



universität  
wien

# MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Digital Field Twin for Weed Detection“

verfasst von / submitted by

Mathias Höller, BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2022 / Vienna, 2022

Studienkennzahl lt. Studienblatt /  
degree programme code as it appears on  
the student record sheet:

UA 066 921

Studienrichtung lt. Studienblatt /  
degree programme as it appears on  
the student record sheet:

Masterstudium Informatik

Betreut von / Supervisor:

Univ.-Prof. Dr. Stefanie Rinderle-Ma



# Acknowledgements

Studying requires cognitive skills. These skills can be improved through ongoing training. But all this is only possible if sufficient cognitive resources are available. The cognitive bandwidth is limited and is diminished by thoughts about vital things. My family provided me with a carefree life with enough resources that allowed me to devote my cognitive bandwidth to study. The resources were also sufficient to experience a lot of joy and fun during the study period. Thank you for all the opportunities and all the support.

This work deals with digital agriculture. My enthusiasm and curiosity for technical devices often motivated me to disassemble equipment to understand how it works. Unfortunately, it was not always possible to reassemble the device in a functional way. Thank you for allowing me to live out my curiosity even when this sometimes results in destruction. Another place where I was able to discover and learn a lot of new things, and still do, is my uncle's farm. As a child, I often worked on the farm with my grandfather to experience and learn to love agriculture. The enthusiasm for technology and agriculture is still great today and has driven me to write this thesis. Many thanks to my grandfather and uncle, who spent a lot of time to teach me about agriculture.

I would like to thank my supervisors Stefanie Rinderle-Ma and Jürgen Mangler for the opportunity to write this thesis and for the support I received in finding and elaborating the topic.

Machine learning is only possible with a sufficient amount of processed data. The Institute of Agricultural Engineering of the University of Natural Resources and Life Sciences Vienna (BOKU) provided me with data collected by Florian Kitzler. Thank you for the openness towards me and the time it took to explain the data and clarify the legal framework.



# Contents

<b>Acknowledgements</b>	<b>i</b>
<b>1. Introduction</b>	<b>1</b>
1.1. Contribution . . . . .	2
1.2. Research Questions . . . . .	3
1.3. Methodology - Design Science . . . . .	3
1.4. Structure of the Thesis . . . . .	4
<b>2. Related Work and Fundamentals</b>	<b>5</b>
2.1. Convolutional Neural Network . . . . .	5
2.1.1. Atrous Spatial Pyramid Pooling (ASPP) . . . . .	7
2.1.2. Transfer Learning . . . . .	7
2.1.3. Dataset Augmentation . . . . .	8
2.1.4. Class Imbalance . . . . .	8
<b>3. Related Projects</b>	<b>11</b>
3.1. Selective Spraying . . . . .	11
3.2. Inter Row Hoeing . . . . .	11
3.3. EU Projects: ROMI and IWMPRAISE . . . . .	12
3.4. Semantic Segmentation . . . . .	12
3.5. Sugarbeet Bonn 2016 . . . . .	12
3.6. Image Database of Plant Seedlings . . . . .	12
3.7. Generative Adversarial Networks . . . . .	13
3.8. Automatic Model based Dataset Generation . . . . .	13
<b>4. Solution Design</b>	<b>15</b>
4.1. Problem . . . . .	15
4.2. Goals . . . . .	15
4.2.1. Data Collection Workflow . . . . .	15
4.2.2. Digital Field Twin . . . . .	17
4.3. Evaluation Design . . . . .	18
4.3.1. Dataset Separation . . . . .	18
4.3.2. Digital Field Twin . . . . .	18
4.3.3. Digital Collection Workflow . . . . .	18
4.3.4. Metric . . . . .	18

<b>5. Implementation</b>	<b>21</b>
5.1. Technologies . . . . .	21
5.1.1. Deep Lab . . . . .	21
5.1.2. PyTorch . . . . .	21
5.1.3. Optimization Strategy Adaptive Moment Estimation . . . . .	21
5.1.4. Tensor Board . . . . .	22
5.1.5. Conda . . . . .	22
5.1.6. CPEE Workflow Engine – Worker Assistance System . . . . .	22
5.2. Components . . . . .	22
5.2.1. Digital Field Twin Generator . . . . .	23
5.2.2. Database . . . . .	28
5.2.3. Separate Dataset . . . . .	28
5.2.4. Data Collection Workflow . . . . .	30
5.2.5. Class Imbalance . . . . .	33
5.3. Model Training . . . . .	33
5.3.1. Plants and Background . . . . .	34
5.3.2. Generate Digital Field Twin . . . . .	34
5.3.3. Copy to Infrastructure . . . . .	34
5.3.4. Configuration File . . . . .	34
5.3.5. Predict and Calculate Performance . . . . .	35
5.3.6. Predict Large Images . . . . .	37
<b>6. Evaluation</b>	<b>39</b>
6.1. Datasets . . . . .	39
6.1.1. Carrot . . . . .	39
6.1.2. Plant Seedlings Dataset . . . . .	39
6.1.3. University of Applied Life Science Vienna (BOKU) . . . . .	40
6.1.4. Self Collected . . . . .	40
6.2. Separate and Recombine Carrot Dataset . . . . .	40
6.2.1. Carrot Dataset with DBSCAN . . . . .	41
6.3. Data Collection Workflow . . . . .	42
6.3.1. Advantages over State of the Art Approach . . . . .	42
6.3.2. Self Collected Soy Dataset 31.05.2021 . . . . .	43
6.3.3. Self Collected Soy Dataset 11.06.2021 . . . . .	44
6.3.4. Kallham Combined . . . . .	44
6.3.5. Generalization of DFT . . . . .	44
6.4. BOKU Dataset . . . . .	45
6.4.1. BOKU Soy Tested on Kallham Soy . . . . .	45
6.4.2. BOKU Sugar Beet . . . . .	46
<b>7. Discussion</b>	<b>53</b>
7.1. Research Question 1 . . . . .	53
7.2. Research Question 2 . . . . .	54
7.3. Research Question 3 . . . . .	55

<b>8. Conclusion</b>	<b>57</b>
<b>Bibliography</b>	<b>59</b>
<b>A. Appendix</b>	<b>63</b>
A.1. Zusammenfassung . . . . .	63





# 1. Introduction

The United Nations predict that in 2100 the earth will be the home of around 11 billion people. In 2021 the world population is roughly 8 billion. In less than 100 years, the population will increase by one third [Pub19]. Economic growth and the associated increase in household income are contributing to the rising demand for high-quality food. Feeding all people is a big challenge. Feeding all people without harming our planet is even more challenging. Besides the rapid growth of the demand for agricultural products we have to fight climate change and reduce greenhouse gas emissions. The current state of how we produce our food with high performance agriculture pollutes the environment and destroys biodiversity [Nig21]. The World Bank forecasts that the demand for food will increase more than 50% in 2050 compared to 2020 [FM17]. Technologies and innovation will be necessary to fill all tables around the world with good, nutritious and sustainable food.

Innovations can be made, for example, in the areas of biotechnology or digitalization. Biotechnology can help breed more robust, highly nutritious plants with high yields. Digitalization provides the farmer with more detailed information about the needs of plants or animals which then helps to act targeted. Accurate actions such as sensor-guided fertilizing, early disease detection in livestock, or data-based disease control reduce the amount of medicine used in livestock or pesticides and fertilizers applied to fields [Nig21]. Plants need sunlight, nutrients and water to grow. They are in competition for these resources with the plants growing in the surrounding area. In order to provide the planted crops with the maximum resources, weeds must be controlled.

Herbicides are widely used to kill weeds in fields by applying chemicals on the surface. Lately, concerns about the negative impact of herbicides on wildlife and the health of humans spurred the search for alternatives. In the Farm-to-Fork strategy paper, the European Union announced the goal to reduce the use and risk of chemicals in agriculture until 2030 by 50% [BIN].

Technology can help reduce the use of chemicals by applying them directly on top of weeds instead of spraying the whole field with herbicides [LB20]. Another way is to kill the weeds manually. But weeding done by humans is too expensive for most crops. Robots can automate the weeding process and reduce the cost of manual weeding. Both the targeted spraying system or autonomous weeding robots need computer vision systems which can localize and classify plants.

This thesis is about detecting weeds and crops in an image taken from a farm field. Semantic Segmentation assigns class labels to every pixel of an image. The goal of the weed detection algorithm trained in this research project is to assign the labels "weed", "background" and "crop" to every pixel of the image fed to the algorithm. The output of the weed detection algorithm is an image where every pixel is colored according to the

## 1. Introduction

corresponding label. To train and quantify the accuracy of the segmentation algorithm, the ground truth has to be known. The ground truth is often created by human annotators. The annotators have to do the same thing as the algorithm, namely assigning labels to every pixel. There are tools available to assist the annotator in his or her work, but the task is still monotonous and time-consuming. According to the Robotics for Microfarmers (ROMI) project annotating one image with weeds, crops and background takes about 20 minutes [Col18]. Figure 1.1 shows an example image containing three classes (weeds, crops and ground) and the ground truth segmentation. In this research project, an

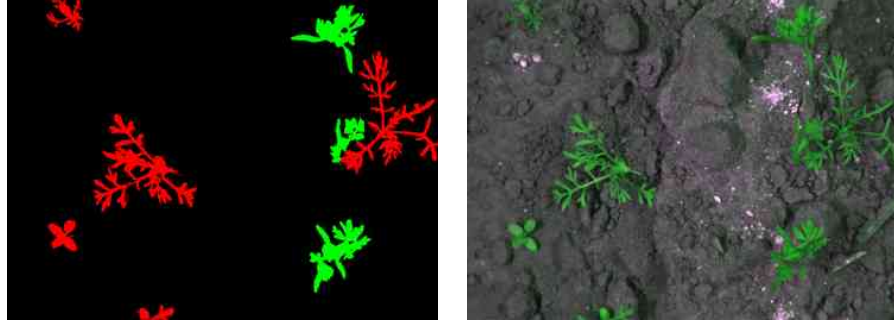


Figure 1.1.: Example of an Image and the Corresponding Annotation Mask of the Crop/-Weed Field Image Dataset [HO15]. Classes: Ground - Black, Weeds - Red, Crops(Carrot) - Green

approach is presented to reduce the need for large annotated datasets. The system consists of two parts. The first part extracts individual plants from already annotated images or collects and annotates new images with a newly designed data collection process performed by farmers on smartphones in the field. The second part is an algorithm that combines the individual plant images with blank ground images into new realistic looking fields. The resulting collection of created images is named "Digital Field Twin" and is used to train deep learning algorithms.

### 1.1. Contribution

This research project shows that it is possible with the help of a worker assistance system to build up a training set based on data recorded and annotated with a smartphone in the field. The artifacts implemented in the scope of this research project are published on GitHub [Hoe22]. It is shown on the example of crop/weed segmentation that it is feasible to train a semantic segmentation model on a synthetic dataset composed of cropped real world plant and background images. Farmers can collect and annotate training data in their fields without having to manually annotate it afterwards. This approach saves time and offers the possibility to train the algorithm on data that are very similar to the target data in terms of spatial and temporal dimensions. The appearance of the plant is dependent on environmental parameters such as nutrients, moisture or sunlight,

but above all, the appearance of the plant changes with time and the different stages of growth.

## 1.2. Research Questions

Human tasks in a process are in general more expensive than automated tasks. Automating these tasks often reduces the time and costs for the process execution. This thesis presents a new approach how to gather new training data in the field with a smartphone. The presented method of data collection is compared to the standard way of taking images and manually labeling data afterwards. The focus of the evaluation lies on the quality of the recorded data to train a semantic segmentation model and less on the time saving in the data collection. Good data is a prerequisite for a good machine learning model and training success.

The first research question is:

**How does the performance of a semantic segmentation model trained on a Digital Field Twin compare to manually annotated training data?**

Well annotated training images are expensive to gather and therefore valuable. For this reason it would be great when a given dataset could be expanded in such a way that a model can better learn the important features which distinguish the classes.

**How do weed/crop segmentation models differ in terms of the similarity to the ground truth when trained on original data or generated data?**

The accuracy of a machine learning algorithm is largely determined by the training data. The chances of success are greater if the training data has a high similarity to the data of the target domain. Deep neural networks work best in a stable environment. Optimal results can be easily reached in a lab environment, where the camera, the background, the objects and the lightning conditions are the same while training and testing the model. The proposed data collection workflow enables farmers to collect training data on their fields and therefore brings the training data very close to the target domain. The third research question shows the influence of the age of a plant on weed detection. **How is the semantic segmentation accuracy influenced by the age of the plants?**

## 1.3. Methodology - Design Science

Design science is used as the scientific method in this research project. Design Science is focusing on the iterative development and evaluation of design artifacts. The research paradigm is mainly found in engineering and computer science research. The designed artifact within the research project has to be innovative, novel and has to solve a yet unsolved problem or improve a existing solution. The design process must produce a

## 1. Introduction

feasible artifact like a construct, a model, a method or an instantiation, that solves a real-world problem. The six steps of design science research are [PTRC07]:

1. Problem Definition and Motivation: The development of technology-based solutions for significant business problems of interest is the goal of design science research. The problem statement should clarify why the problem is worth to be solved.
2. Objective Definition of a Solution: A statement about the achievement of a goal can only be made on the basis of a well-defined goal.
3. Design and Development: Artifacts that contribute to the solution have to be designed and implemented.
4. Demonstration: Show how the created artifacts solve the problem.
5. Evaluation: Evaluation methods are used to show the value, quality and power of the design artifact.
6. Communication: Research findings should be accessible to many. Research only succeeds through teamwork.

### 1.4. Structure of the Thesis

At the start of this thesis a motivation is provided, the introduction states the problem of growing demand for agricultural goods in the world and the need for a more sustainable agriculture. Technologies might help the agriculture to produce high quantities of high quality food and do little harm to the environment during the production process. One paragraph summarizes the contribution to the field machine learning in the domain of weed/crop detection. Further the introduction includes the research question and the methodology used to answer them.

In the related work and fundamentals section the most important technologies used like Convolutional Neural Networks are explained. The related projects section is about similar projects and research regarding targeted weed control. Short descriptions of the projects and the relation to this research project are part of this section.

The forth chapter, Solution Design, dives into the plan how to solve the stated problem of weed/crop segmentation. The Implementation chapter is about the design and implementation of the training data generation and the training of the weed/crop segmentation models. The developed artifacts (trained models, data collection workflow, Digital Field Twin Generator) are tested. The Evaluation section outlines the data used to test the models, describes the testing procedure and presents the evaluation results.

Finally, the most important findings are summarized and possible further developments are outlined.

## 2. Related Work and Fundamentals

### 2.1. Convolutional Neural Network

In the era of digitization more and more data gets collected day by day. Millions of smartphones and any other smart device connected to the internet leaves a data trail. Companies specialized in analyzing this huge amounts of data are among the most valuable companies in the world. In the last decades data availability and cheap computation power increased a lot. This lead to a boost of the field of Machine Learning. Machine Learning is about solving problems like predicting the next move of a player in a game, forecasting what product a costumer likes to buy, which film the user will like best or recognising the face looking into the smartphone camera. These predictions or classifications are based on statistical models whose parameters are learned by minimizing an error function on historical data [ibm21].

In this project a Convolutional Neural Network (CNN), a tool from the machine learning toolbox, is used for image segmentation. CNNs turn inputs (feature maps) like images into useful information by applying multiple filters. CNNs are made up of layers, which in turn are made up of neurons. The input of a layer is a feature map which is transformed by a convolution filter into another feature map which contain more useful information for the specific task. Neurons are rather simple statistical models that output one number. Neurons of one layer are connected to the neurons of the next layer. CNNs are not fully connected which means that the neurons of the next layer do not take into account all outputs of the neurons of the previous layer. This reduces the number of parameters, called weights, that have to be determined while learning, for each neuron. The input signals of a neuron are weighted and aggregated. The wights are tuned while training the network [nvi21]. Figure 2.1 shows a convolutional neural network transforming a traffic sign into a number. Two convolutional filter layers each combined with a sub sampling layer transform the input image (32x32) into feature maps (5x5). The final layer combines all feature maps to classify the number painted on the sign [CNN18].

