MSc Artificial Intelligence
Master Thesis

AI-Driven Segmentation and Classification of Vegetation in the Biesbosch Floodplain Using Remote Sensing Data

by
Eva Claire Virginie Gmelich Meijling
12162124

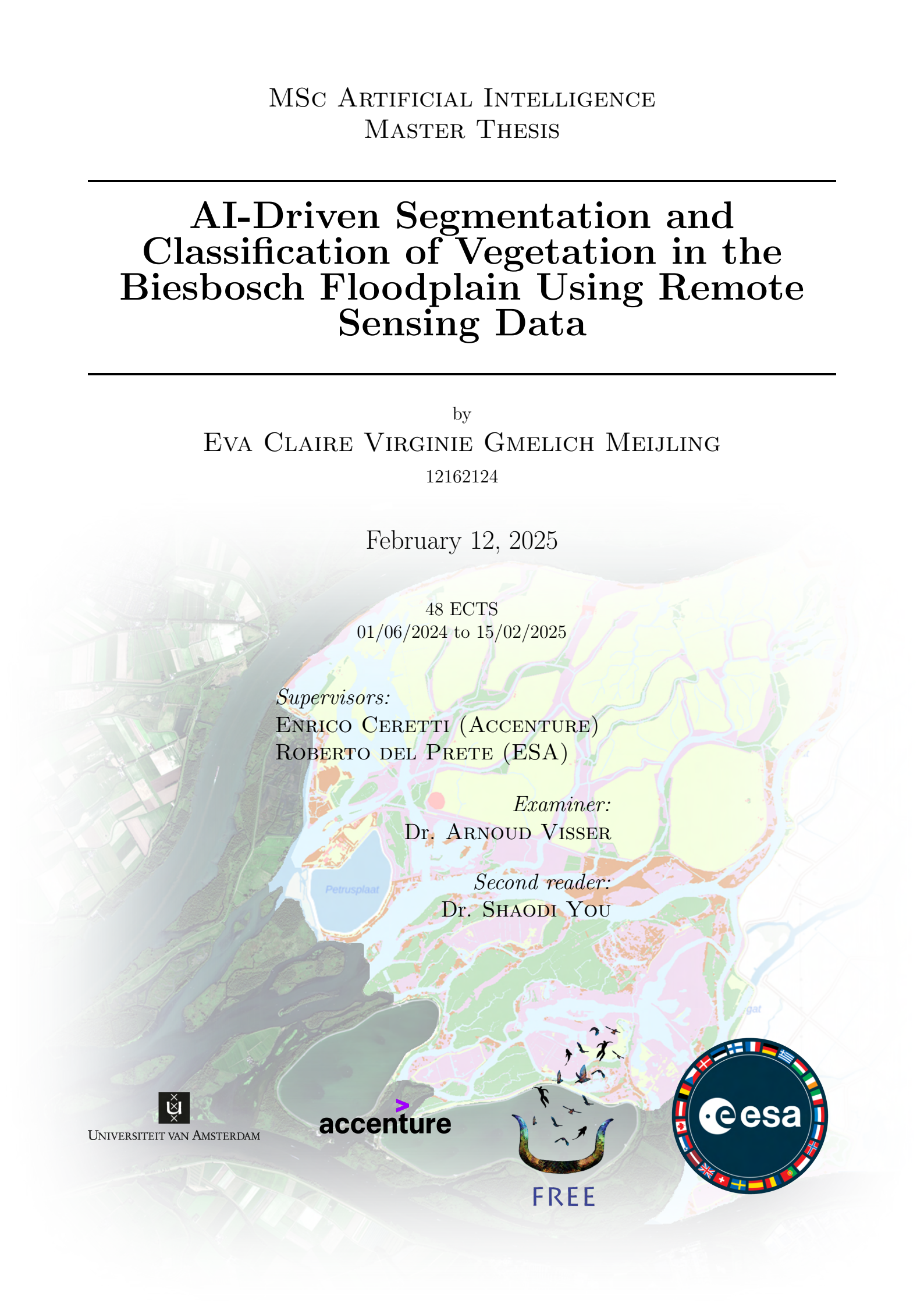
February 12, 2025

48 ECTS
01/06/2024 to 15/02/2025

Supervisors:
Enrico Ceretti (Accenture)
Roberto del Prete (ESA)

Examiner:
Dr. Arnoud Visser

Second reader:
Dr. Shaodi You



Contents

Abstract

Wetlands are dynamic ecosystems that play an important role in flood mitigation. To reduce flood risks and preserve their ecosystems, wetland areas must be monitored effectively, supporting informed decision-making and necessary actions. This study presents a methodology for wetland land cover classification by training a U-Net model from scratch for semantic segmentation across six wetland areas in the Netherlands. The baseline model trained on Sentinel-2 satellite imagery achieves an accuracy of 85.26%.

Annotated datasets for land cover and land use classification in remote sensing are often challenging to obtain due to the labor-intensive and time-consuming process of manual labeling. To address this, the study also examines the impact of pretraining and higher-resolution satellite imagery on the performance of the U-Net. Self-supervised learning (SSL) is used for pretraining, leveraging large volumes of unlabeled remote sensing data to extract features, thereby reducing the reliance on annotated samples. The effect of higher resolution is assessed by downsampling high-resolution labels to a lower resolution and comparing the impact on U-Net performance.

This approach is demonstrated for land cover classification in the Biesbosch floodplain, a dynamic wetland area in the Netherlands, using publicly available optical satellite data. Results show that SSL pretraining with an autoencoder can improve classification accuracy, especially when labeled data is limited. When only 10% of the dataset is annotated, pretraining leads to a 1.23% relative improvement in accuracy compared to training from scratch. However, the performance gap between SSL and purely supervised models diminishes when more labels become available during training, and as training progresses over an increasing number of epochs. Furthermore, this study shows that, when labels are sparse, using higher-resolution imagery with corresponding labels also improves classification accuracy by 4.94% relative improvement compared to lower-resolution imagery.

Limitations in this research include the misalignment between the pretraining data and the target geographic regions, likely reducing the effect of the pretraining. Additionally, an imbalance in the manually annotated dataset may affect the reliability of performance metrics.

Chapter 1

Introduction

Throughout the years, remote sensing has evolved significantly. In the early 20th century, the first steps in capturing spatial data were made with airplanes conducting aerial photography. By the mid-20th century, major advancements occurred with Earth-observing satellites, enabling large-scale and continuous observations of Earth's surface. More recently, since the early 2000s, Unmanned Aerial Vehicles (UAVs), also known as drones, have been deployed to capture high-resolution imagery of more localized areas, offering greater flexibility and accessibility compared to traditional plane-based or satellite-based methods [71].

As shown in Figure 1.1, these three remote sensing modalities operate across different spatial scales and resolutions, capturing varying levels of detail depending on the platform. Satellites can now achieve very high resolutions, comparable to those originally provided by airplanes and even drones. Since many satellites continuously orbit the Earth, they are also among the most accessible data sources for remote sensing applications. Modern optical satellites, such as Sentinel-2 [26] and Landsat [81], achieve resolutions of approximately 10 meters per pixel, providing detailed insights at a regional scale wile remaining openly accessible. The highest-resolution instruments are currently often offered by proprietary platforms, such as Maxar's WorldView-3[1], which provides native resolutions of up to 30 cm per pixel and generates 15 cm high-definition (HD) imagery, enabling precise analyses of urban areas, environmental features, and even applications in defense and military intelligence.

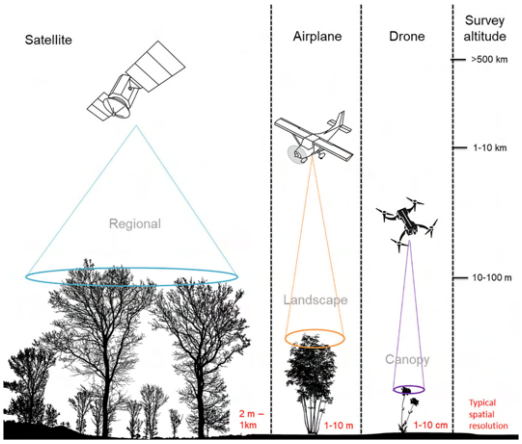


Figure 1.1: The approximate operating altitudes of satellite-, airplane-, and drone-based imaging instruments. The corresponding fields of view used to capture vegetation from these different altitudes provide an indication of the landscape extend captured by sensors onboard each imaging platform [30].

[1] https://www.maxar.com/maxar-intelligence/products/optical-imagery

Remote sensing for Earth observation provides an efficient method for monitoring dynamic ecosystems, overcoming the limitations of ground-based assessments, which often have restricted spatial coverage, high operational costs, and time-consuming procedures [42]. This is particularly valuable for large or inaccessible areas, such as wetlands and floodplains, which frequently undergo rapid and unpredictable environmental changes [29].

Wetlands are ecosystems where water, either permanently or seasonally, saturates the soil [68]. They support diverse plant and animal species, act as natural filters by trapping pollutants, improve water quality, and play a vital role in carbon sequestration [68]. Floodplains, a specific type of wetland found adjacent to rivers, absorb excess water during flooding, playing a crucial role in maintaining environmental stability and mitigating flood risks. Both wetlands and floodplains reduce flood risks by retaining water and regulating sediment transport. These processes are closely linked to the vegetation's roughness, which is influenced by the vegetation's distribution, density, and type.

The Biesbosch, located in the Netherlands, is a wetland that plays an important role in flood mitigation in nearby regions such as Rotterdam and Dordrecht. The distribution and density of the area's vegetation strongly influence the water flow patterns, making accurate classification of vegetation essential to effectively monitor and manage the ecosystem. Monitoring the area helps identify locations where water builds up due to excessive or overly rough vegetation. Action can then be taken, such as mowing by humans or deploying natural grazers. Natural grazing organizations like Free Nature[2] employ natural grazers, including water buffaloes, wild horses, and deer, to manage vegetation. Especially in difficult-to-access areas, these grazers provide an effective alternative to human intervention, while simultaneously promoting biodiversity and improving ecosystem functionality.

Problem areas with high vegetation roughness can be identified using deep learning to perform semantic segmentation on satellite imagery, as depicted in Figure 1.2. Especially with high resolution imagery, the distribution, variation and specific type of vegetation can be accurately determined. Nevertheless, despite the increasing accessibility of Earth observation data and the large amount of public satellite imagery that is available for the Netherlands, extracting insights from remote sensing imagery for land use and land cover detection using deep learning methods still heavily relies on labeled datasets.

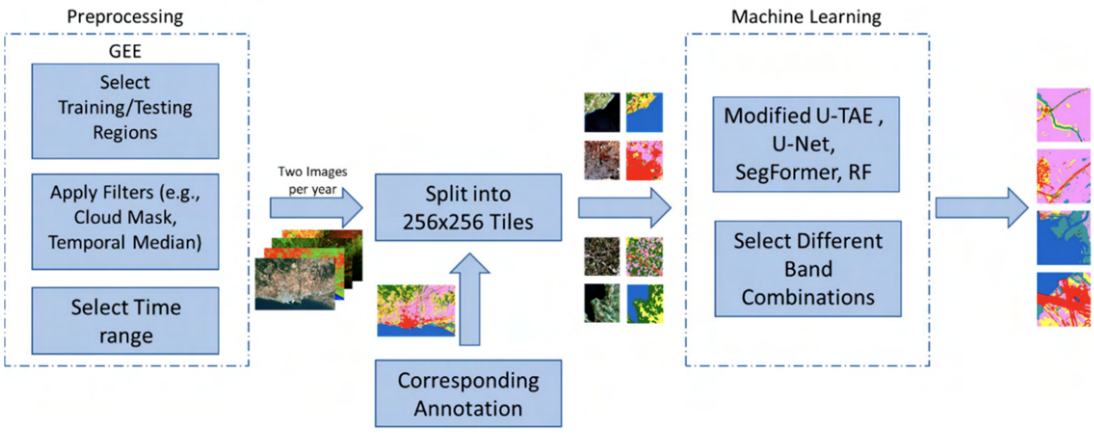


Figure 1.2: Flowchart of land cover classification approach using deep learning [74].

In addition to performing semantic segmentation, this study addresses the challenge of having limited labeled data available. This is achieved by employing self-supervised learning (SSL)

[2] https://www.freenature.nl/nl

and leveraging high-resolution imagery. SSL utilizes large volumes of unlabeled data to extract meaningful features, reducing the need for extensive manual labeling while improving model performance. By integrating an autoencoder with a U-Net architecture, this study evaluates whether SSL can improve the efficiency and accuracy of land cover classification in the Biesbosch when only limited labeled data is available. Additionally, the study examines the impact of increasing the resolution of the data and its corresponding labels on semantic segmentation performance, aiming to reduce reliance on large quantities of labels by emphasizing the use of a smaller set of high-quality annotations.

Therefore, the overarching research question that this thesis addresses is:

How can effective semantic segmentation for land cover classification in dynamic wetlands be achieved while reducing reliance on annotated data?

The following two sub-research questions will help answer this question by exploring how remote sensing and advanced machine learning techniques can address the challenges associated with vegetation classification in wetlands.

1. Can self-supervised learning improve the efficiency and accuracy of land cover classification for estimating vegetation roughness in dynamic wetland environments?

2. How does image resolution influence the amount of labeled data required for accurate vegetation classification?

This thesis will first discuss the context and background of wetland areas, with a specific focus on the Biesbosch. It will then explore the use of remote sensing for wetland monitoring and the role of machine learning in analyzing remote sensing data. Next, the methodology section will detail the baseline U-Net model and outline the two experiments conducted to answer the sub-research questions. An analysis of the results is presented, followed by a discussion and conclusion of the study's limitations, future work, and the implications of using self-supervised learning and high resolution imagery for wetland monitoring and ecosystem management.

Chapter 2

Background

This chapter provides background information on wetlands, focusing on the Biesbosch, the role of remote sensing (RS) in wetland monitoring, and introduces the application of machine learning techniques for remote sensing.

2.1 Wetlands and Floodplains

A wetland is an area where the soil surface is either seasonally or permanently covered with water throughout the year [68]. As shown in Figure 2.1[1], wetlands are vital ecosystems for both humans and nature for several reasons, including their ability to naturally filter pollutants, reduce flood risks by absorbing excess water, and combat climate change by capturing and storing carbon. Despite covering only about 6% of the Earth's land surface, wetlands hold 20-30% of the world's carbon pool [56], helping mitigate climate change by reducing carbon dioxide levels in the atmosphere. As highlighted by the Ramsar Convention[2], an international treaty for the conservation and sustainable use of wetlands, these ecosystems are also critical for biodiversity, providing habitats and breeding grounds for 40% of all plant and animal species.

Floodplains are considered a type of wetland which are low-lying areas adjacent to rivers and streams that become inundated during periods of high water. They play a crucial role in dispersing and slowing floodwaters, thereby protecting surrounding regions from potential flood damage [1].

Although wetlands help combat climate change, they are also highly vulnerable to its effects [9]. Rising temperatures and shifting rainfall patterns are causing many wetlands to lose water, leading to drying or shrinking, especially in areas already facing water shortages [68]. Coastal wetlands, such as mangroves and salt marshes, are particularly at risk from rising sea levels, which flood habitats with saltwater and reduce their size. These changes not only harm the biodiversity that wetlands support, but also weaken their ability to store carbon, further contributing to climate change.

2.1.1 The Biesbosch Wetland

The Biesbosch is located in the Netherlands between the provinces of Zuid-Holland and Noord-Brabant, as shown in Figure 2.2 (a). It is one of Europe's few remaining freshwater tidal areas, shaped by the St. Elisabeth's Flood in 1421[3]. This flood disaster transformed an agricultural area known as the Groote and Zuidhollandse Waard into an inland sea stretching from Dordrecht to Geertruidenberg. Over time, sand and clay carried by water currents settled,

[1] `https://europe.wetlands.org/wetlands/what-are-wetlands/`

[2] `https://www.ramsar.org`

[3] The official Biesbosch National Park website: `https://np-debiesbosch.nl`

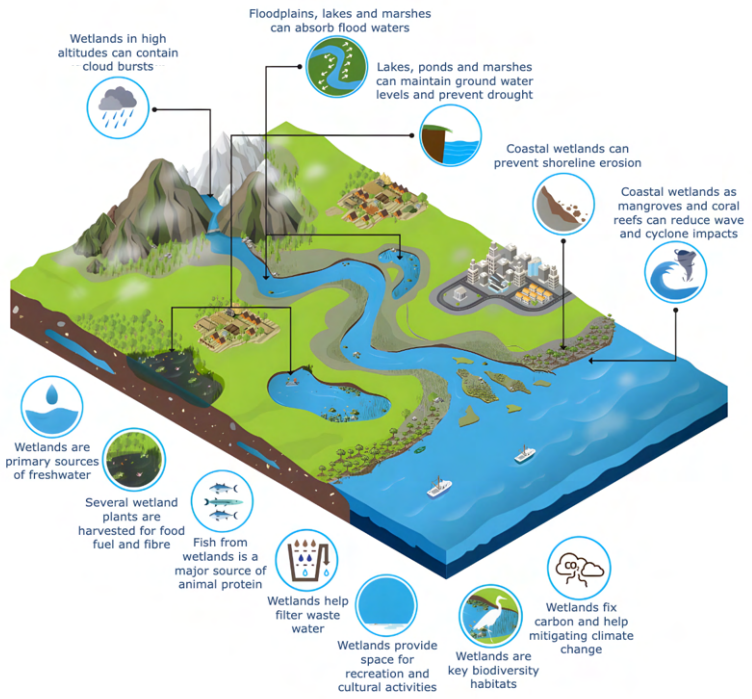


Figure 2.1: Key processes provided by wetlands[1].

making the inland sea increasingly shallow. The first vegetation to colonize the emerging sandbanks during low tide were rushes (in Dutch "biezen"), as they thrive best with their roots in water. The name "Biesbosch" means "forest of rushes", and refers to this type of vegetation that was abundantly present in the area. As more land emerged and became less frequently inundated, the rushes were naturally replaced by reed vegetation that could better tolerate the drier conditions. Subsequently, willows began to grow and were cultivated in embanked fields known as "grienden." These willows were valued for their branches, marking the transition from a rush-dominated landscape to the willow forests that largely define the Biesbosch region today.

By 1850, two-thirds of this inland sea had been reclaimed for agriculture. A significant change occurred between 1850 and 1870 with the construction of the Nieuwe Merwede canal, dividing the Biesbosch into two parts. The area underwent its most dramatic hydrological change with the Delta Works Project, especially the completion of the Haringvlietdam in 1970. Before the Delta Works, tidal differences of two meters were common in the Biesbosch, facilitated by two open sea arms: the Nieuwe Waterweg and the Haringvliet. Originally, the tidal flow entered primarily through the Haringvliet. However, due to the construction of the Haringvliet dam, the connection to the sea was blocked, reducing the tidal movements. In addition to these earlier interventions, more recent modifications such as depoldering have further altered the landscape and water flow in the area, as shown in Figure 2.2. Today, while the Nieuwe Waterweg remains open, tidal differences vary from just 20-30 centimeters in the Dordtse and Brabantse Biesbosch to 60-80 centimeters in the Sliedrechtse Biesbosch, the latter experiencing larger fluctuations due to its more northern location. Despite these changes, the Biesbosch, which was designated as a National Park in 1994, remains one of the largest and most valuable natural areas in the Netherlands, covering approximately 9,700 hectares.

The Biesbosch experiences seasonal variations in water levels, particularly during winter and spring when the Rhine and Maas rivers can cause flooding in many polders. These hydrological and ecological conditions specific to the Biesbosch wetland have given rise to a variety of vegetation types. Hence, understanding these vegetation types and their characteristics is

important for managing the area's water flow, sediment transport, and biodiversity.

A couple of studies have been conducted on the Biesbosch to investigate its hydrology, ecology and vegetation dynamics [36, 73, 76]. Most date from the late 20th century and focus on the impact of tidal reduction caused by the Delta Works, particularly on sedimentation patterns and vegetation succession. Some research has also been done on the restoration of natural processes, such as reintroducing tidal dynamics and creating new wetland habitats within the Biesbosch itself to support biodiversity [22]. Additionally, the role of vegetation in influencing water flow and sediment transport has been extensively explored [75], providing valuable insights for flood management and habitat conservation in this unique freshwater tidal ecosystem. To date, no studies have explicitly utilized remote sensing techniques to investigate the Biesbosch in specific.

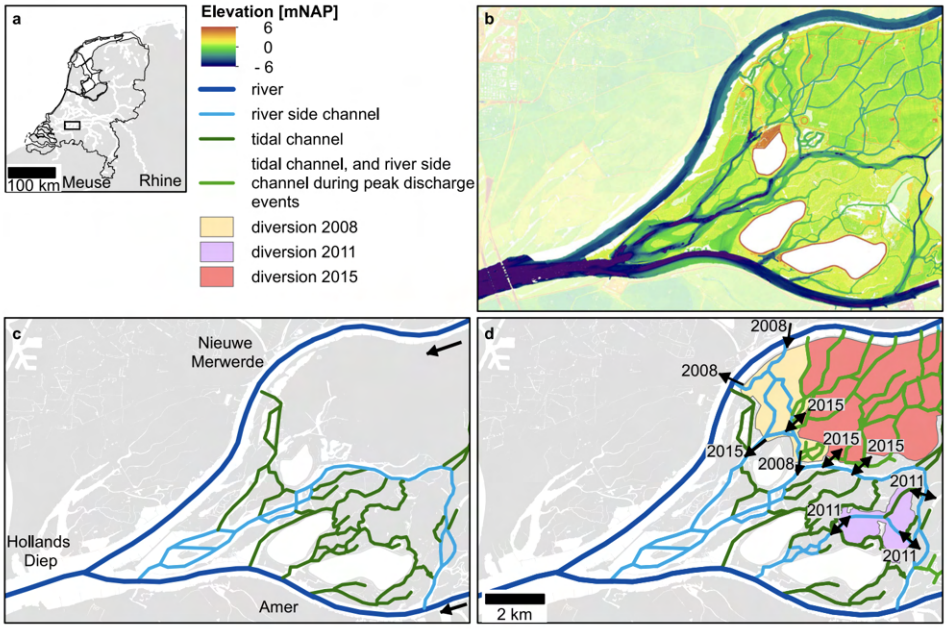


Figure 2.2: Figure from [76], where (a) depicts the location of The Biesbosch in The Netherlands (b) the elevation above Dutch Ordnance Datum (m NAP), (c) schematic overview of the main channels and their types before depoldering, and (d) a schematic layout of the channels after depoldering in the Brabantse Biesbosch area. The year of opening and primary flow direction of the new channels are indicated in (d).

2.1.2 Wetland Vegetation Types

Variations in vegetation height and density result in different levels of roughness, which in turn affects water flow through the area. Wetlands are characterized by either permanently or seasonally water saturated soil, and not unified by a common vegetation type. Instead, every wetland area hosts a diverse range of plant species depending on the local hydrological and environmental conditions [72].

Vegetation in the Biesbosch

The Biesbosch contains many different vegetation types, highly influenced by the tidal environment. Historically, there were a lot of reed and sedge marshes. Over time, human activities such as land reclamation for agriculture, griend management, and flooding control have changed the types of vegetation. Currently, the vegetation includes alluvial forests, reed and

sedge marshes, and grasslands[4]. The floodplain forests, found on higher terrain, have transitioned from managed grienden into naturalized willow forests. However, many willows are aging and lack regeneration. Reed and sedge marshes dominate lower floodplain areas and transition into pioneer vegetation in shallow, silty riverbanks. Managed grasslands, such as hayfields and summer polders, are present in areas where the water levels are highly controlled.

Vegetation Roughness

There is significant variation in plant density and stiffness between vegetation types, leading to different levels of flow resistance [23]. Dense, tall reeds slow water more effectively than shorter, flexible plants. Woody vegetation with thick stems and complex canopies further reduces flow, promotes sediment deposition, and shapes the ecosystem's long-term structure.

Vegetation roughness, defined as the degree to which plant structures obstruct water flow, plays a key role in the Biesbosch. It influences water movement, sediment transport, and flood dynamics. Classifying vegetation roughness accurately is therefore essential for effective flood management and ecosystem conservation.

In wetlands, the roughness of the vegetation affects water velocity and turbulence [44]. Above the vegetation, velocity increases with height following a logarithmic pattern. Within the vegetation, velocity increases almost linearly but stabilizes at low discharge (small water volumes). As discharge increases (higher water volumes), turbulence intensifies. This turbulence creates mixing, where faster-moving water from above interacts with slower-moving water within the vegetation, causing the different layers in the water to mix. This effect is strongest near the top of the vegetation, where the flow transitions from being slowed by plants to moving more freely.

2.2 Remote Sensing for Wetlands

Wetlands face growing threats from human development and climatic impacts on their ecosystems [68]. Monitoring these environments is crucial for understanding changes and making informed decisions to mitigate or reduce these threats. If not properly observed, changes in wetlands can lead to disruptions in surrounding areas, such as increased flooding risks. Jafarzadeh et al. [42] reviewed 334 studies on wetland monitoring using remote sensing (RS) over the last three decades. More than half of the studies focused on classifying wetland zones, while others explored internal changes, vegetation mapping, and the delineation of wetland extent.

Remote sensing has enabled the collection of data about objects or areas without direct physical contact [64]. It employs sensors to detect and measure various forms of energy, such as electromagnetic radiation, which is either emitted, reflected, or scattered by the objects being observed. This technology has become indispensable in monitoring and analyzing the Earth's surface, especially in environmental and land cover studies.

2.2.1 Types of Remote Sensing

Remote sensing studies on wetlands often utilize either passive or active techniques, or a combination of both. These methods can identify for instance: vegetation patterns, map wetland boundaries, or assess structural features such as canopy height and surface roughness.

[4] https://www.brabant.nl/onderwerpen/natuur-landschap/natura-2000-gebieden/biesbosch/

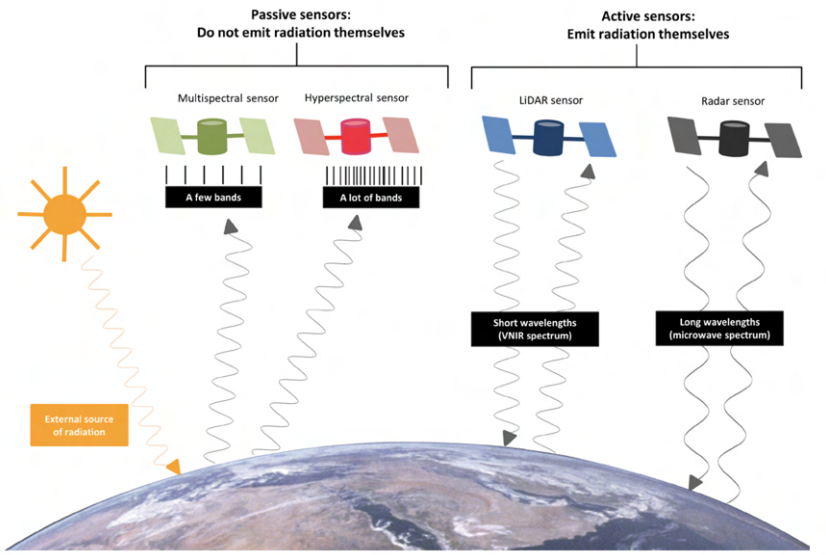


Figure 2.3: Passive and active sensors for remote sensing.

Passive Sensors

Optical remote sensing has been in use for over fifty years [45] and function as passive devices, as shown in Figure 2.3 by WWF Germany[5], detecting sunlight that is reflected by the Earth's surface [43]. They capture data across multiple spectral bands, each sensitive to different wavelengths, enabling detailed analysis of surface features such as vegetation, water bodies, and urban areas. Multispectral remote sensing is based on capturing data in discrete spectral bands, chosen according to their relevance for specific observations and their alignment with atmospheric windows wavelength ranges where Earth's atmosphere is transparent to incoming and outgoing radiation. An overview of such spectral bands and their common applications in RS, can be found in Table 2.1.

Table 2.1: Spectral bands captured by optical sensors and their applications in the wetlands domain (roughly classified) [2].

