

Detection of pests in agriculture using machine learning

Emma Olsson

Master of Science Thesis in Electrical Engineering
Detection of pests in agriculture using machine learning

Emma Olsson

LiTH-ISY-EX--22/5537--SE

Supervisor: **Magnus Malmström**
ISY, Linköpings universitet

Examiner: **Fredrik Gustafsson**
ISY, Linköpings universitet

*Division of Automatic Control
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden*

Copyright © 2022 Emma Olsson

Abstract

Pest inventory of a field is a way of knowing when the thresholds for pest control is reached. It is of increasing interest to use machine learning to automate this process, however, many challenges arise with detection of small insects both in traps and on plants.

This thesis investigates the prospects of developing an automatic warning system for notifying a user of when certain pests are detected in a trap. For this, sliding window with histogram of oriented gradients based support vector machine were implemented. Trap detection with neural network models and a check size function were tested for narrowing the detections down to pests of a certain size. The results indicates that with further refinement and more training images this approach might hold potential for fungus gnat and rape beetles.

Further, this thesis also investigates detection performance of Mask R-CNN and YOLOv5 on different insects in fields for the purpose of automating the data gathering process. The models showed promise for detection of rape beetles. YOLOv5 also showed promise as a multi-class detector of different insects, where sizes ranged from small rape beetles to larger bumblebees.

Acknowledgments

I would like to express my gratitude to my examiner Fredrik Gustafsson and supervisor Magnus Malmström for their continuous guidance and gear supply. Furthermore, thanks to Agtech 2030 and ISY for providing images and collaboration. Thanks to the people at Jordbruksverket that offered support, advice, annotated traps and trap images. Thanks to the private advisors that gave me valuable advice and lent me gear as well as helped me with maintaining traps and cameras. Thanks to the farmers and Uddetorps natural farming school for letting me place traps and photograph insects in your fields. Thanks to my family and friends that gave me endless support and help.

Linköping, November 2022
Emma Olsson

Contents

Notation	ix
1 Introduction	1
1.1 Background	1
1.2 Aim	2
1.3 Limitations	2
2 Related work	3
2.1 Pests management	3
2.2 Object detection and classification	3
2.2.1 Pest detection	4
2.2.2 Detection of small objects	4
2.2.3 Pest classification	5
3 Theory	7
3.1 Insects	7
3.1.1 Pest information	8
3.1.2 Capturing Pests	8
3.2 Classical machine learning methods	10
3.2.1 Configurations for machine learning	10
3.2.2 Support vector machine	10
3.2.3 Histogram of oriented gradients	12
3.2.4 Sliding window	12
3.3 Neural network based machine learning	13
3.3.1 Neural network configurations	13
3.3.2 Convolutional neural network	14
3.3.3 Feature pyramid network	15
3.3.4 Residual neural network	15
3.3.5 Mask R-CNN	16
3.3.6 You Only Look Once	17
3.4 Performance measure	17
4 Gathering data	19

4.1	Method of collecting data	19
4.2	Results	20
4.2.1	Data collection	21
4.2.2	Cameras	23
4.3	Discussions	24
4.3.1	Data collection	24
4.3.2	Camera evaluation	24
5	Pre-processing data	25
5.1	Method of pre-processing data	25
5.2	Results	26
5.3	Discussions	26
6	Classification and Detection	27
6.1	Method of detection and classification of insects	27
6.1.1	Detection and classification of insects in traps	27
6.1.2	Trap detection	29
6.1.3	Detection of insects in field images	29
6.2	Results	30
6.2.1	Trap images classification and detection	30
6.2.2	Field images classification and detection	35
6.3	Discussions	37
6.3.1	Trap images	37
6.3.2	Field images	38
6.3.3	Field detection future use	39
6.3.4	Work in a wider context	40
7	Conclusion	41
7.1	Conclusion	41
7.2	Future work	42
A	Additional material	45
A.1	Extended pest chart	45
A.2	Additional images dataset	45
A.3	Additional Results	45
	Bibliography	51

Notation

ABBREVIATIONS

Abbreviation	Meaning
Adam	Adaptive Moment Estimation
ANN	Artificial Neural Networks
BN	Batch Normalization
CNN	Convolutional Neural Network
COCO	Common Objects in Context
DetNet	Deterministic Networking
EFPN	Extended Feature Pyramid Network
FPN	Feature Pyramid Network
FTP	File Transfer Protocol
FTT	Feature Texture Transfer
GPU	Graphics processing unit
HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
KNN	k-nearest neighbor
LIDAR	Light Detection and Ranging
Mask R-CNN	Region-based Convolutional Neural Network
NB	Naive Bayes
NMS	Non-maximum Suppression
NN	Neural Network
RBF	Radial Basis Function
ReLU	Rectified Linear Units
ResNet	Residual Neural Network
ROI	Region of Interest
SGD	Stochastic gradient descent
SVM	Support Vector Machine
YOLO	You Only Look Once

1

Introduction

With warmer temperatures and milder autumns the occurrence of some pests has increased. At the same time, beneficial insects have seen a decline for some species due to climate change and the use of pesticides. When the beneficial predators start to disappear there are less natural ways of keeping the pest populations down. Understanding the full impact pests have on crops is an ongoing, time-consuming research. The areas to investigate can be large while the pests are small. Automation of insect detection and classification as well as a rough estimate to how severe the infestation is going to be, can be useful for agricultural practices both in the organic and conventional fields.

1.1 Background

Use of pesticides in agriculture has been a frequently discussed topic for many years. Both for its beneficial usage to stop infestations of pest from lowering the harvest and also for its negative side effects on the biodiversity [7]. Especially the declining wild bee populations have driven the agricultural sector to review its use of many pesticides [7]. Organic farming is on the rise in response to the negative effects of pesticides and the conventional agriculture sector is striving to minimize the impact of pesticides on the surrounding area. Precision farming, banning of some pesticides, compulsory courses before use and strict practices are some examples of this. Some pests have also developed a resistance to some pesticides, which makes it important to not apply pesticides too early or too often [32]. When pesticides need to be used, it is best to apply when the pests are the most vulnerable and before they can damage the crops. Also, some pests appear only at corners or in spots across the fields at first, which makes it unnecessary to apply pesticides to the whole field. It is thus beneficial to detect the infested areas in which to apply pesticides with precision. A study made by Johanna Orsholm

[32] showed that 85 % of the participants (113 oilseed farmers) would like to use alternative methods for keeping pests under control instead of pesticides. One such method is to favor beneficial insects that prey on pests. This is usually done by avoiding to use pesticides when the crops bloom, avoid plowing after harvesting some crops and to favor wild flowers near the fields. In order to apply counter measures at the proper time farmers can enlist the help of pest advisors to monitor the appearance of pests in the fields, or monitor it by themselves. Depending on the pest this can involve visits to the fields and traps at varying intensity, sometimes only once or twice in the beginning of cultivation and at other times several visits a week. When the fields are many and the visits need to be during many weeks, the task can be very time-consuming.

1.2 Aim

This research aims to explore solutions to alleviate the tasks of manually detecting, classifying and counting pests in traps and in fields. The long-term aim is to utilize this research for developing an automated method to study the correlation of occurrence of pests with damage to plants and crops. This lies beyond this project. This project will evaluate the different methods for automatic detection and classification of different species of insects. Here, it is of interest to investigate how to avoid wrongly classifying small black dots as insects, and what precision of the classification that is obtained. If a method is proven to work with enough precision, a probable sighting of a pest can be sent by the method to be verified by an user and aid decisions for if pest control is needed.

1.3 Limitations

To gather images of all the pests that can cause damage to farmers is a task too great for this project to undertake. This project will limit itself to a smaller number of pests which are fairly common to encounter in Swedish commercial agriculture. Since not all pests will be visible during the time period of this project, focus will be on pests that appear during spring and summer. The gathering of images of pests will take place mainly in the central part of Sweden, in regions such as Västergötland and Östergötland. The number of traps that will be put out is a limiting resource. The pests usually undergo several stages of evolution in their life-cycle, from egg, to larva, to full grown insect and so on. Therefore, this project will be limited to the fully grown pests. The amount of data is limited and the insects in images are not labeled by an entomologist. Results should be viewed with discretion.

2

Related work

Detection along with classification of insects and calculation of threshold values of pests have traditionally been done manually. As the interest of automating this process has increased, so has the interest to use ML for detection and classification of insects.

2.1 Pests management

Today there are services in Sweden that keep track of when the pests approach certain areas. Notifications and information are sent out to subscribers through "växtskyddsbrev" and posted on their website [32, 33]. This information is great for farmers, but it does not alleviate the need for the farmers to do an inventory of pests in their own fields. For example, the inventory of rape beetles should be done 2-3 times a week from oilseed plants buds appearing until they turn into flowers. Some farmers hire an agricultural- or plant protection consultant to perform the inventory for them instead. Preventing damage from certain pests can be troublesome, both in terms of the time consumed when checking for infestations and the difficulty of knowing when to apply measures. [32]

2.2 Object detection and classification

Object classification and detection have been researched extensively in the field of computer vision. Medium to large size objects are the most commonly researched, while small size objects are researched more for medical, aerial and satellite images. Classification and detection of humans, animals, plants and objects in different settings are popular research topics. Insect image classification and detection have been researched to a degree, but not as extensively as other

topics.