**Field of View** The term neural network comes from the similarity of the statistical model and the biological model of the brain. There is another similarity of the human perception system and the field of view of a neural network. The intuition is that it is easier for us to identify a car as such when we see the whole car then when we only look at the headlights of a car. Figure 2.2 shows a street scene with pedestrians, cars, buildings and a pedestrian crossing. In the right image, each pixel is colored with the color of the corresponding class. A pixel (right image) in the center of the red car (left image) has been assigned the class car with the color purple. For this purpose, the environment of the pixel of different size can be included in the decision, the so-called field of view. The

## 2. Related Work and Fundamentals

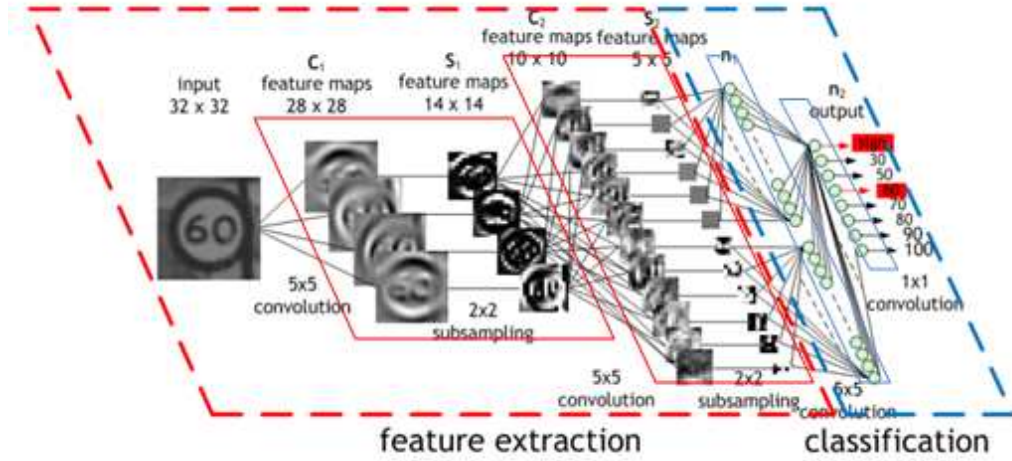


Figure 2.1.: Model of a CNN Transforming a Traffic Sign Into a Number

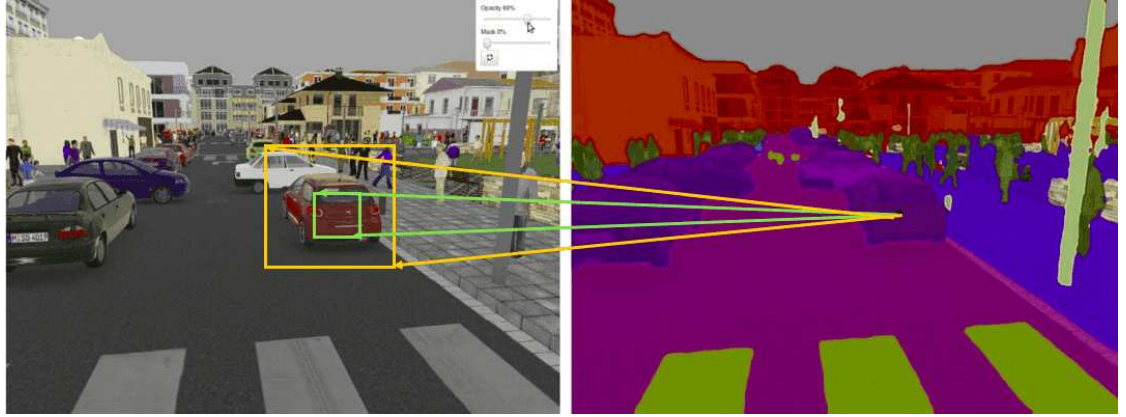


Figure 2.2.: Field of View Example

green and red rectangles in the left image show two different sized fields of view. On the one hand a larger Field of View results in better segmentation, on the other hand it requires more computation power and memory [nvi21].

Atrus Convolution, used in Deeplab, increases the field of view while maintaining the same computational complexity.

André Araujo et. al. showed that it is the same with CNNs, a larger field-of-view results in a better classification accuracy [ANS19]. The drawback of a larger field-of-view is that the parameters per neuron in the CNN, which have to be adjusted to minimize the error function, increase with a larger field-of-view and with it the amount of computation and memory needed for training. The size of the field-of-view can be influenced by the number of layers or the size of the convolution matrix.

### 2.1.1. Atrous Spatial Pyramid Pooling (ASPP)

Standard convolution is often used in image processing to apply filters like Gaussian filters to blur an image or median filter to reduce the noise of an image. A 3x3 filter matrix of a box filter, also called mean filter, has a value of  $1/9$  in each cell. The filter is shifted pixel by pixel over the image. For each position the filter is element-wise multiplied by the corresponding pixels in the image. In the example case of a 3x3 box filter one pixel of the image and all the neighboring pixels are multiplied by  $1/9$  and added up. The average of one pixel and their direct neighbors is calculated. This leads to a blurrier image after the box filter is applied [Gon18].

The same concept of convolution is used with Deep Convolutional Neural Network (DCNN) to connect layers and create feature maps. DeepLab uses a special version of convolution to widen the field of view without increasing the amount of computation. The idea of Atrous Convolution is to create a sparse filter matrix. The parameter rate defines how many zeros should be inserted into the filter matrix between the non-zero values. In the example of the standard convolution only the neighboring pixels were used (which makes sense to blur the image). If the area of view should be increased, the filter matrix has to be increased as well, for example to 5x5. Instead of 9 values (3x3) the new filter has 25 values (5x5). These values are weights in the case of a DCNN and have to be adjusted and calculated while training. Increasing the amount of these values also increases the amount of computation and memory which is necessary for learning. Atrous Convolution can "see" similar to a 5x5 convolution but still has 9 parameters to calculate by instantiate the new cells introduced from 3x3 to 5x5 with zeros and skip the calculation of this values. This leads to a good accuracy/efficiency trade-off according to [CZP<sup>+</sup>18].

Deeplabv3 uses as base for the feature encoder, called backbone, different networks like the RESNET 101 and adds a Atrous Spatial Pyramid Pooling on top of it. The idea of this is to have different sized field of views calculated in parallel for the same feature map. This makes the semantic segmentation model less dependent of different scaled images or objects.

Figure 5.3 shows the Deeplab v3 Model. Block1, 2 and 3 encode the features of the image while shrinking the image. The feature maps get smaller by traversing the network up to Block 4. Atrous Convolution encodes features at different scales. The outputs of the Atrous Spatial Pyramid Pooling are concatenated and aggregated by a 1x1 convolution.

### 2.1.2. Transfer Learning

Transfer learning is about reusing parameters learned in one setting to understand a new setting faster. Starting with a pre-trained neural network helps reducing the time and data necessary to achieve satisfying results. In object detection the transferred information used as a starting point can be seen as the basic understanding of visual scenes. The following rule is an example of information which is important in order to understand visual scenes: Pixels close to each other are more likely parts of the same object than

## 2. Related Work and Fundamentals

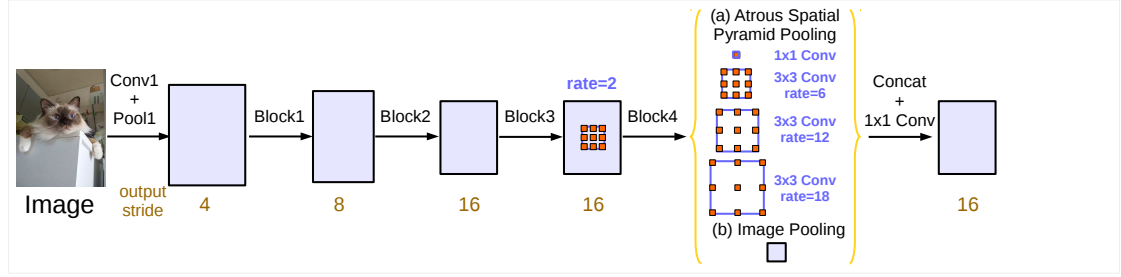


Figure 2.3.: Deeplab v3 Model [CZP<sup>+</sup>18]

pixels far apart. These rules are encoded implicitly in the trained parameters [Goo16]. Transfer learning is applied in this work to reduce data volume and training time.

### 2.1.3. Dataset Augmentation

Deep learning algorithms need a large amount of labeled data to learn and generalize well. The algorithm generalizes well, if it has good performance on data which was not processed by the algorithm, thus not contained in the training set. There are several variables which influence the performance of the algorithm: layers, training epochs, neural network architecture or training data. Collecting more data is often the key to a better generalization and thus a better performance of the algorithm. But annotated data samples are expensive. Collecting new data is often not possible. The idea of dataset augmentation is to apply transformations like rescaling, rotation, random translation or random perturbation of the colors. The transformed data is then added to the training set. Data augmentation extends the size of the training data set without collecting new real data and enhances the classification accuracy of the algorithm [Goo16].

In this work data augmentation is applied to every single plant image in order to achieve better results with the same amount of annotated data.

### 2.1.4. Class Imbalance

Class Imbalance is given in a lot of real world data sets where the occurrence of objects in the images differ a lot or, in the case of semantic segmentation, the size of the objects is very different. Algorithms can reach good average performance without taking into account the least occurring class on an imbalanced data set.

Example: A dataset contains class A and B. The algorithm is trained to classify the objects as A or B. The training set includes 1000 objects of class A and 10 objects of class B. Obviously the dataset is very unbalanced, 100 times more class A objects than class B objects are part of the dataset. When classifying all objects as class A, the algorithm is right in 1000 cases and wrong in only 10 cases.

In the case of autonomous driving a small area and low occurrence are pedestrians. Therefore, misclassification of pedestrians, respectively the pixels showing the pedestrian, has not a big influence on the over all error of the semantic segmentation because of



the little number of pixels showing pedestrians. [CRH<sup>+</sup>19] showed that class imbalance in street scenes, used to train a autonomous driving algorithm, are a reason for bad performance on objects with little frequency in the data set. One solution is to use weighted loss functions where the classes are weight by the inverse of the frequency of the class in the data set.

Another approach to reduce the error due to class imbalance is the adaption of the data itself. A balanced data set can be constructed by splitting and recombining the classes. This approach is used with the digital field twin to get a class balanced data set with similar area showing crops and weeds.

Crop plants take up significantly more space in fields than weeds, because weeds are controlled by the farmer. For this reason, a dataset containing weeds and useful plants is usually class imbalanced.



## 3. Related Projects

This section lists and describes research projects that are related to weed/crop detection.

### 3.1. Selective Spraying

Herbicides are traditionally applied to the entire field to rid it of weeds. The crop plants are resistant to the chemical, so they remain largely unharmed. The targeted application of herbicides only to partial areas on which weeds are growing has ecological and economic advantages. Liu and Bruch used weed detection in a Romain Lettuce field to demonstrate a state-of-the-art workflow that includes field image labeling, training, and evaluation of a deep learning model for weed detection. 3000 labeled examples were used to train and test the model. The first 500 images were manually labeled by hand. The remaining 2500 images were prelabeled by an algorithm trained on the first 500 labeled images. The resulting model can detect Romain Lettuce with an mean average precision of 92% in unseen images from the training examples. This means that 92% of the predictions that a certain part of the image shows a Romain Lettuce are true. Factors like environmental lightning, microclimate, occluded plant leaves, or spectral properties at different growth stages of plants are stated as challenges for developing crop/weed detection algorithms. Herewith was shown that deep learning models can reach high accuracy in weed/crop detection [LB20].

The data collection workflow proposed in this work could be executed in the field, right before the autonomous weeding process starts. The small temporal difference between data collection, training on the collected data and autonomous weeding could help master the challenges stated by Liu and Bruch.

### 3.2. Inter Row Hoeing

A common way of manual weed control with computer support is inter-row hoeing. Crops such as beets, corn or soybeans are grown in rows. Inter-row hoeing kills the weeds by tilling the ground between the rows. The row spacing varies depending on the crops. Crops such as corn have up to 80cm of space between the rows. Small row distance requires precise operation of the hoe to minimize the tilling and the damage of crops. Computer programs help detecting rows and guide the driver or even autosteer the tractor to precisely drive and till in between the rows [AHIS]. The problem with the described approach is that weeds that grow very closely to the crops or between the crops in the row are not killed [Me].

#### 3.3. EU Projects: ROMI and IWMPRAISE

The projects ROMI and IWMPRAISE, funded by the European Union, are working on the development of automatic weeding systems [Col18, iwm21]. Both have in common that computer vision and deep learning is used to determine the position of weeds. The weeding systems kill the weeds manually.

In this work images are also labeled pixel by pixel by a deep learning algorithm.

#### 3.4. Semantic Segmentation

Semantic Segmentation classifies the image pixel by pixel. Every pixel in the image gets a class label. The following list shows some examples where semantic segmentation is used:

- Background - Foreground Separation: Images, particularly portraits, look better when the background is blurred and the object in the foreground stays sharp. The background of the image has to be determined on pixel level so that only the background pixels are blurred [goo21].
- Organ Segmentation: The segmentation of medical full body scans and the coloring of different body parts instead of gray tones, helps simplify and speed up the diagnosis [SPC<sup>+</sup>20].
- Hair Color Augmentation: Trying on different hair colors via smartphone requires fast and accurate hair segmentation [hai21].
- Self-driving cars/Cityscapes: Cars have to understand the surroundings in street scenes to react accordingly. Cityscapes is a large dataset containing more than 5000 pixel-wise labeled images of street scenes with 30 classes (e.g. sky, road, people, bridge) [COR<sup>+</sup>16].

#### 3.5. Sugarbeet Bonn 2016

The Sugar Beets 2016 dataset provided by the University of Bonn is collected by an autonomous field robot. Lidar sensors, GPS receivers and different camera systems mounted on the robot gather data about the plants in the field. The robot started to collect data in the field when the first plants emerged and stopped when accessing the field was no longer possible without damage to the crops. In a period of 3 months in spring 2016, the field was surveyed three times a week on average. The dataset also contains an annotation mask in which the plants are labeled pixel by pixel [CLS<sup>+</sup>].

#### 3.6. Image Database of Plant Seedlings

Mosgaard Giselsson et. al published a database containing over 500 images of plants of 12 species. The database contains 12 common weed and crop plants from danish agriculture.

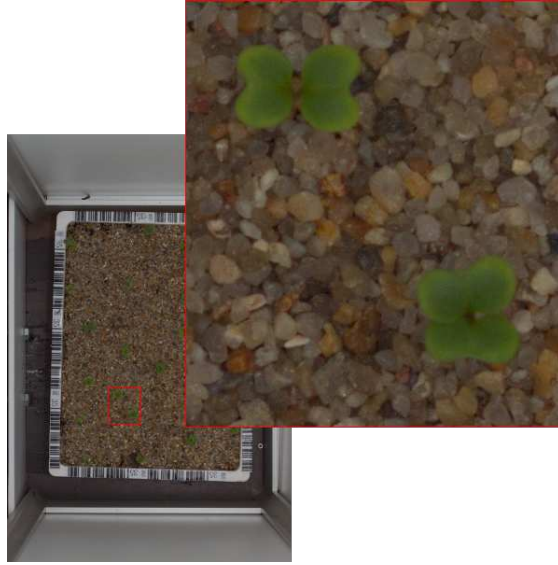


Figure 3.1.: Sample Image from Seedlings Database by Mosgaard Giselsson et. al [GJJ<sup>+</sup>17]

The seedlings are grown in trays containing one species and marked with a bar code. Every two to three days over a period of 20 days the trays were placed into a device which took images of the plants. The images were then segmented into foreground (plants) and background by a Naive Bayes model [GJJ<sup>+</sup>17]. The segmented and cropped images of plants are used in this research project to build a Digital Field Twin.

### 3.7. Generative Adversarial Networks

Fawakherji et al. have used a GAN to generate realistic agricultural scenes [FPP<sup>+</sup>]. Generative Adversarial Networks (GANs) are a generative modeling approach to produce samples similar to or even indistinguishable from a given training dataset. Generated samples and samples drawn from the training dataset are randomly given to a discriminator network. The discriminator network classifies the samples as fake or real. The generator network learns to generate more realistic samples and tries to fool the discriminator network. The network converges towards a state in which, for the discriminator, the fake samples are indistinguishable from the real samples [Goo16].

The extension of the training set with the generated scenes resulted in a better semantic segmentation accuracy. In this work, the goal is to recombine real images to realistic agricultural scenes with a simple insertion algorithm.

### 3.8. Automatic Model based Dataset Generation

Cicco et. al used a 3D plant model and texture from real world images to build synthetic training data for weed/crop detection models. The realistic agricultural scenes are based

### *3. Related Projects*

on parameterized plant models and terrain generation (ground image). The parameters of the plant model are randomized so that different agricultural scenes can be generated. An artificial camera captures images of the 3D agricultural scene model, which are used to train a semantic segmentation algorithm. The described method could significantly reduce the effort required for manual annotation of data by humans while maintaining the accuracy of the classification [CPGP].

