Zurich University of Applied Sciences

Department Life Sciences and Facility Management

Institute of Natural Resource Sciences

TERM PAPER 2

Pixel Classification of Remote Sensing Data - Assessing Impervious and Pervious Surfaces

JANUARY 17, 2025

Author:
Julian Kraft[1]

Tutor:
Dr. Johann Junghardt[1]

Affiliations:
[1]Institute of Natural Resource Sciences

Imprint

Project type:	Term Paper 2
Title:	Pixel Classification of Remote Sensing Data - Assessing Impervious and Pervious Surfaces
Date:	January 17, 2025
Keywords:	classification, deeplearning, machinelearning, remotesensing, landcoverclassification, perviousness, surface sealing
Copyright:	Zurich University of Applied Sciences

Author:	Julian Kraft[1] (kraftjul@students.zhaw.ch)
Tutor:	Dr. Johann Junghardt[1] (johann.junghardt@zhaw.ch)
Affiliations:	[1]Institute of Natural Resource Sciences

Abstract

Understanding the distribution of impervious and pervious surfaces is critical for effective urban planning, environmental management, and rainfall impact analysis. This study explores the use of convolutional neural networks (CNN) for pixel-based classification of aerial remote sensing data to assess surface sealing. Leveraging high-resolution SwissImage RS data, the analysis employs a simplified ResNet-18 architecture adapted for four-channel inputs, including RGB and near-infrared bands. A comprehensive workflow was developed, encompassing data preprocessing, augmentation, and hyperparameter tuning. The best-performing model achieved a classification accuracy of 0.927 for simplified surface perviousness, demonstrating the potential of deep learning to improve upon traditional geoprocessing methods. While challenges such as mixed pixels and class imbalances remain, this research highlights promising avenues for future advancements in remote sensing through the integration of advanced neural architectures and self-supervised learning.

Contents

Code, and LaTeX source are to be found on GitHub:
https://github.com/juliankraft/TermPaper2_RasterClassification

List of Figures

List of Tables

1 Introduction

Understanding surface sealing is a critical aspect of urban planning, environmental monitoring, and sustainable development. Sealed surfaces, such as roads, parking lots, and buildings, reduce natural soil permeability, disrupt water infiltration, and contribute to urban heat island effects, flooding, and habitat loss. Accurate detection and mapping of sealed surfaces at high spatial resolutions are essential for informed decision-making and policy development.

Traditionally, remote sensing techniques have been employed to address this challenge. These methods rely on spectral data, including near-infrared (NIR) bands, which are particularly effective for differentiating between sealed and unsealed surfaces due to their ability to capture subtle variations in surface reflectance. However, conventional approaches often involve rule-based geoprocessing or manual interpretation, which can be time-consuming and limited in scalability (Kadhim et al., 2016).

In recent years, the advent of deep learning has revolutionized remote sensing by enabling more automated, accurate, and scalable analysis. Neural networks have proven particularly effective in tasks like image classification, semantic segmentation, and object detection. While many studies in this domain focus on object-based or scene-based approaches (Thapa et al., 2023), pixel-level classification is an area with significant potential for granular analysis of surface characteristics. By leveraging the contextual information of neighboring pixels, pixel-based methods can provide a more detailed understanding of surface sealing, which is vital for applications requiring high spatial accuracy (Zheng & Chen, 2023).

Convolutional Neural Networks (CNN) have been widely utilized for remote sensing applications, including surface sealing detection and land cover classification. Prominent examples include ResNet-18 and ResNet-50, which have demonstrated strong performance in tasks requiring high-resolution image classification due to their ability to effectively learn hierarchical feature representations (Natya et al., 2022). VGG19, known for its simplicity and depth, has also been employed in remote sensing studies to classify urban landscapes and detect sealed surfaces, benefiting from its consistent feature extraction capabilities (Alem & Kumar, 2022). These architectures highlight the effectiveness of deep CNNs in capturing both spatial and contextual information for detailed surface analysis.

1.1 Background and Methodology

This term paper builds on previous research conducted for the canton of Basel-Landschaft, Switzerland. The original study employed a geoprocessing approach to determine the perviousness of surfaces on a per-pixel basis using aerial imagery, including NIR bands. This method focused on leveraging spectral characteristics to classify surfaces, achieving valuable insights into land cover and sealing patterns.

This paper explores an alternative methodology by applying a CNN, a simplified ResNet-18 architecture, to the same dataset. CNNs have demonstrated remarkable success in image analysis tasks by learning hierarchical feature representations directly from raw data. By training a neural network on the aerial imagery, this study aims to automate the process of surface classification to determine perviousness.

Unlike modern deep learning approaches that emphasize object-based or scene-based classifications (Thapa et al., 2023), this research adopts a pixel-based classification framework. Pixel-based classification not only allows for finer spatial resolution but also incorporates contextual information from neighboring pixels. This is achieved by inputting small patches of image data into the network, enabling it to capture local spatial patterns and textures critical for distinguishing between sealed and unsealed surfaces.