2.2.1 Pest detection

Thenmozhi Kasinathan, et al. [22] proposes an approach for insect detection algorithm that segments out the insect and chooses the contour of the insect, yielding high detection accuracy. Weiguang Ding and Graham Taylor [13] instead used a combination of convolutional neural network (CNN) and the sliding window detection pipeline to detect moths in traps. After colour correction of the image, a CNN predicts the likelihood that an image patch contains pests. The remaining patches after filtering with Non-maximum Suppression (NMS), finding the patches with higher probability than their neighbours, are the proposed detection. Best accuracy was achieved with patches of 21×21 for CNN, at object level 93.1 % and for image level 97.2%. [13]

Wenyong Li, et al. [27] states some of the challenges of pest detection such as construction of large-scale datasets or multi-scale detection. Obtaining new images, relabeling and retraining a model when new classification tasks are added can take a considerable amount of time. They mention data augmentation and domain adaptations as some proposed solutions for increasing efficiency. With multi-scale pest detection, geometric details might disappear with deeper layers and information is contained at different layers. Therefore it is proposed to use anchor-free networks and predictions with multi-layer features. Occlusion of pests can cause a CNN to be less robust. The robustness could be increased by letting the neural network (NN) learn parts of the insect in a specific layer. Since there is a limited amount of hardware available out in the field, lightweight models are preferable. Hence models that require GPU and high-power hardware resources should be avoided. Feature pyramids are promising for solving the multi-scale insect detection problem, either by fusing feature maps from different depths of the network with cross-layer connections or by constructing a spatial pyramid based on the receptive field's parallel branches. [27]

2.2.2 Detection of small objects

The detection of middle sized and larger objects achieved better results with deeper CNNs, however it has been shown that detection of small objects did not perform as well due to loss of information in the deeper layers [14]. Some object detection methods, such as Faster CNN (R-CNN), often utilize *max-pooling* for obtaining feature maps that have rich semantic information while decreasing the computational cost. However, the feature maps spatial resolution decrease when using max-pooling which makes it hard to discover exact locations of small objects. Even if high spatial resolution is retained in the feature map the model will have too little semantic information for identifying small object in such cases. [14]

In light of this, Yini Gao [14] proposes an approach for detection of golf balls, with Deterministic networking (DetNet) and Feature Pyramid Network (FPN), that allows for gaining better semantic information while keeping the feature

map's spatial resolution. The DetNet backbone takes inspiration from Residual neural network (ResNet) structure and utilizes mostly dilated convolutional layers. They use the concept of dilated rate, which enables convolutional operations to skip neurons. Yini Gao also mentions that You Only Look Once (YOLO) version 3 with Darknet-53 reaches impressive results when applied for object detection on small objects. [14]

Chunfang Deng, et al. [12] recognizes the problem with detection of small objects that are represented with very few pixels and proposes the Extended FPN (EFPN). Reason being that feature coupling of various scales does not perform well enough on small objects even if the scale-level detection has been greatly improved for FPN. EFPN utilizes a pyramid level with very high resolution specifically for detection of very small objects. EFPN is memory as well as computational efficient while yielding impressive results for small objects, in some cases better than FPN. EFPN can use a Feature Texture Transfer (FTT), designed to discard noise that otherwise would be carried along to next level and manages to *super-resolve* features at the same time as finding credible regional details. A special loss function called *foreground-background-balanced loss function* is introduced. Designed to regulate differences in foreground and background, since normal global loss will have trouble learning such small fractions of an image well. This approach utilizes global reconstruction loss and positive patch loss on both foreground and background to gain a higher feature quality. [12]

2.2.3 Pest classification

Thenmozhi Kasinathan, et al. [22] compares the classification approaches Artificial neural networks (ANN), k-nearest neighbor (KNN), support vector machine (SVM), Naive Bayes (NB) and CNN on datasets with close-up images of insects. The SVM classifier, utilizing nine shape features of the radial basis function (RBF) kernel, reached an accuracy of 79.9% for 5 classes, but the accuracy decreased rapidly to 71.8% when adding 4 more classes. The highest classification accuracy was made with the CNN model that scored between 90-91.5% accuracy in the range 9 to 24 classes. Noteworthy is that different approaches of machine learning algorithms performed best on different types of insects, where SVM gave the most accuracy for winged insects such as fruit flies, and KNN performed better for butterfly species. [22]

Wenyong Li, et al. [27] presents an overview of how well different methods of classification perform for insects in field images. A fine-tuned ResNet-50 with medium large dataset of approximately 2000-3000 images of paddy crop pests achieved an accuracy of 95.01%, while tiny YOLOv3 reached an accuracy of 98% for a dataset of 400 images of sticky traps. Faster R-CNN achieved 99 % accuracy on 500 images and classification of cotton pests with 500 images using few-shot training reached an accuracy of 95.4%. The quality of the dataset, regardless of the amount of images, and the number of classes impacted on the performance. Further, it is mentioned that describing image features using natural language could be beneficial for CNN models. [27]

Dan Jeric Arcega Rustia, et al [38] proposes a semi-supervised learning ap-

proach for monitoring insects on a farm using a pseudo-labelling algorithm with accuracy as high as 96.3%. This is promising for automated collection of training data for the model.

Johanna Orsholm [33] investigates another promising approach for classification of different insects using auditory features. Utilizing audio recordings for machine learning is an approach that was tested for classification of wingbeat frequencies to differentiate between pests and pollinators by earlier studies, reaching an accuracy of 99 %, and similar results were obtained when using feature extractions in another study. To enable classification based on wingbeat frequency, the frequency needs to differ from other insects in the vicinity and too great variations might cause difficulties. During experiments, the sound of wingbeats from rape beetles and weevil were very similar, possibly making it hard to differentiate. Collecting microphone based data could be complicated due to background noise when this approach is applied in the field. A company called FaunaPhotonics takes another approach and use a *light detection and ranging* (LiDAR) to capture the wingbeat frequency, and the LiDAR can also provide information such as relations between body and wings even at long distances. [33]

3


Theory

Automatic detection and classification of pests can be realized with machine learning or with approaches from the sub-field of machine learning called deep learning. In this chapter, information about pollinators, pests, methods for capturing pests and machine learning are introduced.

3.1 Insects

Some of the insects that appear in crops can be considered as either pests or beneficial insects, where the latter pollinates flowers and in some cases regulate pest populations. An insect considered to be a pest can also be a pollinator, and vice versa [7]. Rape beetle is a pest that causes damage to young rape plants, but while crawling over the flowers it also pollinates the plants during the blooming season. In the Table 3.1 the only pollinator considered for detection is listed.

Table 3.1: Pollinator chart [7, 10]. °C is the temperature needed for a pollinator to emerge from various winter dwellings.

Image	Pollinator	Length mm	Plants preferred	°C
	Bumblebee	7-24	Oilseed plants, flowers	15

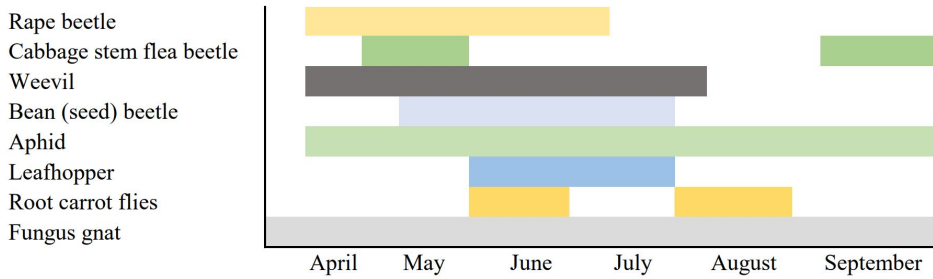


Figure 3.1: First appearance and re-appearance, as well as approximated presence during which can be seen in crops or traps [5].

3.1.1 Pest information









The prognosis for pests predicted arrival to the fields each year will vary due to weather conditions. Figure 3.1 contains the likely first sightings of some of the considered pests if weather conditions and temperatures agrees. Here, it is mainly rape beetles and weevils that can be spotted early in rape fields. Table 3.2 contains information about the pests, their preferred crop and a grading for how much of a threat the insect is. This measurement is based on how severe the damage to the listed plants and crops could be if the conditions are favorable [5, 6]. Further, it can be seen that for some pests the size and shape is similar. A rape beetle can be hard to differentiate from flea beetles and weevils caught in traps.

3.1.2 Capturing Pests

In general most winged pests can be captured with a yellow cup, even a sun-bleached one, or a yellow or orange sticky trap. To protect pollinators a net can be placed on the cup. Some specific cases relevant to this project are listed below. See Figure 4.1 for images of the traps mentioned.

- **Rape Beetle:** Capture of this pest is usually done with a yellow cup filled with water or a soap mixture. After one season in the field the sun-bleached cup might no longer attract rape beetles. Place trap on the east or northeast side of the field and on the same height as the plant or the flowers. For capture using sticky traps, place with a tilt of 45° and facing out from the field. [32, 33]
- **Carrot root flies/Leafhopper:** Can be captured using sticky traps, either placed upright or tilted, close by the plant.
- **Frit fly:** Captured using a blue cup, placed directly in the ground.

Table 3.2: Pest information chart [5, 6, 21, 31]. TL = Threat level. The grading range from a low threat, denoted 1, to higher threat, denoted 5. Some pests are more common as well as a larger threat to certain crops depending on the region. The regions are denoted (M) for the middle part of Sweden and (S) for the south part of Sweden following crop names when applicable. The * denotes those crops that could suffer the most damage. An extended table can be found in the Appendix, see Table A.1.

Pest information chart				
Image	Pest	Length mm	TL	Plants infested
	Rape beetle	2-3	3	Oilseed plants (spring and autumn rape*)
	Flea beetle	1,5-5	2	Oilseed plants, cruciferous vegetables
	Weevil	2,5-3	1	Oilseed plants
	Bean (seed) beetle	3-5	5	Legumes (field beans*)
	Carrot root flies	5-7	3	Carrots (S), parsnip (M)
	Leafhopper	2-5	5	Potatoes
	Aphid	2-3	3	Spring wheat, spring grain, oats
	Fungus gnat	3-4	2	Seedlings, young plants

3.2 Classical machine learning methods

Relevant machine learning techniques for classification and object detection for this project are presented in this section.

3.2.1 Configurations for machine learning

Common features used with machine learning techniques are introduced below.

Loss functions: For classification, a loss function computes a penalty based on the difference between the true annotation and calculated prediction. This penalty is then utilized for optimizing the model by alterations of the parameters.

For classifiers with large margin, like SVM, *Hinge-Loss* is a good consideration that finds the maximum margin between the different classes [16]. The correct class is denoted y , the ground truth, and is either 1 or -1. The predicted value is \hat{y} , calculated with $f(x)$, see (3.2) for SVM. Multi-class loss can be calculated as "one versus rest". [8]

$$L(y, \hat{y}) = \max(0, 1 - y\hat{y}) \quad (3.1)$$

Optimizers: A training algorithm is required to find good values of the parameters for the network. With the output given from the loss function, a gradient descent optimization algorithm updates parameters of the model to reduce the loss. A basic method is the *stochastic gradient descent* (SGD), where the weights are either updated after running through the whole dataset or after each batch from the dataset [16, 18].

