	×	×	×

3.3 Results

This chapter presents a few additional results that I did not include in the manuscript. In Section 3.3.1, I compare my results to the results from Wieland et al. [3] and explain some observed differences. In Section 3.3.2, I present results for all conducted experiments, including the experiments that I omitted in the manuscript.

3.3.1 Comparison with Results from Wieland et al.

As mentioned previously, there is no corresponding Sentinel-1 SLC data for scene 31 at the required date and the scene was removed from the testing split. In addition, the convolutional neural network architecture used in this thesis was further developed after the publication of Wieland et al. [3]. The encoder, decoder, loss function, optimizer, initial learning rate, and weight decay remain the same, but some more advanced training strategies, such as warm restarts, were added. These two factors make a direct comparison between the results difficult. I did not test the more recent architecture using the original testing dataset including scene 31. This could have helped to determine the impact of the updated model architecture on the results.

Furthermore, the model used in this thesis produced artifacts in the segmentation, where the boundaries of the tiles were sometimes visible in the predicted water masks. Figure 3-13 shows two scenes where these artifacts are clearly visible as straight boundaries between false positive (or false negative) predictions and the background. In scene 75, a straight vertical boundary between the false positives and true negatives can be seen in the agricultural fields. The false negatives in the water body in the north-western corner of the scene also show very square-shaped artifacts. In scene 88, the artifacts are visible in the false positive predictions and, to a lesser extent, in the false negative predictions. During my time at DLR, I was not able to fully eliminate these artifacts.

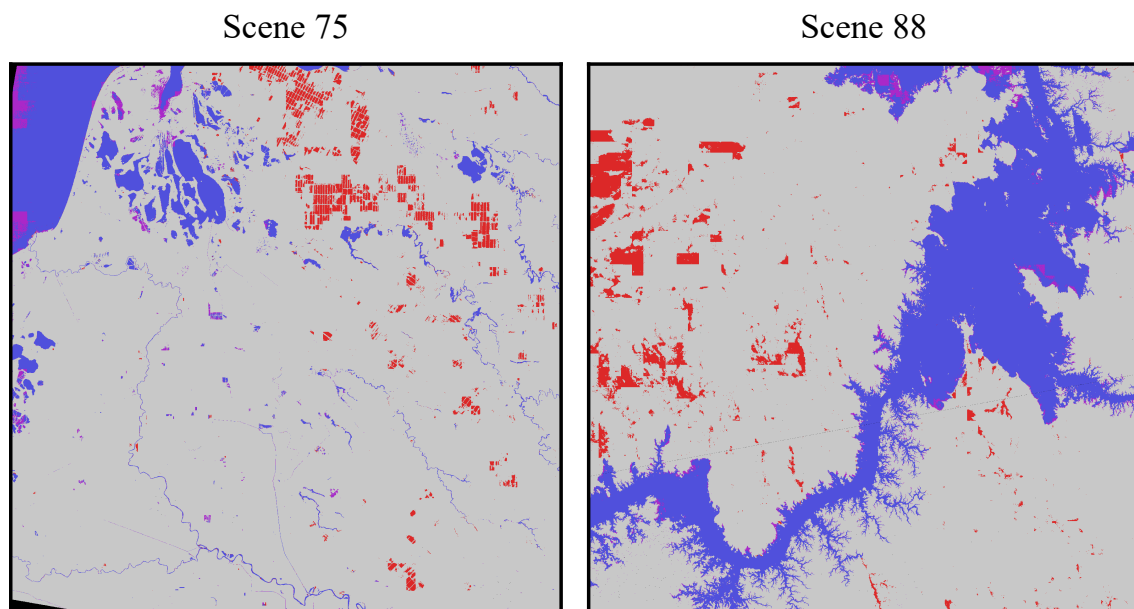


Figure 3-13: Examples of scenes containing artifacts in the segmentation results.

Despite these limitations, general trends in the results can still be compared. Table 3-8 lists the Intersection over Union (IoU), precision, and recall for both the *VV* experiment and the Baseline Model 2 (BM2) from

Wieland et al. [3]. The VV experiment performed better in all three metrics. However, based on the metrics alone, it is difficult to determine whether this improvement is caused by the missing scene or by the changes to the model architecture.

Figure 3-14 presents the IoU for the individual test scenes. As can be seen, the IoU is similar for both experiments for most scenes. Scenes with high IoU values in their study generally also perform well in the present study, while the same challenging scenes achieve lower IoU values. For most, but not all scenes, my model performs equally well or slightly better. For a third of the scenes, the IoU is worse for my model. This shows that a more complex CNN architecture does not necessarily result in better performance. A similar conclusion was reached by Bereczky et al. [21], who found in their tests of CNN architectures with different levels of complexity that a more sophisticated architecture does not necessarily result in better segmentation performance.

The missing scene 31 has a relatively low IoU in the experiment of Wieland et al. [3]. As a result, including this scene in the aggregation could reduce the overall IoU compared to my VV experiment, where it is not included.

Table 3-8: Intersection over Union (IoU), Precision and Recall comparison between my VV experiment and Wieland et al. [3] Baseline Model 2 (BM2)

	IoU	Precision	Recall
Wieland et al.	0.844	0.950	0.822
Patzwahl et al.	0.911	0.956	0.951

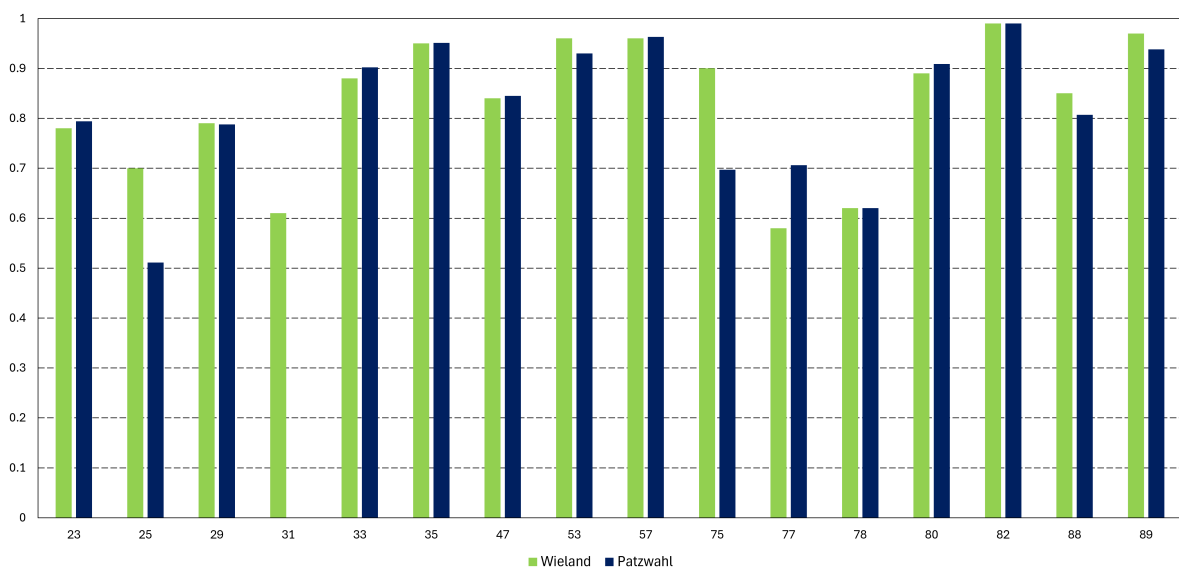


Figure 3-14: Intersection over Union (IoU) comparison between my VV experiment and Wieland et al. [3] Baseline Model 2 (BM2).

3.3.2 Additional Results

Due to their significantly worse performance, the experiments without the backscatter intensity data (VV-VH) are not further analyzed in the manuscript. This section presents the results of all conducted experiments.

Figure 3-15 shows the Intersection over Union (IoU), F1-score, Precision, and Recall for the different experiments across all test scenes. These graphs are included here for reference, as the results are discussed in more detail in the manuscript.