This study contributes to the growing body of research on deep learning in remote sensing by demonstrating the feasibility and potential of a simplified ResNet-18 model for pixel-level classification. Moreover, the inclusion of NIR data ensures that the model can effectively differentiate between surface types based on their spectral properties, a key advantage for urban and environmental applications. The results of this study aim to provide a foundation for future work in this area, such as integrating multi-temporal data or experimenting with more advanced architectures to further enhance performance.

2 Methods

2.1 Data

The used data for this project is the SwissImage RS (SwissTopo, 2024) data from the Swiss Federal Office of Topography (SwissTopo). It is a raster dataset with a resolution of 0.1m containing four bands: RGB and NIR. In order to cover the area of interest (AOI) 6 tiles of the dataset are needed. Over this large AOI there are three areas labeled with the corresponding labels Figure 1. These labels were provided by a team of researchers from the ZHAW. The three areas are distributed over a residential area (83487m^2), an industrial area (132642m^2) and a rural area (82740m^2). Two kinds of labels are available: The land cover category Figure 2 and an assessment of the degree of perviousness Figure 3. In Figure 4 the distribution of the data available for each label is shown for both the land cover category (`category`) and the sealing assessment (`sealed`). A third kind of labels was generated by simplifying the degree of perviousness into two classes: pervious and impervious (`sealed_simple`). This was done by reclassifying the unknown areas to sealed ones since they only consist of BuildingDistortions and ConstructionSites.

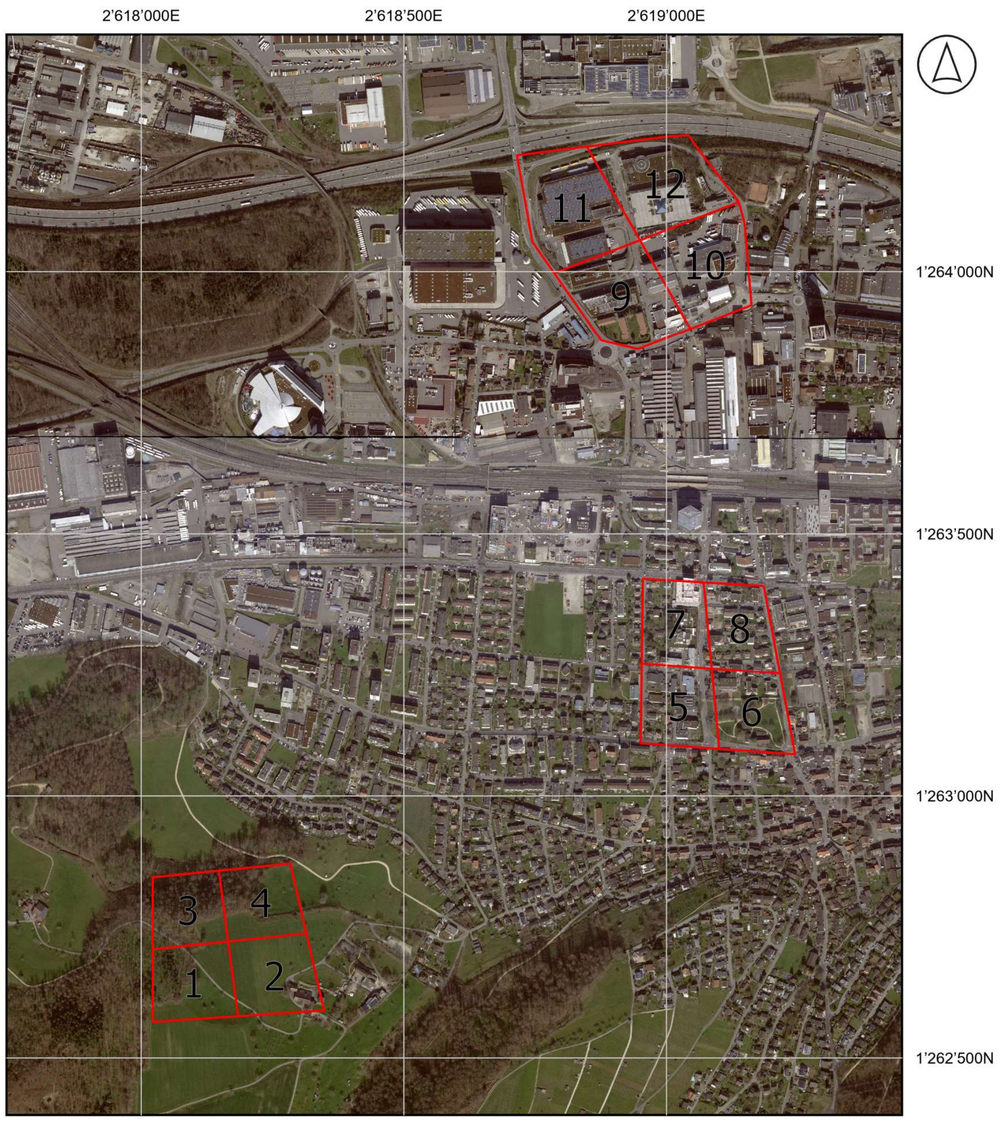

Figure 1: Map of the AOI, Pratteln, a municipality in the canton of Basel-Landschaft, Switzerland. The three marked areas, each split in four tiles, are the areas with corresponding labels. 1 - 4 are in the rural area, 5 - 8 in the residential area and 9 - 12 in the industrial area.