Non-maximum Suppression: NMS is commonly used for pre-processing the output to remove redundant bounding boxes, keeping the box with local maximum and boxes with less overlap than a set threshold, see Figure 3.2, [14]. If an object lies such that its bounding box would significantly overlap another object, NMS would suppress it by setting the detection score to 0. However, using Softer-NMS, the detection could be kept [39]. Softer-NMS instead sets the score as a function of the threshold, keeping the detection score if the overlap is less and assigning a low score when the overlap is high.

Anchor boxes: Anchors are predefined bounding boxes tiled across an image that allow for finding the probability of different object positions at different scales across an image. The anchors then tie the original image locations to the found features [4, 18]. Anchor box strategy, using k -means clustering algorithm, can make a model more precise and stable [14].

3.2.2 Support vector machine

Support vector machine (SVM) solves classification problems through finding a hyperplane that separates two classes [8]. The ideal hyperplane is found by creating two parallel support vectors passing through the points closest to the hyperplane from both classes and calculate the distance from the points to the hyperplane, which is the margin that then will be maximized, see Figure 3.3. The reason for using support vectors with maximum margins is to find a generalized

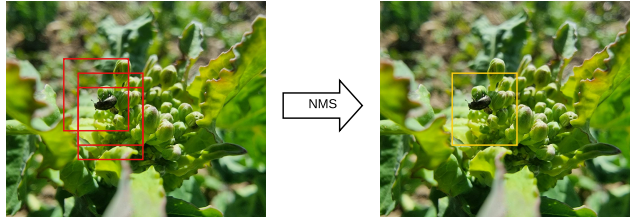


Figure 3.2: Before and after applying NMS. The probability scores of the bounding boxes are compared and removed until the one with the highest score remains.

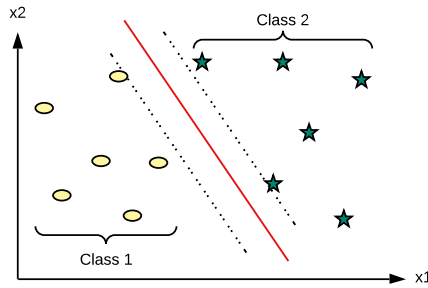


Figure 3.3: A visual representation of how SVM divides 2 classes using a hyperplane, the red line, and two support vectors, the dotted lines. The equation 3.2 is used for finding these lines.

model. For data that is not linearly separable, an extra dimension can be added to find the hyperplane and then project it onto the original dimension. [8]

For a linear kernel, with an input vector X the equation for SVM is the following, where b is the distance from the origin and the hyperplane's normal vector is represented with w . [8]

$$f(x) = w^T X + b \quad (3.2)$$

Radial basis function (RBF) kernel is a multiple kernel function that is often used for non-linear data. This is practical for the cost function optimization to work efficiently in higher dimensions while doing the transformation implicitly. [8]

$$k(x_1, x_2) = \exp\left(\frac{-\|x_1 - x_2\|^2}{2\sigma^2}\right) \quad (3.3)$$

Here the squared Euclidean distance between the feature vectors x_1 and x_2 is $\|x_1 - x_2\|^2$ and the hyperparameter σ controls the shape of the decision boundary, used to limit the impact of a single example during training. A low number gives a low impact reaching far and a high number gives a high impact close to the data point. [8]

3.2.3 Histogram of oriented gradients

Histogram of oriented gradients (HOG) uses the structure or shape of an object to describe the features and is used for object detection [25]. It holds similarities to other models such as Canny edge detector, since it involves gradient computation. The difference is that HOG uses both angle and magnitude for generating histograms, with a dense grid of cells and normalization of overlapping local contrast which leads to better accuracy. When training a HOG feature dependent model, the training set should contain positive images with the object of interest and negative images with anything but the object. [25]

HOG approaches the problem cell for cell to find what is commonly called the *HOG descriptor* for an image [30]. After converting the image into grey-scale, normalizing the images and finding each pixels gradient, the image is split up in cells, also called spatial regions. Per cell, a local histogram is computed before combining cells into bigger blocks. Setting the number of pixels in the cells to a higher number gives a weaker HOG than setting to a low number, same with the number of cells in a block and the number of bins consisting of a set number of degrees. The number of bins is connected to the gradient vector computation of change in direction, the angle, also known as the orientations of the cells pixels. [30]

3.2.4 Sliding window

A sliding window approach for images utilizes a rectangle of fixed dimensions that is moved a set distance each time across the image left to right, as well as from top to bottom, see Figure 3.4, [13, 26]. An image classifier can then be applied to each of these small window regions and by combining sliding window with image pyramids objects of different scale can be detected. The size of the window should fit the whole object that is to be detected. It is best if the window fits tightly around it, since if the window is too large or too small the object might not be detected properly. The speed of a sliding window approach will vary with the specification of the image pyramid, if it is used, as well as with the size of the rectangle and the image. When sliding window is used with a detection or classification model, such as a SVM trained with HOG feature extracted images, the window size should be the same dimension as the training images. [13]

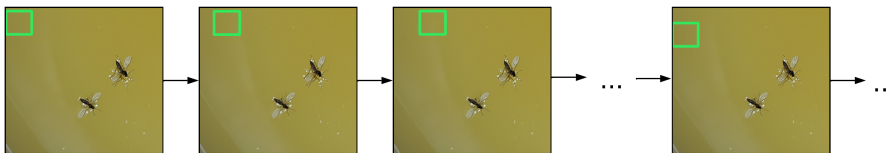


Figure 3.4: Example of sliding window on image.

3.3 Neural network based machine learning

Neural network (NN) is a model used in machine learning. It has a loose equivalent to a biological brain, the neurons of a NN are computational nodes connected to one another and can forward signals as transmitted over synapses. Improvement and extensions to NNs have been done during the years since the first NN was presented, and the development is an ongoing process.

3.3.1 Neural network configurations

In this chapter follows some concepts for configuring a NN. To get the desired behaviour from a NN the optimizers, activation and loss functions need to be configured.

Transfer learning: Transfer learning removes the need for a large dataset, without losing the generalization or infringe the performance. Reusing the information from a previously trained model allows the training process to converge faster towards minimum without needing more annotated images. Two common approaches are to either fine tune the whole network or to fine tune only a selection of layers with the weights of the earlier layers frozen [14]. Pre-trained weights can be obtained from training on big datasets such as Common Objects in Context (COCO) [28].

Activation function: Activation functions are non-linear functions that define the output of a neuron through a function of the input x . Rectified linear units (ReLU), $f(x) = \max(0, x)$, have been proven to improve the optimization speed, outperforming both sigmoid and tanh activation functions. [16]

The activation function *softmax* is often used to find the probability estimation of an object belonging to the different classes, the predicted probability Q . The class label is denoted y , known as true class or ground label for the input x . Before normalization, the \hat{y} is referred as a latent variable. When using one-hot encoding the y is a vector with size n , each object will have a known class label set to 1 and 0 for the rest. [16]

$$\text{Softmax} = Q(\hat{y}) = \frac{\exp(\hat{y}_i)}{\sum_{j=1}^n \exp(\hat{y}_j)} \quad (3.4)$$

Loss functions: Used with softmax for multi-class classification problems, *cross-entropy* calculates the loss through distance between two probability distributions, for one prediction, in order to optimize the model [8].

$$H(y, Q) = -\sum_{i=1}^N y_i \log(Q(\hat{y}_i)) \quad (3.5)$$

Binary cross-entropy, a binary version of (3.5) also referred to as the log loss function L_{cls} , gives an indication of how close the prediction is to the ground truth for one class as follows [15, 42]. Here the N instead represents only two classes.

$$L_{cls} = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)] \quad (3.6)$$

A variation of (3.6) is the *average binary cross-entropy loss*, L_{mask} , and is the loss function for segmentation mask that utilizes per-pixel *sigmoid* [18, 41, 42]. The i is extended to the cell (i, j) , $\hat{y} = \hat{y}^k$ with k -th mask and N is replaced with the dimension of the mask $m \times m$, giving a size of Km^2 on the whole output with K classes. Both L_{mask} and L_{box} , see (3.8), are not classification losses.

$$L_{mask} = -\frac{1}{m^2} \sum_{1 \leq i, j \leq m} [y_{ij} \log \hat{y}_{ij}^k + (1 - y_{ij}) \log (1 - \hat{y}_{ij}^k)] \quad (3.7)$$

Localization loss calculates the penalty based on the networks ability to correctly locate an object, comparing the predicted bounding boxes with the ground truth. *Smooth L1 localization loss*, as shown in (3.8), is less affected by outliers and good for avoiding overfitting [15]. Also referred to as L_{box} . The input x is the offset values for width, height and positions of (x, y) of the box center.

$$L_{box} = \begin{cases} 0.5x^2 & \text{if } |x| < 1, \\ |x| - 0.5 & \text{otherwise.} \end{cases} \quad (3.8)$$

Classification loss and localization loss are often combined for object detection problems. Mask R-CNN is such a model, calculating the loss with $L = L_{cls} + L_{mask} + L_{box}$, due to optimizing results for instance segmentation. There is no competition between classes during mask generation with L_{mask} , as opposed to the combination of *multinomial cross-entropy loss* and per-pixel *softmax*. The mask is generated for each class and each Region of Interest (ROI). [15, 18]

Optimizers: The SGD explained in Section 3.2.1 can be implemented using *early stopping* in order to stop the optimization process as the performance no longer improves [16, 18]. Adaptive Moment Estimation (Adam) is an improved algorithm built on SGD [24]. In Adam the adaptive learning rate for each parameter is adjusted in response to changes calculated on the first-order gradients [24]. The vanishing gradient problem is common for small object detection with NNs. It can be dealt with using Batch Normalization (BN). BN first normalizes and zero-centers the input before each layer. [14]

3.3.2 Convolutional neural network

As mentioned in Section 2.2.3, the CNN architecture scored high accuracy for many classes and it is a foundation for improved versions for classification and detection problems, such as Mask R-CNN [22]. The architecture generally adapted for CNN can be seen in Figure 3.5, [23]. Following the input layer of a CNN is hidden layers that alternates between convolution and subsampling layers [23].