Spectral Band	**Wavelength Range**	**Common Applications**
Ultraviolet (UV)	<400 nm	Ozone and atmospheric monitoring
Coastal/Aerosol Band	Around 400 nm	Water clarity, chlorophyll, aerosols
Visible Bands	400–700 nm	Surface features, vegetation, water
Blue Band	400–500 nm	Water quality, atmospheric corrections
Green Band	500–600 nm	Vegetation health, biomass
Red Band	600–700 nm	Vegetation stress, chlorophyll
Red-Edge Band	690–750 nm	Vegetation stress, biochemical properties
Near-Infrared (NIR)	700–1300 nm	Vegetation health, leaf structures
Shortwave Infrared	1300–2500 nm	Water content, soil moisture, minerals

Multispectral optical satellite data can be retrieved from, among others, Sentinel, Landsat or WorldView satellites. Sentinel-2 includes 13 bands, covering visible, NIR, SWIR, and red-edge wavelengths. Landsat-8/9 captures data in visible, NIR, SWIR, and thermal bands.

[5]Official technical report by WWF Germany on remote sensing for conservation, retrieved from `https://www.wwf.de/`.

WorldView-3 provides a wide range of spectral capabilities, including visible, NIR, SWIR, and additional specialized bands for vegetation and material analysis.

These multispectral sensors enable the calculation of vegetation indices, which are mathematical combinations of spectral bands designed to enhance vegetation signals in remote sensing data. One of the most widely used vegetation indices is the Normalized Difference Vegetation Index (NDVI). NDVI is calculated using the near-infrared (NIR) and red (RED) bands, as plants reflect strongly in the NIR band while absorbing light in the RED band. The formula for NDVI is:

$$NDVI = \frac{(NIR - RED)}{(NIR + RED)} \tag{2.1}$$

NDVI provides a measure of vegetation health, with higher values indicating denser, healthier vegetation. Figure 2.4 shows an NDVI map of the Biesbosch, where the three main water basins, surrounding cropland, and river networks are clearly distinguishable due to their significant differences in vegetation cover and reflectance characteristics. Other vegetation indices, such as the Enhanced Vegetation Index (EVI) and the Soil-Adjusted Vegetation Index (SAVI), are tailored to specific conditions like dense canopies or areas with high soil reflectance.

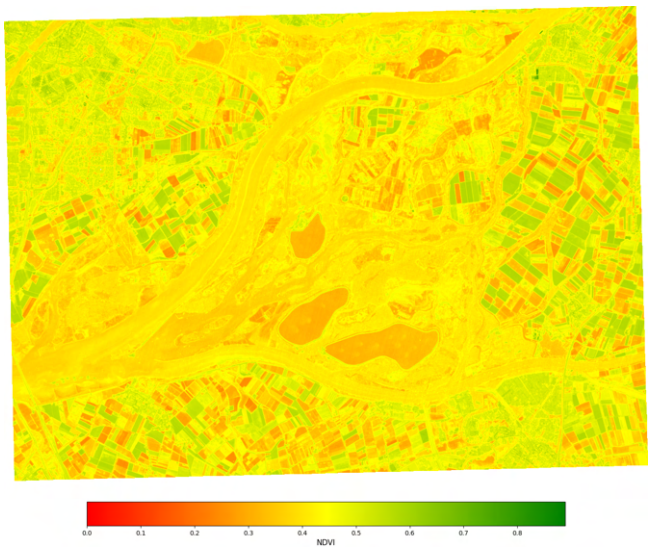


Figure 2.4: NDVI map of the Biesbosch region derived from Sentinel-2 data. Red indicates low vegetation health or sparse vegetation, and green represents high vegetation health or dense vegetation. The color bar below the map shows the corresponding NDVI values

As previously discussed, multispectral sensors offer data across 4 to 20 bands, enabling general land use and cover classification. Hyperspectral sensors, on the other hand, provide much finer spectral resolution with hundreds of contiguous narrow bands, making them suitable for identifying subtle differences in vegetation types or stress levels [34]. Their continuous spectral coverage allows for greater flexibility in selecting wavelengths that align with atmospheric windows, reducing the impact of absorption and scattering. However, the accessibility and large-scale application of hyperspectral imaging are constrained by the absence of a globally operational satellite that provides high spatial and spectral resolution.

To fully leverage satellite data for such applications, appropriate pre-processing is required to ensure consistency and accuracy. These processing levels range from raw reflectance measurements to fully corrected surface reflectance datasets, each suited for different analytical needs.

Common Processing Levels

- **Top-of-Atmosphere (TOA):** Reflectance values measured at the satellite sensor, uncorrected for atmospheric effects. Suitable for large-scale studies but less accurate for surface-specific analyses.
- **Surface Reflectance (SR):** Corrected for atmospheric distortions to reflect actual surface properties. Used for precise vegetation indices and time-series analysis.
- **Harmonized SR (HSR):** Ensures consistency across multiple satellite platforms or within a single sensor over time. Ideal for global monitoring and multi-sensor studies.

Harmonized data improves accuracy, facilitates interoperability, and is critical for detecting environmental changes. In this research, pre-processed Harmonized Surface Reflectance (HSR) data from Sentinel-2 is used to ensure temporal consistency across different acquisition dates.

One of the advantages of optical remote sensing is its ability to offer high spatial resolution and frequent revisit times, making it ideal for continuous environmental monitoring. However, a key limitation of this method is its susceptibility to cloud cover, which can obstruct the view of the surface. Contrary to passive sensors, active sensors do not rely on sunlight and can operate in all weather conditions, providing a reliable alternative for Earth observation.

Active Sensors

Active sensors, such as radar and LiDAR, became widely adopted in the last two to three decades. As shown in Figure 2.3, active sensors emit energy and measure the reflected or backscattered signal [45]. They both look at the time delay in defining the object.

Radar uses microwave signals that can penetrate clouds and are less affected by atmospheric conditions, making it particularly valuable for land mapping in cloudy regions. One of the most widely used radar techniques is Synthetic Aperture Radar (SAR) [57], with Sentinel-1 being one of the primary satellites providing SAR data. SAR systems use the motion of the radar sensor to simulate a large antenna using the Doppler effect to process frequency shifts caused by the sensor's movement relative to the target, allowing for high-resolution images of the Earth's surface. This technique is particularly effective for mapping surface roughness, soil moisture, and vegetation structure, even in cloudy or dark conditions.

Light Detection and Ranging (LiDAR) is another active sensing technology that uses laser pulses to measure distances, creating high-precision three-dimensional maps. While primarily deployed on aircraft and drones for detailed terrain mapping, LiDAR sensors are also carried on satellites like GEDI and ICESat-2 for global-scale monitoring. It is ideal for detailed topographic mapping, vegetation height estimation, and modeling urban environments.

Processing data from active sensors can be more complex than passive optical data [34]. Radar data, for instance, requires advanced techniques to interpret backscatter signals, which depend on surface roughness, moisture content, and angle of observation. LiDAR data, while providing high spatial resolution, it often has low spatial coverage and requires significant computational resources to process raw point clouds into usable formats such as digital elevation models (DEMs) or vegetation metrics.

2.3 Machine Learning for Remote Sensing

Since the launch of the first satellite in 1957, the volume and complexity of remote sensing data has continued to grow. For instance, NASA's Earth Observing System Data and Information System (EOSDIS), which stores NASA's Earth science data, is projected to exceed 300

petabytes by 2030 [63]. This increasing volume requires increasingly advanced tools to manage and analyze the data efficiently. Artificial intelligence (AI) has grown to play a vital role in generating insights by extracting, processing, and interpreting these vast datasets.

Since the early 2010s, research has focused on reducing data processing latency by enabling real-time analysis directly on satellites instead of transmitting raw data to Earth [31]. On-the-edge processing, as discussed by [28], optimizes data transmission by sending only relevant or pre-processed information, improving the efficiency and accelerating ground-based analysis. Advances in AI further enhance this approach by enabling more efficient data filtering, feature extraction, and anomaly detection directly onboard, making satellite systems more autonomous and responsive

Simultaneously, AI's ability to detect patterns and trends in large datasets has seen significant advancements. The development of deep learning methods, such as multilayered neural networks, vision transformers, generative adversarial networks (GANs), and large-scale visual segmentation models, has greatly enhanced the capacity of AI in remote sensing [43]. These advancements allow for more accurate and actionable insights that were previously difficult to achieve with traditional RS methods.

With the introduction of foundation models in 2021, a paradigm shift took place in AI-driven remote sensing. Unlike task-specific models, which are optimized for specific applications, foundation models, such as CROMA [27], are pretrained on a diversity of datasets, allowing them to generalize across multiple remote sensing tasks with minimal fine-tuning. As highlighted by Lu et al. [50], these models excel in transfer learning, few-shot adaptation, and multimodal integration, making them highly effective for diverse remote sensing tasks. However, their limitations include domain gaps between natural image datasets, on which they are often pretrained. Additionally, they require high computation resources, task-specific fine-tuning for optimal performance, and pose challenges in interpretability.

Looking ahead, as the volume of remote sensing data continues to grow, the speed at which data is captured and processed, often referred to as the *ingest rate*, will become increasingly critical. AI-based methods will need to evolve further to handle the vast amounts of data generated, especially for real-time applications. In addition to increasing ingest rates, we can expect the development of even more sophisticated AI techniques, such as advanced unsupervised learning models, which can autonomously interpret and predict environmental changes without large labeled datasets. These evolving methods will further improve the accuracy and speed of AI-based analysis, making it an even more powerful tool in understanding and responding to complex environmental processes. More details on ML techniques applied in RS are explained in the next chapter.

Chapter 3

Machine Learning for Remote Sensing

This chapter explores the application of machine learning (ML) to remote sensing (RS). It reviews a range of ML methodologies applied in RS and introduces Self-Supervised Learning (SSL) as an approach to overcome the challenge of having limited annotated data.

3.1 Machine Learning Methods in Remote Sensing

ML methods have become common in analyzing RS data, enabling many use cases for Earth Observation like land cover mapping, object detection, and change detection [43]. Three prominent ML approaches in RS are classification, object detection, and semantic segmentation. Depending on the use case, data resolution, and spatial complexity of the landscape, the chosen ML methods can be pixel-based, object-based, or scene-based [51, 18].

Classification assigns a label to an image, region or individual pixels, depending on the use case. Traditional pixel-based classification operates solely on spectral features, which can lead to misclassification in high-resolution imagery where spatial context is crucial [4]. Object-based classification overcomes this limitation by grouping pixels into meaningful segments before applying classification. By incorporating texture and shape, it is particularly useful for applications like land use mapping. Scene-based classification, on the other hand, analyzes entire images for broad land cover categorization [18]. It uses deep learning models like CNNs and vision transformers to capture spatial patterns and contextual relationships on a larger scale.

Object detection is also object-based but differs from classification in that it focuses on identifying and localizing specific objects within an image rather than assigning it to a broad category. It detects distinct elements like buildings, roads, or water bodies by analyzing spatial relationships, texture, and shape features. In RS, object detection is particularly useful for applications like urban monitoring, building planning, disaster prediction and even military applications [47].

Semantic segmentation is inherently a pixel-based process and assigns a class label to each pixel in an image. While the process operates at the pixel level, convolutional neural networks (CNNs) are used to capture the broader structural and spatial context. By doing so, it considers relationships between neighboring pixels and multi-scale features. This approach enables delineation of complex patterns and structures, making it effective for tasks like land use and land cover (LULC) mapping, where understanding the spatial distribution of classes (e.g., vegetation, water, urban areas) at a granular level is necessary [84]. Furthermore, semantic segmentation can also be used for change detection, where pixel-level comparisons over time

show changes in land cover or infrastructure [32, 35].

The aforementioned ML tasks can be addressed with conventional and Deep Learning (DL) algorithms. The choice mainly depends on factors such as the complexity and volume of the data, as well as the available computational resources. As can be seen in Figure 3.1, the main difference lies in how the features are extracted and processed. Conventional ML methods require manual feature engineering based on spectral, spatial or temporal properties, whereas DL models automatically learn these features from raw remote sensing data [3].

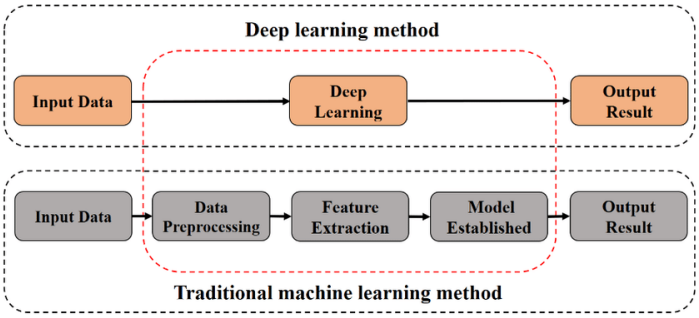


Figure 3.1: Comparison of deep learning (top) and traditional machine learning (bottom) workflows [14].

3.1.1 Conventional Machine Learning Algorithms

Conventional ML algorithms are capable of processing both multi-temporal and multi-sensor remote sensing data [43]. In the 1960s, the Support Vector Machine (SVM) was introduced and later popularized with the kernel trick in the 1990s [10]. As shown by among others Gualtieri et al. [33] and Huang et al. [39], SVMs have since been widely applied in remote sensing. However, although SVMs perform well on small, high-dimensional datasets [60], they are sensitive to the selection of the kernel and can be computationally inefficient with large data [54]. SVMs' soft margin approach handles overlapping data, but struggles with extreme outliers and noisy datasets [70]. Therefore, ensemble methods such as Random Forest (RF) [11] and Extreme Gradient Boosting (XGBoost) [16] are often a preferred alternatives in more complex scenarios.

RF, introduced by [11], is well-regarded for its ability to handle high-dimensional data by creating an ensemble of decision trees. This technique enhances robustness and provides variable importance metrics, making it valuable for feature selection in high-dimensional datasets. XGBoost, developed by [16], extends this approach by iteratively correcting errors in the model, achieving high accuracy even in cases with subtle spectral differences, such as differentiating similar land cover types.

Similar to SVMs, RF and XGBoost have been widely applied in RS tasks like biomass estimation and urban land cover mapping. In research by Antunes et al. [6], above-ground biomass estimation in the Amazon rainforest demonstrated the effectiveness of both RF and XGBoost, with XGBoost showing superior performance due to its capability to handle complex relationships and residual errors, particularly when integrating SAR and optical data. Similarly, in a study by Shao et al. [17] focusing on urban impervious surface mapping, XGBoost outperformed RF by achieving higher accuracy and precision using fused data from Sentinel-1 and Landsat 8.

Nevertheless, both RF and XGBoost face computational challenges, especially with large datasets or real-time predictions [24]. RF is robust but may bias toward categorical features

and requires pre-processing for missing values. And, although XGBoost often outperforms RF, it remains sensitive to hyperparameter tuning and prone to overfitting without proper regularization.

3.1.2 Deep Learning-Based Methods

The limitations of these shallow-structured conventional machine learning tools such as SVMs, RF and XGBoost are addressed by more advanced, DL based methods [3]. DL has shown to be very promising in the field of RS due to its ability to learn hierarchical representations and process large, complex datasets [65, 7, 3]. Contrary to the conventional ML methods, DL does not rely on handcrafted features but can automatically extract them from raw data, as can be seen in Figure 3.1.

Such DL models are based on artificial neural networks, which excel at identifying patterns and extracting features from large and complex datasets. A typical DL model undergoes three key phases: training, validation, and testing. During the training phase, the model's parameters, the weights and biases, are iteratively adjusted through a process called backpropagation [43]. Backpropagation works by comparing the model's predictions to the ground truth labels using a loss function, calculating the error, and propagating this error backward to update the parameters. This process enables the model to learn patterns and relationships within the data. As training progresses, the model becomes increasingly adept at transforming input data into representations suitable for downstream tasks such as land cover classification or change detection. Depending on this task, the final layers of the model map learned representations to meaningful outputs, such as classification labels, bounding boxes, or pixel-wise segmentations.

Over the years, many DL architectures have been discovered and deployed. Table A.1, based on the overview provided by [43], summarizes the most common algorithms and their most common use cases for RS.

3.2 Machine Learning for Wetland Monitoring

ML, particularly DL methods, have shown 16–21% higher accuracy than traditional approaches in land-cover and wetland classification by capturing complex patterns in RS data [53]. However, as highlighted in a meta-analysis by Jafarzadeh et al. [42], DL is not always the preferred choice due to interpretability challenges, large data requirements, and high computational costs. Their review of 344 studies (1990–2022), mostly published in the *Remote Sensing* journal, found that ensemble learning, particularly Random Forest (RF), remains the most widely used ML approach for wetland research due to its robustness with multi-source RS data.

The meta-analysis also identified land use and land cover (LULC) classification as the most common RS application in wetlands (51%), followed by change detection (14%) and vegetation mapping (12%). Additionally, the study found that supervised learning remains the dominant approach, regardless of whether conventional ML or DL is used. However, self-supervised learning is emerging as a promising alternative to reduce reliance on large labeled datasets.

Since this study focuses on mapping vegetation types, a method was selected that effectively captures their spatial distribution and classification. As discussed in Section 3.1, semantic segmentation is well-suited for this task, as it assigns class labels at the pixel level, enabling detailed vegetation mapping, while also considering the spatial aspect. The following subsections explore ML-driven semantic segmentation for wetlands, comparing supervised and self-supervised approaches.

3.2.1 Supervised Learning for Semantic Segmentation

Supervised learning techniques for semantic segmentation rely on annotated datasets, where each pixel in an image is labeled with its corresponding class. For remote sensing applications, annotated datasets typically consist of aerial imagery, captured through passive or active sensors as explained in Chapter 2, and labeled according to specific classification schemes depending on the use case, ranging from general land cover types to highly detailed vegetation classifications.

In 2015, Long et al. [49] introduced Fully Convolutional Networks (FCNs), demonstrating how CNNs could be adapted for pixel-wise predictions by replacing fully connected layers with convolutional layers. Following this, several enhancements were introduced to improve segmentation performance. For example, ParseNet [48] addressed the limitations of FCNs by incorporating global context information. An extension on the FCN architecture was made with U-Net [66], which introduced symmetric encoder-decoder structures with skip connections to retain fine-grained spatial details, originally made for biomedical image segmentation.

U-Net, as shown in Figure 3.2, consists of an encoder that extracts compact feature representations by downsampling the input, while its decoder reconstructs detailed segmentation maps through upsampling. It's skip connections bridge corresponding encoder and decoder layers, enabling the model to retain both high-level abstract features and fine-grained spatial details. This design makes U-Net highly effective for tasks requiring precise segmentation, such as delineating features in satellite date, such as vegetation boundaries [13, 20].

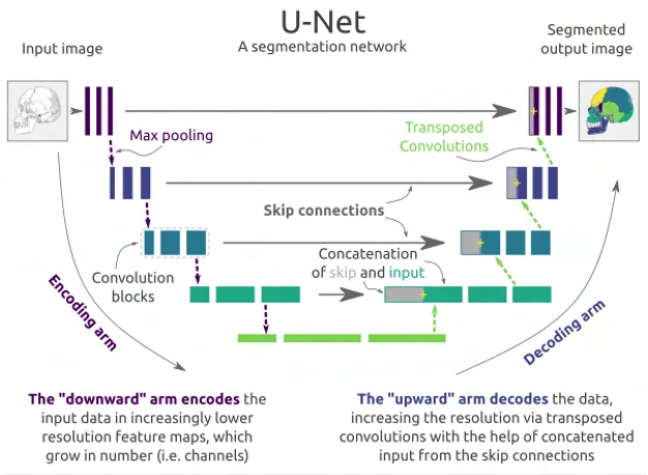


Figure 3.2: U-Net architectur for image segmentation [20].

Pech-May et al. [61] show the ability of a U-Net architecture to effectively identify flooded areas using SAR images, obtaining an IoU score of 73.02% and an accuracy of 94.31%. Waldeland et al. [78] demonstrate that U-Net can also be used for large-scale vegetation height mapping from Sentinel-1 and Sentinel-2 data, where it effectively maps forest extent, achieving a root mean square error (RMSE) of 3.5 m for Sentinel-2 and 4.6 m for Sentinel-1, highlighting the advantage of U-Net's spatial feature learning. Similarly, Fawzy et al. [25] apply U-Net for urban land cover classification using Very High-Resolution (VHR) satellite images, where it outperformed traditional methods with an overall accuracy of 87.50%. These studies illustrate U-Net's versatility in remote sensing applications, demonstrating its effectiveness in capturing fine spatial structures across diverse environments, including vegetation mapping, flood detection, and urban classification, making it a strong candidate for wetland vegetation segmentation.

However, as discussed by Dahiya et al. [20], two challenges of U-Net are image and label quality. Image quality varies due to lighting, noise, and resolution differences. Label quality is

limited as ground truth annotations are difficult to obtain and require expert knowledge. While increasing the training set size can sometimes offset minor labeling errors, Agnew et al. [5] show that incorrect labels remain highly detrimental to model performance. Annotating wetlands is particularly demanding, often requiring input from experts due to its heterogeneous vegetation and subtle class distinctions [29].

Publicly available labeled datasets can alleviate the cost of annotation but often mismatch target resolutions or class definitions, limiting direct applicability [53]. Models trained on lower-resolution imagery, for example, may fail to capture the finer details essential for accurate segmentation when applied to high-resolution scenes. Data augmentation and transfer learning are among the most common approaches for improving model performance with limited labeled data in the target resolution [67]. Self-supervised learning offers a way to leverage large volumes of unlabeled data to improve feature extraction, reducing dependency on manual annotations [38]. Similarly, few-shot and zero-shot learning approaches aim to improve classification by training models to generalize from limited examples.

3.2.2 Self-Supervised Learning for Pretraining

Self-supervised learning (SSL) is a powerful approach to address the challenge of limited annotated datasets in RS [69, 8]. By pretraining backbone networks such as CNNs or transformers on domain-specific data, SSL methods enable models to learn representations of the desired data and corresponding use case before any labels are provided to the model.

Contrastive learning [46] and (masked) autoencoding [19, 59, 83] are two examples of SSL techniques that learn features from unlabeled data. Contrastive learning trains a model to distinguish between similar and dissimilar pairs of data points, while masked auto-encoding learns to reconstruct missing parts of an input image, forcing the model to capture meaningful spatial and semantic information.

An autoencoder is a NN designed to learn efficient representations of data through unsupervised learning [18]. They are commonly used for tasks such as feature extraction, dimensionality reduction, and anomaly detection. Similar to the U-Net, an autoencoder consists of two primary components: an encoder and a decoder. The encoder processes the input data and compresses it into a latent-space representation, capturing the most important features while discarding redundant information. This compressed representation, often called the bottleneck or bridge, contains essential patterns needed to reconstruct the original input. The decoder takes this latent-space representation and reconstructs the input data as accurately as possible, performing the inverse operation of the encoder. The final output layer produces the reconstructed version of the input data, which the model attempts to make as close as possible to the original input. During training, the autoencoder minimizes the reconstruction error, which measures the difference between the original input and the reconstructed output, encouraging the model to learn meaningful and robust representations.

Chapter 4

Method

4.1 Model Architecture

Although pretrained encoder backbones (e.g., ResNet, EfficientNet) are commonly used for feature extraction in deep learning, this study does not utilize them. Standard backbones, typically trained on natural images, may fail to extract relevant features from multispectral and high-resolution remote sensing data. Instead, this research tests how domain-specific pretraining improves remote sensing for wetlands by capturing unique RS spectral and spatial characteristics that generic models might overlook. Both the feature extractor and the semantic segmentation model are (pre-)trains from scratch. To ensure seamless integration, two architectures with parallel structures were chosen. Specifically, an autoencoder was employed for SSL pretraining, as its learned encoder weights can be directly transferred to the encoder of the U-Net, which shares a similar design. This integration is illustrated in Figure I.1 and Figure I.2, and further detailed in Chapter 4.

The following sections show the architecture of these models in detail. The code is based on that of Sreenivas Bhattiprolu[1], originally developed for microscopic image analysis. Despite the differences in data domains, the U-Net's architecture is also well-suited for remote sensing tasks, as both applications involve pixel-level feature extraction and segmentation as discussed in Section 3.2.1. The final code base for this research can be found on GitHub[2].

4.1.1 Autoencoder for Pretraining

The used autoencoder is a fully convolutional architecture designed for image reconstruction. It consists of two main components: an encoder and a decoder, made up from convolutional blocks for feature extraction and reconstruction. Figure 4.1 provides a schematic overview of the autoencoder architecture.

To accommodate different image resolutions, the input data was divided into patches of varying sizes. For medium-resolution imagery, patches of 256 × 256 pixels were used with a batch size of 8. For very-high-resolution imagery, larger patches of 1024 × 1024 pixels were used, requiring a batch size of 4 due to higher memory demands.

To optimize autoencoder training performance, various learning rates and dropout probabilities were evaluated. For the learning rate, a fixed rate of 0.001 yielded the best results compared to 0.0001, as shown in Figure B.1 and Table B.1. As for the dropout probability, values of 0%, 15%, and 25% were tested, with a dropout rate of 15% yielding the best results, as illustrated in Figure B.2 and Table B.2.

[1] `https://github.com/bnsreenu/python_for_microscopists/tree/master/235-236_pre-training_unet_using_autoencoders`

[2] `https://github.com/Evameijling/WetlandSemanticSegmentation.git`

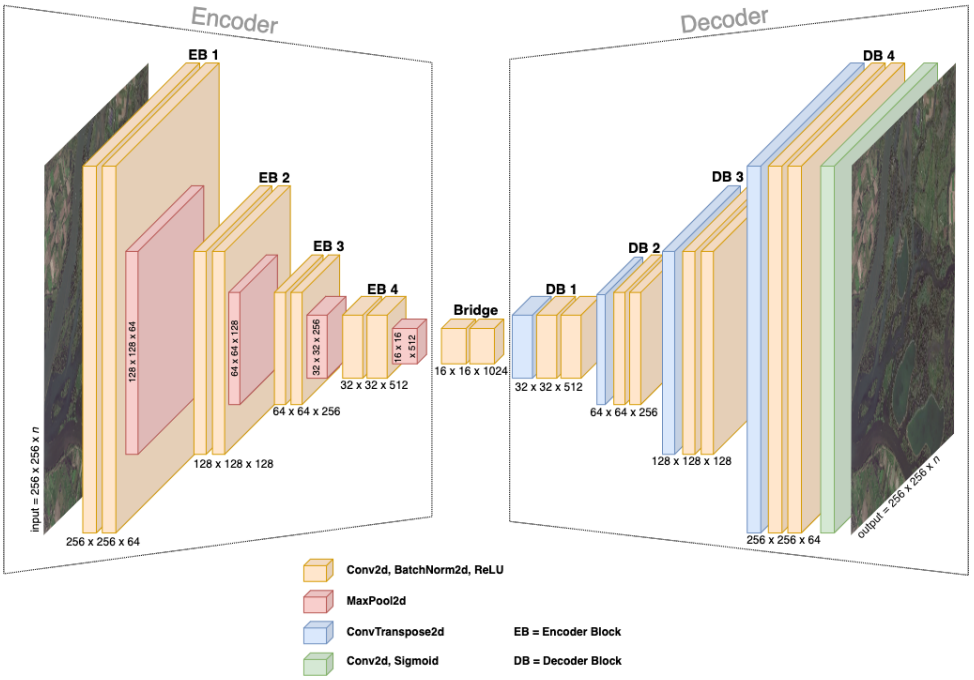


Figure 4.1: Schematic representation of the autoencoder architecture. The encoder reduces the spatial dimentions by a factor of 2 at each block, compressing the input from 256x256 pixels to 16x16 pixels in the bridge. At the same time, the number of channels increases by a factor of 2, from 64 in the first encoder block, to 512 in the bridge. The decoder restores the spatial dimensions by a factor of 2 at each block, reconstructing the output from 16x16 to 256x256, while the number of channels decreases by a factor of 2, from 512 to 64.

Encoder

As explained in Chapter 3, the autoencoder's encoder extracts hierarchical features from the input images and reduces their spatial dimensions. This encoder consists of four convolutional blocks, each followed by a max-pooling layer for down-sampling. The components of each encoder block are as follows:

- **Convolutional Block:** Each block contains two convolutional layers with a kernel size of 3×3 followed by batch normalization. A ReLU activation function is applied after each convolution. After the second convolution, a dropout layer with a probability of 0.25 is added to prevent overfitting.
- **Max-Pooling Layer:** After each convolutional block, a 2×2 max-pooling operation reduces the spatial dimensions by half.

