Computer Vision Applications for Spectral Imaging

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
M.Sc. Leon Amadeus Varga
aus Herrenberg

Tübingen
2023

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	01.02.2024
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter/-in:	Prof. Dr. rer. nat. Andreas Zell
2. Berichterstatter/-in:	Prof. Dr. -Ing. Andreas Geiger

Abstract

The success of deep-learning-based algorithms significantly boosted the performance of computer vision methods based on color images in recent years. The development of hyperspectral and multispectral camera systems allowed many new applications of spectral imaging. Still, the combination of both, computer vision methods and spectral imaging, are in its infancy, especially since deep-learning-based approaches are not well-established for spectral imaging.

In this work, recent computer vision developments are applied to different spectral imaging tasks. Four challenges for the algorithms (lack of data sets, task-specific features, complicated data augmentation, and large channel dimension) are identified and tackled.

In the first part of this work, a simple convolutional neural network is proposed and evaluated on two hyperspectral imaging applications in food inspection. In this context, a data set of ripening fruit is introduced, which is used throughout the rest of the work. In the second part, self-supervised pretraining for hyperspectral imaging is introduced based on the example of three state-of-the-art contrastive learning methods (SimCLR, SimSiam, Barlow Twins). Some modifications, like data augmentations, are required for this. Afterward, the main contribution of this paper, a wavelength-aware 2D convolution for hyperspectral imaging, is proposed. The key idea of the method is the introduced bias "Similiar wavelengths show similar features". This bias leads to a significant trainable parameter reduction and supports the training of camera-agnostic models. The last part of this work discusses and evaluates the usefulness of multispectral cameras for maritime search and rescue missions. Therefore, a data set with humans in open water was recorded and published. In this context, a method is presented which can reduce the background bias, a problem of these remote sensing recordings. In the end, the work is concluded with a summary, a short discussion, and an outlook.

None of the defined challenges were fully overcome. Still, the presented approaches show how a solution could look and prepare future research in these directions.

Kurzfassung

Der Erfolg von Deep-Learning-basierten Algorithmen hat in den letzten Jahren die Leistung von Bildverarbeitungsmethoden auf der Grundlage von Farbbildern erheblich verbessert. Die Entwicklung von hyperspektralen und multispektralen Kameras ermöglichte neue Anwendungsgebiete für die spektrale Bildverarbeitung. Dennoch steckt die Kombination von Computer-Vision-Methoden und Spectral Imaging noch in den Kinderschuhen, zumal Deep-Learning-basierte Ansätze für Spectral Imaging noch nicht vollständig etabliert sind.

Jüngste Entwicklungen im Bereich der Computer Vision werden in dieser Arbeit für verschiedene Anwendungen der Spectral Imaging erprobt. Vier Herausforderungen für die Algorithmen (fehlende Datensätze, aufgabenspezifische Merkmale, komplizierte Daten-Erweiterung und große Channeldimension) werden identifiziert und angegangen.

Im ersten Teil dieser Arbeit wird ein einfaches Convolutional Neural Network vorgestellt und auf zwei Hyperspectral-Imaging-Anwendungen aus der Lebensmittelkontrolle evaluiert. In diesem Zusammenhang wird ein Datensatz von reifenden Früchten vorgestellt, der auch im weiteren Verlauf der Arbeit verwendet wird. Im zweiten Teil wird das self-supervised Pretraining für Hyperspectral Imaging am Beispiel von drei modernen Contrastive-Learning-Methoden (SimCLR, SimSiam, Barlow Twins) vorgestellt. Einige Modifikationen, wie z.B. Data Augmentation, sind hierfür notwendig. Danach wird der Hauptbeitrag dieser Arbeit, eine wellenlängenbasierte 2D-Convolution für die hyperspektrale Bildgebung, vorgestellt. Die Kernidee der Methode ist der eingeführte Bias "Ähnliche Wellenlängen zeigen ähnliche Merkmale". Dieser Bias führt zu einer signifikanten Reduktion der trainierbaren Parameter und unterstützt das Training von kamera-agnostischen Modellen. Im letzten Teil dieser Arbeit wird die Tauglichkeit von Multispektralkameras für maritime Such- und Rettungseinsätze diskutiert und evaluiert. Dazu wurde ein Datensatz mit Menschen im Wasser aufgenommen und veröffentlicht. In diesem Zusammenhang wird eine Methode vorgestellt, mit der Background-Bias, ein Problem bei diesen Fernerkundungsaufnahmen, reduziert werden kann. Am Ende wird die Arbeit mit einer Zusammenfassung, einer kurzen Diskussion und einem Ausblick abgerundet.

Keine der definierten Herausforderungen konnte vollständig bewältigt werden. Dennoch zeigen die vorgestellten Ansätze, wie eine Lösung aussehen könnte und bereiten zukünftige Forschung in diese Richtung vor.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Contributions . . . 2
- 1.3 Outline . . . 5

2 Foundations **7**
- 2.1 Spectral Imaging . . . 8
  - 2.1.1 Hyperspectral Imaging . . . 8
  - 2.1.2 Multispectral Imaging . . . 11
  - 2.1.3 Challenges of Spectral Imaging . . . 11
- 2.2 Data Sets . . . 13
- 2.3 Hyperspectral Measurement System . . . 15
- 2.4 Multispectral Cameras . . . 16
- 2.5 Drone Platforms . . . 17

3 Supervised Learning for Hyperspectral Classification **19**
- 3.1 Related Work . . . 20
- 3.2 Models . . . 21
  - 3.2.1 k-Nearest-Neighbors . . . 21
  - 3.2.2 Support Vector Machine . . . 21
  - 3.2.3 ResNet-18 . . . 21
  - 3.2.4 DeepHSNet . . . 22
- 3.3 Measuring the Ripeness of Fruit . . . 23
  - 3.3.1 Related Work for Fruit Inspection . . . 23
  - 3.3.2 Hyperspectral Imaging . . . 24
  - 3.3.3 Fruit Ripening . . . 24
  - 3.3.4 Data Set . . . 26
  - 3.3.5 Experiments . . . 27
  - 3.3.6 Ablation Study . . . 30
  - 3.3.7 Investigation of the Learned CNN Features . . . 32
  - 3.3.8 Visualization of the Ripening Process . . . 33
  - 3.3.9 Conclusion . . . 35
- 3.4 Food Inspection for Soybean (*Glycine max*) . . . 35
  - 3.4.1 Materials and Methods . . . 36
  - 3.4.2 Results . . . 39

3.4.3 Discussion . . . 42
3.4.4 Conclusion . . . 45
3.5 Summary . . . 45

4 Self-Supervised Learning for Hyperspectral Classification 47
4.1 Introduction . . . 47
4.2 Related Work . . . 48
4.3 Experiments . . . 48
4.3.1 Data Set . . . 49
4.3.2 Models . . . 49
4.3.3 Self-supervised Pretraining . . . 50
4.3.4 Evaluation . . . 52
4.4 Results . . . 53
4.5 Ablation Study . . . 55
4.5.1 Classifier Model . . . 55
4.5.2 Self-supervised Pretraining Method . . . 56
4.5.3 Augmentations . . . 58
4.6 Semi-supervised Pretraining . . . 60
4.7 Conclusion . . . 63

5 Wavelength-aware 2D Convolutions for Hyperspectral Imaging 65
5.1 Introduction . . . 65
5.2 Related Work . . . 67
5.3 Proposed Method . . . 68
5.3.1 Motivation . . . 68
5.3.2 Method . . . 70
5.3.3 Mapping-Based Baselines . . . 74
5.4 Experiments . . . 74
5.4.1 Application A: Fruit Ripeness Prediction . . . 74
5.4.2 Application B: Satellite Imagery Segmentation . . . 80
5.5 Ablation Study . . . 83
5.5.1 Interpretation of the WROIs . . . 83
5.5.2 Impact of Parameter G . . . 84
5.5.3 Training of the Gaussian Distributions . . . 85
5.5.4 Impact of the Method Extension . . . 86
5.6 Conclusion . . . 86

6 UAV-Based Applications of Multispectral Imaging 89
6.1 Introduction . . . 89
6.2 Generation of a Maritime Benchmark for Detecting Humans in Water . . . 90
6.2.1 Related Work . . . 91
6.2.2 Data Set Generation . . . 92

6.2.3 Data Set Task . . . 93
6.2.4 Evaluation . . . 94
6.2.5 Extension of the Data Set (SeaDronesSeev2) . . . 94
6.2.6 Conclusion . . . 95
6.3 Comprehensive Analysis of the Object Detection Pipeline on UAVs . . . 96
6.3.1 Related Work . . . 97
6.3.2 Materials and Methods . . . 98
6.3.3 Experiment Setup . . . 98
6.3.4 Results . . . 101
6.3.5 Discussion . . . 103
6.3.6 Conclusion . . . 104
6.4 Tackling the Background Bias of Remote Sensing Object Detection . . . 105
6.4.1 Introduction . . . 105
6.4.2 Related Work . . . 106
6.4.3 Method: Cropping Window (CroW) . . . 108
6.4.4 Experiment . . . 108
6.4.5 Results . . . 109
6.4.6 Evaluation for Multispectral Recordings . . . 110
6.4.7 Conclusion . . . 111
6.5 Summary . . . 111

7 Conclusions **113**
7.1 Summary . . . 113
7.2 Discussion . . . 114
7.3 Future Work . . . 116

A Appendices **117**
A.1 Spectral Signatures of Noticeable Features for Soybean . . . 118
A.2 Full List of Augmentations Tested for SSL . . . 119
A.3 HRSS Results with Standard Deviation . . . 120
A.4 Additional Examples of a learned Camera Filters with HyveConv . . . 121

Bibliography **127**

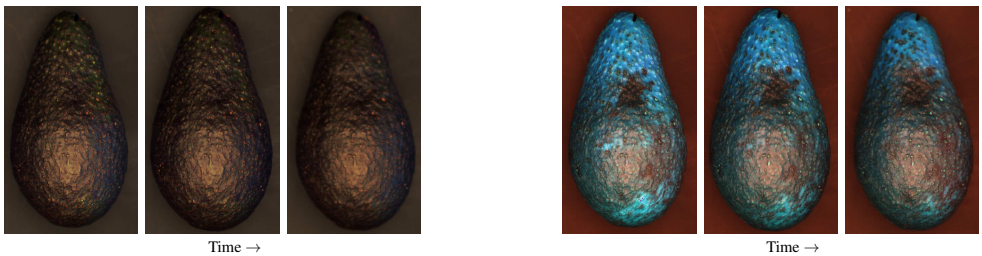
Chapter 1

Introduction

1.1 Motivation

The recent technological development of hyperspectral and multispectral cameras leads to many new applications of spectral imaging. By providing information about wavelengths outside the human-visible spectrum ($\approx$ 400 - 700 nm), these systems allow decisions that are not possible with purely human-based perception (e.g., see Fig. 1.1). Recent developments in computer vision algorithms utilizing neural networks allowed the surpassing of human experts in restricted classification tasks (He *et al.*, 2015). The trend towards deep learning enlivened research in many computer vision areas. Algorithms, which seemed impossible ten years ago, are now within reach. Still, the straightforward combination of spectral imaging and deep learning receives little attention. Through this combination, however, superhuman vision seems possible.

The emerging trend towards spectral imaging is visible in many industry sectors, especially those that can be expected to grow in the near future (e.g., recycling, food inspection, and non-invasive medical applications). In this work, we present applications of spectral imaging and review current algorithmic computer vision developments. We show how hyperspectral imaging can be used to optimize the food chain, supporting the trend towards precise farming, and we present how multispectral cameras could support maritime search and rescue missions.



(a) Human perception (b) Near Infrared

Figure 1.1: Recordings of an avocado on different days. In the **(a)** color images no change is visible. Only the **(b)** false-color images ($\approx$ 700 - 1000 nm) show a growing dark area, which correlates with the ripeness of the fruit.

1.2 Contributions

This thesis contributes to the research community with the following peer-reviewed works. The asterisk (*) indicates equal contributions:

1. Leon Amadeus Varga, Jan Makowski and Andreas Zell. **"Measuring the Ripeness of Fruit with Hyperspectral Imaging and Deep Learning"**, IEEE International Joint Conference on Neural Networks (IJCNN), 2021.

   We present a system to measure the ripeness of fruit with a hyperspectral camera and a suitable deep neural network architecture. This architecture did outperform competitive baseline models on the prediction of the ripeness state of fruit. For this, we recorded a data set of ripening avocados and kiwis. Furthermore, a technique is introduced to visualize the ripening process.

2. Leon Amadeus Varga and Andreas Zell. **"Tackling the Background Bias in Sparse Object Detection via Cropped Windows"**, IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, 2021.

   Object detection on unmanned aerial vehicles is still a challenging task. The recordings are mostly sparse and contain only small objects. In this work, we propose a simple tiling method that improves the detection capability in the remote sensing case. By reducing the background bias and enabling the usage of higher image resolutions during training, our method can improve the performance of models substantially. The procedure was validated on three different data sets and outperformed similar approaches in performance and speed.

3. Leon Amadeus Varga*, Benjamin Kiefer*, Martin Messmer*, and Andreas Zell. **"SeaDronesSee: A Maritime Benchmark for Detecting Humans in Open Water"**, IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2022.

   Unmanned aerial vehicles are of crucial importance in search and rescue missions in maritime environments. Modern computer vision algorithms are of great interest in aiding such missions. However, they are dependent on large amounts of real-case training data from UAVs, which is only available for traffic scenarios on land. Therefore, this paper introduces a large-scaled visual object detection and tracking benchmark (SeaDronesSee) aiming to bridge the gap from land-based vision systems to sea-based ones. In addition, we are providing meta information for altitude, viewing angle and other meta data. Multiple state-of-theart computer vision algorithms were evaluated on this newly established benchmark serving as baselines. We provide an evaluation server where researchers can upload their prediction and compare their results on a central leaderboard.

4. Leon Amadeus Varga, Sebastian Koch and Andreas Zell. **"Comprehensive Analysis of the Object Detection Pipeline on UAVs"**, MDPI Remote Sensing Journal 14, no. 21, 2022.

   The quality of the images directly affects the performance of object detectors. This paper aims to tune the detection throughput and accuracy of existing object detectors in the remote sensing scenario by optimizing the input images tailored to the object detector. We empirically analyze the influence of two selected camera calibration parameters (camera distortion correction and gamma correction) and five image parameters (quantization, compression, resolution, color model, and additional channels) for these applications. We show that not all parameters have an equal impact on detection accuracy and data throughput. Using a suitable compromise between parameters, we can achieve higher detection accuracy for lightweight object detection models while keeping the same data throughput.

5. Leon Amadeus Varga, Martin Messmer, Nuri Benbarka and Andreas Zell. **"Wavelength-Aware 2D Convolutions for Hyperspectral Imaging"**, IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023.

   Deep Learning could drastically boost the classification accuracy for hyperspectral imaging. Still, the training on the mostly small hyperspectral data sets is not trivial. Two key challenges are the large channel dimension of the recordings and the incompatibility between cameras of different manufacturers. By introducing a suitable model bias and continuously defining the channel dimension, we propose a 2D convolution optimized for these challenges of hyperspectral imaging. Besides the shown superiority of the model, the modification adds additional explanatory power. In addition, the model learns the necessary camera filters in a data-driven manner.

6. Leon Amadeus Varga*, Hannah Frank* and Andreas Zell. **"Self-supervised Pretraining for Hyperspectral Classification of Fruit Ripeness"**, Internationale Konferenz zur Optischen Charakterisierung von Materialien (OCM), 2023.

   The ripeness of fruit can be measured in a non-destructive way using hyperspectral imaging and deep learning methods. However, the lack of labeled data samples limits hyperspectral image classification. This work explores self-supervised learning (SSL) as pretraining for HSI classification of fruit ripeness. State-of-the-art SSL methods are implemented, and augmentation techniques for HSI are developed. A 3D-2D hybrid convolutional network is proposed to support the pretraining procedure. The pretraining is evaluated on the fruit ripeness prediction task. This work shows that it is possible to transfer the ideas of SSL to HSI. Pretraining stabilizes classifier training and improves the classifier performance.

Further, it can partially compensate for the need for large labeled data sets in HSI classification.

7. Stefan Thomas, Leon Amadeus Varga, Nico Harter, Andreas Zell and Ralf Voegele. **"Detection of Phakopsora pachyrhizi infestation in soybean via hyperspectral imaging and data analysis"** Nature Scientific Report Journal (under review).

   Phakopsora pachyrhizi, the causative agent of the Asian Soybean Rust, is one of the most prominent causes of yield loss in soybean production worldwide. In this study, a combination of hyperspectral imaging with advanced data analysis methodology is shown to accurately detect soybean rust symptoms in early stages and differentiate them from other factors at leaf scale. Results show that even with a comparably small subset of training data the analysis through neural networks can outperform classical machine learning methods and human experts in the early stage detection of infection.

In addition, our latest work, which discusses a similiar topic and was prepared during the research of this dissertation, could not be included here due to time constraints:

8. Hannah Frank*, Leon Amadeus Varga* and Andreas Zell. **"Hyperspectral Benchmark: Bridging the Gap between HSI Applications through Comprehensive Dataset and Pretraining"** International Journal of Computer Vision (under review).

   Hyperspectral Imaging serves as a non-destructive spatial spectroscopy technique with a multitude of potential applications. However, a recurring challenge lies in the limited size of the target datasets, impeding exhaustive architecture search. Consequently, when venturing into novel applications, reliance on established methodologies becomes commonplace, in the hope that they exhibit favorable generalization characteristics. Regrettably, this optimism is often unfounded due to the fine-tuned nature of models tailored to specific HSI contexts. To address this predicament, this study introduces an innovative benchmark dataset encompassing three markedly distinct HSI applications: food inspection, remote sensing, and recycling. Furthermore, the enhanced diversity inherent in the benchmark dataset underpins the establishment of a pretraining pipeline for HSI.

1.3 Outline

This thesis presents different applications of *spectral imaging (SI)* and computer vision. We evaluate deep-learning-based algorithms for *hyperspectral imaging (HSI)* and *multispectral imaging (MSI)* in different scenarios, e.g., under laboratory conditions or open fielad applications.

In chapter 2, we prepare the foundations for the thesis and discuss spectral imaging with the key challenges for computer vision. Hyperspectral imaging and multispectral imaging are forms of spectral imaging. We will mostly focus on HSI as this is more different from Color Imaging and, therefore, more challenging. At the end of this chapter, we present the used hardware (e.g., cameras, hyperspectral measurement system, and drone platforms) utilized throughout the rest of the work.

In chapter 3, supervised learning for hyperspectral image classification is discussed with two examples in the area of food inspection. Besides acquiring the data sets for these applications, a shallow *Convolutional Neural Network (CNN)*, called DeepHSNet, is presented, which could outperform comparable approaches for these two applications. Further, this chapter presents how HSI and machine learning can be utilized to build non-destructive measurement systems for the upcoming trend of Precision Farming. The network architecture DeepHSNet is the basis for further developments in this thesis. Also, the recorded data set of ripening fruit will be used for further evaluation.

This chapter is based on our publications:

- Varga, L. A., Makowski, J., and Zell, A. (2021). Measuring the ripeness of fruit with hyperspectral imaging and deep learning. In *International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021*, pages 1–8. IEEE,

- Thomas, S., Varga, L. A., Harter, N., Zell, A., and Voegele, R. T. (submitted). Detection of phakopsora pachyrhizi infestation in soybean via hyperspectral imaging and data analysis. *Nature Scientific reports*.

A challenge of spectral imaging discussed in chapter 2, is the small labeled data sets, as labels are often costly. In chapter 4, the recent trend of *Self-supervised Learning (SSL)* based on *Constrative Learning* is reviewed and adapted for HSI. Here, we show that self-supervised Pretraining stabilizes the training of larger neural network models for the small HSI data sets. Further, the impact of different data augmentation techniques on the HSI classification was evaluated.

This is based on our publication:

- Varga, L. A., Frank, H., and Zell, A. (2023a). Self-supervised pretraining for hyperspectral classification of fruit ripeness. In J. Beyerer, T. Längle, and M. Heizmann, editors, *OCM 2023 - Optical Characterization of Materials : Conference Proceedings*, pages 97–108.

In chapter 5, we propose a wavelength-aware 2D-Convolution for HSI. The model bias "similar wavelengths show similar features" allows a significant parameter reduction. This proposed method is our key contribution as it combines our main assumption that neural networks can handle the large channel dimension of a hyperspectral cube without dimension reduction with a suitable bias. With reduced model complexity, the model can still outperform comparable models and generalizes significantly better on unseen hyperspectral cameras. The straightforward definition of the learned wavelength ranges as Gaussian-based camera filters makes the interpretation of the channel dimension handling more intuitive. The learned camera filters could be the basis for a multispectral camera for the specific application. This chapter is based on our publication:

- Varga, L. A., Messmer, M., Benbarka, N., and Zell, A. (2023b). Wavelength-aware 2d convolutions for hyperspectral imaging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3788–3797.

Hyperspectral cameras often require a laboratory setup, reasoned by the line-scan acquisition mode and the light source. Multispectral cameras, in contrast, usually are more robust with light, are cheaper, and support imaging acquisition mode. Therefore, they fit better for most applications. In chapter 6, we show multispectral and color cameras mounted on drones could be used for maritime *search and rescue (SAR)* missions. We describe the data set generation of SeaDronesSee and the extension SeaDronesSeev2. Further, we evaluate the impact of selected camera parameters, focusing on the camera type (e.g., grayscale, color, or multispectral) selection, on the performance of an object detection pipeline in the remote sensing application. In this context, we also evaluate the advantage of MSI for maritime SAR. Finally, we discuss a common problem of object detection in remote sensing, namely the background bias, and propose an easy-to-implement method that reduces its impact. This chapter is based on our publications:

- Varga, L. A., Kiefer, B., Messmer, M., and Zell, A. (2022b). SeaDronesSee: A maritime benchmark for detecting humans in open water. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pages 3686–3696. IEEE,

- Varga, L. A., Koch, S., and Zell, A. (2022a). Comprehensive analysis of the object detection pipeline on UAVs. *Remote Sensing*, **14**(21),

- Varga, L. A. and Zell, A. (2021). Tackling the background bias in sparse object detection via cropped windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 2768–2777.

In the end, chapter 7 summarizes and discusses the achieved results and provides an outlook on ongoing and possible future research based on the presented work.

Chapter 2

Foundations

In this chapter, we will discuss the foundations of this work, which are necessary for further chapters. First, we describe *spectral imaging (SI)* via the examples *hyperspectral imaging (HSI)* and *multispectral imaging (MSI)*. This also includes the particular challenges of these recordings. Afterward, we introduce the used data sets, which are publicly available or were published by us. Finally, we present the utilized hardware, which covers the hyperspectral measurement system, camera systems, and drone platforms.

It is assumed that the reader is familiar with the foundations of neural networks. Therefore these will not be explained here.

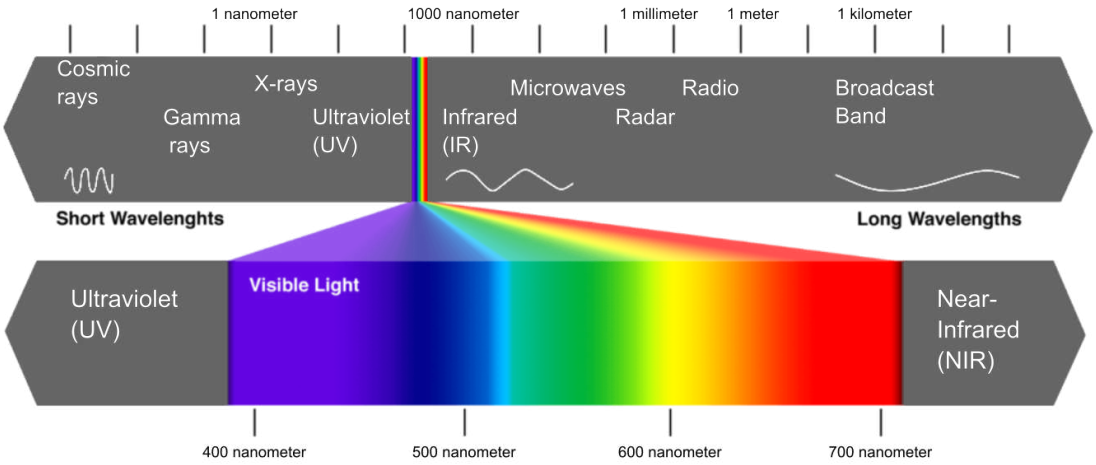


Figure 2.1: Spectrum ranges. Modified graphic based on the figure of Elite Optoelectronics Co. (2020).

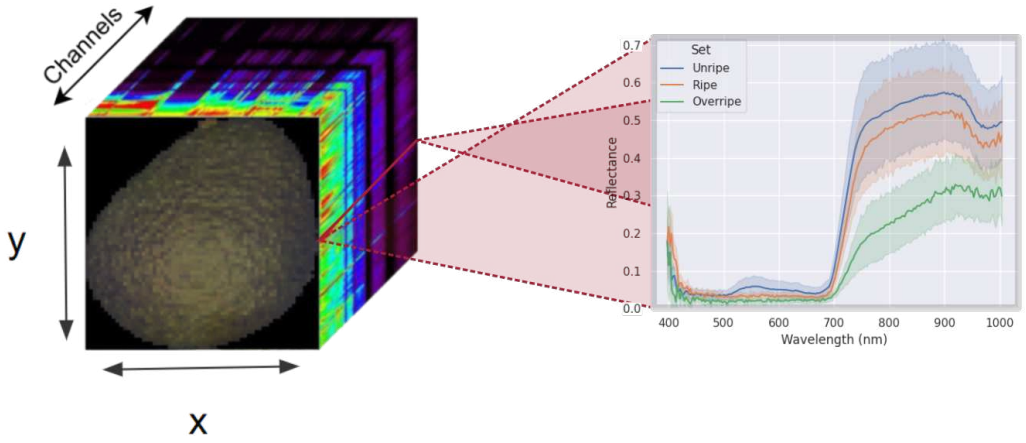
2.1 Spectral Imaging

Color and grayscale recordings are the most common form of images. Nearly every smartphone allows capturing color images within seconds. These recordings cover only the (for human) visible light (between 380 to about 750 nm).

Spectral imaging tries to mimic the underlying spectrum for each pixel, as discussed by Robles-Kelly and Huynh (2013). Therefore many more channels are required. Further, these recordings typically include channels outside the visible light, which can be informative in some applications.

The wavelength ranges are categorized, as shown in Fig. 2.1. Especially, the *Near-infrared (NIR)* range from around 750 nm to 2500 nm, and the *Near-ultraviolet (NUV)* range from 300 nm to 400 nm are interesting for spectral imaging. Besides, the X-Ray range is essential for some applications but will not be considered in this work.

Spectral imaging is usually achieved with at least one band outside the visible range. Hyperspectral imaging and multispectral imaging are forms of spectral imaging, which differ in the number of channels. As multispectral imaging has typically between 5 and 10 channels, hyperspectral imaging incorporates around 200 channels. The different channel numbers have advantages and disadvantages, which will be discussed in the following.

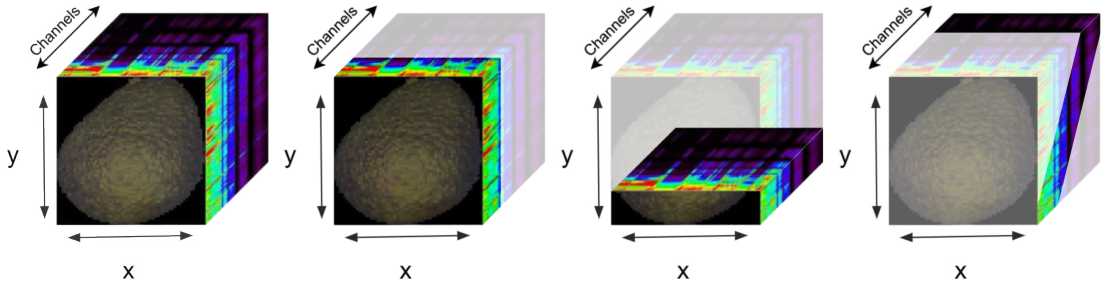


(a) Hyperspectral cube (b) Spectra of ripening Avocados

Figure 2.2: A hyperspectral cube of a ripening avocado and spectra of different ripening states. Each pixel of a hyperspectral cube is defined by a spectrum.

2.1.1 Hyperspectral Imaging

Hyperspectral cameras record around 200 channels. With this number of channels, they can sufficiently approximate the spectrum in a restricted range. Therefore, hyperspectral



(a) Snapshot (b) Spectral (c) Spatial (d) Spatial-Spectral

Figure 2.3: Visualization of the different scanning approaches

imaging can be interpreted as spatial spectroscopy. The only difference is that spectroscopy often covers a more extensive wavelength range with a higher spectral resolution. The product of HSI is often called a hyperspectral cube, as shown in Fig. 2.2.

Like a color image, the cube has two spatial (x and y) and one channel dimension (λ). The difference is that the channel dimension is larger, which leads to the cubic form instead of the color image plane. This kind of data raises a couple of handling challenges, which we will discuss in Sec. 2.1.3.

Besides data handling challenges, data acquisition is more complicated than for color images. As the channel dimension is larger, it is currently hard to capture the whole hyperspectral cube at once, called Snapshot Imaging. There are different approaches to achieving the hyperspectral cube in total, which are visualized in Fig. 2.3.

- **Snapshot Imaging (a)**: Snapshot imaging would be the favored approach. This captures the hyperspectral cube at once. Currently, no cameras in the consumer market are available, allowing Snapshot Imaging for hyperspectral imaging in a reliable way.

- **Spectral Scanning (b)**: Spectral scanning captures the spatial dimension at once but requires scanning the spectral dimension. This approach can be, for example, implemented by switching camera filters.

- **Spatial Scanning (c)**: This approach simultaneously captures the whole spectral dimension but still requires Spatial scanning. These kinds of cameras are called line-scan cameras because they capture a spatial line. For a recording of an entire scene, moving the camera or the object orthogonal to this line is necessary. In practice, a linear actuator or a conveyor belt is used.

- **Spatial-Spectral Scanning (d)**: The last category combines Spatial scanning and Spectral scanning. It captures a part of the spatial and a part of the spectral dimensions at once. This approach is rare and currently more common for prototypes,

as the acquisition is complicated, but it can achieve higher spatial and spectral resolutions.

Most available hyperspectral cameras use spatial scanning as an acquisition technique. In Sec. 3, we also used cameras using this technique for two applications. In the first application, a linear actuator was used to record the whole sample. For the second application, the camera was moved over the recorded object. The resulting output, meaning the hyperspectral cube, is for all techniques the same if we neglect minor recording artifacts.

For HSI, the light source is essential. It has to emit a homogenous light in the whole recorded wavelength range. Otherwise, relevant features could be underexposed or overexposed. For NIR, Halogen lamps produce a reliable light. For UV, LED lamps are the better choice. A diffuse light source, which could be indirect illumination, is preferable as this reduces shadows. Shadows are problematic as these produce mixtures of the underlying spectra and can invalidate the recording. Hyperspectral recordings should be referenced to allow the comparability of the spectra of different recordings. The referencing is achieved with the help of a white reference R_{white} (maximal possible intensity) and a dark reference R_{dark} (minimal possible intensity). Often an average of more than one recording is used as a reference. The normalized hyperspectral cube for the hyperspectral cube C can be computed by:

$$C_{norm} = \frac{C - R_{dark}}{R_{white} - R_{dark}} \tag{2.1}$$

As HSI is a non-destructive measurement and the additional channels can support classification tasks, many applications for HSI were found. Initially, HSI was mainly used for remote sensing with satellites (Baumgardner *et al.*, 2015), as the first hyperspectral cameras were quite expensive. In recent years, new technologies boosted the hyperspectral camera market resulting in cheaper and more practicable cameras. This allowed the usage of HSI in many new industry sectors.

One of these sectors is food inspection (Park and Lu, 2015; Yang, 2011), which we used mainly for our experiments. It covers different areas from precision farming to quality measurements in the food chain. Most of the systems here are inline or infield. It is still a growing sector with a lot of potentials. We selected this for our experiments as it is, in comparison to the other applications, relatively easy to collect samples.

A further promising sector is the medical usage. First studies showed that HSI could support early cancer diagnosis, analysis of diabetic foot, or can guide surgery as discussed by Lu and Fei (2014). As the measurement is not invasive, it could support medical decisions without additional risk.

Further, inline sorting is also an upcoming sector. In every application where the additional spectral information is helpful, HSI is worth considering. Examples are plastic sorting in the recycling industry (Tatzer *et al.*, 2005) or inline quality management of medicine production (Vakili *et al.*, 2015).

In contrast to color images, the applied computer vision algorithms for HSI are mostly

restricted to classification or segmentation tasks, as these are required for these applications. More complicated tasks, like object detection or object tracking, are uncommon as the acquisition procedure of the hyperspectral cameras is not practicable for this.

2.1.2 Multispectral Imaging

In contrast to hyperspectral cameras, multispectral cameras are restricted to five to twenty channels. Therefore, they combine a couple of advantages of color and hyperspectral cameras. Like hyperspectral cameras, they cover wavelength ranges outside the visible spectrum. Since they often operate in Snapshot Imaging mode, no special handling during the acquisition is necessary. For multispectral cameras, the light source is usually less critical since the camera filters of the channels are broader and therefore collect more incident light. In summary, multispectral cameras are more practicable in usage. They can even be used in embedded applications, e.g., mounted on a drone, as shown in chapter 6. Further, the price of multispectral cameras is usually lower than that of hyperspectral cameras.

The main drawback of multispectral cameras is the limited spectral resolution. The necessary spectral resolution depends highly on the applications. For some applications of spectral imaging, a lower spectral resolution is acceptable, and only the correct wavelength ranges are essential. The questions for the necessary wavelength ranges and required spectral resolution should be asked in the conception phase of the acquisition pipeline.

A promising practice is to use HSI in the first step to identify the necessary ranges and spectral resolution. For the final deployment, a multispectral camera can be used based on the findings of the first phase. In chapter 5, we propose a method to support this workflow. With these two phases, it is possible to keep the flexibility of the hyperspectral camera for the evaluation and to provide the practical usability of the multispectral cameras if possible.

In contrast to hyperspectral cameras, multispectral cameras can support, besides classification and segmentation, more complex object detection tasks. Multispectral tracking is still uncommon as many cameras capture with low *Frames per Second (FPS*).

2.1.3 Challenges of Spectral Imaging

In this section, we want to identify challenges that arise with HSI and in minor form with MSI for computer vision methods. In the later chapters, we tackle some of these with our works and provide solutions or an approach for further investigation.

Lack of Data Sets The number of publicly available data sets of hyperspectral recordings is small. Color image classification and detections showed, for example, with ImageNet of Deng *et al.* (2009) and Microsoft COCO of Lin *et al.* (2014b), how large

data sets can boost the research in a specific topic. These data sets propose evaluation procedures, ensuring results' comparability between different approaches. The HSI community lacks this kind of data set. One reason is the still high price of the cameras and the complicated acquisition procedure. Further, the labeling procedure is often time-consuming. As for most applications of HSI, super-human results are wanted, and the labeling usually requires additional tools to support the labeling.

Task-Specific Features For most applications of HSI, the required features are highly task-specific. Features that are important for one application could be meaningless for another. Primarily the wavelength ranges differ widely between different tasks. For color image tasks, at least the kernels of the first backbone layers are often interchangeable, as these support the detection of simple geometry structures (e.g., lines or corners), which are necessary for nearly every more complex object. The pretraining of the large backbones is therefore helpful, as shown by Hendrycks *et al.* (2019). For small data sets, pretraining is crucial as it stabilizes the training. For HSI, this kind of pretraining is, reasoned by the task-specific features, very uncommon.

Complicated Data Augmentations Many works showed how proper data augmentation techniques boost the performance of models for color images (e.g., bag-of-freebies proposed by Bochkovskiy *et al.* (2020)). The meaning of color images is quite robust against modifications. In contrast, the meaning of the spectrum of HSI is rather sensitive. Minor changes can already destroy the meaning. It is, for example, not easy to alter the object color as a modification of the channels in the visible range also affects other channels in a hard-to-predict way. Still, alterations in the spatial dimension (e.g., rotation or flipping) are easy to apply.

Large Channel Dimension Finally, the most obvious difference to color images is the larger number of channels. Handling the many channels requires special attention. So, architectures of neural networks, which were optimized for color images, often perform poorly for hyperspectral recordings as these expect only three input channels and are designed for large data sets. This also applies to classicial machine learning approaches. As a consequence, dimension reductions as preprocessing are widespread for hyperspectral data.

We often compared HSI to color image classification. The trend of success for color image classification in recent years has been astonishing, and the basic structure is similar (with two spatial dimensions and one channel dimension). A similar performance boost for hyperspectral applications would be favored. For MSI, most challenges also apply (like lack of data sets, task-specific features, and complicated data augmentations). In this work, we present different applications of hyperspectral and multispectral imaging. Furthermore, we try to tackle some of the challenges in the further chapters:

- In chapter 3, we propose a data set that can be used to test models in a hyperspectral classification task based on ripening fruit.
- In chapter 4, we evaluate self-supervised methods for hyperspectral data. In this context, the impact of different data augmentations for HSI is validated.
- In chapter 5, a 2D convolution is proposed, which utilizes a model bias designed for hyperspectral recordings. This bias allows handling the large input channels as the features are learned based on wavelengths.
- Chapter 6 shows the usage of embedded multispectral cameras onboard on *unmanned aerial vehicle (UAV)* systems. This covers generating the data set and evaluating object detection models on the multispectral data.

In this work, we focus on our algorithmic research done for HSI in the chapters 3, 4 and 5. In contrast, the MSI-based work, mentioned in chapter 6, is described in less detail.

2.2 Data Sets

Table 2.1: Hyperspectral data sets, which are publicly available.

Name	**Description**	**Task**	**Samples**	**Wavelength range [nm]**
HRSS (Graña, M and Veganzons, MA and Ayerdi, B (2014))	Satellite recordings	Image segmentation	6 scenses with $\approx$ 130.000 annotated pixels	$\approx$ 400 - 2500
HS-SOD (Imamoglu *et al.* (2018))	Hyperspectral Saliency Detection	Saliency detection	60 images	$\approx$ 350 - 1100
FS-ALMI (Wang *et al.* (2022a))	Microsopic images of lung tissue	Temporal classification	32 frames with a resolution of 256×256 pixels	$\approx$ 599 - 780
WHU-Hi (Hu *et al.* (2020b))	UAV-based crop classification	Image segmentation	3 scenes with $\approx$ 1.035.000 annotated pixels	$\approx$ 400 - 1000
DeepHS Fruit v2 (ours)	Recordings of ripening fruit	Image classification	1018 annotated images and 3653 unannotated images	$\approx$ 400 - 1700

As mentioned in the previous section, there is a lack of publicly available hyperspectral data sets. Many hyperspectral algorithms were evaluated on the *HRSS* data set, a segmentation task based on satellite recordings. As we show in chapter 5.4.2, many methods achieve $> 99\%$ on this data set, so the task is not challenging anymore. But, it is still the most common benchmark for HSI methods.

The data sets *HS-SOD* and *FS-ALMI* are examples of most public hyperspectral data sets. They are tiny (100 samples), which reduces the reliability of the results. Further, their application is very specific and restricted. *HS-SOD* uses HSI for saliency detection with only 60 samples, and *FS-ALMI* focuses on the temporal classification of lung cancer. So the methods performing well on these tasks are typically optimized for these specific applications and perform poorly on other HSI applications.

A better candidate seems to be *WHU-Hi*, which covers an image segmentation task recorded with UAVs. Still, the data set is not established, and the quality of the recordings is mixed. But it could become more important in the future.

As a last candidate, we mention the data set *DeepHS Fruit*, which we published. It covers a classification task of the ripeness level of fruit. Currently, the second version of this data set is available. In chapter 3.3, we describe the data acquisition of the first version of the data set. The extension to the second version can be found in chapter 4.3. This version covers five fruit types (avocados, kiwis, mangos, papayas, and persimmons) recorded with three hyperspectral cameras (Specim FX 10, Innospec Redeye, and Corning microHSI 410 Vis-NIR). A full description of this data set can be found in the mentioned chapters. At this point, the three key points of the data set are mentioned:

- It is a classification task, which is fundamental for more complicated tasks like object detection or object tracking. Further, the evaluation on different fruit types reduces the problem of task-specific features. As the ripeness classification of different fruit types requires slightly other features, the models must show more adaptations capabilities. This differs from other task-specific hyperspectral data sets.

- Due to the destructive labeling procedure, this data set contains many unlabeled samples. These recordings cannot be utilized for supervised learning (see chapter 3). In chapter 4, we use the unlabeled recordings with the help of self-supervised learning.

- Finally, the images of this dataset were taken with several different hyperspectral cameras. In chapter 5, we discuss how to train a model that accepts images from different hyperspectral cameras. Since these images usually differ in their channel dimensions, this is not straightforward.

This work will mainly use the *DeepHS Fruit* data set as the mentioned key points support the experiments. Especially in chapter 5, we claim generalizability for the proposed method. Therefore we also provide experiments with the small but established *HRSS* data set.

For multispectral imaging, there is a similar problem. Most multispectral data sets are very specific and small. In contrast to the established HRSS data sets, there is no well-accepted multispectral data set which is used widely. As we focus in this work primarily on hyperspectral recordings, we only highlighted these at this point.