A convolution operation is one of the basic principles of a CNN, and a convolutional layer is constructed with several convolutional kernels. A kernel consisting of a matrix acts as a filter and slides over the image in a similar manner as to a sliding window, see Figure 3.4, to generate a filtered output. The output and the image are then matched, retaining the information that is found matching as the output of the convolution. The kernel's matrix contains the corresponding weights of the NN after being trained for recognizing certain features, such as edges, pixel differences for sharpening and many more. [16]

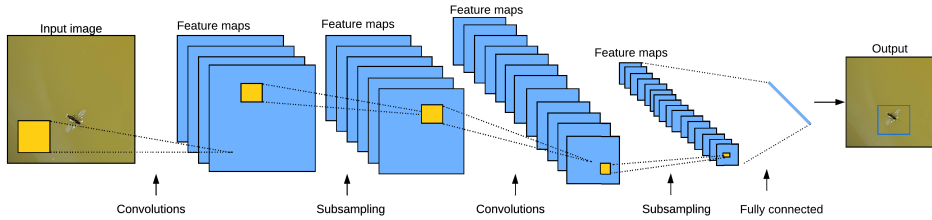


Figure 3.5: CNN's general structure.

Pooling, applied following a convolutional layer, uses sub-sampling on the feature maps to reduce the computational cost through decreasing the amount of connections in between convolutional layers. In addition, the use of pooling can reduce overfitting which creates a more generalized model. Max-, average- and stride pooling are commonly used. Padding, such as zero-padding, is used to avoid losing information about the image's corners when using filters as well as avoiding the image from decreasing in size after a convolution is applied. [16]

3.3.3 Feature pyramid network

Tsung-Yi Lin et al. [29] proposed the FPN that utilizes predictions from feature maps at different depths of the pyramid for a more generic feature extraction. The general structure of FPN has bottom-up paths connected laterally with top-down paths [29]. Each pyramid level of both paths can consist of several feature maps of the same resolution. The FPN can with the lateral connections combine both the low and the high spatial resolution feature maps through upsampling, retaining the rich semantic information as well as having both the current and the deeper stages features to base predictions on [14]. This while being faster than a feature pyramid and more accurate than single feature map or pyramidal feature hierarchy. An extension of FPN, EFPN, includes object detection [12].

3.3.4 Residual neural network

ResNet was developed in response to the vanishing gradient problem, see Section 3.3.1, that arises for very deep CNN's during backpropagation [17]. As a minimum is sought with the loss function during training a too large amount of layers will make the gradient shrink and eventually disappear, causing the optimization to stop prematurely. ResNet retains the gradient by generating multiple layers that are initially skipped in favor of reusing the previous layer's activation functions and expands the "residual" convolutional layers during re-training to investigate features that a shallow CNN architecture would have overlooked. Furthermore, the architecture of ResNet allows for more parameters. [17]

A simple ResNet backbone structure with implemented FPN would have down sampling bottom-up paths latterly connected to the up sampling top-down path [17, 29]. The up and down sampling is a factor 2, where the down sampling

uses stride and the up sampling can be done with nearest neighbor. Stride is the amount of pixels that a kernel is moved across the image. The last residual blocks (C) output of each level except the first in the bottom-up path, C2-C4, is added latterly to the corresponding output of each level (M) in the top-down path, M4-M2, after undergoing a 1×1 convolution layer. The M5 is the C5 after applying a 1×1 convolution layer. The input P5-P2 to the rest of the NN, such as Mask R-CNN, is the final feature map after applying a 3×3 convolution to the merged output of each level. [17, 29]

Residual blocks, commonly called skip connections, are important building blocks of ResNet that will not cause added computational cost and retains essential features to the final layers. The intermediate input x is added to the convolutional blocks output $F(x)$ of the previous layers. [17]

3.3.5 Mask R-CNN

Mask R-CNN is an extended version of Faster R-CNN created upon a CNN structure [39]. The general structure of Mask R-CNN can be seen in Figure 3.6 [18]. Mask R-CNN uses instance segmentation, that annotates each pixel in an image to a class while keeping localization information, which allows for counting instances of separate objects belonging to the same class [11]. Region proposals are generated after scanning the image and top-down FPN extracts semantic feature maps from the candidate object boundary boxes proposals. Anchors are used to tie the original image locations to the features found by RPN [4, 18]. The proposals are then classified and the bounding boxes are refined and pixel level masks applied. ROI align is used for extraction of relevant areas of feature maps. ResNet-50, ResNet-101 and MobilNet are all possible to be used as a backbone [39]. As mentioned in Section 2.2.3, ResNet-50 did achieve high accuracy for insect classification [27].

One implementation of Mask R-CNN is Matterports [4], which use a ResNet-101 as backbone and a learning rate set to 0.001 for re-training the top layers of the model. It uses transfer learning where weights previously are trained on a COCO dataset [28]. It uses Softer-NMS and the structure of FPN is such that it will extract feature maps from each layer, excluding the first layer [4].

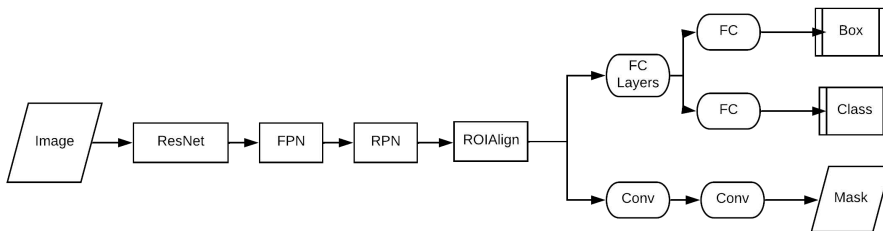


Figure 3.6: Mask R-CNN network structure [18]. Fully connected (FC).

3.3.6 You Only Look Once

As the name implies, each image is observed once to detect and classify objects [36]. The first version of YOLO is fast, lightweight and straightforward. After resizing, the input image passes through one convolutional network before outputting the detections. In contrast to sliding window and techniques using region proposals that only sees part of an image at a time, YOLO sees the whole image both during training and testing [36]. There are now improved versions of the original adaptation of YOLO such as YOLOv5, [20], where the accuracy is significantly improved. Common backbones to use with YOLO implementations are ResNet and DarkNet-53, and for certain applications, a detector head combining YOLO with FPN and similar have been tried with state-of-the-art results. YOLO can be used on images or video. [36, 38]

3.4 Performance measure

The three standard metrics accuracy, recall and precision will be used to evaluate the classification model, and since the project uses unbalanced datasets the precision and recall are especially good to evaluate. Evaluation of accuracy alone does not give a proper measurement regarding the effectiveness of the model.

$$Precision = \frac{TP}{TP + FP} \quad (3.9)$$

$$Recall = \frac{TP}{TP + FN} \quad (3.10)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.11)$$

True Positives, False Positives, True Negatives and False Negatives describes the outcome of classification predictions and are represented by TP, FP, TN and FN respectively. The predicted outcome of a binary or multi-class classification can then be visualized in a confusion matrix and for a good result the majority of predictions should end up in the diagonal. To measure how good a classifier is of an unbalanced dataset, a F1-score can be used. F1-score combines the outcome of precision (P) and recall (R) for each class, one class at the time, which gives that class a score that can be evaluated against the remaining classes.

$$F1 = 2 \frac{P * R}{P + R} \quad (3.12)$$

Evaluation of the precision of bounding boxes can be done with the Intersection over Union (IOU), that uses the ground truth bounding box and the predicted bounding boxes for estimating the intersection [14].

$$IOU = \frac{\text{Region of Overlap}}{\text{Region of Union}} \quad (3.13)$$

IOU can be used during training of a model to reward a heavy overlap of the ground truth.

4

Gathering data

Below is an outline of the methods for gathering data used in this project. The results will be presented and discussed, along with an overview of the databases.

4.1 Method of collecting data

There will be two types of images used in this project: Trap images and Field images. See Figure 4.1 for an example image of each type.

- **Trap images:** These images will be of sticky traps or cup traps. The traps are placed near crops prone to visits from pests, on the author's home farm and at other farms, agreeing to participate in the project. The images will be collected with cameras facing the traps or be photographed during regular visits. Many trap images and traps are also provided by advisors (Jordbruksverket or private), and by previous Agtech 2030 project.
- **Field images:** The field images will be collected by manually taking pictures of insects on plants when finding them. Some of the field images collected are also intended for use in a pilot project in Agtech 2030 [31].

The gathering stage of Trap and Field images can be highly impacted by weather conditions and the actual occurrence of the pests. It is required for the pests to emerge from their various winter dwellings. For different pest this happens at different times as the temperature rises during the spring. The gathering of images will begin in late March and continue until late August. For this project Hunter 3G camera [1] will be used. The Hunter 3G has a 100 degrees field of view, can save 12 MP on the SD card and is configured during this project to deliver images through file transfer protocol (FTP) and e-mail with 1920*1440 pixels. A M5stack Ov3660 [2] camera is evaluated during this project as well, supporting

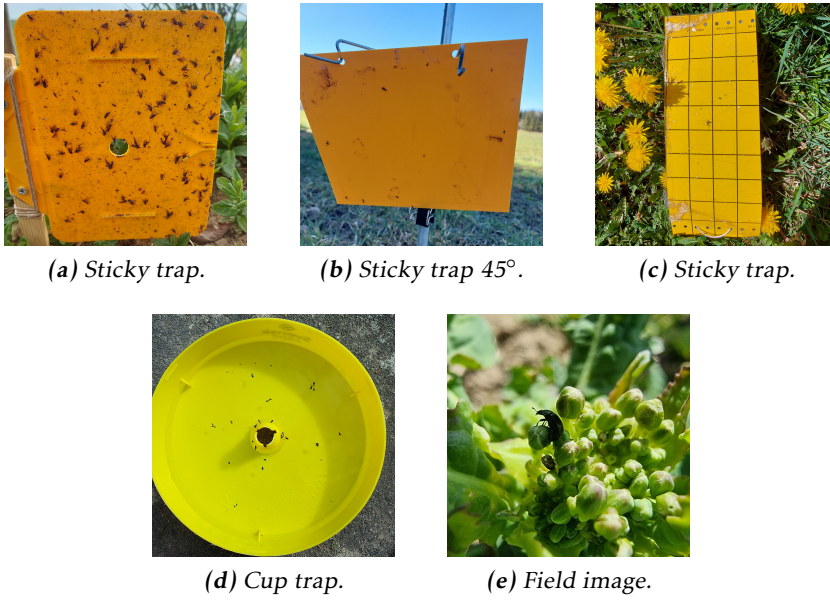


Figure 4.1: An example of each image type used in this project and where a)-d) are a representation of the different type of traps used.