New weeds or crops require a new parameterized plant model. The approach proposed in this work is based on a simple algorithm without knowledge about the plants. Adapting the algorithm is not necessary for new plants or weeds.

## 4. Solution Design

Design Science is used as the scientific method for this research project. Feasible artifacts are developed and evaluated to solve a problem identified in a specific domain [PTRC07]. This section contains a clear problem description, the defined goals of this research project, the out-coming artifacts and the plan of how the implemented artifacts are evaluated.

### 4.1. Problem

Weeds compete with crops in terms of sunlight, water, and nutrients. To ensure optimal growing conditions of the field, weed control is essential. Common ways of weed control are based on chemicals applied on crops and weeds or manual weeding, often done by humans. The drawbacks of these methods are that chemicals are sometimes not working due to herbicide-resistant plants and are suspected to have a negative environmental impact. Manual weeding is labor-intensive and therefore expensive for the farmer. Furthermore manual weeding is hard work and could harm the workers health. Autonomous weeding robots could be a solution for this problem. Weeding robots have to distinguish between weeds, crops and ground. Camera systems and artificial intelligence can be used to build visual weed detection systems. Training a weed detection algorithm requires thousands of images. Collecting and labeling the images takes a lot of time and is therefore expensive.

### 4.2. Goals

Convolutional Neural Networks (CNNs) are suitable for building fast and accurate weed/crop detection systems. A downside of CNNs is that a large amount of expensive labeled training data is necessary for training. CNNs work best in a stable environment, where training and target domain are similar and do not change rapidly. The goal of this work is to develop a method to expand labeled training data and ease the collection of new labeled plant images. The following artifacts are implemented and evaluated:

- Workflow to collect labeled images of crops and weeds via smartphone.
- Process to create a digital field twin.

#### 4.2.1. Data Collection Workflow

The Data Collection Workflow should ease the process of collecting images of plants and labeling them in the field. The collection workflow could be executed right before the weeding robot starts to control weeds in the field. The benefit of this approach is that

#### 4. Solution Design

farmers can collect and label data themselves and the collected plant images are very similar to the plants which should be detected by a weeding robot. Training data drawn from the same distribution as the data which should be labeled in the end improves the accuracy of the algorithm.

The plants are collected one by one. The first step is that the user prepares the background so that extracting the plant in the image computationally becomes easy. The ground is covered by black paper as shown in Figure 4.1. As a second task, the user takes an image of the plant on a smartphone. The background is then removed by thresholding. Now the plant is cropped and shown to the user. After the user confirms that the background removal worked well, attributes about the plants are inputted by the user. The attributes are: class (weed, crop or plant species), diameter, and age of the plant. The attributes are used to create realistic field scenes from the collected plants. The data is stored in a database for easy and structured access.

Sometimes weeds grow very close to a crop and is hardly possible to put the black paper under the crop and weed separately. In this case the weed can be ripped out and photographed by clamping the roots in a clip.



Figure 4.1.: Left: Background Preparation with Black Paper; Upper Right: Photo Taken on a Smartphone; Lower Right: Cropped Plant Image

Due to the fact that the ground is removed to get cropped images of plants, the ground has to be collected separately. Ground images are collected in the time span between seeding and the first sign of plants on the surface with a smartphone. This ensures that only soil and no other classes are included in the image. In addition, other ground surfaces without vegetation were photographed to obtain different backgrounds, e.g. field paths. The Data Collection Workflow will be implemented with the Cloud Process Execution



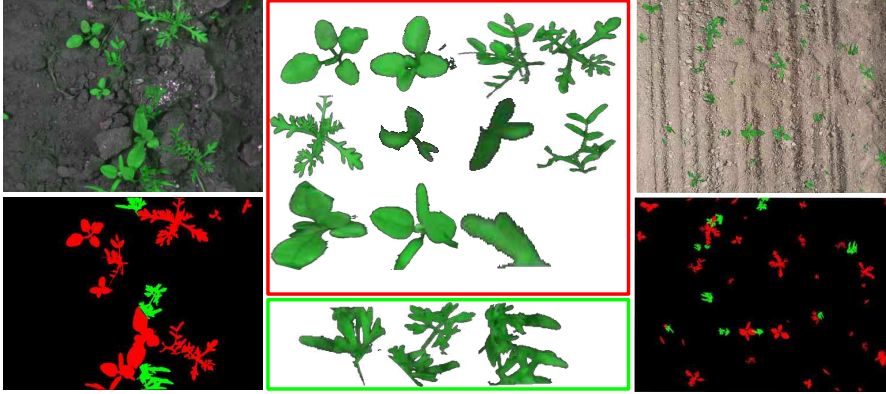


Figure 4.2.: Left: Carrot Dataset Annotation and Real Image [HO15]; Middle: Separated Weeds with Red Border, Separated Crops with Green Border; Right: Synthetic Image Generated from the Depicted Separated Plants in the Middle and a Ground Image

Engine [cpe21].

The workflow engine has the advantage that the process is explicitly modeled and easy to adapt. New services, like further instructions how to use the tool or a new storage service, can be added with few changes in the process model. No fiddling around in a code fragment is needed for changes. Furthermore it is easy to change to control flow. A logger tracks the process execution. The logs can be investigated to gain insights what make a process execution result in good or bad training data. This analysis of the process is not part of this thesis.

#### 4.2.2. Digital Field Twin

The Digital Field Twin combines the collected data (weeds, crops and grounds) to realistic agricultural scenes. A digital field twin is a collection of syntactic images that show a field of crops and weeds with certain attributes (age, species). The digital field twin is used to train a semantic segmentation algorithm.

Aside from the data collection via smartphone the cropped plant images can also be generated from annotated datasets. Annotation masks color code pixels according to their class. The annotation masks are split up into the three classes: crops, weeds and ground. Then the weed and crop class masks are separated into single cropped plant masks. The separated crop and weed masks are used to crop the original image into single plant images.

Two different approaches are implemented to separate the crop and weed annotation masks depending on the available information. The first possibility is that the data contains files with the polygons drawn by the annotator to edge the plants. These polygons are used to separate the plants. If the polygon information is not part of the dataset, a density based clustering (DBSCAN) is performed on the annotation mask. Each cluster in the DBSCAN result corresponds to one plant instance.

### 4.3. Evaluation Design

The DeeplabV3 with a ResNet-101 backbone implemented with pytorch pre-trained on a subset of COCO train2017 is used to evaluate the proposed method [CZP<sup>+</sup>18, pyt21b, coc20, Min19].

#### 4.3.1. Dataset Separation

The first evaluation is related to the separation algorithm. The two methods for separating annotated datasets, DBSCAN and Polygon, are compared by splitting a dataset once with each algorithm. The separated datasets are used to create DFTs and train a semantic segmentation model. The models are used to predict new unseen images from the original dataset. The carrot dataset is used for this experiment, because the images were taken sequentially in a row in groups a few meters apart. The last group is not included in the training data and is used as a test set [HO15].

#### 4.3.2. Digital Field Twin

The goal of this experiment is to compare the accuracy of models trained using original data or DFT as training data. Sugar beet datasets from BOKU and the University of Bonn are used for this purpose [HO15]. The BOKU datasets are from 2020 and 2021. The models are trained with the data from the Uni Bonn and the BOKU 2020 data. A subset of the BOKU data from 2021 will be used to test the segmentation accuracy of the trained models. This will test how well the models can segment unseen images from other fields, that means how well the models can generalize the learned knowledge.

#### 4.3.3. Digital Collection Workflow

This test is intended to show whether the collection of single plants with the Data Collection Workflow is suitable for training a semantic segmentation model. On a soybean field in Kallham, Upper Austria, the Data Collection Workflow is executed on two different days to collect soybean plants. The plants are combined with weeds from the Plant Seedlings dataset to form a DFT [GJJ<sup>+</sup>17]. 10 test images are taken at different locations in the field where no single plants were collected. The models are also being tested on images of BOKU soybean fields to determine how well the models work on other fields.

#### 4.3.4. Metric

The chosen performance metric is the Jaccard Index, also called Intersection over Union (IOU). This metric is widely used in the area of semantic segmentation and is therefore suitable for comparison with similar projects working on weed detection algorithms [AHIS]. The IoU score measures the similarity of the predicted segmentation and the ground truth area. The metric can be calculated efficiently, is easy to understand, and summarizes the algorithm's performance in a single number.

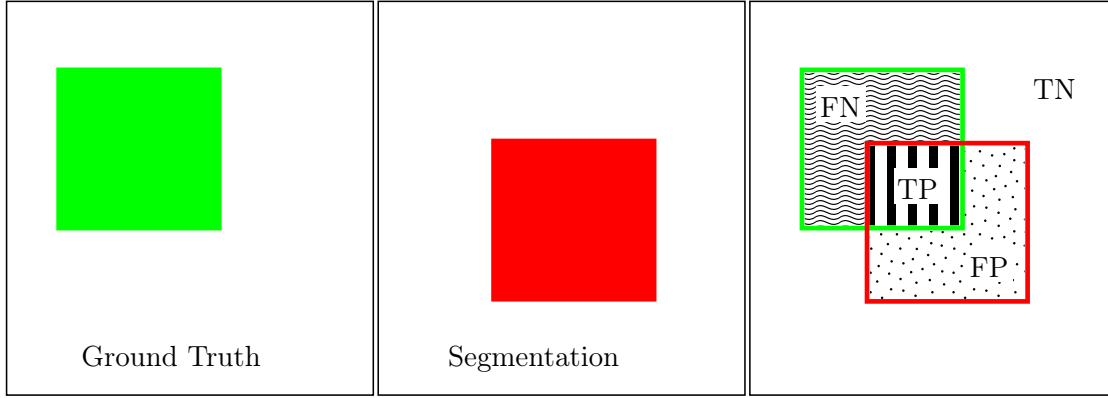


Figure 4.3.: Illustration of the Intersection over Union Score for one Class

A single number metric can ease the comparison of different models and the decision on which one is best [Ng18].

The IoU is calculated per class and is then aggregated by the mean of the three classes: weed, crop and ground. The average Intersection over Union score is:

$$IoU = \frac{1}{N} \sum_{i=1}^N \left( \frac{TP_i}{TP_i + FP_i + FN_i} \right) \quad (4.1)$$

where  $i$  is the class and TP, FN, FP are the number of true positive, false negative and false positive pixels. Semantic Segmentation assigns a class to every pixel in an image. In the example shown in Figure 4.3 the task is to segment an image showing a square hut on a field. The classes are hut and background. The score is calculated for the class hut. The left figure shows the correct segmentation, with the pixels of the hut marked in green. A possible segmentation output of an algorithm is shown in the middle, with the pixels of the hut marked in red. The IOU is calculated based on the left and middle figure. The intersection of the red and green area is True Positive (TP). TP means the segmentation of the algorithm is correct for this area. False Negative (FN) are pixels that are classified as background but are part of the hut. False Postive (FP) is the opposite of FN, so pixels are classified as hut but are part of the background. The TP area is divided by the area of  $FN + TP + FP$ . The IOU score is 0.2. (See 4.3 for an illustration of the IoU score for one class)

The average IoU is used to compare different models. The per class IoU shows the difference in the segmentation accuracy over different classes.

Interpretation: The Jaccard Index measures the similarity of two sets. In the case of semantic segmentation the two sets are the annotation mask, determined by the semantic segmentation algorithm, and the ground truth mask. A IoU score of 1 or near 1 means that the prediction of the classes is the same or very similar to the ground truth.



## 5. Implementation

### 5.1. Technologies

In this chapter the most important technologies used to implement the artifacts in this project are presented.

#### 5.1.1. Deep Lab

DeepLab was presented in 2015 by Chen et. al. and was iteratively improved in the following years [CZP<sup>+</sup>18]. The DeepLab system addresses the task of semantic segmentation. DeepLab3+, the latest release of the DeepLab system, is listed as fifth best semantic segmentation algorithm on the dataset used at the PASCAL Visual Object Challenge 2012 [EVGW<sup>+</sup>, pas21]. DeepLab is open source and available on Github [ten22]. The original implementation uses the deep learning framework developed by Google named Tensorflow. In this research project a pytorch implementation of DeepLabv3 was used [pyt21a]. The implementation of the network itself was not changed in this project but fine tuned on new data.

#### 5.1.2. PyTorch

PyTorch is a open source framework for machine learning developed by researchers at Facebook [pyt21b]. A big community is implementing and training models which are publicly available. Due to the wide distribution there are many resources like tutorials or error reports which are helpful in using the framework. The framework is used to load the images, determine the segmentation output, calculate the error compared to the ground truth and minimize the error. The trained models with a parameter setting can be stored and reloaded later to compare different training steps and datasets. The framework provides functions to calculate the performance of models by loading a model, predicting a segmentation and determining the difference to the ground truth annotation masks.

#### 5.1.3. Optimization Strategy Adaptive Momemt Estimation

The goal of semantic segmentation or in general classification is to predict the class of a pixel, image, text, music or any other data. In the case of supervised learning the ground truth is necessary for learning. The ground truth class of an image showing a cat is the string "cat". Figure 1.1 shows an example of a ground truth, the annotation mask, for a semantic segmentation of weed/crop image. The model is trained by showing the model the ground truth and the original image. In this work, the original image is an image of a

## 5. Implementation

field that contains ground, weeds and crops. The ground truth is an images where each pixel is colored green (crop), red (weed) or black (ground) (see Figure 1.1). The semantic segmentation model predicts the classes of every pixel and compares the prediction with the ground truth. This comparison is done by a mathematical equation which gives an error. Calculating the gradient of the error function is a complex calculation and need a lot of computation power. The gradient is necessary to know which parameters have to be tuned to minimize the error. The calculation of the error and the optimization step is implemented within the PyTorch framework. In this project a type of stochastic gradient with adaptive learning rates is used, namely Adaptive Momemt Estimation [KB14].

### 5.1.4. Tensor Board

TensorBoard is a logger with a web interface designed to track trainings of machinelearning models [ten21]. Charts show the error over time on training and test set. The interface gives a good overview on the training progress. Different training runs can be shown in the same figure to compare different settings like parameters or datasets. Tensor Board provides a convenient way of keeping track of the different training iterations. During this project over 100 different training runs were executed. In this case it is important to have a tool which summarizes the training runs in a compact interface.

### 5.1.5. Conda

Conda is a environment and package management system. Different environments can be created without any interference on the same machine [con21]. This is especially helpful in the machine learning context, since large computing capacities are needed and these are often available on a shared server. Further conda manages packages and dependencies.

### 5.1.6. CPEE Workflow Engine – Worker Assistance System

The Cloud Process Execution Engine (CPEE) is a open-source, modular and lightweight workflow execution engine supporting multiple protocols for service interaction [cpe21]. In this research project it is used to orchestrate the services that takes part in the Data Collection Workflow and guides the user through the workflow. The CPEE has a logging infrastructure that could be customised and help to gain information about the process. The logging tool has not been used in this project but could be used to learn from the execution logs how the process is executed by the user and what parameters lead to good training data or what are promising ways to improve the collection process.

## 5.2. Components

This section describes all components that were implemented for this project. The components are implemented as independent, lightweight programs that take care of one task, so called micro services. The code is published on GitHub [Hoe22].

Figure 5.1 shows all components which take part in the collection of data and the Digital

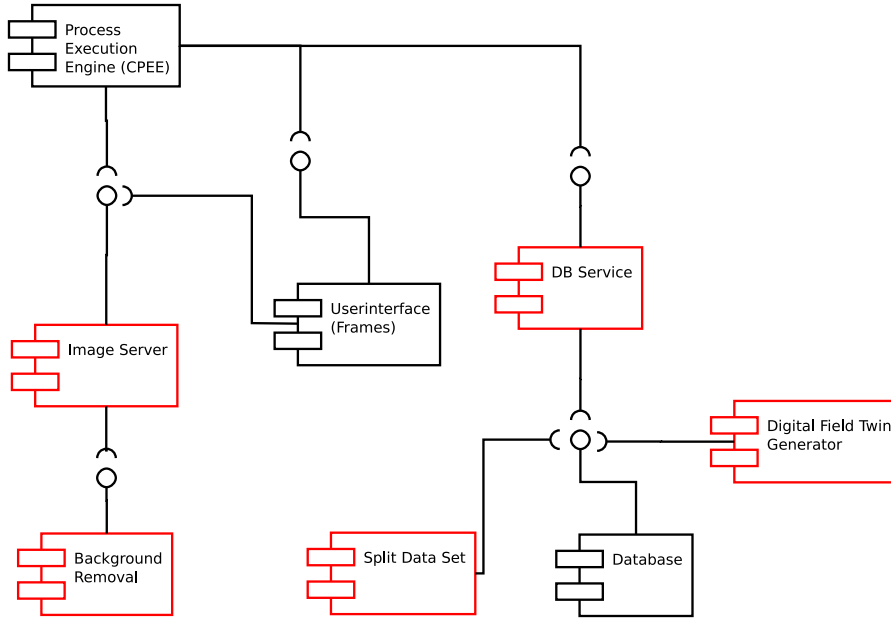


Figure 5.1.: Architecture and Developed Components (red)

Field Twin generation. The collection of new data is coordinated by the Data Collection Workflow executed by the CPEE. The workflow instructs a person via the Userinterface how to collect the crop data, sends information about the collected data to the DB Server and handles the communication to the Image Server. Annotated data can be split up via the Split Dataset component and thus become part of a DFT. The Digital Field Twin Generator gets the location and metadata of the desired weeds, crops and ground images from the Database Service. The images are loaded from the File System and combined to a DFT.

