МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ» ФАКУЛЬТЕТ ІНФОРМАТИКИ КАФЕДРА ІНФОРМАТИКИ

Кваліфікаційна робота

освітній ступінь – бакалавр

на тему: **Normalization as a Key Enabler for Transferable Machine Learning in Multi-Temporal Cross-Dataset Satellite Imagery: Evidence in Cloud Detection**

Комп’ютерні науки - 122

Керівник: _____

_____ (підпис)

“____” _____ 2025.

Полякова Любов Василівна

“____” _____ 2025.

Київ 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ» ФАКУЛЬТЕТ ІНФОРМАТИКИ КАФЕДРА ІНФОРМАТИКИ

ЗАТВЕРДЖУЮ

Завідувач кафедри інформатики

доцент, кандидат наук.

Гороховський С. С.

«___» _____ 2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на кваліфікаційну роботу студентці

Поляковій Любові Василівні

факультету інформатики

4 курсу бакалаврської програми

Тема: «Normalization as a Key Enabler for Transferable Machine Learning in Multi-Temporal Cross-Dataset Satellite Imagery: Evidence in Cloud Detection»

Зміст ТЧ до кваліфікаційної роботи:

1. Introduction

2. Background and literature review

3. Materials and methods

4. Results

5. Conclusions

References

Додатки (за необхідністю)

Дата видачі «___» _____ 2025 р.

Керівник _____

Завдання отримав _____

Графік підготовки кваліфікаційної роботи до захисту

Графік узгоджено «___» _____ 2025 р.

№	Назва етапу кваліфікаційної роботи	Термін виконання етапу	Підпис наукового керівника	Дата ознайомлення наукового керівника	Приміт ка
1	Отримання теми кваліфікаційної роботи	30.10.2024			
2	Ознайомлення з темою кваліфікаційної роботи	01.11.2024 – 07.11.2024			
3	Розробка плану роботи	08.11.2024 – 14.11.2024			
4	Написання вступу та огляду літератури	15.11.2024 – 15.12.2024			
5	Опис матеріалів та методів	16.12.2024 – 15.01.2025			
6	Проведення експериментів та збір результатів	16.01.2025 – 15.03.2025			
7	Аналіз результатів, написання висновків	16.03.2025 – 15.04.2025			
8	Оформлення роботи (текст, ілюстрації, список джерел)	16.04.2025 – 30.04.2025			
9	Попередній захист кваліфікаційної роботи.	28.05.2025			
10	Захист кваліфікаційної роботи.	05.06.2025			

Contents

1. INTRODUCTION

A common and persistent challenge in remote sensing and computer vision is the difficulty of ensuring that models trained on specific datasets generalize effectively to unseen or differing datasets. Often, models exhibit robust performance on test images from the same dataset and temporal context used during training but struggle significantly when evaluated against images from different temporal periods or acquired from different sources. This problem is particularly acute in satellite imagery tasks, such as cloud detection, where changes in atmospheric conditions, illumination, sensor characteristics, and geographic location introduce substantial variability.

Transferability of machine learning models - specially convolutional neural networks used in semantic segmentation tasks - is crucial for practical applications. The conventional approach to improving generalization involves retraining or fine-tuning models with additional labeled data. However, this approach is often costly, both in terms of computational resources and time. Hence, there is a clear need for methods that enhance model generalization without the recurring overhead of fine-tuning.

Normalization and standardization techniques offer potential solutions to address this challenge. These preprocessing methods adjust the spectral and radiometric characteristics of imagery to a consistent scale, mitigating discrepancies arising from varied temporal, atmospheric, and sensor conditions. By standardizing input data, models can learn representations that are more robust and less sensitive to dataset-specific features.

In this thesis, we explore the use of normalization and standardization to improve the transferability of deep learning models for cloud detection from multi-temporal satellite imagery. Specifically, we evaluate whether applying normalization techniques during preprocessing can reduce the necessity of model fine-tuning when encountering temporally shifted and externally sourced satellite images. Through experimental evaluation, we aim to demonstrate that proper normalization significantly enhances

model robustness and generalization, offering practical benefits in terms of efficiency and scalability in remote sensing applications.

2. BACKGROUND AND LITERATURE REVIEW

2.1 Remote Sensing

Remote sensing is the science and technology of obtaining information about objects or areas from a distance, typically through sensors placed on satellites, aircraft, or drones, without direct physical contact with the target being studied [1]. It involves detecting and measuring electromagnetic radiation emitted or reflected by the Earth's surface or atmosphere, and then processing and analyzing this information to monitor, characterize, and understand various environmental phenomena and physical processes [2].

This technology provides critical insights in diverse applications, including land-use mapping, agriculture, forestry, geology, hydrology, meteorology, oceanography, disaster management, and climate change monitoring [3]. It utilizes various platforms and sensors, such as multispectral and hyperspectral imagers, thermal infrared sensors, Synthetic Aperture Radar (SAR), and LiDAR (Light Detection and Ranging), each suitable for specific types of analyses based on spatial, spectral, radiometric, and temporal resolution [3][4].

2.1.1 Electromagnetic spectrum in remote sensing

At its core, remote sensing relies on detecting, measuring, and analyzing electromagnetic radiation across various wavelengths to extract meaningful information about Earth's surface and atmosphere. The electromagnetic (EM) spectrum encompasses the entire range of electromagnetic radiation, extending from short-wavelength gamma rays to long-wavelength radio waves [2].

Several regions of the EM spectrum are typically utilized in remote sensing, including visible (0.4–0.7 μm), near-infrared (NIR; 0.7-1.3 μm), shortwave infrared (SWIR; 1.3-3 μm), thermal infrared (TIR; 3-14 μm), and microwave (1 mm-1 m)

wavelengths [3]. Each spectral region provides unique information based on how electromagnetic radiation interacts differently with matter through absorption, reflection, emission, transmission, and scattering processes [4].

- Visible Spectrum (0.4-0.7 μm): Primarily used for applications such as vegetation mapping, land-use classification, and detecting surface features due to distinct reflectance characteristics in visible wavelengths [5].
- Near-Infrared (NIR, 0.7-1.3 μm): Highly sensitive to vegetation health and biomass, as chlorophyll-rich vegetation reflects strongly in this spectral region, making NIR valuable in agriculture and forestry studies [1].
- Shortwave Infrared (SWIR, 1.3-3 μm): Useful in distinguishing mineralogy, moisture content, and vegetation stress due to distinct absorption features in soils, rocks, and vegetation at these wavelengths [2].
- Thermal Infrared (TIR, 3-14 μm): Enables the detection of emitted radiation from Earth's surface, useful in monitoring surface temperature, heat stress in vegetation, urban heat islands, geological studies, and hydrological applications [3].
- Microwave Spectrum (1 mm-1 m): Capable of penetrating clouds, vegetation, and even surface layers, making it particularly useful in all-weather monitoring, soil moisture mapping, ice cover analysis, and terrain elevation models through radar imaging technologies like Synthetic Aperture Radar (SAR) [4].

Understanding the electromagnetic spectrum's characteristics, along with sensor design and spectral response, is fundamental to selecting appropriate remote sensing systems and methods for specific applications [1][3].

2.1.2. Spectral bands used for cloud detection

Cloud detection using remote sensing involves distinguishing clouds from the Earth's surface by exploiting the reflective, absorptive, and emissive properties of clouds across different spectral bands of the electromagnetic spectrum [6]. Accurate cloud identification is essential for weather forecasting, climate studies, and satellite image interpretation.

Several key spectral bands are widely utilized for cloud detection:

1. Visible Spectrum (0.4-0.7 μm)

Visible bands, particularly at approximately 0.6 μm, are effectively used to detect clouds due to their high albedo compared to land and water surfaces. Brightness differences allow for cloud identification in daytime imagery; however, these bands become ineffective during nighttime [6] [8]

2. Near-Infrared (NIR, 0.8-1.3 μm)

NIR spectral bands, specifically around 0.8 μm and 1.3 μm, are sensitive to cloud optical thickness and particle size, facilitating distinction between clouds and snow or ice-covered surfaces [7]. Additionally, they help discriminate clouds from vegetation, which exhibits high reflectance in the NIR region [1].

3. Shortwave Infrared (SWIR, 1.6–2.2 μm)

SWIR bands near 1.6 μm and 2.2 μm are frequently used for cloud phase discrimination - separating water clouds from ice clouds - due to distinct absorption differences between ice and liquid water at these wavelengths [6][7]

4. Thermal Infrared (TIR, 8-14 μm)

Thermal infrared bands are vital for cloud detection and classification both day and night because clouds emit radiation differently from the Earth's surface, typically exhibiting lower brightness temperatures. Channels around 10.8 μm and 12 μm are particularly useful for identifying cloud-top heights and distinguishing clouds from land and sea surfaces due to their distinct emissivity and temperature contrast [2][8].

5. Water Vapor Absorption Bands (~6.7 μm)

The water vapor absorption band centered near 6.7 μm is effective in detecting upper tropospheric clouds and moisture, especially cirrus clouds, which significantly absorb and emit radiation at these wavelengths, aiding in high-altitude cloud detection [9].

Combining observations from these spectral bands into multispectral or multisensor approaches significantly enhances cloud detection accuracy. Algorithms frequently utilize threshold-based and machine learning techniques to exploit spectral differences for cloud identification and classification [6][7].

2.2 Machine Learning for Cloud Detection in Remote Sensing

2.2.1 Overview of cloud detection challenges in satellite imagery

Cloud detection in satellite imagery is a critical yet complex task in atmospheric science, remote sensing, and Earth observation. Accurate cloud identification is essential for applications such as land surface monitoring, climate modeling, weather forecasting, and aerosol retrievals. However, numerous challenges arise due to the variable nature of clouds, sensor limitations, and atmospheric conditions.