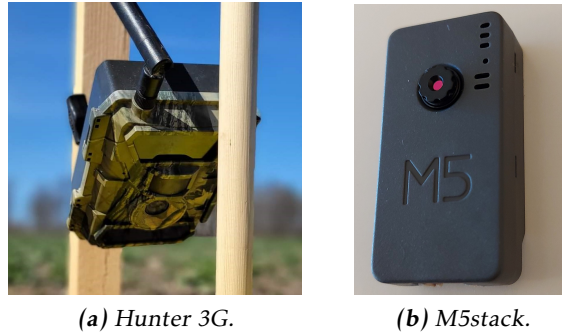


Figure 4.2: The cameras evaluated during this project.

66.5 degrees field of view and a resolution of 1600 x 1200 pixels. The cameras are shown in Figure 4.2. Additional images will be captured with other cameras and mobile cameras, as well as with a built-in Macro lens at 5 MP.

4.2 Results

Result of the data collection and the camera evaluation are presented below.

4.2.1 Data collection

An overview of how the traps were placed in fields can be seen in Figure 4.3. The timeline for collection of trap images and field images along with information about in what type of crops they were collected in are presented in Figure 4.4. One Hunter 3G camera was placed in an autumn rapeseed field during March-May and another was placed between a spring rapeseed field and an autumn rapeseed field, during April-June.

A summary of the databases used for training that were collected and finalized can be seen in Table 4.1. Examples of the crop images in these are presented in Figure 4.5 and Figure A.1. A large portion of the images collected did not get included for various of reasons. For fritfly, an early consideration, the number of possible images were less than 3 after collection. 4 databases for trap detections were also created within the project, were two include 4 classes.

Table 4.1: Training database summary for insects. *N* is the total number of images. *BB* stands for bounding box and *P* is polygon in the dataset names. *BB3* is one dataset consisting of 6 insect types. The resolutions of the individual images might differ, \geq or $>$ indicating if more images of a certain type. * mark that geometric transformation has been performed.

Trap cropped insect	N	Resolution
Rape beetle H3G	1136*	20x20 to 25x25 px
Negative images H3G	2362	20x20 to 25x25 px
Rape beetle	474	50x50 px
Rape beetle	3792*	50x50 px
Flea beetle	162	50x50 px
Weevil	285	50x50 px
Decomposed/Winged rape beetle	504*	50x50 px
Negative images, cup trap	5380	50x50 px
Fungus gnat	402*	112x112 px
Negative images, sticky trap	728	112x112 px
Field image	N*	Resolution
P1_Rape beetle	120	Macro 5 MP \geq 2.1 MP, 64 MP
P2_Rape beetle	720	Macro 5 MP, 2.1 MP \geq 64 MP
BB2_Rape beetle	707	Macro 5 MP, 2.1 MP \geq 64 MP
BB1_Rape beetle/BB3, Rape beetle	120	Macro 5 MP \geq 2.1 MP, 64 MP
BB3, Weevil	102	2.1 MP \geq 64 MP, Macro 5 MP
BB3, Pea leaf weevil	106	Macro 5 MP \geq 64 MP
BB3, Bean (seed) beetle	114	Macro 5 MP
BB3, Aphid	120	Macro 5 MP \geq 64 MP
BB3, Bumblebee	120	2.1 MP, 64 MP

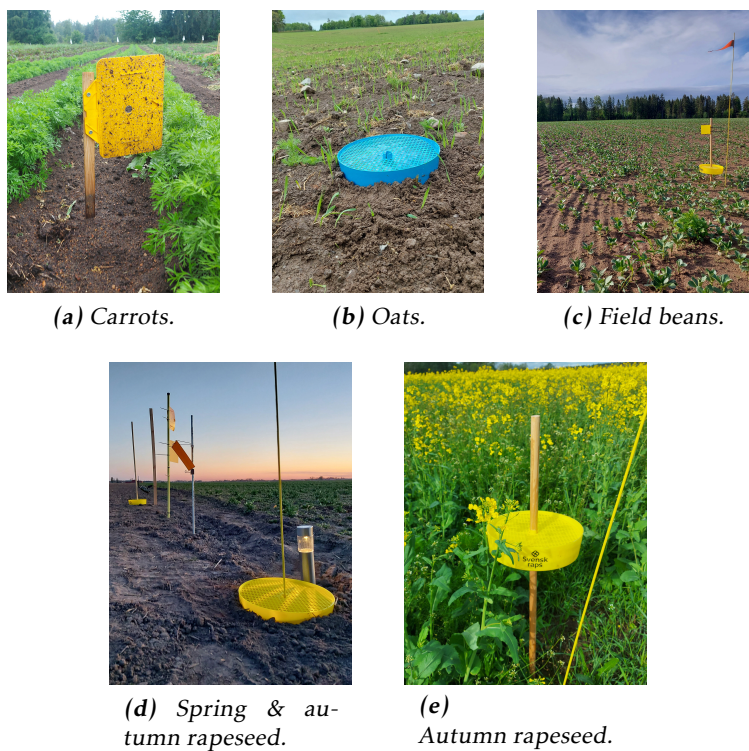


Figure 4.3: Example of trap placement in fields.

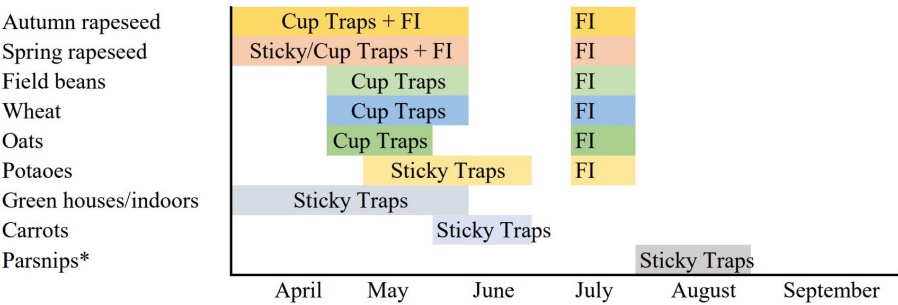


Figure 4.4: Data collection timeline in different crops in Västra Götaland, where field images FI. The cup trap images provided by Jordbruksverket placed in autumn and spring rapeseed fields were collected from April until June in Östergötland. *The parsnips sticky traps were in Östergötland, monitored with a camera and managed by an advisor and Agtech 2030.

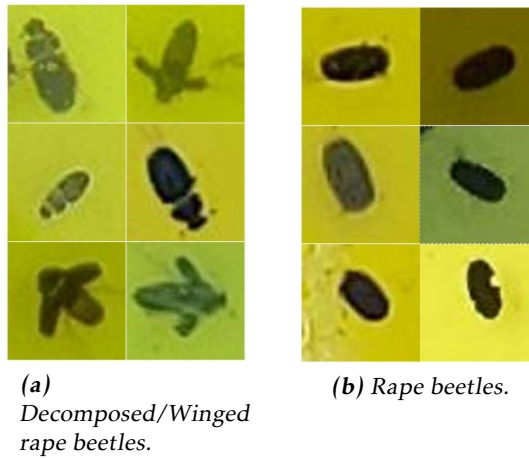


Figure 4.5: Example of positive images in the datasets for rape beetles. The differences between these possible rape beetles are large enough to effect the accuracy of a model and the results when trying to sort the insects based on size. Most rape beetles were too difficult to identify when on the side and were mostly left out from training set since sometimes very similar to flea beetles or unknown beetle types. Some of the crops in a)-b) are taken from original images provided by Jordbruksverket. See Figure A.1 for more examples from the datasets.

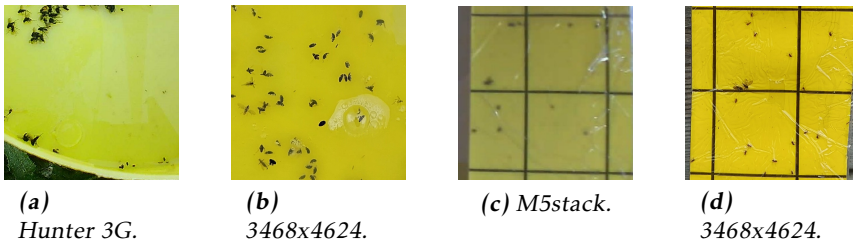


Figure 4.6: Images of Hunter 3G and M5stack alongside mobile images of same type of trap. The images are cropped, original resolution in pixels mentioned.

4.2.2 Cameras

An overview of the evaluated cameras visual image resolution from the collection of data is presented in Figure 4.6. M5stack images are not used in any datasets, see discussion in Section 4.3.2.

4.3 Discussions

In this section the results of data collection and camera evaluation are discussed.

4.3.1 Data collection

It was not a trivial task to find images of relevant pests in large quantities, good quality and correctly identified, as well as allowed for commercial use. Furthermore, the image conditions would rarely match the purpose. Pests with very few images, found or collected, were no longer considered. The collection of data went well for some pests, e.g., rape beetle. This was due to great advice and help from pest advisors, a large portion of luck during trap and field collections and thanks to Jordbruksverket that provided traps with fungus gnat and images of pests in cup traps. The few online insect images found became redundant and a database gathered within the project was pursued. More time than intended was spent on gathering data and redoing databases, since the learning curve of how to collect properly and what kind of images could be used was steep. It was not trivial to learn how to annotate the pests or how to find them. Lastly, the amount of time needed to manage traps and do pest inventory was misjudged.

4.3.2 Camera evaluation

The Hunter 3G cameras durable design, paired with long-life batteries, worked well in this project. The transferred images resolution is a bottleneck, but much information about the present state of the trap were provided with the images sent every day. The resolution could be enough for detection based on size. For example, in Figure 4.6 a) a possible weevil is in the top center, and some possible rape beetles are in the right bottom corner. However, details needed for classification of similar sized insect were not seen consistently for small insects.

The M5stack Ov3660 camera is suitable for similar tasks, with resolution enough for detection of insects based on size, but the resolution did not offer the option to differentiate well between small insects, see Figure 4.6. Smartphone cameras, the built-in Macro lens and system cameras provided more details.