### 5.2.1. Digital Field Twin Generator

The purpose of the Digital Field Twin Generator is to create a agricultural scene composed of crop, weed and ground images. The goal is to generate a large and diverse dataset with thousands of images based on tens to hundreds of single plant/ground images. The resulting dataset can be used to train a weed detection deep learning model.

The Digital Field Twin (DFT) is generated by randomly inserting images of a plant with transparent background in a ground image. The plant images are transformed in different ways to increase the variety in the DFT. The generator takes a list of crops and weeds that should be combined to the DFT. The second parameter is the number of generated images building the resulting DFT. The algorithm is shown in the following pseudo code:

## 5. Implementation

**input** : A crop species *crop*, one or multiple weed species *Lweed* and the number of images that should be generated *Nim*

**output** : Digital Field Twin

```
for 0 to Nim do
  gim ← GetRandGroundIm();
  for igim ← 0 to number_of_ground_segments do
    cropped_ground_image ← CropGroundIm(ground_image, igim);
    cropped_ground_image ← ChangeLighness(cropped_ground_image);
    cropped_ground_image ← Flip(cropped_ground_image);
    classbalance ← {ground : 1, weed : 0, crop : 0};
    while classbalance[ground] < 0.95 do
      if classbalance[weed] < classbalance[crop] then
        | pim ← GetRandPlantIm(Lweed)
      else
        | pim ← GetRandPlantIm(crop)
      end
      normalize plant_image size;
      ajust the color of the plant image plant_image to the background
      cropped_plant_image;
      plant_image ← ChangeColor(plant_image);
      plant_image ← Rotate(plant_image);
      plant_image ← Flip(plant_image);
      cropped_ground_image ← insert plant_image in
      cropped_plant_image at random position;
      classbalance ← CalculateClassBalance(cropped_plant_image)
    end
    store cropped_ground_image;
    igim++;
  end
end
```

Algorithm 1: Pseudo Code Digital Field Twin Generator

## Details

### Transform Images

The semantic segmentation algorithm learns parameters to predict whether a pixel belongs to the ground, a crop or a weed. When the training images come from a distribution which is different than the target domain, this can lead to low performance on images from the target domain. When the training images are taken on a cloudy day and the prediction should be made on images taken on a sunny day, the distributions, where the images come from, are different. Adding synthetic images, where the brightness or color is slightly changed, makes the training images distribution more similar to the input images distribution.



Name	<b>GetRandGroundIm</b>
Input	String with a note string that the DB records have to match with.
Output	Ground image with information about the width of the image in cm.
Description	The function queries a random entry with the given note string from the database. The database returns the width and the filename of the image. The loaded image and the width information is returned.

Name	<b>CropGroundIm</b>
Input	Ground image with original size.
Output	Part of the image with the dimensions 513px x 513px.
Description	Crop a square part of the image with a length of a side of 513px.

Name	<b>ChangeLightness</b>
Input	Cropped ground image.
Output	Image where the lightness of the image (L channel in LAB color representation) is randomly changed in the range [0.5, 1.5].
Description	Multiply the L channel of an image in the LAB representation to make it brighter or darker randomly in the range [0.5, 1.5].

Name	<b>Flip</b>
Input	Image.
Output	Flipped image.
Description	The image is flipped horizontal or vertical with a probability of 0.3.

Name	<b>GetRandPlantIm</b>
Input	Parameter hash with the obligate key species and the optional keys: note, min_age, max_age and timestr.
Output	A list with the location of an image with the properties defined by the passed parameters and the properties of the plant image (e.g. age, width in cm, notes).
Description	The request is translated to an SQL statement and send to the database. If no plant image in the database fits the defined properties the returned list is empty.

## 5. Implementation

Name	<b>Normalize Plant Image</b>
Input	Image of the plant, width of the ground image and the width of the plant image
Output	The plant image is resized to fit the width of the ground image.
Description	The resolution of the plant image and the ground image are different in most cases. A cm in the ground image doesn't equal an cm in the plant image. The plant has to be resized to fit the proportions of the ground image and look realistically after insertion. Some randomness is introduced in the scaling by multiplying the resize factor with a number from the range $[0.8, 1.2]$ .
Name	<b>Align Plant Color to the Background</b>
Input	Image of the plant and ground
Output	The lightness of the plant image is adapted to the ground image.
Description	The lightness of the plant and ground image could be very different, this leads to very unrealistic scenes (e.g. the ground is black and the plant very bright). Both images are transformed into the LAB color representation. The mean value of the L channel are calculated. There by it is important to ignore the pixels in the plant image where the alpha channel indicates transparency. The L channel of the plant image is multiplied by $mean\_L_{ground}/mean\_L_{plant} * [0.5, 1.35]$ . A certain fuzziness is given by a multiplicative factor from the range $[0.5, 1.35]$ .
Name	<b>Change Color</b>
Input	Image of the plant
Output	The image of the plant with slight color changes.
Description	The color channels (A,B channel) are multiplied by a random factor in the range of $[0.85, 1.15]$ . All pixels in one channel are multiplied by the same number. A and B channel can have different numbers to alter the color.
Name	<b>Calculate Class Balance</b>
Input	Annotation mask of the scene (ground with plants)
Output	A list with three entries, the portion of pixels of the class. The first entry is the portion of pixels which are red and therefore belong to a weed. Pixels of crops are the second value. The last value represents ground pixels.
Description	The resulting dataset should show a similar amount of crop and weed pixels. The class balance toggles whether the next plant to be inserted is a weed or a crop. The insertion loop stops when the amount of ground pixels are less then 95% of all pixels in the image.

The position of the plants should have no influence on the prediction of the algorithm. But if all or most of the crops in the training dataset are in the right upper corner and most weeds are in the lower left corner of the image the algorithm could minimize the error by using the position of the plant to predict the class (weed, crop). Flipping, rotation and translation alters the images such that position and orientation are randomized and thus not useful to make good predictions on the test dataset. Transforming the images creates new images with slightly different lightning, colors and orientation. This helps the CNN to generalize better and learn the right parameter to distinguish between ground, weed and crop. The class predictions should be invariant of orientation, position or lightning. Further, small errors in the image like blur or noise should have minimal impact at the accuracy of the algorithm.

**Flipping and Rotation** Flipping and rotating images of plants taken from bird's eye view is easy. There are no wrong rotations or flips. The plants are rotated randomly between 0 and 360° and flipped horizontally and vertically by a probability of 30%. This transformation adds "new" plants to the dataset which then looks different for the network and thus helps the model to generalize well. Orientation and rotation should have no influence on the prediction of the algorithm.

**LAB color representation** The natural lightning of the sun changes over the day and depends on the season. Clouds and dust in the air also influence the lightning conditions of a scene. Since crops are planted outdoors and the single plant images are collected outdoors with a smartphone it is important that the variability of exposure and color values in the real world is also reflected in the training data. Data collected on a cloudy day can be modified in such a way that it looks more like a photo that was taken on a sunny day. Therefore the images are transformed into to LAB color spectrum where L is lightness and A,B defines the color of the pixel. The canvas of the Digital Field Twin is a ground image where plant images are placed on top of it. The lightness, thus the L channel, is multiplied by a random number between 0.5 and 1.5. The numbers are chosen in a way that the variation is large but the images still look realistically. The lightness of the plant images is randomly altered with consideration of the brightness of the background to avoid very bright plants on dark grounds. The following equation is applied to change the lightness of plant images:  $L = L_{PLANT} * (MEAN(L_{GROUND}) / MEAN(L_{PLANT})) * RANDOM(0.5, 1.35)$  The color values A and B of the plant images are slightly modified in the range [0.85, 1.15].

**Blur and Add Noise** Training datasets are often collected in a well designed setting. The camera is stable and the objects do not move so that there are no artifacts like blur or noise in the image. In nature, images like the ones taken on a smartphone are sometimes not captured with optimal parameters. Maybe the photographer has shaking hands so the camera moves. This could lead to a blurry image. Another error in the image could be caused by some dirt on the camera lens. Since these distortions are not part of a training set, which was collected under controlled circumstances, the neural network has a bad

## 5. Implementation

performance on images with errors. To close the gap between the image statistics of the natural dataset and the controlled dataset some errors are added artificially [VCS16].

### 5.2.2. Database

The images are identified by an UUID and stored in the file system. A SQLite database contains all information about the plant which is necessary to build a DFT with the plants. Each entry for a plant image has a unique UUID, a diameter in centimeter, an age in days and a species. The UUID is the file name to store the image. There is also a table for background images that stores basic information like the width in centimeter of the ground image and a note string.

A server provides an interface to the database so that the collected data from the smartphone can be send and stored to the datastore.

### 5.2.3. Separate Dataset

The DFT is build of images, showing a single plant, inserted in ground images. Annotation masks denote the class of each pixel by color. Normally the three classes ground, weeds and crops are color coded in black, red and green. The color coded annotation mask can be used to separate a annotated dataset into single plant images. Two different algorithms were implemented for dataset separation. Depending on the available information of the annotation step one of the algorithms works best. If additional information about the polygon, drawn by the annotator to border a plant, is included in the dataset, this can be facilitated for separation. The carrot dataset published by S. Haug and J. Ostermann is an example for a dataset with polygon information [HO15]. Otherwise a DBSCAN algorithm is used to separate and cluster the plants in the annotation mask.

**Components** The date set separation is structured as a pipe filter application. The separation step is either a DBSCAN followed by a separation of the detected clusters or information about the polygon drawn for annotating the image can be used to crop the plant instances. The cropped plant images are stored to the file system with information encoded in the filename. The ratio between the width of the original image and the cropped image, the age of the plant, the filename of the original image, the species and a sequential number per original image are part of the filename of the cropped plant images. The DB Connector stores the information about the cropped plant images to the DB and copies the images to a directory, where all plant images are stored, so that the image server can find and load them.

**Polygon** People classify the images by drawing a polygon that edges an area containing only one plant. Background segmentation is sometimes used to support the annotator by removing the background area from the marked area. This eases the annotation step, because fine details of plants can be ignored. The marked area is then colored accordingly. Some datasets (carrot) include the data about the points that form the polygon and the class of the plant in this area. The example images 5.1 show an annotation mask and the

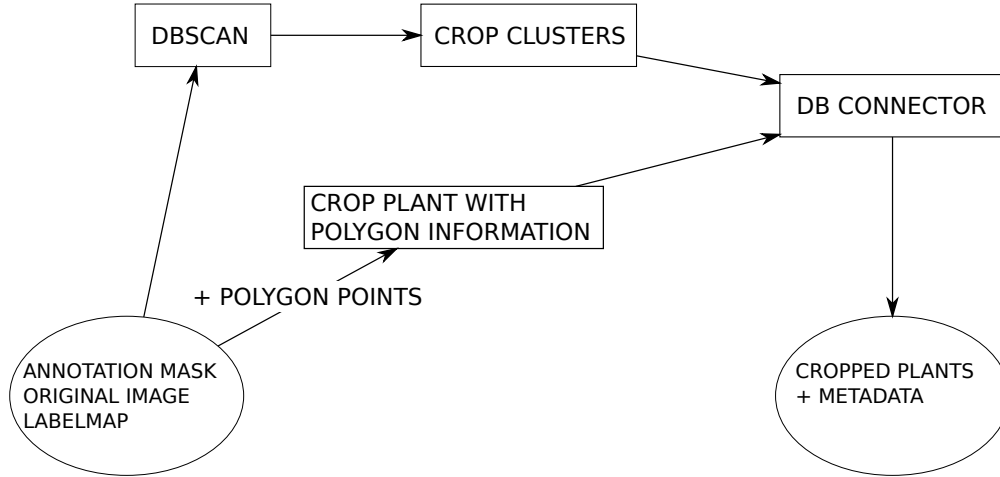


Figure 5.2.: Pipeline to Transform a Annotated Dataset Into a Well Structured Data Collection of Single Plant Images.

corresponding polygon. The example polygon has 14 edges therefore 14 calculation steps. In the lower right corner the annotation mask with the corresponding weed is shown. The algorithm constructs the black and white polygon image with values 0 and 1 from the list of points given with the dataset. Multiplying the polygon image (last b-w image) with the annotation mask cuts out the weed. The resulting image only contains one plant and is used to crop the plant in the original image. The first image in Table 5.2 shows the cropped original image. The black area is made transparent in the original image. The image returned by the algorithm shows only one class with transparent background. The most interesting part of this algorithm is how the polygon image is build from a list of points. The problem is named point in a polygon and is well described in literature [poi18]. The idea of this algorithm is to do ray tracing in X-direction of the image starting from each point and counting how often the ray crosses an edge of the polygon. If the crossing number is odd then the point is outside of the polygon, otherwise the point is inside. The implemented algorithm is based on the PNPOLY - Point Inclusion in Polygon Test from W. Randolph Franklin [poi18].

**DBSCAN** The DBSCAN is utilized to separate the plants, if no polygon information is present. The clustering method finds clusters of pixels with the same color and little special distance in the annotation mask. Nearby pixels of the same class, denoted as color, build up a cluster that correspond to a plant. Plants of the same class which are overlapping result in one cluster. Each cluster found by the DBSCAN is cropped in the original image and stored with transparent background. The image shown with the polygon example results in a different outcome when the DBSCAN is applied. On the upper side of the image a second weed is very close to the example weed so that the DBSCAN results in one cluster instead of two separate weeds. Clusters of the same plant are still very well suited to build a Digital Field Twin. Single parts of a plant like a single

## 5. Implementation

```

func Polygon( $p_1, p_2, sum_{mask}$ ):
     $im1 \leftarrow$  color the area between  $p_1$  and  $p_2$  white (upward_crossing,
        downward_crossing);
     $im2 \leftarrow$  color the area normal to the line defined by the points  $p_1$  and  $p_2$  in the
        direction of +x white;
     $mask \leftarrow$  combine  $im1$  and  $im2$  by a boolean AND operation;
     $sum_{mask} \leftarrow$  XOR  $mask$  with the  $sum_{mask}$ 
return  $sum_{mask}$ 
 $sum_{mask} \leftarrow$  black image;
for Point of Array  $P$  with index  $i$  up to  $i < size - 1$  do
    |  $sum_{mask} \leftarrow$  Polygon( $p_y(i), p_y(i + 1), sum_{mask}$ );
end
// close the polygon by adding the last segment between the last and
    the first point
 $sum_{mask} \leftarrow$  Polygon( $p_y.last, p_y.first, sum_{mask}$ );

```

Algorithm 2: Point in a Polygon

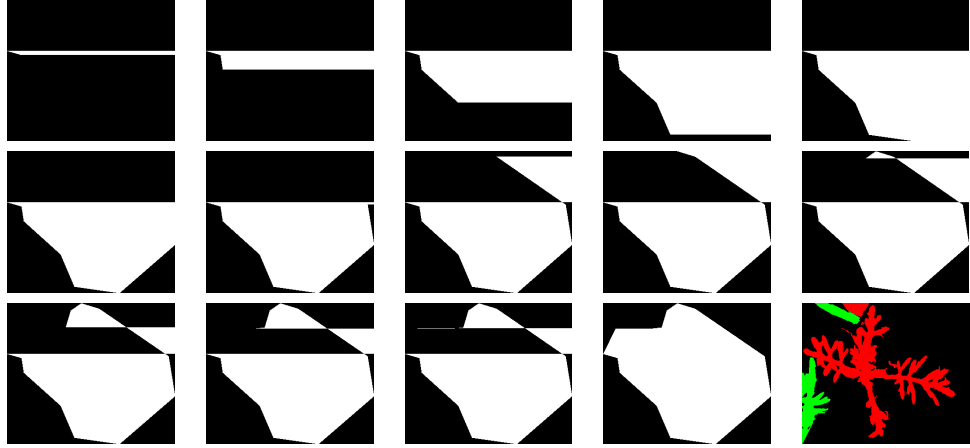


Table 5.1.: Point in a Polygon Example

leaf leads to less realistic generated field images. For this reason the parameters for the DBSCAN were chosen in a way that parts of plants which are not connected to the plants (e.g. because the stem is not annotated or covered by soil) are clustered together with the main plant. (see Figure 5.2)

Both algorithms output an image showing a single plant or few plants of one class with transparent background. These images can then be used to build up a Digital Field Twin.