1. Spectral Similarity Between Clouds and Bright Surfaces

One of the primary challenges in cloud detection is the spectral similarity between clouds and bright surfaces, such as snow, ice, and desert regions. These surfaces often exhibit high reflectance in visible and near-infrared (NIR) wavelengths, making it difficult to distinguish them from clouds using simple thresholding techniques [6][7]. For instance, thin cirrus clouds may appear spectrally similar to snow in the visible spectrum, necessitating the use of shortwave infrared (SWIR) or thermal infrared (TIR) channels for improved discrimination.

2. Thin and Subpixel Clouds

Thin clouds, especially cirrus clouds, often lack sufficient optical thickness to be easily detected by traditional algorithms. These clouds may partially obscure the surface or go undetected in lower-resolution imagery. Additionally, subpixel clouds, which occupy only a fraction of a pixel, may be misclassified or missed entirely due to the limitations of spatial resolution and mixed pixel effects [7] [9].

3. Cloud Edge and Partly Cloudy Pixels

The detection of cloud edges and partially cloudy pixels is especially problematic because these areas often deviate from the assumptions of homogeneity used in retrieval algorithms. Misclassification can result from sharp reflectance gradients or heterogeneous cloud fields, leading to inaccurate cloud masks or retrieval artifacts [7].

4. Multilayer and Overlapping Clouds

Satellites observing the Earth in a nadir view often struggle to identify multilayer cloud systems. When a thin cirrus cloud layer overlays a lower cumulus layer, the reflected or emitted signal may represent a mixture, complicating the retrieval of accurate cloud properties [10] [11]. This poses a significant challenge to both passive and active remote sensing methods.

5. Atmospheric and Surface Contaminants

Atmospheric aerosols (e.g., smoke, dust) and surface phenomena such as sunglint can be falsely identified as clouds. These false positives are particularly common in optical sensors that rely heavily on reflectance data. Algorithms such as the Clear Sky Restoral (CSR) mechanism in MODIS Collection 6 aim to reduce such errors by employing additional spectral and spatial information [7].

6. Sensor and Algorithm Limitations

Variability in sensor spectral capabilities, viewing angles, and calibration uncertainties further complicates cloud detection. Algorithms developed for one sensor (e.g., MODIS) may not perform equally well on others (e.g., VIIRS or AVHRR), necessitating sensor-specific tuning and validation [6][9].

Despite significant advances in algorithm development and multispectral imaging, cloud detection in satellite imagery remains a challenging task. Addressing these issues requires continued refinement of detection algorithms, incorporation of ancillary data, and use of synergistic approaches combining passive and active sensors.

2.2.2 Importance of accurate cloud segmentation

Accurate cloud segmentation is crucial for reliable remote sensing applications, particularly in climate studies, weather forecasting, and environmental monitoring. Precise segmentation ensures correct retrieval of cloud optical and microphysical properties, such as cloud optical thickness, particle size, and cloud water path, all critical parameters influencing climate modeling and atmospheric studies. Inaccurate segmentation can significantly degrade the quality of derived satellite products, leading to errors in interpreting atmospheric and surface conditions, which subsequently impact environmental decision-making processes. Furthermore, accurate cloud segmentation aids in distinguishing clouds from other atmospheric phenomena such as aerosols, smoke, and sunglint, thus improving the overall reliability of satellite-derived data products [7].

2.2.3 Multi-temporal and seasonal image variation

Satellite imagery collected over different time periods and seasons is subject to significant natural variability due to changes in illumination geometry, atmospheric conditions, surface phenology, and land cover status. These multi-temporal and seasonal variations present both challenges and opportunities in remote sensing analysis.

Natural seasonal processes such as vegetation cycles, snow cover dynamics, and soil moisture changes affect spectral reflectance characteristics captured by satellite sensors. For instance, deciduous vegetation demonstrates strong temporal variability in the visible and near-infrared (NIR) bands, peaking in reflectance during the growing season and declining during senescence [2]. Similarly, seasonal snow and ice cover, common in temperate and boreal regions, significantly increase surface albedo and influence energy interactions in the visible and shortwave infrared (SWIR) ranges [2].

These variations affect both radiometric consistency and classification accuracy across time series analyses. Image misalignment caused by differing solar angles, sensor view geometry, or atmospheric scattering and absorption can lead to errors in change detection and temporal pattern recognition [2]. For example, vegetation indices such as NDVI and EVI are sensitive not only to plant health but also to sun-sensor-target geometry, which shifts across seasons [2].

To mitigate these issues, radiometric and atmospheric corrections, as well as temporal normalization techniques, are essential. Empirical line calibration and image normalization using regression help reduce the effects of atmospheric and seasonal variability. Furthermore, composite imagery - such as multi-date mosaics or cloud-free seasonal composites - is commonly used to build consistent datasets for applications in land cover mapping and change detection [2].

In cloud detection tasks specifically, seasonal variation in cloud frequency and type complicates consistent detection. For instance, persistent cloud cover during monsoon or winter seasons reduces the number of usable cloud-free observations, particularly in tropical and high-latitude regions [7]. Addressing these limitations often requires combining multi-temporal data with advanced detection algorithms that account for seasonal patterns [11].

Thus, understanding and correcting for multi-temporal and seasonal variation is critical for accurate environmental monitoring, change detection, and long-term land surface analysis in remote sensing.

2.3 Machine-Learning Architectures for Cloud Detection in Satellite Imagery

2.3.1 Common neural networks for segmentation

Convolutional neural networks (CNNs), particularly deep learning, has significantly advanced the state of cloud segmentation in satellite imagery. CNNs excel at modeling complex spatial patterns and generalizing across different atmospheric conditions, sensor platforms, and geographic regions. Among the various architectures developed, several models have emerged as particularly effective in this domain, including U-Net, CloudSegNet, CS-CNN, and Mask R-CNN.

1. U-Net and Its Variants

U-Net, introduced by Ronneberger et al. (2015), is a convolutional neural network architecture originally developed for biomedical image segmentation. Its design features

a symmetrical encoder - decoder structure with skip connections that preserve fine-grained spatial details lost during downsampling, making it particularly suitable for pixel-wise classification tasks, including remote sensing applications such as cloud segmentation.

Building on this architecture, Wieland et al. (2019) implemented a modified U-Net for multi-sensor cloud and cloud shadow segmentation, using the SPARCS dataset for training and validating across multiple platforms including Landsat TM, ETM+, OLI, and Sentinel-2. Their model segmented five land cover classes (cloud, cloud shadow, snow/ice, land, and water), achieving a mean accuracy of 89%, Kappa of 0.82, and a Dice coefficient of 0.85. Notably, the model showed strong generalization across sensors without retraining and significantly outperformed traditional rule-based methods such as Fmask in both speed and accuracy on GPUs [14].

2. CS-CNN

CS-CNN (Cloud Segmentation CNN) is a custom-built CNN architecture developed for rapid cloud mask generation from geostationary satellite data. Unlike traditional pixel-based classifiers, CS-CNN performs full-image segmentation in a single forward pass, making it computationally efficient. It avoids the need for handcrafted features by learning spatial and spectral dependencies directly from the input data.

Drönner et al. (2018) tested CS-CNN on MSG-SEVIRI imagery and compared its output with the operational CLAAS-2 cloud mask. The model achieved an overall accuracy of 94% and was particularly effective at distinguishing clouds in challenging areas like coastlines and bright surface regions. Its architecture is streamlined for fast inference (25 ms per 508×508 image using GPU), which is critical in near real-time applications [12].

3. CloudSegNet

CloudSegNet is a lightweight encoder-decoder neural network designed specifically for cloud detection in ground-based sky images. Its architecture includes a simple set of convolutional and upsampling layers, making it ideal for tasks with limited

labeled data. It is optimized for both day and night image segmentation and achieves state-of-the-art results in binary cloud classification tasks.

Hensel et al. (2021) evaluated CloudSegNet against more complex models like Mask R-CNN using hemispherical sky images. CloudSegNet consistently outperformed Mask R-CNN in terms of accuracy and F1-score while requiring fewer training epochs and less computational power. For instance, after 3500 training epochs, it achieved an F1-score of 0.846 compared to 0.538 for Mask R-CNN, showing better generalization with fewer parameters [13].

4. Mask R-CNN

Mask R-CNN is a two-stage object detection and segmentation network that performs well on structured objects with defined boundaries. It includes a region proposal network (RPN) followed by a mask prediction branch for pixel-level segmentation. Although highly effective in many computer vision tasks, its application to cloud segmentation has shown mixed results [13].

In the same evaluation by Hensel et al. (2021), Mask R-CNN lagged behind CloudSegNet due to its complexity and higher data demands. The architecture struggled to consistently segment large or diffuse cloud formations, possibly due to insufficient training data and the difficulty in modeling cloud edges with bounding boxes.

2.3.2 U-Net for cloud segmentation

Given the wide success and adaptability of U-Net in remote sensing segmentation tasks, this architecture was selected for the experiments in this study. U-Net's ability to perform accurate pixel-level classification, maintain spatial resolution through skip connections, and generalize across sensors makes it particularly suitable for multi-temporal cloud detection in satellite imagery.

U-Net is an encoder-decoder network designed for semantic segmentation tasks. The encoder progressively extracts feature maps by applying convolutional and pooling layers, while the decoder reconstructs the segmented output through upsampling and convolutions, with skip connections that allow the network to retain important spatial

details. These skip connections are essential for retaining fine-grained features in the output, which is crucial for accurately segmenting objects like clouds, which have irregular and complex shapes [14] [15].

In cloud segmentation, U-Net performs pixel-wise classification, where each pixel is classified as cloud, cloud shadow, or background (land, water, etc.). The network's strength lies in its ability to generalize across different sensors and atmospheric conditions, making it a versatile solution for multi-sensor cloud detection tasks [13].

Studies comparing U-Net with other cloud detection methods consistently highlight its superior performance in both accuracy and efficiency. For instance, Wieland et al. (2019) demonstrated that U-Net outperformed traditional methods like Fmask and Random Forest in terms of accuracy (Dice coefficient of 0.85, Kappa coefficient of 0.82) and inference speed when applied to Landsat and Sentinel-2 imagery [14].

