Flower Mapping in Grasslands with Drones and Deep Learning

Johannes Gallmann[1] (Orcid ID: 0000-0003-4782-2768)

Beatrice Schüpbach[2] (Orcid ID: 0000-0003-4090-8155)

Katja Jacot[2]

Matthias Albrecht[2] (Orcid ID: 0000-0001-5518-3455)

Jonas Winizki[2]

Helge Aasen[3] (Orcid ID: 0000-0003-4343-0476)

Correspondence: Helge Aasen, Universitätstrasse 2, 8092 Zürich, Switzerland, helge.aasen@usys.ethz.ch

Headline: Automated Flower Abundance Mapping

---

[1] Department of Computer Science, ETH Zürich, Zürich, Schweiz

[2] Agricultural Landscape and Biodiversity Group, Agroscope, Zürich, Schweiz

[3] Department of Agricultural Science, ETH Zürich, Zürich, Schweiz

Abstract

- Manual assessment of flower abundance of different flowering plant species in grasslands is a time consuming process.
- We present an automated approach to determine the flower abundance in grasslands from drone images using a deep learning (Faster R-CNN) object detection approach, which is trained and evaluated on data of five flights and two sites. Our deep learning network is able to identify and classify individual flowers.
- The novel method allows generating spatially explicit maps of flower abundance that meets or exceeds the accuracy of the manually counted extrapolation method and is less labor intensive. The results are very good for some types of flowers with precision and recall being close to or higher than 90 %. Other flowers are detected poorly due to reasons such as lack of enough training data, appearance changes due to phenology or flowers being too small to be reliably distinguishable on the aerial images.
- The method is able to give precise estimates of the abundance of many flowering plant species. The collection of more training data will allow better predictions in the future for the flowers that are not well predicted yet. The developed pipeline can be applied to any sort of aerial object detection problems.

1 Keywords

Aerial Images, Drones, Faster R-CNN, Flower Abundance Mapping, Machine Learning, Object Detection, Remote Sensing, Unmanned Aerial Vehicles

2 Introduction

The service done by pollinators in farmlands is estimated to value more than 150 Billion Euros a year worldwide (Gallai, Salles, Settele, & Vaissière 2009). Their declining numbers (Hallmann et al. 2017) motivate many ecologists to study their interplay with the environment. This includes the assessment of flower abundance and distribution, which is an extremely time consuming task.

In the last 10 years, rapid development in sensor technology and robotics have enhanced the capabilities of unmanned aerial vehicles (UAVs) (Anderson & Gaston 2013; Pajares 2015; Sanchez-Azofeifa et al. 2017; Aasen, Honkavaara, Lucieer, & Zarco-Tejada 2018). Today it is both technologically possible and affordable to take ultra-high spatial resolution images of large areas (several deca-ha with ground resolution of 1 cm / pixel). When flying lower and slower, even resolutions of down to millimetres can be reached. Consequently, UAVs have also been used in many ecological settings. These include invasive species mapping (Martin et al. 2018; Müllerová et al. 2017; Hill et al. 2017; Kattenborn, Eichel, & Fassnacht 2019; de Sá et al. 2018), wild live assessment (Andrew & Shephard 2017; Hollings et al. 2018; Christiansen et al. 2019; Eikelboom et al. 2019; Rey, Volpi, Joost, & Tuia 2017) and plant biodiversity estimation (Getzin, Wiegand, & Schöning 2012).

Recently, deep learning based classification methods have appeared that are able to utilize the details of ultra high resolution image data. We use the Faster R-CNN object detection pipeline (Ren, He, Girshick, & Sun 2015). It utilizes deep convolutional neural networks (CNNs) to detect and classify objects in RGB images. A deep CNN is a network with many layers. It takes the pixels of an image as input and as output predicts the likelihood for each class label it has been trained on. Internally it applies thousands of learned filters to all regions of the image and in the end combines them to find the likelihood of each class label. Recently, such approaches have also been introduced to detect and count animals (Eikelboom et al. 2019; Rey, Volpi, Joost, & Tuia 2017) and plants (Eikelboom et al. 2019; Kattenborn, Eichel, & Fassnacht 2019; Osco et al. 2020) in an ecological context.

Remote flower mapping in a grassland containing many species is a challenging task, since the structures are fine and flowers might be occluded by other plants. Current approaches of automated flower mapping work with image resolutions in the range of centimeters or even meters per pixel (Landmann et al. 2015; Chen, Jin, & Brown 2019; Abdel-Rahman et al. 2015) and are therefore not suited to detect individual flowers and differentiate between flower species of similar color. Other approaches are handcrafted for a single species (Campbell & Fearns 2018; Horton, Cano, Bulanon, & Fallahi 2017) and are not applicable to a wide range of use cases.

In this article we present a deep learning based method to collect information about flower abundance and distribution in grasslands from drone images. To evaluate its performance we address several questions:

(i) Is it possible to identify flowers in overhead images with flowers spanning only a few pixels at a ground resolution of 1.5 mm per pixel using deep CNNs?

(ii) How do UAV based automated counts compare to manual in field measurements by an expert?

(iii) How does automatically generated flower abundance maps of a whole field compare to the educated guess by an expert?

3 Materials and Methods

3.1 Overview

The proposed method can be divided into the three main phases of data collection (section 3.2), model training (section 3.3) and application to unseen images (section 3.4) as depicted in fig. 1.

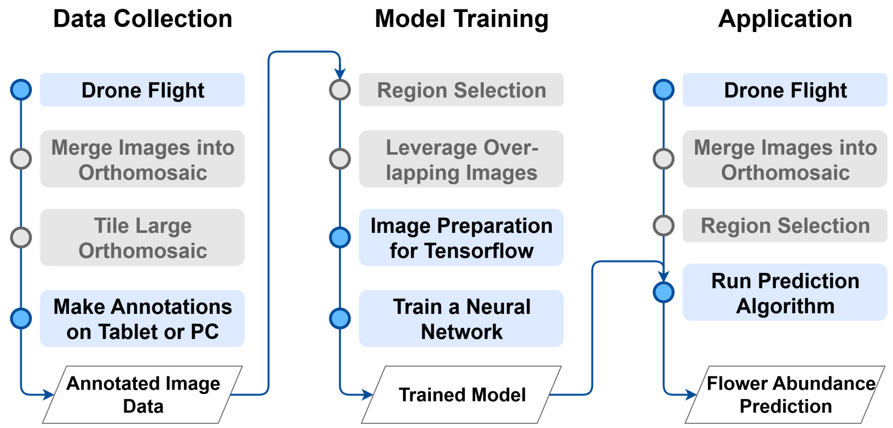


Figure 1: Overview of the proposed method. Grey colored steps might not be necessary for some use cases.

Ranunculus sp **(n = 474)**	***Lotus corniculatus*** **(3271)**	***Galium mollugo*** **(659)**	***Crepis biennis*** **(159)**	***Centaurea jacea*** **(805)**
-*Ranunculus bulbosus* (442) -*Ranunculus friesianus* (8) -*Ranunculus acris* (24)	-*Lotus corniculatus* (2926) -*Lathyrus pratensis* (345)	-*Galium mollugo* (202) -*Achillea millefolium* (338) -*Daucus carota* (65) -*Carum carvi* (54)	-*Crepis biennis* (89) -*Leontodon hispidus* (10) -*Tragopogon pratensis* (8) -*Picris hieracioides* (52)	-*Centaurea jacea* (786) -*Lychnis flos-cuculi* (19)

Table 1: Plant species that are combined into one group.

3.2 Data Collection

3.2.1 Dataset

The dataset on which the method is evaluated consists of 10000 annotated flowers. The aerial images are captured at two sites and five days from a flight height of 19 meters with a ground sampling distance of approximately 1.5 mm/pixel. For the collection of the flower dataset a drone model called TransformerUAV (Copting GmbH 2017) as well as a DJI Matrice 600 PRO (SZ DJI Technology Co., Ltd. 2018) were used. Both drones are programmed to fly along a predefined route such that the area is fully covered and the images have an overlap of 60 % to 90 %. Attached to the drone is a Sony ILCE-7RM2 (Sony Corporation 2015) camera that takes 42.2 Megapixel photos in combination with a Zeiss Batis 1.8/85 telephoto lens (Carl Zeiss AG 2017). The weather was sunny on all flight days. One of the two sites has been managed extensively during the last 15 years such that the plant diversity in this meadow is very high. A total of 40 different flowering plant species have been found between May 23rd and July 3rd. Approximately half of these 40 flower species are omitted in the analysis because too few samples are present in the dataset to reasonably train a neural network. We excluded all flowers with less than 50 samples in total from the experiments. As summarized in table 1, some flowers are combined into groups because they have few annotated samples or they look similar to other flowers. Because the individual flowers within an inflorescence can rarely be identified in the drone images, all inflorescences are annotated as one flower instance. Subsequently, when referring to the term flower, inflorescences are included as well.

