Machine Learning for Crop and Weed Classification

FINAL INTERNSHIP REPORT

Lodewijk Voorhoeve
WAGENINGEN UNIVERSITY / POLITECNICO DI MILANO | 31-01-2018

Machine Learning for Crop and Weed Classification

Final Internship Report

Study program:	Biosystems Engineering (MSc); Track: Farm Technology
Course Code:	FTE-70430; ECTS: 30
Student:	Lodewijk Voorhoeve
Registration No.:	920704904080
Supervisors:	Matteo Matteucci (Politecnico di Milano)
	Gert Kootstra (Wageningen UR)
Internship Coordinators:	Arni Janssen, Frank Helderman (Wageningen UR)
Date:	31-01-2018
Version:	1.0
Status:	Released

Acknowledgements

This opportunity, to follow an academic internship at Politecnico di Milano, has been a great experience for me. The expert advice from professor Matteo Matteucci has given me insights in the possibilities of computer vision and machine learning, and its applications in agriculture. With his guidance, and the help of many students at the AIR-Lab at Politecnico, I have been able get acquainted with programming in python, several machine learning concepts and how to use this for the task of image classification. Though I feel as I've not nearly spent enough time and effort to use all this available expertise, I hope that some of it is reflected in this report.

My time in Milan and its surroundings has been a great experience as well. The magnificent lake of Como and its mountains were a welcome distraction. Though I faced some difficulties in settling and finding my ways, it taught me a lot too. Finally, I am grateful to the Erasmus organisation. This academic internship would not have been possible without them, which supported me financially during these five months.

Nomenclature

ML	Machine Learning
CV	Computer Vision
CWFID	Crop Weed Field Image Dataset
SB2016	Sugar Beets 2016 (dataset)
Logit	Multiple Logistic Regression (classifier)
RFC	Random Forest Classifier
OOB	Out-of-Bag (error rate for Random Forest Classifier)
SVM	Support Vector Machine (classifier)
NN	Neural Network
FCN	Fully Connected Network
CNN	Convolutional Neural Network
U-Net	U-shaped neural network architecture, proposed by [1]
VGG16	16-layer deep neural network architecture, proposed by [2]

Abstract

The challenge of crop/weed classification is addressed by considering various machine learning techniques. The goal is to be able to identify a crop or weed in an image and display its location, this method is called pixel-wise classification. The algorithms for this work are based on earlier work of Stefano Cereda [3], a former MSc. Student in Computer and Information Science at Politecnico di Milano. The goal of this academic internship is to gain knowledge and skills in programming and machine learning, as a basis for a possible thesis in this subject.

The key-component of any classification challenge is the available data. Two publicly available datasets are used to train and evaluate machine learning algorithms on, namely: the Crop/Weed Field Image Dataset (CWFID, published in 2014) and the Sugar Beets 2016 dataset (SB2016, published in 2017). CWFID consists of 60 top-down images with three classes: crop, weed and soil. All vegetation which does not belong to the crop class is considered weed. The SB2016 dataset, consisting of 283 annotated images, reportedly contains 10 classes, among which are crop, different types of weed and soil. Later analysis displayed a total of 16 different weeds, adding to that crop and soil creating a total of 18 different classes.

The pixel-wise classification is handled by two approaches. The first approach uses three classical machine learning methods: Multiclass Logistic Regression (Logit), Random Forest Classifier (RFC) and Support Vector Machine (SVM). The method of how these classifiers are trained and their results are explained in depth within this report. They consist of: masking the soil which reveals the vegetation-only pixels, tiling the image according to a grid of unmasked pixels, extracting features and then training a classifier on these features. For CWFID, the procedures were very similar to that used by Cereda [3], with the exception of added features and a mask-cleaning operation. The smoothed SVM still has the highest overall performance, but the newly added classifiers show a similar performance with a slightly higher Jaccard-index due to improved masking. For the Sugar Beets 2016 dataset, the Random Forest Classifier with label smoothing, using a window of 100 pixels and a grid of 10 pixels, obtained a 91.21% accuracy and 97.16% recall on crops vs weed. However, the results display a rather poor evaluation on the different types of weeds per label, as visualised in appendix C. The poor results on the different weed types is most likely caused by the severe class imbalance (certain weed types are underrepresented).

An introduction and first attempts show the potential for using Neural Networks on this task. The results look promising though more knowledge and algorithm-training time are required to see the full potential of these techniques, which have already been studied and proved in literature by using various neural network architectures. The used neural network is based on a pre-trained model, called VGG16, which is adjusted to the task of pixel-wise classification for crop and weed detection. The work of Cereda shows that other network architectures, such as U-Net, perform well on segmentation tasks with a small dataset. However, these networks require considerable computational power and time to train. More experience and training time are required to further prove the potential of various networks.

In the conclusion and future work, the possibilities of implementing machine learning networks for the task of crop-weed detection is reflected upon. An interesting example is the work of [4] where in-field labelling is used which takes only about 1 minute. In general, it can be concluded that the evaluated baseline classifiers show similar performance on both datasets, compared to previous work which made use of several neural network architectures. These considerations, and the knowledge gained during this internship, can be used as a starting point or reference for a thesis in crop-weed detection.

Table of Contents

List of Figures

List of Tables

1. Introduction

Rising demands in food quantity and quality, due to an increasing and more informed population, pose challenges and possibilities for agriculture. Among these challenges is the reduction of negative environmental impacts caused by intensified production. One way to achieve this is to focus on precise treatment of the field and crops with the help of modern technology: Precision Agriculture (PA) [5]. An example of PA is the use of Unmanned Aerial Vehicles (UAV's) to monitor the production field. The information that is acquired can then be used to adapt fertilizer and herbicide distribution patterns. To add more targeted techniques that allow for automated plant specific treatment, ground vehicles can be used. These vehicles would need to be able to differentiate between plant species, or at least between crop and weed, to apply the correct treatment.

Differentiation between plant species can be achieved by Computer Vision (CV) and Machine Learning (ML). Computer vision is a technique to analyse a digital image so that an action can be connected to it. With the use of Machine Learning, the analysis of digital images can be automated, since an algorithm can learn what it must look for in the image data. One of the challenges for this automation is the availability of data with corresponding annotation, in both quantity and quality. In the recent years, two datasets of interest have been released: the Crop/Weed Field Image Dataset (CWFID) [6] in 2015 and the Sugar Beets 2016 dataset [7].

Several papers have been released based on the CWFID dataset. Haug at al. [8], who performed classification of crop and weed without segmentation on the CWFID with a Random Forest Classifier (RFC), achieved a classification accuracy of 93.8%. Di Cicco et al. [9] proposed methods on augmenting the dataset (creating 3D models with textures) published by [6] and achieved 91.3% accuracy using the RGB Basic SegNet, a Neural Network for image segmentation. Milioto et al. [10] used a Convolutional Neural Network on the Sugar Beets 2016 (SB2016) dataset and experimented with the application in different locations, outside of the training area, achieving 94.7% test accuracy [10].

Recently a former MSc student of Politecnico di Milano, Stefano Cereda, graduated on a comparison of different classifiers for the job of crop and weed classification [3]. In his work, Cereda describes several classical methods such as a Random Forest Classifier (RFC), Support Vector Machine (SVM) and Gradient Boosting (XGB) as baseline classifiers. He compared their performance to several Artificial Neural Networks (ANN's). This comparison shows which methods perform best, and how they can be used for the task of crop and weed classification.

This report contains findings and experiments based on Cereda's work. The aim is to use the newly available Sugar Beets 2016 dataset for classification using the existing classifiers implemented by Cereda. Basline classifiers such as Random Forest and Multiclass Logistic Regression are used to present initial results. The newly available TensorFlow library is used to display the ability of Neural Networks on this classification task. For this, an existing, pre-trained network is used to speed up the training process. The results of these classifiers are discussed, and future possibilities are presented which could form a basis for further research.

2. Materials and Methods

The approach for adapting the existing code and implementing new classification methods is described in this chapter. First, the two publicly available datasets are analysed and presented. They form the basis for classification and evaluation of the designed approach. Then, the basic functionality and pipeline of the previously designed code is presented, along with how the data is split (train/test), how the mask is created and cleaned, how the images are tiled and finally how the features are extracted. This gives the basis for the classifiers, which will use the training data to fit on. Each classifier, being Logit, RFC and SVM, is briefly discussed after which the training results are displayed. Finally, the evaluation method and smoothing technique is discussed which will give the reader a basis to analyse the results, presented in chapter 3. Apart from the baseline classifiers discussed earlier, a brief introduction of Neural Networks is presented.

2.1. Old and New Dataset: CWFID and Sugar Beets 2016

Machine Learning is data driven, this means that the performance depends on the availability and quality of data. In this case the aim is to use supervised learning, which also requires a ground-truth annotation. While there are several datasets available for tasks such as hand-written digit recognition, face recognition, urban street scene-labelling and medical imagery, the field of agriculture lacks available data. In the past, two datasets were made publicly available: first the Crop/Weed Field Image Dataset (CWFID) in 2014 and later followed by the Sugar Beets 2016 dataset, both acquired with the Autonomous Field Robot BoniRob [11]. Both datasets are described along with examples of their content.

2.1.1. CWFID (2014)

The Crop/Weed Field Image Dataset (CWFID) [6] contains 3 classes (soil, crop, weed) and 60 top-down images. It was released in 2014 along with a publication by Haug and Ostermann [6].

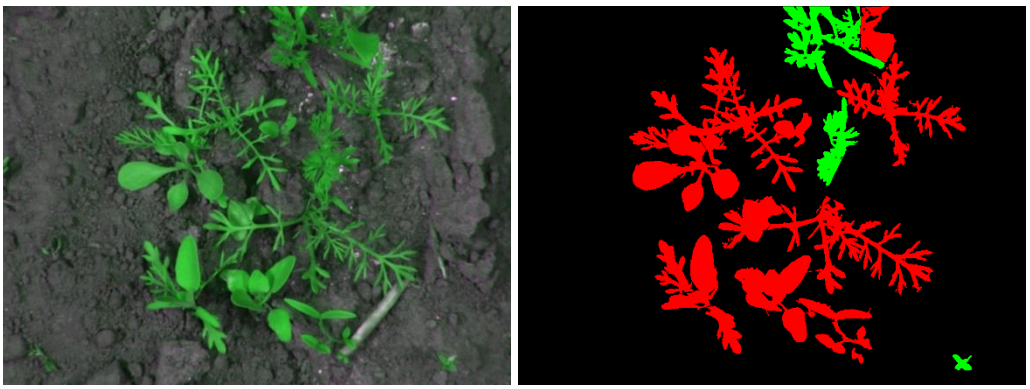


Figure 1 - Original image with R and NIR-channel (left) and Ground-Truth annotation (right) from CWFID

All images have a corresponding annotation with three different classes: soil (black), crop (green) and weed (red). An interesting observation: since the original image contains 3 channels, of which the first one is red, the second one Near InfraRed (NIR) and the third also red, it seems like the image is coloured. However, the colour green is achieved by putting the NIR-image data in the green channel. In 2014, Haug et al. published a paper [8] using this dataset but adding another class: Chamomile (a type of weed). With this class, which visually shows resemblance to carrots, the crop, they achieved a classification accuracy of 93.8%. However, the released dataset does not contain this extra annotated class. Cereda reached a classification accuracy of 89.4% using a Support Vector Machine on this dataset and 89.7% classification accuracy using a U-Net-based Neural Network.

2.1.2. Sugar Beets 2016

The Sugar Beets 2016 dataset [7] counts 17 classes (soil, crop, weeds) and 283 annotated images. However, in their report, Chebrolu et al. [7] report 10 different classes, which is assumed to be an error in the annotation (the authors have not replied to the question). Apart from a Red-Green-Blue (RGB) image, a NIR-image and Ground-Truth coloured annotation are available.