2.3 Hyperspectral Measurement System

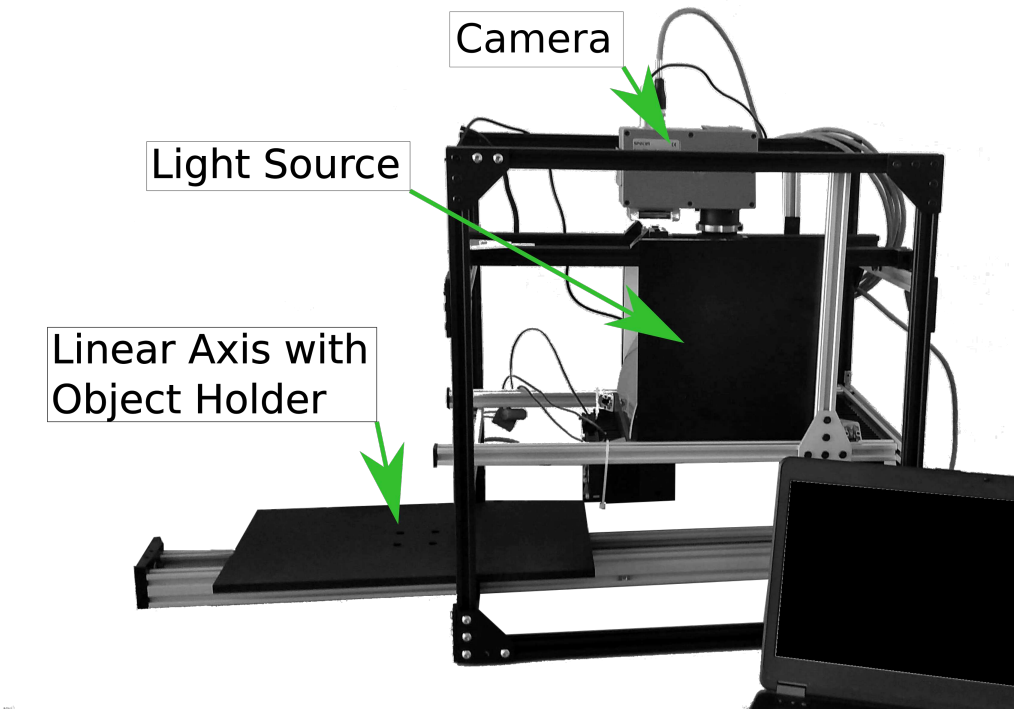


Figure 2.4: The recording system. With the object holder and linear axis, the light source and the camera.

As mentioned earlier, hyperspectral cameras rely on sufficient illumination and often require continuous movement of the test object. A hyperspectral measurement system, which was built for the data acquisition of this thesis, is described here. During the various acquisitions, the system was continuously improved. We focus here on the final version. Three main components are visible in Fig. 2.4. The first component is the object holder, which is moved by a linear actuator. The linear actuator is necessary for the line scan operation mode of the hyperspectral cameras. The linear axis would be unnecessary for a snapshot hyperspectral camera instead of the line scan camera. The latter, however, still seem to have better sensitivity.

The second component is the light source. For HSI, a sufficiently bright and homogeneous light source is indispensable. In the final version of the measurement system, halogen lamps were used, producing sufficient illumination. In the first version of the system, a combination of halogen lamps and LED lamps was tested, but the halogen lamps are enough for our spectral range. In addition, we used a *Polytetrafluoroethylene* curvature reflector to create diffuse light, which is preferable.

The last component is the camera. In total, we used three different hyperspectral cameras. Two cameras covered the visible range (400 - 700 nm) and the lower part of

(a) Specim FX 10 (b) Corning HSI-1d-Vis

Figure 2.5: Two of the used hyperspectral cameras.

the near-infrared range up to 1000 nm. The last camera was only used for the first data acquisition but also recorded wavelengths up to 1700 nm.

Most recordings were done with a *Specim FX 10*. The *Specim FX 10* records 224 spectral channels and covers a spectral range from 400 to 1000 nm. This range holds the VIS range with the addition of the lower NIR range. The spatial dimension contains 1024 pixels.

The second camera is the *Corning HSI-1d-Vis*. The spectral channels are similar to the *Specim FX 10* with 249 channels and a range from 400 to 900 nm. The spatial resolution is a bit higher, with 1400 pixels. Even though the acquired recordings look similar to the specifications of the two cameras, the recordings differ, e.g., discussed in chapter 5.

The last camera is a *INNO-SPEC Redeye 1.7* with 252 spectral channels. This camera misses the visible range but covers the near-infrared range with a wavelength range of 950 to 1700 nm.

All used hyperspectral cameras require line-scan mode. For an optimal acquisition, the exposure time of the camera must fit the illumination to avoid under- or over-exposed areas. Further, the speed of the line actuator has to fit the exposure time. Otherwise, the recordings are spatially distorted.

2.4 Multispectral Cameras

We employed multispectral cameras besides color cameras for an embedded application onboard drones (see chapter 6). We used two multispectral camera types.

A *MicaSense RedEdge-MX* was used for most multispectral recordings. This camera records five channels. Besides the three color channels (red with 668 nm, green with 560 nm, and blue with 475 nm), two channels cover the near-infrared (red edge with 717 nm and near-infrared with 842 nm). Further, a few recordings were also performed with a *MicaSense Altum*. In addition to the five channels of the *MicaSense RedEdge-MX*, this camera records a thermal channel (with 11 µm).

Figure 2.6: A MicaSense RedEdge-MX camera used for most of the multispectral recordings.

In contrast to the hyperspectral cameras, these cameras work in snapshot mode, which is much easier for infield applications.

2.5 Drone Platforms

As carrier systems for the multispectral cameras in the embedded applications, two Quantum System Trinity F90+ drones were used (Fig. 2.7). These fixed-wing drone ,which have a wingspan of 2.4 m and a maximal take-off weight of 5 kg, allow vertical take-off and combine, therefore, the benefits of multi-copters and gliding flight. The take-off and landing area can be small, and the flight time is much higher ($\approx$ 90 minutes) than for multi-copters. These are a perfect fit for the search and rescue task discussed in chapter 6. Besides the multispectral cameras as payload for our experiments, the Quantum System Trinity F90+ drones carried a high resolution camera (Sony UMC-R10C).

Figure 2.7: The Quantum System Trinity F90+ vertical-take-off drone, which carried the MicaSense RedEdge-MX camera on the data acquisition for the SeadDronesSee data set.

Figure 2.8: The ElevonX Sierra drone carring a Nvidia AGX Xavier and a Allied Vision 1800 U-1236 camera on a field mission.

We recorded data for offline processing with these carrier systems and other quadrocopters (e.g., DJI Matrice 100, DJI Matrice 210, and a DJI Mavic 2 Pro). We created the maritime search and rescue data set *SeaDronesSee*, discussing this in chapter 6 with a focus on the multispectral recordings.

For online processing, we built two prototypes based on *ElevonX Sierra VTOL* drones (see Fig. 2.8). This drone is a vertical-take-off fixed-wing drone, too. The wingspan of the *ElevonX Sierra VTOL* is around 3 m. Further, it supports a payload weight of 3 kg, which allows it to carry a *Nvidia Jetson AGX Xavier* or *Nvidia Jetson AGX Orin* computing unit with a high-resolution *Allied Vision 1800 U-1236* camera. The prototype is able of online object detection of the drone recordings with the presentation on a ground station.

In this thesis, we focus on hyperspectral and multispectral imaging. Therefore, we highlight only the parts of this project related to the multispectral recordings (in chapter 6), which are good examples of how spectral imaging can be used in an embedded application.

Chapter 3

Supervised Learning for Hyperspectral Classification

This chapter will discuss supervised methods for *hyperspectral imaging (HSI*). In contrast to the self-supervised learning in the next chapter, these methods require labeled samples. Therefore, the ground truth is needed, which is often costly in acquisition for HSI. Two applications of the area *food inspection/precision farming* will be presented. In the first application, the ripeness prediction of exotic fruit is tackled as a classification task. The second application validates the usefulness of HSI for detecting *Phakopsora Pachyrhizi* infestations on soybean via a patch-level classification task.

For both applications, data sets were acquired. The ripening fruit data set *DeepHS fruit* was recorded with our hyperspectral measurement system (see Sec. 2.3). The Department of Phytopathology of the University of Hohenheim acquired the second data set. Here, we supported with technical knowledge and provided the data evaluations.

In the ripening fruit task context, a shallow convolution neural network *DeepHSNet* was designed for the supervised classification. The proposed network architecture is tiny in comparison to other state-of-the-art computer vision architectures, but it could produce reliable results for the first application. We validated the model in the second application with a patch-based classification task. Therefore, it is a baseline for our developments for the following chapters (see chapter 4 and chapter 5).

This chapter is based on the following publications:

- Varga, L. A., Makowski, J., and Zell, A. (2021). Measuring the ripeness of fruit with hyperspectral imaging and deep learning. In *International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021*, pages 1–8. IEEE

- Thomas, S., Varga, L. A., Harter, N., Zell, A., and Voegele, R. T. (submitted). Detection of phakopsora pachyrhizi infestation in soybean via hyperspectral imaging and data analysis. *Nature Scientific reports*

3.1 Related Work

In this section, we discuss related work for supervised HSI classification. The more specific works, which are more connected to each application, will be addressed separately.

Classical machine learning approaches, like *Support Vector Machine (SVM)* proposed by Cortes and Vapnik (1995) or *k-Nearest-Neighbors (k-NN)* described by Fix and Hodges (1989), are still widespread for the classification of hyperspectral recordings. As we will see later, they can produce reliable results and are easy to apply. Therefore, they are still a very common baseline.

In recent years, deep-learning-based methods could outperform these in many hyperspectral applications. Most of these evaluations were done on hyperspectral remote sensing data. Chen *et al.* (2014) were one of the earliest adopters of deep learning for hyperspectral recordings. Their approach was based on a PCA followed by stacked autoencoders and a final logistic regression. Makantasis *et al.* (2015) showed how a simple 2D *Convolutional Neural Network (CNN)* can outperform SVM and k-NN approaches for tunnel inspection. CNNs, which led to a breakthrough in color image classification, significantly impacted the classification of HSI. Their main benefit is the incorporation of spatial information. Noisy pixels can be stabilized with the help of their neighboring pixels. This additional information and the higher complexity of the methods supported their breakthrough. In this chapter, we focus on 2D CNNs. More complex approaches, e.g., 3D convolutions or vision transformers, will be discussed later (in chapter 4 and chapter 5).

Besides introducing convolution layers, many modifications of the simple 2D CNNs were proposed. For example, Zhang *et al.* (2020) presented the HTD-Net framework, which focuses on hyperspectral data and uses an autoencoder to enhance the training data with additional samples. Or Li *et al.* (2017) proposed a 2D convolution-based approach called CNN-PPF, which utilizes the correlation of neighboring pixels by introducing a pixel-pair feature training. A further example is presented by Song *et al.* (2018) with residual blocks for HSI classification, which allow the extraction of features on different hierarchical layers. This is similar to the ResNet architecture of He *et al.* (2016).

We will discuss the most significant developments of HSI classification in further detail in chapter 5. Still, these developments were mainly applied for the remote sensing application and tested on the segmentation task of the *Hyperspectral Remote Sensing Scenes (HRSS)* data set.

However, the use-case of remote sensing differs widely from the classification task of food inspection. As a result, we focus in this chapter on generating two hyperspectral data sets for food inspection. Further, we propose a simple 2D CNN called DeepHSNet, which is optimized for these applications. It is evaluated against comparable approaches. We kept the model as simple as possible to better understand the model's internals. A comparison with the more complicated state-of-the-art methods of the remote sensing application can be found in a later chapter (see chapter 5). At this point, we concentrate on a simple model for the food inspection task.

3.2 Models

For both applications we evaluated similar models, which we will discuss here shortly. Afterward, the two applications' data acquisition and experiments will be examined. Two classical machine learning approaches are used, which are still very common for HSI classification. Further, a common *Convolutional Neural Network (CNN)* architecture is selected. Finally, a small CNN architecture designed for these applications is proposed.

3.2.1 k-Nearest-Neighbors

k-Nearest Neighbors (k-NN) of Fix and Hodges (1989) is evaluated as baseline model. It defines the requested sample's label by the k nearest neighbors of the sample in the chosen embedding. The parameter k is thereby critical, and we selected it by a grid search with cross-validation on the training set. As a distance measurement, the Euclidean distance between the mean pixels of the samples was selected.

3.2.2 Support Vector Machine

A still very common representative of the classical machine learning algorithms is the *Support Vector Machine (SVM)*. This method defines the classification problem as finding the most suitable support vectors to separate the labeled samples. With the help of the kernel trick, it is possible to use more complex kernels than just a linear separation. A radial basis kernel was used for our experiments as described by Cristianini and Shawe-Taylor (2010). The parameter C, which defines the penalty for misclassification, was evaluated by grid search with cross-validation on the training set.

3.2.3 ResNet-18

The next candidate is a 2D CNN and a representative of the ResNet family, which was proposed by He *et al.* (2016). This family defines a convolutional neural network architecture with identity shortcut connections. We selected the smallest member, called ResNet-18, with just 18 layers. The smallest ResNet-18 was chosen, as it is least likely to overfit. This problem is challenging for small hyperspectral data sets, even if data augmentation and early stopping are used.

For the ResNet-18, the first layer of the network was adapted to the hyperspectral images as input, meaning we increased the number of input channels to the size of the hyperspectral cube, as the default ResNet architecture is designed for only three input channels. The modified ResNet18 model has around 11 million trainable parameters.

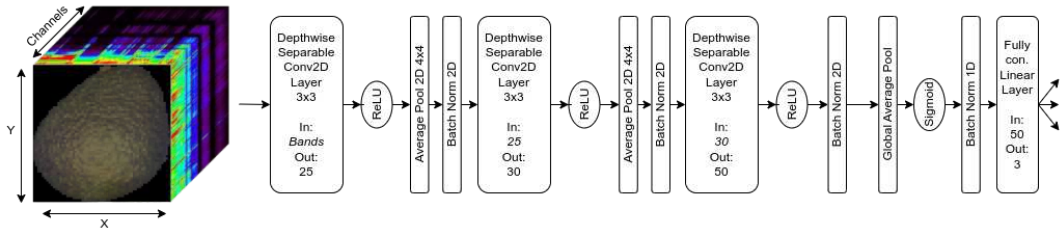


Figure 3.1: Architecture of DeepHSNet.

3.2.4 DeepHSNet

DeepHSNet is a small convolutional neural network, which we designed for the application of ripening fruits and additionally evaluated with the soybean inspection.

Here some architecture decisions are explained and discussed why they are beneficial for hyperspectral data.

An RGB color image is a cuboid with two spatial dimensions and one channel dimension with three channels (red, green, blue). A hyperspectral image has significantly more channels than a color image, so the input data is much larger for the same spatial resolution. For computational reasons, extracting the necessary information at an early step is essential. In many approaches, this is done by a preprocessing step (like PCA (Pearson, 1901), Factor Analysis, or IBRA), where the most significant bands are selected and used for further inspection.

We assume allowing the network to select the essential channels on its own is beneficial for a neural network-based approach. It was shown for many other applications that deep-learning-based methods can handle high dimensional data well. At this early stage of our research, we selected an approach that works directly on the hyperspectral cube without any dimension reduction as preprocessing. We investigate this further in chapter 5 with a more sophisticated method.

Further, the data sets are often tiny compared to common color image data sets. This makes overfitting likely and also leads to unreliable results. Unlike standard color images, no large data sets are available for training or pretraining the models. We avoid this problem with a minimal and shallow architecture in this chapter and for the proposed model. Only three convolution layers are used. Further, the convolutions are separated into two smaller separable convolutions to reduce the number of parameters. This technique was proposed by Guo *et al.* (2018a). In addition, a global average pooling layer is used instead of a large fully connected header as defined by Lin *et al.* (2014a). We will tackle the problem of a missing pretraining in the next chapter with self-supervised Training (see 4) and reduce the number of trainable parameters with a suitable bias further in the chapter afterward (see 5).

The complete architecture is visible in Fig. 3.1. The input is a hyperspectral cube. The recording consists of two spatial dimensions and the channel dimension. Three con-

volutional layers extract feature maps from the input. The convolutions are separated into two smaller separable convolutions to reduce the number of parameters. Instead of the frequently used max-pooling layer, we used average-pooling layers, because they gave empirically better results in our experiments. An explanation might be that the winner-takes-all strategy of max-pooling layers is counter-productive for this task. Furthermore, batch normalization of Ioffe and Szegedy (2015) was used to speed up the training process. The final classification happens in the head of the CNN, consisting of a global average pooling layer and a fully connected layer. In the presented form, the network classified three different categories (e.g. unripe, ripe, overripe) visible in the output of the final layer, which suits the fruit classification task. This results in around 32,000 trainable parameters. We developed this architecture for hyperspectral recordings with about 200 channels of wavelengths. If the number of channels differs significantly, the hidden layers must be adapted. An implementation of the model can be found on https://github.com/cogsys-tuebingen/deephs_fruit.

3.3 Measuring the Ripeness of Fruit

In the fruit industry, one of the goals is to determine how ripe a fruit is. Furthermore, it is helpful for supermarkets to know the ripeness level of fruit in order to avoid selling far overripe fruit or giving significant discounts shortly before. The ripeness can easily be inferred from the skin color for some fruit types, like bananas. This is not trivial for others like avocados, mangos, and kiwis. The fruit industry primarily uses destructive indicator measurements. So only random samples are possible here. To give a solution, we verify whether HSI and deep neural networks could predict the ripeness level of fruit. Here, we contribute a hyperspectral data set, which we will extend and use further throughout this work. We also evaluate different models on the data set, thereby showing the advantage of a small neuronal network called DeepHSNet, which will be used further throughout this work.

3.3.1 Related Work for Fruit Inspection

This work covers the idea of determining the ripeness level of fruit by using hyperspectral recordings. Other works already showed that it is possible to predict the ripeness of fruit by this kind of data. Pinto *et al.* (2019) and Olarewaju *et al.* (2016) used HSI to determine the ripeness level of avocados. Zhu *et al.* (2017) predicted the firmness and the soluble solids content of kiwis with hyperspectral recordings. In these three works, the authors used approaches without neural networks. So far, most fruit classification data analysis has been done with classical machine learning algorithms, which were often supported by small data sets. In contrast to these works, we concentrate on deep learning approaches.

Mollazade *et al.* (2012) showed the prediction capability of a simple neural network for the moisture content of tomatoes. Gao *et al.* (2020) could predict the ripeness state

Table 3.1: Typical wavelength ranges for hyperspectral cameras

Name	Near-ultraviolet (NUV)	Visible (VIS)	Near-infrared (NIR)
Wavelength	200-380 nm	380-740 nm	740-2500 nm

of strawberries with HSI and a pretrained AlexNet, which is a deep convolutional neural network proposed by Krizhevsky *et al.* (2017). The ideas of both works are very similar to ours. In contrast to them, we focus on two new fruits, avocados and kiwis. For both it was already validated, that a prediction with hyperspectral data is possible, which was shown by Pinto *et al.* (2019), Olarewaju *et al.* (2016) and Zhu *et al.* (2017). In contrast to the mentioned works, we used a larger variety of models and recorded a large data set, which we made public. We further analyzed if hyperspectral data is necessary for this task or if pure color images are sufficient. The other works missed this validation.

3.3.2 Hyperspectral Imaging

Hyperspectral imaging (HSI) is a non-destructive measurement technique that has become increasingly popular recently. We fully discussed it in Sec. 2.1.1. The ranges of the wavelengths reveal different chemical properties of the inspected substance. For example, Mitsui *et al.* (2008) showed that the *Near-infrared (NIR*) range indicates the presence of hydroxyl groups. Hydroxyl groups are an essential part of organic chemistry and can indicate the presence or absence of H_2O. With this in mind, it is obvious why the NIR range is vital for fruit inspection. This chapter uses HSI to predict the ripeness level of avocados and kiwis in a non-destructive way.

3.3.3 Fruit Ripening

Here we give a short overview of the ripening process of fruit. There is a distinction between non-climacteric and climacteric fruit. Non-climacteric fruit do not ripen after harvesting, which was, e.g., shown by Alexander and Grierson (2002). Therefore, the focus here is on climacteric fruit. The chemical ripening process highly depends on the fruit type. The three main processes found within fruit are described by Toivonen and Brummell (2008) as:

- Deconstruction of the cell walls so the fruit becomes softer.
- Hydrolyzitation of starch hydrolyzes to sugar, which leads to sweetness.
- Deconstruction of chlorophyll and synthesis of other pigments result in color change.

With the following indicators, the ripeness level of fruit is commonly measured:

- *Soluble solids content (SSC)* is based on the creation of sugar during ripening. Sugars are the majority of soluble solids in most fruit.
- Fruit flesh firmness shows the degeneration of the cell walls.
- Starch content indicates the degeneration of the starch.

Especially SSC and fruit flesh firmness are widely used because they are reliable indicators for many fruit types. Nowadays, their measurement is destructive. So, it is only possible to measure random samples. The works of Pinto *et al.* (2019), Olarewaju *et al.* (2016) and Zhu *et al.* (2017) already showed that it is possible to predict the ripeness level by using HSI for some fruit.

In this work, the focus lies on two types of fruit. Avocados and kiwis are both fruit with a critical ripening process. The time window between unripe and overripe is small for both. Accordingly, this work focuses on the end of the ripening process. Our goal is to predict the perfect consumption date as a classification problem.

(a) Avocados (b) Kiwis

Figure 3.2: Two of the fruit crates at day 1 of the first measurement series.

Avocado

The avocado is the berry of an evergreen laurel plant. There are more than 400 different types of avocado, *Hass* and *Fuerte* being the most common. Through this broad diversity of species, the appearance of avocados may vary widely. Avocados only ripe after harvesting because the tree produces an inhibitor that prevents the fruit from ripening, which was already shown by Lewis (1978). Besides the small consumption window, the avocado was chosen because of its relatively high price. Pinto *et al.* (2019) and Olarewaju

et al. (2016) showed that it is possible to conclude the ripeness level with HSI. Nevertheless, the most common ripeness measurement technique for avocados is the firmness of the fruit flesh.

Kiwi

Like the avocado, the looping berry fruit plant kiwi has many subspecies. The best known are probably *Actinidia deliciosa* and *Actinidia chinensis*. Their appearance is very similar, only the color of the fruit flesh differs. HSI for the ripeness determination of kiwis is uncommon. Useful indicators for the ripeness of kiwis are the SSC and the firmness of the fruit flesh as shown by Martinsen and Schaare (1998).

3.3.4 Data Set

In this section, the data acquisition is described, so it is possible to reproduce the data or adapt the procedure for other fruit. The measurement setup we described in chapter 2.3 was used. Our hyperspectral recordings are available on https://github.com/cogsys-tuebingen/deephs_fruit. This data set is used in the further analysis. The data set contains 1038 recordings of avocados and 1522 recordings of kiwis. It covers the ripening process from unripe to overripe for both fruit types. Because of the destructive manner of the labeling process, only 180 avocado recordings and 262 kiwis recordings are labeled by indicator measurements. The data set was recorded in two separate measurement series. We applied a division into a training set ($\frac{3}{4}$), validation set ($\frac{1}{8}$), and test set ($\frac{1}{8}$), evenly distributed among the different states of ripening.

Label Acquisition

Aside from the camera system, we used a refractometer to measure the SSC. A refractometer can indicate the concentration of a specific substance in the sample. For the fruit flesh firmness, we used a penetrometer. A penetrometer can measure penetration resistance. Both techniques are currently the usual way to measure fruit ripeness. But they are also destructive.

Data Acquisition

The two measurement series covered a total of 28 days in the years 2019 and 2020. We acquired fresh avocados and kiwis for the two series from a supermarket, supporting our measurement plans. Each day the following procedure was followed:

1. Record the temperature
2. Start the measurement setup for the warm-up of the lamps
3. Calibrate the linear actuator

4. For both cameras:
    a) Adjust the focus of the camera on the surface of a reference object
    b) Record a white reference (average of 10 measurements)
    c) Record a dark reference (average of 10 measurements)
    d) Record the front and the back of each fruit, in order to double the data without much effort
5. Select fruit for destructive indicator measurement
    a) Weigh the fruit
    b) Determine the fruit flesh firmness with a penetrometer
    c) (Only for kiwis:) Measure the sugar content via the refractometer
    d) Record the overall ripeness level of the fruit by appearance and taste

The number of destructively measured fruit was adapted to each day's ripening progress. The output of the two series is a collection of hyperspectral recordings of kiwis and avocados. Each recording contains only one fruit.

Data Preparation

To improve the quality of the recorded data, we used background extraction. We excluded the background with a simple pixel-based neural network that we trained to differentiate between background and fruit. Further, the smallest rectangle around the fruit was extracted from the recordings to remove most of the background. We observed that the results are better if the intensity of the remaining background is forced to zero. Therefore, the results are the smallest possible recordings of the fruit with an empty background.

For the labels, we defined categories. We aimed to classify whether the fruit is unripe, ripe, or overripe. Consequently, a regression problem is unnecessary, and we reduced the complexity to three classes for the firmness, the sweetness, and the overall ripeness level. For the category firmness, the classes were based on the penetrometer measurements. The sweetness category, which is only helpful for kiwis, was based on the refractometer tests. The last category, ripeness, was based on appearance and taste. The class assignments are visible in Tab. 3.2.

3.3.5 Experiments

In this section, the data set and experiment setup is described.

Table 3.2: Class Assignments for avocados and kiwis. For avocados, the sweetness is not a valid indicator. * marks all fruit with optimal ripeness.

Fruit type	Class	**Firmness**	**Sweetness**	**Overall Ripeness**
Avocado	Unripe	$>$1200 g/cm^2	-	Green and unripe taste
	Perfect	*	-	*
	Overripe	$<$900 g/cm^2	-	Brown spots in fruit flesh and overripe taste
Kiwi	Unripe	$>$1500 g/cm^2	$<$15.5 °*Brix*	Sweetness is missing
	Perfect	*	*	*
	Overripe	$>$1000 g/cm^2	$<$17 °*Brix*	Acerbity is missing, plain sweet taste

Training

For training, the size of the classes in the categories was balanced. Thus, there was no bias towards one class. We used rotation, flipping, random noise, and random cut as data augmentation techniques, which don't change the label. The neural networks were optimized with Adabound, presented by Luo *et al.* (2019), using 1×10^{-2} as the learning rate. Focal loss of Lin *et al.* (2017) was used as a loss function. We used early stopping based on the validation loss to prevent over-fitting as described by Prechelt (1998). For training, we used a batch size of 32. The hyperspectral images were resized to 64×64 pixels.

Test

We tested five models on our data set. Besides the already described models (SVM, k-NN, ResNet-18, and DeepHSNet), we further used an AlexNet, which was used by Gao *et al.* (2020) for strawberry ripeness prediction. The smallest ResNet-18 was used because a larger representative of the ResNet family would more likely tend to over-fitting. For the ResNet-18 and the AlexNet, the first layer of the network was adapted to the hyperspectral images as input.

The test set was $\frac{1}{8}$ of the labeled hyperspectral recordings. Test time augmentation was used for the evaluation, proposed by Howard (2014). The test results are given in Table 3.3 and Table 3.4. For each neural network, three values are provided. The *Raw* value gives the accuracy when the network accesses the raw hyperspectral recording.

Table 3.3: Test accuracy over all categories for avocados. The highest accuracies for each configuration are given in **bold**.

Category		Firmness		Ripeness	
Camera		INNO-SPEC Redeye	Specim FX 10	INNO-SPEC Redeye	Specim FX 10
SVM		77.8%	73.3%	44.4%	66.7%
k-NN		73.3%	77.8%	**88.9%**	60.0%
ResNet-18 11M parameters	RGB	66.7%	66.7%	66.7%	53.3%
	PCA	44.4%	53.3%	44.4%	60.0%
	Raw	66.7%	80.0%	33.3%	80.0%
AlexNet 58M parameters	RGB	44.4%	33.3%	33.3%	33.3%
	PCA	44.4%	33.3%	33.3%	33.3%
	Raw	44.4%	33.3%	33.3%	60.0%
DeepHSNet (our)	RGB	77.8%	53.3%	55.6%	40.0%
	PCA	44.4%	80.0%	44.4%	66.7%
32K parameters	Raw	**88.9%**	**93.3%**	**88.9%**	**93.3%**

In the *RGB* case, the hyperspectral recordings were reduced to color images in a preprocessing step. And for the *PCA* case, a *Principal Component Analysis (PCA*), as proposed by Pearson (1901), was used to reduce the channel size of the hyperspectral recordings to 5. The PCA technique is often used for hyperspectral recordings to extract only the necessary information in an early step. This will be discussed in detail in chapter 5.

Our model outperformed the reference models in most cases. Moreover, it produced the most stable results. With our model, it was possible to predict the firmness of avocados with an accuracy of over 93.3 % and further predict the ripeness level in 3 categories with over 90 %. Predicting the ripeness level of the kiwis is more complicated than for the avocados. Thus, the prediction accuracy for them was significantly lower for all models. However, our model could still predict the firmness of unseen kiwis with an accuracy of nearly 70% and the ripeness with nearly 80%.

Further, the *Raw* configuration was, in most cases, better than the reduced configurations (*RGB* or *PCA*). The network could select the most influential bands in the *Raw* case. *RGB* was in some cases better than the *PCA* approach. The *RGB* reduction doesn't use the largest variance in contrast to *PCA*. Instead, it uses the CIE color-matching functions to calculate the impact of each wavelength. Most likely, *PCA* has troubles with the noisy channels of the recordings and removes necessary information by the reduction, which is still available in the *RGB* reduction.

Table 3.4: Test accuracy over all categories for kiwis. The highest accuracies for each configuration are given in **bold**.

Category		Firmness		Sweetness		Ripeness	
Camera		INNO-SPEC Redeye	Specim FX 10	INNO-SPEC Redeye	Specim FX 10	INNO-SPEC Redeye	Specim FX 10
SVM		44.4%	60.9%	44.4%	**82.6%**	33.3%	45.8%
k-NN		**55.6%**	60.9%	22.2%	73.9%	55.6%	50.0%
ResNet-18	RGB	44.4%	56.5%	55.6%	47.8%	44.4%	54.2%
	PCA	33.3%	60.9%	44.4%	47.8%	66.7%	33.3%
11M parameters	Raw	**55.6%**	60.9%	**66.7%**	47.8%	66.7%	58.3%
AlexNet	RGB	33.3%	52.2%	44.4%	47.8%	33.3%	33.3%
	PCA	33.3%	52.2%	44.4%	47.8%	33.3%	33.3%
58M parameters	Raw	33.3%	52.2%	44.4%	47.8%	66.7%	33.3%
DeepHSNet (our)	RGB	44.4%	65.2%	55.6%	60.9%	44.4%	62.5%
	PCA	44.4%	34.9%	44.4%	47.8%	33.3%	33.3%
32K parameters	Raw	44.4%	**69.6%**	**66.7%**	**82.6%**	**77.8%**	**66.7%**

3.3.6 Ablation Study

The architecture DeepHSNet was designed for this specific HSI application. Therefore, a more in-depth analysis of the architecture should show the impact of the different parts. In the following, elements of the architecture are altered to evaluate the effect of these. Given is the test accuracy for the prediction of the avocado firmness. The architecture decision used for DeepHSNet is marked with a grey background.

Depthwise Separable Convolution

The idea behind *Depth-Wise Separable Convolution (DSCNV)* of Guo *et al.* (2018a) is to split up the regular convolution into the spatial and a depth-wise convolution, which corresponds to the channel dimension. It is a common technique to reduce the number of parameters, which can prevent overfitting. Still, the splitted convolutions should be as powerful as the single convolution. A minor degeneration without *DSCNV* is noticeable. So, it seems helpful for this application.

Convolution type	**Accuracy**
DSCNV	93 %
Normal convolution	80 %

Model-Head

The head of the network uses the feature map of the convolutional backbone to determine the classification result. We inspected three head architectures. A fully connected head, a

Global Average Pooling of Lin *et al.* (2014a) head, and a head based on Global Average Pooling with an additional linear layer. The Global Average Pooling reduces the number of parameters, which prevents overfitting. Just Global Average Pooling leads to the worst results. The extra linear layer seems necessary.

Head architecture	**Accuracy**
Global Average Pooling with additional layer	93 %
Global Average Pooling	80 %
Fully connected layers	87 %

Augmentation

The influence of the different augmentation techniques is visible here. Random cut and test time augmentation are essential in this scenario. On the other hand, the effect of the transformation augmentations is minor, so fruit alignment is less of an issue in this data set. In chapter 4, we will evaluate the impact of different augmentation techniques on HSI classification in more detail.

Augmentation variant	**Accuracy**
Full augmentation	93 %
Without test time augmentation	71 %
Without random noise	73 %
Without random cut	69 %
No transformation augmentation	80 %

Loss Function

The Focal loss is a cross entropy loss that weighs the impact of a sample corresponding to their classification error. This improves the behavior with unbalanced classes, as shown by Lin *et al.* (2017). Although we avoided class bias, the Focal loss still improved the result.

Loss function	**Accuracy**
Focal loss	93 %
Cross entropy loss	80 %

Optimizer

We tested different optimizers for the training process. The Adabound optimizer with a learning rate of 0.01 worked best in our case. During the later works, we found out that Adabound is not very reliable. The step sizes are erratic, and finding an optimal solution is not guaranteed. Therefore, we used the more established Adam optimizer for most of the other works because it was much more reliable. Still, for these experiments, Adabound produced better results, so we used Adabound for these experiments. But we highly recommend starting with Adam or an SGD with a learning rate scheduler for an unknown scenario.

Optimizer	Accuracy
Adabound with learning rate 0.01 (Luo *et al.* (2019))	93 %
Adabound with default parameters (Luo *et al.* (2019))	80 %
Adam (Kingma and Ba (2015))	80 %
Stochastic gradient descent (Kiefer and Wolfowitz (1952))	80 %

Pooling Layers

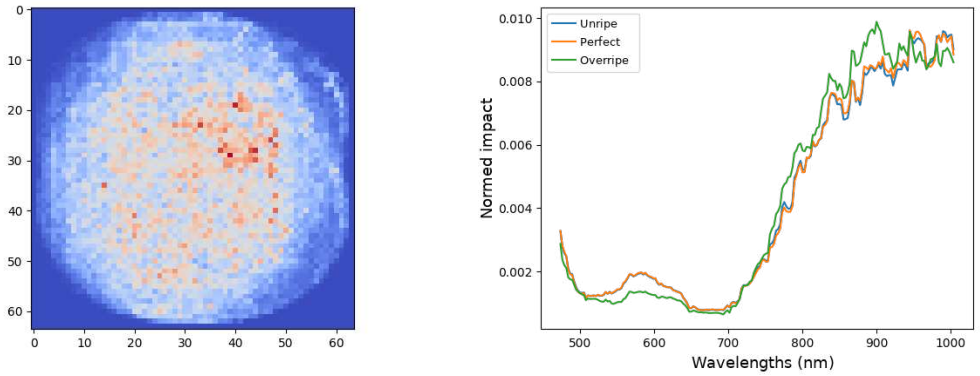
We compared max pooling layers with average pooling layers. The results with average pooling layers were minimally better. For this problem, it seems more important not to consider only the extreme value.

Pooling	Accuracy
Average pooling	93 %
Max pooling	87 %

3.3.7 Investigation of the Learned CNN Features

Besides the ablation study, we want to show that the trained DeepHSNet network learns meaningful features for the classification, which validates the correctness of the prediction. We used Integrated gradient, proposed by Sundararajan *et al.* (2017), to see what parts of the hyperspectral cube are essential to determine the state of the fruit. This technique can show neurons' influence on the network's decision. It is possible to validate the decision process of the neural network to a certain extent.

In Fig. 3.3a, the spatial distribution of the impact for the avocado ripeness prediction is presented. The effect is evenly distributed over the whole fruit. The wavelength-based impact is visualized in Fig. 3.3b. The main decision happens over 800 nm. This discovery fits with the findings of Pinto *et al.* (2019). Additionally, to a small extent, the range of the visible light between 520 nm and 650 nm was used by the network to differentiate between unripe and perfect fruit. This range matches the visible change



(a) (b)

Figure 3.3: The impact of the input on the predicted class for an avocado recorded with the *Specim FX 10*. The impact of the spatial **(a)** and the spectral **(b)** features is visualized.

of avocados. Overall the features learned by the convolutional neural network seem plausible. In chapter 5, we will propose a method that allows this investigation by design by learning these ranges separately.

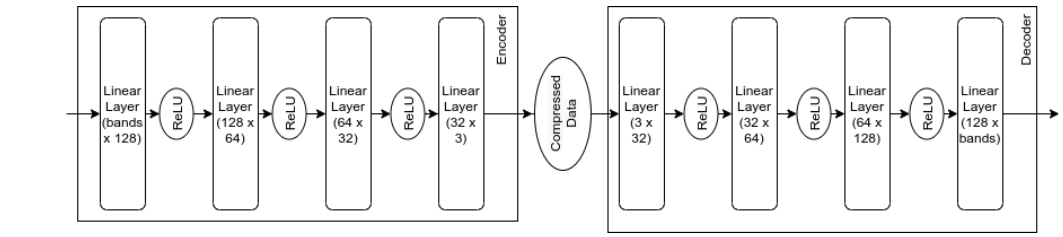
3.3.8 Visualization of the Ripening Process

Furthermore, we introduce a technique to generate false-color images of hyperspectral recordings for specific tasks. We used a two-stage training process and a two-level classifier, presented in Fig. 3.4. In the first step, we trained a pixel-based autoencoder (Fig. 3.4a) to encode and decode hyperspectral images of fruit. The unlabeled data can also be used here. We used the mean-squared error for training. The latent space size was three, so the interpretation as a color image is possible. In the second step, we used the encoder's embedding as the input for a classifier network (Fig. 3.4b) and trained the classifier to differ between ripeness levels. Here a Focal loss was used. For the second step, the labeled data is necessary. The weights of the encoder were not fixed in the second step. So, the embedding representation was adapted to fit better to the classification task. As a result, the encoder is specialized to differentiate ripeness levels.

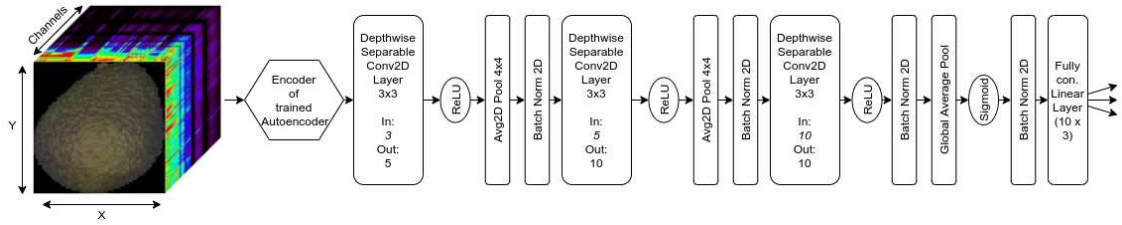
An encoder we have trained in this way can produce false-color images containing the necessary information for the classification task in a color image.

For avocados, an example is visible in Fig. 3.5. The ripe parts grow from the bottom to the top of the fruit. Another example is also visible in Fig. 3.5. Here the encoder was specialized for firmness prediction. The output visualizes the firmness distribution of a kiwi. A damaged part slowly grows over the fruit.

This technique can benefit from a large amount of unlabeled data and allows the creation of visual representations of the ripening process. A performance boost with this



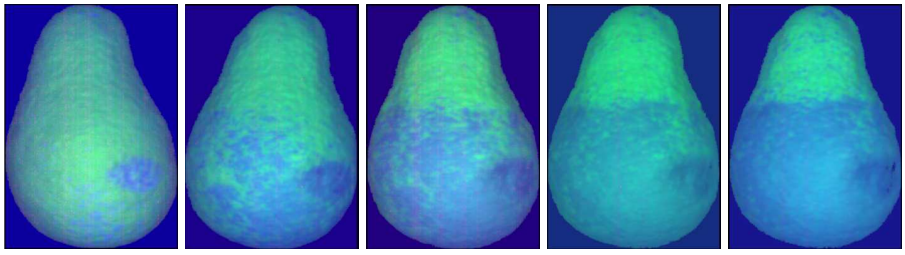
(a) Autoencoder



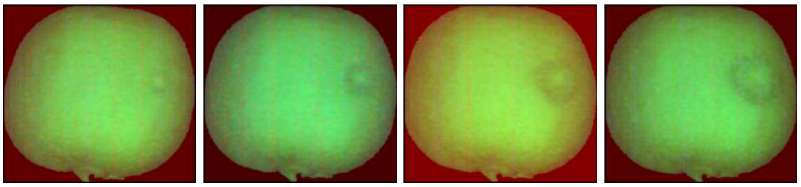
(b) Classifier

Figure 3.4: The architecture of the pretrained approach. The **(a)** autoencoder architecture is trained unsupervised in the first stage. In the second stage, the **(b)** classifier is trained. The pretrained encoder is used as backbone for the classfier network.

training was not observed, which could be connected to the small latent space of the autoencoder. Furthermore, in the presented form, it only uses pixel-based pretraining. So, it is not further considered in the discussion of self-supervised methods (chapter 4).



(a) Day 3 (b) Day 6 (c) Day 7 (d) Day 8 (e) Day 9



(f) Day 3 (g) Day 4 (h) Day 5 (i) Day 6

Figure 3.5: Visualization of the ripeness development of an avocado (**(a)** - **(e)**) and the firmness distribution of a damaged kiwi (**(f)** - **(i)**)

3.3.9 Conclusion

We showed that convolutional neural networks could be used on hyperspectral data to classify exotic fruit into three classes (unripe, ripe, and overripe). We published a data set of ripening avocados and kiwis. Our DeepHSNet classifier network shows superb performance in the classification of ripeness states for avocados and good performance for kiwis. We could validate the results with a more in-depth look into the trained features. The following section will validate this architecture in a second HSI classification task. Further, this architecture will be used throughout the following two chapters.

Besides the architecture, we presented a technique to produce false-color images for specific use cases.

As the labeling procedure is destructive, the labeled data is sparse. Self-supervised approaches utilizing the unlabeled boosting the models' performance are particularly promising for this application and will be discussed in chapter 4.

3.4 Food Inspection for Soybean (*Glycine max*)

Soybean (*Glycine max*) has worldwide importance as a source of high-quality plant oil, with a production of over 100 million tons in south and north America, as shown by United States Department of Agriculture (2022). Barro *et al.* (2021) showed *Asian Soybean Rust (ARS*) is a disease with the potential to cause up to 90% yield loss in soybean production, leading to massive economic damages when left untreated. Due to the short lifecycle of the causative agent *Phakopsora pachyrhizi* and the high damage potential, timely treatment with fungicides is complex, and the availability of resistant soybean cultivars is limited (Childs *et al.*, 2018; Goellner *et al.*, 2010).