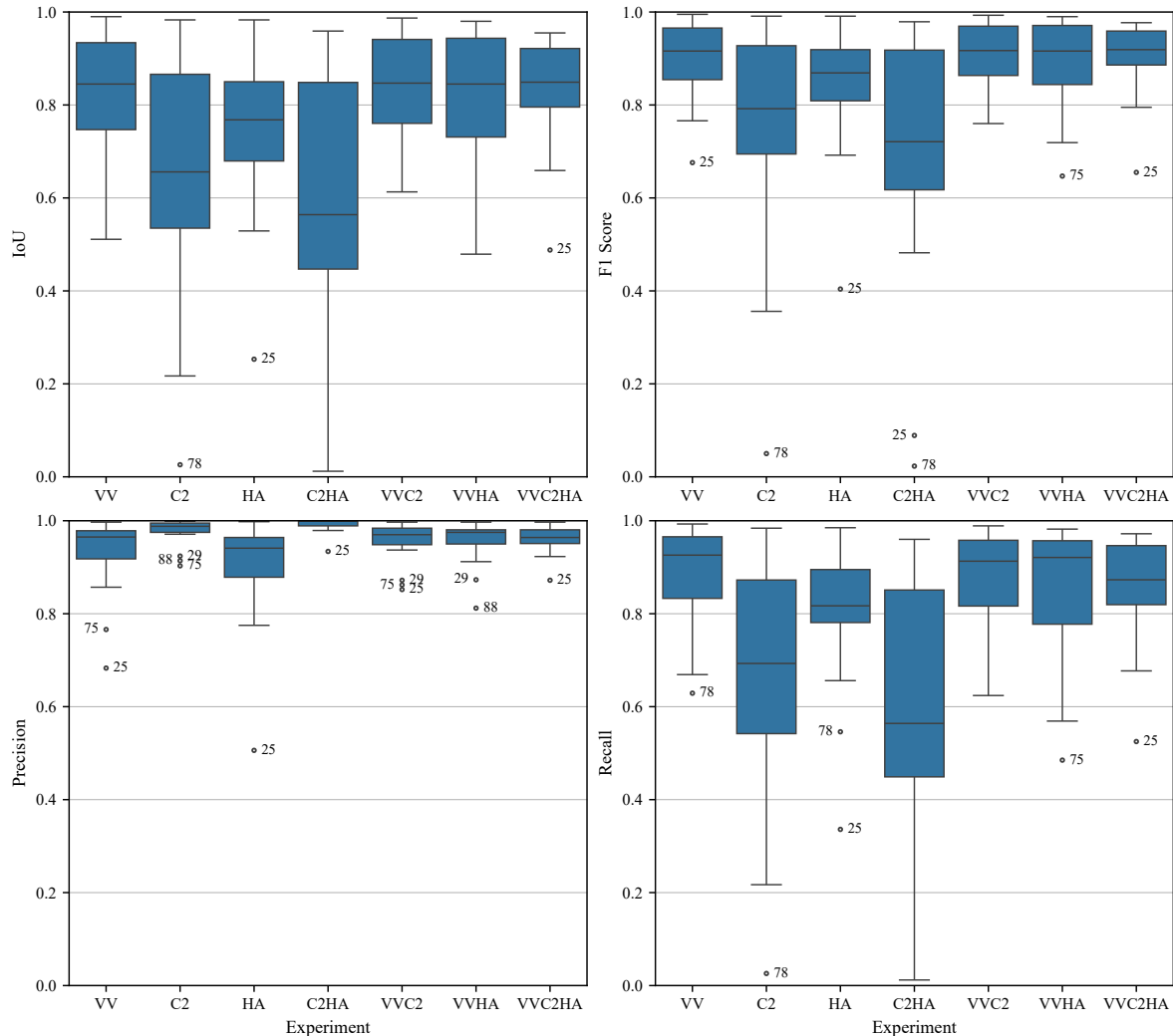


Figure 3-15: Intersection over Union (IoU), F1-score, Precision, and Recall for different experiments across all test scenes. Horizontal lines show medians, boxes represent the interquartile range, and whiskers represent 1.5 times the interquartile range. Outliers are labelled with the corresponding scene ID.

Figure 3-16 presents the Intersection over Union (IoU), Precision, and Recall for each test scene and all conducted experiments. At first glance, it can be seen that the Precision is generally high across the different scenes and experiments. Scene 25, 29, 75, 77, and 88 show slightly lower Precision for some experiments, particularly for VV and HA. Scene 25 and 78 show the lowest values, particularly for C2 and C2HA. The Recall values are overall lower and show more variability compared to the Precision. As a result, the IoU is mostly influenced by the Recall rather than the Precision. Additionally, some scenes show relatively worse performance across all experiments. While the Precision is generally high for experiments that do not include VV-VH, the Recall is lower.

The impact of including polarimetric data is shown in Figure 3-17. It shows the change in the different metrics for each experiment compared to the baseline experiment VV. For most scenes, the inclusion of polarimetric data has little to no effect. However, a drastic decrease in Recall can clearly be seen for all experiments

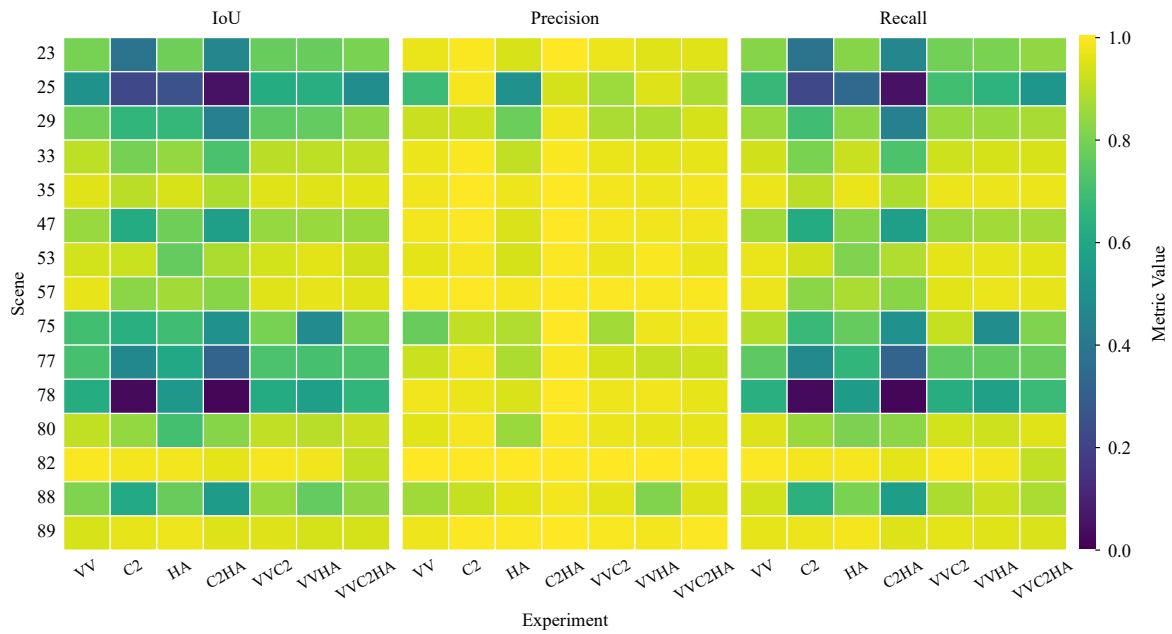


Figure 3-16: Intersection over Union (IoU), Precision and Recall across scenes for all experiments