### 5.2.4. Data Collection Workflow

The Data Collection Workflow enables farmers to collect new images and annotate them in the field. The process is designed in a way so that it can be executed in a web browser



Table 5.2.: Comparison of Dataset Separation DBSCAN and Polygon; Image Sources: 1,2: [HO15],3: [CLS<sup>+</sup>]

on a smartphone. The process is implemented in the process execution engine CPEE. The workflow engine orchestrates the teamwork of computers running the automatic tasks and humans doing the manual tasks. The following table lists the tasks which are part of the Data Collection Workflow with information about the executor and a description.

The services have a REST interface which is used to coordinate the services and transfer data between the services.

Name	Description	Task Type
Show Instructions	The first step is to instruct the user what to do. The scene has to be prepared in a manner that only one plant is in the image and the background is black. The ground and other plants are covered with black paper.	Human
Take Image	The second task is to take an image showing the black background and the plant.	Human
Crop Image	Trigger the cropping algorithm and wait till it has finished.	Automatic
Show Cropped Image	Show the cropped image to the user. The user gives feedback about the cropped image. If the cropped image of the plant is bad (e.g. a lot of background pixels are present) the user can start at Take Image again.	Human
Input Plant Attributes	Additional information about the plant is collected: age of plant, species, width in cm and notes.	Human
Store to Database	The ID of the collected plant image and the information about the plant is send to the database service.	Automatic

Table 5.3.: Data Collection Workflow Tasks

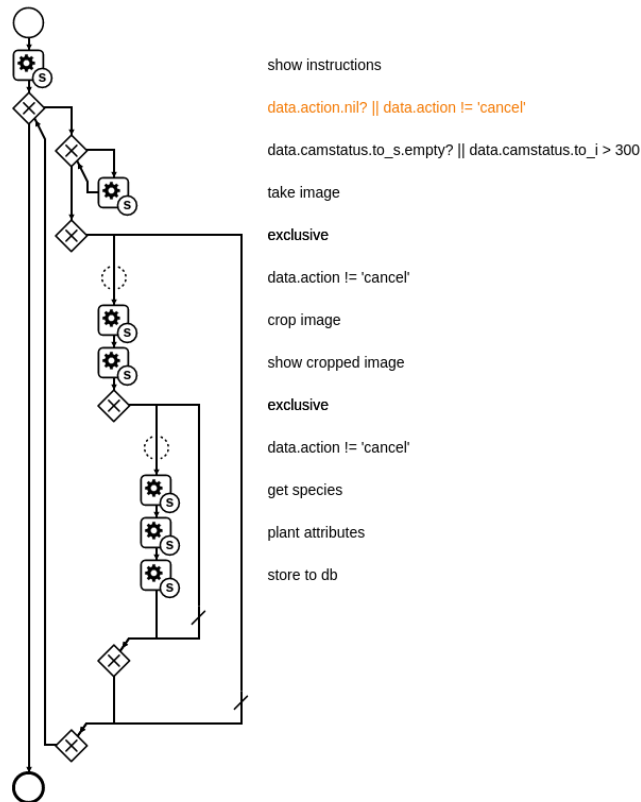


Figure 5.3.: CPEE Process Model of the Data Collection Workflow [cpe21]

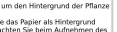
## Background Removal


The main part of the data collection workflow is the cropping algorithm, also called background removal algorithm. The goal is to extract the pixels showing parts of the plant and make all the background pixels transparent. In order to collect only single plants and make the background as homogeneous as possible a black paper is placed

Ziel dieses Prozesses ist es Trainingsdaten von Unkräutern und Nutzpflanzen mittels Smartphone zu sammeln.

Sie benötigen schwarzes Papier, um den Hintergrund der Pflanze am Foto zu schneiden.

Das nachfolgende Foto zeigt, wie das Papier als Hintergrund verwendet werden soll. Bitte beachten Sie beim Aufnehmen des Fotos, dass nur die Pflanze und schwarzes Papier sichtbar sind. Weiterhin werden folgende Daten erhoben: Alter der Pflanze, Art der Pflanze und Durchmesser in cm.





Species:

soy

Diameter:

10

Age (Days):

0

Notes:

kaliham soy field

START

Abbrechen

Fertig

Abbrechen

Fertig

Abbrechen

Fertig

Abbrechen

Show Instructions

Take Image

Show Cropped Image

Input Plant Attributes

Table 5.4.: User Interface Data Collection Workflow



under the plant. A few images were taken in which the background is made black with a black paper. All pixels of the plant were colored white and background were colored black, manually. GIMP was used to border the plant and color the background and plant in the corresponding color [gim21]. The manually annotated images were then used to find out which color spaces are best to determine a boundary to separate background pixels from plant pixels by color value. The background removal algorithm does not utilize location information of the pixels. Visual inspection on a scatter plot matrix of the color spaces HSV, RGB and LAB was done. Further, color indices for vegetation segmentation like CIVE, EXG, EXR and EXGR were investigated. These indices are used to find out the area of plants covering the ground in a satellite image or to follow the growing process in a field by taking images with drones and calculating the leave area over time [HLES18]. The manually classified images were used to create histograms for each color space. The class (fore- and background) was encoded in color. A Support Vector Machine was trained to find a linear plane which separates the two classes plant and background best. As a starting point visual investigation of the scatter plot matrix was used. The LAB color representation resulted in the best separating hyperplane. The benefit of a linear plane is that it can be easily expressed as a linear equation, which in turn can be implemented efficiently with parallel image processing libraries. The separating hyper plane is:

$$\begin{aligned}
 w &= [-0.015931472997181118, -0.3315611013490525, 0.24868342258628218] \\
 b &= -4.992750111484608 \\
 im_L &\geq (im_B * \frac{w_2}{w_0} + im_A * \frac{w_1}{w_0} + \frac{b}{w_0})
 \end{aligned} \tag{5.1}$$

where  $im$  is the image of a plant with black background in LAB color representation,  $w$  and  $b$  define the position and orientation of the separating hyperplane.

White paper was also investigated to make the background white, but it turned out that it is easier to find a boundary between background color and plant color when the background is black.

#### 5.2.5. Class Imbalance

This script calculates how imbalanced the different classes weed, crop and ground are within one dataset. The annotation mask where each pixel is colored according to the class is the input for this algorithm. Basically, the program counts the pixels which are red (weed), green (crop) and black (ground) and divides the number of pixels per class by the overall pixels of the image. The three percentages of pixels per class in an image are then added up over one dataset and divided by the number of images to get the average percentage per class of one dataset.

The DFT is created such that the classes weeds and crops are balanced over the dataset.

### 5.3. Model Training

The training process has the following activities: determine the plant and background images that will be used to generate the Digital Field Twin, create the DFT, copy the

## 5. Implementation

DFT to the training infrastructure, create a configuration file and monitor the training via the webinterface, namely Tensorboard. Subsequently, the activities involved in model training are described.

### 5.3.1. Plants and Background

Good predictions in machine learning can only be based on good data. Good means that the characteristics, for example the parameters that determine whether a part of an image is soil or plant, are the same in the training data as in the target data. This means that the data, especially the images of the plants, must be similar to the crops that are to be detected.

The database stores metadata such as age, size, species and additional notes of the single plants. The metadata is used to find the plants which are most similar to the target field. The crops and weeds in the training data, the age and size should be matched to the target field.

If the data originates from the same field, the similarity is most likely great and the quality of the training data good.

### 5.3.2. Generate Digital Field Twin

Based on the parameters from the previous step thousands of images are created with different combinations of ground, weed and crop. Additionally, transformations are applied on the images. The details about the Digital Field Generator is described in section Digital Field Generator 5.2.1.

### 5.3.3. Copy to Infrastructure

Due to the large number of parameters that have to be determined to minimize the error of the neural network many computational operations must be performed. Graphics processing unit are much better suited for the training of neural networks than conventional CPUs. For this reason, the training is carried out on a computer with a graphical processing unit from Nvidia: GeForce RTX 2070 SUPER.

### 5.3.4. Configuration File

A lot of trial and error runs are necessary to learn what parameters, training data or number of training data works best, because for every domain and use case this might differ. There is no recipe for a well performing model. There are guides about what to try and how to interpret the outcome. It is important in a trial and error scenario to have enough information about a training process so that it can be reconstructed, compared to others and most importantly, the results can be used to learn why one training session performed better than another. Important questions about the training process are: what kind of data was used to train this model, how long was the training or what was the starting model?

Name	Description
Name	This name is used to store the output of a training, the model, and log entries are entered under this name.
description	Short description what data is used and what is different to the the other runs.
data_directory	Path to the training data directory counting an image and annotation folder.
validation_directory	Path to a directory holding data that should be used for validating each epoch. If this parameter is empty 20% of the data in data_directory is used as validation data.
exp_directory	Path to the directory which holds the resulting models.
epochs	Number of epochs the training lasts.
batch_size	Number of images that could be processed on the GPU in parallel. This is limited by the memory of the GPU.
model	Path to a model that is used as a starting point for training. If empty, the Deeplabv3 resnet101 pretrained on COCO train2017 is used [CZP <sup>+</sup> 18, pyt21b, coc20, Min19].

Table 5.5.: Model Training Activities

The configuration file holds all the information that is altered over different training sessions. The config file is a YAML file.

```

name:          carrot_digfield_small_subset_10
description:    no model loaded; train on carrot_digfield_small_0-10_polygon
digital field twin created on 0-10 original carrot images; 2250 images;
validated on same dataset;
tested on the remaining 10 images from the dataset 51-60
data_directory: testdata/carrot_digfield_small/0-10_subset_polygon
validation_directory:
exp_directory:  outdata
epochs:        100
batch_size:    2
model:

```

### 5.3.5. Predict and Calculate Performance

The performance of the trained models on test data is determined by a script. The script takes either a single image or a directory with images and optionally the ground truth annotation masks. Images that are larger then 513 x 513 are cropped at a random position to the correct size. A parameter controls how often the image is cropped. If the ground truth is not given the script runs the semantic segmentation on the image or parts of the image, but does not determine its performance. The main performance metric is Intersection over Union as described in 4.3.4. Further, the overall pixel wise accuracy

## 5. Implementation

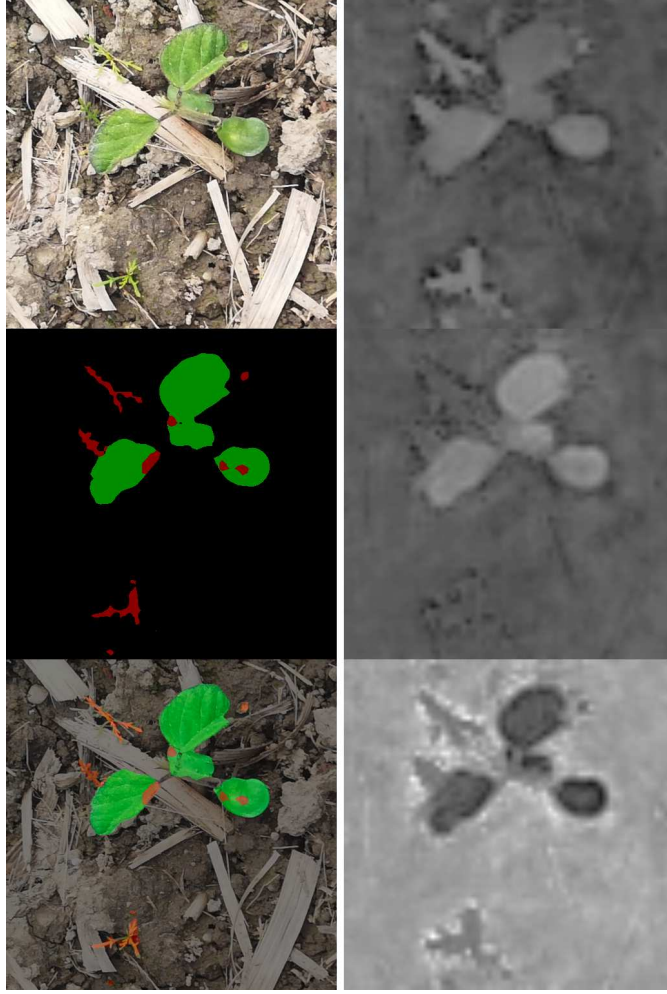


Figure 5.4.: Images Created During Semantic Segmentation

is calculated. The performance metric is stored in a CSV file. Each line in the result corresponds to one image with the columns: global pixel accuracy, IoU weed, IoU crop, IoU ground and the mean IoU. Moreover, a couple of images are stored while calculating the performance of the semantic segmentation. The ground truth annotation mask, the original image, the predicted annotation mask and a combination of the original image with the predicted annotation mask are stored to the result directory. The final class per pixel is determined by using the class with the highest activation, thus the brightest pixel. A black and white image encodes the activation intensity by the shade of gray of each pixel. The b-w images are stacked together. The upper most part shows the crop class, the middle part shows the weed class and the lowest part of the image gives the ground class. The b-w image gives an intuition about how sure the model was about the decision to assign a class to a pixel (see Figure 5.4).



Figure 5.5.: Segmented Drone Image of a Soybean Field

### 5.3.6. Predict Large Images

Images taken by a smartphone, autonomous machine or drone are larger than the 513x513px, which are the input size for the semantic segmentation model. Therefore a program is implemented to split up the larger image into several smaller ones, predict the small images and then recombine the images again. Plants that are cropped into the half by splitting up the large image into smaller ones are harder to segment for the model. The splitting is done two times with different offset to avoid that plants are cut off. Squares with a length of the edge of 513px are cut out. The first cropping starts at the top left corner at position 0,0, the square is shifted by 513 px in horizontal and vertical direction until the end of the image is reached or the remaining part is smaller than 513px in one direction. The second cropping starts with an offset of 256px in each direction, so at the position 256x256. The square is shifted  $x-1$  and  $y-1$  times by 513px in each direction where  $x$  and  $y$  are the number of shifts in  $x$  and  $y$  direction in the first cropping iteration.

The cropped images are segmented independently of each other and recombined by adding up the per class activation for each pixel, identified by the position in the original image. See Figure 5.5 for an example of an image taken by a drone on a soy field. The image was taken by a DJI Mavic 2 Pro on a soy field in Kallham on 01.06.2021. The model is trained on images collected by smartphone with the Data Collection Workflow on the same day at the same field.



## 6. Evaluation

This section includes the results of different training runs and the corresponding tests to evaluate the performance of the semantic segmentation model. Different techniques to collect and annotate data are compared to each other to find out how well the proposed Digital Field Twin performed compared to the state-of-the-art way.

### 6.1. Datasets

Neural networks learn from data. The data used for training and testing is described in the following paragraphs.

#### 6.1.1. Carrot

The Carrot dataset<sup>1</sup> is publicly available and is interned as a reference dataset to compare different crop/weed segmentation algorithms. The images are taken by a field robot. In total the dataset contains 60 images and shows 162 crops and 332 weeds. The weeds and crops are close together and do overlap sometimes.

All images are taken in the same row of plants. At the beginning of the row 20 images were taken. The remaining 40 images were taken in junks of 10 images approximately 12 meters apart.

This setup is well suited to test the use case described with the data collection workflow of collecting single plants on one end of the field, train a model on this data and run weed/crop detection on the rest of the field.

#### 6.1.2. Plant Seedlings Dataset

This dataset<sup>2</sup> contains about 960 different images of plants belonging to 12 species (Black-grass, Charlock, Cleavers, Common Chickweed, Common wheat, Fat Hen, Loose Silky-bent, Maize, Scentless Mayweed, Shepherd's Purse, Small-flowered Cranesbill, Sugar Beet) in different growth stages [GJJ<sup>+</sup>17]. The images are annotated, classified and are used to create a digital field twin. The weeds are particularly interesting since a traditional field consists of a variety of weeds in different growth stages and one crop sowed to grow and be harvested. Weeds are normally rare in a traditional field, since the goal of the farmer is to reduce the weeds so that the crops have a maximum of the light, nutrition and water that is available at the field. Therefore its good to have a big collection of weed images to build a DFT.