3.2.2 Traditional Data Acquisition

Traditionally, information about flower abundance is acquired by counting or estimating the flowers by hand within survey plots which are distributed inside the area of interest. We used 15 survey plots that are one by one meters wide. Once the flowers present within the survey plots are counted or estimated, these numbers are extrapolated to the size of the whole area of interest. If the positions of the survey plots are well chosen, this method should produce a good estimate of the abundance of flowers. We carried out the traditional approach of manual counting in parallel to each iteration of the drone based data acquisition method and used it as a baseline.

3.2.3 Drone Based Data Acquisition

Before the drone flight, ground control points (GCPs) are placed inside the test region. For planning a proper placement of the GCPs, refer to Roth, Hund, & Aasen (2018). GCPs are small signs with a unique pattern facing upwards so that they can be recognized on the drone images. The exact GPS positions of all these GCPs are collected with a differential GPS with a precision of a few centimeters. Later they are used in the Agisoft software (Agisoft 2019) as described below. Having the GCPs in place, the drone can be flown along a predefined route across the field with a camera attached that takes a large amount of highly overlapping aerial images of the field. Using a drone with RTK GNSS potentially allows to omit the need for GCPs.

After the flight, the relative positions of the large amount of overlapping aerial images are reconstructed and merged together into a large orthomosaic. An orthomosaic is a detailed, accurate photo representation of an area, created out of many photos that are stitched together and geometrically corrected. We use the structure from motion (SfM) appraoach (Ullman 1979; Harwin & Lucieer

2012) implemented in the software Agisoft Metashape Version 1.5.3 (Agisoft 2019). Agisoft takes all aerial images as inputs. It aligns all photos and generates a point cloud model. We used a sparse point cloud. From the sparse point cloud either a digital surface model or a mesh of the topography of the research area can be created. Then, based on the topography and the reconstructed relative positions and orientations of the images, an orthomosaic is generated. We enabled the option *blending disabled* to use the original information of the images in the orthomosaic.

Agisoft automatically detects the unique pattern on the GCPs to map the GPS coordinates to each of them. The advantage of providing the positions of the GCPs in the field is that Agisoft creates an orthomosaic that is orthorectified and georeferenced. Georeferencing of the orthomosaic is later needed to display the user's position in the Android FieldAnnotator application as well as to be able to copy annotations to the single orthorectified images that are georeferenced (cf. 3.2.4 and 3.2.5 for further reading).

3.2.4 Annotating

Having an orthomosaic of the region of interest, flowers have to be annotated. The annotated flowers are needed as training data for the machine learning model. Since the survey plots needed for the traditional data acquisition are visible on the drone images, we annotated all flowers within these plots. This allows us to verify whether the number of flowers visible on the drone images are comparable to the number of flowers that are manually counted by hand.

For annotating, we use the LabelMe program (Wada 2016) and an Android tablet application called FieldAnnotator which we specifically developed for this purpose. The advantage of being able to make the annotations on a tablet is that they can be made directly in the field. This might be necessary because some flowers can be very hard to distinguish in the image alone. If one can compare the image to the actual flowers on site, the quality of the training data can be improved and it is made sure that the number of false annotations is minimized. Android tablets are not capable of handling large orthomosaics (around 50000 times 50000 pixels for a 30 times 30 meters area). Therefore a script tiles the orthomosaic into small chunks of 256 times 256 pixels in various zoom levels before these tiles are then imported into the FieldAnnotator application. The resulting annotations are stored in a json file.

3.2.5 Leveraging Overlapping Images

Since the camera attached to the drone captures a large amount of highly overlapping images, the idea is to use the overlapping images as additional training data. Since the flowers are pictured from a slightly different angle on each image and the background changes from image to image, this provides valuable additional training data. Grasslands have a very complex structure and it is hard to reconstruct the exact geometry of the images. Therefore, the copied annotations are slightly shifted within the overlapping images. To correct for the shift, a script lets the user view and adjust all annotations in the LabelMe application. These slight adjustments of the annotations take significantly less time than collecting new data.

3.3 Model Training

3.3.1 Selecting Regions of Interest in Annotated Images

In case images are only partly annotated, we developed a script that allows the user to cut out certain regions (polygon shaped) from the images. Only the image pixels within these selected regions are kept while the rest of the image pixels are overridden with black. This ensures that the Tensorflow model (Abadi et al. 2015) does not learn to classify non-annotated flowers as the background class.

3.3.2 Image Preparation for Tensorflow

The training data consisting of image files alongside with json files containing the annotations has to be converted into a format that is supported by Tensorflow. To do so, our pipeline automatically carries out the subsequent steps. First, the images are split up into tiles. The default tile size is set to 450 times 450 pixels. These image tiles are then upscaled by a factor of two to 900 times 900 pixel tiles as suggested by Hu & Ramanan (2017) and justified in supplementary material A.1.1. The tiles are overlapping such that flowers positioned on the edge of two tiles are not lost as training data but are always present as a whole in at least one tile. Additionally, all annotations (including point and polygon annotations) are converted to bounding boxes. Finally, the images are split up into train, test and validation set.

3.3.3 Neural Network Training

The core of the pipeline consists of a CNN. We use the Faster R-CNN architecture. This architecture outputs the bounding box coordinates of the objects it recognizes on an input image. The Faster R-CNN architecture requires more compute power than other architectures but it has been shown that it performs well on aerial and other high resolution images (Carlet & Abayowa 2017; Huang et al. 2017). Since the default configuration of the Faster R-CNN architecture is not optimized to detect very small objects (Huang et al. 2017; Zhang et al. 2017) of only a few pixels in diameter, such as flowers in aerial images, we adjusted some parameters (cf. A.1.1 in the supplementary materials for experiment results on different parameter combinations). Additionally, we use data augmentation techniques to increase the diversity of our dataset. The following data augmentation options are used: random horizontal and vertical flips, random brightness adjustments, random contrast adjustments, random saturation adjustments and random box jittering.

During training, the validation set is used to decide when to change the learning rate and when to stop training. Every 2500 steps the training is paused and the prediction algorithm followed by the evaluation algorithm is run on the validation set. The learning rate is adjusted if for the last 15000 steps no further improvements were made. After adjusting the learning rate two times from 3×10^{-4} to 3×10^{-5} and from 3×10^{-5} to 3×10^{-6}, the training is stopped if for 15000 steps again no improvement on the performance has been made. The number of 15000 steps is chosen empirically. Reducing the learning rate twice by a factor of 10 is directly adapted from the Faster R-CNN default configuration. The evaluation metric can be chosen as either the F1 score or the mean average precision (mAP). Section 3.4 further explains the prediction and evaluation process.

The number of training examples can vary heavily from class to class. Therefore each class is assigned a weight. The weight is inversely proportional to the number of training examples and influences the loss function during training. This ensures that the network does not just optimize to detect the most common classes. Each mistake on a less common class has a much higher penalty to the loss function as a consequence. Once a network is fully trained it can be exported as an inference graph. This exported inference graph is then used by the prediction and evaluation scripts described in section 3.4.

3.4 Application to Unseen Images

3.4.1 Predictions

The trained network can be used to make predictions on images of arbitrary size (e.g. orthomosaics) provided they have a similar ground sampling distance to the training images. The pipeline handles the tiling of large images as well as the reassembling of the prediction results from the single tiles. Optionally a region of interest can be selected within an image. As a consequence, only the flower abundance within this region of interest is assessed by the prediction algorithm.

The prediction algorithm draws the bounding boxes of all detected flowers onto the image and saves the statistics about the flower abundance to a json file. To improve the prediction accuracy, the tiles have an overlap of 100 pixels by default. This ensures that as long as a flower is not larger than 100 pixels in diameter, it is fully visible on at least one tile. Error prone predictions close to or on the edge of a tile can therefore be ignored because they are fully covered on the adjacent tile. Nevertheless, having this overlap introduces the problem of duplicate predictions. This is mitigated by applying non maximum suppression with an intersection-over-union threshold of 0.3 similar to Ozge Unel, Ozkalayci, & Cigla (2019). Meaning that for all predictions that have an overlap of more than 30 %, only the one with the highest confidence score is kept.

3.4.2 Evaluations

To evaluate the performance of a model, the predictions on the test set are compared to the ground truth of the test set. The main metrics of interest are precision and recall. To compute precision and recall values, the true positive (TP), false positive (FP) and false negative (FN) predictions have to be known. In order to obtain these values the predictions are sorted by their confidence. Then it is looped through all the predictions and each of them is compared to all ground truth bounding boxes of the same label. To compare two bounding boxes, the intersection-over-union (iou) formula is used:

$$iou = \frac{intersection\ area}{area\ of\ union}$$

If the greatest iou value is greater than some threshold (default of 0.3), the corresponding ground truth box is marked as used and the prediction is marked as true positive. If the largest iou value is lower than the threshold, the prediction is marked as false positive. After this process is done for each prediction, all ground truth entries that are not marked as used are counted as false negatives. Having the TP, FP and FN numbers, the precision and recall values can easily be calculated using the following formulas:

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

A good way to rate the performance of a model is to compute the F1 score. The F1 score is calculated as follows:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

The better the precision and recall values are, the better the F1 score gets. It rates precision and recall equally and reaches its maximum of 1 at perfect precision and recall. As an alternative to the F1 score, the mean average precision (mAP) as defined in the PASCAL VOC Challenge Development Kit (Everingham & Winn 2011) can be used to rate a model's performance.