Phakopsora pachyrhizi is an obligate biotrophic pathogen. While little is known about the sexual reproduction of the fungus, the asexual reproduction on soybean is well understood (Goellner *et al.*, 2010). Urediospores are dispersed by wind and germinate under suitable conditions on soybean leaves. Unlike most other rust fungi, which infect their respective host plants through stomates (Voegele, 2006), *Phakopsora pachyrhizi* forms an appressorium, which directly penetrates an epidermal cell of the soybean leaf (Goellner *et al.*, 2010). The fungal mycelium grows in the intercellular space of the mesophyll tissue with haustoria providing nutrients derived from the plant cells (Goellner *et al.*, 2010). Finally, the fungal mycelium forms uredosori, which break through the epidermis and release new urediospores 5-8 days after the initial infection (Goellner *et al.*, 2010; Voegele, 2006; Koch *et al.*, 1983).

Optical sensors have shown to be an effective tool for detecting plant diseases in previous studies (Mahlein, 2016; Roitsch *et al.*, 2019). Among the different optical sensors, hyperspectral sensors have the distinct advantage of producing a detailed profile of the plant's reflectance signature, allowing for precise measuring of metabolic and structural changes within the plant, which can be linked to specific plant–pathogen interactions

(Alisaac *et al.*, 2018; Thomas *et al.*, 2018a). Nevertheless, the application of hyperspectral imaging for plant disease detection in early stages is challenging due to large amount of data generated by the sensor, which leads to the requirement of advanced data analysis methods to efficiently extract the relevant data for disease detection (Mahlein *et al.*, 2019).

Classical machine learning approaches, like *Support Vector Machine (SVM)* (Cortes and Vapnik, 1995) or *k-Nearest-Neighbor (k-NN)* (Fix and Hodges, 1989), are capable of handling this amount of data and achieved reliable results in hyperspectral classification applications (Guo *et al.*, 2018b; Kuo *et al.*, 2013). In recent years, deep learning-based models outperformed other machine learning methods at the cost of increasing model complexity (Li *et al.*, 2019b). Convolution layers allow trainable filters for neural networks, simplifying the consideration of spatial information (Matsugu *et al.*, 2003). The spatial information boosted the performance of neural networks even further. However, the training procedure of all presented methods is supervised. Therefore, annotated training samples are required.

This section focuses on the potential of hyperspectral imaging sensors to detect an ASR infection in early stages, before symptoms become visible. To achieve this, the resulting datasets of two experiments, in which time-series measurements on inoculated soybean plants have been performed, were analyzed with classical machine learning methods and deep learning. The applied support vector machine, k-NN, and neural network analysis were individually tested for their capability of early detection of disease symptoms and their detection accuracy and compared with each other. A limited set of training data, focusing on the last three days of the time-series measurement, was generated via a human expert and split into three distinct classes with high variability in each class due to multiple factors – such as different plant parts or symptom progression – being included in the respective classes. This allows an estimation of the capabilities of the supervised classification methods to overcome in-class data variance and performance in detecting early-stage disease symptoms based on a limited training data set.

3.4.1 Materials and Methods

In this section, the data preparation and data analysis methods are described.

Plant cultivation, pathogen material and inoculation procedure

The soybean (*Glycine max*) plants were grown under greenhouse conditions with two seeds planted per pot. Greenhouse conditions consisted of long light (16 / 8 h) with an average temperature of 22°C and 40%–50% relative humidity. Once the plants reached *Biologische Bundesanstalt für Land- und Forstwirtschaft, Bundessortenamt und CHemische Industrie (BBCH)* stage 11, meaning the start of development of leaves, plants were separated to guarantee one soybean plant of similar growth per pot.

Phakopsora pachyrhizi spores from the University of Hohenheim laboratory collection, stored at -80°C, were suspended in $0,01\%$ tween solution with a concentration of 1 mg / 1 mL directly after thawing. Soybean plants were inoculated with the spore suspension at BBCH stages 61 and 70 (development of flowers) for experiments 1 and 2, respectively. The spore suspension was uniformly applied through a nebulizer. Tween solution without spores was applied to the control plants in order to exclude possible effects of the solution on early measurements. After inoculation, both inoculated and control plants were placed for 24 h in a high humidity environment ($> 90\%$ humidity) without light at 21 °C.

Hyperspectral imaging measurement

All measurements were performed via the Corning microHSI 410 Vis-NIR pushbroom hyperspectral sensor, fixed to a BiSlide Positioning Stage linear actuator to permit precise sensor movement. Additionally, 4 Illuminator 70W 3100K halogen lamps were fixed to the linear actuator to guarantee constant and even illumination of the measurement area while scanning the samples. The equipment was controlled remotely via the software FluxTrainer of Luxflux GmbH. Both control and inoculated plants were framed over the entire duration of the time-series measurements to ensure minimal leaf movement and high compatibility of leaf placement throughout the time series.

During each measurement, the light sources were activated 30 minutes before the measurement start to prevent changes in illumination values of the halogen lamp due to temperature changes. The framed plants were placed in the measurement area and covered to avoid heat stress. At the same time, white and dark references for reflectance calculation were acquired at the same distance from the hyperspectral sensor as the measured plant samples. A polytetrafluorethylene-based material was used as a white reference. The dark reference was obtained by closing the shutter of the camera. Each reference value was averaged over 100 frames.

Two experiments (1 and 2) were performed as time-series measurements via hyperspectral imaging. Each experiment consisted of eight soybean plants, of which two were used as control while six were inoculated with *Phakopsora pachyrhizi* spores. An average of nine soybean leaves per plant were measured over the respective experiments (minimum three leaves, maximum 15 leaves), each consisting of 80,000 to 130,000 pixels, respectively. The time-series measurement started 1 *Day after Inoculation (DaI)* with a measurement every 24 h until 10 DaI for both experiments.

Data analysis

The FluxTrainer software was used for visual assessment and spectral information extraction of the hyperspectral datasets through an expert. Furthermore, the training data for the data analysis methods were selected via this software. The training data was divided into three classes: healthy plant tissue, disease symptoms, and background. Each

of the respective classes consists of over 150.000 annotated pixels, including samples for all features within the respective class (see Fig. A.1). The pixels were annotated from the images at 10 DaI at experiment 1 and 8, 9 and 10 DaI from experiment 2, respectively, when developed, and late-stage symptoms were clearly visible in order to ensure high quality training data.

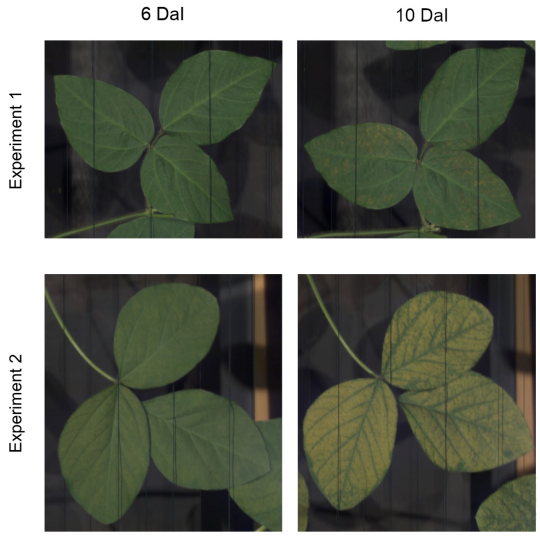


Figure 3.6: Pseudo RGB images of soybean plants with typical ASR symptoms for experiment 1 and 2, respectively, at 6 and 10 *Days after Inoculation (DaI)*.

Supervised machine learning methods

The classification was handled as a supervised pixel-based classification task. Each pixel of the input recording should be classified into one of the three classes (healthy, diseased, or background). Five models were selected to compare classical machine learning and recent deep learning approaches.

SVM and k-NN represented the classical machine learning approaches. These were tested in two configurations. The methods in their default configuration were used in the first set of experiments. In the second set of experiments, the configurations of the methods were optimized for this specific task. This step required additional knowledge of the application and the data. As a result, a *Principle Component Analysis (PCA)* (Pearson, 1901) with 20 output components and an additional normalization were added as preprocessing steps. Both configurations were considered in the further analysis.

Three candidates represented deep learning approaches. The first neural network was a fully connected neural network with four layers. The fully connected layers were separated by ReLU activation functions (Nair and Hinton, 2010) and Batch Normalization layers (Ioffe and Szegedy, 2015).

DeepHSNet, the second neural network, was already validated on the ripening fruit data set. As well as the ResNet18 architecture. Both were described in further detail in Sec. 3.2.

The neural networks were trained with the Adam optimizer (Kingma and Ba, 2015). As for the previous experiments, a learning rate of 1×10^{-2} and a batch size of 64 were used. The learning rate was divided by ten after every 30 epochs.

SVM, k-NN, and the fully connected neural network classified each pixel separately. DeepHSNet and ResNet18 utilized 63x63 pixel patches to classify the center pixel. A class bias was avoided by oversampling all classes to the same number of samples.

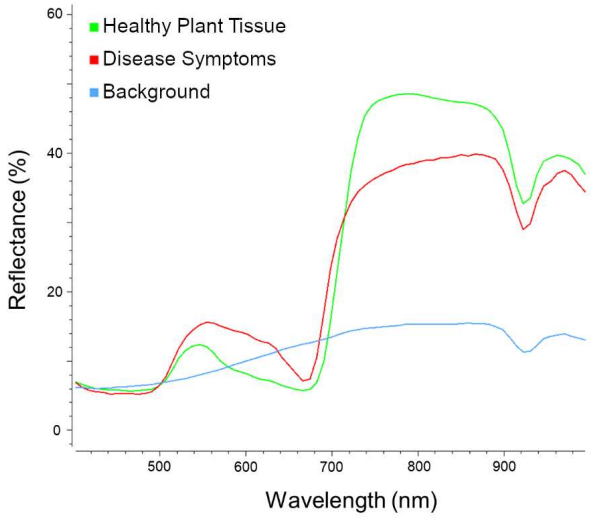


Figure 3.7: Characteristic spectral reflectance signatures of the three classes.

3.4.2 Results

Visual assessment of the hyperspectral datasets

The control plants showed no signs of disease symptoms throughout the experiments. Despite the leaves framing to minimize leaf movement over the time-series measurements, it was not entirely possible to prevent leaf movement in all cases. Nevertheless, it was possible to clearly discern and track each leaf throughout the measurement series as leaf positions stayed relatively stable and plant orientation was kept constant during the measurements.

Plants inoculated with the *Phakopsora pachyrhizi* showed steadily progressing disease symptoms over the measured period in both experiments. In experiment 1, disease symptoms could initially be visually assessed at 6 DaI and slowly progressed until the end of the measurement period. In experiment 2, disease symptoms were detected visually at 5

DaI and progressed quickly throughout the experiment to the point where entire leaves were symptomatic. While disease progression in both experiments was similar, disease severity in experiment 2 was significantly higher than in experiment 1 (see Fig. 3.6). As shown in Fig. 2, it was also impossible to completely prevent the leaf movement of the inoculated plants. However, the change in leaf position in the images is insignificant for a leaf comparison between different images in the time-series measurement.

After manual assessment of the visual and spectral data within the image, three main classes – healthy plant tissue, disease symptoms, and background – were identified within the image as training data for use with the supervised data analysis methods employed within the study (see Fig. 3.7). Each of the individual classes showed considerable variability within the images due to factors such as plant geometry, leaf placement, leaf shadows, and symptom development, e.g., the different features included in the class healthy plant tissue consist of leaf, stem, soybean pod and leaf vein (see Fig. A.1).

Analysis of the hyperspectral datasets through supervised machine learning and neural networks

The datasets of both experiments were analyzed with multiple supervised data analysis methods. SVM and k-NN were selected as classic machine learning methods, which have been successfully used in many studies. These methods were tested in a default configuration and a fine-tuned configuration. The classical machine learning methods were compared with three neural networks, a fully-connected neural network, ResNet18, and DeepHSNet.

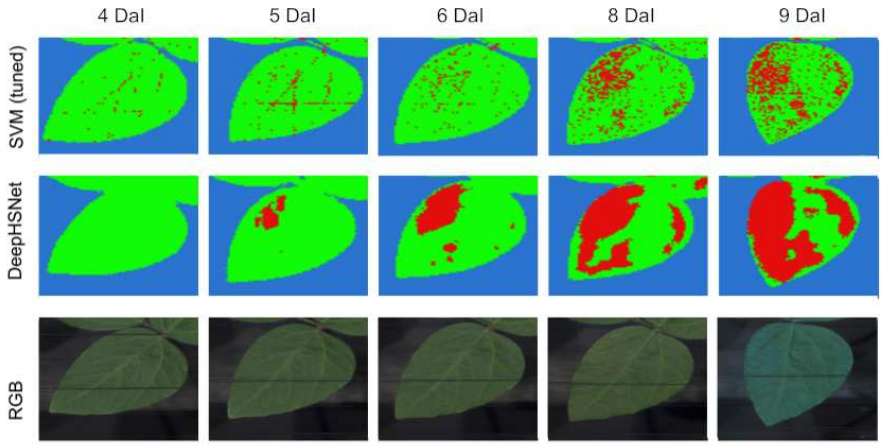


Figure 3.8: Pseudo RGB and false color images of an inoculated soybean leaf in experiment 1. The false color images represent the classification results of DeepHSNet and SVM, with: green = healthy plant tissue, red = disease symptoms, blue = background.

Both machine learning methods performed adequately in their default configuration with an accuracy of 86.8% and 87.21% for k-NN with 10% and 30% of the annotated

Table 3.5: Accuracy of the data analysis methods for *P. Pachyrhizi* symptom detection within the study.

Method	**% of annotated data used as training data**	**Accuracy (%) ↑**
k-Nearest Neighbor	10	86.80
	30	87.21
k-Nearest Neighbor (finetuned)	10	99.50
	30	99.58
Support Vector Machine	10	61.64
	30	78.73
Support Vector Machine (finetuned)	10	99.58
	30	99.62
Fully connected network	10	99.94
ResNet-18	10	99.97
DeepHSNet	10	**99.99**

data used as training data, respectively, while SVM achieved an accuracy of 61.94% and 78.73% (see Tab. 3.5). By fine-tuning the configurations of the methods for the specific task, the performance of both methods was improved significantly. k-NN achieved 99.58% accuracy with 30% of the annotated data used as training data, and the accuracy of SVM was improved to 99.62% (see Tab. 3.5). However, the neural networks outperformed the machine learning methods with an accuracy of 99.94% (fully connected), 99.97% (ResNet-18), and 99.99% (DeepHSNet) with only 10% of the annotated data as training data (see Tab. 3.5).

DeepHSNet and SVM were chosen as candidates for machine learning and deep learning methodology to test for early disease symptom detection due to the excellent accuracy within the given data. In experiment 1 first disease symptoms were detected via DeepHSNet at 5 DaI, one day before symptoms became visible with the human eye, at locations of the leaf that showed visible symptoms on the following day (Fig. 4). Throughout the time-series measurement, the DeepHSNet classification results correlated with visible symptoms on the inoculated leaves of the experiment (Fig. 4). In experiment 2 the classification showed first results at 4 DaI before visible symptoms could be observed at 5 DaI and had overall comparable performance to the results in experiment 1 (Fig. 5). However, through the higher disease severity the classification results showed entire leaves as symptomatic in the later stages of the time-series measurement (Fig. 5). After manual inspection of the symptomatic leaves it was concluded, that the classification result is correct in these cases.

First disease symptoms were detected via SVM at 4 DaI and 3 DaI for experiments 1 and 2, respectively, one day prior to the detection via DeepHSNet (see Fig. 3.8 and Fig. 3.9). However, as shown in Fig. 3.8, pixels classified as symptomatic at 4 DaI

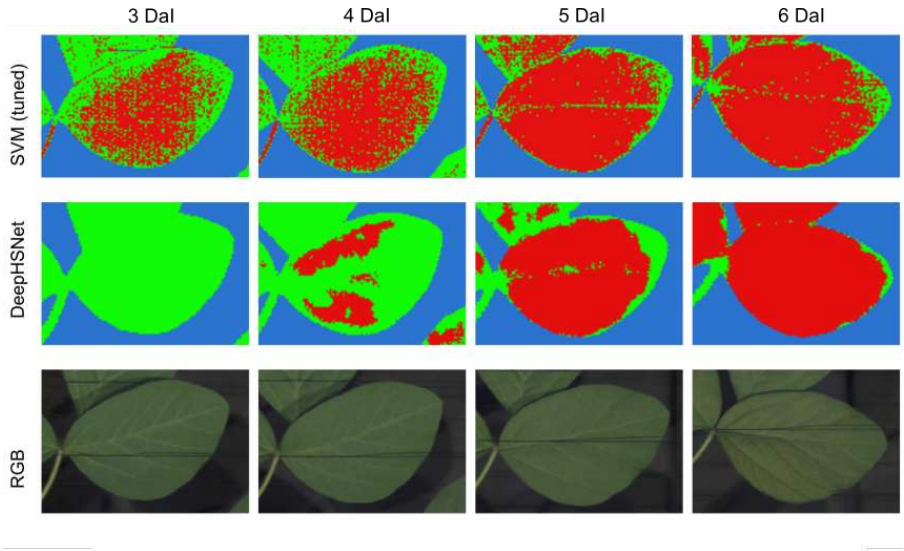


Figure 3.9: Pseudo RGB and false color images of an inoculated soybean leaf in experiment 2. The false color images represent the classification results of DeepHSNet and SVM, with: green = healthy plant tissue, red = disease symptoms, blue = background.

and 5 DaI via SVM did not correlate with observable symptomatic areas at later stages within the time-series measurement. Only at 6 DaI and after that, when symptoms were visible with the human eye, did the classification results start to match with the symptomatic areas of the leaves (see Fig. 3.8). Additionally, the SVM-based prediction could not reliably differentiate between symptomatic areas and leaf areas partially covered by the frame, as shown in Fig. 3.9 at 4 DaI and 8 DaI. In experiment 2, most leaves were classified as symptomatic from 3 DaI on. As the entire leaves eventually became symptomatic due to the high disease severity within the second experiment, the classification result matches the observed symptomatic areas (see Fig. 3.9). However, due to the nature of disease development, manual symptom assessment cannot be used to properly assess accuracy for experiment 2. Furthermore, the SVM based classification is prone to misclassify the soybean stems as symptomatic areas (see Fig. 3.9).

3.4.3 Discussion

In this section, the potential of hyperspectral imaging in combination with data analysis methods for early detection of ASR symptoms on soybean leaves has been investigated. Furthermore, supervised machine learning and neural networks have been compared for detection accuracy of *Phakopsora pachyrhizi* infection based on a limited set of annotated training data.

The two experiments performed within the study showed a typical and similar progression of ASR symptoms on soybean leaves, with first symptoms being visible to the

human eye at 6 DaI and 5 DaI for experiment 1 and experiment 2, respectively. A notable difference is, however the disease severity in the two experiments. While plants in experiment 1 had a relatively low disease severity at the end of the time-series measurement (10 DaI), soybean leaves in experiment 2 were nearly completely covered with symptoms (see Fig. 3.6). As both experiments used the same methodology and spore material for inoculation, there is no obvious explanation for this divergence between the experiments. One possible explanation could be that experiment 1 was performed in March while experiment 2 was performed in June. This might have influenced the spore germination rate despite the plants being kept under controlled conditions in the greenhouse.

The neural networks generally outperformed the machine learning methods among the applied data analysis methods. The classical machine learning methods achieved satisfying results after task-specific configuration fine-tuning. All methods were able to reach an accuracy well over 99%. The neural networks still achieved slightly better results without requiring task-specific fine-tuning and less training data.

SVM, which is widely used in multiple studies as a supervised data analysis method for plant disease detection (Huang *et al.*, 2019; Nagasubramanian *et al.*, 2018; Rumpf *et al.*, 2010), is often the first choice for this kind of classification task and also performed well in our experiments with fine-tuned configuration. As shown by the study of Thomas *et al.* (2022), SVM is a suitable tool for detecting brown rust symptoms on wheat leaves for Hyperspectral Images. This work has shown that SVM produces comparable results for ASR symptoms on soybean. Still, DeepHSNet predicted symptoms more accurately. After manual investigation of the classification results, the authors present the hypothesis that in the case of soybean leaves, a differentiation of ASR symptoms and the plant's leaf veins is the probable cause for the performance difference, as the respective spectral signatures of these features share high similarities (see Fig. A.1).

A further explanation for the difference, especially in the detection accuracy of early disease symptoms, is the selection of training data. While Thomas et al. selected training data specifically for optimized disease detection via machine learning methods, the current study focused on a limited set of training data from images with late disease stages. It did not split the resulting training data up as described in the previous study to optimize disease detection for the specific algorithms (Thomas *et al.*, 2022). The high variability of spectral signatures within each selected class might be more challenging for machine learning methods compared to neural networks (see Fig. A.1). Each of the three relevant classes (see Fig. 3.7) for a classification approach with results that are applicable for use in agricultural practice consists of a diverse set of plant features, disease symptom progression states, and background (see Fig. A.1). While it would have been possible to further separate these features into distinct classes, such an approach would not be well suited for practical application in phenotyping experiments or field environments. One of the biggest hurdles for applying hyperspectral imaging in agricultural practice at the time this study is conducted is the increased data variance through environmental factors in field and greenhouse applications compared to laboratory experiments (Lowe *et al.*, 2017; Thomas *et al.*, 2018b).

From the results of this study, it can be hypnotized that neural networks are better suited for complex classes with high spectral data variance in plant disease detection than classical machine learning approaches. Surprisingly the neural networks, especially DeepHSNet, could accurately classify the image data over the entire time series of both experiments, despite a limited amount of annotated training data. It is acknowledged in the scientific community that the requirement of large amounts of annotated data is one of the downsides of neural networks, which is based on the larger search space and model complexity of the neural networks. Nevertheless, in the current study, neural networks could classify the presented data over both experiments accurately and even detected disease symptoms before they became visible to the human eye.

The quality of the prediction results for SVM and DeepHSNet (see Fig. 3.8 and Fig. 3.9) differs significantly, especially for the early stages of infection. The symptomatic areas predicted by DeepHSNet match better with those identified by a human. The authors assume two components responsible for the better performance of DeepHSNet. The spatial information, which DeepHSNet utilizes, seems necessary for this task. It adds the pixel context into consideration. The second component is the higher model complexity. DeepHSNet can mimic more complex class boundaries in contrast to SVM. The current thesis is that the second component is more important than spatial information due to the quantitative results (see Tab. 3.5). The fully connected network has a higher model complexity than SVM but cannot use spatial information in the pixel-wise approach. Still, it outperforms SVM. Further, the performance boost between SVM and the fully connected network is more significant than between the fully connected network and DeepHSNet. DeepHSNet, with both components, outperformed the other models within the qualitative (see Fig. 3.8 and Fig. 3.9) and the quantitate evaluation (see Tab. 3.5).

The DeepHSNet-based classification results showed symptomatic areas in both experiments one day before they became visible to the human eye (see Fig. 3.8 and Fig. 3.9). While symptom detection prior to manual assessment is a key feature in hyperspectral imaging disease detection, which has been observed in multiple studies (Bauriegel and Herppich, 2014; Khan *et al.*, 2021; Wang *et al.*, 2019), it has been shown in previous studies, that these results are difficult to achieve with supervised data analysis methodology (Thomas *et al.*, 2018b). The main problem when applying supervised methods is that it is challenging for experts to label symptomatic areas of the leaves, especially under greenhouse and field conditions, which do not yet show visible symptoms. As symptoms cannot be observed directly, it is necessary to precisely measure the leaf to convey the leaf's area to be labeled from images at later points in the time-series (Bohnenkamp *et al.*, 2019). Alternatively, it is possible to use unsupervised methods to detect such areas under controlled conditions and use the resulting data as annotated training data to apply supervised methods under more complex circumstances.

In light of these facts, the early detection of *Phakopsora pachyrhizi* symptoms through the DeepHSNet classification with annotated data from visible symptoms in a training dataset with high class variability is a promising step for the application under field

conditions in agricultural practice, where a multitude of environmental factors increase the complexity of the data set.

3.4.4 Conclusion

The results of this study show that symptoms of ASR on soybean leaves can be detected accurately via analysis of HSI. Despite specifically limited training data selection in late-stage disease development with high in-class data variability, both machine learning and deep learning methods could detect and identify disease symptoms with over 99% accuracy. However, the deep learning methods outperformed the machine learning methods for early disease detection applications while accurately detecting disease symptoms about one day before they became visible to the human eye. This shows the potential of HSI in combination with deep learning approaches for practical application in agriculture on field level, where a high data variability is imposed on the measurements due to environmental factors.

3.5 Summary

In this chapter, hyperspectral imaging was used for two applications of *food inspection/-precision farming*. We showed that a non-destructive ripeness prediction of avocados and kiwis is possible. Further, hyperspectral imaging allows reliable detection of *Phakopsora Pachyrhizi* infestations on soybean.

Further, a convolution neural network *DeepHSNet* was proposed and validated on the hyperspectral task. As this network showed reliable results, it will be used as a baseline for our further investigations in the following chapters (see chapter 4 and chapter 5).

Chapter 4

Self-Supervised Learning for Hyperspectral Classification

In this chapter, self-supervised learning for hyperspectral imaging is discussed based on our published work:

- Varga, L. A., Frank, H., and Zell, A. (2023a). Self-supervised pretraining for hyperspectral classification of fruit ripeness. In J. Beyerer, T. Längle, and M. Heizmann, editors, *OCM 2023 - Optical Characterization of Materials : Conference Proceedings*, pages 97–108

4.1 Introduction

As discussed in the previous chapter, knowing the ripeness of fruit is of great interest in the food industry. For this, chemical and physical indicators like the sugar content and fruit flesh firmness are usually employed, all of which are obtained by destructive measurement.

It is also possible to predict the ripeness of fruit using *Hyperspectral Imaging (HSI)* and supervised methods (see chapter 3). However, the supervised manner requires labels. Obtaining the actual ripeness state of a fruit still comes with destroying it, making the labeling process tedious and labeled samples scarce. Training networks on small training sets can be challenging, and overfitting becomes likely. Therefore, it is desirable to also use unlabeled recordings that can be obtained without much effort.

Self-supervised Learning (SSL) methods have produced astonishing results in computer vision (e.g. *SimCLR* by Chen *et al.* (2020), *SimSiam* by Chen and He (2021), *Barlow Twins* by Zbontar *et al.* (2021)). As there is no pretraining step for hyperspectral recordings, we evaluate in this chapter the effect of SSL-based pretraining in the case of hyperspectral image classification, which can stabilize the training and improve the network's predictions. So, we apply three state-of-the-art approaches on the hyperspectral data set of ripening fruit. As most current SSL approaches target color images, we must adapt the hyperspectral recordings methods, especially the data augmentations need modifications. Further, we test the effect on three network architectures.

4.2 Related Work

For works related to the HSI classification or the ripeness prediction of fruit, we refer to Sec. 3.1, as HSI classification is currently mostly done with supervised approaches. Herefore, the relevant works are similar.

Self-supervised learning, on the other side, has been very popular in recent years. SSL allows the network to learn useful representations of the data by solving a pretext task that can help perform the actual downstream tasks. Various self-supervised learning methods have been developed.

Most of them rely on Siamese network architectures, as proposed by Bromley *et al.* (1993), that are trained to maximize the similarity between their outputs for two different distortions of the same sample. The main challenge is preventing collapsing, which occurs when the network branches ignore their two inputs and produce identical or constant output vectors.

Recently, various approaches have occurred: Contrastive learning (Hadsell *et al.*, 2006) methods like, e.g. *SimCLR* by Chen *et al.* (2020), avoid collapse by repulsing negative pairs (views of different images), in addition to attracting the positive pairs (views of the same image). The works Wu *et al.* (2018), He *et al.* (2020) and Chen *et al.* (2020) followed this approach. However, these methods either require a large memory bank or large batch sizes.

Other, more recent methods do not rely on negative samples. Instead, they introduce asymmetry to avoid collapse. For example, the *SimSiam* method by Chen and He (2021) uses an additional network and a stop-gradient operation in one branch, respectively. An alternative approach is provided by so-called information maximization methods, which were, e.g., proposed by Zbontar *et al.* (2021) Ermolov *et al.* (2021) and Bardes *et al.* (2022). They decorrelate the output vectors of the two branches, maximizing their information content and therefore avoiding (informational) collapse. *Barlow Twins* by Zbontar *et al.* (2021) is one prominent example.

All of those methods are initially designed for regular color images. Research on self-supervised pretraining when using hyperspectral images is still in its infancy. Only a few approaches have been proposed very recently (e.g., the works of Yue *et al.* (2022), Zhao *et al.* (2022) and Hou *et al.* (2022)), all addressing hyperspectral remote sensing scenes. Hence, their use-case differs widely from the classification of fruit. They focus on a patch-based task in contrast to a sample-based task, so a direct comparison is not possible.

4.3 Experiments

This section describes the extension of data set *DeepHS Fruit*. Further, the methods used for the experiments are defined. Finally, the evaluation metric of the experiments are discussed.

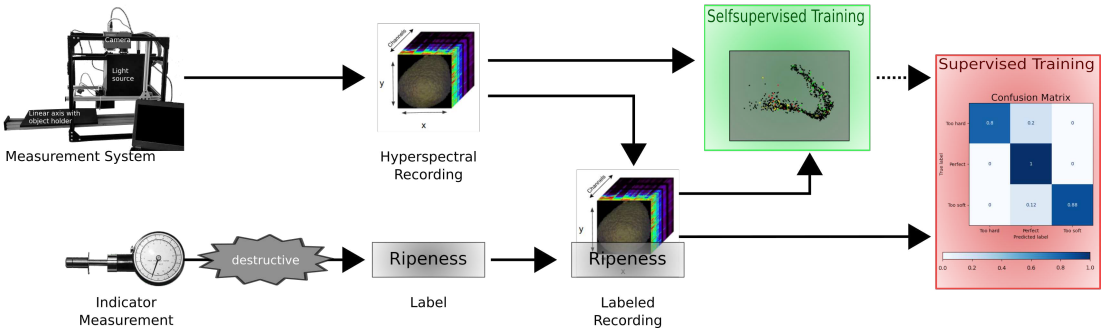


Figure 4.1: The fruits were recorded by HSI. Some of them were selected and measured destructively to obtain the corresponding ripeness label. The other fruit recordings remained unlabeled. First, the initial model was pretrained using self-supervised learning (SSL) on all recordings, not requiring any labels. Then, we set it up for the actual classification task and further fine-tuned the model by supervised training, using only the labeled data.

4.3.1 Data Set

This work extended the already publicly available hyperspectral fruit data set, *DeepHS Fruit* (see chapter 3.3.4, by additional recordings of avocados, kiwis, mangos, persimmon, and papayas. We used the same measurement setup and followed the procedure described for the first version. Each fruit was recorded by the *Specim FX 10* with 224 bands (398 nm - 1004 nm) and the *Corning microHSI 410 Vis-NIR Hyperspectral Sensor* with 249 bands (408 nm - 901 nm). Labels (firmness, sugar level, and overall ripeness) were obtained by destructive measurement. As in version 1, the labeled data was divided into a fixed training set ($^3/_4$), validation set ($^1/_8$), and test set ($^1/_8$).

The resulting *DeepHS v2* data set consists of 4671 recordings in total, 1018 labeled. For supervised classification, only the labeled subset was used. For self-supervised pretraining, the unlabeled samples were also used. Here, the samples were divided into a training ($^4/_5$) and a validation set ($^1/_5$). Only the Specim and Corning camera recordings were used for the experiments. Further, the size of the classes in the three categories was balanced, so there was no bias towards one class.

4.3.2 Models

Three different classifier models were used: *DeepHSNet*, which we already evaluated on the supervised task (see chapter 3), a proposed hybrid model and *ResNet-18* of He *et al.* (2016).

DeepHSNet In chapter 3, we proposed the *DeepHSNet* network, specialized for HSI data and evaluated on two hyperspectral applications. It is a small *Convolutional Neural*

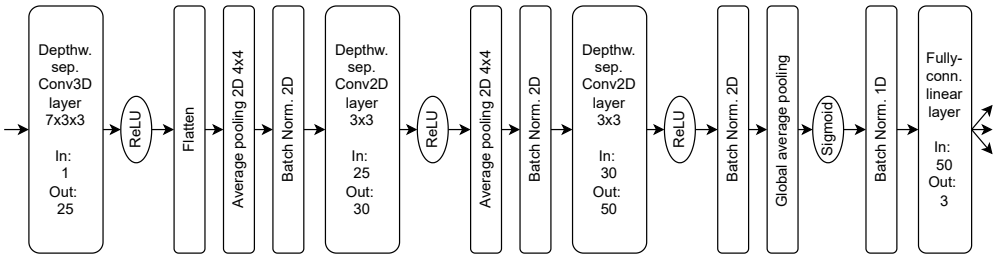


Figure 4.2: Architecture of the 3D-2D hybrid model.

Network (CNN) with three 2D convolutional layers for feature extraction and a fully-connected linear layer for actual classification.

3D-2D Hybrid Model For the self-supervised experiments, we propose a slightly modified variant, a hybrid model, using a 3D convolution instead of a 2D convolution in the first layer, which was inspired by the *HybridSN* by Roy *et al.* (2020b). The first layer has been extended to a three-dimensional $7 \times 3 \times 3$ kernel. The architecture is shown in Fig. 4.2.

Overall, the resulting 3D-2D hybrid model comprised of a 3D convolutional layer for spectral-spatial feature learning, followed by two 2D convolutional layers for more abstract spatial feature learning, and finally the fully-connected layer, again operating on the spectral dimension, for actual classification. As a consequence of using the 3D convolution, we obtained a much larger model concerning the baseline ($\approx$ 20 times as many parameters)

ResNet-18 Finally, we also evaluate our methods using a *ResNet* architecture of He *et al.* (2016), since it has proven good performance on image classification tasks in general, as shown in the work He *et al.* (2016) and is also commonly used as backbone for SSL (e.g. Chen *et al.* (2020); Chen and He (2021); Zbontar *et al.* (2021)). Here, the *ResNet-18* was used. It is a deep convolutional neural network with skip connections and 18 layers, making it the smallest member of the *ResNet* family. Nonetheless, compared to the other two models, it is more complex and has significantly more parameters ($11,900,000$), but has no 3D convolutions.

4.3.3 Self-supervised Pretraining

The pretraining (as shown in Fig. 4.1) was performed using one of the three SSL methods: *SimCLR* (Chen *et al.*, 2020), *SimSiam* (Chen and He, 2021), *Barlow Twins* (Zbontar *et al.*, 2021).

All employ a siamese network architecture, like described by Bromley *et al.* (1993), where each branch is built by the encoder, the convolutional part of the classifier model, followed by a projection head. For the latter, we used an *Multi Layer Perceptron (MLP*)

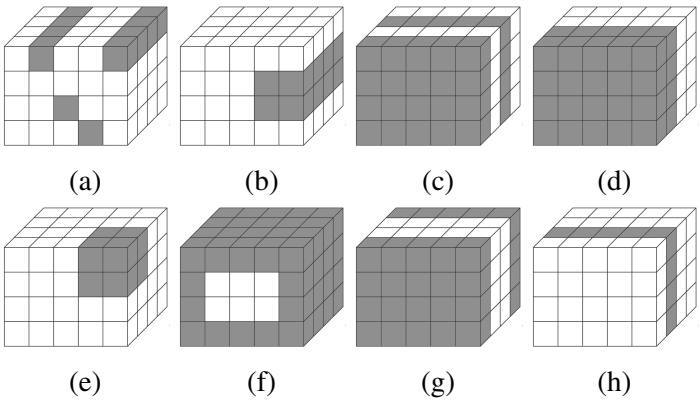


Figure 4.3: Augmentations visualized on a hyperspectral data cube: Modification of **(a)** random pixels, **(b)** consecutive pixels, **(c)** random channels, **(d)** consecutive channels, **(e)** a "subcube" (consecutive pixels and channels), **(f)** edge pixels, **(g)** edge channels, and **(h)** color information channels. Modified pixels and channels are marked in grey.

with two layers. A ReLU non-linearity and batch normalization (Ioffe and Szegedy, 2015) was applied for each layer. The input dimension was 50 (for the baseline or hybrid model, and 512 for the *ResNet-18*), the hidden dimension was 16, and the embedding dimension was eight. For *SimSiam*, we used an additional prediction MLP, consisting of a single linear layer with input and output dimension of eight. The temperature parameter for *SimSiam* was $\tau = 0.1$. For *Barlow Twins*, a weighting factor $\lambda = 0.01$ was used.

A critical component of SSL are the data augmentations. We evaluated 21 augmentation techniques (the full list is visible in Tab. A.1), including four basic image transformations (rotating, flipping, cropping, random noise), two more specific ones (wavelength-dependent noise and pixel-wise intensity scaling), 13 augmentations that modify parts of the hyperspectral cube (i.e., drop or blur specific pixels, channels, or an entire sub-cube (Haut *et al.*, 2019)), as well as two mixing augmentations (inspired by *MixUp* (Zhang *et al.*, 2018) and *ScaleMix* (Wang *et al.*, 2022b)). Representatives of each category are visualized in Fig. 4.3.

Based on the ablation studies (see Sec. 4.5.3), only a subset of the augmentations (random rotations with probability 50%, random cropping with probability 30%, modification of the hyperspectral cube, and mixing with probability 20%) was actually used for pretraining of the final experiments.

The networks were optimized with SGD, defined by Kiefer and Wolfowitz (1952), with a weight decay of 10^{-4}, a momentum of 0.9, and a learning rate of 10^{-2}, decayed with the cosine decay schedule without restart, as desribed by Loshchilov and Hutter (2017). We trained for 80 epochs with an effective batch size of 32.

4.3.4 Evaluation

For the evaluation of self-supervised pretraining, the produced embeddings were considered. They were evaluated qualitatively (based on 3D visualizations) and quantitatively (based on the k-Nearest-Neighbor accuracy). For the visualization, the feature values of the embedding were plotted in three-dimensional space after applying PCA. k-NN classification, as proposed by Fix and Hodges (1989), was employed for the embedded labeled samples, using $k = 5$, the cosine distance and leave-one-out cross-validation. A similar evaluation metric was already used by Wu *et al.* (2018) and Chen and He (2021).

Additionally, we measured the performance for classification without and with pretraining.

For the pretrained model, fine-tuning instead of full downstream training was performed. Fine-tuning refers to training the model on the downstream classification task by using the pretrained parameters as an initialization. For this, the whole model was built based on the pretrained encoder and a randomly initialized fully-connected part. Then the entire model was trained on the classification task. However, early experiments showed that training the classifier with regular settings (as without pretraining) would simply overwrite the pretrained backbone weights and lead to similar classification results for the initial and pretrained model. Therefore, the training procedure for the pretrained classifier was adapted to:

1. Freeze the backbone and train only the weights of the fully connected part of the network (with the same learning rate and the number of epochs as for the default case or linear evaluation).

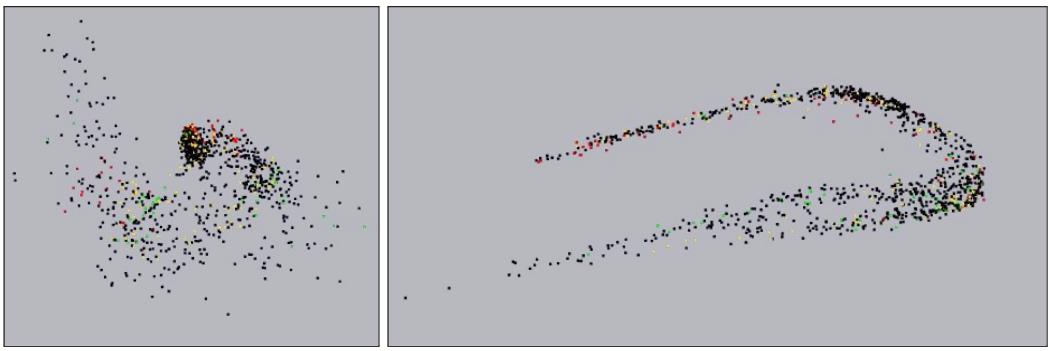
2. Fine-tune the weights of the whole network using a smaller learning rate and shorter training, respectively.

The first step mainly served us to temporarily assign meaningful weights to the otherwise randomly initialized fully-connected layer and therefore stabilize the gradients for further fine-tuning of all weights. The second step of fine-tuning on the labeled data per se is commonly used in the SSL literature, e.g., by Chen *et al.* (2020); Chen and He (2021); Zbontar *et al.* (2021); Caron *et al.* (2020); Grill *et al.* (2020). However, in contrast to most of the literature, since the data set used here is already relatively small, we decided to use the whole instead of only a small fraction of the labeled training data. Again, all other settings remain unaltered from the default case.

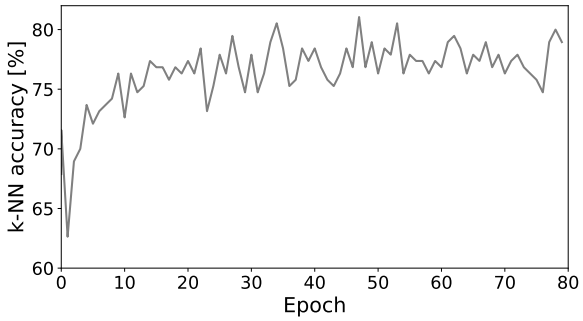
Without pretraining, the randomly initialized model was trained using the settings of the supervised task (see Sec. 3.3.5).

After the supervised training, the model was evaluated on the test set. Test time augmentations (Howard, 2014) were applied with probability 50%.

Using five different seeds each, we conducted experiments for all possible combinations of fruit types, cameras, and categories.



(a) Embedding, before (left) and after pretraining (right).



(b) k-NN accuracy.

Figure 4.4: **(a)** 3D visualization of the embedding before and after pretraining via *Barlow Twins* – coloring by ripeness levels: unripe (green), ripe (yellow), overripe (red) and unlabeled (black). **(b)** k-NN accuracy on the ripeness levels of the labeled samples (train and validation set) during pretraining with *SimCLR*. For the hybrid model and the avocados, recorded by the *Specim* camera.