---

<sup>1</sup><https://github.com/cwfid/dataset>

<sup>2</sup><https://vision.eng.au.dk/plant-seedlings-dataset/>

## 6. Evaluation

Date	Location	Species	Number of Images
31.05.2021	Kallham	Soy	25
31.05.2021	Kallham	Weed	10
11.06.2021	Kallham	Soy	48
11.06.2021	Kallham	Weed	0
20.05.2021	Türkenschanzpark, Vienna	Weed	10

Table 6.1.: Data Collected with the Data Collection Workflow

### 6.1.3. University of Applied Life Science Vienna (BOKU)

The BOKU<sup>3</sup> in Vienna operates field trials at the Experimental Farm in Groß-Enzersdorf next to Vienna, Austria. The crops sugar beet and soy are sown and photographed in regular intervals with a self build machine. The device has wheels and can be pushed across the field by the operator. Two stereo cameras take pictures at a constant height. The images are manually annotated with the CVAT tool [ope22].

### 6.1.4. Self Collected

The data collection workflow is used to take photos<sup>4</sup> in the field and annotate the images. For the experiment a soy field is used and photographed at two different growth stages. A Huawei P20 was used to take the photos. Table 6.1 lists the date when the workflow was executed, the location, the species of the plant and the number of images.

## 6.2. Separate and Recombine Carrot Dataset

The cropped and separated plants can be extracted from already annotated data. The carrot dataset contains images, annotation step and information about the annotation step. The polygons to outline the plants are stored in a yaml file for each image. This allows the separation of the plants based on the polygons.

The experiment described in the following investigates how the digital field twin constructed from separated plants performs compared to the original data.

The carrot dataset is composed of 60 images. The last 10 images, which are taken at the end of the field, are used to test and compare the performance of trained models in the end. Different portions of the remaining 50 training images are used for training. For each portion of the carrot dataset one digital field twin is generated and a training set of the original images is used for training. In the first setting shown in table 6.4 a digital field twin is build based on the plants extracted from the first 10 images of the carrot dataset. The separation of the first ten images resulted in 10 carrots and 25 weeds. The 1-10 DFT is build of 35 separated plant and 18 ground images.

<sup>3</sup>Dataset provided by the Institute of Agricultural Engineering <https://boku.ac.at/en/nas/ilt>

<sup>4</sup><https://github.com/hoellermathias/DigitalFieldTwin>



## 6.2. Separate and Recombine Carrot Dataset

Dataset	Images	#Carrots	#Weeds	#Grounds
1 - 10 Subset Carrot	10	25	72	18
1 - 20 Subset Carrot	20	54	109	28
1 - 30 Subset Carrot	30	77	158	28
1 - 40 Subset Carrot	40	107	225	30
1 - 50 Subset Carrot	50	132	254	28

Table 6.2.: Training Subsets of Carrot Crop/Weed Dataset

The training images have the dimension of 513x513px. 2250 images are generated for the digital field twin. The model is trained on 1800 images of the training data and validated on the remaining 450 images for 100 epochs. The model with the best mean intersection over union score on the validation dataset is selected and used for determining the performance on the test set. The test results, meaning the performance on the last 10 images (51-60), showed that the recombination of plants with different backgrounds and a couple of transformations lead to good training data. The test showed that there is a significant increase in the Intersection over Union score (IoU) from the 1-10 subset (0.6) to the 1-20 subset (0.71), but that there is hardly an increase by adding more plant images to the 1-20 subset. The subsets 20, 30, 40 and 50 all showed a performance ( $> 0.70$ ) slightly higher than the score reached with a Random Forest classifier in the paper which presented the dataset (0.67).

This experiment showed that it is possible to reach good semantic segmentation results with a Digital Field Twin build on single, separated plant images inserted into ground images.

Table 6.5 shows the intersection over union score of the models trained on Digital Field Twins build of different subsets of the carrot dataset. The diagram in the first row on the right displays the cross entropy loss of the 1-20 Subset DFT. The diagrams in the rows 2,3 and 4 show the IoU score of the training set on the left side and the validation set on the right side.

Training time of the runs shown in table 6.4 is more or less 24 hours, because the number of epochs is fixed to 100. The charts of the training error and the charts showing the IoU per class (weed, crop, ground) flatten after around 20-30 epochs of training. This indicates that the parameters of the network are hardly changing and thus learns little. Stopping the training in an earlier epoch could drastically reduce the training time. The data indicates that the time could be reduced to 1/3 - 1/5 of the current total training time of nearly 24 hours. The model trained for 4,5 hours on the 1 - 20 subset results in an mean IoU on the last 10 images of 0.69. This performance is 4% lower than the mean IoU of the model trained for 24 hours.

### 6.2.1. Carrot Dataset with DBSCAN

Datasets do not always contain the information about the polygon drawn by the annotator to outline a plant. A clustering algorithm can be used to group the pixels belonging to one

## 6. Evaluation

plant together. The carrot dataset is separated with both methods: separation using the polygon data or the DBSCAN algorithm. To compare the performance of the DBSCAN algorithm to the polygon annotation the same data in both setups was separated and recombined. The 1-20 subset, the first 20 images of the dataset, were separated and resulted in 48 crops and 79 weeds. Compared to the separation based on the polygon annotation the number of single plants is lower by 22%. This is due to plants which are very close to each other or even overlapping. This plants are clustered as one plant and leads to a weaker separation of the individual plants. The parameters for the DBSCAN algorithm are tuned such that the outputted plants are complete. Complete means that leafs which are not connected to the center of the plant, because the stamp is classified as background, are part of the same cluster thus the same plant.

The model trained on a Digital Field Twin generated on the plants extracted with the DBSCAN from the 0-20 subset of the carrot dataset performed only slightly worse then the polygon separated data. The mIoU score of the model trained on the DBSCAN data tested on the last 10 images (51-60) reached 67%, which is circa the performance of the reference algorithm presented by S. Haug and J. Ostermann [HO15].

The clustering algorithm is a good alternative to separate an annotated dataset, if there is no polygon information given with the dataset.

### 6.3. Data Collection Workflow

Collecting data in the field and testing the trained model on images taken on the same field, just a couple of meters apart from the data collection spot, is close to a real world use case. The benefits are that the training data and the target domain, so the rest of the field, are close and look similar. The plants are in a similar growth state and typically the same sort of seeds were sown. Train neural networks is more promising if the training domain is close to the target domain. This experiment shows that a CNN trained on synthetic images generated with data collected by the data collection workflow shows good results.

The data is collected on the same soy field on different days thus different growing stages. The data was taken on the 31.05.2021 and on the 11.06.2021 on a soy field. Three Digital Field Twins are build with weeds from the plant database published by Mosgaard Giselsson et. al and different self collected soy plants [GJJ<sup>+</sup>17]:

- 25 soy crop images taken on the 31.05.2021
- 25 soy crop images taken on the 11.06.2021
- 50 soy plants, 25 plants form the 11.06.2021 and the 31.05.2021 each

#### 6.3.1. Advantages over State of the Art Approach

Ground Truth images with little errors in the training data are a prerequisite for a model doing good predictions. Poor data leads to a poor prediction. If the image of fields are

poorly annotated and an algorithm is trained on this data a good segmentation of crops, weeds and ground is impossible. Therefore, the annotation step is crucial for the success of machine learning projects. In the traditional setup, where the images are annotated by hand, it is often hard, especially when plants overlap, to find the border between a crop and a weed. The method proposed in this paper only collects single plants without the need of drawing a border between two plants.

Annotating images showing a field with plants is considered as boring and time consuming.

#### 6.3.2. Self Collected Soy Dataset 31.05.2021

The performance (IoU score) of the model trained on the DFT 31.05.2021 tested on 11.06.2021 test data is 0.54.

The test data images of the field are collected on the same field but at another spot, so that the plants are similar but not identical to the training data. Figure 6.7 shows some example images. The first row shows the input image. The second row shows the manually annotated ground truth masks. The third row shows the prediction of the model trained on a Digital Field Twin of 25 soy plants and hundreds of weeds. The IoU score is given in the last row for the comparison of the ground truth and the prediction. The average IoU score of the five predictions is 0.81.

The average IoU score over all classes of the test data collected on the 31.05.2021, thus on the same day as the training data, is 0.71. The per class IoU scores for weed, crop and ground are 0.4, 0.73 and 0.99. Nearly all of the pixels showing ground are labeled correctly. The soy plants are labeled with high precision in most of the cases but some of the images have low IoU scores. 11 of 19 test images have a IoU score for crop class over 0.9. Two images show only weeds and thus have a default IoU score of 0 for the missing class. If the average IoU of the crop class is corrected by the two images without the class crop the performance is even higher at 0.83. Three other images have a low IoU score of class crop (0.23, 0.32, 0.44). These images have in common that either parts of the plants are eaten by vermin or are covered by soil or two plants are overlapping. This significantly alters the appearance of a soybean plant and makes it difficult to assign the correct label to the crop pixels. The class weed has the worst IoU score with 0.4. A reason for this low value is that some image do not contain weeds, thus have 0 IoU score for the weed class. The corrected value is only slightly higher with 0.47. Ground pixels are very well classified. The weeds in the test set are small and hard to detect. Misclassification of the classes weed and crop are mostly misclassified among each other. Soy plants which are not detected as such are then misclassified as weed. The low IoU score of the weed class shows that the labeling is not accurate, so the annotation mask and the ground truth mask have less than 50% in common in term of the pixels labeled as weed. After manual determination of the weeds detected, it was found that a large proportion of the weeds had been identified but the marked area does not congruent. Only 2 out of 42 were not detected at all. Weeds that were partially marked as such were counted as detected.

This experiment shows that the temporal difference (same day or 12 days later) makes a huge difference to the prediction performance.

## 6. Evaluation

### 6.3.3. Self Collected Soy Dataset 11.06.2021

A DFT is generated based on the weeds from the Seedlings dataset and the soy plants collected with the collection workflow on 11.06.2021 at a private field in upper Austria. The digital field twin is build of 48 unique soy plants and 2632 different weed images resulting in 2250 images with a size of 513x513px. 1800 images are used as training set. The remaining 450 images build up the validation set. The performance is tested on seven hand annotated images showing soy and weeds collected on the same field where the single plants were collected but on a different spot in the field. The IoU score of the model trained for 20 epochs and around 4 hours is 0.7. The performance increased to 0.73 after 100 epochs and 24 hours of training.

Figure 6.8 shows five test images with the corresponding original image, segmentation mask superimposed over original image, ground truth and predicted segmentation mask. The number under every column shows the intersection over union score per image.

### 6.3.4. Kallham Combined

The results of the Kallham Soy DFTs showed that the proposed data collection workflow combined with the Digital Field Twin generator works well when the target domain is very similar to the training domain. The Kallham Combined DFT was build with soy plants from 11.06.2021 and 31.05.21. This should show how the performance changes if the training and target domain get larger and more diverse. In this case the dimension age of the crop plant is increased by one in the train and target domain. The soy plants in the training data and the test data are collected on different days on the same field. The model was trained for 100 epochs on a DFT containing 2250 images including 50 different soy plant images, 25 taken on 11.06.21 and 25 taken on 31.05.21. 32 different ground images and 2887 images of weed plants are used in the generation process. The performance was tested on the test data from the experiments described above (11.06.2021 Self Collected and 31.05.2021 Self Collected).

The IoU score on the 31.05.21 test set is 0.63 and on the 11.06.21 test set it is 0.66. Both are worse than the models trained only on the soy plants of one day.

### 6.3.5. Generalization of DFT

To examine the generalization of the models, trained on a DFT build with data collected by the Data Collection Workflow, to other unseen soy fields, the models were tested on images of soy plants from the University of Natural Resources and Life Sciences, Vienna. The results showed that the models do not generalize well and reach little IoU score on other fields. The IoU score for the crop and weed class calculated on 10 images of the BOKU soy dataset is 0 for both models (310521 and 110621).

## 6.4. BOKU Dataset

### 6.4.1. BOKU Soy Tested on Kallham Soy

A model trained on manually annotated images of soy plants and weeds, collected by the University of Natural Resources and Applied Life Sciences in Vienna (BOKU), is compared to a model trained on a DFT build with the same plants extracted of the same dataset. The original data is separated by the DBSCAN Data Separation script. The cropped plant images (weeds and soy) and meta data are stored to the database.

The soy dataset contains 88 images taken 16, 20, 27, 34 and 43 days after sowing. The number of images per day is: 40, 10, 22, 13 and 3. Both training sets, the original dataset and the DFT, consist of 2250 images with a resolution of 513x513px contain 77 soy plants, 99 Avena fatua plants and 34 unknown weeds.

The original training data is created by cropping the original annotation mask and images to the size of 513x513px. The length of the edges of the square cropping window is randomly selected from the range of  $[0.5, 1.5] * 513\text{px}$  and resized to 513x513px in the end. The cropped images are flipped horizontally and vertically with a probability of 30%.

Images showing only ground are not suited to train the model to distinguish between the classes crop, weed and ground. Therefore the cropped image is only stored and added to the training set if more than one class is visible in the part of the image. The areas occupied by the classes differ greatly. The percentage of the area showing weeds, crops and ground are 0.006%, 0.4%, 99.6%. The training set created by cropping the original dataset is slightly better class balanced with the portion of data per class weed, crop and ground: 0.32%, 2.2%, 97.47%. The generation of a DFT is described in detail in section 5.3.2. The resulting DFT has a class ratio of 1.9%, 4.2%, 93.9%.

A third model is trained on a combination of the original cropped data and the generated DFT. The combined training set contains 50% of the images of the other training sets. The class balance is 1.13% weed, 3.18% crop and 95.69% ground. All models are trained for 50 epochs. The models are tested on the 310521 and 110621 test dataset which was collected via smartphone camera on a field in Kallham. Table 6.9 shows the results on the test set after 50 epochs of training. The bar chart 6.1 shows that on both soy testsets the DFT performs better than the original data. The best accuracy on both test sets was reached by the model trained on a combination of original data and the DFT. The low score in weed segmentation of all variants may be due to the differences between the weeds in the training set and the test set. Only weeds from the BOKU dataset were used for the DFT generation.

This experiment showed that the model trained only on the cropped original data performs far worse (0.4 original - 0.7 DFT, 0.3 original - 0.5 DFT) than the DFT or the model trained on the combined dataset when it is used to predict images from a another field (Kallham 110621 and 310521). The accuracy of the model trained on the DFT build on the BOKU soy data is similar to the model trained on the data collected via the Data Collection Workflow in Kallham.

## 6. Evaluation

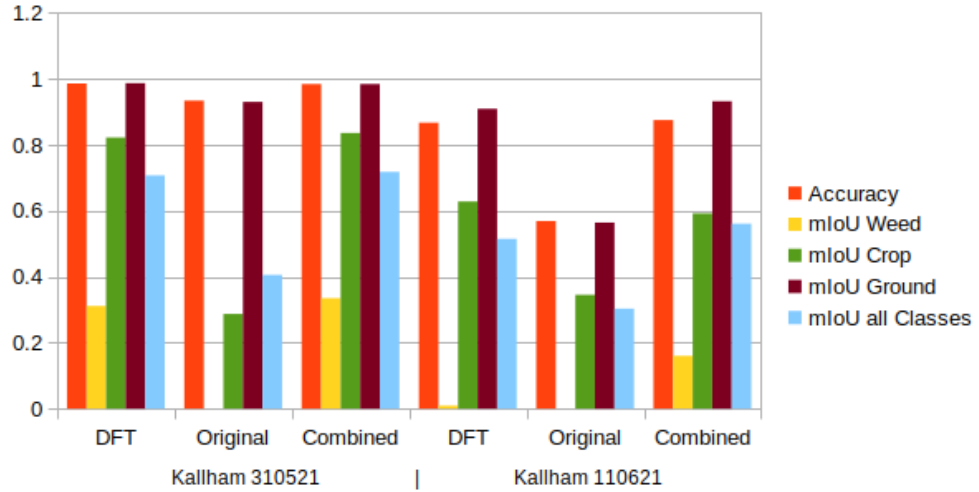


Figure 6.1.: Test results of the Models Trained on the BOKU Soy Dataset Tested on the Kallham Soy Datasets.

### 6.4.2. BOKU Sugar Beet

Models are trained on the sugar beet 2020 dataset, collected and annotated by the University of Natural Resources and Applied Life Sciences in Vienna (BOKU). One model is trained on a DFT build on the plants cropped from the original dataset. The second model is trained on 2250 images cropped to the size 513x513px from the original data. The performances of the trained models are compared by using the same test sets to determine the IoU score on different images for each model.

The main difference of the datasets are the backgrounds. The DFT generator algorithm inserts plant images into images of ground without plants. The background images are collected on different days and in different places. All background images stored in the database are used for the generation of the DFT. In total 44 different ground images are used in the DFT. A lot of different background images should enable the model to learn a general understanding of ground. The model should be more robust to shades, different lightning conditions, stones or straw lying on the surface.