3.4.3 Visualizations

The pipeline offers various options for visualizing the results. Apart from drawing the predictions as colored bounding boxes onto the images, erroneous predictions can be highlighted. Additionally, heatmaps that visualize the density distribution of the flowers can be generated from the prediction output. The size of the kernel for the flower density mapping is customizable. Optionally the heatmap can be drawn directly onto the image. The heatmaps can be generated for an individual class or for all classes. If the input images are georeferenced, there is the option to generate one heatmap from a collection of images. If the images are overlapping, the heatmap indicates the average number of flowers found at a particular position. Furthermore, the user can provide the geo coordinates of the upper left and lower right corner of the desired output region. The script will then output a heatmap of exactly that region. This allows for time series generations. Example results of such time series generations can be viewed in section 4.3.

4 Results

4.1 Manual Counting vs Drone Image Based Tablet Annotations

Since the exact same areas are annotated on the tablet as they are manually counted by hand, we are able to directly compare the numbers of flowers annotated on the drone images on the tablet to the numbers of flowers manually counted by hand. Table 2 lists a representative subset of all flowers found within the test fields.

Some flowers are hardly visible on the drone images and therefore significantly less instances are counted in the tablet annotations compared to the

Flower	**Manually Counted**	**Tablet Annotations**
Leucanthemum vulgare	724	960
Onobrychis viciifolia	483	105
Lotus corniculatus	1943	748
Salvia pratensis	142	127
Ranunculus sp	431	474
Knautia arvensis	371	471
Trifolium pratense	129	72
Medicago lupulina	117	5
Centaurea jacea	25	28

Table 2: Comparison of selected manually counted total numbers to tablet annotations.

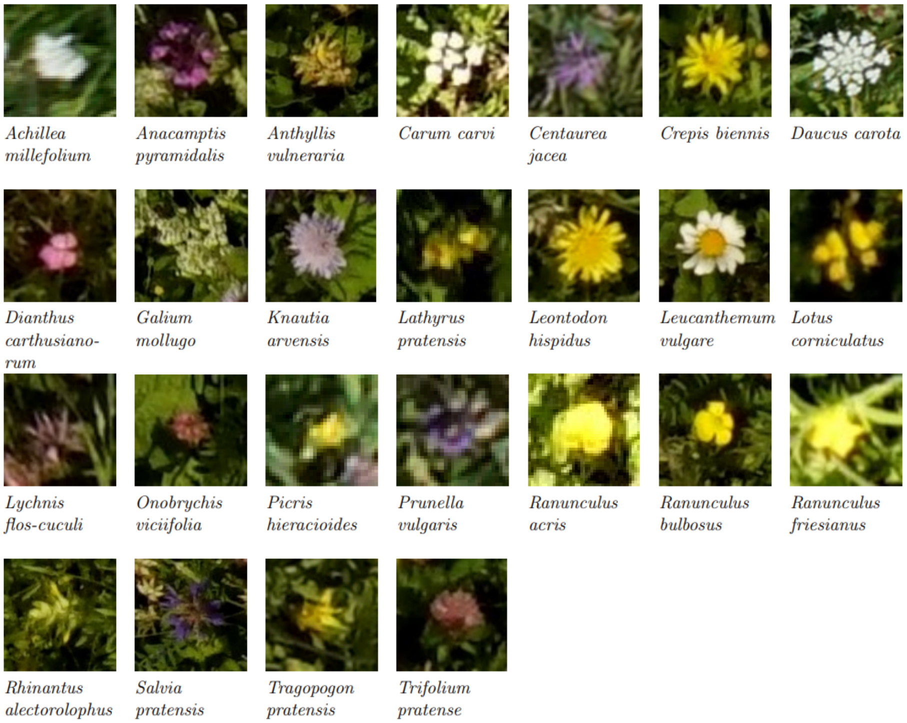
manually counted data. *Onobrychis viciifolia*, *Medicago lupulina* and to some extent *Trifolium pratense* fall under this category. The flowers of *Medicago lupulina* are too small to be reliably identifiable on the drone images. *Trifolium pratense* and *Onobrychis viciifolia* would be large enough but often they are hardly distinguishable from the background. Refer to table 2 for visualizations of 25 flowers found within the test fields. For other flowers (*Leucanthenum vulgare*, *Ranunculus sp*, *Knautia arvensis* and *Centaurea jacea*) there are more flowers annotated on the tablet than manually counted by hand (cf. section 5.1).

4.2 Prediction on a Meadow

The idea of the experiments in this section is to simulate an as realistic as possible situation. The data of one of the five flights is entirely used as test data. 90 % of the data of all other flights is used as training data and 10 % as validation data.

4.2.1 Performance inside Survey Plots

We compared the tablet annotations with the deep learning predictions within the survey plots. The prediction performance for each flower can be obtained from table 4. A prediction is considered if its confidence score is greater than



Achillea millefolium
Anacamptis pyramidalis
Anthyllis vulneraria
Carum carvi
Centaurea jacea
Crepis biennis
Daucus carota
Dianthus carthusianorum
Galium mollugo
Knautia arvensis
Lathyrus pratensis
Leontodon hispidus
Leucanthemum vulgare
Lotus corniculatus
Lychnis flos-cuculi
Onobrychis viciifolia
Picris hieracioides
Prunella vulgaris
Ranunculus acris
Ranunculus bulbosus
Ranunculus friesianus
Rhinantus alectorolophus
Salvia pratensis
Tragopogon pratensis
Trifolium pratense

Figure 2: Excerpts from aerial images of the most common flowers.

0.2. The overall precision and recall are 87 % and 84.2 % respectively. The vast majority of the flowers present in the test data of June 14th are *Knautia arvensis*, *Leucanthemum vulgare* and *Lotus corniculatus*. These three flowers perform well and therefore the good overall score is mainly determined by these three flowers. All the other flowers perform worse than the overall performance indicates.

Table 3 shows the confusion matrix of this experiment. It is striking that there are only a few confusions between different flowers (brown). The much more common cases are that flowers are predicted where there are none (red) and flowers are not predicted where they should be (orange). The green entries denote the correctly predicted flowers.

Table 4 shows that the flowers with little training data tend to not perform well. The question is whether this is due to the lack of enough training data or because assigning an inversely proportional weight to each class during training

	A. vulneraria	***C. jacea***	***C. biennis***	***D. carthusianorum***	***G. mollugo***	***K. arvensis***	***L. vulgare***	***L. corniculatus***	***O. viciifolia***	***P. vulgaris***	***Ranunculus sp***	***R. alectorolophus***	***S. pratensis***	***T. pratense***	**False Negatives**
A. vulneraria	1	-	-	-	-	-	-	3	-	-	-	-	-	-	2
C. jacea	-	27	-	-	-	17	-	-	-	1	-	-	3	3	2
C. biennis	-	-	14	-	-	-	-	5	-	-	-	-	-	-	2
D. carthusianorum	-	3	-	8	-	1	-	-	10	-	-	-	-	6	6
G. mollugo	-	-	-	-	8	-	-	-	-	-	-	-	-	-	8
K. arvensis	-	-	-	-	-	412	1	-	-	-	-	-	2	-	23
L. vulgare	-	-	1	-	1	4	906		-	-	-	-	1	-	109
L. corniculatus	-	-	6	-	-	-	1	877	-	-	-	-	-	-	142
O. viciifolia	-	1	-	-	-	1	-	-	45	-	-	-	-	11	37
P. vulgaris	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Ranunculus sp	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R. alectorolophus	-	-	-	-	-	-	-	1	-	-	-	8	-	-	17
S. pratensis	-	-	-	-	-	-	-	-	-	-	-	-	12	-	3
T. pratense	-	-	-	-	-	1	-	-	-	-	-	-	-	4	2
False Positives	4	6	17	-	32	24	31	117	3	-	1	5	6	18	-

