Development and Applications of Machine Learning Methods for Hyperspectral Data

---

Felix M. Riese

Doctoral Thesis
Karlsruhe, 2020

Development and Applications of Machine Learning Methods for Hyperspectral Data

Doctoral Thesis

for the fulfillment of the requirements
for the academic degree

Doctor of Engineering (Dr.-Ing.)

Accepted by
the KIT Department of Civil Engineering,
Geo and Environmental Studies of the
Karlsruhe Institute of Technology (KIT)

Submitted by

Felix M. Riese

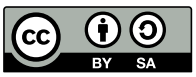
born in Stuttgart, Germany

Day of examination	19 May 2020
Main referee	Prof. Dr.-Ing. habil. Stefan Hinz Institute of Photogrammetry and Remote Sensing (IPF) Karlsruhe Institute of Technology (KIT)
Co-referee	Prof. Dr. Jocelyn Chanussot Grenoble Images Speech Signal and Control (GIPSA) Grenoble Institute of Technology (INP)

Karlsruhe 2020

Felix M. Riese
Development and Applications of Machine Learning Methods for Hyperspectral Data
Doctoral Thesis
Day of examination: 19 May 2020
Referees:
Prof. Dr.-Ing. habil. Stefan Hinz
Prof. Dr. Jocelyn Chanussot

Karlsruhe Institute of Technology (KIT)
Department of Civil Engineering, Geo and Environmental Studies
Institute of Photogrammetry and Remote Sensing (IPF)
Kaiserstr. 12
76131 Karlsruhe



> *Stay hungry. Stay foolish.*

— **Steve Jobs**
(Co-Founder of Apple Inc.)

Abstract

Hyperspectral remote sensing of the Earth relies on data from passive optical sensors that are mounted on platforms such as satellites and Unmanned Aerial Vehicles (UAVs). Hyperspectral data includes information to identify materials and to monitor environmental variables, such as soil texture, soil moisture, chlorophyll *a*, and land cover. Data analysis methods are necessary to retrieve information from hyperspectral data. One powerful tool in the analysis of hyperspectral data is Machine Learning (ML), a subset of Artificial Intelligence. ML models can solve non-linear correlations and are scalable on increasing dataset sizes. Every dataset and every ML estimation task brings new challenges that require innovative solutions. The aim of the studies presented in this thesis is the development and applications of ML methods on hyperspectral remote sensing data. These studies address the following three main challenges: (I) datasets with only a few labeled datapoints, (II) the limited potential of shallow ML approaches on hyperspectral data, and (III) the challenge of dataset shift between training and test dataset.

The studies on the challenge (I) result in the development and publication of a Self-Organizing Map (SOM) framework for unsupervised, supervised, and semi-supervised learning. The SOM is applied to a hyperspectral dataset in the (semi-)supervised regression of soil moisture, outperforming a Random Forest (RF) regressor. The SOM framework shows adequate performance in the (semi-)supervised classification of land cover. It provides additional visualization capabilities to improve the understanding of the underlying dataset. In the studies addressing the challenge (II), three innovative 1-dimensional Convolutional Neural Network (CNN) architectures are developed. The CNNs are applied in the context of a soil texture classification to a freely available hyperspectral dataset. Their performance is compared with two existing CNN approaches and a RF classifier. Two main findings can be summarized. Firstly, the CNN approaches show significantly better performance than the applied shallow approach RF. Secondly, adding the information about hyperspectral band numbers to the input layer of a CNN improves the performance on the individual classes. The studies on the challenge (III) are based on a UAV dataset, acquired on five different measurement areas in Peru in 2019. Dataset shift is detected with qualitative methods and with unsupervised ML approaches, such as Principal Component Analysis and Autoencoder. Based

on the results, a supervised regression of soil moisture is performed on different combinations of measurement areas. Additionally, to study the effects of dataset shift on the regression, the dataset is augmented with Monte Carlo methods. The applied SOM regressor is relatively robust against soil moisture sensor noise and performs well on small datasets, while the applied RF performs best on the full dataset. Dataset shift makes this regression task difficult; some combinations of measurement areas form a significantly better training dataset than others. To conclude, the presented studies tackling the three main challenges show promising results. The developed ML methods can be further enhanced in future research.

Zusammenfassung

Die hyperspektrale Fernerkundung der Erde stützt sich auf Daten passiver optischer Sensoren, die auf Plattformen wie Satelliten und unbemannten Luftfahrzeugen montiert sind. Hyperspektrale Daten umfassen Informationen zur Identifizierung von Materialien und zur Überwachung von Umweltvariablen wie Bodentextur, Bodenfeuchte, Chlorophyll *a* und Landbedeckung. Methoden zur Datenanalyse sind erforderlich, um Informationen aus hyperspektralen Daten zu erhalten. Ein leistungsstarkes Werkzeug bei der Analyse von Hyperspektraldaten ist das Maschinelle Lernen, eine Untergruppe von Künstlicher Intelligenz. Maschinelle Lernverfahren können nichtlineare Korrelationen lösen und sind bei steigenden Datenmengen skalierbar. Jeder Datensatz und jedes maschinelle Lernverfahren bringt neue Herausforderungen mit sich, die innovative Lösungen erfordern. Das Ziel dieser Arbeit ist die Entwicklung und Anwendung von maschinellen Lernverfahren auf hyperspektrale Fernerkundungsdaten. Im Rahmen dieser Arbeit werden Studien vorgestellt, die sich mit drei wesentlichen Herausforderungen befassen: (I) Datensätze, welche nur wenige Datenpunkte mit dazugehörigen Ausgabedaten enthalten, (II) das begrenzte Potential von nicht-tiefen maschinellen Lernverfahren auf hyperspektralen Daten und (III) Unterschiede zwischen den Verteilungen der Trainings- und Testdatensätzen.

Die Studien zur Herausforderung (I) führen zur Entwicklung und Veröffentlichung eines Frameworks von Selbstorganisierten Karten (SOMs) für unüberwachtes, überwachtes und teilüberwachtes Lernen. Die SOM wird auf einen hyperspektralen Datensatz in der (teil-)überwachten Regression der Bodenfeuchte angewendet und übertrifft ein Standardverfahren des maschinellen Lernens. Das SOM-Framework zeigt eine angemessene Leistung in der (teil-)überwachten Klassifikation der Landbedeckung. Es bietet zusätzliche Visualisierungsmöglichkeiten, um das Verständnis des zugrunde liegenden Datensatzes zu verbessern. In den Studien, die sich mit Herausforderung (II) befassen, werden drei innovative eindimensionale Convolutional Neural Network (CNN) Architekturen entwickelt. Die CNNs werden für eine Bodentexturklassifikation auf einen frei verfügbaren hyperspektralen Datensatz angewendet. Ihre Leistung wird mit zwei bestehenden CNN-Ansätzen und einem Random Forest verglichen. Die beiden wichtigsten Erkenntnisse lassen sich wie folgt zusammenfassen: Erstens zeigen die CNN-Ansätze eine deutlich bessere Leistung als der angewandte nicht-tiefe Random Forest-Ansatz. Zweitens verbessert das Hinzu-

fügen von Informationen über hyperspektrale Bandnummern zur Eingabeschicht eines CNNs die Leistung im Bezug auf die einzelnen Klassen. Die Studien über die Herausforderung (III) basieren auf einem Datensatz, der auf fünf verschiedenen Messgebieten in Peru im Jahr 2019 erfasst wurde. Die Unterschiede zwischen den Messgebieten werden mit qualitativen Methoden und mit unüberwachten maschinellen Lernverfahren, wie zum Beispiel Principal Component Analysis und Autoencoder, analysiert. Basierend auf den Ergebnissen wird eine überwachte Regression der Bodenfeuchte bei verschiedenen Kombinationen von Messgebieten durchgeführt. Zusätzlich wird der Datensatz mit Monte-Carlo-Methoden ergänzt, um die Auswirkungen der Verschiebung der Verteilungen des Datensatzes auf die Regression zu untersuchen. Der angewandte SOM-Regressor ist relativ robust gegenüber dem Rauschen des Bodenfeuchtesensors und zeigt eine gute Leistung bei kleinen Datensätzen, während der angewandte Random Forest auf dem gesamten Datensatz am besten funktioniert. Die Verschiebung der Verteilungen macht diese Regressionsaufgabe schwierig; einige Kombinationen von Messgebieten bilden einen deutlich sinnvolleren Trainingsdatensatz als andere. Insgesamt zeigen die vorgestellten Studien, die sich mit den drei größten Herausforderungen befassen, vielversprechende Ergebnisse. Die Arbeit gibt schließlich Hinweise darauf, wie die entwickelten maschinellen Lernverfahren in der zukünftigen Forschung weiter verbessert werden können.

Acknowledgement

I am writing and handing in this thesis during the time of the Coronavirus disease in 2019 and 2020. I am thankful for the people in the health system and in jobs of systemic importance who contribute towards stopping this crisis.

I am deeply grateful for the three very intense Ph.D. years. I feel lucky to be surrounded by so many inspiring and supportive people that I was able to meet along the way. While this thesis is my work, my research would not have been possible without the people I met. I can not acknowledge every person that has inspired and supported me during my Ph.D., but I will mention the most significant ones.

First of all, I want to thank Stefan Hinz for supervising my Ph.D. at the Institute of Photogrammetry and Remote Sensing (IPF) of the Karlsruhe Institute of Technology (KIT), for giving me the freedom to pursue my research interests, and for being an absolute *enabler*. Stefan enabled me to give talks at different international conferences, to participate in the open-source community, and to spend time abroad on different occasions. These learning opportunities are truly valuable.

Secondly, I want to thank Sina Keller for being my direct supervisor. She invested countless hours discussing research and giving feedback. Sina inspired me with her strong drive and motivation to publish our research, and she was in charge of the TRUST project's acquisition and management. Thanks to Sina, I changed my field of research and started my research at the IPF, which turned out to be a great experience. Sina resolved many blockers that came up during my Ph.D.; and with her help, I already had data within the first month of my Ph.D. to start my initial research. I have been fortunate to have Sina as my supervisor!

Thirdly, I want to thank my direct office colleagues Johanna Guth, Philipp Maier, and Tobias Guth, from the IPF. The combination of deep focus, intense discussions, and relaxed coffee breaks on our balcony made me enjoy my time at the institute even more. Furthermore, I thank Chris Michel for the useful discussions and his feedback on my research. I want to thank Jens Leitloff for keeping each other up-to-date with Machine Learning research, for his critical feedback on my research, for his outside-the-box ideas, and for our two shared talks at the PyCon and the M3 conference.

I am grateful for the opportunity to go to Australia and visit the company FluroSat in Sydney. The Graduate School for Climate and Environment (GRACE) at the KIT, with Andreas Schenk and Ilse Engelmann, funded my stay abroad and several external trainings. I thank Anastasia Volkova, the CEO of FluroSat, for *heaps* of amazing experiences I made during my visit. Further, I thank Juan Delard de Rigoulières Mantelli, Victor Proto, Leandro Giovannini, Thomas Sauvajon, and the whole team.

Next, I want to thank Jocelyn Chanussot for being my co-referee, despite his incredible workload and the external circumstances. I thank Heike Birkel for the perfect administrative support over the three years. Special thanks to Sven Wursthorn, who kept the IPF infrastructure running and the servers working around the clock. The successful measurement campaign in Peru would not have been successful without Samuel Schroers, Philipp Wagner, and Julian Bocanegra.

The work of this thesis benefited a lot from open-source software. I am thankful for the developers, for example, of Python, Python packages such as *scikit-learn*, *pandas*, *NumPy*, *matplotlib*, as well as LaTeX.

I want to thank my colleagues and friends from the Collège des Ingénieurs. Namely, I thank Elmar Mitterreiter and Florian Schäfer for their motivating support and their feedback on my research. Furthermore, I want to thank my friends for their support during my Bachelor's, Master's, and Ph.D. in Karlsruhe for making this time unforgettable. Special thanks to Timothy Gebhard and Carl Degitz for their feedback on my research.

I am grateful for the unlimited support of my parents Susanne and Matthias Riese, and my sister Annika Riese.

I thank my partner Teresa for her support and her humor. Meeting her was a big highlight of my three Ph.D. years.

Thank you! Merci! Gracias! Grazie! Obrigado! Cheers mates! Danke!

Felix M. Riese
Karlsruhe, June 2020

Contents

1 Introduction

> *Prediction, not narration, is the real test of our understanding of the world.*
>
> — **Nassim Nicholas Taleb**
> (Statistician, Author)

1.1 Background

Vision is of utmost importance for humans. The human eye covers the electromagnetic spectrum from about 380 nm to 750 nm, referred to as the Visible Spectrum (VIS), in three different types of retinal cells [1, 2, 3]. Each type of retinal cell corresponds to a specific spectral band, characterized by the mean wavelength and bandwidth. The combination of these three spectral bands enables humans to distinguish between millions of colors and, therefore, to differentiate visually between materials. Humans can recognize, for example, roads, grass, and bare soil from color and spatial patterns. They perceive roads as *grey*, grass as *green*, and bare soil as *brown*.

The Red–Green–Blue (RGB) color model is an adaptation of the three-spectra model of the human eye. For example, the majority of today's consumer-grade cameras are based on passive, optical RGB sensors. Hyperspectral sensors extend this color model and the human vision by (i) increasing the number of spectral bands from three to about 10 to 1000 bands, (ii) decreasing the width of the spectral bands to a few Nanometers, and (iii) extending the spectral range into the infrared spectrum [2]. Depending on the type of hyperspectral sensor, it covers the Visible and Near-Infrared (VNIR), Short-Wavelength Infrared (SWIR), or Thermal Infrared (TIR) spectrum. The increased amount of information about the electromagnetic spectrum, compared to human vision and RGB cameras, improves the identification and differentiation of materials and monitoring of physical processes.

Hyperspectral remote sensing of the Earth applies hyperspectral sensors from satellites, airplanes, and Unmanned Aerial Vehicles (UAVs) [4, 5]. With these platforms,

hyperspectral data can be acquired over large areas and in short time scales. Hyperspectral remote sensing derives information about, for example, soil texture and soil moisture, chlorophyll *a* concentrations in inland waters, and changes in land cover and vegetation classes [6, 7, 8, 9]. This large variety of possible applications makes hyperspectral remote sensing a valuable tool in many research fields of environmental sciences such as hydrology, ecology, and geology [2, 10].

In hyperspectral remote sensing, technological advances of the last decades have significantly lowered the costs of data acquisition with satellites and UAVs [11]. These technological advances have substantially increased computing power and data storage capabilities (*big data*). The resulting growth of data availability requires scalable data analysis tools. The increased computing power makes it possible for Machine Learning (ML), which is a subset of Artificial Intelligence (AI), to become a standard data analysis tool in many research fields such as hyperspectral remote sensing [10, 12, 13, 14, 15, 16].

ML methods can perform tasks such as supervised classification and regression with non-linear correlations [17]. For these tasks, ML models are built based on given training datasets consisting of input data and labels. The ML model learns patterns in the dataset. The aim in ML estimation is to train a ML model that generalizes to new input data. This means that the ML estimation can infer meaningful outputs between untrained input data or test datasets [18]. In the example of soil moisture regression from hyperspectral data, a hyperspectral image of a measurement area is the input data. The labels, referred to as ground truth or reference data, can be in situ soil moisture measurements.

1.2 Challenges and Overall Aim

This thesis focuses on ML estimations, such as the previous example of soil moisture regression from hyperspectral data. During several field experiments, heterogeneous datasets are acquired on different scales, in different countries, and with different hyperspectral sensors. Every dataset and every corresponding estimation task brings along new challenges that require new innovative, methodological solutions [10]. Three main challenges are the focus of this thesis:

(I) ML model training on datasets with only a few labeled datapoints,

(II) the limited potential of shallow ML approaches on hyperspectral data, and

(III) the challenge of distribution differences between training and test dataset.

The overall aim of this thesis is to improve the quality of ML estimations based on hyperspectral data. This aim is achieved by (a) developing ML methods for hyperspectral data to address the three main challenges, (b) applying innovative methods in current ML research to the field of hyperspectral remote sensing, and (c) providing these datasets, software, and evaluation scripts freely accessible to everyone.

1.3 Outline and Main Contributions

In the following, the structure of this thesis is outlined, and the main contributions in every chapter are summarized. Figure 1.1 illustrates an overview of the structure, divided into six chapters and an appendix. Appendix A lists all publications published within the scope of this thesis.

Chapter 2 – Framework The full range of the ML estimation based on hyperspectral data is presented as a hyperspectral estimation framework. The framework structures the process from a given estimation task to the final estimation in four levels. (i) The sensor level describes the acquisition of hyperspectral data and corresponding ground truth point data. (ii) The data level covers pre-processing, the challenge of dataset shift, data augmentation, and dataset splitting. (iii) The feature level includes the unsupervised dimensionality reduction, unsupervised clustering, feature engineering as well as feature selection. (iv) The model level consists of supervised learning, semi-supervised learning, model selection, the optimization of hyperparameters, and model evaluation metrics. The hyperspectral estimation framework is presented in Chapter 2. Each challenge is tackled with the presented framework and with strategies presented in Chapters 3 to 5.

Chapter 3 – Challenge (I) The acquisition of hyperspectral remote sensing data with modern satellite and UAV technology is getting increasingly affordable over large areas [19]. At the same time, the acquisition of reference data is still expensive in time and resources. Therefore, datasets often include significantly more hyperspectral data than reference data. Both hyperspectral data and reference data are needed for proper training of ML models. This is the first challenge addressed in this thesis. In Chapter 3, a framework of Self-Organizing Maps (SOMs) is introduced as a semi-supervised ML approach. SOMs are underrepresented in today's research [20]. This chapter fills this gap with the following four main contributions:

- the publication of a hyperspectral field campaign dataset,
- the development and publication of a (semi-)supervised SOM framework,
- the application of this framework in the regression of soil moisture, and

- the application of this framework in the classification of land cover.

Chapter 4 – Challenge (II) The second challenge is the limited potential of shallow ML models on high dimensional data. Shallow learners such as Random Forest and Support Vector Machine show adequate performance in hyperspectral applications. They often depend on feature engineering or dimensionality reduction, which adds inductive bias. Inductive bias can support the generalization of the ML model but also limit its potential [21]. Deep learning approaches, such as Convolutional Neural Networks (CNNs), are one solution for this challenge. In the field of hyperspectral remote sensing, CNNs are mostly applied on 2-dimensional (2D) images rather than on 1-dimensional (1D) spectra. The studies presented in Chapter 4 fill this gap in hyperspectral remote sensing research with the following two main contributions:

- the development and publication of deep 1D CNN architectures, and
- the application of these models on an existing soil texture dataset.

Chapter 5 – Challenge (III) The third challenge is dataset shift in hyperspectral datasets. In general, dataset shift occurs if the distributions of the input data and labels of a training dataset differ from the distributions of a test dataset. A consequence of dataset shift for ML estimation is the ability to generalize to new input data [22]. Dataset shift is a common challenge in ML research and hyperspectral remote sensing, but researchers mostly ignore it. Chapter 5 fills this gap with the following four main contributions:

- the publication of a hyperspectral UAV dataset,
- the detection of dataset shift with unsupervised ML approaches,
- the development of a data augmentation method, and
- the study of the effects of dataset shift on supervised soil moisture regression.

Chapter 6 – Conclusions and Outlook The findings of this thesis are summarized in Chapter 6. The conclusion is followed by an outlook of extensions of the presented studies and proposed future studies.

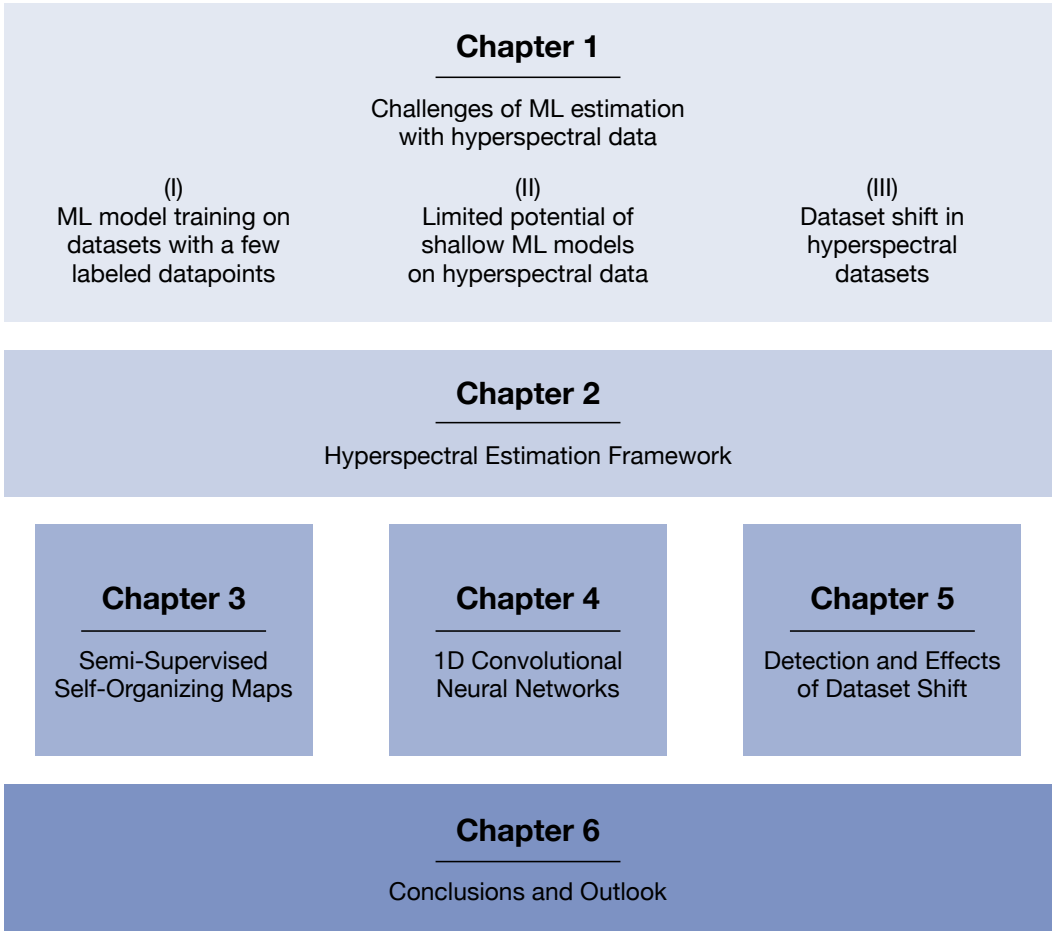


Figure 1.1: Overview of the logical structure of this thesis. In Chapter 1, three main challenges are introduced in the estimation based on hyperspectral data. A hyperspectral estimation framework is the basis for the subsequent studies and is described in Chapter 2. Three studies are presented which address the three main challenges in Chapters 3 to 5. A conclusion of results and an outlook is given in Chapter 6.

2 Hyperspectral Estimation Framework

> *The signal is the truth. The noise is what distracts us from the truth.*
>
> **— Nate Silver**
> (Statistician, Political Forecaster)

This chapter includes material from

Felix M. Riese and Sina Keller. "Supervised, Semi-Supervised, and Unsupervised Learning for Hyperspectral Regression". In: *Hyperspectral Image Analysis: Advances in Machine Learning and Signal Processing*. Ed. by Saurabh Prasad and Jocelyn Chanussot. Cham: Springer International Publishing, 2020. Chap. 7, pp. 187–232. Reprinted with permission. It is cited as [7] and marked in blue.

2.1 Introduction

Hyperspectral data, in general, requires different handling compared to other data when applying Machine Learning (ML). Hyperspectral data is high-dimensional, its spectral bands can be highly correlated, and the amount of ground truth for supervised estimation tasks is often small. In this chapter, a hyperspectral estimation framework is introduced, which covers the ML estimation based on hyperspectral data from the data acquisition to the final estimation. The focus lies on data-driven ML models since they are capable of dealing with non-linear estimation tasks (e.g. [23]). The terms *hyperspectral regression* and *hyperspectral classification* are introduced for the regression or classification solely based on hyperspectral data.

The structure of this chapter is illustrated in Figure 2.1. At first, the fundamentals of hyperspectral regression and classification with different learning techniques and definitions of technical terms are presented in Section 2.2. The target variables

for the ML estimations, applied in this thesis, are described in Section 2.3. After the fundamentals, the four levels of the framework are introduced: sensor level, data level, feature level, and model level. On the sensor level in Section 2.4, hyperspectral data is combined with ground truth. The data level in Section 2.5 is structured into four parts: pre-processing, dataset shift, data augmentation, and dataset splitting. This level is followed by the feature level in Section 2.6, divided into dimensionality reduction, clustering, and feature engineering, as well as feature selection. The fourth level is the model level in Section 2.7. Two supervised models are presented, semi-supervised learning is summarized, and the model selection, optimization as well as evaluation are addressed.

2.2 Fundamentals

In recent years, the hyperspectral remote sensing community mainly focused on classification tasks. Classification refers to the estimation of discrete classes, for example, to distinguish between land cover classes like *water*, *vegetation*, *road*, and *building*. Both classification and regression are about building predictive models. The difference is that, in classification, the target space is discrete (e.g., land cover classes), whereas in regression, the targets are continuous (e.g., soil moisture). [7, Sec. 7.2]

In the context of ML regression, different approaches can be applied depending on the objective and the availability of reference data. In Figure 2.2, these different approaches are visualized schematically. We can distinguish [... three] cases. In case (a), reference data, meaning labels containing the ground truth, is available for all (hyperspectral) input datapoints. In this context, supervised learning models are suitable. A supervised model is able to learn from all available input-output data pairs. In case (b), we have an incompletely labeled dataset. That is, some of the samples are missing the correct ground truth labels. In this context, we can rely on semi-supervised learning models. They learn from the complete input-output pairs as well as from the datapoints without labels. [...] Finally, in the case (c) when no labels are available, unsupervised learning can be applied. Unsupervised learning is useful, for example, for dimensionality reduction and clustering. [7, Sec. 7.2]

The mathematical notation conventions used in this chapter are consistent with [24] [and are listed in Table B.1]: $X = (x_1, \dots x_n)$ is a set of n input datapoints $x_i \in \mathcal{X}$ for all $i \in [n] := \{1, \dots n\}$. Every datapoint x_i consists of m input features. In hyperspectral regression [and classification], the input features represent the m hyperspectral bands and it is $\mathcal{X} \subset \mathbb{R}^m$. In supervised learning, case (a), $y_i \in \mathcal{Y}$

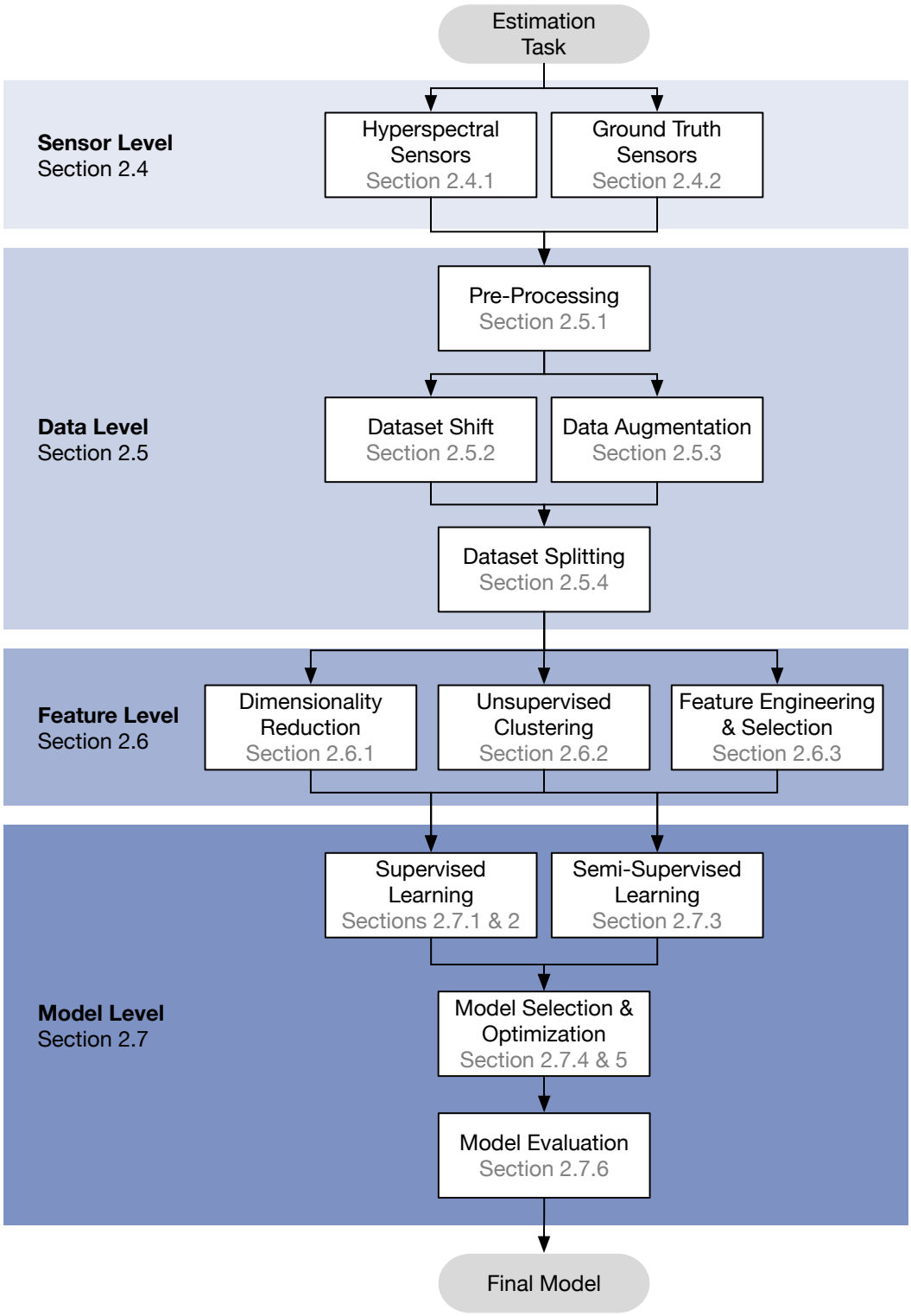


Figure 2.1: Hyperspectral estimation framework divided into the sensor level, the data level, the feature level and the model level. (adapted with permission from [7]).

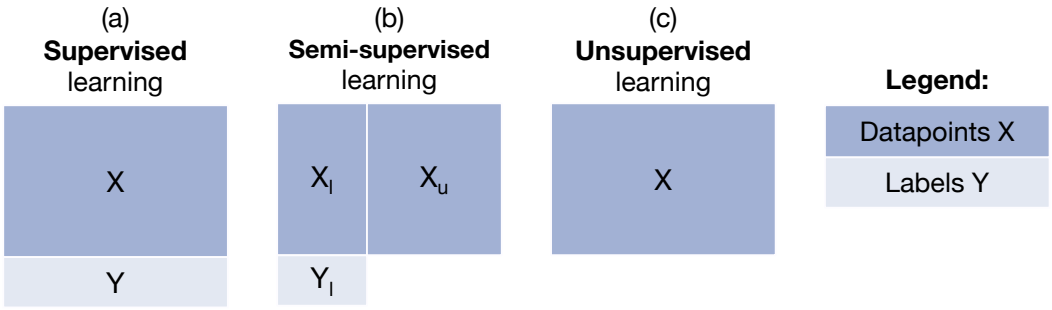


Figure 2.2: Depending on the availability of labels for our training data, we can distinguish three types of learning algorithms: (a) supervised learning, (b) semi-supervised learning, and (c) unsupervised learning. (adapted with permission from [7])

with $Y = (y_1, \ldots, y_n)$ are the labels of the datapoints x_i and the training set is given as pairs (x_i, y_i). In semi-supervised and active learning, cases (b) and (c), the dataset X is divided into two parts. The first part consists of the datapoints $X_l := (x_1, \ldots, x_l)$ with the corresponding labels $Y_l := (y_1, \ldots, y_l)$ and the second parts consists of the datapoints $X_u := (x_{l+1}, \ldots, x_{l+u})$ without any labels. It is $l + u = n$. Again, we have $y_i \in \mathcal{Y}$ for $i = 1, \ldots, l$. For regression, the labels are continuous in the cases (a) to (c) which means $\mathcal{Y} \subset \mathbb{R}$. Note that also more-dimensional labels can be used in regression. In the d-dimensional case, it is $\mathcal{Y} \subset \mathbb{R}^d$. Within the scope of this chapter, we will stick to 1-dimensional (1D) labels, meaning $d = 1$. [For classification, the labels are discrete in the cases (a) to (c).] We refer to this combination of hyperspectral input data and desired output data as *datapoint*. In the unsupervised case (d), the dataset only consists of input datapoints X without any labels. [7, Sec. 7.2]

In the field of hyperspectral remote sensing and in the analysis of hyperspectral data, there are many applications for ML. [...] A general overview of remote sensing image processing with a focus on traditional ML models and physical models is given in [10]. Most current studies address ML classification with hyperspectral data (e.g., overview in [13]) whereas only few studies focus on hyperspectral regression (e.g., [5, 4]). The ML models used for the respective regression [and classification] tasks are described in Sections 2.7 and 2.7.3. [7, Sec. 7.2]

Depending on the hyperspectral regression [or classification] task, we need to select an appropriate ML model [17]. At best, the selected ML model is able to learn all relevant nuances of the training dataset (low bias) and is able to generalize well on unknown datasets (low variance). Accomplishing low bias and low variance at the same time is impossible. Thus, a trade-off between bias and variance [25, 26, 27] has to be addressed while selecting an appropriate model (see Section 2.7.4).

When an ML model is characterized by a low bias (high variance), it is able to adapt well to the training dataset which also includes noise. Such an ML model tends towards overfitting. An ML model with low variance is more robust against noise and outliers. Such an ML model is not able to adapt well to the nuances of the training dataset which is called underfitting. [7, Sec. 7.2]

2.3 Target Variables

The ML applications presented in this thesis focus on the estimation of three variables: soil moisture, soil texture, as well as land cover and land use. In the following, these three target variables are defined, and their relevance is described. Soil moisture is described in Section 2.3.1, soil texture is described in Section 2.3.2, and land cover and land use are described in Section 2.3.3.

2.3.1 Soil Moisture

Soil moisture is defined as the water contained in the part of the soil surface, which is unsaturated [28]. One measure of soil moisture is volumetric soil moisture, meaning the volume of water in a specific volume of soil. Therefore, soil moisture values are dimensionless and commonly provided in percentage. Soil moisture is an essential variable in hydrological, ecological, and climatological systems [28, 29]. Precise data about the soil moisture distribution helps in the modeling of these systems.

In hydrology, the soil moisture dynamics of a river catchment gives a measure of the saturation of the soil. With the knowledge about the degree of soil saturation, the amount of infiltration and runoff from the water of a rainfall event, for example, can be estimated [28]. In climatology, soil moisture is a crucial variable in the global water and energy cycle since the water and heat exchange between the Earth's surface, and the atmosphere depends on the soil moisture [30]. In ecology and, especially, in ecosystems with limited water availability, knowledge about soil moisture helps in plant irrigation management, drought forecasting, and crop yield [31].

Soil moisture on the soil surface is spatially and temporally highly variable. The variability mainly depends on topography, soil texture, organic matter content, soil macroporosity, vegetation density and type, land use as well as climatological and meteorological factors such as precipitation and solar radiation [32, 33, 28]. The water transmission and retention of the soil, for example, depends on

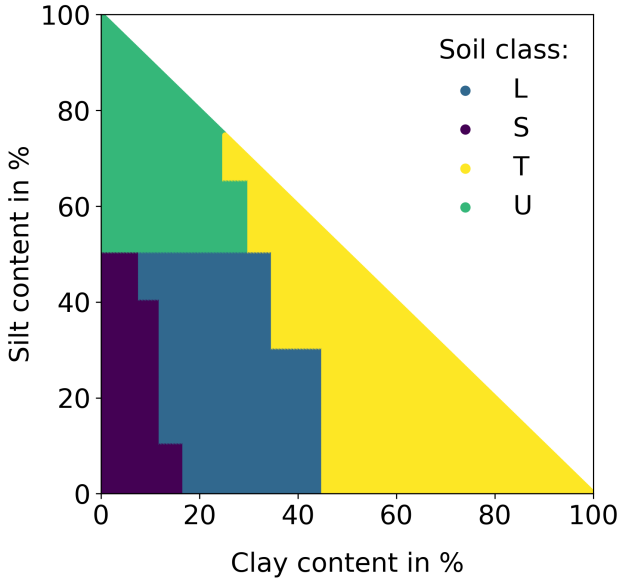


Figure 2.3: Main classes of soil texture according to Bodenkundliche Kartieranleitung ed. 5 (KA5): L (mainly loam), S (mainly sand), T (mainly clay), and U (mainly silt).

the soil texture: soils with larger particle sizes have a lower capacity to hold water than soils with smaller particle sizes [34].

2.3.2 Soil Texture

Soil is the composition of soil particles with various sizes. These soil particles can be classified according to their diameters into clay ($< 0.002\,\text{mm}$), silt ($0.002\,\text{mm}$ to $0.05\,\text{mm}$), and sand ($0.05\,\text{mm}$ to $2\,\text{mm}$). The relative content of clay, silt, and sand is referred to as soil texture. It can be classified, for example, according to the Bodenkundliche Kartieranleitung ed. 5 (KA5) taxonomy [35] or the United States Department of Agriculture (USDA) soil taxonomy [36]. Within the scope of this work, the KA5 taxonomy is used. The KA5 soil classes are listed in [35] with their clay, silt, and sand contents. Figure 2.3 illustrates the four KA5 main classes.

Soil texture is a vital soil property for hydrology and agriculture [37, 38]. The ability to store water depends on the soil texture, which directly influences the availability of water for crops in agriculture. Smaller particle sizes, meaning a higher clay content, leads to higher water and nutrient holding capacity and to less oxygen. Overall, soil texture influences the soil response to environmental conditions like droughts and rainfall.

2.3.3 Land Cover and Land Use

Land cover is defined as the material that covers the Earth's surface. Examples for land cover classes are *forest land*, *water*, and *wetland*. Land use is the human's use and modifications of the Earth's surface. Exemplary land use classes are *cropland*, *orchards*, *residential buildings*, and *industrial buildings*. Since land cover and land use classes overlap, both terms are used synonymously [39].

Changes in land cover significantly impact ecological systems; they affect global warming, erosion of soils, and biodiversity [40]. Further, information about changes in land cover and land use is essential for governments to monitor and manage countries and cities.

2.4 Sensor Level

For the estimation based on hyperspectral data, this data needs to be acquired with appropriate sensors. The first level of the presented hyperspectral estimation framework is, therefore, the sensor level. Datasets for the estimation of hyperspectral data consist of data from a hyperspectral sensor and, depending on the application, of ground truth data. An overview of the hyperspectral sensors applied in this work is presented in Section 2.4.1. The ground truth data is mostly acquired point-wise. In this work, the estimation of soil moisture, soil texture, and land cover is studied based on hyperspectral data. In Section 2.4.2, the most relevant measurement techniques are presented.

For the datasets used in this work, hyperspectral data and ground truth data are combined on different scales. In Chapter 3, the Karlsruhe Lysimeter (KarLy) soil moisture dataset on a small field scale is presented and applied [41, 42]. The Land Use/Cover Area Frame Survey (LUCAS) dataset is used in Section 4.3 for the estimation of soil texture [43, 44, 45]. This dataset includes data from soil samples all over Europe. The Aerial Peruvian Andes Campaign (ALPACA), introduced and used in Chapter 5, consists of hyperspectral Unmanned Aerial Vehicle (UAV) data combined with soil moisture in situ measurements on field scale. An additional dataset is used in further studies [46, 23, 47] for the estimation of soil moisture: the Field experiment dataset of surface-subsurface infiltration dynamics acquired by hydrological, remote sensing, and geophysical measurement techniques (HydReSGeo) [48].

2.4.1 Hyperspectral Sensors

Hyperspectral sensors are passive, optical sensors. These sensors are designed for their particular purposes and applications. Depending on the number of bands and the width of the bands, a sensor can be considered monospectral, multispectral, or hyperspectral. Sensors with one band are considered *monospectral*, sensors with approximately 2 to 15 relatively wide bands are considered *multispectral*, and sensors with approximately 100 to 1000 narrow bands are considered *hyperspectral*. Within the scope of this thesis, several different hyperspectral and multispectral sensors are applied. The sensors are differentiated by the platform on which they are mounted, the spectral range, spectral resolution (number and width of bands) as well as the output type. The following platforms are considered: a static tripod, a handheld sensor, UAV, airplane, and satellite. Additionally, hyperspectral sensors can be applied in a laboratory setup.

Three spectral ranges are considered: Visible and Near-Infrared (VNIR) from approximately 400 nm to 1400 nm, Short-Wavelength Infrared (SWIR) from approximately 1400 nm to 3000 nm, and Thermal Infrared (TIR) from approximately 3 μm to 15 μm. The output format of the sensors can be a single spectrum (spectrometer), line data (line scanner), and images (snapshot). A qualitative overview of the four primary hyperspectral sensors applied in this work regarding the spatial, temporal, and spectral resolutions is shown in Figure 2.4. The spatial resolution refers to the sensor's pixel size, the temporal resolution refers to the number of measurements that are possible during a specific time interval, and the spectral resolution refers to the number and width of the bands. The more bands and the smaller the width of the bands, the larger the spectral resolution gets. All applied sensors and their properties are listed in Table 2.1.

2.4.2 Ground Truth Measurements and Sensors

This work focuses on the estimation of soil moisture, soil texture, and land cover. The relevance of these variables is explained in Section 2.3. The ground truth of these variables comes from direct observation instead of inference. While the ground truth of land cover is generated manually by experts, the ground truth of soil moisture and soil texture requires specific sensors. This section provides an overview of measurement techniques for soil moisture and soil texture. In comparison to area-wide hyperspectral sensors, these ground truth measurements are performed point-wise.

Table 2.1: Overview of the hyperspectral sensors applied within the scope of this thesis.

Sensor name	Sensor platform	Sensor type	Spectral range in nm	Number of bands	Own studies
Hyperspectral Devices ROX	Tripod	Spectrometer	400 to 950	769	[9]
Cubert UHD 285	Tripod	Snapshot (50×50 pixels)	450 to 950	125	Chapter 3 and [42, 41, 23, 6, 7, 48, 49]
ASD FieldSpec 4	Handheld	Spectrometer	350 to 2500	2151	[46, 48, 49]
FOSS XDS Rapid Content Analyzer	Laboratory	Spectrometer	400 to 2500	4200	Chapter 4 and [8]
AVIRIS	Airplane	Line scanner	400 to 2500	224	Chapter 3 and [6]
ESA Sentinel-2	Satellite	Line scanner	420 to 2370	13	[50, 51]
Headwall Hyperspec SWIR	UAV	Line scanner	900 to 2500	170	Chapter 5
FLIR Tau 2 640	Tripod	Snapshot (640×512 pixels)	7500 to 13 500	1	[23, 52, 48]

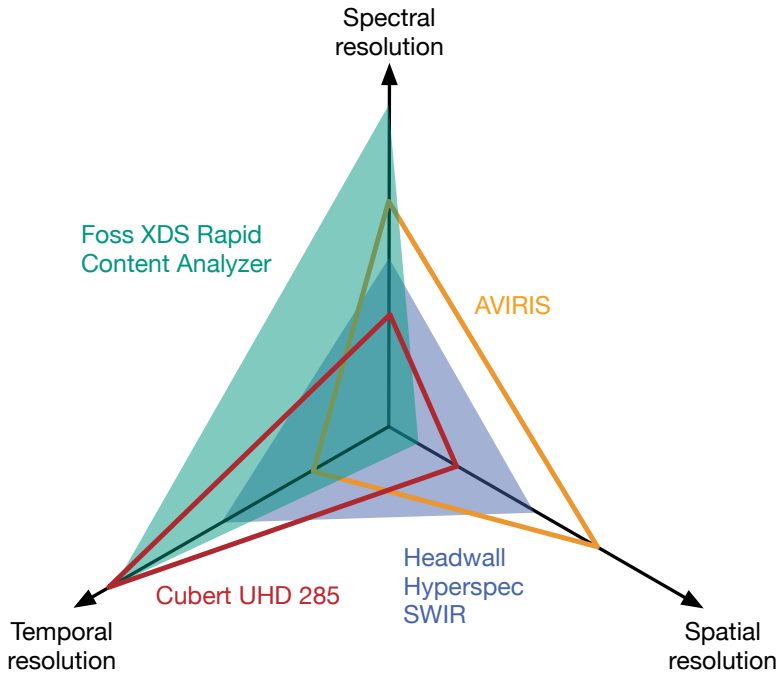


Figure 2.4: Qualitative overview of the hyperspectral sensors mentioned in this work in regards to the spatial, temporal and spectral resolutions.

Soil moisture can be measured, for example, point-wise with gravimetric methods, with nuclear methods, electromagnetic measurement techniques, tensiometer methods, and hygrometric methods [53]. The measurements are either performed directly at the measurement site (in situ) or performed indirectly by analyzing a soil sample in the laboratory. In contrast, area estimations are performed regionally on one area per measurement. Area estimations with remote sensing techniques can only measure soil moisture of the upper few centimeters of the soil. A more detailed overview of soil moisture estimation from remote sensing can be found in [54]. Since every soil moisture measurement technique comes with its limitations, there is not a single superior measurement technique [55]. In the following, the gravimetric method, are described as well as the two electromagnetic measurement techniques, which are applied in the presented studies in Chapters 3 and 5: Time Domain Reflectometry (TDR) sensors and capacitive sensors.

Gravimetric methods are based on drying a soil sample in the laboratory. The weight of the soil sample before and after the drying process is measured. The difference of both weights refers to the water content in the soil sample and, therefore, to its soil moisture. This measurement technique is invasive since it requires a soil sample. Therefore, it is mostly used as a baseline and for the calibration of other methods. Electromagnetic measurement techniques for soil moisture include resistive sensors,

capacitive sensors, TDR sensors, and frequency domain sensors [55]. These methods are based on the electromagnetic characteristics of water and, therefore, soil moisture. Capacitive sensors measure soil moisture by measuring the capacitance between electrodes in the soil. TDR sensors emit an electric pulse into an electrode located in the soil. The reflection of this pulse correlates with the soil moisture around the TDR sensor. TDR sensors are widely applied since they are non-invasive and easy to apply [56]. A quality assessment of TDR sensors is presented in [57]. Further details and other methods can be found in [28].

The soil texture of a soil sample can be determined with a laboratory method, referred to as Particle Size Analysis (PSA) [58]. In the PSA, the different particle sizes within the soil sample are separated from each other. The resulting distribution can be used with a taxonomy such as the KA5 taxonomy [35] or USDA soil taxonomy [36], as described in Section 2.3.2. This laboratory method is applied in the dataset used in the studies of Chapter 4, as described in [44, 43].

Land cover and land use ground truth can be acquired in a field survey, resulting in a land cover map. The exact classes have to be set beforehand. The number of different classes depends on the application of the dataset to be acquired.