After the final encoder block, the bridge layer captures the most compressed and abstract representation of the input image, preserving the most essential features before passing them to the decoder.

Decoder

The decoder reconstructs the input image from the encoded features. It utilizes the same convolutional blocks as in the encoder for consistency in feature extraction. The decoder blocks consist of the following components:

- **Transposed Convolutional Layers:** Each decoder block begins with a 2×2 transposed convolution, which up-samples the feature maps by doubling their spatial dimensions.

- **Convolutional Blocks:** After up-sampling, each decoder block contains two convolutional layers with a kernel size of 3×3, followed by batch normalization. A ReLU activation function is applied after each convolution, and a dropout layer with a probability of 0.25 is included to reduce overfitting.

The final layer is a 3×3 convolution followed by a sigmoid activation function to produce the reconstructed output with values in the range [0, 1].

4.1.2 U-Net for Semantic Segmentation

U-Net Architecture

The U-Net architecture used for supervised semantic segmentation has the same encoder design as the autoencoder discussed in Section 4.1.1. This shared encoder design allows for easy weight transfer between the two models, enabling the U-Net to leverage pretrained features for segmentation tasks. The U-Net differs from the autoencoder by incorporating skip connections, which help retain fine-grained spatial details necessary for accurate segmentation. Figure 4.2 provides a schematic overview of the U-Net architecture.

Similar to the optimization of the autoencoder, various learning rates and dropout probabilities were evaluated for the training of the U-Net as well. For the learning rate, cosine annealing yielded the best results compared to fixed rates of 0.001 and 0.0001, as well as warm cosine annealing, as shown in Figure B.3 and Table B.3. As for the dropout probability, values of 0%, 15%, 25%, 35%, and 45% were tested, with a dropout rate of 15% yielding the best results, as illustrated in Figure B.4 and Table B.4.

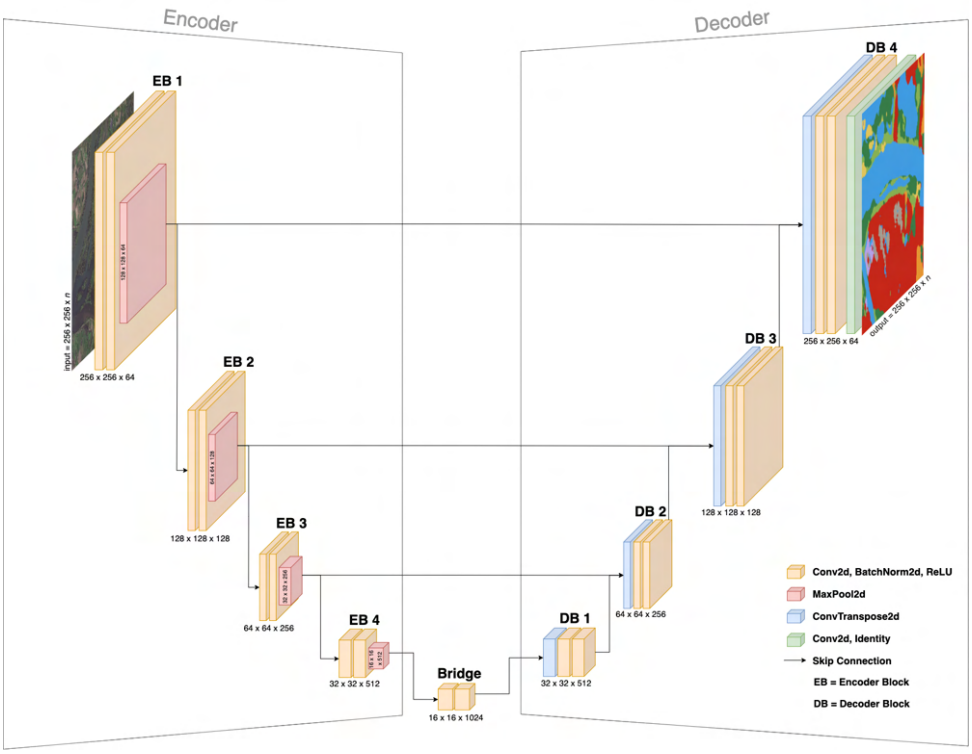


Figure 4.2: Schematic representation of the U-Net architecture. The encoder reduces spatial dimensions by a factor of 2 at each block while increasing the number of channels, compressing the input from 256×256 pixels to 16×16 pixels in the bridge. Skip connections link corresponding encoder and decoder layers, helping to retain spatial information. The decoder reconstructs the segmentation map, restoring the spatial dimensions back to 256×256 while progressively reducing the number of channels.

Encoder

The encoder in the U-Net is identical to that used in the autoencoder, as described earlier. It extracts hierarchical features fromthe input satellite image and reduces spatial dimensions through a sequence of convolutional blocks and max-pooling operations. The components of each encoder block are as follows:

- **Convolutional Block:** Each block contains two convolutional layers with a kernel size of 3×3, followed by batch normalization. A ReLU activation function is applied after each convolution, and a dropout layer with a probability of 0.25 is added to mitigate overfitting.
- **Max-Pooling Layer:** A 2×2 max-pooling operation follows each convolutional block, reducing spatial dimensions by half.

The encoder produces feature maps at multiple resolutions, which are saved and later used in the skip connections to help reconstruction. After the final encoder block, a bridge layer captures the most abstract and high-level features of the input before the decoder begins reconstructing the segmentation map.

Decoder

The decoder reconstructs the segmentation map (the labeled ground truth mask) from the encoded features of the satellite image. It differs from the autoencoder decoder by incorporating skip connections, which concatenate feature maps from the encoder to the up-sampled feature maps in the decoder. This helps the model retain spatial context and fine-grained details. The decoder consists of:

- **Skip Connections:** Feature maps from the encoder blocks at corresponding spatial resolutions are concatenated with the up-sampled feature maps in the decoder. These connections enhance spatial precision and provide access to both low-level and high-level features. In this implementation, skip connections transfer all feature channels from the encoder to the decoder (a 1:1 ratio).
- **Transposed Convolutional Layers:** Each decoder block begins with a 2×2 transposed convolution to up-sample the feature maps by doubling their spatial dimensions.
- **Convolutional Blocks:** After up-sampling and concatenation, each decoder block contains two convolutional layers. The convolutional layers use a kernel size of 3×3 with batch normalization and ReLU activation, similar to the encoder. The decoder also includes a dropout layer with a probability of 0.25 to reduce overfitting.

The final decoder block is followed by a 1×1 convolutional layer, which maps the feature maps to the desired number of output classes. This allows the U-Net to produce a segmentation map where each pixel is assigned to one of the class from the classification scheme it was trained on.

4.2 Learning Objective

For both models, specific loss functions were applied to optimize the learning process. This section outlines the loss functions used and the metrics employed to evaluate the performance of each model.

4.2.1 Autoencoder Loss

The autoencoder loss function is a combination of three components: Huber Loss, Structural Similarity Index Measure (SSIM) Loss, and Edge Loss. These losses are weighted by the coefficients α, β, and γ, respectively, to balance reconstruction accuracy and feature preservation.

- **Huber Loss [40]:** The Huber Loss, $\mathcal{L}_{\text{Huber}}$, is used to handle outliers by combining the properties of the Mean Squared Error (MSE) and Mean Absolute Error (MAE). This loss helps in robust reconstruction by minimizing the impact of outliers. It is defined as:

$$\mathcal{L}_{\text{Huber}}(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{if } |y - \hat{y}| \leq \delta, \\ \delta|y - \hat{y}| - \frac{1}{2}\delta^2 & \text{otherwise,} \end{cases}$$

  where y is the ground truth, $\hat{y}$ is the predicted output, and δ is a threshold parameter.

- **SSIM Loss [80]:** The Structural Similarity Index (SSIM) Loss, $\mathcal{L}_{\text{SSIM}}$, focuses on preserving structural information in the image by comparing luminance, contrast, and structure between the original and reconstructed image. It is defined as:

$$\mathcal{L}_{\text{SSIM}}(y, \hat{y}) = 1 - \text{SSIM}(y, \hat{y}),$$

  where the SSIM between the ground truth y and the predicted output $\hat{y}$ is given by:

$$\text{SSIM}(y, \hat{y}) = \frac{(2\mu_y\mu_{\hat{y}} + \epsilon_1)(2\sigma_{y\hat{y}} + \epsilon_2)}{(\mu_y^2 + \mu_{\hat{y}}^2 + \epsilon_1)(\sigma_y^2 + \sigma_{\hat{y}}^2 + \epsilon_2)}.$$

  Here, μ_y and $\mu_{\hat{y}}$ are the means, σ_y^2 and $\sigma_{\hat{y}}^2$ are the variances, $\sigma_{y\hat{y}}$ is the covariance, and ϵ_1 and ϵ_2 are stabilizing constants.

- **Edge Loss:** The Edge Loss, $\mathcal{L}_{\text{Edge}}$, enforces the preservation of edges by comparing the gradients of the ground truth and predicted images. It is defined as:

$$\mathcal{L}_{\text{Edge}}(y, \hat{y}) = \|\nabla_x y - \nabla_x \hat{y}\|_1 + \|\nabla_y y - \nabla_y \hat{y}\|_1,$$

  where ∇_x and ∇_y represent the gradients along the x- and y-directions, respectively. This loss helps in maintaining sharp edges in the reconstructed images.

The total loss function, $\mathcal{L}_{\text{total}}$, is a weighted sum of these three components:

$$\mathcal{L}_{\text{total}} = \alpha\mathcal{L}_{\text{Huber}} + \beta\mathcal{L}_{\text{SSIM}} + \gamma\mathcal{L}_{\text{Edge}},$$

where α, β, and γ are the weights assigned to each loss component. In this work, we use $\alpha = 0.5$, $\beta = 0.4$, and $\gamma = 0.1$ to balance the contributions of each component. With Huber Loss handling noise, SSIM Loss preserving structural details, and Edge Loss ensuring sharp transitions, this combination of losses is well-suited for accurate satellite imagery reconstruction.

4.2.2 U-Net Loss

The U-Net model employs Dice Loss, $\mathcal{L}_{\text{Dice}}$, which is particularly effective in addressing class imbalance. This ensures that less frequent features, such as small vegetation types or bare ground, are well represented alongside majority classes like water and built areas. Additionally, its emphasis on spatial overlap and boundary alignment makes it well-suited for capturing the complex and heterogeneous structures found in satellite imagery.

Dice Loss is derived from the Dice coefficient [21], a measure used to evaluate the similarity between two sets. The Dice coefficient between the predicted segmentation map $\hat{y}$ and the ground truth y is defined as:

$$\text{Dice}(\hat{y}, y) = \frac{2\sum_{i=1}^{N} \hat{y}_i y_i + \epsilon}{\sum_{i=1}^{N} \hat{y}_i + \sum_{i=1}^{N} y_i + \epsilon},$$

where:

- N is the total number of pixels in the image.
- $\hat{y}_i$ is the predicted probability for the i-th pixel.
- y_i is the ground truth label for the i-th pixel (1 for foreground, 0 for background).
- ϵ is a small constant (e.g., 10^{-6}) to avoid division by zero.

The **Dice Loss** is then defined as:

$$\mathcal{L}_{\text{Dice}} = 1 - \text{Dice}(\hat{y}, y).$$

For the highly unbalanced labeled data, as discussed in Section 5.3, a Weighted Dice Loss is applied to mitigate class imbalance by assigning lower weights to dominant classes, defined as:

$$\mathcal{L}_{\text{WeightedDice}} = 1 - \frac{1}{C}\sum_{c=1}^{C} w_c \frac{2\sum_{i=1}^{N} \hat{y}_{i,c} y_{i,c} + \epsilon}{\sum_{i=1}^{N} (\hat{y}_{i,c} + y_{i,c}) + \epsilon},$$

where w_c represents the weight assigned to class c.

4.2.3 Performance Metrics

The following metrics are used to evaluate the performance of the models:

Metrics for the Autoencoder

- **PSNR (Peak Signal-to-Noise Ratio):** Quantifies the quality of the reconstructed images.

  $$\text{PSNR} = 20 \cdot \log_{10}\left(\frac{\text{MAX}_I}{\sqrt{\text{MSE}}}\right)$$

  where MAX_I represents the maximum possible pixel value. Since the images are normalized in the range $[0, 1]$, $\text{MAX}_I = 1$.

- **SSIM (Structural Similarity Index):** Assesses the similarity between the reconstructed image and the original image. The formula is the same as shown in Subsection 4.2.1, SSIM Loss.

- **Accuracy (Bandwise):** Measures the proportion of reconstructed pixel values that fall within a predefined tolerance (τ) for each spectral band.

  $$\text{Accuracy} = \frac{1}{B}\sum_{b=1}^{B} \frac{\sum_{i=1}^{N} \mathbb{I}(|\hat{y}_{i,b} - y_{i,b}| \leq \tau)}{N}$$

  where B is the number of spectral bands, N is the total number of pixels per band, $y_{i,b}$ and $\hat{y}_{i,b}$ are the ground truth and reconstructed values for pixel i in band b, respectively,

and $\mathbb{I}(\cdot)$ is an indicator function that equals 1 if the absolute difference between the reconstructed and ground truth values is within the tolerance threshold ($\tau = 0.1$), and 0 otherwise.

Metrics for the U-Net

- **Standard Metrics:** These commonly used metrics evaluate the overall classification performance of the model.
  - **Accuracy:** Measures the proportion of correctly classified pixels.

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i)}{N}$$

    where N is the total number of pixels, and $\mathbb{I}(\cdot)$ is an indicator function that equals 1 if the predicted label $\hat{y}_i$ matches the ground truth y_i, and 0 otherwise.
  - **Precision (Macro-Averaged):** Measures the proportion of correctly predicted pixels for each class and averages across all classes.

$$\text{Precision} = \frac{1}{C} \sum_{c=1}^{C} \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c}$$

    where C is the number of classes.
  - **Recall (Macro-Averaged):** Calculates the proportion of correctly predicted pixels among all actual positive pixels per class, then averages across classes.

$$\text{Recall} = \frac{1}{C} \sum_{c=1}^{C} \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c}$$

  - **F1 Score (Macro-Averaged):** Combines precision and recall per class and averages across classes.

$$\text{F1} = \frac{1}{C} \sum_{c=1}^{C} 2 \cdot \frac{\text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}$$

    Note: The F1 score is mathematically equivalent to the Dice coefficient in the context of segmentation. Since Dice is the standard metric for evaluating segmentation quality, only the Dice coefficient is reported in the results to avoid redundancy.

- **Dice coefficient [21]:** Measures the overlap between the predicted segmentation and ground truth, commonly used in segmentation tasks. The Dice coefficient, $\text{Dice}(\hat{y}, y)$, is defined using the same formula presented in Subsection 4.2.2.

- **IoU (Intersection over Union)**, also known as the Jaccard Index [41]: Measures the ratio of the intersection to the union of the predicted and ground truth segmentation masks. It is defined as:

$$\text{IoU} = \frac{\sum_{i=1}^{N} \hat{y}_i y_i}{\sum_{i=1}^{N} (\hat{y}_i + y_i - \hat{y}_i y_i)}$$

  where N is the total number of pixels, and $\hat{y}_i$ and y_i represent the predicted and ground truth segmentation values for pixel i, respectively.

These metrics are tailored to evaluate the specific objectives of the autoencoder and U-Net, ensuring comprehensive assessments of reconstruction quality and segmentation performance.

Chapter 5

Experimental Setup

The U-Net model described in Section 3.2.1 is used for semantic segmentation of land use in satellite images. This chapter explores two methods to improve it. Specifically, the following two factors are investigated:

- **Dependency on labeled data:** As discussed in Chapter 3, the performance of semantic segmentation models heavily depends on the labeled training data. Oftentimes, the annotated data is scarce or misaligned with the data intended to be used during inference, such as the desired classification scheme, resolution, and geographical coverage. This experiment addresses this limitation by exploring the use of widely available unlabeled RS data through pretraining.
- **Dependency on resolution:** In this experiment, the relationship between the resolution of satellite imagery and the performance of semantic segmentation models is explored. Higher-resolution shows more detailed information, which hypothetically could improve model accuracy and reduce the dependency on labeled data. This experiment investigates how using a different image resolutions (medium resolution vs very high resolution) impacts the performance of the U-Net model under both from-scratch and pretrained conditions.

In addition, this chapter also describes the datasets used in the baseline and experimental setups, detailing the retrieval and pre-processing steps required to prepare the data for analysis.

All experiments were conducted on a high-performance computing cluster[1] equipped with NVIDIA A100 and V100 GPUs and AMD EPYC CPUs.

5.1 Data

The optical satellite data used in this research was obtained from open-source platforms, including Google Earth Engine[2] and Satellietdataportaal[3]. The specific Sentinel-2 dataset used in this research is shared here[4]. Due to licensing restrictions, the Very High Resolution satellite (VHR) imagery retrieved from Satellietdataportaal cannot be shared. Labels were derived through a combination of Google Earth Engine and manual annotation using tools such as Blackshark[5] and Roboflow[6]. Eventually, only the labels obtained using Roboflow were used,

[1] https://www.surf.nl/en/services/snellius-the-national-supercomputer
[2] https://earthengine.google.com
[3] https://viewer.satellietdataportaal.nl/
[4] https://drive.google.com/drive/folders/1gETPmb8uniyRd0q6pjHlkyOxX0KEOuOA?usp=share_link
[5] https://blackshark.ai
[6] https://roboflow.com

as these were all manually identified and therefore provided a more accurate representation of the ground truth compared to the AI-generated labels from Blackshark, as will be discussed in Section 5.1.2. In initial experiments, the pretrained U-Net was tested on a land-cover dataset using Gaofen-2 (GF-2) satellite images [82], yielding promising results, particularly with performance gains from pretraining. However, the dataset was not suitable for this research due to differences in geographical coverage, resolution, and classification scheme, which did not align with the Biesbosch area. The results of these preliminary experiments are presented in Appendix C.

5.1.1 Sentinel-2 imagery with Dynamic World labels

Data Retrieval

Sentinel-2 imagery used in this research was obtained via the Google Earth Engine API, sourced from the COPERNICUS/S2_SR_HARMONIZED collection, which provides Harmonized Level-2A Sentinel-2 data, as explained in Chapter 2. These images have a spatial resolution of 10 meters per pixel and include 26 spectral bands, listed in Appendix D.1. Sentinel-2 satellites have a revisit time of 5 days, and over the Biesbosch region, passes occur consistently between 10:00 and 11:00 in the morning local time, ensuring uniform illumination and minimal shadow interference.

For this research, similar wetland areas in the Netherlands were selected, including the Biesbosch, to acquire specific domain knowledge and ensure good coverage of the desired classes. All selected areas are part of the Natura 2000 network[7] and are considered wetlands according to the Nationaal Georegister[8], based on the Ramsar Convention on Wetlands (updated in 2024)[9]. An exception is the Gelderse Poort, which, while not Ramsar-designated, was included due to its high comparability with the Biesbosch as a floodplain shaped by the confluence of the Rhine and Waal rivers. The locations of these areas within the Netherlands are displayed in Figure D.1. Figure D.2 presents Sentinel-2 images of each area captured during various months of the year, illustrating the variability in both the regions and their vegetation across different seasons.

To ensure consistency in data quality, only Sentinel-2 scenes with less than 5% cloud cover were selected, covering the period from January 1, 2017, to November 1, 2024. To account for the multiple image tiles covering each wetland area, the Copernicus Browser[10] was utilized to manually identify and select the most relevant tile IDs for each area and date, ensuring non-duplicate tiles per region, per date.

Land cover labels for each Sentinel-2 image were sourced from the Dynamic World dataset (GOOGLE/DYNAMICWORLD/V1), which provides near real-time, 10 m resolution global land cover classifications using deep learning [12]. The dataset uses Sentinel-2 spectral data and a pretrained convolutional neural network trained on a combination of expert and non-expert annotations to assign probabilities for 9 land cover types to each pixel. The Dynamic World classification achieves an overall agreement of 73.8% with expert-labeled validation data, performing best on categories like water, trees, and built areas while having more difficulty with classes such as grass and shrub & scrub. The classes in the Dynamic World classification scheme are extensively detailed in Table D.3. For each pixel, the class with the highest probability (ranging from pixel values 0 to 8) is selected to create classification masks.

[7] https://www.natura2000.nl/gebieden
[8] https://nationaalgeoregister.nl/
[9] https://www.ramsar.org
[10] https://browser.dataspace.copernicus.eu/

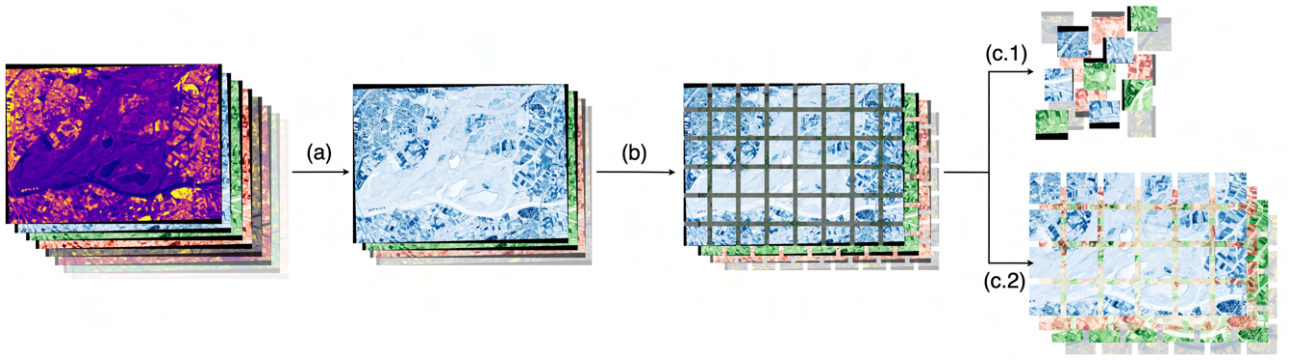


Figure 5.1: Overview of the pre-processing steps applied to both Sentinel-2 and Pléiades NEO datasets. (a) Selection of relevant spectral bands tailored to the classification task. (b) Division of images into patches of 256 × 256 pixels for consistency and usability in training. (c.1) Exclusion of patches with dimensions smaller than 256 × 256 or containing excessive black pixels (> 10% for Sentinel-2 and > 30% for Pléiades NEO). (c.2) Retention of patches that meet size and quality requirements to ensure consistency in high-resolution data.

Data Pre-processing

Before the data could be fed to the model, it underwent several pre-processing steps as shown in Figure 5.1. This structured approach to pre-processing ensured diverse and representative data splits, enhancing the model's ability to generalize across different wetland environments. First, band selection was performed. From the 26 available bands in the Sentinel-2 imagery, 9 bands were selected based on their relevance to wetland classification tasks and the need to balance data size for computational efficiency. Comprehensive meta-analysis [42] on remote sensing for wetland classification shows that in addition tot the RGB bands and SWIR-bands, the red-edge and near-infrared bands are the most effective optical bands for wetland delineation [52]. Hence, the selected bands for the Sentinel-2 dataset are B2 (blue), B3 (green), B4 (red), B5-B7 (Red Edge 1-3), B8 (NIR), and B11-B12 (SWIR 1 and 2). These bands are known for their sensitivity to vegetation, water bodies, and soil characteristics, which are essential for distinguishing wetland environments. A full list of the 26 available bands is provided in Table D.1. One exception of when not all these bands are used is in Experiment 2, as described in Chapter 5. To enable a fair comparison between high- and low-resolution data, the same number of bands (4) are used for both: the RGB and NIR bands. This is because, for the high-resolution data from Pléides NEO, additional bands were not available due to storage constraints in the experimental setup.

Second, the images were divided into patches of 256 × 256 pixels to match the input size required by the model, resulting in input dimensions of 256 × 256 × n, where n is the number of selected bands. Patches containing more than 10% black pixels or those not meeting the 256 × 256 size requirement were excluded from the dataset.

Lastly, the data was divided into three sets. The Biesbosch region was used as the test set to evaluate the model's performance on unseen data. Due to its similarity to the Biesbosch, Lauwersmeer was used as the validation set to tune model parameters. The remaining regions - Gelderse Poort, Oostvaardersplassen, Loosdrechtse Plassen, and Land van Saeftinghe - were used as the training set. Because the splits were based on geographical regions, which greatly differ in size, the train/validation/test split was not balanced. A larger number of patches came from retrieved the Biesbosch imagery, making the test set more extensive. The complete training set contained 1,701 images, the validation set 948 images, and the test set 1,140 images.

In Experiment 2, only the Biesbosch region was used for training, validation, and testing to ensure a fair comparison with the high-resolution data. As described in Chapter 5, this

experiment used 12 training images, 2 validation images, and 2 test images. Here, the division into train, validation, and test sets was randomly selected.

5.1.2 Pléiades NEO imagery with Manual Labels

Data Retrieval

Pléiades NEO imagery[11] was accessed via FTP from the Satellietdataportaal platform. This open-access platform provides high-resolution optical satellite imagery of the Netherlands and the Caribbean Netherlands and is only accessible to users with a Dutch IP address. It offering spatial resolutions of up to 0.3m, significantly higher than Sentinel-2. Pléiades NEO provides 6 spectral bands: Red, Green, Blue (RGB), Near Infrared (NIR), Red Edge, and Deep Blue. Pléiades NEO has a revisit time of approximately 6 weeks.

Due to the high storage requirements of high-resolution imagery and the primary goal of Experiment 2 to compare high and low-resolution data rather than evaluate the U-Net's performance, the focus was limited to the Biesbosch area. From this area, all available data was retrieved, spanning between the beginning of 2023 to the end of 2024.

To ensure sufficient data availability while maintaining quality, a cloud cover limit of 30% was applied during pre-processing. This threshold balanced data availability and minimized obstruction from clouds, ensuring clear visibility of the target features.

Land Cover Labels for this dataset were manually created using the Beeldenboek bij Rijkswaterstaat[12] as a reference, with the annotation process carried out in Roboflow. To explore more efficient labeling methods, semi-automated tools like Blackshark.ai were tested. This tool generates ground truth labels based on a few user-provided examples. However, as shown in Figures E.3 and E.4, its application to multiple wetland areas often resulted in incorrect labels and significant inconsistency across different dates. These discrepancies confused the model and negatively impacted performance.

Given these limitations, manual annotation using Roboflow was ultimately chosen to ensure label accuracy and consistency. While more time-intensive, this approach provided reliable ground truth data for subsequent analysis, leading to a more robust model performance compared to training on Blackshark-generated labels.

Data Pre-processing

The pre-processing of the Pléiades Neo imagery and manual labels is similar to that of Sentinel-2, as discussed in Section 5.1.1 and shown in Figure 5.1. Pre-processing includes band selection, aligning aerial and satellite images, cropping to remove pixels from the imagery and/or labels with invalid pixel values, and splitting into training, validation, and test sets. Due to storage limitations, only the first four spectral bands of the six available bands were used: Red, Green, Blue, and Near Infrared. Unlike the 256×256 pixel patches of the Sentinel-2 data, the Pléiades Neo data was tiled into 1024×1024 pixel patches to account for its higher resolution and to capture more spatial information within each patch. Patches containing more than 5% invalid pixels or those not meeting the required size were excluded from the dataset. The data was randomly split into 1,027 training images, 205 validation images, and 136 test images. For training the autoencoder used to pretrain the U-Net for the Pléiades NEO images, a larger dataset of high resolution images from Satellietdataportaal was utilized, covering the same wetland areas as those included in the Sentinel-2 dataset. The autoencoder was trained for nine epochs on

[11] https://earth.esa.int/eogateway/missions/pleiades-neo

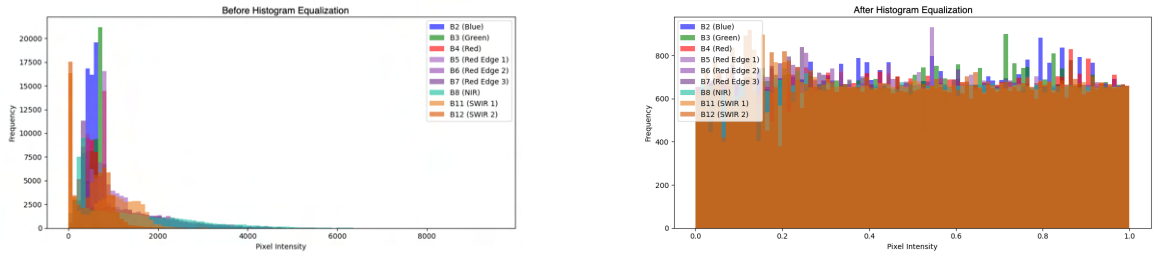
[12] https://open.rijkswaterstaat.nl/overige-publicaties/2020/beeldenboek-vegetatiebeheer-grote/

64,485 training patches of 256 × 256 pixels using a batch size of 8. The validation set consisted of 11,761 patches, and the test set comprised 53,118 patches. A similar train/validation/test split was applied as with the Sentinel-2 dataset, with Lauwersmeer designated as the validation set and Biesbosch as the test set.