5

Pre-processing data

5.1 Method of pre-processing data

To be able to utilize the gathered data for training and testing of the machine learning model the data must first be pre-processed. Since the detection will revolve around small objects, placing the bounding box consistently close around the object will help to avoid confusing the model [14]. Open software Makesense is used for manually annotating field images [3]. It is free, and can be used directly in the browser.

The datasets are then divided into training and test sets, along with an additional validation set for tuning hyperparameters of SVM model. All test sets are set aside manually before any training, and are not present in Table 4.1. Most images used for testing are captured at a date later than when the last training image used for that model was captured. The validation set is a small portion removed from the test set. The amount of test images are low due to limited amount of images where the pests are present as well as visible enough, while the number of insects per image are high in some cases. Original mobile cup images containing visible rape beetles amount to less than 90 images, and images set aside for test are approx. 25 %. In Figure 6.3, the models were evaluated on 62 cropped images for Hunter 3G models, and the other rape beetle models were tested on 772 cropped images.

To improve learning and detection a number of techniques can be applied. Firstly, data augmentation in the form of geometric transformations will be used to combat overfitting that otherwise is prone to happen due to the smaller size of these datasets. Secondly, in order to combat false positives generated by glare in images, the glare in training images could be removed.

5.2 Results

Annotations with polygons or bounding boxes of very occluded pests or pests out of focus increased false detections. Geometric transformations, flipping and rotation of the images, were performed for certain insects depending on desired dataset composition.

False detections due to glare occurred mostly in rectangle sticky trap images due to the material being prone to reflecting light, and also since the traps are commonly wrapped in plastic to handle more easily. For cups containing water the amount of false detections due to glare depended on the sunlight, use of flash or amount of bubbles. After removal of glare, false detections were caused by marks left by the removal.

5.3 Discussions

Annotation workflow with Makesene worked well. The placement and size of the bounding boxes or polygons were chosen to include invisible parts of pests to a degree and excluding pests too occluded or out of focus after initial results. For the rectangular sticky trap the placement of the bounding box had to be very large whenever the trap was not placed straight in the image, including more background, in contrast to the placement of the polygon that were placed along the edges.

Data augmentation with geometric transformations allowed for the models to be more general, have a more even distribution between classes and have a better accuracy. For insects such as weevils or bean (seed) beetles the dataset for field images would have been too small to include for detections alongside other pests for YOLOv5 without geometric transformations. Same held true for cropped trap images of fungus gnat as well as Hunter 3G rape beetles and mobile rape beetles. The geometric transformations helped improve accuracy in these cases.

SVM had a tendency of getting false detections due to both glare and marks left behind after removal of glare. Either inclusion of negative patches containing glare or muted areas were needed, and since some images have low amount of glare the removal of glare was deemed unnecessary. Photographing the traps in suitable conditions would be the best way to combat glare.

6

Classification and Detection

In this chapter the method, results and discussions of classification and detection of insects and traps are presented.

6.1 Method of detection and classification of insects

In this section the approaches tested for detection and classification of insects in trap or field images are presented as well as of the traps themselves.

6.1.1 Detection and classification of insects in traps

For testing detection and classification of insects in trap images the sliding window method will be implemented, with the option of image pyramids for improving the detection. This approach is inspired by [22], [13] and [37]. The insects of interest in this project can be quite small, represented by only a few pixels, and in one trap image there can be several breeds of insects similar in shape, colour and size. A classifier will be applied to the detection window, checking each detection to see if it passes both the size check and the set confidence level or probability.

- **SVM & HOG** Denoted SWSH for short, this approach will use the SVM model, Scikit [35], for classification. The loss function is *Hinge-Loss*, as shown in (3.1). The SVM will be trained with a linear or a RBF kernel on HOG feature extracted images with both positive and negative images, see Table 6.1 for HOG settings. In order to suppress bounding boxes NMS will be used, with a threshold of 0.1.
- **CNN** A CNN model can be used as classifier instead on the window. This option is more preferable if many classes are to be detected at once.

Table 6.1: Values for HOG feature extraction, see Section 3.2.3.

Orientations	Pixels per cell	Cells per block	Threshold
9	8, 8	2, 2	.3

- **Check size** The original image is cropped to the detection and uses Otsu method [34] to threshold the gray-scale image. Then check if detection is within threshold for density of white pixels, the area, the width and height as well as the difference between width and height. Input images size, trap type and, if checked, trap size can be used to automatically change the thresholds.

The approach can be combined with trap detection, see Section 6.1.2. For Hunter 3G images, the SWSH will always stop the computation at the bottom white bar. A flowchart of the method can be seen in Figure 6.1.

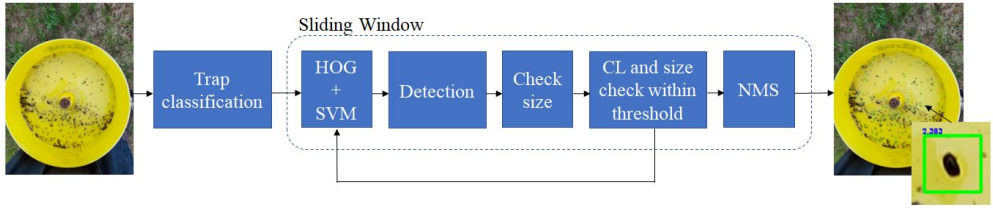


Figure 6.1: Sliding window classification and detection pipeline, here with SWSH. CL is confidence level. After the trap classification step, see Section 6.1.2, the size of the image is checked for choosing the settings for size check. When used on training images the SWSH crops its own new training images, allowing to train on the models weaknesses in regards to false detections.

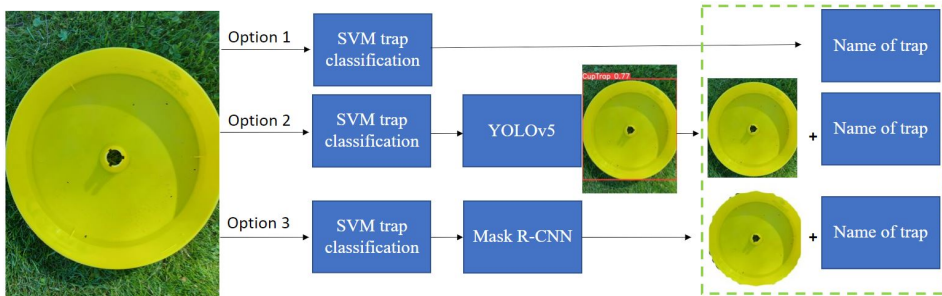


Figure 6.2: Flow of trap detection. Three paths are possible. As standard setting, the SVM trap detection will always output the trap type. Mask R-CNN and YOLOv5 can also output the size of the trap and crop to detection. The more stable option out of 2 and 3 will be kept in final implementation.

6.1.2 Trap detection

Automatic classification and detection of traps, and lack of one, will be tested. A chart of the trap detection is in the Figure 6.2. This allows for switching between methods for detection of insects in field images or insects captured in traps, switching between settings depending on what kind of trap. Automatic detection of traps will be tested or implemented to differentiate traps from background in order to extract only the trap and for cropping the image to detection when applicable. The purpose is to avoid having too much background in which it is possible to get multiple wrong detections. Also, cropping the image can decrease some of the detection time needed for SWSH method.

- **Mask R-CNN:** Tested and implemented to detect traps for removing background and cropping image to detection. Since the generated mask can be uneven in edges when training is not enough, a solution could be to dilate the mask just enough to be sure it would cover the area of interest without adding too much background. This implementation will build on Matterports [4] implementation of the Mask R-CNN, see Section 3.3.5, as well as some of the modifications as by Sergio Canu's tutorial for training and testing with custom datasets [9]. As described in Section 6.1.1, this can be used in the SWSH method and the code was modified for this.
- **SVM:** Tested for automatic classification of traps in order to use with other models. Implemented with the SWSH method, see Section 6.1.1.
- **YOLOv5:** Tested for detection of traps in order to crop image to detection. Also tested for automatic classification of traps for a future program as with SVM. The training and testing is done with the model YOLOv5 created by Ultralytics [20].

6.1.3 Detection of insects in field images

Two models will be tested for their performance on field images, with 5MP Macro images, 2.1 MP video captured images and on 64MP images. Depending on insect, there might only be one type of resolution used. Below are the two models that will be tested for detection of insects. Both models are used without extra modifications except those needed for training with custom datasets or displaying testing images.

- **Mask R-CNN:** Tested for its detection performance. Trained in same manner as for trap detection, see Section 6.1.2.
- **YOLOv5:** Suitable for images and video, real-time applications and some edge devices. Therefore tested for future prospects of using it as a tool to help automating the collection of images containing insects as well as its performance as a classifier and detector for field use. If used on a trap, meant to detect new additions to the trap. The model was trained in same manner as for trap detection, see Section 6.1.2.

Table 6.2: Performance of trap classification and detection. The number of images tested is N and A is the accuracy. A failed result for Mask R-CNN means no trap detected or that the mask did not cover the whole area of interest. If the area covered was slightly too small it is marked a Pass *. The 4 different classes are cup, field, square and rectangle sticky. See Figure A.2 for results with Mask R-CNN.

Model	N	Pass	Pass *
Mask R-CNN, 2 classes, incl. H3G Cup	22	11	8

Model	N	Pass	A
SVM, 4 classes	43	40	0.93
YOLOv5, 2 classes incl. H3G Cup	9	9	1
YOLOv5, 4 classes incl. H3G	16	15	0.94

6.2 Results

The results for models applied for classification and detection on images are presented in this section. The results for models are divided into two cases, one for trap images and one for field images. Detection results done on images sent from Hunter 3G cameras will be marked H3G. Observe, as mention in Section 1.3, that even if the true number of rape beetles in a cup trap is sometimes known for an image, the annotations in this chapter most likely have mistakes in cup images as well as in field images.

6.2.1 Trap images classification and detection

The different trap detection methods were tested in order to choose which to use during testing with sliding window, see Table 6.2. Both SVM and YOLOv5 were chosen to be used for implementation.