4.4 Results

To evaluate the pretraining per se, we visualized the embeddings in 3D and monitored the k-NN accuracy during pretraining (see Fig. 4.4).

The spatial arrangement in the 3D space correlates with the ripeness level; samples of the same ripeness level are brought closer together. This fits the development of the k-NN accuracy, which increases as pretraining advances and finally converges towards 80%. This shows that pretraining can extract meaningful features and find useful representations for the data without using label information.

Additionally, the pretrained model was evaluated on the downstream classification task. Especially, classification performance with pretraining and additional fine-tuning was compared to classification without pretraining. We present the classification accuracy per fruit in Tab. 4.1. The pretraining led, for all examples, to a performance improvement. We achieved an overall classification accuracy of 58.3%. Comparing the baseline

model initially designed for pure classification to our newly proposed hybrid model with pretraining, overall, we could observe an improvement of approx. 3% in classification accuracy. For some fruit, it could be increased by more than 10%. Where this was not the case, the *Interquartile range (IQR)* was reduced, indicating that pretraining increased stability.

The experiments, visible in Fig. 4.5, show that pretraining even could compensate for the need for large amounts of labeled samples.

Table 4.1: Classification accuracies (median, IQR) for regular classifier training versus *SimCLR* pretraining plus fine-tuning, for the *HS-CNN* (baseline) and hybrid model. One example for the five different fruit: Avocado (ripeness, *Specim*), kiwi (sugar, *Specim*), mango (firmness, *Specim*), kaki (sugar, *Specim*), papaya (ripeness, *Corning*), and over all fruit, categories and camera types. Highest accuracies in **bold**.

		Avocado	Kiwi	Mango	Kaki	Papaya	Overall
Baseline	Without pretraining	83.3% (±4.2%)	65.2% (±4.3%)	50.0% (±33.3%)	50.0% (±4.3%)	77.8% (±11.1%)	55.6% (±32.2%)
	With pretraining	87.5% (±0.0%)	73.9% (±8.7%)	50.0% (±8.3%)	**66.7%** (±8.7%)	**88.9%** (±0.0%)	**58.3%** (±32.2%)
Hybrid	Without pretraining	75.0% (±4.2%)	73.9% (±13.0%)	50.0% (±33.0%)	58.3% (±13.0%)	**88.9%** (±11.1%)	54.2% (±33.3%)
	With pretraining	**91.7%** (±4.2%)	**78.3%** (±4.3%)	50.0% (±16.7%)	58.3% (±4.3%)	**88.9%** (±11.1%)	**58.3%** (±36.1%)

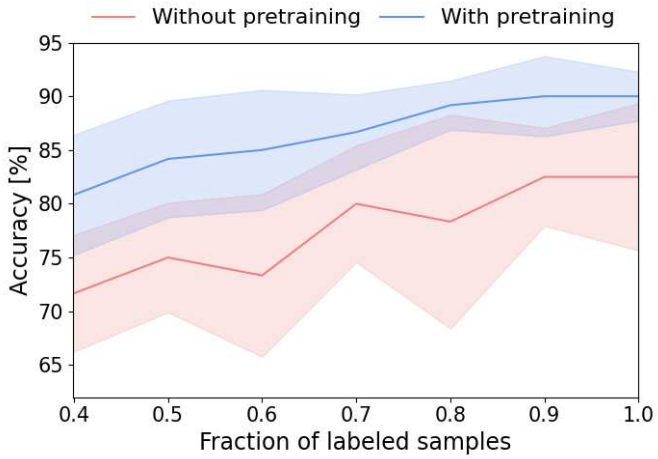


Figure 4.5: Classification accuracy (mean and standard devation) versus fraction of labeled samples used for classifier training for the baseline model with default classifier training (red) and hybrid model with pretraining (via *SimCLR*) plus fine-tuning (blue). Example: Avocado, *Specim* camera, ripeness classification.

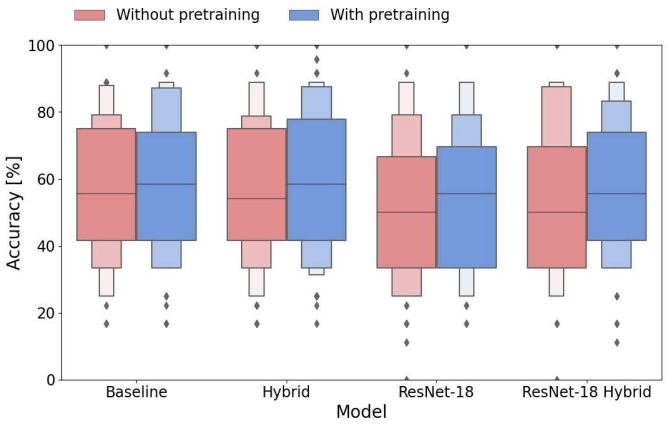


Figure 4.6: Classification accuracies for the *DeepHSNet* baseline, *ResNet-18* model, and a hybrid version of both models without pretraining (red) and with pretraining via *SimCLR* (blue).

4.5 Ablation Study

In this section, the impact of the classifier model, the SSL method and the augmentation techniques is analyzed.

4.5.1 Classifier Model

For each of the three models, the classification accuracy with and without pretraining is visualized in Fig. 4.6. Further, a hybrid version of the *ResNet-18* model was tested.

For classification without pretraining, the *DeepHSNet* performs best among all three models (55.6% accuracy). With pretraining, the performance can be improved only by a small amount, probably due to the affected backbone extracting only spatial and no spectral features. The later FC layers, which process the spectral information, are most important for classification, whereas the backbone with the 2D convolutions, extracting spatial features, hardly matters. Since in our experiments, only the backbone was pretrained, but the FC layer for actual classification had to be trained from scratch on the classification task anyway, it was hard or even impossible to obtain significant improvements using the baseline model.

With a hybrid model of the *DeepHSNet*, we tried to overcome this issue by using a 3D instead of 2D convolution to include the spectral information already in the backbone and therefore give it more relevance. With 54.2% classification accuracy for regular classification, the modified hybrid model performs slightly worse than the baseline, caused by overfitting. For the case with pretraining, the hybrid model performs equal to the baseline model (58.3%), but more importantly, the accuracy improved by a larger amount of

4.1%. So, as intended, pretraining is more effective for the hybrid variant.

Finally, we considered the even larger and more complex *ResNet-18*. With a classification accuracy of only 50%, the *ResNet-18* performs much worse than the other two models for standard classification. Again, we hold overfitting responsible for the bad performance. However, the *ResNet-18* does outperform both models concerning the most significant improvement by more than 5% relative to without pretraining. Further, we evaluated whether the hybrid modification, which was used for *DeepHSNet*, could also be helpful for the *ResNet-18* architecture. The influence seems neglectable. We assume, as the *ResNet-18* backbone is much larger, it can incorporate the spectral features learned during pretraining.

Overall, it has been found that pretraining improved the classification accuracy relative to regular classification for all models. There was a correlation between the model size and classification performance: The improvement is more significant for larger models, as these benefit more from the pretraining. Based on those observations, we claim that pretraining stabilizes the subsequent classifier training, can prevent overfitting, and enables the training of larger models for HSI.

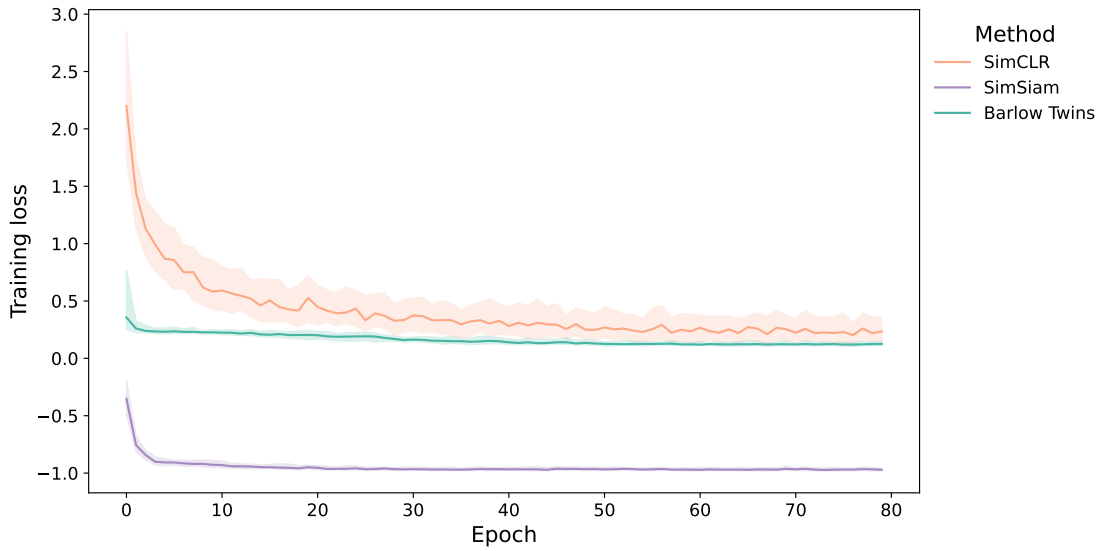


Figure 4.7: Training loss for pretraining via *SimCLR* (orange), *SimSiam* (violet), *Barlow Twins* (green) using the hybrid model. Example: Avocado, *Specim* camera, on the ripeness classification task.

4.5.2 Self-supervised Pretraining Method

Secondly, we compare the three employed self-supervised methods (*SimCLR*, *SimSiam*, and *Barlow Twins*). All use a Siamese Network structure, but they mainly differ in their

way of collapse prevention. While *SimCLR* relies on negative samples and, therefore, must use large batches, as described by Chen *et al.* (2020). *SimSiam* was designed by Chen and He (2021) with an asymmetry (Chen and He, 2021) to avoid trivial solutions. *Barlow Twins* of Zbontar *et al.* (2021) does this by the construction of the loss. So the latter do not require large batches of samples, nor do *SimCLR* or *Barlow Twins* have any particular asymmetry in the Siamese Network structure.

Further, of course, the loss function is an essential aspect of the method. *SimCLR* uses the *InfoNCE* or *NT-Xent* loss, as defined by Chen *et al.* (2020). *SimSiam* by default employs the (negative) cosine similarity, as proposed by Chen and He (2021). For the *Barlow Twins* method, Zbontar *et al.* (2021) considered the similarity of the cross-correlation matrix to the identity matrix. While *SimCLR* and *SimSiam* have a symmetric loss, for *Barlow Twins*, no loss symmetrization is employed.

Fig. 4.7 shows the development of these losses over the pretraining process for the example of avocado ripeness classification using the *Specim* camera recordings. The *NT-Xent* loss of the *SimCLR* method shows the most variance relative to the other losses. It starts at the highest value and, on average, converges at a value slightly below 0.5, taking much more training epochs than the other two methods. *Barlow Twins* with its loss based on the cross-correlation matrix, very early reaches a small training loss value around 0.2, not being able to decrease further towards the minimum possible value of 0. In contrast to the other two methods, *SimSiam* measures its loss as the negative cosine similarity that can have values between -1 and $+1$. Considering this, the training loss looks quite good, converging very quickly towards a value near the minimum.

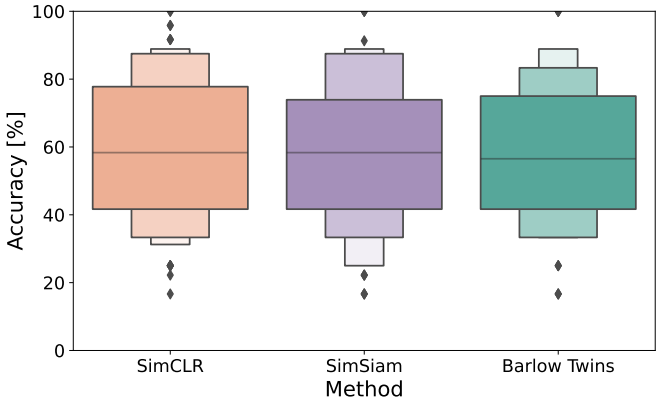


Figure 4.8: Classification accuracies for pretraining via *SimCLR*, *SimSiam*, *Barlow Twins* using the hybrid model. Over all fruit, categories and both cameras.

Although their training loss curves look very different, in the end, the performance of the three methods on the ripeness classification task is relatively similar. The classification accuracies over all fruit, categories, and cameras are visualized in Fig. 4.8, for pretraining with either *SimCLR*, *SimSiam* or *Barlow Twins*, respectively. Over all fruit,

categories, and both cameras together, *SimCLR* performs best, slightly better than *SimSiam* which both have a median classification accuracy of 58.3%. *Barlow Twins* obtains a lower median accuracy of 56%. It is also worth mentioning that the different methods perform differently for specific examples. There is not one single "best" method for pretraining, at least for this application.

However, at least some interesting observations could be made concerning their theoretical background. First of all, the methods are often compared for asymmetry. It is not possible to make a clear statement about this here since an asymmetric and symmetric method (*SimSiam* and *SimCLR*) demonstrated nearly the same performance while another symmetric method (*Barlow Twins*) performed worse. Because of the good performance of *SimCLR*, we assume that contrastive learning, using negative samples, was indeed helpful and can not simply be replaced by asymmetry or other collapse prevention approaches. However, this contradicts what Chen and He (2021) claims for *SimSiam*. For *SimCLR*, Chen *et al.* (2020) state the requirement of large batches of contrastive samples and, therefore theoretically, should perform poorly on small batch sizes. In contrast, *Barlow Twins*(Zbontar *et al.*, 2021) and *SimSiam*(Chen and He, 2021) claim to not require large batches. However, for an effective batch size of 32, we observed almost the contrary: *SimCLR* outperformed the other two methods with respect to the resulting classification accuracies. Still, it could explain the high variance in the training loss for *SimCLR*. According to their authors, the methods further have different requirements regarding model size or dimensionality of embeddings. Zbontar *et al.* (2021) observed improved performance for increased dimensionality of the embeddings. Therefore, the small embedding dimension of eight chosen for this application is a possible explanation for the relatively bad performance of the *Barlow Twins* method. Analogously, Chen *et al.* (2020) of *SimCLR* claim their method greatly benefits from bigger models. Surprisingly, *SimCLR* worked relatively well on our rather small baseline and hybrid model. However, it may explain the observations that pretraining (via *SimCLR*) is more effective for larger models. Apart from the batch size and model architecture, all other settings were chosen to be the same over all SSL methods for our experiments, like the same learning rate, and optimizer. Also, the very same set of augmentations was applied with the same probabilities, whereas it is valid to assume that the individual methods would have benefited from using their best-suited settings, respectively.

4.5.3 Augmentations

Further, we evaluated the influence of the 21 proposed data augmentation techniques by grouping them and using only one group for pretraining, respectively. Fig. 4.9 shows the resulting classification accuracies for the avocado fruit as a representative example.

The basic augmentations (rotating, flipping, cropping, and cutting) showed the highest accuracy ($> 80\%$) and therefore seemed to be most important. The pixel augmentations, like the modification of edge pixels and dropping random or consecutive pixels, were also helpful for pretraining. On the other hand, dropping multiple consecutive channels led

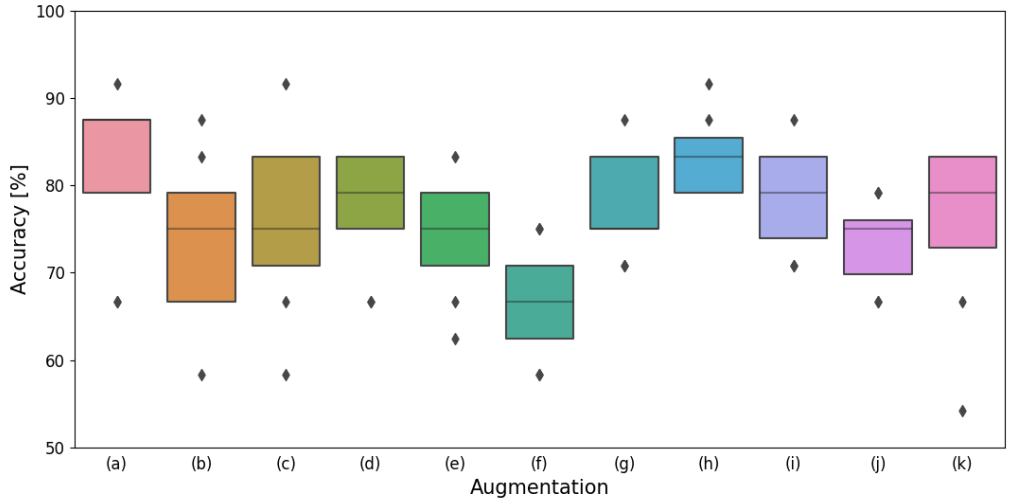


Figure 4.9: Classification accuracies for self-supervised pretraining (via *SimCLR*) using only the group of **(a)** basic augmentations, **(b)** noise augmentations, **(c)** augmentations that blur or drop random pixels, **(d)** drop consecutive pixels, **(e)** blur or drop random channels, **(f)** drop consecutive channels, **(g)** drop a sub-cube, **(h)** blur or drop edge pixels, **(i)** blur or drop edge channels, **(j)** blur or drop visible color information channels and **(k)** mixing augmentations. Over all three SSL methods. Example: Avocado, *Specim*, ripeness classification.

to the worst classification accuracy ($< 70\%$). Also, dropping or blurring color channels decreased performance, since color information was indeed used by the network (e.g. green vs. brown color for avocado, affecting the wavelengths between $400 - 700$ nm, was also important in our other experiments for this task).

The general trend with respect to channel augmentations would be that they can be helpful to make the model more robust to potentially noisy channels. Still, they can also be harmful when taking away too much important information or distorting the spectrum. Altering the spectrum curves and distorting the spectrum may also have been the case for adding any random noise to the whole data cube. The noise augmentations also showed a relatively low classification accuracy. We have shown that including knowledge about the data set and the specific application is very effective. Augmentations can be used in SSL to "tell" the model which parts of the data cube are specifically important (by leaving them unaltered) and which are not interesting by altering them the most. We found that for hyperspectral image data, it makes more sense to introduce distortions systematically instead of completely random (e.g. `BlurEdgePixels` versus `RandomNoise`).

Of course, the composition of the individual augmentations may also be crucial for pretraining and, therefore, the classification performance. However, there are combinatorially many cases, and in additional experiments, it was observed that the augmentations performed rather inconsistent for individual methods, fruit, and between camera

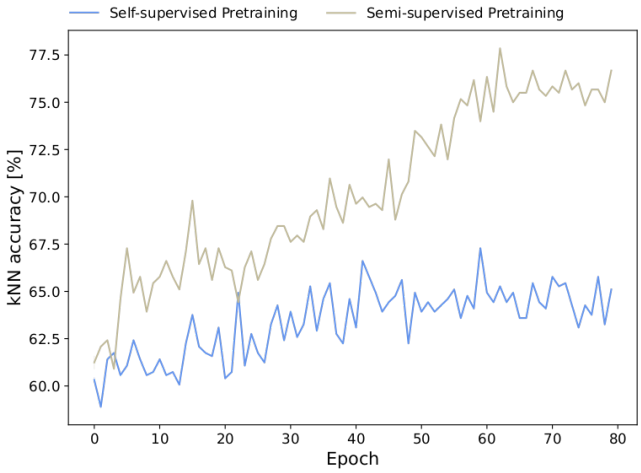


Figure 4.10: k-NN accuracy on the ripeness labels (of the labeled training and validation samples) during pretraining on *Specim* recordings of all fruit using the *SimCLR* method, for the self-supervised (blue) vs. semi-supervised (brown) variant.

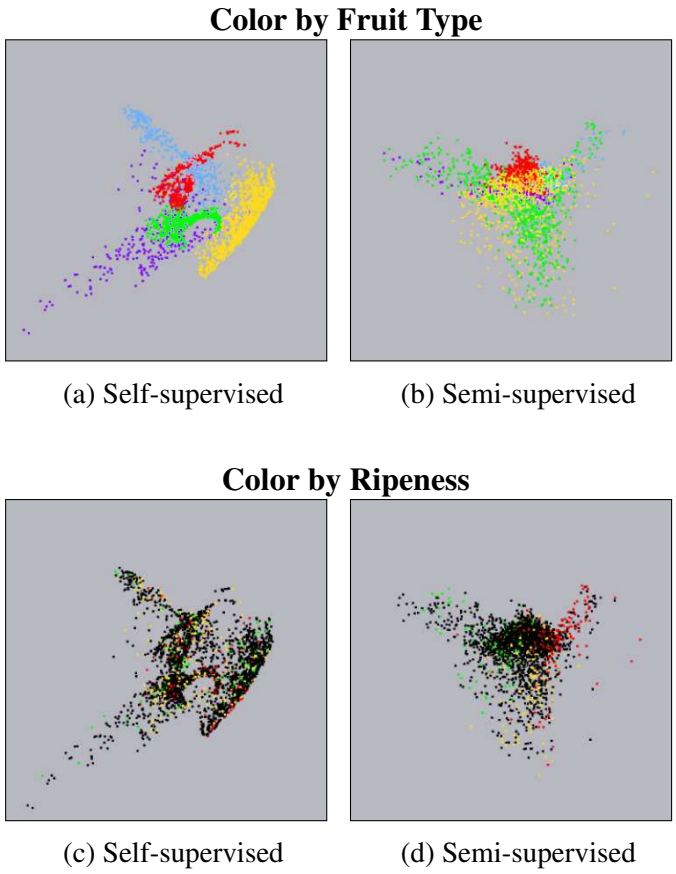
types, so there was no single "best" combination of augmentations to be found. For our evaluations, instead, we simply picked and combined the best-performing individual augmentations, partially with a lower probability, and found that this did work well in general.

4.6 Semi-supervised Pretraining

This section introduces a semi-supervised variant for pretraining that makes additional usage of the available label information. The goal is to steer the pretraining in the desired direction by emphasizing that only the features determining the respective level of the current category (ripeness, firmness, or sugar) are relevant, but similarities based on other aspects are not.

To find out whether using the label information made sense in practice, we applied it to the example of pretraining on all fruit types. This shows a situation where additional information can be helpful. Here, in contrast to the previous experiments, the models were trained on the recordings of all fruit types instead of only one. Resulting in a model that can classify the fruit's ripeness level regardless of the fruit type (with respect to the fruit types in the data set). In practice, models trained for a specific fruit type are enough. But still, this is theoretically an interesting task as the model could generalize the essential features between fruit types even if they are, in most cases, completely different.

Fig. 4.10 shows the k-NN accuracy development and Fig. 4.11 visualizes the resulting embeddings, using self-supervised and semi-supervised pretraining, respectively.



(a) Self-supervised (b) Semi-supervised

(c) Self-supervised (d) Semi-supervised

Figure 4.11: Embedding after (a, c) Self-supervised versus (b, d) Semi-supervised Pretraining using the *SimCLR* method on *Specim* camera recordings of all fruit. Coloring by (a, b) fruit type: avocado (green), kiwi (yellow), mango (red), kaki (blue) and papaya (purple) or (c, d) ripeness: unripe (green), ripe (yellow), overripe (red).

During self-supervised pretraining, the k-NN accuracy improved, but only by a relatively small amount of 5%, reaching a maximum very early. In contrast, when using the label information in addition in the context of semi-supervised pretraining, it increased more drastically and reached a high level of $> 77\%$. This was expected, since bringing samples of the same labels together as an immediate consequence makes them "nearest neighbors" and therefore should increase the k-NN accuracy (compare to Sec. 4.3.4).

This is also in accordance with the resulting embeddings, visualized in Fig. 4.11. For the self-supervised case (Fig. 4.11a and 4.11c), there is a clear separation between fruit types. Due to the clustering by fruit types, the ripeness levels are mixed up, and separation by ripeness level is impossible. Separation by the fruit type is more prominent, which let the models focus on this criterion. Semi-supervised pretraining (Fig. 4.11b and 4.11d), by also considering the ripeness labels, was able to find an embedding where the three ripeness levels are separated and in turn, of course, the fruit types got mixed up.

Table 4.2: Classification accuracies (median, in percent) for the hybrid model and pretraining using the self-supervised versus the semi-supervised approach. For the five examples (1. avocado, *Specim*, ripeness, 2. avocado, *Specim*, firmness, 3. avocado, *Corning*, ripeness, 4. kiwi, *Specim*, sugar, 5. papaya, *Corning*, ripeness) and over all fruit, cameras and categories.

	1.	**2.**	**3.**	**4.**	**5.**	**All**
Self-supervised Pretraining	83.3 %	88.9 %	88.9 %	78.3 %	77.8 %	56.5 %
Semi-supervised Pretraining	83.3 %	88.9 %	88.9 %	78.3 %	**88.9 %**	**58.3 %**

The additional usage of the label information apparently helped the model to decide on features determining the ripeness level to be important while ignoring similarities and differences based on the fruit type. Motivated by those results, we implemented semi-supervised pretraining also for our regular experiments.

To include the label information in an originally unsupervised setting, we needed to load the data in pairs. For all samples, pairs of the same sample, and additionally for a fraction (here: 20%) of the labeled samples, all pairs of two samples sharing the same label were generated.

Consequently, the loss was computed on (views of) the two samples in the pair instead of the two views of the same sample. In general, this semi-supervised loss could be denoted as

$$\mathcal{L} = \mathcal{L}_u + \mathcal{L}_s \tag{4.1}$$

where $\mathcal{L}_u$ is the unsupervised loss as used for self-supervised pretraining, here calculated based on the pairs of equal samples, and $\mathcal{L}_s$ is the supervised loss computed on the pairs of different but equally labeled samples. The model was trained to maximize the similarity for pairs of augmentations of the same sample and for different samples with the same label.

For comparability, we loaded the data in pairs for both semi-supervised and self-supervised pretraining.

Tab. 4.2 shows the resulting median classification accuracy for the five examples and over all fruit, classification categories and both cameras for the hybrid model with self-supervised vs. semi-supervised Pretraining via *SimCLR*, respectively.

The accuracy stayed the same for four out of the five examples. For the fifth example, there is an improvement in classification performance for semi-supervised relative to self-supervised pretraining. Also, over all fruit, categories, and cameras, the accuracy could be increased by about 2%.

The addition of label information has a positive impact on the pretraining, especially concerning the k-NN accuracy and the generated embeddings. However, the resulting classification accuracies that we are actually interested in the end, could only be slightly improved. Actually, semi-supervised pretraining brings another problem: The necessity to load the data in pairs and load the pairs of labeled samples in addition which is com-

putationally more expensive. Therefore and because the method would then not strictly be "self-supervised" any more, we decided to not use it for the base case. However, the semi-supervised approach could be interesting for further research.

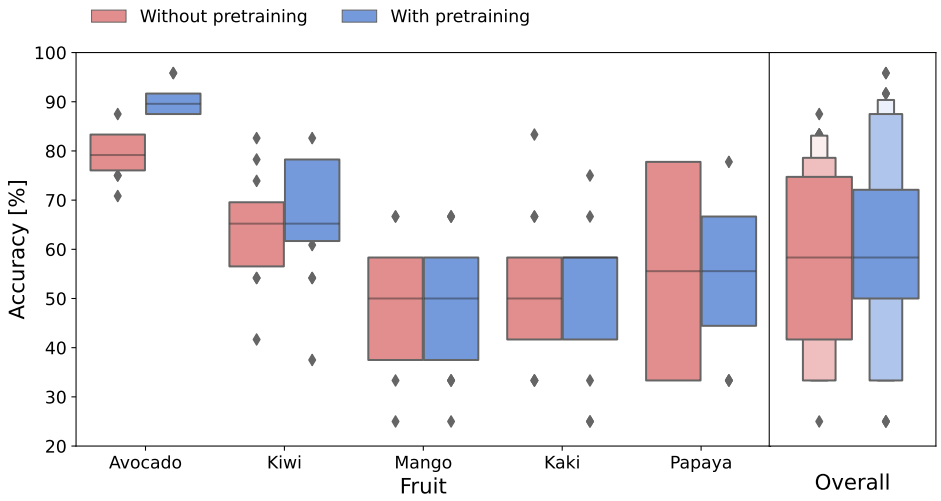


Figure 4.12: Classification accuracies for the baseline model without pretraining (red) versus the hybrid model with *SimCLR* pretraining (blue). For the *Specim* camera and the five different fruit (avocado, kiwi, mango, kaki, papaya), classified by all three categories (ripeness, firmness and sugar content).

4.7 Conclusion

This chapter extended the hyperspectral data set of ripening fruit by two new measurement series and three new fruit types.

Further, we showed that it is possible to transfer the ideas of SSL to hyperspectral data. SSL pretraining extracts essential features in an unsupervised manner and allows using larger models. It can stabilize classifier training and improves classification accuracy in some situations. Therefore, pretraining can partially compensate for the need for large labeled data sets in HSI classification.

Fig. 4.12 shows the improvements achieved using SSL pretraining for the ripeness classification for the five different fruit. The classification accuracy could be boosted by more than 10% for the avocados and the kiwis. The classification itself is not stable for mangos, kakis, and papayas, but pretraining could reduce the variability for the papayas and overall. Summarizing, the pretraining allows a more reliable ripeness classification for specific exotic fruit.

Chapter 5

Wavelength-aware 2D Convolutions for Hyperspectral Imaging

This chapter will revisit the supervised classification task of hyperspectral recordings (see chapter 3). A wavelength-aware 2D convolution will be proposed and discussed. This convolution is optimized for *hyperspectral imaging (HSI*) and provides useful properties for this data structure. As this is one of the main contributions of our work, we will discuss this method in full detail.

This chapter is based on our publications:

- Varga, L. A., Messmer, M., Benbarka, N., and Zell, A. (2023b). Wavelength-aware 2d convolutions for hyperspectral imaging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3788–3797

5.1 Introduction

Hyperspectral recordings, as described in chapter 2 approximate the spectrum for each pixel of an image. For a higher spectral resolution, the number of channels is increased (to ≈ 200). Further, the range of recorded wavelengths is extended. The additional wavelengths carry information that can be helpful for complex classification tasks. As a result, these systems can perform tasks that aren't possible with pure human perception, allowing superhuman performance in specific applications.

A problem arises from the fact that the recordings created by different manufacturers' hyperspectral cameras are a priori incompatible. There is no standardization regarding the distribution of the channels in the wavelength space. Therefore, two near-infrared cameras from different manufacturers covering the same spectral range have different wavelength assignments for the channels. A model, which identifies the features based on the channel index, will fail on the recordings of another camera. In general, a solution for this problem is standardizing the recording to defined wavelengths. A basic and reliable approach is linear interpolation as described by Steffensen (2013), which can be tedious to fine-tune. Further, the incompatibility of the recordings acquired by different

hyperspectral cameras complicates the generation of large data sets, as only one type of camera can be used. This differs from color image data sets.

In addition, as already stated in chapter 2, a significant drawback of hyperspectral cameras is their complicated acquisition mode. Applying hyperspectral cameras often requires complex data acquisition (e.g., line-scan operation mode) and labeling procedures. This leads to small data sets. The small data sets and the complicated features, often necessary for the tasks, support overfitting. Besides these characteristics, the larger channel dimension of hyperspectral recordings requires special attention.

In this work, we want to tackle the mentioned problems and propose a modified 2D convolution layer optimized for hyperspectral recordings. Reducing the parameters significantly by inferring a proximity bias for the channel dimension, the method can outperform comparable approaches. Further, the model incorporates the channels' wavelength information. This capability allows the training of camera-agnostic models, meaning the models can perform their tasks on recordings of different cameras. As an additional outcome, the models learn the camera filters necessary for the learned task. These camera filters could be used to build a multispectral camera for a specific use-case.

Besides the theoretical background, we prove our claims with empirical experiments on two hyperspectral applications.

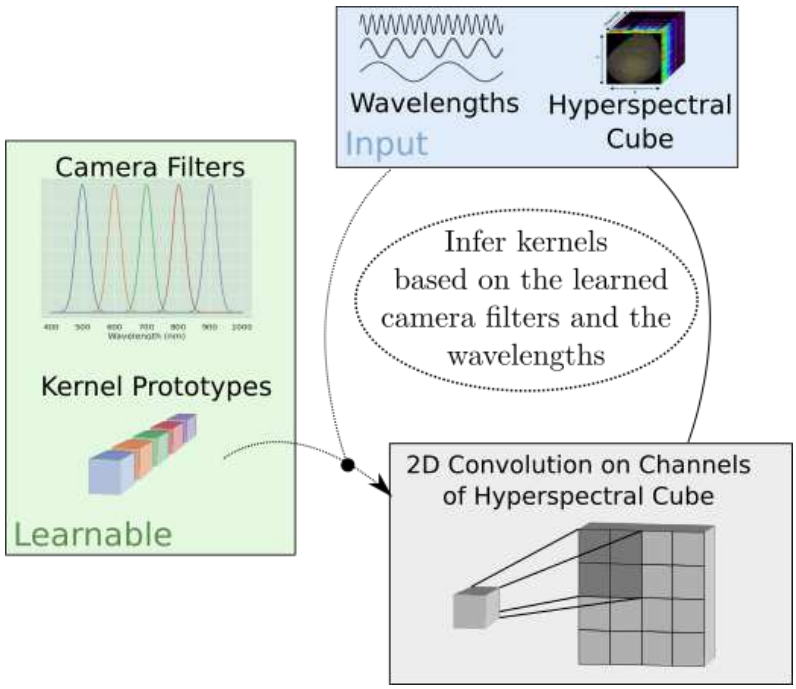


Figure 5.1: Hyperspectral Visual Embedding Convolution (HyveConv) at a glance. Further details in Sec. 5.3.2.

5.2 Related Work

As we return to the supervised HSI classification, most related works mentioned in Sec. 3.1 are also relevant to this chapter. In this chapter, the focus lies on the convolution layer. Therefore, we extend the related work by discussing the most recent developments for HSI in this direction.

Convolutional neural networks for HSI can be divided into methods based on 2D convolutions, 3D convolutions, a mixture of both, imitating 3D convolutions with 2D convolutions, and vision transformers. 2D convolutions perform only spatial convolutions. So the exchange of information between channels is limited and often conducted by the final fully connected layers. Makantasis *et al.* (2015) were the first who utilized 2D convolutions for hyperspectral recordings. 2D convolutions are still very common for hyperspectral recordings, because they have less trainable parameters and can still incorporate spatial information.

In contrast, 3D convolutions can perform convolutions in all three dimensions of the hyperspectral cube. So they can incorporate additional information but are also parameter hungry. Large models are hard to train on the small hyperspectral data sets. Therefore, many approaches try to optimize the model power-size ratio. Smaller models with the same performance are preferred because they tend to overfit less and often produce more stable results over different training runs. Ben Hamida *et al.* (2018) used 3D convolutions to classify hyperspectral remote sensing data. He et al. introduced a multiscale 3D convolutional neural network, which applies 3D convolutions with different kernel sizes. This boosts the performance of the 3D convolutions. Roy *et al.* (2020a) proposed FuSENet, which fuses the output of 3D convolutions by using residual fuse blocks and introducing Squeeze-and-Excitation blocks for HSI, which will be discussed later in full detail.

The third category combines 3D convolutions and 2D convolutions. Roy *et al.* (2020b) proposed HybdriSN. This model has a 3D convolution backbone. The output of this backbone is processed by a 2D convolution and a fully connected head. This was used as the template for the hybrid version of DeepHSNet in chapter 4.

SpectralNET (Chakraborty and Trehan, 2021) belongs to the fourth category. It mimics 3D convolutions with wavelet transformations. It utilizes 2D convolutions for the spatial dimension and the transformed spectral dimension. Finally, the results of both are combined.

Vision transformers, the most recent computer vision trend, also impacted the HSI classification. There are already some adaptations for hyperspectral recordings. Qing *et al.* (2021) proposed SAT Net, which is based on the self-attention mechanism of transformers. Hong *et al.* (2022) introduced a special spectral embedding and a skip-connection, which boosted the performance of the transformers. But for this application, the transformer models often cannot outperform the convolution neural networks as the data sets are tiny. Therefore, smaller convolutional neural networks are usually preferred.

Our method is based on 2D convolutions and, therefore, part of the first group. We

utilize the wavelength metainformation of the input channels to learn a continuous representation of the features in the input channel dimension. Our approach only affects the first convolution layer, so it is compatible with other works mentioned.

Our method utilizes Gaussian distributions to represent the feature distribution in the input channel dimension. Still, our approach is not related to Bayesian convolutional neural networks, like discussed by Shridhar *et al.* (2019). These networks try to approximate the true posterior and incorporate the uncertainty into the inference process, which is not part of our approach.

Hu *et al.* (2020a) proposed a similar approach. Their Squeeze-and-Excitation block allows the network to learn a channel interdependency. Our approach differs in three key points. First, their method uses the input to predict weightings for each feature channel. Our method uses metainformation, the channels' wavelength, to weigh the kernels. Further, our convolution introduced the bias that channels with similar wavelengths should use similar kernels. This proximity relation is helpful for hyperspectral records, shown in section 5.3.1. Last, our method also allows the interpretation of the learned features. The selected wavelengths can be visualized and analyzed, as shown in section 5.5. Both methods share the idea of introducing an interdependency in the channel dimension. In the experiments, we can show that our approach outperforms their approach in the hyperspectral application.

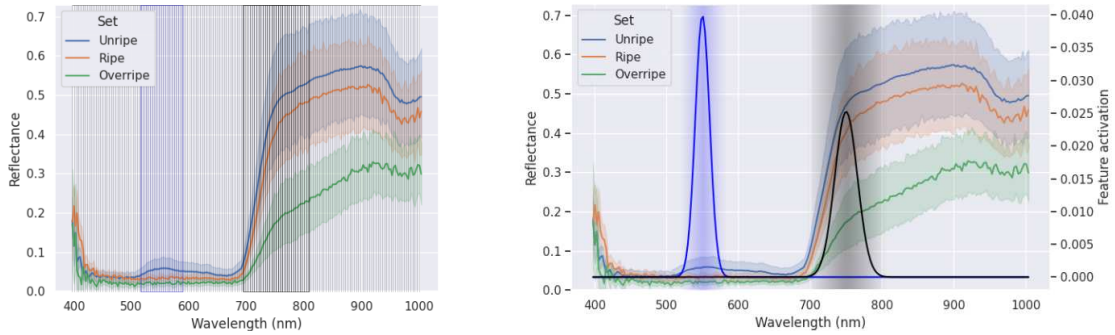
5.3 Proposed Method

Our approach is based on 2D convolutions. Regular 2D convolutions handle input based on the input channel, which is not optimal for hyperspectral recordings. We emphasize key problems and justify our modifications. Further, we propose the method itself. The procedure is evaluated in the section afterward with experiments on real and synthetic hyperspectral data sets with different applications.

5.3.1 Motivation

The reflected light, recorded by a camera, is a spectrum of many wavelengths. An RGB image oversimplifies this spectrum with three sampling bands (red, green, and blue). Hyperspectral recordings cover many more bands and mimic a much better spectrum approximation. Fig. 5.2a shows the spectra of avocados with different ripening states recorded with HSI. The continuity of the underlying spectrum is captured sufficiently.

As a result, we encounter two problems. First, the numerous input data channels (around 200 bands) would demand a large first convolution layer. For the proposed method, the network should have access to the entire hyperspectral cube without a dimension reduction as preprocessing. A dimension reduction could reduce the size of the hyperspectral cube. But by using a dimension reduction, the original data can only be approximated. We argue that if the network can use the full potential of the data, this



(a) Conv2D (b) HyveConv (ours)

Figure 5.2: The channel dimension handling visualized for normal 2D convolutions and our approach. Two example features are presented. Instead of learning each channel separately, the ranges of the features are learned.

usually is beneficial for the selected features, as deep learning approaches can handle high dimensional data well, e.g. shown by Hinton and Salakhutdinov (2006). We prove this in the experiment empirically (in section 5.4.1).

The second problem is that the recorded wavelengths of hyperspectral cameras are not standardized. Recordings of a manufacturer's NIR camera are usually not compatible with recordings of a NIR camera of another manufacturer, even though both cameras share the same wavelength range. Their channel-wavelength assignments are often shifted and have different gradients. An example can be found in Fig. 5.6b and will be discussed in further detail later. As 2D convolutions are based on the index of channels, standardizing the data by a preprocessing step, like linear interpolation (Davis, 1975), is necessary. These preprocessing steps harm end-to-end training. We propose a method capable of handling different hyperspectral cameras by design.

To solve the mentioned problems, our convolution learns a wavelength range of interest (WROI) for each feature instead of the specific input channel. By having a continuous representation of the channel dimension, it can sample the kernels based on the wavelengths of the input channels.

By adding the bias, that the network should apply similar kernels for similar wavelengths, it is possible to significantly reduce the number of parameters. This bias restricts the freedom of the model, but in the context of a continuous spectrum in the channel dimension, this is reasonable and seems to be a key point for handling hyperspectral recordings.

In summary, we propose a method that adds a bias regarding adjacent channels. It eliminates the need for dimensionality reduction for hyperspectral images. And it enables the training of hyperspectral camera-independent models. We provide empirical evidence for these claims in section 5.4. But first, the method itself is described.

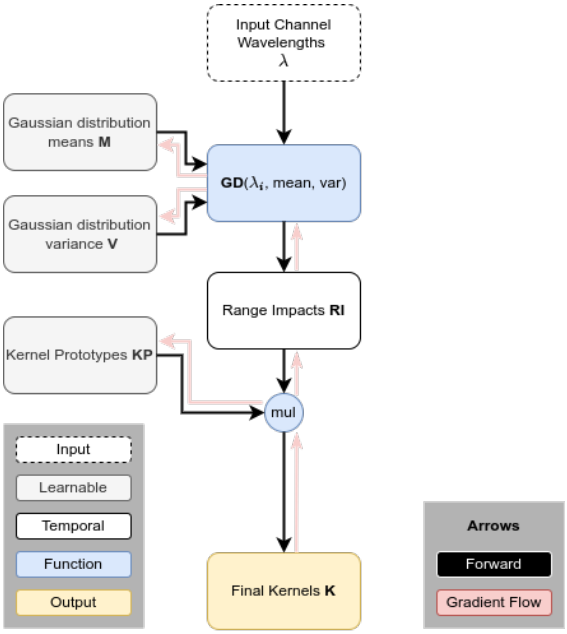


Figure 5.3: Flow of Hyperspectral Visual Embedding Convolution (HyveConv)

5.3.2 Method

The fundamental idea of this approach is to learn kernels and their target wavelength range in combination (Fig. 5.2b) instead of learning each input channel kernel independently (Fig. 5.2a). A learnable Gaussian distribution represents a wavelength range. The weighting factor for the corresponding kernel for this input channel is given by sampling the distribution at the input channel wavelength. The resulting kernel is then calculated by multiplying the factor and the kernel. Finally, the kernel is used for a 2D convolution on the specific input channel. Fig. 5.3 shows the procedure of the method.