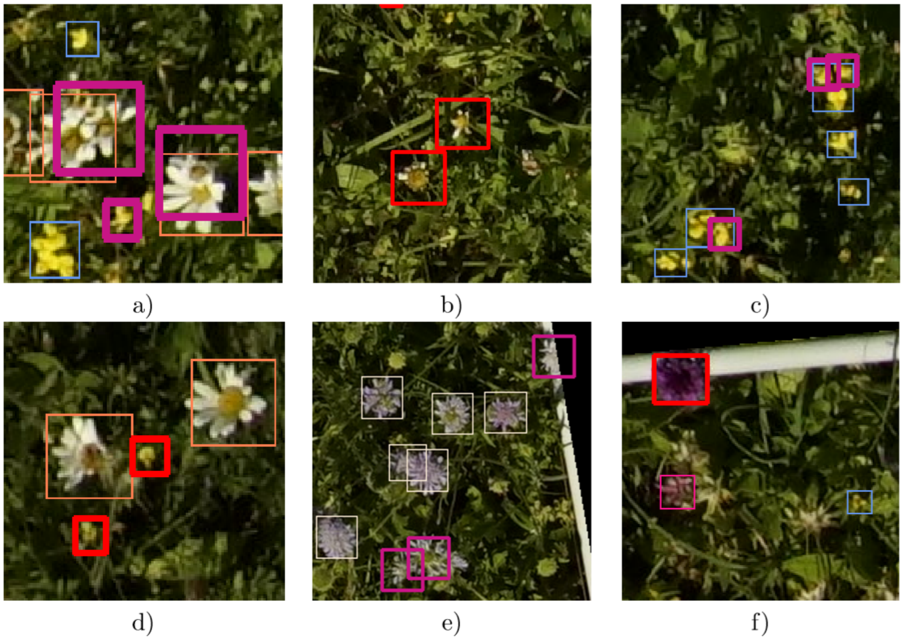
Table 3: The table shows the confusion matrix. The columns represent what the model predicted and the rows represent what the model should have predicted (the ground truth). The green, red, orange and brown numbers denote TP, FP, FN and confusions between two flowers respectively.

is not sufficient to regularize the loss function. Therefore we trained a separate network in which the three best performing flowers (*Leucanthemum vulgare*, *Lotus corniculatus* and *Knautia arvensis*) are ignored and treated as background. With the mAP rising from 25.2 % to 31.5 % (f1 score improves from 47 % to 51.3 %) a certain improvement can be seen but the performance is still significantly below what is satisfactory. Therefore the possibility of leveraging two separately trained networks is not further evaluated.

When looking at the predictions, there are various sources of errors apparent. Some examples can be seen in fig. 3. For *Leucanthemum vulgare*, a typical error occurs where two instances are very close to each other as in image a). In that case often only one of the two flowers is detected. The missing annotation is not caused by the non maximum suppression algorithm as a closer look discloses. Another typical source of errors are flowers that are on the verge of fading. In the case of image b) two flowers are detected that are not annotated in the ground truth because the botanical expert considered the flowers to be faded already. Even when manually counting the flowers by hand it is sometimes difficult to decide if a flower should be counted or not because of

Flower	**Train Instances**	**Test Instances**	**Precision**	**Recall**	**mAP**	**F1 Score**
A. vulneraria	196	6	20.0 %	16.7 %	0.056	0.182
C. jacea	742	53	73.0 %	50.9 %	0.382	0.6
C. biennis	124	21	36.8 %	66.7 %	0.325	0.475
D. carthusianorum	20	34	100.0 %	23.5 %	0.235	0.381
G. mollugo	546	16	19.5 %	50.0 %	0.1	0.281
K. arvensis	429	438	89.6 %	94.1 %	0.879	0.918
L. vulgare	928	1022	96.5 %	88.6 %	0.861	0.924
L. corniculatus	2153	1026	87.4 %	85.5 %	0.772	0.864
O. viciifolia	92	95	77.6 %	47.4 %	0.407	0.588
R. alectorolophus	23	26	61.5 %	30.8 %	0.218	0.41
S. pratensis	133	15	50.0 %	80.0 %	0.436	0.615
T. pratense	109	7	9.5 %	57.1 %	0.104	0.163
Overall	5495	2759	87.0 %	84.2 %	0.398	0.855

Table 4: Performance of the prediction algorithm on all flower species present in the field on June 14th. The numbers in the *Train Instances* and *Test Instances* columns refer to the ground truth annotations. The overall scores of the performance metrics are weighted means.



a) b) c) d) e) f)

Figure 3: Selection of typical mispredictions. All thin bounding boxes are correct predictions. The bold red bounding boxes denote false positive and the bold violet bounding boxes denote false negative predictions. There are various explanations for the mispredictions: Overlapping flowers (a), partially withered flowers (b and e), collections of flowers (c), missing ground truth annotations (d) and flowers that are missing in the training data (f).

the seamless transition from blooming to faded. Two main problems exist for *Lotus corniculatus*. Firstly, the blooms of *Lotus corniculatus* are often arranged as small inflorescences as visible in image a) in the bottom left or in image c). In some cases the network predicts the blooms of an inflorescence as individual instances while in the ground truth the whole inflorescence is annotated as one instance. The opposite case is common as well. The second problem of *Lotus corniculatus* are false positive predictions caused by missing ground truth annotations (as in image d)). These problems are further discussed in section 5.1. The main error source of *Knautia arvensis* is again blooms that look different because they are wilting as for example in image e). In image f) the model erroneously predicts a *Knautia arvensis* where there is a *Anacamptis pyramidalis*. *Anacamptis pyramidalis* is not included in the training because too few training instances exist.

4.2.2 Performance outside Survey Plots

We compared the predictions of the deep learning model on the full test field to the extrapolation of the manually counted flowers. The numbers of manually counted flowers are extrapolated to the size of the whole field which is 730 square meters. Table 5 lists all flowers that are detected reasonably well inside

Flower	**Drone Based Prediction**	**Extrapolation of Manual Counting**	**Relative Difference**
Centaurea jacea	456	505	10.7 %
Knautia arvensis	8059	8308	3.1 %
Leucanthemum vulgare	7044	10778	53.0 %
Lotus corniculatus	50365*	51139	1.5 %
Onobrychis viciifolia	595	3761	532 %
Salvia pratensis	209	673	222 %

Table 5: Predictions on the whole field of 730 square meters. The 50365 predicted *Lotus corniculatus* are calculated as the multiplicative of the actual predictions of the network (19389) and a ratio of 2.6. The numbers in table 2 suggest that there are 2.6 blooms per prediction on average.

the survey plots by the deep learning model. For each flower species the number of deep learning detections in the whole field is listed as well as the number of flowers predicted by the extrapolation of the manual counting.

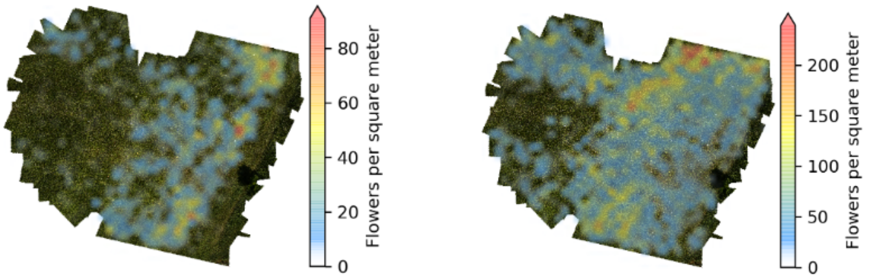
For *Centaurea jacea*, *Knautia arvensis* and *Lotus corniculatus* the number of drone based predictions is very similar to the extrapolation of the manually counted number of flowers. The results are within 11 %, 3 % and 2 % respectively. According to heatmaps generated from the drone based predictions (cf. section 4.3), these are also the flowers that are relatively evenly distributed. The extrapolation of the manually counted number of *Leucanthemum vulgare* is 53 % higher than the number of drone based predictions. The question is, which prediction is more accurate. Assuming that the performance of the prediction algorithm is similar on the whole field as it is inside the annotated survey plots, the extrapolation of the manual counting must be inaccurate. Even when adding 8 % to the number of drone based predictions to compensate for the relatively low recall value of *Leucanthemum vulgare*, the results still have a 47 % gap. The extrapolation is based on the manually counted number of flowers which is lower than the number of tablet annotations within the survey plots as pointed out in section 4.1. If the tablet based numbers were taken, the result of the extrapolation would be an additional 51 % higher making them a total of 131 % higher than the drone based predictions.

The main reason for the bad results of *Onobrychis viciifolia* is that it is very hard to distinguish on the drone images. The most probable reason for the unsatisfactory results of *Salvia pratensis* is that the amount of training data is too low to accomplish good results. A likely additional reason is an unrepresentative choice of survey plot locations for these flowers.

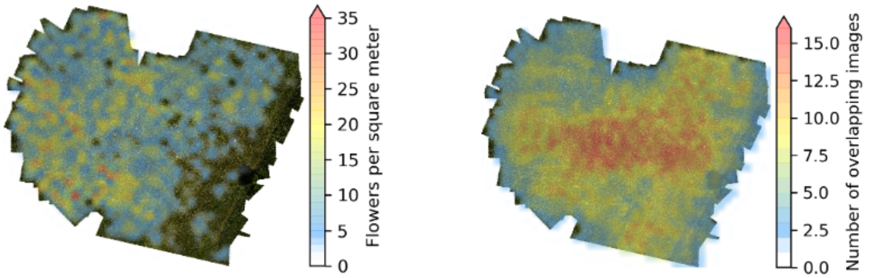
4.3 Density Distribution Maps

The heatmaps in fig. 4 depict the abundance of some selected individual flowers in one of our test fields on June 14th. The three heatmaps for *Leucanthemum vulgare*, *Lotus corniculatus* and *Knautia arvensis* are generated from the orthomosaic.