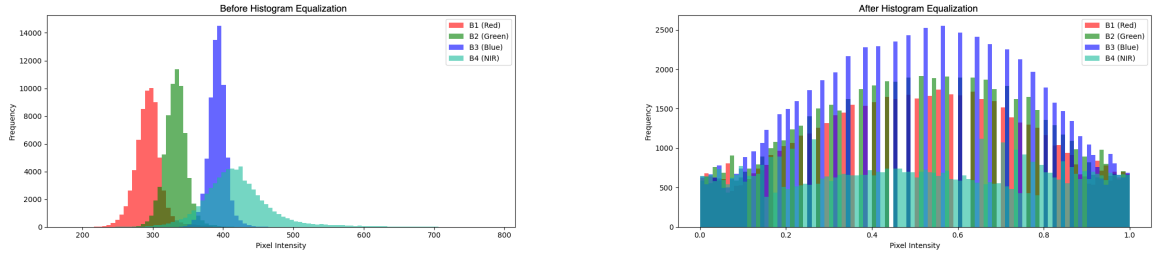
5.2 Experiment 1: Impact of Pretraining (Label Dependency)

The first experiment examines the impact of pretraining a U-Net model using an autoencoder, where the encoder weights from the autoencoder are transferred to initialize the U-Net. The performance of U-Net models trained from scratch is then compared to those initialized with pretrained weights across different dataset subset sizes, as detailed in Table G.1. To assess whether pretraining provides greater performance improvements when applied to VHR data, the U-Net's performance is evaluated on two datasets: medium-resolution imagery from Sentinel-2 and VHR imagery from Pléiades NEO, as discussed in Section 5.1.

For both datasets, histogram equalization is applied to normalize intensity distributions, improving contrast and feature clarity. This pre-processing step enhances the model's ability to distinguish features by emphasizing subtle variations in pixel values and reducing the impact of spectral inconsistencies typical in satellite imagery. Figure 5.2 shows the histograms of both resolutions before and after equalization, highlighting the improved contrast and redistributed intensity values.



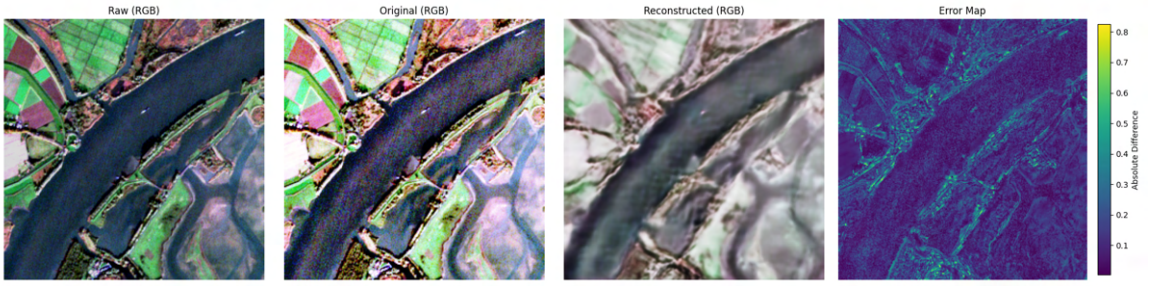
(a) Medium-resolution pixel distribution (Sentinel-2).



(b) VHR pixel distribution (Pléiades NEO)

Figure 5.2: Histograms of pixel intensity distributions before (left) and after (right) histogram equalization for Sentinel-2 medium-resolution (a) and Pléiades NEO very high-resolution (VHR) data (b). Both datasets have 16-bit depth, with Sentinel-2 (9 bands) showing a skewed distribution and Pléiades NEO (4 bands, higher spatial resolution) exhibiting distinct peaks. Histogram equalization successfully redistributes the pixel intensities, improving contrast and normalizing brightness variations across bands in both datasets.

A detailed reconstruction of all nine Sentinel-2 bands can be seen in Figure G.1, where each band is well reconstructed, consistently preserving key spectral information across all bands. Once the raw input patches have undergone histogram equalization, they are fed to the autoencoder. Applying the train/validation/test split for both datasets as described in Section 5.1, the autoencoder learns features during training and reconstructs the input image in as shown in Figure 5.3. The training of the autoencoder for the medium-resolution imagery converged at 200 epochs, while for the VHR imagery, convergence was reached at just 9 epochs.



(a) Reconstruction result for medium-resolution Sentinel-2 data.

(b) Reconstruction result for VHR Pléiades NEO.

Figure 5.3: Reconstruction results from the autoencoder for medium-resolution Sentinel-2 data (a) and very high-resolution (VHR) data (b). The first column shows the raw RGB input with contrast stretching for visibility, followed by the histogram-equalized original RGB image. The third column presents the reconstructed image from the autoencoder, while the last column displays the error map, where blue indicates minimal pixel differences and yellow highlights larger discrepancies.

Table 5.1 summarizes the performance of the autoencoder on the two datasets: medium-resolution (Sentinel-2) and very high-resolution (Pléiades NEO). Despite the visually convincing quality of the VHR reconstruction shown in Figure 5.3, its numerical performance appears significantly lower. This discrepancy will be further analyzed in Chapter 7.

Resolution	**Accuracy ↑**	**PSNR ↑**	**SSIM ↑**	**Huber ↓ Loss**	**SSIM ↓ Loss**	**Edge ↓ Loss**	**Mixed ↓ Loss**
Sentinel-2	0.6076	17.8110	0.4964	0.0083	0.2518	0.1060	0.1009
Pléiades NEO	0.3667	14.5375	0.4627	0.0111	0.2686	0.2342	0.1610

Table 5.1: Performance metrics of the autoencoder for medium-resolution (Sentinel-2) and very high-resolution (Pléiades NEO) imagery on their corresponding test sets.

5.3 Experiment 2: Impact of Resolution

The second experiment examines how varying image resolutions affect the performance of the U-Net model for vegetation classification in the Biesbosch. To achieve this, datasets from two sources were utilized: VHR images (0.3m × 0.3m) from Pléiades NEO and medium-resolution images (10m × 10m) from Sentinel-2. Data selection focused on four temporally diverse dates, two in April and two in September, to capture seasonal variation. As shown in Figures E.1 and E.2, each of the four satellite images was divided into a 4 × 4 grid of patches, from which four distinct patches were selected per image. This approach ensured that the final 16 selected patches collectively covered the entire study area while capturing data from four different points in time, balancing spatial coverage and labeling efficiency. To ensure a fair comparison, the corresponding dates for the Pléiades NEO and Sentinel-2 datasets were selected to be temporally close, as shown in Table E.1. These 16 tiles were then labeled using Roboflow, as shown in Figure E.5. These manually annotated labels were transferred from the high resolution imagery to the corresponding low resolution imagery, and down-sampled, as shown in Figure 5.4. By following these steps, the experiment aimed to assess the relationship between image resolution and the ability to classify detailed vegetation features, using consistent pre-processing of image-label pairs and training steps for both resolutions to ensure a controlled comparison.

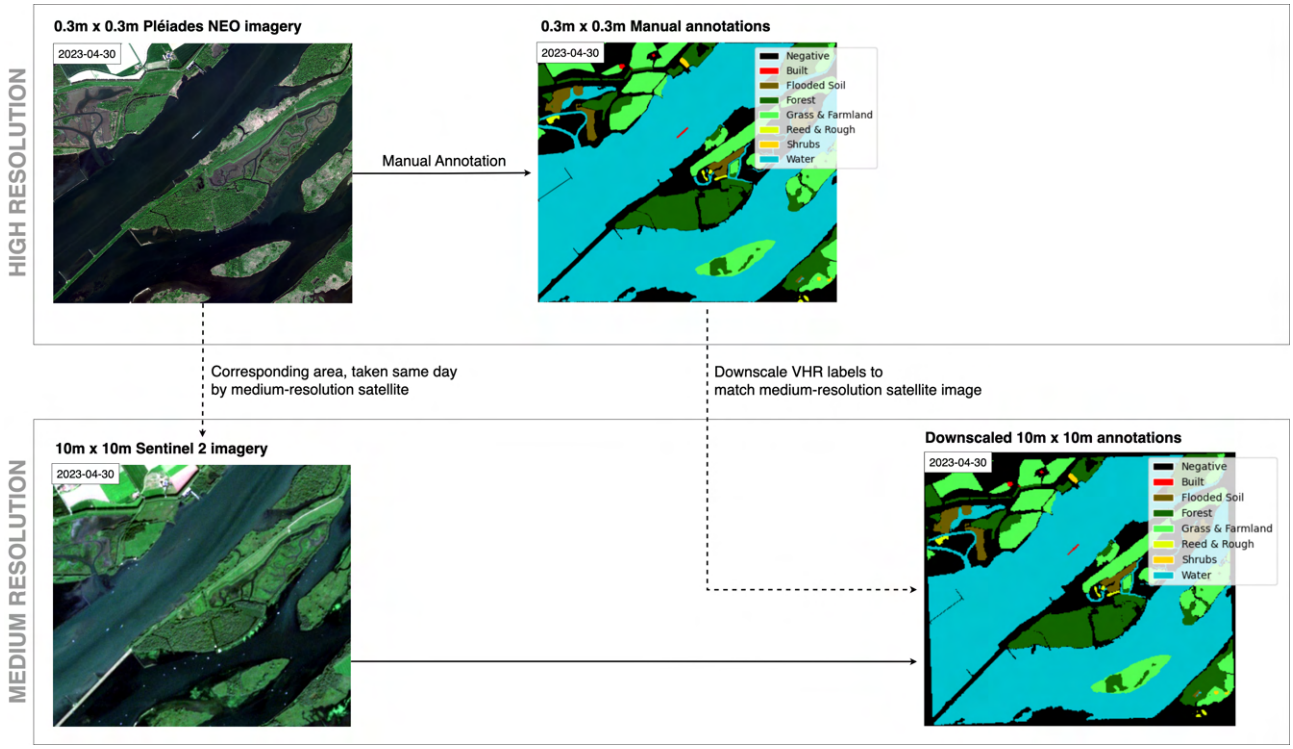


Figure 5.4: VHR imagery from Pléiades NEO (0.3m x 0.3m) was manually annotated to generate high-resolution land cover labels. A corresponding medium-resolution Sentinel-2 image (10m x 10m) from the same date was found to ensure temporal consistency. The high-resolution annotations were then downscaled to match the lower resolution of the Sentinel-2 image. This process was repeated to generate multiple VHR and medium-resolution image-label pairs.

Since this experiment focuses only on the Biesbosch area, the number of images in the train, validation, and test sets for both resolutions is relatively low compared to the baseline and Experiment A. Table 5.2 provides the exact numbers of the splits. The VHR imagery was divided into patches of 1024 × 1024 pixels to cover a larger geographical area. However, due to the significant resolution difference between Sentinel-2 and Pléiades NEO, the VHR dataset yielded substantially more patches. Specifically, within a single Sentinel-2 patch (10m

resolution), approximately 111 Pléiades NEO patches (0.3m resolution) fit, highlighting the scale disparity between the two datasets.

	Pleides-NEO	Sentinel-2
Training patches	1027	12
Validation patches	205	2
Testing patches	138	2

Table 5.2: Number of input patches generated from high- and low-resolution datasets. The Pléiades NEO imagery (0.3mx0.3m) generated more patches due to its higher spatial resolution.

Figure 5.5 shows the significant class imbalance across the dataset splits for both resolutions. A more detailed overview of the exact percentages can be found in Tables E.2 and E.3. This imbalance originates from the manual annotation where Water and Grass & Farmland were the most easily distinguishable classes and were therefore labeled more frequently. Additionally, these classes are naturally more prevalent in the Biesbosch region, which further contributes to their overrepresentation in the dataset. To mitigate the effects of this imbalance, class weighting was applied in the Dice loss function where the weight for Grass & Farmland was reduced by a factor of 10 relative to other classes to balance their influence during model training.

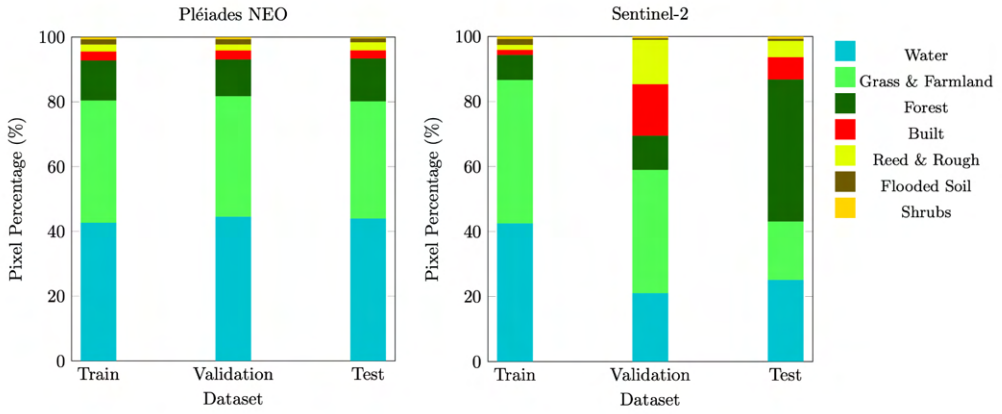


Figure 5.5: Class distribution in the training, validation, and test datasets for both Pléiades NEO (left) and Sentinel-2 (right). The dominant land cover classes, Water and Grass & Farmland, occupy the largest pixel proportions across all splits, reflecting their prevalence in the Biesbosch region. This class imbalance is addressed by incorporating weighted loss functions during training.

To prevent background pixels from causing a bias in optimization and metric computation, they are excluded from both loss computation and accuracy calculations. This is done by applying a binary mask that disregards these pixels. As a result, only annotated land cover classes are considered during training and evaluation.

Furthermore, weighted accuracy was used to address class imbalance, adjusting each class's contribution based on its frequency in the dataset. Instead of a simple pixel-wise accuracy, underrepresented classes received higher weights to ensure fair influence on overall performance. Conversely, dominant classes, such as Water and Grass & Farmland, were weighted lower in the high-resolution dataset, where they were more prevalent.

Chapter 6

Results

This chapter presents the results of the U-Net trained from scratch, alongside findings from two experiments: one evaluating the dependency on labels through pretraining, and the other analyzing the impact of resolution on model performance.

6.1 Baseline Performance of U-Net trained from Scratch

Table 6.1 presents the U-Net's performance after 200 epochs, at which point training had converged, using the dataset and hyperparameters described in Chapter 4. The test set results show an accuracy of 0.8526 and a Dice score of 0.648. The lower performance on the validation set across all metrics is likely due to its focus on a different geographical area, which may be too distinct from the training set for the model to generalize effectively. Figure 6.1 visualizes the model's predictions compared to the ground truth. The prediction correctness map highlights areas of uncertainty and misclassification, particularly around class boundaries.

Model	Fraction	Accuracy ↑	Dice ↑	IoU ↑	Precision ↑	Recall ↑	Dice Loss ↓
U-Net	Train	0.8522	0.7287	0.6096	0.7226	0.7687	0.4295
	Val	0.7870	0.5940	0.4670	0.6412	0.5909	0.5712
	Test	0.8526	0.6480	0.5346	0.6616	0.6694	0.4865

Table 6.1: Performance metrics for U-Net after training for 300 epochs with patch size 256 by 256 and batch size 8 on 100% of the dataset: 1701 training images, 948 validation images, 1140 test images. The dropout rate is 0.15, the learning rate uses cosine annealing from 0.001 to 0.0001 and the weight decay 1e-4.

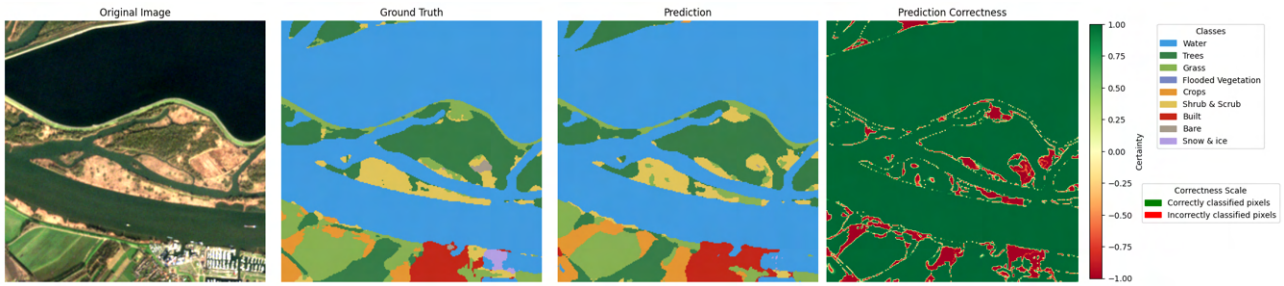
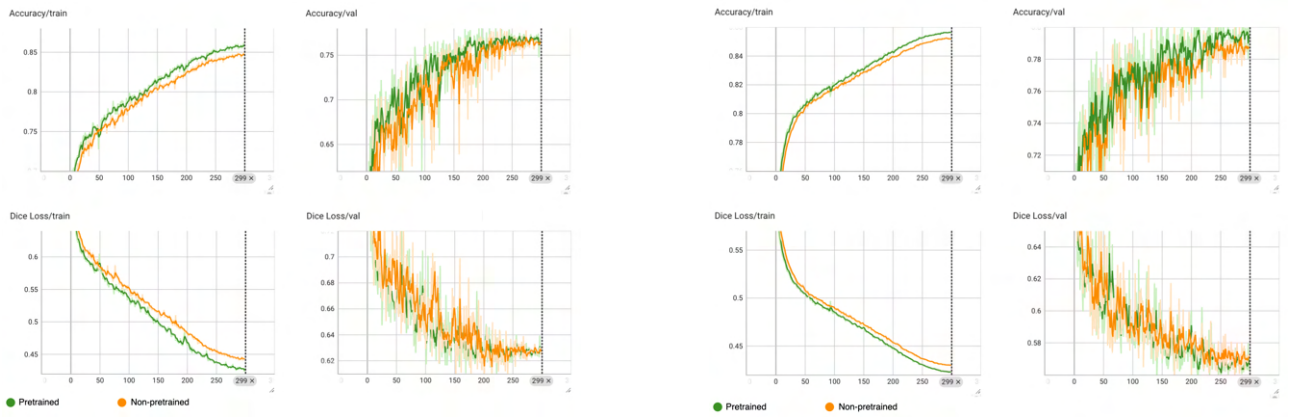


Figure 6.1: From left to right: the test set image in RGB, the ground truth classification by Dynamic World, the prediction of the U-Net, and last, the prediction correctness. Here, green indicates correctly classified pixels, red represents misclassified pixels, and the color intensity reflects the model's certainty.

6.2 Experiment 1: Impact of Pretraining (Label Dependency)

In Figure 6.2, the performance comparison between pretrained and non-pretrained U-Net architectures is depicted for a subset of 10% and 100% of the Sentinel-2 dataset. The results show that the performance gap between the pretrained and non-pretrained U-Net diminishes as the number of available labels increases, suggesting that the features learned by the autoencoder become less impactful when sufficient annotated labels are provided, as can also be seen in Table 6.2. This aligns with the understanding that, for tasks like semantic segmentation, the availability of labeled data primarily drives the training of the U-Net model, as the annotated labels contain the most essential features needed for the downstream task.



(a) 10% subset (b) 100% subset

Figure 6.2: Training and validation accuracy and dice loss for pretrained and non-pretrained U-Net models on 10% (left) and 100% (right) subsets of the Sentinel-2 dataset. The performance gap between pretrained and non-pretrained models is more pronounced for the smaller dataset, while the difference diminishes as more labeled data is available, indicating that the impact of pretraining decreases with increasing label availability.

When analyzing the performance changes across different test set subsets, as shown in Table 6.2, the relevant percentage improvement remain relatively small compared to those observed in the training set (Appendix G). While the gap between the non-pretrained and pretrained U-Net results narrows as the subset size increases, the pretrained model continues to outperform the non-pretrained model, though only slightly at 100%.

Subset	Accuracy	Dice	IoU	Precision	Recall	Loss
1%	+2.33%	+5.41%	+5.96%	+4.43%	+10.90%	-8.05%
10%	+1.23%	+2.45%	+3.57%	+1.65%	+1.08%	-3.32%
30%	+0.98%	+2.24%	+3.22%	+2.09%	+1.93%	-3.58%
50%	+0.79%	+2.57%	+3.38%	+1.95%	+3.27%	-3.01%
70%	+0.96%	+2.90%	+3.73%	+3.29%	+1.83%	-3.14%
100%	+0.50%	+1.32%	+1.67%	+1.33%	+0.79%	-1.44%

Table 6.2: Relative percentage improvement for all performance metrics on the test set for the non-pretrained versus the pretrained U-Net.

The results for the training, validation, and test sets across all subsets, comparing pretrained and non-pretrained models, are provided in Appendix G.

The effect of pretraining was also evaluated on the VHR imagery from Pléiades NEO, using corresponding manually annotated labels, as described in Section 5.3. A significant improvement was observed, with a 46.20% relative increase in weighted accuracy, rising from 60.35% (non-pretrained) to 88.23% (pretrained), as shown in Table 6.3.

Model Type	Weighted ↑ Accuracy	Dice ↑	IoU ↑	Precision ↑	Recall ↑	Dice Loss ↓
Non-pretrained	0.6035	0.2827	0.2243	0.3889	0.3158	0.5114
Pretrained	0.8823	0.4457	0.3919	0.5079	0.4551	0.5457

Table 6.3: Performance comparison of non-pretrained and pretrained U-Net on VHR imagery and labels.

6.3 Experiment 2: Impact of Resolution

Experiment 2, as outlined in Chapter 4, evaluated the performance of U-Net models trained on VHR and medium-resolution datasets to investigate the impact of image resolution on segmentation accuracy.

Table 6.4 compares the performance of non-pretrained VHR labels with non-pretrained medium-resolution labels on both test sets. Due to differences in dataset size and resolution, the number of training epochs was adjusted accordingly. The performance was documented at the epoch where the weighted accuracy began to converge, which was 50 epochs for VHR and 200 epochs for medium resolution.

Resolution	Weighted ↑ Accuracy	Dice ↑	IoU ↑	Precision ↑	Recall ↑	Dice Loss ↓
Medium	0.5751	0.3454	0.2687	0.4148	0.4395	0.7241
Very High	0.6035	0.2827	0.2243	0.3889	0.3158	0.5114

Table 6.4: Performance comparison between medium-resolution (Sentinel-2) and VHR (Pléiades NEO) imagery.

The most effective way to compare the performance of both approaches is through visual evaluation, as the metrics are heavily influenced by class imbalance and the incomplete presence of labels in the ground truth, as discussed in Section 5.3. These factors impose a strict upper limit on the achievable accuracy, making quantitative comparisons less reliable. Figure 6.3 shows the semantic segmentation result of both resolutions.

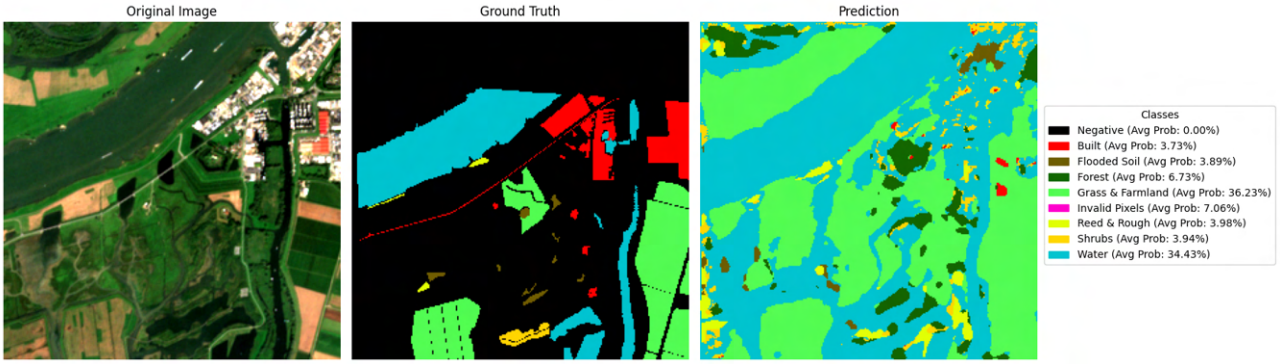
From Figure 6.3, it is clear that although the performance metrics for both resolutions appear comparable, and in some cases even slightly higher for the medium-resolution imagery and labels, the VHR imagery and labels provide more precise segmentation results.

In addition to more accurately capturing the different classes, as can be visually assessed by comparing the original image with the prediction in Figure 6.3, the VHR imagery also preserves finer details. For example, the boat on the water appears as a bright white patch in the VHR image and is labeled red in the prediction, despite not being included in the ground truth labels. In contrast, the medium-resolution prediction fails to capture the boats present on the river.

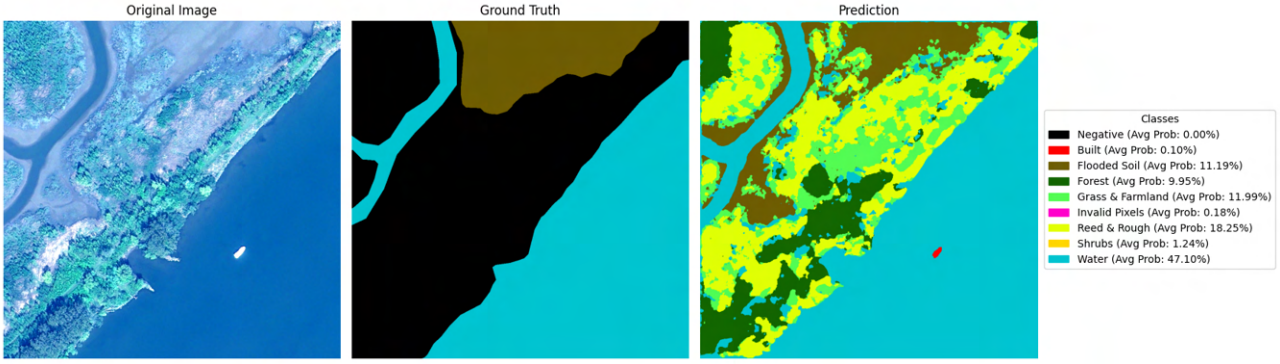
Furthermore, for this specific use case in the Biesbosch, the class "flooded soil" is particularly important, as it frequently transitions to water during high tides and helps in monitoring flood extent and tidal dynamics. This transition is best captured by the VHR imagery (represented by the brown-colored class in Figure 6.3), highlighting the advantage of using VHR data for

wetland monitoring. Although some flooded soil is present in the medium-resolution image, it does not appear in the prediction.

See the Appendix H for more comparisons between segmentation results of the medium-resolution imagery (Section H.1) versus the VHR imagery (Section H.2).



(a) Medium-resolution segmentation results



(b) VHR segmentation results

Figure 6.3: Comparison of medium-resolution and VHR segmentation results. While performance metrics appear similar, VHR imagery (b) provides finer details and more precise segmentation. The legend shows the average predicted probability for each class across the image, highlighting model confidence in class assignments.

Chapter 7

Discussion

In this chapter, the key findings and limitations of the research are discussed, the results are analyzed in relation to the research questions described in Chapter 1, and directions for future research are explored.

7.1 Analysis of the Results

The baseline U-Net model achieves an accuracy of 85.26% in distinguishing different vegetation classes, demonstrating its potential for wetland monitoring. Visual inspection of the resulting segmentation maps (Figure 6.1) shows that the model distinguishes vegetation classes well enough to identify areas that may need more attention, such as for conservation or grazing.