In the following, the method is described in further detail. Afterward, an extension is explained, which adds additional synergy effects for the kernels.

A 2D convolution calculates the cross-correlation between trainable kernels and the input data. For C_{in} input channels, C_{out} output channels and kernel size of $K_x \cdot K_y$, this results in a matrix W for the trainable weights:

$$W \in \mathbb{R}^{C_{in} \times C_{out} \times K_x \times K_y} \tag{5.1}$$

The number of trainable parameters of a convolution depends on the number of input channels. For the first layer, the input channels are defined by the channel dimension of the input data. For hyperspectral recordings, this is around 200.

Depthwise-separable convolutions, proposed by Chollet (2017) reduce the number of parameters by splitting up a convolution into a spatial- and a channel-based convolution. The overall relation between input channels and necessary weights still exists. For a

depthwise-separable convolution, the learnable parameters are defined as:

$$W_{ds} = (W_{spatial} \in \mathbb{R}^{C_{in} \times 1 \times K_x \times K_y}, \\ W_{depth} \in \mathbb{R}^{C_{in} \times C_{out} \times 1 \times 1}) \tag{5.2}$$

To tackle the issue of a too large first layer, we learn wavelength ranges of interest (WROIs) for the kernels instead of channel-wise kernels. We assume that adjacent channels of the wavelength space typically share similar features. An example of this behavior can be found in Fig. 5.2a, where adjacent channels have nearly identical reflectance values, originating from the high resolution in the wavelength dimension and the continuity of the signal. Both points can be expected for HSI, so the bias to use similar kernels in neighboring bands seems suitable and even crucial.

Our convolution decouples the number of kernels from the number of input channels. Instead of kernels per input channel, wavelength ranges of interest (WROI) and their kernels are learned. G defines the number of WROIs. Their learnable kernels are called kernel prototypes (KP). This results in the matrix for the kernel prototype weights:

$$KP \in \mathbb{R}^{G \times C_{out} \times K_x \times K_y} \tag{5.3}$$

For the learnable distributions, Gaussian distributions (GDs) are used. A Gaussian distribution (GD) can mimic the wanted behavior, that the impact decays to the borders of a WROI. Further, it is defined by just two parameters, the mean μ and the variance σ^2. Both parameters are differentiable and interpretable. A Gaussian distribution combines a learnable mean μ and a learnable variance σ^2. The value of the Gaussian distribution for a value x is defined as Eq. 5.4.

$$GD(x, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x-\mu}{2\sigma^2}\right) \tag{5.4}$$

To predict the kernels for specific channel wavelengths λ, the first step is to calculate the Gaussian distributions at these wavelengths λ. The result is the range impact matrix RI, which defines the impact of all kernel prototypes on all input channels:

$$RI \in \mathbb{R}^{C_{in} \times G} \\ \text{with } RI_{ij} = GD(\lambda_i, \mu_j, \sigma_j^2) \tag{5.5}$$

Afterward, the learnable kernel prototypes can be weighted with this matrix to produce the final kernels K. These kernels are then used for a 2D convolution on the input.

$$K = RI \cdot KP \in \mathbb{R}^{C_{in} \times C_{out} \times K_x \times K_y} \tag{5.6}$$

The result of this convolution is input to further layers. Our convolution can reduce the trainable parameters to W_{hyve} with $G << C_{in}$. In Tab. 5.1 the number of trainable

parameters of the different models is compared.

$$W_{hyve} = \left(KP, M \in \mathbb{R}^G, V \in \mathbb{R}^G_{>0}\right) \tag{5.7}$$

To support full end-to-end learning, a gradient for Gaussian distributions and kernel prototypes is needed. Fig. 5.3 shows the gradient flow in our convolution model. The multiplication divides the gradient of the final kernel K on the learnable kernel prototypes KP and the range impact matrix RI. The matrix RI holds entries, which were sampled, of the Gaussian distributions regarding the wavelengths of the input. This allows us to infer the impact of the input wavelengths on the gradient. Further, the gradient can be passed to the means M and variances V of the Gaussian distributions. The channel wavelengths λ are part of the input data and do not need a gradient. So, all learnable components of the convolution are trainable based on the gradient of the final kernel K, and end-to-end training of the model is possible.

This also affects the way the model learns the kernels. Fig. 5.2b shows how the training of our convolution varies from the training of a 2D convolution. This simplified example shows just two WROIs (black and blue). A significant feature which is, e.g., visible around 750 nm, will also be visible in some channels around this wavelength. In this example, the feature is visible in the range from 700 nm to 800 nm. So, the 2D convolution (a) has to learn similar features for around 30 independent kernels (assuming input data with 200 channels between 400 nm and 1000 nm). In contrast, our convolution (b) has to learn only one kernel and the distribution of the feature in the wavelength dimension.

The hyperparameter G seems very crucial for this approach. But G is interpretable and explainable. Further, the default configuration seems robust. More information regarding G can be found alation study (see section 5.5).

As the default value for G we recommend 5. Meaning the model can select 5 WROIs in the wavelength range. This leads to $G << C_{in}$.

Initialization of the Gaussian

For our implementation, we initialized the Gaussian distributions of the WROIs in a manner that the whole inspected wavelength range is covered. This is achieved by distributing the means $\mu_{t=0}$ evenly between the minimal (w_{min}) and the maximal inspected wavelengths (w_{max}). Further, the variance $\sigma^2_{t=0}$ is initialized with overlap:

$$\sigma^2_{t=0} = \frac{1}{G}^2 \cdot (w_{max} - w_{min}) \tag{5.8}$$

Further, negative variances are prevented by using softplus proposed by Shridhar *et al.* (2019).

Table 5.1: Parameters of a single convolution for the configuration: $C_{in} = 200$, $C_{out} = 25$, $K_x = 3$, $K_y = 3$, $G = 5$

Conv2D	**Depthwise -separable Conv2D**	**HyveConv (ours)**	**HyveConv++ (ours)**
$200 \cdot 25 \cdot 3 \cdot 3$ $= 45000$	$200 \cdot 1 \cdot 3 \cdot 3$ $+200 \cdot 25 \cdot 1 \cdot 1$ $= 6800$	$5 \cdot 25 \cdot 3 \cdot 3$ $+2 \cdot 5$ $=$ **1135**	$5 \cdot 25 \cdot 3 \cdot 3$ $+2 \cdot 5$ $+1 \cdot 25 \cdot 3 \cdot 3$ $+1 \cdot 1 \cdot 3 \cdot 3$ $+2$ $=$ **1371**

Extension: Additional Kernel Sharing

The learned WROIs allow the model to share kernels through the channel dimension of the input C_{in}. As an extension, we propose sharing parts of kernels through the channel dimension of the output C_{out} and overall kernels of the convolution layer. For this, the previous method is enhanced with additional kernel prototypes. These additional kernel weights are weighted with the learnable factors $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$. The sum of all kernel prototypes is then used further.

$$\begin{aligned} KP_{++}^{i,j,m,n} &= KP^{i,j,m,n} + \alpha \cdot KP_{c\_out}^{1,j,m,n} + \beta \cdot KP_{conv}^{1,1,m,n} \\ KP_{c\_out} &\in \mathbb{R}^{1 \times C_{out} \times K_x \times K_y} \\ KP_{conv} &\in \mathbb{R}^{1 \times 1 \times K_x \times K_y} \end{aligned} \tag{5.9}$$

The kernel prototypes KP_{++} replace the kernel prototypes KP in Eq. 5.6 resulting in Eq. 5.10, which predicts the final kernels.

$$K_{++} = RI \cdot KP_{++} \in \mathbb{R}^{C_{in} \times C_{out} \times K_x \times K_y} \tag{5.10}$$

With this extension, our approach has the following trainable parameters:

$$W_{hyve++} = (KP, M, V, \alpha, KP_{c\_out}, \beta, KP_{conv}) \tag{5.11}$$

This slightly increases the trainable parameters but provides the model with additional synergy effects for kernels within a convolution layer.

Keeping the impact of the shared kernel prototypes at the beginning small is essential. Otherwise, the training is very unstable. Therefore, we recommend an initial value for α_0 and β_0 of 0.1. An evaluation of the impact of the proposed extension can be found in Sec. 5.5.

5.3.3 Mapping-Based Baselines

Our approach can be seen as a convolution, which learns a wavelength-based mapping in addition to the spatial convolutions. One could argue that the proposed method is complicated for its goal. Therefore, we introduce two simpler baselines with similar effects. We extract the learnable mapping into a previous layer for these baselines. As a result, the mapping is learned in a trainable preprocessing step.

The first baseline, called *Input Mapping*, consists of a 1x1 2D-convolution, which is put in front of the first layer and reduces the input channels to a size that is easier to handle.

The second baseline adds the introduced proximity bias to the mapping convolution layer and learns wavelength-based weights. As described for HyveConv (see Eq. 5.6), learnable Gaussian distributions are used to achieve the wavelength-based behavior for this baseline.

For both baselines, a hidden layer size of $G = 5$ was chosen, so the hidden layer size fits the number of WROIs used by our proposed method.

In the experiments discussed in Sec. 5.4, the performance of the models could not keep up with our proposed method HyveConv. We assume HyveCone can easily incorporate spatial information for wavelength mapping and is less prone to input noise.

5.4 Experiments

The proposed method is evaluated on two hyperspectral applications. The first application covers a classification task of ripening fruit recorded under laboratory conditions. The second covers a well-established segmentation task of satellite remote sensing data.

As the first data set contains recordings of the same scene with two different hyperspectral cameras, this data set is used to evaluate the proposed method's general performance and to prove the claim of camera-agnostic property. The second data set, which is well established, is used to validate the method in additional use case and to compare it with the results of other state-of-the-art approaches.

For the following experiments, the extended version of the proposed method (see Eq. 5.10) with the following parameters was used: $G = 5$, $\alpha_0 = 0.1$ and $\beta_0 = 0.1$. Each configuration was tested with three random seeds. The random seed affects the network initialization, the training sample order, and the data augmentation order.

5.4.1 Application A: Fruit Ripeness Prediction

The first application's task is to classify the fruit's ripeness level based on the extended DeepHS Fruit data set. In this set of experiments, the performance of the proposed method is evaluated and compared with similar approaches.

Data Set For the first set of experiments, called the application of fruit ripeness prediction, our data set *DeepHS Fruit v2* (see chapter 4) was used. Here, the ripeness is classified into three categories (firmness, sweetness, and overall ripeness). All setups cover three classes (unripe, perfect, and overripe). This data set has fixed training and test sets.

Most of the recordings were done with a Specim FX 10. This hyperspectral camera covers the wavelength range of 397.66 nm to 1003.81 nm with 224 bands. In addition, there are many recordings of a Corning microHSI 410 Vis-NIR Hyperspectral Sensor. It covers the wavelengths between 408.03 nm and 901.26 nm with 249 bands. The two cameras' ranges are significantly overlapping, which is perfect for the camera-agnostic experiments. All recordings are already normalized with a white and a dark reference.

We use the training and test pipeline proposed in Sec. 3.3.5. The neural networks were trained with the Adam optimizer (Kingma and Ba, 2015) as it provided more stable results. A learning rate of 1x10-2 and a batch size of 64 were used. The learning rate was divided by ten after every 30 epochs.

Models The model DeepHSNet, presented in section 3.3.5, was used as a basis for our approach. It is a shallow convolutional neural network consisting of three depthwise-separable convolutions, a global average pooling layer, and a fully connected head. The complexity of the model is low, which helps in understanding the model's internals. Further, the model could already achieve satisfying results for the prediction of the ripeness level of fruit. It is optimized for the small size of hyperspectral data sets.

For the proposed method, we replaced the first convolution layer with a HyveConv++ layer, keeping all dimensions of the convolution fixed. The rest of the model stays the same. The training schedule was kept the same.

Further, we used a ResNet18 (He *et al.* (2016)), which is still a commonly used backbone, and SpectralNET with factor analysis (Chakraborty and Trehan (2021)), which achieves state-of-the-art performance on the remote sensing data set. Additionally, we used the *Squeeze-and-Excitation (SE)* method of Hu *et al.* (2020a) in two variants. First, in combination with a ResNet18 network, like proposed by Hu et al., and second with the DeepHSNet model. The SE block adds interdependency in the channel dimension of the kernels. The feature map is used to scale each channel of the same feature map using a small, fully-connected network. This allows the network to highlight or ignore specific channels based on the input. Our approach does not connect the kernels directly to the input. Instead, our model can learn the kernels linked to the input wavelengths. In our opinion, this bias tends less to overfitting and is, therefore, more helpful for HSI. Nevertheless, both approaches introduce channel interdependence, and the SE block improved HSI classification (see second application), so this approach should be considered as a baseline.

Finally, we evaluated the performance of the baselines *Input Mapping* and *Input Mapping with Bias*, which were described in Sec. 5.3.3.

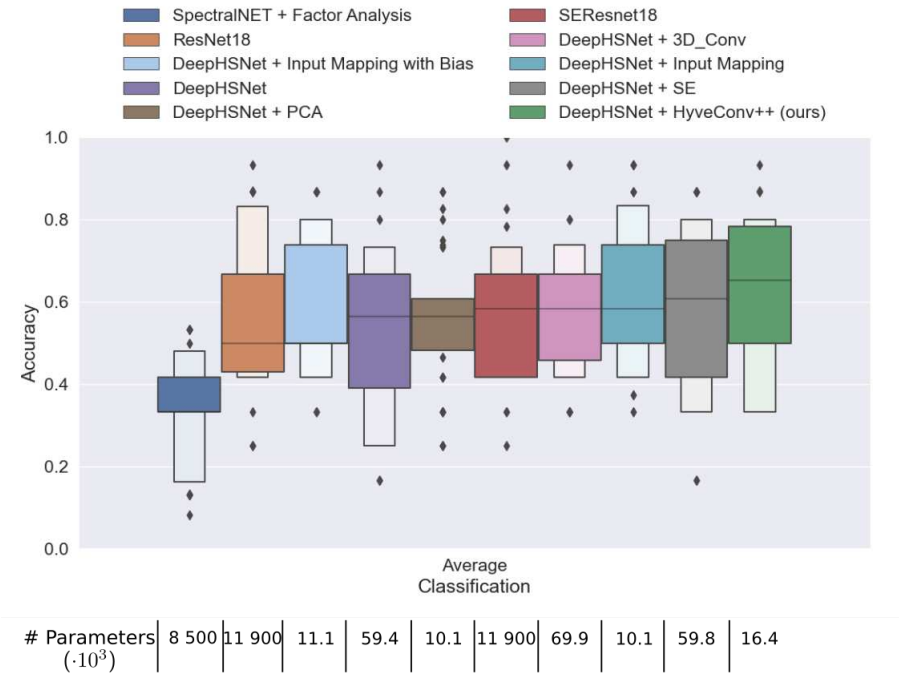
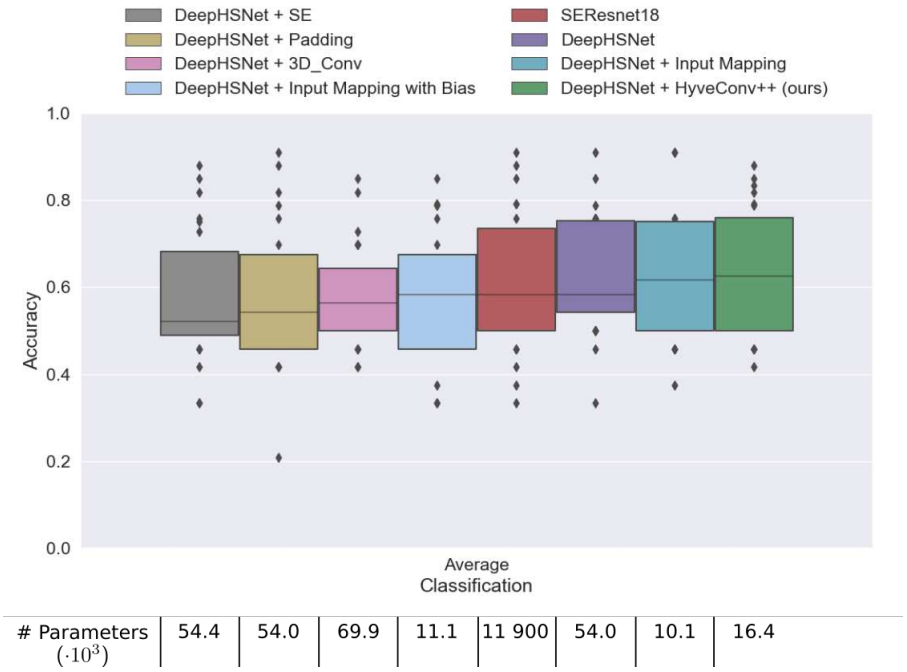


Figure 5.4: Overall accuracy on the ripening fruit data set with the Specim FX 10 recordings.

One Camera In the first experiments, the recordings of the Specim FX 10 were used only. These experiments indicate the performance of the tested models in the default single-camera use case. The results are visible in Fig. 5.4. Our model could outperform the other models' average accuracy, given by the median over all categories and fruit types. A DeepHSNet with Squeeze-and-Excitation blocks (DeepHSNet + SE) performed second best. Both approaches introduce an interdependency between the channels, which seems helpful. Our model bias that similar wavelengths should have similar features further boosts performance. As a result, our model produced better results with fewer parameters. The baselines *Input Mapping* and *Input Mapping with Bias* showed no significant improvement compared to the origin DeepHSNet. As discussed in Sec. 5.3.3, we assume HyveConv can handle noise channels better by adding the spatial information more directly. Still, these baselines show that not only the reduction of the parameters was the reason for the performance boost. Larger models, like ResNet18 or SpectralNET could not benefit from the larger number of parameters. We assume that the used early-stopping and data augmentations could not prevent overfitting entirely.

Multiple Cameras In the second set of experiments, recordings of two different cameras were used (Specim FX 10 and Corning microHSI 410 Vis-NIR). These cameras differ in the recorded wavelengths. In these experiments, the synergy capability of the



# Parameters ($\cdot 10^3$)	54.4	54.0	69.9	11.1	11 900	54.0	10.1	16.4

Figure 5.5: Overall accuracy on the ripening fruit data set with recordings of both cameras.

models is evaluated, meaning how well the models can incorporate the recordings of both cameras. The number of recordings of each camera is not balanced ($\approx 2:1$), which adds another challenge. The training, validation, and test set contained recordings of both cameras in these experiments.

We compared our method with the best models of the previous experiments. Most models cannot handle the inhomogeneous data out of the box, as the number of channels and the channel-wavelength-assignment differs. Therefore, a linear-interpolation step was introduced for all models except the HyveConv++ approach, the *Input Mapping + Bias*, and the Padding approach, which simply adds padding for the smaller recordings. The three mentioned methods can handle the recordings of both cameras by design.

The results are visible in Fig. 5.5. Three plateaus are visible. The *Input Mapping* approach and our proposed HyveConv++ performed best. HyveConv++ performed slightly better.

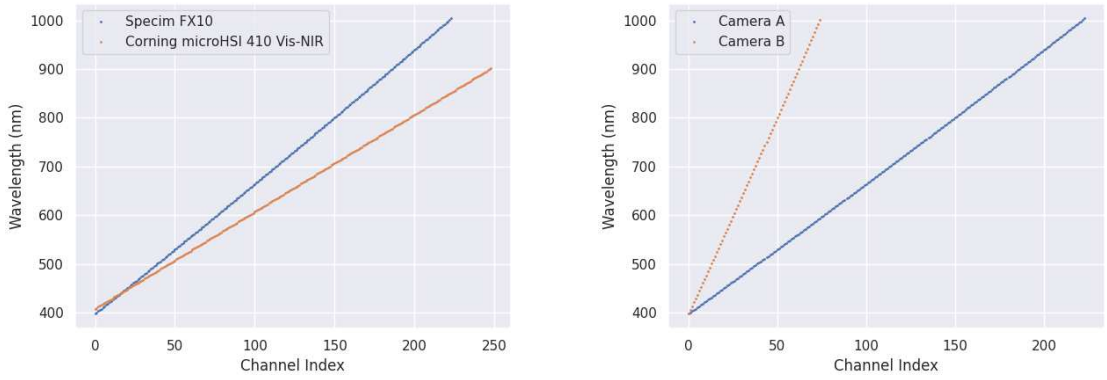
On the second plateau, four models can be found. The models are the baseline DeepHSNet, the ResNet18 with Squeeze-and-Excitation, and the *Input Mapping with Bias* approach.

The third plateau contains approaches that degraded the performance compared to the baseline DeepHSNet. Here, we can find a 3D convolutions model, a Squeeze-and-Excitation DeepHSNet model, and the padding approach.

To summarize, linear interpolation works for this application better than a padding-

based approach. Further, our method could still improve performance slightly, especially if we keep in mind that the configuration of linear interpolation is crucial. An inappropriate definition of the quantization steps can harm the expressiveness of the recordings.

HyveConv++ can boost the camera-agnostic behavior for HSI by using the wavelength information for the convolution. An additional experiment, which supports this claim and is based on synthetic data, will be discussed next.



(a) (b)

Figure 5.6: Assigment of wavelengths to the channel indices for **(a)** a real setup (Specim FX 10 and Corning microHSI 410 Vis-NIR) and a **(b)** synthetic setup.

Synthetic Multiple Cameras There is a lack of hyperspectral data sets with multiple camera recordings of the same scene. Thus, we generated a synthetic data set to support the claim of camera-agnostic behavior. For this, a data set was created based on the Specim FX 10 recordings of the ripening fruit.

Two cameras were simulated using two channels' subsets (shown in Fig. 5.6b). We selected the channels of the subsets based on different step sizes for the channel indices. This mimics the real setup's behavior, as shown in Fig. 5.6a. The wavelengths of the latter channels differ more in contrast to the actual setup. Further, based on the sampling method of the channels of the two subsets, camera B contains fewer channels than Camera A. And the spatial resolution of both cameras is identical, which is uncommon for two different cameras. In addition, the number of recordings per camera is perfectly balanced, which is also very unusual. Therefore, this experiment setup has slightly different requirements than a configuration with two real cameras, but still, it can at least indicate whether the models can learn from both synthetic cameras.

Again, this data set provides inhomogeneous recordings as the channel dimension size, and the wavelength assignment differs for the two synthetic cameras. Thus, linear interpolation was introduced for most of the models as an additional step.

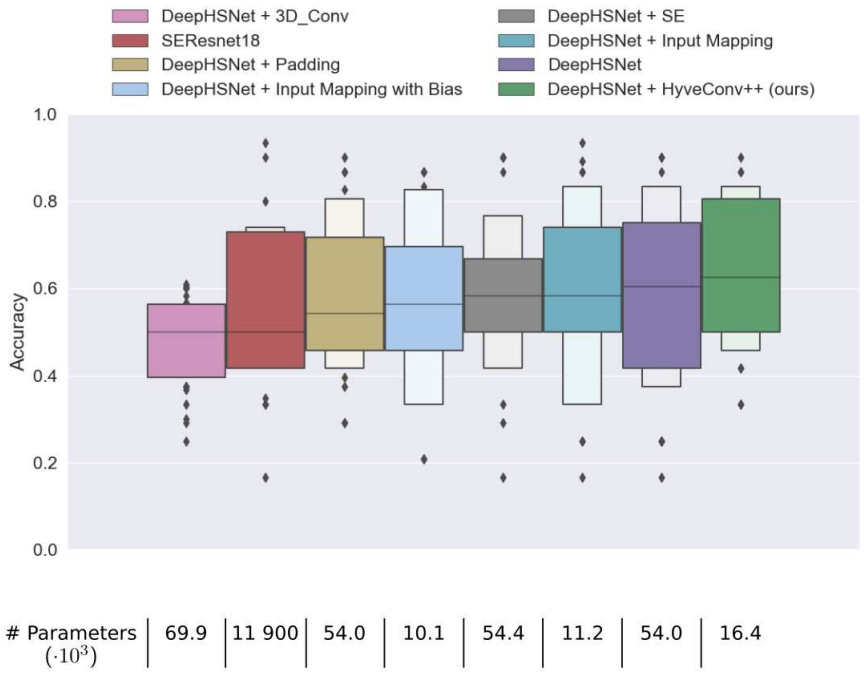


Figure 5.7: Overall accuracy on the ripening fruit data set with synthetic recordings, generated for two cameras.

Fig. 5.7 shows the outcome of the experiments on synthetic data. The result is different from the previous experiments. This time only our method could outperform the baseline DeepHSNet. The *Input Mapping* approach, which performed very well in the last experiment, was significantly worse than DeepHSNet. The Padding approach was still one of the worst models. This confirms the camera-agnostic property of our model. It performed most stable in these two setups. In the next step, the situation with an unknown camera is inspected.

With unknown camera The camera-agnostic behavior for unknown cameras should be evaluated in the fourth set of experiments. Therefore, the test set contained recordings of both cameras, but the training set and validation set contained only recordings of the Specim FX 10 at the beginning.

This experiment should check the model generalizability for another camera. Therefore, we evaluated how the model's performance depends on the number of recordings of the second camera. The results are visible in Fig. 5.8. A ratio of 0.0 means none of the recordings of the second camera were added to the training and validation set. In contrast, a ratio of 1.0 means all of the available recordings were added. This number does not indicate the ratio between the recordings of the two cameras.

For ratio of 0.0, the models saw only recordings of a single camera during training

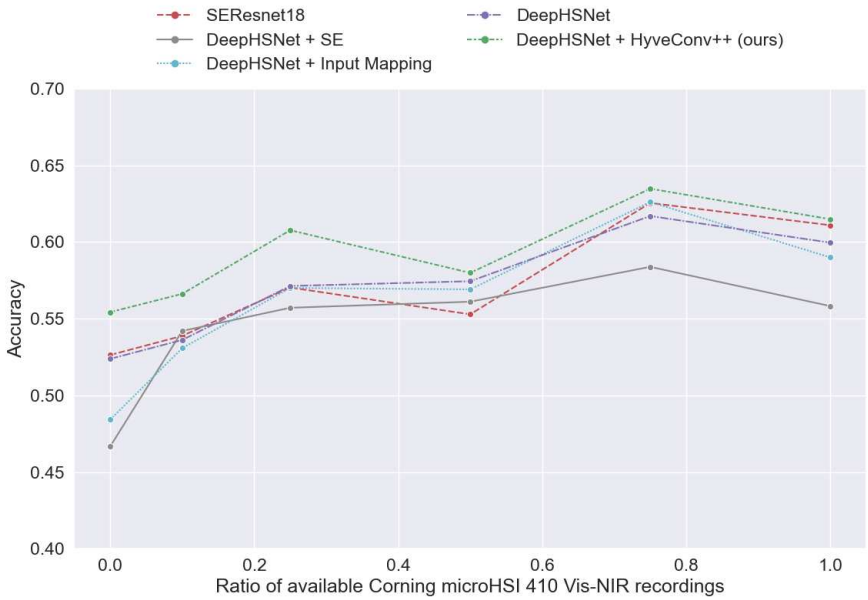


Figure 5.8: Overall accuracy on the ripening fruit data set evaluated on recordings of both cameras. The training set was restricted to the recordings of the Specim FX 10 with additional recordings of the Corning microHSI 410 Vis-NIR.

without the information that there is a second camera. This is an arduous setup. We recommend that the training set at least cover two cameras to enforce better generalization.

We evaluated the best models of the previous experiments on the task. For all models except our HyveConv++ approach, linear interpolation was necessary.

For all models, we can see a general trend, that the performance improves with more recordings of the second camera. For most of the models, we have a dip for 0.5 of the available recordings of the second camera. We do not have a founded explanation for this behavior. A this point, a quarter of the training and validation set belongs to the second camera (as the data set contains half as many recordings of the second camera). At this point, we would mark it as an outlier, even if it is consistent over many models.

Fig. 5.8 indicates that our proposed method could perform best independent of the amount of samples of the second camera. Even without any recordings of the second camera in the training and validation set, it could still perform significantly better than the compared models.

5.4.2 Application B: Satellite Imagery Segmentation

The experiments on the second data set should validate the proposed method's performance compared to other HSI state-of-the-art approaches.

Table 5.2: Classification accuracies (%) of different models in terms of *Overall Accuracy (OA*), *Cohen Kappa (Kappa*), and *Averaged Classwise Accuracy (AA*) with 10% annotated data as training data. Based on the evaluations of Chakraborty and Trehan (2021). Two configurations of our model are presented. (*) not fine-tuned (**) larger hidden layers. **Bold** numbers indicate the best accuracy for each configuration. The full results can be found in Appendix A.3.

Methods	**# of params**	**Indian Pines**			**Pavia University**			**Salinas**		
		OA	Kappa	AA	OA	Kappa	AA	OA	Kappa	AA
SVM Cortes and Vapnik (1995)		81.67	78.76	79.84	90.58	87.21	92.99	94.46	93.13	93.01
2D-CNN Makantasis *et al.* (2015)	561,300	80.27	78.26	68.32	96.63	95.53	94.84	96.34	95.93	94.36
3D-CNN Ben Hamida *et al.* (2018)	991,596	82.62	79.25	76.51	96.34	94.90	97.03	85.00	83.20	89.63
M3D-CNN He *et al.* (2017)	372,544	81.39	81.20	75.22	95.95	93.40	97.52	94.20	93.61	96.66
FuSENet Roy *et al.* (2020a)	100,880	97.11	97.25	97.32	97.65	97.69	97.68	99.23	99.97	99.16
HybridSN Roy *et al.* (2020b)	5,122,176	98.39	98.16	98.01	**99.72**	**99.64**	99.20	**99.98**	**99.98**	**99.98**
SpectralNET Chakraborty and Trehan (2021)	6,800,336	**98.76**	98.59	98.61	99.71	99.62	99.43	99.96	99.96	99.97
HyveConv++ ours (*)	**16,700**	98.18	98.41	98.28	99.30	99.30	**99.49**	99.24	99.64	99.94
HyveConv++ ours (**)	**25,200**	98.33	**98.64**	**98.69**	99.42	99.49	99.46	99.89	99.79	99.74

Data Set The *Hyperspectral Remote Sensing Scenes (HRSS*) data set is a collection of hyperspectral satellite images collected by M. Graña, M.A. Veganzons, and B. Ayerdi. The task is to classify the nature of the ground in different settings. Each scene consists of an image with ground truth labels. The most common scenes are *Indian Pines (IP*), *Pavia University (UP*), and *Salinas (SA*). Each scene is handled separately. We followed the data handling procedure of Chakraborty and Trehan (2021). The segmentation task is converted into a classification task of 64x64 patches.

Indian Pines was recorded with the AVIRIS sensor, which covers the range of 400 nm to 2500 nm with 224 channels as written by Baumgardner *et al.* (2015). Twenty-four noisy channels were already removed from these recordings. The classification happens within 16 classes. Salinas was also recorded by this sensor and covers six classes. Pavia University was recorded with the ROSIS sensor, covering the range from 430 nm to 830 nm with 103 bands and distinguishing nine classes.

Models We used two configurations of our model. Both used the whole hyperspectral cube without dimension reduction. The first configuration (*) is the same model used for the ripening classification task. Only the final output layer was adapted to the number of classes. The second configuration (**) was slightly adapted for this application. The main change here was to increase the number of channels in the hidden layers, reasoned by the higher number of classes for this application.

Table 5.3: Classification accuracies (%) with 30% annotated data as training data. Based on the evaluations of Chakraborty and Trehan (2021). Two configuration of our model are presented. (*) not fine-tuned (**) larger hidden layers. **Bold** indicates the best accuracy per configuration. The full results can be found in Appendix A.3.

Methods	**# of params**	**Indian Pines**			**Pavia University**			**Salinas**		
		OA	Kappa	AA	OA	Kappa	AA	OA	Kappa	AA
SVM Cortes and Vapnik (1995)		87.24	85.27	85.15	95.65	94.63	94.60	94.95	94.48	97.93
2D-CNN Makantasis *et al.* (2015)	561,300	88.90	87.01	85.70	96.50	96.55	96.00	96.75	96.71	98.57
3D-CNN Ben Hamida *et al.* (2018)	991,596	90.23	89.70	89.87	97.90	97.22	97.30	95.54	94.81	97.09
M3D-CNN He *et al.* (2017)	372,544	95.67	94.70	94.60	97.60	96.50	98.00	94.99	95.40	96.28
FuSENet Roy *et al.* (2020a)	100,880	99.01	98.60	98.64	99.42	99.21	99.33	99.68	99.74	99.69
ImprovedTransformerNet Qing *et al.* (2021)	150,000,000	99.22	99.19	99.08	99.64	99.49	99.67	99.91	99.78	99.63
HybridSN Roy *et al.* (2020b)	5,122,176	99.75	99.71	99.63	99.98	**99.98**	99.97	**100**	**100**	**100**
SpectralNET Chakraborty and Trehan (2021)	6,800,336	**99.86**	**99.84**	**99.98**	**99.99**	**99.98**	**99.98**	**100**	**100**	**100**
HyveConv++ ours (*)	**16,700**	99.85	99.75	99.7	99.97	99.96	99.97	99.98	99.99	99.99
HyveConv++ ours (**)	**25,200**	**99.86**	**99.84**	99.57	99.96	**99.98**	99.94	99.93	99.92	99.99

Further, we tested a variety of different models. The models cover classical machine learning (Cortes and Vapnik (1995)), 2D convolutions (Makantasis *et al.* (2015)), 3D convolutions (Ben Hamida *et al.* (2018)), mixture of 2D and 3D convolutions (He *et al.* (2017); Roy *et al.* (2020a)), 2D convolutions with optimized preprocessing (Roy *et al.* (2020b); Chakraborty and Trehan (2021)) and one vision transformer based approach (Qing *et al.* (2021)). Especially, HybridSN, proposed by Roy *et al.* (2020b), and SpectralNET, presented by Chakraborty and Trehan (2021), achieve state-of-the-art results for this application. These significantly different approaches cover the development of HSI classification.

Results Tab. 5.2 and Tab. 5.3 provide an overview of the performance of the models on the remote sensing application. The two tables use 10% and 30% of the annotated data as training and validation data. The rest is used as test set.

First of all, it is noticeable that larger models do not always outperform smaller models here. FuSENet performs much better than 3D-CNN and M3-CNN even though these approaches are significantly larger. FuSENet is based on the already discussed Squeeze-and-Excitation (SE) approach, which was highlighted for the first application. FuSENet used this approach with a PCA to preprocess the HSI remote sensing classification.

HybridSN combines 3D and 2D convolutions. SpectralNET utilizes wavelet transformations for the spectral features. Both perform reliably on all scenes of HRSS.

A vision transformer model was included in the experiments to cover the most recent trend of computer vision. In Tab. 5.3, ImprovedTransformerNet represents the vision transformers. The size of this model is much larger than the other evaluated models. For the small hyperspectral data sets, this is not optimal. Therefore, the model was only evaluated on the larger training set of 30% annotated samples. The transformer achieved good results but not perfect results. Further, the training procedure was unstable. This fits our assumption that a well-chosen bias is important for small hyperspectral data sets.

Finally, our model was evaluated with two configurations. The model without modifications (*) could already achieve second/third rank in the overall ranking (see Tab. 5.2 and Tab. 5.3). With minor modifications (**), it achieved state-of-art performance with 250 times fewer parameters. These experiments showed that our proposed method generalizes well to different hyperspectral applications. Further, it seems necessary for HSI to use convolutions optimized for the hyperspectral cube instead just using 3D convolutions or vision transformers.

5.5 Ablation Study

In the previous section, our model outperformed comparable models in two applications. In this section, the interpretability of the learned WROIs, also called camera filters, is presented. In addition, an analysis of the influence of the hyperparameter G, defining the number of WROIs and the impact of the proposed method extension (Eq. 5.11) is analyzed. Finally, it is checked whether the training of the Gaussian distribution is necessary.

5.5.1 Interpretation of the WROIs

Fig. 5.9 presents the training and the resulting WROIs for an example run for the ripeness classification of avocados. 5.9a and 5.9b show the value over epochs for the mean and the variance, respectively. The variance is a good indicator for the WROI search procedure. A decreasing variance indicates that the model has found a feature and narrows the corresponding wavelength range. The final WROIs are visible in Fig. 5.9c and the visualizations of these are shown in Fig. 5.10. Gaussian distributions 2 and 3 have a significant overlap of over 50% and a position swap in epoch 3. Therefore, they may cover the same feature, and a reduction of the WROI number G seems possible.

The final camera filters cover the visible light in the ranges of blue and green. Further, overtones of water were selected by the model. Wellburn (1994) showed these ranges could indicate the degeneration of chlorophyll. These ranges also fit the findings of the works of Pinto *et al.* (2019) and Melado-Herreros *et al.* (2021). The WROI selection of the trained model could be used to build a multispectral camera optimized for this use case. As we already discussed earlier, a multispectral camera with 5-10 custom-defined wavelengths usually is easier to apply in an inline scenario.

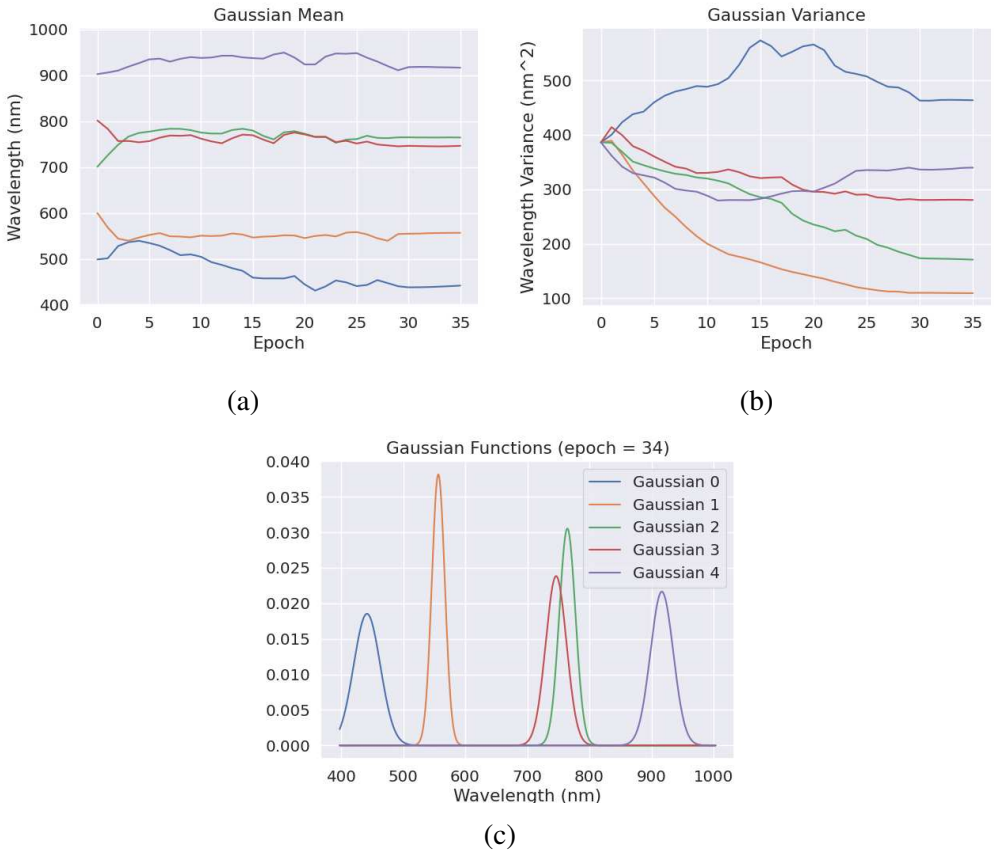


Figure 5.9: Training of the Gaussian distributions. **(a)** and **(b)** show the development of the mean and variance over training epochs. **(c)** shows the final Gaussian distributions.

We showed that the learned features are explainable and can be used further. Further examples can be found in the appendix (see Sec. A.4).

5.5.2 Impact of Parameter G

Fig. 5.11 visualizes the impact of the parameter G, which defines the number of possible WROIs, on the performance of our model. This was tested on the ripening fruit data set and averaged over all setups (fruit type and classification type) with the recordings of the Specim FX 10.

A clear trend is visible. Increasing the number of Gaussian distributions increases the performance of the model. Two giant steps are visible: a Plateau between $G = 2$ and $G = 4$ and for $G \geq 5$. The increase is continuous for $G > 5$, but only slightly. Each jump in performance shows that the additional WROI provides helpful new information. Overall, the hyperparameter G seems stable above the threshold of 5. So, for a ripeness prediction for the evaluated fruit types, $G = 5$ is recommended. For avocados, the classification

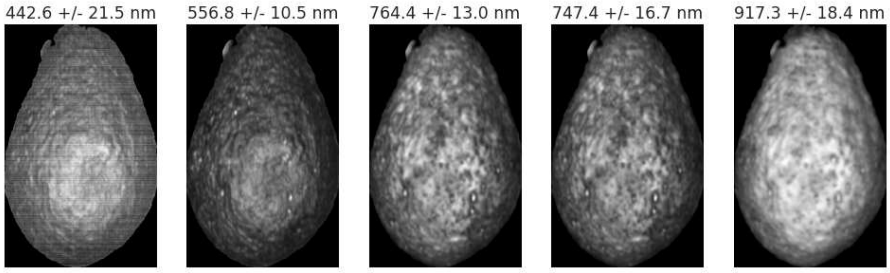
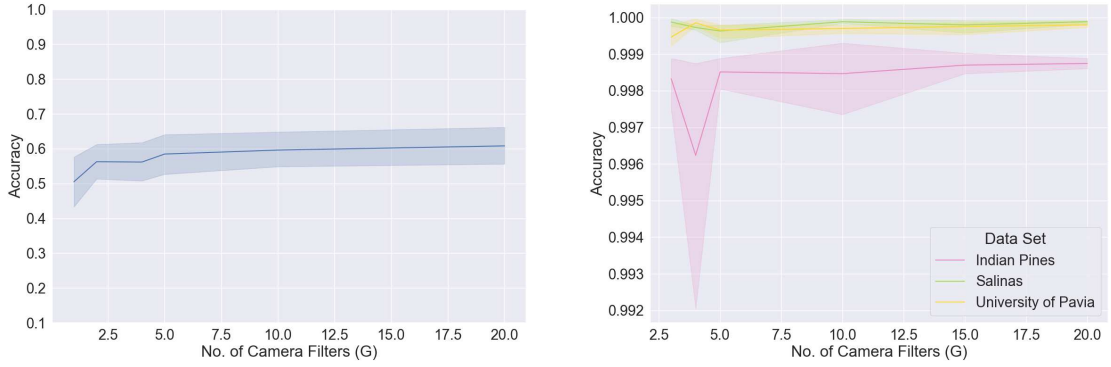


Figure 5.10: Training of the Gaussian distributions: Visualization of the learned camera filters of the training of Fig. 5.9.