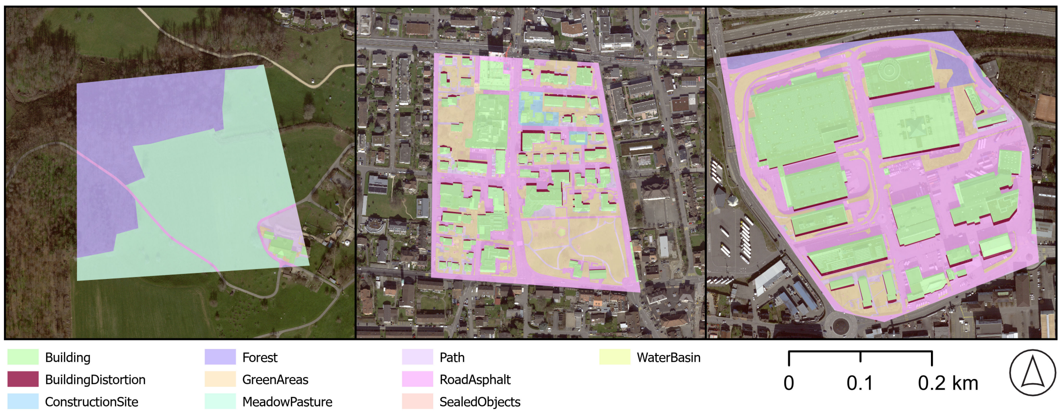

Figure 2: The three areas with the corresponding land cover categories.

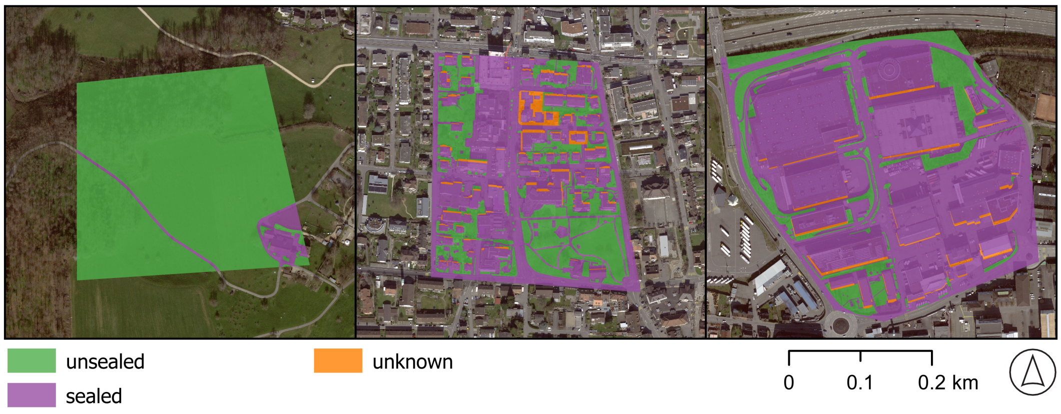

Figure 3: The three areas with the corresponding degree of perviousness.

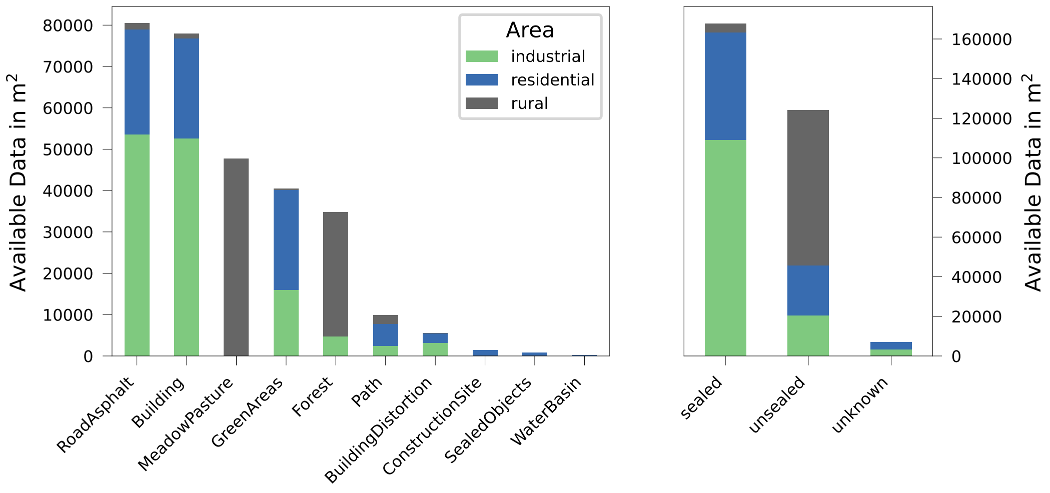

Figure 4: Available data by label, colored by the area.