2.5 Data Level

In the following, the data level as the second level of the presented hyperspectral estimation framework is described. The data level is divided into four parts: pre-processing, dataset shift, data augmentation, and dataset splitting. In Section 2.5.1, the pre-processing is described focusing on collecting, validating, and preparing the data. The second part of the data level addresses the challenges of dataset shift in Section 2.5.2. This includes possible approaches to cope with dataset shift. Data augmentation is presented in Section 2.5.3, including basic image manipulations, Monte Carlo (MC) augmentation, and Generative Adversarial Networks (GANs). The data level is concluded by several approaches of dataset splitting in Section 2.5.4. [7, Sec. 7.3]

2.5.1 Pre-Processing

> The first part of the presented hyperspectral [... estimation framework] is pre-processing [10, 59]. We divide the pre-processing into three steps: reading in data, preparing data, and validating data. First, we need to read in the data. In

Python, datasets can be conveniently read in using existing and established software packages such as *Pandas* [60] or *TensorFlow* [61]. [7, Sec. 7.3.1]

The second step of pre-processing is the validation of the data. We highly recommend to explore the dataset before further processing [...]. The exploration procedure could include a check of the value range of the input features and the target variable. The data validation can be achieved by an analysis of the datasets statistics and by a visualization of the dataset. Thus, we obtain an overview of the used dataset. Additionally, we recognize possible challenges in the dataset such as outliers, missing values or labels as well as dataset shift at an early stage. The latter is addressed in detail in Section 2.5.2. A useful example to motivate the investigation of the dataset with statistical methods and visualizations is given in [62]. [7, Sec. 7.3.1]

The last step of pre-processing is the preparation of the data. Depending on the results of the data validation and the applied ML models (see Sections 2.7 and 2.7.3), the dataset might need to be normalized or transformed. The data normalization makes the training less dependent on the scale of the input data. Typical normalization techniques scale the numerical data, for example, linearly between 0 and 1, or around 0 with a standard deviation of 1. Additionally, it might be necessary to transform categorical data to numerical values since some ML models like Artificial Neural Networks (ANNs) (see Section 2.7.2) only work with numerical data. A common way to achieve this is *one hot encoding*. Each categorical feature is represented by one entry in a binary vector. [7, Sec. 7.3.1]

2.5.2 Dataset Shift

Most ML models rely on the *Independent and Identically Distributed (i.i.d.)* assumption. The i.i.d. assumption refers to the independent collection of the training dataset and new, unknown datasets (see Section 2.5.4) which are identically distributed. [7, Sec. 7.3.2]

In this context, the term *training dataset* refers to the dataset that is available during the training of the ML model. For example, in hyperspectral regression of soil moisture, the hyperspectral data as well as the ground truth labels of soil moisture should cover all (in reality) possible values. Otherwise, dataset shift occurs and the estimation performance might suffer [22, 63]. In general, three main types of dataset shift exist: [7, Sec. 7.3.2]

- Covariate shift [64, 65] is defined as a change of the input feature distribution P(X). It is the best studied type of dataset shift in the literature. For example in hyperspectral regression of soil moisture, rainfall events between two measurement days affect the input feature distribution of this two-day dataset.
- Prior probability shift [66, 63] is defined as a change of the target variable distribution P(Y) without a change in X. This change mostly occurs in the application of generative models. For example, in the hyperspectral regression of soil moisture, the distribution of soil moisture can vary due to the underlying soil structure while the soil surface remains unchanged.
- Concept shift [67] or concept drift is a change in the relationship between the input data and the target variable. The concept shift is the most challenging type of dataset shift to handle. For example, in hyperspectral regression of chlorophyll *a* concentration, the relationship between hyperspectral input data and chlorophyll *a* concentration as target variable can change due to undetectable hydrochemical processes. [...]

Several causes of dataset shift exist. One cause is the sample selection bias. Sample selection bias can occur in the scope of different data measurements. In hyperspectral regression [or classification], it often occurs as a result of the parallel use of different hyperspectral sensors and changes of the measuring site. Another cause for dataset shift [... are] non-stationary environments. Non-stationary environments appear when the training environment differs from the test environment. This distinction can be temporal or spatial. Since hyperspectral satellites record data at different locations and during different seasons, dataset shift commonly occurs. [7, Sec. 7.3.2]

Various ways exist to deal with the challenges of dataset shift. In most ML studies for hyperspectral regression [or classification], dataset shift is simply ignored. In this case, the applied model is static with regards to the dataset shift. Such models can be used further as a baseline model allowing the detection of dataset shift and enabling the evaluation of approaches aiming at the reduction of the effects of dataset shift. [7, Sec. 7.3.2]

A first approach to reduce the effects of dataset shift is to re-fit or update the ML model to new data. In the case of time series, this means re-fitting or updating the ML model on more recent data. In the case of 2-dimensional (2D) areal data, this means re-fitting on more training areas. Another approach is to re-weight the training dataset based on temporal (time series) or spatial (2D data) features. For example, training data of time series can be re-weighted so that newer datapoints are more important in the training than preceding ones. Fur-

ther, the ML model can be set up to inherently learn temporal changes to reduce the bias of seasonality and timing. [7, Sec. 7.3.2]

2.5.3 Data Augmentation

Limited dataset size, in general, is a challenge for ML models. Especially for deep architectures of ANNs and for estimations based Convolutional Neural Networks (CNNs), a large dataset size is crucial (see Section 2.7.2). Sensor uncertainties and noise are other challenges in the estimation based on hyperspectral data. In this section, three data augmentation approaches are presented: data augmentation with basic image manipulations, MC data augmentation, and GANs. An overview of data augmentation approaches in deep learning is presented in [18, 68].

Data augmentation with basic image manipulations includes flipping, rotating, and translation of images as well as color space transformations [68]. That way, one image of a dataset can be used several times in the training of a ML model. Current ML Python packages such as *TensorFlow* [61] already come with built-in image manipulation methods. In [69], the implementation of these manipulation methods is provided for multispectral satellite data.

Monte Carlo (MC) data augmentation is based on the generation of random samples according to distributions which depend on the experiment at hand. For example, a normal distribution of sensor uncertainty is often assumed to simulate sensor noise in the data augmentation. The normal distribution results from the central limit theorem (e.g. [70]). The probability density for the normal distribution p(x) is defined, with the mean μ and the variance σ^2 of the measured soil moisture values x, as

$$\mathrm{p(x)} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\mathrm{x}-\mu)^2}{2\sigma^2}\right). \tag{2.1}$$

For MC data augmentation, the generation of pseudo data, a random data generator is needed. In general, it is recommended to apply existing, well-tested random data generators. Within the scope of this work, a random generator from the Python libraries SciPy and NumPy [71] is used, which is based on PCG64 [72]. In reference [47], the augmentation of soil moisture data from two sources is studied in the estimation of soil moisture from hyperspectral data. Two different MC augmentation datasets, augmentation over time and space, are included.

An overview of hyperspectral data augmentation is presented in [73]. MC data augmentation is applied in Section 5.5.1.

Generative Adversarial Networks (GANs) [74] are an upcoming ML model which was introduced for unsupervised data augmentation. GANs consist of two ANNs: a generator network and a discriminator network. The generator network learns to generate new data samples. In combination with the existing (real) training data, these new (fake) samples constitute the input data for the discriminator network. The discriminator network learns to differentiate between real and fake input data. During the training of a GAN, both networks learn to improve their performance regarding their respective task. Finally, a trained GAN is able to generate new training data which can be used to augment the existing training dataset [...]. [7, Sec. 7.6.1]

A detailed overview on GANs is presented in [75]. In hyperspectral regression, GANs are not commonly used so far, although there are different applications in classification. For example, the implementation of GANs and their applications on open hyperspectral classification datasets is presented in [76]. A more complex approach combining GANs with Semi-Supervised Learning (SSL) is presented in [77]. [7, Sec. 7.6.1]

2.5.4 Dataset Splitting

To evaluate the generalization abilities of an ML model, the full available dataset needs to be split into smaller datasets. In general, dataset splitting should meet the i.i.d. assumption (see Section 2.5.2). [... There are two commonly applied types of dataset splitting.] In the first type, the full dataset is split into two subsets: *training* and *test*. In the second type, the three subsets training, *validation*, and test are generated. In both split types, the training dataset is used repeatedly to train the ML model. The test dataset is used only once to evaluate the final ML model. The split types differ with respect to the way the ML models are optimized (see Section 2.7.5). In the 3-subset split, the validation dataset is repeatedly used for the evaluation of the generalization abilities of the ML model in the optimization process. In the 2-subset split, the training dataset is used for both training and evaluation in the optimization process by applying a k-fold cross-validation. Within the k-fold cross-validation, the training dataset is randomly partitioned into k subsets of similar size. One of the k subsets is then used for the evaluation of the ML model, while the remaining $k-1$ subsets are used for the training of the ML model. This selection is repeated so that every subset is used once as validation subset. Note that it is not trivial to apply k-fold cross-validation on time series due to possible casual relationships. [7, Sec. 7.3.3]

After deciding on the number of dataset subsets, the splitting approach needs to be defined. [...] They are described in detail in [78] with their respective strengths and weaknesses. The most commonly applied splitting approach is a random split or random sampling (e.g., [79]). The subsets are randomly sampled which leads to subsets with relatively similar target variable distributions. However, for spatially or temporarily correlated data like 2D hyperspectral image data or time series, a pixel-wise random split can lead to biased subsets. Since a significant number of datapoints in one subset have direct spatial or temporal neighbors, the datapoints are highly correlated in between the subsets. Training on one datapoint and evaluating the model performance on a neighboring datapoint leads to highly biased results. [7, Sec. 7.3.3]

In [7], further splitting approaches are presented and discussed: systematic splitting, patch splitting and stratified splitting. Within the scope of this thesis, we rely on random splitting as discussed in the relevant sections.

2.6 Feature Level

The feature level is the third level of the presented hyperspectral estimation framework. It consists of unsupervised dimensionality reduction (Section 2.6.1), unsupervised clustering (Section 2.6.2), and feature engineering as well as feature selection (Section 2.6.3).

2.6.1 Dimensionality Reduction

Since correlations and redundancies between input features can occur, the virtual dimensionality of a dataset is often smaller than the given dimensionality [80]. The term *dimensionality reduction* refers to the reduction of the dimension m of the input data to a smaller dimension $m_r \leq m$ toward the virtual dimensionality. In addition, the term *compression* focuses on the reduction of the dimension m of the data to the smallest possible $m_{min} \leq m_r \leq m$. In most cases after applying dimensionality reduction, it is only possible to reconstruct *similar* data, not the original input data. The topic of dimensionality reduction and compression in general is reviewed in detail in [81, 82]. Note that the term *feature extraction* is often used instead of dimensionality reduction (see e.g., [83]). [7, Sec. 7.4.1]

A more recent overview of dimensionality reduction approaches is presented in [16]. The authors provide an overview of state-of-the-art dimensionality reduction ap-

proaches, divided into shallow and deep approaches. Overall, they compare 15 approaches in terms of their impact on the accuracy of the exemplary classification task.

We discuss in the following the most relevant approaches of dimensionality reduction in hyperspectral regression [... and classification]. A commonly applied approach is the Principal Component Analysis (PCA) [84]. The PCA transforms the input data orthogonally based on the variance along newly found axes. These new axes are referred to as principal components. The principal components are sorted by decreasing variance. That is, the first principal component has the largest variance. Therefore, the set of the first few principal components contain most of a dataset's variance and at best, most of the information contained in the dataset. [...] [7, Sec. 7.4.1]

Autoencoder (AE) [85] is an ANN approach for dimensionality reduction. An AE consists of an input layer of input dimension m, followed by several hidden layers with smaller dimension $m_{hidden} < m$ and an output layer of size m. The dimension reduction of input to hidden layers is called *encoding*. In the encoding, the AE finds a lower-dimensional representation of the input data. The dimension increase of the encoded data in the original dimension m is called *decoding*. The AE is trained in an unsupervised manner with the hyperspectral data for both input and (desired) output data. Then, the encoding part of the trained AE can be used for dimensionality reduction on the (hyperspectral) input data. In sum, the full AE with encoding and decoding can also be used for noise removal (denoising) of the hyperspectral input data. More details about ANN are presented in Section 2.7.2. Since an AE consist of many free parameters, large training datasets are necessary for the training. Note that only the number of hyperspectral input datapoints needs to be large for the AE. The dataset that includes ground truth labels can be small. Since the combination of large input data and small ground truth data is characteristic for multi- and hyperspectral satellite data, AE is well-suited in this context. [...] [7, Sec. 7.4.1]

Finally, we list two additional dimensionality reduction approaches. The t-Distributed Stochastic Neighbor Embedding (t-SNE) [86] is a non-linear approach which reduces high-dimensional input data to a dataset with the dimension $m_r \in \{2, 3\}$. Therefore, this approach is well suited not only for dimensionality reduction but for the visualization of a dataset as well. A recently presented dimensionality reduction approach is called Uniform Manifold Approximation and Projection (UMAP) [87]. UMAP is comparable with the t-SNE algorithm incorporating several advantages in terms of speed and performance. Since UMAP is a relatively new approach, it has to be investigated further in context of hyperspectral regression [and classification]. [7, Sec. 7.4.1]

2.6.2 Unsupervised Clustering

Clustering a dataset means the grouping datapoints with respect to a pre-defined similarity metric. Datapoints are clustered, mostly in an unsupervised manner, based on the input features such as hyperspectral bands.

The original publication [7] describes and compares the k-means clustering [88], the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [89], and unsupervised Self-Organizing Map (SOM) [90]. We omit this part because Chapter 3 introduces a framework of unsupervised, supervised, and semi-supervised SOMs in detail [6].

2.6.3 Feature Engineering and Feature Selection

Feature Engineering In Sections 2.6.1 and 2.6.2, we apply dimensionality reduction and clustering to generate new features. In contrast to these data-driven approaches, feature engineering is based on prior knowledge. The generated features can be categorized as spectral features or spatial features. The engineering of spectral features is inspired by physical processes. Spectral features are commonly characterized by a ratio or the normalized difference of hyperspectral bands. The most popular example in hyperspectral regression [and classification] is the Normalized Difference Vegetation Index (NDVI) [91] which corresponds to photosynthesis processes. Spatial features are often generated based on contextual information of neighboring pixels (datapoints). Examples for spatial features are objects, edges, and contours. They are generally created by the application of filters. Note that spatial features can only be generated [... for] hyperspectral images when their corresponding spatial resolution is adequate. [...] [7, Sec. 7.4.3]

In recent years, the application of (manual) feature engineering has decreased in hyperspectral regression [and classification]. Deep ANNs, which are able to learn new low-level and high-level features automatically (see Section 2.7.2), are the main reason for this development. Admittedly, incorporating domain knowledge into data-driven ML models might still improve their performance. Especially, the estimation of physical parameters in hyperspectral regression can be improved by including domain knowledge if such knowledge is available. An overview of the domain knowledge integration into ML models is given in [92] including a review and a consistent taxonomy on previous research. The authors distinguish between four possible approaches to include prior knowledge into ML models [92]: [7, Sec. 7.6.2]

- Integration of the knowledge into the training data by feature engineering (see Section 2.6.3) and simulations (see Section 2.5.3).
- Integration of the knowledge into the hypothesis space, for example by choosing an appropriate ML model such as CNNs for 2D hyperspectral data in the case of locality and translation invariance (see Section 2.7.2).
- Integration of the knowledge into the training algorithm, for example by modifying the loss function.
- Integration of the knowledge into the final hypothesis, for example by including physical constraints on the output variable.

Domain knowledge is included in a data-driven ML model, mainly to increase the model's ability to generalize on new, unknown data. The domain knowledge itself adds a bias to the ML model, which is referred to as *inductive bias* (e.g. [21]).

Feature Selection In contrast to feature engineering, feature selection describes the process of selecting a subset of all available input features which can be used as input data for supervised ML models. In context of hyperspectral regression [and classification], the term *band selection* is often used instead of the term feature selection. The main advantage of feature selection over feature engineering or dimensionality reduction is that the features (hyperspectral bands) are physically meaningful. For example, principal components can not be interpreted physically (see Section 2.6.1). Therefore, feature selection applied on data of one sensor can be transferred to data of another sensor with slightly different hyperspectral bands. [...] [7, Sec. 7.4.3]

An overview of feature selection is presented in [93]. A review on several applications of feature engineering and feature selection in the context of remote sensing image processing is provided in [10]. [...] The use of feature selection and feature engineering depends on the dataset as well as on the applied supervised ML model. [7, Sec. 7.4.3]

2.7 Model Level

Hyperspectral regression and classification is based on mapping hyperspectral input data with desired output data with a specific ML model. Two supervised ML approaches are presented in the following: tree-based models (Section 2.7.1) and ANNs (Section 2.7.2). An overview of semi-supervised learning based on hyperspectral data is given in Section 2.7.3. The selection of an appropriate ML model

is addressed in Section 2.7.4, followed by the hyperparameter optimization (Section 2.7.5) and the model evaluation metrics (Section 2.7.6). This chapter is followed by Chapter 3, which is about a specific type of ML model, the supervised SOMs. Afterward, deep 1D CNNs are introduced and applied in Chapter 4.

2.7.1 Tree-based Models

Tree-based regression is based on Decision Trees (DTs). DTs consist of a root node and leave nodes connected by branches. The basic idea is to split the training dataset at every branch into subsets based on the input features, for example hyperspectral bands. In the best case, this split leads to leafs at the end of the branches containing similar values of the respective physical parameter to be estimated. The algorithm of DT regression is defined as follows [94] [7, Sec. 7.5.1.2]:

1. Start with the root node.
2. Start with the most significant input feature (hyperspectral band) of the training data, for example according to the Gini impurity.
3. Divide the input data with a (binary) cut c_1 on that input feature x_i, for example according to the Gini impurity.
4. Divide data along the next best feature on cut c_j for $j = 2, 3, \ldots$ which are calculated similarly to step 3.
5. Stop if a condition is met, for example, maximum number of nodes, maximum depth, or maximum purity.
6. Then, the ground truth labels of the datapoints are averaged for every individual leaf. Finally, every leaf contains one output value.

DTs can also be applied for the classification of discrete target variables. The algorithm changes in step 6. The averaging of the ground truth labels over every individual leaf is replaced, for example, by a majority vote.

In the context of regression, the trained DT is applied for the estimation of the physical parameter. Every input datapoint is mapped onto a leaf containing the respective output value. In steps 2 and 4, the DT algorithm finds the most important feature at each branch in order to divide the dataset into more homogeneous subsets. For this reason, most software implementations of the DT algorithm return a trained estimator and an importance ranking of each input feature. This ranking is called feature importance. In the case of regression with hyperspectral data, the importance ranking refers to the hyperspectral bands. The implementation of the feature importance differs depending on the applied software. For example, the

feature importance can be based on the permutation of the respective values of each input feature. The bigger the influence of an input feature on the regression [or classification] performance, the more important it is. [7, Sec. 7.5.1.2]

[...] To address the issue of overfitting of a single DT, an ensemble of trees can be used. In the following, we focus on two ensembling techniques: bagging and boosting. The main idea of bootstrap aggregation, or bagging, is to average over a number of estimators trained on slightly different training datasets. In case of tree-based regression, the average is calculated over multiple DTs with different setups or training datasets. The trees are trained in parallel. Random Forest (RF) is one implementation of bagging with DTs [95]. Its algorithm is defined in the initialization (step 1) and three repeated steps (steps 2 to 4) [7, Sec. 7.5.1.2]:

1. Initialization: Set the number of trees B, the number of features is m.
2. Bootstrap: Sample learning batch containing n datapoints with replacement from a dataset with n datapoints. There should be $n_1 \approx 2/3n$ different samples.
3. Feature bagging: At every node, a random subset of $m_{bag} = \sqrt{m}$ features are used for the splitting. This leads to a decreasing correlation between the different trees.
4. Regression [or classification]: See DT algorithm above.

Every tree only uses between 60 % to 70 % of the datapoints for the training process. The remaining 30 % to 40 % of the datapoints can be used to evaluate the estimation performance of the respective trees. The regression error of these trees on their ignored datapoints is called *out-of-bag error* and is a good estimate for the generalization error of the ML model. This reduces the need for an extra validation dataset. [7, Sec. 7.5.1.2]

Extremely Randomized Trees (ET) are a modification of the RF algorithm [96]. Compared to the RF algorithm, the splitting process for each node is modified. Randomized thresholds are calculated for each feature of the random subset. Finally, the best threshold is used for the split in the respective node. This modification leads to less variance and increases the bias [of the model]. [...] [7, Sec. 7.5.1.2]

Boosting is another technique to improve the regression based on DTs [97, 98]. It relies on learning multiple estimators which are incrementally generated and improved. Gradient Tree Boosting (GTB) as an example of a boosting algorithm applies a gradient descent optimization [99, 100]. Shallow trees are fitted iteratively on the negative gradient of the loss function. [7, Sec. 7.5.1.2]

With respect to bias and variance, the two tree ensembling techniques differ. In bagging, fully grown DTs are used. By decreasing the correlation between the trees, the variance of the estimator also decreases. The bias remains unchanged. In boosting, relatively shallow trees are used which implies a high bias and a low variance. The ensemble of these shallow trees, the boosted model, reduces mostly the bias. [...] [7, Sec. 7.5.1.2]

In hyperspectral regression, tree bagging techniques such as RF or extremely randomized trees are one of the most-frequently used regression models. For example, RF models are applied to estimate biomass with a smaller dataset of WorldView-2 satellite images [101]. The feature importance was used to create new input features. A similar approach was pursued for the estimation of sugarcane leaf nitrogen concentration based on Earth-Observing One (EO-1) Hyperion hyperspectral data [102]. Compared with RF models, ET perform consistently better several regression tasks such as estimating, for example, soil moisture and chlorophyll *a* concentration [9, 23, 103, 104, 105, 47]. According to these studies, the additional randomization seems to improve the regression performance. [7, Sec. 7.5.1.2]

Similarly, RF is one of the standard approaches in hyperspectral classification. For example, RF is applied in the classification of land cover classification [106, 107]. In [108], the classification of vegetation types based on Airborne Visible Infrared Imaging Spectrometer (AVIRIS) data and the classification of land cover based on Hyperion data is presented.

It appears that boosting is less common in hyperspectral regression [and classification]. GTB, for example, is applied in context of soil characterization with hyperspectral data in the range of 350 nm to 2500 nm [109, 110]. To optimize the application of GTB, a combination of gradient boosting with Partial Least Squares (PLS) is introduced for high-dimensional data [110]. [7, Sec. 7.5.1.2]

2.7.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) based on perceptrons and backpropagation have been around since the 1960s. With increasing computing power and the increasing availability of large training datasets, ANNs significantly have grown in popularity in the 2010s. Subsequently, we give an overview of the different types of ANNs and their applications in hyperspectral regression [and classification]. A comprehensive introduction to ANNs can be found, for example, in [25, 18]. [7, Sec. 7.5.1.5]

Inspired by biological neural networks, ANNs are networks of artificial neurons. These neurons consist of input connections from predecessor neurons as well as an activation function depending on the weighted inputs and a defined threshold. With the activation function, the neuron output is calculated which is then forwarded to the subsequent neurons. The connections between the neurons are weighted. Through the adaptation of these weights, the ANN is adapted to a regression problem, for example through backpropagation. The most common ANN architecture for regression tasks is a fully-connected network. It consists of several neurons organized in consecutive and connected layers. [7, Sec. 7.5.1.5]

A deep neural network is a network with several hidden layers. Hidden layers are located between input and output layers. Deep ANNs are able to learn hierarchical, meaning that lower-level features are learned in the first layers while higher-level features are composed in the following layers. This way, a deep ANN is able to adapt to more complex tasks. [...] An overview of deep learning applications in image analysis with hyperspectral data is given in [12]. The application of deep learning in hyperspectral classification tasks is illustrated in [14]. [7, Sec. 7.5.1.5]

A typical challenge of applying deep learning is that with increasing number of layers and neurons, the number of trainable parameters increases. This may lead to overfitting. To prevent overfitting, regularization techniques such as L2 regularization, dropout and batch-normalization are introduced. L2 regularization adds an L2 term of the weights to the loss function, dropout deactivates neurons randomly during the training iterations and batch-normalization normalizes the output of each layer per batch of datapoints. [7, Sec. 7.5.1.5]

There are several types of ANN with different characteristics. Convolutional Neural Networks (CNNs) [111] are designed to reduce the number of weights and therefore the free parameters. The main idea of CNNs is to extract local features which are translation invariant. CNNs consist of filter layers which are convolved with the input data. In most cases, the input data for CNNs consists of 2D images. Because the filters are convolved with the data instead of having one filter per input dimension, this technique is called *weight sharing*. Weight sharing significantly reduces the number of free parameters which need to be trained. These filters are learned in the training process in contrast to the hand-engineered filters in classical image processing. Many popular CNN architectures also include pooling layers. Pooling layers reduce the input data by a factor. A CNN often includes fully-connected layers at the end. [7, Sec. 7.5.1.5]

In general, ANNs and CNNs can be trained from scratch by initializing all weights randomly and iteratively adapting the weights to the training dataset. Transfer

learning is an alternative approach [112]. Networks are pre-trained on an existing and similar dataset and are then refined on the actual task. For example, a popular dataset for 2D images is ImageNet [113]. Pre-trained networks such as VGG16 [114] and ResNet50 [115] are freely available and can be used for own classification or regression tasks. Transfer learning can save significant amounts of time compared to training from scratch since less training iterations of the network weights are necessary. [7, Sec. 7.5.1.5]

Recurrent Neural Networks (RNNs) [116] are another type of ANN. They learn from sequences like time series. For long sequences, the gradients can vanish. Long Short-Term Memory (LSTM) networks [117] solve this issue with the help of gates (update, forget, output). In recent years, RNN and LSTM, have been used mainly in natural language processing. In the context of hyperspectral remote sensing, time series such as satellite images of different dates pose possible applications of these network architectures. [7, Sec. 7.5.1.5]

ANNs are widely applied in hyperspectral regression [and classification]. An early overview of their use and their opportunities in remote sensing is given by [118]. For example, ANNs are applied on hyperspectral spectroscopy to estimate rice nitrogen status [119]. As a finding of this study, the authors emphasize the extensive need of hyperparameter tuning of the ANN. Furthermore, the results imply that dimensionality reduction (see Section 2.6.1) can increase the estimation performance. A comparison of backpropagation ANNs and further regression models for the estimation of pigment content in rice leaves and panicles is shown in [120]. With hyperspectral input data, ANNs noticeably outperform the compared models. Regarding the estimation of chlorophyll *a* and soil moisture, ANNs perform strongly especially with input data normalization and dimensionality reduction [9, 23]. [7, Sec. 7.5.1.5]

The primary applications of CNNs on hyperspectral data cover, so far, only classification tasks (e.g., [8] [and Chapter 4]). An example of hyperspectral regression of soil clay content with 1D CNNs on a large European dataset is presented in [121]. Instead of applying a commonly applied spatial 2D CNN, the introduced CNN convolves along the spectral axis. Furthermore, this study is a good example that using transfer learning provides acceptable results. Up to now (2020), there is no relevant published study about the application of basic RNNs in hyperspectral regression. However, LSTMs are used to estimate crop yield in combination with CNNs in [122]. The results of this study emphasize the potential of LSTMs. [7, Sec. 7.5.1.5]

Innovations with respect to ANN architectures are continuously presented and applied in ML research. For example, hierarchical neural networks such as attention networks [123] are a promising architecture alternative to LSTMs (see

Section 2.7.2) when analyzing sequential data. Further developments involve capsule networks [124], which use vectors rather than scalars to represent input features. Capsule networks might improve the estimation performance of networks, for example in the hyperspectral image classification [125]. [7, Sec. 7.6.3]

A further trend in ML is the automation of the ML workflow, often referred to as Automated Machine Learning (AutoML). AutoML can include the automation of steps like pre-processing (Section 2.5.1), dimensionality reduction (Section 2.6.1), feature engineering and feature selection (Section 2.6.3), ML model selection and optimization (Sections 2.7.4 and 2.7.5). An overview of AutoML is given in [126]. While AutoML simplifies and speeds up the application of ML for the user, we emphasize that AutoML is not a universal solution for all hyperspectral regression [and classification] problems. Two relevant implementations of AutoML are auto-sklearn [127, 128] and TPOT [129, 130]. Both implementations are based on the widely-used scikit-learn [131]. Another example of AutoML is MorphNet [132]. MorphNet is focused on shrinking and expanding ANN structures to adapt the ANN for maximum performance with respect to constraints on computing resources. [7, Sec. 7.6.3]

2.7.3 Semi-Supervised Models

In Section 2.7, we have assumed that every datapoint x_i in our training dataset comes with an associated ground truth label y_i. In practice, this may not always be the case. For example, hyperspectral satellite images as input data cover large areas while soil moisture ground truth might be limited to several point-wise measurements. It is possible to use only the datapoints x_i with existing label y_i to apply supervised learning (see Section 2.7). Semi-Supervised Learning (SSL) is a solution that also benefits from datapoints without labels. The mathematical description can be found in Section 2.2 as well as in [24]. For SSL, a certain *smoothness* of the data is assumed. That means that if two datapoints $x_1, x_2 \in \mathcal{X}$ are close in the $\mathcal{X}$ space, their corresponding labels $y_1, y_2 \in \mathcal{Y}$ are close in the $\mathcal{Y}$ space as well. [...] [7, Sec. 7.5.2]

Up to now (2020), there is a lack of relevant SSL applications with respect to hyperspectral regression. In the following, we give an overview of the most important types of SSL approaches and their applications with hyperspectral input data. A review of general SSL applications is given in [133]. [7, Sec. 7.5.2.1]

Generative models rely on the *cluster assumption*: the datapoints of clusters and datapoints in similar clusters share similar labels [24]. Reasonable assumptions about the

dataset distributions are crucial for the success of generative models. As a result, only few ML applications with generative models exist. In [134], the authors present an example of the application of generative models. The authors present a classification of agricultural classes with multispectral data of an airborne sensor. [7, Sec. 7.5.2.1]

According to an equivalent formulation of cluster assumption, the decision boundary of an estimator should lie in a low-density region of the feature space [24]. To achieve this aim, maximum margin algorithms such as Support Vector Machines (SVMs) can be applied. This type of SSL algorithms is called low-density separation algorithms. One possible implementation are Transductive Support Vector Machine (TSVM). TSVM maximize the margins between unlabeled as well as labeled datapoints [135, 136]. An example of a TSVM application on Landsat 5 is shown in [137]. Six land cover classes are estimated based on a missing-label dataset as ill-posed classification task. [7, Sec. 7.5.2.1]

In graph-based methods, each datapoint of the dataset is represented as one node of a graph [138, 139]. The nodes are linked to each other with edges. The edges between two nodes are labeled with the distance between the respective two nodes. Graph-based methods rely on the *manifold assumption*. This means that the high-dimensional datapoints lie roughly on a low(er)-dimensional manifold [24]. Therefore, manifold regularization can be applied to the graph both on the labeled and on the unlabeled subset of the dataset [140]. This includes a term to enforce the smoothness of the dataset. As a remark, these methods imply high computational costs due to matrix inversion on large datasets despite efficient methods. In [141], the graph-based method proposed by [142] is applied to the AVIRIS *Indian Pines* land cover dataset. Another graph-based method is LapSVM. It is based on Laplacian SVMs and was introduced in [143]. LapSVM is applied by [144] on urban monitoring and cloud screening with a variety of data. [7, Sec. 7.5.2.1]

Additionally, we point to one further SSL approach which is applied on hyperspectral data. In [145], a Semi-Supervised Neural Network (SSNN) is introduced. The SSNN is set up to compensate shortcomings of the existing SSL approaches. The results show non-monotonic improvement of class accuracies as well as indicate the added value of ANNs in semi-supervised tasks. In [146], CNNs and RNNs are applied in combination with label clustering to deal with the missing labels. The Supervised Self-Organizing Maps (SuSi) framework [see Chapter 3] also contains semi-supervised estimators for regression and classification [6]. [7, Sec. 7.5.2.1]

2.7.4 Model Selection

In Section 2.7, we have introduced a number of supervised and semi-supervised models and provided references to exemplary applications [...]. However, we have not yet discussed the selection of a particular ML model for a given regression problem. This selection is based on criteria which are constrained by the dataset and the respective application. In the following, we list some important selection criteria of an ML regression model: [7, Sec. 7.5.3]

- What type of input data are we using: 1D, 2D, 3-dimensional (3D), time series? CNNs are good for datasets if locality and translation invariance can be assumed. LSTMs are particular useful at capturing long-term dependencies in time series data.
- What are the spectral, spatial and temporal dimension of the input data? Red–Green–Blue (RGB), multispectral or hyperspectral? Some ML models perform better with low-dimensional data. Therefore, dimensionality reduction (see Section 2.6.1) might be a good idea.
- Is the given regression problem linear or non-linear? We recommend applying the simplest model first. If the regression problem is expected to be linear, a linear model should be used.
- What is the size of the training dataset? Deep learning techniques require large amounts of data to be trained from scratch. Transfer learning can be applied to solve the shortcoming in this particular case (see Section 2.7.2).
- Is low bias or low variance more important for the estimation? Setting up ML models is often a tradeoff between bias and variance (see Section 2.2). For example in DT ensemble methods (see Section 2.7.1), bagging applies deeper trees with lower variance while boosting applies shallow trees with less bias.
- How important is it to apply ML models which are transparent and interpretable? Models such as k-Nearest Neighbors (k-NN) and DTs are easy to interpret for humans, while ANNs and SVMs are considered as *black box* models.

After choosing the model that meets the selection criteria, it needs to be optimized and then be evaluated. The hyperparameter optimization is described in Section 2.7.5. The metrics to evaluate ML models in terms of regression performance can be found in Section 2.7.6. [7, Sec. 7.5.3]

2.7.5 Hyperparameter Optimization

ML models are defined by two sets of parameters: hyperparameters and model parameters. Hyperparameters are set before the training and model parameters are learned during the training process of the ML model. In the following, we give a brief overview of different possibilities to optimize hyperparameters. A more detailed explanation of hyperparameter optimization can be found in [147]. [7, Sec. 7.5.3.1]

A simple way to tune hyperparameters is manually setting and testing them, for example, with cross-validation. Since this approach is very time-consuming, a better way is to automate the tuning process. For example, in the grid search approach, a pre-set hyperparameter space is automatically evaluated. The grid search operates well on small hyperparameter spaces, but it is very time-consuming for larger hyperparameter spaces. A randomized search speeds up the grid search approach by randomly iterating through the hyperparameter space instead of testing all combinations [148]. [7, Sec. 7.5.3.1]

A more sophisticated type of hyperparameter optimization is the Bayesian optimization. It collects more information about the dataset and the ML model with each iteration by building hypotheses about sets of hyperparameters before the actual run. [...] [7, Sec. 7.5.3.1]

2.7.6 Model Evaluation Metrics

ML models should be evaluated based on several different metrics. In the following, the evaluation metrics for regression and classification models are described.

Metrics in Regression For all of the n input datapoints x_i with their true labels y_i, the ML model returns the estimation $\hat{y}_i$ based on x_i. The Mean Absolute Error (MAE) is defined as

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \,. \tag{2.2}$$

The MAE is one of the easy-to-use evaluation metrics since it sums up the absolute differences $y_i - \hat{y}_i$ of the true label value and the estimated label value. [7, Sec. 7.5.3.2]

Many ML applications require the model to have as little outliers as possible. This can be achieved by including the squared error instead of the absolute error. The Mean Squared Error (MSE) is defined as

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \tag{2.3}$$

Estimation errors below 1 are less important in the MSE implementation than errors above 1. One drawback of the MSE is the unit of the error being the squared unit of the target variable to be estimated. The Root Mean Squared Error (RMSE) solves this issue. It is defined as [7, Sec. 7.5.3.2]

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \tag{2.4}$$

All the presented metrics, MAE, MSE, and RMSE, are easy to understand. MAE and RMSE return an error measure in the unit of the target variable y. As a remark, without knowing the distribution and scale of the target variable with its minimum and maximum, these metrics are difficult to interpret. [7, Sec. 7.5.3.2]

The Coefficient of Determination (R^2) is a relative measure to resolve the unit issue as stated above. It is defined as

$$R^2 = 1 - \frac{n \cdot \text{MSE}}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \qquad \text{with } \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i. \tag{2.5}$$

In this definition, it normally returns a value between 0 and 1; $R^2 = 1$ indicates that the ML model estimation is in perfect agreement with the data. However, negative values might occur and indicate a bad estimation performance. [...] [7, Sec. 7.5.3.2]

Metrics in Classification In classification, the labels y_i of the input datapoints x_i are discrete. In contrast to the regression case, the true and the estimated labels can not be compared with a distance metric. Instead, estimated labels can be equal or unequal to their respective true labels. The Overall Accuracy (OA) is defined as

$$\text{OA} = \frac{\text{number of correctly classified datapoints}}{n}, \tag{2.6}$$

with the total number of datapoints n. The OA is independent of the numbers of datapoints per individual class. To include the class distribution of a dataset into the evaluation of a ML model, the Average Accuracy (AA) can be used. The AA is defined as

$$\mathrm{AA} = \frac{1}{\mathrm{n}} \sum_{\mathrm{c}} \frac{\text{number of correctly classified datapoints of class c}}{\text{number of datapoints of class c}}, \tag{2.7}$$

with the sum over all different classes of a given dataset. To include the hypothetical probability of chance agreement θ into an evaluation, the Cohen's Kappa Coefficient (κ) can be applied. The probabilities θ for each class in the dataset are derived from the observed data. With that, κ is defined as

$$\kappa = \frac{\mathrm{OA} - \theta}{1 - \theta}. \tag{2.8}$$

2.8 Summary, Applications, and Extension

This chapter presents a hyperspectral estimation framework with four different levels: sensor level, data level, feature level, and model level. In Figure 2.1, this framework is illustrated in all considered parts. On the sensor level (Section 2.4), the hyperspectral data is combined with ground truth, for example, point-wise measurements of soil moisture or soil texture. The following data level (Section 2.5) is divided into pre-processing, the challenge of dataset shift, data augmentation, and approaches for dataset splitting. On the feature level (Section 2.6), dimensionality reduction, clustering, and feature engineering, new features are generated and afterward selected. The actual estimation is implemented on the model level (Section 2.7) with supervised and semi-supervised learning models. The model selection, optimization, and evaluation are described for the given estimation task and dataset, resulting in a final model and a final estimation.

This chapter includes material from [7], which provides further material:

- additional supervised learning models such as linear regression, partial least squares, SVMs, and k-NN,
- best practices as well as strengths and weaknesses of different models and approaches of the presented framework,
- a comparison of the supervised learning models based on the KarLy dataset in the regression of soil moisture based on hyperspectral data, and
- code examples for every part of the framework in Python [149].

The presented hyperspectral estimation framework is applied in this work as follows. In Chapter 3, a supervised and semi-supervised learning approach is presented that covers the feature and model level of this framework. A deep supervised classification approach is presented in Chapter 4 on the model level that replaces the feature level. Based on the ALPACA dataset, the hyperspectral estimation framework is applied to all levels in Chapter 5. The hyperspectral estimation framework is applied in other studies as well. Within the scope of this doctoral thesis, the author supervised two students during their master theses [50, 51]. In the following, their work and their findings are summarized as applications of the hyperspectral estimation framework.

Application on land use change detection The first master thesis is about the multi-temporal change detection of land use in Peru [50]. On the sensor level, multi-spectral satellite data from Sentinel-2 is used as input data. A land-use map is the ground truth for this estimation. On the data level, two different dataset splitting approaches are studied: random split and patch splitting (see Section 2.5.4). The main contributions of this master thesis are on the model level: the development and optimization of LSTMs, based on [150, 151], for the classification of land use. The results are promising since the sequential, multi-temporal LSTM estimator outperforms the mono-temporal baseline estimator. Further, the LSTM is able to learn the concept of clouds in the images without supervision.

Application on soil texture classification The second master thesis supervised within the scope of this thesis is about the classification of soil texture in eastern Germany [51]. Again, Sentinel-2 data is used as input data, and a freely available soil texture map is used as ground truth. On the sensor level, the impact of the ground truth quality on the estimation quality is studied. On the data level, the influence of Sentinel-2 atmospheric corrections is compared. Also, two different dataset splitting approaches are implemented: random splitting and patch splitting [152]. On the feature level, spectral indices are applied to study the separability of Sentinel-2 pixels between bare soil and vegetation. On the model level, several supervised models are applied, such as SVMs, RF, ANNs, and a voting classifier. The main results can be summarized as follows: (a) the impact of pixels covered by vegetation is more significant than the impact of the ground truth quality, (b) the atmospheric corrections show no increase in estimation performance in this study, and (c) the voting classifier shows the best overall estimation performance.

Extension of the Framework As an extension to the four levels of the hyperspectral estimation framework, we propose a fifth level: the *decision level*. With the estimations provided by the ML model on the model level, decisions can be triggered based on the applications. For example, agronomic decisions can be made based on

hyperspectral estimations of crop type, crop cover, and crop yield [153]. Another example is that a land cover estimation map provided on the model level can trigger administrative decisions. Further, *decision fusion* of different data sources and different estimations is included in the decision level [154, 155].

3 Semi-Supervised Self-Organizing Maps for Regression and Classification

> *Some people call this artificial intelligence, but the reality is this technology will enhance us. So instead of artificial intelligence, I think we'll augment our intelligence.*
>
> — **Virgina Rometty**
> (CEO of IBM, 2012-2020)

This chapter includes material from

Felix M. Riese, Sina Keller, and Stefan Hinz. "Supervised and Semi-Supervised Self-Organizing Maps for Regression and Classification Focusing on Hyperspectral Data". In: *Remote Sensing* 12.1 (2020). It is cited as [6] and marked in green.

3.1 Introduction

Machine Learning (ML) approaches benefit from large datasets, especially the commonly applied deep Artificial Neural Networks (ANNs) and Convolutional Neural Networks (CNNs), as described in Section 2.7.2. In hyperspectral remote sensing, the acquisition of reference data (ground truth) can be time-consuming and costly. In the example of the acquisition of soil moisture ground truth, each soil moisture measurement is taken manually, point by point. That leads to hyperspectral datasets with sufficient hyperspectral data, for example, a hyperspectral Unmanned Aerial Vehicle (UAV) image, and only a few datapoints which are labeled with their respective soil moisture measurement. Therefore, there is a need for ML approaches that can handle small datasets and datasets with only a few labeled datapoints.

Self-Organizing Maps (SOMs) are a type of ANN that can handle datasets with few references. SOMs are mostly used for unsupervised clustering and visualization; only a few studies have focused on supervised SOMs up to now (see Section 3.2). The SOM was introduced in [156, 90, 157, 158]. It is a shallow ANN architecture consisting of an input layer and a 2-dimensional (2D) grid as the output layer. The latter is fully connected to the input layer. The neurons on the output grid are interconnected to each other through a neighborhood relationship. Changes on the weights of one output neuron also affect the neurons in its neighborhood. This unique characteristic decreases overfitting of the training datasets. Furthermore, the 2D output grid visualizes the results of the SOM comprehensibly. This plain visualization does not exist in the majority of ANNs. [6, Sec. 1]

In this [... chapter], we introduce the Supervised Self-Organizing Maps (SuSi) framework for regression and classification with hyperspectral data. Figure 3.1 illustrates the schema of the SuSi framework. The SuSi framework combines unsupervised, supervised, and semi-supervised learning for regression and classification. In [41], the supervised regression part of the SuSi framework has been initially introduced. It has been applied for the estimation of soil moisture [23] and the estimation of water quality parameters [9]. This initial SOM implementation has served as the basis for the SuSi framework that we have further improved in terms of its regression performance. In addition, we introduce novel parts of the SuSi framework in this contribution such as the supervised classification as well as the Semi-Supervised Learning (SSL). To evaluate these novel parts, we apply the SuSi framework on two exemplary hyperspectral datasets and compare the respective regression and classification results to a Random Forest (RF) model. The actual framework is available as an open-source Python package on GitHub [159]. This ensures the maintainability and the opportunity for future upgrades by the authors as well as any member of the community. [6, Sec. 1]

The main contributions regarding methodological developments and software of this chapter are:

- the publication of the Karlsruhe Lysimeter (KarLy) dataset,
- the development and open-source publication of the SuSi framework including unsupervised, supervised, and semi-supervised SOMs, and
- the applications of this framework approach in the regression of soil moisture and the classification of land cover based on hyperspectral data.

This chapter is structured as follows. The related work of SOMs is presented in Section 3.2, including a comparison of existing SOM frameworks with the SuSi framework. The basis of the SuSi framework, the unsupervised SOM, is described in

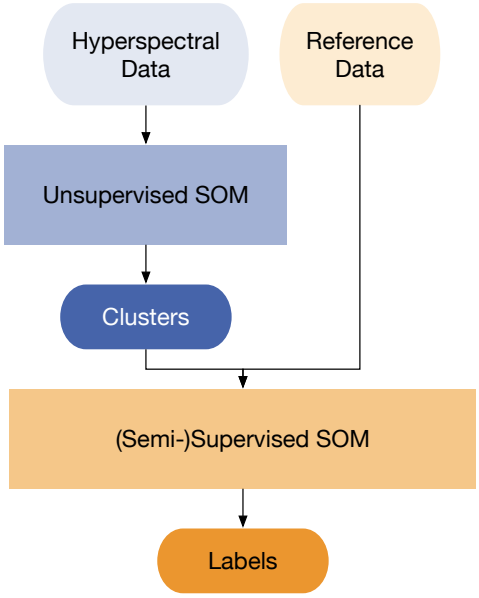


Figure 3.1: Schema of the Supervised Self-Organizing Maps (SuSi) framework consisting of the unsupervised Self-Organizing Map (SOM) (blue) and the (semi-)supervised SOM (orange). Depending on the degree of reference data availability, a supervised or semi-supervised SOM is applied. (adapted from [6])

detail in Section 3.3. For supervised regression and classification, the unsupervised SOM is upgraded with an additional output layer to a supervised SOM in Section 3.4. The semi-supervised extension is described in Section 3.5. In Section 3.6, the supervised and the semi-supervised regression of soil moisture is performed on a hyperspectral dataset. The classification SOM is applied in Section 3.7 in the supervised and semi-supervised classification of land cover based on hyperspectral data. Finally, the chapter is concluded in Section 3.8.

3.2 Related Work

In the following subsection, we give a brief overview of various SOM applications in different fields of research. Most SOMs are applied in unsupervised learning like clustering, visualization, and dimensionality reduction [160]. A comprehensive overview of SOMs as unsupervised learners and their applications in the research field of water resources is presented by [20]. SOMs are also used in the context of maritime environment research [161]. In addition, a main application of SOMs is clustering data [162] and cluster-wise regression [163]. [6, Sec. 2.1]

SOMs can be combined with other ML techniques. For example, the output of the unsupervised SOM is used by Hsu et al. [164] as input for a support vector

regressor to forecast stock prices. The clustering of the unsupervised SOM significantly improved the forecast. The authors in [165] present the combination of SOMs with linear regression in hydrology. The authors point out the value of the visualization capabilities of the unsupervised SOM. SOMs can also be used for data fusion, e.g., for plant disease detection [166]. Hagenbuchner and Tsoi [167] add majority voting to SOMs for the application as a supervised classifier. The combination of SOMs and nearest-neighbor classification is shown by Ji [168] for the classification of land use based on Landsat satellite data. Additional combinations of unsupervised SOMs and supervised algorithms used for classification are presented in [169, 170, 171, 172]. One example for the application of SOMs to solve nonlinear regression tasks is presented by Hecht et al. [173] in the field of robotics. SOMs are also applied in the supervised regression of physical parameters in environmental research [41, 23, 9]. [6, Sec. 2.1]

SSL learns from labeled as well as from unlabeled data. An introduction to SSL is given in [24] [and in Section 2.7.3]. In [174], an unsupervised SOM and a supervised ANN are combined for the semi-supervised classification in software fault detection. The authors point out the generalization abilities of their proposed framework. An approach entirely based on SOMs is presented in [175]. A Semi-Supervised Self-Organizing Map (SS-SOM) is introduced and applied on several real-world datasets. The implementation of the SS-SOM is provided and can therefore be compared to existing packages. [6, Sec. 2.1]

Python and R are widely used programming language in the context of ML applications. Programming frameworks like scikit-learn [131] in Python have simplified the application of existing ML techniques considerably. While in the programming language R the *kohonen* package [176] provides a standardized framework for SOMs, several minor SOM packages exist in Python addressing only specific ML tasks. [6, Sec. 2.1]

In the following, we compare the introduced SuSi framework with existing Software packages in Python, R, and C. In our comparison, we include the Python packages *SOMPY* [177], *SimpSOM* [178], *MiniSom* [179], *TensorFlow SOM* [180], PyMVPA [181], NeuPy [182], the R *kohonen* package [176] and the C tool *SS-SOM* [175]. All entitled packages are freely available, regularly maintained (in 2019), and include unsupervised clustering. For example, the packages `abhinavralhan/kohonen-maps` and `lmjohns3/kohonen` are not regularly maintained and therefore left out in this study. The packages are analyzed on the basis of the following criteria: [6, Sec. 2.1]

- Syntax: Is the syntax simple to use, e.g., in the style of scikit-learn [131]?