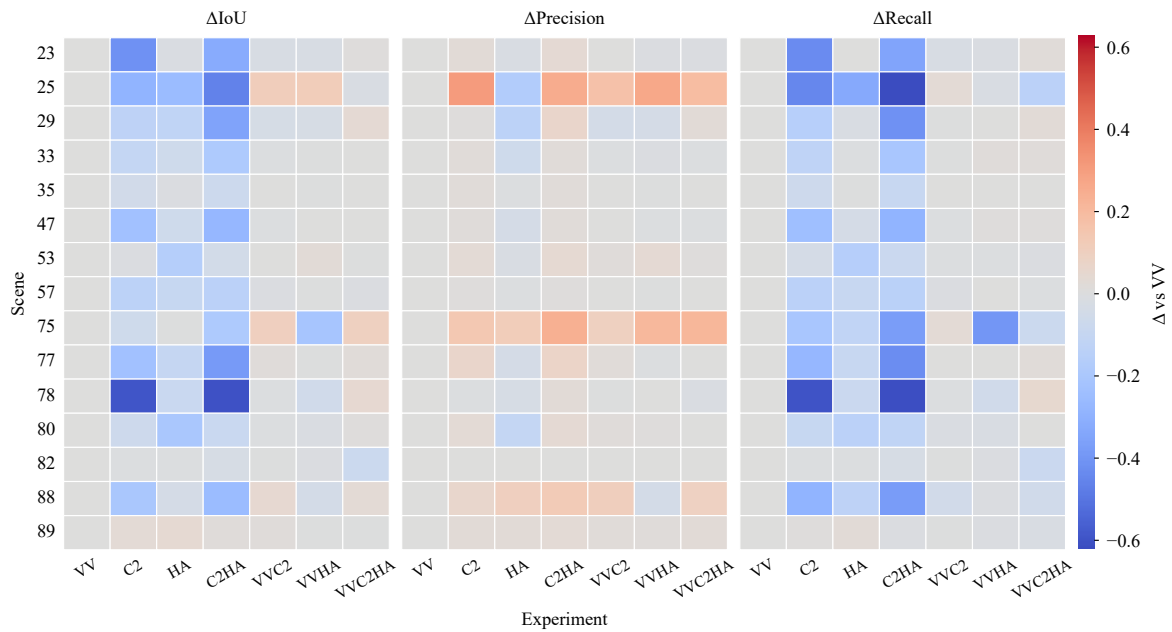


Figure 3-17: Change in Intersection over Union (IoU), Precision and Recall relative to baseline experiment (VV). Positive values (red): increase in FP or FN compared to the baseline; negative values (blue): reduction in FP.

without VV-VH (blue squares). For some scenes, an improvement in Precision can be observed (orange squares). These include scene 35, 75, and 88 which I discuss in more detail in the manuscript.

Figure 3-18 presents the false positives FP (%) across different land cover types and the false negatives FN (%) for all experiments, as well as the change relative to the baseline experiment (VV). Even though the experiments that only include polarimetric information reduce the number of false positives, they also significantly increase the number of false negatives. Only using C_2 features or using the combination $HAC2$, reduces false positives in bare landscapes, cropland and herbaceous wetland. Only using $H-A-\alpha$ introduces more false positives for some land cover types such as wetlands and mangroves but reduces the number of false positives for bare landscapes. The $HAC2$ input combination has the highest number of false negatives of all investigated combinations. This may be due to the redundant information introduced by including both the covariance matrix and the polarimetric decomposition products. This could also explain why the combination of all input features $VVHAC2$ does not perform better than just $VVC2$. Unlike all other combinations, $VVHA$ does not reduce the number of false positives in *Bare / Sparse Vegetation*. Investigating the reason for this difference could also be a topic for future research.

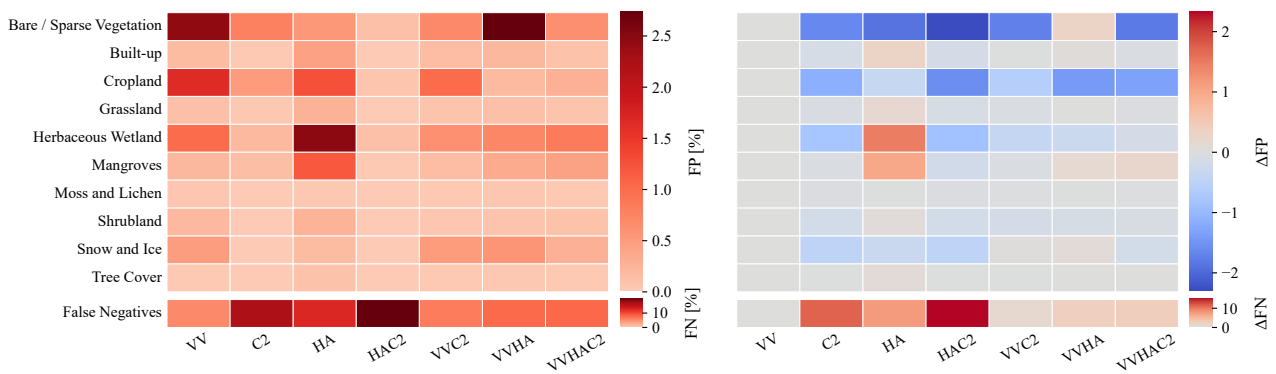


Figure 3-18: Left: false positives FP (%) across land cover types and false negatives FN (%) for each experiment. Right: change relative to baseline experiment (VV). Positive values (red): increase in FP or FN compared to the baseline; negative values (blue): reduction in FP.

All combinations reduce the number of false positives for *Cropland* compared to the baseline. Looking closer at the distribution of false positives, the cropland areas with the most false positives are bare at the time of Sentinel image acquisition. There are two scenes in the dataset that are dominated by Cropland: scene 53 and 75. Both scenes are shown in Figure 3-19. Scene 53 shows a lake and agricultural land in Heilongjiang Province, China. The Sentinel imagery for this scene was acquired in early spring, when crops had not yet begun to grow. Scene 75 shows an area at the shore of the Sea of Azov, northeast of Krasnodar, Russia. The Sentinel imagery for this scene was acquired in September, when a majority of crops were already harvested. Both areas are major rice production sites. In the scene in China, the rice was not yet growing, while in the scene in Russia, the rice had already been harvested. As a semiaquatic plant, rice requires constant irrigation, which is why rice fields are often flooded. Even though the fields in both scenes do not seem to be flooded at the time, the fields may still have small pools of standing water that contribute to a false-positive detection.

Herbaceous Wetland is the third land cover type that seems to benefit from the inclusion of polarimetric data. However, as I discuss in the manuscript and Section 3.1, the results for Wetland should be interpreted with care since both the reference masks and the land cover map contain inconsistencies in wetland areas. Figure 3-19 shows one example of an area with extensive herbaceous wetland. A dedicated study could further investigate the potential of polarimetric information for water body segmentation in wetland environments. For this purpose, the reference water masks would need to be revised to create more reliable and consistent masks for training and evaluation.