The original sugar beet dataset collected in 2020 contains 267 images with a resolution of 1936px X 1216px. The images were collected in a sugar beet field 16, 20, 27, 34 and 43 days after sowing. 19, 62, 80, 81 and 25 images are included in the dataset for the corresponding day of collection. The separation of the dataset results in 1419 cropped plant images where 292 are sugar beet plants. Both models are trained for 50 epochs.

The model are tested on 30 sugarbeet images taken on the experimental farm of BOKU. Table ?? shows the results on the sugarbeet images taken on 20.05.2021. Pixel wise accuracy over all classes and Intersection over Union score for each class and averaged over all classes are shown in the diagram. Models tested on data from BOKU taken on 20.05.2021 in a sugarbeet field and sugar beet images from the University of Bonn. The

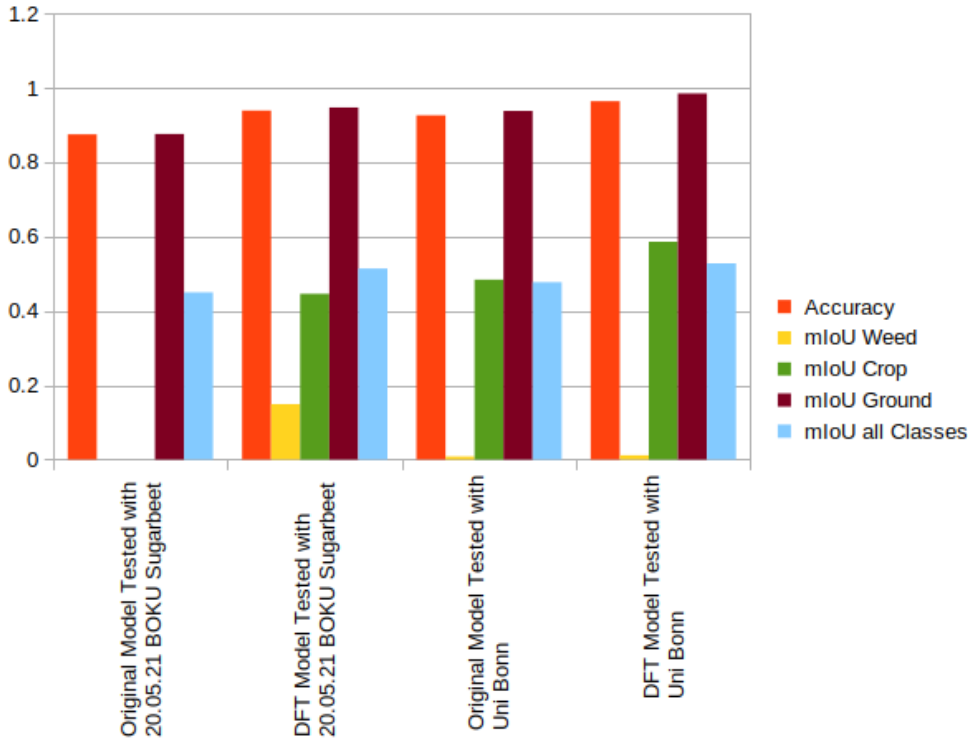


Figure 6.2.: Results of Semantic Segmentation of a Model Trained with BOKU 2020 Sugarbeet Data DFT and Original

DFT trained models perform better compared to the original counterparts but all models cannot accurately segment the images. The model trained with the DFT can assign all classes better than the model trained with the original images. With the mean IoU score of 0.51 DFT and 0.45 original, both models have a large error and cannot reliably segment the images.

The test on the sugar beet data from the University of Bonn showed a similar picture as the BOKU data. The DFT model is slightly better but still not able to segment the image with high accuracy (mIoU of 0.5). It is noticeable that the ground is recognized much better by the models trained with DFT. The influences of different soils on the performance of the model can be reduced using a DFT, but there are still some errors related to soil misclassification.

## 6. Evaluation

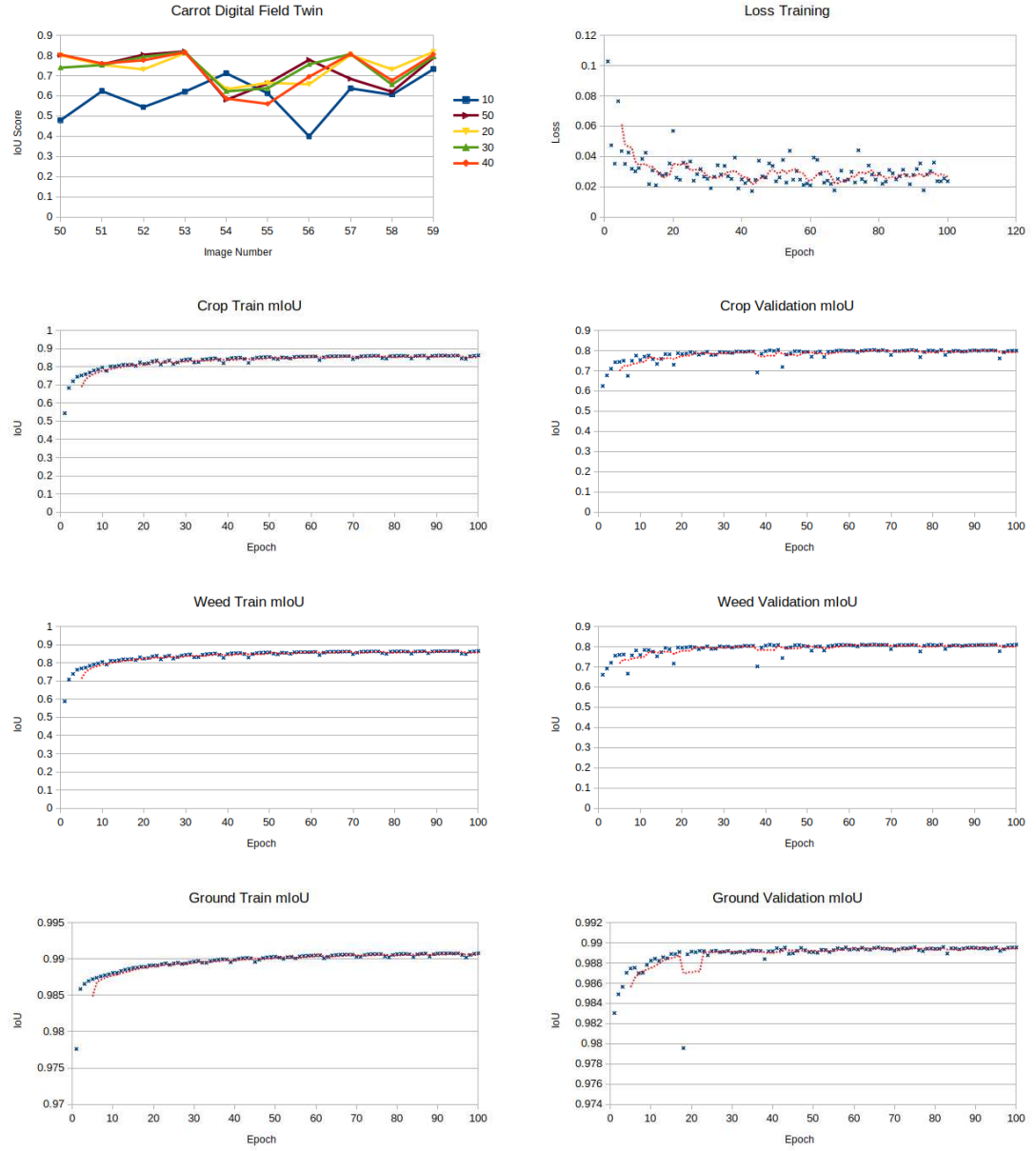


Table 6.3.: Training with the Carrot Dataset



Dataset	Best Epoch	Total Training Time	mean IoU
1 - 10 Subset	72	23:25:13	0.59
1 - 20 Subset	66	23:33:41	0.71
1 - 30 Subset	98	23:34:26	0.73
1 - 40 Subset	97	23:30:59	0.72
1 - 50 Subset	99	23:14:21	0.72

Table 6.4.: Results of Models Trained with DFTs Generated with Different Subsets of the Carrot Dataset

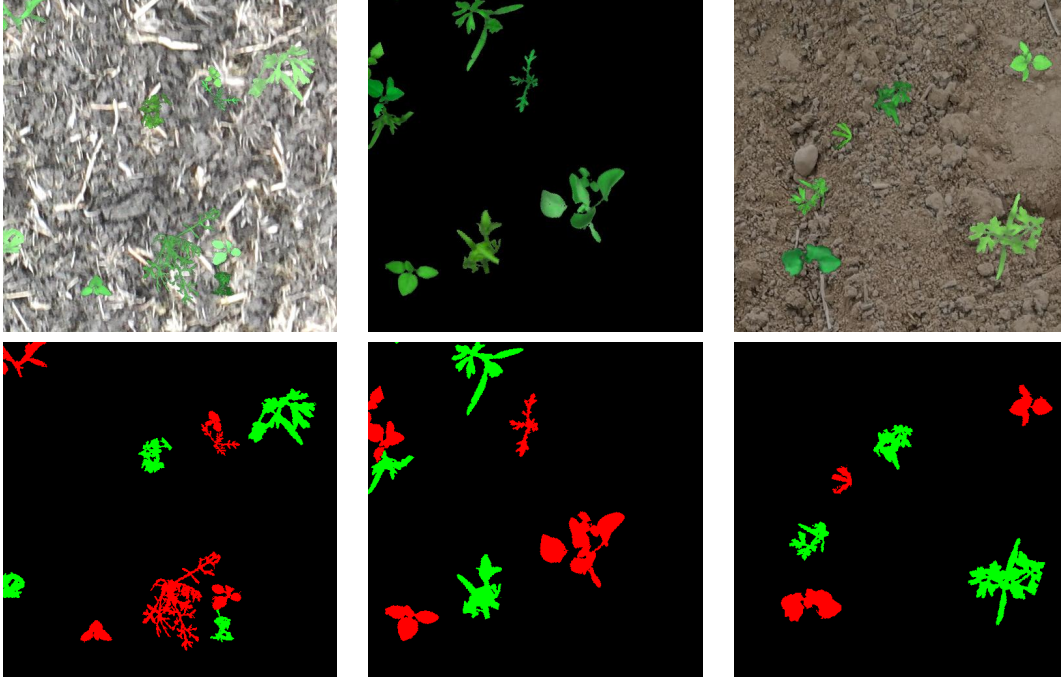


Table 6.5.: Images of 1-20 Carrot DFT



Table 6.6.: Example Images of the Digital Field Twin Build of Soy Plant Images Taken 31.05.21 and Weeds from the Seedling Database [GJJ<sup>+</sup>17]

## 6. Evaluation

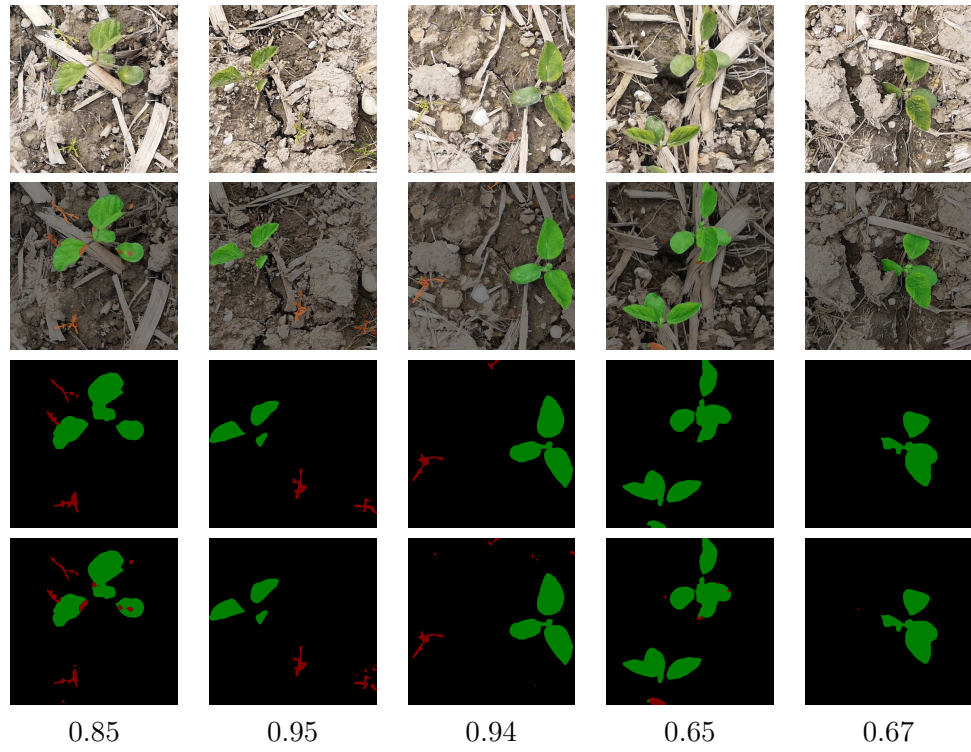


Table 6.7.: Example Images of the Test Dataset of Soy images taken on the 31.05.2021 on a Farm in Kallham, Upper Austria.

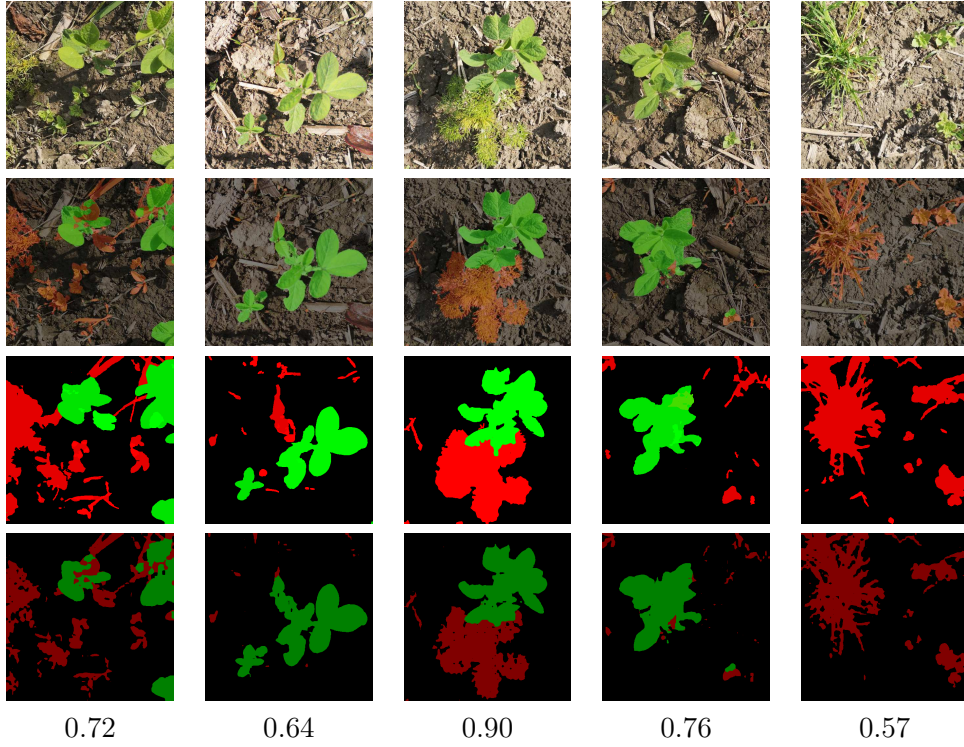


Table 6.8.: Test Set Predictions with the Model Trained with Self Collected Soy Dataset  
11.06.2021

Name	Test Data	Accuracy	mIoU Weed	mIoU Crop	mIoU Ground	mIoU all Classes
DFT	310521	0.986	0.311	0.822	0.987	0.707
Original	310521	0.934	0	0.287	0.93	0.405
Combined	310521	0.984	0.335	0.836	0.984	0.718
DFT	110621	0.867	0.009	0.628	0.909	0.515
Original	110621	0.569	0	0.345	0.564	0.303
Combined	110621	0.875	0.160	0.592	0.932	0.561

Table 6.9.: Test Results of the Models Trained on the BOKU Soy Sataset Tested on the Kallham soy datasets.



## 7. Discussion

Neural networks perform best in stable environments. This is the reason why autonomous driving still encounters great difficulties. The machine learning model has to segment the scene in order to decide for example if the car should break because of an obstacle. However, there are a lot of different and currently changing parameters in a street scene like different types of cars, trucks, busses, pedestrians, buildings or bicycles.