- Documentation: Is the documentation provided in the form of a comprehensive paper or an online documentation?
- Code: Is the code well documented and structured? Are code examples and example plots available?
- ML: Is unsupervised clustering, supervised regression, or supervised classification included in the package?
- Installation: Is the installation simple for the user, e.g., with PyPI?
- Programming language: In what language can the package be used?

The analysis is performed in favor of the packages, meaning that in cases where a criterion is only partially satisfied, it is still considered as satisfied. Table 3.1 summarizes the analysis results. Thus far, no supervised SOM package for Python is available that matches the defined criteria and requirements (see Table 3.1). [6, Sec. 2.1]

3.3 Unsupervised SOMs for Clustering

In the following subsection, we describe the architecture and mathematics of the unsupervised part of the SuSi framework. Further insights about the theory of SOMs can be found for example in [158]. The 2D grid of a SOM, the map, can be implemented in different topologies. In this [... chapter], we use the simplest topology: the 2D rectangular grid consisting of $n_{row} \times n_{column}$ nodes. This grid is fully connected to the input layer. This means that each node is connected to all [... m] input features via [... m] weights. The variable naming conventions of the SuSi framework are given in Table B.1. The training process of the unsupervised SOM is illustrated in Figure 3.2 and consists of the following steps: [6, Sec. 2.3]

1. Initialize the SOM.
2. Get random input datapoint.
3. Find Best Matching Unit (BMU) (see Section 3.3.1).
4. Calculate learning rate (see Section 3.3.2) and neighborhood function (see Section 3.3.3).
5. Calculate neighborhood distance weight matrix (see Section 3.3.4).
6. Modify SOM weight matrix (see Section 3.3.5).
7. Repeat from step 2 until the maximum number of iterations is reached.

The initialization approach of a SOM mainly affects the speed of the training. We initialize the SOM weights of the SuSi framework randomly at this stage of

Table 3.1: Analysis of the SuSi package with existing SOM packages. All packages are provided for the programming language Python (except kohonen and SS-SOM), they are freely available and regularly maintained (in 2019). (reprinted from [6])

Package Reference	SuSi	SOMPY [177]	SimpSOM [178]	MiniSom [179]	TensorFlow SOM [180]	PyMVPA [181]	NeuPy [182]	kohonen [176]	SS-SOM [175]
Simple syntax	✓		✓	✓		✓	✓		
Useful documentation	✓			✓		✓	✓		✓
Well documented code	✓		✓	✓	✓		✓		
Unsupervised clustering	✓	✓	✓	✓	✓	✓	✓	✓	✓
Supervised regression	✓							✓	
Supervised classification	✓							✓	✓
Semi-supervised regression	✓								
Semi-supervised classification	✓								✓
Simple installation	✓		✓	✓		✓	✓	✓	
Python version	3	2	3	3	3	2	3	R	C

development. Attik et al. [183] and Akinduko et al. [184], for example, proposed more sophisticated initialization approaches like applying a principal component analysis. We describe the training of an unsupervised SOM in Section 3.3.1. For every part of the unsupervised SOM, several possible formulas exist.

The most relevant formulas for the unsupervised SOM for this chapter are provided in the following subsections. Alternative formulas are partially provided in Appendix C.1.1 and in detail in [6].

3.3.1 Finding the Best Matching Unit

During the search for the BMU, the current input datapoint is compared to all [... m]-dimensional weight vectors on the SOM grid. The SOM node that is the closest one to the input node according to the chosen distance metric is the BMU. Several distance metrics can be applied. The most common distance metric is the *Euclidean distance* defined as

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{\sum_{i=1}^{m} (x_i - x_i')^2}, \tag{3.1}$$

with a dimension [... m] of the vectors $\mathbf{x}, \mathbf{x}'$. Further SOM distance metrics are provided in [...] [6]. The Euclidean distance is the default distance metric of the SuSi framework. Note that, in Section 3.3.4, the Euclidean distance is applied on the 2D coordinates of the SOM nodes and not on the [... m]-dimensional weight vectors. [6, Sec. 2.3.1]

3.3.2 Learning Rate

Decreasing learning rates are often implemented in ANNs for a faster convergence and to prevent oscillations. The learning rate α of the SOM training is a function that decreases from a value α_0 with an increasing number of iterations. In the following, we present the default implementation of the learning rate in the SuSi framework. Alternative learning rate functions are summarized in [...] [6]. The

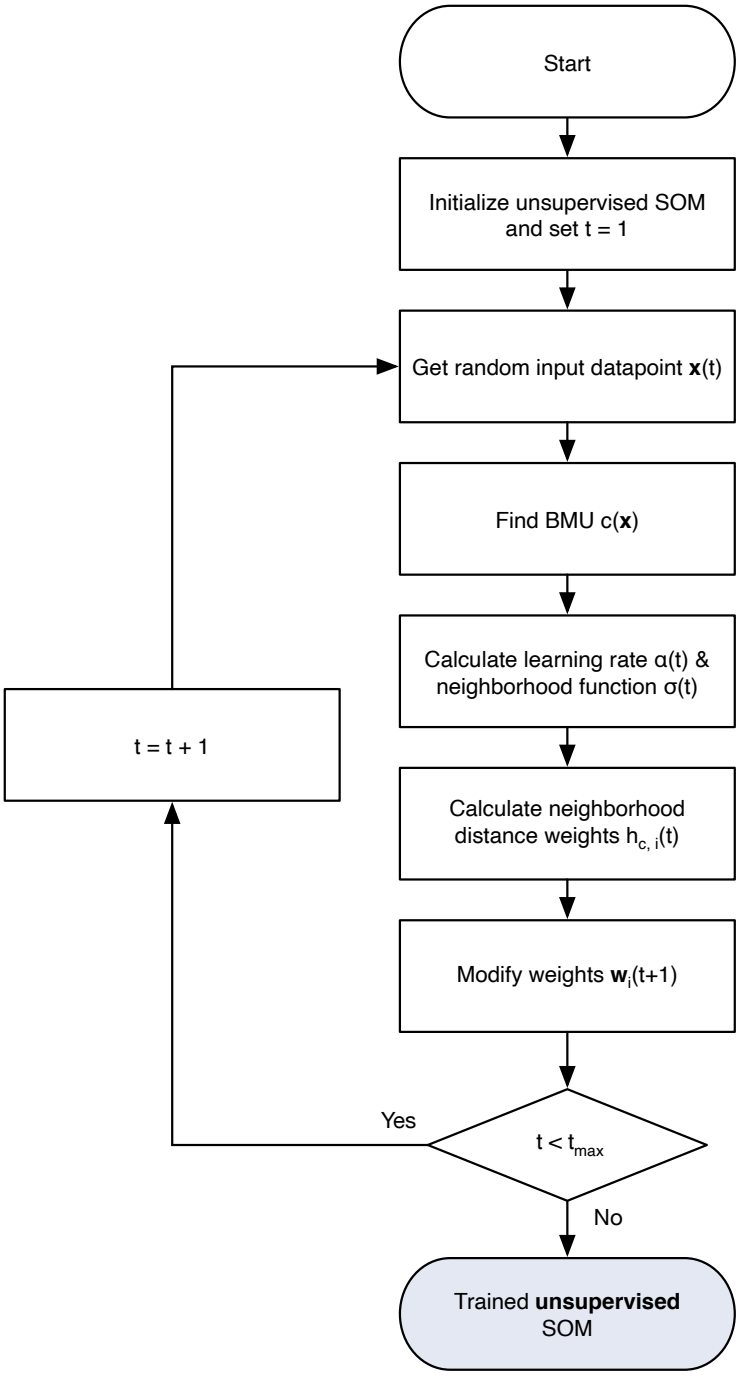


Figure 3.2: Flowchart of the unsupervised SOM algorithm resulting in the trained unsupervised SOM (blue). (adapted from [6])

default implementation the learning rate is taken from [185] and includes a start value for the learning rate as well as an end value α_{end}: [6, Sec. 2.3.2]

$$\alpha(t) = \alpha_0 \cdot \left(\frac{\alpha_{end}}{\alpha_0}\right)^{t/t_{max}}. \tag{3.2}$$

3.3.3 Neighborhood Function

Similar to the learning rate, the neighborhood function is monotonically decreasing. In the following, we provide the default neighborhood function of the SuSi framework. Additional neighborhood functions can be found in [...] [6]. The neighborhood function in [186] is defined, with σ_0 as initial value of the neighborhood function [...], as [6, Sec. 2.3.3]

$$\sigma(t) = \sigma_0 \cdot \left(1 - \frac{t}{t_{max}}\right). \tag{3.3}$$

3.3.4 Neighborhood Distance Weight

The neighborhood distance weight is a function of the number of iterations and the distance $d(c, i)$ between the BMU c and every other node i on the SOM grid. The distance $d(c, i)$ between the BMU c and node i is defined as the Euclidean distance (see Equation (3.1)) on the 2D map grid. Note that this distance $d(c, i)$ is not the distance between the [... m]-dimensional weights. In the following, we provide the default neighborhood distance weight formula of the SuSi framework. Another formula is provided in [...] [6]. Matsushita and Nishio [186] proposed a *Pseudo-Gaussian* neighborhood distance weight. The weight between the BMU c and the node i on the SOM grid is defined as

$$h_{c,i}(t) = \exp\left(-\frac{d^2}{2 \cdot \sigma(t)^2}\right), \tag{3.4}$$

with the neighborhood function from Equation (3.3) and the Euclidean distance $d(c, i)$ on the SOM grid. The implications of the chosen neighborhood distance weight definitions on the SOM are investigated in e.g., [187, 188]. [6, Sec. 2.3.4]

3.3.5 Adapting Weights

The two most used approaches to adapt the SOM weights are the *online* and the *batch* mode. All weights of the SOM are adapted based on the learning rate and the neighborhood distance weight. The *online mode* is described in detail in [158]. After each iteration, all weights of the SOM are adapted to the current datapoint $\mathbf{x}(t)$ as follows:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \alpha(t) \cdot h_{c,i}(t) \cdot \left(\mathbf{x}(t) - \mathbf{w}_i(t)\right), \tag{3.5}$$

with neighborhood function $h_{c,i}(t)$, learning rate $\alpha(t)$, and weight vector $\mathbf{w}_i(t)$ of node i at iteration t. The *online* mode is the default mode of the SuSi framework. An implementation of the *batch mode* is described in [... [6]]. [6, Sec. 2.3.5]

3.3.6 Trained Unsupervised SOM

After reaching the maximum number of iterations t_{max}, the unsupervised SOM is fully trained. The unsupervised SOM can now calculate the BMU for every input datapoint, regardless if the datapoint is also part of the training dataset. This supervised approaches in Sections 3.4 and 3.5 are based on the trained unsupervised SOM. Since the number of nodes on the SOM grid can be larger than the number of datapoints in a specific dataset, several nodes can exist that are not linked to a datapoint of that respective dataset. [6, Sec. 2.3.6]

3.4 Supervised SOMs for Regression and Classification

To apply the SuSi framework for solving supervised regression or classification tasks, we attach a second SOM to the unsupervised SOM. Figure 3.3 illustrates the flowchart of the second, supervised SOM. The two SOMs differ with respect to the dimension of the weights as well as their training. The weights of the unsupervised SOM are of the same dimension as the input data. Thus, adapting these weights often changes the BMU for each input datapoint. In contrast, the weights of the supervised SOM have the same dimension as the target variable of the respective task. [6, Sec. 2.4]

One has to distinguish between two cases: regression and classification. In the regression case, the weights of the supervised SOM are one-dimensional and contain a continuous number. In the classification case, the weights of the supervised SOM

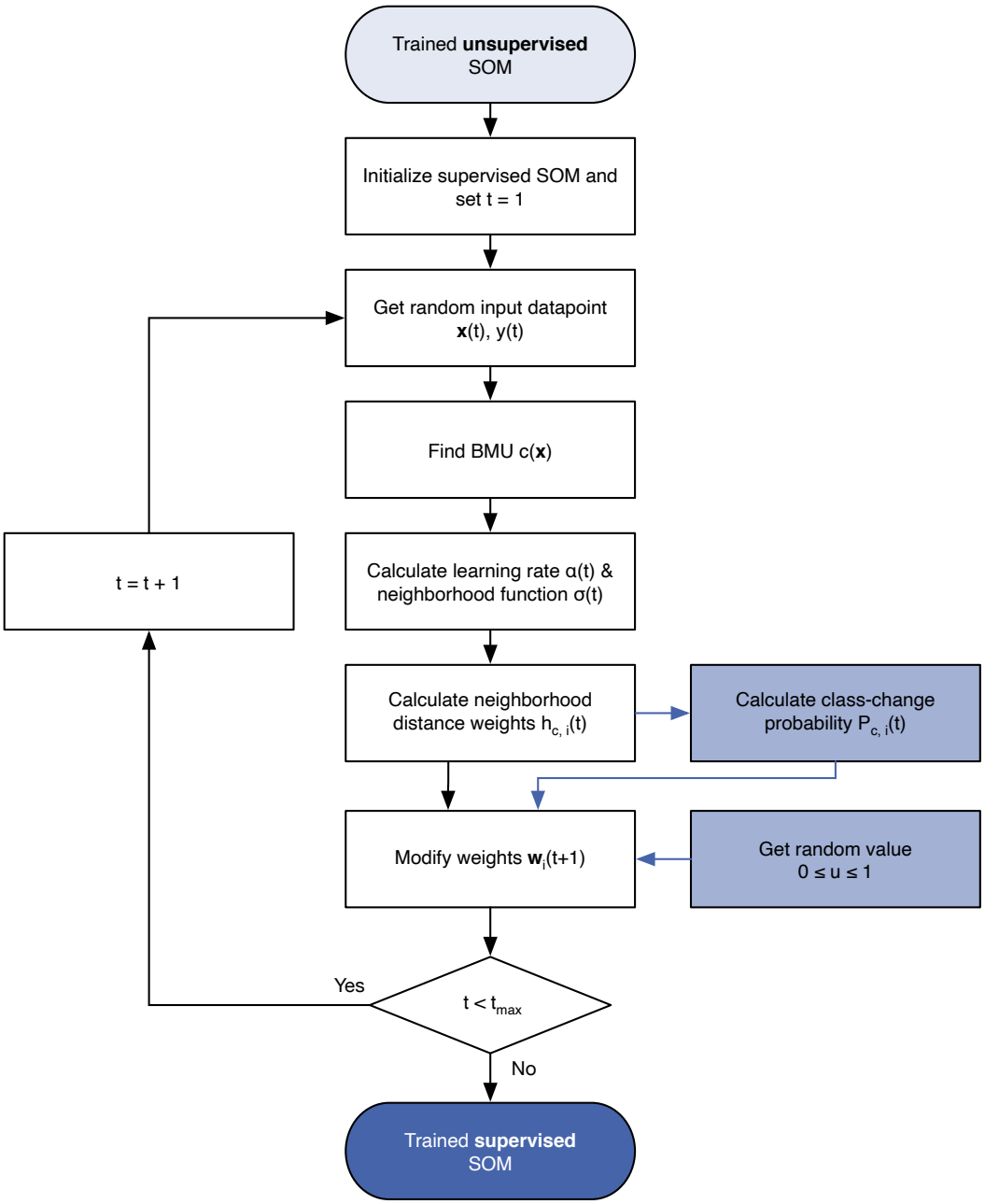


Figure 3.3: Flowchart of the algorithms for the regression SOM (black) and the classification SOM (black and blue). The *Trained unsupervised SOM* (dark blue) is the result of the unsupervised SOM algorithm illustrated in Figure 3.2. (adapted from [6])

contain a class. By combining the unsupervised and the supervised SOM, the former is used to select the BMU for each datapoint while the latter links the selected BMU to a specific estimation. In the following, we describe the different implementations for regression and classification tasks. [6, Sec. 2.4]

3.4.1 Implementation of the Regression SOM

An initial implementation of the regression SOM is described in [41] using the example of the soil moisture regression based on hyperspectral data. The training of the regression SOM proceeds analogous to the unsupervised SOM: first, the SOM is initialized randomly. Again, it iterates randomly through the dataset (see Step 1). In each iteration, the BMU is found for the current datapoint based on the trained unsupervised SOM (see Steps 2 and 3). The BMUs do not change for the datapoints during the training since the unsupervised SOM is fully trained. Then, the neighborhood function, the learning rate, and the neighborhood distance weight matrix are calculated similarly to the algorithm of the unsupervised SOM (see Steps 4 and 5). Finally, the weights are adapted to the label y(t) of the input datapoint $\mathbf{x}(t)$ (see Step 6). [6, Sec. 2.4.1]

In the case of the regression SOM, the label is a continuous value and the weights of the regression SOM can be modified similarly to the process described in Section 3.3.5. After the training (and in the case of a one-dimensional, target variable), the regression SOM consists of a map with a continuous distribution of the regression target variable. To apply the trained regression SOM to a new dataset, the BMUs needs to be found by the unsupervised SOM. For each datapoint in the new dataset, the estimated output value of the SuSi framework is the weight of the found BMU on the regression SOM. The regression SOM is illustrated in Figure 3.3. [6, Sec. 2.4.1]

3.4.2 Implementation of the Classification SOM

In the case of a classification task, the labels are discrete. In contrast to the commonly used majority voting approach (see [167]), we have implemented a training process similar to the adaptation approach of the unsupervised SOM (see Section 3.3.5): [6, Sec. 2.4.2]

1. Initialize the classification SOM.
2. Get random input datapoint with label.
3. Find BMU based on trained unsupervised SOM.

4. Calculate learning rate and neighborhood function.
5. Calculate neighborhood distance weight.
6. Calculate class-change probability matrix.
7. Modify classification SOM weight matrix.
8. Repeat from step 2 until the maximum number of iterations is reached.

The classification SOM is illustrated in Figure 3.3. The initialization in step 1 contains a simple majority vote: each node is assigned to the class representing the majority of datapoints allocated to the respective node. Steps 2 to 5 are implemented similarly to the regression SOM in Section 3.4.1. To modify the discrete weights of the classification SOM, we introduce the class-change probability $P_{c,i}(t)$ in step 6. In the regression SOM, the SOM nodes around the BMU are adapted to the current datapoint with a certain probability depending on the learning rate and the neighborhood distance weight. In the following, we explain our proposed modification in the case of discrete models. [6, Sec. 2.4.2]

For datasets with imbalanced class distributions, meaning datasets with significantly different number of datapoints per class, we provide the possibility to re-weight the dataset. The optional class weight is defined as

$$w_{class(j)} = \begin{cases} n/(n_{classes} \cdot n_j), & \text{if class weighting,} \\ 1, & \text{otherwise,} \end{cases} \tag{3.6}$$

with the number of datapoints [... n], the number of datapoints [... n_j] of class j, and the number of classes $n_{classes}$. Similar to Equation (3.5), we define a term that affects the modification of the SOM weights. Since the modifications need to be discrete, we rely on probabilities. The probability for a class change of a node i with BMU c($\mathbf{x}$) of the datapoint $\mathbf{x}(t)$ with label y(t) is defined as

$$P_{c,i}(t) = w_{y(t)} \cdot \alpha(t) \cdot h_{c,i}(t), \tag{3.7}$$

with the class weight $w_{y(t)}$ (see Equation (3.6)), the learning rate $\alpha(t)$ (see Section 3.3.2), and the neighborhood distance weight $h_{c,i}(t)$ (see Section 3.3.4). To decide if a node changes its assigned class, a binary decision rule is created based on this probability. A simple threshold of e.g., 0.5 would lead to a static SOM after a certain number of iterations. Therefore, we include randomization into the decision process. For every node in every iteration, a random number $u_i(t)$ is generated that is uniformly distributed between 0 and 1. The mod-

ification of the weights is then implemented based on the class change probability $P_{c,i}(t)$ defined in Equation (3.7) as follows:

$$w_i(t+1) = \begin{cases} y(t), & \text{if } u_i(t) < P_{c,i}(t), \\ w_i(t), & \text{otherwise,} \end{cases} \tag{3.8}$$

with the label $y(t)$ linked the to datapoint $\mathbf{x}(t)$ of the current iteration t. After the maximum number of iterations is reached, the classification SOM is fully trained. Then, every node on the SOM grid is assigned to one class of the dataset. To apply the classification SOM on unknown data, the BMU needs to be found for each datapoint with the unsupervised SOM. This process is similar to the process of the trained regression SOM. The estimation of the classification SOM for this datapoint is equivalent to the weight of the neuron in the classification SOM at the position of the selected BMU. [6, Sec. 2.4.2]

3.5 Semi-Supervised SOMs for Regression and Classification

In hyperspectral remote sensing, the acquisition of the reference data like soil moisture measurements is often costly and time-expensive. With hyperspectral sensors on UAVs or satellites, the hyperspectral input data can be recorded over large areas. This kind of data acquisition can lead to a dataset with significantly more hyperspectral input data than reference data. To use the possibly larger input dataset, SSL can be applied that can learn from labeled as well as from unlabeled data. In comparison, supervised learning approaches (see Section 3.4) can only learn from labeled data. [6, Sec. 2.5]

The semi-supervised SOM is a combination of an unsupervised part (see Section 3.3) and a supervised part (see Section 3.4). The unsupervised part is applied on the full hyperspectral input dataset since it does not rely on labels. The supervised part afterwards is applied on the smaller, labeled part of the training dataset. To decrease the importance of unlabeled training datapoints, a sample weight is implemented which gives more weight to labeled datapoints than to unlabeled datapoints. This weighting is used for semi-supervised regression as well as for semi-supervised classification. A class weighting, as it is implemented in the supervised classification SOM (see Section 3.4.2), can only improve the semi-supervised classification performance, if the labeled datapoints are not balanced over the different classes. [6, Sec. 2.5]

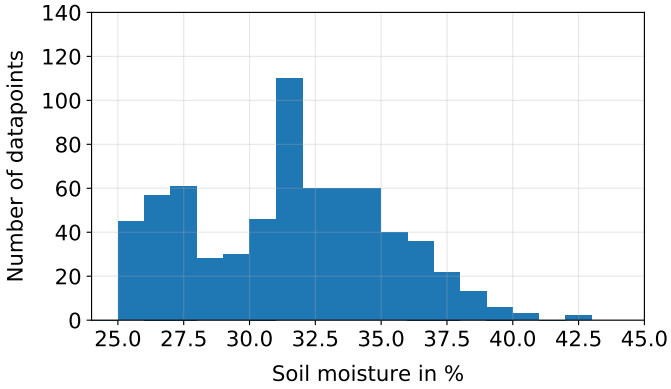


Figure 3.4: Soil moisture histogram of the KarLy dataset [41, 42]. (adapted from [6])

3.6 Soil Moisture Regression

In this section, the SuSi framework is applied on a hyperspectral dataset in the regression of soil moisture. The relevance of soil moisture is addressed in Section 2.3.1 and the dataset is introduced in Section 3.6.1. The supervised and semi-supervised regressions are presented in Sections 3.6.2 and 3.6.3. In both regressions, the SuSi framework is compared to a RF regressor.

3.6.1 Soil Moisture Dataset

The Karlsruhe Lysimeter (KarLy) dataset consisting of a bare soil sample is introduced in [41] and is openly available in [42]. The dataset was acquired in May 2017 in a five-day field campaign. A Time Domain Reflectometry (TDR) sensor measures soil moisture in an irrigated soil sample in a range of 25 % to 42 % soil moisture (see Figure 3.4).

The hyperspectral data is acquired with the UHD 285 hyperspectral snapshot camera by Cubert (Ulm, Germany). In Section 2.4.1, the UHD 285 is compared to the hyperspectral sensors applied within the scope of this thesis. The UHD 285 covers the Visible and Near-Infrared (VNIR) spectrum with a range of 450 nm to 950 nm in 125 spectral bands. Its bandwidth is 8 nm, with a sampling interval of 4 nm. One image includes 50×50 pixels. In this work, the sensor is mounted on a tripod in a static setup. The UHD 285 is calibrated with a white reference, which is located within the field of view in every measurement. For every image, a mean spectrum is calculated, as described in [41]. Overall, the dataset consists of 679 datapoints with a hyperspectral spectrum of 125 bands and one soil moisture label.

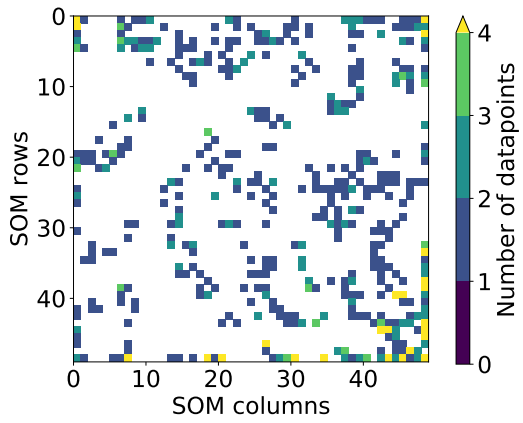
3.6.2 Supervised Regression of Soil Moisture

For the supervised regression of soil moisture, the KarLy dataset (see Section 3.6.1) is used with all datapoints including all labels. The dataset is split randomly into a training and a test subset in the ratio 1 : 1 [6]. The regression models are trained on the training subset and evaluated on the test subset. A normalization per feature is applied for the SOM regressor, the data for the RF regressor is not normalized [6]. A similar study with an initial version of the regression SOM is published in [41].

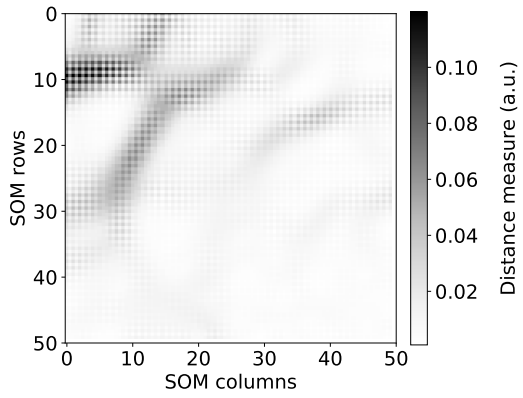
The hyperparameters of the regression SOM are set after a minor optimization and are provided in Table C.1. These hyperparameters can be further optimized depending on the applied dataset and underlying task. The results of the regression SOM are compared to the results of a RF regressor [95]. The RF regressor is set up with 10 000 estimators and the scikit-learn default hyperparameters (see [131]). For the evaluation, we choose the Coefficient of Determination (R^2). To ensure generalization, the evaluation is repeated for five different random seeds which are applied on the SOM, the RF as well as the dataset split. The final results are provided as the mean R^2 its standard deviation. [6, Sec. 3.2]

The regression SOM achieves $R^2 = (95.3 \pm 0.8)\,\%$ on the test subset. This score implies that the regression SOM is able to generalize very well on this dataset. Interestingly, the result for the training subset is only marginally better with $R^2_{\text{train}} = (97.7 \pm 0.6)\,\%$. In comparison, the RF regressor results in $R^2 = (93.0 \pm 2.2)\,\%$ on the test dataset and $R^2_{\text{train}} = (99.1 \pm 0.2)\,\%$ on the training dataset. The SOM outperforms the RF while showing less overtraining, meaning a smaller difference between R^2 and R^2_{train}. In this case, the R^2_{train} score could function as *out-of-bag estimate* for the dataset similar to [95]. When dealing with small datasets, the SOM provides the advantage of not requiring a split of the dataset. [6, Sec. 3.2]

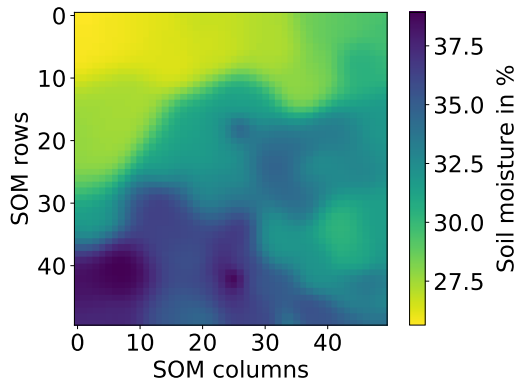
Figure 3.5a shows the distribution of the SOM BMUs of the soil moisture dataset. Instead of finding one single maximum, we recognize rather a random and uniform distribution. The u-matrix shows the Euclidean distances between the SOM nodes (see Figure 3.5b). In summary, areas on the SOM grid with low output values show larger distances to the neighboring SOM nodes. Figure 3.5a illustrates further that, despite the fact that the dataset is smaller than the number of SOM nodes, the training takes advantage of the whole SOM grid. The spread over the whole SOM grid ensures a generalization. The continuous regression output for each SOM node is presented in Figure 3.5c. Although the SOM nodes outnumber the input datapoints, each SOM node is linked to a soil moisture value. [6, Sec. 3.2]



(a)



(b)



(c)

Figure 3.5: (a) Regression SOM distribution of the BMUs of the dataset; (b) u-matrix; (c) distribution of the SOM regression output calculated for each node. (reprinted from [6])

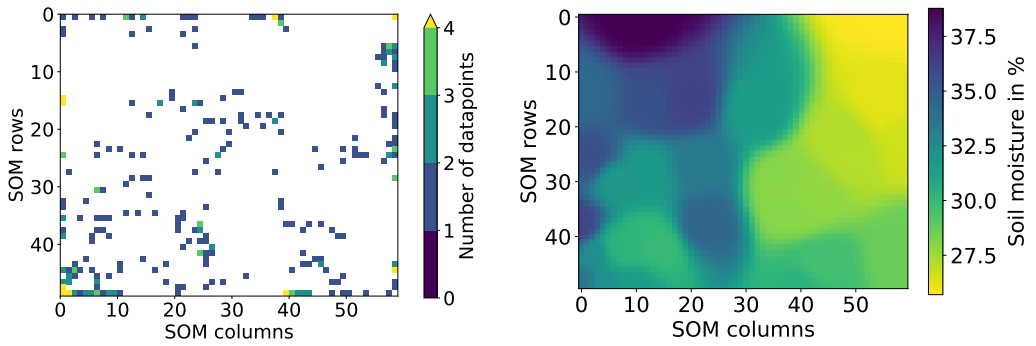


Figure 3.6: Semi-supervised regression SOM distributions of (a) the Best Matching Units (BMUs) of the test dataset and (b) the regression output calculated for each node. (reprinted from [6])

3.6.3 Semi-Supervised Regression of Soil Moisture

To evaluate the semi-supervised regression SOM, we use the soil moisture dataset and is applied in a supervised regression (see Section 3.6.2). For the training, the dataset is split into a training and a test subset in the ratio 1 : 1. Only 10 % of the datapoints in the training dataset are labeled. Five different random seeds for the SuSi framework as well as for the dataset split are used to reduce effects of randomization. The hyperparameters of the semi-supervised regression SOM are set after a minor optimization and are provided in Table C.1. For the semi-supervised regression SOM, the hyperspectral input data are normalized. [6, Sec. 3.4]

The semi-supervised regression SOM achieves $R^2 = (81.9 \pm 3.2)\,\%$. The distribution of the BMUs of the soil moisture dataset is shown in Figure 3.6a (supervised regression: Figure 3.5a) and the continuous regression output for each SOM node can be found in Figure 3.6b (supervised regression: Figure 3.5c). Comparing the plots of the semi-supervised and the supervised case, we find that the distribution of the dataset over the full SOM grid as well as generating a continuous output map remain the same for the semi-supervised case. [6, Sec. 3.4]

To evaluate the regression performance of the semi-supervised SOM, we apply an RF regressor with five different random seeds and 1000 estimators. The RF is trained only on the labeled datapoints of the training dataset and tested on the full test dataset. It achieves $R^2 = (71.9 \pm 6.0)\,\%$. This result shows that the semi-supervised regression SOM is able to learn additional information from the unlabeled data and outperforms the RF on such a small labeled dataset. [6, Sec. 3.4]

3.7 Land Cover Classification

In the following, the land cover classification with SOMs is presented. In Section 2.3.3, the relevance of land cover classification is addressed. The dataset is described in Section 3.7.1, and unsupervised clustering and visualization are performed in Section 3.7.2. The supervised classification is presented in Section 3.7.3 and the semi-supervised classification is described in Section 3.7.4.

3.7.1 Land Cover Dataset

The Salinas valley dataset [189] for classification is a freely available land cover dataset consisting of 512×217 pixels collected by the 224-band Airborne Visible Infrared Imaging Spectrometer (AVIRIS) in California, United States of America. The spatial resolution of this dataset is 3.7 m. Of the 224 bands in the range of 400 nm to 2500 nm, the 20 water absorption bands are discarded, namely the bands 108 to 112, 154 to 167 and 224. The dataset consists of 54 129 datapoints with reference data of 16 classes including several vegetation classes and bare soil. In Figure 3.7, an overview of the dataset is illustrated. Compared to the regression dataset, this dataset is significantly larger. [6, Sec. 2.2]

For some studies, the hyperspectral data are normalized per feature to improve the performance of some estimators which are based on distance metrics. The mean of the respective dataset is therefore set to zero and the standard deviation to one. [6, Sec. 2.2]

3.7.2 Unsupervised Clustering and Visualization

We exemplary apply the unsupervised SOM on the Salinas dataset described in Section 3.7.1. The hyperparameters for the unsupervised SOM are provided in Table C.1. After the training of the unsupervised SOM, the dominant (most prevalent) class for every single SOM node is shown in Figure 3.8a. The SOM performs the clustering unsupervised, solely relying on the input data. The labels of the input data are only added in the unsupervised part due to visualization purposes. Similar classes form clusters of different shapes. With this visualization, information about similarities between the classes can be gained as well as possible mislabeled datapoints and classes [160]. Classes such as *celery* and *soil vinyard*

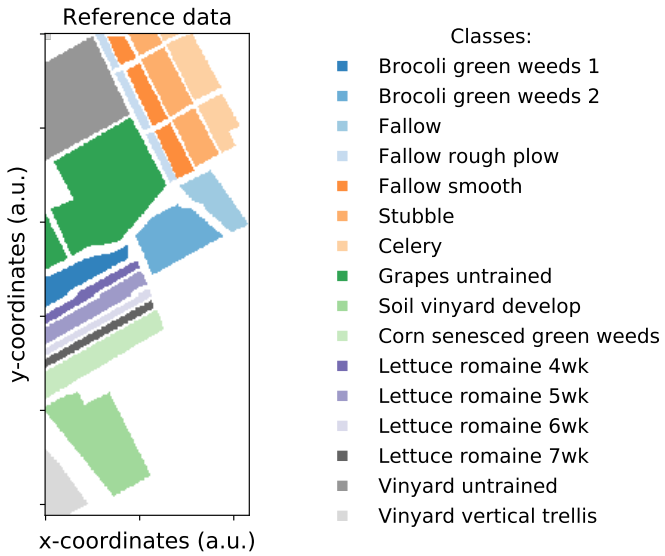


Figure 3.7: Overview of the Salinas Airborne Visible Infrared Imaging Spectrometer (AVIRIS) dataset with 16 classes and a spatial resolution of 3.7 m [189]. The coordinates are not available and therefore illustrated in Arbitrary Units (a.u.).

develop form more isolated clusters while a class like *corn senesced green weeds* is more spread over the SOM grid. [6, Sec. 3.1]

In Figure 3.8b, the u-matrix is shown. The u-matrix contains the vector norms between the neighboring SOM nodes. Larger values imply larger spectral differences. The given u-matrix shows that the spectra of classes *stubble* and *celery* are significantly separated from the rest classes. [6, Sec. 3.1]

The SOM grid distribution per class is shown in Figure 3.9. For each class, mostly one or two soft clusters are visible. Differences of overlapping classes like *grapes untrained* and *vinyard untrained* are more visible in this visualization than in the dominant-class plot in Figure 3.8a. [6, Sec. 3.1]

3.7.3 Supervised Classification of Land Cover

We use the Salinas land cover dataset (see Section 3.7.1) for the evaluation of the supervised classification SOM. An RF classifier is used as a baseline. The dataset is split into a training and a test dataset in the ratio 30 : 70. The split ratio is selected to ensure enough datapoints in the training dataset and to ensure a proper evaluation. The evaluation results are the average results of five different random seeds (see Section 3.6.2). Scaling is applied on the hyperspectral input data for the SOM as described in Section 3.7.1. [6, Sec. 3.3]

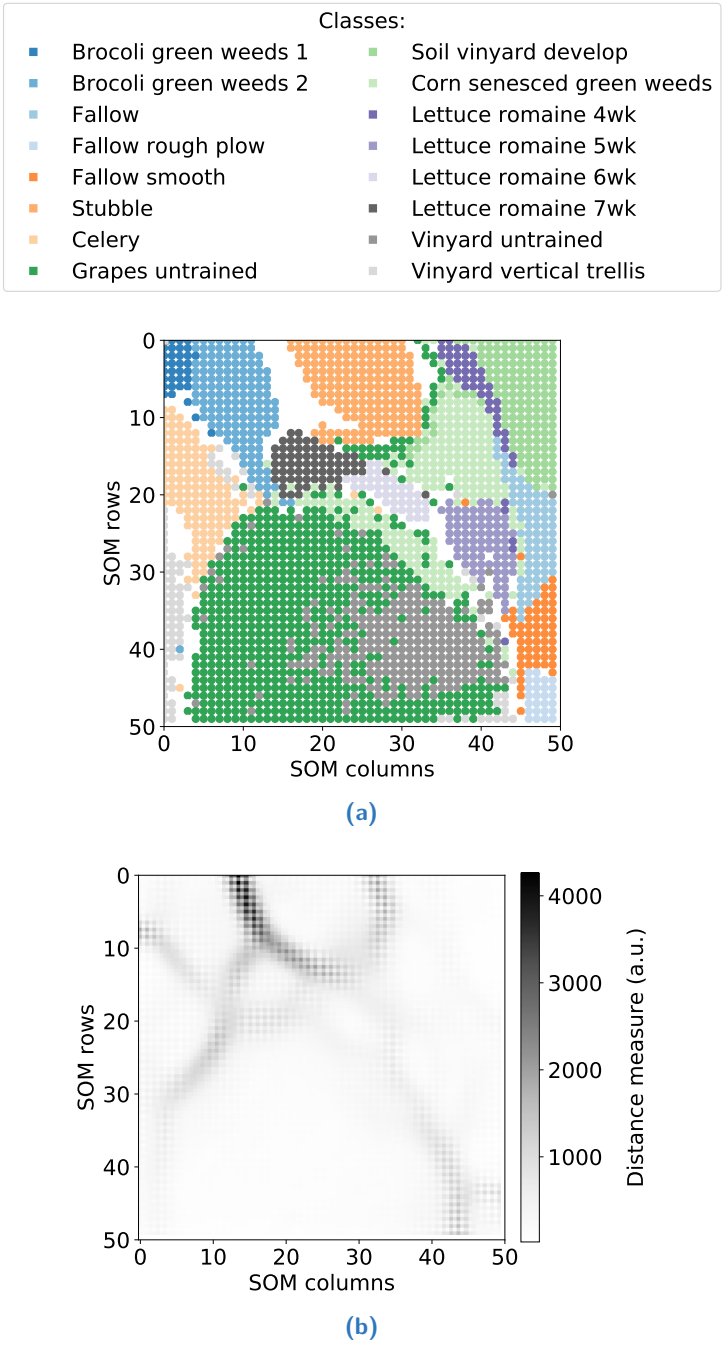


Figure 3.8: (a) Dominant classes per SOM node on the grid of the unsupervised SOM. The color white is used if no datapoint is mapped to a SOM node; (b) u-matrix of the trained unsupervised SOM. ((a) adapted and (b) reprinted from [6])

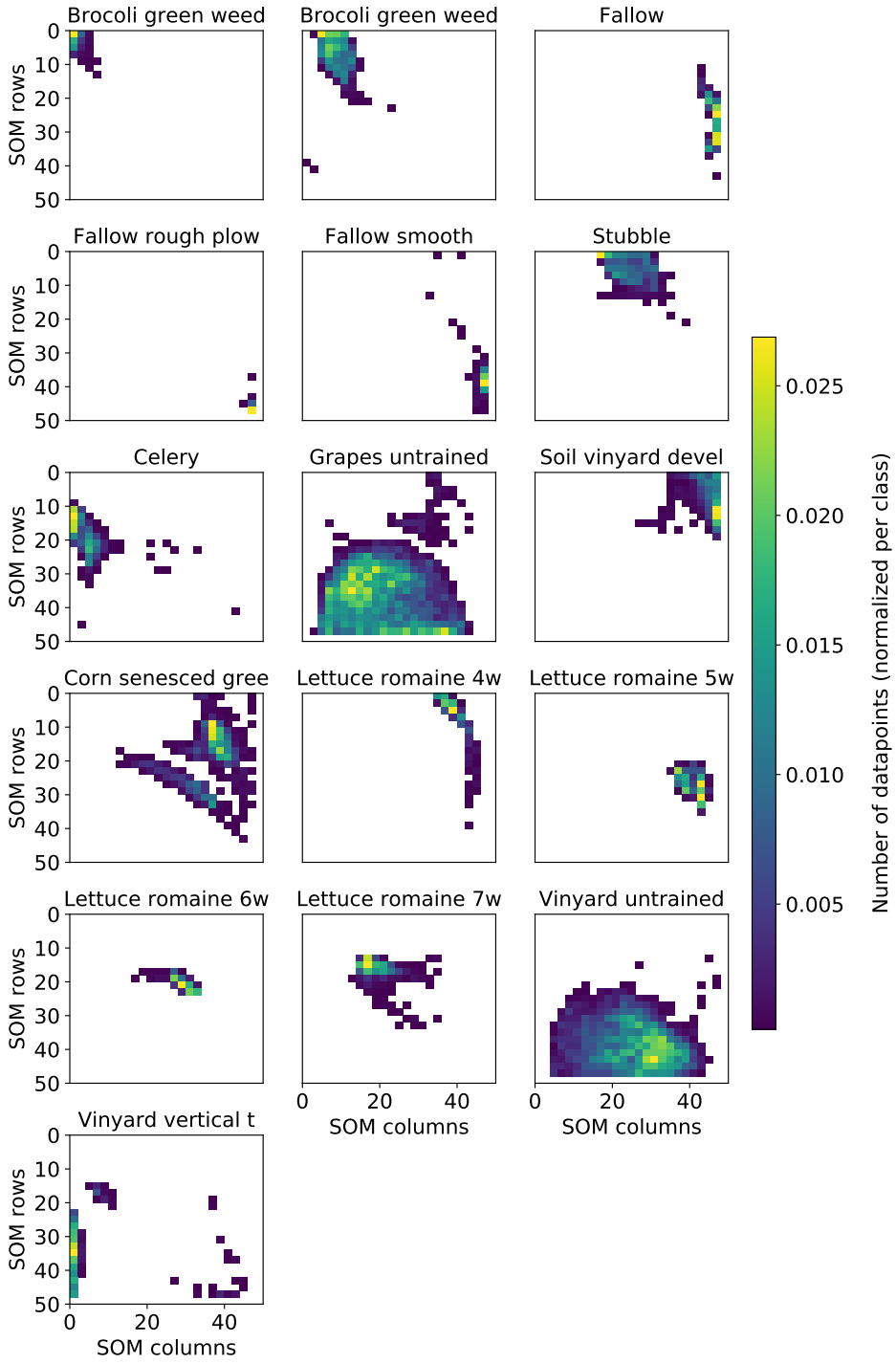


Figure 3.9: 2D histograms for every class of the Salinas dataset on the 2D grid of the unsupervised SOM. (adapted from [6])

Similar to Section 3.6.2, the hyperparameters of the classification SOM are set after a minor optimization and are provided in Table C.1. The hyperparameters of the classification SOM can be further optimized. The RF classifier is set up with 100 estimators and the scikit-learn default hyperparameters (see [131]). For the evaluation, we choose the metrics Overall Accuracy (OA), Average Accuracy (AA), and Cohen's Kappa Coefficient (κ) [... as defined in Section 2.7.6].