2.2 Programming Language and Frameworks

To build and train the deep learning model, the programming language Python was used. The Frameworks PyTorch and Lightning are widely used and powerful tools for building deep learning models.

2.3 Model Architecture

For this study, an adapted version of the ResNet-18 architecture was employed to classify aerial imagery with high spatial resolution. ResNet-18 is a Convolutional Neural Network (CNN) model widely recognized for its residual connections, which mitigate the vanishing gradient problem during training. These connections allow for deeper networks while maintaining efficient backpropagation.

To suit the unique requirements of this research, several modifications were made to the standard ResNet-18 architecture:

- **Input Channels**: The original ResNet-18 was modified to accept four input channels (e.g., red, green, blue, and near-infrared bands) instead of the default three channels (RGB). This adjustment allows the model to leverage the additional spectral information for improved classification performance on remote sensing data.
- **Layer Configuration**: To adjust for the limited data available one of the usually four fully connected layers was removed from the model. And the number of planes per block was reduced significantly from 64, 128, 256, 512 to 4, 8, 16.

- **Output Design**: The model was tailored for pixel-wise classification tasks. The final fully connected layer was replaced with a convolutional layer to output class probabilities over a patch of pixels. The size of the output patch is adjustable via the `output_patch_size` parameter.
- **Pooling and Feature Extraction**: An adaptive average pooling layer was utilized to condense spatial features while maintaining flexibility in input dimensions. This pooling layer ensures compatibility with variable input sizes.
- **Regularization and Initialization**: Batch normalization layers were used to stabilize training and improve generalization. Weights in the convolutional layers were initialized using Kaiming normal initialization, enhancing the model's convergence during training.
- **Optimization and Training**: The model was trained using the AdamW optimizer, which combines adaptive learning rate methods with weight decay for regularization. A weighted cross-entropy loss function was employed to handle class imbalances effectively.

The modified ResNet-18 model was implemented using PyTorch and integrated into the Lightning framework for efficient training and evaluation. The design leverages the contextual and spectral richness of the four-channel input data, making it well-suited for pixel-level surface sealing classification tasks.

2.4 Data Processing and Augmentation

To feed data into the neural network, it is necessary to process it into a format that can be used for training. A standardized sample size is needed to ensure that the model can be trained on the data.

2.4.1 Preprocessing

In a first step the data was processed using ArcGIS Pro. The process included a few steps implemented as a model with the Model Builder Figure 5. The 6 tiles where mosaicked together and the area of interest was clipped. The area labels 1 - 12 and both versions of available data labels where transformed to raster datasets and added as additional bands to the dataset. The data was then exported as a GeoTIFF file. In order to use the data in a neural network an additional step was necessary. Using Python, the data was transformed into Zarr format. This format is a chunked, compressed, N-dimensional array storage format with multi-scale support. This allows for lazy loading and therefore for a more memory efficient data access during training.

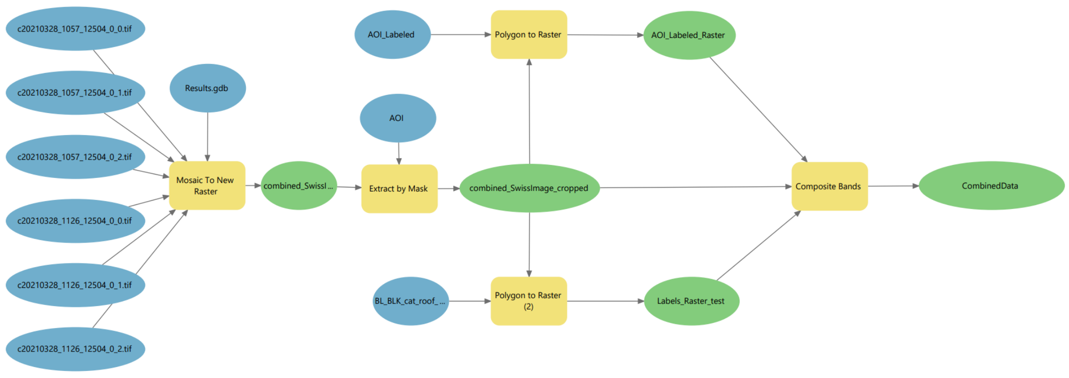

Figure 5: ArcGIS model used for preprocessing the data.

2.4.2 Processing and Augmentation

To feed the data into a neural network a PyTorch DataLoader was implemented. The data was sampled as cutouts of size 51x51 pixels, predicting the 5x5 center pixels as output. This cutout size and output size where implemented as a parameter but no other values where tested. The DataLoader indexes the available data depending on the cutout size and the corresponding area labels depending on the purpose:

• Areas for Training:	1, 2, 5, 6, 9, 10
• Areas for Validation:	3, 7, 11
• Areas for Testing:	4, 8, 12

It handles the data augmentation on the fly, during training steps. For the data augmentation a parent class BaseAugmentor was implemented from which the different augmenters inherit. The implemented augmentors - refer to Table 1 - are then chained into a object of the class AugmentorChain which can be used in the DataLoader. Only the Flip- and RotateAugmentor where used for the training of the models.