Moreover, U-Net's architecture allows for fast inference even with large datasets, which is crucial for real-time applications such as disaster response and environmental monitoring. When run on GPUs, U-Net achieved significantly faster processing times compared to rule-based methods, with inference times as low as 2.8 seconds per megapixel [14].

A significant advantage of U-Net is its ability to generalize across different satellite sensors, including Landsat, Sentinel-2, and MODIS. For example, U-Net trained on Landsat OLI data showed strong performance in detecting clouds across other sensors like Sentinel-2, without the need for retraining. This cross-sensor adaptability is a key benefit for cloud detection, especially when dealing with long-term time series or datasets from different platforms [15].

2.4 Normalization and Standardization in Image Preprocessing

In the context of image preprocessing, normalization and standardization are pivotal steps to adjust the image data, making it more suitable for analysis or machine

learning models. These processes help to deal with varying illumination, sensor conditions, and topographic influences in remote sensing and other image applications.

Normalization involves scaling pixel values to a consistent range, typically between 0 and 1, or -1 and 1. This method is essential when combining images from different sensors or conditions to ensure they are comparable. In remote sensing, a common practice is BRDF (Bidirectional Reflectance Distribution Function) normalization, which corrects for directional effects due to topography and atmospheric conditions. For example, Landsat and Sentinel-2 imagery often undergoes BRDF normalization to minimize discrepancies caused by varying sun angles and sensor viewing geometry [2] [19]. This method utilizes fixed coefficients, such as k0, k1, and k2, in relation to geometric and radiative factors like the solar zenith angle (θs), view zenith angle (θv), and azimuth angle (φ). Additionally, some studies also use percentile-based methods, such as the 1-99 percentile range, to limit the influence of outliers, particularly in areas with high variability in reflectance [16].

Standardization adjusts pixel values so that they have a mean of zero and a standard deviation of one. This method is particularly helpful when data are expected to follow a Gaussian distribution or when combining data from different sensors or times. For remote sensing data, standardization is often applied after other preprocessing steps, including atmospheric correction and cloud masking. A technique commonly employed in topographic normalization, Minnaert correction, relies on a parameterization scheme where reflectance values are adjusted based on local topographic factors like slope and aspect. This helps correct the anisotropic nature of surface reflectance, improving data consistency across different terrain types [2] [19].

Several methods can be used for normalization and standardization, including:

- Min-max normalization: Scales the pixel values to a specified range, typically [0, 1]. This is done by subtracting the minimum value from each pixel and then dividing by the range (max-min) [2].

- 1-99 percentile normalization: This method normalizes the image using the 1st and 99th percentiles of pixel intensity values, reducing the influence of extreme outliers that could distort analysis [16].
- Standard deviation and mean normalization: In this method, the pixel values are adjusted based on the mean and standard deviation of the image data, ensuring that the final distribution of values has a mean of zero and a standard deviation of one. This is particularly useful when the dataset is expected to be Gaussian [17].

These preprocessing techniques-particularly normalization and standardization - are crucial when developing machine learning models that are expected to generalize across different sensors, time periods, and atmospheric conditions. The effectiveness of such techniques is demonstrated in practice through empirical evaluation on real-world satellite data, as described in the following sections.

Before presenting the experimental setup and results, it is essential to define the evaluation metrics used to assess model performance.

2.5 Evaluation Metrics in Image Segmentation

To evaluate the performance of the segmentation models in this study, the Intersection over Union (IoU) metric was used. IoU, also known as the Jaccard Index, is a widely adopted evaluation criterion in semantic segmentation tasks due to its sensitivity to both over- and under-segmentation.

IoU is calculated for each class as the ratio between the intersection and the union of the predicted segmentation and the ground truth:

$$IoU = \frac{TP}{(TP + FP + FN)}$$

where:

- TP (true positives) are correctly predicted pixels,

- FP (false positives) are pixels incorrectly labeled as belonging to a class,
- FN (false negatives) are pixels that belong to a class but were not predicted as such.

IoU values range from 0 to 1, where 1 indicates perfect overlap between the predicted and ground truth regions. In this study, IoU was computed per class (e.g., clear, cloud shadow, thin cloud, thick cloud), and the mean IoU across all classes was reported as the final performance metric.

This metric was selected for its robustness to class imbalance and its ability to provide an intuitive measure of spatial agreement between predictions and reference annotations, which is especially important in cloud segmentation tasks with diffuse and overlapping boundaries.

3. MATERIALS AND METHODS

3.1 Datasets Description

3.1.1 CloudSen12+ dataset

For this research, the dataset CloudSEN12+ was used as the primary source for training, validation, and testing of deep learning models for cloud and cloud shadow segmentation in Sentinel-2 imagery. CloudSEN12+ is a significantly enhanced and expanded version of the original CloudSEN12 dataset. It represents the largest collection of expert-labeled pixels for cloud and cloud shadow detection in Sentinel-2 satellite data to date [20].

The dataset includes 50,249 Sentinel-2 Level-1C (L1C) image patches, which collectively cover an area of approximately 1,283,256 km². These samples are geographically distributed across all continents except Antarctica, providing extensive variability in land cover, atmospheric conditions, and cloud types. This diversity is essential for building models that generalize well across different environmental contexts [20].

Each image patch in the dataset includes:

- Thirteen Sentinel-2 spectral bands (resampled to 10-meter resolution using nearest-neighbor interpolation),
- One human-annotated semantic mask,
- One machine-predicted cloud mask generated using the CloudSEN12 UnetMobV2 model.

The dataset is structured into two major categories based on spatial resolution:

- p509: patches of 509 × 509 pixels,
- p2000: larger patches of 2000 × 2000 pixels, introduced specifically to improve the recognition of cloud shadows, which often require greater spatial context for accurate labeling [20].

Each of these patch categories is further divided by label type:

- High: every pixel is manually annotated using a rigorous protocol,
- Scribble: partial annotations covering only ~5% of each patch, useful for model validation,
- Nolabel: no human annotation but includes machine-predicted masks for potential weak supervision.

Only the ‘high’ and ‘scribble’ subsets are split into train, validation, and test folders. The ‘nolabel’ subset is used only in training and contains predictions to support semi-supervised learning approaches [20].

Each labeled pixel is assigned to one of four semantic cloud classes:

- Clear (0) – no cloud or shadow contamination,
- Thick Cloud (1) – optically thick clouds fully obstructing the Earth’s surface,
- Thin Cloud (2) – semi-transparent clouds allowing partial visibility,
- Cloud Shadow (3) – areas darkened due to cloud occlusion, dependent on solar geometry and cloud structure.

To ensure annotation quality, a five-step protocol was followed:

1. Sampling – priority was given to rare and ambiguous cloud formations, such as contrails and cirrus clouds.
2. Agreement – annotators established a unified labeling guide and agreed on optimizing the F2-score, which prioritizes recall.
3. Training – annotators underwent structured training with example-based practice and feedback.
4. Production – annotation was done manually, sometimes initialized using model predictions.
5. Quality Control – double-blind reviews were performed; samples with uncertain quality were re-evaluated [20].

Additionally, a semi-automatic label quality evaluation system was used to detect possible annotation errors. This system includes a Trustworthiness Index (TI) and a Hardness Index (HI). Labels were flagged as potentially erroneous if they had low agreement with model predictions (TI < 0.3) and high annotation complexity (HI > 0.5). As a result, 3,570 image patches (17.12% of the dataset) were reviewed, and 452 were confirmed to have labeling errors and were subsequently corrected [20].

CloudSEN12+ is publicly available under the CC0 license, meaning it can be used, modified, and redistributed freely without restriction. The full dataset and metadata can be accessed via the Science Data Bank under DOI: 10.57760/sciencedb.17702.

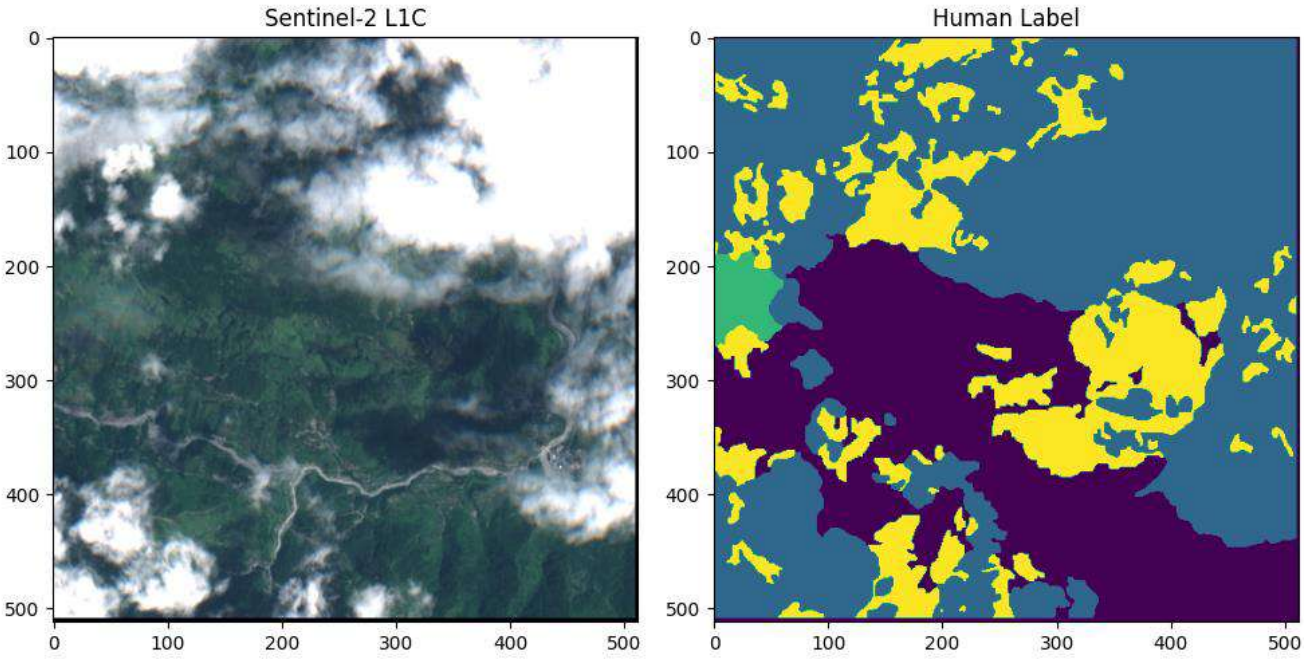
For this research, the dataset used is CloudSEN12+, which was accessed programmatically via the tacoreader library.