It is important to note that no other study uses exactly the same classification scheme, resolution, and labels with a U-Net model, making direct comparisons to the state-of-the-art difficult. One of the most recent and comparable studies is by Mainali et al. [53], which presents a high-resolution (1m) wetland mapping model using a U-Net-based deep learning approach, trained on Sentinel-2, LiDAR, NAIP, and geomorphological data. The model achieves high accuracy (94%), precision (96.5%), recall (90.2%), and an IoU of 0.873 in its primary study area. However, when applied to a new geographical region, precision drops to approximately 80% and recall significantly decreases to 48%, emphasizing the impact of changing the geographical domain - a challenge also observed in this research. The basic model in the research by Mainali et al. uses only NAIP and Sentinel-2 data, achieving 91.6% accuracy, 90.5% precision, 91.3% recall, and an IoU of 0.833. The noticeable difference in performance compared to their more advanced model trained on four modalities instead of two highlights the difficulty of directly comparing results. Performance improvements may be partly attributed to the inclusion of additional data sources, resolution differences, and/or the geographical areas on which the model is trained rather than to the model architecture alone.

Furthermore, most existing research on wetland mapping focuses on classification rather than segmentation, and those that do employ segmentation often rely on traditional machine learning approaches [55, 52]. Also, wetland monitoring suffers from a lack of standardized global definitions and comprehensive classification schemes, complicating cross-study comparisons.

One key challenge in the experimental setup of the baseline experiment is the accuracy of the Dynamic World labels, which do not always align with the satellite imagery upon visual inspection. This discrepancy imposes an upper bound on the achievable accuracy of the model.

Moreover, although this study does not explore variations in the depth of the U-Net architecture, future work could investigate modifying the encoder-decoder depth of the U-Net (and autoencoder) to enhance feature extraction.

Lastly, unlike the conventional 70/15/15 train/validation/test split, the dataset in this study is divided based on geographical regions rather than randomly. The test set is about 70% the

size of the training set, which may influence performance metrics as the large number of test images increases the chance of accumulating misclassifications, especially if the distribution of classes or environmental conditions differs from the training set.

The influence of pretraining on a U-Net model is shown in Experiment 1, which indicates that self-supervised learning enhances training performance, especially when only a small subset of labeled data is available. However, its benefits are less apparent on the validation and test sets, as the large size of these sets provides robust evaluations that probably dilute observable differences between self-supervised and purely supervised approaches. Additionally, because the autoencoder uses the same train/validation/test split as the U-Net for Sentinel-2 data, it is not exposed to the geographical areas present in the validation and test sets. This limited exposure hampers its ability to extract spatial features relevant to the target area. As noted by Bai, Yu et al. [7], pretraining on large-scale datasets from domains different from remote sensing can lead to a domain mismatch, causing degradation in generalization performance during fine-tuning instead of an increase in performance. Since the autoencoder in this study is pretrained on data that does not encompass the geographical areas of interest, its effectiveness is potentially restricted. Moreover, the poor performance metrics of the VHR autoencoder compared to the medium-resolution one suggest that reconstruction loss does not necessarily reflect the usefulness of the learned representations for the downstream segmentation task. This reinforces the idea that pretraining quality should not be judged solely on reconstruction accuracy, as a higher reconstruction loss for VHR does not directly imply worse performance when fine-tuned for U-Net segmentation. Alternative approaches, such as feature prediction loss as proposed by Pihlgren et al. [62], provide a potential improvement by aligning learned representations with high-level semantic features rather than pixel-wise reconstructions.

To address the issues encountered in this experiment, future research should consider using separate train/validation/test splits for pretraining to ensure that the pretraining data includes representative geographical areas, thereby aligning it with the target application and maximizing the benefits of transfer learning.

The influence of resolution on semantic segmentation of vegetation in wetland areas is explored in Experiment 2, confirming research by Mainali et al. [53] that higher-resolution imagery significantly improves segmentation performance compared to medium- or low-resolution. However, this difference is less apparent in the performance metrics, which are heavily influenced by shortcomings in the manual annotation, such as the presence of unlabeled pixels and high class imbalance. Although the methodology of this research attempts to account for these factors by using, among others, a weighted loss, the performance metrics remain unreliable as a definitive measure of which resolution performs better. Instead, visual inspection of the segmentation maps alongside the satellite imagery provides a more accurate assessment of the resolution's impact.

One limitation of this experiment is the limited temporal spread of the data, as it includes imagery only from April and September. Expanding the dataset to cover a broader range of seasons could improve model generalization by capturing seasonal variations in vegetation dynamics, thereby providing a more comprehensive assessment of how resolution influences segmentation performance across different temporal conditions.

7.2 Answering the Research Questions

The results of this study provide key insights into how effective semantic segmentation for land cover classification in dynamic wetlands is achieved while reducing reliance on annotated data.

RQ1: Can self-supervised learning improve the efficiency and accuracy of land cover classification for estimating vegetation roughness in dynamic wetland environments? Experiment 1 indicates that self-supervised learning enhances training performance, especially when only a small subset of labeled data is available. However, its impact on the validation and test sets is less pronounced, likely due to a significant domain shift between the pretrained data and the validation/test data in the downstream segmentation task. It is shown that, when labeled data is abundant, the U-Net model learns more from direct supervision than from pretraining. Additionally, if the model is trained for a sufficiently long duration, the influence of pretraining diminishes as the U-Net adapts to the labeled data.

RQ2: How does image resolution influence the amount of labeled data required for accurate vegetation classification? Experiment 2 demonstrates that higher-resolution imagery results in better segmentation performance due to its ability to capture finer spatial details. This, in turn, increases the potential benefits of pretraining, as the autoencoder learns more meaningful representations from high-resolution data. However, the poor performance metrics of the VHR autoencoder suggest that reconstruction loss alone is not always a reliable indicator of segmentation effectiveness. Furthermore, higher-resolution imagery may require fewer labeled samples because the model leverages spatial patterns more effectively, although this advantage is highly dependent on annotation quality and class imbalance. The results also suggest that the combination of pretraining and high-resolution imagery is most beneficial in settings with limited labeled data, as the pretraining phase helps extract structural information that compensates for the lack of supervision.

These findings indicate that the effectiveness of pretraining is conditional on both the availability of labeled data and the resolution of the imagery. When labeled data is abundant, the benefits of pretraining diminish as the model relies more on supervised learning. Similarly, if the model is trained for a sufficiently long duration, pretraining has a limited effect because the U-Net learns task-specific features directly from annotations. However, when labeled data is scarce and imagery is high resolution, pretraining provides a meaningful boost by leveraging spatial detail that would otherwise be difficult to capture. A key takeaway from both experiments is that pretraining and resolution interact in important ways: pretraining has a greater impact when using high-resolution imagery, as it captures more spatial detail that is leveraged during self-supervised learning. This reinforces the need for strategic dataset design, where resolution and pretraining decisions are made jointly based on the amount of available annotations and specific application requirements.

Overall, these findings address the overarching research question by demonstrating that effective semantic segmentation for land cover classification in dynamic wetlands can be achieved through the combined use of self-supervised pretraining and high-resolution imagery, thereby reducing reliance on extensive annotated data.

7.3 Future Research

Beyond addressing the limitations of this study, future research could explore change detection as a valuable downstream task. This can be achieved using a Siamese Network, as demonstrated by [35], which integrates structural change detection with pixel-level segmentation. Additionally, investigating alternative architectures such as DeepLabV3+, which leverages atrous convolutions and conditional random fields for improved boundary detection and outperforms U-Net in both speed and accuracy [15, 37], could help overcome challenges related to computation, class imbalance, and segmentation precision.

Furthermore, the upcoming Copernicus Hyperspectral Imaging Mission for the Environment (CHIME), scheduled for launch in 2028, presents an opportunity to integrate hyperspectral data, which could enhance vegetation classification and wetland monitoring through improved spectral resolution [58]. Similarly, self-supervised learning datasets like SSL4EO-S12 [79] provide a promising foundation for pretraining models on large-scale remote sensing imagery, improving generalization and representation learning for segmentation tasks.

Moreover, building on findings from Waldeland et al. [77], future studies might assess the integration of multispectral data with ancillary sources like SAR or LiDAR to capture vegetation height using a U-Net. Lastly, the emergence of foundation models in remote sensing enables transfer learning across diverse datasets [27]. Fine-tuning such a foundation model to perform semantic segmentation could allow for a more robust and adaptable approach compared to training a task-specific model like U-Net.

Chapter 8

Conclusion

8.1 Conclusion

This study demonstrates that a U-Net model trained from scratch on Sentinel-2 data can achieve an accuracy of 85.26% for land cover classification in dynamic wetland environments. When only 10% of the dataset is annotated, self-supervised pretraining using an autoencoder results in a 1.23% relative improvement in accuracy compared to training from scratch. In scenarios with sparse labels, the use of high-resolution imagery (0.3m by 0.3m) with corresponding labels enhances segmentation accuracy by a 4.94% relative improvement over medium-resolution data (10m by 10m). Pretraining has the most pronounced effect on high-resolution imagery, yielding a relative improvement of 46.20%. These results highlight the benefits of pretraining and higher-resolution data. The primary contribution of this research lies in the comparative analysis of these factors, rather than in benchmarking model performance against state-of-the-art methods. Such direct comparisons are challenging due to variations in classification schemes, data modalities, and geographical domains. These findings emphasize the importance of strategically leveraging both pretraining and image resolution to enhance segmentation performance while reducing reliance on extensive annotated datasets.

8.2 Acknowledgments

I would like to thank my supervisor, Arnoud Visser, for his guidance, critical insights and support the past months. I appreciate his patience and encouragement during challenging times and his efforts to help me gain confidence in my work.

I would also like to express my gratitude to my Accenture supervisor, Enrico Ceretti, for helping me find my way at Accenture and for sticking with me, even though this topic was not in his field of expertise. I am equally thankful to all the great people from the Data & AI team for the enjoyable (long) days at the office and for giving me my first insight into corporate life.

Furthermore, I would like to thank Roberto del Prete for showing interest in my work and recognizing its potential. His technical expertise in the field of remote sensing (a domain that was entirely new to me) greatly improved my research. Moreover, his introduction to the European Space Agency community guided me to start my journey as a visiting researcher at ESRIN in Frascati, Italy.

Last, but not least, I want to thank my family, friends, and most importantly my partner Simón Rodriguez Cedeño, who has been there for me every step of the way. Whether it was celebrating my highs or providing emotional support during my lows - you've helped me make this work possible.

Appendix A

Deep Learning Methods in Remote Sensing

Table A.1: (Non-Exhausive) Overview of Deep Learning Methods in Remote Sensing, based on review by Janga et al. [43].

DL Method	Description	Example Applications
Deep Convolutional Neural Networks (DCNNs)	Capture spatial hierarchies in data through convolutional layers	Land cover classification, vegetation analysis
Residual Networks (ResNets)	Use skip connections to address vanishing gradient issues, enabling very deep architectures	High-resolution image analysis, vegetation mapping
You Only Look Once (YOLO)	Provides real-time object detection by processing the entire image in a single forward pass	Object detection tasks such as urban feature identification and wildlife monitoring
Faster R-CNN	Combines region proposal networks and CNNs for accurate object detection	Detailed feature extraction in high-resolution images
Self-Attention Mechanisms (e.g., Vision Transformers)	Model global dependencies using self-attention, improving performance for complex spatial relationships	Advanced land cover segmentation, vegetation health analysis
Long Short-Term Memory Networks (LSTMs)	Process sequential data to monitor temporal changes	Time-series analysis of vegetation dynamics, climate change impact studies
Generative Adversarial Networks (GANs)	Generate synthetic data to augment training datasets, addressing challenges of limited labeled data	Data augmentation, enhancing small remote sensing datasets
Deep Reinforcement Learning (DRL)	Learn optimal strategies through interaction with dynamic environments	Resource allocation, adaptive monitoring strategies

Appendix B

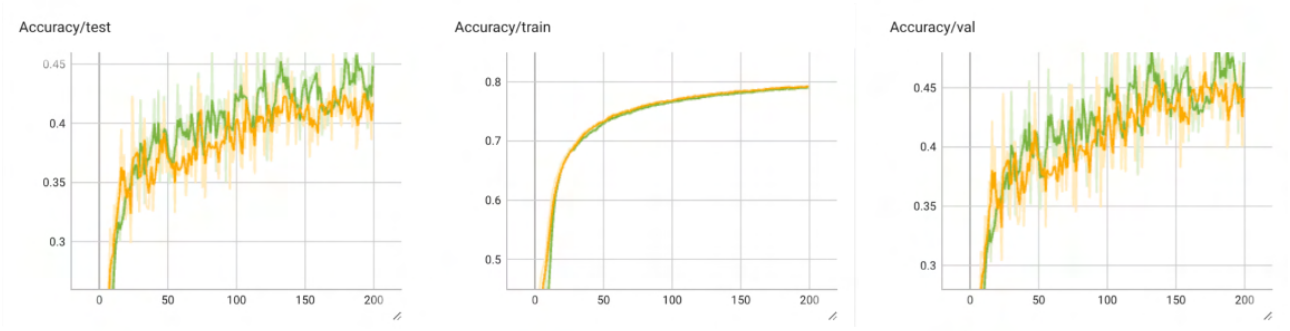
Hyperparameter Tuning

B.1 Autoencoder

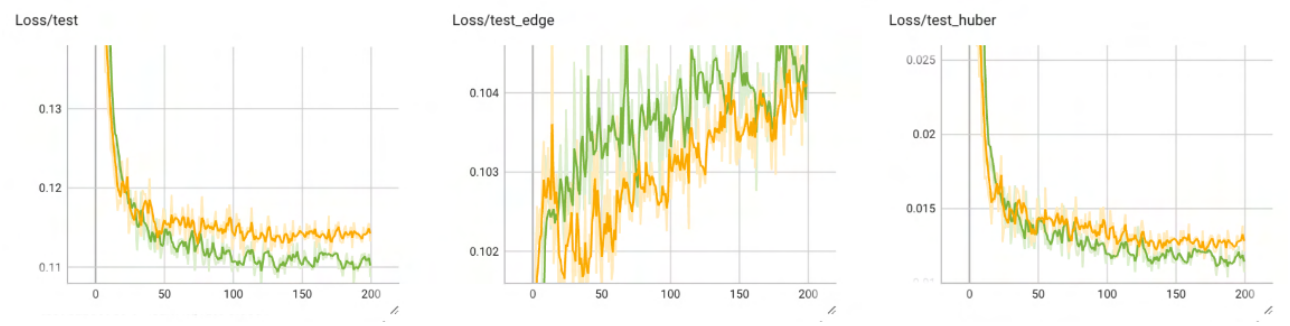
B.1.1 Learning Rate

Learning Rate	Accuracy ↑	PSNR ↑	SSIM ↑	Huber ↓ Loss	SSIM ↓ Loss	Edge ↓ Loss	Mixed ↓ Loss
0.001	0.4401	16.1603	0.4243	0.0121	0.2879	<u>0.1041</u>	0.1132
0.0001	<u>0.4881</u>	<u>16.7063</u>	<u>0.4529</u>	<u>0.0107</u>	<u>0.2735</u>	0.1067	<u>0.1088</u>

Table B.1: Performance metrics on the test set after 200 epochs for different learning rate strategies on 50% of the dataset.



(a) Test, Validation and Training Accuracy



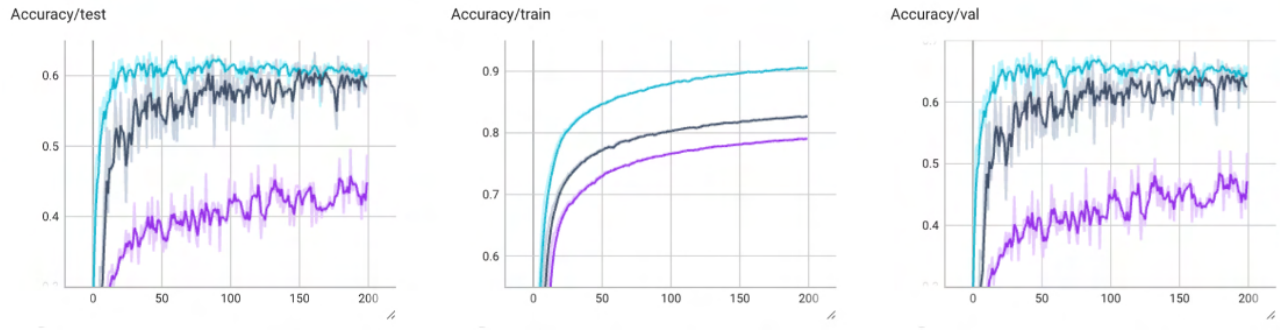
(b) Test, Validation and Training Dice Loss

Figure B.1: Accuracy, Loss, and over 200 epochs using 50% of the Sentinel-2 dataset. The figure compares the performance of two different, constant learning rate strategies: 0.001 (orange) and 0.0001 (green). In al runs, the dropout probability has been kept at a consistent 25%.

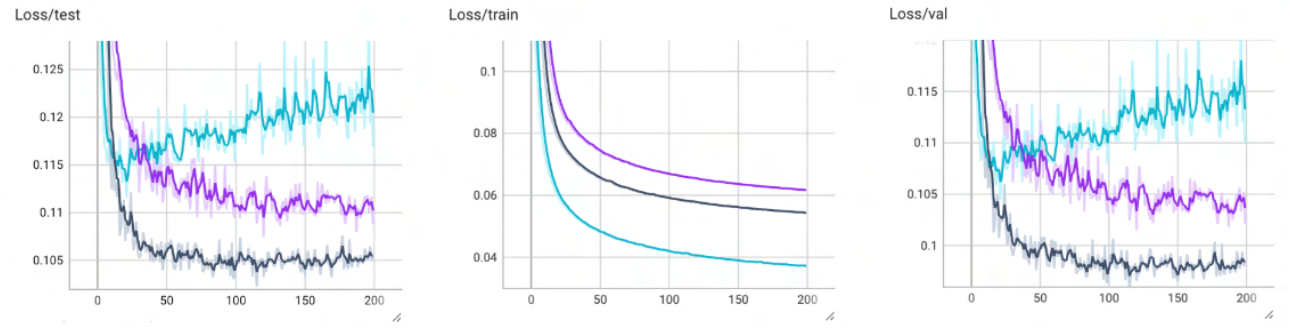
B.1.2 Dropout Probability

Learning Rate	**Accuracy ↑**	**PSNR ↑**	**SSIM ↑**	**Huber ↓ Loss**	**SSIM ↓ Loss**	**Edge ↓ Loss**	**Mixed ↓ Loss**
0%	<u>0.6158</u>	17.0287	0.4111	0.0100	0.2945	0.1179	0.1169
15%	0.5848	<u>17.3022</u>	<u>0.4714</u>	<u>0.0094</u>	<u>0.2643</u>	<u>0.1074</u>	<u>0.1055</u>
25%	0.4881	16.7063	0.4529	0.0107	0.2735	0.1067	0.1088

Table B.2: Performance metrics on the test set after 200 epochs for different dropout probabilities on 50% of the dataset.



(a) Test, Validation and Training Accuracy



(b) Test, Validation and Training Dice Loss

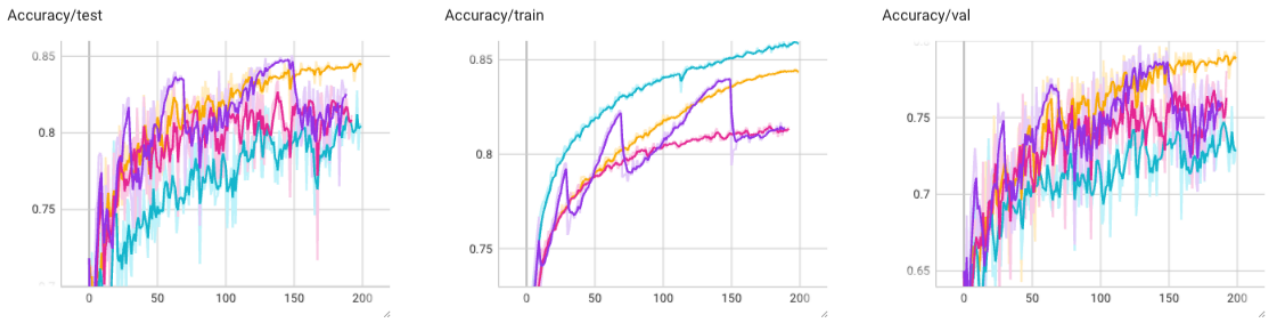
Figure B.2: Test, validation and training accuracy and loss over 200 epochs using 50% of the Sentinel-2 dataset. The figure compares the performance of three different dropout probabilities: 0% (blue), 15% (purple) and 25% (black). All runs were trained using a learning rate of 0.0001.

B.2 U-Net

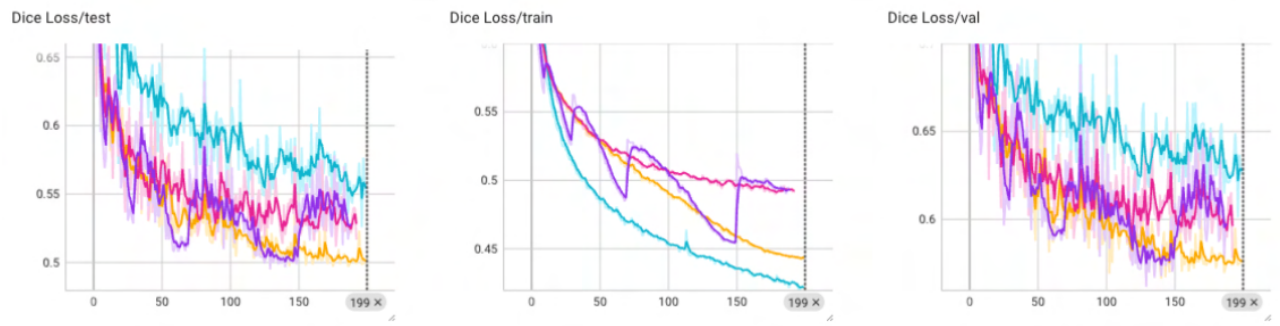
B.2.1 Learning Rate

Learning Rate	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Loss ↓
0.001	0.8117	0.6109	0.4925	0.6109	0.6029	0.6746	0.5167
0.0001	0.8030	0.5852	0.4641	0.5852	0.6443	0.5789	0.5486
Cosine Annealing	0.8425	0.6419	0.5241	0.6419	0.6757	0.6432	0.5028
Warm Cosine Ann.	0.7880	0.5603	0.4417	0.5603	0.6152	0.5834	0.5633

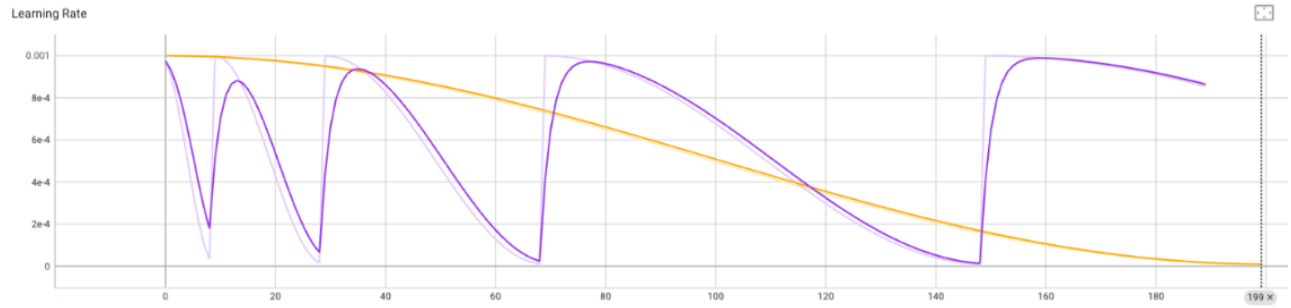
Table B.3: Performance metrics on the test set after 200 epochs for different learning rate strategies.



(a) Test, Validation and Training Accuracy



(b) Test, Validation and Training Dice Loss



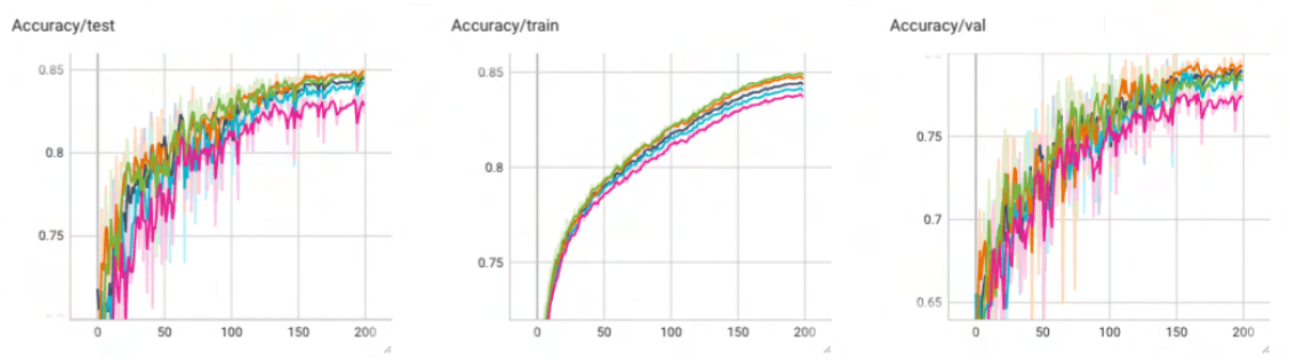
(c) Learning Rate

Figure B.3: Accuracy, Dice Loss, and **Learning Rate Schedules** over 200 epochs using 50% of the Sentinel-2 dataset. The figure compares the performance of four different learning rate strategies: constant learning rates of 0.001 (pink) and 0.0001 (blue), cosine annealing (orange), and warm cosine annealing (purple). In al runs, the dropout probability has been kept at a consistent 25%.

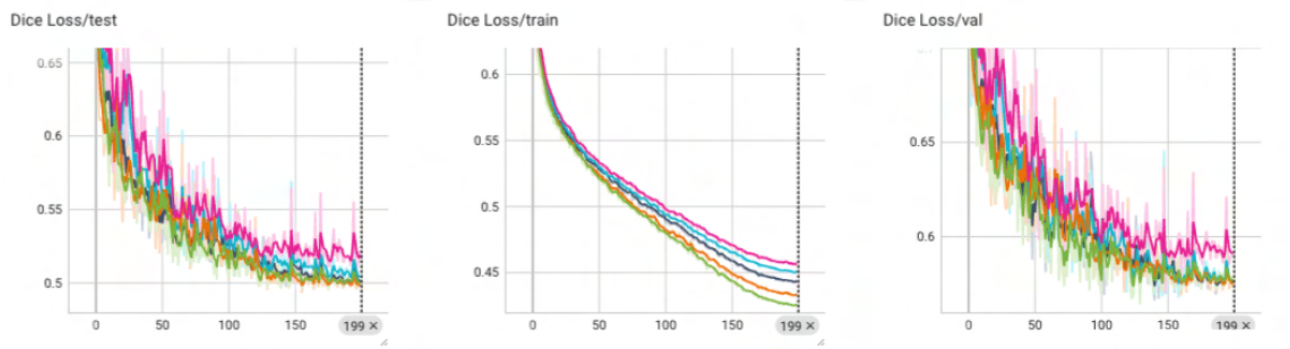
B.2.2 Dropout Probability

Learning Rate	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Loss ↓
0%	0.8448	0.6337	0.5198	0.6337	0.6690	0.6374	0.5019
15%	<u>0.8468</u>	0.6416	<u>0.6416</u>	0.5852	0.6741	<u>0.6432</u>	<u>0.4996</u>
25%	0.8425	<u>0.6419</u>	0.5241	<u>0.6419</u>	<u>0.6757</u>	<u>0.6432</u>	0.5028
35%	0.8402	0.6343	0.5182	0.6343	0.6646	0.6412	0.5057
45%	0.8241	0.6141	0.4950	0.6141	0.6670	0.6185	0.5226

Table B.4: Performance metrics on the test set after 200 epochs for different learning rate strategies.



(a) Test, Validation and Training Accuracy



(b) Test, Validation and Training Dice Loss

Figure B.4: Test, validation and training accuracy and loss over 200 epochs using 50% of the Sentinel-2 dataset. The figure compares the performance of five different dropout probabilities: 0% (green), 15% (orange), 25% (black), 35% (blue), and 45% (pink). All runs were trained using a cosine annealing learning rate schedule.