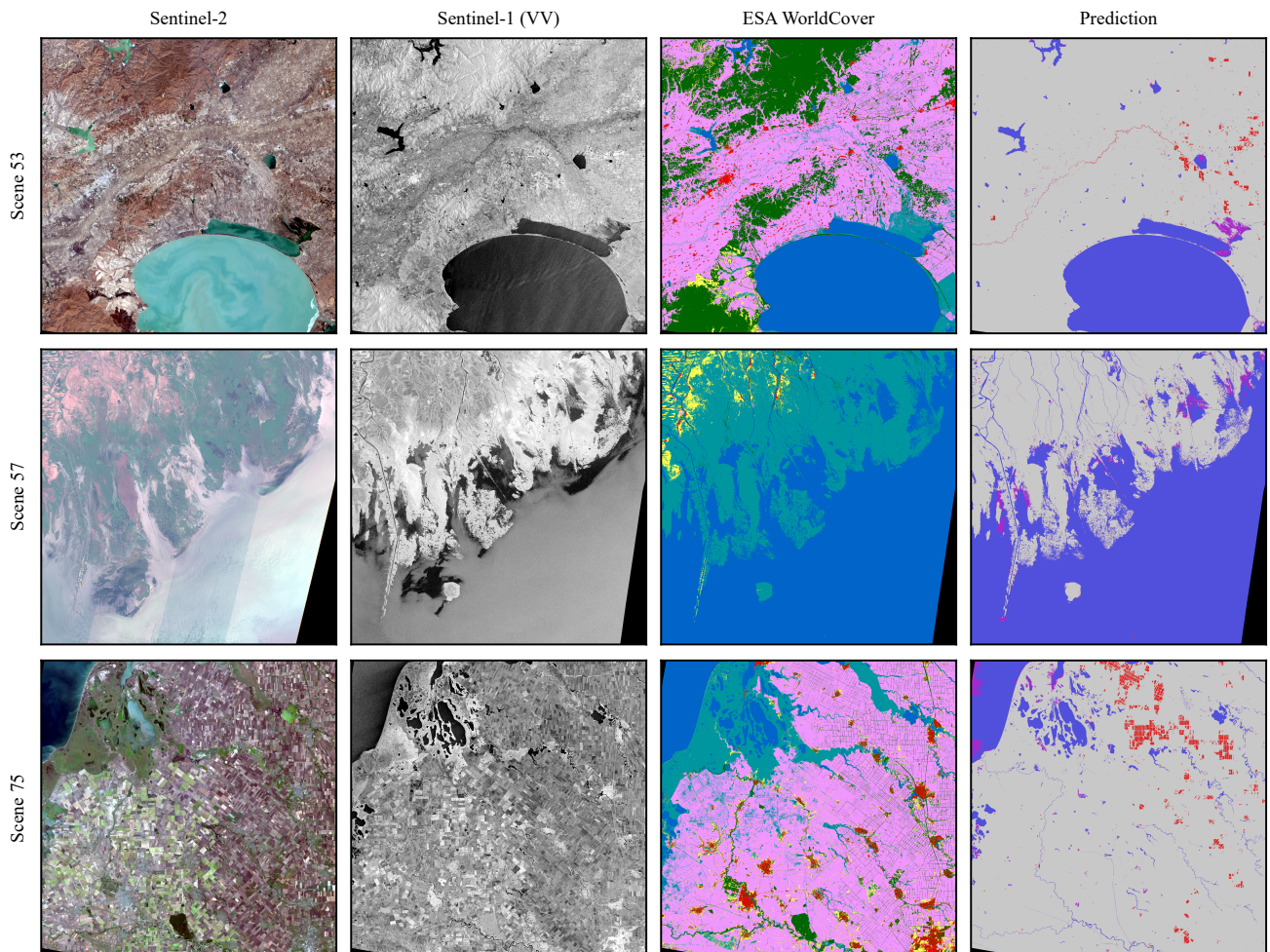


Figure 3-19: Examples of scenes with bare agricultural fields and wetlands. Land cover types: Bare / Sparse Vegetation (gray), Built-up (red), Cropland (pink), Grassland (Yellow), Herbaceous Wetland (teal), Mangroves (green), Moss and Lichen (beige), Permanent Water Bodies (blue), Shrubland (orange), Snow and Ice (white), Tree Cover (dark green). Blue: true positives (water), gray: true negatives (non-water), red: false positives (wrongly detected water), purple: false negatives (missed water).

3.4 Bibliography

- [1] 'Journal of Selected Topics in Applied Earth Observations and Remote Sensing,' GRSS-IEEE. (3rd Jan. 2020), [Online]. Available: <https://www.grss-ieee.org/publications/journal-of-selected-topics-in-applied-earth-observations-and-remote-sensing/> (visited on 15/07/2026).
- [2] 'Information for Authors,' *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 13, pp. C3–C3, 2020. DOI: 10.1109/JSTARS.2020.2984062.
- [3] M. Wieland, F. Fichtner, S. Martinis *et al.*, 'S1S2-Water: A Global Dataset for Semantic Segmentation of Water Bodies From Sentinel- 1 and Sentinel-2 Satellite Images,' *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 17, pp. 1084–1099, 2024. DOI: 10.1109/JSTARS.2023.3333969.
- [4] C. D. S. Ecosystem. 'Copernicus DEM - Global and European Digital Elevation Model | Copernicus Data Space Ecosystem.' (), [Online]. Available: <https://dataspace.copernicus.eu/explore-data/data-collections/copernicus-contributing-missions/collections-description/COP-DEM> (visited on 12/07/2026).
- [5] M. Wieland, *S1S2-Water GitHub Repository*, 31st Jul. 2024.
- [6] 'STAC: SpatioTemporal Asset Catalogs.' (), [Online]. Available: <https://stacspect.org/en/> (visited on 14/06/2024).
- [7] 'PySTAC Documentation — pystac 1.15.1 documentation.' (), [Online]. Available: <https://pystac.readthedocs.io/en/stable/> (visited on 21/07/2026).
- [8] M. Wieland, F. Fichtner, S. Martinis *et al.*, *S1S2-Water: A global dataset for semantic segmentation of water bodies from Sentinel-1 and Sentinel-2 satellite images*, version v1.0.1, Zenodo, 24th May 2024. DOI: 10.5281/zenodo.8314174.
- [9] 'Copernicus Data Space Ecosystem APIs.' (), [Online]. Available: <https://dataspace.copernicus.eu/analyse/apis> (visited on 19/06/2024).
- [10] 'STEP – Science Toolbox Exploitation Platform.' (), [Online]. Available: <https://step.esa.int/main/> (visited on 14/06/2024).
- [11] M. Miltiadou, *MiltoMiltiadou/SNAPGraphMassiveProcessing*, 25th Dec. 2025.
- [12] 'Cloud Optimized GeoTIFF.' (), [Online]. Available: <https://www.cogeo.org/> (visited on 14/06/2024).
- [13] D. Zanaga, R. Van De Kerchove, W. De Keersmaecker *et al.*, *ESA WorldCover 10m 2020 v100*, version v100, 2021. DOI: 10.5281/zenodo.5571936.
- [14] R. Van De Kerchove, D. Zanaga, L. Li, N. Tsendbazar and M. Lesiv, *ESA WorldCover Product User Manual v1.0*, 19th Oct. 2021.
- [15] F. a. A. O. of the United Nations (FAO). 'Land Cover Classification System (LCCS) | Land & Water | Food and Agriculture Organization of the United Nations | Land & Water | Food and Agriculture Organization of the United Nations.' (2026), [Online]. Available: <https://www.fao.org/land-water/land/land-governance/land-resources-planning-toolbox/category/details/en/c/1036361/> (visited on 27/03/2026).
- [16] E. W. Project. 'WorldCover Viewer,' *ESA WorldCover Viewer*. (2021), [Online]. Available: <https://viewer.esa-worldcover.org/worldcover> (visited on 31/03/2026).
- [17] N. Tsendbazar, L. Li, M. Koopman *et al.*, *ESA WorldCover Product Validation Report v1.1*, 22nd Oct. 2021.
- [18] 'A Pulse of Water for Lake Powell - NASA Science.' (1st Oct. 2019), [Online]. Available: <https://science.nasa.gov/earth/earth-observatory/a-pulse-of-water-for-lake-powell-145667/> (visited on 15/04/2026).

- [19] 'Usbr.gov/uc/water/hydrodata/reservoir_data/919/dashboard.html#pool_elevation/' (), [Online]. Available: https://www.usbr.gov/uc/water/hydrodata/reservoir_data/919/dashboard.html#pool_elevation/ (visited on 15/04/2026).
- [20] 'PyTorch Lightning 2.6.1 Documentation.' (), [Online]. Available: <https://lightning.ai/docs/pytorch/stable/> (visited on 06/08/2026).
- [21] M. Bereczky, M. Wieland, C. Krullikowski, S. Martinis and S. Plank, 'Sentinel-1-Based Water and Flood Mapping: Benchmarking Convolutional Neural Networks Against an Operational Rule-Based Processing Chain,' *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 15, pp. 2023–2036, 2022. DOI: 10.1109/JSTARS.2022.3152127.
- [22] I. Loshchilov and F. Hutter. 'SGDR: Stochastic Gradient Descent with Warm Restarts.' arXiv: 1608.03983 [cs.LG]. (3rd May 2017), [Online]. Available: <http://arxiv.org/abs/1608.03983> (visited on 30/07/2026), pre-published.
- [23] P. Izmailov, D. Podoprikin, T. Garipov, D. Vetrov and A. G. Wilson. 'Averaging Weights Leads to Wider Optima and Better Generalization.' arXiv: 1803.05407 [cs.LG]. (25th Feb. 2019), [Online]. Available: <http://arxiv.org/abs/1803.05407> (visited on 30/07/2026), pre-published.