The dataset was loaded using the following command:

```
dataset = tacoreader.load("tacofoundation:cloudsen12-l1c")
```

This retrieves the Level-1C (Top-of-Atmosphere reflectance) Sentinel-2 image patches along with the associated annotations from the TACO Foundation repository. Each patch contains:

- Sentinel-2 reflectance bands (13 channels),
- A manually labeled cloud mask



Picture 3.1.1.1 Example of Sentinel-2 L1C Image and Corresponding CloudSEN12+ Human-Labeled Mask

Once loaded, the dataset was explored using Python libraries such as rasterio, matplotlib, and pandas. A sample image and its corresponding label were read and displayed to verify data integrity.

To ensure the analysis focused on high-quality, representative data, the metadata was filtered based on several criteria:

- Spatial resolution: Only image patches with a size of 509 × 509 pixels (referred to as *p509*) were selected, as they offer full pixel-wise annotations necessary for supervised deep learning training.
- Label quality: Only patches categorized as “high” quality were included, meaning that every pixel within the patch had been manually annotated and reviewed according to the strict CloudSEN12+ protocol.
- Temporal filtering: Only images captured during the months of June and July were selected. This choice minimizes seasonal variability and ensures that both training and testing samples have consistent atmospheric and cloud conditions. Using images from the same time period helps avoid introducing additional

temporal noise into model evaluation and highlights the model's true learning performance when trained and tested under similar conditions.

- Geographical filtering: Patches were further filtered based on latitude, selecting only those located within the temperate climate zones (between 30° and 60° latitude in both the Northern and Southern Hemispheres). This was done to match environmental conditions closer to those found in Ukraine, where temperate continental climate patterns predominate, ensuring greater relevance of the model results for applications in that region.

```
is_june_to_july = metadata["s2_date"].dt.month.isin([6, 7])
is_temperate_zone = metadata["latitude"].between(30,60) | metadata["latitude"].between(-60, -30)

filtered_train = metadata[(metadata["tortilla:data_split"] == "train") &
                          (metadata["real_proj_shape"] == 509) &
                          is_june_to_july &
                          is_temperate_zone &
                          (metadata["label_type"] == "high")]

filtered_validation = metadata[(metadata["tortilla:data_split"] == "validation") &
                               (metadata["real_proj_shape"] == 509) &
                               is_june_to_july &
                               is_temperate_zone &
                               (metadata["label_type"] == "high")]

filtered_test_summer = metadata[(metadata["tortilla:data_split"] == "test") &
                                (metadata["real_proj_shape"] == 509) &
                                is_june_to_july &
                                is_temperate_zone &
                                (metadata["label_type"] == "high")]
```

The filtered dataset was then split into:

- A training set
- A validation set
- A test set representing summer conditions

This split strategy ensures balanced temporal and geographical representation and is aligned with the dataset's design structure described in the original CloudSEN12+ paper [20].

3.1.2 Sentinel Hub image

To test the trained model on out-of-sample data with different temporal characteristics and not included in the CloudSEN12+ dataset, a Sentinel-2 image was obtained from Sentinel Hub. This approach was selected to demonstrate that while the model performs well on data from the same source (i.e., CloudSEN12+), it struggles with images that have different temporal and geographical characteristics, especially when the data is not normalized or standardized.

The objective of using the Sentinel-2 image was to show that, even if a model works well with test data from the same dataset (with consistent temporal properties), it will not generalize well to images from different temporal contexts, regions, or sources without proper normalization.

1. Data Acquisition from Sentinel Hub

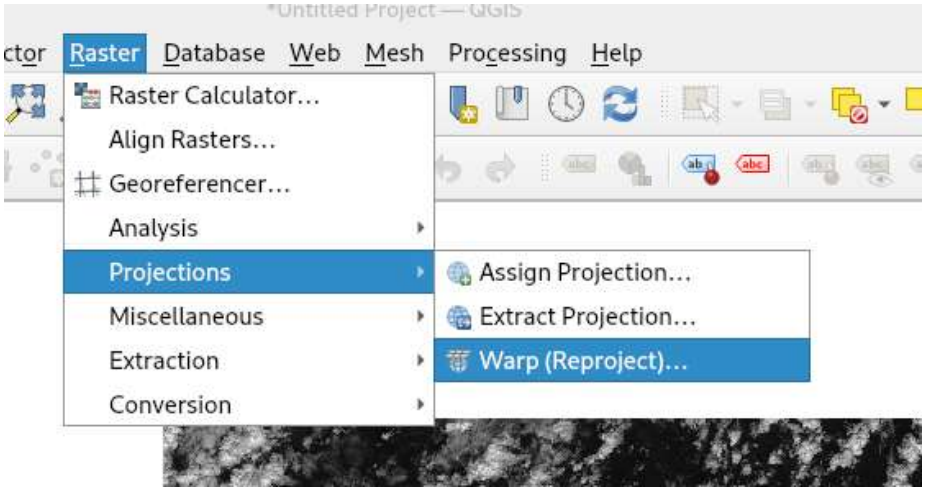
The Sentinel-2 image was downloaded from Sentinel Hub as a Level-1C (L1C) product, which contains 13 Sentinel-2 bands covering a broad spectrum of wavelengths essential for cloud and land classification tasks. For the purpose of this research, an August image over the territory of Latvia was selected due to its variety of cloud types, which were deemed relevant to the cloud detection task. These varying cloud types provided a challenging test case for the model trained on CloudSEN12+, which has a different temporal and spatial context.

2. Data Preprocessing in QGIS

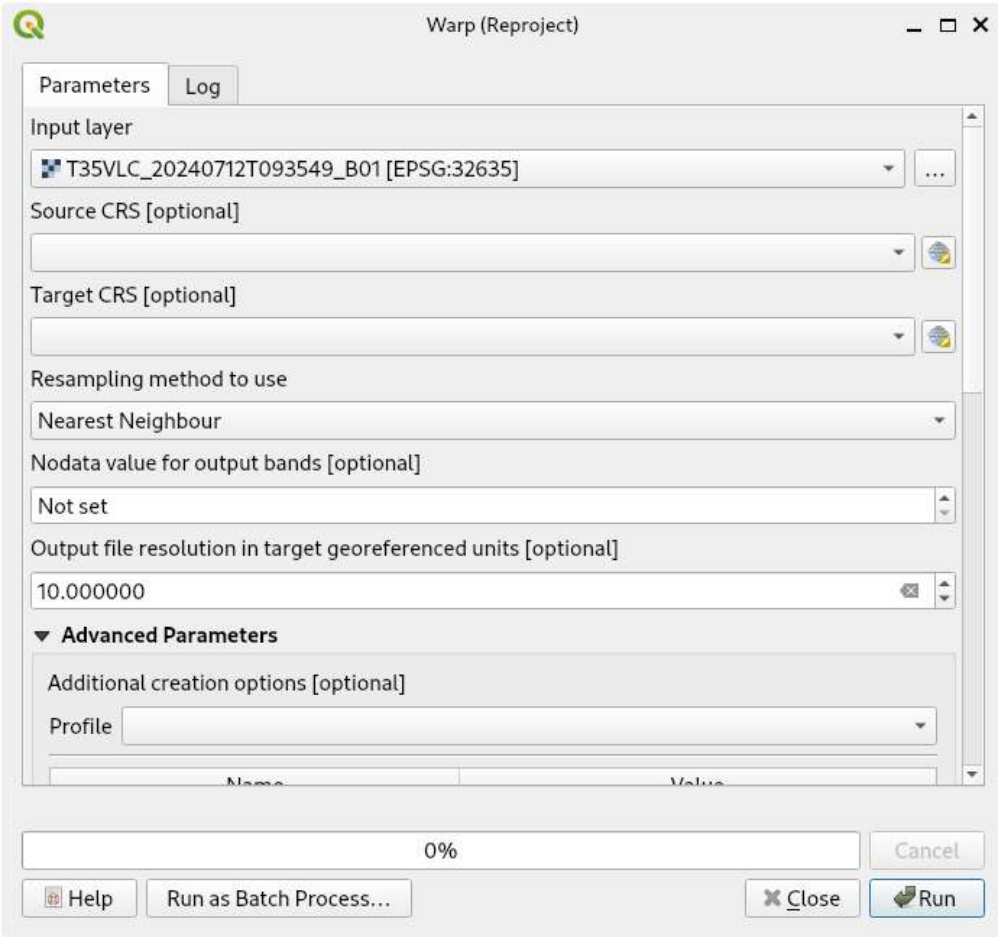
Once the Sentinel-2 image was obtained, QGIS was used to preprocess the data for model evaluation. The preprocessing steps included:

1. Spatial resolution adjustment: The original Sentinel-2 image from Sentinel Hub contained bands with varying spatial resolutions (10m, 20m, and 60m). However, since the CloudSEN12+ dataset is structured with a uniform 10-meter spatial resolution for all bands, it was necessary to resample the Sentinel-2 image to match this resolution. This ensures consistency between the training data and the new test data, enabling the model to handle the data appropriately. The resampling was done using the Warp tool in QGIS.

The Warp tool was accessed under Raster → Projections → Warp, where the spatial resolution was set to 10 meters for all bands.