The classification results of the complete dataset are shown in the prediction map in Figure 3.10. The SOM achieves a test OA of $(76.9 \pm 0.2)\,\%$, AA = $(81.4 \pm 0.7)\,\%$ and $\kappa = (74.3 \pm 0.2)\,\%$. The training OA is $(77.2 \pm 0.7)\,\%$, the training AA is $(81.6 \pm 0.9)\,\%$, and the κ score on the training subsets is $(74.6 \pm 0.8)\,\%$. In contrast, the RF classifier achieves a test OA of $(93.5 \pm 0.1)\,\%$, AA = $(96.4 \pm 0.1)\,\%$ and $\kappa = (92.7 \pm 0.2)\,\%$, while the RF training metrics are all at about 100 %. The RF classifier performs significantly better than the classification SOM. Note that the classification SOM of the SuSi framework has not yet been fully optimized. Similar to the regression results (see Section 3.6.2), the differences between test and training metrics are lower for the SOM. Two findings can be derived. Firstly, the SOM is robust against overtraining in this study implied by the only marginal difference between training and test accuracies. Secondly, the SOM is not optimized on this classification task in terms of the classification algorithm itself as well as the hyperparameters. To optimize the algorithm and the large number of different hyperparameters for the classification SOM, we expect the potential users of the open-source SuSi framework [159] to share their experiences and to improve the framework. [6, Sec. 3.3]

Figure 3.11a shows the distribution of the classification output of the SOM. Nodes assigned to the same classes are closer together on the SOM grid due to the inclusion of the neighborhood during the training process. The u-matrix is shown in Figure 3.11b. Both plots in Figure 3.10 can be compared to the clustering results in Figure 3.8. Similar structures occur. Obvious differences can be explained by the different number of datapoints used for the clustering as well as by the application of different random seeds. The ability to differentiate between the 16 classes is illustrated in the confusion matrix in Figure 3.12. Classes like *grapes untrained* and *vinyard untrained* are often confused with the respective other as is also found in Section 3.7.2. Figure 3.13 shows the confusion matrix of the RF is shown. As expected from the OA and AA, most of the classes are classified correctly to nearly 100 %. The two classes *grapes untrained* and *vinyard untrained* are confused with each other, which is a comparable effect previously seen in the SOM. [6, Sec. 3.3]

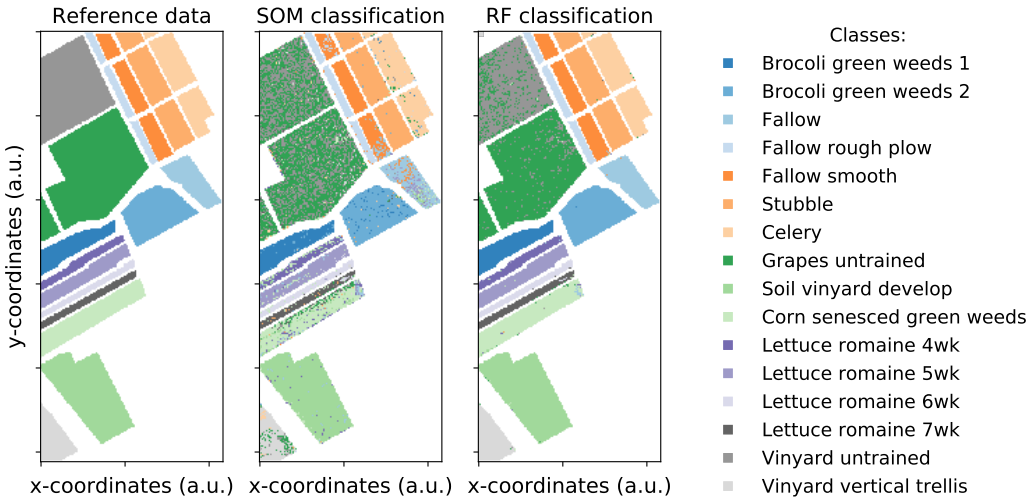


Figure 3.10: Map of the reference data (left) and the classification result of the classification SOM (center) and the Random Forest (RF) classifier (right) on the Salinas Valley dataset. The white area is ignored. The coordinates are not available and therefore illustrated in Arbitrary Units (a.u.). (reprinted from [6])

3.7.4 Semi-Supervised Classification of Land Cover

In the following, we provide a proof-of-concept for the semi-supervised classification SOM. The Salinas dataset (see Section 3.7.3) provides the basis for the evaluation. To study the semi-supervised functionality of the SuSi framework, we generate a new training dataset consisting of the whole Salinas dataset with the following modification: only two datapoints of every class are labeled, which results in a labeled dataset with 32 labeled pixels (0.06 %) and 54 097 unlabeled pixels. The semi-supervised SOM is evaluated based on the full Salinas dataset with all 54 129 labeled pixels. The hyperparameters of the semi-supervised classification SOM are set after a minor optimization and are provided in Table C.1. Similarly to the studies before, five different random seeds are applied for the classifiers as well as for the randomized choice of the 32 labeled pixels. To evaluate the classification results of the semi-supervised SOM, a supervised RF is only trained on the 32 labeled datapoints and evaluated on the full datasets with all 54 129 labeled pixels. [6, Sec. 3.5]

The test OA of the semi-supervised classification SOM is (67.3 ± 3.0) %, the test AA is (78.6 ± 2.6) %, and the test $\kappa = (64.5 \pm 3.3)$ %. Compared to the supervised classification results in Section 3.7.3, the OA is significantly lower while the AA does not differ much. Since the labeled training dataset consists of two datapoints of every class, the AA is expected to be larger than the OA for a very unbalanced dataset as for the Salinas dataset. A κ of more than 60% shows that the semi-supervised SOM is significantly better than a random classification. The training

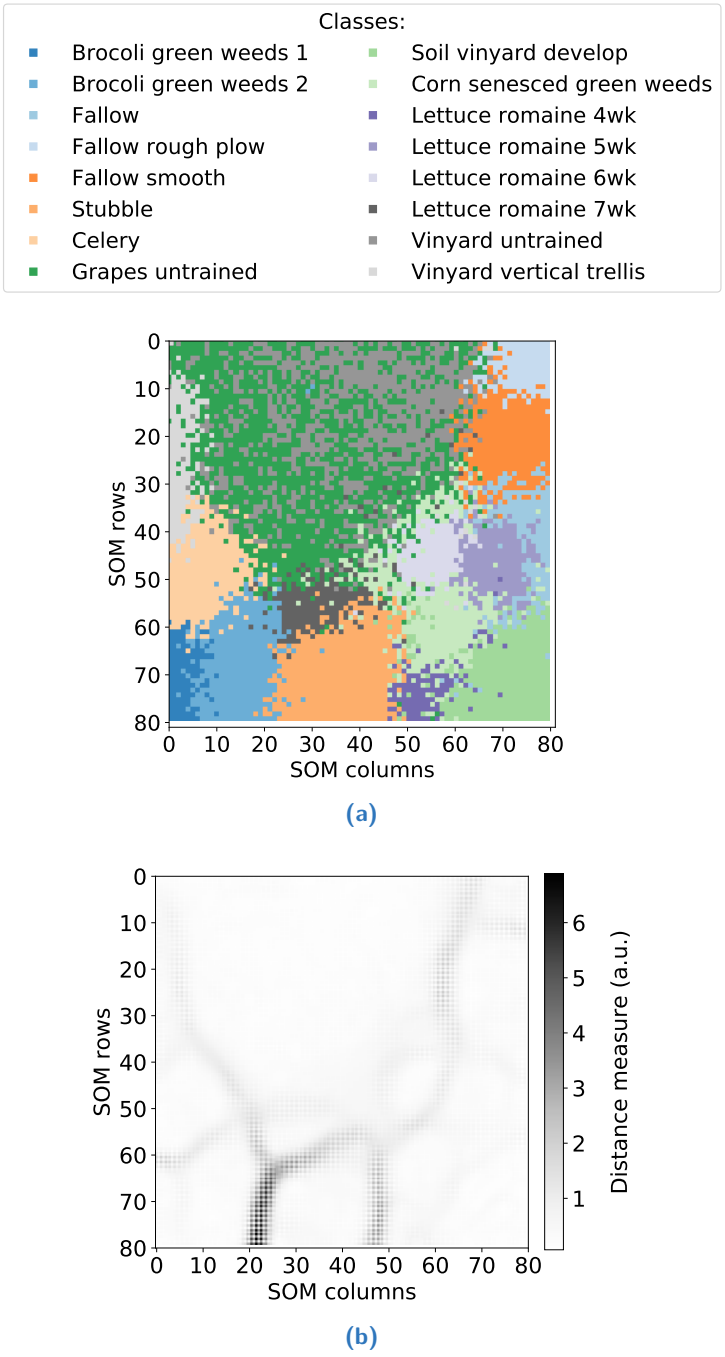


Figure 3.11: (a) Classification SOM distribution of the classes linked to each node as output of the classification calculated; (b) u-matrix of the classification SOM. ((a) adapted and (b) reprinted from [6])

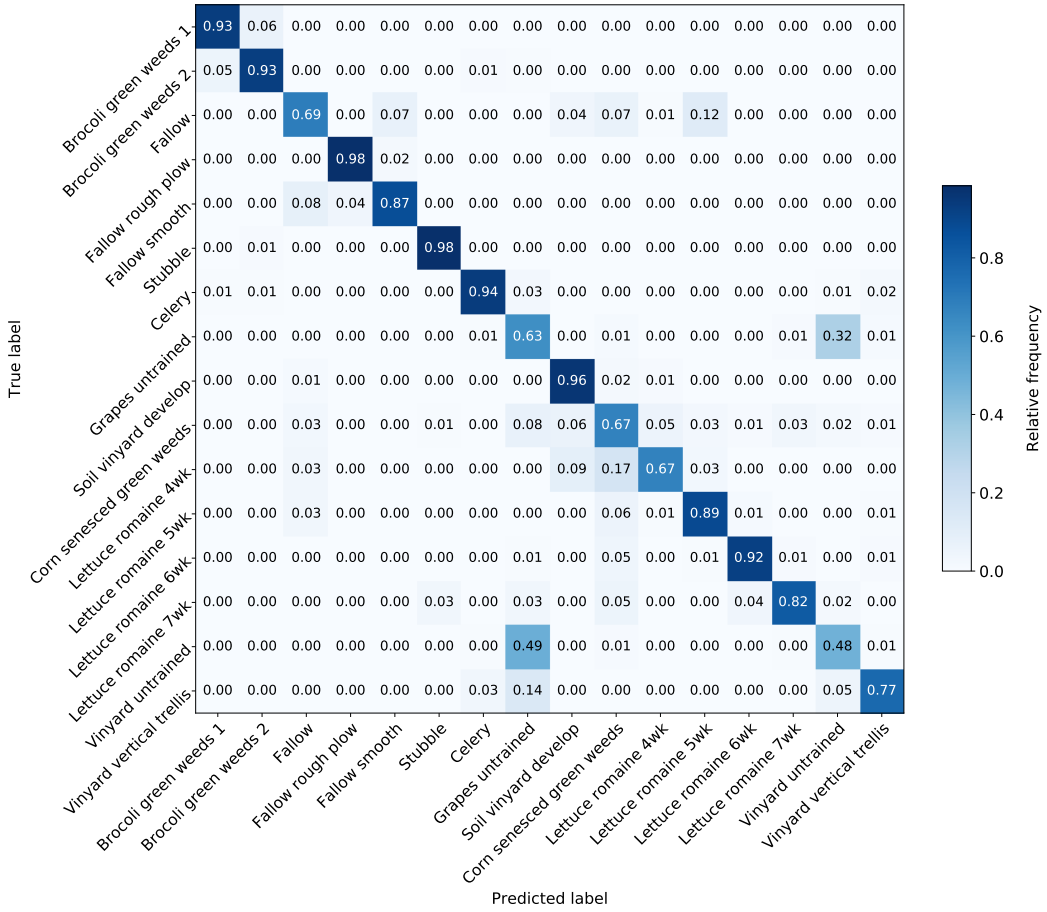


Figure 3.12: Normalized confusion matrix of the classification SOM. (adapted from [6])

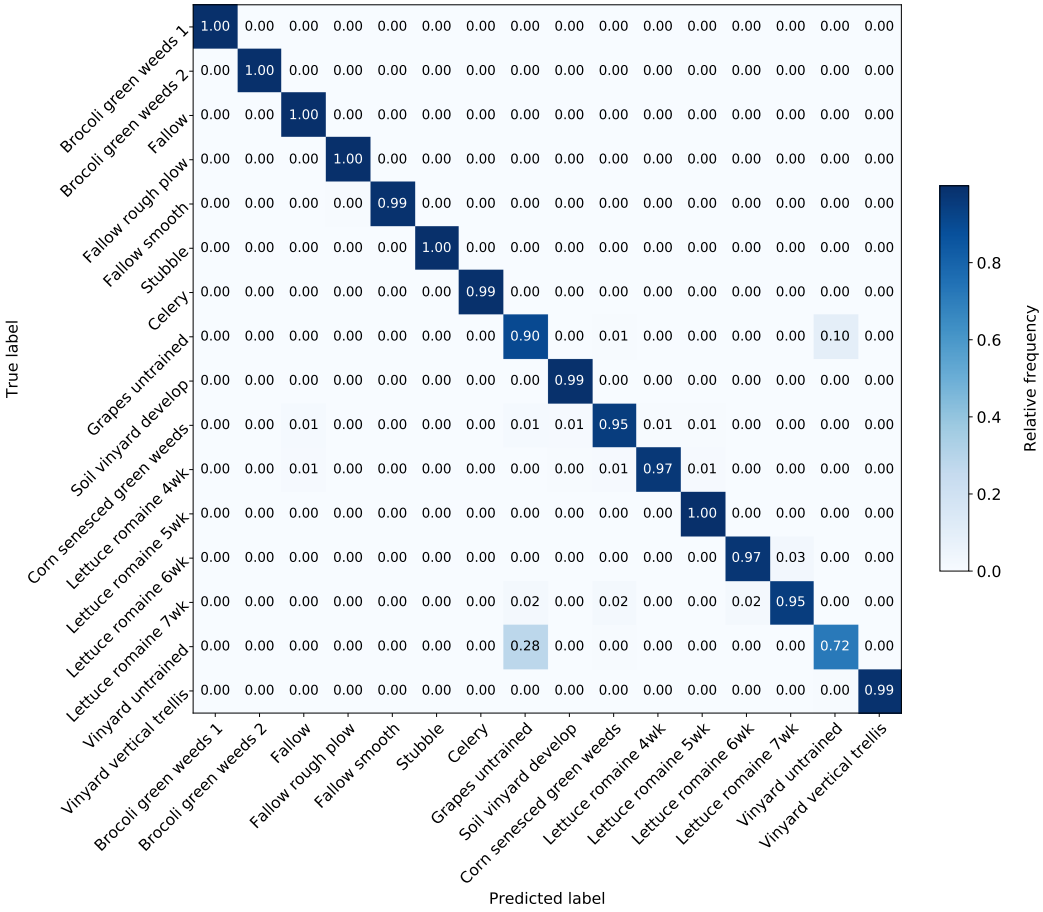


Figure 3.13: Normalized confusion matrix of the RF classifier. (adapted from [6])

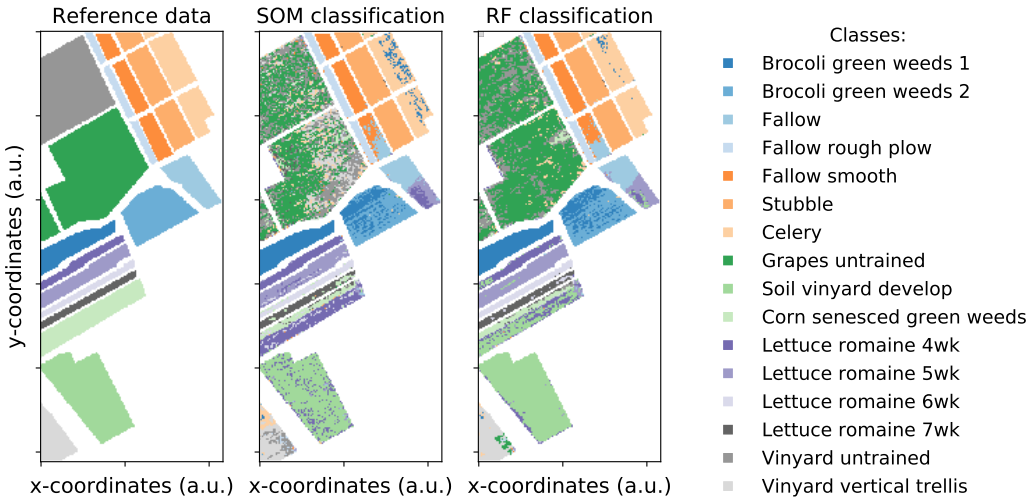


Figure 3.14: Prediction map of the semi-supervised SOM and RF classifier. The coordinates are not available and therefore illustrated in Arbitrary Units (a.u.). (reprinted from [6])

metrics are calculated based on the 32 labeled datapoints and are for the OA and AA about $(99.4 \pm 1.2)\,\%$ and κ of $(99.3 \pm 0.0)\,\%$. This indicates the capability of the SOM to adapt to the poorly-labeled dataset. [6, Sec. 3.5]

The supervised RF achieves a test OA of $(72.9 \pm 2.8)\,\%$, a test AA of $(81.3 \pm 2.0)\,\%$, and a test κ of $(70.2 \pm 3.0)\,\%$. Similar to Section 3.7.3, the train metrics are about 100 %. The RF classifier outperforms the semi-supervised approach only by a few percentage points. The differences between the performances of the SOM and the RF are significantly smaller than in the supervised case (see Section 3.7.3). This finding implies that a supervised approach might be a similar or even better choice in a case of a dataset with only a few labels. [6, Sec. 3.5]

Figure 3.14 shows the prediction map of the semi-supervised SOM with reasonable results. In general, the fields of the different classes in the Salinas dataset can be recognized as being of one major class. In Figure 3.15a, the output grid of the semi-supervised SOM with each node assigned to one class is illustrated. Compared to the supervised case in Figure 3.11a, clear circles can be seen that are artifacts of the early stages, and therefore larger learning rates, of the training of the semi-supervised SOM. With more labeled datapoints, a more diverse output grid could be trained. The distribution of the different labeled datapoints on the SOM grid is shown in Figure 3.15b. The dataset is spread over the full SOM grid with small clusters at the borders of the grid. [6, Sec. 3.5]

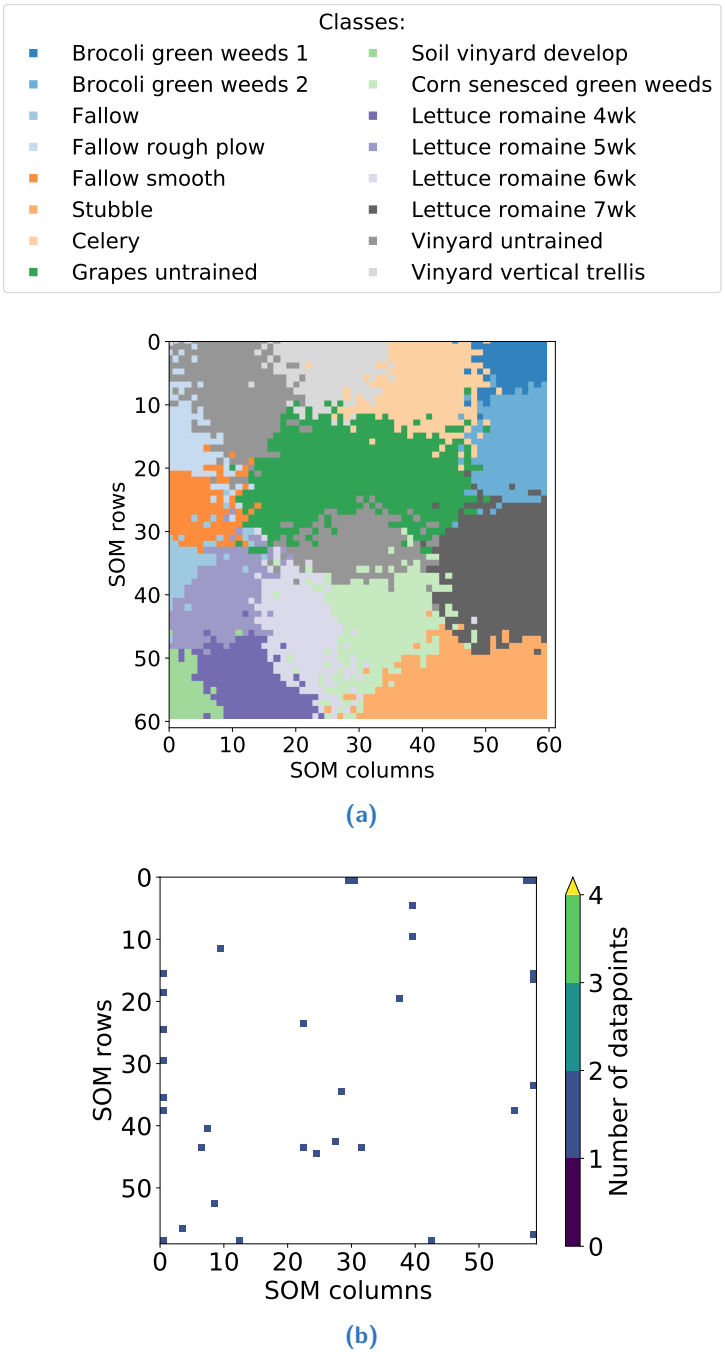


Figure 3.15: (a) Distribution of the semi-supervised SOM classification output calculated for each node; (b) semi-supervised regression SOM distribution of the BMUs of the dataset. ((a) adapted and (b) reprinted from [6])

3.8 Conclusions and Outlook

ML approaches, such as the commonly applied deep ANNs, require large training datasets. In hyperspectral regression, many available datasets include only a few labeled datapoints. While the acquisition of hyperspectral data gets increasingly affordable, reference data, such as soil moisture point measurements, is still time-expensive and costly to acquire. In this chapter, the SuSi framework is presented, based on SOMs. It can be applied for unsupervised clustering and visualization as well as supervised and semi-supervised regression and classification tasks.

We compare the framework with existing Python, R and C packages in Section 3.2 based on a selected list of requirements and based on the ease of use. The mathematical concept of the framework is presented in Sections 3.3 to 3.5. [...] [6, Sec. 4]

We demonstrate regression and classification results of the supervised SOM in the SuSi framework in Sections 3.6.2 and 3.7.3. The regression is performed on a small dataset while the classification SOM is applied on a relatively large dataset. All evaluation results of this [... chapter] are summarized in Table 3.2. The regression SOM outperforms the baseline classifier RF while simultaneously showing less overtraining. Similar to the supervised SOM regression performance, the supervised classification SOM achieves satisfactory results with potential to improve. We find that the performance metric based on the training dataset could function as an *out-of-bag estimate* for the dataset. This suggests that, in the case of the supervised regression and classification SOMs, we do not have to split the dataset necessarily, which might improve the training especially on small datasets. [6, Sec. 4]

The unsupervised and supervised capabilities of the SuSi framework are combined for solving semi-supervised tasks. Similar to the supervised regression and classification applications, we apply the semi-supervised regression and classification SOM on two different datasets to evaluate their performances. Both datasets are modified for the semi-supervised evaluation: only a few datapoints in the training dataset remain labeled. While the semi-supervised regression SOM clearly outperforms the RF baseline classifier, the semi-supervised classification SOM achieves satisfying results that are still below the RF performance. [6, Sec. 4]

In the future, the SuSi framework will be extended, optimized, and upgraded. In particular, the supervised and semi-supervised classification has great potential for methodological improvements. One promising approach is the batch mode [158] for adapting the SOM weights [... see [6]]. The handling of missing and incomplete data, as described in [167], is one example for a possible extension. Another

possible extension of the semi-supervised SOM is active learning [190]. In active learning, the ML model actively queries for unlabeled datapoints to be labeled that are most helpful for the underlying task. One further advantage of the SOM is the 2D output grid that can be used for visualization of the results of the SOM. This visualization can enhance the understanding for the underlying dataset. For example, the general ability to learn from datasets can be extended according to [165]. Furthermore, we plan to apply the SuSi framework on further datasets and to share our best practices in the context of handling the SuSi framework to ensure its effectiveness in hyperspectral remote sensing. [6, Sec. 4]

Table 3.2: Evaluation results of the soil moisture regression examples in Sections 3.6.2 and 3.6.3 as well as of the land cover classification examples in Sections 3.7.3 and 3.7.4. (reprinted from [6])

	Metric	Dataset	Supervised		Semi-Supervised	
			SOM	RF	SOM	RF
Regression	R^2 in %	test	95.3 ± 0.8	93.0 ± 2.2	81.9 ± 3.2	71.9 ± 6.0
		training	97.7 ± 0.6	99.1 ± 0.2	-	-
Classification	OA in %	test	76.9 ± 0.2	93.5 ± 0.1	67.3 ± 3.0	72.9 ± 2.8
		training	77.2 ± 0.7	100	99.4 ± 1.2	-
	AA in %	test	81.4 ± 0.7	96.4 ± 0.1	78.6 ± 2.6	81.3 ± 2.0
		training	81.6 ± 0.9	100	99.4 ± 1.2	-
	κ in %	test	74.3 ± 0.2	92.7 ± 0.2	64.5 ± 3.3	70.2 ± 3.0
		training	74.6 ± 0.8	100	99.3 ± 0.0	-

4 1D Convolutional Neural Networks for Hyperspectral Data

> *The difficulty lies not so much in developing new ideas as in escaping from old ones.*
>
> — **John Maynard Keynes**
> (Economist)

This chapter includes material from

Felix M. Riese and Sina Keller. "Soil Texture Classification with 1D Convolutional Neural Networks based on Hyperspectral Data". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W5 (2019), pp. 615–621. It is cited as [8] and marked in orange.

4.1 Introduction

The texture of soil influences the soil's capability to store water and its fertility. Therefore, the classification of soil texture is important for agricultural applications as well as for the monitoring of environmental processes [see Section 2.3]. The term *soil texture* refers to the relative content of soil particles of various sizes. It is determined by the percentages of clay, sand and silt in the soil. Soil texture can be classified with respect to these three properties e.g. according to the KA5 taxonomy defined by Boden [35]. [8, Sec. 1]

The monitoring of soil texture with in-situ measurements is expensive and is not feasible on large areas. To cover such large areas, optical remote sensing provides a good alternative. For example, hyperspectral sensors are such optical remote sensing devices which measure solar reflectance spectra of objects. The information of soil texture derived from the soil reflectance corresponds to specific absorption features

of clay or other soil mineral and organic constituents [191]. For a classification of soil texture based on hyperspectral data, a model has to be developed that is able to link different reflectance spectra to the respective soil textures. [8, Sec. 1]

Shallow and deep Machine Learning (ML) approaches are applied in this classification of soil texture based on hyperspectral data. Shallow learning approaches such as Random Forest (RF) and Support Vector Machine (SVM) have shown good performance in [192, 108]. Their potential is limited since shallow learning approaches are often depending on feature engineering and dimensionality reduction. Deep learning approaches, such as Convolutional Neural Networks (CNNs), are widely applied in hyperspectral remote sensing in recent years [12, 15], as described in Section 2.7.2. These CNNs approaches are mostly applied on 2-dimensional (2D) hyperspectral images rather than on 1-dimensional (1D) spectra. This chapter fills this gap with the following main contributions:

- the pre-processing, splitting and class aggregation of the Land Use/Cover Area Frame Survey (LUCAS) soil dataset [193],
- the development of three innovative 1D CNN approaches,
- the open-source publication of the code of these approaches [194], and
- the application and evaluation of these approaches, as well as the comparison to existing ML approaches, in the soil texture classification.

This chapter is structured as follows. The related work is presented in Section 4.2, including ML classification of soil texture with a focus on deep learning and studies based on the LUCAS dataset. The LUCAS soil dataset is described in Section 4.3, followed by the pre-processing and dataset splitting. The methodology of this chapter is described in Section 4.4. The results of the soil texture classification are presented in Section 4.5 and discussed in Section 4.6. Finally, Section 4.7 concludes this chapter.

4.2 Related Work

In this section, we briefly review the published research which is related to the presented classification of soil texture based on hyperspectral data. A first review of geological remote sensing is given by Cloutis [191]. Traditional approaches like nearest mean, nearest neighbor, maximum likelihood, hidden Markov models and spectral angle matching for the classification of soil texture show acceptable results [195, 196, 197]. [8, Sec. 2]

Many research disciplines apply 1D CNNs for different purposes [198]. For example, 1D CNNs can be used in the real-time monitoring and classification of electrocardiograms [199]. Structural damage detection in civil infrastructure is another application [200]. Another example is the real-time fault detection of motor faults [201] and modular multilevel converters [202].

The increasing popularity of deep learning approaches in many research disciplines has also reached the field of remote sensing. Deep learning approaches turn out to solve classification tasks better than shallow methods [81]. Zhu et al. [15] give a detailed overview of deep learning in remote sensing and Petersson et al. [12] review the application of deep learning in hyperspectral image analysis. The application of 2D CNNs for classification and regression tasks based on hyperspectral images is proposed among others by Makantasis et al. [203]. The two dimensions refer to the two spatial dimensions of hyperspectral images. Since hyperspectral images consist of several spectral channels, one additional dimension is possible: the spectral dimension. This spectral dimension can be utilized as a third dimension of a CNN or can be analyzed on its own by 1D CNNs. Hu et al. [204] propose the use of 1D CNNs based on the spectral dimension of hyperspectral images. This network is described in Section 4.4 in detail. [8, Sec. 2]

In most publications, the applied ML approaches are trained on a specific training dataset. Zhao et al. [205] propose the use of pre-trained networks for the hyperspectral image classification, so called transfer learning. In transfer learning, it is assumed that the trained features of a neural network are comparable between different image datasets. Therefore, this approach is time-saving and enables training on smaller datasets. The latter is possible since the training of the neural network is mostly done with another dataset beforehand (pre-trained). Transfer learning of a 1D CNN is proposed by Liu et al. [121]. They apply the CNN for the regression of clay content in the soil based on the LUCAS soil dataset. We describe this approach in detail in Section 4.4 and compare it to other methods with respect to our classification task. [8, Sec. 2]

Based on the LUCAS dataset, a variety of studies exists. For example, the estimation of N_2O is shown by Lugato et al. [206]. Several studies focus on the soil organic carbon content [207, 208, 209]. Studies about land cover and land use diversity benefit from the large area covered by the LUCAS dataset. They calculate landscape indices [210, 211] and combine land use data with Landsat images [212]. The soil erodibility is studied by Panagos et al. [213].

As stated in Section 4.1, the soil texture information is addressed in several studies applying ML techniques. For example, Ballabio et al. [214] perform a regression of

soil properties such as the three layers of soil texture (clay, sand, silt) plus coarse fragments with the Multivariate Adaptive Regression Splines (MARS) model. A recent study applies 1D CNNs to estimate the clay content [121]. [8, Sec. 3.1]

4.3 The LUCAS Soil Dataset

The Land Use/Cover Area Frame Survey (LUCAS) Soil dataset is a large and comprehensive survey of topsoil [43, 44, 45]. The dataset was collected in different locations all over Europe between 2009 and 2012. Further measurements have been performed in 2018 but are not included into this publication. The LUCAS dataset consists of about 22 000 datapoints that include physico-chemical properties like the percentage of coarse fragments, the particle size distributions clay, sand and silt, the pH value, the organic carbon content, the carbonate content, the total nitrogen content, the extractable potassium content, the phosphorus content, the cation exchange capacity and metals. Additionally, this dataset includes continuous reflectance spectra [...], referred to as hyperspectral data in the following. [...] The new 2018 dataset will include, among others, soil biodiversity properties and soil moisture data [45]. [8, Sec. 3.1]

The hyperspectral data of the LUCAS is acquired with a XDS Rapid Content Analyzer by FOSS NIRSystems (Laurel, US) [44]. The sensor is used in a laboratory setup. In Section 2.4.1, the XDS Rapid Content Analyzer is compared to the other sensors applied within the scope of this thesis. This sensor covers a spectral range of 400 nm to 2500 nm with a spectral resolution of 0.5 nm and 4200 spectral bands.

Pre-Processing The calibration and the corrections, which are already included in the LUCAS dataset, are described in Tóth et al. [44]. For the presented studies, three pre-processing steps are applied, as illustrated in Figure 4.1.

1. Dimensionality reduction to reduce the number of spectral bands of the hyperspectral data from 4200 to 256 with minimal information loss by averaging 16 to 17 neighboring bands to one new band. This dimensionality reduction is necessary for practical reasons, e.g. to reduce the computation time and to avoid overtraining by minimizing the weights of the networks. [8, Sec. 3.2]
2. Removal of the duplicates of the multiple hyperspectral datapoints per soil sample and removal of unused features to generate a minimal classification dataset. This step reduces the bias of the training and evaluation of ML techniques. [8, Sec. 3.2]

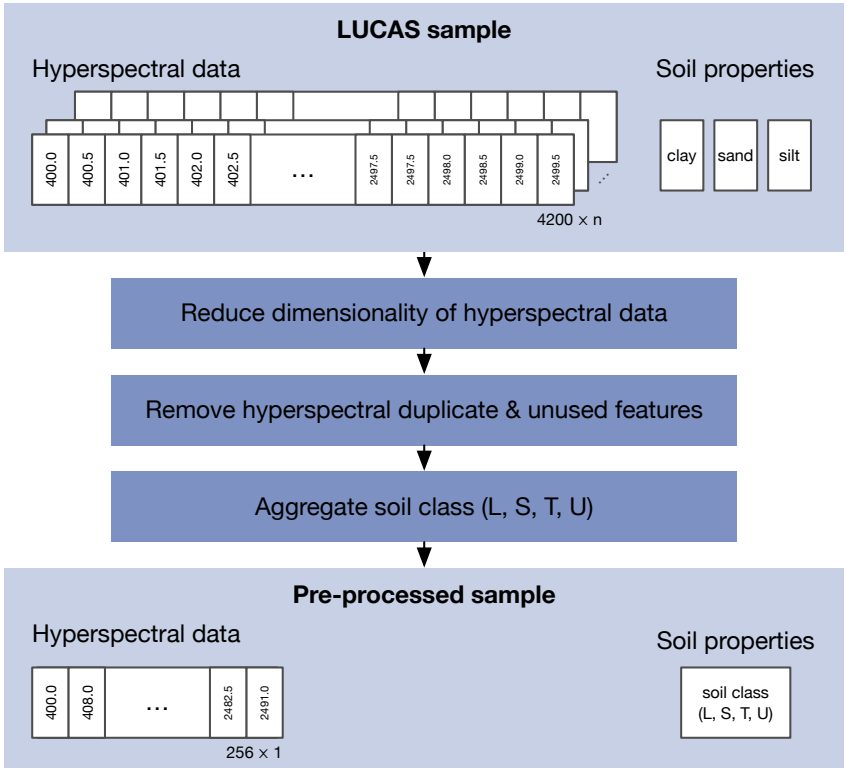


Figure 4.1: Preprocessing workflow with three steps: the dimensionality reduction, the removal of the duplicates and the aggregation of the general soil classes. (adapted from [8])

3. Aggregation the general soil classes L, S, T, U for the supervised classification performed below. [8, Sec. 3.2]

The four soil classes L, S, T, U are derived from the Bodenkundliche Kartieranleitung ed. 5 (KA5) soil taxonomy [35] based on the distribution of clay, sand, and silt contents. L stands for loam (German: *Lehm*), S stands for sand (German: *Sand*), T stands for clay (German: *Ton*), and U stands for silt (German: *Schluff*). The four main soil classes and the 31 subclasses are listed in [35]. Compared to the original publication of this study [8], the definition of the four soil classes is slightly changed. In [8], the four soil classes L, S, T, and U are derived from the first letter of the respective subclass. In the presented study, the four soil classes are derived directly from the definition of the KA5 main group soil classes. This adaptation of the study affects the soil class distributions and all derived results in the following sections. Figure 4.2 illustrates the distribution of the four soil classes in the LUCAS dataset.

Dataset Splitting To evaluate the performance of the different classification approaches, the pre-processed dataset is split into three disjoint subsets: the training subset, the validation subset and the test subset. We choose random splitting with a ratio of approximately 60 : 20 : 20. In total, the training subset consists of

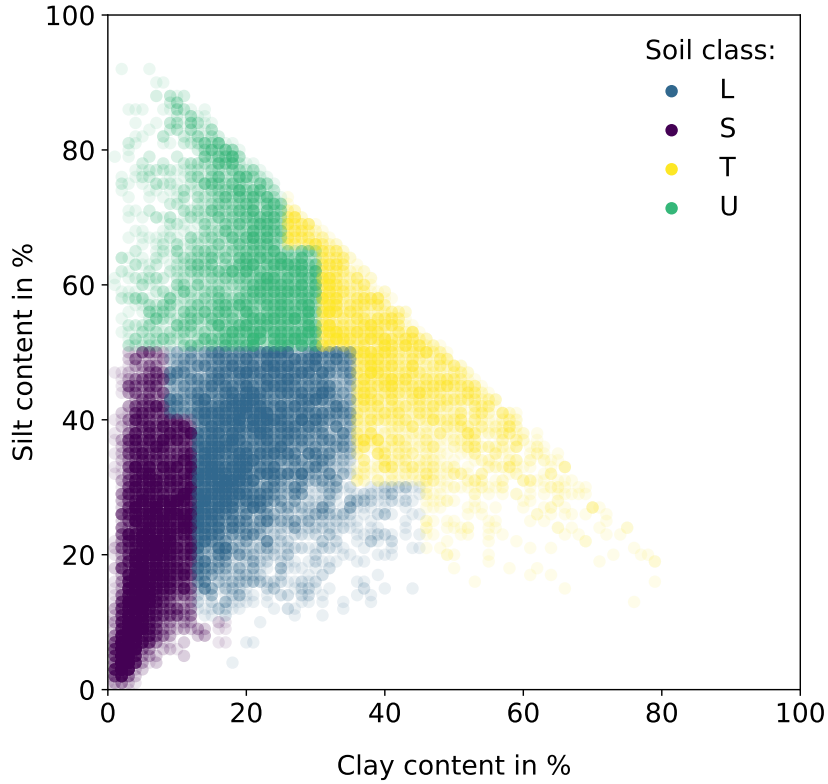


Figure 4.2: Clay and silt content of all pre-processed datapoints of the LUCAS dataset. The color of the datapoints symbolizes the respective soil class introduced in this chapter. (adapted from [8])

9759 datapoints, the validation subset contains 3109 datapoints and the test subset contains 3208 datapoints. The class distributions of the three datasets are shown in Figure 4.3. One datapoint consists of 256 hyperspectral reflectance values and one of the four soil classes L, S, T, U. [8, Sec. 3.2]

4.4 Supervised Classification Models

For supervised learning based on hyperspectral images, various methods exist. For example, Keller et al. [9, 23] combine ten shallow learning techniques for the regression of environmental variables. For the presented soil texture classification task, we study several ML approaches. All approaches are CNNs except for the RF classifier. The RF classifier is established in remote sensing applications (see e.g. [108]). Therefore, the classification accuracy of the several CNN approaches is compared against the results of the RF classifier. [8, Sec. 4]

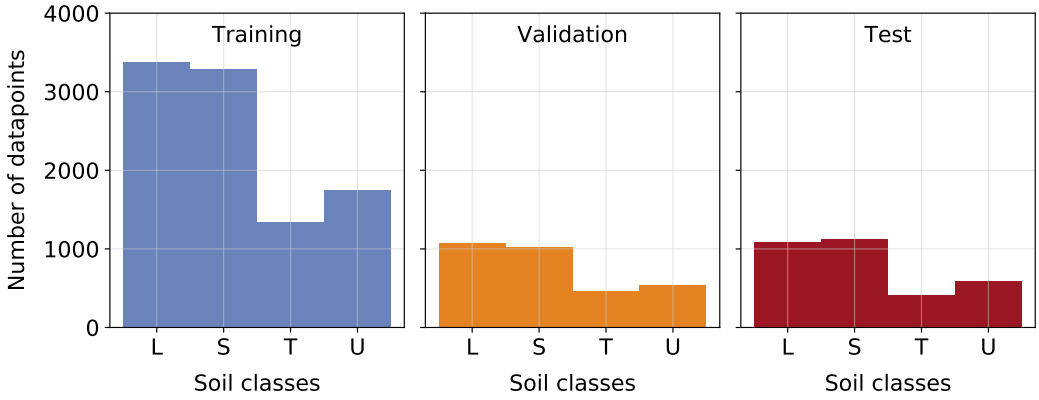


Figure 4.3: Class distribution of the LUCAS dataset divided into training, validation, and test subsets. (adapted from [8])

In addition to the RF classifier, we study five different 1D CNN architectures. Two of them have been introduced by Hu et al. [204] and Liu et al. [121] and are modified for the underlying classification task. The 1D CNN of Hu et al. [204] consists of one 1D Convolutional (CONV) layer followed by one max-pooling layer and one Fully-Connected (FC) layer. The 1D CNN of Liu et al. [121] was introduced as a regression approach for the estimation of clay content based on the LUCAS dataset. It consists of four 1D CONV layers each followed by a max-pooling layer. At the end of each network by Hu et al. [204] and Liu et al. [121], we implement one FC layer with a softmax activation and four outputs. This prepares these CNNs for the classification task of this study. [8, Sec. 4]

Innovative CNN Architectures In addition to these two existing CNN approaches, we introduce three 1D CNN architectures for the soil texture classification. All three architectures are inspired by the LeNet5 network [215]. In order to distinguish between the three implemented CNNs, we refer to the three architectures as LucasCNN, LucasResNet and LucasCoordConv. In Figure 4.4, the architectures of the three LUCAS networks are illustrated. The *LucasCNN* consists of four CONV layers, each followed by a max-pooling layer with a kernel size of 2. After flattening the output of the fourth CONV layer, two FC layers are implemented and, as before, one FC layer with a softmax activation and four outputs is placed at the end of the network. [8, Sec. 4]

For the *LucasResNet*, we add an identity block to the LucasCNN. The input vector is bypassing the four CONV layer and is concatenated to the activation of the last CONV layer and before the first FC layers. The special feature of the *LucasCoordConv* is one coordinates layer placed before the first CONV layer of the LucasCNN. Liu et al. [216] introduced such a coordinates layer first. The network architecture after

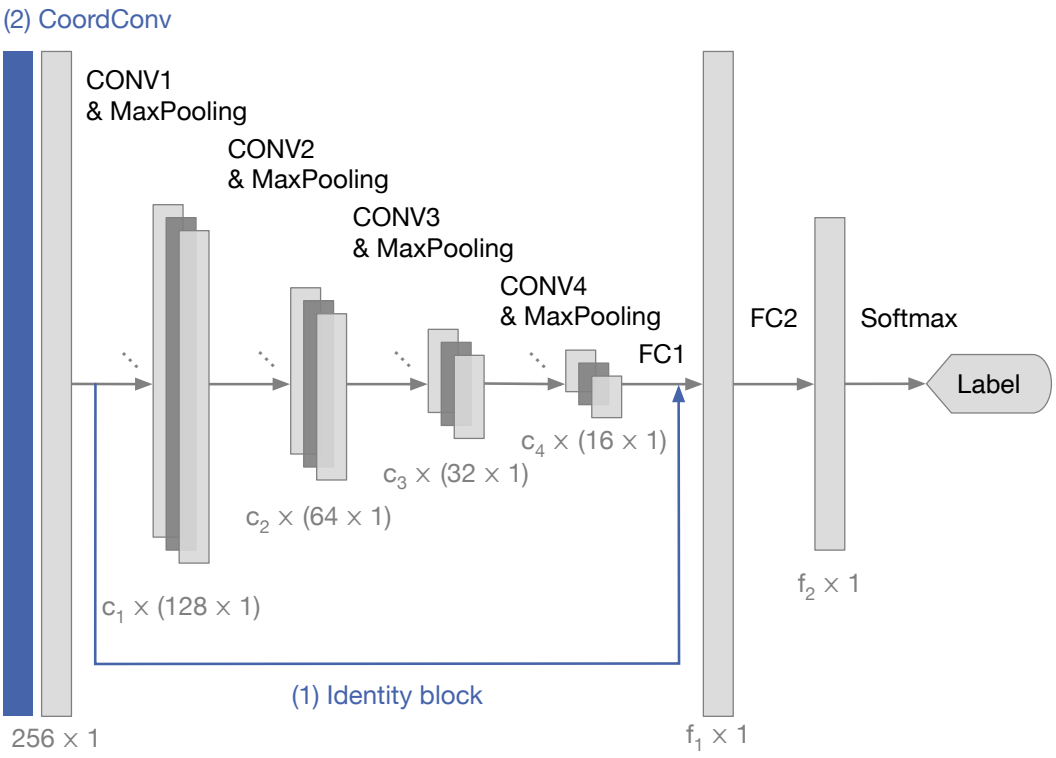


Figure 4.4: Flowchart of the LucasCNN (grey). The network consist of Convolutional (CONV), Fully-Connected (FC) layers and max-pooling layers. The i-th CONV layer consists of c_i filters and the j-th FC layer consists of f_j units. For the LucasResNet, an (1) identity block is implemented. For the LucasCoordConv, a (2) coordinates layer is inserted before the first CONV layer. At the end of each network, a softmax layer provides the classification output as a 4-dimensional vector. (adapted from [8])

the first CONV layer remains the same as in the LucasCNN. The code of all presented implementations of 1D CNNs is published on GitHub [194]. [8, Sec. 4]

Model Optimization ML models are characterized by two types of parameters: model parameters and hyperparameters. Model parameters are adapted during the training of the model and hyperparameters are set beforehand. For the RF classifier, we use the implementation of Pedregosa et al. [131] with 10 000 estimators. This configuration achieves good results e.g. in a regression task based on hyperspectral data [23]. All hyperparameters of the two existing CNNs are adopted from the respective introducing publications. The hyperparameters of the three new approaches LucasCNN, LucasResNet and LucasCoordConv are determined with a hyperparameter optimization process [see Section 2.7.5]. [8, Sec. 5]

The training dataset is used for the training of each CNN while their evaluation is performed on the validation dataset. The hyperparameters of the

all five 1D CNN approaches are shown in Table C.2. The test dataset is not used for this procedure. [8, Sec. 5]

4.5 Results

Table 4.1 lists the results for the presented classification of soil texture. The results are compared based on the Overall Accuracy (OA), Average Accuracy (AA), and Cohen's Kappa Coefficient (κ), which are defined in Section 2.7.6. The shallow learning approach RF shows the worst performance of all six applied classifiers with an OA of 0.61. The CNN approach by Liu et al. [121] shows slightly better results with an OA of 0.64.

The four CNN approaches Hu et al. [204], LucasCNN, LucasResNet, and LucasCoordConv show similar results with only minor differences. The best performance in terms of OA shows the CNN by Hu et al. [204] and LucasCNN with OA of 0.71. The CNN by Hu et al. [204] and the LucasCoordConv show the best performance in terms of κ, with κ of about 0.59. The best classification performance on the individual classes shows the LucasCoordConv with an AA of 0.69.

The confusion matrices of all six ML approaches are shown in Figure 4.5. Ideally, a confusion matrix would show only values of 1.0 on the diagonal and 0.0 in the other cells. In general, the four soil texture classes are more confused into the classes L and S than in the classes T and U. Especially the RF and the CNN by Liu et al. [121] show issues in the classification of the class T, which is confused with the class L in 35 % to 48 % of the datapoints. Most classifiers can classify the classes L and S in over 65 %, while class U is only correctly classified in 40 % to 59 % of the datapoints. The LucasCoordConv shows the best individual, relative performance in the classification of the class S.

Table 4.1: Classification results based on the test subset. (adapted from [8])

Model	OA	AA	κ
RF	0.61	0.55	0.44
Liu et al. [121]	0.64	0.58	0.48
Hu et al. [204]	**0.71**	0.68	**0.59**
LucasCNN	**0.71**	0.68	0.58
LucasResNet	0.70	0.66	0.57
LucasCoordConv	0.70	**0.69**	**0.59**

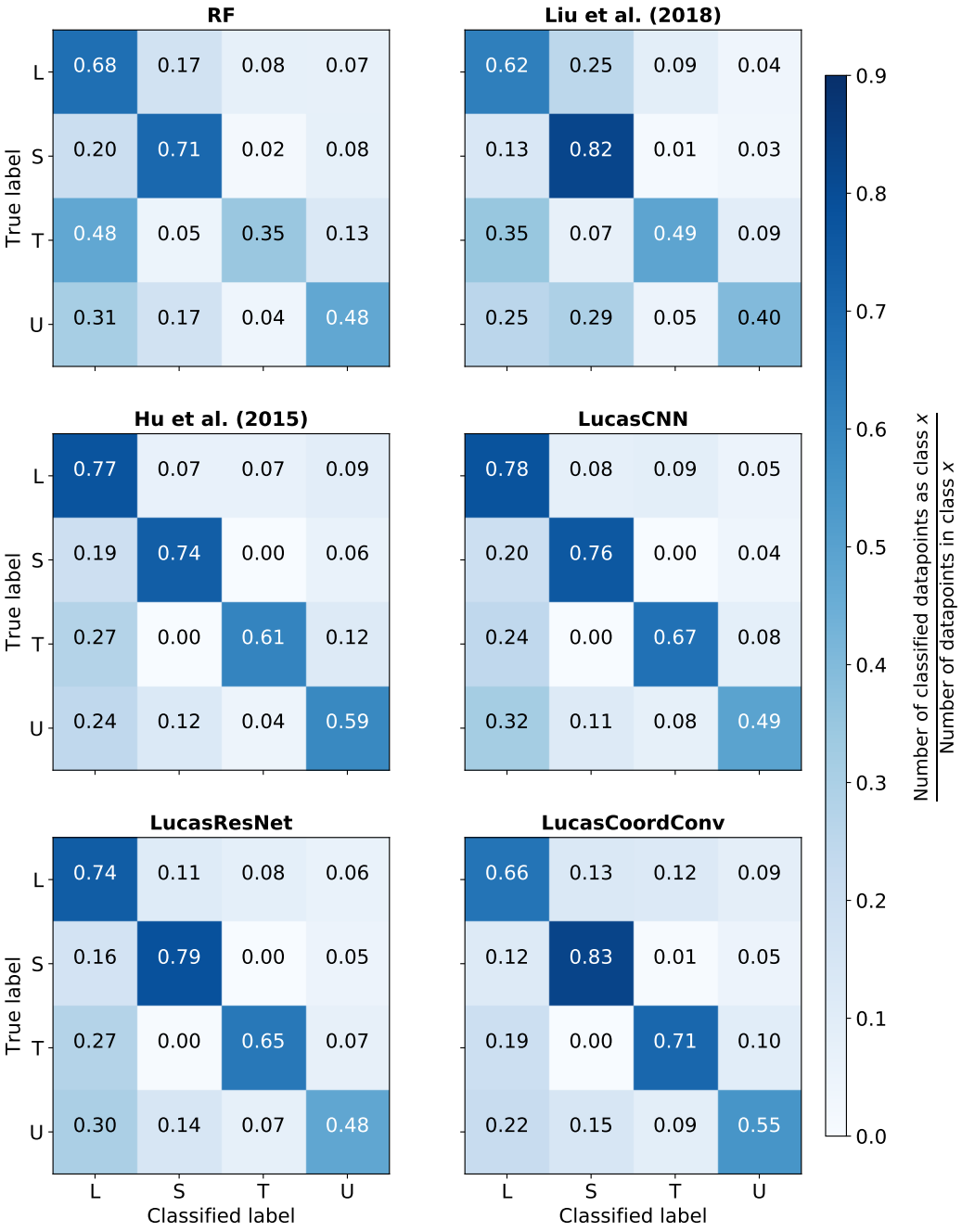


Figure 4.5: Normalized confusion matrices based on the test dataset. (adapted from [8])

Figure 4.6 shows the misclassified datapoints for every ML approach. In general, the datapoints close to the class borders are more often misclassified than other datapoints (see also Figure 4.2). Exceptions to this observation are the class T for the RF and the CNN by Liu et al. [121], the class U for the LucasCNN and LucasResNet, and the class L for the LucasCoordConv.