Table 5 contains a time series of an excerpt of our main test site. It illustrates the difference of the abundance evolution of *Leucanthemum vulgare* and *Lotus corniculatus*. It is conspicuous that *Lotus corniculatus* is much more evenly distributed than *Leucanthemum vulgare*. While *Leucanthemum vulgare* has a peak population on June 6th, on July 3rd the population is almost completely



Leucanthemum vulgare　　*Lotus corniculatus*



Knautia arvensis　　Image coverage

Figure 4: Heatmaps of our main test site for various flower species. The last image depicts the image coverage of the field. In the images, survey plots as well as GCPs are visible.

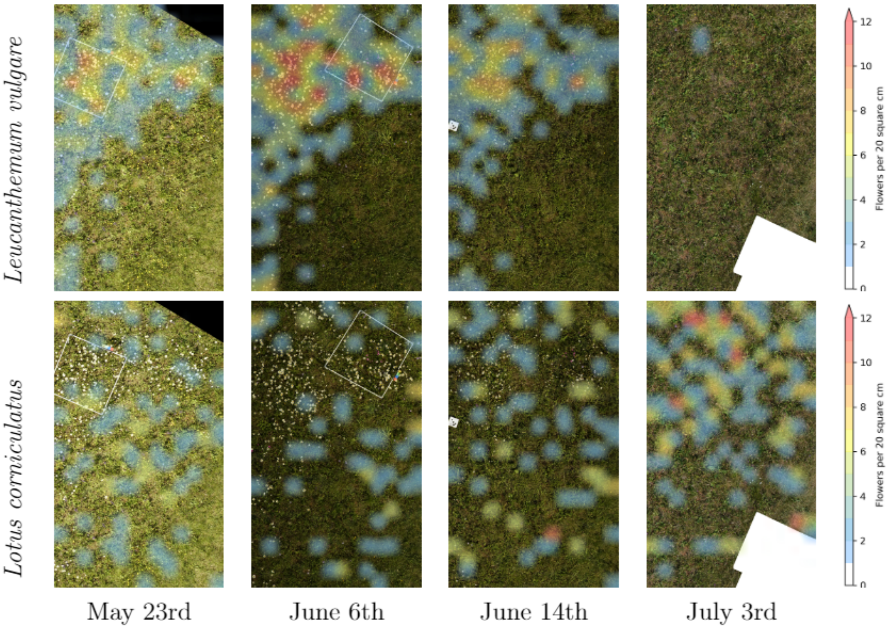


Figure 5: Timeseries of the distribution of *Leucanthemum vulgare* and *Lotus corniculatus* in our main test field.

faded. The peak population of *Lotus corniculatus* is much less pronounced.

5 Discussion

5.1 Different Approaches for Flower Abundance Mapping

We evaluated different approaches to map flowers in grasslands. We used manual counting inside survey plots as a baseline and compared it to tablet annotations on images of the survey plots and automated deep learning based mapping inside the survey plots from drone images. Then we compared the extrapolations of the manual counting to the predictions of the deep learning model on the whole test field. Section 4.1 shows that some flowers have more tablet annotations in the images than are manually counted by hand inside the survey plots. This can be explained by the fact that manually counting flowers by hand requires a high level of concentration. Mistakes happen very easily if a lot of flowers are present within a small area. Annotating on an image has the advantage that flowers are marked and therefore the risk of counting twice or forgetting to count a flower is minimized. When combining these falsely counted numbers with non optimally chosen survey plot locations, the extrapolations of the manually counted flowers have the potential to be very inaccurate.

With a reliable flower detection model, the results can be much more accurate than with the extrapolation from the manual counting. Moreover, the drone based approach has other advantages. The potential to have spatially explicit maps of flowers goes beyond what can be done with the traditional approach of extrapolating the manually counted numbers of flowers within the survey plots. Once a trained network is available, manually labelling the species to train the network is no more necessary. It is sufficient to fly the drone over the meadow and let the deep learning algorithm predict the species. The prediction time of the trained deep learning network for one square meter is approximately 7.4 seconds using a GTX 1080 GPU (Nvidia Corporation 2016). On the contrary, manually counting the flowers by hand within a survey plot can take between one and ten minutes, depending on the flower density. The predictions of the network have to be controlled by a good botany expert.

Whether it is possible to achieve reliable predictions for a certain flower on drone images depends on several factors. First, enough training data of the flower in question needs to be available. The results suggest that with a few hundred instances good performance can be achieved. Second, also the morphology of the flower has an impact. Flowers such as *Galium mollugo* are difficult for an object detection network to predict reliably. The cause seems to be that this flower can sometimes be very small and in other cases multiple instances of the same flower species cover a large area of partly overlapping inflorescences in which it is difficult to separate the single instances. In such cases it would be interesting to see how an image segmentation network such as U-Net (Ronneberger, Fischer, & Brox 2015), which predicts regions (pixels) that belong to a certain class, would perform. Third, the size of a flower should span a certain minimum amount of pixels. The good results of *Lotus corniculatus* suggest that a diameter of around 5 to 10 pixels is sufficient. These results are likely to be positively influenced by the distinct color and the strong contrast to the background of *Lotus corniculatus*. Other flowers of similar size such as *Onobrychis viciifolia* or *Trifolium pratense* perform significantly worse. These flowers are much harder to distinguish from the background. It is evident that distinguishability (mainly driven by contrast) is the fourth main factor which determines the prediction performance of a network for a particular flower.

When taking a closer look at the results, a substantial portion of mispredictions that negatively influences the model performance scores such as mAP and

F1 score is not fatal. This includes for example false positives that are in fact missing annotations in the ground truth such as the examples in table 3. False positive predictions of flowers that are on the verge of fading fall under this category as well. The mispredictions caused by the confusion between single flowers and inflorescences of *Lotus corniculatus* as described in section 4.2.1 are not severe either. If such mispredictions were ignored, the performance scores would be better.

These mispredictions exemplify the challenges that exist for the training data collection. Even when being able to directly compare the image on the tablet to the flowers on site, it is sometimes not clear how to annotate a flower. *Lotus corniculatus* is a good example. They are often arranged as inflorescences. It is not uncommon however that there are single flowers that do not belong to the same inflorescence. Since it is often not possible to distinguish the single flowers within an inflorescence, the whole inflorescence is annotated as one flower instance. Unfortunately there are border cases in which a single flower very close to another inflorescence is annotated as a separate instance in the ground truth but the prediction algorithm includes that flower in the inflorescence and predicts only one bounding box. This results in false negative predictions for the single flowers very close to the collection as the examples in image c) in table 3 show. The opposite case that multiple single flowers are predicted separately while they are annotated as an inflorescence with a single bounding box is common as well. The second main problem for *Lotus corniculatus* is that some instances are hardly visible on the images because they are very small. Sometimes they are partly hidden by other vegetation and occasionally weak motion blur is present which makes it even harder to distinguish between flower and background. This problem also manifests itself in false positive and false negative predictions. False positive predictions are mainly caused by background areas that look similar to a blurred flower and real flowers which are not present in the ground truth annotations (as in image d)). The false negative predictions are often flowers that are small and hardly distinguishable.

As demonstrated on the example of *Lotus corniculatus* in table 4, an average number of flowers per annotation can be calculated from the training data and manually counted data. This value can then be multiplied with the total amount of predictions to get the number of flowers.

5.2 Influences of the Network Configuration and Image Resolution

It is advised to scale up all images with objects that are smaller than 40 pixels in diameter by a factor of two in order to improve the performance of a network (Hu & Ramanan 2017). This is the case for the vast majority of flowers dealt with in this study. The Faster R-CNN architecture is not designed to detect very small objects such as flowers of just a few pixels in decimeter (Huang et al. 2017; Zhang et al. 2017). Therefore scaling up the images is an appropriate counter measure which helped to improve our results.

Data Augmentation options are a convenient way of artificially increasing the amount of training data. One should be careful with applying too many augmentation options. Since the flowers do not span a large number of pixels, they are predicted based on minuscule details. Changing these details too much might be counterproductive. Flips and random box jittering can be applied without hesitation. They do not alter the important details but only the orientation or the position of the bounding box. Brightness, contrast and saturation adjustment should be applied moderately. In our experiments the maximal change is a delta of 25 %.

5.3 Practical Considerations

Our main test grassland site was around 30 times 30 meters large. In order to have enough overlapping images to generate an orthomosaic of this area, a drone has to fly over the meadow for about 20 minutes. This means that it is difficult to scale this approach to larger areas. A way of overcoming this problem is to take sample pictures with less or no overlap or at random locations of a larger meadow and therefore omitting the generation of an orthomosaic. Knowing the flight height and the lens angle of the camera, one can calculate the covered area of the image. Running the prediction algorithm on these sample images and extrapolating the numbers of predictions to the size of the whole meadow can still achieve very good results. The advantage over the manual counting flower abundance determination approach is that a much larger sample size can be collected. The effort to collect the vegetation data is smaller and more precise. This enables to spend more time for controlling, extrapolating and analysing the data, which finally earns a better result. What remains to be evaluated is whether the prediction algorithm generates similar results close to the edges of an image compared to the center. The viewing angle changes across an image which changes also the appearance of the imaged objects (Aasen & Bolten 2018; Roth, Aasen, Walter, & Liebisch 2018; Aasen 2016). Consequently there could be a degradation in prediction performance. The orthomosaics are created only from the center regions of the single images.