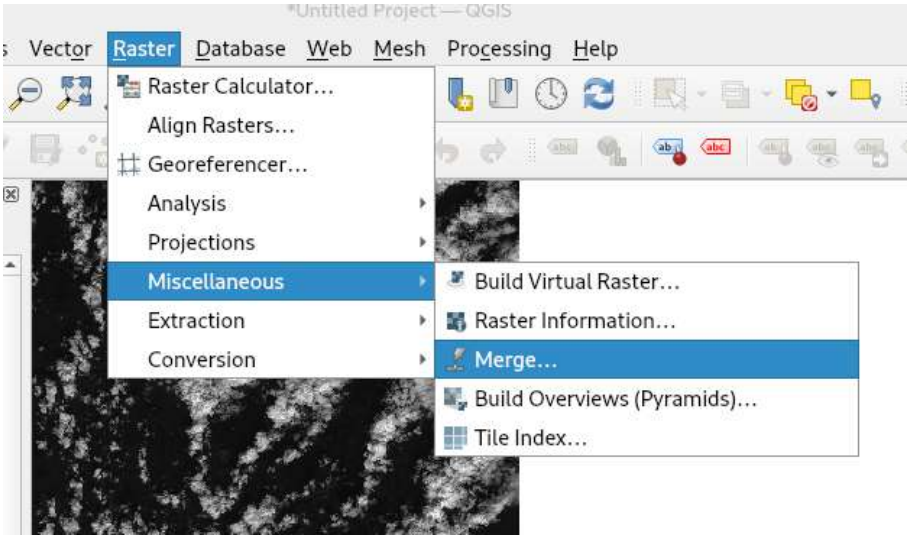


Picture 3.1.2.1 QGIS Warp Tool Location for Reprojecting Rasters

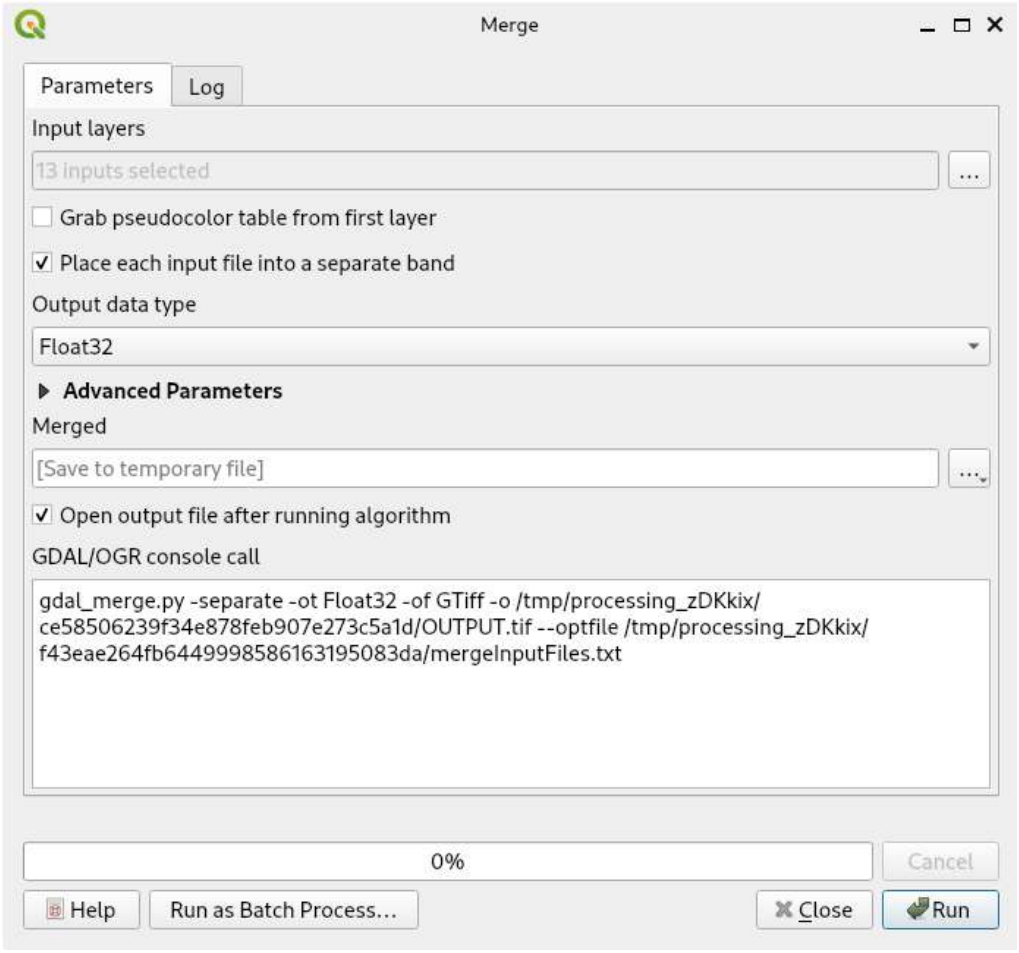


Picture 3.1.2.2 QGIS Warp (Reproject) tool interface

2. Merging multiband images: The Sentinel-2 data was provided as 13 separate .jp2 files, each corresponding to a different spectral band. These individual bands were merged into a single multiband image using the Merge tool in QGIS:
    a. The tool was accessed under Raster → Miscellaneous → Merge.
    b. Each input file was placed in a separate band, ensuring that the resulting image contained all 13 bands as individual layers within a single multiband image.

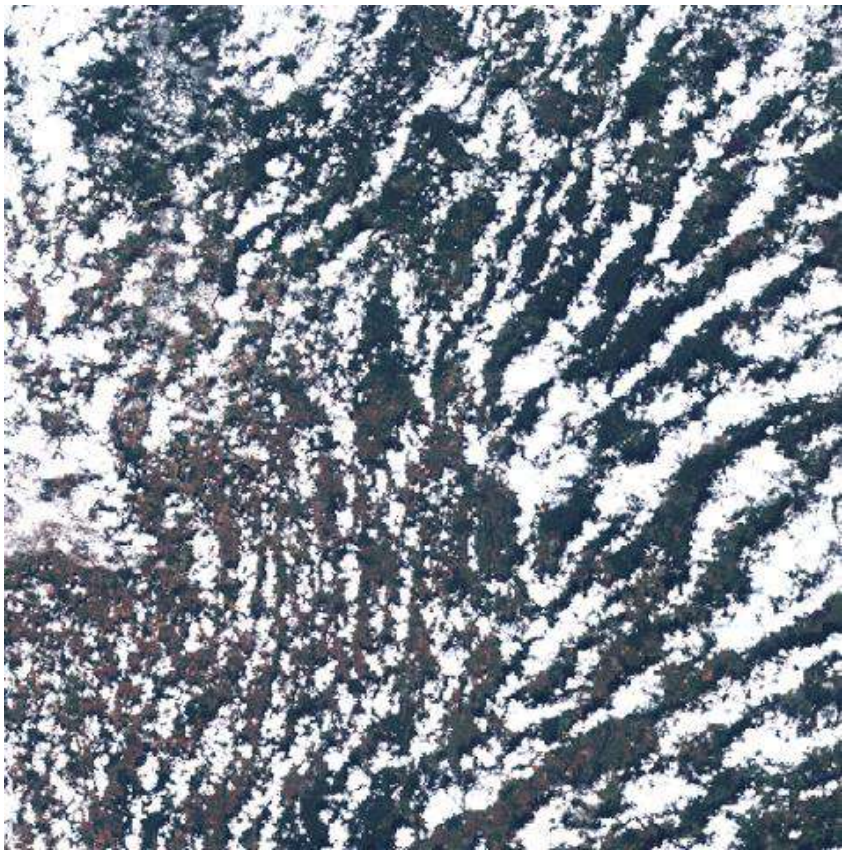


Picture 3.1.2.3 Accessing the Merge Tool in QGIS via Raster Menu



Picture 3.1.2.4 Interface of the Merge tool in QGIS

After completing the preprocessing steps, the output was a merged image file named merged_layer.tif, containing the 13 bands, all at a uniform 10-meter resolution. This processed image was then used for testing the model's ability to generalize on out-of-sample data.



Picture 3.1.2.5 merged_layer.tif

For the testing of the model on the Sentinel-2 image, a ground truth mask was created using the scene classification data available in the Level-2A (L2A) product that also can be downloaded from the Sentinel Hub along with L1C product containing 13 bands. The L2A product is a processed Sentinel-2 image that includes the classification of each pixel into various scene categories. The mask used in this study was located in the L2A folder.

This mask initially contained a detailed classification of pixels into 12 different classes, as outlined in the table below:

Value	Scene Classification	HTLM color code	Color
0	No Data (Missing data)	#000000	
1	Saturated or defective pixel	#ff0000	
2	Topographic casted shadows (called "Dark features/Shadows" for data before 2022-01-25)	#2f2f2f	
3	Cloud shadows	#643200	
4	Vegetation	#00a000	
5	Not-vegetated	#ffe65a	
6	Water	#0000ff	
7	Unclassified	#808080	
8	Cloud medium probability	#c0c0c0	
9	Cloud high probability	#ffffff	
10	Thin cirrus	#64c8ff	
11	Snow or ice	#ff96ff	

Picture 3.1.2.6 Scene Classification Color Legend for Sentinel-2 L2A Cloud Mask

However, for the purpose of cloud and cloud shadow segmentation, this classification was remapped into three primary categories: Clear, Cloud Shadow, and Clouds. This simplification aligns the data with the output classes used in model training, making it easier to evaluate the model's performance on multi-temporal data.

Remapping Procedure

The remapping process involved reading the mask data, then categorizing the original classes into three broader categories as follows:

- Clear: Classes corresponding to non-cloud and non-shadow pixels, including No Data, Saturated pixels, Vegetation, Not-vegetated, Water, Unclassified, Topographic casted shadows and Snow or Ice.
- Cloud shadow: Classes related to topographic or cloud shadows, specifically Cloud shadows.
- Clouds: Pixels representing clouds or those with a high likelihood of containing clouds, including Cloud medium probability, Cloud high probability, and Thin cirrus.