4.6 Discussion

The CNN of Hu et al. [204] shows the best overall performance in this classification study. It is the most basic CNN implementation in this study and uses the largest kernel size of 28 (compared to 3) and pooling size 6 (compared to 2). This result implies that the soil texture classification task based on the LUCAS dataset might be solved without applying deeper CNN architectures.

The introduced CNNs—LucasCNN, LucasResNet, and LucasCoordConf—show similar results in this study. Compared to the CNN by Liu et al. [121], their architectures are quite similar. The three introduced CNNs include two FC layers at the end and are trained with fewer epochs and, partially, with smaller batch sizes. The increase of classification performance from the CNN by Liu et al. [121] to, for example, the LucasCNN emphasizes the importance of an FC layer after the CONV layers. The value of a second FC layer, as implemented in LucasCNN, LucasResNet, and LucasCoordConv, does not show any performance increase compared to the CNN by [204], which includes only one FC layer.

The identity block in the LucasResNet does not significantly change the performance in terms of the evaluation metrics or the confusion matrices. Further, the CoordConv layer, as implemented in the LucasCoordConv, improves the accuracy of the individual soil texture classes. The best individual class performance for the classes L and S can be explained with the class distribution in the LUCAS dataset, shown in Figure 4.3. Since the classes L and S contain the most datapoints in the dataset, the trained ML approaches can adapt better to these classes.

The class definitions, as described in Section 4.3, rely on the KA5 taxonomy. These classes have been defined by humans. The differences between the soil textures at the class borders are marginal by definition. The distributions of the misclassified datapoints in Figure 4.6 show that the datapoints at the class borders can not be classified well. Instead of discrete classes, continuous clay, silt, and sand contents could be used in future work to resolve this finding.

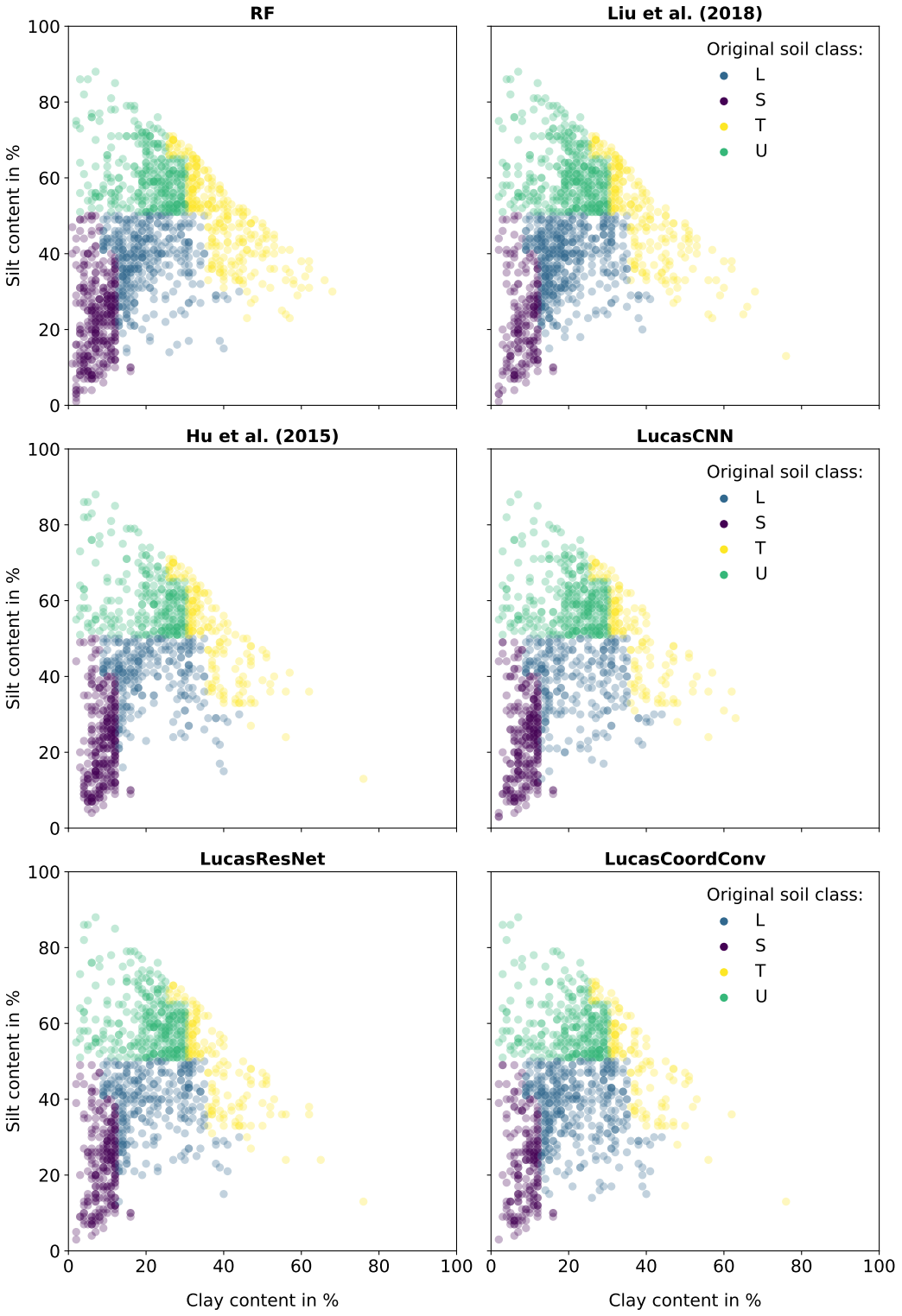


Figure 4.6: Misclassified datapoints with respect to their silt and clay contents for the six applied classifiers. The displayed class colors illustrate the true, original class of a datapoint.

Regarding the original publication [8], the results are comparable. In the original publication, the CNN by Hu et al. [204] also shows the best OA and κ while the LucasCoordConv shows the best AA. Overall, the values for OA are slightly lower in this study, while the values for the AA have increased slightly. The performance of the CNN by Liu et al. [121] is significantly lower with the soil texture classes in the presented study.

4.7 Conclusions and Outlook

> In this [... chapter], we address the classification of soil texture based on hyperspectral data with 1D CNNs. We use the freely available LUCAS soil dataset and describe its pre-processing and splitting in detail. For the classification of the dataset, we apply a RF classifier as well as two existing 1D CNNs by Hu et al. [204] and Liu et al. [121]. In addition, we introduce three new approaches LucasCNN, LucasResNet and LucasCoordConv. [8, Sec. 6]
>
> After the hyperparameter optimization of the three new approaches, we compare the classification performance of all six approaches based on the metrics OA, AA and κ as well as the confusion matrices. We conclude, that the RF classifier is incapable of handling this classification task sufficiently. [...] The most basic CNN approach by Hu et al. [204] achieves the best performance in OA and κ. The introduced LucasCoordConv, which includes a coordinates layer according to Liu et al. [216], performs best regarding the AA. This means that this approach performs best on each individual class. [8, Sec. 6]
>
> This study presents a further step towards the classification of hyperspectral data based on CNNs. Although up to now, 1D CNNs are often underrated in context of hyperspectral classification tasks, we demonstrate their potential on the LUCAS dataset. In general, the application of 2D and 3D CNNs on point measurements as the LUCAS dataset is not possible by definition. However, the results of this publication can be of value for studies focusing methodologically on 3D CNNs utilizing the spectral dimension as third dimension, e.g. Chen et al. [217]. In future work, we can further enhance the introduced LucasCNN, LucasResNet and LucasCoordConv and include additional variables of the rich LUCAS dataset. Regularization methods like dropout and batch normalization can help to generalize the presented CNN approaches. Additionally, techniques like transfer learning with 1D CNNs and their applications on new datasets like the LUCAS 2018 [45] dataset are promising. Furthermore, the de-

veloped methods of this publication can be applied on upcoming hyperspectral satellite data like Environmental Mapping and Analysis Program (EnMAP). [8, Sec. 6]

5 Dataset Shift in Hyperspectral Regression

> *The conditions under which the system was developed will differ from those in which we use the system.*
>
> — **Quionero-Candela et al.**
> (*Dataset Shift in Machine Learning* [63])

5.1 Introduction

Generalization is the aim of the training of a Machine Learning (ML) model. The Independent and Identically Distributed (i.i.d.) assumption requires that the training dataset of a ML estimation needs to be as similar as possible to the reality. In the case of most ML studies, the training dataset needs to be as similar as possible to the test dataset for the evaluation. If the i.i.d. assumption is not met, meaning that there are differences between the training and the test dataset, this is called dataset shift.

In hyperspectral remote sensing, datasets cover only a limited subset of a scene. If a training dataset is taken from this limited scene, its distributions will differ from the rest of the scene, and they will notably differ from areas outside of this scene. These differences can lead to dataset shift [218]. For the same reason, small datasets and datasets with a small number of labeled datapoints are prone to dataset shift. In this context, the term *small* means small in relation to the dimensionality of the data (comparing n to m, see Section 2.2) and small in relation to the number of classes (in the classification case) or the target value range (in the regression case). An overview of dataset shift and its different types is presented in Section 2.5.2.

Figure 5.1 illustrates the structure of the studies in this chapter. In this chapter, the detection and effects of dataset shift are studied based on a hyperspectral Unmanned Aerial Vehicle (UAV) dataset. In Section 5.2, the related work is summarized, which is relevant to the presented studies. The Aerial Peruvian Andes Campaign (ALPACA)

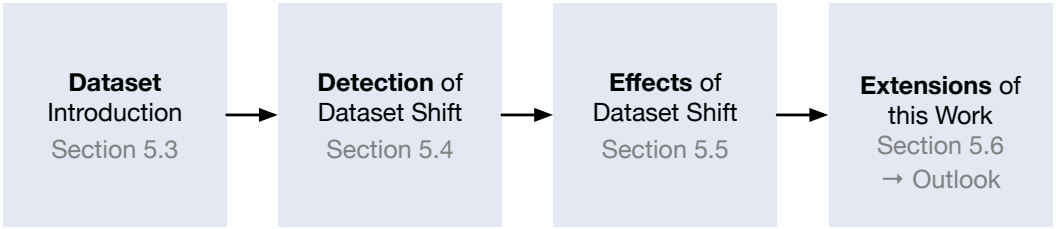


Figure 5.1: Structure of the studies in Chapter 5 addressing the detection and effects of possible dataset shift of the ALPACA dataset.

dataset is introduced in Section 5.3, which includes hyperspectral UAV data and in situ soil moisture data from five different measurement areas. These measurement areas differ, for example, in terms of altitude, microclimate, vegetation, soil moisture, and soil type. These differences can cause dataset shift. The detection of a possible dataset shift in the ALPACA dataset is presented in Section 5.4. Different kinds of dataset shift are addressed based on the distributions of the hyperspectral and the soil moisture data. Additionally, unsupervised learning approaches are applied for a more detailed look at dataset shift in the ALPACA dataset. The effects of dataset shift on the supervised regression of soil moisture, as well as the effects of sensor uncertainty, are studied and discussed in detail in Section 5.5. Finally, a summary and an outlook towards possible solutions of dataset shift are presented in Section 5.6.

The main contributions of this chapter are:

- the introduction of the novel ALPACA dataset [219, 220],
- the development of a data augmentation method to increase the dataset size and to include the sensor uncertainties,
- the detection of dataset shift in the presented ALPACA dataset with the unsupervised ML approaches, and
- a study of the effects of dataset shift on soil moisture regression result based on the presented ALPACA dataset with supervised ML models.

5.2 Related Work

The different types of dataset shift are described in detail in Section 2.5.2 [22, 63, 64, 65, 66, 67]. In the literature on ML for hyperspectral data, only a few studies exist on the topic of dataset shift. Further, most of the existing studies are performed for classification rather than for regression tasks. In the following, the most relevant studies for this chapter are summarized.

Population drift, also known as covariate shift, is addressed in [221] in the classification of land cover and land use based on Airborne Visible Infrared Imaging Spectrometer (AVIRIS) and Hyperion data. On the multitemporal dataset, the covariate shift originates from spectral variations over location and time. The limited amount of labeled training data, in combination with the covariate shift, challenges the proposed classification. In [222], sample selection bias is detected in the classification of clouds based on Medium Resolution Imaging Spectrometer (MERIS) data. The performance of the proposed semi-supervised Support Vector Machine (SVM) is limited, depending on the quality of the labels. The authors state that for more substantial sample selection biases, this approach might not be appropriate.

In [223], the land cover ground truth is available for a specific date and satellite image. Over time, the land cover can change and, therefore, differs from the available ground truth. A domain-adaptation classifier based on SVMs is proposed to evaluate the classification quality on a more recent Landsat 5 satellite image. Additionally, a circular validation strategy is proposed to evaluate the performance of the classifier. The classification is difficult if the more recent satellite image is considerably different, and the classification performance highly depends on its initialization [223]. Active learning is proposed in [218] to reduce the impact of covariate shift on the classification performance of land cover and land use. The number of labels in the given dataset is extended, queried by the active learning approach. For the presented application, this approach solves the challenge of dataset shift. Note that this approach can only be applied if it is possible to acquire new labels for the dataset [218]. Errors and uncertainties in training data of remote sensing datasets, in general, are addressed in [224, 225].

The principles of the Monte Carlo (MC) data augmentation is described in detail in Section 2.5.3. In the ML estimation based on hyperspectral data, MC data augmentation is applied, for example, in the generation of synthetic hyperspectral images [226] and ground truth [47], for hyperspectral image denoising [227] and data fusion [228].

5.3 The ALPACA Dataset

The Aerial Peruvian Andes Campaign (ALPACA) dataset is the result of a 5-day measurement campaign[1] in April 2019 in Peru. Five different measurement areas are

[1] The Peru measurement campaign was organized and conducted by Felix M. Riese, Samuel Schroers, Philipp Wagner and Julian Bocanegra as part of the German Federal Ministry of Education and Research (BMBF) *Trust* project (see, e.g., [229]).

Table 5.1: Overview of the measurement areas in Peru for Unmanned Aerial Vehicle (UAV) flights and soil moisture measurements. The measurement areas are all located in the catchment area of the river Lurín in Peru near the villages San Damian, San Andrés de Tupicocha and Chorillos. The elevation of the measurement areas is provided in meters Above Mean Sea Level (AMSL).

Area	Closest village	AMSL in m	UAV flight day, 2019	UAV area in 1000 m^2	Soil Moisture Values	Weather
A_1	S. Damian	3600	25 April	5.8	30	Cloudy
A_2	Tupicocha	3550	19 April	9.5	57	Changing
A_3	Tupicocha	3550	18 April	7.0	77	Sunny
A_4	Chorillos	2750	24 April	2.3	22	Changing
A_5	S. Damian	3650	20 April	3.0	50	Cloudy
			Overall:	27.6	236	

located in the catchment area of the Lurín River in the Andean highlands between 2700 m to 3700 m Above Mean Sea Level (AMSL) close to Lima. Data from several sensors is included in the ALPACA dataset. This chapter focuses on hyperspectral images and in situ soil moisture measurements. The relevance of the variable *soil moisture* is described in Section 2.3.1, and an overview of hyperspectral and soil moisture sensors is presented in Section 2.4. In the following, the hyperspectral data is described in Section 5.3.1, and the soil moisture data is described in Section 5.3.2. The pre-processing and the data fusion of the hyperspectral data and the soil moisture data are described in Section 5.3.3.

5.3.1 Hyperspectral Data

The *Hyperspec SWIR* sensor is a hyperspectral line scanner by Headwall Photonics (Boston, US). In Section 2.4.1, the Hyperspec SWIR sensor is compared to the other hyperspectral sensors applied in the studies within the scope of this thesis. The sensor covers the Short-Wavelength Infrared (SWIR) spectrum in a range of 900 nm to 2500 nm in 170 spectral bands. Each line of the line scanner consists of 384 pixels, and the radiometric resolution is 16 bit. In this work, the Hyperspec SWIR is mounted on a Matrice 600 Pro UAV by DJI (Shenzhen, China). The spatial resolution of the Hyperspec SWIR line scanner depends on the altitude above ground of the UAV. The Hyperspec SWIR line scanner provides hyperspectral images with a spatial resolution of 3 cm (edge length of one pixel) for the ALPACA dataset.

The sensor calibration is performed with the software *Hyperspec III* with a white reference and a dark reference. The white reference consists of a tarp with known

reflectance properties. For the dark reference, the shutter of the Hyperspec SWIR sensor is closed. This calibration is performed directly before each flight and data acquisition to calibrate the sensor in similar weather conditions as in the measurement.

The sensor data of the Hyperspec SWIR line scanner is calibrated and corrected by the software *SpectralView*. The correction starts with a radiance correction with a dark reference value taken before each flight. The spectral calibration, converting sensor counts to reflectance values, is performed based on the previously mentioned white reference. This white reference has to be located within the measurement area of the sensor during the data acquisition. Afterward, orthorectification is performed (see, e.g., in [230]). For the orthorectification, precise data about the orientation, movements, and altitude of the UAV is used as well as a Digital Elevation Model (DEM) of the measurement area. The DEM provided with SpectralView comes with a spatial resolution of about 30 m. This resolution makes the orthorectification difficult if the data is acquired in measurement areas with significant altitude differences. One example of such significant altitude differences can be seen in measurement area A_5. This correction results in a georeferenced hyperspectral image. In Figure 5.2, the hyperspectral image of the measurement area A_1 is shown.

The resulting hyperspectral images contain the reflectance spectra of the land cover in the respective measurement area. In the case of the ALPACA dataset, that includes bare soil, vegetation, and rocks. In the estimation of soil moisture, one main challenge is to extract information from the spectra that is related to soil moisture rather than soil texture, vegetation cover, and rocks. As shown in Table 5.1, the weather conditions are different for the five measurements. For the calibration of the hyperspectral line scanner to be valid, the weather conditions are supposed to be constant during one measurement (in this case: one UAV flight). The measurements at the measurement areas A_2 and A_4 are performed with changing weather conditions. These conditions add additional uncertainty to the hyperspectral data of these measurements.

5.3.2 Soil Moisture Data

In situ soil moisture measurements are taken with a handheld *ThetaProbe* [231, 232] sensor. The ThetaProbe is based on the measurement of the relative permittivity of the soil, which corresponds to the volumetric soil moisture. Per area, between 22 to 77 soil moisture measurements are available. The point measurements are acquired with various distances of 5 m to 15 m between each measurement. In

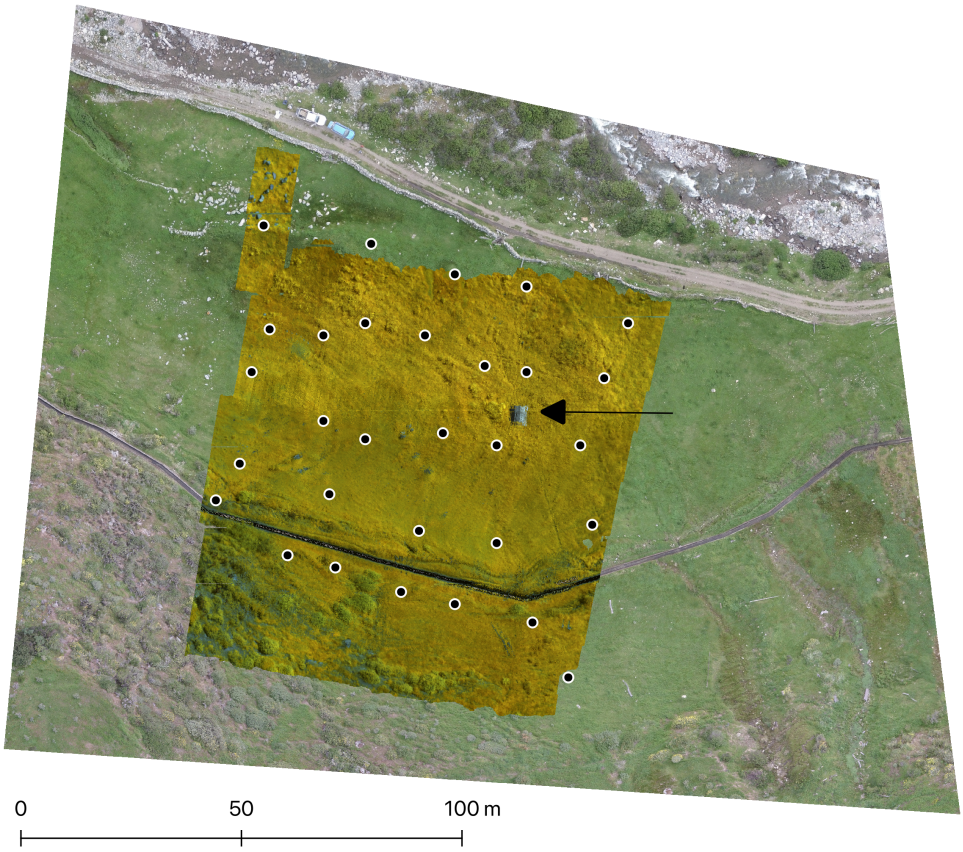


Figure 5.2: UAV images of measurement area A_1 of the ALPACA dataset. The larger background image is an RGB image, and the smaller image in the center is the hyperspectral image of the Headwall Hyperspec SWIR line scanner after calibration, illustrated in pseudo colors. The soil moisture measurements are illustrated as black dots. The horizontal line through the lower part of the hyperspectral image is an artificial water canal. Above the canal, the white reference as a 3 m $\times$ 3 m tarp divided into three different stripes is shown (black arrow).

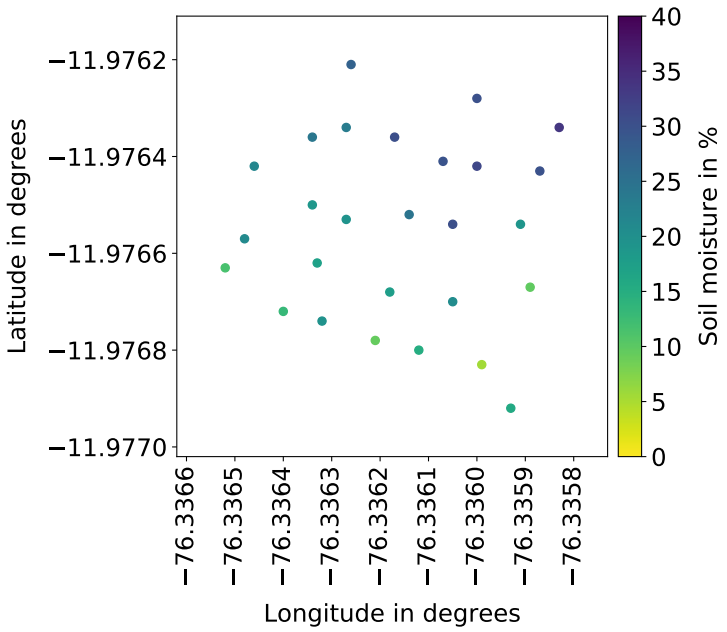


Figure 5.3: Distribution of the soil moisture point measurements of measurement area A_1 in the coordinate system World Geodetic System 1984 (WGS84). Note that only the datapoints are included in the regression study which overlap with the hyperspectral UAV image.

Figure 5.3, the spatial distribution of measured soil moisture values on measurement area A_1 is shown. The spatial soil moisture distributions of measurement areas A_2 to A_5 can be found in Figures C.1 to C.4.

The uncertainty of the sensor is stated by the manufacturer as sensor accuracy of $\pm 5\,\%$ soil moisture with the manufacturer's calibration [232]. After an (optional) manual calibration for the soil type of the respective measurement area, a sensor accuracy of $\pm 2\,\%$ can be reached [232]. Within the scope of this measurement campaign and this study, no such soil type-specific calibration is performed due to the lack of knowledge about the prevalent soil types. The fact that a calibration lowers the measurement error implies that one part of the sensor uncertainty can be removed with calibration and is, therefore, a *systematic bias*. Only if the soil type is similar for all measurement areas, this bias can be neglected in the evaluation of the hyperspectral estimation of soil moisture. Other parts of the measurement error can be of systematic or stochastic nature. That includes sensor noise, temporal and spatial soil moisture variability, uncertainties in locating the aerial coordinates of the soil moisture measurements, and orthorectification of the hyperspectral UAV images.

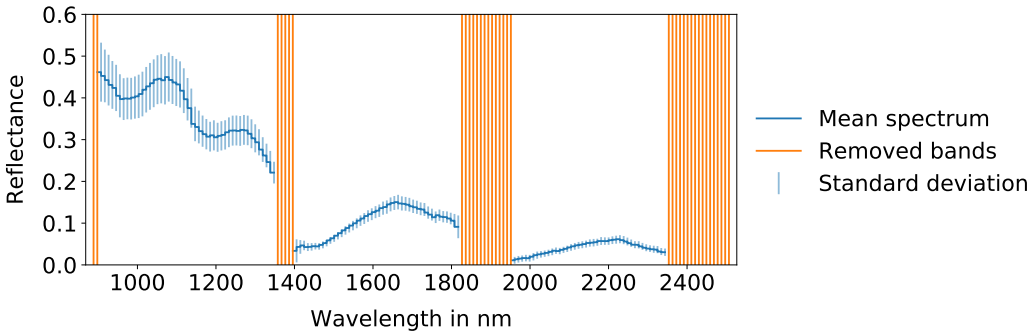


Figure 5.4: Mean spectrum of measurement area A_1 (blue curve) with standard deviation (vertical blue lines). The spectral bands 0, 1, 49 to 53, 98 to 111, and 153 to 169 (vertical orange lines) from all available bands 0 to 169 are removed in the pre-processing.

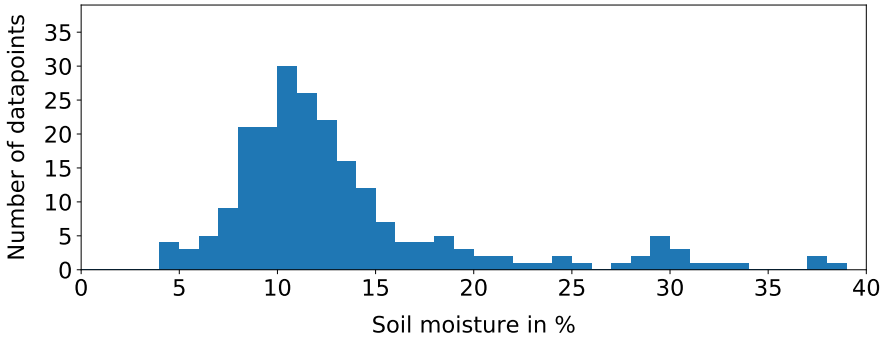


Figure 5.5: Histogram of the soil moisture data of all five measurement areas of the ALPACA dataset. Only datapoints with a soil moisture value of less than 40 % soil moisture are considered.

5.3.3 Pre-Processing and Data Fusion

The hyperspectral images from the Hyperspec SWIR line scanner are pre-processed to remove noise and incomplete data from the dataset. An overview of pre-processing, in general, is presented in Section 2.5.1. The ALPACA dataset consists of 170 hyperspectral channels with channel numbers 0 to 169. The channels with the respective numbers 0, 1, 49 to 53, 98 to 111, and 153 to 169 are ignored because they correspond to water absorption bands of the SWIR spectrum and contain significantly more noise than signal. An exemplary spectrum with the removed bands is shown in Figure 5.4. The total number of bands is reduced from 170 to 132.

In the case of the soil moisture measurements, all measurements with soil moisture of more than 40 % are ignored in the following studies. The ignored 39 datapoints are considered as outliers; the dataset size is decreased to overall 197 datapoints. This outlier removal reduces the challenge of dataset shift, as described in Section 5.4.1. Figure 5.5 gives an overview of all measured soil moisture values.

The data fusion is performed after the pre-processing. In [233], this stage is referred to as observation-level fusion. In the data fusion, the soil moisture data is merged with the hyperspectral data. The channel-wise mean spectrum of the 5×5 pixels of the hyperspectral image around each soil moisture measurement is calculated. The data fusion results in a dataset consisting of 197 datapoints, each consisting of 132 spectral bands and one soil moisture value.

5.4 Detection of Dataset Shift

In the following, the ALPACA dataset is analyzed for the detection of dataset shift. An overview of the detection of dataset shift is given in [234]. For the detection of dataset shift in this context, the five different measurement areas are considered as five different subsets of data. Therefore, dataset shift occurs in this context, if the measurement areas show significant differences in the hyperspectral and soil moisture data. The detection presented in the following is independent of solving the underlying regression task. In Section 5.4.1, the possible dataset shift of the ALPACA dataset is qualitatively addressed based on the distributions of the hyperspectral and soil moisture data. Three unsupervised ML approaches—Principal Component Analysis (PCA), Autoencoder (AE), and Self-Organizing Map (SOM)—are applied in Section 5.4.2 to detect dataset shift in the high-dimensional hyperspectral data in more detail. These approaches use the full spectral data of all measurement areas.

5.4.1 Qualitative Detection of Dataset Shift

The ALPACA dataset consists of measurements from five different measurement areas. In the following, possible occurring dataset shifts of the ALPACA dataset are addressed qualitatively. In Figure 5.6, the mean spectrum of each measurement area is shown. The spectra differ in terms of absolute values (intensity) and different characteristics (shape). More sophisticated ML approaches are applied in Section 5.4.2 to take full advantage of the high-dimensional hyperspectral data.

The histograms of the measured soil moisture values for each measurement area are shown in Figure 5.7. Only the soil moisture values of measurement area A_1 show a uniform distribution, while the other measurement areas show one single peak per distribution with few outliers. There are multiple possible reasons for these differences. For example, the different measurement areas are located in different

microclimates, and their soil textures might differ. These can be called non-stationary environments, and they result in a concept shift between each measurement area.

The ALPACA dataset can be considered a small dataset, especially when dividing it into the five different measurement areas. When splitting this dataset into a training and a test dataset (see Section 2.5.4), the test dataset might not represent the training dataset well. This issue can be considered a sample selection bias, the i.i.d. assumption can not be applied (see Section 2.5.2).

5.4.2 Unsupervised Learning Results

In Section 5.4.1, the dataset shift in the hyperspectral data of the ALPACA dataset is addressed based on the mean spectra of each measurement area. The three ML approaches PCA, AE, and unsupervised SOMs are applied to take full advantage of the high-dimensional data in the detection of dataset shift. All pixels from all measurement areas are used for the respective training. That also includes pixels without soil moisture reference data. The goal of this study is to analyze the three approaches in the context of dataset shift detection.

While the main application of PCA and AE is dimensionality reduction, the goal of this study is to analyze the dataset. An overview of these dimensionality reduction approaches is presented in Section 2.6.1 and [81]. The SOM is presented as an unsupervised, supervised, and semi-supervised approach in Chapter 3. For this study, the unsupervised SOM is applied to cluster the ALPACA dataset. The results of the three unsupervised approaches are summarized in the following and discussed in Section 5.4.4.

PCA The principal components are ordered by the variance they explain. The first two principal components calculated on this dataset include about 82 % of the dataset's variance and are shown in Figure 5.8a. If the hyperspectral data only consists of the (desired) signal, therefore does not contain any noise, its variance can be used as a measure for the signal. In this case, the signal is the target variable soil moisture. Since the hyperspectral data includes noise to some extend (see Section 5.3.1), the first two principal components are influenced by that noise as well. The measurement areas A_1, A_3, and A_4 are distributed more at the border of the value range, areas A_2 and A_5 are distributed over the full value range.

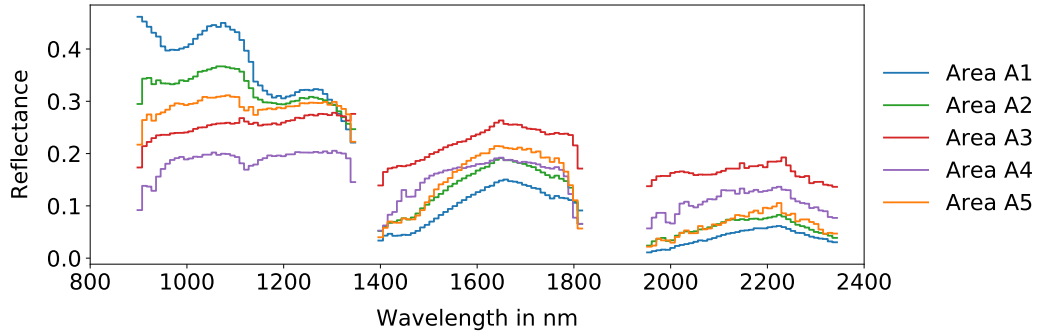


Figure 5.6: Mean spectra of the different measurement areas A_1 to A_5.

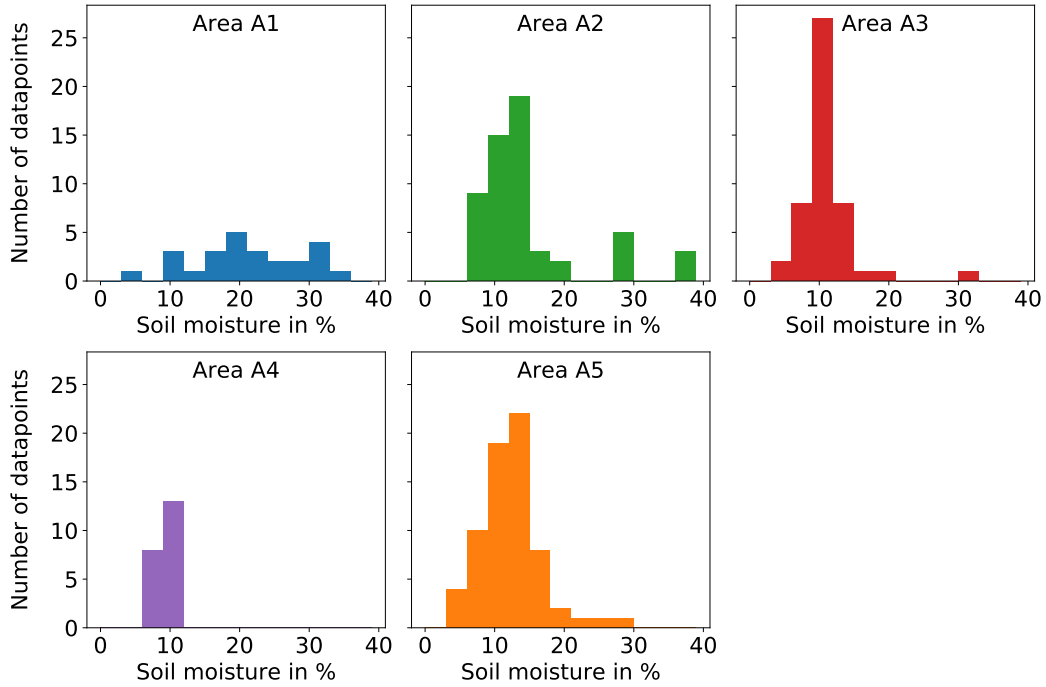


Figure 5.7: Histogram of the measured soil moisture values for each measurement area. Only datapoints with a soil moisture value of less than 40 % soil moisture are considered. The soil moisture data of all measurement areas combined is shown in Figure 5.5.

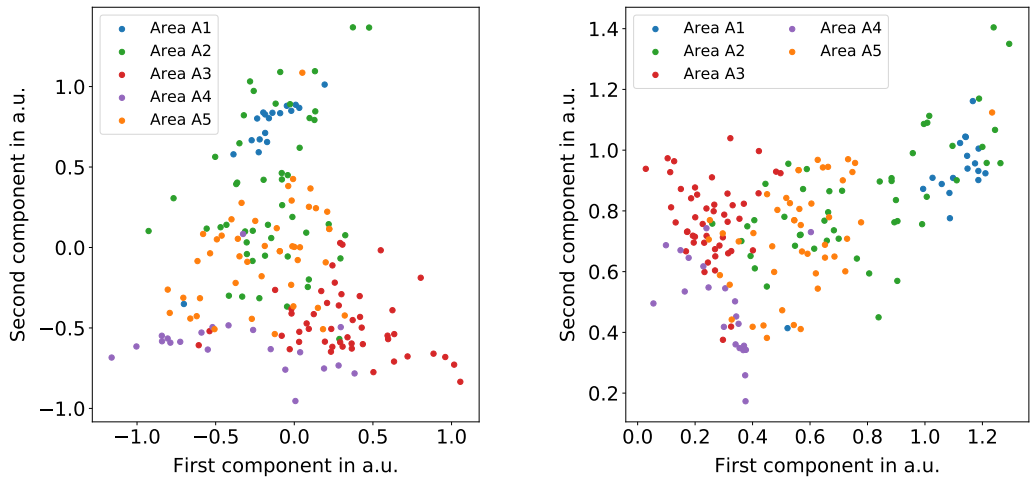


Figure 5.8: First and second components of (a) PCA and (b) AE in Arbitrary Units (a.u.).

AE The AE is set up with five fully-connected layers with {200, 100, 2, 100, 200} neurons. The architecture can be referred to as deep AE. The two components in the middle of the AE include the smallest possible representation of the training dataset. The two components are illustrated in Figure 5.8b. The distributions of the five measurement areas behave similarly as the PCA distribution in Figure 5.8a: the measurement areas A_1, A_3 and A_4 are more distinctly clustered than areas A_2 and A_5.

SOM The unsupervised SOM is set up with a grid of 40×40 neurons and 30 000 iterations for the training. The other hyperparameters are the default hyperparameters of the Supervised Self-Organizing Maps (SuSi) Python package [159, 6] (Version 1.0.8). For the evaluation of the unsupervised SOM, only the pixels with corresponding measured soil moisture value are included. In Figure 5.9, the histogram of the Best Matching Units (BMUs) for every datapoint of the dataset is shown, divided into the different measurement areas. Measurement areas A_1, A_3 and A_4 are linked to only a small number of SOM neurons while measurement areas A_2 and A_5 are distributed broadly over the SOM grid.

5.4.3 Quantification of Dataset Shift

In this section, a novel approach is presented to detect dataset shift quantitatively. This detection is based on the output of the unsupervised SOMs. As described in Section 5.4.2, Figure 5.9 illustrates the clustering of the datapoints of the ALPACA dataset with the unsupervised SOM. Assuming that no dataset shift is given, the five

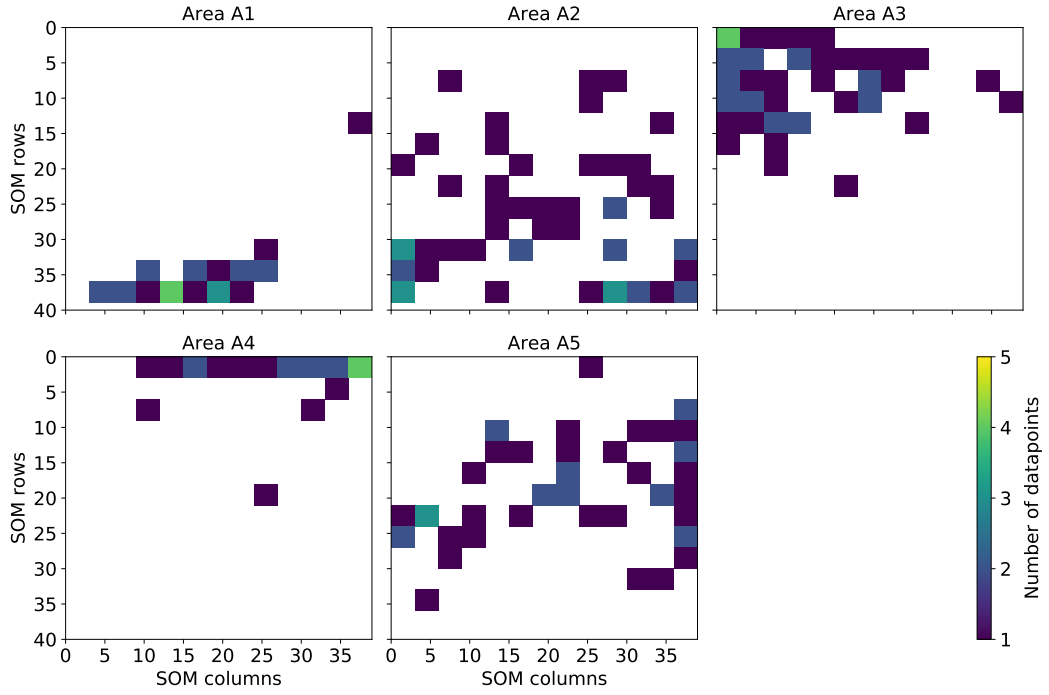


Figure 5.9: 2-dimensional (2D) histograms, including the BMUs of each datapoint per measurement area of the ALPACA dataset. The bin size of the 2D histograms corresponds to three SOM neurons on both axes.

measurement areas are expected to be clustered by (continuous) soil moisture. In the case of a dataset shift between these five different measurement areas, the dataset is clustered by those areas. The spectral separability based on the unsupervised SOM, which is a key in ML classification, can help to quantify a possible dataset shift in ML regression. In the following, we introduce two novel metrics to quantify dataset shift: the relative grid area ρ covered on a 2D grid and the relative grid overlap ξ on a 2D grid. Both metrics are based on a 2D histogram of datapoints on the SOM grid. The metrics depend on the bin size of the histogram and, therefore, on the size of the SOM grid. For example, in Figure 5.9, the bin size is 3×3 SOM neurons. For studies on smaller datasets (e.g., Figure 3.5a in Section 3.6), we recommend a bin size that covers more than one neuron for a more meaningful visualization and calculation of ρ and ξ. In studies on large datasets (e.g., Figure 3.9 in Section 3.7), the bin size can be equal to one neuron.

Relative grid area ρ The relative grid area ρ relates to the spread of the datapoints from a measurement area on the SOM grid. A larger ρ implies a smaller dependence of the SOM clustering on the different measurement areas compared to an observed variable such as soil moisture. This relation means that a larger

ρ implies a smaller degree of dataset shift. Based on a fixed bin size, the relative grid area $\rho(i)$ of measurement area A_i can be calculated as

$$\rho(i) = \frac{a_i}{a}, \tag{5.1}$$

with the number of filled bins a_i of measurement area A_i and with the number of bins a of the full 2D area.

Relative grid overlap ξ The spread of the datapoints of one measurement area over the SOM grid can be compared to the spread of other measurement areas. This comparison is the motivation for the relative grid overlap ξ. It is defined, with the number of filled bins a_i of measurement area A_i, as

$$\xi(i) = \frac{\text{Number of overlapping bins: } A_i \cap \text{remaining areas}}{a_i}. \tag{5.2}$$

A relative grid overlap ξ of 100 % implies that two measurement areas are spectrally very similar, while an overlap of 0 % means that the SOM can separate the given measurement area A_i from others well. A large relative grid overlap ξ, therefore, implies a smaller degree of dataset shift between this area A_i to the remaining dataset.

Results Table 5.2 shows the results of the quantitative analysis of the dataset shift in the ALPACA dataset. Measurement areas A_1 and A_4 show the smallest relative grid area ρ with values of 8.3 %. The maximum for ρ can be observed in measurement area A_2, with about 25.4 %. In terms of the relative overlap ξ, A_2 shows the smallest value of 11.6 %, while A_4 gives $\xi = 28.6$ %. The results of the dataset detection are discussed in the following section.

Table 5.2: Quantitative detection of dataset shift for the five measurement areas. For the comparison, the relative grid area ρ and the relative grid overlap ξ on the 2D SOM grid are applied. The values implying a larger dataset shift are highlighted in bold.

Measurement area	ρ in %	ξ in %
A_1	**8.3**	14.3
A_2	25.4	**11.6**
A_3	20.7	14.3
A_4	**8.3**	28.6
A_5	21.3	16.7

5.4.4 Discussion of the Dataset Shift Detection

Overall, all three unsupervised approaches applied in Section 5.4.2 are able to detect dataset shift in the hyperspectral data of the ALPACA dataset. The results from the dimensionality reduction approaches PCA and AE in Section 5.4.2 imply that the measurement areas A_1, A_3, and A_4 are more distinguishable on the spectral feature space than areas A_2 and A_5. The SOM can separate measurement areas A_1, A_3, and A_4 from each other. The unsupervised SOM shows a much clearer separation between the different measurement areas. This result can be interpreted in two ways. On the one hand, the SOM shows better clustering capabilities than using the first two components from PCA and AE. On the other hand, the SOM is more sensitive to the spectral differences that originate from the different measurement areas instead of the spectral differences that originate from the different soil moisture values and other influencing factors.

The datapoints of the measurement areas A_2 and A_5 are overlapping on the SOM grid. That implies that their spectra overlap as well. This result is similar to the result in Section 5.4.1. In Figure 5.6, the mean spectra of measurement areas A_2 and A_5 overlap in the spectrum of about 1400 nm to 2300 nm.

Section 5.4.3 introduces two quantitative metrics for the degree of dataset shift. The relative grid area ρ implies the most significant degrees of dataset shift for the measurement areas A_1 and A_4. The relative grid overlap ξ shows the largest degree of dataset shift for the measurement area A_2. Overall, two main findings can be summarized from the quantitative dataset shift detection. Firstly, both metrics imply a significant degree of dataset shift, since the value ranges are all below 30 %. Secondly, both metrics show different minima and should be applied together to capture the full range of dataset shift characteristics.

The findings of this section motivate to perform supervised regressions of soil moisture on the measurement areas individually. In Section 5.5, individual regressions and regressions on combinations of the measurement areas are performed. A focus lies on measurement areas A_1, A_3 and A_5, since the data of measurement areas A_2 and A_4 was acquired during changing weather conditions (see Section 5.3.1). The effects of the detected concept shift in the soil moisture data is studied by performing regressions on the measurement areas individually and in different combinations in Section 5.5. The detected sample selection bias is studied in the following by testing several random seeds for the dataset split.

5.5 Effects of Dataset Shift on Supervised Regression

In this section, supervised regression of soil moisture is performed based on the available hyperspectral data. Only a few measured soil moisture values are available per measurement area (see Table 5.1). This low number of labels is a challenge when splitting the dataset (see Section 5.4.1). Additionally, the soil moisture data is measured with an uncertainty, depending on different factors like sensor calibration, sensor noise, the soil properties of the measurement area, and the measurement process itself. This uncertainty is discussed in Section 5.3.2.

In the following, the ALPACA dataset is augmented with MC methods, as described in Section 5.5.1. The data augmentation increases the dataset size and includes the soil moisture sensor uncertainty. The regression results based on the original dataset and the MC augmented dataset are presented in Section 5.5.2 and discussed in detail in Section 5.5.3.

5.5.1 Monte Carlo Data Augmentation

In this chapter, MC data augmentation is applied to increase the size of the dataset and to study the effects of soil moisture sensor uncertainties. MC data augmentation, in general, is described in Section 2.5.3. The MC generator produces pseudo soil moisture data for existing hyperspectral pixels with corresponding measured soil moisture values. The pseudo data should incorporate uncertainties of the soil moisture measurements (see Section 5.3.2).

For this study, the datapoints from measurement area A_1, A_3, and A_5, as well as different combinations of all five measurement areas, are included. Only datapoints with soil moisture values below 40 % are considered. For each of the measured soil moisture values, ten MC datapoints are generated. The mean μ of Equation (2.1) is the measured soil moisture value itself. The standard deviation σ is set to 2 % based on the measurement uncertainty of the applied soil moisture sensor (see Section 5.3.2). The 2 % only include the stochastic uncertainty, ignoring the systematic error. This is a conservative assumption and can be modified in future studies. The distributions of the measured soil moisture data and the MC pseudo data are shown for measurement area A_1 in Figure 5.10 (see also Figures C.5 and C.6 for measurement areas A_3 and A_5). The pseudo soil moisture data described in this subsection is used in Section 5.5.2 for the supervised estimation of soil moisture.