(a) (b)

Figure 5.11: Impact of number of WROIs on DeepHS Fruit v2 **(a)** and the HRSS **(b)**.

seems more straightforward, and therefore already, $G = 4$ could be sufficient, as shown in the previous paragraph.

There is a similar trend for the second application. $G = 5$ seems to work reliably. As a result, $G = 5$ seems a good first choice, which the first training results can optimize. This shows how easily a Parteo optimization can be performed for the optimal number of camera filters.

5.5.3 Training of the Gaussian Distributions

We evaluated whether training of the Gaussian distributions is necessary. As a comparison, we used the initial distribution previously mentioned, which should cover the whole inspected wavelength range evenly distributedly. The result is visible in Fig. 5.12. There is a clear improvement in performance and stability. Thus, trainable Gaussian distributions seem helpful.

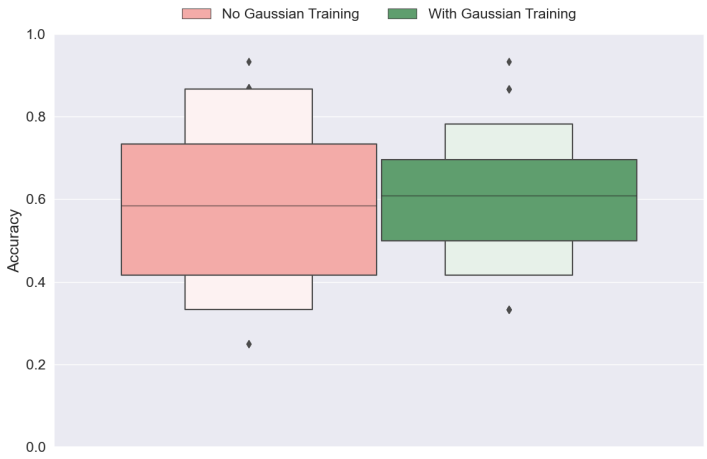


Figure 5.12: Trainable and non-trainable Gaussian distributions. (DeepHS Fruit v2, Specim FX 10)

5.5.4 Impact of the Method Extension

We checked whether the proposed extension of the basic method is necessary. We compared the baseline model (DeepHSNet) with the basic method (HyveConv) and the extended method (HyveConv++). Further, we evaluated only the extension. The Specim FX 10 recordings of fruit ripening data set were used.

Fig. 5.13 shows the results. The continuous definition of the input channel space and the additional bias of HyveConv boosts the accuracy by around 2%. Allowing the convolution to share features through different output channels and the whole convolution layers (HyveConv++) increases the model's performance by 4%. Therefore, the extension seems reasonable. On the other side, the extension on its own without HyveConv decreases the performance by 6%. Thus the extension is only in combination as HyveConv++ helpful.

5.6 Conclusion

In this chapter, we proposed a 2D convolution optimized for HSI. By using a continuous representation of the input space and adding a suitable model bias, it is possible to reduce the number of parameters significantly. Further, sampling the kernels by the input wavelengths allows the training of a camera-agnostic model. The whole model is end-to-end trainable with one interpretable hyperparameter G. This parameter defines the number of wavelength ranges of interest, called camera filters. Experiments on two different hyperspectral applications confirmed the advantage of this method.

The convolution is proposed for HSI. Still, it could also be helpful in other scenarios of image data with many channels and some proximity relationship between them.

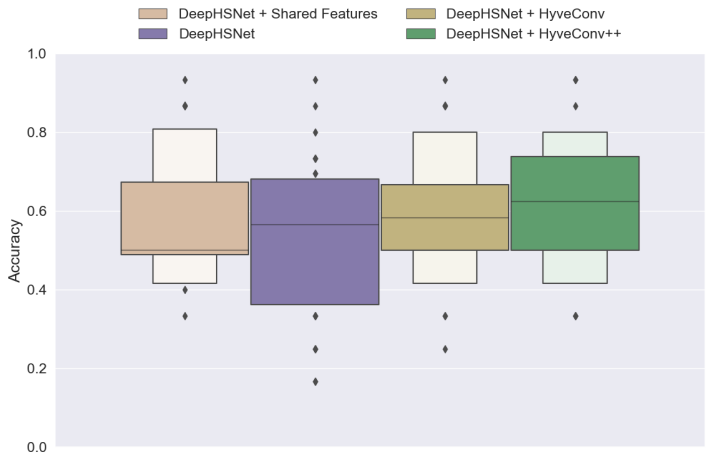


Figure 5.13: HyveConv with and without extension. (DeepHS Fruit v2, Specim FX 10)

Multispectral and color cameras are typically based on wavelength filters. The WROIs, learned by our method, are these kinds of filters. Therefore, the model learns the filters for a specific task in a data-driven way. Based on the learned filters an optimized multispectral camera for a particular task could be built.

The next chapter will discuss the embedded use case of multispectral cameras onboard drones for search and rescue missions in maritime environments. In this application, hyperspectral cameras would be hard to handle. Further, this application is very different from the laboratory setting, which was primarily presented until now. This setting shows how spectral imaging can be utilized in a real-world scenario.

Chapter 6

UAV-Based Applications of Multispectral Imaging

This chapter shows how to use multispectral cameras for embedded applications (like onboard drones). The discussion is based on our published works:

- Varga, L. A., Kiefer, B., Messmer, M., and Zell, A. (2022b). SeaDronesSee: A maritime benchmark for detecting humans in open water. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pages 3686–3696. IEEE
- Varga, L. A., Koch, S., and Zell, A. (2022a). Comprehensive analysis of the object detection pipeline on UAVs. *Remote Sensing*, **14**(21)
- Varga, L. A. and Zell, A. (2021). Tackling the background bias in sparse object detection via cropped windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 2768–2777

6.1 Introduction

In this chapter, the embedded application of multispectral cameras is discussed. As mentioned earlier, multispectral cameras are easier to handle in real-world applications but provide a smaller spectral resolution (see chapter 1). We explain how multispectral cameras can be utilized via the example of maritime *search and rescue (SAR)* missions. Besides the classification, the localization of the objects in the recording is essential for SAR missions. Accordingly, object detection is the discussed computer vision task in this chapter.

For this reason, our acquisition of the object detection data set *SeaDronesSee* is presented. During the data acquisition, the main goal was to capture color images, but multispectral recordings were also acquired. In this thesis, we will focus on the multispectral recordings. Thus, we refer an interested reader for more information about the whole data set to the related publication (see (Varga *et al.*, 2022b)).

Further, we analyzed the impact of different camera parameters (quantization, compression, resolution, image distortions, gamma error, color model, and multispectral cameras) on the performance of object detectors. In this thesis, we will only focus on the last two, the impact of the color model and the usefulness of multispectral recordings. For a full evaluation, we also refer to the related publication (see (Varga *et al.*, 2022a)).

The last part of this chapter shows that the background bias in remote sensing recordings is a severe problem for object detectors. Further, an easy-to-apply cropping-based technique is presented. Hence the method is mainly evaluated on color images instead of multispectral recordings. This part is kept short. The interested reader is again referred to the related publication (see (Varga and Zell, 2021)).

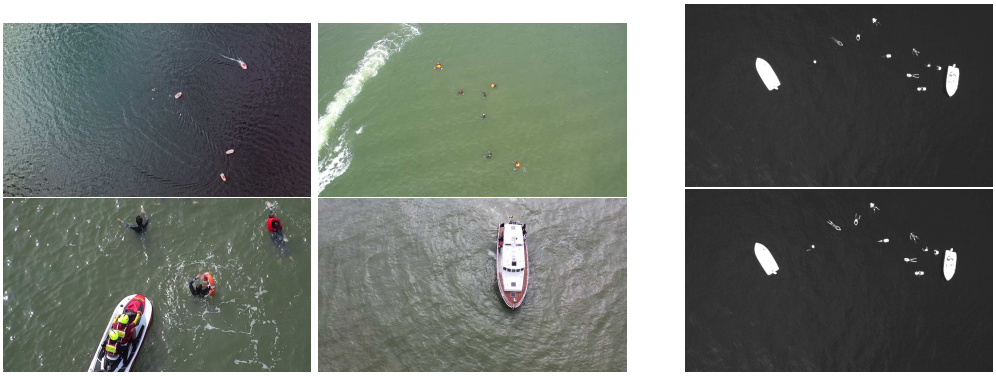
6.2 Generation of a Maritime Benchmark for Detecting Humans in Water

Unmanned aerial vehicles (UAV) equipped with cameras have grown into an essential asset in a wide range of fields, such as agriculture (Adão *et al.*, 2017), delivery (San *et al.*, 2018), surveillance, and *search and rescue (SAR)* missions (Geraldes *et al.*, 2019). Due to their speed and flexibility, UAVs can assist in SAR missions by providing an overview of the scene (Mishra *et al.*, 2020; Karaca *et al.*, 2018; Albanese *et al.*, 2020).

Especially in maritime scenarios, where wide areas need to be quickly overseen and searched, the efficient use of autonomous UAVs is crucial, as shown by the review of Yeong *et al.* (2015). Detection, localization, and tracking of people in open water are thereby key components (Gallego *et al.*, 2019; Nasr *et al.*, 2019).

Currently, the vision systems used for this task are implemented via data-driven methods such as deep neural networks. These methods depend on large-scale data sets. However, there is a great lack of large-scale data sets in maritime environments. Most data sets captured from UAVs are land-based, often focusing on traffic environments, such as VisDrone (Zhu *et al.*, 2018) and UAVDT (Du *et al.*, 2018). Many of the few data sets captured in maritime environments fall in the remote sensing category, often leveraging satellite-based synthetic aperture radar (Crisp, 2004). These are only valuable for ship detection (Corbane *et al.*, 2010) as they don't provide the resolution needed for SAR missions. Furthermore, satellite-based imagery is susceptible to clouds and only provides top-down views. Finally, many current approaches in the maritime setting rely on classical machine learning methods, incapable of dealing with the large number of influencing variables and calling for more elaborate models (Prasad *et al.*, 2019).

This work aims to close the gap between large-scale land-based data sets captured from UAVs to maritime-based data sets. We introduce a large-scale data set of people in open water called SeaDronesSee. Using multispectral cameras with near infrared channels to detect humans in maritime settings can be advantageous (Gallego *et al.*, 2019). For that reason, we also captured, besides color images, multispectral images using a MicaSense



(a) (b)

Figure 6.1: **(a)** Typical image examples with varying altitudes and angles of view. **(b)** Examples of the red-edge (717 nm, top) and near-infrared (842 nm, bottom) light spectra of an image captured by the MicaSense RedEdge-MX.

RedEdge-MX. This enables the development of detectors taking into account the non-visible light spectra near-infrared (842 nm) and red-edge (717 nm).

We focused on object detection and task tracking with color images in the related publication. We provided baselines with state-of-the-art approaches for the application tracks and analyzed the statistical properties of the recorded data. But we neglected a further analysis of the multispectral recording. In this thesis, we will only briefly discuss the color image recordings and focus on analyzing the multispectral recordings.

6.2.1 Related Work

This section reviews major labeled data sets in computer vision from UAVs and maritime scenarios usable for supervised learning models.

Over the last few years, several data sets captured from UAVs have been published. The most prominent are those that depict traffic situations, such as VisDrone (Zhu *et al.*, 2018) and UAVDT (Du *et al.*, 2018). Both data sets focus on object detection and object tracking in unconstrained environments. Pei *et al.* (2019) collected videos showing campus traffic participants for human trajectory prediction usable for object detection. UAV123 (Mueller *et al.*, 2016) is a single-object tracking data set consisting of 123 video sequences with corresponding labels. The clips mainly show traffic scenarios and everyday objects. Both Hsieh *et al.* (2017) and Mundhenk *et al.* (2016) captured a data set showing parking lots for car counting tasks and constrained object detection. Li and Yeung (2017) provided a single-object tracking data set showing traffic, wildlife and sports scenarios.

Another active area of research focuses on drone-based wildlife detection. van Gemert *et al.* (2014) release a data set for low-altitude detection and cattle counting tasks. Ofli

Table 6.1: Overview of used cameras. The multispectral camera is highlighted.

Camera	Resolution	Frames per Second
Hasselblad L1D-20c	3,840×2,160	30
Sony UMC-R10C	5,456×3,632	1
Zenmuse X5	3,840×2,160	30
Zenmuse XT2	3,840×2,160	30
MicaSense RedEdge-MX	1,280× 960	1

et al. (2016) release the African Savanna data set as part of their crowd-sourced disaster response project.

Li *et al.* (2018) provided a maritime-related data set of ships with images mainly taken from Google Earth and a few UAV-based images. In (Xia *et al.*, 2018), the authors provide satellite-based images from natural scenes, primarily land-based but also harbors.

The work of Lygouras *et al.* (2019) is the most similar to our work. They also consider the problem of human detection in open water. However, their data mainly contains images close to shores and swimming pools. Furthermore, it is not publicly available.

Besides the publications proposing UAV data sets, many works tackle subtasks of maritime SAR with drones. Lvsouras and Gasteratos (2020) combine detection and tracking to achieve better results. Gallego *et al.* (2019) focused on the detection part but did the first experiments with a multispectral camera in maritime environments. Lygouras *et al.* (2019) utilized a *Global Navigation Satellite System (GNSS*) to combine the detection algorithms with real time coordinates. Queralta *et al.* (2020) built a multi-UAV system for assisting SAR missions in maritime areas. Roberts *et al.* (2016) focused on mission planning and proposed a framework for efficient coordination of multi-UAV systems in maritime SAR missions. Ghazali *et al.* (2016) proposed an approach to find the exact position of a subject with different search patterns. These works show there is an upcoming trend for UAV-based maritime SAR approaches.

6.2.2 Data Set Generation

The footage for the first data set version was gathered over several days on Lake Constance. The acquisitions were performed on different days to increase the variance in the data (like weather, wave structures, or light conditions). Over 20 test subjects were transported by boats to the area of interest. At the target, the quadcopters were launched. At the same time, the fixed-wing UAV Trinity F90+ was launched from the coast. Life jackets with different colors were provided for the subjects.

Four drones (DJI Matrice 100, DJI Matrice 210, DJI Mavic 2 Pro, and a Quantum Systems Trinity F90+) with different mounted cameras (see Tab. 6.1) were used. These different cameras diminish the effect of a camera bias.

Table 6.2: The classes, which were annotated for SeaDronesSee version 1.

Name	Description	With life jacket	On boat
Swimmer	A human swimming in the water without life jacket	×	×
Floater	A human swimming in the water with life jacket	✓	×
Swimmer†	A human on a boat without life jacket	×	✓
Floater†	A human on a boat with life jacket	✓	✓
Life jacket	A life jacket	-	-
Boat	A boat swimming in water	-	-

The multispectral camera MicaSense RedEdge-MX was referenced with a white reference before each flight. As the RedEdge-MX captures every band individually, we merge the bands using the development kit provided by MicaSense. Especially these recordings are relevant for the multispectral evaluation in Sec. 6.3. Besides the camera recordings, meta information, like GPS position and camera gimbal pitch, were captured. Meta information is not relevant to this work. Therefore we refer to our publication (Varga *et al.* (2022b)) for more details about these.

Annotation Method

The frames were annotated with the help of human experts. The six classes (visible in Tab. 6.2) were annotated in each frame. These classes were selected as they represent all objects of interest in the scenery. Furthermore, we marked regions with confusing objects (e.g., boats on land) as ignored regions.

Data Set Split

The color and the multispectral cameras are handled separately but with the same procedure. For the color recordings, we roughly balance the training, validation, and test set such that the subsets have similar distributions concerning altitude and angle of view. These two factors are significant for the appearance of the scene. For both recording types, we randomly select $4/7$ for the training set, $1/7$ for the validation set and another $2/7$ for the testing set. We also provide an object tracking task with a similar split for this data set. As we focus in this chapter on the object detection task of the multispectral cameras, we refer for more information on the tracking task and how the video recordings were handled to our publication (Varga *et al.*, 2022b).

6.2.3 Data Set Task

UAV-based maritime SAR missions have become popular recently, as shown in Sec. 6.2.1. These missions can be divided into subtasks (e.g., detection, optimal search pattern, long-distance communication). We focus here on the detection part.

Recordings of maritime scenes captured by UAVs have a variety of special characteristics, which make them especially hard for detection tasks:

- People may be hardly visible or occluded by waves or sea foam.
- The water surface is specular.
- Objects of interest are mostly small or tiny.

We note that these factors are on top of general UAV-related detection difficulties.

Color Object Detection

There are 5,630 images (training: 2,975; validation: 859; testing: 1,796). In the second iteration of the data set (SeaDronesSeev2), we adapted the classes to the actual application. This is further discussed in Sec. 6.2.5.

Multispectral Object Detection

The MicaSense RedEdge-MX records five channels, namely blue (475 nm), green (560 nm), red (668 nm), red-edge (717 nm and near-infrared (842 nm). The camera was mounted to the drone without a gimbal. Therefore, only recordings facing downward (90°) were captured. For this subset of recordings, this is acceptable as this is also most common for area searches.

The multispectral recordings are annotated in the same manner as the color recordings. Also, the training, validation, and test set split were done similarly. This data set is much smaller than the color data set. It contains 432 images with 1,901 instances. See Fig. 6.1 for an example of the individual bands. It was also extended in the second iteration.

6.2.4 Evaluation

In the related publication (Varga *et al.*, 2022b), we provide experiments with state-of-the-art object detectors and object trackers on the color recordings. Here, we focus on multispectral recordings. The experiments of object detectors on the multispectral recordings will be presented in Sec. 6.3.4.

6.2.5 Extension of the Data Set (SeaDronesSeev2)

A second acquisition series was performed to extend the variety in the data set. The first series was conducted in the summer of 2020 at Lake Constance. For the second series, the open sea was favored. Therefore, a measurement series in the North Sea was planned and carried out in the summer of 2021. The annotated data set was published as SeaDronesSeev2 with object detection and a multispectral object detection track via the MaCVi 2023 workshop (1st Workshop on Maritime Computer Vision, 2023). The

Table 6.3: The classes of SeaDronesSee version 2, which were annotated.

Name	Description	Superclass
Swimmer	A human swimming in the water	Person
Boat	A boat swimming in water	Vessel
Jetski	A Jetski swimming in water	
Life-Saving Suppliance	A life jacket or other life saver	Object
Buoy	A buoy swimming in water	

Figure 6.2: Examples of objects. Note that these examples are crops from high resolution images.

second version contains the recordings of both locations. The subset split (for training, validation, and test set) was performed as described in Sec. 6.2.2 with particular attention that annotations of the test set weren't public already. SeaDronesSeev2 contains 16,611 color images (training: 8,930; validation: 1,547; testing: 6,134) with 102,393 instances. The multispectral detection task contains 807 images (training: 461; validation 115; testing: 231) with 2,895 annotations.

The acquisition procedure and the used hardware, e.g., cameras and drones, did not differ from the described hardware, see Sec. 6.2.2.

A further difference to version 1 can be found in the annotated classes. Based on discussions with lifeguards of the *Deutsche Lebens-Rettungs-Gesellschaft e.V. (DLRG)*, we decided to adapt the classes to actual SAR applications. The classes of SeaDronesSeev2 can be found in Tab. 6.3. As a result, the distinction between swimmers with and without a life jacket is gone, as this is not so helpful for an actual rescue mission. Further, persons on boats are also not relevant for the detection task. And the new class buoy was introduced as there were no buoys on the Lake Constance recordings, and they are essential for orientation. In summary, the most critical part is to detect swimmers in the water. The objects nearby are informative to understand the situation better but are not as crucial as the swimmers.

6.2.6 Conclusion

This serves as an introductory benchmark in UAV-based computer vision problems in maritime scenarios. We built the first large scaled-data set for detecting and tracking humans in open water. Moreover, we provide multispectral imagery for detecting humans

in open water. These images are very promising in maritime scenarios, having the ability to capture wavelengths, which set apart objects from the water background. We hope this data set will boost future research in this direction.

Furthermore, we improved the data set (SeaDronesSee) with a second iteration (SeaDronesSeev2), containing more annotated samples and revised classes.

In the following sections, we will use this data set to analyze the object detection pipeline comprehensively. Afterward, we propose a technique to reduce the impact of background bias in remote sensing recordings.

6.3 Comprehensive Analysis of the Object Detection Pipeline on UAVs

Micro-aerial vehicle (MAV) or *unmanned aerial vehicle (UAV)* systems can support surveillance tasks in many applications. Examples are traffic surveillance in urban environments (Zhu *et al.*, 2020) or the described search and rescue missions in maritime environments.

For these tasks, object detection pipelines support the human operator and simplify the task. The camera system and the object detector are key components of these detection pipelines. The interaction of these components, generally based on image data, is crucial for the performance of the whole pipeline.

In most projects, the configuration of these components has to be conducted at an early stage. A recording of a necessary data set should be already done with the final camera system. Moreover, the camera system should be optimized based on an existing data set. This situation leads to a chicken-and-egg problem.

The emerging trend toward smart cameras, combining the camera and the computing unit, confirms the need for classification and detection pipelines in embedded systems. The lack of flexibility and the high price of these closed systems remain significant drawbacks and prevent their usage in many research projects or prototyping.

In the publication (Varga *et al.*, 2022a), we analyzed seven camera parameters (quantization, compression, color model, resolution, multispectral recordings, and camera calibration) and measured their impact on the performance of the detection pipeline in the remote sensing application. The selected parameters are easy to adjust for many systems or worth considering. Their effect on the detection performance was analyzed by varying these parameters separately. With optimized parameters, we showed that the pipeline is used more efficiently. This allowed, e.g., a higher throughput of the detection pipeline. By finding the sweet spots and analyzing the impact of the seven camera parameters, we provided recommendations for further projects, which we will discuss in Sec. 6.3.5.

In this section, we focus on the experiments and the results performed with multispectral recordings. Further, we present the experiments related to the color model, as these are related to defining the number of used channels. For the other parameters, we refer the interested reader to (Varga *et al.*, 2022a).

The experiments were performed in a desktop and an embedded environment, as both are relevant for remote sensing. An embedded object detection pipeline mounted on a UAV was used to prove the usability of the results.

6.3.1 Related Work

The image processing pipeline significantly impacts the performance of object detectors. Many parameters have impact on the image quality and influence this pipeline. However, there is still a lack of impact analysis for UAV-based systems. Yahyanejad *et al.* (2011) have measured the impact of lens distortion correction for thermal images acquired by UAVs. However, their results are only helpful for a thermal camera system.

Other existing camera parameter evaluations are mainly based on the autonomous driving scenario, e.g., the works of Blasinski *et al.* (2018), Carlson *et al.* (2018) Liu *et al.* (2019), Liu *et al.* (2020), Saad and Schneider (2019) or Secci and Ceccarelli (2020). The findings of the autonomous driving experiments cannot be directly applied to the remote sensing use case. The scene is often homogenous for autonomous driving, and the object sizes differ only slightly. Further, the distance to the objects is often only a couple of meters. For remote sensing recordings, the distance to the scene is often over 30 m, as shown in Varga *et al.* (2022b), resulting in tiny objects. Depending on the camera angle, the images can vary enormously. Because of these differences, the results of one scenario are not entirely valid in the other but can give the first sign.

Blasinski *et al.* (2018) proposed a rendering framework to evaluate camera architectures in simple autonomous driving scenarios. Secci and Ceccarelli (2020) evaluated camera failures in the autonomous driving application. Besides these, the works of Liu *et al.* (2019), Carlson *et al.* (2018) and Saad and Schneider (2019) showed the still existing gap between synthetic and real-world data. Thus, our experiments are based entirely on real-world recordings.

Buckler *et al.* (2017) analyzed the image pipeline regarding performance and energy consumption. In contrast to their work, we close the gap for a missing camera parameter evaluation in the remote sensing application. Further, we consider parameters that are configurable for most off-the-shelf camera systems.

In contrast to work that accelerates inference by, e.g., quantizing the neural network like the works of Li *et al.* (2019a) or Cai *et al.* (2020), our work focuses on the parameters of the object recognition pipeline that affect the overall throughput time of the system, rather than on the inference time itself.

In summary, our work differs from those mentioned above in the following aspects. We focus on the remote sensing use case for UAVs and use established real-world data sets instead of synthetic data for the evaluation. Finally, the evaluated parameters are configurable for most cameras.

6.3.2 Materials and Methods

Three data sets and four object detectors were used. The experiments were done on a desktop and an embedded environment. The selected camera parameters are described in further detail.

6.3.3 Experiment Setup

The setup contains the data sets, the object detectors, and the hardware environment. For each configuration, we trained with three different random seeds. The random seed affects the weight initialization, the order of the training samples, and data augmentation techniques. The three initializations show the stability of each configuration.

Data Sets We evaluated our experiments on three remote sensing data sets. Dota-2 is based on satellite recordings (Ding *et al.*, 2022). VisDrone (Zhu *et al.*, 2020), and SeaDronesSee were remotely sensed by *Micro aerial vehicles (MAVs*). The first two data sets are well-established and commonly used to benchmark object detection for remote sensing. In contrast, SeaDronesSee focuses on a maritime environment, which differs from the urban environment of VisDrone and Dota-2. Therefore, it introduces new challenges and properly extends our experiments.

Models For our experiments, we selected four object detectors. The two-stage detector Faster-RCNN, introduced by Ren *et al.* (2017), can still achieve state-of-the-art results with minor modifications, as shown in the VisDrone challenge of Zhu *et al.* (2020). By comparison, the one-stage detectors are often faster than the two-stage detectors. Yolov4 by He *et al.* (2016) and EfficientDet by Tan *et al.* (2020) are one-stage detectors and focus on efficiency. Both perform well on embedded hardware. Modified versions of CenterNet, proposed by Duan *et al.* (2019), also a one-stage detector, could achieve satisfying results in the VisDrone challenge by Zhu *et al.* (2020). These four models cover a variety of approaches and can give a reliable impression of the parameter impacts.

We used multiple backbones for Faster R-CNN, EfficientDet, and CenterNet to provide insight into different network sizes. A full description of the training procedures and hyperparameters can be found in Varga *et al.* (2022a).

Hardware Setup We did most of the experiments in a desktop environment. To evaluate the performance in a more appropriate setting, we performed experiments on an Nvidia Xavier AGX board. The small size and the low weight allow the usage for onboard processing in MAVs or other robots.

The infield experiments were done with a system carried onboard a DJI Matrice 100 drone. An Nvidia Xavier AGX board mounted on the drone performed the calculations, and an Allied Vision 1800 U-1236 camera was used for capturing.

Inspected Camera Parameters

All parameters were selected based on their direct influence on the recording.

Here, we will focus on the parameters which define how the spectrum of the captured light is represented for each pixel. For color cameras, the color model defines this representation. Altering the color model usually does not affect memory consumption, as most models use three distinct values. If the color information can be neglected, the representation can be reduced to a single brightness value, reducing memory. In contrast, Multispectral camera recordings contain channels outside the visible light and color channels. These additional channels can increase detection performance at the price of more values per pixel and increase memory consumption. For the usage of Multispectral recordings, it is necessary to check the integration effort for existing architectures and whether the performance improvement is worth the effort and the additional memory.

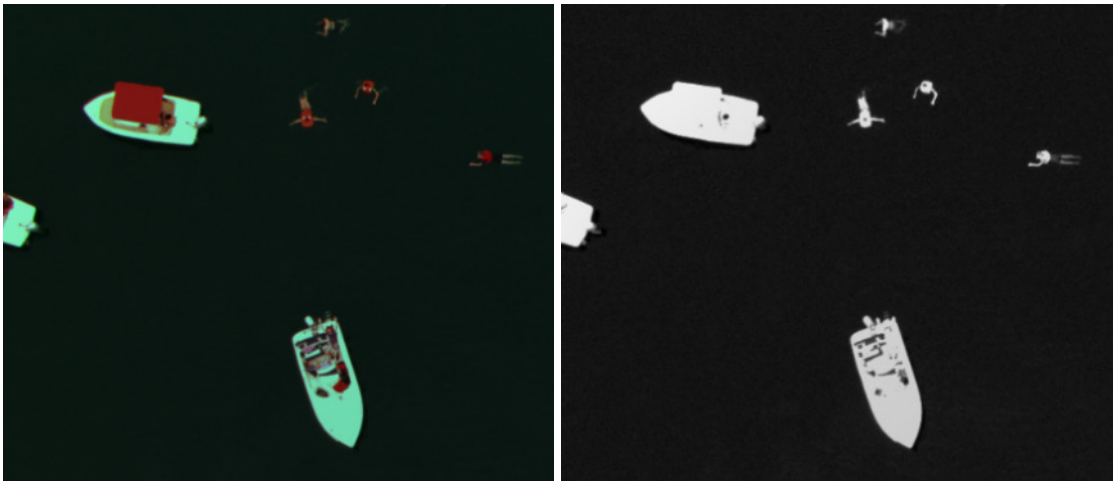
As mentioned in the introduction, we will consider only the parameters related to the spectrum representation (color model and multispectral recordings) in this work. We only give an overview for the following parameters and refer to the published work.

Three parameters, quantization, compression, and resolution, control the required memory consumption. By reducing the memory consumption with one of these parameters, the throughput can be increased, or another parameter can utilize the freed memory. Further, the impact of camera setup and calibration was evaluated. These do not affect the throughput of the pipeline. Nevertheless, it is worth checking how vital a perfect calibration is. The radial image distortion, which originates from the camera lens, and gamma correction, which simulate different exposure settings for a prerecorded data set, show the effect of non-optimal camera calibration.

Color Model The color model defines the representation of the colors. The most common is RGB, described by Hunt (2005). It is an additive color model based on three colors red, green, and blue. This representation is mainly used for object detection and object classification, e.g., mentioned by Ren *et al.* (2017). Besides RGB, non-linear transformations of RGB are also worth considering. HLS and HSV are based on a combination of hue and saturation. Both differ in the brightness's definition value. YCbCr considers human perception and encodes information more suitably for humans.

In some applications, HSV outperforms RGB. Cucchiara *et al.* (2001) and Shuhua and Gaizhi (2010) could confirm this performance improvement. Kim et al. demonstrated RGB outperforms the other color models (HSV, YCbCr, and CIE Lab) in traffic signal detection. It seems that which color model is best depends strongly on the application and the model.

The color model is also a design decision, so we evaluate the performance of RGB, HSV, HLS, and YCbCr color models for object detection in remote sensing. We also review the performance of gray-scale images. To achieve comparable results, the backbones are not pre-trained for these experiments.



(a) (b)

Figure 6.3: Example images for multispectral recordings in maritime environment. In the NIR recordings, the swimmers are fully visible. (**a**) RGB; (**b**) Near-infrared.

Multispectral Recordings In recent years, the usage of multispectral cameras on drones has gotten more popular, e.g. Candiago *et al.* (2015), Deng *et al.* (2018) or Zhang *et al.* (2019). Multispectral recordings have additional channels, which increase the recording size but can also provide important information. The additional channels typically lie outside the visible light (e.g., near-infrared or thermal). In pedestrian and traffic monitoring, multispectral imaging can increase the detector performance, as shown by Karasawa *et al.* (2017), Vandersteegen *et al.* (2018), and Guan *et al.* (2019). And for precise farming, they are one of the key components, as discussed by Candiago *et al.* (2015), Deng *et al.* (2018) or Zhang *et al.* (2019). We want to create awareness that multispectral recordings can be helpful in some situations. But we also want to check how complicated the integration into existing object detection pipelines is. So, the focus is not on evaluating multimodal models, as described by Ophoff *et al.* (2019). Our experiments are limited to the early fusion approach like Zhang *et al.* (2021) as we see multispectral recordings as a special case of color images.

We used multispectral images from the SeaDronesSee data set for our experiments. These were acquired with a MicaSense RedEdge-MX and thus provided wavelengths of 475 nm (blue), 560 nm (green), 668 nm (red), 717 nm (red edge), and 842 nm (near infrared). We evaluate how the additional wavelengths affect the performance of the object detectors in detecting swimmers. In theory, the near-infrared channel provides a foreground mask for objects in water (see example in Fig. 6.3). These experiments should provide a general trend, even if the exact results apply only to the maritime environment. It is worth considering if additional channels could carry helpful information.

6.3.4 Results

The experiments of the different backbones per network were grouped. Only the largest backbones of CenterNet and Faster R-CNN are listed separately. We use the mean average precision (mAP) as a metric with an intersection over union (IoU) threshold of 0.5, as proposed by Lin *et al.* (2014b). The mean average precision metric averages the precision values of the predictions over the recall values and all classes. The IoU threshold defines the minimum overlap between a prediction and a ground-truth bounding box to be counted as a positive sample. It is the most common metric to measure the performance of object detectors.

For the Dota data set, the performance of the models differs from the performance in the other data sets. The objects in the Dota data set are tiny. As a result, it challenges the object detectors differently.

Color Model

We can conclude two essential outcomes of the experiment for color models. First, we must distinguish between two situations visible in Fig. 6.4. In the first situation, the object detectors cannot take advantage of the color information. Therefore, we assume the color is, in this case, not beneficial. This applies to the VisDrone data set and the Dota data set. Both data sets provide objects and backgrounds with a high color variance. For example, a car in the VisDrone data set can have any color. Thus, the texture and shape are more relevant. That means the object detector can achieve excellent results with the gray-scale recordings. The color information for the Dota data set could only be utilized by the largest models (Faster R-CNN/ResNext101_32x8d and CenterNet/Hourglass104). The other models couldn't incorporate color and relied only on texture or shape.

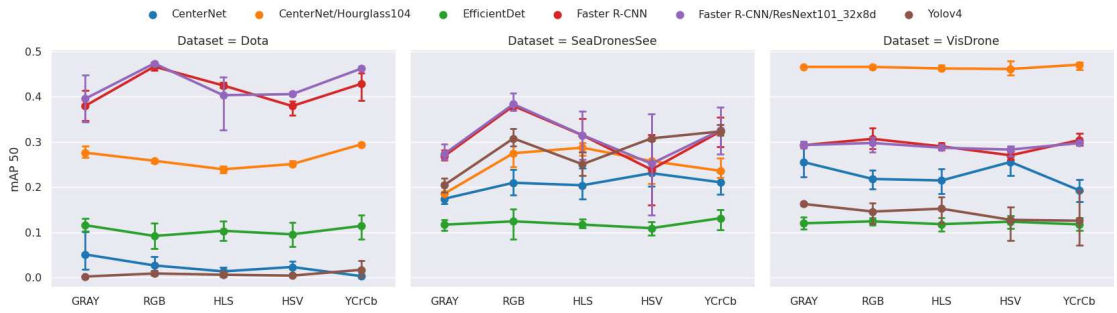


Figure 6.4: Color Model: the impact of the color models. The y-axis shows the mean average precision (mAP).

In the SeaDronesSee experiments, we observed that the color experiments outperformed the gray-scale experiments. The explanation can be found in the application. It covers a maritime environment. As a result, the background is always a mixture of blue

and green. So, the color information is reliable in distinguishing between background and foreground. For SeaDronesSee, all models use color, which is evident in the drop in performance for the gray-scale experiments.

If the color is essential, RGB produces the best results, followed by YCrCb. However, since most data augmentation pipelines have been optimized for RGB, we recommended using RGB as the first choice.

In conclusion, knowing whether color information is relevant to the application is crucial. Gray-scale images are usually sufficient, leading to a third of the image size, and may even allow more straightforward cameras.

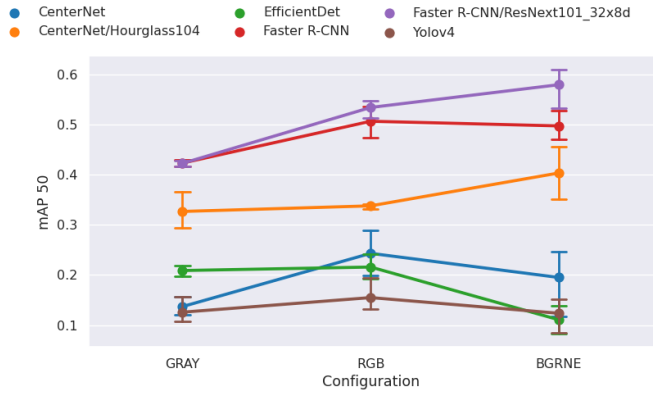


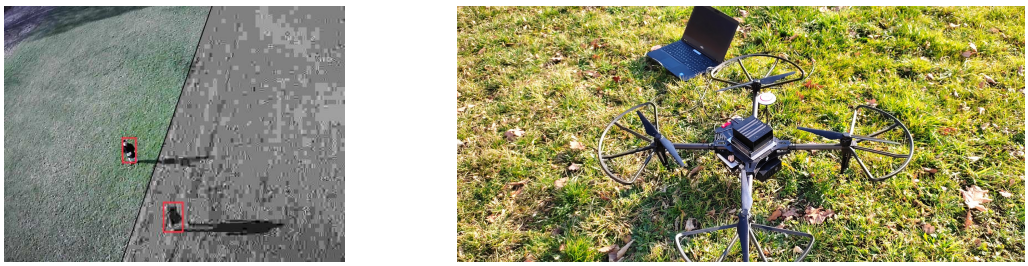
Figure 6.5: Performance of object detectors on multispectral data. ('BGRNE'—multispectral recordings, 'RGB'—color images, 'GRAY'—gray-scale).

Multispectral Recordings

These experiments are based on multispectral data from the SeaDronesSee data set. The usefulness of the additional channels depends strongly on the application. For the maritime environment, additional information is beneficial, especially in the near-infrared channel. Curcio and Petty (1951) showed that this wavelength range is absorbed by water, resulting in a foreground mask (an example is shown in Fig. 6.3).

The results in Fig. 6.5 fit the findings for the color models (see Sec. 6.3.4). For the maritime environment, color information is essential, as indicated by the gap for gray-scale recordings. The multispectral recordings (labeled as 'BGRNE') can only improve the performance of the larger models. For the smaller models, they even lead to a decrease in performance. To support the multispectral recordings as input, we had to increase the input channels of the first layer and lose the pretraining.

In summary, additional channels can be helpful depending on the use case. It is essential to remember that larger backbones make better use of the extra channels. Furthermore, it is a trade-off between a fully pre-trained backbone and additional information.



(a) (b)

Figure 6.6: Prototype system using an optimized detection pipeline. (**a**) Original input and optimized input; (**b**) Detection pipeline onboard a DJI Matrice 100.

6.3.5 Discussion

In the experiments of the publication, we showed that the inspected parameters have an unequal impact on the performance of a model. Some have a minor effect on the performance but significantly impact the required space (Quantization, Compression). Others have a negligible influence on the performance (Color Model, Calibration Parameters), and some are fundamental for the remote sensing application (Resolution). In section 6.3.4, we showed that more complex camera systems, e.g., multispectral cameras, can boost performance; however, only if the additional channels carry helpful information. As a result, we want to emphasize keeping the whole object detection pipeline in mind when building an object detection system.

As the parameters have an unequal impact on the performance, we can reduce the memory consumption of the parameters, which have a minor effect on the performance. The freed space can be used to increase the throughput of the system or can be utilized to boost the crucial parameters by assigning more memory space to these.

Based on previous investigations and assumptions, we determine the optimal parameter configuration for the remote sensing application. These parameters are proposed as a recommendation for future work. The objective was to find a suitable compromise between the detection performance and the required space for a recording, which limits the pipeline's throughput and defines the frames per second of the system. The combination of the optimal parameters utilizes the pipeline more efficiently. For a maximum throughput with a negligible loss of performance, we recommend the following:

1. A quantization reduction from 8 bit to 4 bit is sufficient, as the models do not utilize the entire color spectrum. This was shown by Varga *et al.* (2022a).

2. A JPEG compression with a compression quality of 90 reduces the performance slightly but the memory footprint significantly. This was shown by Varga *et al.* (2022a).

3. For the urban surveillance use-case, color is not essential. Thus, for the VisDrone

data set, we recommend gray-scale images. For the other data sets, color is essential. This was shown by Varga *et al.* (2022a).

The mentioned recommendations lead to a reduction in memory consumption. This could be used to increase the frames per second of the system. This thesis is proven in Table 6.4. The first setup (marked *), which uses reduced quantization, JPEG compression, and optional color reduction, significantly increases the throughput (including the inference and the pre-and post-processing) of the smaller models. In contrast to the baselines, we speed up the detection pipeline ($\approx$+15%) with a small loss in performance ($\approx -5\%$). For the larger models, inference accounts for most of the throughput, so there is no significant speed-up.

If the best performance is the target of the project, we recommend utilizing the freed space with the additional recommendation:

4. Utilize the highest possible resolution because this is a crucial parameter for small objects of remote sensing applications. This was shown Varga *et al.* (2022a).

Combining the first three adjustments and increasing the resolution (marked **) in Table 6.4, the detector's performance can be improved ($\approx$+25%) while maintaining the base throughput.

Further, we can conclude the following statements based on the experiments. These do not affect the trade-off between performance and throughput but are still important for future projects.

5. As has been shown in Varga *et al.* (2022a), a minor error in the camera calibration (distortion or gamma) is not critical.