The following code snippet was used to carry out the remapping process:

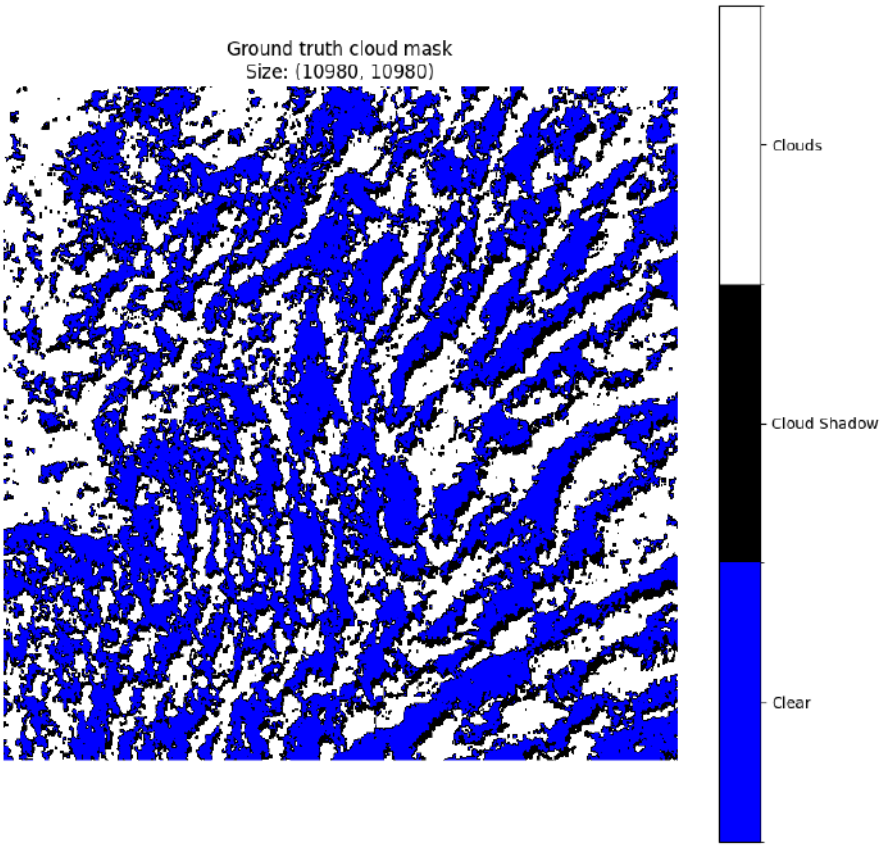
```
remapped_mask = np.zeros_like(mask_data)

clear_classes = [0, 1, 2, 4, 5, 6, 7, 11]
remapped_mask[np.isin(mask_data, clear_classes)] = 0

shadow_classes = [3]
remapped_mask[np.isin(mask_data, shadow_classes)] = 1

cloud_classes = [8, 9, 10]
remapped_mask[np.isin(mask_data, cloud_classes)] = 2
```

Although the labeling in the L2A product differs from the classes used in CloudSEN12+, the remapping procedure was designed to make the labels as similar as possible. This ensures that the model, which was trained on the CloudSEN12+ dataset, could be tested effectively on the Sentinel-2 image by aligning the relevant cloud categories in both datasets. By doing so, the comparison between the model's performance on CloudSEN12+ data and the Sentinel-2 test data is more meaningful and interpretable.



Picture 3.1.2.7 mask.tif

3.2 Model Architecture and Setup

In this research, the deep learning model used for cloud and cloud shadow segmentation is based on the U-Net architecture, which has proven effective in semantic segmentation tasks, particularly in remote sensing and medical image analysis. The U-Net model is designed to perform pixel-wise classification, which is essential for tasks such as cloud segmentation, where precise delineation of cloud boundaries is critical.

The model was implemented using the following libraries:

- PyTorch: A widely used deep learning framework for building and training neural networks. PyTorch's flexibility and dynamic computation graph made it suitable for the experiment.
- Segmentation Models PyTorch (SMP): This library provides a collection of popular architectures, including U-Net, with pre-built implementations. It simplifies the process of using well-known segmentation models and loss functions.
- PyTorch Lightning: This high-level framework abstracts much of the boilerplate code in PyTorch, enabling faster experimentation and easier model training, evaluation, and logging.
- TacoReader: A library for loading and processing the CloudSEN12+ dataset, facilitating easy access to Sentinel-2 imagery and associated metadata.

The model architecture chosen for this research is the U-Net, a convolutional neural network (CNN) designed specifically for image segmentation tasks. U-Net is well-suited for this type of problem because of its encoder-decoder structure and the use of skip connections, which help retain spatial information during downsampling and upsampling operations.

The U-Net model was configured using the Segmentation Models PyTorch (SMP) library, with the MobileNetV2 encoder and ImageNet weights for transfer learning. The model’s input consists of 13 Sentinel-2 bands, and the output layer has 4 channels representing the four classes: Clear, Cloud Shadow, Thin Cloud, and Thick Cloud.

```
class CloudSegmentationModel(pl.LightningModule):
    def __init__(self, lr=1e-3):
        super().__init__()
        self.model = smp.Unet(
            encoder_name="mobilenet_v2",
            encoder_weights="imagenet",
            in_channels=13,
            classes=4
        )
```

The Dice Loss function was used as the loss function for training, which is commonly employed in segmentation tasks due to its effectiveness in handling class imbalance:

```
self.criterion = smp.losses.DiceLoss(mode="multiclass")
```

To evaluate model performance, the Intersection over Union (IoU) metric was used for multi-class segmentation, along with accuracy for overall performance measurement:

```
self.train_iou = torchmetrics.JaccardIndex(task="multiclass", num_classes=4)
self.val_iou = torchmetrics.JaccardIndex(task="multiclass", num_classes=4)
```

The CloudDataset class was implemented to load the image data and corresponding labels efficiently. The data was preprocessed, including resizing the images to 256 × 256 patches to make the training process faster. Additionally, the pixel values of the Sentinel-2 bands were divided by 10,000 to convert the raw reflectance values into a suitable range for model input. The CloudDataModule class was used to manage data loading, batching, and splitting between training, validation, and test datasets:

```python
class CloudDataset(torch.utils.data.Dataset):
    def __init__(self, subset: pd.DataFrame, dataset):
        self.subset = subset.reset_index(drop=True)
        self.dataset = dataset

    def __len__(self):
        return len(self.subset)

    def __getitem__(self, index):
        sample_idx = self.subset.iloc[index].name
        s2_l1c = self.dataset.read(sample_idx).read(0)
        s2_label = self.dataset.read(sample_idx).read(1)

        with rio.open(s2_l1c) as src, rio.open(s2_label) as dst:
            X = src.read(list(range(1, 14))).astype(np.float32) / 10000
            y = dst.read(1).astype(np.int64)

        X = X[:, 0:256, 0:256]
        y = y[0:256, 0:256]

        X = torch.tensor(X, dtype=torch.float32)
        y = torch.tensor(y, dtype=torch.long)

        return X, y
```

```python
class CloudDataModule(pl.LightningDataModule):
    def __init__(self, dataset, batch_size: int = 4):
        super().__init__()
        self.batch_size = batch_size
        self.dataset = dataset

        self.train_dataset = CloudDataset(filtered_train, dataset)
        self.validation_dataset = CloudDataset(filtered_validation, dataset)
        self.test_summer_dataset = CloudDataset(filtered_test_summer, dataset)

    def train_dataloader(self):
        return torch.utils.data.DataLoader(
            dataset=self.train_dataset,
            batch_size=self.batch_size,
            shuffle=True,
            num_workers=0
        )

    def val_dataloader(self):
        return torch.utils.data.DataLoader(
            dataset=self.validation_dataset,
            batch_size=self.batch_size,
            num_workers=0
        )

    def test_dataloader(self):
        return [
            torch.utils.data.DataLoader(self.test_summer_dataset, batch_size=self.batch_size, num_workers=0),
        ]
```

This data module provided an easy interface for feeding data into the model during training and evaluation, ensuring that the data was split properly and that batch loading was efficient.

The model was trained using PyTorch Lightning, which provided an easy-to-use interface for managing epochs, training loops, and model checkpoints. The ModelCheckpoint callback was used to save the best model based on validation loss:

```
best_model_checkpoint = ModelCheckpoint(
    dirpath=MODEL_DIR,
    filename="best_model-{val_loss:.4f}",
    monitor="val_loss",
    save_top_k=1,
    mode="min",
    verbose=True
)
```

Finally, the trainer.fit(model, cloud_data) method was used to start training the model:

```
trainer = pl.Trainer(
    max_epochs=0,
    accelerator="gpu" if torch.cuda.is_available() else "cpu",
    callbacks=[best_model_checkpoint, last_model_checkpoint]
)
```

3.3 Experimental Setup

This section describes the experimental procedure designed to evaluate how the lack of input normalization affects the generalization ability of cloud segmentation models trained on satellite imagery. Two evaluation scenarios were considered: intra-dataset testing, using CloudSEN12+ images from the same time period as the training data (June), and cross-dataset testing, using an external Sentinel-2 image retrieved from the Sentinel Hub platform, captured in August.

Training on Unnormalized CloudSEN12+ (June)

The model was trained using Sentinel-2 Level-1C imagery from the CloudSEN12+ dataset, filtered to include only high-quality, fully labeled samples captured during the month of June in temperate zones. These filters ensured that the training data were temporally consistent and geographically representative of the target application region

(e.g., Ukraine). Each image patch was 509 × 509 pixels in size and contained 13 spectral bands, which were later cropped to 256 × 256 for training efficiency.

In the preprocessing stage, the reflectance values were scaled by a factor of 1/10,000, following the Sentinel-2 data specification. **However, no further normalization or standardization was applied, making this setup suitable for assessing the baseline performance and transferability of the model under raw input conditions.**

A custom PyTorch Dataset class (CloudDataset) was implemented to read and preprocess each image-label pair. Each sample was loaded using rasterio, and the full 13-band image was cropped and converted into a PyTorch tensor. The label mask was also extracted and converted into an integer class tensor with four classes: Clear (0), Thick Cloud (1), Thin Cloud (2), Cloud Shadow (3).