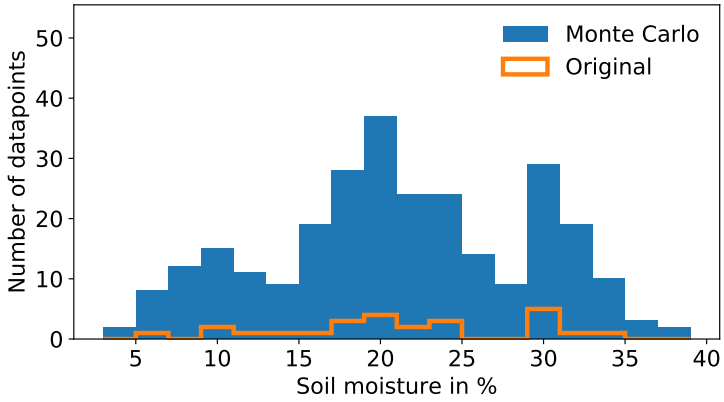


Figure 5.10: MC histogram of measurement area A_1. Only the 25 datapoints of area A_1 are included with a soil moisture value below 40 %. Ten MC pseudo datapoints are generated for every datapoint with a standard deviation of 2 %.

5.5.2 Regression Models and Results

For the regression of soil moisture in this chapter, data of the five measurement areas is applied in different combinations: every measurement area individually, areas A_1 and A_3 ($A_{1,3}$), areas A_1 and A_5 ($A_{1,5}$), areas A_1, A_3 and A_5 ($A_{1,3,5}$) as well as all measurement areas together. Since only datapoints with less than 40 % soil moisture are considered, the total number of soil moisture values per measurement area decreases: A_1 consists of 25 soil moisture values, A_3 consists of 48 soil moisture values, and A_5 consists of 47 soil moisture values. The combination of all measurement areas together consists of 197 soil moisture values. For every combination, the original dataset and the respective dataset with additional ten MC pseudo datapoints per measured datapoint are studied.

The regression is performed with two regressors: a Random Forest (RF) regressor (see Section 2.7.1) and a supervised SOM based on the SuSi framework (see Section 3.4.1). As a standard approach in hyperspectral remote sensing, a RF regressor is used with 100 trees (see Section 2.7.1). All other hyperparameters of the RF are the default hyperparameters implemented in the Python package *scikit-learn* [131] (Version 0.22.1). The SOM is applied with a grid of 10×10 neurons. The unsupervised part of the SOM is trained with 2000 iterations and the supervised part of the SOM with 1000 iterations. All other SOM hyperparameters are the default hyperparameters implemented in the *SuSi* package [159, 6] (Version 1.0.8). Further hyperparameter optimizations are possible in future studies. For the subsequent studies, no normalization, feature engineering, or dimensionality reduction is applied.

The dataset of the measured soil moisture values is split into a training subset and a test subset. The effect of the test subset size is studied by applying four different test subset sizes from 15 % to 30 %. The training subset size is defined as 100 % minus the test subset size. For the MC dataset, the MC pseudo data is added to the dataset after the split. For every original datapoint, all MC datapoints are added into the same subset. That way, one hyperspectral spectrum is only part of either the training or the test subset. Several parts of the estimation study depend on randomization: the generation of the MC pseudo data, the dataset split, and the training of RF and SOM. The random seeds of all randomized parts are fixed and can be controlled to study different combinations. Every estimation is performed with 40 different random seeds. Overall, every regression experiment is performed $4 \cdot 40 = 160$ times.

For the evaluation in this regression study, the Coefficient of Determination (R^2), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE) are considered as evaluation metrics, which are defined in Section 2.7.6. The results are listed in Table 5.3 as median values of the evaluation metrics of all different random seeds and test subset sizes. The histograms of the R^2, MAE, and RMSE distributions as a measure for the regression performance are shown for the combination of measurement areas $A_{1,3}$ in Figure 5.11 and the other six combinations in Figures C.7 to C.12. An exemplary estimation result of the SOM is shown in Figure 5.12. The measured soil moisture datapoints are emphasized within the MC datapoints.

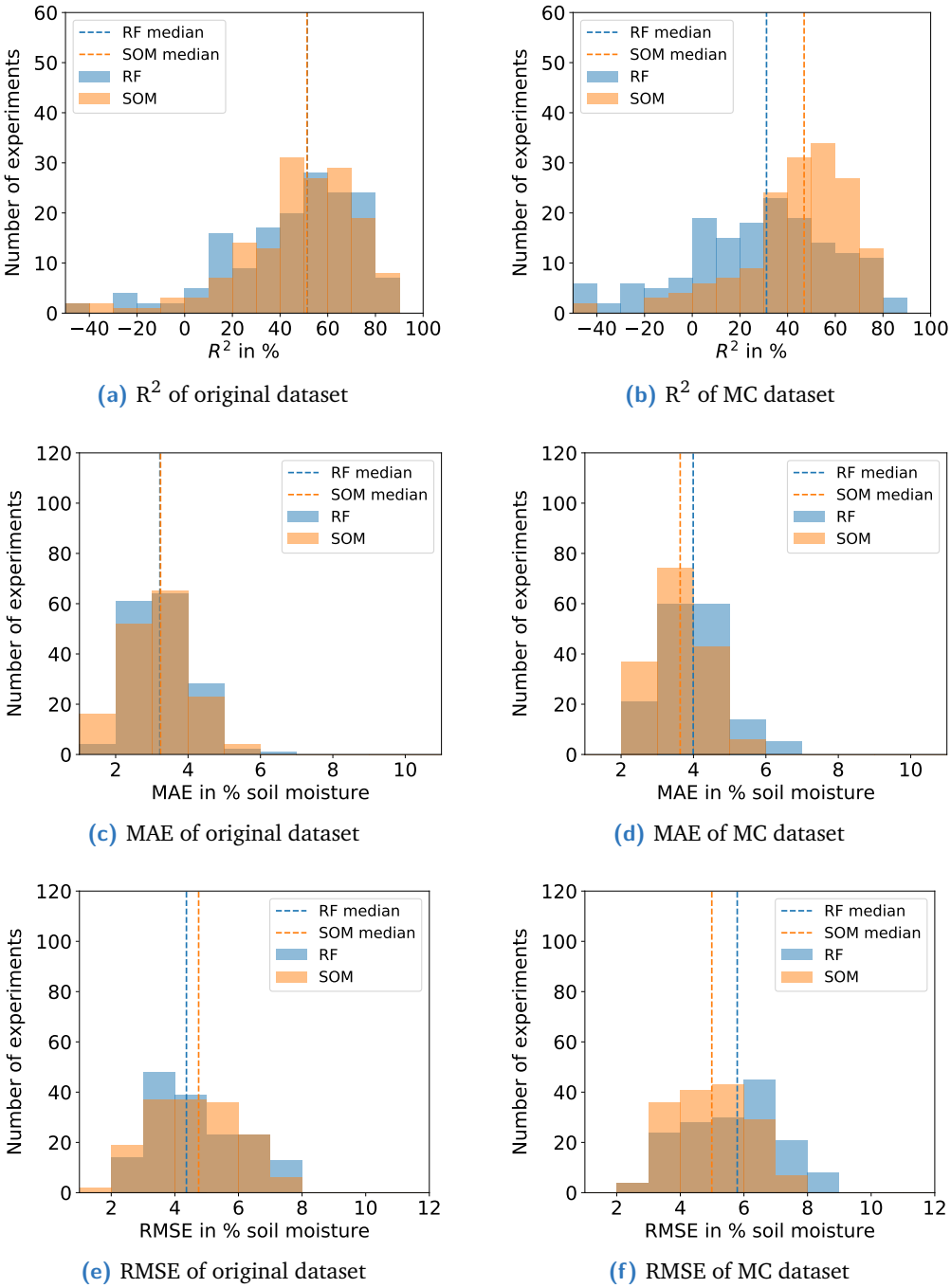
Exemplary SOM output grids are shown in Figure 5.13 for a SOM trained on the original soil moisture dataset, and a SOM trained on the MC augmented dataset. The combination of measurement areas $A_{1,3}$ is used for the training of these SOMs. They provide a continuous soil moisture map with local minima and maxima. The SOM, which is trained on the original augmented dataset, covers a broader range of soil moisture. The same two exemplary regression SOMs are applied for the soil moisture estimation on the full hyperspectral image of measurement area A_1 in Figure 5.14. The SOM, which is trained on the MC augmented dataset, shows a smoother estimation map than the SOM, which is trained on the original dataset.

5.5.3 Discussion

Overall, the regression performances of RF and SOM lead to median R^2 values with at best 51.4 %. That implies that the given regression task is difficult for both regression approaches. This challenge is expected because of the small dataset size and the effect of dataset shift as described in Section 5.4.4.

Table 5.3: Median R^2, MAE and RMSE for the regression models SOM and RF based on the original datasets and the MC augmented datasets of the measurement areas A_1, A_3 and A_5 individually and in the different combinations $A_{1,3}$, $A_{1,5}$, $A_{1,3,5}$, and all areas.

Area(s)	Dataset	R^2 in %		MAE in % soil moisture		RMSE in % soil moisture	
		RF	SOM	RF	SOM	RF	SOM
A_1	Original	17.9	30.0	5.5	4.8	6.2	5.9
	MC	−20.0	23.8	6.4	5.1	7.8	6.3
A_3	Original	−37.1	−10.6	**2.5**	**1.9**	**3.3**	**2.6**
	MC	−32.5	−9.0	3.1	2.6	4.1	3.3
A_5	Original	−19.8	−17.8	2.9	2.8	3.6	3.6
	MC	−23.6	−10.0	3.5	3.2	4.3	4.1
$A_{1,3}$	Original	**51.4**	**51.3**	3.2	3.2	4.4	4.7
	MC	31.1	46.9	4.0	3.6	5.8	5.0
$A_{1,5}$	Original	32.9	31.7	3.9	4.0	5.1	5.3
	MC	17.8	29.7	4.6	4.3	5.8	5.6
$A_{1,3,5}$	Original	38.0	39.5	3.3	3.3	4.7	4.8
	MC	26.3	34.8	3.9	3.8	5.3	5.2
All	Original	29.3	23.1	3.7	3.9	5.6	5.8
	MC	17.6	21.1	4.2	4.2	6.1	6.1



(a) R^2 of original dataset

(b) R^2 of MC dataset

(c) MAE of original dataset

(d) MAE of MC dataset

(e) RMSE of original dataset

(f) RMSE of MC dataset

Figure 5.11: Estimation results of the supervised regressors RF and SOM based on the data of the combination of measurement areas $A_{1,3}$. The results are illustrated as histograms of the three evaluation metrics R^2, MAE and RMSE with and without MC augmented data. (a) R^2 of the original dataset and (b) R^2 with MC data augmentation. The left-most bin is an overflow bin, collecting all values $R^2 < -50\,\%$. (c) MAE of the original dataset, (d) MAE with MC data augmentation, (e) RMSE of the original dataset, and (f) RMSE with MC data augmentation.

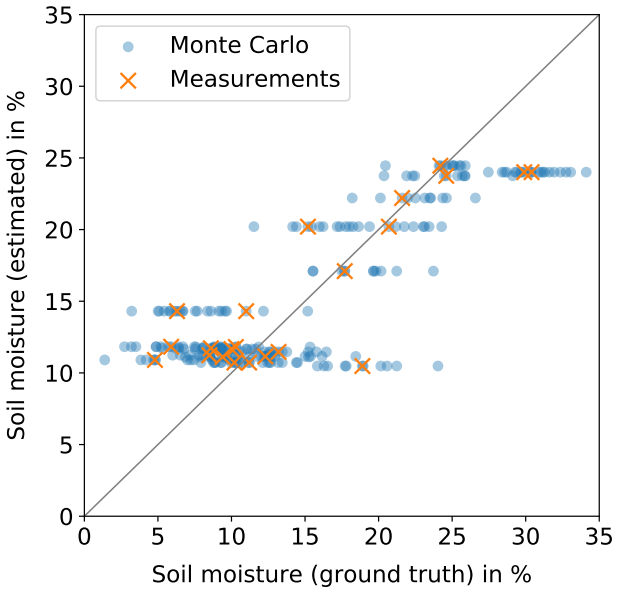


Figure 5.12: Estimated soil moisture over measured soil moisture values based on the MC augmented soil moisture dataset of measurement areas $A_{1,3}$. The plot shows a discretization of estimated soil moisture values, because the underlying hyperspectral input data is not augmented. The estimation in this exemplary plot results in $R^2 = 66.1\,\%$. The diagonal grey line through the origin marks the ideal estimation outcome.

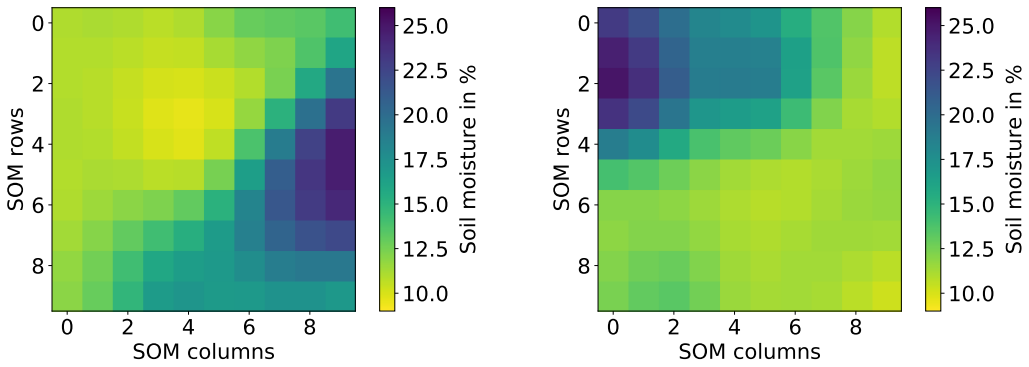
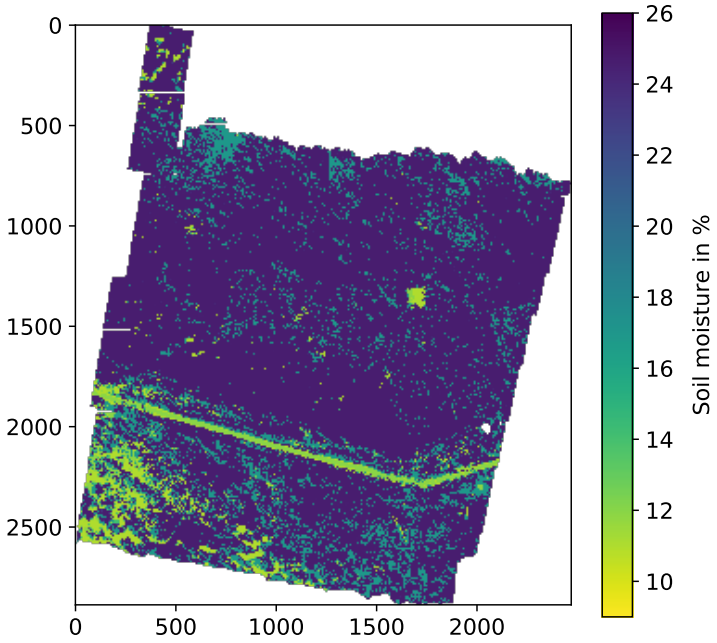
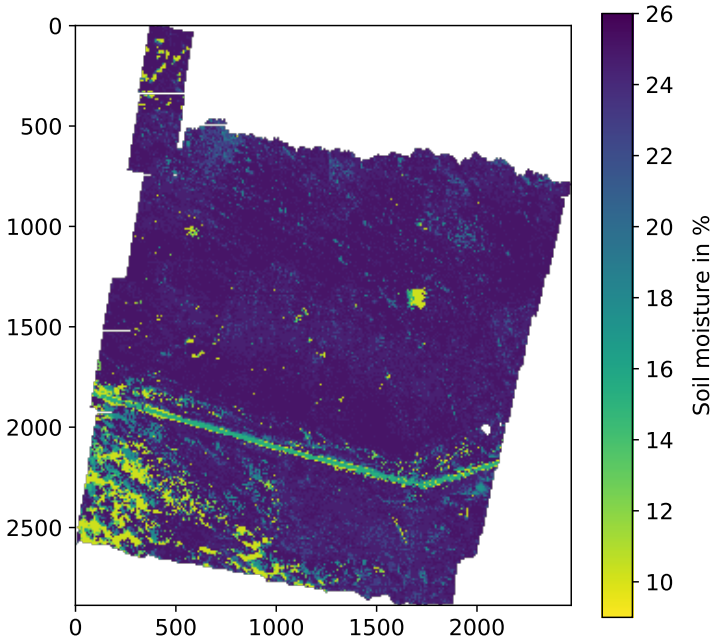


Figure 5.13: Exemplary SOM output grids with 10×10 neurons of the two supervised SOM regressors trained on (a) the original dataset consisting of the measured soil moisture values and (b) the MC augmented dataset of the combination of measurement areas $A_{1,3}$. The exact distribution of the soil moisture values on the SOM output grids depends on the random initialization of each SOM. Therefore, only qualitative differences between the different output grids can be considered.



(a) Trained on original dataset



(b) Trained on MC dataset

Figure 5.14: Exemplary SOM estimation maps of measurement area A_1 by the two supervised SOM regressors trained on (a) the original dataset consisting of the measured soil moisture values and (b) the MC augmented dataset of the combination of measurement areas $A_{1,3}$. The horizontal canal in the lower third of the map and the white reference in the upper right part of the map needs to be ignored for the evaluation.

The regression, based on measurement area A_1 individually, leads to significantly better results than the regression, based on the individual measurement areas A_3 and A_5. The uniform soil moisture distribution of measurement area A_1 (see Figure 5.7) and the constant weather conditions are both positively impacting the regression.

The performance on the individual measurement areas A_3 and A_5 results in negative values of the median R^2, which implies that an estimation is not possible. The median MAE is between 1.9 % to 3.5 % soil moisture, which is rather low compared to the other results. Also, these values for the median MAE are in a similar range as the soil moisture sensor uncertainty of ± 2 % (see Section 5.3.2). Since measurement area A_3 and A_5 show a sharp peak in their soil moisture distributions (see Figure 5.7) and a much broader range of soil moisture values (0 % to 40 % soil moisture) is considered, outliers in the soil moisture distributions have a significant impact on the respective values of R^2. Therefore, the median MAE is the more meaningful evaluation metric for the individual estimations based on measurement areas A_3 and A_5 in this study. The dataset size of each of both measurement areas A_3 and A_5 is nearly twice as large as the dataset size of measurement area A_1. The regression results imply that the dataset size is not the most important factor for the regression quality.

Combining the three measurement areas A_1, A_3 and A_5 to $A_{1,3}$, $A_{1,5}$ and $A_{1,3,5}$ leads to increased regression performance. The combination $A_{1,3}$ shows the best overall results with a median R^2 of about 51 % for RF and SOM. As shown in Section 5.4, measurement areas A_1 and A_3 can be differentiated well based on their spectra. The distribution of the datapoints of measurement area A_5, for example, based on unsupervised AE or SOM, is much broader than the distributions of A_1 or A_3. That implies that the combination of A_1 and A_3 (to $A_{1,3}$) adds more additional information to the regression task than the combination of A_1 and A_5. The combination of all five measurement areas shows a lower regression performance with values for the median R^2 from 23.1 % to 29.3 % for the original dataset. Since the regression performance from $A_{1,3}$ to $A_{1,3,5}$ decreased, the performance drop from $A_{1,3,5}$ to all areas is expected.

Overall, the two regression approaches RF and SOM show similar regression performances on the original dataset. There are two exceptions to this finding. On measurement area A_1, the SOM performs significantly better than the RF regressor, while the RF outperforms the SOM on the combination of all measurement areas. This finding implies that the SOM is able to learn better from smaller datasets. Further, the SOM might benefit from more intensive hyperparameter tuning on the heterogeneous combination of all areas, while the RF shows better out-of-the-box performance.

Comparing the regressions of the original datasets and the MC augmented datasets, the performances drop. The MC augmentation adds noise to the datasets, which does not include any new information except the uncertainty of the original soil moisture measurements. The performance drops of the RF regressor from 10 % to 20 % are more significant than the performance drops of the SOM from 2 % to 6 %. This finding implies that the SOM is more robust against noise in the soil moisture data. The SOM firstly clusters the data based on the hyperspectral data, which is not augmented in this study. Datapoints with similar spectra are clustered on a similar region on the SOM grid, which leads to similar soil moisture values on the supervised SOM grid (see Section 3.4.1).

The measurement accuracy of the measured soil moisture data is described in Section 5.3.2. The estimation errors in Table 5.3 of median MAE values for the individual measurement areas A_3 and A_5 in a range of 1.9 % to 3.5 % soil moisture are, therefore, within a possible measurement error without the systematic bias. The median MAE values for the combinations of measurement areas $A_{1,3}$, $A_{1,5}$, $A_{1,3,5}$, and all areas are within the sensor uncertainty, including the systematic bias, meaning 5 %.

The SOM estimation maps of measurement area A_1 in Figure 5.14 are trained on the original, and the MC augmented datasets of the best performing combination $A_{1,3}$. Overall, both estimation maps show similar distributions. The estimation map trained on the MC augmented dataset shows a significantly smoother distribution of soil moisture. One reason for this result is the distribution of the SOM output grid. For the generation of the estimation map, every pixel of the corresponding hyperspectral image is assigned with a soil moisture value from the corresponding SOM output grids in Figure 5.13. While the distributions of the SOM output grids depend on randomization, only qualitative differences can be discussed. The SOM output grid trained on the original dataset shows a soil moisture minimum in the center of the grid with only a few neurons distance to the soil moisture maximum on the grid. In contrast, the SOM output grid trained on the MC augmented dataset shows a smooth distribution with minimum and maximum in opposite corners of the output grid.

5.6 Conclusions and Outlook

This section concludes the studies presented in this chapter and gives an outlook of possible future dataset shift studies based on the ALPACA dataset.

Conclusions This chapter introduces the ALPACA dataset [219] in Section 5.3. It consists of hyperspectral images and soil moisture data of five different measurement areas in Peru. In Section 5.4, dataset shift is detected based on a qualitative approach and unsupervised learning. The qualitative approach is based on the soil moisture distributions and the mean spectra of the five measurement areas. Three unsupervised approaches—PCA, AE, and SOM—are applied to the hyperspectral data of all five measurement areas to evaluate the dataset shift findings. Two novel metrics for the quantitative detection of dataset shift are introduced, based on the unsupervised SOM: the relative grid area ρ and the relative grid overlap ξ. Concept drift in the soil moisture data, as well as sample selection bias, are detected, originating from the dataset size.

In Section 5.5, the effects of the dataset shift on the regression of soil moisture with the ALPACA dataset is evaluated. The regression is performed on three measurement areas individually and different combinations of the measurement areas. The standard approach RF is applied as well as a supervised SOM. Both show comparable results. The SOM outperforms the RF on the smallest dataset, measurement area A_1. Additionally, MC data augmentation is applied on these datasets. The data augmentation increases the dataset size and includes uncertainty on the soil moisture sensors. As expected, the regression performance drops if the datasets are MC augmented. While the performance of the RF decreases significantly, the SOM performance only decreases by a few percents. The results imply that the SOM is more robust against noise in the soil moisture measurements. Further, the added noise from the MC data augmentation smoothens out the area-wise estimation of soil moisture.

Outlook The soil moisture regression based on the ALPACA dataset presented in Section 5.5 can be combined with the dimensionality reduction approaches PCA and AE. Note that the unsupervised SOM is included in the supervised SOM. Additionally, the unsupervised approaches t-Distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP) can be applied for the dimensionality reduction, as described in Section 2.6.1. The reduced dimensionality can lead to an increased regression performance, depending on the dataset and the supervised ML model.

The novel dataset shift metrics ρ and ξ can be combined with other approaches to one dataset shift metric [234]. In future work, this metric can be applied to different datasets and regression tasks, reducing the time and effort to detect dataset shift. Additionally, the metrics can be extended for ML classification tasks, which are the main application in hyperspectral remote sensing.

The ALPACA dataset includes significantly more hyperspectral pixels than in situ soil moisture measurements. In the studies presented in this chapter, only the pixels around the soil moisture measurements are considered for the training of the ML models. In a future study, Semi-Supervised Learning (SSL) can be applied to make use of all available data of the ALPACA dataset. As described in Section 5.2, other studies perform SSL and active learning to resolve dataset shift [222, 218]. A promising approach is the semi-supervised SOM presented in Section 3.5 and applied in Section 3.6.3 in the semi-supervised regression of soil moisture on the Karlsruhe Lysimeter (KarLy) dataset is a promising candidate. In a future study, this semi-supervised SOM can be applied to the ALPACA dataset to minimize the effect of dataset shift.

Another way to extend the presented studies is to apply MC data augmentation on the hyperspectral data. Uncertainties originating from the hyperspectral sensor itself and georeferencing have to be modeled to augment hyperspectral data. In future work, MC augmented hyperspectral data can be included similarly as the MC augmented soil moisture data to study the influence of sensor uncertainty on the estimation result.

6 Conclusions and Outlook

> *All innovation begins with vision. It's what happens next that is critical.*
>
> **— Eric Ries**
> (Entrepreneur, Author of "The Lean Startup")

6.1 Conclusions

The studies presented in this thesis address three main challenges in the research field of hyperspectral remote sensing (see Section 1.2). The following subsections summarize the main contributions and results of these studies and frame them in the presented hyperspectral estimation framework.

6.1.1 Conclusions on Challenge (I)

The first challenge of Machine Learning (ML) model training on datasets with only a few labeled datapoints is the main focus of Chapter 3. The Supervised Self-Organizing Maps (SuSi) framework is presented, consisting of unsupervised, supervised, and semi-supervised Self-Organizing Maps (SOMs). There is a lack of literature in today's ML research regarding SOMs, especially in the context of hyperspectral remote sensing. Chapter 3 fills this gap by showing the potential of SOMs and providing researchers with the framework to apply SOMs in their studies. In the following, this chapter is concluded by summarizing the four main contributions.

KarLy Dataset Two datasets are used to study the ability of the SuSi framework in the supervised estimation. For the exemplary regression of soil moisture based on hyperspectral data, we published the Karlsruhe Lysimeter (KarLy) dataset in [42]. For the exemplary classification of land cover, the available Salinas dataset is applied [189].

SuSi Framework The SuSi framework is presented with its mathematical foundation of the unsupervised SOM and (semi-)supervised extensions in Sections 3.3 to 3.5. Compared to other SOM frameworks, the SuSi framework is the most powerful and easy-to-use framework (see Table 3.1). The implementation of the SuSi framework is made freely available as a Python package [159]. In the regression and classification studies, the performances of the SOM of the SuSi framework are compared with the performances of the standard approach Random Forest (RF). All results are given in Table 3.2; the regression and classification studies are summarized in the following.

(Semi-)Supervised Regression Based on the KarLy dataset, a supervised and a semi-supervised regression are performed (see Section 3.6). For the semi-supervised case, the labels of 90 % of the training datapoints are manually removed. In both the supervised and semi-supervised regression, the SOMs outperform the RF regressor. Further, the SOMs show significantly smaller differences between the performance on the training and the test subset. That implies that the SOM is less prone to overtraining in the presented studies. Another finding is that the SOM performs well on the KarLy dataset even if only 10 % of the datapoints are labeled. Overall, the developed SuSi framework shows a strong performance in the supervised and, especially, in the semi-supervised regression.

(Semi-)Supervised Classification The SuSi framework's performance in supervised and semi-supervised classification is evaluated based on the Salinas land cover dataset (see Section 3.7). In the supervised classification, the RF outperforms the presented classification SOM. For the semi-supervised classification, only two labels per class (0.06 %) are included in the training dataset. Both classifiers show satisfying performances. One major finding of the presented classification studies is the visualization capability of the SuSi framework. With the unsupervised SOM included in the supervised SOM estimator, the clustering of the training dataset can be used for the visualization of the dataset (see Figure 3.9). That enables, for example, to compare different classes and their spectral separability. The visualization module of the unsupervised SOM is used, for example, in Section 5.4.2.

6.1.2 Conclusions on Challenge (II)

Chapter 4 addresses the second challenge, which is the limited potential of shallow ML approaches, such as RF, on hyperspectral data. Deep Convolutional Neural Networks (CNNs) solve this challenge by learning new features based on the training dataset. In the field of hyperspectral remote sensing, CNNs are mostly applied on

2-dimensional (2D) images rather than on 1-dimensional (1D) spectra. The studies presented in this chapter fill this gap in hyperspectral remote sensing research for the example of the soil texture classification with 1D CNNs. The 1D CNNs are evaluated based on the Land Use/Cover Area Frame Survey (LUCAS) soil dataset. In the following, the two main contributions of Chapter 4 are summarized.

Innovative CNN Architectures Three innovative 1D CNN architectures are introduced within this study. They are inspired by the LeNet5 network [215], which is successfully applied in ML research. The LucasCNN consists of four Convolutional (CONV) layers, each followed by a max-pooling layer. Two Fully-Connected (FC) layers are put after the CONV layers with a softmax activation at the end of the network. The LucasResNet extends the LucasCNN architecture by an identity block, bypassing the four CONV layers. For the LucasCoordConv, the LucasCNN is extended by a CoordConv layer [216]. This layer includes the input feature coordinates in the network, in this case, the band numbers. The implementations of these three 1D CNN architectures are published in [194].

LUCAS Soil Texture Classification The three introduced CNN architectures are evaluated in the classification of soil texture on the LUCAS dataset. Their performance is compared with the performance of two existing approaches [204, 121] and a shallow RF classifier. The RF performs worst of all six approaches. An existing CNN approach [204] shows the best overall performance, closely followed by the introduced LucasCNN, LucasCoordConv, and LucasResNet. This existing approach is the most basic one, with only one CONV and one FC layer. The second existing CNN [121] performs significantly worse than the other approaches, which implies the importance of FC layers at the end of the network. Overall, the findings of the 1D CNNs are promising. A simple CNN architecture outperforms the more complex architectures, and FC layers improve the performance.

6.1.3 Conclusions on Challenge (III)

The third challenge, differences between the distributions of training and test dataset, is addressed in Chapter 5. These differences are referred to as dataset shift (defined in Section 2.5.2). Dataset shift is a common challenge in ML research and hyperspectral remote sensing, but it is mostly ignored. The studies presented in this chapter address this gap. In the following, the four main contributions of Chapter 5 are summarized.

ALPACA Dataset The Aerial Peruvian Andes Campaign (ALPACA) dataset was acquired during a field campaign in Peru in April 2019. It consists of hyperspectral Unmanned Aerial Vehicle (UAV) data of five measurement areas and soil moisture point measurements for each area. The dataset is published in [219]. Based on this dataset, the studies for the detection and the effects of dataset shift are performed.

Detection of Dataset Shift Differences between the five measurement areas are visible in terms of their soil moisture distributions and their mean spectra (see Figures 5.6 and 5.7). Concept drift and sample selection bias are detected between the five measurement areas. Further, three unsupervised ML approaches analyze the hyperspectral data of the ALPACA dataset: Principal Component Analysis, Autoencoder, and unsupervised SOM. Based on the unsupervised SOM, two quantitative metrics for dataset shift are developed: the relative grid area ρ and the relative grid overlap ρ. The two main findings are: (1) the SOM can separate the five measurement areas solely based on their hyperspectral data best, and (2) the results from the quantitative analysis directly imply significant dataset shift.

MC Data Augmentation The size of the ALPACA dataset leads to the previously mentioned sample selection bias between the measurement areas. To increase the dataset size and to include information about the soil moisture sensor uncertainty, the dataset is augmented with Monte Carlo (MC) methods. The influence of dataset size and sensor uncertainty is studied in a soil moisture regression, as described in the following.

Effects of Dataset Shift A supervised regression is performed to study the effects of dataset shift and the soil moisture sensor uncertainty. Because of its success in Section 3.6, a supervised SOM of the SuSi framework is applied and compared with a RF regressor. Different combinations of the five measurement areas are studied as well as the MC data augmentation. The SOM and the RF show similar performances. Overall, the main findings are (1) the SOM is more robust against sensor uncertainty, (2) the SOM performs better on smaller datasets while the RF shows the best performance on the dataset of all areas combined, and (3) the MC data augmentation makes the estimation on a full hyperspectral image significantly smoother.

6.1.4 Framing the Three Challenges

The three main challenges and the presented corresponding studies are framed in Figure 6.1 within the presented hyperspectral estimation framework (see Chapter 2). The framework consists of four levels: sensor, data, feature, and model level. Each of the Chapters 3 to 5 has a different focus regarding the four levels of the framework. On the sensor level, the datasets and the included sensors are heterogeneous (see

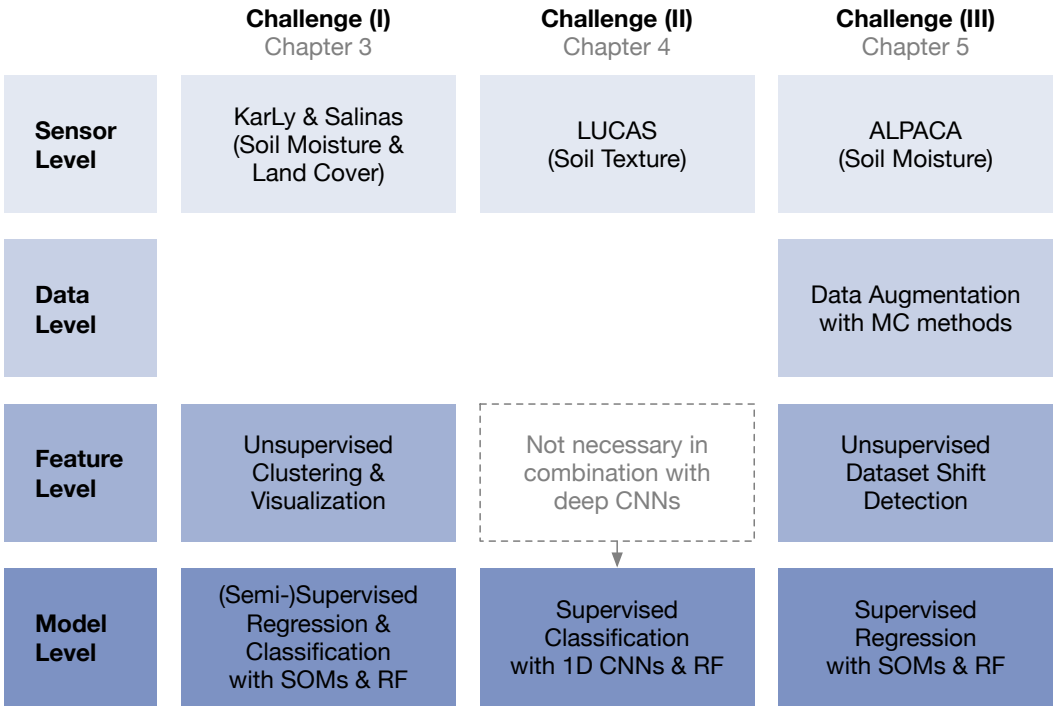


Figure 6.1: The studies in Chapters 3 to 5 within hyperspectral estimation framework, presented in Chapter 2. The framework consists of four levels: the sensor level, the data level, the feature level, and the model level.

Section 2.4 and Figure 2.4). The datasets vary, for example, in terms of their size, the number of labels, hyperspectral sensors, and location. On the data level, all studies include pre-processing and dataset splitting. Further, Chapter 5 presents a data augmentation method and addresses the qualitative detection of dataset shift. On the feature level, no feature engineering or feature selection is performed in this thesis. Unsupervised approaches for the clustering, visualization, and dimensionality reduction are applied in Chapters 3 and 5. In contrast, the feature level is not necessary for deep learning approaches, as presented in Chapter 4. On model level, different supervised and semi-supervised estimations are performed with shallow and deep ML approaches based on hyperspectral data. In every study, the performance of the applied ML models is compared with the performance of a RF estimator. SOMs are applied in (semi-)supervised estimations presented in Chapter 3 in the regression of soil moisture and the classification of land cover. Chapter 4 presents the supervised classification of soil texture. The supervised regression of soil moisture is presented in Chapter 5.

6.2 Enhancements and Outlook

This final section addresses the possible enhancements of the presented studies within this thesis and proposes future research.

Extension of the SuSi Framework The presented SuSi framework has excellent potential for further enhancements. Notably, the implementation of the (semi-)supervised classification SOM can be further improved, as presented in Section 3.7. Instead of the applied *online* mode, the *batch* mode can be used and further developed (see Section 3.3.5). The SuSi Python implementation is easily extensible because it is freely available and can be extended by the whole community [159].

Physical Explainability ML approaches, such as deep Artificial Neural Networks and CNNs, are successfully applied in hyperspectral remote sensing (see Section 2.7.2). These models are considered *black-box* models, which can not be easily interpreted physically [235]. The SuSi framework is a promising approach that includes methods for the interpretation of the model. For example, the powerful visualization abilities of the SuSi Framework are applied in Sections 3.7.2 and 5.4. In future studies, these additional findings of the SuSi can be further enhanced and used for the explainability of physical processes.

Synthesis of Chapters 3 to 5 on the ALPACA dataset The ALPACA dataset (Chapter 5) includes only a few labels for each of its five measurement areas. While the detection and effects of dataset shift are addressed in detail, the results of this chapter can be combined with the presented methods and findings of Chapters 3 and 4. Semi-supervised regression with the SuSi framework (Chapter 3) has great potential on the ALPACA dataset and can be applied in future studies.

Additionally, the 1D CNNs (Chapter 4) can be applied to the high-dimensional hyperspectral data of the ALPACA dataset. The input dimension of the CNN architectures has to be changed from 256 to 132, and the hyperparameters need to be optimized. Besides, the hyperspectral images of the ALPACA dataset can be used to develop 3-dimensional CNNs with the findings on 1D CNNs of Chapter 4 and successfully applied 2D CNNs (e.g., [69]).

The two mentioned master theses [50, 51] show the potential of Sentinel-2 satellite data (see Section 2.8). In future studies, the developed methods from Chapters 3 to 5 can be refined and applied to hyperspectral satellite data. Further, satellite data can be included in the ALPACA dataset. The satellite data can be used with the measured soil moisture data as a training dataset for a supervised model. Additional studies are possible to enhance the satellite data with the hyperspectral UAV data.

Bibliography

[1] Lindsay T. Sharpe, Andrew Stockman, Herbert Jägle, and Jeremy Nathans. “Opsin genes, cone photopigments, color vision, and color blindness”. In: *Color vision: From genes to perception* (1999), pp. 3–51 (cit. on p. 1).

[2] James B. Campbell and Randolph H. Wynne. *Introduction to remote sensing*. 5th ed. New York: Guilford Press, 2011 (cit. on pp. 1, 2).

[3] Dietrich Hofmann. “Messung von optischen, lichttechnischen Größen”. In: *Handbuch Meßtechnik und Qualitätssicherung*. Springer, 1983, pp. 430–454 (cit. on p. 1).

[4] Paul M. Treitz and Philip J. Howarth. “Hyperspectral remote sensing for estimating biophysical parameters of forest ecosystems”. In: *Progress in Physical Geography: Earth and Environment* 23.3 (1999), pp. 359–390 (cit. on pp. 1, 10).

[5] Iftikhar Ali, Felix Greifeneder, Jelena Stamenkovic, Maxim Neumann, and Claudia Notarnicola. “Review of Machine Learning Approaches for Biomass and Soil Moisture Retrievals from Remote Sensing Data”. In: *Remote Sensing* 7.12 (2015), pp. 16398–16421 (cit. on pp. 1, 10).

[6] Felix M. Riese, Sina Keller, and Stefan Hinz. “Supervised and Semi-Supervised Self-Organizing Maps for Regression and Classification Focusing on Hyperspectral Data”. In: *Remote Sensing* 12.1 (2020) (cit. on pp. 2, 15, 24, 32, 39–70, 96, 101, 143, 147, 149, 150).

[7] Felix M. Riese and Sina Keller. “Supervised, Semi-Supervised, and Unsupervised Learning for Hyperspectral Regression”. In: *Hyperspectral Image Analysis: Advances in Machine Learning and Signal Processing*. Ed. by Saurabh Prasad and Jocelyn Chanussot. Cham: Springer International Publishing, 2020. Chap. 7, pp. 187–232 (cit. on pp. 2, 7–11, 15, 17–36, 143, 147).

[8] Felix M. Riese and Sina Keller. “Soil Texture Classification with 1D Convolutional Neural Networks based on Hyperspectral Data”. In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W5 (2019), pp. 615–621 (cit. on pp. 2, 15, 30, 71–80, 83, 84, 143, 151).

[9] Sina Keller, Philipp M. Maier, Felix M. Riese, et al. “Hyperspectral Data and Machine Learning for Estimating CDOM, Chlorophyll a, Diatoms, Green Algae, and Turbidity”. In: *International Journal of Environmental Research and Public Health* 15.9 (2018), p. 1881 (cit. on pp. 2, 15, 28, 30, 40, 42, 76, 144).

[10] Gustavo Camps-Valls, Devis Tuia, Luis Gómez-Chova, Sandra Jiménez, and Jesús Malo. “Remote sensing image processing”. In: *Synthesis Lectures on Image, Video, and Multimedia Processing* 5.1 (2011), pp. 1–192 (cit. on pp. 2, 10, 17, 25).

[11] José M. Bioucas-Dias, Antonio Plaza, Gustavo Camps-Valls, et al. “Hyperspectral remote sensing data analysis and future challenges”. In: *IEEE Geoscience and remote sensing magazine* 1.2 (2013), pp. 6–36 (cit. on p. 2).

[12] Henrik Petersson, David Gustafsson, and David Bergström. “Hyperspectral Image Analysis Using Deep Learning - A Review”. In: *2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*. IEEE. 2016, pp. 1–6 (cit. on pp. 2, 29, 72, 73).

[13] Utsav B. Gewali, Sildomar T. Monteiro, and Eli Saber. “Machine learning based hyperspectral image analysis: A survey”. In: *arXiv preprint arXiv:1802.08701* (2018) (cit. on pp. 2, 10).

[14] Nicolas Audebert, Bertrand Le Saux, and Sébastien Lefèvre. “Deep Learning for Classification of Hyperspectral Data: A Comparative Review”. In: *IEEE geoscience and remote sensing magazine* (2019), pp. 159–173 (cit. on pp. 2, 29).

[15] Xiao Xiang Zhu, Devis Tuia, Lichao Mou, et al. “Deep Learning in Remote Sensing: A Comprehensive Review and List of Resources”. In: *IEEE Geoscience and Remote Sensing Magazine* 5.4 (2017), pp. 8–36 (cit. on pp. 2, 72, 73).

[16] Behnood Rasti, Danfeng Hong, Renlong Hang, et al. “Feature Extraction for Hyperspectral Imagery: The Evolution from Shallow to Deep (Overview and Toolbox)”. In: *IEEE Geoscience and Remote Sensing Magazine* (2020) (cit. on pp. 2, 22).

[17] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014 (cit. on pp. 2, 10).

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016 (cit. on pp. 2, 20, 28).

[19] Laura Colini, Claudia Spinetti, Stefania Amici, et al. “Hyperspectral spaceborne, airborne and ground measurements campaign on Mt. Etna: multi data acquisitions in the frame of Prisma Mission (ASI-AGI Project n. I/016/11/0)”. In: *Quaderni di Geofisica* 119 (2014), pp. 1–51 (cit. on p. 3).

[20] Aman M. Kalteh, Peder Hjorth, and Ronny Berndtsson. “Review of the self-organizing map (SOM) approach in water resources: Analysis, modelling and application”. In: *Environmental Modelling & Software* 23.7 (2008), pp. 835–845 (cit. on pp. 3, 41).

[21] Tom M. Mitchell. *The need for biases in learning generalizations*. New Brunswick, New Jersey, USA: Department of Computer Science, Rutgers University, 1980 (cit. on pp. 4, 25).

[22] Jose G. Moreno-Torres, Troy Raeder, Rocío Alaiz-Rodríguez, Nitesh V. Chawla, and Francisco Herrera. “A unifying view on dataset shift in classification”. In: *Pattern Recognition* 45.1 (2012), pp. 521–530 (cit. on pp. 4, 18, 86).

[23] Sina Keller, Felix M. Riese, Johanna Stötzer, Philipp M. Maier, and Stefan Hinz. “Developing a machine learning framework for estimating soil moisture with VNIR hyperspectral data”. In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-1 (2018), pp. 101–108 (cit. on pp. 7, 13, 15, 28, 30, 40, 42, 76, 78, 144).

[24] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. Adaptive computation and machine learning. Cambridge, MA, USA: MIT Press, 2006, p. 508 (cit. on pp. 8, 31, 32, 42).

[25] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. Springer series in statistics New York, 2001 (cit. on pp. 10, 28).

[26] Stuart Geman, Elie Bienenstock, and René Doursat. "Neural Networks and the Bias/Variance Dilemma". In: *Neural Computation* 4.1 (1992), pp. 1–58 (cit. on p. 10).

[27] Andreas Merentitis, Christian Debes, and Roel Heremans. "Ensemble Learning in Hyperspectral Image Classification: Toward Selecting a Favorable Bias-Variance Tradeoff". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7.4 (2014), pp. 1089–1102 (cit. on p. 10).

[28] George P. Petropoulos, Hywel M. Griffiths, Wouter Dorigo, Angelika Xaver, and Alexander Gruber. "Surface soil moisture estimation: significance, controls, and conventional measurement techniques". In: *Remote sensing of energy fluxes and soil moisture content*. 1st ed. Boca Raton: CRC Press, 2013. Chap. 2, pp. 29–48 (cit. on pp. 11, 17).

[29] Wouter Dorigo, Wolfgang Wagner, Clement Albergel, et al. "ESA CCI Soil Moisture for improved Earth system understanding: State-of-the art and future directions". In: *Remote Sensing of Environment* 203 (2017), pp. 185–215 (cit. on p. 11).

[30] Jagdish Shukla and Yale Mintz. "Influence of land-surface evapotranspiration on the earth's climate". In: *Science* 215.4539 (1982), pp. 1498–1501 (cit. on p. 11).

[31] Edwin T. Engman. "Applications of microwave remote sensing of soil moisture for water resources and agriculture". In: *Remote Sensing of Environment* 35.2-3 (1991), pp. 213–226 (cit. on p. 11).

[32] James S. Famiglietti, James W. Rudnicki, and Matthew Rodell. "Variability in surface moisture content along a hillslope transect: Rattlesnake Hill, Texas". In: *Journal of Hydrology* 210.1-4 (1998), pp. 259–281 (cit. on p. 11).

[33] Olivier Hébrard, Marc Voltz, Patrick Andrieux, and Roger Moussa. "Spatio-temporal distribution of soil surface moisture in a heterogeneously farmed Mediterranean catchment". In: *Journal of Hydrology* 329.1-2 (2006), pp. 110–121 (cit. on p. 11).

[34] Rocco Panciera. "Effect of land surface heterogeneity on satellite near-surface soil moisture observations". PhD thesis. University of Melbourne, Department of Civil and Environmental Engineering, 2009 (cit. on p. 12).

[35] Ad-Hoc-Arbeitsgruppe Boden. *Bodenkundliche Kartieranleitung. KA5*. Ed. by Wolf Eckelmann, H. Sponagel, W. Grottenthaler, et al. 5th ed. Schweizerbart'sche Verlagsbuchhandlung, 2006 (cit. on pp. 12, 17, 71, 75).

[36] Soil Survey Staff. *Soil taxonomy: A basic system of soil classification for making and interpreting soil surveys*. 2nd ed. U.S. Department of Agriculture Handbook 436. Natural Resources Conservation Service, 1999 (cit. on pp. 12, 17).

[37] Daniel Hillel. *Applications of soil physics*. New York: Academic Press, 1980 (cit. on p. 12).

[38] Anthony R. Dexter. "Soil physical quality: Part I. Theory, effects of soil texture, density, and organic matter, and effects on root growth". In: *Geoderma* 120.3-4 (2004), pp. 201–214 (cit. on p. 12).