Figure 2 - Original image, NIR-image and Ground-Truth annotation image of the Sugar Beets 2016 dataset

Apart from image data, the dataset contains information on the time of capture, motor encoders, position of the robot, depth sensors and more. However, this information will not be used during this project. There are no directly comparable results available for this dataset. However, in [10] the authors use a larger dataset that also contains SB2016, and report results of 86% crop and 66% weed precision, and 97% crop and 85% weed recall, with a Mean-Intersection-over-Union of 81% for crop, weeds and soil.

2.2. Alteration of the work of Cereda on the Baseline Classifiers

Cereda performed experiments using a Random Forest Classifier (RFC), a Support Vector Machine (SVM) and Gradient Boosting (XGB) as a baseline for his work [3]. The code is written in Python, using several libraries such as the Scikit-Learn library for Machine Learning. The process of training and prediction are based on the work described by Haug et al. in [8].

Feature Nr.	Description	Scale Invariant
1	perimeter (length of contour)	No
2	area (number of pixels covered by leaf)	No
3	length of skeleton	No
4	compactness (area / perimeter2)	Yes
5	solidity (area / area of convex hull)	Yes
6	convexity (perimeter / perimeter of convex hull)	Yes
7	length of skeleton / perimeter	Yes
8	minimum of vegetation pixel intensities	Yes
9	maximum of vegetation pixel intensities	Yes
10	range of vegetation pixel intensities	Yes
11	mean of vegetation pixel intensities	Yes
12	median of vegetation pixel intensities	Yes
13	standard deviation of vegetation pixel intensities	Yes
14	kurtosis of vegetation pixel intensities	Yes
15	skewness of vegetation pixel intensities	Yes

Table 1 - Features use by Cereda and Haug et al.

For training, images are masked to remove the soil and maintain vegetation. Then, the vegetation pixels are divided into tiles of 80x80 pixels, based on a sparse grid of key-points with a distance of 10x10 pixels (more dense than the 15x15 grid proposed in [8]). Each tile is then used for feature

extraction using the features listed in Table 1. The extracted features, based on the ground-truth annotation, are grouped in one file. This file is then used by a classifier to train on. The trained classifier in turn is used to predict unseen data. A visualisation of what these features are based on can be seen in Figure 3, as proposed in [8] and used in the work of Cereda [3].

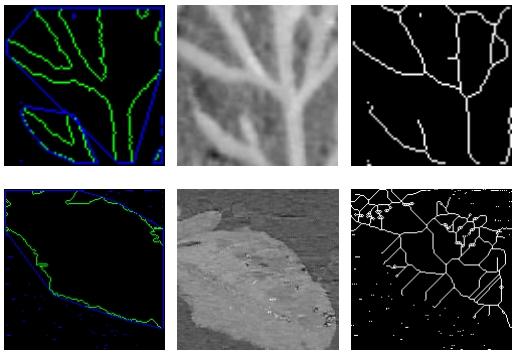


Figure 3 - From left to right: Convex Hull (blue) of border, NIR-channel (pixel-based features) and Skeleton, from CWFID (top row) and SB2016 (bottom row)

In order to use the code, it has been stripped down to the baseline-classifier code only, omitting all parameters and variables for the Neural Networks. This helped my understanding of the code and working principles within Python, the Machine Learning library from Scikit-Learn and many other libraries. Added to this code is a framework to select the Sugar Beets 2016 dataset, background removal based on the available NIR and RGB-images, a simple and fast mask cleaning method, feature selection based on the 18 classes and a modified prediction algorithm to handle 18 classes.

In an attempt to improve classification, 6 scale-free features were added based on the Harris corner detection algorithm [12], implemented according to instructions in [13]. The results of this algorithm, pixel location and intensity values, are stored in an array. These values are then converted to single numbers using a summation value of all corners, the mean corner value and the max corner value. Two experimental values for k were selected. Here, k is a freely chosen parameter that defines the subtraction power of squared covariance matrix. A visualisation of the Harris Corner Detector is visualised in Figure 4. Experiments with and without the 6 added features on both a Logistic Regression and Random Forest classifier are displayed in Table 2. A feature importance list, given in Table 3, was extracted using a Forest of Trees method, displays the informative value of each feature to the classification process. Only the scale-free features (18 in total) are displayed, of which feature 12 to 17 are the Harris features (bold-marked in the table). Since the Harris features gave better training accuracy for the Random Forest classifier, they were used throughout the remaining training process.

Figure 4 - Example of Harris Corner Detection with OpenCV. Source: MeccanismoComplesso.org

WINDOW GRID, FEATURES	LOGIT	RFC (TREES)	SVM
W80 G5 F15	0.790	0.962 (700)	N.A.
W80 G5 F21	0.790	0.971 (700)	N.A.
W80 G10 F15	0.787	0.895 (700)	N.A.
W80 G10 F21	0.787	0.913 (700)	N.A.
W80 G15 F15	0.789	0.855 (700)	0.852
W80 G15 F21	0.789	0.867 (700)	0.909

Table 2 - Comparison of features during training of Logit and RFC on the CWFID dataset

RANK	FEATURE NR.	IMPORTANCE
1	feature 5	10.68%
2	**feature 13**	8.72%
3	**feature 17**	8.64%
4	feature 9	7.67%
5	feature 6	5.82%
6	feature 4	5.49%
7	**feature 16**	5.42%
8	feature 0	5.25%
9	**feature 14**	5.06%
10	feature 3	4.98%
11	feature 1	4.58%
12	feature 7	4.16%
13	feature 11	4.03%
14	feature 8	4.03%
15	feature 2	3.94%
16	**feature 12**	3.91%
17	**feature 15**	3.85%
18	feature 10	3.76%

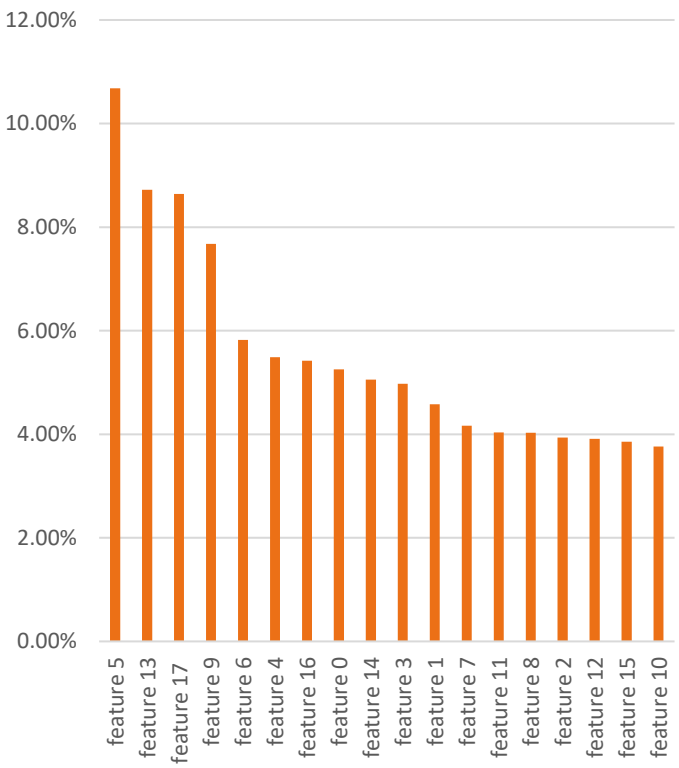


Table 3 - Feature importance through Forest of Trees method, bold-marked are Harris features

2.2.1. Datasets and Splits: CWFID and Sugar Beets 2016

Haug and Ostermann describe in [6] two possible splits for the CWFID dataset. The first split is based on a real-life use case, where it is assumed that the robot is trained on the beginning of a row. The split is image 1 to 20 for training, 21 to 60 for testing. The second split is from a computer vision point of view and uses a random 66% of the images for training, the remaining 33% for testing. For this project, only the second split is used and evaluated. The authors of [7] do not provide a suggestion for a data split, therefore the 66%-33% random split will be used here as well. Apart from the 283 available annotated images, there are several hundreds of unannotated images available. The authors state that in the future, more annotated images will be available.

2.2.2. Background Removal with Mask and Mask Cleaning

The first step of the algorithm is removing the area which does not contain vegetation. A classical method for this is called Excessive Green (ExG), proposed by Woebbecke et al. [14], which overexpresses green pixels in an image, given in Eq. 1. Apart from that, Kataoka et al. [15] proposed the Color Index of Vegetation Extraction (CIVE), given in Eq. 2. Hague et al. [16] proposed converting an RGB image to a grayscale one using an index, based on the work of Marchant and Onyango [17], and developed Eq. 3, in which the variable a depends on the camera type. In the code, the results of these formulas are combined and normalized into an array. For the CWFID dataset, this algorithm is not used since the images only contain an R and NIR channel. However, for the Sugar Beets 2016 dataset, this approach is used to remove the background.

$$ExG = 2 * G - R - B \tag{1}$$

$$CIVE = 0.441R - 0.811G + 0.385B + 18.78745 \tag{2}$$

$$y = \frac{G}{R^a * B^{1-a}} \; with \; a = 0.667 \tag{3}$$

$$NDVI = \frac{NIR - R}{NIR + R} \tag{4}$$

Since a NIR-image is available for both datasets, the Normalized Difference Vegetation Index (NDVI) can be calculated according to Eq. 4. The results of either the NDVI or the combination of RGB-based values are converted to a binary image using Otsu's Method [18]. This binary image is then used to create a mask which covers all non-vegetation pixels. In order to reduce the noise, a simple opening-closing operation was added, which reduces the presence of small unmasked areas. An example of an area in a not-cleaned and cleaned mask is given in Figure 5. When using the SB2016 dataset, it became clear that the NIR-image caused trouble for the Otsu-thresholding due to appearance of non-vegetation which reflected the NIR and distorted the image. See appendix D for examples of image distortion. For this reason, only the RGB-image was used when training and evaluating SB2016. On top of this, Excessive Green alone gave the best masking results, setting the other indices to zero.

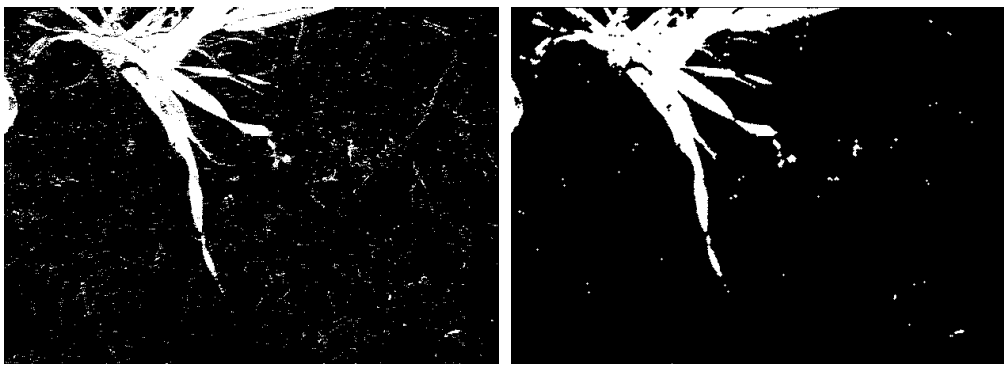


Figure 5 - cropped image (right top corner) of uncleaned mask (left) and cleaned mask (right)

2.2.3. Image Tiling

After the background removal, the masked image is tiled. Tiling means extracting an area of the image around a selected point (i.e. pixel). The tile area is later analysed, and features are extracted. In order to extract tiles, Haug and Ostermann proposed a grid of sparse key points with a distance of 15 pixels, horizontally and vertically, and used a tile-size of 80x80 pixels. Cereda used a more dense grid of 10 by 10 pixels, which increased the number of tiles, with the same tile size. In the code, a tile is extracted above every key-point which contains vegetation (i.e. is unmasked), therefore tiles can overlap. An

example is displayed in Figure 6. Different grid densities and tile sizes are used and evaluated. The results of this are displayed in chapter 3.