The training process was managed using the PyTorch Lightning framework. A LightningDataModule class (CloudDataModule) was implemented to handle train/validation/test splits, dataloading, and batching. The model itself was built on the U-Net architecture, with a MobileNetV2 encoder pre-trained on ImageNet. The loss function used was Dice Loss (multiclass mode), and the optimization was carried out using Adam with a learning rate of 0.001. Evaluation metrics included accuracy and Intersection over Union (IoU), computed using the torchmetrics package.

Cross-Dataset Testing on Sentinel Hub (August)

To test the model's generalization to unseen data from a different temporal context, the trained model was applied to a Sentinel-2 L1C image from August, retrieved from Sentinel Hub and preprocessed in QGIS. Preprocessing steps included:

- Resampling all 13 spectral bands to 10-meter spatial resolution.
- Merging the bands into a single multiband GeoTIFF file.

- Extracting ground truth from the Scene Classification Layer (SCL) of the Level-2A product, which was remapped from its original 12-class format into three categories: Clear, Cloud Shadow, and Clouds.

The ground truth mask was resampled to 10 meters and compared against the predicted segmentation mask using IoU and accuracy.

3.4. Normalization Pipeline

To address the generalization issues observed during cross-dataset evaluation, a normalization procedure was implemented. This section describes the applied normalization technique, how it was integrated into the data pipeline, and how retraining and testing were conducted using the normalized data.

Techniques Used

The normalization method chosen for this experiment was percentile-based min-max normalization, applied per band across the training dataset. Specifically, pixel values in each Sentinel-2 spectral band were scaled using the 1st and 99th percentiles of reflectance values observed in the training set. This method reduces the influence of extreme outliers while preserving the underlying signal distribution, which is important for heterogeneous remote sensing data.

Each band was rescaled to the [0, 1] range using the following transformation:

$$X_{norm} = \frac{clip(X, P_{99}, P_1)}{P_{99} - P_1}$$

where P_1 and P_{99} are the 1st and 99th percentiles, respectively.

Implementation Details

To compute the normalization statistics, a set of training samples was processed to extract the 1st and 99th percentiles for each of the 13 Sentinel-2 bands. These values were stored in a dictionary band_percentiles, which was passed to the dataset class. During data loading, each band was normalized individually using these statistics.

The key function used was:

```
def normalize_image(band, percentile_1st, percentile_99th):
    band = np.clip(band, percentile_1st, percentile_99th)
    return (band - percentile_1st) / (percentile_99th - percentile_1st)
```

This normalization was integrated into the __getitem__ method of the custom CloudDataset class, ensuring that every input sample—whether for training, validation, or testing—was normalized before being passed to the model.

The training pipeline (based on PyTorch Lightning) was unchanged in terms of model architecture and training configuration. A U-Net model with a MobileNetV2 encoder was used, trained with Dice Loss and evaluated using IoU and accuracy metrics.

Retraining and Testing Methodology

After implementing normalization, the model was retrained on the same filtered CloudSEN12+ June dataset as in the unnormalized baseline. The structure of the experiment remained identical to maintain comparability. Evaluation was again conducted on two levels:

1. Intra-dataset testing (CloudSEN12+ June test set)
2. Cross-dataset testing (Sentinel Hub August image)

During cross-dataset testing, the Sentinel-2 image was normalized using the same 1st–99th percentiles computed from the CloudSEN12+ training set to ensure consistency. The image was then tiled into 256 × 256 patches, processed by the model, and reassembled into a full predicted mask.

A remapped ground truth mask derived from the Scene Classification Layer (SCL) was used for evaluation. It included three classes: Clear (0), Cloud Shadow (1), and Clouds (2).

Finally, per-class IoU, overall IoU, and accuracy were calculated using scikit-learn's jaccard_score and accuracy_score functions. This setup provided a direct comparison between models trained with and without normalization under identical conditions.

4. RESULTS

4.1. Baseline Results (No Normalization)

In this section, the results of the model trained on unnormalized data and tested both on the CloudSEN12+ dataset (June) and a Sentinel Hub image (August) are presented.

Intra-Dataset (CloudSEN12+ June)

The model was first evaluated on the CloudSEN12+ test set, which is temporally and spatially aligned with the training set. The evaluation on this dataset yielded the following results:

- Mean Accuracy: 0.9373
- Mean IoU: 0.7435

These results demonstrate that the model performed well when both the training and testing datasets were temporally aligned, as expected for a model trained without any normalization.

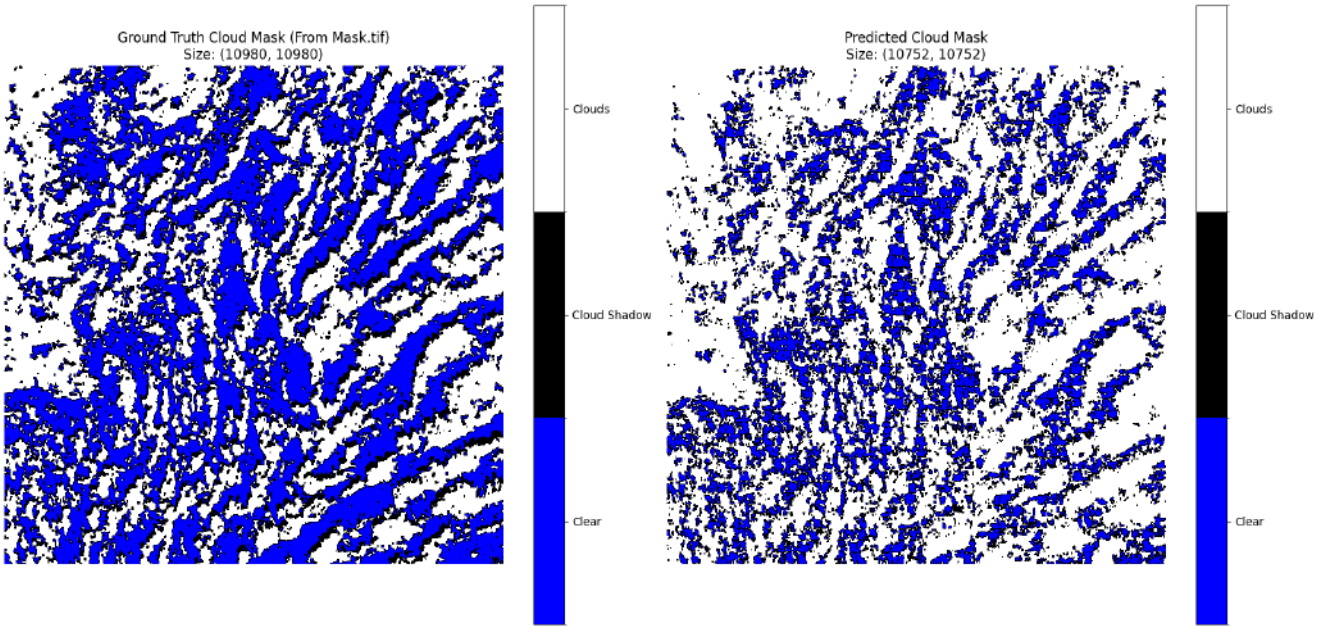
Cross-Dataset (Sentinel Hub August)

When tested on the external Sentinel Hub image from August, the model's performance drastically dropped:

- IoU Scores:
  - Clear: 0.4354
  - Cloud Shadow: 0.2079
  - Clouds: 0.4722
  - Overall IoU: 0.3718
- Accuracy: 0.6568

These results confirm the hypothesis that the model struggled to generalize to unseen data when no normalization was applied, demonstrating poor performance when

the temporal shift between training and testing data was introduced. Specifically, the model performed poorly on the Cloud Shadow class, which had an IoU of only 0.2079, indicating that the model was unable to handle domain shifts effectively without normalization.



Picture 4.3.1 Ground Truth Cloud Mask (left) and Predicted Cloud Mask (right) on Sentinel Hub image (without normalization)

The left image shows the ground truth cloud mask derived from the Sentinel-2 Scene Classification Layer (SCL) associated with the Level-2A product. The original SCL classes were reclassified into three categories to match the model output: clear (blue), cloud shadow (black), and clouds (white). The spatial extent is (10,980 × 10,980) pixels. This remapping facilitates direct comparison with the predicted segmentation.

The right image presents the predicted cloud mask generated by the U-Net model trained on unnormalized CloudSEN12+ data (June). This predicted mask was created by tiling and processing the Sentinel Hub image in 256 × 256 pixel patches. Although the model correctly identifies general cloud structures, the segmentation is noticeably noisier and less complete compared to the ground truth. In particular, many regions of cloud shadow are misclassified or missed entirely, and false positives appear in several clear-sky areas.

These qualitative results align with the quantitative metrics: while the cloud class was identified with moderate accuracy (IoU ≈ 0.47), the model failed to generalize to the shadow class (IoU ≈ 0.21), demonstrating the model's limited ability to transfer to unseen, temporally shifted data without normalization.

These results highlight that the model trained on unnormalized CloudSEN12+ data performed poorly when tested on data from a different temporal and data source context, even when visual and geographic conditions were similar. The particularly low performance on the Cloud Shadow class further illustrates the model's sensitivity to domain shift.

4.2. Normalized Training Results

This section presents the results of the model trained on normalized CloudSEN12+ data and evaluated on both the CloudSEN12+ test set (June) and the Sentinel Hub image (August). The goal of this experiment was to assess whether input normalization improves model generalization, particularly in cross-dataset settings with temporal shifts.