4 Appendix

4.1 STAC Metadata

Code 4-1: STAC metadata catalog: catalog.json (abridged)

```
{ "id": "sis2_water_pol",
  "type": "Catalog",
  "title": "sis2_water_pol",
  "stac_version": "1.0.0",
  "description": "S1S2-Water-Pol: A global dataset for semantic segmentation of water
  ⇨ bodies from Sentinel-1 and Sentinel-2 satellite images including
  ⇨ polarimetrically decomposed Sentinel-1 data (covariance matrix and
  ⇨ dual-polarimetric decomposition parameters H, A, alpha",
  "links": [
    {
      "rel": "root",
      "href": "./catalog.json",
      "type": "application/json",
      "title": "catalog"
    },
    {
      "rel": "self",
      "href": "./catalog.json",
      "type": "application/json"
    },
    {
      "rel": "item",
      "href": "./93/sentinel12_93_meta.json",
      "type": "application/json",
      "title": "metadata"
    },
    ... items 91 to 5 ...
    {
      "rel": "item",
      "href": "./1/sentinel12_1_meta.json",
      "type": "application/json",
      "title": "metadata"
    }
  ]
}
```

```
}
```

Code 4-2: STAC metadata item after changes: sentinel12_75_meta.json

```
{
  "type": "Feature",
  "stac_version": "1.0.0",
  "id": "sentinel12_75_meta",
  "links": [
    {
      "rel": "root",
      "href": "../catalog.json",
      "type": "application/json"
    },
    {
      "rel": "parent",
      "href": "../catalog.json",
      "type": "application/json"
    },
    {
      "rel": "self",
      "href": "sentinel12_75_meta.json",
      "type": "application/geo+json"
    }
  ],
  "assets": {
    "s1_img_pol": {
      "href": "75/sentinel12_s1_75_img_pol.tif",
      "type": "image/tiff; application=geotiff; profile=cloud-optimized",
      "roles": [
        "data"
      ]
    },
    "s1_msk": {
      "href": "75/sentinel12_s1_75_msk.tif",
      "type": "image/tiff; application=geotiff; profile=cloud-optimized",
      "roles": [
        "data"
      ]
    }
  }
}
```

```
    },
    "s1_valid": {
      "href": "75/sentinel12_s1_75_valid.tif",
      "type": "image/tiff; application=geotiff; profile=cloud-optimized",
      "roles": [
        "data"
      ]
    },
    "copdem30_slope": {
      "href": "75/sentinel12_copdem30_75_slope.tif",
      "type": "image/tiff; application=geotiff; profile=cloud-optimized",
      "roles": [
        "data"
      ]
    },
    "copdem30_elevation": {
      "href": "75/sentinel12_copdem30_75_elevation.tif",
      "type": "image/tiff; application=geotiff; profile=cloud-optimized",
      "roles": [
        "data"
      ]
    }
  },
  "stac_extensions": [],
  "properties": {
    "flood": false,
    "split": "test",
    "datetime": "2020-01-01T00:00:00.000000Z",
    "landcover": "Cropland rainfed",
    "s1_srcids": [
      "S1A_IW_SLC__1SDV_20190911T033207_20190911T033234_028968_034906_549E"
    ],
    "date_s1": "20190911"
  },
  "bbox": [
    4200019.424143931,
    5630687.230521306,
    4355505.013540178,
    5788927.5411777515
  ],
```

```
"geometry": {  
  "type": "Polygon",  
  "coordinates": [  
    [  
      [  
        4200019.424143931,  
        5630687.230521306  
      ],  
      [  
        4200019.424143931,  
        5788927.5411777515  
      ],  
      [  
        4355505.013540178,  
        5788927.5411777515  
      ],  
      [  
        4355505.013540178,  
        5630687.230521306  
      ],  
      [  
        4200019.424143931,  
        5630687.230521306  
      ]  
    ]  
  ]  
}
```

4.2 Script for downloading SLC products from CDSE

Code 4-3: Python Script for downloading SLC data: download_slc.py

```
import os
import re
import requests
import pathlib
from creds import *
import pyproj
from datetime import datetime, timedelta
from shapely.ops import transform
from pystac_client import Client
from shapely.geometry import shape
from ukis_valhalla import resto

def get_access_token(username: str, password: str) -> str:
    data = {
        "client_id": "cdse-public",
        "username": username,
        "password": password,
        "grant_type": "password",
    }
    try:
        r = requests.post("https://identity.dataspace.copernicus.eu/auth/realms/CDSE/
        ↪ protocol/openid-connect/token",
            data=data,
        )
        r.raise_for_status()
    except Exception as e:
        raise Exception(
            f"Keycloak token creation failed. Reponse from the server was: {r.json()}"
        )
    return r.json()["access_token"]

def get_refresh_token(username: str, password: str) -> str:
    data = {
        "client_id": "cdse-public",
        "username": username,
        "password": password,
        "grant_type": "password",
```

```
    }
    try:
        r = requests.post("https://identity.dataspace.copernicus.eu/auth/realms/CDSE/ ↵
        ↪ protocol/openid-connect/token",
        data=data,
        )
        r.raise_for_status()
    except Exception as e:
        raise Exception(
            f"Keycloak token creation failed. Reponse from the server was: {r.json()}"
        )
    return r.json()["refresh_token"]

def refresh_access_token(refresh_token: str) -> str:
    data = {
        "client_id": "cdse-public",
        "refresh_token": refresh_token,
        "grant_type": "refresh_token",
    }

    try:
        r = requests.post(
            "https://identity.dataspace.copernicus.eu/auth/realms/CDSE/protocol/openi ↵
            ↪ d-connect/token",
            data=data,
        )
        r.raise_for_status()
    except Exception as e:
        raise Exception(
            f"Access token refresh failed. Reponse from the server was: {r.json()}"
        )

    return r.json()["access_token"]

# Get all items from STAC catalog
catalog = Client.open("catalog.json")
items = [item.to_dict() for item in catalog.get_all_items()]

# Projection from UTM to Web Mercator
```



```
from_crs = pyproj.CRS('EPSG:3857')
to_crs = pyproj.CRS('EPSG:4326')

# Get Areas of Interest, their date and geometry from STAC items
aois = [(re.search('_(.*)_', item["id"]).group(1),
            datetime.strptime(item["properties"]["date_s1"], '%Y%m%d'),
            transform(pyproj.Transformer.from_crs(from_crs, to_crs,
            ↪ always_xy=True).transform, shape(item["geometry"])).wkt,
            shape(item['geometry']).wkt
            ) for item in items]

# Get the Sentinel-1 SLC IDs from date (+-1 day) and geometry
results = {}
for aoi in aois:
    results[aoi[0]] = list(resto.search_aoi(sensor=1, geom=aoi[2], from_date=aoi[1],
    ↪ to_date=aoi[1] + timedelta(days=1), producttype="SLC"))

# Download SLC based on the ID (key: S1S2-Water scene ID, value: Sentinel-1 SLC
↪ product ID)
for key, values in results.items():
    token = get_access_token(os.environ["CDSE_USERNAME"], os.environ["CDSE_PASSWORD"])
    pathlib.Path(f'SLC/{key}').mkdir(parents=True, exist_ok=True)
    for value in values:
        filename = f"{key}/{value}.zip"

        # Only download if folder does not already exist
        if os.path.exists(filename):
            print(f"{filename} already downloaded.")
        else:
            url = f"https://download.dataspace.copernicus.eu/odata/v1/Products({value}
            ↪ )/$value"
            headers = {"Authorization": f"Bearer {token}"}

            # Create a session and update headers
            session = requests.Session()
            session.headers.update(headers)

            # Perform the GET request
            response = session.get(url, stream=True)
```

```
# Check if the request was successful and download file
if response.status_code == 200:
    print(f"Downloading {filename}...")
    with open(filename, "wb") as file:
        for chunk in response.iter_content(chunk_size=8192):
            if chunk: # filter out keep-alive new chunks
                file.write(chunk)
else:
    print(f"Failed to download file. Status code: {response.status_code}")
    print(response.text)
```