Figure 6 - From left to right: key-points from the grid, tiles around key-points and the image mask

2.2.4. Feature Extraction

The tiles from the masked image are used for feature extraction. This process identifies the features, listed in Table 1, within each tile. The computed features are connected to the ground-truth annotation of the image and stored in a file. This file is part of the pipeline, which comes with the Scikit-Learn library and is useful for storing and accessing information (the method is called 'pickling'). Apart from extra classes (18 for the Sugar Beets 2016 dataset), no features were added yet.

2.2.5. Training Classifiers: Random Forest, Logistic Regression and Support Vector Machine

The computed features and corresponding ground-truth labels form the input for the classifier. Three classifiers are used: Logistic Regression (Logit), Random Forest Classifier and Support Vector Machine. The code is based on the Scikit-Learn library.

2.2.5.1. Logistic Regression

Logistic regression is a predictive analysis used to describe the relationship between one dependent variable and one or more independent variables. In the case of multiple classes, a multinomial or softmax regression is used, which generalizes logistic regression to predict the probabilities of more than two different outcomes.

For the case of crop/weed detection with 18 different classes, the linear model of Scikit-learn library (Python) is used. The hyperparameter for this model is de C-value (cost), which can be automatically selected using the cross-validation method *StratifiedKFold*. In the current setup of the code, the cross-validated version of the multiclass logistic regression is used, which produces a single training accuracy value after fitting the model.

2.2.5.2. Random Forest Classifier

A Random Forest Classifier constructs a group of decision trees that outputs the class which is most often predicted by the individual trees (mode). Each decision tree is a predictive model which uses observations (features) of an item to generate a prediction of the label. The observations are done in the 'branches', whereas the prediction of a class label is visible in the 'leaves'. The downside of decision trees is that they tend to overfit (creating too many branches). Random Forests can reduce this problem by generating a high number of trees, therefore limiting the branches per tree while maintaining

The modifiable hyperparameters used for this method are the maximum number of features, the criterion and the number of estimators (decision trees). The selection method involves fitting different setups of the model with either the square root or the base-2-logarithm of the number of features. This yields four different random forest ensembles which each produce an Out-of-Bag (OOB) error

while the number of decision trees is subsequently increased. A graphical representation can directly display the best possible combination which can be used to fit the model. Examples of this are displayed in Figure 7 and Figure 8.

The current setup has a function to explore hyperparameters and select the number of decision trees based on a graphical output of the OOB error rate. This is a memory intensive operation and requires, with the current dataset and a dense grid of key-points, at least 8GB RAM.

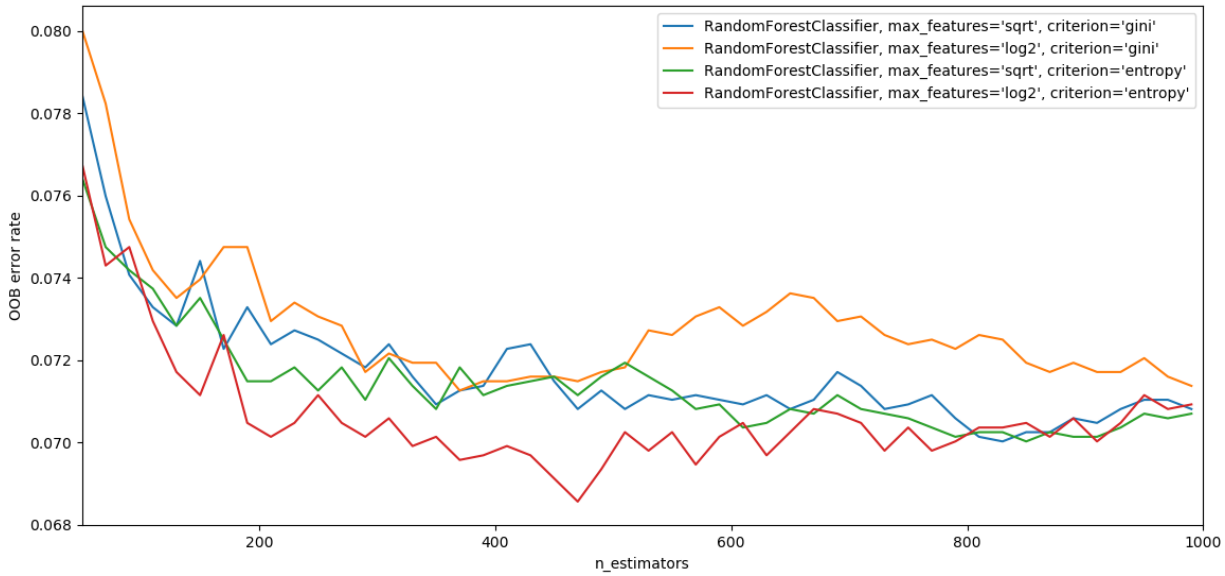


Figure 7 - Hyperparameter exploration of Random Forest Classifier for SB2016, using W80 and G15

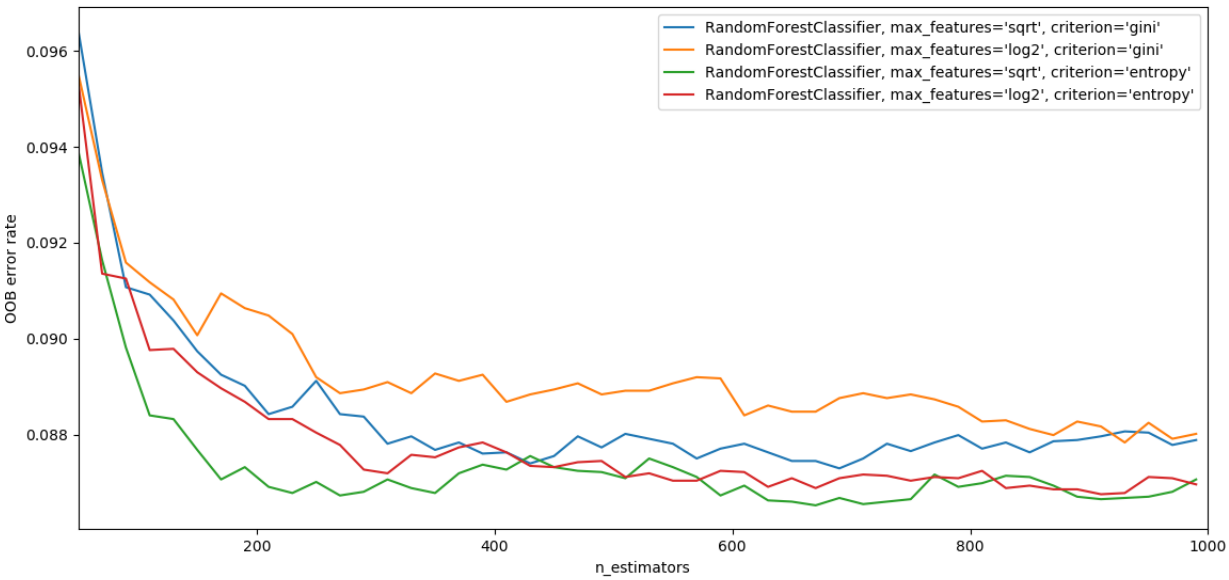


Figure 8 - Hyperparameter exploration of Random Forest Classifier for CWFID, using W80 and G10

2.2.5.3. Support Vector Machine

A support vector machine (SVM) for classification creates decision boundaries to fit the training data, using only certain data points, the so-called support vectors. This makes the method category-based and thus non-probabilistic. In order to get the probability estimate of predicted data, five-fold cross-validation is used. One constraint of using the SVM is that the data should be regularized by using a standard scaler.

Hyperparameters for support vector machines are the cost or soft-margin parameter C; the kernel and the parameters of that kernel. Commonly (and also in the Scikit-learn library), a Guassian-based kernel, which has only one tuning parameter (γ). The kernel-type is either linear, (polynomial) or radial-basis. The best hyperparameters can be selected by performing a grid search, which is evaluated using cross-validation. In the current setup, hyperparameters can be selected by performing a grid-search operation with cross-validation. This happens step wise, in which first the kernel is selected, then subsequently trying combinations of the hyperparameters C and γ.

2.2.5.4. Training Results

Each classifier was training on different feature files for both the CWFID and SB2016 dataset. The modified parameters are window size W (varying from 8 to 100 pixels) and grid step G (varying from 3 to 15 pixels). For each model, a training accuracy is produced and presented in Table 4 for CWFID and SB2016. However, when training of a model was not possible due to a lack of memory (14GB of committed RAM), N.A. (not available) or N.E. (not evaluated) was assigned. K-fold Cross validation of SVM W30 G15 for SB2016, with a split (K) of 3, gave an accuracy of 0.832, 0.829 and 0.795, indicating that the true accuracy of the model might be 3-5% lower. Cross validating the SVM is time consuming, therefore it was not used for further training procedures. The SVM was only trained on the grid-size 15 and 10 feature files, since the required memory and computation time were too high. Also training the RFC on grid-size 3 for SB2016 was not possible due to memory limitations.

WINDOW **G**RID **CWFID**	**LOGIT**	**RFC (**TREES**)**	**SVM**
W8 G3	0.728	0.751 (700)	N.A.
W16 G5	0.742	0.779 (700)	N.A.
W30 G5	0.761	0.836 (700)	N.E.
W30 G10	0.761	0.803 (700)	N.E.
W30 G15	0.761	0.796 (700)	N.E.
W40 G5	0.770	0.881 (700)	N.E.
W40 G10	0.769	0.825 (700)	N.E.
W40 G15	0.770	0.811 (700)	N.E.
W80 G5	0.790	**0.971 (700)**	0.939
W80 G10	0.787	0.913 (700)	0.920
W80 G15	0.789	0.867 (700)	0.909
W100 G5	0.782	**0.980 (700)**	**0.957**
W100 G10	0.782	0.937 (700)	0.940
W100 G15	0.792	0.897 (700)	0.930

WINDOW **G**RID **SB2016**	**LOGIT**	**RFC (**TREES**)**	**SVM**
W8 G3	0.720	N.A.	N.A.
W16 G5	0.765	0.814 (400)	N.A.
W30 G5	0.797	0.865 (500)	N.A.
W30 G10	0.798	0.845 (600)	0.860
W30 G15	0.798	0.837 (600)	0.862
W40 G5	0.808	0.890 (600)	N.A.
W40 G10	0.807	0.864 (600)	0.878
W40 G15	0.806	0.853 (600)	0.880
W80 G5	0.840	**0.925 (400)**	N.A.
W80 G10	0.839	0.905 (600)	0.914
W80 G15	0.840	0.900 (600)	0.914
W100 G5	0.847	0.928 (400)	N.A.
W100 G10	0.845	**0.911 (600)**	**0.918**
W100 G15	0.846	0.901 (400)	0.919

Table 4 - Training results on CWFID (left) and SB2016, bold marked is used for prediction later on

2.2.6. Prediction

The prediction of unseen test data happens through a similar procedure of feature extraction. First, the background of the test image is removed, after which the image is tiled on vegetation-only pixel locations. The tiles are then used for feature extraction. Now, the trained classifier is used to assign probabilities for each tile. The maximum probability among the different classes is assigned along with the class label. A thresholding option allows for setting a minimum probability value for the tile outcome. If the minimum value is not reached, the tile will not get a class assigned to it. After the classification of each tile, a smoothing step can be executed, as presented in the next paragraph. Finally, interpolation is used to create a coloured annotation with the same size of the input image.