Two Scikit-learn [35] methods for training a SVM model were tested on a validation set, a LinearSVC and a GridSearch, which finds the best hyperparameters for SVM. Regarding the mobile crop images, the GridSearch was done with a subset of the negative images. The best hyperparameters found through GridSearch were a RBF kernel, see Section 3.2.2, with a C of 100 and gamma of 0.0001. C is used to regulate the margin against the classification accuracy and gamma is equivalent to $\frac{1}{2\sigma^2}$. This RBF kernel did however give an accuracy of 0% for H3G images, and instead the LinearSVC was chosen.

The models were then retrained with different compositions of the datasets and tested to find which combination of positive and negative images to use, see Figure 6.3. Geometric transformation were applied to all positive rape beetle images. For the amount of cropped images in each category, see Table 4.1. For the RBF kernel, the addition of more negative images such as negative weevils increases the accuracy although decreasing the recall. Adding more positive images such as decomposed rape beetles or in the case of H3G with LinearSVC additional mobile or camera rape beetles had the opposite effect.

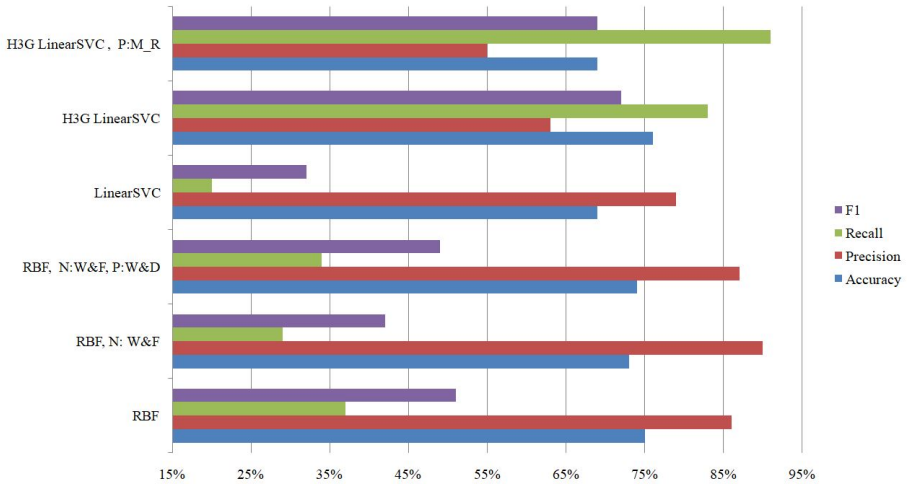


Figure 6.3: Classification metrics from rape beetle test on cropped images, 62 images for H3G and 772 images otherwise. Negative is denoted N, P is positive and M_R indicates that the original mobile and camera rape beetle images were included. W&F stands for weevil and flea beetles, W&D stands for winged and decomposed rape beetles.

Table 6.3: SWSH settings test on RBF model for rape beetles. Step size is the amount of pixels that the sliding window will travel each iteration.

Settings	Accuracy	Precision	Recall
Nothing	0.19	0.20	0.73
YOLOv5	0.21	0.22	0.73
YOLOv5 & size check settings	0.32	0.37	0.73
YOLOv5 & YOLOv5 size check	0.32	0.37	0.73

Size check settings

Area	Density	Width/Height	$ w - h $	Step Size
$110 < a < 420$	$0.07 < d < 0.25$	$17 < w/h < 31$	$\text{diff} < 22$	7

The RBF model trained on only the negative images along with the geometric transformed rape beetle images was chosen. This model was then tested with SWSH on 4 test images, set aside for seeing the impact of different settings, see Table 6.3. When YOLOv5 is combined with size check, the settings are adjusted based on the trap size found with YOLOv5.

The time for completing the SWSH for one cup trap image was approximate 5 min using LinearSVC and around 30 min using RBF kernel. For H3G images however, the SWSH took only seconds or less than 2 min. The time would also be less with a larger step size, but so would also the detections.



(a) 36 known rape beetles, however only the clearly visible rape beetles will be included as FN. The detections are 15 TP, 11 FN and 9 FP.

		True Class	
		Pos	Neg
Predicted class	Pos	82	92
	Neg	51	2
		Accuracy: 37%	
		Precision: 47%	
		Recall: 62%	

(b) Confusion matrix.

Figure 6.4: Results from the test with SWSH on the 15 testimages. TP are the green boxes, and FP are red and FN are purple. Original image a) provided by Jordbruksverket.

Rape beetle detection using SWSH and a model trained on RBF kernel with 1.0 CL and a loose YOLOv5 size check was tested on 15 mobile and camera cup trap test images, see Figure 6.4. 5 of these images had no rape beetles, and the two with no false detections were included as TN. This model was trained only on negative images and geometric transformed rape beetles. The reason for not including negative flea beetles and weevils is that this removes many true detections, although it would greatly improve the accuracy. As can be seen in Figure 6.4 and Figure 6.5 a), many false detections are due to flea beetles and also not identified beetles on their side, of these some in the test images might actually be rape beetles. The inclusion of weevil and flea beetles as negative images gives the same accuracy of 46% as without on 4 of the selected test images. The FP get less the more flea beetles and weevils that are added. Without geometric transformations the precision increased with 3% from 56% and with geometric transformations it increased to 75%, however with a much lower recall.

Rape beetle H3G detection using SWSH, with a model trained using a LinearSVC kernel, was tested on 4 cup trap test images, see Figure 6.6. Trained on negative H3G images and on geometric transformed rape beetle H3G images. The settings used were a YOLOv5 size check, step size 7 and 0.1 CL. Here the width w and height h check are approximately between $|5 < w/h < 12|$, and the difference check is $|w - h| < 7$.

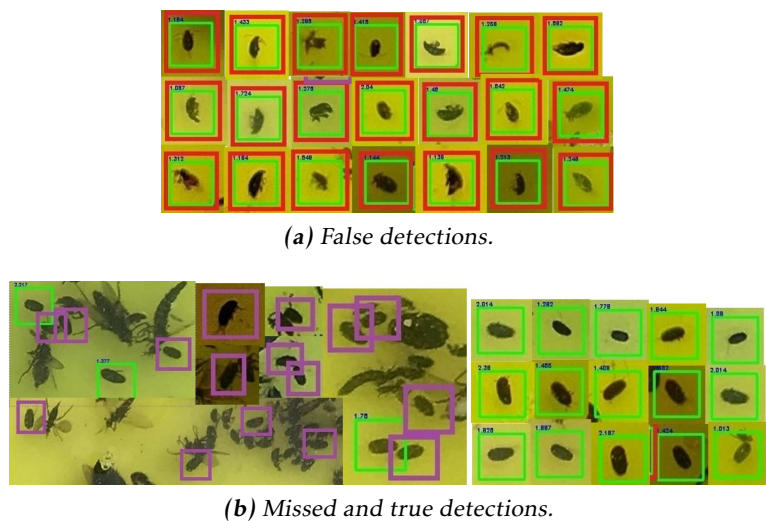


Figure 6.5: Some zoom and crops of the detections in the test images. Green boxes are TP, and the red are FP and FN are purple. Original images provided by Jordbruksverket.

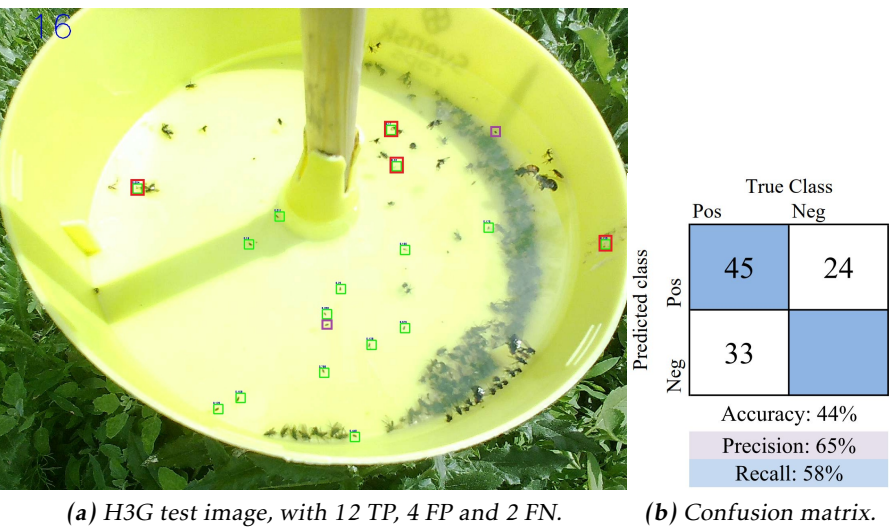
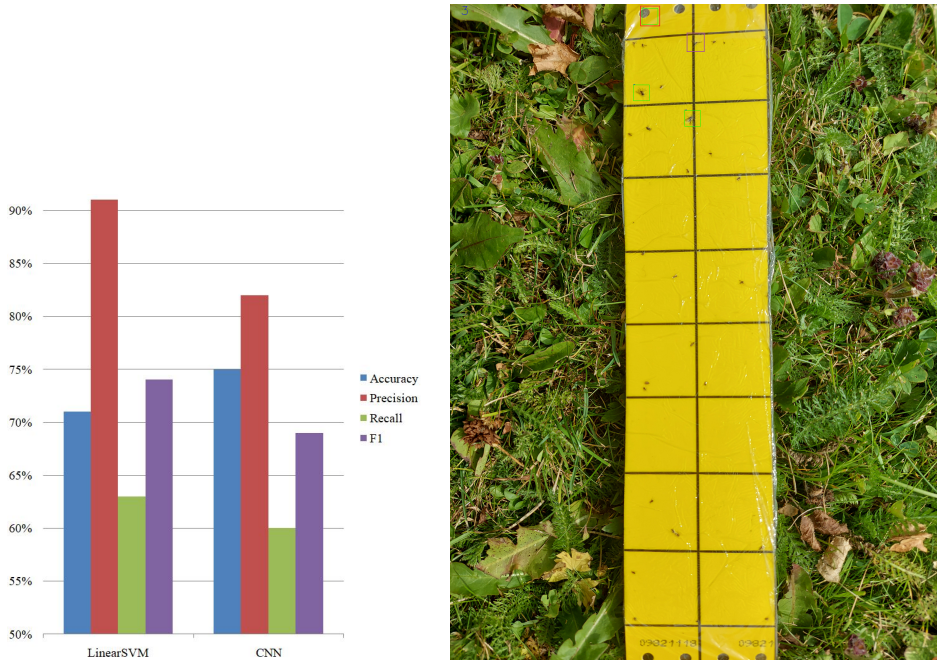


Figure 6.6: Test with 4 H3G images using SWSH method on cup traps. Green boxes marks TP, and FP red and FN purple. Some FN and FP might be missed. No TN were included for this small test.