[39] Peter Fisher, Alexis J. Comber, and Richard Wadsworth. "Land use and land cover: contradiction or complement". In: *Re-presenting GIS* (2005), pp. 85–98 (cit. on p. 13).

[40] Giles M. Foody. "Status of land cover classification accuracy assessment". In: *Remote sensing of environment* 80.1 (2002), pp. 185–201 (cit. on p. 13).

[41] Felix M. Riese and Sina Keller. "Introducing a Framework of Self-Organizing Maps for Regression of Soil Moisture with Hyperspectral Data". In: *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*. Valencia, Spain, 2018, pp. 6151–6154 (cit. on pp. 13, 15, 40, 42, 50, 53, 54, 143).

[43] Gergely Tóth, Arwyn Jones, and Luca Montanarella. "The LUCAS topsoil database and derived information on the regional variability of cropland topsoil properties in the European Union". In: *Environmental Monitoring and Assessment* 185.9 (2013), pp. 7409–7425 (cit. on pp. 13, 17, 74).

[44] Gergely Tóth, Arwyn Jones, and Luca Montanarella. *LUCAS Topsoil Survey: Methodology, Data, and Results*. Tech. rep. JRC83529. Joint Research Centre of the European Commission, 2013 (cit. on pp. 13, 17, 74).

[45] Alberto Orgiazzi, Cristiano Ballabio, Panagiotis Panagos, Arwyn Jones, and Oihane Fernández-Ugalde. "LUCAS Soil, the largest expandable soil dataset for Europe: a review". In: *European Journal of Soil Science* 69.1 (2018), pp. 140–153 (cit. on pp. 13, 74, 83).

[46] Sina Keller, Felix M. Riese, Niklas Allroggen, Conrad Jackisch, and Stefan Hinz. "Modeling Subsurface Soil Moisture Based on Hyperspectral Data: First Results of a Multilateral Field Campaign". In: *Tagungsband der 37. Wissenschaftlich-Technische Jahrestagung der DGPF e.V.* Vol. 27. Munich, Germany, 2018, pp. 34–48 (cit. on pp. 13, 15, 143, 144).

[47] Felix M. Riese and Sina Keller. "Fusion of hyperspectral and ground penetrating radar data to estimate soil moisture". In: *2018 9th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. Amsterdam, Netherlands, 2018, pp. 1–5 (cit. on pp. 13, 20, 28, 87, 144).

[50] Benno A. Ommerborn. "Multitemporale Erkennung von Landnutzungsänderungen mithilfe von neuronalen Netzen und Sentinel-2 Fernerkundungsdaten". MA thesis. Karlsruhe, Germany: Karlsruhe Institute of Technology (KIT), 2018 (cit. on pp. 15, 37, 116).

[51] Carmen Rothfuß. "Estimation of Soil Texture with Machine Learning and Multispectral Data: A Case Study in Saxony, Germany". MA thesis. Karlsruhe, Germany: Karlsruhe Institute of Technology (KIT), 2019 (cit. on pp. 15, 37, 116).

[53] Fedro S. Zazueta and Jiannong Xin. “Soil moisture sensors”. In: *Soil Science* 73 (1994), pp. 391–401 (cit. on p. 16).

[54] George P. Petropoulos, Gareth Ireland, and Brian Barrett. “Surface soil moisture retrievals from remote sensing: Current status, products & future trends”. In: *Physics and Chemistry of the Earth, Parts A/B/C* 83 (2015), pp. 36–56 (cit. on p. 16).

[55] Willem Verstraeten, Frank Veroustraete, and Jan Feyen. “Assessment of evapotranspiration and soil moisture content across different scales of observation”. In: *Sensors* 8.1 (2008), pp. 70–117 (cit. on pp. 16, 17).

[56] David A. Robinson, Scott B. Jones, Jon M. Wraith, Daniel Or, and Shmulik P. Friedman. “A review of advances in dielectric and electrical conductivity measurement in soils using time domain reflectometry”. In: *Vadose Zone Journal* 2.4 (2003), pp. 444–475 (cit. on p. 17).

[57] Thomas Gräff, Erwin Zehe, Stefan Schlaeger, et al. “A quality assessment of Spatial TDR soil moisture measurements in homogenous and heterogeneous media with laboratory experiments.” In: *Hydrology & Earth System Sciences* 14.6 (2010) (cit. on p. 17).

[58] Glendon W. Gee and James W. Bauder. “Particle-size analysis”. In: *Methods of soil analysis: Part 1 Physical and mineralogical methods* 5 (1986), pp. 383–411 (cit. on p. 17).

[59] Maider Vidal and José Manuel Amigo. “Pre-processing of hyperspectral images. Essential steps before image analysis”. In: *Chemometrics and Intelligent Laboratory Systems* 117 (2012), pp. 138–148 (cit. on p. 17).

[60] Wes McKinney. “Data Structures for Statistical Computing in Python”. In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 51–56 (cit. on p. 18).

[61] Martín Abadi, Paul Barham, Jianmin Chen, et al. “TensorFlow: A system for large-scale machine learning”. In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. Savannah, GA: USENIX Association, 2016, pp. 265–283 (cit. on pp. 18, 20).

[62] Justin Matejka and George Fitzmaurice. “Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics Through Simulated Annealing”. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. Denver, Colorado, USA: ACM, 2017, pp. 1290–1294 (cit. on p. 18).

[63] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009 (cit. on pp. 18, 19, 85, 86).

[64] Hidetoshi Shimodaira. “Improving predictive inference under covariate shift by weighting the log-likelihood function”. In: *Journal of statistical planning and inference* 90.2 (2000), pp. 227–244 (cit. on pp. 19, 86).

[65] Amos Storkey. “When training and test sets are different: characterizing learning transfer”. In: *Dataset shift in machine learning* (2009), pp. 3–28 (cit. on pp. 19, 86).

[66] Tom Fawcett and Peter A. Flach. “A Response to Webb and Ting’s On the Application of ROC Analysis to Predict Classification Performance Under Varying Class Distributions”. In: *Machine Learning* 58.1 (2005), pp. 33–38 (cit. on pp. 19, 86).

[67] Gerhard Widmer and Miroslav Kubat. “Learning in the presence of concept drift and hidden contexts”. In: *Machine Learning* 23.1 (1996), pp. 69–101 (cit. on pp. 19, 86).

[68] Connor Shorten and Taghi M. Khoshgoftaar. “A survey on image data augmentation for deep learning”. In: *Journal of Big Data* 6.1 (2019), p. 60 (cit. on p. 20).

[70] John R. Taylor. *Error analysis*. 2nd ed. Sausalito, California: University Science Books, 1997 (cit. on p. 20).

[71] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, et al. “SciPy 1.0: fundamental algorithms for scientific computing in Python”. In: *Nature Methods* (2020), pp. 1–12 (cit. on p. 20).

[72] Melissa E. O’Neill. *PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation*. Tech. rep. HMC-CS-2014-09095. Claremont, CA: Harvey Mudd College, 2014 (cit. on p. 20).

[73] Jakub Nalepa, Michal Myller, and Michal Kawulok. “Hyperspectral data augmentation”. In: *arXiv preprint arXiv:1903.05580* (2019) (cit. on p. 21).

[74] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680 (cit. on p. 21).

[75] Antonia Creswell, Tom White, Vincent Dumoulin, et al. “Generative adversarial networks: An overview”. In: *IEEE Signal Processing Magazine* 35.1 (2018), pp. 53–65 (cit. on p. 21).

[76] Lin Zhu, Yushi Chen, Pedram Ghamisi, and Jón Atli Benediktsson. “Generative Adversarial Networks for Hyperspectral Image Classification”. In: *IEEE Transactions on Geoscience and Remote Sensing* 56.9 (2018), pp. 5046–5063 (cit. on p. 21).

[77] Zhi He, Han Liu, Yiwen Wang, and Jie Hu. “Generative Adversarial Networks-Based Semi-Supervised Learning for Hyperspectral Image Classification”. In: *Remote Sensing* 9.10 (2017) (cit. on p. 21).

[78] Stephen V. Stehman. “Basic probability sampling designs for thematic map accuracy assessment”. In: *International Journal of Remote Sensing* 20.12 (1999), pp. 2423–2441 (cit. on p. 22).

[79] Alan H. Fielding and John F. Bell. “A review of methods for the assessment of prediction errors in conservation presence/absence models”. In: *Environmental Conservation* 24.1 (1997), pp. 38–49 (cit. on p. 22).

[80] Chein-I Chang. “A Review of Virtual Dimensionality for Hyperspectral Imagery”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 11.4 (2018), pp. 1285–1305 (cit. on p. 22).

[81] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. “Reducing the Dimensionality of Data with Neural Networks”. In: *Science* 313 (2006), pp. 504–507 (cit. on pp. 22, 73, 94).

[82] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. “Dimensionality reduction: a comparative”. In: *Journal of Machine Learning Research* 10.66-71 (2009), p. 13 (cit. on p. 22).

[83] Xiuping Jia, Bor-Chen Kuo, and Melba M. Crawford. “Feature Mining for Hyperspectral Image Classification”. In: *Proceedings of the IEEE* 101 (2013), pp. 676–697 (cit. on p. 22).

[84] Karl Pearson. “On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572 (cit. on p. 23).

[85] Geoffrey E. Hinton and Richard S. Zemel. “Autoencoders, Minimum Description Length and Helmholtz Free Energy”. In: *Advances in Neural Information Processing Systems 6*. Ed. by J. D. Cowan, G. Tesauro, and J. Alspector. Morgan-Kaufmann, 1994, pp. 3–10 (cit. on p. 23).

[86] Laurens van der Maaten and Geoffrey Hinton. “Visualizing data using t-SNE”. In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605 (cit. on p. 23).

[87] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. “UMAP: Uniform Manifold Approximation and Projection”. In: *The Journal of Open Source Software* 3.29 (2018), p. 861 (cit. on p. 23).

[88] James MacQueen. “Some methods for classification and analysis of multivariate observations”. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Vol. 1. 14. Berkeley, Calif.: University of California Press, 1967, pp. 281–297 (cit. on p. 24).

[89] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. “A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. Portland, Oregon: AAAI Press, 1996, pp. 226–231 (cit. on p. 24).

[90] Teuvo Kohonen. “The self-organizing map”. In: *Proceedings of the IEEE* 78.9 (1990), pp. 1464–1480 (cit. on pp. 24, 40).

[91] John W. Rouse Jr., Rüdiger H. Haas, John A. Schell, and Donald W. Deering. “Monitoring vegetation systems in the Great Plains with ERTS”. In: *Third Earth Resources Technology Satellite-1 Symposium*. Greenbelt, 1974, pp. 309–317 (cit. on p. 24).

[92] Laura von Rueden, Sebastian Mayer, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. “Informed Machine Learning-Towards a Taxonomy of Explicit Integration of Knowledge into Machine Learning”. In: *arXiv preprint arXiv:1903.12394* (2019) (cit. on p. 24).

[93] Isabelle Guyon and André Elisseeff. “An introduction to variable and feature selection”. In: *Journal of machine learning research* 3 (2003), pp. 1157–1182 (cit. on p. 25).

[94] Leo Breiman, Jerome Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and regression trees*. Routledge, 1984 (cit. on p. 26).

[95] Leo Breiman. “Random Forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32 (cit. on pp. 27, 54).

[96] Pierre Geurts, Damien Ernst, and Louis Wehenkel. “Extremely randomized trees”. In: *Machine Learning* 63.1 (2006), pp. 3–42 (cit. on p. 27).

[97] Harris Drucker and Corinna Cortes. “Boosting decision trees”. In: *Advances in neural information processing systems*. 1996, pp. 479–485 (cit. on p. 27).

[98] Robert E. Schapire. “A Brief Introduction to Boosting”. In: *International Joint Conferences on Artificial Intelligence*. Vol. 99. 1999, pp. 1401–1406 (cit. on p. 27).

[99] Leo Breiman. *Arcing The Edge*. Tech. rep. 486. Statistics Department, University of California at Berkeley, 1997 (cit. on p. 27).

[100] Jerome H. Friedman. “Stochastic gradient boosting”. In: *Computational Statistics & Data Analysis* 38.4 (2002), pp. 367–378 (cit. on p. 27).

[101] Onisimo Mutanga, Elhadi Adam, and Moses Azong Cho. “High density biomass estimation for wetland vegetation using WorldView-2 imagery and random forest regression algorithm”. In: *International Journal of Applied Earth Observation and Geoinformation* 18 (2012), pp. 399–406 (cit. on p. 28).

[102] Elfatih M. Abdel-Rahman, Fethi B. Ahmed, and Riyad Ismail. “Random forest regression and spectral band selection for estimating sugarcane leaf nitrogen concentration using EO-1 Hyperion hyperspectral data”. In: *International Journal of Remote Sensing* 34.2 (2013), pp. 712–728 (cit. on p. 28).

[103] Philipp M. Maier and Sina Keller. “Machine Learning Regression on Hyperspectral Data to Estimate Multiple Water Parameters”. In: *2018 9th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. Amsterdam, Netherlands, 2018, pp. 1–5 (cit. on p. 28).

[104] Philipp M. Maier and Sina Keller. “Application of Different Simulated Spectral Data and Machine Learning to Estimate the Chlorophyll a Concentration of Several Inland Waters”. In: *2019 10th Workshop on Hyperspectral Imaging and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. IEEE. Amsterdam, Netherlands, 2019, pp. 1–5 (cit. on p. 28).

[105] Philipp M. Maier and Sina Keller. “Estimating Chlorophyll a Concentrations of Several Inland Waters with Hyperspectral Data and Machine Learning Models”. In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W5 (2019), pp. 609–614 (cit. on p. 28).

[106] Pall Oskar Gislason, Jón Atli Benediktsson, and Johannes R. Sveinsson. “Random forests for land cover classification”. In: *Pattern Recognition Letters* 27.4 (2006), pp. 294–300 (cit. on p. 28).

[107] Victor Francisco Rodriguez-Galiano, Bardan Ghimire, John Rogan, Mario Chica-Olmo, and Juan Pedro Rigol-Sanchez. “An assessment of the effectiveness of a random forest classifier for land-cover classification”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 67 (2012), pp. 93–104 (cit. on p. 28).

[108] JiSoo Ham, Yangchi Chen, Melba M. Crawford, and Joydeep Ghosh. “Investigation of the Random Forest Framework for Classification of Hyperspectral Data”. In: *IEEE Transactions on Geoscience and Remote Sensing* 43.3 (2005), pp. 492–501 (cit. on pp. 28, 72, 76).

[109] David J. Brown, Keith D. Shepherd, Markus G. Walsh, M. Dewayne Mays, and Thomas G. Reinsch. “Global soil characterization with VNIR diffuse reflectance spectroscopy”. In: *Geoderma* 132.3-4 (2006), pp. 273–290 (cit. on p. 28).

[110] Lanfa Liu, Min Ji, and Manfred Buchroithner. “Combining Partial Least Squares and the Gradient-Boosting Method for Soil Property Retrieval Using Visible Near-Infrared Shortwave Infrared Spectra”. In: *Remote Sensing* 9 (2017), p. 1299 (cit. on p. 28).

[111] Yann LeCun, Bernhard Boser, John S. Denker, et al. “Backpropagation Applied to Handwritten Zip Code Recognition”. In: *Neural Computation* 1.4 (1989), pp. 541–551 (cit. on p. 29).

[112] Sinno Jialin Pan and Qiang Yang. “A Survey on Transfer Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2009), pp. 1345–1359 (cit. on p. 30).

[113] Jia Deng, Wei Dong, Richard Socher, et al. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 248–255 (cit. on p. 30).

[114] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *3rd International Conference on Learning Representations, ICLR*. San Diego, CA, USA, 2015 (cit. on p. 30).

[115] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778 (cit. on p. 30).

[116] David Servan-Schreiber, Axel Cleeremans, and James L. McClelland. “Learning sequential structure in simple recurrent networks”. In: *Advances in neural information processing systems*. 1989, pp. 643–652 (cit. on p. 30).

[117] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (1997), pp. 1735–1780 (cit. on p. 30).

[118] Peter M. Atkinson and Adrian R. L. Tatnall. “Introduction Neural networks in remote sensing”. In: *International Journal of Remote Sensing* 18 (1997), pp. 699–709 (cit. on p. 30).

[119] Qiu-Xiang Yi, Jing-Feng Huang, Fu-Min Wang, Xiu-Zhen Wang, and Zhan-Yu Liu. "Monitoring Rice Nitrogen Status Using Hyperspectral Reflectance and Artificial Neural Network". In: *Environmental Science & Technology* 41.19 (2007), pp. 6770–6775 (cit. on p. 30).

[120] Ling Chen, Jing-feng Huang, Fumin Wang, and Yan-Lin Tang. "Comparison between back propagation neural network and regression models for the estimation of pigment content in rice leaves and panicles using hyperspectral data". In: *International Journal of Remote Sensing* 28.16 (2007), pp. 3457–3478 (cit. on p. 30).

[121] Lanfa Liu, Min Ji, and Manfred Buchroithner. "Transfer Learning for Soil Spectroscopy Based on Convolutional Neural Networks and Its Application in Soil Clay Content Mapping Using Hyperspectral Imagery". In: *Sensors* 18.9 (2018), p. 3169 (cit. on pp. 30, 73, 74, 77, 79, 81, 83, 113, 151).

[122] Jiaxuan You, Xiaocheng Li, Melvin Low, David Lobell, and Stefano Ermon. "Deep Gaussian Process for Crop Yield Prediction Based on Remote Sensing Data". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017, pp. 4559–4566 (cit. on p. 30).

[123] Kelvin Xu, Jimmy Ba, Ryan Kiros, et al. "Show, attend and tell: Neural image caption generation with visual attention". In: *International conference on machine learning*. 2015, pp. 2048–2057 (cit. on p. 30).

[124] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. "Dynamic routing between capsules". In: *Advances in neural information processing systems*. 2017, pp. 3856–3866 (cit. on p. 31).

[125] Kaiqiang Zhu, Yushi Chen, Pedram Ghamisi, Xiuping Jia, and Jón Atli Benediktsson. "Deep convolutional capsule network for hyperspectral image spectral and spectral-spatial classification". In: *Remote Sensing* 11.3 (2019), p. 223 (cit. on p. 31).

[126] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, eds. *Automated Machine Learning: Methods, Systems, Challenges*. Springer International Publishing, 2019 (cit. on p. 31).

[127] Matthias Feurer, Aaron Klein, Katharina Eggensperger, et al. "Efficient and Robust Automated Machine Learning". In: *Advances in Neural Information Processing Systems 28*. Curran Associates, Inc., 2015, pp. 2962–2970 (cit. on p. 31).

[128] Matthias Feurer, Aaron Klein, Katharina Eggensperger, et al. "Auto-sklearn: Efficient and Robust Automated Machine Learning". In: *Automated Machine Learning: Methods, Systems, Challenges*. Ed. by Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. Springer International Publishing, 2019, pp. 113–134 (cit. on p. 31).

[129] Randal S. Olson, Nathan Bartley, Ryan J. Urbanowicz, and Jason H. Moore. "Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science". In: *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. Denver, Colorado, USA: ACM, 2016, pp. 485–492 (cit. on p. 31).

[130] Randal S. Olson and Jason H. Moore. “TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning”. In: *Automated Machine Learning: Methods, Systems, Challenges*. Ed. by Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. Springer International Publishing, 2019, pp. 151–160 (cit. on p. 31).

[131] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on pp. 31, 42, 54, 61, 78, 101).

[132] Ariel Gordon, Elad Eban, Ofir Nachum, et al. “MorphNet: Fast & simple resource-constrained structure learning of deep networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1586–1595 (cit. on p. 31).

[133] Avital Oliver, Augustus Odena, Colin A. Raffel, Ekin D. Cubuk, and Ian J. Goodfellow. “Realistic Evaluation of Deep Semi-Supervised Learning Algorithms”. In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio, H. Wallach, H. Larochelle, et al. Curran Associates, Inc., 2018, pp. 3235–3246 (cit. on p. 31).

[134] Qiong Jackson and David A. Landgrebe. “An adaptive classifier design for high-dimensional data analysis with a limited training data set”. In: *IEEE Transactions on Geoscience and Remote Sensing* 39.12 (2001), pp. 2664–2679 (cit. on p. 32).

[135] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, 1998 (cit. on p. 32).

[136] Kristin P. Bennett and Ayhan Demiriz. “Semi-supervised Support Vector Machines”. In: *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II*. Cambridge, MA, USA: MIT Press, 1999, pp. 368–374 (cit. on p. 32).

[137] Lorenzo Bruzzone, Mingmin Chi, and Mattia Marconcini. “A Novel Transductive SVM for Semisupervised Classification of Remote-Sensing Images”. In: *IEEE Transactions on Geoscience and Remote Sensing* 44.11 (2006), pp. 3363–3373 (cit. on p. 32).

[138] Fan R. K. Chung and Fan Chung Graham. *Spectral graph theory*. 92. American Mathematical Society, 1997 (cit. on p. 32).

[139] Michael Irwin Jordan. *Learning in graphical models*. Vol. 89. Springer Science & Business Media, 1998 (cit. on p. 32).

[140] Mikhail Belkin and Partha Niyogi. “Semi-Supervised Learning on Riemannian Manifolds”. In: *Machine Learning* 56.1-3 (2004), pp. 209–239 (cit. on p. 32).

[141] Gustavo Camps-Valls, Tatyana V. Bandos Marsheva, and Dengyong Zhou. “Semi-Supervised Graph-Based Hyperspectral Image Classification”. In: *IEEE Transactions on Geoscience and Remote Sensing* 45.10 (2007), pp. 3044–3054 (cit. on p. 32).

[142] Dengyong Zhou, Olivier Bousquet, Thomas N. Lal, Jason Weston, and Bernhard Schölkopf. “Learning with local and global consistency”. In: *Advances in neural information processing systems*. 2004, pp. 321–328 (cit. on p. 32).

[143] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. “Manifold regularization: A geometric framework for learning from labeled and unlabeled examples”. In: *Journal of machine learning research* 7 (2006), pp. 2399–2434 (cit. on p. 32).

[144] Luis Gómez-Chova, Gustavo Camps-Valls, Jordi Munoz-Mari, and Javier Calpe. "Semisupervised Image Classification With Laplacian Support Vector Machines". In: *IEEE Geoscience and Remote Sensing Letters* 5.3 (2008), pp. 336–340 (cit. on p. 32).

[145] Frédéric Ratle, Gustavo Camps-Valls, and Jason Weston. "Semisupervised Neural Networks for Efficient Hyperspectral Image Classification". In: *IEEE Transactions on Geoscience and Remote Sensing* 48.5 (2010), pp. 2271–2282 (cit. on p. 32).

[146] Hao Wu and Saurabh Prasad. "Semi-Supervised Deep Learning Using Pseudo Labels for Hyperspectral Image Classification". In: *IEEE Transactions on Image Processing* 27.3 (2018), pp. 1259–1270 (cit. on p. 32).

[147] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. "Algorithms for hyper-parameter optimization". In: *Advances in neural information processing systems*. 2011, pp. 2546–2554 (cit. on p. 34).

[148] James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization". In: *Journal of Machine Learning Research* 13.Feb (2012), pp. 281–305 (cit. on p. 34).

[150] Marc Rußwurm and Marco Körner. "Multi-temporal land cover classification with sequential recurrent encoders". In: *ISPRS International Journal of Geo-Information* 7.4 (2018), p. 129 (cit. on p. 37).

[151] Marc Rußwurm and Marco Körner. "Convolutional LSTMs for cloud-robust segmentation of remote sensing imagery". In: *arXiv preprint arXiv:1811.02471* (2018) (cit. on p. 37).

[153] Jerry L. Hatfield, Anatoly A. Gitelson, James S. Schepers, and Charlie L. Walthall. "Application of spectral remote sensing for agronomic decisions". In: *Agronomy Journal* 100.Supplement_3 (2008), S–117 (cit. on p. 38).

[154] Mathieu Fauvel, Jocelyn Chanussot, and Jón Atli Benediktsson. "Decision fusion for the classification of urban remote sensing images". In: *IEEE Transactions on Geoscience and Remote Sensing* 44.10 (2006), pp. 2828–2838 (cit. on p. 38).

[155] Saurabh Prasad and Lori Mann Bruce. "Decision fusion with confidence-based weight assignment for hyperspectral target recognition". In: *IEEE Transactions on Geoscience and Remote Sensing* 46.5 (2008), pp. 1448–1456 (cit. on p. 38).

[156] Teuvo Kohonen. "Self-organized formation of topologically correct feature maps". In: *Biological Cybernetics* 43.1 (1982), pp. 59–69 (cit. on p. 40).

[157] Teuvo Kohonen. *Self-Organizing Maps*. Vol. 30. SSINF. Heidelberg, Berlin: Springer, 1995 (cit. on p. 40).

[158] Teuvo Kohonen. "Essentials of the self-organizing map". In: *Neural Networks* 37 (2013), pp. 52–65 (cit. on pp. 40, 43, 48, 68).

[160] Alfred Ultsch. "Self-organizing neural networks for visualisation and classification". In: *Information and classification*. Springer, 1993, pp. 307–313 (cit. on pp. 41, 57).

[161] Victor J. A. S. Lobo. “Application of Self-Organizing Maps to the Maritime Environment”. In: *Information Fusion and Geographic Information Systems*. Ed. by Vasily V. Popovich, Christophe Claramunt, Manfred Schrenk, and Kyrill V. Korolenko. Berlin, Heidelberg: Springer, 2009, pp. 19–36 (cit. on p. 41).

[162] Juha Vesanto and Esa Alhoniemi. “Clustering of the self-organizing map”. In: *IEEE Transactions on Neural Networks* 11.3 (2000), pp. 586–600 (cit. on p. 41).

[163] Jorge Muruzábal, Diego Vidaurre, and Julián Sánchez. “SOMwise regression: a new clusterwise regression method”. In: *Neural Computing and Applications* 21 (2012), pp. 1229–1241 (cit. on p. 41).

[164] Sheng-Hsun Hsu, J. J. Po-An Hsieh, Ting-Chih Chih, and Kuei-Chu Hsu. “A two-stage architecture for stock price forecasting by integrating self-organizing map and support vector regression”. In: *Expert Systems with Applications* 36.4 (2009), pp. 7947–7951 (cit. on p. 41).

[165] Kuo-lin Hsu, Hoshin V. Gupta, Xiaogang Gao, Soroosh Sorooshian, and Bisher Imam. “Self-organizing linear output map (SOLO): An artificial neural network suitable for hydrologic modeling and analysis”. In: *Water Resources Research* 38.12 (2002), pp. 1–17 (cit. on pp. 42, 69).

[166] Dimitrios Moshou, Cedric Bravo, Roberto Oberti, et al. “Plant disease detection based on data fusion of hyper-spectral and multi-spectral fluorescence imaging using Kohonen maps”. In: *Real-Time Imaging* 11.2 (2005), pp. 75–83 (cit. on p. 42).

[167] Markus Hagenbuchner and Ah Chung Tsoi. “A supervised training algorithm for self-organizing maps for structures”. In: *Pattern Recognition Letters* 26.12 (2005), pp. 1874–1884 (cit. on pp. 42, 50, 68).

[168] Chang Yoon Ji. “Land-Use Classification of Remotely Sensed Data Using Kohonen Self-organizing Feature Map Neural Networks”. In: *Photogrammetric Engineering & Remote Sensing* 66.12 (2000), pp. 1451–1460 (cit. on p. 42).

[169] Pablo Martinez, J. Anthony Gualtieri, Pedro L. Aguilar, et al. “Hyperspectral Image Classification using a Self-Organizing Map”. In: *Proceedings of the Tenth JPL Airborne Earth Science Workshop*. Vol. 10. 2001, pp. 267–274 (cit. on p. 42).

[170] Nicola Zaccarelli, Giovanni Zurlini, G. Rizzo, Euro Blasi, and M. Palazzo. “Spectral Self-Organizing Map for hyperspectral image classification”. In: *Sensors for Environmental Control*. World Scientific Publishing Company Incorporated, 2003, pp. 218–223 (cit. on p. 42).

[171] Yanfei Zhong, Liangpei Zhang, Bo Huang, and Pingxiang Li. “An Unsupervised Artificial Immune Classifier for Multi/Hyperspectral Remote Sensing Imagery”. In: *IEEE Transactions on Geoscience and Remote Sensing* 44.2 (2006), pp. 420–431 (cit. on p. 42).

[172] Françoise Fessant, Patrice Aknin, Latifa Oukhellou, and Sophie Midenet. “Comparison of Supervised Self-Organizing Maps Using Euclidian or Mahalanobis Distance in Classification Context”. In: *Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*. Ed. by José Mira and Alberto Prieto. Vol. 2084. Berlin, Heidelberg: Springer, 2001, pp. 637–644 (cit. on p. 42).

[173] Thomas Hecht, Mathieu Lefort, and Alexander Gepperth. “Using self-organizing maps for regression: the importance of the output function”. In: *European Symposium on Artificial Neural Networks (ESANN)*. Bruges, Belgium, 2015, pp. 1–6 (cit. on p. 42).

[174] Golnoush Abaei, Ali Selamat, and Hamido Fujita. “An empirical study based on semi-supervised hybrid self-organizing map for software fault prediction”. In: *Knowledge-Based Systems* 74 (2015), pp. 28–39 (cit. on p. 42).

[175] Pedro H. M. Braga and Hansenclever F. Bassani. “A Semi-Supervised Self-Organizing Map for Clustering and Classification”. In: *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2018, pp. 1–8 (cit. on pp. 42, 44).

[176] Ron Wehrens and Johannes Kruisselbrink. “Flexible Self-Organizing Maps in kohonen 3.0”. In: *Journal of Statistical Software* 87.7 (2018), pp. 1–18 (cit. on pp. 42, 44).

[181] Michael Hanke, Yaroslav O. Halchenko, Per B. Sederberg, et al. “PyMVPA: a unifying approach to the analysis of neuroscientific data”. In: *Frontiers in neuroinformatics* 3 (2009), p. 3 (cit. on pp. 42, 44).

[183] Mohammed Attik, Laurent Bougrain, and Frédéric Alexandre. “Self-organizing Map Initialization”. In: *Artificial Neural Networks: Biological Inspirations – ICANN 2005*. Ed. by Włodzisław Duch, Janusz Kacprzyk, Erkki Oja, and Sławomir Zadrożny. Vol. 3696. Berlin, Heidelberg: Springer, 2005, pp. 357–362 (cit. on p. 45).

[184] Ayodeji A. Akinduko, Evgeny M. Mirkes, and Alexander N. Gorban. “SOM: Stochastic initialization versus principal components”. In: *Information Sciences* 364-365 (2016), pp. 213–221 (cit. on p. 45).

[185] Guilherme A. Barreto and Aluizio F. R. Araújo. “Identification and Control of Dynamical Systems Using the Self-Organizing Map”. In: *IEEE Transactions on Neural Networks* 15.5 (2004), pp. 1244–1259 (cit. on pp. 47, 149).

[186] Haruna Matsushita and Yoshifumi Nishio. “Batch-Learning Self-Organizing Map with Weighted Connections Avoiding False-Neighbor Effects”. In: *The 2010 International Joint Conference on Neural Networks (IJCNN)* (2010), pp. 1–6 (cit. on p. 47).

[187] Helge Ritter, Thomas Martinetz, and Klaus Schulten. *Neural computation and self-organizing maps: an introduction*. Addison-Wesley Reading, MA, 1992 (cit. on p. 47).

[188] Roberto Horowitz and Luis Alvarez. “Convergence properties of self-organizing neural networks”. In: *Proceedings of 1995 American Control Conference-ACC’95*. Vol. 2. IEEE. 1995, pp. 1339–1344 (cit. on p. 47).

[190] Yuhang Zhang, Hsiuhan Lexie Yang, Saurabh Prasad, et al. “Ensemble multiple kernel active learning for classification of multisource remote sensing data”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8.2 (2014), pp. 845–858 (cit. on p. 69).

[191] Edward A. Cloutis. “Review Article Hyperspectral geological remote sensing: evaluation of analytical techniques”. In: *International Journal of Remote Sensing* 17.12 (1996), pp. 2215–2242 (cit. on p. 72).

[192] Farid Melgani and Lorenzo Bruzzone. “Classification of Hyperspectral Remote Sensing Images With Support Vector Machines”. In: *IEEE Transactions on Geoscience and Remote Sensing* 42 (2004), pp. 1778–1790 (cit. on p. 72).

[195] Xudong Zhang, Veeraraghavan Vijayaraj, and Nicolas H. Younan. “Hyperspectral Soil Texture Classification”. In: *IEEE Workshop on Advances in Techniques for Analysis of Remotely Sensed Data, 2003*. 2003, pp. 182–186 (cit. on p. 72).

[196] Xudong Zhang, Nicolas H. Younan, and Charles G. O’Hara. “Wavelet Domain Statistical Hyperspectral Soil Texture Classification”. In: *IEEE Transactions on Geoscience and Remote Sensing* 43.3 (2005), pp. 615–618 (cit. on p. 72).

[197] Dhruba P. Shrestha, Dante E. Margate, Freek van der Meer, and Viet Anh Hoang. “Analysis and classification of hyperspectral data for mapping land degradation: An application in southern Spain”. In: *International Journal of Applied Earth Observation and Geoinformation* 7.2 (2005), pp. 85–96 (cit. on p. 72).

[198] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, et al. “1D Convolutional Neural Networks and Applications: A Survey”. In: *arXiv preprint arXiv:1905.03554* (2019) (cit. on p. 73).

[199] Serkan Kiranyaz, Turker Ince, and Moncef Gabbouj. “Real-time patient-specific ECG classification by 1-D convolutional neural networks”. In: *IEEE Transactions on Biomedical Engineering* 63.3 (2015), pp. 664–675 (cit. on p. 73).

[200] Onur Avci, Osama Abdeljaber, Serkan Kiranyaz, and Daniel Inman. “Structural damage detection in real time: implementation of 1D convolutional neural networks for SHM applications”. In: *Structural Health Monitoring & Damage Detection, Volume 7*. Springer, 2017, pp. 49–54 (cit. on p. 73).

[201] Turker Ince, Serkan Kiranyaz, Levent Eren, Murat Askar, and Moncef Gabbouj. “Real-time motor fault detection by 1-D convolutional neural networks”. In: *IEEE Transactions on Industrial Electronics* 63.11 (2016), pp. 7067–7075 (cit. on p. 73).

[202] Serkan Kiranyaz, Adel Gastli, Lazhar Ben-Brahim, Nasser Al-Emadi, and Moncef Gabbouj. “Real-time fault detection and identification for MMC using 1-D convolutional neural networks”. In: *IEEE Transactions on Industrial Electronics* 66.11 (2018), pp. 8760–8771 (cit. on p. 73).

[203] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis. “Deep Supervised Learning for Hyperspectral Data Classification Through Convolutional Neural Networks”. In: *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. 2015, pp. 4959–4962 (cit. on p. 73).

[204] Wei Hu, Yangyu Huang, Li Wei, Fan Zhang, and Hengchao Li. “Deep Convolutional Neural Networks for Hyperspectral Image Classification”. In: *Journal of Sensors* 2015 (2015), pp. 1–12 (cit. on pp. 73, 77, 79, 81, 83, 113, 151).

[205] Bei Zhao, Bo Huang, and Yanfei Zhong. “Transfer Learning With Fully Pretrained Deep Convolution Networks for Land-Use Classification”. In: *IEEE Geoscience and Remote Sensing Letters* 14.9 (2017), pp. 1436–1440 (cit. on p. 73).

[206] Emanuele Lugato, Lily Paniagua, Arwyn Jones, Wim de Vries, and Adrian Leip. "Complementing the topsoil information of the Land Use/Land Cover Area Frame Survey (LUCAS) with modelled N2O emissions". In: *PLOS ONE* 12.4 (2017), pp. 1–16 (cit. on p. 73).

[207] Panos Panagos, Cristiano Ballabio, Yusuf Yigini, and Martha B. Dunbar. "Estimating the soil organic carbon content for European NUTS2 regions based on LUCAS data collection". In: *Science of The Total Environment* 442 (2013), pp. 235–246 (cit. on p. 73).

[208] Marco Nocita, Antoine Stevens, Gergely Tóth, et al. "Prediction of soil organic carbon content by diffuse reflectance spectroscopy using a local partial least square regression approach". In: *Soil Biology and Biochemistry* 68 (2014), pp. 337–347 (cit. on p. 73).

[209] Fabio Castaldi, Sabine Chabrillat, Arwyn Jones, et al. "Soil Organic Carbon Estimation in Croplands by Hyperspectral Remote APEX Data Using the LUCAS Topsoil Database". In: *Remote Sensing* 10.2 (2018), p. 153 (cit. on p. 73).

[210] Alessandra Palmieri, Paolo Dominici, Marjo Kasanko, and Laura Martino. "Diversified landscape structure in the EU Member States". In: *Statistics in focus* 21 (2011) (cit. on p. 73).

[211] Alessandra Palmieri, Laura Martino, Paolo Dominici, and Marjo Kasanko. "Land Cover and Land Use Diversity Indicators in LUCAS 2009 data". In: *Land Quality and Land Use Information in the European Union* (2011), pp. 59–68 (cit. on p. 73).

[212] Dirk Pflugmacher, Andreas Rabe, Mathias Peters, and Patrick Hostert. "Mapping pan-European land cover using Landsat spectral-temporal metrics and the European LUCAS survey". In: *Remote Sensing of Environment* 221 (2019), pp. 583–595 (cit. on p. 73).

[213] Panos Panagos, Katrin Meusburger, Cristiano Ballabio, Pasqualle Borrelli, and Christine Alewell. "Soil erodibility in Europe: A high-resolution dataset based on LUCAS". In: *Science of The Total Environment* 479 (2014), pp. 189–200 (cit. on p. 73).

[214] Cristiano Ballabio, Panos Panagos, and Luca Monatanarella. "Mapping topsoil physical properties at European scale using the LUCAS database". In: *Geoderma* 261 (2016), pp. 110–123 (cit. on p. 73).

[215] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cit. on pp. 77, 113).

[216] Rosanne Liu, Joel Lehman, Piero Molino, et al. "An intriguing failing of convolutional neural networks and the CoordConv solution". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio, H. Wallach, H. Larochelle, et al. Curran Associates, Inc., 2018, pp. 9628–9639 (cit. on pp. 77, 83, 113).

[217] Yushi Chen, Hanlu Jiang, Chunyang Li, Xiuping Jia, and Pedram Ghamisi. "Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks". In: *IEEE Transactions on Geoscience and Remote Sensing* 54.10 (2016), pp. 6232–6251 (cit. on p. 83).

[218] Devis Tuia, Edoardo Pasolli, and William J. Emery. “Dataset shift adaptation with active queries”. In: *2011 Joint Urban Remote Sensing Event*. IEEE. 2011, pp. 121–124 (cit. on pp. 85, 87, 110).

[221] Suju Rajan, Joydeep Ghosh, and Melba M. Crawford. “Exploiting class hierarchies for knowledge transfer in hyperspectral data”. In: *IEEE Transactions on Geoscience and Remote Sensing* 44.11 (2006), pp. 3408–3417 (cit. on p. 87).

[222] Luis Gómez-Chova, Gustavo Camps-Valls, Lorenzo Bruzzone, and Javier Calpe-Maravilla. “Mean map kernel methods for semisupervised cloud classification”. In: *IEEE Transactions on Geoscience and Remote Sensing* 48.1 (2009), pp. 207–220 (cit. on pp. 87, 110).

[223] Lorenzo Bruzzone and Mattia Marconcini. “Toward the automatic updating of land-cover maps by a domain-adaptation SVM classifier and a circular validation strategy”. In: *IEEE Transactions on Geoscience and Remote Sensing* 47.4 (2009), pp. 1108–1122 (cit. on p. 87).

[224] Giles M. Foody, Mahesh Pal, Duccio Rocchini, Carol X. Garzon-Lopez, and Lucy Bastin. “The sensitivity of mapping methods to reference data quality: Training supervised image classifications with imperfect reference data”. In: *ISPRS International Journal of Geo-Information* 5.11 (2016), p. 199 (cit. on p. 87).

[225] Arthur Elmes, Hamed Alemohammad, Ryan Avery, et al. “Accounting for training data error in machine learning applied to Earth observations”. In: *Remote Sensing* 12.6 (2020), p. 1034 (cit. on p. 87).

[226] Gen-Tao Chiang, Martin Dove, Stuart Ballard, Charles Bostater, and Ian Frame. “A grid enabled Monte Carlo hyperspectral synthetic image remote sensing model (GRID-MCHSIM) for coastal water quality algorithm”. In: *Remote Sensing of the Ocean, Sea Ice, and Large Water Regions 2006*. Ed. by Charles R. Bostater Jr., Xavier Neyt, Stelios P. Mertikas, and Miguel Vélez-Reyes. Vol. 6360. International Society for Optics and Photonics. SPIE, 2006, pp. 60–70 (cit. on p. 87).

[227] Linlin Xu, Fan Li, Alexander Wong, and David A. Clausi. “Hyperspectral image denoising using a spatial–spectral monte carlo sampling approach”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8.6 (2015), pp. 3025–3038 (cit. on p. 87).

[228] Qi Wei, Nicolas Dobigeon, and Jean-Yves Tourneret. “Bayesian fusion of hyperspectral and multispectral images”. In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 3176–3180 (cit. on p. 87).

[229] Christian D. León, Hannah Kosow, Yvonne Zahumensky, et al. “Solutions and planning tools for water supply and wastewater management in prosperous regions tackling water scarcity”. In: *Proceedings of the GRoW Midterm Conference - Global analyses and local solutions for sustainable water resources management, Frankfurt am Main, 20-21 February 2019. Ed.: A. Kramer*. Frankfurt am Main, Deutschland: adelphi, Berlin, 2019, pp. 28–31 (cit. on p. 87).

[230] Andrea S. Laliberte, Jeffrey E. Herrick, Albert Rango, and Craig Winters. “Acquisition, orthorectification, and object-based classification of unmanned aerial vehicle (UAV) imagery for rangeland monitoring”. In: *Photogrammetric Engineering & Remote Sensing* 76.6 (2010), pp. 661–672 (cit. on p. 89).

[231] Graham J. Gaskin and James D. Miller. “Measurement of soil water content using a simplified impedance measuring technique”. In: *Journal of agricultural engineering research* 63.2 (1996), pp. 153–159 (cit. on p. 89).

[232] James D. Miller and Graham J. Gaskin. *ThetaProbe ML2x: Principles of operation and applications*. Tech. rep. 2. Aberdeen, Scotland: Macaulay Land Use Research Institute, 1999 (cit. on pp. 89, 91).

[233] Michael Schmitt and Xiao Xiang Zhu. “Data fusion and remote sensing: An ever-growing relationship”. In: *IEEE Geoscience and Remote Sensing Magazine* 4.4 (2016), pp. 6–23 (cit. on p. 93).

[234] Stephan Rabanser, Stephan Günnemann, and Zachary Lipton. “Failing Loudly: An Empirical Study of Methods for Detecting Dataset Shift”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 1394–1406 (cit. on pp. 93, 109).

[235] Christoph Molnar. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. `https://christophm.github.io/interpretable-ml-book/`. 2019 (cit. on p. 116).

[236] José Alberto Silva de Sá, Brígida Ramati Pereira da Rocha, Arthur Almeida, and José Ricardo Souza. “Recurrent Self-Organizing Map for Severe Weather Patterns Recognition”. In: *Recurrent Neural Networks and Soft Computing*. Rijeka: IntechOpen, 2012. Chap. 8, pp. 151–174 (cit. on p. 149).

Webpages

[42] Felix M. Riese and Sina Keller. *Hyperspectral benchmark dataset on soil moisture*. 2018. URL: `https://github.com/felixriese/hyperspectral-soilmoisture-dataset` (cit. on pp. 13, 15, 53, 111, 144).

[48] Sina Keller, Felix M. Riese, Niklas Allroggen, and Conrad Jackisch. *HydReSGeo: Field experiment dataset of surface-subsurface infiltration dynamics acquired by hydrological, remote sensing, and geophysical measurement techniques*. 2020. URL: `https://doi.org/10.5880/fidgeo.2020.015` (cit. on pp. 13, 15, 144).

[49] Felix M. Riese. *Hyperspectral Processing Scripts for the HydReSGeo Dataset*. 2020. URL: `https://github.com/felixriese/hyperspectral-processing` (cit. on pp. 15, 145).

[52] Felix M. Riese. *Processing Scripts for Thermal Infrared Cameras*. 2019. URL: `https://github.com/felixriese/thermal-image-processing` (cit. on pp. 15, 145).

[69] Jens Leitloff and Felix M. Riese. *Examples for CNN training and classification on Sentinel-2 data*. 2018. URL: https://github.com/jensleitloff/CNN-Sentinel (cit. on pp. 20, 116, 144).

[149] Felix M. Riese and Sina Keller. *Hyperspectral Regression: Code Examples*. 2019. URL: https://github.com/felixriese/hyperspectral-regression (cit. on pp. 36, 145).

[152] Oliver Sefrin, Felix M. Riese, and Sina Keller. *Soil Texture Processing*. 2019. URL: https://github.com/oliversefrin/soil-texture-processing (cit. on pp. 37, 145).

[159] Felix M. Riese. *SuSi: Supervised Self-organizing Maps in Python*. 2019. URL: https://github.com/felixriese/susi (cit. on pp. 40, 61, 96, 101, 112, 116, 144).

[177] Vahid Moosavi, Sebastian Packmann, and Iván Vallés. *SOMPY: A Python Library for Self Organizing Map (SOM)*. Commit of 4 March 2019. 2018. URL: https://github.com/sevamoo/SOMPY (visited on Mar. 4, 2019) (cit. on pp. 42, 44).

[178] Federico Comitani. *SimpSOM: a lightweight implementation of Kohonen Self-Organising Maps*. Release 1.3.3. 2018. URL: https://github.com/fcomitani/SimpSOM (visited on Mar. 4, 2019) (cit. on pp. 42, 44).

[179] Giuseppe Vettigli. *MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map*. Release 2.1.5. 2019. URL: https://github.com/JustGlowing/minisom (visited on Mar. 4, 2019) (cit. on pp. 42, 44).

[180] Chris Gorman. *A multi-gpu implementation of the self-organizing map in TensorFlow*. 2018. URL: https://github.com/cgorman/tensorflow-som (visited on Nov. 7, 2018) (cit. on pp. 42, 44).

[182] Yurii Shevchuk et al. *NeuPy*. 2015. URL: http://neupy.com/ (visited on Oct. 1, 2019) (cit. on pp. 42, 44).

[189] Grupo de Inteligencia Computacional de la Universidad del País Vasco. *Salinas AVIRIS Dataset*. URL: http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes#Salinas (visited on Mar. 13, 2019) (cit. on pp. 57, 58, 111).

[193] Felix M. Riese. *LUCAS Soil Texture Processing Scripts*. 2020. URL: https://github.com/felixriese/lucas-processing (cit. on pp. 72, 145).

[194] Felix M. Riese. *CNN Soil Texture Classification*. 2019. URL: https://github.com/felixriese/CNN-SoilTextureClassification (cit. on pp. 72, 78, 113, 144).

[219] Felix M. Riese, Samuel Schroers, Jan Wienhöfer, and Sina Keller. *Aerial Peruvian Andes Campaign (ALPACA) Dataset 2019*. KITopen. 2020. URL: https://doi.org/10.5445/IR/1000118082 (cit. on pp. 86, 109, 114, 144).

[220] Felix M. Riese. *Processing Scripts for the ALPACA Dataset*. 2020. URL: https://github.com/felixriese/alpaca-processing (cit. on pp. 86, 145).