2.2.7. Smoothing and Threshold

The smoothing method designed by Cereda was applicable only to a binary problem and thus only used for CWFID. For SB2016, a new smoothing algorithm was designed which compared the prediction of a central key-point to its direct surroundings. Figure 9 shows the central key-point *K* with its surrounding points *ABCD*. Each point can belong to any of the 18 available classes, each with their own probability value of that class. If two or more points surrounding *K* have a different class, their combined mean probability is compared to the probability of point *K*. When the mean probability exceeds the one of point *K*, a new class is assigned. In the results, a comparison between smoothed and unsmoothed results is included.

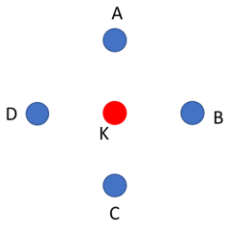


Figure 9 - Grid representation of points surrounding key-point K, used for smoothing

Since there are 18 different classes, it is important that a key-point is only assigned when the probability is truly high, and not just a marginally higher probability compared to the other classes. For this, a threshold can be applied. When the threshold of 50% probability is not met, the point is assigned to have value *nan*, which can be considered as having 0 probability to any class when it enters the smoothing phase. The predictions were tested with and without thresholds and a comparison is displayed in chapter 3.

2.3. Neural Networks

Neural Networks (NN's) is the collective name for deep learning technologies. A network consists of multiple neurons that are ordered in layers, generally consisting of an input layer, hidden layers and an output layer. Each neuron can have one or more connections with neurons in a preceding or next layer, but not with neurons of its own layer. Each connection is coupled with a weight-value that manipulates the input or output. Within a neuron, a threshold value (or bias) defines whether the neuron will pass information onto the connected neuron(s). Currently, Neural Networks outperform all classical machine learning methods for speech recognition, image classification and many other tasks. Typical of Neural Networks is their black-box principle: the user has little control over how the hyperparameters (weights and biases) are learned by the network. This is also one of the reasons they are the preferred method for difficult-to-characterise problems, such as image recognition, for which hand-crafting of features is laborious and difficult work.

The task of crop/weed detection makes use of images. For this, a special type of Neural Network is available: the Convolutional Neural Network (CNN), designed for image classification. The difference with traditional NN's lies in how the information is processed: a NN would process an image as one large array of numbers, whereas a CNN breaks the image up into tiles. This allows for parallel processing and thus reduces the time to detect objects within an image, regardless of its location. This makes them ideal for image classification, which means determining what the main content of an image is. However, since the location information is not used (and thus lost), CNN's are not usable for object localization.

To solve this problem, the sliding-window approach was designed. This strategy breaks the image up into parts with a central point, similar to the strategy used with classical methods. However, since each image is divided into many small images, the computation time is very high and thus

unfavourable. It would also require a mask-like operation which undermines the benefits of the automated learning of NN's. To solve these issues, Fully Connected Networks (FCN's) were designed.

FCN's use a similar architecture to CNN's, but their output is a pixel-wise classification instead of a single class per image. Long et al. describe how coarse outputs per layer are connected back to pixels. To convert a CNN into an FCN, the Fully Connected layers have to be changed into Convolutional Layers. This allows the output of a spatial map. In technical terms: the softmax layer, which outputs the class-core of an image, is replaced with a 1x1 convolution layer, with a depth that is equal to the number of classes.

2.3.1. Transfer Learning: using pre-trained model VGG16

To compare the baseline classifiers with the performance of a neural network, a Fully Connected Network based on VGG16 was selected. VGG16 is a network architecture, published in 2014 by Zisserman and Simonyan [2], that was designed to show the importance that depth has when it comes to classification. This network is 16 layers deep and can be used for image classification. Long et al. [19] used this network to show that classification networks could be modified to the task of semantic segmentation. The network, created by Sagieppel [20] and available on Github, requires only the transformation of the annotation images from RGB to binary images, creating a greyscale for each class instead of an assigned colour. The network, designed in TensorFlow, was initially trained on CWFID and SB2016 for just 300 to 550 steps to demonstrate the initial performance.

A second training phase was carried out for 2000 steps on SB2016, with a 10 times smaller learning rate in an attempt to achieve better convergence. The training loss is visualised in Figure 10, where the plateau is very visible in the left graph. The right graph of Figure 10 shows a truncated version where the smaller training loss is visibly not able to converge much further (a minimum was found). The model was saved and can be restarted for training with a smaller learning rate, starting at the last recorded loss. Since TensorFlow offers the possibility to train on GPU's, this would be an interesting opportunity to explore the true potential of this network. However, within the scope of this project it was not possible to evaluate this performance. Preliminary results are displayed in chapter 3.

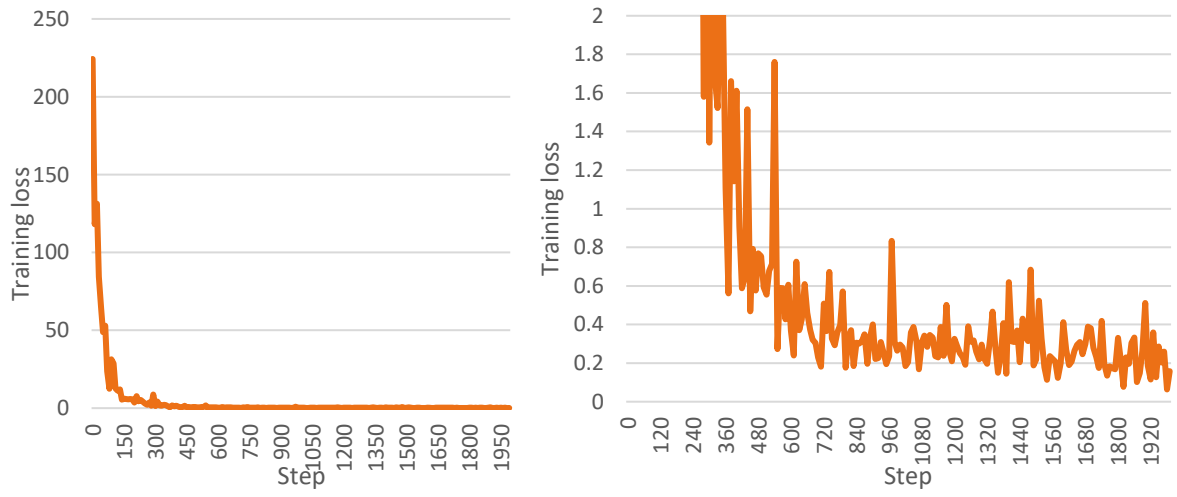


Figure 10 - Training loss on SB2016 dataset (left: full scale, right: truncated to see plateau), learning rate = $1*e^{-5}$

2.4. Evaluation Metrics

The baseline classifiers and the neural network are evaluated using several standard methods: the Intersection over Union (IoU, also known as Segmentation Accuracy (SA) or Jaccard Index (JI)) Eq. 5, accuracy Eq. 6, precision Eq. 7, recall (or sensitivity) Eq. 8 and F1-score (or Dice Similarity Score, DSS)

Eq. 9. F or precision agriculture, it is important to keep the number of False Negatives small, which means keeping the number of crops classified as weeds (and thereby removing them) as small as possible. A high recall for crops is thus preferred to a high weed accuracy. Since the SB2016 dataset has 18 different labels (see appendix A) and a strong class imbalance (see appendix B), the following metrics will be evaluated as follows:

1. Evaluation of Vegetation vs. Soil accuracy, using the IoU measure that Cereda [3] also applied.
2. Evaluation of Crop vs. Weed accuracy, using all evaluation metrics mentioned.
3. Evaluation of per-class accuracy with the IoU score of vegetation-only labels, excluding the soil class for improved interpretability.

$$IoU = \frac{true\ positive}{true\ positive + false\ positive + true\ negative} \tag{5}$$

$$Accuracy = \frac{true\ positive + true\ negative}{true\ positive + false\ positive + true\ negative + false\ negative} \tag{6}$$

$$Precision = \frac{true\ positive}{true\ positive + false\ positive} \tag{7}$$

$$Recall = \frac{true\ positive}{true\ positive + false\ negative} \tag{8}$$

$$F1\ score = \frac{2 * precision * recall}{precision + recall} \tag{9}$$

3. Results

Three baseline classifiers and one neural network have been evaluated on both datasets. The baseline classifiers were selected according to the training results displayed in chapter 2. The top-3 classifiers were evaluated, and their results were compared with the references. The neural network was trained for a limited number of steps due to computation limitations. Several predicted images show the capability of the classifiers in identifying soil, crop and weed pixels. However, the networks were poorly able to describe the different types of weeds, as visualised in both the images and the data.

3.1. Baseline Classifiers

Three baseline classifiers were evaluated, namely the Multiclass Logistic Regression, Random Forest Classifier and Support Vector Machine. For each classifier, hyperparameters were optimized during the training stage in order to achieve the highest training accuracy. In some cases, cross validation was applied to verify the training accuracy. Apart from the classifier parameters, the pre-processing and post-processing steps were alternated to verify the effects on the classification results. Appendix A shows the available classes in the annotation files for the SB2016 dataset. In appendix B, the annotated pixel balance is presented for the train and test set. From here, it becomes clear that there is a severe class imbalance, favouring the crop and weed1 class. The soil class is the largest class but is mostly filtered out during the masking procedure. All other classes, weed2 to weed16, show a 5% or less presence within the vegetation group.

3.1.1. Results on CWFID Classification

For CWFID, the obtained results will be compared with two baseline classifiers of Cereda: the SVM and RFC. Cereda [3] used a window size of 80 and grid size of 10 pixels for classification of CWFID. His results of crop vs. weed, from split 2 (67% training, 33% testing, random split), are displayed in Table 5. For both the unsmoothed and smoothed predictions, the SVM outperforms the RFC by a small margin on this dataset.

UNSMOOTHED					
CLASSIFIER	Accuracy	Precision	Recall	F1	Jaccard
RFC	0.855	0.886	0.888	0.881	0.879
SVM	0.87	0.887	0.911	0.894	0.879
SMOOTHED					
RFC	0.871	0.897	0.9	0.893	0.879
SVM	0.894	0.904	0.93	0.91	0.879

Table 5 - Results from Cereda [3] *on CWFID split 2 crop vs. weed, smoothed and unsmoothed*

The only difference between the evaluation of CWFID from Cereda is the addition of the Harris features and a mask-cleaning phase. The mask-cleaning resulted in an improved Jaccard-index, whereas the added features did not contribute to better classification accuracy. The results are displayed in Figure 11 and Table 6. Prediction examples are given in Figure 12.