Table 1: Implemented augmentors

Augmentor	Description
FlipAugmentor	Randomly flips the image vertically and horizontally
RotateAugmentor	Randomly rotates the image 0, 90, 180 or 270°
PixelNoiseAugmentor	Adds random noise to the image (parameter: scale)
ChannelNoiseAugmentor	Adds random noise to the channels (parameter: scale)

2.5 Fitting the Model

Fitting the model was done on the IUNR HPC cluster using node 301 a HPE Apollo 6500 Gen10+ node running Rocky Linux 8. The node is equipped with 8 NVIDIA L40S GPUs (48GB each), dual AMD EPYC 7742 processors, 512 cores, and 5800 GB of storage, providing the computational power needed for high-performance tasks. The training was done using a potential limit of 300 epochs and an early stopping Callback set to stop after 10 epochs without improvement. Batch size was set to 128 for training and 256 for validation and prediction. During training only the best model and the last model where saved. On completion of the training the best model was loaded and used for prediction on the whole dataset. This predictions where saved as a Zarr file for later use in the evaluation. For each the category classification and the perviousness classification two models where trained with and without data augmentation. The data augmentation was done using the Flip- and RotateAugmentor. The learning rate was set to 0.001 and weight decay to 0.01.

For the simplified perviousness classification a limited hyperparameter search was done. All combinations of the following parameters where tested:

• Learning rate:	0.001, 0.0001
• Weight decay:	0, 0.1, 0.01
• Data augmentation:	no augmentation, Flip- and RotateAugmentor

2.6 Evaluation

For the performance evaluation of the models, the Python library Scikit-Learn was used. The predictions where loaded from the Zarr file and compared to the ground truth labels. Accuracy, F1-Score weighted and per class where calculated for every model and stored to a CSV file for later comparison. For the best model the accuracy was as well calculated grouped per corresponding land cover category to provide insight which category was classified best.

3 Results

The best performing model achieved an accuracy and of 0.9274 and a weighted F1 Score of 0.9275. The results for all trained models are summarized in Table 2. All trained models for the simplified perviousness lead the ranking, followed by the models for the perviousness classification and the category classification with accuracies of around 0.88 and 0.59, respectively. Only the label type `sealed_simple` will be considered for further analysis, as it is the most relevant for the study area.

Table 2: Accuracy and F1 Score for all trained models, arranged by accuracy.

LabelType	Augmented	LearningRate	WeightDecay	Accuracy	F1 Score
sealed_simple	True	0.0001	0.1	0.9274	0.9275
sealed_simple	False	0.0001	0.01	0.9109	0.9112
sealed_simple	True	0.001	0.1	0.9069	0.9072
sealed_simple	True	0.0001	0.01	0.9034	0.9038
sealed_simple	True	0.001	0	0.9028	0.9032
sealed_simple	False	0.001	0.1	0.9021	0.9024
sealed_simple	True	0.0001	0	0.9007	0.9011
sealed_simple	False	0.0001	0.1	0.9000	0.9004
sealed_simple	False	0.001	0.01	0.8999	0.9001
sealed_simple	False	0.001	0	0.8996	0.9000
sealed_simple	True	0.001	0.01	0.8980	0.8984
sealed_simple	False	0.0001	0	0.8977	0.8980
sealed	False	0.001	0.01	0.8817	0.8716
sealed	True	0.001	0.01	0.8763	0.8619
category	True	0.001	0.01	0.5902	0.5722
category	False	0.001	0.01	0.5806	0.5665

3.1 Hyperparameter Tuning

The Hyperparameter Tuning results included in the Table 2 and visualized in Figure 6 show that the `learning_rate` of 0.0001 performs better and that a `weight_decay` of 0.1 leads to the best results if combined with data augmentation. Over all data augmentation seems to have a positive effect on the accuracy but there is an exception for `weight_decay` of 0.01 where the accuracy is higher without data augmentation.

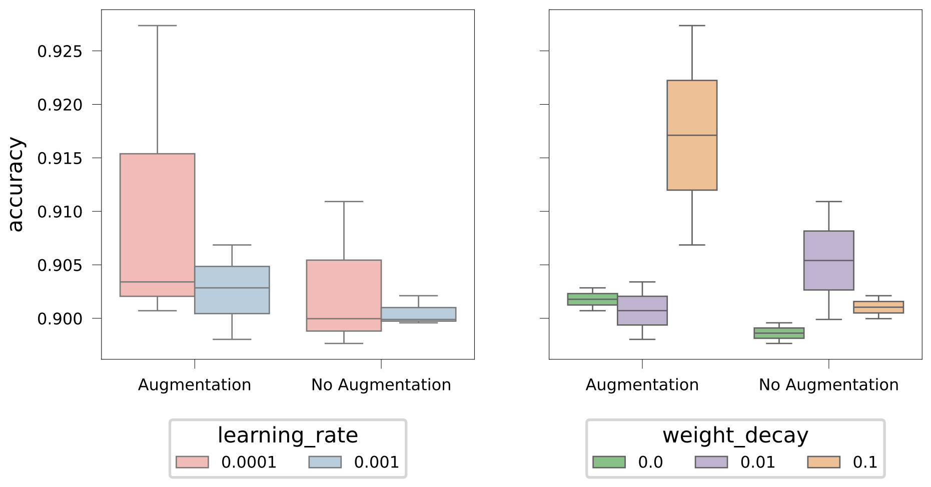

Figure 6: Accuracy of the models for the different hyperparameter grouped by use of data augmentation.