List of Abbreviations

1D	1-dimensional
2D	2-dimensional
3D	3-dimensional
AA	Average Accuracy
AE	Autoencoder
AI	Artificial Intelligence
ALPACA	Aerial Peruvian Andes Campaign
AMSL	Above Mean Sea Level
ANN	Artificial Neural Network
a.u.	Arbitrary Units
AutoML	Automated Machine Learning
AVIRIS	Airborne Visible Infrared Imaging Spectrometer
BMBF	German Federal Ministry of Education and Research
BMU	Best Matching Unit
CNN	Convolutional Neural Network
CONV	Convolutional
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DEM	Digital Elevation Model
DT	Decision Tree
EnMAP	Environmental Mapping and Analysis Program
EO-1	Earth-Observing One
ET	Extremely Randomized Trees
FC	Fully-Connected
GAN	Generative Adversarial Network
GTB	Gradient Tree Boosting
HydReSGeo	Field experiment dataset of surface-subsurface infiltration dynamics acquired by hydrological, remote sensing, and geophysical measurement techniques
i.i.d.	Independent and Identically Distributed
IPF	Institute of Photogrammetry and Remote Sensing
k-NN	k-Nearest Neighbors
KA5	Bodenkundliche Kartieranleitung ed. 5
κ	Cohen's Kappa Coefficient

KarLy	Karlsruhe Lysimeter
KIT	Karlsruhe Institute of Technology
LSTM	Long Short-Term Memory
LUCAS	Land Use/Cover Area Frame Survey
MAE	Mean Absolute Error
MARS	Multivariate Adaptive Regression Splines
MC	Monte Carlo
MERIS	Medium Resolution Imaging Spectrometer
ML	Machine Learning
MSE	Mean Squared Error
NDVI	Normalized Difference Vegetation Index
OA	Overall Accuracy
PCA	Principal Component Analysis
PLS	Partial Least Squares
PSA	Particle Size Analysis
R^2	Coefficient of Determination
ReLU	Rectified Linear Unit
RF	Random Forest
RGB	Red–Green–Blue
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SOM	Self-Organizing Map
SSL	Semi-Supervised Learning
SSNN	Semi-Supervised Neural Network
SS-SOM	Semi-Supervised Self-Organizing Map
SVM	Support Vector Machine
SWIR	Short-Wavelength Infrared
SuSi	Supervised Self-Organizing Maps
TDR	Time Domain Reflectometry
TIR	Thermal Infrared
t-SNE	t-Distributed Stochastic Neighbor Embedding
TSVM	Transductive Support Vector Machine
UAV	Unmanned Aerial Vehicle
UMAP	Uniform Manifold Approximation and Projection
USDA	United States Department of Agriculture
VNIR	Visible and Near-Infrared
VIS	Visible Spectrum
WGS84	World Geodetic System 1984

List of Figures

List of Tables

List of Publications

A

In the following, the publications are listed which have been published by or with the author Felix M. Riese within the scope of the presented thesis over the time period 2017-2020. The ideas presented in this thesis have been partly published in the following publications. The respective passages are marked in the thesis with a bar in the corresponding color.

Main Publications:

- Felix M. Riese, Sina Keller, and Stefan Hinz. "Supervised and Semi-Supervised Self-Organizing Maps for Regression and Classification Focusing on Hyperspectral Data". In: *Remote Sensing* 12.1 (2020). **Peer-reviewed**, cited as [6] and marked in green.
- Felix M. Riese and Sina Keller. "Supervised, Semi-Supervised, and Unsupervised Learning for Hyperspectral Regression". In: *Hyperspectral Image Analysis: Advances in Machine Learning and Signal Processing*. Ed. by Saurabh Prasad and Jocelyn Chanussot. Cham: Springer International Publishing, 2020. Chap. 7, pp. 187–232. Cited as [7] and marked in blue.
- Felix M. Riese and Sina Keller. "Soil Texture Classification with 1D Convolutional Neural Networks based on Hyperspectral Data". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W5 (2019), pp. 615–621. **Peer-reviewed**, received best paper award, cited as [8] and marked in orange.

Further Publications:

- Sina Keller, Felix M. Riese, Niklas Allroggen, Conrad Jackisch, and Stefan Hinz. "Modeling Subsurface Soil Moisture Based on Hyperspectral Data: First Results of a Multilateral Field Campaign". In: *Tagungsband der 37. Wissenschaftlich-Technische Jahrestagung der DGPF e.V.*. Vol. 27. Munich, Germany, 2018, pp. 34–48. Cited as [46].
- Felix M. Riese and Sina Keller. "Introducing a Framework of Self-Organizing Maps for Regression of Soil Moisture with Hyperspectral Data". In: *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*. Valencia, Spain, 2018, pp. 6151–6154. Cited as [41].

- Sina Keller, Felix M. Riese, Johanna Stötzer, Philipp M. Maier, and Stefan Hinz. "Developing a machine learning framework for estimating soil moisture with VNIR hyperspectral data". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-1 (2018), pp. 101–108. **Peer-reviewed** and cited as [23].
- Sina Keller, Philipp M. Maier, Felix M. Riese, Steffen Norra, Andreas Holbach, Nicolas Börsig, et al. "Hyperspectral Data and Machine Learning for Estimating CDOM, Chlorophyll a, Diatoms, Green Algae, and Turbidity". In: *International Journal of Environmental Research and Public Health* 15.9 (2018), p. 1881. **Peer-reviewed** and cited as [9].
- Felix M. Riese and Sina Keller. "Fusion of hyperspectral and ground penetrating radar data to estimate soil moisture". In: *2018 9th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. Amsterdam, Netherlands, 2018, pp. 1–5. Cited as [47].

Published Datasets:

- Felix M. Riese and Sina Keller. *Hyperspectral benchmark dataset on soil moisture*. 2018. URL: `https://github.com/felixriese/hyperspectral-soilmoisture-dataset`. Cited as [42], referred to as KarLy dataset and introduced in Section 3.6.1.
- Sina Keller, Felix M. Riese, Niklas Allroggen, and Conrad Jackisch. *HydReSGeo: Field experiment dataset of surface-subsurface infiltration dynamics acquired by hydrological, remote sensing, and geophysical measurement techniques*. 2020. URL: `https://doi.org/10.5880/fidgeo.2020.015`. Cited as [48], referred to as HydReSGeo and introduced in [46].
- Felix M. Riese, Samuel Schroers, Jan Wienhöfer, and Sina Keller. *Aerial Peruvian Andes Campaign (ALPACA) Dataset 2019*. KITopen. 2020. URL: `https://doi.org/10.5445/IR/1000118082`. Cited as [219], referred to as ALPACA dataset and introduced in Section 5.3.

Published Open-Source Software:

- Jens Leitloff and Felix M. Riese. *Examples for CNN training and classification on Sentinel-2 data*. 2018. URL: `https://github.com/jensleitloff/CNN-Sentinel`. Cited as [69].
- Felix M. Riese. *CNN Soil Texture Classification*. 2019. URL: `https://github.com/felixriese/CNN-SoilTextureClassification`. Cited as [194].
- Felix M. Riese. *SuSi: Supervised Self-organizing Maps in Python*. 2019. URL: `https://github.com/felixriese/susi`. Cited as [159].

- Oliver Sefrin, Felix M. Riese, and Sina Keller. *Soil Texture Processing*. 2019. URL: https://github.com/oliversefrin/soil-texture-processing. Cited as [152].
- Felix M. Riese. *Processing Scripts for Thermal Infrared Cameras*. 2019. URL: https://github.com/felixriese/thermal-image-processing. Cited as [52].
- Felix M. Riese and Sina Keller. *Hyperspectral Regression: Code Examples*. 2019. URL: https://github.com/felixriese/hyperspectral-regression. Cited as [149].
- Felix M. Riese. *Hyperspectral Processing Scripts for the HydReSGeo Dataset*. 2020. URL: https://github.com/felixriese/hyperspectral-processing. Cited as [49].
- Felix M. Riese. *Processing Scripts for the ALPACA Dataset*. 2020. URL: https://github.com/felixriese/alpaca-processing. Cited as [220].
- Felix M. Riese. *LUCAS Soil Texture Processing Scripts*. 2020. URL: https://github.com/felixriese/lucas-processing. Cited as [193].

Variable Naming Conventions

B

Table B.1: Variable naming conventions of the presented thesis. The variables are described in more detail, for example, in Sections 2.2 and 3.3. (adapted from [6, 7])

Chapter	Variable	Description
All	x	Input datapoint with $x \in \mathbb{R}^m$
	y	Label of a datapoint
	m	Number of features of a datapoint
	n	Number of datapoints
	X	Set of n input datapoints with $X := (x_i, \ldots, x_n)$
	Y	Set of labels
	$\mathcal{X}$	Feature space with $\mathcal{X} \subset \mathbb{R}^m$
	$\mathcal{Y}$	Target variable space
3	n_{row}	Number of rows on the SOM grid
	n_{column}	Number of columns on the SOM grid
	t	Number of current iteration
	t_{max}	Number of maximum iterations, $t < t_{max}$
	$\mathbf{x}(t)$	Datapoint at iteration t with $\mathbf{x} \in \mathbb{R}^n$
	$y(t)$	Label of datapoint $\mathbf{x}(t)$
	$c(\mathbf{x})$	BMU of datapoint $\mathbf{x}(t)$ with $c \in \mathbb{R}^2$
	$\alpha(t)$	Function of the learning rate
	α_0	Start value of the learning rate
	$\sigma(t)$	Neighborhood function
	σ_0	Start value of the neighborhood function
	σ_{end}	End value of the neighborhood function
	$h_{c,i}$	Neighborhood distance weight between BMU c and node i
	$\mathbf{w}_i(t)$	Weight of node i at iteration t with $\mathbf{w}_i \in \mathbb{R}^n$
5	A_i	Measurement area with $i \in \{1, 2, 3, 4, 5\}$
	ρ	Relative grid area on a 2D histogram
	ξ	Relative grid overlap on a 2D histogram

C Supplementary Material

C.1 Supplementary Material of Chapter 3

This section includes material from

Felix M. Riese, Sina Keller, and Stefan Hinz. "Supervised and Semi-Supervised Self-Organizing Maps for Regression and Classification Focusing on Hyperspectral Data". In: *Remote Sensing* 12.1 (2020). It is cited as [6] and marked in green.

C.1.1 Additional SOM Formulas

One additional learning rate formula, was applied by [236], is given as [6]

$$\alpha(t) = \alpha_0 \cdot \exp(-t/t_{max}). \tag{C.1}$$

Another neighborhood function (see [185]) is defined as [6]

$$\sigma(t) = \sigma_0 \cdot \left(\frac{\sigma_{end}}{\sigma_0}\right)^{t/t_{max}}. \tag{C.2}$$

Further supplementary material can be found in [6].

C.1.2 SuSi Hyperparameters

Table C.1: Hyperparameters of the different SOM evaluations divided into general, unsupervised and supervised hyperparameters. The training time (last row) is not a hyperparameter but is included to present the computation times of the Supervised Self-Organizing Maps (SuSi) framework training. These training times are based on an Apple MacBook Pro 13-inch, 2017, with 3.1 GHz Dual-Core Intel Core i5 and 16 GB memory. (reprinted from [6])

Hyperparameters		Unsupervised	Supervised		Semi-Supervised	
		Clustering	Regression	Classification	Regression	Classification
General	Number of rows	50	50	80	50	60
	Number of columns	50	50	80	60	60
	Distance metric	(3.1)	(3.1)	(3.1)	(3.1)	(3.1)
	Learning rate start	0.5	0.7	0.7	0.6	0.6
	Learning rate end	0.05	0.07	0.07	0.07	0.07
	Nbh. distance weight	(3.4)	(3.4)	(3.4)	(3.4)	(3.4)
Unsupervised	Initialization	random	random	random	random	random
	Number of iterations	50 000	10 000	60 000	10 000	30 000
	Training mode	online	online	online	online	online
	Neighborhood mode	(3.3)	(3.3)	(3.3)	(C.2)	(C.2)
	Learning rate	(3.2)	(3.2)	(3.2)	(C.1)	(C.1)
Supervised	Initialization	-	random	random	random	random
	Number of iterations	-	20 000	60 000	50 000	70 000
	Training mode	-	online	online	online	online
	Neighborhood mode	-	(3.3)	(3.3)	(3.3)	(3.3)
	Learning rate	-	(3.2)	(3.2)	(3.2)	(3.2)
	Class weighting	-	-	no	-	no
	Training time in s	200	30	> 1800	38	220

C.2 Supplementary Material of Chapter 4

This section includes material from

Felix M. Riese and Sina Keller. "Soil Texture Classification with 1D Convolutional Neural Networks based on Hyperspectral Data". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W5 (2019), pp. 615–621. It is cited as [8].

Table C.2: Hyperparameters of the new 1-dimensional (1D) Convolutional Neural Network (CNN) approaches LucasCNN, LucasResNet and LucasCoordConv as well as the existing CNN approaches by Hu et al. [204] and Liu et al. [121]. The number of filters in the i-th CONV layer is defined as c_i and the number of units in the i-th FC layer is defined as f_j. As activations, Rectified Linear Units (ReLUs) and tanh are applied. The hyperparameter emphasized in **bold** are changed compared to the original publication [8]. (adapted from [8])

Hyperparameters	Lucas-CNN	Lucas-ResNet	Lucas-CoordConv	Hu et al. [204]	Liu et al. [121]
Number of epochs	**120**	**150**	120	200	235
Batch size	100	64	32	100	100
Kernel size	3	3	3	28	3
Pooling size	2	2	2	6	2
Activations	ReLU	ReLU	ReLU	tanh	ReLU
Padding	valid	same	valid	valid	valid
c_1	32	32	32	20	32
c_2	32	32	**32**	-	32
c_3	64	64	64	-	64
c_4	64	64	**64**	-	64
f_1	120	150	256	100	-
f_2	160	100	128	-	-
Loss	categorical cross entropy				
Optimizer	Adam				

C.3 Supplementary Material of Chapter 5

C.3.1 Soil Moisture Measurements

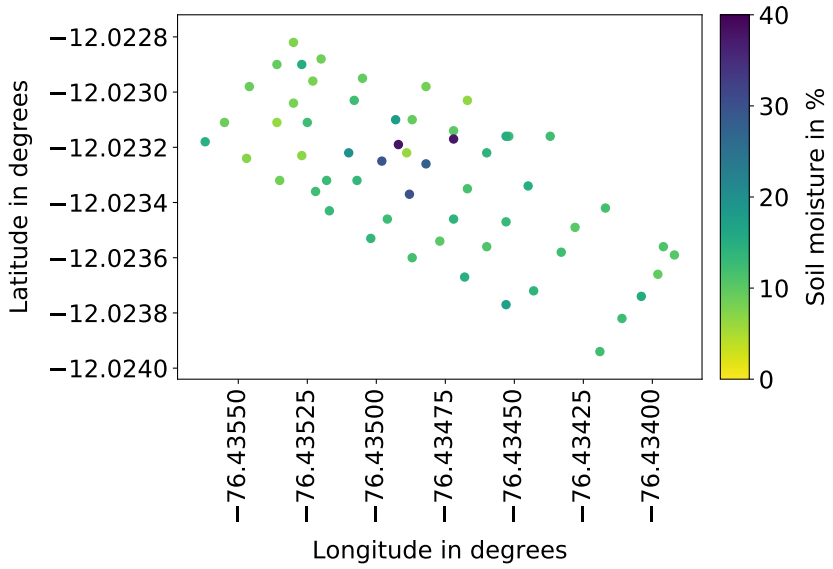


Figure C.1: Distribution of the soil moisture point measurements of measurement area A_2 in the coordinate system World Geodetic System 1984 (WGS84). Note that only datapoints are included in the regression study which overlap with the hyperspectral Unmanned Aerial Vehicle (UAV) image.

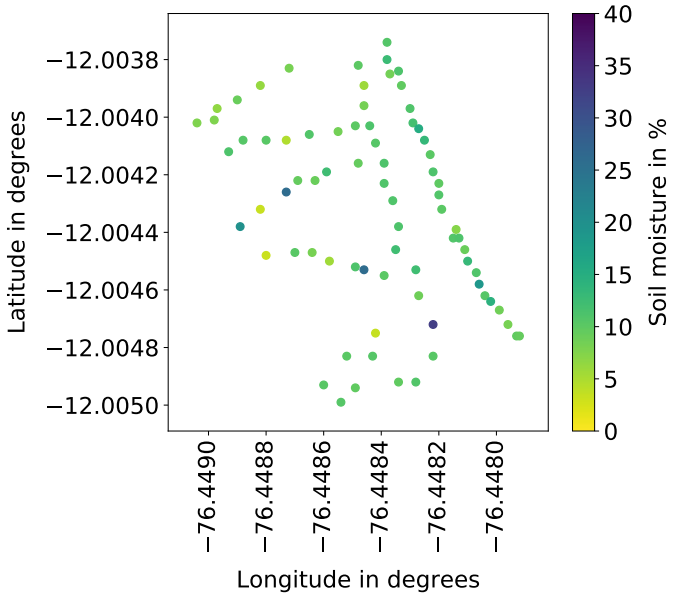


Figure C.2: Distribution of the soil moisture point measurements of measurement area A_3 in the coordinate system WGS84. Note that only datapoints are included in the regression study which overlap with the hyperspectral UAV image.

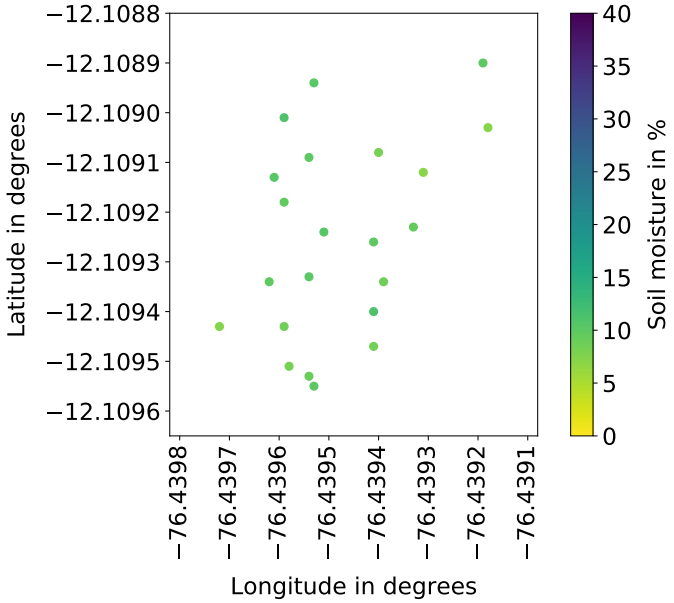


Figure C.3: Distribution of the soil moisture point measurements of measurement area A_4 in the coordinate system WGS84. Note that only datapoints are included in the regression study which overlap with the hyperspectral UAV image.

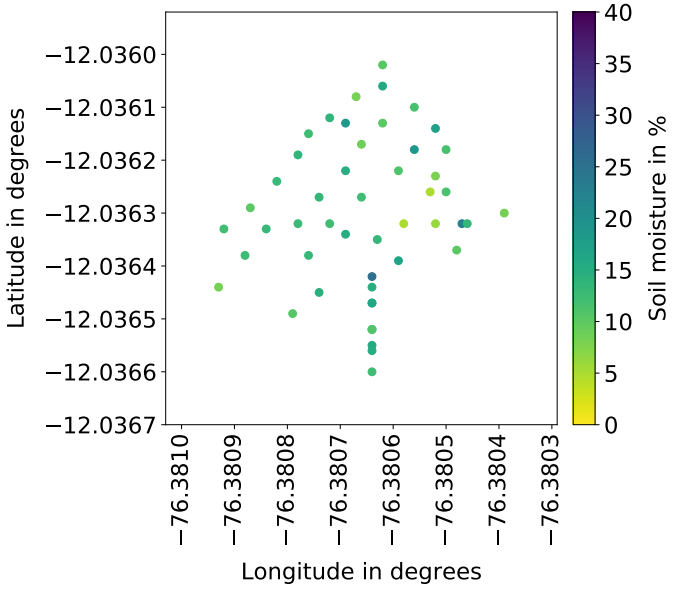


Figure C.4: Distribution of the soil moisture point measurements of measurement area A_5 in the coordinate system WGS84. Note that only datapoints are included in the regression study which overlap with the hyperspectral UAV image.

C.3.2 Monte Carlo Augmentation

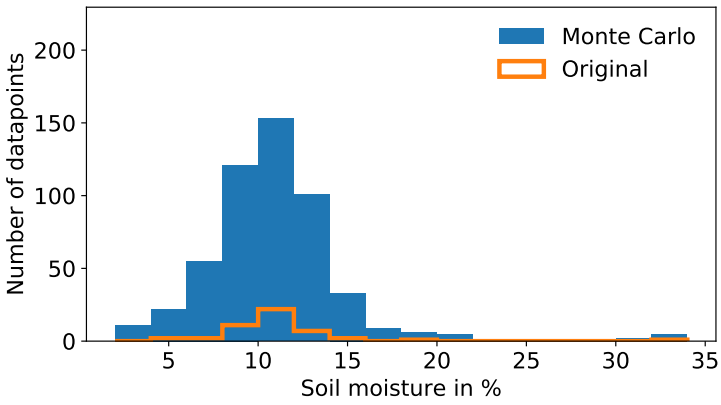


Figure C.5: Monte Carlo (MC) histogram of measurement area A_3. Only the 48 datapoints of A_3 are included with a soil moisture value below 40 %. Ten MC pseudo datapoints are generated for every datapoint with a standard deviation of 2 %.

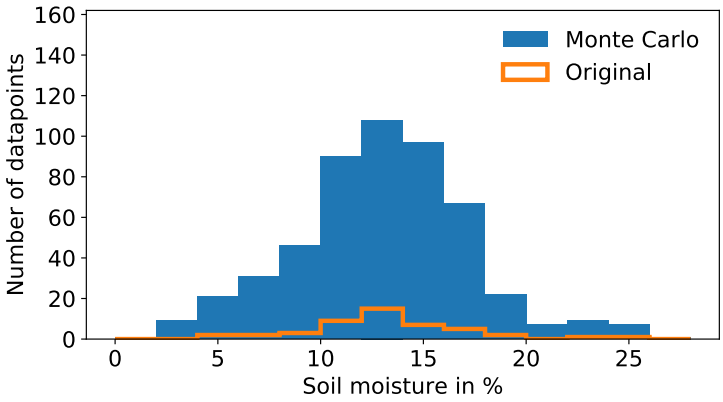
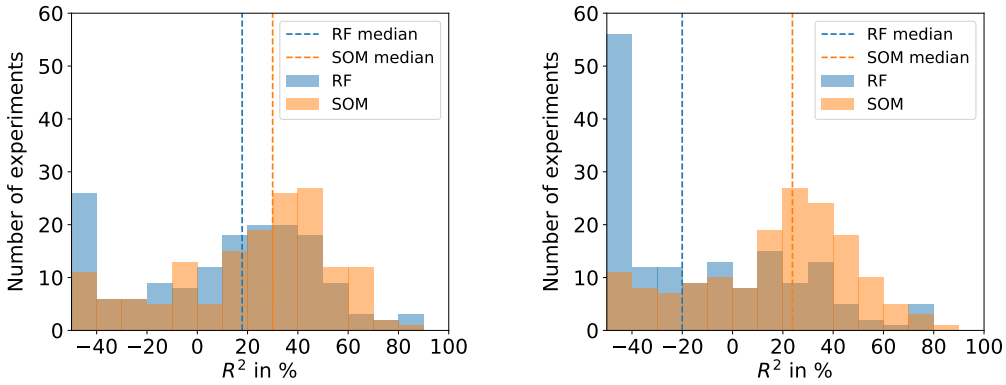


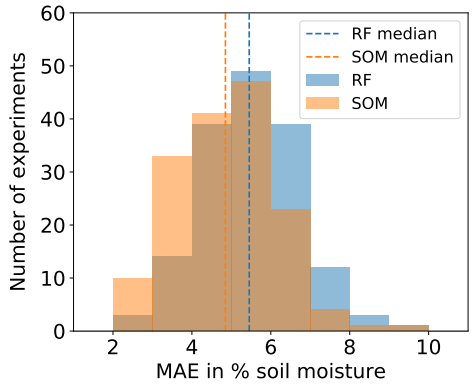
Figure C.6: MC histogram of measurement area A_5. Only the 47 datapoints of A_5 are included with a soil moisture value below 40 %. Ten MC pseudo datapoints are generated for every datapoint with a standard deviation of 2 %.

C.3.3 Regression Results

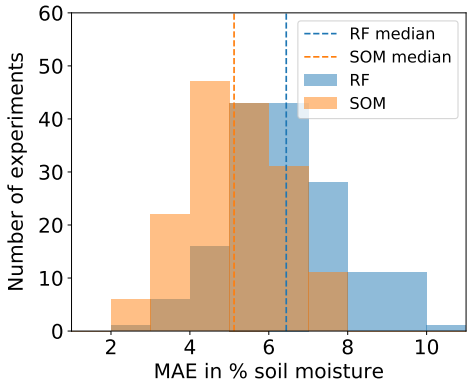


(a) R^2 of original dataset

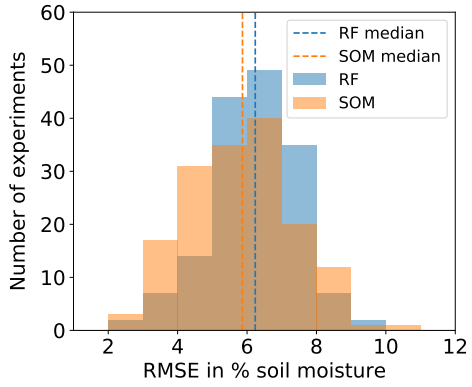
(b) R^2 of MC dataset



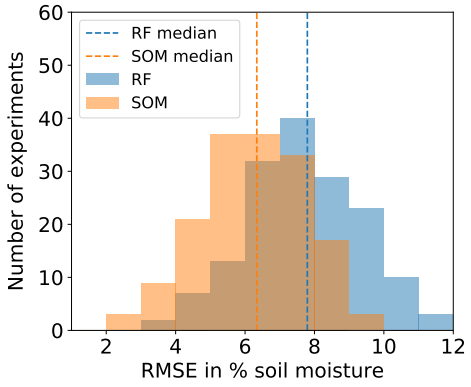
(c) MAE of original dataset



(d) MAE of MC dataset

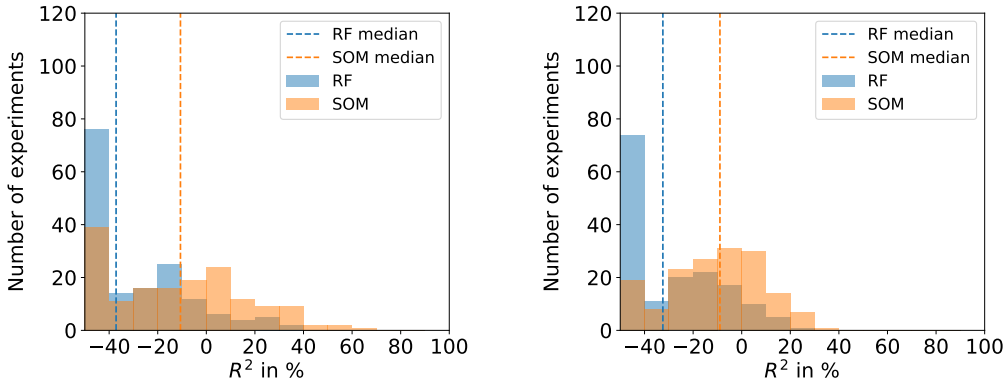


(e) RMSE of original dataset



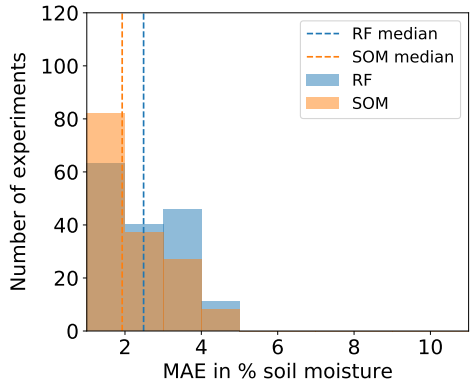
(f) RMSE of MC dataset

Figure C.7: Estimation results of the supervised regressors Random Forest (RF) and Self-Organizing Map (SOM) based on the data of measurement area A_1. The results are illustrated as histograms of the three evaluation metrics R^2, MAE and RMSE with and without MC augmented data. (a) R^2 of the original dataset and (b) R^2 with MC data augmentation. The left-most bin is an overflow bin, collecting all values $R^2 < -0.5\,\%$. (c) MAE of the original dataset, (d) MAE with MC data augmentation, (e) RMSE of the original dataset and (f) RMSE with MC data augmentation.

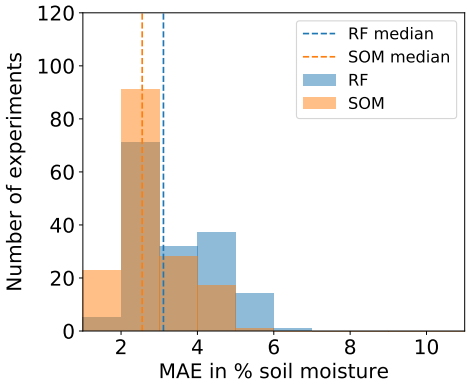


(a) R^2 of original dataset

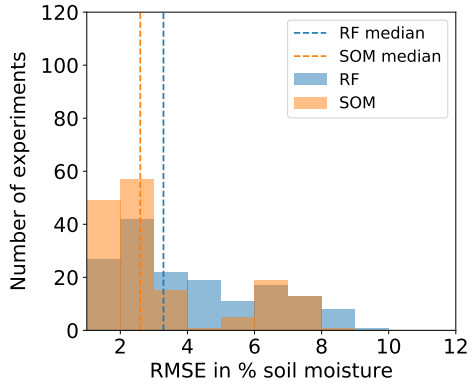
(b) R^2 of MC dataset



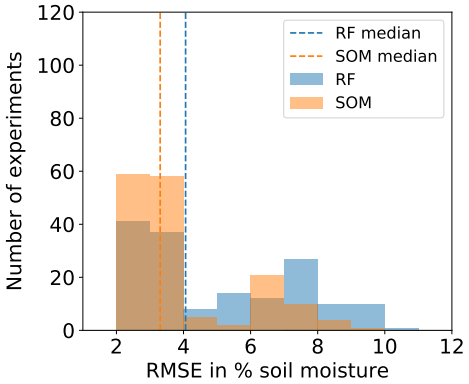
(c) MAE of original dataset



(d) MAE of MC dataset

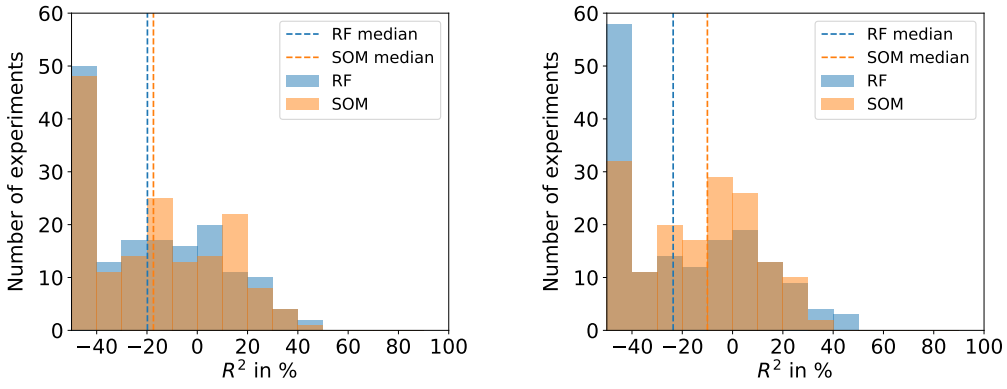


(e) RMSE of original dataset



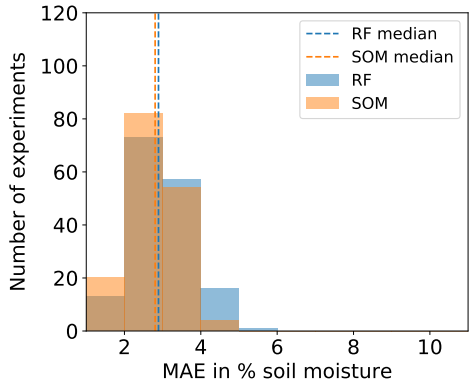
(f) RMSE of MC dataset

Figure C.8: Estimation results of the supervised regressors RF and SOM based on the data of measurement area A_3. The results are illustrated as histograms of the three evaluation metrics R^2, MAE and RMSE with and without MC augmented data. (a) R^2 of the original dataset and (b) R^2 with MC data augmentation. The left-most bin is an overflow bin, collecting all values $R^2 < -0.5\,\%$. (c) MAE of the original dataset, (d) MAE with MC data augmentation, (e) RMSE of the original dataset and (f) RMSE with MC data augmentation.

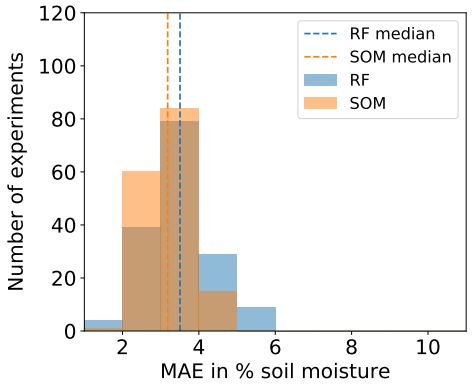


(a) R^2 of original dataset

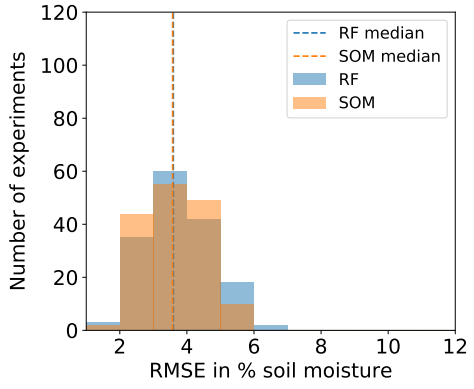
(b) R^2 of MC dataset



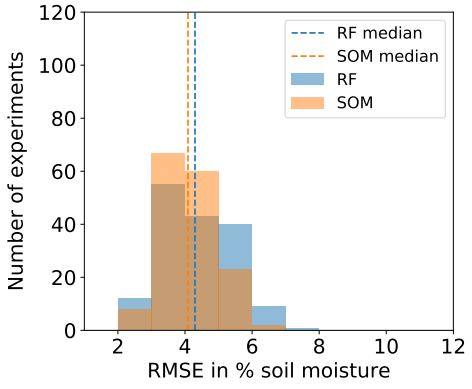
(c) MAE of original dataset



(d) MAE of MC dataset

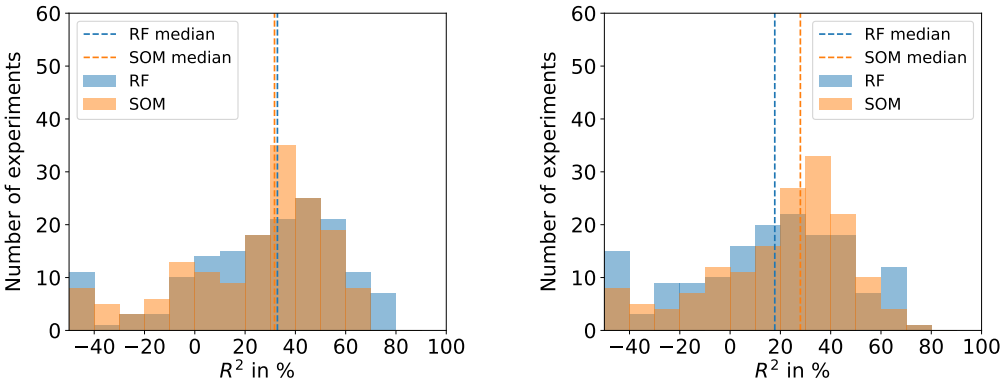


(e) RMSE of original dataset



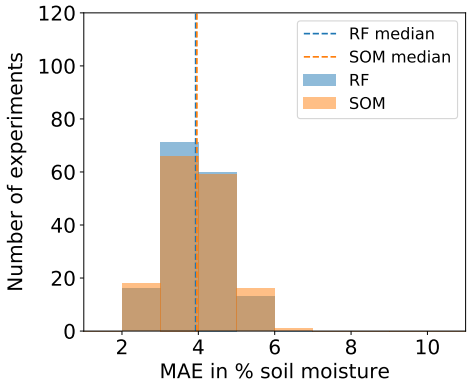
(f) RMSE of MC dataset

Figure C.9: Estimation results of the supervised regressors RF and SOM based on the data of measurement area A_5. The results are illustrated as histograms of the three evaluation metrics R^2, MAE and RMSE with and without MC augmented data. (a) R^2 of the original dataset and (b) R^2 with MC data augmentation. The left-most bin is an overflow bin, collecting all values $R^2 < -0.5\,\%$. (c) MAE of the original dataset, (d) MAE with MC data augmentation, (e) RMSE of the original dataset and (f) RMSE with MC data augmentation.

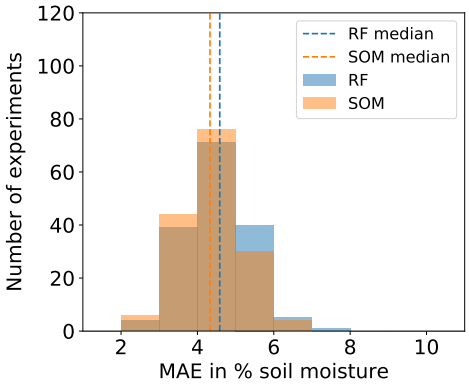


(a) R^2 of original dataset

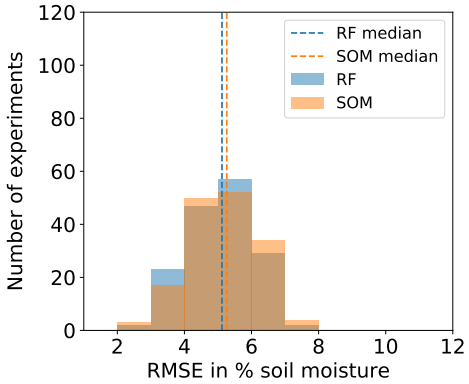
(b) R^2 of MC dataset



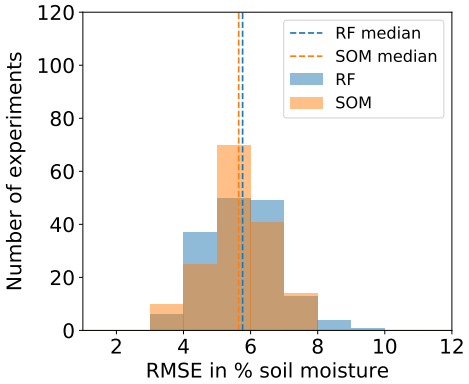
(c) MAE of original dataset



(d) MAE of MC dataset

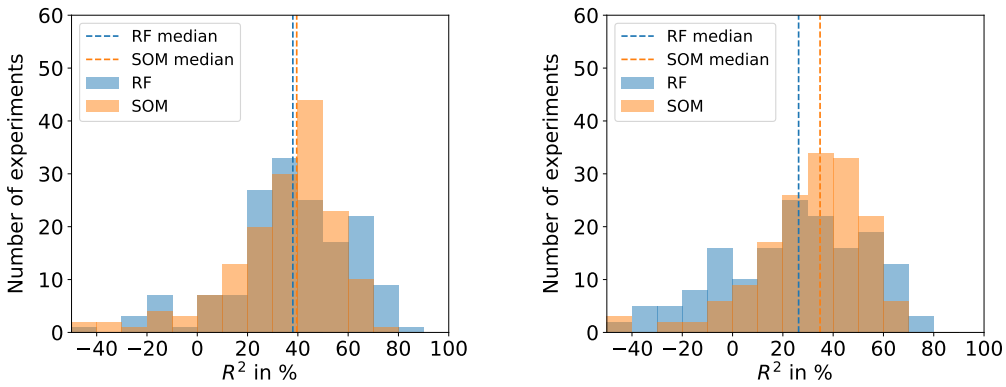


(e) RMSE of original dataset



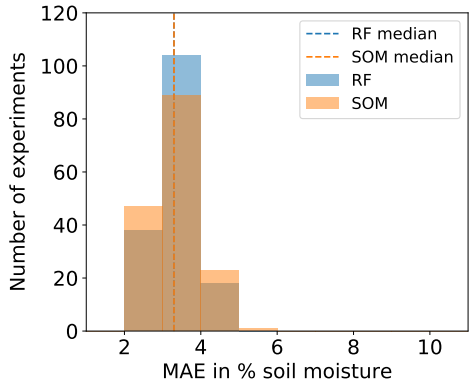
(f) RMSE of MC dataset

Figure C.10: Estimation results of the supervised regressors RF and SOM based on the data of the combination of measurement areas $A_{1,5}$. The results are illustrated as histograms of the three evaluation metrics R^2, MAE and RMSE with and without MC augmented data. (a) R^2 of the original dataset and (b) R^2 with MC data augmentation. The left-most bin is an overflow bin, collecting all values $R^2 < -0.5\,\%$. (c) MAE of the original dataset, (d) MAE with MC data augmentation, (e) RMSE of the original dataset and (f) RMSE with MC data augmentation.

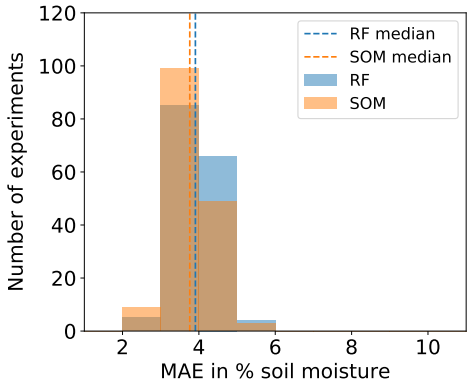


(a) R^2 of original dataset

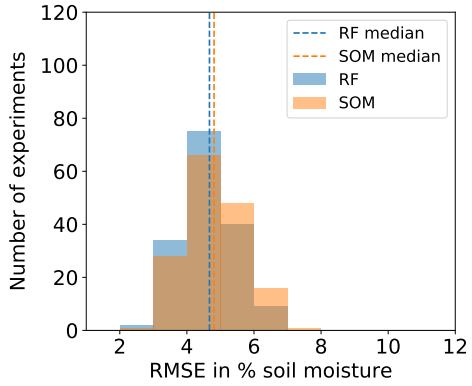
(b) R^2 of MC dataset



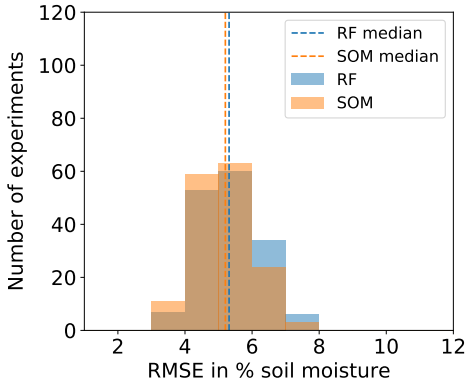
(c) MAE of original dataset



(d) MAE of MC dataset

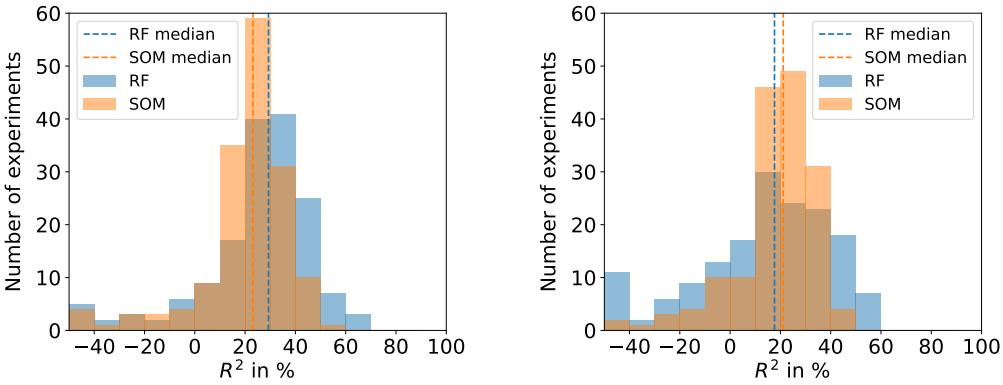


(e) RMSE of original dataset



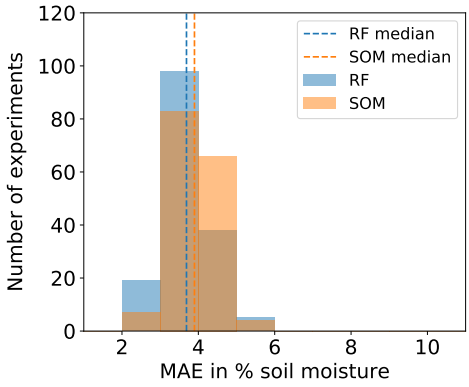
(f) RMSE of MC dataset

Figure C.11: Estimation results of the supervised regressors RF and SOM based on the data of the combination of measurement areas $A_{1,3,5}$. The results are illustrated as histograms of the three evaluation metrics R^2, MAE and RMSE with and without MC augmented data. (a) R^2 of the original dataset and (b) R^2 with MC data augmentation. The left-most bin is an overflow bin, collecting all values $R^2 < -0.5\,\%$. (c) MAE of the original dataset, (d) MAE with MC data augmentation, (e) RMSE of the original dataset and (f) RMSE with MC data augmentation.

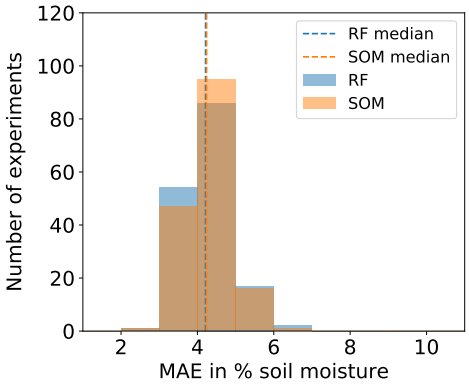


(a) R^2 of original dataset

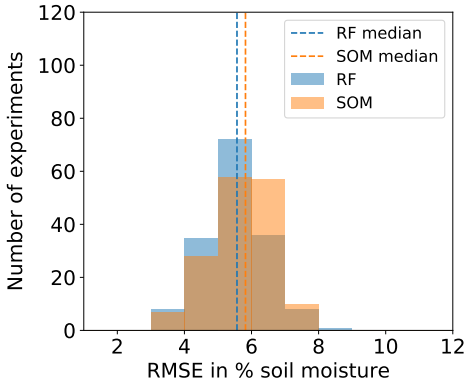
(b) R^2 of MC dataset



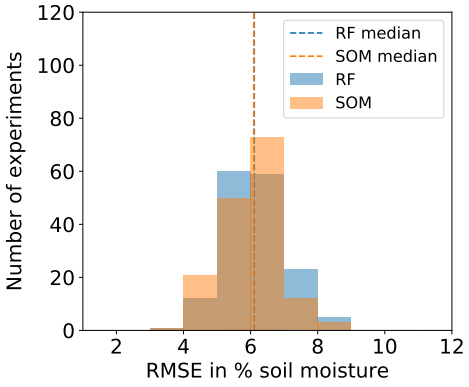
(c) MAE of original dataset



(d) MAE of MC dataset



(e) RMSE of original dataset



(f) RMSE of MC dataset

Figure C.12: Estimation results of the supervised regressors RF and SOM based on the data of the combination of all five measurement areas. The results are illustrated as histograms of the three evaluation metrics R^2, MAE and RMSE with and without MC augmented data. (a) R^2 of the original dataset and (b) R^2 with MC data augmentation. The left-most bin is an overflow bin, collecting all values $R^2 < -0.5\,\%$. (c) MAE of the original dataset, (d) MAE with MC data augmentation, (e) RMSE of the original dataset and (f) RMSE with MC data augmentation.

Colophon

This thesis was typeset with LaTeX 2_ε. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.