Various metrics are used to describe a model's performance. Precision, recall, F1 score and mAP all describe a certain aspect of a model's performance. It depends on the application case, which metric is most important. Precision and recall can easily be controlled with the minimum confidence parameter. The higher the minimum confidence parameter of the prediction script is set, the higher the precision gets. Lowering the minimum confidence score increases the recall. For the abundance determination use case as in this study a balanced precision/recall ratio is advantageous, because false negatives and false positives are likely to cancel each other out and therefore a good estimate of the abundance can be given. The F1 score is mainly determined by precision and recall. The higher these two values are, the higher is the F1 score. A balanced ratio of precision and recall rewards the score even more. Consequently, the F1 score is a good indicator of a model's performance.

We have used independent train, test and validation sets for our evaluations. In the future, the results should be validated in more ways, e.g. by using cross-validation or by testing the models on more unseen test sites as well as including data with different environmental conditions.

The method developed in this study opens a wide range of use cases beyond the substitution of manual flower counting. Weed control could be realized in a precision agriculture setting. Detecting invasive neophyte plants in difficult-to-access areas could replace manual checks. The multitemporal abundance maps have the potential to map flowering dynamics quantitatively and spatially assess co-occurrence of different flowers and assess the influence of climate conditions of different years on the abundance. By detecting certain indicator species, conclusions may be drawn about the soil properties. The presence of *Leucanthemum vulgare* for example is an indicator for nutrient-poor meadows. In the context of quality assessment of meadows in connection with direct payments by the state, drone usage is imaginable. Apart from flowering plant detection, the method can be applied to other areas such as monitoring of wildlife aggregations as described by Lyons et al. (2019).

For some use cases it might be beneficial to have real time detections. The method developed in this study is not designed for that. By using the default configuration of the Faster R-CNN architecture without upscaling the images, the prediction algorithm can be sped up by a factor of four at least. The drawback is that the accuracy is lower. Nevertheless, for some use cases this might be acceptable. Using a more light weight object detection network design such as the SSD architecture (Liu et al. 2016) can deliver further speed ups. However, the accuracy is expected to be lower than with Faster R-CNN.

More training data would have been beneficial to better train the underrepresented flowers and catch flowers during their entire phenology. Unfortunately, this was not possible due to the failure of the initially used drone. However, with the now designed framework new training data can be created and pooled with the current training data to expand the training dataset and allow better predictions in the future. The suite of tools developed in this study is easy to install and can be applied to any sort of object detection problem on aerial images. The time consuming task of training data collection by annotating aerial images can be carried out on the FieldAnnotator application for Android or with the widely used LabelMe application for desktop operating systems. The script that copies annotations onto overlapping images can be a powerful way of increasing the amount of training data without major efforts.

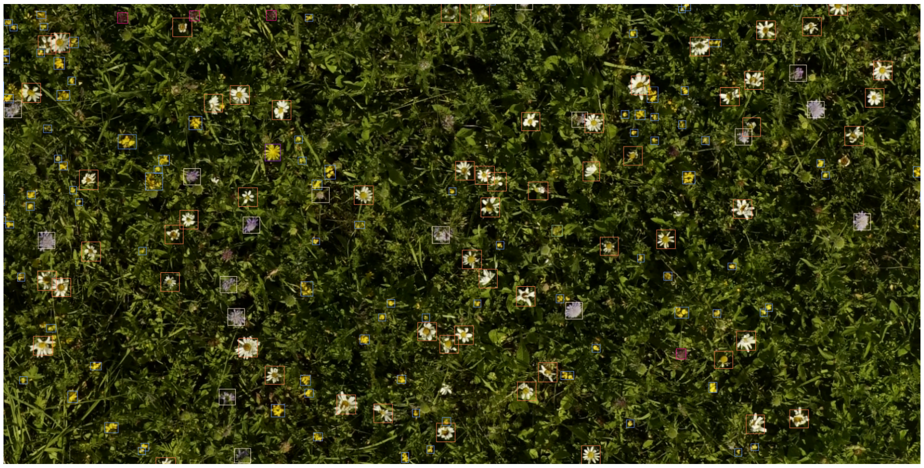


Figure 6: Typical prediction example.

6 Acknowledgements

We thank Bettina Keller and Pascal Kipf who helped assessing the data in the field. Thanks also to Alexander Indermaur, who assisted the drone flights. Special thanks also to Norbert Kirchgessner who organized the computer setup, Lukas Roth who assembled the drone of ETH Zürich and Achim Walter who hosted Johannes Gallmann in his group during the work.

7 Author's contributions

Helge Aasen, Johannes Gallmann, Matthias Albrecht and Beatrice Schüpbach conceived the ideas and designed the methodology; Johannes Gallmann, together with Katja Jacot from Agroscope, collected the data; Jonas Winizki was responsible for the drone flights and Agisoft image processing; Johannes Gallmann analyzed the data, programmed the software and led the writing of the manuscript. All authors contributed critically to the drafts and gave final approval for publication.

8 Data Availability

All code as well as documentation of the code and tutorial videos how to apply the code to new use cases are available on Github (`https://github.com/tschutli/Phenotator-Toolbox`). All our data including images and trained networks can be found at `https://datadryad.org/stash/share/O5OlneE_hOQvhFOBR73XEU1HDOZChlwWYdxKbAOyHlI`.

References

Aasen, H. (2016). Influence of the viewing geometry on hyperspectral data retrieved from uav snapshot cameras. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 3(7). doi:10.5194/isprs-annals-III-7-257-2016.

Aasen, H. & Bolten, A. (2018). Multi-temporal high-resolution imaging spectroscopy with hyperspectral 2d imagers–from theory to application. *Remote sensing of environment*, 205, 374–389. doi:10.1016/j.rse.2017.10.043.

Aasen, H., Honkavaara, E., Lucieer, A., & Zarco-Tejada, P.J. (2018). Quantitative remote sensing at ultra-high resolution with uav spectroscopy: A review of sensor technology, measurement procedures, and data correction workflows. *Remote Sensing*, 10(7). doi:10.3390/rs10071091.

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z. et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Abdel-Rahman, E.M., Makori, D.M., Landmann, T., Piiroinen, R., Gasim, S. et al. (2015). The utility of aisa eagle hyperspectral data and random forest classifier for flower mapping. *Remote Sensing*, 7(10), 13298–13318. doi:10.3390/rs71013298.

Agisoft, L. (2019). Agisoft metashape user manual. *Professional Edition, Version 1.5*, 1, 71.

Anderson, K. & Gaston, K.J. (2013). Lightweight unmanned aerial vehicles will revolutionize spatial ecology. *Frontiers in Ecology and the Environment*, 11(3), 138–146. doi:10.1890/120150.

Andrew, M.E. & Shephard, J.M. (2017). Semi-automated detection of eagle nests: an application of very high-resolution image data and advanced image analyses to wildlife surveys. *Remote Sensing in Ecology and Conservation*, 3(2), 66–80. doi:10.1002/rse2.38.

Campbell, T. & Fearns, P. (2018). Simple remote sensing detection of corymbia calophylla flowers using common 3 –band imaging sensors. *Remote Sensing Applications: Society and Environment*, 11, 51 – 63. doi:10.1016/j.rsase.2018.04.009.

Carl Zeiss AG (2017). Zeiss Batis 1.8/85, Technische Daten/Technical Specifications. `https://www.zeiss.com/content/dam/camera-lenses/files/service/download-center/datasheets/batis-lenses/datasheet-zeiss-batis-1885.pdf`. Accessed: 2020-01-24.

Carlet, J. & Abayowa, B. (2017). Fast vehicle detection in aerial imagery. *CoRR*, abs/1709.08666.

Chen, B., Jin, Y., & Brown, P. (2019). An enhanced bloom index for quantifying floral phenology using multi-scale remote sensing observations. *ISPRS Journal of Photogrammetry and Remote Sensing*, 156, 108 – 120. doi:10.1016/j.isprsjprs.2019.08.006.

Christiansen, F., Sironi, M., Moore, M.J., Di Martino, M., Ricciardi, M. et al. (2019). Estimating body mass of free-living whales using aerial photogrammetry and 3d volumetrics. *Methods in Ecology and Evolution*, 10(12), 2034–2044. doi:10.1111/2041-210X.13298.