4.3 SNAP Graphs

Code 4-4: SNAP Graph: Slice Assembly

```
<graph id="Graph">
  <version>1.0</version>
  <node id="Read">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <useAdvancedOptions>false</useAdvancedOptions>
      <file>${in}</file>
      <copyMetadata>true</copyMetadata>
      <bandNames/>
      <pixelRegion>0,0,73463,15199</pixelRegion>
      <maskNames/>
    </parameters>
  </node>
  <node id="Read(2)">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <useAdvancedOptions>false</useAdvancedOptions>
      <file>${out}</file>
      <copyMetadata>true</copyMetadata>
      <bandNames/>
      <pixelRegion>0,0,73463,13680</pixelRegion>
      <maskNames/>
    </parameters>
  </node>
  <node id="SliceAssembly">
    <operator>SliceAssembly</operator>
    <sources>
      <sourceProduct refid="Read"/>
      <sourceProduct.1 refid="Read(2)"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <selectedPolarisations/>
    </parameters>
  </node>
  <node id="Write">
```

```
<operator>Write</operator>
<sources>
  <sourceProduct refid="SliceAssembly"/>
</sources>
<parameters class="com.bc.ceres.binding.dom.XppDomElement">
  <file>${out}</file>
  <formatName>BEAM-DIMAP</formatName>
</parameters>
</node>
<applicationData id="Presentation">
  <Description/>
  <node id="Read">
    <displayPosition x="23.0" y="20.0"/>
  </node>
  <node id="Read(2)">
    <displayPosition x="26.0" y="97.0"/>
  </node>
  <node id="SliceAssembly">
    <displayPosition x="147.0" y="63.0"/>
  </node>
  <node id="Write">
    <displayPosition x="277.0" y="65.0"/>
  </node>
</applicationData>
</graph>
```

Code 4-5: SNAP Graph: Apply Orbit File, Calibration, TOPSAR Deburst

```
<graph id="Graph">
  <version>1.0</version>
  <node id="Read">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <useAdvancedOptions>false</useAdvancedOptions>
      <file>${in}</file>
      <copyMetadata>true</copyMetadata>
      <bandNames/>
      <pixelRegion/>
      <maskNames/>
    </parameters>
  </node>
  <node id="Apply-Orbit-File">
    <operator>Apply-Orbit-File</operator>
    <sources>
      <sourceProduct refid="Read"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <orbitType>Sentinel Precise (Auto Download)</orbitType>
      <polyDegree>3</polyDegree>
      <continueOnFail>false</continueOnFail>
    </parameters>
  </node>
  <node id="Calibration">
    <operator>Calibration</operator>
    <sources>
      <sourceProduct refid="Apply-Orbit-File"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <sourceBands/>
      <auxFile>Latest Auxiliary File</auxFile>
      <externalAuxFile/>
      <outputImageInComplex>true</outputImageInComplex>
      <outputImageScaleInDb>false</outputImageScaleInDb>
      <createGammaBand>false</createGammaBand>
      <createBetaBand>false</createBetaBand>
    </parameters>
  </node>
</graph>
```

```
<selectedPolarisations/>
<outputSigmaBand>true</outputSigmaBand>
<outputGammaBand>false</outputGammaBand>
<outputBetaBand>false</outputBetaBand>
</parameters>
</node>
<node id="TOPSAR-Deburst">
  <operator>TOPSAR-Deburst</operator>
  <sources>
    <sourceProduct refid="Calibration"/>
  </sources>
  <parameters class="com.bc.ceres.binding.dom.XppDomElement">
    <selectedPolarisations/>
  </parameters>
</node>
<node id="Write">
  <operator>Write</operator>
  <sources>
    <sourceProduct refid="TOPSAR-Deburst"/>
  </sources>
  <parameters class="com.bc.ceres.binding.dom.XppDomElement">
    <file>${out}</file>
    <formatName>BEAM-DIMAP</formatName>
  </parameters>
</node>
<applicationData id="Presentation">
  <Description/>
  <node id="Read">
    <displayPosition x="27.0" y="24.0"/>
  </node>
  <node id="Apply-Orbit-File">
    <displayPosition x="114.0" y="24.0"/>
  </node>
  <node id="Calibration">
    <displayPosition x="235.0" y="24.0"/>
  </node>
  <node id="TOPSAR-Deburst">
    <displayPosition x="328.0" y="24.0"/>
  </node>
  <node id="Write">
```



```
        <displayPosition x="462.0" y="24.0"/>
    </node>
</applicationData>
</graph>
```

Code 4-6: SNAP Graph: Subset

```
<graph id="Graph">
  <version>1.0</version>
  <node id="Read">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <useAdvancedOptions>false</useAdvancedOptions>
      <file>${in}</file>
      <copyMetadata>true</copyMetadata>
      <bandNames/>
      <pixelRegion>0,0,26085,28879</pixelRegion>
      <maskNames/>
    </parameters>
  </node>
  <node id="Subset">
    <operator>Subset</operator>
    <sources>
      <sourceProduct refid="Read"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <sourceBands/>
      <tiePointGrids/>
      <region>0,0,0,0</region>
      <referenceBand/>
      <geoRegion>POLYGON ((37.88931655883789 36.051578521728516, 39.10976028442383
        ↪ 36.051578521728516, 39.10976028442383 37.046531677246094, 37.88931655883789
        ↪ 37.046531677246094, 37.88931655883789 36.051578521728516, 37.88931655883789
        ↪ 36.051578521728516))</geoRegion>
      <subSamplingX>1</subSamplingX>
      <subSamplingY>1</subSamplingY>
      <fullSwath>false</fullSwath>
      <copyMetadata>true</copyMetadata>
    </parameters>
  </node>
  <node id="Write">
    <operator>Write</operator>
    <sources>
      <sourceProduct refid="Subset"/>
    </sources>
  </node>
</graph>
```

```
</sources>
<parameters class="com.bc.ceres.binding.dom.XppDomElement">
  <file>${out}</file>
  <formatName>BEAM-DIMAP</formatName>
</parameters>
</node>
<applicationData id="Presentation">
  <Description/>
  <node id="Read">
    <displayPosition x="30.0" y="32.0"/>
  </node>
  <node id="Subset">
    <displayPosition x="121.0" y="32.0"/>
  </node>
  <node id="Write">
    <displayPosition x="221.0" y="33.0"/>
  </node>
</applicationData>
</graph>
```

Code 4-7: SNAP Graph: Polarimetric Speckle Filter

```
<graph id="Graph">
  <version>1.0</version>
  <node id="Read">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <useAdvancedOptions>false</useAdvancedOptions>
      <file>${in}</file>
      <copyMetadata>true</copyMetadata>
      <bandNames/>
      <pixelRegion>16,0,28087,9293</pixelRegion>
      <maskNames/>
    </parameters>
  </node>
  <node id="Polarimetric-Speckle-Filter">
    <operator>Polarimetric-Speckle-Filter</operator>
    <sources>
      <sourceProduct refid="Read"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <filter>Refined Lee Filter</filter>
      <filterSize>5</filterSize>
      <numLooksStr>1</numLooksStr>
      <windowSize>5x5</windowSize>
      <targetWindowSizeStr>3x3</targetWindowSizeStr>
      <anSize>50</anSize>
      <sigmaStr>0.9</sigmaStr>
      <searchWindowSizeStr>15</searchWindowSizeStr>
      <patchSizeStr>5</patchSizeStr>
      <scaleSizeStr>1</scaleSizeStr>
    </parameters>
  </node>
  <node id="Write">
    <operator>Write</operator>
    <sources>
      <sourceProduct refid="Polarimetric-Speckle-Filter"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
```

```
<file>${out}</file>
<formatName>BEAM-DIMAP</formatName>
</parameters>
</node>
<applicationData id="Presentation">
  <Description/>
  <node id="Read">
    <displayPosition x="26.0" y="50.0"/>
  </node>
  <node id="Polarimetric-Speckle-Filter">
    <displayPosition x="124.0" y="50.0"/>
  </node>
  <node id="Write">
    <displayPosition x="325.0" y="49.0"/>
  </node>
</applicationData>
</graph>
```