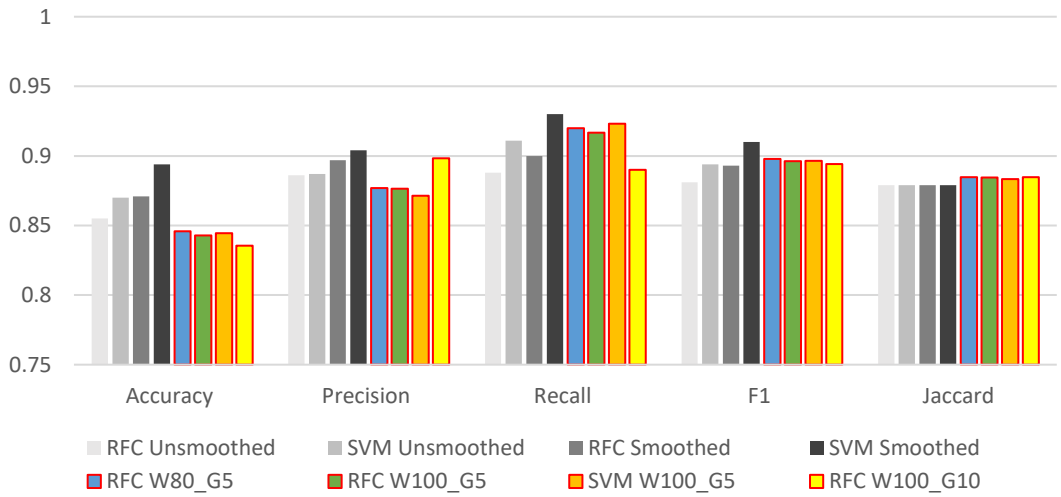


Figure 11 - Comparison of classification results on CWFID with the classifiers of Cereda (grey-scale) and the new evaluation

UNSMOOTHED					
CLASSIFIER	Accuracy	Precision	Recall	F1	Jaccard
RFC	0.855	0.886	0.888	0.881	0.879
SVM	0.87	0.887	0.911	0.894	0.879
SMOOTHED					
RFC	0.871	0.897	0.9	0.893	0.879
SVM	0.894	0.904	0.93	0.91	0.879
SMOOTHED NEW					
RFC W80_G5	0.846	0.877	0.920	0.898	0.885
RFC W100_G5	0.843	0.876	0.917	0.896	0.885
SVM W100_G5	0.844	0.871	0.923	0.896	0.883
RFC W100_G10	0.835	0.898	0.890	0.894	0.885

Table 6 - Comparison of classification results on CWFID

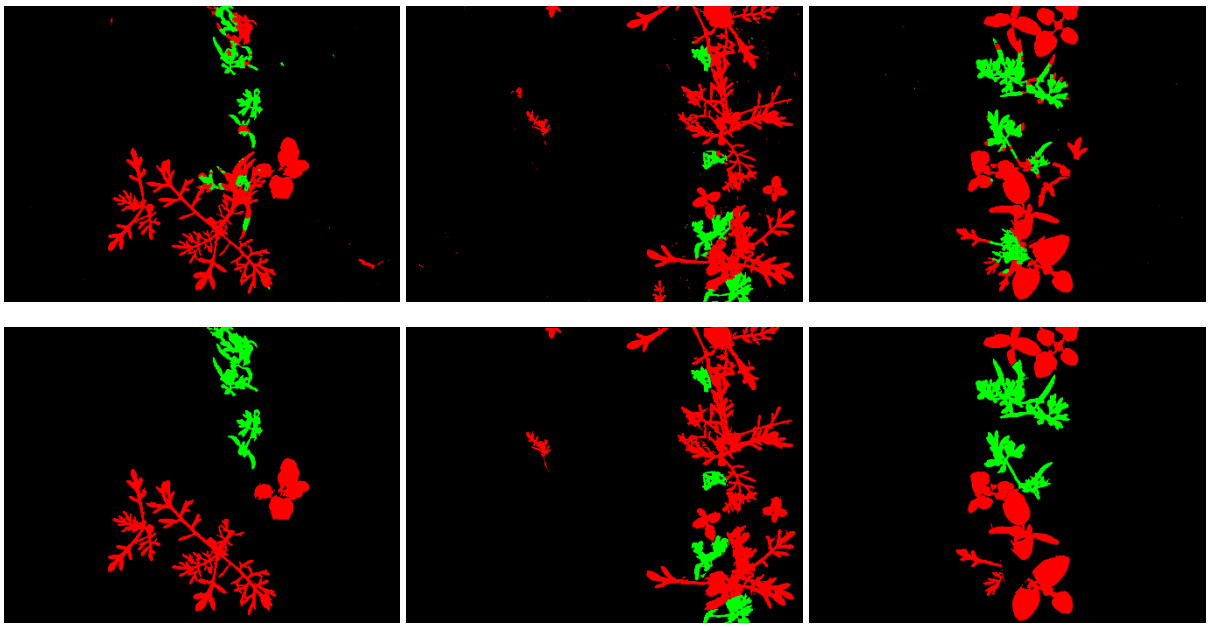


Figure 12 - Predicted images of CWFID with SVM W100_G5 (top) and annotations (bottom) of image 9, 28 and 56

3.1.2. Results on SB2016 Classification

For the SB2016 dataset there are no directly comparable results. However, the authors of [10] report testing accuracies for which they used a dataset that contained the images of SB2016, their total dataset being an extension of it (10036 images, split 70% train, 15% validation, 15% test). The results, displayed in Table 7, show three different inputs for their network with SegNet architecture. RGB-input means solemnly RGB-images; RGB+NIR also includes Near-Infrared images; 14 Indices stands for the 14 image representations that they used to feed their network, among which are HSV transform, Excessive Green, CIVE and Sobel representations (see [10], chapter 3 A for more details). Their Mean-Intersection-over-Union (M-IOU) represents the mean of soil, crop and weeds.

NETWORK	M-IOU	IOU			PRECISION			RECALL		
		Soil	Weeds	Crops	Soil	Weeds	Crops	Soil	Weeds	Crops
RGB	0.60	0.99	0.21	0.60	1.00	0.29	0.66	0.99	0.42	0.82
RGB+NIR	0.77	0.99	0.49	0.82	1.00	0.53	0.84	0.99	0.88	0.97
14 INDICES	**0.81**	**0.99**	**0.59**	**0.84**	**1.00**	**0.66**	**0.86**	**1.00**	**0.85**	**0.97**

Table 7 - Test results from [10] to compare with

The top-3 classifiers from the training phase were evaluated for prediction. Their scores were converted to the same measure as used by [10] for comparison and are visualised in Figure 13. Next to that, the scores are also indicated in the way that Cereda presented them for CWFID (Table 8). For this evaluation, all weeds were put in the same class.

SVM W100_G10 SMOOTH+THRESHOLD

	IoU	Accuracy	Precision	Recall	F1
CROP	0.87	0.90	0.89	0.97	0.93
WEED	0.29	0.90	0.94	0.77	0.85
SOIL	0.95	0.98	1.00	0.99	0.99

RFC W100_G10 SMOOTH+THRESHOLD

	IoU	Accuracy	Precision	Recall	F1
CROP	0.88	0.91	0.91	0.97	0.94
WEED	0.29	0.91	0.93	0.80	0.86
SOIL	0.95	0.98	1.00	0.99	0.99

RFC W80_G5 SMOOTH+THRESHOLD

	IoU	Accuracy	Precision	Recall	F1
CROP	0.88	0.91	0.90	0.97	0.93
WEED	0.29	0.91	0.93	0.79	0.85
SOIL	0.95	0.98	1.00	0.99	0.99

Table 8 - Classifier scores on SB2016 (crop, weed and soil)

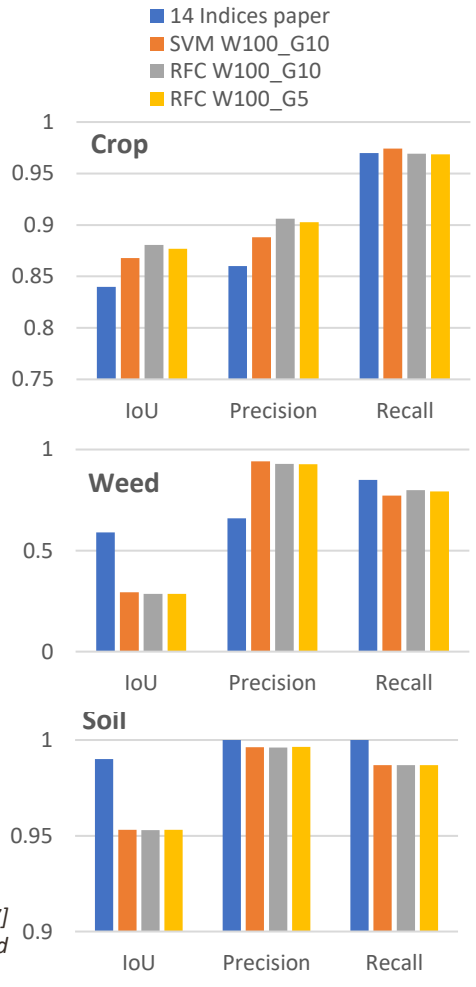


Figure 13 - Comparison with [7] on SB2016 (crop (top), weed (middle) and soil (bottom))

The results in Table 8 and Figure 13 are those after smoothing and a probability threshold of 0.5, and gave the best overall comparison scores (visualised in appendix E). The SVM W100_G10 had the

highest crop recall, which is considered important since it reduces classification of sugar beets as weeds. Overall, the classifiers seem quite balanced, outperforming the SegNet from [10] while using less training data and a smaller training split (67% versus 70% used by [10]). Apart from the joint class of weeds, the separate weed classes have been analysed. The performance appeared to be poor on prediction, as can be seen in Figure 15, and for more detail and all three classifiers in appendix C. Appendix C also contains a comparison of per-class pixel presence within the train and test set of the data. Smoothing had some mitigating effect, while thresholding reduced mainly the classification performance of the RFC's, as can be seen in Figure 14. Prediction examples are given in Figure 16.

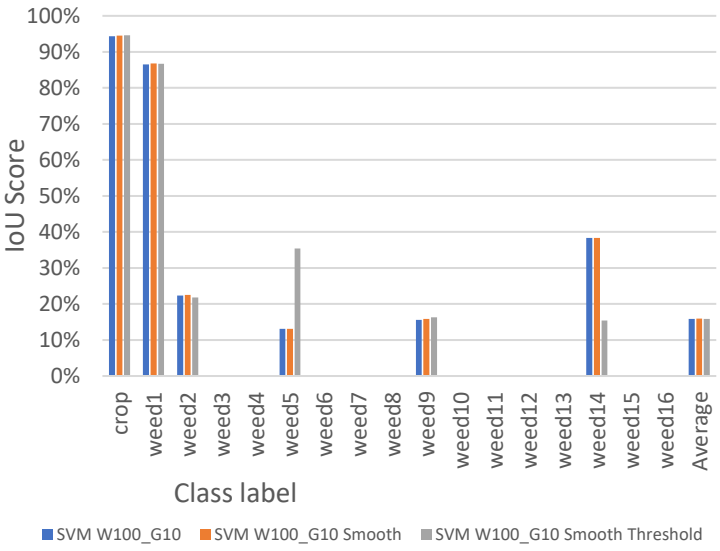


Figure 15 - Per-class IoU-score of SVM W100_G10 with and without smoothing and threshold

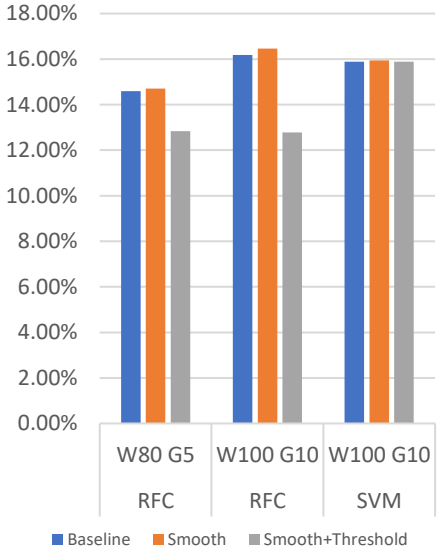


Figure 14 - Average IoU score from all vegetation labels

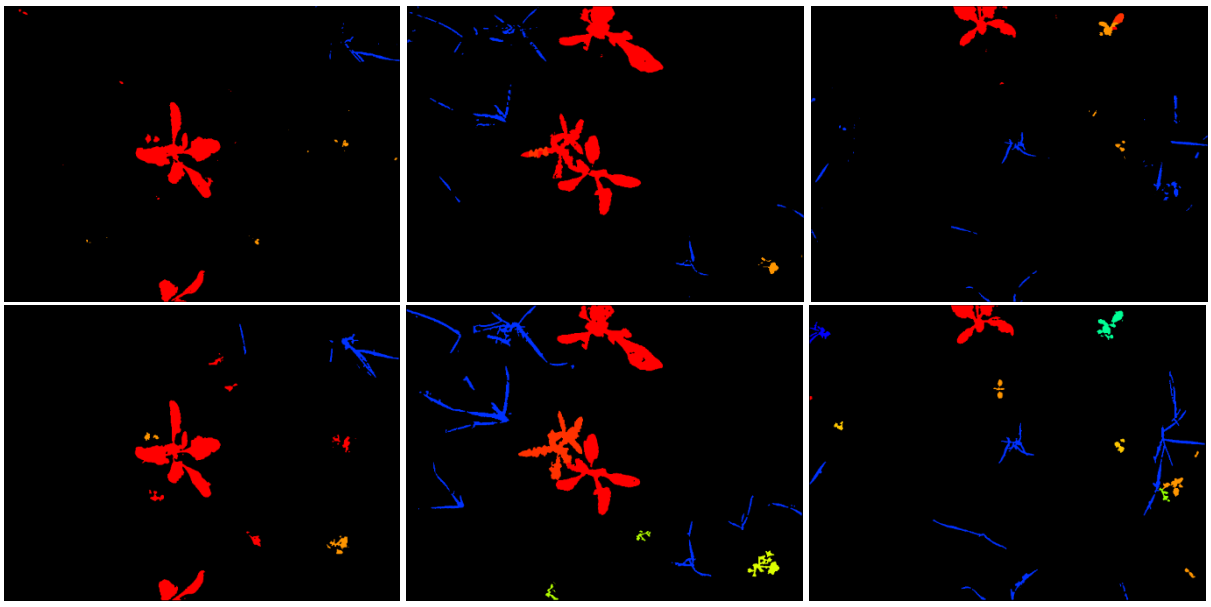


Figure 16 – Prediction examples of image 42, 73 and 122 from the test set (top) and annotation (bottom)