3.2 Best Model

The best performing model was the one for the simplified perviousness classification with the following hyperparameter:

- Learning rate: 0.001
- Weight decay: 0.01
- Data augmentation applied

It achieved an accuracy of 0.927 and a weighted F1 Score of 0.927. The F1 Score per class was 0.92 for class ’unsealed’ and 0.93 for class ’sealed’. The accuracy per category is shown in Figure 7. For a visual inspection of the predictions refer to Figure 8.

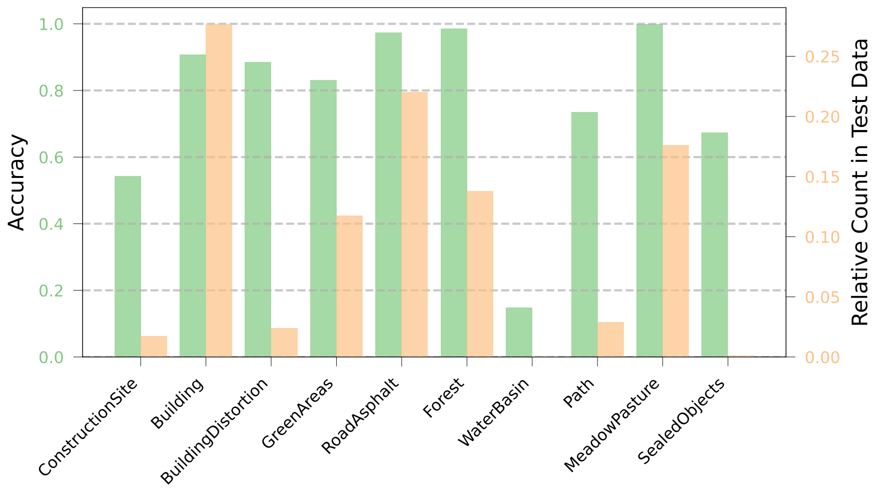

Figure 7: Accuracy per category for the best performing model including relative count of available data for each category.