Code 4-8: SNAP Graph: Polarimetric Decomposition

```
<graph id="Graph">
  <version>1.0</version>
  <node id="Read">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <useAdvancedOptions>false</useAdvancedOptions>
      <file>${in}</file>
      <copyMetadata>true</copyMetadata>
      <bandNames/>
      <pixelRegion>20,0,68552,25665</pixelRegion>
      <maskNames/>
    </parameters>
  </node>
  <node id="Polarimetric-Decomposition">
    <operator>Polarimetric-Decomposition</operator>
    <sources>
      <sourceProduct refid="Read"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <decomposition>H-Alpha Dual Pol Decomposition</decomposition>
      <windowSize>5</windowSize>
      <outputHAAalpha>true</outputHAAalpha>
      <outputBetaDeltaGammaLambda>false</outputBetaDeltaGammaLambda>
      <outputAlpha123>false</outputAlpha123>
      <outputLambda123>false</outputLambda123>
      <outputTouziParamSet0>true</outputTouziParamSet0>
      <outputTouziParamSet1>false</outputTouziParamSet1>
      <outputTouziParamSet2>false</outputTouziParamSet2>
      <outputTouziParamSet3>false</outputTouziParamSet3>
      <outputHuynenParamSet0>true</outputHuynenParamSet0>
      <outputHuynenParamSet1>false</outputHuynenParamSet1>
    </parameters>
  </node>
  <node id="Write">
    <operator>Write</operator>
    <sources>
      <sourceProduct refid="Polarimetric-Decomposition"/>
    </sources>
  </node>
</graph>
```



```
</sources>
<parameters class="com.bc.ceres.binding.dom.XppDomElement">
  <file>${out}</file>
  <formatName>BEAM-DIMAP</formatName>
</parameters>
</node>
<applicationData id="Presentation">
  <Description/>
  <node id="Read">
    <displayPosition x="20.0" y="58.0"/>
  </node>
  <node id="Polarimetric-Decomposition">
    <displayPosition x="125.0" y="59.0"/>
  </node>
  <node id="Write">
    <displayPosition x="334.0" y="61.0"/>
  </node>
</applicationData>
</graph>
```

Code 4-9: SNAP Graph: Terrain Correction

```

<graph id="Graph">
  <version>1.0</version>
  <node id="Read">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <useAdvancedOptions>false</useAdvancedOptions>
      <file>${in}</file>
      <copyMetadata>true</copyMetadata>
      <bandNames/>
      <pixelRegion>24,0,36296,9022</pixelRegion>
      <maskNames/>
    </parameters>
  </node>
  <node id="Terrain-Correction">
    <operator>Terrain-Correction</operator>
    <sources>
      <sourceProduct refid="Read"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <sourceBands/>
      <demName>Copernicus 30m Global DEM</demName>
      <externalDEMFile/>
      <externalDEMNoDataValue>0.0</externalDEMNoDataValue>
      <externalDEMApplyEGM>true</externalDEMApplyEGM>
      <demResamplingMethod>BILINEAR_INTERPOLATION</demResamplingMethod>
      <imgResamplingMethod>BILINEAR_INTERPOLATION</imgResamplingMethod>
      <pixelSpacingInMeter></pixelSpacingInMeter>
      <pixelSpacingInDegree></pixelSpacingInDegree>
      <mapProjection>PROJCS["WGS 84 / Auto UTM", &#xd;
GEOGCS["WGS84(DD)", &#xd;
  DATUM["WGS84", &#xd;
    SPHEROID["WGS84", 6378137.0, 298.257223563]], &#xd;
  PRIMEM["Greenwich", 0.0], &#xd;
  UNIT["degree", 0.017453292519943295], &#xd;
  AXIS["Geodetic longitude", EAST], &#xd;
  AXIS["Geodetic latitude", NORTH], &#xd;
  AUTHORITY["EPSG", 4326]], &#xd;

```

```

PROJECTION["Transverse_Mercator"], &#xd;
PARAMETER["central_meridian", 165.0], &#xd;
PARAMETER["latitude_of_origin", 0.0], &#xd;
PARAMETER["scale_factor", 0.9996], &#xd;
PARAMETER["false_easting", 500000.0], &#xd;
PARAMETER["false_northing", 10000000.0], &#xd;
UNIT["m", 1.0], &#xd;
AXIS["Easting", EAST], &#xd;
AXIS["Northing", NORTH]]</mapProjection>
  <alignToStandardGrid>false</alignToStandardGrid>
  <standardGridOriginX>0.0</standardGridOriginX>
  <standardGridOriginY>0.0</standardGridOriginY>
  <nodataValueAtSea>false</nodataValueAtSea>
  <saveDEM>false</saveDEM>
  <saveLatLon>false</saveLatLon>
  <saveIncidenceAngleFromEllipsoid>false</saveIncidenceAngleFromEllipsoid>
  <saveLocalIncidenceAngle>false</saveLocalIncidenceAngle>
  <saveProjectedLocalIncidenceAngle>false</saveProjectedLocalIncidenceAngle>
  <saveSelectedSourceBand>true</saveSelectedSourceBand>
  <saveLayoverShadowMask>false</saveLayoverShadowMask>
  <outputComplex>false</outputComplex>
  <applyRadiometricNormalization>false</applyRadiometricNormalization>
  <saveSigmaNought>false</saveSigmaNought>
  <saveGammaNought>false</saveGammaNought>
  <saveBetaNought>false</saveBetaNought>
  <incidenceAngleForSigma0>Use projected local incidence angle from
    ↪ DEM</incidenceAngleForSigma0>
  <incidenceAngleForGamma0>Use projected local incidence angle from
    ↪ DEM</incidenceAngleForGamma0>
  <auxFile>Latest Auxiliary File</auxFile>
  <externalAuxFile/>
</parameters>
</node>
<node id="Write">
  <operator>Write</operator>
  <sources>
    <sourceProduct refid="Terrain-Correction"/>
  </sources>
  <parameters class="com.bc.ceres.binding.dom.XppDomElement">
    <file>${out}</file>
  </parameters>
</node>

```

```
<formatName>BEAM-DIMAP</formatName>
</parameters>
</node>
<applicationData id="Presentation">
  <Description/>
  <node id="Read">
    <displayPosition x="29.0" y="32.0"/>
  </node>
  <node id="Terrain-Correction">
    <displayPosition x="136.0" y="32.0"/>
  </node>
  <node id="Write">
    <displayPosition x="302.0" y="31.0"/>
  </node>
</applicationData>
</graph>
```