Appendix C

(Pretrained) U-Net Results on Gaofen-2 imagary and LULC-labels

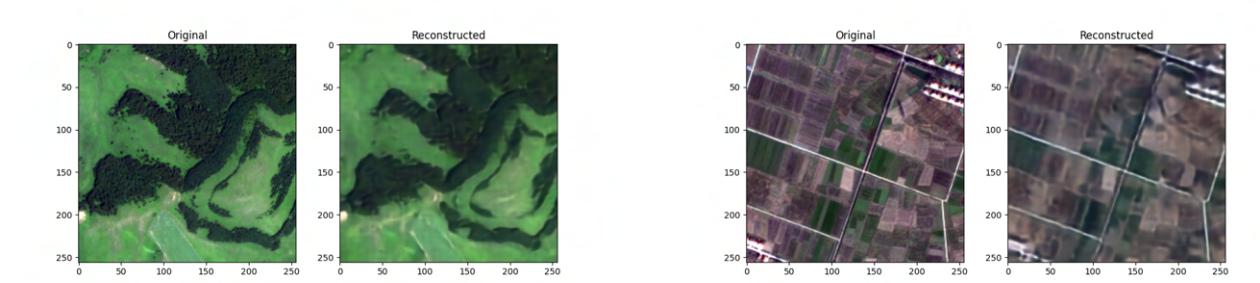


Figure C.1: Reconstruction of the Gaofen-2 satellite imagery using an autoencoder trained from scratch.

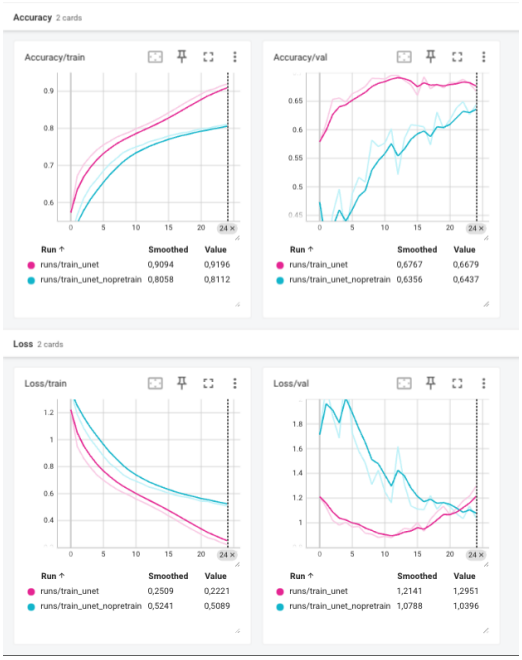


Figure C.2: Training accuracy and loss of U-Net, comparing the performance of pretrained and non-pretrained models.

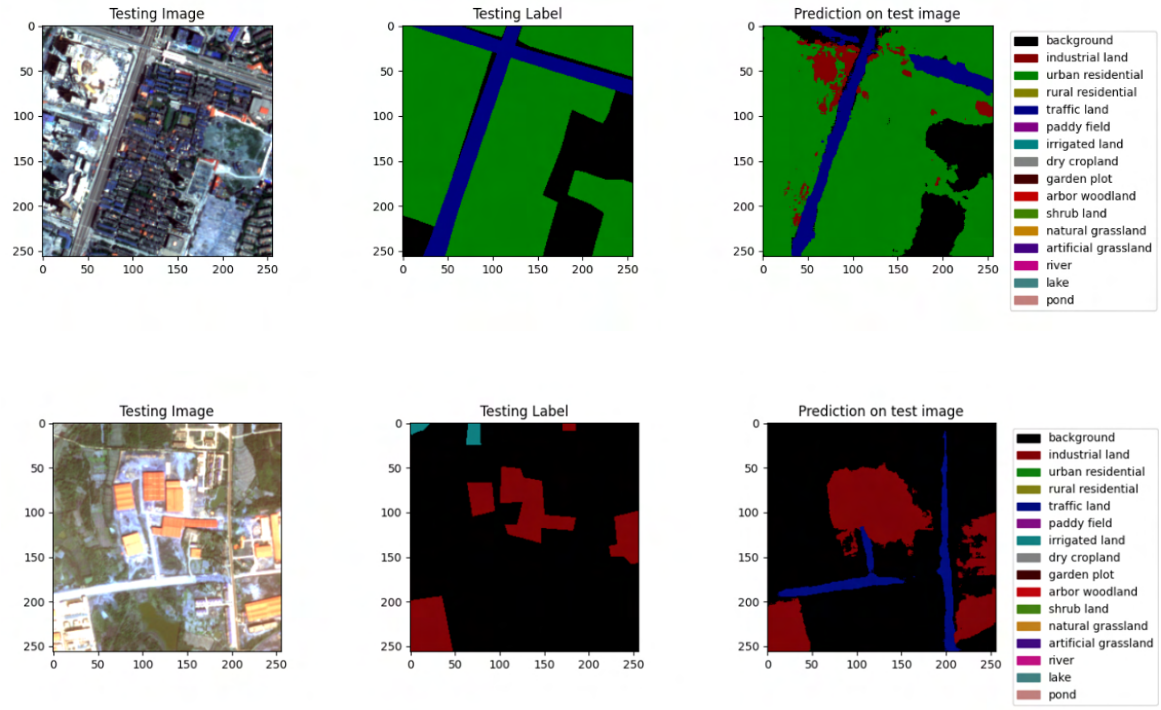


Figure C.3: Results without pretraining on the autoencoder.

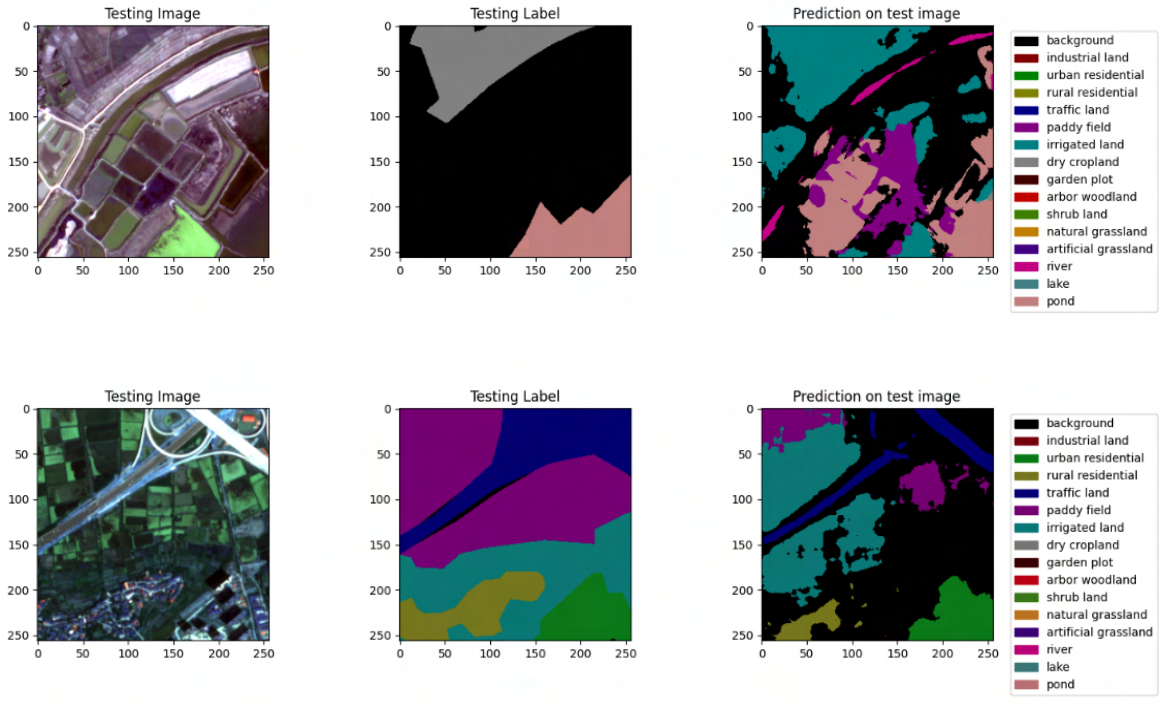


Figure C.4: Results with pretraining on the autoencoder.

Figure C.5: U-Net predictions (after 25 epochs with batch size 16) with and without pretraining on the autoencoder.

Appendix D

Sentinel-2 imagery with Dynamic World labels

D.1 Available Sentinel-2 bands

Name	Scale	Pixel Size	Description
B1	0.0001	60 meters	Aerosols
B2	0.0001	10 meters	Blue (496.6nm (S2A) / 492.1nm (S2B))
B3	0.0001	10 meters	Green (560nm (S2A) / 559nm (S2B))
B4	0.0001	10 meters	Red (664.5nm (S2A) / 665nm (S2B))
B5	0.0001	20 meters	Red Edge 1 (703.9nm (S2A) / 703.8nm (S2B))
B6	0.0001	20 meters	Red Edge 2 (740.2nm (S2A) / 739.1nm (S2B))
B7	0.0001	20 meters	Red Edge 3 (782.5nm (S2A) / 779.7nm (S2B))
B8	0.0001	10 meters	Near Infrared (835.1nm (S2A) / 833nm (S2B))
B8A	0.0001	20 meters	Red Edge 4 (864.8nm (S2A) / 864nm (S2B))
B9	0.0001	60 meters	Water vapor (945nm (S2A) / 943.2nm (S2B))
B11	0.0001	20 meters	Short Wave Infrared 1 (1613.7nm (S2A) / 1610.4nm (S2B))
B12	0.0001	20 meters	Short Wave Infrared 2 (2202.4nm (S2A) / 2185.7nm (S2B))
AOT	0.001	10 meters	Aerosol Optical Thickness
WVP	0.001	10 meters	Water Vapor Pressure
SCL	-	20 meters	Scene Classification Map
TCI_R	-	10 meters	True Color Image, Red Channel
TCI_G	-	10 meters	True Color Image, Green Channel
TCI_B	-	10 meters	True Color Image, Blue Channel
MSK_CLDPRB	-	20 meters	Cloud Probability Map
MSK_SNWPRB	-	10 meters	Snow Probability Map
QA10	-	10 meters	Always empty
QA20	-	20 meters	Always empty
QA60	-	60 meters	Cloud mask
MSK_CLASSI_OPAQUE	-	60 meters	Opaque clouds classification band (0=no clouds, 1=clouds)[1]. Masked out before February 2024
MSK_CLASSI_CIRRUS	-	60 meters	Cirrus clouds classification band (0=no clouds, 1=clouds). Masked out before February 2024
MSK_CLASSI_SNOW_ICE	-	60 meters	Snow/ice classification band (0=no snow/ice, 1=snow/ice). Masked out before February 2024

D.2 Selection of Wetland areas for Sentinel-2

Figure D.1: Wetland areas included in the Sentinel 2 dataset

(a) Loosdrechtse Plassen 2019-02-15

(b) Lauwersmeer 2019-04-01

(c) Oostvaardersplassen 2023-06-24

(d) Land van Saeftinghe 2024-08-12

(e) Biesbosch 2021-10-24

(f) Gendtse Polder 2021-12-21

Figure D.2: The six geographical areas used for the Sentinel 2 dataset.

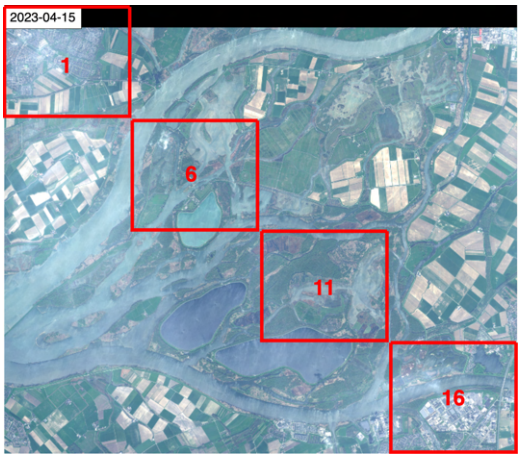
D.3 Dynamic World Classification Scheme

LULC Type	Color Coding	Description
Water	`#419bdf`	Permanent and seasonal water bodies
Trees	`#397d49`	Includes primary and secondary forests, as well as large-scale plantations
Grass	`#88b053`	Natural grasslands, livestock pastures, and parks
Flooded vegetation	`#7a87c6`	Mangroves and other inundated ecosystems
Crops	`#e49635`	Include row crops and paddy crops
Shrub & Scrub	`#dfc35a`	Sparse to dense open vegetation consisting of shrubs
Built Area	`#c4281b`	Low- and high-density buildings, roads, and urban open space
Bare ground	`#a59b8f`	Deserts and exposed rock
Snow & Ice	`#b39fe1`	Permanent and seasonal snow cover

Appendix E

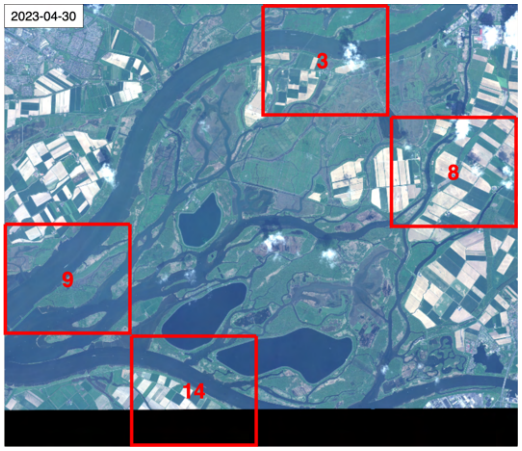
Pléiades NEO imagery with Manual Labels

E.1 Selection of Biesbosch areas for Pléiades NEO

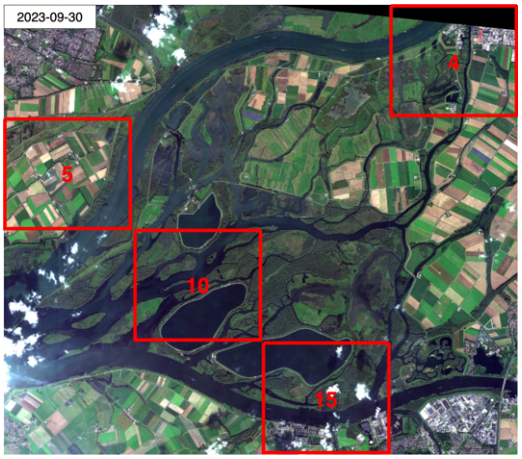


(a) 2023-04-15

(b) 2024-09-21



(c) 2023-04-20



(d) 2024-09-30

Figure E.1: Satellite images of the Biesbosch area, obtained from Pleiades NEO, used in this study.

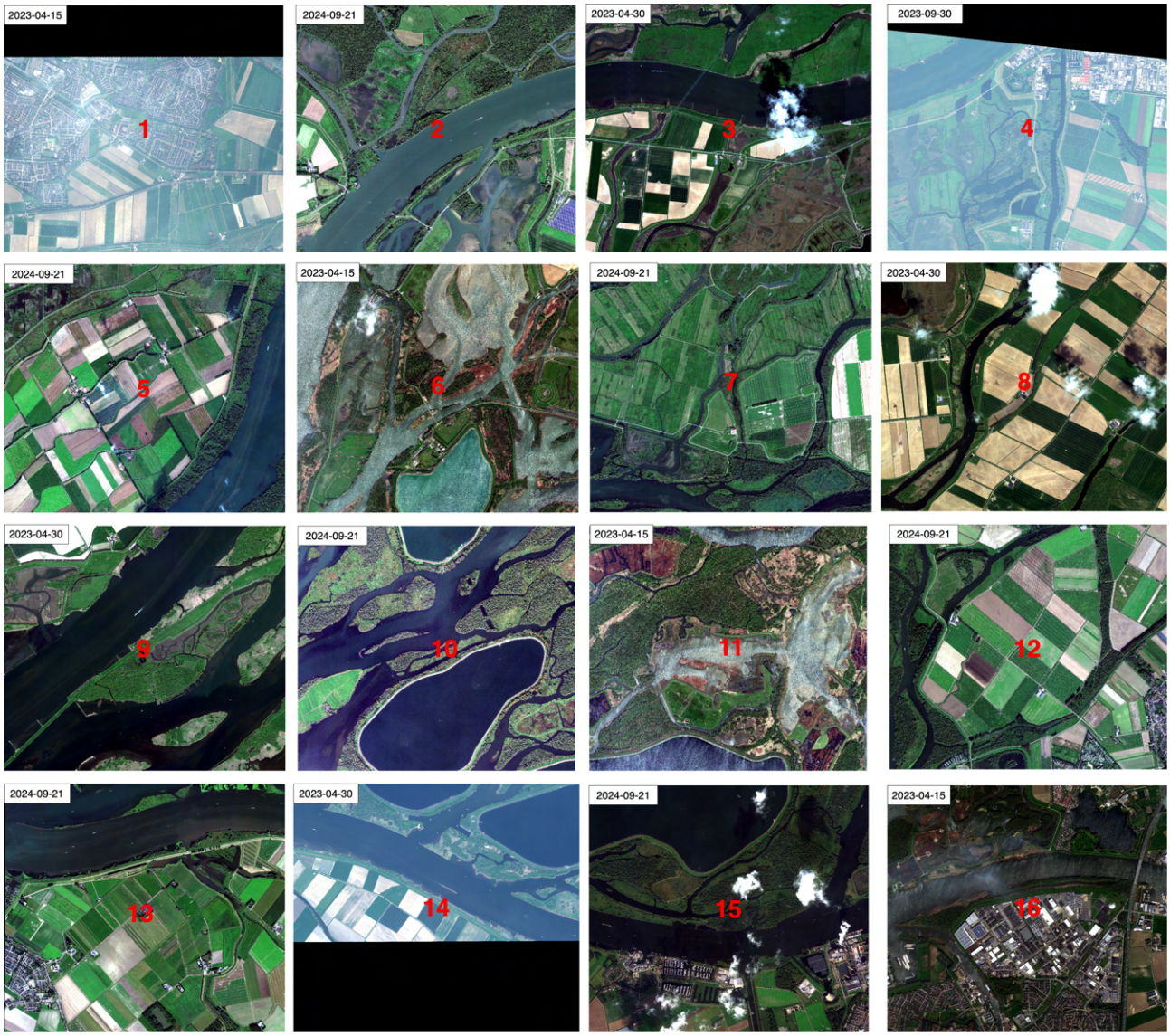
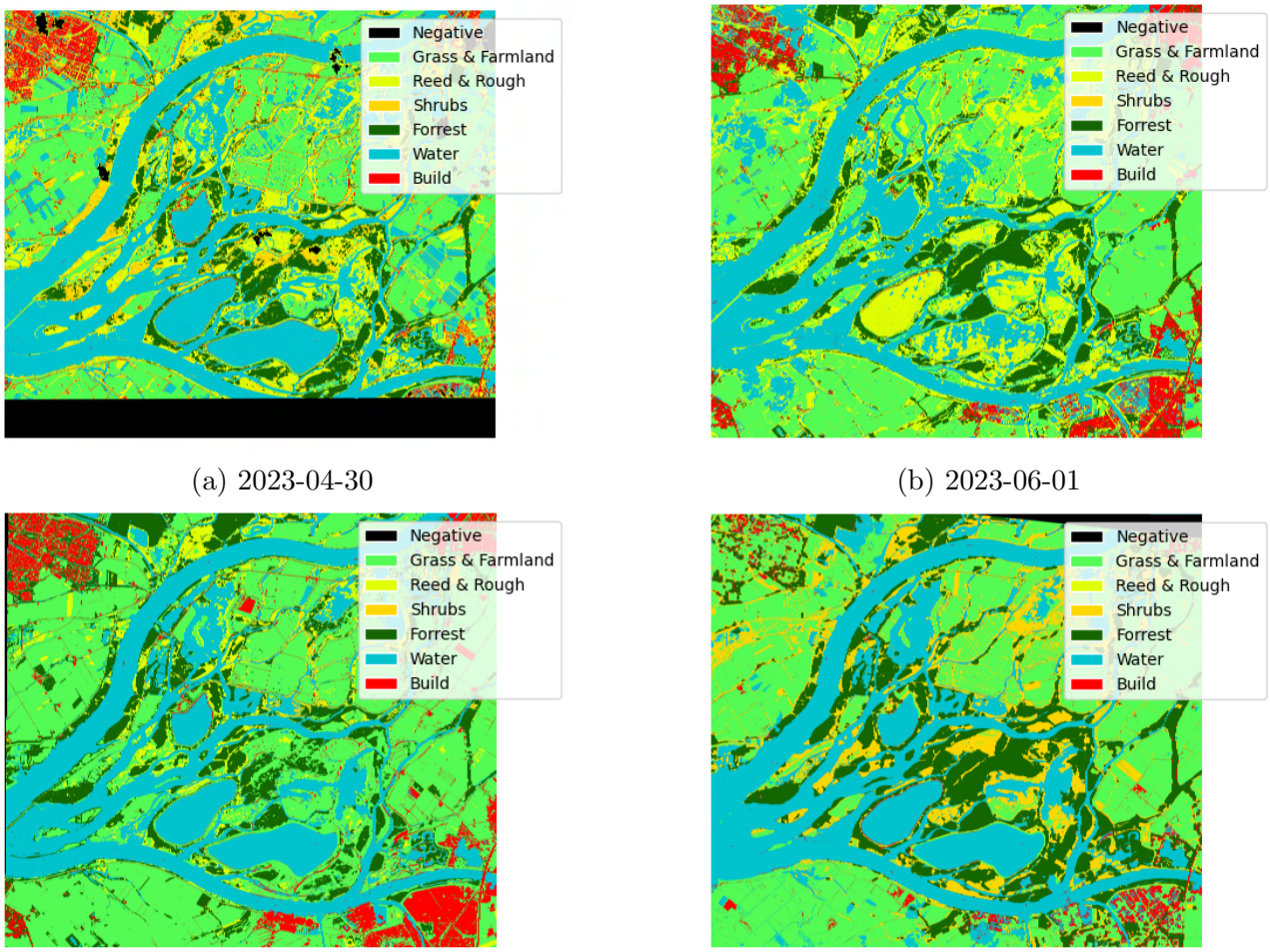


Figure E.2: Selection of tiles for manual annotation using Roboflow

E.2 Rijkswaterstaat Classification Scheme

LULC Type	Color Coding	Description
Grass and Farmland	`#53FB54`	Bare terrains, open or dense grass, and seasonal herbs
Reed and Rough Vegetation	`#E0FF06`	Wetland vegetation and tall rough vegetation
Shrubbery	`#FFD600`	Dense shrub vegetation, 2-5 meters in height
Forest	`#146600`	Tree-dominated vegetation, 5-15 meters or taller
Water	`#00C3CE`	Open water
Paved Surfaces	`#FF0000`	Hardened surfaces with minimal vegetation

E.3 Blackshark.ai labels



(a) 2023-04-30

(b) 2023-06-01

(c) 2024-09-21

(d) 2023-09-30

Figure E.3: The label mask for the Biesbosch area, created using Blackshark, highlights a notable inconsistency: despite some satellite images being taken in close proximity to one another, the labels show significant variation. This underscores the limitations and potential vulnerabilities of relying solely on AI labeling tools to generate ground truth labels.

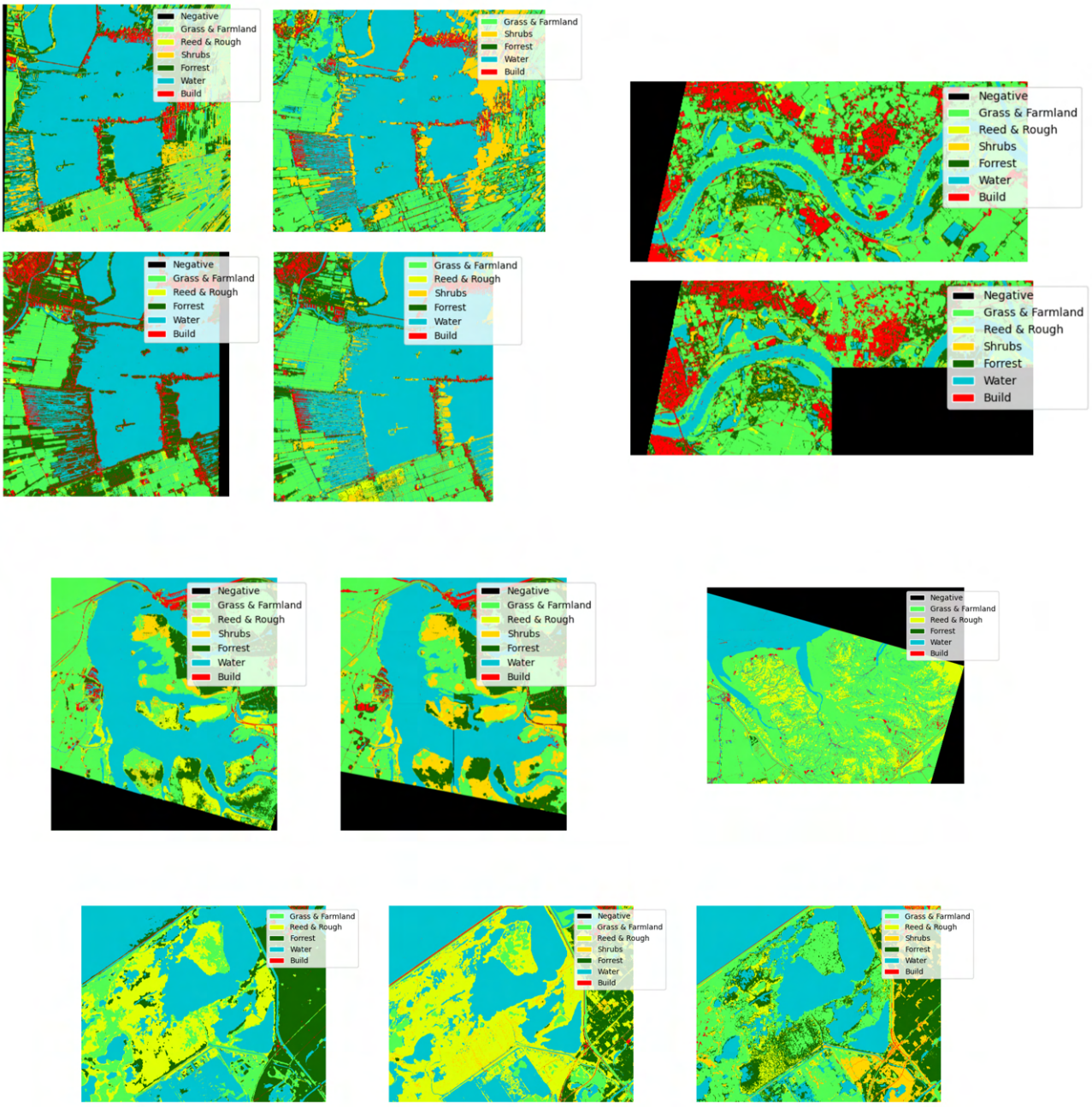


Figure E.4: Labels generated for the wetland areas based on high resolution satellite data.