3.2. Neural Networks: FCN-VGG16 Results

The neural network, a Fully Connected Network with pre-trained VGG16 weights, was trained to recognize and predict images of CWFID and SB2016 for semantic segmentation. However, training was only performed for a limited number of steps due to a lack of computational power (evaluation happened on a CPU whereas evaluation of a GPU would be much faster). The results, as shown in Table 9, display a poor understanding of the different classes. The soil class seems well understood, but since around 90% of all the pixels are soil, this is not so surprising. Evaluation of the CWFID images showed some understanding of the difference between soil and vegetation (Figure 17), whereas the images of SB2016 where not clear at all (therefore not shown).

SB2016

IOU AFTER STEP	Step 300	Step 2000
SOIL	0.871565	0.9423837
CROP	0.063144	0.0093313
WEED1	0.030080	0.0014139
WEED2	0.003957	0.0001361
WEED3	0.001022	0
WEED4	0.000314	0
WEED5	0.000061	0
WEED6	0.000000	0
WEED7	0.000000	0
WEED8	0.000646	0
WEED9	0.003676	0.0001933
WEED10	0.000106	0
WEED11	0.000005	0
WEED12	0.000133	0.0006363
WEED13	0.000058	0
WEED14	0.000173	0
WEED15	0.000000	0
WEED16	0.000030	0

CWFID

IOU AFTER STEP	Step 300	Step 550
BACKGROUND	0.909008	0.914832
CROP	0.035141	0.037246
WEED	0.207400	0.219902

Table 9 - Training results for SB2016 (left) and CWFID (right)

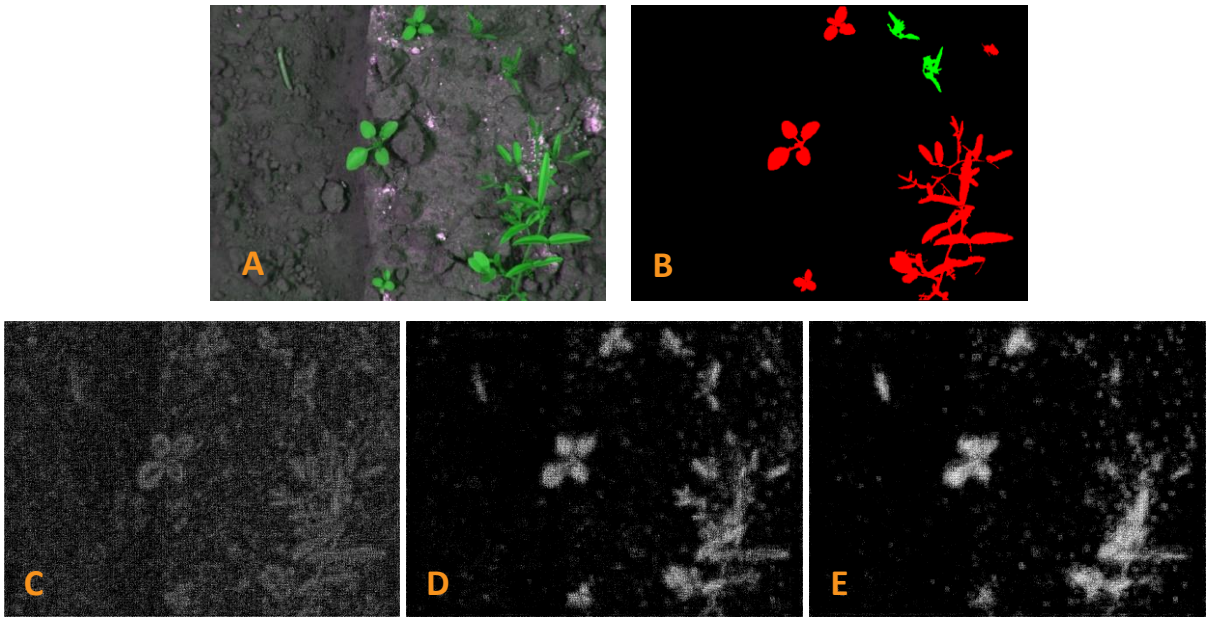


Figure 17 - CWFID image 23 (A) after 50 (C), 300 (D) and 550 (E) training steps with FCN-VGG16 with annotation (B)

4. Conclusion, Discussion and Future Possibilities

The algorithms designed by Cereda for the task of crop/weed classification were successfully transformed into a multi-class classification pipeline and applied to both datasets. The results show a similar performance on the CWFID dataset, where only small modifications in the used features and mask-cleaning were applied. Evaluation of the SB2016 dataset was compared with results from a similar dataset and showed some improvement, which is interesting considering the reduced number of training images and vastly more simple construction of the baseline classifiers, versus the 14-indices input of the SegNet architecture. However, it becomes clear that the evaluated classifiers were not able to differentiate between the different weed classes. The evaluated neural network did not perform well at all, but this is most likely due to inefficient training and a lack of tuning of the hyperparameters. Further analysis and better training might yield more promising results.

4.1. Discussion

Seeing the results of the baseline classifiers and comparing them to the work of Cereda [3] and Milioto [10], it seems that these classifiers have the ability to produce similar results. This could be a good starting point for future research, as the pipeline as well as the classifiers are easier to interpret and experiment with.

The need and design for intra-weed classification seems to require more attention. A possible strategy for this could be to use clustering or adapt classifiers to deal with a heavy class-imbalance.

4.2. Future Possibilities

In their 2017 paper on semi-supervised learning, Lottes and Stachniss [4] showed that with minimal labelling time, they were able to reach a 95% classification accuracy on sugar beet fields in Germany and Switzerland. They achieve this by using printed markers and placing them on the field, pointing towards sugar beet plants. This is the only labelling effort that they apply within their classification system. Quote: "…due to the comparably high precision for the weeds we can adjust the threshold for the class assignment according to Eq. (8) and detect around 85% of the weeds correctly at recall of 99.9% for sugar beets, which means that only 1 of 1000 crops is wrongly considered as weed by robot." An example of their work-method is displayed in Figure 18. A similar strategy would be interesting to explore, as this would allow for better generalization and thus more diverse application.

Figure 18 - Images from [2] showing the in-field labelling method

5. Bibliography

[1] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” May 2015.

[2] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” pp. 1–14, 2014.

[3] S. Cereda, “A comparison of different neural networks for agricultural image segmentation,” Politecnico di Milano, 2017.

[4] P. Lottes and C. Stachniss, “Semi-supervised online visual crop and weed classification in precision farming exploiting plant arrangement,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5155–5161.

[5] R. Bongiovanni and J. Lowenberg-Deboer, “Precision Agriculture and Sustainability,” *Precis. Agric.*, vol. 5, no. 4, pp. 359–387, Aug. 2004.

[6] S. Haug and J. Ostermann, “A Crop/Weed Field Image Dataset for the Evaluation of Computer Vision Based Precision Agriculture Tasks,” Springer, Cham, 2015, pp. 105–116.

[7] N. Chebrolu, P. Lottes, A. Schaefer, W. Winterhalter, W. Burgard, and C. Stachniss, “Agricultural robot dataset for plant classification, localization and mapping on sugar beet fields,” *Int. J. Rob. Res.*, vol. 36, no. 10, pp. 1045–1052, 2017.

[8] S. Haug, A. Michaels, P. Biber, and J. Ostermann, “Plant classification system for crop /weed discrimination without segmentation,” in *IEEE Winter Conference on Applications of Computer Vision*, 2014, pp. 1142–1149.

[9] M. Di Cicco, C. Potena, G. Grisetti, and A. Pretto, “Automatic Model Based Dataset Generation for Fast and Accurate Crop and Weeds Detection,” 2016.

[10] A. Milioto, P. Lottes, and C. Stachniss, “Real-time Semantic Segmentation of Crop and Weed for Precision Agriculture Robots Leveraging Background Knowledge in CNNs,” Sep. 2017.

[11] E. J. van Henten, D. Goense, and C. Lokhorst, Eds., *Precision agriculture ’09*. The Netherlands: Wageningen Academic Publishers, 2009.

[12] C. Harris, C. Harris, and M. Stephens, “A combined corner and edge detector,” *PROC. FOURTH ALVEY Vis. Conf.*, pp. 147--151, 1988.

[13] OpenCV, “Harris Corner Detection — OpenCV 3.0.0-dev documentation.” [Online]. Available: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_features_harris/py_features_harris.html. [Accessed: 24-Jan-2018].

[14] D. M. D. M. Woebbecke, G. E. G. E. Meyer, K. Von K. Von Bargen, and D. A. D. A. Mortensen, “Color Indices for Weed Identification Under Various Soil, Residue, and Lighting Conditions,” *Trans. ASAE*, vol. 38, no. 1, pp. 259–269, 1995.

[15] T. Kataoka, T. Kaneko, H. Okamoto, and S. Hata, “Crop growth estimation system using machine vision,” in *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, vol. 2, pp. b1079–b1083.

[16] T. Hague, N. D. Tillett, and H. Wheeler, “Automated Crop and Weed Monitoring in Widely Spaced Cereals,” *Precis. Agric.*, vol. 7, no. 1, pp. 21–32, Mar. 2006.

[17] J. A. Marchant and C. M. Onyango, “Shadow-invariant classification for scenes illuminated by

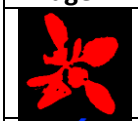
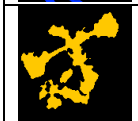
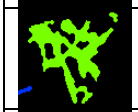
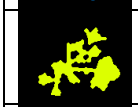
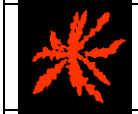
daylight," *J. Opt. Soc. Am. A*, vol. 17, no. 11, p. 1952, Nov. 2000.

[18] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," vol. 9, no. 1. IEEE Transactions on Systems, Man and Cybernetics, pp. 62–66, 01-Jan-1979.

[19] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation." pp. 3431–3440, 2015.

[20] Sagieppel, "Fully convolutional neural network (FCN) for semantic segmentation with tensorflow," 2017. [Online]. Available: https://github.com/sagieppel/Fully-convolutional-neural-network-FCN-for-semantic-segmentation-Tensorflow-implementation.