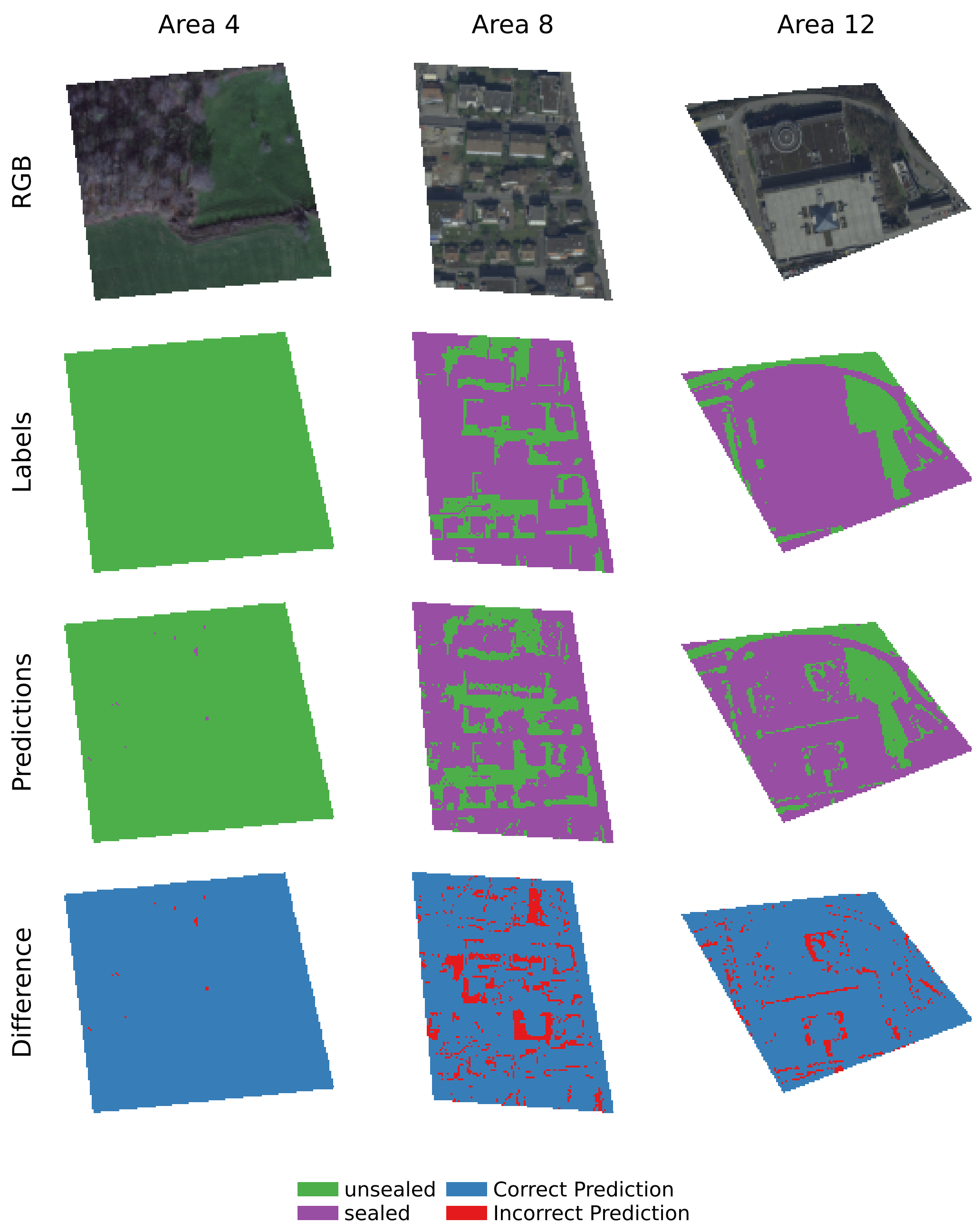

Figure 8: Visual inspection of the predictions for the test areas of the best model.

4 Discussion

4.1 Best Model Performance

The best performing model achieved a reasonable high accuracy of 0.92. The validation metrics loss and accuracy are shown in Figure 9 suggest a reasonable progress during the fitting process. Since they both are not all the way flattening out, it might even be possible to further improve the model using the same architecture and hyperparameters by training it for more epochs. This could be achieved by using a higher value for the `patience` parameter – the next value to be tested could be 20 instead of 10.

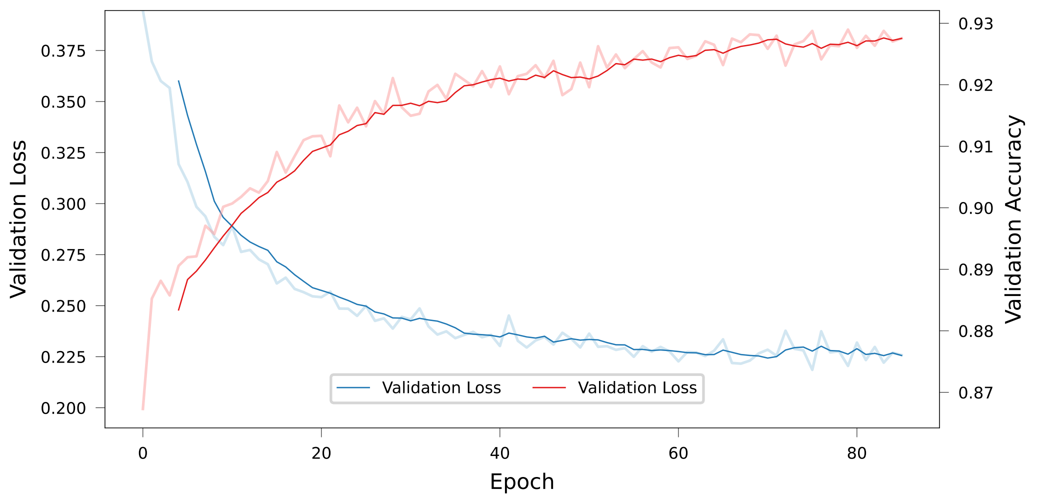

Figure 9: Validation loss and accuracy for the best model.

The accuracy per class, shown in Figure 7 does align with expectations. Classes like `Building`, `GreenAreas`, `RoadAsphalt`, `Forrest`, and `MadowPasture` are predicted with high accuracy. While classes, where there is very little data available like `WaterBasin` or classes where the data is more vague like `ConstructionSites` are predicted with lower accuracy.

From the visual inspection of the predictions in Figure 8, it looks like the borders between the classes are difficult to predict. This can be explained by the fact that the classes are not always very precise and this would specifically effect this areas. An other explanation could be the resolution of 10cm which leads to pixel in reality consisting of multiple classes – this issue referred to as mixed pixels – could be the part of the reason for the lower accuracy in the border areas.

4.2 Hyperparameter Tuning

The hyperparameter tuning process was successful in finding the best hyperparameters within the grid search. From Figure 6 it can be seen that lower values for the `learning_rate` seem to work better and that a higher regulation during training – a higher value `weight_decay` – leads to better results. Very interesting is the fact that the benefit of the data augmentation seems to correlate with more regularization during training. The grid of evaluated hyperparameters was rather limited dew to limited computational resources in the remaining time. A more extensive search could potentially lead to even better results. What might really be worth trying is to test for even smaller values of `learning_rate`, higher values for `weight_decay` and to use the other options for the data augmentation – pixel- and channel noise in different combinations.