E.4 Roboflow labels

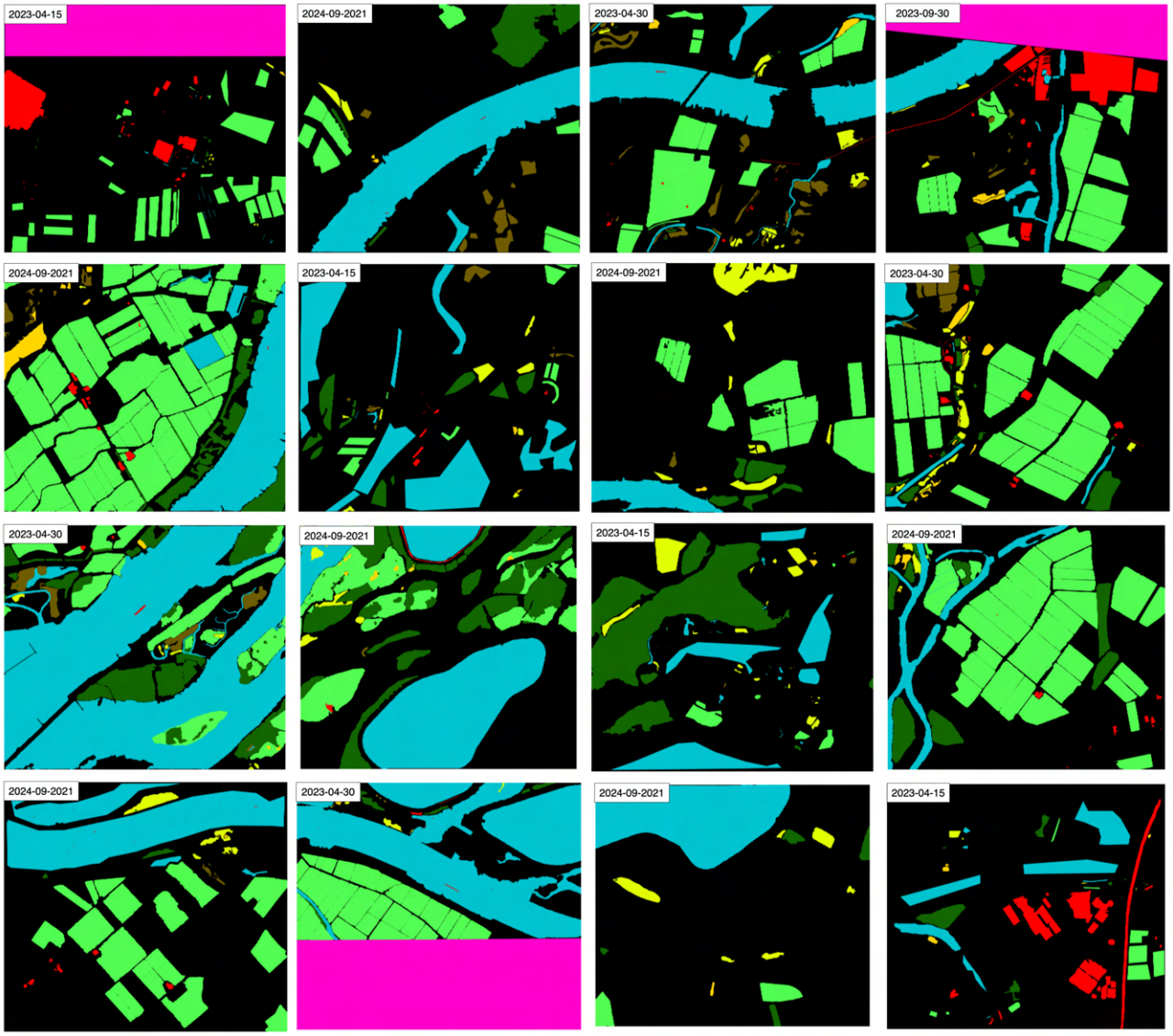


Figure E.5: Annotated masks for selection of Pléiades NEO imagery.

Pléiades NEO	Sentinel-2
2023-04-15	2023-04-05
2023-04-30	2023-04-30
2023-09-30	2023-09-09
2024-09-21	2024-09-21

Table E.1: Dates corresponding to the Pléiades NEO and Sentinel-2 datasets used in Experiment 2 to examine the impact of image resolution on vegetation classification in the Biesbosch.

Table E.2: Class distribution across VHR label dataset.

Class	**Train (%)**	**Validation (%)**	**Test (%)**
Negative	0.00	0.00	0.00
Built	2.76	2.79	2.54
Flooded Soil	1.65	1.68	1.19
Forest	12.38	11.35	13.21
Grass & Farmland	37.72	37.19	36.20
Invalid Pixels	0.02	0.03	0.02
Reed & Rough	2.14	1.82	2.50
Shrubs	0.65	0.61	0.38
Water	42.67	44.54	43.95

Table E.3: Class distribution across medium-resolution label dataset.

Class	**Train (%)**	**Validation (%)**	**Test (%)**
Negative	0.00	0.00	0.00
Built	1.56	15.85	6.85
Flooded Soil	1.81	0.45	0.70
Forest	7.69	10.51	43.68
Grass & Farmland	44.11	37.90	17.93
Invalid Pixels	0.00	0.00	0.00
Reed & Rough	1.52	13.67	5.01
Shrubs	0.78	0.58	0.70
Water	42.53	21.06	25.14

Appendix F

Baseline U-Net

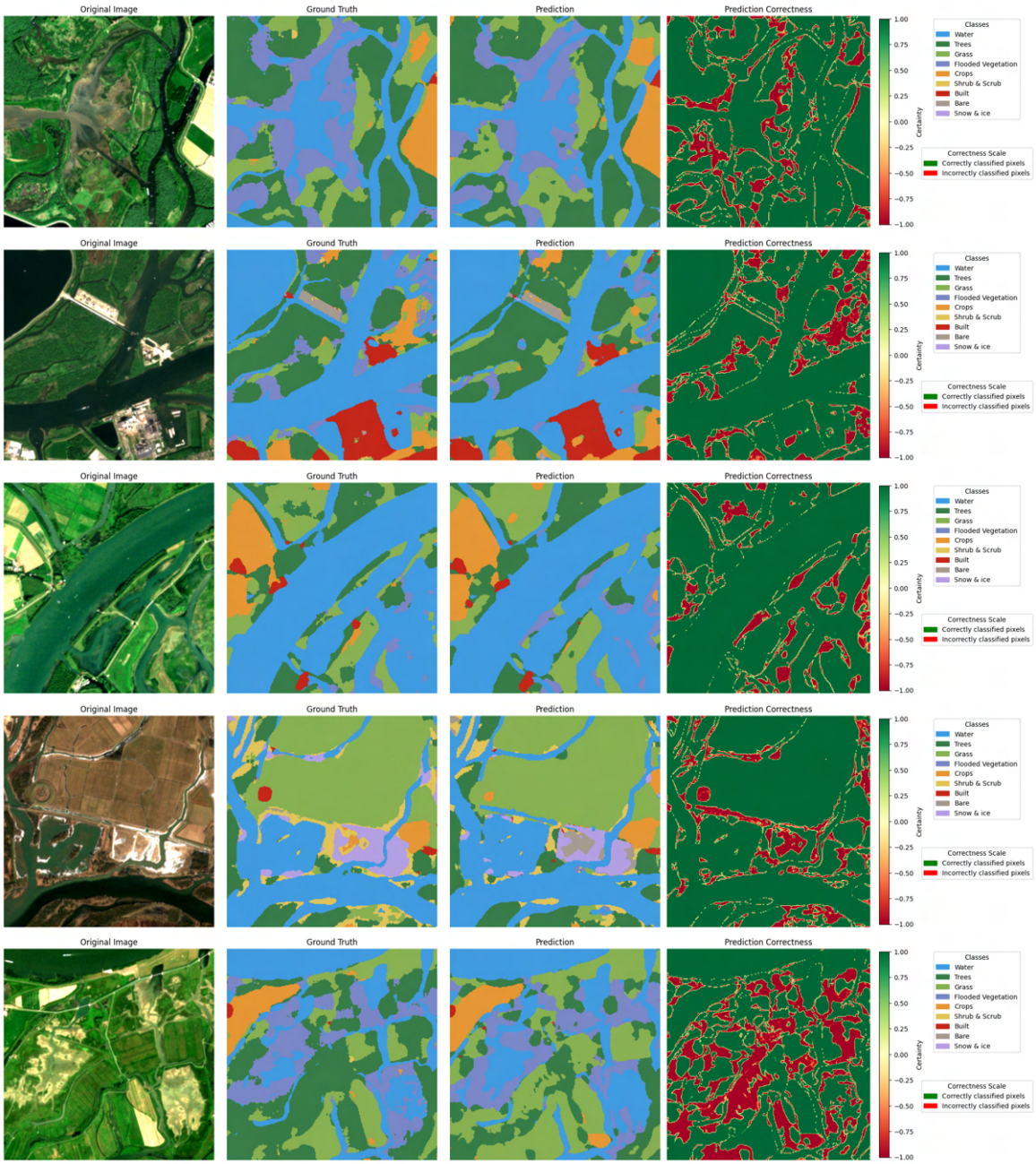


Figure F.1: Baseline U-Net results for the medium-resolution imagery and labels.

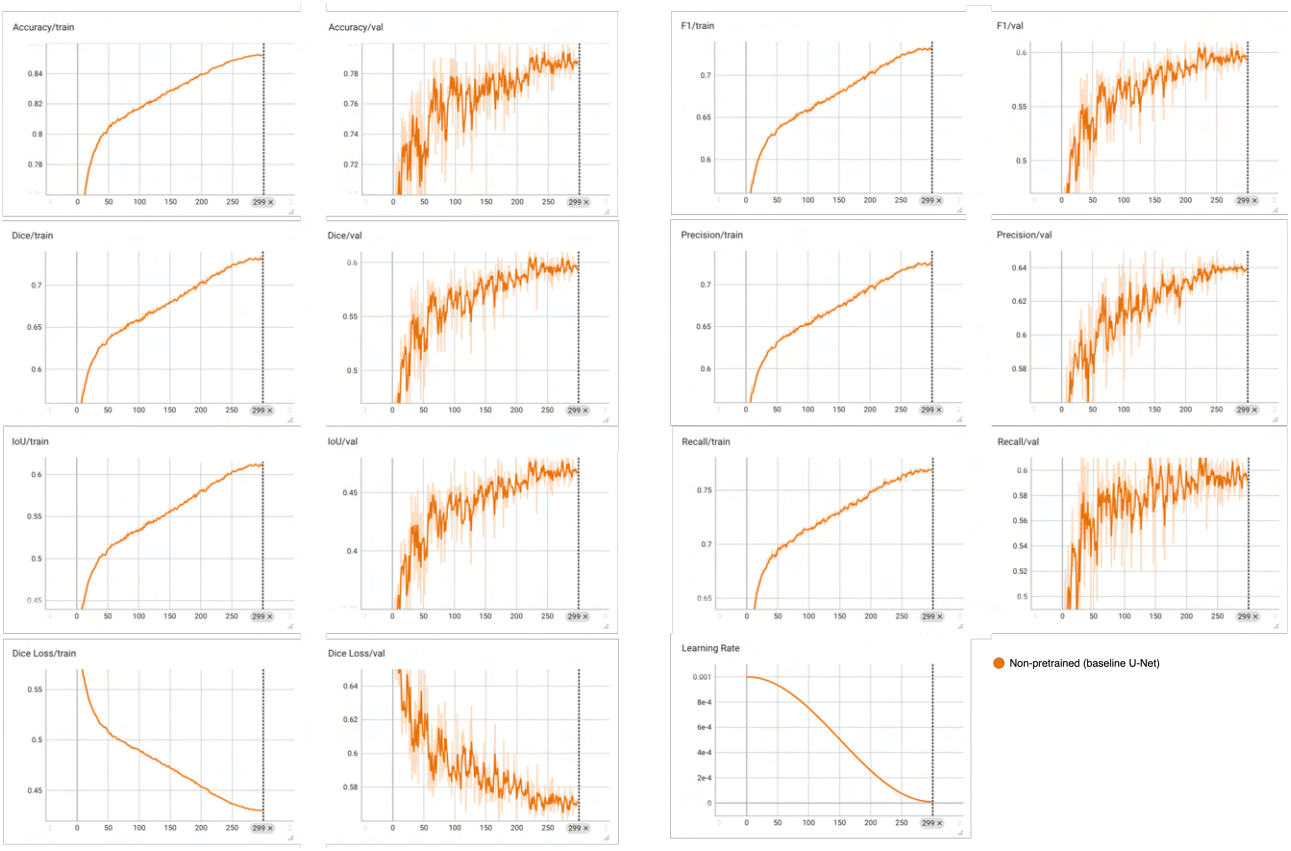


Figure F.2: Performance metrics on training and validation set of baseline U-Net.

Appendix G

Experiment 1

Fraction	**1%**	**10%**	**30%**	**50%**	**70%**	**100%**
Train	17	170	510	850	1190	1701
Val	9	94	284	474	663	948
Test	11	114	342	570	798	1140

Table G.1: Dataset split across different subsets.

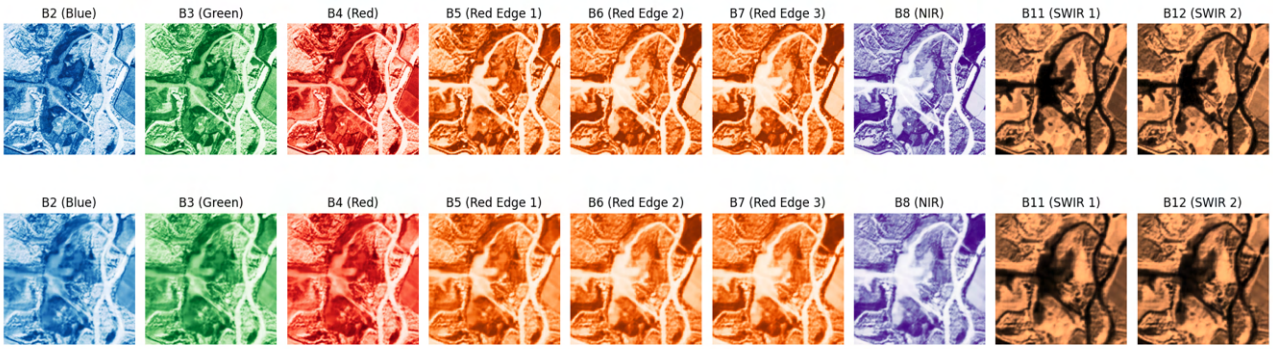


Figure G.1: The reconstruction of the 9 individual spectral bands from Sentinel-2 imagery using an autoencoder. The colors assigned to the bands are for intuitive representation, as the direct visual appearance of the bands differs from natural colors.

Model Type	Fraction	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Dice Loss ↓
Non-pretrained	Train	0.8289	0.6760	0.5583	0.6760	0.6723	0.7047	0.4768
	Val	0.6873	0.4436	0.3343	0.4436	0.5373	0.4747	0.6911
	Test	0.6623	0.4399	0.3306	0.4399	0.4681	0.4766	0.6581
Pretrained	Train	0.8482	0.7126	0.5916	0.7126	0.7021	0.7815	0.4384
	Val	0.7175	0.4620	0.3517	0.4620	0.5839	0.4607	0.6883
	Test	0.7338	0.4670	0.3601	0.4670	0.5040	0.4846	0.6261

Table G.2: U-Net performance on the 1% subset with and without pretraining. The U-Net was trained for 300 epochs with a dropout rate of 0.15, a batch size of 8, and a cosine annealing learning rate schedule from 0.001 to 0.0001. The pretrained model used features extracted from an autoencoder trained on the 100% of the training set for 200 epochs with a mixed loss function ($\alpha = 0.5$, $\beta = 0.4$, $\gamma = 0.1$), a learning rate of 0.0001, and a batch size of 8.

Model Type	Fraction	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Dice Loss ↓
Non-pretrained	Train	0.8459	0.7055	0.5834	0.7055	0.7074	0.7399	0.4433
	Val	0.7613	0.5148	0.3975	0.5148	0.5905	0.5322	0.6298
	Test	0.8201	0.5750	0.4652	0.5750	0.6247	0.5866	0.5349
Pretrained	Train	0.8563	0.7228	0.6042	0.7228	0.7191	0.7479	0.4286
	Val	0.7667	0.5141	0.3986	0.5141	0.5923	0.5229	0.6297
	Test	0.8232	0.5818	0.4710	0.5818	0.6313	0.5835	0.5415

Table G.3: U-Net performance on the 10% subset with and without pretraining. The U-Net was trained for 300 epochs with a dropout rate of 0.15, a batch size of 8, and a cosine annealing learning rate schedule from 0.001 to 0.0001. The pretrained model used features extracted from an autoencoder trained on the 100% of the training set for 200 epochs with a mixed loss function ($\alpha = 0.5$, $\beta = 0.4$, $\gamma = 0.1$), a learning rate of 0.0001, and a batch size of 8.

Model Type	Fraction	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Dice Loss ↓
Non-pretrained	Train	0.8505	0.7320	0.6122	0.7320	0.7262	0.7715	0.4250
	Val	0.7833	0.5818	0.4564	0.5818	0.6347	0.5760	0.5810
	Test	0.8411	0.6416	0.5235	0.6416	0.6616	0.6470	0.5041
Pretrained	Train	0.8588	0.7484	0.6319	0.7484	0.7414	0.7864	0.4098
	Val	0.7897	0.5685	0.4461	0.5685	0.6508	0.5544	0.5865
	Test	0.8446	0.6369	0.5208	0.6369	0.6666	0.6427	0.5099

Table G.4: U-Net performance on the 30% subset with and without pretraining. The U-Net was trained for 300 epochs with a dropout rate of 0.15, a batch size of 8, and a cosine annealing learning rate schedule from 0.001 to 0.0001. The pretrained model used features extracted from an autoencoder trained on the 100% of the training set for 200 epochs with a mixed loss function ($\alpha = 0.5$, $\beta = 0.4$, $\gamma = 0.1$), a learning rate of 0.0001, and a batch size of 8.

Model Type	Fraction	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Dice Loss ↓
Non-pretrained	Train	0.8530	0.7344	0.6161	0.7344	0.7324	0.7625	0.4219
	Val	0.7969	0.6037	0.4776	0.6037	0.6387	0.6091	0.5691
	Test	0.8396	0.6381	0.5199	0.6381	0.6511	0.6536	0.5017
Pretrained	Train	0.8597	0.7533	0.6369	0.7533	0.7467	0.7874	0.4092
	Val	0.7859	0.5927	0.4661	0.5927	0.6435	0.5857	0.5740
	Test	0.8421	0.6435	0.5250	0.6435	0.6683	0.6529	0.5020

Table G.5: U-Net performance on the 50% subset with and without pretraining. The U-Net was trained for 300 epochs with a dropout rate of 0.15, a batch size of 8, and a cosine annealing learning rate schedule from 0.001 to 0.0001. The pretrained model used features extracted from an autoencoder trained on the 100% of the training set for 200 epochs with a mixed loss function ($\alpha = 0.5$, $\beta = 0.4$, $\gamma = 0.1$), a learning rate of 0.0001, and a batch size of 8.

Model Type	Fraction	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Dice Loss ↓
Non-pretrained	Train	0.8551	0.7321	0.6140	0.7321	0.7245	0.7694	0.4239
	Val	0.7849	0.5954	0.4678	0.5954	0.6382	0.5947	0.5735
	Test	0.8459	0.6559	0.5369	0.6559	0.6702	0.6706	0.4932
Pretrained	Train	0.8633	0.7533	0.6369	0.7533	0.7483	0.7835	0.4106
	Val	0.7888	0.5961	0.4692	0.5961	0.6443	0.5933	0.5700
	Test	0.8482	0.6639	0.5448	0.6639	0.6601	0.6984	0.4858

Table G.6: U-Net performance on the 70% subset with and without pretraining. The U-Net was trained for 300 epochs with a dropout rate of 0.15, a batch size of 8, and a cosine annealing learning rate schedule from 0.001 to 0.0001. The pretrained model used features extracted from an autoencoder trained on the 100% of the training set for 200 epochs with a mixed loss function ($\alpha = 0.5$, $\beta = 0.4$, $\gamma = 0.1$), a learning rate of 0.0001, and a batch size of 8.

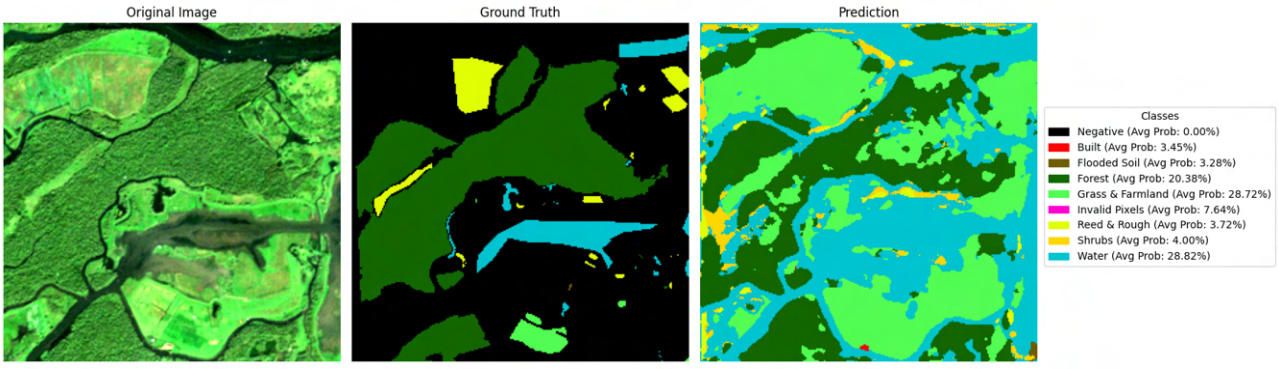
Model Type	Fraction	Accuracy ↑	Dice ↑	IoU ↑	F1 ↑	Precision ↑	Recall ↑	Dice Loss ↓
Non-pretrained	Train	0.8522	0.7287	0.6096	0.7287	0.7226	0.7687	0.4295
	Val	0.7870	0.5940	0.4670	0.5940	0.6412	0.5909	0.5712
	Test	0.8526	0.6480	0.5346	0.6480	0.6616	0.6694	0.4865
Pretrained	Train	0.8565	0.7383	0.6198	0.7383	0.7322	0.7748	0.4233
	Val	0.7956	0.5963	0.4721	0.5963	0.6423	0.5959	0.5647
	Test	0.8542	0.6518	0.5378	0.6518	0.6923	0.6483	0.4905

Table G.7: U-Net performance on the 100% subset with and without pretraining. The U-Net was trained for 300 epochs with a dropout rate of 0.15, a batch size of 8, and a cosine annealing learning rate schedule from 0.001 to 0.0001. The pretrained model used features extracted from an autoencoder trained on the 100% of the training set for 200 epochs with a mixed loss function ($\alpha = 0.5$, $\beta = 0.4$, $\gamma = 0.1$), a learning rate of 0.0001, and a batch size of 8.

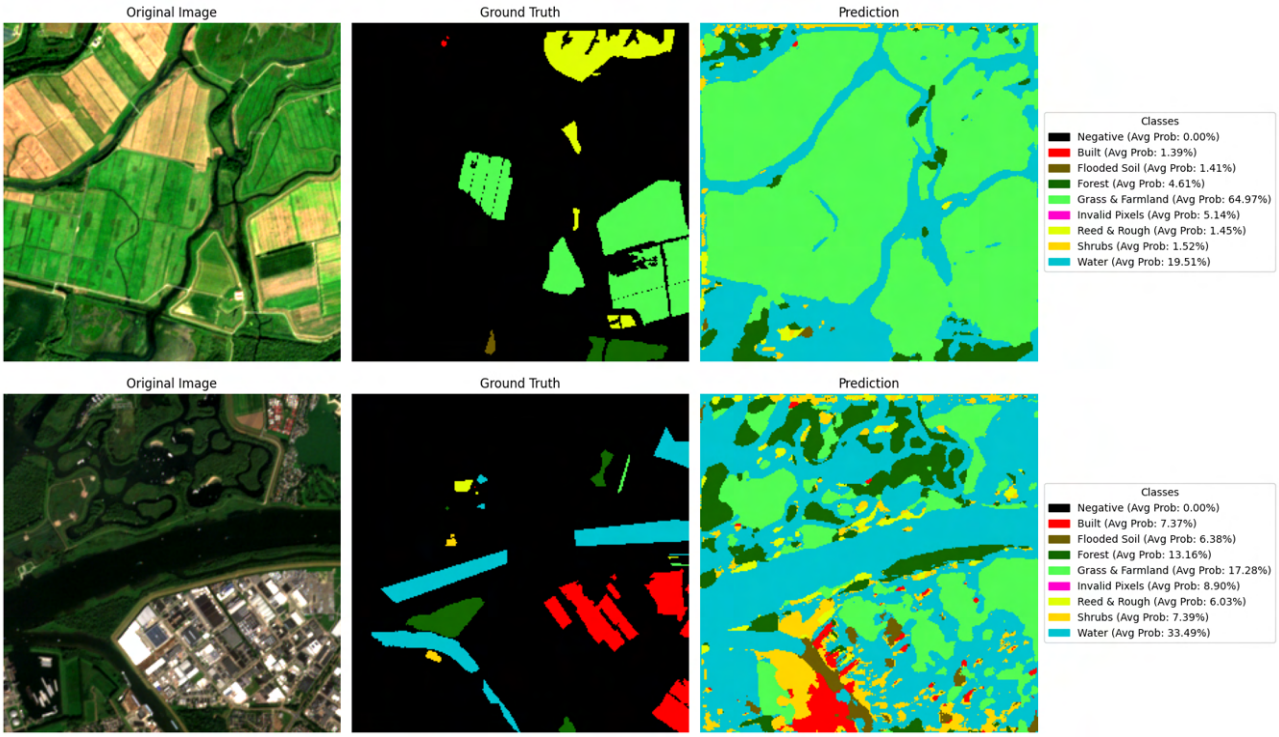
Appendix H

Experiment 2

H.1 Medium-resolution imagery and labels



(a) Test set



(b) Validation set

Figure H.1: Semantic segmentation results on the medium-resolution satellite imagery. The model was trained for 200 epochs with a batch size of 8, as discussed in Chapters 4 and 5.

H.2 VHR imagery and labels

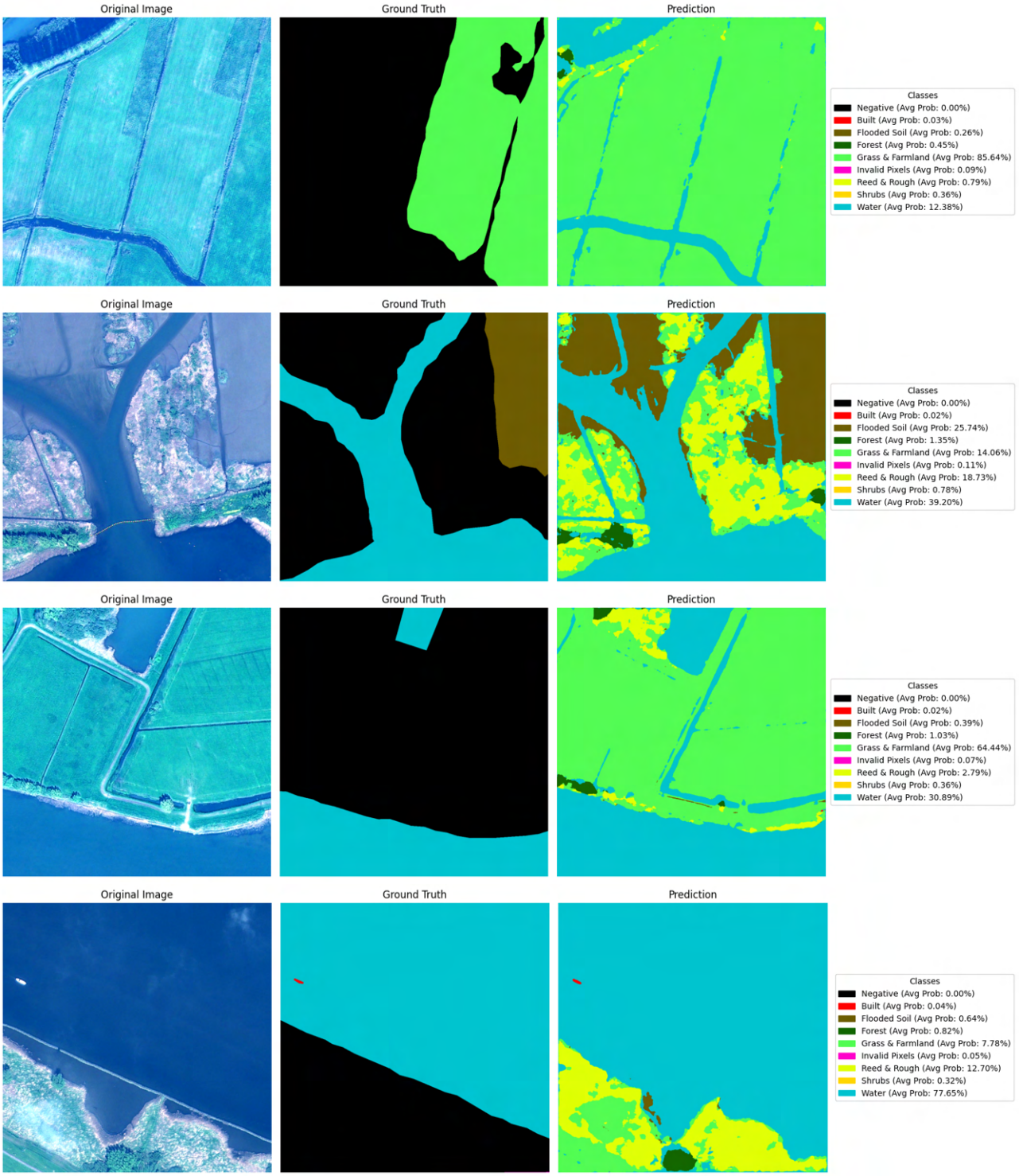


Figure H.2: Semantic segmentation results on the test set of the VHR satellite imagery. The model was trained for 50 epochs with a batch size of 4, as discussed in Chapters 4 and 5.