Appendix A – Sugar Beets 2016 Dataset Annotation Overview

Image	R G B colour code	BGR	Nr
	255, 0 , 0	0, 0, 255	1
	0, 50, 255	255, 50, 0	2
	255, 200, 0	0, 200, 255	3
	255, 150, 0	0, 150, 255	4
	120, 255, 0	0, 255, 120	5
	255, 100, 0	0, 100, 255	6
	0, 150, 255	255, 150, 0	7
	220, 255, 0	0, 255, 220	8
	180, 255, 0	0, 255, 180	9
	255, 50, 0	0, 50, 255	10
	80, 255, 0	0, 255, 80	11
	150, 150, 150	150, 150, 150	12
	0, 255, 150	150, 255, 0	13
	150, 255, 0	0, 255, 150	14
	0, 250, 255	255, 250, 0	15
	0, 255, 200	200, 255, 0	16
Weed (no example found)	0, 0, 255	255, 0, 0	17
Soil (black)	0, 0, 0	0, 0, 0	18

Appendix B – Sugar Beets 2016 Dataset Train-Test Analysis

The dataset was randomly split (seed 1) in train and test data, using 67% for training and 33% for testing. The annotations were then analysed per class, revealing a strong class imbalance.

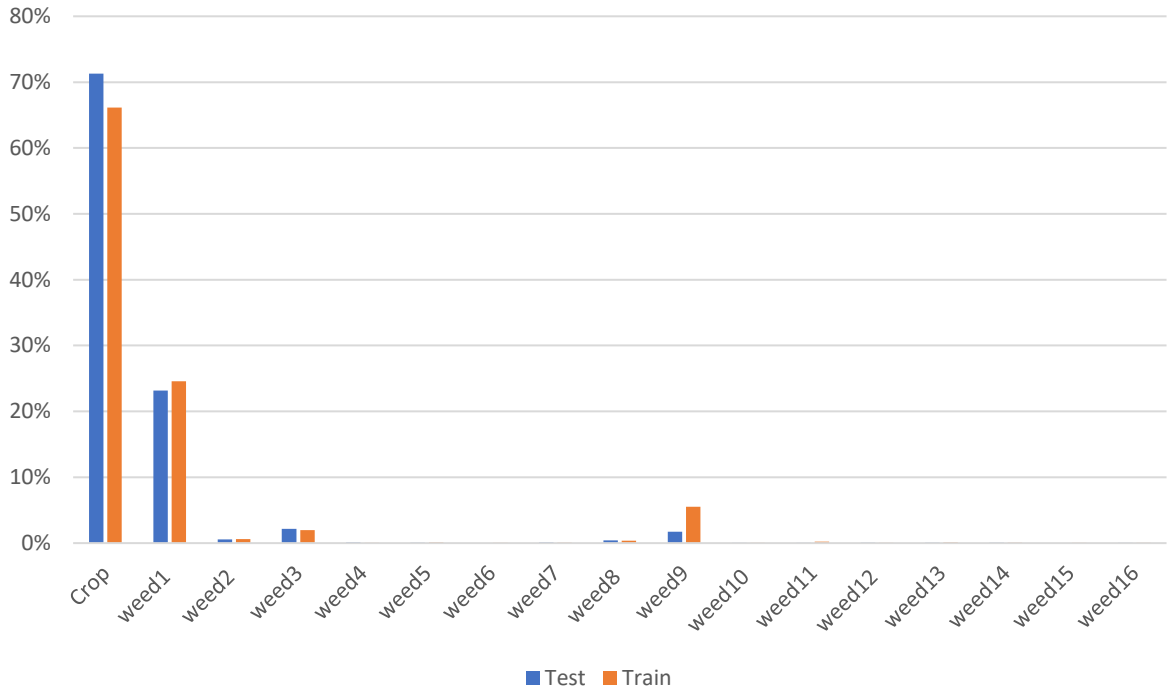


Figure 19 - Annotation pixel analysis, all classes except soil

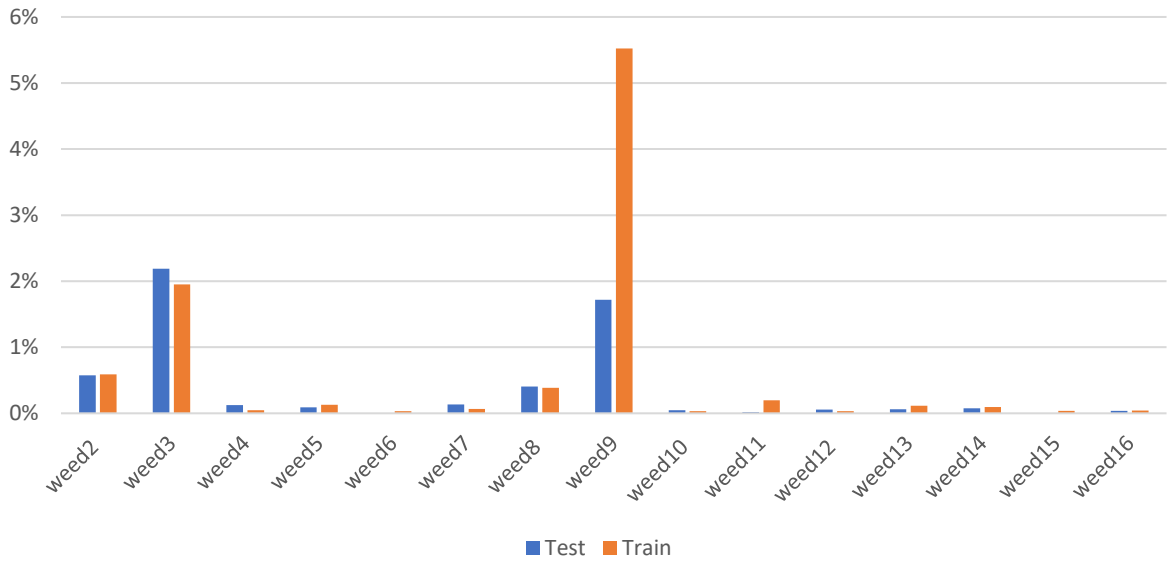


Figure 20 - Annotation pixel analysis, 15 least represented classes

Appendix C – Intersection over Union, per class

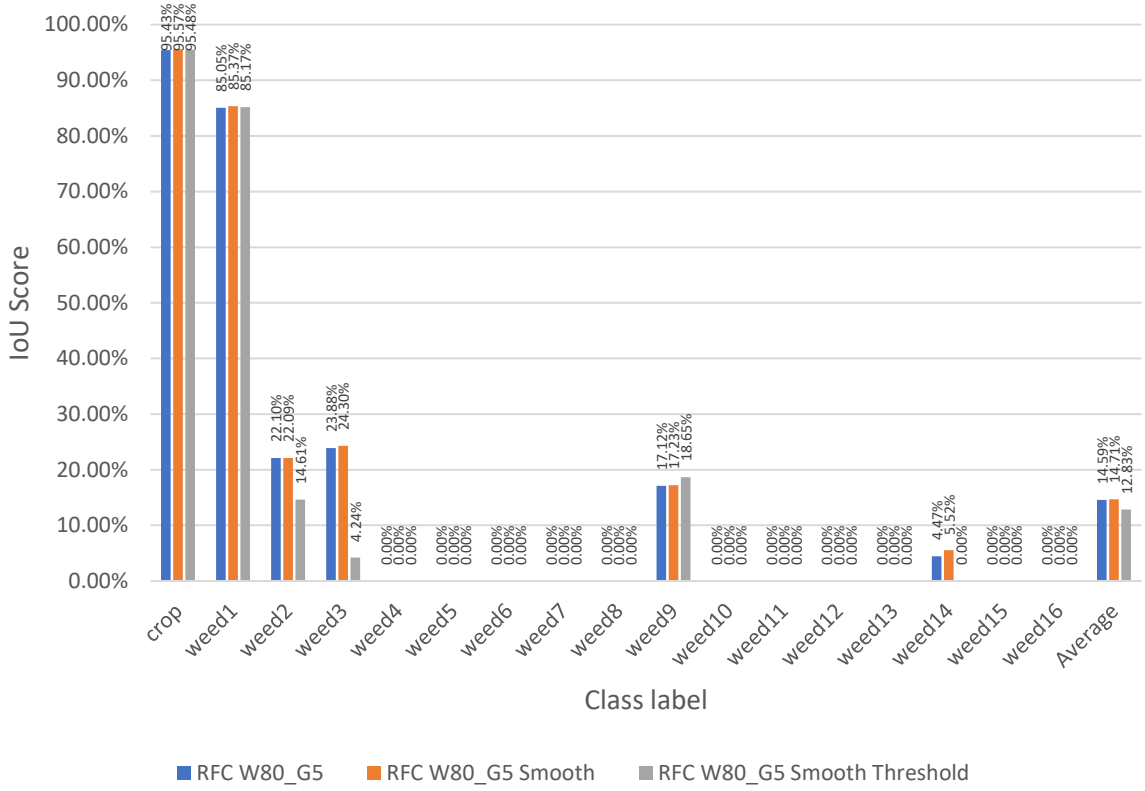


Figure 21 - RFC W80_G5 Intersection over Union, per label

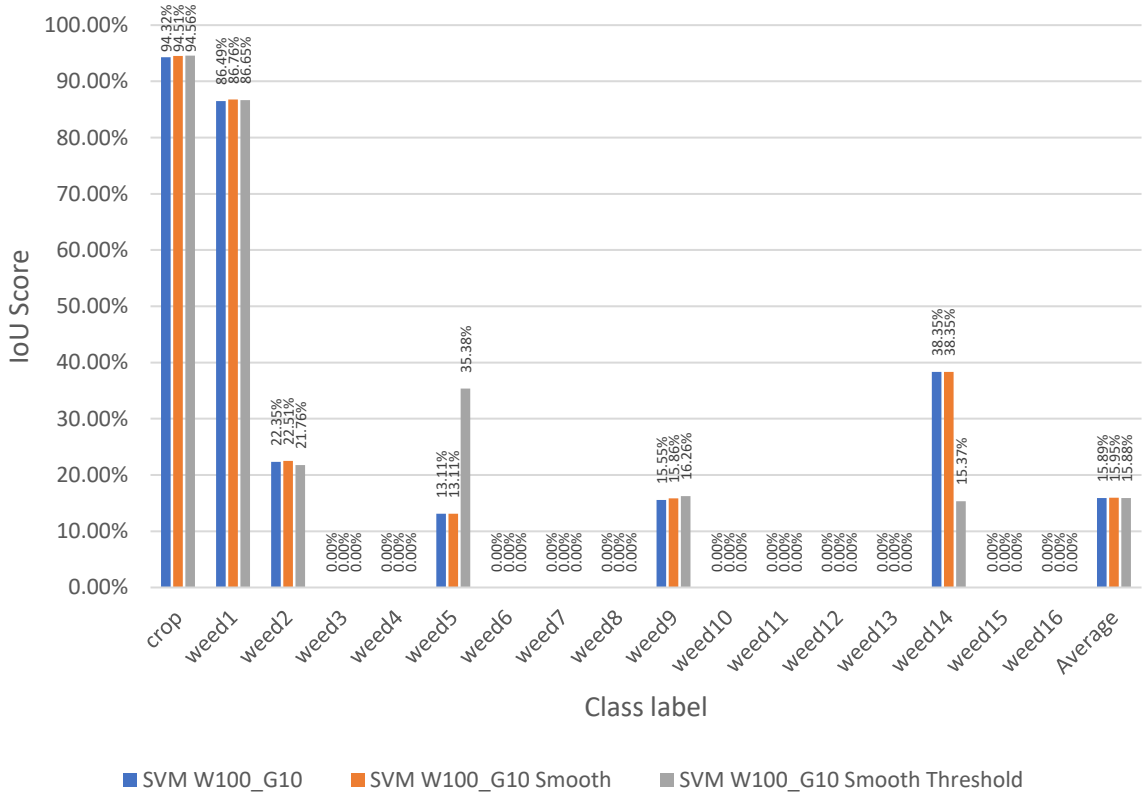