A team from the Max Planck Institute for Intelligent Systems has shown that even small changes in the image can cause the failure of the algorithm. The team designed a sticker that confuses a neural network trained to segment street scenes such that the model is more or less out of order. Designing robust, fail safe deep learning algorithms is very difficult. A small sticker on the T-shirt of a pedestrian does not disturb the human perception system, while an autonomous car can be very negatively affected [GK21].

An algorithm that distinguishes weeds from crops has significantly lower requirements for robustness or accuracy compared to a system for autonomous cars. Missing a weed or misclassifying a crop plant as a weed is of little consequence, at worst it will reduce the yield of a field. In the context of autonomous cars, misinterpreting a situation such as overlooking a pedestrian or classifying a plastic bag as a solid obstacle can lead to serious accidents. For this reason, agriculture offers a good environment to apply machine learning safely and with added value.

A field is a very inert system that does not change abruptly over time. Plants usually do not look noticeably different in one day. Only environmental influences such as sunlight or rain can change the appearance of the scene and might make it more difficult for neuronal networks to assign the classes correctly.

However, data augmentation can help to simulate the effects of different influences on the fields, e.g. solar radiation. The system presented in this project to create the Digital Field Twin changes the size, the colors and the brightness of the images (ground and plants) to adapt the training data so that the trained model can better cope with differences regarding the ground or the lighting.

### 7.1. Research Question 1

**How do weed/crop segmentation models differ in terms of the similarity to the ground truth when trained on original data or generated data?**

Research projects showed that synthetic data and data augmentation improve the performance of weed/crop detection models [CPGP, FPP<sup>+</sup>]. The algorithms to create synthetic fields are based on a mathematical model of a plant or Generative Adversarial Networks. Both techniques are able to extend a training dataset so that it performs

## 7. Discussion

better on unseen test data. However, the algorithms are complex. Adding new plants requires the design of a new mathematical model of a plant. In contrast, the approach presented in this work is a simple image manipulation algorithm. Creating a Digital Field Twin with new crops or add new weeds is possible without adapting the algorithm. The experiment with the carrot dataset and the model trained on the soy data from BOKU tested on an image taken on another soy field showed that the model trained on the DFT performs significantly better than the model trained with the original data.

### 7.2. Research Question 2

**How does the performance of a semantic segmentation model trained on a Digital Field Twin compare to manually annotated training data?**

Andrew Ng suggests to choose validation and test set from the same distribution representing the target domain best [Ng18]. Similarity of training and target domain makes it easier to train a model with high accuracy. As mentioned above, fields are stable and are therefore a good environment for machine learning. Fields with the same crop, the same sowing time and similar weather conditions have a high degree of similarity. A field is quite homogeneous in these parameters like temperature, precipitation or sunshine hours. As shown in the experiment with the carrots dataset, it is easily possible to train a model with images taken on one side of a field, which can segment the images of the plants on the other end of the field with high accuracy ( $> 0.70$  mIoU score).

The Digital Field Twin is built from single plants that are loaded from a database considering particular metadata (e.g. age of the plant). Training data can be generated in a way so that it is similar to the current state of the field. It can be adjusted, which weeds occur, how old the crops are or where the plant images were taken. An individual model can be trained for this field and does not have to work for all ages of a crop. This simplifies training and increases accuracy.

In order to further increase the similarity of the training data with the target data, new data can be recorded and annotated directly at the field. The average duration to annotate an image of a field is between 5 and 30 minutes according to [CPGP]. Annotating fields of weeds and crops requires trained annotators and a computer with appropriate software. The workflow presented in this work is executed on a smartphone and needs about 1 minute to collect and annotate one new plant. The data collection workflow results in well annotated, finely bordered plants, which remain complete. In contrary, extracting images from annotated datasets has the disadvantage that sometimes the plants are cut off at the edges of the image or are not perfectly separated due to clustering of the annotation mask.

The model trained on 25 images of a soybean plants in Kallham collected via smartphone, in combination with weed images from a dataset collected in Denmark, achieved high accuracy ( $>0.7$  mIoU) in semantic segmentation of images of the same soybean field in Kallham.

### 7.3. Research Question 3

**How is the semantic segmentation accuracy influenced by the age of the plants?**

The digital field twin created with 25 images of soybean plants collected at Kallham performed well with images taken in the same field and on the same day. Performance dropped from 0.71 mIoU to 0.54 mIoU when the model was tested with images taken 11 days later in the same field. This shows that the age of the plants has a large effect on prediction accuracy.

For targeted weed control, it is necessary to decide where exactly a weed grows. As can be seen in some example pictures in Table 6.8, some parts of a plant are classified partly as a crop, partly as a weed. This problem must be solved so that the algorithm can be used as a basis for targeted weed control. The environmental impact is reduced with the DFT by using a variety of data augmentation techniques, but remains a challenge. Unstable solar radiance or unknown ground can lead to significant losses in classification accuracy. Furthermore, the economic benefit is only given if the savings exceed the additional costs for the technical effort. Vision-based weed detection systems require cameras, processing units, and an actuator (e.g., hoe or individually controllable valves). These high-tech machines are much more complex and therefore more error-prone than systems currently in use, which process the entire field uniformly.





## 8. Conclusion

Agriculture is caught between the increasing demand for agricultural products and the desire to make production methods more environmental and climate friendly. Machine learning has the potential to make the production of our food greener. Targeted actions with effect on single plants instead of broadcast treatment of an entire field would save resources like pesticides and fertilizer. Proper actions in the field require decisions like: is a plant sick, what nutrients are missing or is it a weed?

In this context, data to train a neural network in good quality and sufficient quantity is needed. The separation of field images into single plants and the recombination of the cropped plant images can increase the power of a training dataset for weed/crop detection. The Digital Field Twin uses data augmentation techniques and different ground images to diversify the dataset. The model trained on a Digital Field Twin is even better in semantic segmentation of unseen field images as a model trained only on the original data.

The proposed Data Collection Workflow simplifies the collection and annotation of new plant images. The workflow can be executed for collecting plants on a part of the field via smartphone by a farmer. The cropped images can be recombined to a DFT and used to train semantic segmentation model for crop/weed detection. An experiment on a soy field showed that the model performs well on other parts of the same field. The advantage of this approach is that the model is trained for a specific field at a certain point in time and is therefore simpler to train. The tests also showed that the model only provide good results on the same field but other fields are only very poorly segmented. Further research would be needed to find a good combination of similar plant images from a database and newly collected plants. A community could use there smartphones to populate a database with images of there fields and annotate them right away. This database could help other farmers to make the generation of a Digital Field Twin even easier and more accurate.



# Bibliography

- [AHIS] Muhammad Asif, Samreen Hussain, Amber Israr, and Muhammad Faraz Shaikh. A vision system for autonomous weed detection robot. 2.
- [ANS19] André Araujo, Wade Norris, and Jack Sim. Computing receptive fields of convolutional neural networks. *Distill*, 2019. <https://distill.pub/2019/computing-receptive-fields>.
- [BIN] John BINNS. Farm to fork strategy. [https://ec.europa.eu/food/farm2fork\\_en](https://ec.europa.eu/food/farm2fork_en).
- [CLS<sup>+</sup>] Nived Chebrolu, Philipp Lottes, Alexander Schaefer, Wera Winterhalter, Wolfram Burgard, and Cyrill Stachniss. Agricultural robot dataset for plant classification, localization and mapping on sugar beet fields.
- [CNN18] Convolutional Neural Network (CNN). <https://developer.nvidia.com/discover/convolutional-neural-network>, Oct 2018. [Online; accessed 2. Nov. 2021].
- [coc20] COCO - Common Objects in Context. <https://cocodataset.org/#home>, Dec 2020. [Online; accessed 10. May 2021].
- [Col18] D. Colliaux. Demonstration of a 3D image segmentation application for weeding. <https://media.romi-project.eu/documents/index.html>, 2018. [Online; accessed 10. May 2021].
- [con21] Conda. <https://docs.conda.io/en/latest>, Sep 2021. [Online; accessed 2. Nov. 2021].
- [COR<sup>+</sup>16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [cpe21] CPEE - The Engine. <https://cpee.org>, May 2021. [Online; accessed 10. May 2021].
- [CPGP] M. D. Cicco, Ciro Potena, G. Grisetti, and A. Pretto. Automatic model based dataset generation for fast and accurate crop and weeds detection.

## Bibliography

- [CRH<sup>+</sup>19] Robin Chan, Matthias Rottmann, Fabian Hüger, Peter Schlicht, and Hanno Gottschalk. Application of decision rules for handling class imbalance in semantic segmentation, 2019.
- [CZP<sup>+</sup>18] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.
- [EVGW<sup>+</sup>] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
- [FM17] Emiko Fukase and Will Martin. Economic growth, convergence, and world food demand and supply, 2017.
- [FPP<sup>+</sup>] M. Fawakherji, C. Potena, I. Prevedello, A. Pretto, D. D. Bloisi, and D. Nardi. Data augmentation using GANs for crop/weed segmentation in precision farming. In *2020 IEEE Conference on Control Technology and Applications (CCTA)*, pages 279–284.
- [gim21] GIMP - GNU IMAGE MANIPULATION PROGRAM. <https://www.gimp.org>, Jul 2021. [Online; accessed 26. Aug. 2021].
- [GJJ<sup>+</sup>17] Thomas Mosgaard Giselsso, Rasmus Nyholm Jørgensen, Peter Kryger Jensen, Mads Dyrmann, and Henrik Skov Midtby. A Public Image Database for Benchmark of Plant Seedling Classification Algorithms. *arXiv*, Nov 2017.
- [GK21] G. Gigerenzer and H. Kober. *Klick*. C. Bertelsmann Verlag, 2021.
- [Gon18] Rafael C Gonzalez. *Digital image processing*. Pearson, New York, NY, fourth edition. edition, 2018.
- [Goo16] Ian Goodfellow. *Deep learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts London, 2016.
- [goo21] Learning to Predict Depth on the Pixel 3 Phones, May 2021. [Online; accessed 10. May 2021].
- [hai21] Papers with Code - Real-time deep hair matting on mobile devices. <https://paperswithcode.com/paper/real-time-deep-hair-matting-on-mobile-devices>, May 2021. [Online; accessed 10. May 2021].
- [HLES18] Mohamed Hassanein, Zahra Lari, and Naser El-Sheimy. A New Vegetation Segmentation Approach for Cropped Fields Based on Threshold Detection from Hue Histograms. *Sensors (Basel)*., 18(4), Apr 2018.

- [HO15] Sebastian Haug and Jörn Ostermann. A crop/weed field image dataset for the evaluation of computer vision based precision agriculture tasks. In *Computer Vision - ECCV 2014 Workshops*, pages 105–116, 2015.
- [Hoe22] M Hoeller. DigitalFieldTwin. <https://github.com/hoellermathias/DigitalFieldTwin>, Feb 2022. [Online; accessed 2. Feb. 2022].
- [ibm21] What is Machine Learning? <https://www.ibm.com/cloud/learn/machine-learning>, Aug 2021. [Online; accessed 12. Aug. 2021].
- [iwm21] Integrated Weed Management: PRActical Implementation and Solutions for Europe. <https://iwmpraise.eu>, May 2021. [Online; accessed 11. May 2021].
- [KB14] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv*, Dec 2014.
- [LB20] Bo Liu and Ryan Bruch. Weed Detection for Selective Spraying: a Review. *Curr. Robot. Rep.*, 1(1):19–26, Mar 2020.
- [Me] Bo Mel and er. Guideline for the implementation of inter-row cultivation in small grain cereals. <https://iwmpraise.eu/guideline-for-the-implementation-of-inter-row-cultivation-in-small-grain-cereals/>. [Online; accessed 21. May 2021].
- [Min19] Manpreet Singh Minhas. Transfer learning for semantic segmentation using pytorch deeplab v3. <https://github.com/msminhas93/DeepLabv3FineTuning>, Sep 2019.
- [Ng18] Andrew Ng. Machine Learning Yearning. <https://github.com/ajaymache/machine-learning-yearning>, 2018. [Online; accessed 10. May 2021].
- [Nig21] Urs Niggli. *Alle satt?: Ernährung sichern für 10 Milliarden Menschen*. Residenz Verlag, 2021.
- [nvi21] Image Segmentation Using DIGITS 5 | NVIDIA Developer Blog. <https://developer.nvidia.com/blog/image-segmentation-using-digits-5>, Jun 2021. [Online; accessed 12. Aug. 2021].
- [ope22] openvinotoolkit. cvat. <https://github.com/openvinotoolkit/cvat>, Jan 2022. [Online; accessed 8. Jan. 2022].
- [pas21] Papers with Code - PASCAL VOC 2012 val Benchmark (Semantic Segmentation). <https://paperswithcode.com/sota/semantic-segmentation-on-pascal-voc-2012-val>, Aug 2021. [Online; accessed 12. Aug. 2021].
- [poi18] PNPOLY - Point Inclusion in Polygon Test - WR Franklin (WRF). [https://wrf.ecse.rpi.edu/Research/Short\\_Notes/pnpoly.html](https://wrf.ecse.rpi.edu/Research/Short_Notes/pnpoly.html), Dec 2018. [Online; accessed 12. Aug. 2021].

## Bibliography

- [PTRC07] Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3):45–77, December 2007.
- [Pub19] United Nations Publications. *World Population Prospects 2019: Highlights*. UN, 2019.
- [pyt21a] PyTorch. [https://pytorch.org/hub/pytorch\\_vision\\_deeplabv3\\_resnet101](https://pytorch.org/hub/pytorch_vision_deeplabv3_resnet101), Aug 2021. [Online; accessed 12. Aug. 2021].
- [pyt21b] pytorch. vision. <https://github.com/pytorch/vision>, May 2021. [Online; accessed 10. May 2021].
- [SPC<sup>+</sup>20] Oliver Schoppe, Chenchen Pan, Javier Coronel, Hongcheng Mai, Zhouyi Rong, Mihail Ivilinov Todorov, Annemarie Müskes, Fernando Navarro, Hongwei Li, Ali Ertürk, and Bjoern H. Menze. Deep learning-enabled multi-organ segmentation in whole-body mouse scans. *Nat. Commun.*, 11(5626):1–14, Nov 2020.
- [ten21] TensorBoard | TensorFlow. <https://www.tensorflow.org/tensorboard>, Jun 2021. [Online; accessed 2. Nov. 2021].
- [ten22] tensorflow. models. <https://github.com/tensorflow/models/tree/master/research/deeplab>, Jan 2022. [Online; accessed 7. Jan. 2022].
- [VCS16] Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. Examining the Impact of Blur on Recognition by Convolutional Networks. *ResearchGate*, Nov 2016.

# A. Appendix

## A.1. Zusammenfassung

Kamerabasierte Systeme in der Landwirtschaft können die Menge an benötigten Unkrautbekämpfungsmitteln reduzieren. Die Systeme müssen dabei beurteilen können, ob es sich bei einer Pflanze um Unkraut oder eine Nutzpflanze handelt. Die Grundlage hierfür sind Neuronale Netze, die sich ihr „Wissen“ auf Basis von Daten aneignen. Diese Daten müssen zunächst gesammelt und anschließend aufbereitet werden. Die Aufbereitung dieser Daten ist in der Regel sehr zeitintensiv und benötigt in vielen Fällen spezifisches Fachwissen. Die gegenständliche Masterarbeit beschäftigt sich mit dem Einsatz von Neuronalen Netzen für die Erkennung von Unkraut auf Fotos eines Ackers. Es wird aufgezeigt, dass Datensets erweitert werden können, indem die Pflanzen (Unkräuter und Nutzpflanzen) aus den Bildern ausgeschnitten, in einer Datenbank gespeichert und in anderer Form wieder zusammengesetzt werden. Diese neu generierten Datensets werden „Digital Field Twin“ genannt und erzielen genauere Ergebnisse bei der Segmentierung von neuem Bildmaterial der selben Nutzpflanze. Diese Bilder können einer bestehenden Datenbank entnommen werden oder von Landwirten am Feld mittels Smartphone aufgenommen werden. Für Letzteres wurde ein eigener Arbeitsablauf erstellt und implementiert, welcher den Anwender am Smartphone durch den Prozess des Fotografierens und Beschriften einer neuen Pflanze führt. Das Sammeln neuer Daten bedarf keiner händischen Kennzeichnung der Pflanzen im Bild. Am Beispiel eines echten Sojafeldes wird in der Arbeit aufgezeigt, dass mit den Daten, die mit einem Smartphone in einem Teilstück des Feldes aufgenommen wurden, ein Model trainiert werden kann, welches Bilder von anderen Teilstücken des Feldes mit hoher Genauigkeit segmentieren kann.