Copting GmbH (2017). TransformerUAV. `https://www.copting.de/index.php/produktuebersicht/uav-copter-drohnen/transformer-uav`. Accessed: 2020-01-24.

de Sá, N.C., Castro, P., Carvalho, S., Marchante, E., López-Núñez, F.A. et al. (2018). Mapping the flowering of an invasive plant using unmanned aerial vehicles: Is there potential for biocontrol monitoring? *Frontiers in Plant Science*, 9, 293. doi:10.3389/fpls.2018.00293.

Eikelboom, J.A.J., Wind, J., van de Ven, E., Kenana, L.M., Schroder, B. et al. (2019). Improving the precision and accuracy of animal population estimates with aerial image object detection. *Methods in Ecology and Evolution*, 10(11), 1875–1887. doi:10.1111/2041-210X.13277.

Everingham, M. & Winn, J. (2011). The pascal visual object classes challenge 2012 (voc2012) development kit. *Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep.*

Gallai, N., Salles, J.M., Settele, J., & Vaissière, B.E. (2009). Economic valuation of the vulnerability of world agriculture confronted with pollinator decline. *Ecological Economics*, 68(3), 810 – 821. doi:10.1016/j.ecolecon.2008.06.014.

Getzin, S., Wiegand, K., & Schöning, I. (2012). Assessing biodiversity in forests using very high-resolution images and unmanned aerial vehicles. *Methods in Ecology and Evolution*, 3(2), 397–404. doi:10.1111/j.2041-210X.2011.00158.x.

Hallmann, C., Sorg, M., Jongejans, E., Siepel, H., Hofland, N. et al. (2017). More than 75 percent decline over 27 years in total flying insect biomass in protected areas. *PLoS ONE*, 12, 1–21. doi:10.1371/journal.pone.0185809.

Harwin, S. & Lucieer, A. (2012). Assessing the accuracy of georeferenced point clouds produced via multi-view stereopsis from unmanned aerial vehicle (uav) imagery. *Remote Sensing*, 4(6), 1573–1599. doi:10.3390/rs4061573.

Hill, D.J., Tarasoff, C., Whitworth, G.E., Baron, J., Bradshaw, J.L. et al. (2017). Utility of unmanned aerial vehicles for mapping invasive plant species: a case study on yellow flag iris (iris pseudacorus l.). *International Journal of Remote Sensing*, 38(8-10), 2083–2105. doi:10.1080/01431161.2016.1264030.

Hollings, T., Burgman, M., van Andel, M., Gilbert, M., Robinson, T. et al. (2018). How do you find the green sheep? a critical review of the use of remotely sensed imagery to detect and count animals. *Methods in Ecology and Evolution*, 9(4), 881–892. doi:10.1111/2041-210X.12973.

Horton, R., Cano, E., Bulanon, D., & Fallahi, E. (2017). Peach flower monitoring using aerial multispectral imaging. *Journal of Imaging*, 3(1). doi:10.3390/jimaging3010002.

Hu, P. & Ramanan, D. (2017). Finding tiny faces. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1522–1530. doi:10.1109/CVPR.2017.166.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A. et al. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7310–7311. doi:10.1109/CVPR.2017.351.

Kattenborn, T., Eichel, J., & Fassnacht, F.E. (2019). Convolutional neural networks enable efficient, accurate and fine-grained segmentation of plant species and communities from high-resolution uav imagery. *Scientific Reports*, 9(1), 2045–2322. doi:10.1038/s41598-019-53797-9.

Landmann, T., Piiroinen, R., Makori, D.M., Abdel-Rahman, E.M., Makau, S. et al. (2015). Application of hyperspectral remote sensing for flower mapping in african savannas. *Remote Sensing of Environment*, 166, 50 – 60. doi: https://doi.org/10.1016/j.rse.2015.06.006.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. et al. (2016). Ssd:

Single shot multibox detector. *European conference on computer vision*, 21–37. Springer. doi:10.1007/978-3-319-46448-0_2.

Lyons, M.B., Brandis, K.J., Murray, N.J., Wilshire, J.H., McCann, J.A. et al. (2019). Monitoring large and complex wildlife aggregations with drones. *Methods in Ecology and Evolution*, 10(7), 1024–1035. doi:10.1111/2041-210X.13194.

Martin, F.M., Müllerová, J., Borgniet, L., Dommanget, F., Breton, V. et al. (2018). Using single- and multi-date uav and satellite imagery to accurately monitor invasive knotweed species. *Remote Sensing*, 10(10). doi:10.3390/rs10101662.

Müllerová, J., Brůna, J., Bartaloš, T., Dvořák, P., Vítková, M. et al. (2017). Timing is important: Unmanned aircraft vs. satellite imagery in plant invasion monitoring. *Frontiers in Plant Science*, 8, 887. doi:10.3389/fpls.2017.00887.

Nvidia Corporation (2016). NVIDIA GeForce GTX 1080 User Guide. `https://www.nvidia.com/content/geforce-gtx/GTX_1080_User_Guide.pdf`. Accessed: 2020-02-03.

Osco, L.P., dos Santos de Arruda, M., Junior, J.M., da Silva, N.B., Ramos, A.P.M. et al. (2020). A convolutional neural network approach for counting and geolocating citrus-trees in uav multispectral imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 160, 97 – 106. doi:10.1016/j.isprsjprs.2019.12.010.

Ozge Unel, F., Ozkalayci, B.O., & Cigla, C. (2019). The power of tiling for small object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.*

Pajares, G. (2015). Overview and current status of remote sensing applications based on unmanned aerial vehicles (uavs). *Photogrammetric Engineering & Remote Sensing*, 81(4), 281–330. doi:10.14358/PERS.81.4.281.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 91–99. doi:10.1109/TPAMI.2016.2577031.

Rey, N., Volpi, M., Joost, S., & Tuia, D. (2017). Detecting animals in african savanna with uavs and the crowds. *Remote Sensing of Environment*, 200, 341 – 351. doi:10.1016/j.rse.2017.08.026.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. N. Navab, J. Hornegger, W.M. Wells, & A.F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 234–241. Springer International Publishing, Cham. doi:10.1038/s41592-018-0261-2.

Roth, L., Aasen, H., Walter, A., & Liebisch, F. (2018). Extracting leaf area index using viewing geometry effects—a new perspective on high-resolution unmanned aerial system photography. *ISPRS journal of photogrammetry and remote sensing*, 141, 161–175. doi:10.1016/j.isprsjprs.2018.04.012.

Roth, L., Hund, A., & Aasen, H. (2018). Phenofly planning tool: flight planning for high-resolution optical remote sensing with unmanned areal systems. *Plant methods*, 14(1), 116. doi:10.1186/s13007-018-0376-6.

Sanchez-Azofeifa, A., Antonio Guzmán, J., Campos, C.A., Castro, S., Garcia-Millan, V. et al. (2017). Twenty-first century remote sensing technologies are revolutionizing the study of tropical forests. *Biotropica*, 49(5), 604–619. doi:10.1111/btp.12454.

Sony Corporation (2015). Sony ILCE-7RM2 User Manual. `https://www.sony.com/electronics/support/res/manuals/W001/W0014549M.pdf`. Accessed: 2020-01-24.

SZ DJI Technology Co., Ltd. (2018). DJI Matrice 600 Pro User Manual. `https://dl.djicdn.com/downloads/m600%20pro/20180417/Matrice_600_Pro_User_Manual_v1.0_EN.pdf`. Accessed: 2020-01-24.

Ullman, S. (1979). The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153), 405–426. doi:10.1098/rspb.1979.0006.

Wada, K. (2016). LabelMe: Image Polygonal Annotation with Python. `https://github.com/wkentaro/labelme`. Accessed: 2020-01-24.

Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X. et al. (2017). S^3fd: Single shot scale-invariant face detector. *2017 IEEE International Conference on Computer Vision (ICCV)*, 192–201. doi:10.1109/ICCV.2017.30.

	Original Image	Upscaled Image						Various Image Scales	
Train Overlap			Yes	Yes		Yes	Yes	Yes	Yes
Data Augmentation				Yes	Yes	Yes	Yes	Yes	Yes
Anchor Size 128						Yes			
Stride 16							Yes		
Multiple Scales									Yes
Precision	81.7 %	82.1 %	82.3 %	84.7 %	85.2 %	85.2 %	86.1 %	85.7 %	73.4 %
Recall	79.5 %	83.9 %	81.8 %	**84.3** %	83.8 %	83.6 %	81.7 %	76.6 %	70.9 %
F1 Score	0.806	0.830	0.820	**0.845**	0.845	0.844	0.838	0.809	0.721
mAP	0.575	0.668	0.634	**0.686**	0.680	0.671	0.619	0.588	0.485

Table 6: Performance of various parameter configurations.

A Supplementary Materials

A.1 Supplementary Results

A.1.1 Faster R-CNN Parameter Tuning

Table 6 shows the effects of certain parameter choices for the Faster R-CNN architecture. For each column in table 6, a network is trained and evaluated on the same data. All results are weighted average values across all flower species meaning that all TP, FP and FN values are summed up across all flower species and then the precision, recall and F1 score are calculated using these summed up values. All predictions having a confidence score below 0.2 are ignored for the evaluation. Having a limited amount of data available, the decision is made to use 70 % of the images of each flight for training, 20 % for testing and 10 % for validating. This splitting strategy ensures that the majority of the data is used for training but still large enough portions are available for validation and testing.