Figure 22 - SVM W100_G10 Intersection over Union, per label

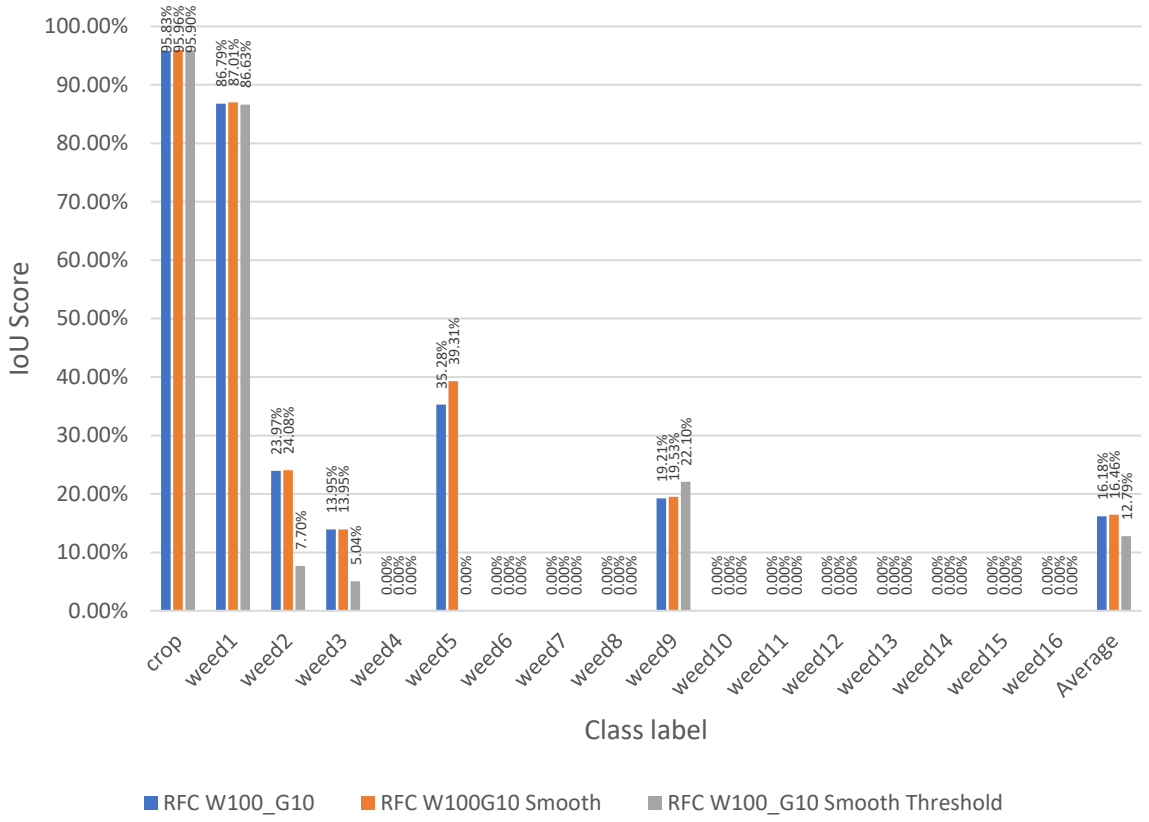


Figure 23 - RFC W100_G10 Intersection over Union, per label

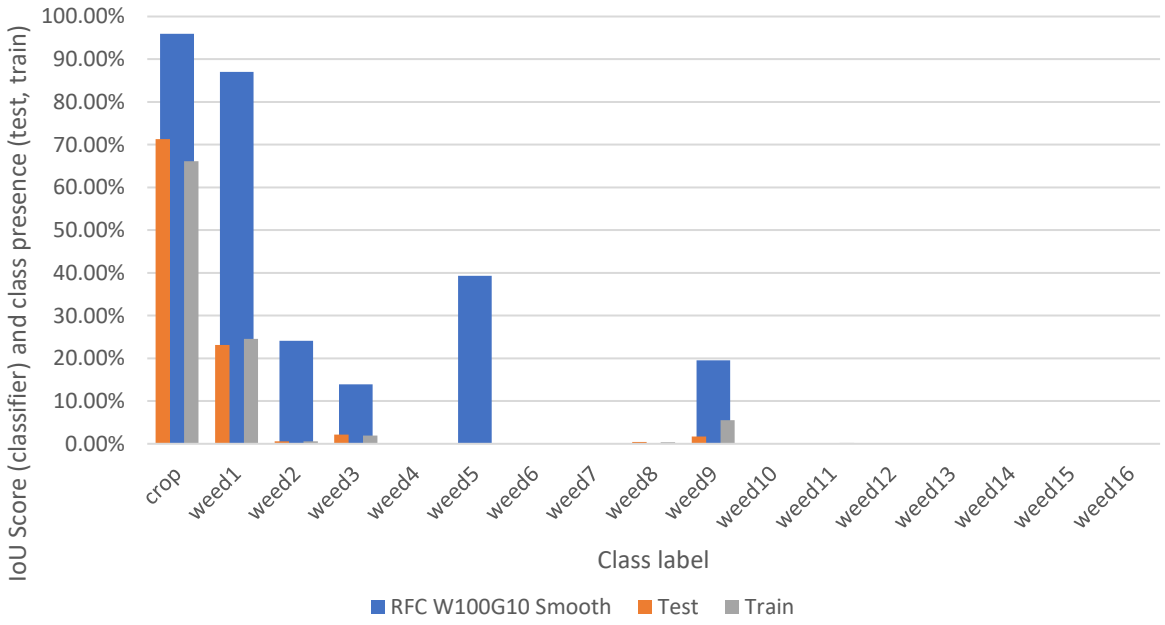


Figure 24 - IoU Score and class presence comparison

Appendix D – Mask Distortion in SB2016 Dataset

In an earlier stage, a classification challenge came to light. The image shows a potential drawback of using a NIR image for the masking process since 'bunny droppings' (or perhaps another creature, I have not been able to find an expert on this field yet) disturb the image. In *Figure 25* the comparison between the original and NIR-image is displayed. In *Figure 26* the ground-truth and prediction show significant differences due to the masking failure of much of the soil and droppings.

Figure 25 - Image 30 of the SB2016 dataset, left: original image, right: NIR-image

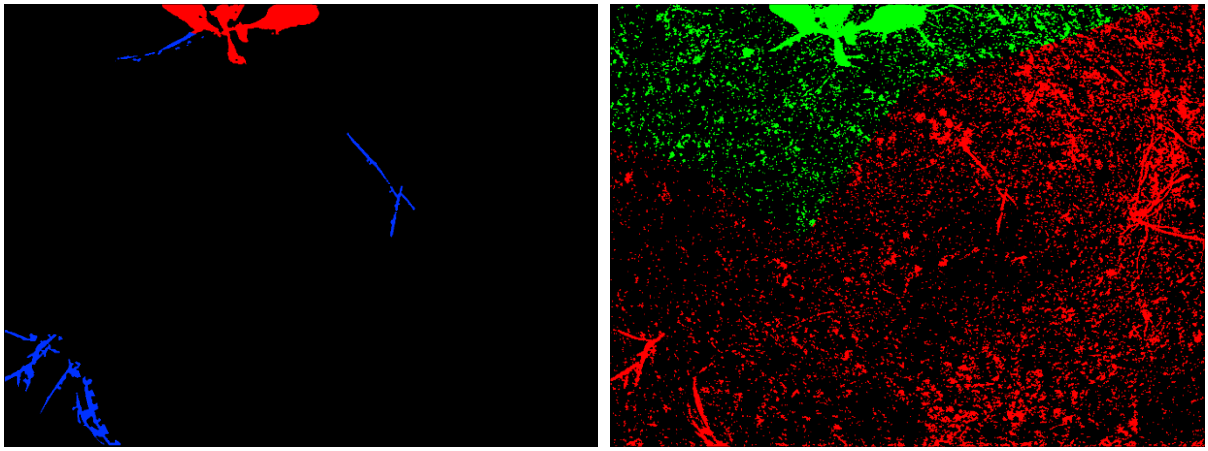


Figure 26 - Prediction of image 30 of the SB2016 dataset. Left: ground-truth, right: prediction (with 3 classes only)

Appendix E – Effects of Smoothing and Threshold

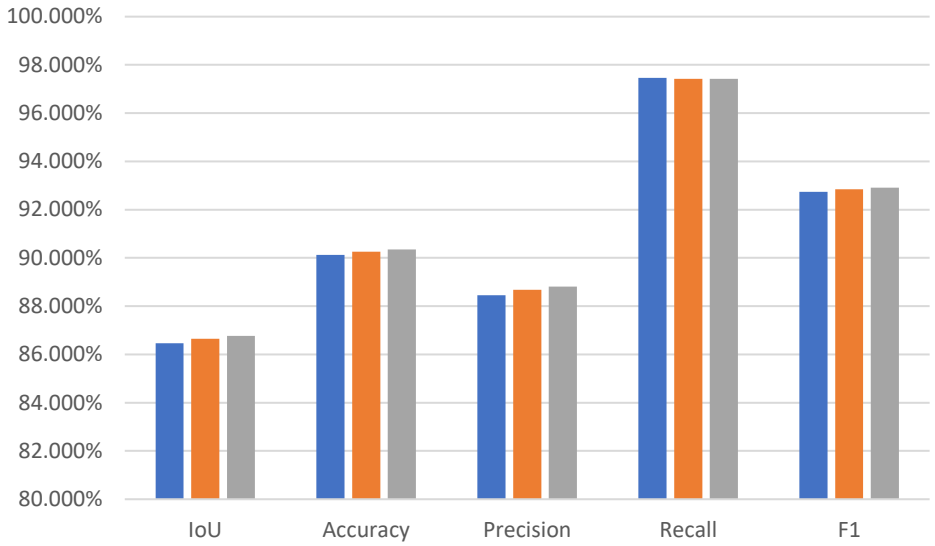


Figure 27 - SVM W100_G10: Crop vs. Weed, SB2016 results

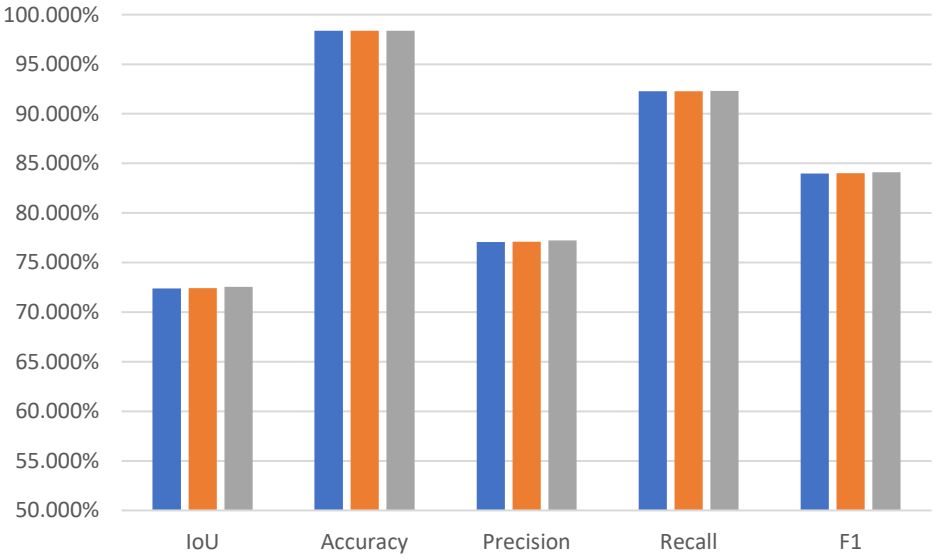


Figure 28 - SVM W100_G10: Vegetation vs. Soil, SB2016 results