Appendix I

Workflow Diagrams

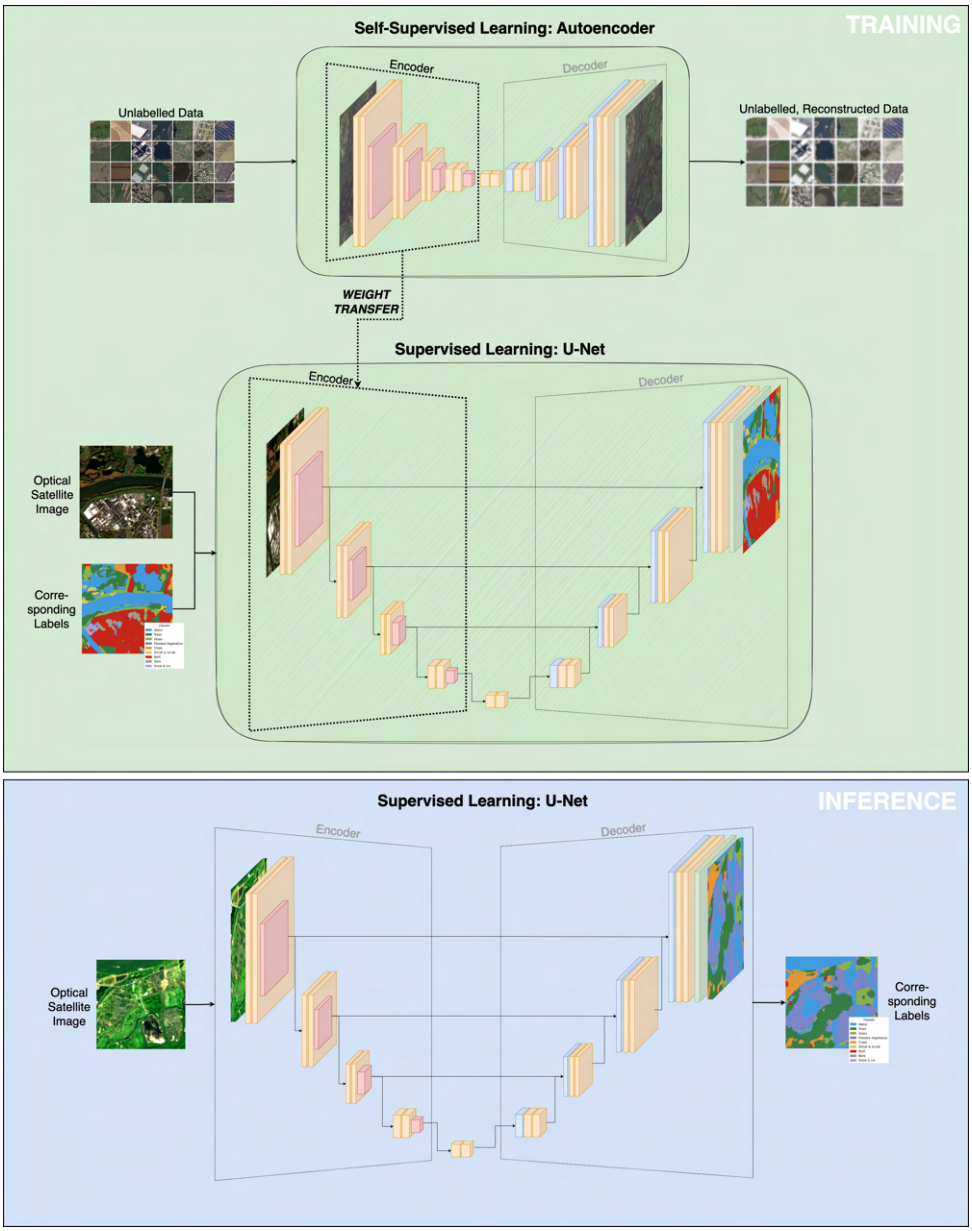


Figure I.1: Replacement of the encoder from the unet with the encoder from the Autoencoder.

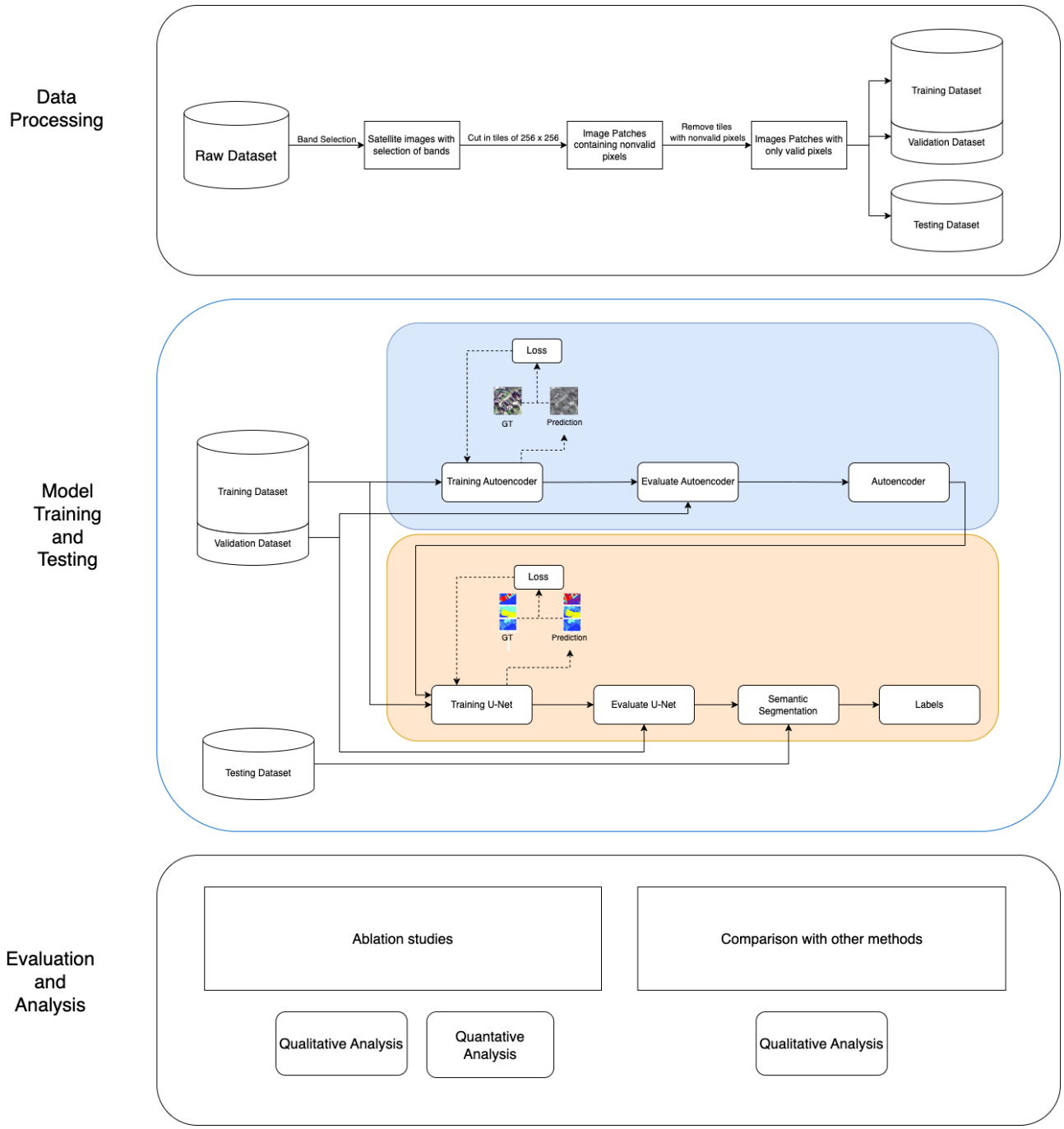


Figure I.2: Workflow diagram of the two architectures, autoencoder and U-Net, used in the research.

Bibliography

[1] M Acreman and J Holden. How wetlands affect floods. *Wetlands*, 33:773–786, 2013.

[2] Elhadi Adam, Onisimo Mutanga, and Denis Rugege. Multispectral and hyperspectral remote sensing for identification and mapping of wetland vegetation: a review. *Wetlands ecology and management*, 18:281–296, 2010.

[3] Adekanmi Adeyinka Adegun, Serestina Viriri, and Jules-Raymond Tapamo. Review of deep learning methods for remote sensing satellite images classification: experimental survey and comparative analysis. *Journal of Big Data*, 10(1):93, 2023.

[4] Nikita Aggarwal, Mohit srivastava, and Maitreyee Dutta. Comparative Analysis of Pixel-Based and Object-Based Classification of High Resolution Remote Sensing Images – A Review. *International Journal of Engineering Trends and Technology*, 38(1):5–11, 2016.

[5] Cathaoir Agnew, Anthony Scanlan, Patrick Denny, Eoin M. Grua, Pepijn Van De Ven, and Ciaran Eising. Annotation Quality vs Quantity for Object Detection and Instance Segmentation. *IEEE Access*, 2024.

[6] Rodrigo Antunes, Luiz Junior, Gilson Costa, Raul Feitosa, Edilson de Souza Bias, Abimael Cereda Junior, Catherine Almeida, Laura E. Cué La Rosa, Patrick Happ, and Leonardo Chiamulera. Leveraging SAR and Optical Remote Sensing for Enhanced Biomass Estimation in the Amazon with Random Forest and XGBoost Models. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume 10, pages 21–27. Copernicus Publications, 2024.

[7] Yu Bai, Yu Zhao, Yajing Shao, Xinrong Zhang, and Xuefeng Yuan. Deep learning in different remote sensing image categories and applications: status and prospects. *International Journal of Remote Sensing*, 43(5):1800–1847, 2022.

[8] Paul Berg, Minh-Tan Pham, and Nicolas Courty. Joint multi-modal self-supervised pre-training in remote sensing: Application to methane source classification. In *IGARSS 2023-2023 IEEE International Geoscience and Remote Sensing Symposium*, pages 6624–6627. IEEE, 2023.

[9] Sudipto Bhowmik. Ecological and economic importance of wetlands and their vulnerability: a review. *Research Anthology on Ecosystem Conservation and Preserving Biodiversity*, pages 11–27, 2022.

[10] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[11] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[12] Christopher F. Brown, Steven P. Brumby, Brookie Guzder-Williams, Tanya Birch, Samantha Brooks Hyde, Joseph Mazzariello, Wanda Czerwinski, Valerie J. Pasquarella, Robert Haertel, Simon Ilyushchenko, Kurt Schwehr, Mikaela Weisse, Fred Stolle, Craig Hanson, Oliver Guinan, Rebecca Moore, and Alexander M. Tait. Dynamic World, Near real-time global 10 m land use land cover mapping. *Scientific Data*, 9(1), 2022.

[13] I. M. Butko, O. I. Golubenko, O. M. Makoveichuk, I. O. Zaitsev, and V. O. Kromkach. Vegetation zone segmentation in multispectral imagery. In *IOP Conference Series: Earth and Environmental Science*, volume 1415. Institute of Physics, 2024.

[14] Chen, Xie, Jingqi Yuan, Readmore Huang, and Li. Research on a real-time monitoring method for the wear state of a tool based on a convolutional bidirectional lstm model. *Symmetry*, 11:1233, 2019.

[15] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[17] Zhenlin Chen, Sahar H. El Abbadi, Evan D. Sherwin, Philippine M. Burdeau, Jeffrey S. Rutherford, Yuanlei Chen, Zhan Zhang, and Adam R. Brandt. Comparing Continuous Methane Monitoring Technologies for High-Volume Emissions: A Single-Blind Controlled Release Study. *ACS ES&T Air*, 1(6):657–670, 2024.

[18] Gong Cheng, Xingxing Xie, Junwei Han, Lei Guo, and Gui Song Xia. Remote Sensing Image Scene Classification Meets Deep Learning: Challenges, Methods, Benchmarks, and Opportunities. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:3735–3756, 2020.

[19] Yezhen Cong, Samar Khanna, Chenlin Meng, Patrick Liu, Erik Rozi, Yutong He, Marshall Burke, David Lobell, and Stefano Ermon. Satmae: Pre-training transformers for temporal and multi-spectral satellite imagery. *Advances in Neural Information Processing Systems*, 35:197–211, 2022.

[20] Raghav Dahiya, Manish Kumar Ojha, Shikhar Saini, and Sanatan Ratna. Satellite Image Segmentation Using U-Net. In *2024 15th International Conference on Computing Communication and Networking Technologies, ICCCNT 2024*. Institute of Electrical and Electronics Engineers Inc., 2024.

[21] Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[22] H Duel, MJ Baptist, GJ Geerling, AJM Smits, and J Van Alphen. Cyclic floodplain rejuvenation as a strategy for both flood protection and enhancement of the biodiversity of the river rhine. *4th Ecohydraulics*, pages 1–24, 2002.

[23] Charlotte Marie Emery, Kevin Larnier, Maxime Liquet, João Hemptinne, Arthur Vincent, and Santiago Peña Luque. Extraction of roughness parameters from remotely-sensed products for hydrology applications. *Hydrology and Earth System Sciences Discussions*, 2021:1–40, 2021.

[24] Sana Fatima, Ayan Hussain, Sohaib Bin Amir, Syed Haseeb Ahmed, and Syed Muhammad Huzaifa Aslam. Xgboost and random forest algorithms: an in depth analysis. *Pakistan Journal of Scientific Research*, 3(1):26–31, 2023.

[25] Mohamed Fawzy and Arpad Barsi. A U-Net Model for Urban Land Cover Classification Using VHR Satellite Images. *Periodica Polytechnica Civil Engineering*, 69(1):98–108, 2024.

[26] Valerie Fernandez, Philippe Martimort, Francois Spoto, Omar Sy, and Paolo Laberinti. Overview of sentinel-2. In *Sensors, Systems, and Next-Generation Satellites XVII*, volume 8889, pages 97–102. SPIE, 2013.

[27] Anthony Fuller, Koreen Millard, and James Green. Croma: Remote sensing representations with contrastive radar-optical masked autoencoders. *Advances in Neural Information Processing Systems*, 36, 2024.

[28] Gianluca Furano, Gabriele Meoni, Aubrey Dunne, David Moloney, Veronique Ferlet-Cavrois, Antonis Tavoularis, Jonathan Byrne, Léonie Buckley, Mihalis Psarakis, Kay-Obbe Voss, et al. Towards the use of artificial intelligence on the edge in space systems: Challenges and opportunities. *IEEE Aerospace and Electronic Systems Magazine*, 35(12):44–56, 2020.

[29] Alisa L. Gallant. The challenges of remote monitoring of wetlands. *Remote Sensing*, 7(8):10938–10950, 2015.

[30] Dunia Gonzales, Natalie Hempel de Ibarra, and Karen Anderson. Remote Sensing of Floral Resources for Pollinators – New Horizons From Satellites to Drones. *Frontiers in Ecology and Evolution*, 10, 2022.

[31] Zhujun Gu and Maimai Zeng. The use of artificial intelligence and satellite remote sensing in land cover change detection: review and perspectives. *Sustainability*, 16(1):274, 2023.

[32] Zhujun Gu and Maimai Zeng. The use of artificial intelligence and satellite remote sensing in land cover change detection: Review and perspectives. *Sustainability*, 16(1):274, 2024.

[33] J Anthony Gualtieri and Robert F Cromp. Support vector machines for hyperspectral remote sensing classification. In *27th AIPR workshop: Advances in computer-assisted recognition*, volume 3584, pages 221–232. SPIE, 1999.

[34] Meng Guo, Jing Li, Chunlei Sheng, Jiawei Xu, and Li Wu. A review of wetland remote sensing. *Sensors*, 17(4):777, 2017.

[35] Diaa Hafez Ibrahim, Reda A El-khoribi, and Farid Ali Mousa. A Novel Deep Learning Method for Detecting Changes in Satellite Imagery. *Journal of Theoretical and Applied Information Technology*, 15(1), 2025.

[36] A. L. Hakstege, S. B. Kroonenberg, and H. Van Wijck. Geochemistry of holocene clays of the rhine and meuse rivers in the central-eastern netherlands. *Geologie en Mijnbouw*, 71:301–315, 1993.

[37] Herlawati and Rahmadya Trias Handayanto. Land Cover Segmentation of Multispectral Images Using U-Net and DeeplabV3+ Architecture. *Jurnal Ilmu Komputer dan Informasi*, 17(1):89–96, 2024.

[38] Yassine Himeur, Bhagawat Rimal, Abhishek Tiwary, and Abbes Amira. Using artificial intelligence and data fusion for environmental monitoring: A review and future perspectives. *Information Fusion*, 86-87:44–75, 2022.

[39] C. Huang, L. S. Davis, and J. R.G. Townshend. An assessment of support vector machines for land cover classification. *International Journal of Remote Sensing*, 23(4):725–749, 2002.

[40] Peter J. Huber. Robust Estimation of a Location Parameter. *The Annals of mathematical statistics,*, 1963.

[41] Paul Jaccard. The Distribution of the Flora in the Alpine Zone. *New Phytologist*, 11(2):37–50, 1912.

[42] Hamid Jafarzadeh, Masoud Mahdianpari, Eric W. Gill, Brian Brisco, and Fariba Mohammadimanesh. Remote Sensing and Machine Learning Tools to Support Wetland Monitoring: A Meta-Analysis of Three Decades of Research. *Remote Sensing*, 14(23), 2022.

[43] Bhargavi Janga, Gokul Prathin Asamani, Ziheng Sun, and Nicoleta Cristea. A Review of Practical AI for Remote Sensing in Earth Sciences. *Remote Sensing*, 15(16), 2023.

[44] Juha Järvelä. Influence of vegetation on flow structure in floodplains and wetlands. In *IAHR Congress RCEM Conf. Proc*, pages 845–856, 2003.

[45] Neha Joshi, Matthias Baumann, Andrea Ehammer, Rasmus Fensholt, Kenneth Grogan, Patrick Hostert, Martin Rudbeck Jepsen, Tobias Kuemmerle, Patrick Meyfroidt, Edward TA Mitchard, et al. A review of the application of optical and radar remote sensing data fusion to land use mapping and monitoring. *Remote Sensing*, 8(1):70, 2016.

[46] Rdvan Salih Kuzu, Oleg Antropov, Matthieu Molinier, Corneliu Octavian Dumitru, Sudipan Saha, and Xiao Xiang Zhu. Forest Disturbance Detection via Self-Supervised and Transfer Learning With Sentinel-1&2 Images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 17:4751–4767, 2024.

[47] Zheng Li, Yongcheng Wang, Ning Zhang, Yuxi Zhang, Zhikang Zhao, Dongdong Xu, Guangli Ben, and Yunxiao Gao. Deep learning-based object detection techniques for remote sensing images: A survey. *Remote Sensing*, 14(10):2385, 2022.

[48] Wei Liu, Andrew Rabinovich, and Alexander C. Berg. Parsenet: Looking wider to see better. *CoRR*, abs/1506.04579, 2015.

[49] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[50] Siqi Lu, Junlin Guo, James R Zimmer-Dauphinee, Jordan M Nieusma, Xiao Wang, Parker VanValkenburgh, Steven A Wernke, and Yuankai Huo. Ai foundation models in remote sensing: A survey. *arXiv preprint arXiv:2408.03464*, 2024.

[51] Lucrêncio Silvestre Macarringue, Édson Luis Bolfe, and Paulo Roberto Mendes Pereira. Developments in Land Use and Land Cover Classification Techniques in Remote Sensing: A Review. *Journal of Geographic Information System*, 14(01):1–28, 2022.

[52] Sahel Mahdavi, Bahram Salehi, Jean Granger, Meisam Amani, Brian Brisco, and Weimin Huang. Remote sensing for wetland classification: a comprehensive review. *GIScience and Remote Sensing*, 55(5):623–658, 2018.

[53] Kumar Mainali, Michael Evans, David Saavedra, Emily Mills, Becca Madsen, and Susan Minnemeyer. Convolutional neural network for high-resolution wetland mapping with open data: Variable selection and the challenges of a generalizable model. *Science of the Total Environment*, 861, 2023.

[54] Ujjwal Maulik and Debasis Chakraborty. Remote Sensing Image Classification: A survey of support-vector-machine-based advanced techniques. *IEEE Geoscience and Remote Sensing Magazine*, 5(1):33–52, 2017.

[55] Niti B. Mishra. Wetlands: Remote Sensing. In *Wetlands and Habitats*, pages 201–212. CRC Press, 2020.

[56] William J. Mitsch, Blanca Bernal, Amanda M. Nahlik, Ülo Mander, Li Zhang, Christopher J. Anderson, Sven E. Jørgensen, and Hans Brix. Wetlands, carbon, and climate change. *Landscape Ecology*, 28(4):583–597, 2013.

[57] Alberto Moreira, Pau Prats-Iraola, Marwan Younis, Gerhard Krieger, Irena Hajnsek, and Konstantinos P. Papathanassiou. A tutorial on synthetic aperture radar. *IEEE Geoscience and Remote Sensing Magazine*, 1(1):6–43, 2013.

[58] Jens Nieke, Laurent Despoisse, Antonio Gabriele, Heidrun Weber, Helene Strese, Nafiseh Ghasemi, Ferran Gascon, Kevin Alonso, Valentina Boccia, Bogdana Tsonevska, et al. The copernicus hyperspectral imaging mission for the environment (chime): an overview of its mission, system and planning status. *Sensors, Systems, and Next-Generation Satellites XXVII*, 12729:21–40, 2023.

[59] Mubashir Noman, Muzammal Naseer, Hisham Cholakkal, Rao Muhammad Anwer, Salman Khan, and Fahad Shahbaz Khan. Rethinking transformers pre-training for multi-spectral satellite imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27811–27819, 2024.

[60] M. Pal and P. M. Mather. Support vector machines for classification in remote sensing. *International Journal of Remote Sensing*, 26(5):1007–1011, 2005.

[61] Fernando Pech-May, Raul Aquino-Santos, Omar Alvarez-Cardenas, Jorge Lozoya Arandia, and German Rios-Toledo. Segmentation and Visualization of Flooded Areas Through Sentinel-1 Images and U-Net. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 17:8996–9008, 2024.

[62] Gustav Grund Pihlgren, Fredrik Sandin, and Marcus Liwicki. Pretraining image encoders without reconstruction via feature prediction loss. In *2020 25th international conference on pattern recognition (ICPR)*, pages 4105–4111. IEEE, 2021.

[63] Hampapuram K Ramapriyan, Jeanne Behnke, Edwin Sofinowski, Dawn Lowe, and Mary Ann Esfandiari. Evolution of the earth observing system (eos) data and information system (eosdis). In *Standard-based data and Information systems for Earth observation*, pages 63–92. Springer, 2009.

[64] Gareth Rees. *Physical principles of remote sensing*. Cambridge university press, 2013.

[65] Ekram M Rewhel, Jianqiang Li, Amal A Hamed, Hatem M Keshk, Amira S Mahmoud, Sayed A Sayed, Ehab Samir, Hind H Zeyada, Sayed A Mohamed, Marwa S Moustafa, et al. Deep learning methods used in remote sensing images: A review. *Journal of Environmental & Earth Sciences*, 5(1):33–64, 2023.

[66] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[67] Anastasiia Safonova, Gohar Ghazaryan, Stefan Stiller, Magdalena Main-Knorn, Claas Nendel, and Masahiro Ryo. Ten deep learning techniques to address small data problems with remote sensing. *International Journal of Applied Earth Observation and Geoinformation*, 125:103569, 2023.

[68] Shokoufeh Salimi, Suhad AAAN Almuktar, and Miklas Scholz. Impact of climate change on wetland ecosystems: A critical review of experimental wetlands. *Journal of Environmental Management*, 286:112160, 2021.

[69] Linus Scheibenreif, Joelle Hanna, Michael Mommert, and Damian Borth. Self-supervised vision transformers for land-cover segmentation and classification. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1421–1430. IEEE, 2022.

[70] Mohammadreza Sheykhmousa, Masoud Mahdianpari, Hamid Ghanbari, Fariba Mohammadimanesh, Pedram Ghamisi, and Saeid Homayouni. Support vector machine versus random forest for remote sensing image classification: A meta-analysis and systematic review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:6308–6325, 2020.

[71] Anita Simic Milas, Arthur P. Cracknell, and Timothy A. Warner. Drones–the third generation source of remote sensing data. *International Journal of Remote Sensing*, 39(21):7125–7137, 2018.

[72] Bart Slagter, Nandin Erdene Tsendbazar, Andreas Vollrath, and Johannes Reiche. Mapping wetland characteristics using temporally dense Sentinel-1 and Sentinel-2 data: A case study in the St. Lucia wetlands, South Africa. *International Journal of Applied Earth Observation and Geoinformation*, 86, 2020.

[73] Derk Jan Stobbelaara and Jack CM Schoenmakersa. The influence of the vegetation structure on the water flow through the Noordwaard (Brabant, The Netherlands). *NCR DAYS 2018*, page 118, 2018.

[74] Anastasios Tzepkenlis, Konstantinos Marthoglou, and Nikos Grammalidis. Efficient deep semantic segmentation for land cover classification using sentinel imagery. *Remote Sensing*, 15(8):2027, 2023.

[75] Eveline C. van der Deijl, Marcel van der Perk, and Hans Middelkoop. Factors controlling sediment trapping in two freshwater tidal wetlands in the Biesbosch area, The Netherlands. *Journal of Soils and Sediments*, 17(11):2620–2636, 2017.

[76] Eveline C. van der Deijl, Marcel van der Perk, and Hans Middelkoop. Pathways of Water and Sediment in the Biesbosch Freshwater Tidal Wetland. *Wetlands*, 39(1):197–215, 2019.

[77] Anders U. Waldeland, Arnt Borre Salberg, Oivind D. Trier, and Andreas Vollrath. Large-Scale Vegetation Height Mapping from Sentinel Data Using Deep Learning. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 1877–1880. Institute of Electrical and Electronics Engineers Inc., 2020.

[78] Anders U. Waldeland, Øivind Due Trier, and Arnt Børre Salberg. Forest mapping and monitoring in Africa using Sentinel-2 data and deep learning. *International Journal of Applied Earth Observation and Geoinformation*, 111, 2022.

[79] Yi Wang, Nassim Ait Ali Braham, Zhitong Xiong, Chenying Liu, Conrad M Albrecht, and Xiao Xiang Zhu. Ssl4eo-s12: A large-scale multimodal, multitemporal dataset for self-supervised learning in earth observation [software and data sets]. *IEEE Geoscience and Remote Sensing Magazine*, 11(3):98–106, 2023.

[80] Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[81] Michael A Wulder, David P Roy, Volker C Radeloff, Thomas R Loveland, Martha C Anderson, David M Johnson, Sean Healey, Zhe Zhu, Theodore A Scambos, Nima Pahlevan, et al. Fifty years of landsat science and impacts. *Remote Sensing of Environment*, 280:113195, 2022.

[82] Qikai Lu Huangfeng Shen Shengyang Li Shucheng You Liangpei Zhang Xin-Yi Tong, Gui-Song Xia. Land-cover classification with high-resolution remote sensing images using transferable deep models. *Remote Sensing of Environment, doi: 10.1016/j.rse.2019.111322*, 2020.

[83] Zhixiang Xue, Xuchu Yu, Anzhu Yu, Bing Liu, Pengqiang Zhang, and Shentong Wu. Self-Supervised Feature Learning for Multimodal Remote Sensing Image Land Cover Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 60, 2022.

[84] Xiaohui Yuan, Jianfang Shi, and Lichuan Gu. A review of deep learning methods for semantic segmentation of remote sensing imagery. *Expert Systems with Applications*, 169:114417, 2021.