6. In specific applications, multispectral cameras can be helpful. For example, we have shown in section 6.3.4 that multispectral recordings boost the performance in maritime environments.

6.3.6 Conclusion

We summarized the results of Varga *et al.* (2022a) and presented them with a focus on the multispectral recordings and the color models.

It is always worth checking whether the color information or additional wavelength channels are helpful for the targeted task. Further, it is crucial to remember that additional channels require special attention. Larger models can handle these better, and the pretraining of at least the first layer is usually unusable.

Besides that, the experiments showed that not all parameters have an equal impact on detection performance, and a trade-off can be made between detection accuracy and data throughput. By using our recommendations, it is possible to maximize the efficiency of the object detection pipeline in a remote sensing application.

Table 6.4: Optimized parameter configuration in a desktop and an embedded environment: The mean average precision (mAP) are presented. For each model, at least two configurations are given (baseline and optimized pipeline). Frames per second (FPS) indicate the inference speed. The percentages in the parentheses indicate the change regarding the baseline. *) the first setup and **) the second setup are desribed in Sec. 6.3.5. **Bold** numbers indicate the maximal value for each configuration.

Data Set	Dota mAP 50 ↑	VisDrone mAP 50 ↑	SeaDronesSee mAP 50 ↑	avg. FPS ↑ (Desktop)	avg. FPS ↑ (Embedded)	Parameters
EfficientDet	23.83	20.71	22.29	25	14	4.5M - 17.7 M
Reduced data *	22.07 (-7 %)	19.53 (-5 %)	22.20 (-1 %)	**28 (+12 %)**	**16 (+14 %)**	
Reduced data + higher res. **	**30.85 (+ 29%)**	**25.52 (+23 %)**	**27.35 (+23 %)**	25	14	
YoloV4	16.19	24.49	40.62	20	4	63.9M
Reduced data *	13.84 (-14 %)	23.61 (-3 %)	36.58 (-9 %)	**21 (+5 %)**	**5 (+20 %)**	
CenterNet	21.35	29.94	31.08	40		14.4M - 49.7 M
Reduced data *	21.22 (-1 %)	27.87 (-6 %)	31.05 (-1 %)	**46 (+15 %)**		
CenterNet/ Hourglass104	44.63	52.07	51.12	6	-	200M
Reduced data *	41.15 (-7 %)	48.59 (-6 %)	47.89 (-6 %)	6		
Faster R-CNN	46.67	35.45	39.94	17		41.4M - 60.3M
Reduced data *	43.92 (-5 %)	33.05 (-6 %)	39.26 (-1 %)	**18 (+6 %)**		
Faster R-CNN/ ResNext101_32x8d	47.78	37.70	40.76	9	-	104M
Reduced data *	44.06 (-7 %)	35.87 (-4 %)	40.59 (-1 %)	9		

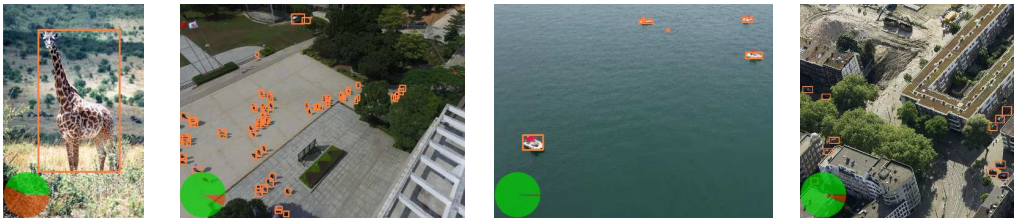
6.4 Tackling the Background Bias of Remote Sensing Object Detection

In this section, we propose a technique to tackle the background bias for object detection based on color images recorded by aerial systems. This evaluation focuses on color images, as many established color data sets are available. The same technique should also be applicable for sparse multispectral recordings, but the lack of data sets prevent a reliable statement. Still, we provide results on the multispectral recordings of SeaDronesSee to support this claim. We will discuss this method only briefly and refer for more information to the related publication (see Varga and Zell (2021)).

6.4.1 Introduction

Nowadays, state-of-the-art object detectors can accurately predict objects in generic object data sets like MS COCO (Lin *et al.*, 2014a) or Pascal VOC (Everingham *et al.*, 2010). For recordings of remote sensing applications, it is still challenging. Remote sensing covers all applications recorded by aerial systems, e.g., UAVs, planes, or satellites.

The recordings of remote sensing are often sparse, as they often show small objects of



(a) MS COCO (b) VisDrone (c) SeaDronesSee (d) DOTA-2

Figure 6.7: Example images of different data sets with orange ground-truth bounding boxes. The orange-green pie-charts show the foreground-background-ratio in the training data set. (orange: foreground; green: background)

interest with a lot of background. This leads to an unbalanced ratio between objects and background, resulting in a background bias. This affects the prediction capability of the object detectors.

Fig. 6.7 shows four example images of different data sets. The pie charts indicate the ratio of foreground to background pixels in the training sets. For MS COCO, it is almost balanced. This is not the case for remote-sensing data sets (like VisDrone by Zhu *et al.* (2020), our SeaDronesSee, and DOTA-2 by Ding *et al.* (2022)). So, remote sensing data sets have an additional challenge in contrast to common object data sets.

Further, the objects in remote sensing data sets are primarily small. This is problematic, especially with the down-scaling often used by object detectors to speed up the inference.

Here, we propose a deterministic cropping-based data augmentation, which reduces the background bias of sparse recordings and allows the training on higher resolutions within the crops.

6.4.2 Related Work

A good solution to reduce the bias, e.g., for a class, is using an adapted loss function. Lin *et al.* (2017) introduced Focal loss, which focuses on the hard examples. Therefore, it can perform well even with class imbalance and as a result, is part of many modern models. We want to tackle the bias between foreground and background, so their loss is also beneficial in this application. Two of our models (CenterNet and EfficientDet) use the Focal loss by default. As shown later, this commonly used loss function can only partially solve this problem.

Data augmentation techniques such as Random Cropping, Random Image Cropping And Patching (RICAP) by Takahashi *et al.* (2020), or CutMix by Yun *et al.* (2019) use, like our approach, crops. Random Cropping cuts random crops out of the input image. Takahashi et al. introduced RICAP, which crops four images and combines them. Yun et al. proposed CutMix as a combination of Mixup and CutOut. Therefore, CutMix

replaces the cutouts with crops of different images. In contrast to our method, these approaches focus on improving the object detector to recognize parts of an object, reducing the contextual impact, and simulating occlusion. These techniques are perfect for medium- or large-scale objects occurring in MS COCO or Pascal VOC. But, these are counterproductive for the detection on remote sensing recordings with mainly small and sparse objects. Our method is deterministic and defines only one representation of the training data. No random augmentation takes place. Thus, combining these augmentation techniques and our approach is still possible and shown.

Besides these augmentation techniques, Hong *et al.* (2019) proposed a patch-level augmentation approach. They address class imbalances as a problem of UAV data sets. After the training procedure, their proposed method generates hard samples with misclassified object instances. These hard samples are used for further training. Similar to their approach, we focus on a class imbalance but on the foreground-to-background imbalance. Xiong *et al.* (2021) proposed a similar approach. Their method over-samples small objects by augmenting the images with additional instances of small objects. For their augmentation, the segmentation masks of the objects are required, which is a costly requirement for the data set. Many data sets do not provide the ground truth segmentation mask. This also applies to the used data sets. Xia *et al.* (2018) recommend for their data set DOTA a cropping technique similar to our approach. We could still boost the performance on the DOTA data set as shown in the experiments.

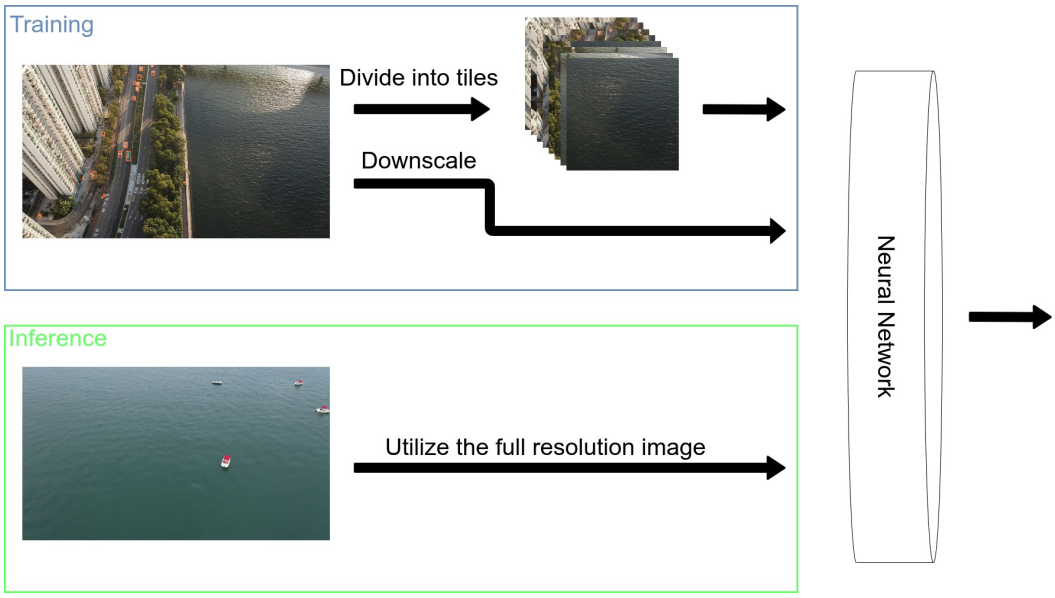


Figure 6.8: The method Cropping Window (CroW) and a visualization of the overlapping cropping pattern.

6.4.3 Method: Cropping Window (CroW)

The diagram in Fig. 6.8 visualizes the method. As shown, we only adapt the training process and not the inference process, which is also the main difference to the method of Unel *et al.* (2019), which focus mainly on inference. To reduce the background bias, we propose a deterministic way to change the representation of the training data.

Our method utilizes additional samples, which are generated via an overlapping cropping pattern. Each image is cropped into sub-images. Only the sub-images containing at least one object are kept, the rest are discarded. The extended training set consists of the original images (with the whole overview) and sub-images (with focus on the objects). As a result, the new training set has a reduced background bias.

In addition, it is possible to train with this technique on higher resolutions. As the crops cover only a subpart of the image, higher resolutions can be utilized for these during the training procedure, as long as the resolution of the training is just restricted by the *Graphical Processing Unit (GPU)* memory.

At inference, the entire frame in the maximal possible resolution is used. So, no merging of predictions is necessary.

Table 6.5: This table shows the mean average precision with a IoU threshold of 0.5 ($mAP^{0.5}$) for different configurations. Value per cell: mean $\pm$ standard deviation. **Bold** numbers indicate the maximal value for each configuration.

Data Set	Dota mAP 50 ↑	VisDrone mAP 50 ↑	SeaDronesSee mAP 50 ↑	avg. FPS ↑ (Desktop)	avg. FPS ↑ (Embedded)	Parameters
EfficientDet-*d*0	19.54 ± 0.64	18.85 ± 3.65	19.35 ± 0.79	46	8	4.5M
With CroW (our)	**25.30 ± 1.93**	**31.21 ± 0.64**	**33.97 ± 4.04**			
EfficientDet-*d*4	19.66 ± 0.60	24.77 ± 1.24	24.82 ± 0.83	21	-	17.7M
With CroW (our)	**27.61 ± 1.09**	**30.41 ± 0.44**	**44.01 ± 2.43**			
YoloV4	17.74 ± 2.00	30.75 ± 1.64	6.44 ± 0.53	20	-	63.9M
With CroW (our)	**28.65 ± 5.40**	**36.41 ± 1.40**	**22.04 ± 0.83**			
CenterNet-ResNet18	24.20 ± 2.27	23.41 ± 1.52	15.18 ± 0.56	78	-	14.4M
With CroW (our)	**26.88 ± 0.73**	**31.49 ± 1.54**	**19.67 ± 0.71**			
CenterNet-ResNet50	29.73 ± 0.50	23.24 ± 2.99	15.85 ± 2.31	33	-	30.7M
With CroW (our)	**35.25 ± 0.57**	**33.07 ± 0.37**	**23.06 ± 7.53**			
CenterNet-ResNet101	26.14 ± 3.18	20.63 ± 1.21	14.19 ± 0.62	22	-	49.7M
With CroW (our)	**33.63 ± 1.61**	**33.68 ± 2.59**	**27.05 ± 6.72**			
CenterNet-Hourglass104	45.77 ± 8.49	51.41 ± 1.06	43.26 ± 0.86	6	-	200M
With Random Cropping	52.47	46.65	50.69			
With CroW (our)	**55.51 ± 0.23**	**53.77 ± 0.22**	**52.40 ± 0.23**			

6.4.4 Experiment

We used three different data sets to evaluate our method. All three are based on aerial recordings. VisDrone by Zhu *et al.* (2020) and our SeaDronesSee consist of *Micro aerial vehicle (MAV)* recordings. DOTA-2 by Ding *et al.* (2022) contains images recorded with satellites. For all experiments, we use the metric framework proposed by Lin *et al.*

(2014a), which is also common for MS COCO. Further, we used three one-stage object detectors for our experiments. The efficiency-optimized object detector EfficientDet by Tan *et al.* (2020), the anchor-free object detector CenterNet by Duan *et al.* (2019), and the well-established Yolov4 by Bochkovskiy *et al.* (2020) are used for the color experiments.

Besides the three mentioned models, we compared our results with approaches of Unel *et al.* (2019), Pailla *et al.* (2019) and similar augmentation techniques such as Random Cropping and Mosaic augmentation (He *et al.*, 2016), which are discussed in total length in the related publication.

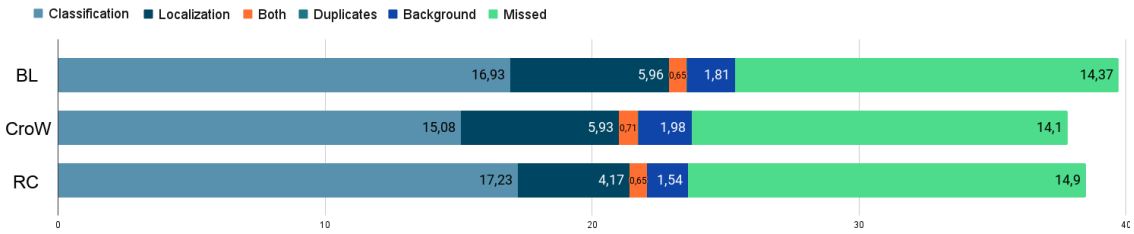


Figure 6.9: TIDE analysis of the trained CenterNet-Hourglass104 on VisDrone-DET. Larger values indicate a larger error. (BL: Baseline; CroW: With Cropping Window (our); RC: With Random Cropping)

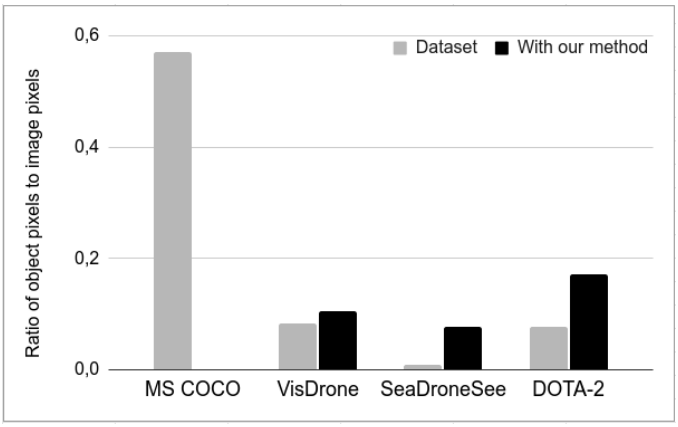


Figure 6.10: Ratio between the annotated pixels and the image pixels for different data sets. In contrast, MS COCO is much more balanced.

6.4.5 Results

In Tab. 6.5 the mean average precision (*mAP*) are given. We stated the mean and standard deviation for each configuration over three runs with different seeds. All models improve for all data sets with 2 to 13 *mAP* points compared to the baseline model. No adaption

of the network architecture is necessary. Therefore, the number of parameters and the inference time do not change.

In Fig. 6.9, a TIDE analysis (Bolya *et al.*, 2020) of a CenterNet-Hourglass104 trained with and without our method is shown. Also, a model trained with Random Cropping is presented. TIDE is based on the *mAP* evaluation and can break down the cause for the missed accuracy. There are two essential differences visible between the baseline and our approach. First, our method reduced the *Missed* error and the *Classification* error. So the second detector was better at classifying classes and missed fewer objects. Further, the *Background* error is increasing minimally. Nevertheless, there is still enough background for learning, even if slightly worse. The focus is moved more to the object of interest. In contrast, Random Cropping improves the localization only slightly and worsens classification. As a remark for the VisDrone data set, the ratio of the classification error is still enormous for all approaches. Fig. 6.10 shows the impact of our method on the foreground-background-ratio for the three data sets.

In summary, we showed the method works reliably in different settings (data sets and models). Further, we proved with TIDE that the impact of background bias was reduced

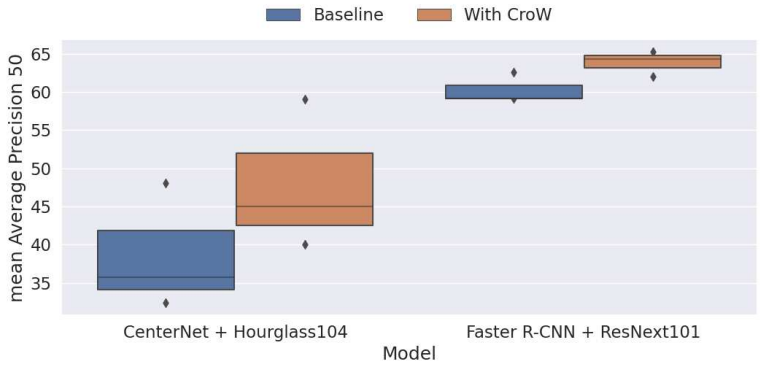


Figure 6.11: CroW for multispectral image object detection.

6.4.6 Evaluation for Multispectral Recordings

To support the claim that this technique is also usable for multispectral imaging, we performed experiments on the multispectral recordings of the SeaDronesSeev2 data set. We selected the largest models of Sec. 6.3.3, as these could utilize the multispectral recordings best, and evaluated the performance with and without the proposed cropping window method. The results are visible in Fig. 6.11. The performance boost is also observable for the multispectral recordings. The technique reduces the background bias and allows the object detectors to focus more on class differences. Although it is only a tiny multispectral data set, it seems promising that this also holds for other remote sensing settings with multispectral data. This experiment also showed that this method

works for two-stage detectors as well. We expected the two-stage approach solves this problem by design, but this seems to be not the case.

6.4.7 Conclusion

We introduced a simple technique to improve the capabilities of object detectors on sparse recordings by tackling background bias. This method is easy to implement into existing object pipelines and enhances performance. By showing this easy-to-achieve improvement, we want to draw attention to the problem of background bias in remote sensing recordings. Most of the experiments were performed on color-images. However, it was still shown that the behavior is also expected for multispectral object detection on sparse recordings.

6.5 Summary

In this chapter, we introduced a data set for maritime search and rescue missions. Part of this data set are multispectral recordings of humans in open water. These images are very promising in maritime scenarios, having the ability to capture wavelengths, which set apart objects from the water background.

We used the data set for a comprehensive analysis of the object detection pipeline. One result of this evaluation is that it is always worth checking, whether additional wavelength channels could boost the model performance further. Further, we defined some recommendations, which help to optimize the efficiency of an object detection pipeline in a remote sensing application.

Afterwards, we proposed a technique to reduce the impact of the background bias in remote sensing recordings. This was mainly evaluated on color images, but also holds for multispectral recordings.

Chapter 7

Conclusions

In this chapter, we summarize and discuss the results of this thesis. Finally, we conclude with an outlook for future work in this research area.

7.1 Summary

In this work, different applications of *hyperspectral imaging (HSI)* and *multispectral imaging (MSI)* were presented. After presenting the foundations in chapter 2, food inspection applications are introduced in chapter 3. Based on two classification examples, a simple convolution neural network, called *DeepHSNet* was developed and evaluated. It can produce reliable results for the ripeness prediction of exotic fruits and the detection of Phakopsora Pachyrhizi infestation in soybean. For both applications, a data set was recorded and annotated. The ripening fruit data set, called *DeepHS Fruit*, was made publicly available and was used further throughout this work.

As annotating hyperspectral data sets is time-consuming, and there are no pretrained models, incorporating unlabeled recordings is favored. One way to achieve this is using self-supervised pretraining, which is analyzed in chapter 4. Three state-of-the-art contrastive learning approaches (SimCLR (Chen *et al.*, 2020), SimSiam (Chen and He, 2021), Barlow Twins (Zbontar *et al.*, 2021)) were selected and evaluated on the extended fruit ripening data set *DeepHS Fruit v2*. It was shown that self-supervised pretraining could stabilize the downstream classification training. All three methods produced comparable results. In this context, the impact of different augmentation techniques and the modification with 3D convolutions was analyzed and proposed.

In chapter 5, a wavelength-aware 2D convolution is presented. Based on learnable Gaussian distributions, the convolution can select the essential wavelength ranges in a data-driven way. The critical component of this approach is introducing the bias 'Similar wavelengths show similar features'. This reduces the trainable parameters and supports the training of camera-agnostic models. The convolution, which is called *HyveConv++*, was evaluated on the extended version of the fruit ripening data set *DeepHs Fruit v2* and a small established hyperspectral data collection of remote sensing scenes (HRSS). On both data sets, it could outperform comparable methods. The learnable Gaussian distributions, called *Wavelength Ranges of Interest (WROI)*, mimic the behavior of camera

filters. So, it is possible to utilize the learned WROIs to build a multispectral camera based on data-driven training.

In chapter 6, it is shown why the simplification of a hyperspectral camera to a multispectral camera is favored. Multispectral cameras are significantly easier to handle and employable in embedded environments. The example of maritime *search and rescue (SAR)* missions shows the benefit of multispectral cameras drones mounted on *unmanned aerial vehicles (UAV)*. The additional channels can add valuable information and boost the model performance for specific tasks, like in the discussed example. Besides acquiring the data set *SeaDronesSee (v2)*, the impact of camera parameters was analyzed. In addition, it was shown that most remote sensing data sets are affected by a background bias. A cropping-based technique, named *CroW*, was presented.

In summary, two large data sets (*DeepHS Fruit v2* and *SeaDronesSee v2*) were published. Further, three algorithmic developments were presented: a technique to reduce background bias in remote sensing recordings (*CroW*), a convolution neural network for hyperspectral imaging (*DeepHSNet*), and a wavelength-aware 2D convolution for hyperspectral imaging (*HyveConv++*). Besides these contributions, the impact of different camera parameters on the performance of the object detection pipeline for remote sensing was evaluated and the usefulness of self-supervised pretraining for hyperspectral imaging was analyzed.

7.2 Discussion

In the introduction of this thesis, we named four challenges of hyperspectral imaging, which also apply in eased form for multispectral imaging.

The lack of publicly available data sets is an obvious challenge for both techniques. As mentioned in chapter 2, small data sets are available for hyperspectral imaging. Still, these are not comparable in size or quality with the large-scale data sets of color image tasks. For example, the HRSS data set is well-established but tiny, and the evaluation metric is inconsistent within the community (e.g., training-test-splits is not fixed). As a result, many approaches are not comparable and many methods just fine-tune for a specific task of one of the small data sets. We tackled this challenge by introducing a new, relatively large, hyperspectral data set (*DeepHS Fruit v2*). The situation is similiar for multispectral imaging, but there is not even an established data set like HRSS.

The task-specific features are the next challenge, which is also one reason for the sparse data sets. The features necessary to solve one data set's task can widely differ from those required for another data set. As long as models are only validated on a single hyperspectral task, it is unclear whether they are only fine-tuned for the specific features of this data set or capable of handling hyperspectral recordings in general. That's also why classical machine learning approaches (e.g., SVM or k-NN), which are less complex, are still favored for unknown hyperspectral applications. Even if incorporating spatial information is much more straightforward for convolutional neural networks. In-

stead of fine-tuning models on specific tasks, the research community's goal should be to design models capable of handling hyperspectral data in general. With the published data set *DeepHS Fruit v2*, we propose a data set that complicates this fine-tuning for a specific setting with the different configurations(fruit type, ripeness type, camera). The model has to adapt to different settings as the important features for different fruit types differ. But it is still far from a general benchmark.

As we mentioned, data augmentation techniques, which are very important for complex models, are not as straightforward as for color images. We performed some experiments regarding data augmentations in chapter 4. But these are far from a comprehensive analysis of this topic. At this point, we can only highlight that it is vital to remember that the hyperspectral cube represents an underlying spectrum for each pixel. Spatial modifications are straightforward and beneficial. Spectral changes require a deeper understanding. But still, dropping not-consecutive channels can help the model to become more robust to noisy channels.

The last named challenge points to the large channel dimension. In a nutshell, this is not a problem for deep-learning models. As we hypothesized, convolutional neural networks can handle high-dimensional hyperspectral cubes well. As a result, dimension reductions (e.g., PCA, Factor Analysis, IBRA) as preprocessing are not necessary or even obstructive, especially as these are often not part of the end-to-end training procedure.

Further, the common trend of larger models does not apply for the small HSI data sets. And as we showed in chapter 5, an appropriate bias seems like a much more promising way for HSI.

Even if we mainly focused on hyperspectral imaging in this discussion, it also largely applies to multispectral imaging, which is more similar to color images. Thus, many developments proven for color images also work for multispectral imaging. But still, there is a lack of established large-scale data sets for multispectral imaging. And again, there are often task-specific features. Finally, multispectral models still have no pretrained weights, which is a significant drawback compared to color models. We published a small multispectral data set as a starting point. But it is far from the well-established color data sets. Further, by the design of the multispectral cameras, these data sets are always very task-specific, and general multispectral models are, therefore, unlikely.

We want to highlight the idea of a hyperspectral-to-multispectral pipeline for spectral imaging applications. For a practical application use case, a straightforward multispectral camera is favored. But a hyperspectral camera is valuable for the in-depth analysis of the application. Thus, finding the required wavelengths with an upstream hyperspectral analysis is self-evident. Custom camera filters for specific wavelength ranges, which are nowadays obtainable, could be used to build a multispectral sensor for a particular task. This combines the advantages of both techniques with the high spectral resolution for the analysis and a straightforward camera for the application. A first approach, which supports this, was presented in chapter 5.

7.3 Future Work

A trend to more spectral imaging has become visible in recent years. More and more companies are trying to utilize hyperspectral cameras for sorting tasks. And multispectral cameras are more often part of new robotic systems. The emerging trend will make optimized algorithms for this kind of recordings more relevant. Still, the development of algorithms for spectral imaging (hyperspectral or multispectral imaging) lacks progress.

A considerable problem is the absence of more general benchmarks for hyperspectral and multispectral imaging. The data set *DeepHS Fruit v2* is just a first step. The next step should be to merge a couple of sparse hyperspectral data sets into a uniform data set allowing a more reliable comparison of approaches. Part of this has to be a well-defined evaluation metric. In this context, an analysis of the current state-of-the-art approaches could show the key components for an optimal hyperspectral model. This could connect the hyperspectral research community and allow for more reliable future research.

Further, more research regarding the particular structure of hyperspectral recordings could lead to better models. With *HyveConv++* we presented an elementary idea. More research could lead to more sophisticated approaches.

In some preliminary experiments, we could also see that the *HyveConv++* approach could be used to train the same models with hyperspectral and multispectral recordings or multispectral and color recordings. This could result in larger data sets but needs further investigation.

For multispectral imaging, it seems necessary to find a way to provide better pretrained weights for multispectral models. Until now, color models have a considerable advantage as these require only finetuning for a task in contrast to full training of the multispectral models. Multispectral cameras could boost performance in many applications but are still not fully present. It could be helpful to create more awareness for this technology.

Also, the research regarding the pretraining of hyperspectral models, performed with contrastive learning, were just the first steps into this direction. Besides contrastive learning, one of the many other techniques (e.g., teacher-student network approaches) is worth considering. Generating useful pretrained weights for a neural network is a separate research topic, but it could be precious for spectral imaging methods.

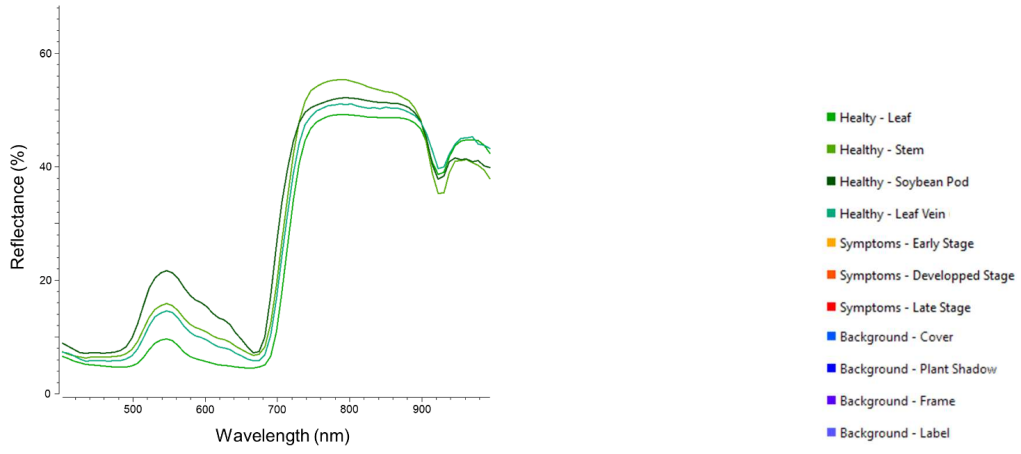
Finally, it would be interesting to validate the idea of the hyperspectral-to-multispectral pipeline on an example application, which would provide valuable insights and shows how spectral imaging could be integrated into many more applications.

Regarding the initial motivation, there are still some further steps for the combination of spectral imaging and computer vision. The current applications need to be more cohesive. The many isolated solutions must be merged to call it a real super-human vision. However, for some specific problems, we can already use spectral imaging to support the decision of human experts.

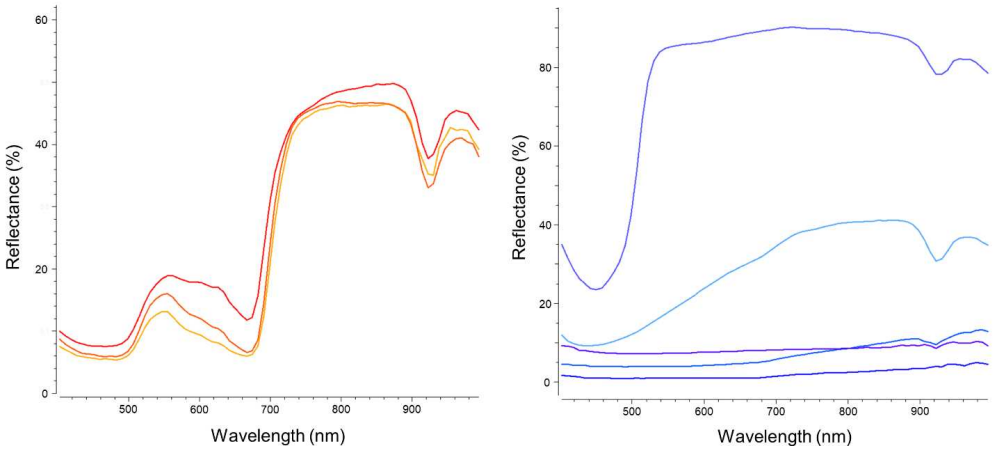
Appendix A

Appendices

A.1 Spectral Signatures of Noticeable Features for Soybean



(a)



(b) (c)

Figure A.1: Spectral signatures of noticeable features within the distinct classes – **(a)** healthy plant tissue, **(b)** disease symptoms and **(c)** background – within the training data.

A.2 Full List of Augmentations Tested for SSL

Table A.1: The list of the evaluated data augmentation techniques indicating spatial (**Spa.**), a spectral (**Spec.**) or a continuous impact on on the hyperspectral cube (**Cont. dist.**).

Augmentation	**Description**	**Spa.**	**Spec.**	**Cont. dist.**
Random Rotate	Random rotation	yes	no	yes
Random Flip	Random flip	yes	no	yes
Random Crop	Use a random crop of the sample	yes	no	-
Random Cut	Drop a random crop of the sample	yes	no	yes
Random Noise	Gaussian noise for all pixels	yes	yes	no
Random Noise Wavelengthbased	Gaussian noise with channel-based standard deviation	yes	yes	no
Random Intensity Scale	Scale the intensities of all channels by a random factor	no	yes	yes
Blur Pixels Rand	Gaussian noise for random pixels	no	no	no
Drop Pixels Rand	Drops random pixels	no	no	no
Drop Channels Cons	Drops random continuous channels	no	yes	yes
Drop Channels Rand	Drops random channels	no	yes	no
Blur Channels Rand	Gaussian noise for random channels	no	yes	no
Rand Occlusion (Haut *et al.*, 2019)	Random 10x10x10 subcube is dropped	yes	yes	yes
Blur Edge Pixels	Gaussian noise to the five edge pixels	yes	no	yes
Drop Edge Pixels	Drops the five edge pixels	yes	no	no
Blur Edge Channels	Gaussian noise for the noise edge channels	no	yes	yes
Drop Edge Channels	Drops the noise edge pixels	no	yes	no
Blur Color Channels	Gaussian noise for the channels in the visible range (400 - 700 nm)	no	yes	yes
Drop Color Channels	Drops the channels in the visible range (400 - 700 nm)	no	yes	yes
Mix Up 2 (Zhang *et al.*, 2018)	Mixes two views of the same sample to the dependency on labels.	yes	yes	yes
Scale Mix (Wang *et al.*, 2022b)	Mixes two scaled views of the same sample.	yes	yes	yes

A.3 HRSS Results with Standard Deviation

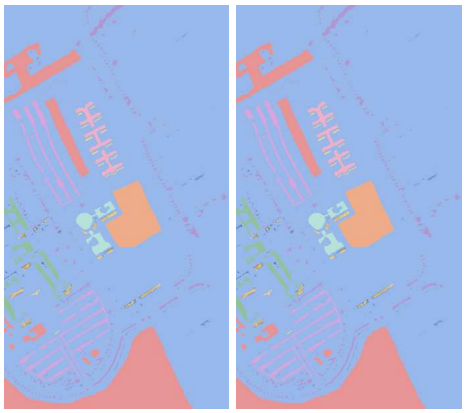
Table A.2: Classification accuracies (%) of different models in terms of *Overall Accuracy (OA)*, *Cohen Kappa (Kappa)*, and *Averaged Classwise Accuracy (AA)* with 10% annotated data as training data. Based on the evaluations of Chakraborty and Trehan (2021). Two configurations of our model are presented. (*) not fine-tuned (**) larger hidden layers. **Bold** numbers indicate the best accuracy for each configuration.

Methods	**# of params**	**Indian Pines**			**Pavia University**			**Salinas**		
		OA	Kappa	AA	OA	Kappa	AA	OA	Kappa	AA
SVM Cortes and Vapnik (1995)		81.67±0.7	78.76±0.8	79.84±3.4	90.58±0.5	87.21±0.7	92.99±0.4	94.46±0.1	93.13±0.3	93.01±0.6
2D-CNN Makantasis *et al.* (2015)	561,300	80.27±1.2	78.26±2.1	68.32±4.1	96.63±0.2	95.53±0.2	94.84±1.4	96.34±0.3	95.93±0.9	94.36±0.5
3D-CNN Ben Hamida *et al.* (2018)	991,596	82.62±0.1	79.25±0.3	76.51±0.1	96.34±0.2	94.90±1.2	97.03±0.6	85.00±0.1	83.20±0.7	89.63±0.2
M3D-CNN He *et al.* (2017)	372,544	81.39±2.6	81.20±2.0	75.22±0.7	95.95±0.6	93.40±0.4	97.52±1.0	94.20±0.8	93.61±0.3	96.66±0.5
FuSENet Roy *et al.* (2020a)	100,880	97.11±0.2	97.25±0.2	97.32±0.2	97.65±0.3	97.69±0.3	97.68±0.4	99.23±0.1	99.97±0.2	99.16±0.1
HybridSN Roy *et al.* (2020b)	5,122,176	98.39±0.1	98.16±0.1	98.01±0.2	**99.72±0.1**	**99.64±0.1**	99.20±0.1	**99.98±0.2**	**99.98±0.2**	**99.98±0.1**
SpectralNET Chakraborty and Trehan (2021)	6,800,336	**98.76±0.2**	98.59±0.1	98.61±0.1	99.71±0.1	99.62±0.1	99.43±0.2	99.96±0.2	99.96±0.1	99.97±0.1
HyveConv++ ours (*)	**16,700**	98.18±0.6	98.41±0.1	98.28±0.1	99.30±0.3	99.30±0.3	**99.49±0.2**	99.24±0.2	99.64±0.0	99.94±0.0
HyveConv++ ours (**)	**25,200**	98.33±0.1	**98.64±0.1**	**98.69±0.1**	99.42±0.2	99.49±0.2	99.46±0.2	99.89±0.0	99.79±0.0	99.74±0.0

Table A.3: Classification accuracies (%) with 30% annotated data as training data. Based on the evaluations of Chakraborty and Trehan (2021). Two configuration of our model are presented. (*) not fine-tuned (**) larger hidden layers. **Bold** indicates the best accuracy per configuration.

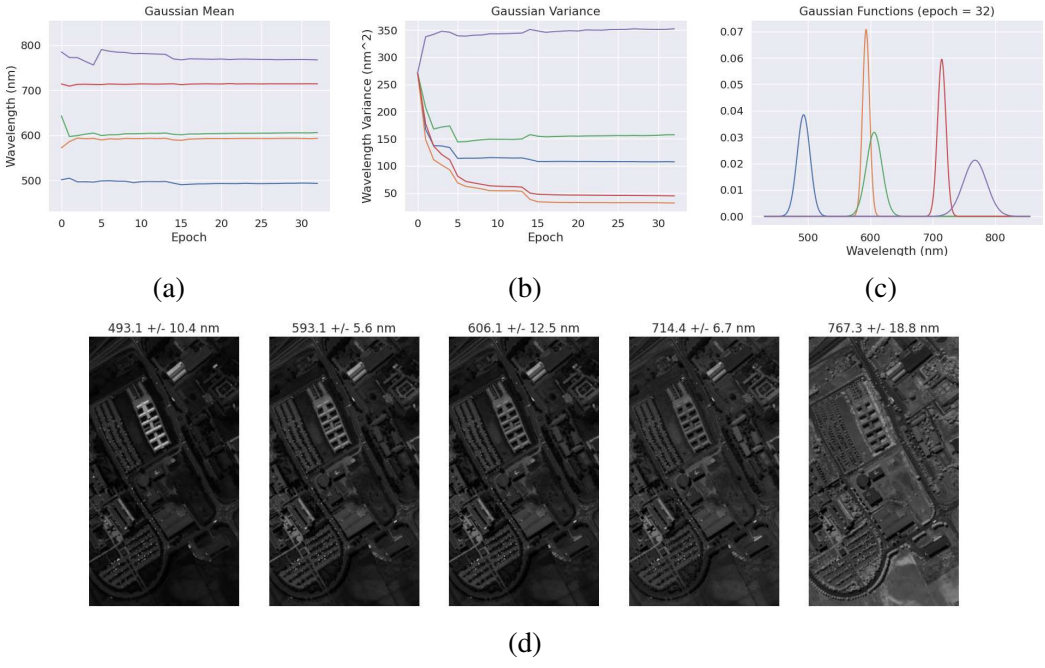
Methods	**# of params**	**Indian Pines**			**Pavia University**			**Salinas**		
		OA	Kappa	AA	OA	Kappa	AA	OA	Kappa	AA
SVM Cortes and Vapnik (1995)		87.24±0.4	85.27±0.5	85.15±1.1	95.65±0.1	94.63±0.2	94.60±0.1	94.95±0.1	94.48±0.1	97.93±0.1
2D-CNN Makantasis *et al.* (2015)	561,300	88.90±1.3	87.01±1.6	85.70±1.0	96.50±0.4	96.55±0.3	96.00±0.1	96.75±0.6	96.71±0.7	98.57±0.2
3D-CNN Ben Hamida *et al.* (2018)	991,596	90.23±0.2	89.70±0.3	89.87±0.1	97.90±0.3	97.22±0.1	97.30±0.1	95.54±0.5	94.81±0.3	97.09±0.6
M3D-CNN He *et al.* (2017)	372,544	95.67±0.1	94.70±0.3	94.60±0.6	97.60±0.2	96.50±0.6	98.00±0.1	94.99±0.3	95.40±0.1	96.28±0.2
FuSENet Roy *et al.* (2020a)	100,880	99.01±0.2	98.60±0.1	98.64±0.1	99.42±0.2	99.21±0.3	99.33±0.2	99.68±0.2	99.74±0.1	99.69±0.1
ImprovedTransformerNet Qing *et al.* (2021)	150,000,000	99.22 %	99.19 %	99.08 %	99.64 %	99.49 %	99.67 %	99.91 %	99.78 %	99.63 %
HybridSN Roy *et al.* (2020b)	5,122,176	99.75±0.1	99.71±0.1	99.63±0.2	99.98±0.1	**99.98±0.2**	99.97±0.2	**100**	**100**	**100**
SpectralNET Chakraborty and Trehan (2021)	6,800,336	**99.86±0.2**	**99.84±0.2**	**99.98±0.1**	**99.99±0.1**	**99.98±0.1**	**99.98±0.1**	**100**	**100**	**100**
HyveConv++ ours (*)	**16,700**	99.85±0.0	99.75±0.0	99.7±0.2	99.97±0.0	99.96±0.0	99.97±0.0	99.98±0.0	99.99±0.0	99.99±0.0
HyveConv++ ours (**)	**25,200**	**99.86±0.0**	**99.84±0.0**	99.57±0.1	99.96±0.0	**99.98±0.0**	99.94±0.0	99.93±0.0	99.92±0.0	99.99±0.0

A.4 Additional Examples of a learned Camera Filters with HyveConv



(a) Prediction (b) Ground Truth

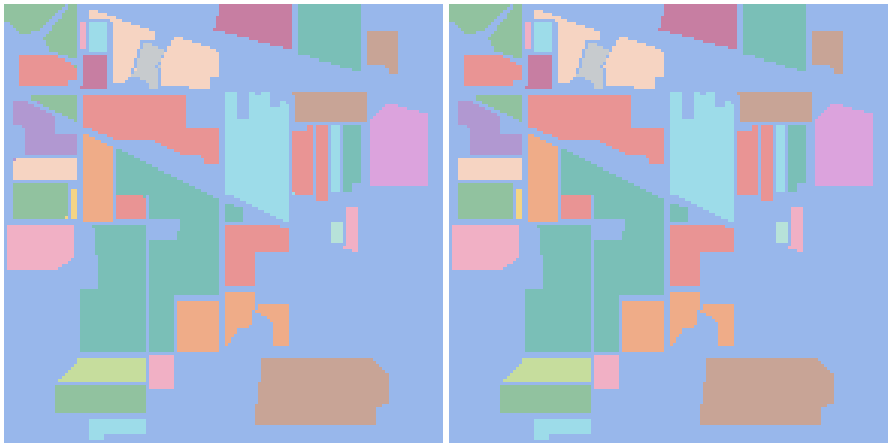
Figure A.2: Prediction of DeepHS_net + HyveConv++ and ground truth for the segmentations mask of the University of Pavia data set.