Intra-Dataset (CloudSEN12+ June)

The normalized model was first evaluated on the normalized CloudSEN12+ test set. The model achieved nearly identical performance:

- Mean Accuracy: 0.9337
- Mean IoU: 0.7364

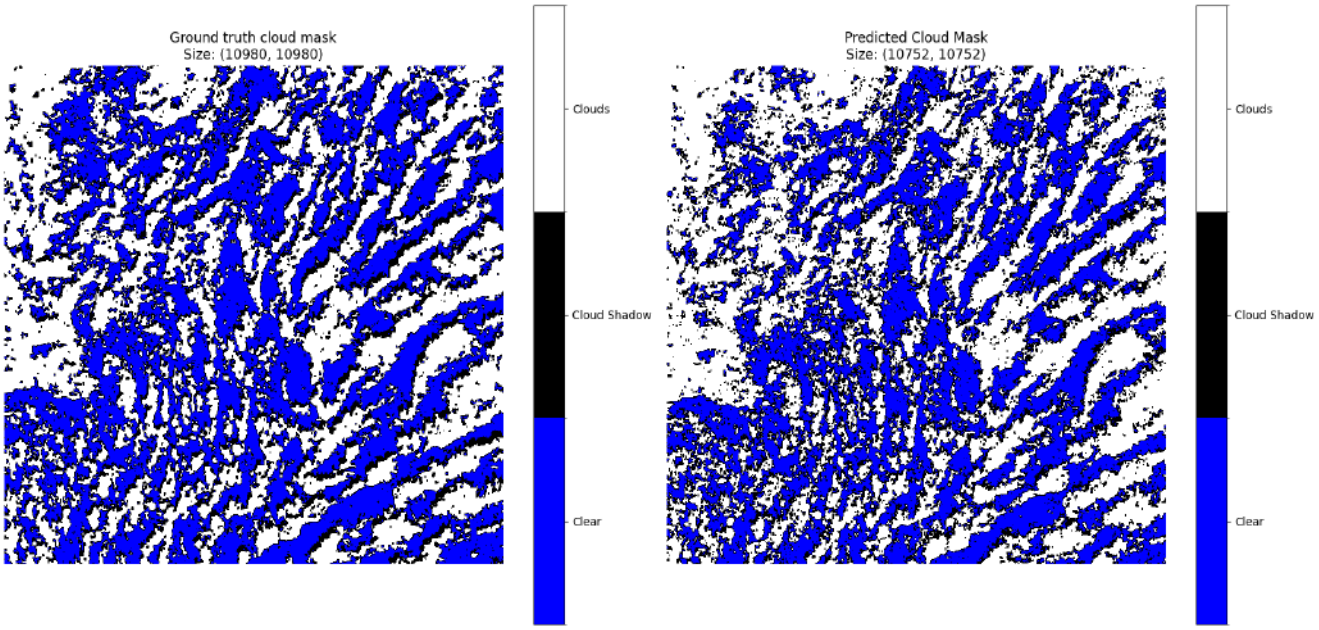
This result confirms that applying normalization did not negatively affect the model's ability to fit the training distribution. It also shows that normalization preserves performance within the training domain, while setting the stage for improved cross-domain robustness.

Cross-Dataset (Sentinel Hub August)

When applied to the Sentinel Hub image from August, the normalized model showed clear improvements over the unnormalized baseline:

- IoU Scores:
  - Clear: 0.6161
  - Cloud Shadow: 0.4529
  - Clouds: 0.5320
- Overall IoU: 0.5336
- Accuracy: 0.7349

Compared to the baseline performance, the normalized model achieved notable gains in generalization, especially for the Cloud Shadow class, which rose from 0.2079 to 0.4529. This improvement suggests that normalization helped mitigate the spectral and radiometric differences introduced by the temporal shift between the training and testing datasets.



Picture 4.2.1 Ground Truth Cloud Mask (left) and Predicted Cloud Mask (right) on Sentinel Hub Image (with normalization)

The left image shows the ground truth mask derived from the Sentinel-2 Scene Classification Layer (SCL), reclassified into three categories matching the model output:

clear (blue), cloud shadow (black), and clouds (white). The full resolution of the mask is (10,980 × 10,980) pixels.

The right image presents the predicted mask from the model trained on normalized CloudSEN12+ data. The mask was generated by tiling and processing the Sentinel Hub image in 256 × 256 pixel patches and then stitched together. Compared to the unnormalized prediction, the segmentation appears more complete, with clearer delineation of cloud shadows and fewer false positives in clear-sky regions.

These results demonstrate that input normalization significantly improves the model's robustness to temporal and sensor-related domain shifts. The enhanced ability to detect cloud shadows and reduce misclassification in clear areas reflects the importance of radiometric standardization in building transferable models for satellite image segmentation tasks.

5. CONCLUSIONS

This research investigated the effectiveness of image normalization as a method to improve the generalization capabilities of deep learning models in cloud detection tasks from multi-temporal satellite imagery. Specifically, the U-Net architecture was employed using the CloudSEN12+ dataset for training, with additional evaluation performed on an external Sentinel-2 image sourced from Sentinel Hub.

The experimental results demonstrated that normalization substantially enhances the generalization potential of the trained model. Without normalization, the model exhibited robust performance on data closely matching the training conditions (CloudSEN12+ June dataset), but significantly deteriorated when tested against temporally and geographically distinct imagery from Sentinel Hub. Particularly, the model struggled to accurately segment cloud shadows, reflected numerically in the extremely low IoU score.

In contrast, the application of percentile-based min-max normalization markedly improved model robustness, as evidenced both visually and numerically. Although the cross-dataset segmentation results remained relatively lower compared to intra-dataset performance—likely due to differences in labeling logic between CloudSEN12+ annotations and the Sentinel Hub Scene Classification Layer—normalization led to measurable enhancements. Notably, cloud shadow detection significantly improved, increasing from nearly no detection capability to a notably better IoU.

The improvements observed underscore the critical role of normalization in mitigating spectral and radiometric discrepancies inherent to multi-temporal remote sensing imagery. While the absolute segmentation results indicate there remains substantial room for improvement—potentially through harmonization of labeling schemes or additional model fine-tuning—**the clear numerical and visual benefits of normalization cannot be overstated.**

REFERENCES

1. Campbell, J. B., & Wynne, R. H. (2011). *Introduction to Remote Sensing* (5th ed.). Guilford Press.

2. Lillesand, T. M., Kiefer, R. W., & Chipman, J. W. (2015). *Remote Sensing and Image Interpretation* (7th ed.). Wiley & Sons.

3. Jensen, J. R. (2015). *Introductory Digital Image Processing: A Remote Sensing Perspective* (4th ed.). Pearson Education.

4. Schowengerdt, R. A. (2007). *Remote Sensing: Models and Methods for Image Processing* (3rd ed.). Academic Press.

5. Congalton, R. G., & Green, K. (2019). *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices* (3rd ed.). CRC Press.

6. Ackerman, S. A., Strabala, K. I., Menzel, W. P., Frey, R. A., Moeller, C. C., & Gumley, L. E. (1998). Discriminating clear sky from clouds with MODIS. *Journal of Geophysical Research: Atmospheres*, 103(D24), 32141–32157.

7. Platnick, S., Meyer, K. G., King, M. D., Wind, G., Amarasinghe, N., Marchant, B., Arnold, G. T., Zhang, Z., Hubanks, P. A., Ridgway, B., & Riedi, J. (2017). *The MODIS cloud optical and microphysical products: Collection 6 updates and examples from Terra and Aqua*. IEEE Transactions on Geoscience and Remote Sensing, 55(1), 502–525.

8. Kidder, S. Q., & Vonder Haar, T. H. (2005). *Satellite Meteorology: An Introduction* (2nd ed.). Academic Press.

9. Schmit, T. J., Gunshor, M. M., Menzel, W. P., Li, J., Bachmeier, A. S., & Gurka, J. J. (2005). *Introducing the next-generation Advanced Baseline Imager on GOES-R*. Bulletin of the American Meteorological Society, 86(8), 1079–1096.

10. Wind, G., Platnick, S., King, M. D., et al. (2010). Multilayer cloud detection with the MODIS near-infrared water vapor absorption band. *Journal of Applied Meteorology and Climatology*, 49(11), 2315–2333.

11. Joiner, J. A., Vasilkov, A., Bhartia, P. K., Wind, G., Platnick, S., & Menzel, W. P. (2010). Detection of multi-layer and vertically-extended clouds using A-Train sensors. *Atmospheric Measurement Techniques*, 3(1), 233–247.

12. Drönner, J., Korfhage, N., Egli, S., Mühling, M., Thies, B., Bendix, J., Freisleben, B., & Seeger, B. (2018). *Fast Cloud Segmentation Using Convolutional Neural Networks*. Remote Sensing, 10(11), 1782.

13. Hensel, S., Marinov, M. B., Koch, M., & Arnaudov, D. (2021). *Evaluation of Deep Learning-Based Neural Network Methods for Cloud Detection and Segmentation*. Energies, 14(19), 6156.

14. Wieland, M., Li, Y., & Martinis, S. (2019). *Multi-sensor Cloud and Cloud Shadow Segmentation with a Convolutional Neural Network*. Remote Sensing of Environment, 230, 111203.

15. Tarrio, K., Tang, X., Masek, J. G., Claverie, M., Ju, J., Qiu, S., Zhu, Z., & Woodcock, C. E. (2020). *Comparison of Cloud Detection Algorithms for Sentinel-2 Imagery*. Science of Remote Sensing, 2, 100010.

16. Raabe, E. A., & Stumpf, R. P. (1997). Image Processing Methods: Procedures in Selection, Registration, Normalization, and Enhancement of Satellite Imagery in Coastal Wetlands. U.S. Geological Survey. Open-File Report 97-287

17. Bishop, M. P., & Colby, J. D. (2006). *Topographic normalization of multispectral satellite imagery*. *Quaternary Science Reviews, 25*(10), 1027–1038.

18. Benhammou, Y. (2022). *Preprocessing Techniques for More Robust Deep Learning Models: Application to Biomedical and Satellite Images* (Doctoral Thesis). Universidad de Granada, Spain.

19. Franch, B., Vermote, E., Skakun, S., Roger, J.-C., Masek, J., Ju, J., Villaescusa-Nadal, J. L., & Santamaria-Artigas, A. (2019). A Method for Landsat and Sentinel 2 (HLS) BRDF Normalization. *Remote Sensing*, 11(6), 632.

20. Aybar, C., Bautista, L., Montero, D., et al. (2024). *CloudSEN12+: The largest dataset of expert-labeled pixels for cloud and cloud shadow detection in Sentinel-2*. Data in Brief, 56, 110852.