As the first column of table 6 shows, using the original image size mainly hurts the recall metric. The low recall value indicates that many flowers are not detected by the trained network. Upscaling the images mitigates this problem as can be seen from the other columns of table 6. In all configurations marked with *Upscaled Image*, the image data is tiled into 450 times 450 pixel tiles. Each such tile is then scaled up to a 900 times 900 pixel tile. The drawback is that this slows down the training and prediction process.

The effect of adding a train overlap is not clear. While the recall, F1 score and mAP slightly drop compared to the configuration with only upscaled images (column 3 vs. column 2 in table 6), in combination with data augmentation options, the performance of these three metrics is better with a train overlap (column 5 vs column 4). The opposite happens with the precision metric.

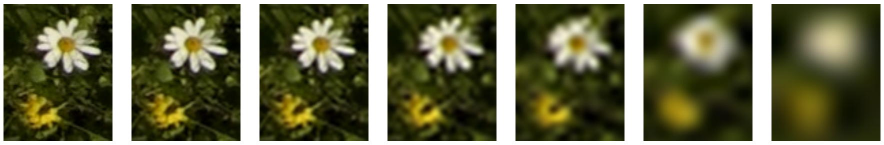
Using data augmentation techniques and a train overlap within the training pipeline results in the best performance in terms of recall, F1 score and mAP. The precision metric is only insignificantly lower than in other configurations. In all other experiments the configuration with data augmentation and train overlap is used.

The fourth last column contains the results with the base anchor size set to 128 pixels instead of default value of 256. Recall, F1 score and mAP are slightly worse than with a base anchor size of 256 pixels. The third last column shows the performance with the `first_stage_features_stride`, `height_stride` and `width_stride` parameters set to 16. The `first_stage_features_stride` defines the output stride of the extracted region proposal network feature map. A bigger `first_stage_features_stride` value has the consequence that the region proposal network of the Faster R-CNN architecture outputs a feature map with a lower resolution. The `height_stride` and the `width_stride` variables control the distance in pixels of two consecutive anchors. An anchor is a location within the image from which various sizes of possible bounding boxes to be evaluated are spanned. Having a large distance between two such anchors might cause the network to miss flowers that are placed in between two such anchors. These changes are suggested in general by Ren, He, Girshick, & Sun (2015) and specifically for small objects by Zhang et al. (2017). The results show that the performance is worse than with these values set to the minimum of 8. The recall value is notably lower than in the configuration with the stride values set to 8. This is an indication that some flowers are missed by the prediction network due to the lower coverage of anchors and the lower resolution of the feature map.

Training networks with images of multiple scales did not result in promising prediction performance. Neither did applying a Lab or HSV color space transformation to the images improve the detection results (experiment results not included in table 6).

A.1.2 Predictions on Simulated Resolutions

The higher the drone can fly the more area can be covered with a single drone flight. Table 7 demonstrates the effect of decreasing ground resolution on an example excerpt of an aerial image containing a *Leucanthemum vulgare* flower and a *Lotus corniculatus* inflorescence. Figure 7 and 8 illustrate the effect of decreasing ground resolution on the F1 score and the mAP respectively. Both figures show that down to a ground resolution of 5 mm per pixel, there is just a marginal decrease in prediction performance. Further decreasing the ground resolution to 10 mm and 20 mm per pixel however has noticeable negative effects on the model's performance. As expected, the performance of small flowers such as *Lotus corniculatus* decreases disproportionately because at a certain ground resolution they simply get indistinguishable. The average size of a *Lotus corniculatus* flower is around 16 mm. The performance of larger



1 mm/pixel 1.4 mm/pixel 2 mm/pixel 3.3 mm/pixel 5 mm/pixel 10 mm/pixel 20 mm/pixel

Table 7: Resolution degradation on an except of an aerial image.

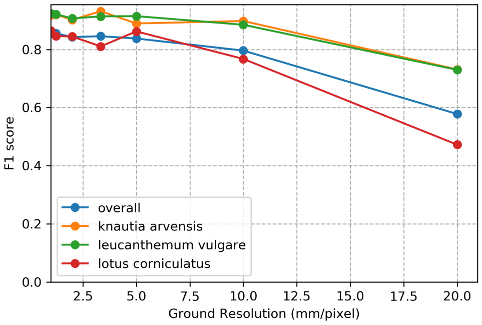


Figure 7: Evolution of F1 score over various simulated ground resolutions.

flowers such as *Leucanthemum vulgare* (40 mm) and *Knautia arvensis* (34 mm) degrades notably slower. The graphs for the precision and recall metrics are omitted since the trends are equivalent to the trends of the F1 score and the mAP metric.

Each training, test and validation image is first scaled down to the desired ground resolution and then scaled up again. After upscaling, all datasets have the same ground sampling distance (pixel size) as the original images again. This ensures that the flower's sizes (in image pixels) are large enough to be detectable by the faster R-CNN network architecture and prevents performance losses caused by this problem as described by Hu & Ramanan (2017). For each ground resolution a network is trained and evaluated with the processed training images.

A.2 FieldAnnotator Android Application

For the annotations, an Android tablet application called FieldAnnotator has been developed. It can be downloaded from the Google Playstore. The advantage of being able to make the annotations on a tablet is that they can be made directly in the field. This is necessary because some flowers can be very hard

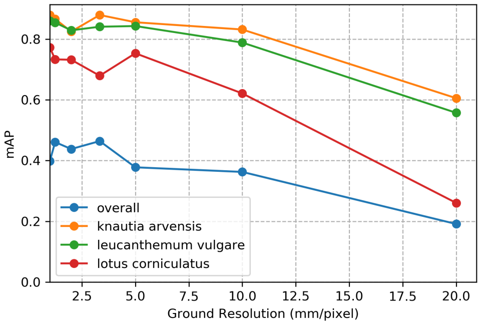


Figure 8: Evolution of mAP over various simulated ground resolutions.

to distinguish in the image alone. If one can compare the image to the actual flowers on site, the quality of the training data can be improved and it is made sure that the number of false annotations is minimized.

A screenshot of the main window of the annotation application can be seen in figure 9. The output folder created by the image preprocessing tool mentioned in the previous subsection can be copied onto the Android tablet and imported into the annotation application (1). The orthophoto is then displayed to the user. If the geo information is included in the metadata file, the user's GPS location is indicated on top of the image (2). This helps the user navigate through the field. The displayed image can be zoomed up to a level where the individual pixels are visible. If the user clicks on any location in the image, the annotation settings on the right appear. The user can select the type of flower from the list (3). Next to each flower in the list, the number of already recorded occurrences are indicated in brackets. If necessary, the position of an annotation can be fine tuned by using the four buttons on the bottom (4). The user can dismiss the annotation in processing by clicking on 'cancel/delete' (5) or save it by clicking on 'save' (6). Optionally, instead of a point annotation, a polygon can be drawn around a region. To do so, the switch at the bottom (7)

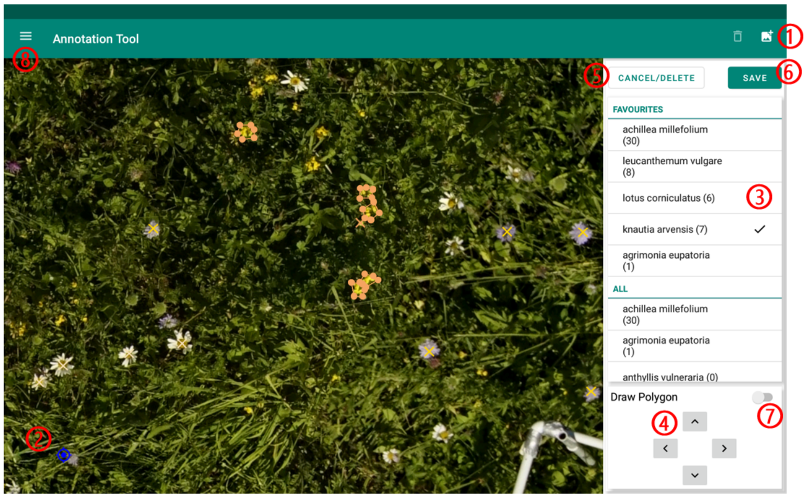


Figure 9: Screenshot of the main window of the FieldAnnotator application for Android.

has to be activated. Already saved annotations can also be edited or deleted again by simply clicking on them inside the image. By clicking on the menu button on the upper left, the settings screen can be opened. There the user can edit the list of flowers either manually or by importing a predefined list from a csv file. Also an export of the flower list to a csv file is possible. Furthermore, some zoom settings such as the maximal zoom level or the zoom level at which the annotations should be displayed to the user can be set. The application continuously saves the annotations to a json file in the project folder. The application is programmed in Kotlin.