(a) (b) (c)

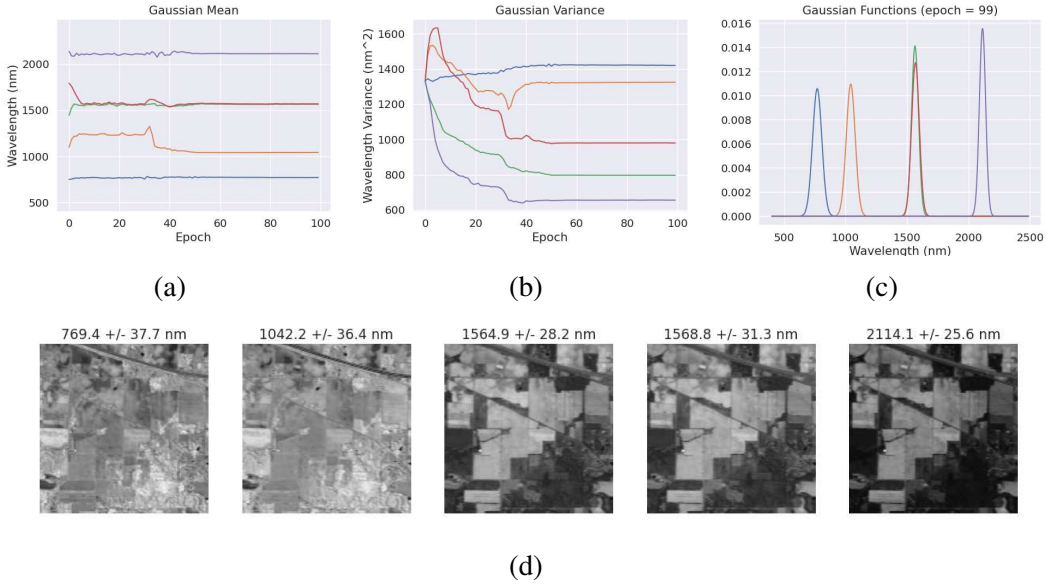
(d)

Figure A.3: Training of the Gaussian distributions for the Pavia University data set. (a) and (b) show the development of the mean and variance over training epochs. (c) shows the final Gaussian distributions, and these are applied as filters in (d).



(a) Prediction (b) Ground Truth

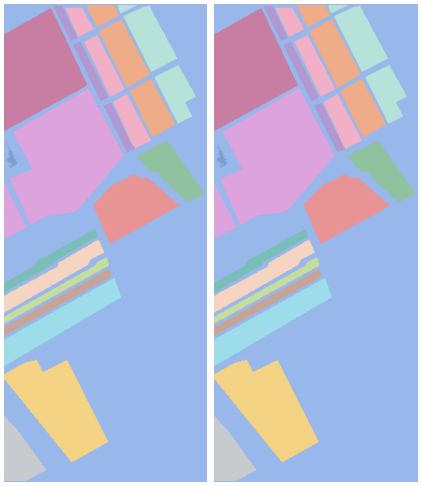
Figure A.4: Prediction of DeepHS_net + HyveConv++ and ground truth for the segmentations mask of the Indian pines data set.



(a) (b) (c)

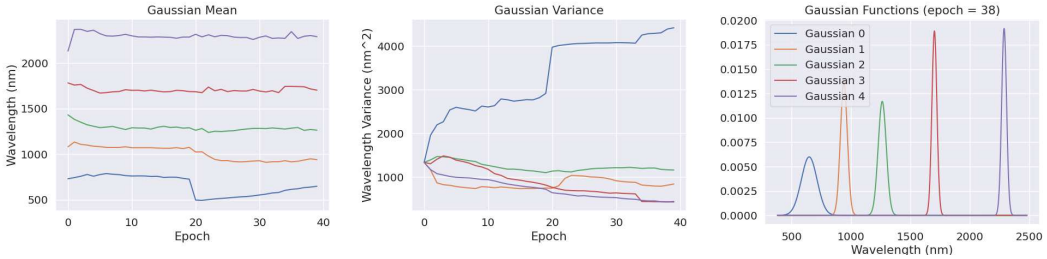
(d)

Figure A.5: Training of the Gaussian distributions for the Indian pines data set. (a) and (b) show the development of the mean and variance over training epochs. (c) shows the final Gaussian distributions, and these are applied as filters in (d).

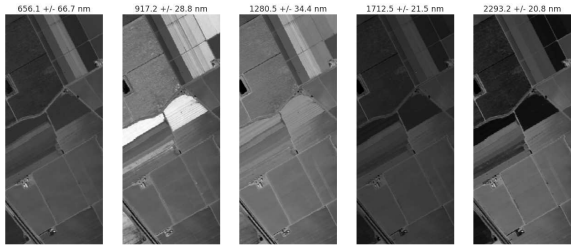


(a) Prediction (b) Ground Truth

Figure A.6: Prediction of DeepHS_net + HyveConv++ and ground truth for the segmentations mask of the Salinas data set.



(a) (b) (c)



(d)

Figure A.7: Training of the Gaussian distributions for the Salinas data set. (a) and (b) show the development of the mean and variance over training epochs. (c) shows the final Gaussian distributions, and these are applied as filters in (d).

Abbreviations

AA	Averaged Classwise Accuracy, Page 81
ARS	Asian Soybean Rust, Page 35
BBCH	Biologische Bundesanstalt für Land- und Forstwirtschaft, Bundessortenamt und CHemische Industrie, Page 36
CNN	Convolutional Neural Network, Page 5
DaI	Day after Inoculation, Page 37
DLRG	Deutsche Lebens-Rettungs-Gesellschaft e.V., Page 95
DSCNV	Depth-Wise Separable Convolution, Page 30
FPS	Frames per Second, Page 11
GNSS	Global Navigation Satellite System, Page 92
GPU	Graphical Processing Unit, Page 108
HRSS	Hyperspectral Remote Sensing Scenes, Page 20
HSI	Hyperspectral Imaging, Page 5
IP	Indian Pines, Page 81
IQR	Interquartile range, Page 54
k-NN	k-Nearest-Neighbor, Page 20
Kappa	Cohen Kappa, Page 81
MAV	Micro Aerial Vehicle, Page 96
MLP	Multi Layer Perceptron, Page 50
MSI	multispectral imaging, Page 5
NIR	Near-Infrared, Page 8

NUV	Near-Ultraviolet, Page 8
OA	Overall Accuracy, Page 81
PCA	Principal Component Analysis, Page 29
SA	Salinas, Page 81
SAR	Search and Rescue, Page 6
SE	Squeeze-and-Excitation, Page 75
SI	Spectral Imaging, Page 5
SSC	Soluble Solids Content, Page 25
SSL	Self-Supervised Learning, Page 5
SVM	Support Vector Machine, Page 20, 21
UAV	Unmanned Aerial Vehicle, Page 13
UP	Pavia University, Page 81
WROI	Wavelength Ranges of Interest, Page 113

Bibliography

Adão, T., Hruška, J., Pádua, L., Bessa, J., Peres, E., Morais, R., and Sousa, J. J. (2017). Hyperspectral imaging: A review on uav-based sensors, data processing and applications for agriculture and forestry. *Remote Sensing*, **9**(11), 1110.

Albanese, A., Sciancalepore, V., and Costa-Pérez, X. (2020). Sardo: An automated search-and-rescue drone-based solution for victims localization. *arXiv preprint arXiv:2003.05819*.

Alexander, L. and Grierson, D. (2002). Ethylene biosynthesis and action in tomato: A model for climacteric fruit ripening. *Journal of Experimental Botany*, **53**(377), 2039–2055.

Alisaac, E., Behmann, J., Kuska, M. T., Dehne, H.-W., and Mahlein, A.-K. (2018). Hyperspectral quantification of wheat resistance to fusarium head blight: Comparison of two fusarium species. *European Journal of Plant Pathology*, **152**, 869–884.

Bardes, A., Ponce, J., and LeCun, Y. (2022). Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Barro, J. P., Alves, K. S., Godoy, C. V., Dias, A. R., Forcelini, C. A., Utiamada, C. M., de Andrade Júnior, E. R., Juliatti, F. C., Grigolli, F. J., Feksa, H. R., *et al.* (2021). Performance of dual and triple fungicide premixes for managing soybean rust across years and regions in brazil: A meta-analysis. *Plant Pathology*, **70**(8), 1920–1935.

Baumgardner, M. F., Biehl, L. L., and Landgrebe, D. A. (2015). 220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3.

Bauriegel, E. and Herppich, W. B. (2014). Hyperspectral and chlorophyll fluorescence imaging for early detection of plant diseases, with special reference to fusarium spec. infections on wheat. *Agriculture*, **4**(1), 32–57.

Ben Hamida, A., Benoit, A., Lambert, P., and Ben Amar, C. (2018). 3-d deep learning approach for remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, **56**(8), 4420–4434.

Blasinski, H., Farrell, J. E., Lian, T., Liu, Z., and Wandell, B. A. (2018). Optimizing image acquisition systems for autonomous driving. *electronic imaging*, **2018**, 161–1–161–7.

Bochkovskiy, A., Wang, C., and Liao, H. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *CoRR*, **abs/2004.10934**.

Bohnenkamp, D., Kuska, M., Mahlein, A.-K., and Behmann, J. (2019). Hyperspectral signal decomposition and symptom detection of wheat rust disease at the leaf scale using pure fungal spore spectra as reference. *Plant Pathology*, **68**(6), 1188–1195.

Bolya, D., Foley, S., Hays, J., and Hoffman, J. (2020). TIDE: A general toolbox for identifying object detection errors. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part III*, volume 12348 of *Lecture Notes in Computer Science*, pages 558–573. Springer.

Bromley, J., Bentz, J. W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., Säckinger, E., and Shah, R. (1993). Signature verification using a "siamese" time delay neural network. *Int. J. Pattern Recognit. Artif. Intell.*, **7**(4), 669–688.

Buckler, M., Jayasuriya, S., and Sampson, A. (2017). Reconfiguring the imaging pipeline for computer vision. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 975–984. IEEE Computer Society.

Cai, Y., Yao, Z., Dong, Z., Gholami, A., Mahoney, M. W., and Keutzer, K. (2020). Zeroq: A novel zero shot quantization framework. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 13166–13175. Computer Vision Foundation / IEEE.

Candiago, S., Remondino, F., De Giglio, M., Dubbini, M., and Gattelli, M. (2015). Evaluating multispectral images and vegetation indices for precision farming applications from uav images. *Remote sensing*, **7**(4), 4026–4047.

Carlson, A., Skinner, K. A., Vasudevan, R., and Johnson-Roberson, M. (2018). Modeling camera effects to improve visual learning from synthetic data. In L. Leal-Taixé and S. Roth, editors, *Computer Vision - ECCV 2018 Workshops - Munich, Germany, September 8-14, 2018, Proceedings, Part I*, volume 11129 of *Lecture Notes in Computer Science*, pages 505–520. Springer.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Chakraborty, T. and Trehan, U. (2021). Spectralnet: Exploring spatial-spectral waveletcnn for hyperspectral image classification. *arXiv preprint arXiv:2104.00341*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. (2020). A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Chen, X. and He, K. (2021). Exploring simple siamese representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 15750–15758. Computer Vision Foundation / IEEE.

Chen, Y., Lin, Z., Zhao, X., Wang, G., and Gu, Y. (2014). Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, **7**(6), 2094–2107.

Childs, S. P., Buck, J. W., and Li, Z. (2018). Breeding soybeans with resistance to soybean rust (phakopsora pachyrhizi). *Plant Breeding*, **137**(3), 250–261.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1800–1807. IEEE Computer Society.

Corbane, C., Najman, L., Pecoul, E., Demagistri, L., and Petit, M. (2010). A complete processing chain for ship detection using optical satellite imagery. *International Journal of Remote Sensing*, **31**(22), 5837–5854.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, **20**(3), 273–297.

Crisp, D. (2004). The state-of-the-art in ship detection in synthetic aperture radar imagery. defence science and technology organization (dsto). *Information Science Laboratory, Research Report No. DSTO-RR-0272*.

Cristianini, N. and Shawe-Taylor, J. (2010). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Cucchiara, R., Grana, C., Piccardi, M., Prati, A., and Sirotti, S. (2001). Improving shadow suppression in moving object detection with hsv color information. In *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585)*, pages 334–339. IEEE.

Curcio, J. A. and Petty, C. C. (1951). The near infrared absorption spectrum of liquid water. *J. Opt. Soc. Am.*, **41**(5), 302–304.

Davis, P. (1975). *Interpolation and Approximation*. Dover Books on Mathematics. Dover Publications.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Deng, L., Mao, Z., Li, X., Hu, Z., Duan, F., and Yan, Y. (2018). Uav-based multispectral remote sensing for precision agriculture: A comparison between different cameras. *ISPRS journal of photogrammetry and remote sensing*, **146**, 124–136.

Ding, J., Xue, N., Xia, G., Bai, X., Yang, W., Yang, M. Y., Belongie, S. J., Luo, J., Datcu, M., Pelillo, M., and Zhang, L. (2022). Object detection in aerial images: A large-scale benchmark and challenges. *IEEE Trans. Pattern Anal. Mach. Intell.*, **44**(11), 7778–7796.

Du, D., Qi, Y., Yu, H., Yang, Y., Duan, K., Li, G., Zhang, W., Huang, Q., and Tian, Q. (2018). The unmanned aerial vehicle benchmark: Object detection and tracking.

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., and Tian, Q. (2019). CenterNet: Keypoint triplets for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2019-Octob, pages 6568–6577. Institute of Electrical and Electronics Engineers Inc.

Elite Optoelectronics Co. (2020). What is the wavelength of the laser? `https://www.s-laser.com/info/what-is-the-wavelength-of-the-laser-50866750.html`; Published: 19.10.2020; Accessed: 04.04.2023.

Ermolov, A., Siarohin, A., Sangineto, E., and Sebe, N. (2021). Whitening for self-supervised representation learning. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 3015–3024. PMLR.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, **88**(2), 303–338.

Fix, E. and Hodges, J. L. (1989). Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. *International Statistical Review / Revue Internationale de Statistique*, **57**(3), 238.

Gallego, A.-J., Pertusa, A., Gil, P., and Fisher, R. B. (2019). Detection of bodies in maritime rescue operations using unmanned aerial vehicles with multispectral cameras. *Journal of Field Robotics*, **36**(4), 782–796.

Gao, Z., Shao, Y., Xuan, G., Wang, Y., Liu, Y., and Han, X. (2020). Real-time hyperspectral imaging for the in-field estimation of strawberry ripeness with deep learning. *Artificial Intelligence in Agriculture*, **4**, 31–38.

Geraldes, R., Goncalves, A., Lai, T., Villerabel, M., Deng, W., Salta, A., Nakayama, K., Matsuo, Y., and Prendinger, H. (2019). Uav-based situational awareness system using deep learning. *IEEE Access*, **7**, 122583–122594.

Ghazali, S. N. A. M., Anuar, H. A., Zakaria, S. N. A. S., and Yusoff, Z. (2016). Determining position of target subjects in maritime search and rescue (msar) operations using rotary wing unmanned aerial vehicles (uavs). In *2016 International Conference on Information and Communication Technology (ICICTM)*, pages 1–4. IEEE.

Goellner, K., Loehrer, M., Langenbach, C., Conrath, U., Koch, E., and Schaffrath, U. (2010). Phakopsora pachyrhizi, the causal agent of asian soybean rust. *Molecular plant pathology*, **11**(2), 169–177.

Graña, M and Veganzons, MA and Ayerdi, B (2014). Hyperspectral Remote Sensing Scenes.

Grill, J., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. Á., Guo, Z., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., and Valko, M. (2020). Bootstrap your own latent - A new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*.

Guan, D., Cao, Y., Yang, J., Cao, Y., and Yang, M. Y. (2019). Fusion of multispectral data through illumination-aware deep neural networks for pedestrian detection. *Inf. Fusion*, **50**, 148–157.

Guo, J., Li, Y., Lin, W., Chen, Y., and Li, J. (2018a). Network decoupling: From regular to depthwise separable convolutions. In *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*, page 248. BMVA Press.

Guo, Y., Han, S., Li, Y., Zhang, C., and Bai, Y. (2018b). K-nearest neighbor combined with guided filter for hyperspectral image classification. *Procedia Computer Science*, **129**, 159–165.

Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742.

Haut, J. M., Paoletti, M. E., Plaza, J., Plaza, A., and Li, J. (2019). Hyperspectral image classification using random occlusion data augmentation. *IEEE Geosci. Remote. Sens. Lett.*, **16**(11), 1751–1755.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 Inter, pages 1026–1034.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. B. (2020). Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9726–9735. Computer Vision Foundation / IEEE.

He, M., Li, B., and Chen, H. (2017). Multi-scale 3d deep convolutional neural network for hyperspectral image classification. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3904–3908.

Hendrycks, D., Lee, K., and Mazeika, M. (2019). Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pages 2712–2721. PMLR.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, **313**(5786), 504–507.

Hong, D., Han, Z., Yao, J., Gao, L., Zhang, B., Plaza, A., and Chanussot, J. (2022). Spectralformer: Rethinking hyperspectral image classification with transformers. *IEEE Trans. Geosci. Remote. Sens.*, **60**, 1–15.

Hong, S., Kang, S., and Cho, D. (2019). Patch-level augmentation for object detection in aerial images. In *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019*, pages 127–134.

Hou, S., Shi, H., Cao, X., Zhang, X., and Jiao, L. (2022). Hyperspectral imagery classification based on contrastive learning. *IEEE Transactions on Geoscience and Remote Sensing*, **60**, 1–13.

Howard, A. G. (2014). Some improvements on deep convolutional neural network based image classification. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Hsieh, M.-R., Lin, Y.-L., and Hsu, W. H. (2017). Drone-based object counting by spatially regularized regional proposal network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4145–4153.

Hu, J., Shen, L., Albanie, S., Sun, G., and Wu, E. (2020a). Squeeze-and-excitation networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, **42**(8), 2011–2023.

Hu, X., Zhong, Y., Luo, C., and Wang, X. (2020b). Whu-hi: Uav-borne hyperspectral with high spatial resolution (H2) benchmark datasets for hyperspectral image classification. *CoRR*, **abs/2012.13920**.

Huang, L., Ding, W., Liu, W., Zhao, J., Huang, W., Xu, C., Zhang, D., and Liang, D. (2019). Identification of wheat powdery mildew using in-situ hyperspectral data and linear regression and support vector machines. *Journal of Plant Pathology*, **101**, 1035–1045.

Hunt, R. (2005). *The Reproduction of Colour*. The Wiley-IS&T Series in Imaging Science and Technology. Wiley.

Imamoglu, N., Oishi, Y., Zhang, X., Ding, G., Fang, Y., Kouyama, T., and Nakamura, R. (2018). Hyperspectral image dataset for benchmarking on salient object detection. In *Tenth International Conference on Quality of Multimedia Experience, QoMEX 2018, Cagliari, Italy, May 29 - June 1, 2018*, pages 1–3. IEEE.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning, ICML 2015*, volume 1, pages 448–456. International Machine Learning Society (IMLS).

Karaca, Y., Cicek, M., Tatli, O., Sahin, A., Pasli, S., Beser, M. F., and Turedi, S. (2018). The potential use of unmanned aircraft systems (drones) in mountain search and rescue operations. *The American journal of emergency medicine*, **36**(4), 583–588.

Karasawa, T., Watanabe, K., Ha, Q., Tejero-de-Pablos, A., Ushiku, Y., and Harada, T. (2017). Multispectral object detection for autonomous vehicles. In W. Wu, J. Yang, Q. Tian, and R. Zimmermann, editors, *Proceedings of the on Thematic Workshops of ACM Multimedia 2017, Mountain View, CA, USA, October 23 - 27, 2017*, pages 35–43. ACM.

Khan, I. H., Liu, H., Li, W., Cao, A., Wang, X., Liu, H., Cheng, T., Tian, Y., Zhu, Y., Cao, W., *et al.* (2021). Early detection of powdery mildew disease and accurate quantification of its severity using hyperspectral images in wheat. *Remote Sensing*, **13**(18), 3612.

Kiefer, J. and Wolfowitz, J. (1952). Stochastic Estimation of the Maximum of a Regression Function. *The Annals of Mathematical Statistics*, **23**(3), 462–466.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Koch, E., Ebrahim-Nesbat, F., and Hoppe, H. H. (1983). Light and electron microscopic studies on the development of soybean rust (phakopsora pachyrhizi syd.) in susceptible soybean leaves. *Journal of Phytopathology*, **106**(4), 302–320.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, **60**(6), 84–90.

Kuo, B.-C., Ho, H.-H., Li, C.-H., Hung, C.-C., and Taur, J.-S. (2013). A kernel-based feature selection method for svm with rbf kernel for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, **7**(1), 317–326.

Lewis, C. E. (1978). The maturity of avocados—a general review. *Journal of the Science of Food and Agriculture*, **29**(10), 857–866.

Li, Q., Mou, L., Liu, Q., Wang, Y., and Zhu, X. X. (2018). Hsf-net: Multiscale deep feature embedding for ship detection in optical remote sensing imagery. *IEEE Transactions on Geoscience and Remote Sensing*, **56**(12), 7147–7161.

Li, R., Wang, Y., Liang, F., Qin, H., Yan, J., and Fan, R. (2019a). Fully quantized network for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2810–2819. Computer Vision Foundation / IEEE.

Li, S. and Yeung, D.-Y. (2017). Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Li, S., Song, W., Fang, L., Chen, Y., Ghamisi, P., and Benediktsson, J. A. (2019b). Deep learning for hyperspectral image classification: An overview. *IEEE Transactions on Geoscience and Remote Sensing*, **57**(9), 6690–6709.

Li, W., Wu, G., Zhang, F., and Du, Q. (2017). Hyperspectral image classification using deep pixel-pair features. *IEEE Transactions on Geoscience and Remote Sensing*, **55**(2), 844–853.

Lin, M., Chen, Q., and Yan, S. (2014a). Network in network. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Lin, T., Maire, M., Belongie, S. J., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014b). Microsoft COCO: common objects in context. In D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer.

Lin, T. Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection.

Liu, Z., Lian, T., Farrell, J. E., and Wandell, B. A. (2019). Soft prototyping camera designs for car detection based on a convolutional neural network. In *2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27-28, 2019*, pages 2383–2392. IEEE.

Liu, Z., Lian, T., Farrell, J. E., and Wandell, B. A. (2020). Neural network generalization: The impact of camera parameters. *IEEE Access*, **8**, 10443–10454.

Loshchilov, I. and Hutter, F. (2017). SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Lowe, A., Harrison, N., and French, A. P. (2017). Hyperspectral image analysis techniques for the detection and classification of the early onset of plant disease and stress. *Plant methods*, **13**(1), 80.

Lu, G. and Fei, B. (2014). Medical hyperspectral imaging: a review. *Journal of Biomedical Optics*, **19**(1), 010901.

Luo, L., Xiong, Y., Liu, Y., and Sun, X. (2019). Adaptive gradient methods with dynamic bound of learning rate. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Lvsouras, E. and Gasteratos, A. (2020). A new method to combine detection and tracking algorithms for fast and accurate human localization in uav-based sar operations. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1688–1696. IEEE.

Lygouras, E., Santavas, N., Taitzoglou, A., Tarchanidis, K., Mitropoulos, A., and Gasteratos, A. (2019). Unsupervised human detection with an embedded vision system on a fully autonomous uav for search and rescue operations. *Sensors*, **19**(16), 3542.

Mahlein, A.-K. (2016). Plant disease detection by imaging sensors–parallels and specific demands for precision agriculture and plant phenotyping. *Plant disease*, **100**(2), 241–251.

Mahlein, A.-K., Kuska, M. T., Thomas, S., Wahabzada, M., Behmann, J., Rascher, U., and Kersting, K. (2019). Quantitative and qualitative phenotyping of disease resistance of crops by hyperspectral sensors: seamless interlocking of phytopathology, sensors, and machine learning is needed! *Current opinion in plant biology*, **50**, 156–162.

Makantasis, K., Karantzalos, K., Doulamis, A., and Doulamis, N. (2015). Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, volume 2015-Novem, pages 4959–4962. Institute of Electrical and Electronics Engineers Inc.

Martinsen, P. and Schaare, P. (1998). Measuring soluble solids distribution in kiwifruit using near-infrared imaging spectroscopy. *Postharvest Biology and Technology*, **14**(3), 271–281.

Matsugu, M., Mori, K., Mitari, Y., and Kaneda, Y. (2003). Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks*, **16**(5-6), 555–559.

Melado-Herreros, A., Nieto-Ortega, S., Olabarrieta, I., Gutiérrez, M., Villar, A., Zufía, J., Gorretta, N., and Roger, J.-M. (2021). Postharvest ripeness assessment of 'hass' avocado based on development of a new ripening index and vis-nir spectroscopy. *Postharvest Biology and Technology*, **181**, 111683.

Mishra, B., Garg, D., Narang, P., and Mishra, V. (2020). Drone-surveillance for search and rescue in natural disaster. *Computer Communications*, **156**, 1–10.

Mitsui, K., Inagaki, T., and Tsuchikawa, S. (2008). Monitoring of hydroxyl groups in wood during heat treatment using NIR spectroscopy. *Biomacromolecules*, **9**(1), 286–288.

Mollazade, K., Omid, M., Tab, F. A., Mohtasebi, S., and Sasse-Zude, M. (2012). Spatial mapping of moisture content in tomato fruits using hyperspectral imaging and artificial neural networks. In *International workshop on Computer Image Analysis in Agriculture*.

Mueller, M., Smith, N., and Ghanem, B. (2016). A benchmark and simulator for uav tracking. In *European conference on computer vision*, pages 445–461. Springer.

Mundhenk, T. N., Konjevod, G., Sakla, W. A., and Boakye, K. (2016). A large contextual dataset for classification, detection and counting of cars with deep learning. In *European Conference on Computer Vision*, pages 785–800. Springer.

Nagasubramanian, K., Jones, S., Sarkar, S., Singh, A. K., Singh, A., and Ganapathysubramanian, B. (2018). Hyperspectral band selection using genetic algorithm and support vector machines for early identification of charcoal rot disease in soybean stems. *Plant methods*, **14**, 1–13.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Nasr, I., Chekir, M., and Besbes, H. (2019). Shipwrecked victims localization and tracking using uavs. In *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pages 1344–1348. IEEE.

Ofli, F., Meier, P., Imran, M., Castillo, C., Tuia, D., Rey, N., Briant, J., Millet, P., Reinhard, F., Parkan, M., *et al.* (2016). Combining human computing and machine learning to make sense of big (aerial) data for disaster response. *Big data*, **4**(1), 47–59.

Olarewaju, O. O., Bertling, I., and Magwaza, L. S. (2016). Non-destructive evaluation of avocado fruit maturity using near infrared spectroscopy and PLS regression models. *Scientia Horticulturae*, **199**, 229–236.

Ophoff, T., Beeck, K. V., and Goedemé, T. (2019). Exploring rgb+depth fusion for real-time object detection. *Sensors*, **19**(4), 866.

Pailla, D. R., Kollerathu, V. A., and Chennamsetty, S. S. (2019). Object detection on aerial imagery using centernet. *CoRR*, **abs/1908.08244**.

Park, B. and Lu, R. (2015). *Hyperspectral Imaging Technology in Food and Agriculture*. Food Engineering Series. Springer New York.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, **2**(11), 559–572.

Pei, Z., Qi, X., Zhang, Y., Ma, M., and Yang, Y.-H. (2019). Human trajectory prediction in crowded scene using social-affinity long short-term memory. *Pattern Recognition*, **93**, 273–282.

Pinto, J., Rueda-Chacón, H., and Arguello, H. (2019). Classification of Hass avocado (persea americana mill) in terms of its ripening via hyperspectral images. *TecnoLógicas*, **22**(45), 109–128.

Prasad, D. K., Dong, H., Rajan, D., and Quek, C. (2019). Are object detection assessment criteria ready for maritime computer vision? *IEEE Transactions on Intelligent Transportation Systems*, **21**(12), 5295–5304.

Prechelt, L. (1998). Automatic early stopping using cross validation: Quantifying the criteria. *Neural Networks*, **11**(4), 761–767.

Qing, Y., Liu, W., Feng, L., and Gao, W. (2021). Improved transformer net for hyperspectral image classification. *Remote. Sens.*, **13**(11), 2216.

Queralta, J. P., Raitoharju, J., Gia, T. N., Passalis, N., and Westerlund, T. (2020). Autosos: Towards multi-uav systems supporting maritime search and rescue with lightweight ai and edge computing. *arXiv preprint arXiv:2005.03409*.

Ren, S., He, K., Girshick, R. B., and Sun, J. (2017). Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, **39**(6), 1137–1149.

Roberts, W., Griendling, K., Gray, A., and Mavris, D. (2016). Unmanned vehicle collaboration research environment for maritime search and rescue. In *30th Congress of the International Council of the Aeronautical Sciences*. International Council of the Aeronautical Sciences (ICAS) Bonn, Germany.

Robles-Kelly, A. and Huynh, C. P. (2013). *Imaging Spectroscopy for Scene Analysis*. Advances in Computer Vision and Pattern Recognition. Springer.

Roitsch, T., Cabrera-Bosquet, L., Fournier, A., Ghamkhar, K., Jiménez-Berni, J., Pinto, F., and Ober, E. S. (2019). Review: New sensors and data-driven approaches—a path to next generation phenomics. *Plant Science*, **282**, 2–10. The 4th International Plant Phenotyping Symposium.

Roy, S. K., Dubey, S. R., Chatterjee, S., and Chaudhuri, B. B. (2020a). Fusenet: fused squeeze-and-excitation network for spectral-spatial hyperspectral image classification. *IET Image Process.*, **14**(8), 1653–1661.

Roy, S. K., Krishna, G., Dubey, S. R., and Chaudhuri, B. B. (2020b). Hybridsn: Exploring 3-d-2-d CNN feature hierarchy for hyperspectral image classification. *IEEE Geosci. Remote. Sens. Lett.*, **17**(2), 277–281.

Rumpf, T., Mahlein, A.-K., Steiner, U., Oerke, E.-C., Dehne, H.-W., and Plümer, L. (2010). Early detection and classification of plant diseases with support vector machines based on hyperspectral reflectance. *Computers and electronics in agriculture*, **74**(1), 91–99.

Saad, K. and Schneider, S. (2019). Camera vignetting model and its effects on deep neural networks for object detection. In *2019 IEEE International Conference on Connected Vehicles and Expo, ICCVE 2019, Graz, Austria, November 4-8, 2019*, pages 1–5. IEEE.

San, K. T., Mun, S. J., Choe, Y. H., and Chang, Y. S. (2018). Uav delivery monitoring system. In *MATEC Web of Conferences*, volume 151, page 04011. EDP Sciences.

Secci, F. and Ceccarelli, A. (2020). On failures of RGB cameras and their effects in autonomous driving applications. In M. Vieira, H. Madeira, N. Antunes, and Z. Zheng, editors, *31st IEEE International Symposium on Software Reliability Engineering, ISSRE 2020, Coimbra, Portugal, October 12-15, 2020*, pages 13–24. IEEE.

Shridhar, K., Laumann, F., and Liwicki, M. (2019). A comprehensive guide to bayesian convolutional neural network with variational inference. *CoRR*, **abs/1901.02731**.

Shuhua, L. and Gaizhi, G. (2010). The application of improved hsv color space model in image processing. In *2010 2nd International Conference on Future Computer and Communication*, volume 2, pages V2–10–V2–13.

Song, W., Li, S., Fang, L., and Lu, T. (2018). Hyperspectral image classification with deep feature fusion network. *IEEE Transactions on Geoscience and Remote Sensing*, **56**(6), 3173–3184.

Steffensen, J. (2013). *Interpolation: Second Edition*. Dover Books on Mathematics. Dover Publications.

Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic attribution for deep networks. In *34th International Conference on Machine Learning, ICML 2017*, volume 7, pages 5109–5118. International Machine Learning Society (IMLS).

Takahashi, R., Matsubara, T., and Uehara, K. (2020). Data Augmentation Using Random Image Cropping and Patching for Deep CNNs. *IEEE Transactions on Circuits and Systems for Video Technology*, **30**(9), 2917–2931.

Tan, M., Pang, R., and Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10778–10787. Computer Vision Foundation / IEEE.

Tatzer, P., Wolf, M., and Panner, T. (2005). Industrial application for inline material sorting using hyperspectral imaging in the nir range. *Real-Time Imaging*, **11**(2), 99–107. Spectral Imaging II.

Thomas, S., Kuska, M. T., Bohnenkamp, D., Brugger, A., Alisaac, E., Wahabzada, M., Behmann, J., and Mahlein, A.-K. (2018a). Benefits of hyperspectral imaging for plant disease detection and plant protection: a technical perspective. *Journal of Plant Diseases and Protection*, **125**, 5–20.

Thomas, S., Behmann, J., Steier, A., Kraska, T., Muller, O., Rascher, U., and Mahlein, A.-K. (2018b). Quantitative assessment of disease severity and rating of barley cultivars based on hyperspectral imaging in a non-invasive, automated phenotyping platform. *Plant methods*, **14**(1), 1–12.

Thomas, S., Behmann, J., Rascher, U., and Mahlein, A.-K. (2022). Evaluation of the benefits of combined reflection and transmission hyperspectral imaging data through disease detection and quantification in plant–pathogen interactions. *Journal of Plant Diseases and Protection*, **129**(3), 505–520.

Thomas, S., Varga, L. A., Harter, N., Zell, A., and Voegele, R. T. (submitted). Detection of phakopsora pachyrhizi infestation in soybean via hyperspectral imaging and data analysis. *Nature Scientific reports*.

Toivonen, P. M. and Brummell, D. A. (2008). Biochemical bases of appearance and texture changes in fresh-cut fruit and vegetables.

Unel, F. O., Ozkalayci, B. O., and Cigla, C. (2019). The power of tiling for small object detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, volume 2019-June, pages 582–591.

United States Department of Agriculture, F. A. S. (2022). Oilseeds: World market and trade.

Vakili, H., Kolakovic, R., Genina, N., Marmion, M., Salo, H., Ihalainen, P., Peltonen, J., and Sandler, N. (2015). Hyperspectral imaging in quality control of inkjet printed personalised dosage forms. *International Journal of Pharmaceutics*, **483**(1), 244–249.

van Gemert, J. C., Verschoor, C. R., Mettes, P., Epema, K., Koh, L. P., and Wich, S. (2014). Nature conservation drones for automatic localization and counting of animals. In *European Conference on Computer Vision*, pages 255–270. Springer.

Vandersteegen, M., Beeck, K. V., and Goedemé, T. (2018). Real-time multispectral pedestrian detection with a single-pass deep neural network. In A. Campilho, F. Karray, and B. M. ter Haar Romeny, editors, *Image Analysis and Recognition - 15th International Conference, ICIAR 2018, Póvoa de Varzim, Portugal, June 27-29, 2018, Proceedings*, volume 10882 of *Lecture Notes in Computer Science*, pages 419–426. Springer.

Varga, L. A. and Zell, A. (2021). Tackling the background bias in sparse object detection via cropped windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 2768–2777.

Varga, L. A., Makowski, J., and Zell, A. (2021). Measuring the ripeness of fruit with hyperspectral imaging and deep learning. In *International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021*, pages 1–8. IEEE.

Varga, L. A., Koch, S., and Zell, A. (2022a). Comprehensive analysis of the object detection pipeline on UAVs. *Remote Sensing*, **14**(21).

Varga, L. A., Kiefer, B., Messmer, M., and Zell, A. (2022b). SeaDronesSee: A maritime benchmark for detecting humans in open water. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pages 3686–3696. IEEE.

Varga, L. A., Frank, H., and Zell, A. (2023a). Self-supervised pretraining for hyperspectral classification of fruit ripeness. In J. Beyerer, T. Längle, and M. Heizmann, editors, *OCM 2023 - Optical Characterization of Materials : Conference Proceedings*, pages 97–108.

Varga, L. A., Messmer, M., Benbarka, N., and Zell, A. (2023b). Wavelength-aware 2d convolutions for hyperspectral imaging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3788–3797.

Voegele, R. T. (2006). Uromyces fabae: development, metabolism, and interactions with its host Vicia faba. *FEMS Microbiology Letters*, **259**(2), 165–173.

Wang, D., Vinson, R., Holmes, M., Seibel, G., Bechar, A., Nof, S., and Tao, Y. (2019). Early detection of tomato spotted wilt virus by hyperspectral imaging and outlier removal auxiliary classifier generative adversarial nets (or-ac-gan). *Scientific reports*, **9**(1), 1–14.

Wang, Q., Fernandes, S., Williams, G. O. S., Finlayson, N., Akram, A. R., Dhaliwal, K., Hopgood, J. R., and Vallejo, M. (2022a). Deep learning-assisted co-registration of full-spectral autofluorescence lifetime microscopic images with h&e-stained histology images. *CoRR*, **abs/2202.07755**.

Wang, X., Fan, H., Tian, Y., Kihara, D., and Chen, X. (2022b). On the importance of asymmetry for siamese representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 16549–16558. IEEE.

Wellburn, A. R. (1994). The spectral determination of chlorophylls a and b, as well as total carotenoids, using various solvents with spectrophotometers of different resolution. *Journal of Plant Physiology*, **144**(3), 307–313.

Wu, Z., Xiong, Y., Yu, S. X., and Lin, D. (2018). Unsupervised feature learning via non-parametric instance discrimination. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 3733–3742. Computer Vision Foundation / IEEE Computer Society.

Xia, G. S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., Datcu, M., Pelillo, M., and Zhang, L. (2018). DOTA: A Large-Scale Dataset for Object Detection in Aerial Images. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3974–3983.

Xiong, W., Ma, Z., and Song, Y. (2021). Robust augmentations for small object detection of aerial images. In *7th IEEE International Conference on Network Intelligence and Digital Content, IC-NIDC 2021, Beijing, China, November 17-19, 2021*, pages 128–132. IEEE.

Yahyanejad, S., Misiorny, J., and Rinner, B. (2011). Lens distortion correction for thermal cameras to improve aerial imaging with small-scale uavs. In *2011 IEEE International Symposium on Robotic and Sensors Environments, ROSE 2012, Montréal, Canada, September 17-18, 2011*, pages 231–236. IEEE.

Yang, H. Q. (2011). Nondestructive prediction of optimal harvest time of cherry tomatoes using vis-nir spectroscopy and plsr calibration. In *Emerging Engineering Approaches and Applications*, volume 1 of *Advanced Engineering Forum*, pages 92–96. Trans Tech Publications Ltd.

Yeong, S., King, L., and Dol, S. (2015). A review on marine search and rescue operations using unmanned aerial vehicles. *International Journal of Marine and Environmental Sciences*, **9**(2), 396–399.

Yue, J., Fang, L., Rahmani, H., and Ghamisi, P. (2022). Self-supervised learning with adaptive distillation for hyperspectral image classification. *IEEE Trans. Geosci. Remote. Sens.*, **60**, 1–13.

Yun, S., Han, D., Chun, S., Oh, S. J., Choe, J., and Yoo, Y. (2019). CutMix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2019-Octob, pages 6022–6031.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12310–12320. PMLR.

Zhang, G., Zhao, S., Li, W., Du, Q., Ran, Q., and Tao, R. (2020). HTD-Net: A deep convolutional neural network for target detection in hyperspectral imagery. *Remote Sensing*, **12**(9).

Zhang, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. (2018). Mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Zhang, L., Zhang, H., Niu, Y., and Han, W. (2019). Mapping maize water stress based on uav multispectral remote sensing. *Remote Sensing*, **11**(6), 605.

Zhang, Y., Sidibé, D., Morel, O., and Mériaudeau, F. (2021). Deep multimodal fusion for semantic image segmentation: A survey. *Image Vis. Comput.*, **105**, 104042.

Zhao, L., Luo, W., Liao, Q., Chen, S., and Wu, J. (2022). Hyperspectral image classification with contrastive self-supervised learning under limited labeled samples. *IEEE Geoscience and Remote Sensing Letters*, **19**, 1–5.

Zhu, H., Chu, B., Fan, Y., Tao, X., Yin, W., and He, Y. (2017). Hyperspectral Imaging for Predicting the Internal Quality of Kiwifruits Based on Variable Selection Algorithms and Chemometric Models. *Scientific Reports*, **7**(1), 1–13.

Zhu, P., Wen, L., Bian, X., Ling, H., and Hu, Q. (2018). Vision meets drones: A challenge. *CoRR*, **abs/1804.07437**.

Zhu, P., Wen, L., Du, D., Bian, X., Hu, Q., and Ling, H. (2020). Vision meets drones: Past, present and future. *CoRR*, **abs/2001.06303**.