4.4 Preprocessing Python Scripts

Code 4-10: Python script to merge neighboring slices: merge.py

```
from rasterio.merge import merge
from rasterio.warp import calculate_default_transform, reproject, Resampling
import rasterio as rio
from pathlib import Path
import rioxarray

def reproject_raster(source, target_crs):
    with rio.open(source) as src:
        transform, width, height = calculate_default_transform(
            src.crs, target_crs, src.width, src.height, *src.bounds
        )
        kwargs = src.meta.copy()
        kwargs.update({
            'crs': target_crs,
            'transform': transform,
            'width': width,
            'height': height
        })

    reprojected_raster = Path(source.parent / f"reprojected_{source.name}")
    with rio.open(reprojected_raster, 'w', **kwargs) as dst:
        for i in range(1, src.count + 1):
            reproject(
                source=rio.band(src, i),
                destination=rio.band(dst, i),
                src_transform=src.transform,
                src_crs=src.crs,
                dst_transform=transform,
                dst_crs=target_crs,
                resampling=Resampling.nearest
            )
    return reprojected_raster

input_path = Path('path/to/directory')
output_path = Path('path/to/directory')
```

```
for folder in input_path.iterdir():
    if folder.is_dir():
        print(f"Processing {folder} ...")
        output_file = output_path / f'{folder.name}_file_ending.tif'

        if output_file.exists():
            print(f"Skipping {output_file.stem}. File already exists.")
            continue

        # Get all raster files in folder
        raster_files = list(folder.rglob('*.tif'))
        raster_to_merge = []

        # Choose the CRS of the first file as target CRS
        with rio.open(raster_files[0]) as ref_raster:
            target_crs = ref_raster.crs

        # Assign CRS
        for file in raster_files:
            with rio.open(file) as raster:
                if raster.crs != target_crs:
                    file = reproject_raster(file, target_crs)
                raster_to_merge.append(rio.open(file))

        # Merge rasters
        merged_raster, output = merge(raster_to_merge)

        # Save merged_raster
        output_meta = raster_to_merge[0].meta.copy()
        output_meta.update({
            "driver": "GTiff",
            "height": merged_raster.shape[1],
            "width": merged_raster.shape[2],
            "transform": output
        })

        with rio.open(output_file, "w", **output_meta) as m:
            m.write(merged_raster)
```

Code 4-11: Python script to collocate and stack image bands: stack.py

```
import os
os.environ['GDAL_MEM_ENABLE_OPEN'] = 'YES'
import rasterio
from ukis_pysat.raster import Image
from pathlib import Path
import rioxarray
import xarray as xr

def stack(tiff_files, output_file):
    # Open the TIFF files
    rasters = [rioxarray.open_rasterio(tiff) for tiff in tiff_files]

    # Reproject all rasters to match the VV/VH image
    rasters_reprojected = [raster.rio.reproject_match(rasters[0]) for raster in
        ↪ rasters]

    # Stack the rasters
    stacked_raster = xr.concat(rasters_reprojected, dim='band')

    # Get reference crs & transform from VV/VH
    with rasterio.open(tiff_files[0], "r") as ref_src:
        ref_crs = ref_src.crs
        ref_transform = ref_src.transform

    # Save as COG
    img = Image(data=stacked_raster.values, dimorder="first", crs=ref_crs,
        ↪ transform=ref_transform)
    img.write_to_file(
        driver="COG",
        path_to_file=output_file,
        dtype=img.arr.dtype,
        compress="LZW",
        kwargs={"BLOCKSIZE": 512, "BIGTIFF": "IF_SAFER", "count":
            ↪ img.arr.shape[0]},
    )
    print(f"Saved stacked image: {output_file}")
```

```
base_directory = "path/to/directory"
output_directory = "path/to/directory"

base_path = Path(base_directory)
output_path = Path(output_directory)
output_path.mkdir(parents=True, exist_ok=True)

for folder in base_path.iterdir():
    if folder.is_dir():
        print(f"Processing {folder} ...")

        # Find files in the folder
        vv_file = next(folder.glob("*img.tif"), None)
        spk_file = next(folder.glob("*Spk_TC.tif"), None)
        ha_file = next(folder.glob("*HA_TC.tif"), None)
        #slope_file = next(folder.glob("*slope.tif"), None) # Not used as slope gets
        ↪ appended during splitting

        if None in [vv_file, spk_file, ha_file]:
            print(f"Skipping {folder}, missing files.")
            continue

        # Skip folder if output files already exist
        output_filename = vv_file.stem.replace("img", "img_pol") + vv_file.suffix
        output_file = output_path / folder.name / output_filename
        if output_file.exists():
            print(f"Skipping. {output_file} already exists.")
            continue

        # Create stack
        tiff_files = [vv_file, spk_file, ha_file]
        stack(tiff_files, output_file)
```

Code 4-12: Python script to replace NaN in stack: replace_nan.py

```
import rasterio
import numpy as np
from pathlib import Path
from ukis_pysat.raster import Image

def rename_file(file, old_ending, new_ending):
    # Construct the new file name
    new_name = file.stem.replace(old_ending, new_ending) + file.suffix
    new_file_path = file.with_name(new_name)

    # Rename the file
    if not new_file_path.exists():
        file.rename(new_file_path)
        print(f"Renamed: {file} -> {new_file_path}")

def replace_nan_with_zero(geotiff_path, output_path, bands_to_process):
    geotiff_path = Path(geotiff_path)
    output_path = Path(output_path)

    with rasterio.open(geotiff_path) as src:
        data = src.read()
        ref_crs = src.crs
        ref_transform = src.transform

        # Process each band and replace NaN with 0
        for band_index in bands_to_process:
            band_data = data[band_index, :, :]
            nan_mask = np.isnan(band_data)
            band_data[nan_mask] = 0
            data[band_index, :, :] = band_data

        # Save the stacked raster as a Cloud Optimized GeoTIFF (COG)
        img = Image(data=data, dimorder="first", crs=ref_crs, transform=ref_transform)
        img.write_to_file(
            driver="COG",
            path_to_file=output_path,
            dtype=img.arr.dtype,
            compress="LZW",
```

```
        kwargs={"BLOCKSIZE": 512, "BIGTIFF": "IF_SAFER", "count":  
        ↪ img.arr.shape[0]},  
    )
```

```
bands_to_modify = [0,1,2,3,4,5,6,7,8] # first band has index 0  
directory = "path/to/directory"  
folders = [f for f in Path(directory).iterdir() if f.is_dir()]  
old_ending = "pol"  
new_ending = "pol_nan"  
  
# rename files to save processed files under original names  
for file in Path(directory).rglob(f"*{old_ending}.tif"):  
    rename_file(file, old_ending, new_ending)  
  
# Process all images and save under the original name  
for folder in folders:  
    for geotiff_file in Path(directory/folder).glob(f"*{new_ending}.tif"):  
        print(f"Processing {geotiff_file} ...")  
  
        new_name = geotiff_file.stem.replace(f"{new_ending}", f"{old_ending}") +  
        ↪ geotiff_file.suffix  
        output_file = Path(directory/folder) / new_name  
  
        if output_file.exists():  
            print(f"Skipping. {output_file} already exists.")  
            continue  
  
        replace_nan_with_zero(geotiff_file, output_file, bands_to_modify)
```

Code 4-13: Python script to create a valid pixel mask: calc_valid_mask.py

```
import rasterio as rio
import numpy as np
from pathlib import Path

def create_mask(geotiff_path, output_path, bands):
    with rio.open(geotiff_path) as src:
        data = src.read()

        mask = np.ones(data.shape, dtype=np.uint8)

        # calculate mask for each band
        for band_index in bands:
            band_data = data[band_index, :, :]
            band_mask = np.where(band_data == 0, 0, 1)
            mask[band_index, :, :] = band_mask

        # create final mask by multiplying all band masks
        final_mask = np.prod(mask, axis=0)

        # Save mask file
        mask_meta = src.meta.copy()
        mask_meta.update({
            "dtype": "uint8", # Data type for mask (0 or 1 values)
            "count": 1        # Single-band mask
        })

        with rio.open(output_path, "w", **mask_meta) as m:
            m.write(final_mask, 1)

directory = "path/to/directory"
folders = [f for f in Path(directory).iterdir() if f.is_dir()]
bands = [0,1,2,3,4,5,6,7,8] # first band has index 0

for folder in folders:
    print(f"Processing {folder} ...")
    path = Path(directory/folder)
```

```
# Create the valid mask for the processed SLC data
for geotiff_file in path.glob("*_pol.tif"):
    output_file = path / geotiff_file.stem.replace("pol", "pol_valid")
    output_path = path / f"{output_file}.tif"
    if output_path.exists():
        print(f"Skipping. {output_file} already exists.")
        continue
    create_mask(geotiff_file, output_path, bands)
    print(f"Mask created: {output_file}.tif")
```