4.3 Comparing the Results to the GIS Approach

Since the validation for the original studies was done for a different are in Rünenberg, using a different dataset, it is hard to compare the results directly. Adding to that, different land cover categories where used and different resolutions. Over all, the original studies achieved an accuracy of 0.9 to 1 for the more obvious classes like `Buildings`, `SealedRoads`, `GreenAreas` and `MadowPastures`. For the more difficult classes like `PathUnsealed` and `SealedObjects` the accuracy was lower as well. The results of this study – as different they are created – reach a similar range of accuracy.

4.4 Prospects

The results of this study show that the approach of using deep learning for perviousness classification is promising. The results are in a similar range as the results of the original study using a GIS approach. Therefore it might be worth to further investigate the potential of deep learning for this task. The next steps besides the mentioned hyperparameter expansion, more data augmentation and more thorough training going for more epochs there are other promising approaches. One could be to try different model concept like U-Net – specifically designed for image segmentation tasks. Another approach could be to use a pre-trained model to build upon – there the issue is to find a model that is trained on similar data and accepts four channels as input. An other idea could be to implement some self supervised learning techniques utilizing not labeled data to improve the model. As for most of the deep learning tasks, the more data the better – so it would be worth to try to get more data of a possibly even higher quality to improve the existing models.

5 Acknowledgment and Declaration

5.1 Acknowledgment

I thank Dr. Johann Junghardt for his support. I would like to express my gratitude to my brother, Dr. Basil Kraft, for his continuous support with his expertise in the field of machine learning.

5.2 Declaration of AI Usage

GitHub Copilot was used to assist writing the code and text for this project.

ChatGPT was used to assist researching as well as writing the code and text for this project.

Sections of the text generated by ChatGPT 4o revised by the author are:

- Abstract
- Introduction
- Model Architecture

Statement of Authorship for Student Work at the School of Life Sciences and Facility Management

By submitting the enclosed

☐ Project
☐ Literature review
☐ Course work
☒ Minor paper
☐ Bachelor’s thesis
☐ Master’s thesis (tick as appropriate)

the student affirms independent completion of the(ir) work without outside help.

The undersigned student declares that all printed and electronic sources used in the text and the bibliography are correctly indicated, i.e., that the work does not contain any plagiarism (no parts that have been taken in whole or in part from another's or his/her own text or from another's or his/her own work without clear identification and without stating the source).

In the event of misconduct of any kind, Paragraph 39 and Paragraph 40 of the General Academic Regulations for Bachelor's and Master's degree programmes at the Zurich University of Applied Sciences (dated 29 January 2008) and the provisions of the Disciplinary Measures of the University Regulations shall apply.

Location, date:	Student signature:
Neuhausen, 2025-01-20	[illegible]

Note on submitting the Statement of Authorship:

Direct submission of the work: This Statement of Authorship is to be inserted in the appendix of the ZHAW version of all work with original signatures and date (copies will not be accepted).

Submission of the work via Complesis: The Statement of Authorship should be made directly in Complesis by clicking as directed and should not be inserted in the appendix of the work.

Erlassverantwortliche/-r	LeiterIn Stabsbereich Bildung	Ablageort	2.05.00 Lehre Studium
Beschlussinstanz	LeiterIn Stab	Publikationsort	Public
Genehmigungsinstanz			

Version	Beschluss	Beschlussinstanz	Inkrafttreten	Beschreibung Änderung
1.0.0	15.03.2022	LeiterIn Stab	15.03.2022	Originalversion
1.1.0	15.04.2024	Leiter:in Stab	01.5.2024	Adding clarification on self-plagiarism

References

Alem, A., & Kumar, S. (2022). Transfer Learning Models for Land Cover and Land Use Classification in Remote Sensing Image. *Applied Artificial Intelligence*, *36*(1), 2014192. https://doi.org/10.1080/08839514.2021.2014192

Kadhim, N., Mourshed, M., & Bray, M. (2016). Advances in remote sensing applications for urban sustainability. *Euro-Mediterranean Journal for Environmental Integration*, *1*(1), 7. https://doi.org/10.1007/s41207-016-0007-4

Natya, S., Manu, C., & Anand, A. (2022). Deep Transfer Learning with RESNET for Remote Sensing Scene Classification. *2022 IEEE International Conference on Data Science and Information System (ICDSIS)*, 1–6. https://doi.org/10.1109/ICDSIS55133.2022.9915967

SwissTopo. (2024). *SWISSIMAGE RS*. Retrieved November 15, 2024, from https://www.swisstopo.admin.ch/de/orthobilder-swissimage-rs

Thapa, A., Horanont, T., Neupane, B., & Aryal, J. (2023). Deep Learning for Remote Sensing Image Scene Classification: A Review and Meta-Analysis. *Remote Sensing*, *15*(19), 4804. https://doi.org/10.3390/rs15194804

Zheng, X., & Chen, T. (2023). High spatial resolution remote sensing image segmentation based on the multiclassification model and the binary classification model. *Neural Computing and Applications*, *35*(5), 3597–3604. https://doi.org/10.1007/s00521-020-05561-8