With a step size of 3 and adding the mobile and camera rape beetles, not geometric transformed, the model increased the recall to 89%, at the expense of the precision that dropped to 40%.

Fungus gnat detection on rectangular sticky traps was also tested using SWSH method, see Figure 6.7 b) - 6.8. Geometric transformations on fungus gnat. With 1.0 CL, 112 WS, a stepsize of 30 and performed on 5 test images, the LinearSVC gave 7 TP, 10 FP, 5 FN resulting in an accuracy of 32% and precision 41%. The size check used for this had the settings $710 < \text{area} < 3100$ and $0.055 < \text{density} < 0.26$.



(a) Classification metrics on cropped test images for LinearSVC and CNN.

(b) LinearSVC. 2 TP, 1 FP and 1 FN.

Figure 6.7: The CNN in a) were trained on a subset of the LinearSVC training images. In b) one of the detection results of fungus gnat using SWSH method can be seen. TP are marked with the green boxes, and FP with red and FN with purple.

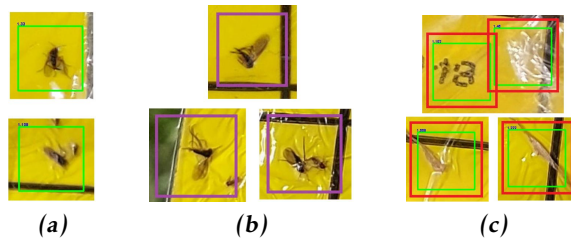


Figure 6.8: From the SWSH test, some examples of true detections in a), missed detections in b) as well as false detections in c).

6.2.2 Field images classification and detection

Detection of insects on plants or in empty yellow cups were tested with Mask R-CNN and YOLOv5. Confidence level for YOLOv5 is 0.45. Missed detections FN were included according to the annotation method, for P2_Rape beetle and BB2_Rape beetle heavier cases of occlusion were included. The results on field images can be seen in Table 6.4 and in Figure 6.9 and Figure 6.10. See translations of Swedish names in Table A.1. In Figure A.4 images from video tests with YOLOv5 are presented.

Table 6.4: Performance comparison. See Table 4.1 for database. The accuracy is denoted (A), precision (P) and recall (R). The BB3 test have six classes; rape beetles, pea leaf weevil, weevil, bean (seed) beetle, aphids and bumblebees.

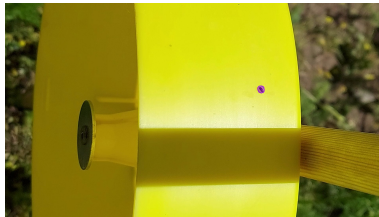
Model	N	Insect	TP	TN	FP	FN	A	P	R
Mask R-CNN	15	P1_Rape beetle	30	1	2	7	0.78	0.94	0.81
Mask R-CNN	17	P2_Rape beetle	29	1	15	14	0.51	0.66	0.67
YOLOv5	15	BB1_Rape beetle	31	1	2	6	0.8	0.94	0.84
YOLOv5	17	BB2_Rape beetle	38	1	6	5	0.78	0.86	0.88
YOLOv5	21	BB3, 6 classes	26	0	5	8	0.67	0.84	0.76



(a) P2_Rape beetle.



(b) BB2_Rape beetle.



(c) P2_Rape beetle.



(d) BB2_Rape beetle.

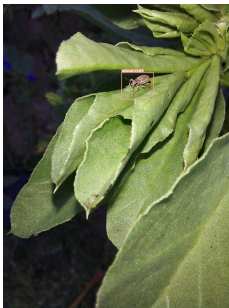
Figure 6.9: Result of detection of rape beetles on field images. In a)-b) both models got 2 TP and the same FP (positioned at red dot in a) marked by Mask R-CNN). c)-d) rotated. For same image in c) BB2_Rape beetle failed, and for image in d) P2_Rape beetle got 1 TP and 1 FP instead.



(a) Rape beetles, P1_Rape beetle, [31].



(b) Rape beetles, BB1_Rape beetle, [31].



(c) Pea leaf weevil (AVirvel), BB3, [31].



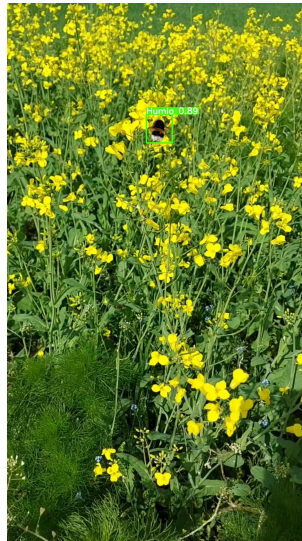
(d) Been (seed) beetle, BB3, [31].



(e) Aphids, BB3, [31].



(f) Failed detection of weevil marked with purple box, BB3.



(g) Bumblebee, BB3.

Figure 6.10: Result of detection on field images. In b) YOLOv5 outperformed a) Mask R-CNN in detection of rape beetles.

6.3 Discussions

In this section the outcome of the tests are discussed.

6.3.1 Trap images

The performance of insect detection and classification were impacted by a number of factors. The most influential factors are listed below.

1. Size of the insect in pixels.
2. Resolution of the insect.
3. Similarities between insects.
4. Occlusions and rotations of insect.
5. Amount of time since caught in trap.
6. Amount of insects caught.
7. Difference in the distance between camera and trap.
8. The amount of background texture inside and outside of trap.
9. Water in cup clear or not.
10. Colour of traps not consistent due to light conditions.
11. Regarding Hunter 3G images, weather conditions.

True for most false detections are that they become similar to a rape beetle in grayscale and when cropped to the detection window. As seen in the results in Figure 6.5, many false detections were caused by flea beetles and weevils due to their similarities to rape beetles. Letters, such as text on bottles or Hunter 3G information bar, see Figure A.3 a), in general often caused wrong detections along with cup edges, bubbles, small yellow flies, wings or tips of larger flies bodies.

To help with the precision and removal of false detections a number of approaches were tried or considered. The ones found most promising are discussed below.

- **Size check of insect with threshold, with or without YOLOv5.**

Density, width, height and area check of insect were most consistent and accurate after using Otsu method to threshold the image. For cup images this proved sometimes as effective in removing FP as using background removal, as opposed to without, see Figure A.3. Fungus gnat often got stuck on the black lines of sticky traps, see Figure 6.7 b), so removing FP of lines with check size was complicated. The check for difference between width and height gave the most impact when trying to detect only weevils. A too strict check size could sort away insects too close to each other. Due to rotations, the width and height check was not used to its full potential, left for future improvement.

- **WS - insect to image ratio.**

Too big or too small WS can cause loss of or false detections. A smaller WS can increase detection of close insects, but would also miss the larger grown insects. The WS size found to be effective for mobile or camera images with rape beetle were 50 pixels, 23 pixels for Hunter 3G.

- **Dataset composition - include or exclude variations of or similar insects.**

Adding more flea beetles and weevils as negative patches lessened the amount of wrong detections of these. However, due to the very similar shape, it would always cause the models to detect less rape beetles. Similar, inclusion of older or winged rape beetles increased true detection at the cost of more wrong detections of flies and beetles. The beetles on the side should be included in future datasets as negative or positive images after identification. Previous datasets tested before final test mostly did not contain negative crops of beetles partly cut off or tip of flies bodies. Adding these removed some of such false detections, however some rape beetle detections were lost.

Knowing the real dimensions of the pests and the corresponding dimensions in pixels allows for a size check threshold for an insect. This is helpful in achieving better accuracy and removing false detections. When the insect size in pixels is unknown and the distance between camera and trap is not consistent, use of a general threshold for check size increased performance. Finding the trap size allows for more specific settings sorting away more false detections, however, it needs to be more refined to not miss detections due to very strict settings. Size check is useful for lessening the impact of data shortage. The models confidence levels needs to be increased for better accuracy, for this more training images would be needed. The results point towards adding a large amount of new weevils and flea beetles to the negative images, with at least as many new rape beetles to the positive images. The datasets of rape beetles were too small and not varied enough to include the needed amount of negative images, regarding both H3G as well as mobile and camera images.

The SWSH method could only analyze a small part of the image at the time. One of the larger obstacles is detection of pests clustered close together with other insects. For solving this problem, a machine learning method seeing larger sections or the whole image at once might be of interest to test.

6.3.2 Field images

Similar to the case with trap images, the performance of insect detection and wrong classification of small black dots was impacted by the resolution of the images and by the distance between camera and insect. Insects with similar size and colour to the rape beetle such as a small fly caused a false detection, but larger flies did not. Young oil seed plants caused false detections due to holes in leaves or shadows in similar size to the pest caused by the flower buds. Especially for 2.1 MP images using Mask R-CNN, that proved more sensitive to differences

in resolution. Both Mask R-CNN and YOLOv5 performed best with less inclusion of occlusion, and Mask R-CNN performed best when few insects were present in one image. The training images of very good clarity were not of enough quantity to reach good polygons on every rape beetle for Mask R-CNN, causing missed detections.

YOLOv5 was also tested briefly as a multi-class detector on images and on video. As seen in the Table 6.4, the overall accuracy is not as high as the other YOLOv5 models. Nevertheless, as seen in Figure 6.10 and in Figure A.4, most of the insects could be detected well, as long as focus was kept, except for the class weevil. In contrast to the class pea leaf weevils, the weevil class often had large distance between camera and insect as well as two types of weevils with different shape and colours included. For classes containing very similar and high quality data the accuracy scored higher. Bumblebees could be detected at varying distances.

It is worth investigating if the performance could increase for Mask R-CNN and YOLOv5 if;

- Using pre-trained weights obtained from training on COCO datasets including only small objects.
- Including background images, with and without insects not of interest, into training.
- Increase the training images without occlusions.
- Using strictly one type of insect per class.
- Utilizing mostly one type of resolution and distance for smaller insects. Or, including more images with varying resolution and distance.

The YOLOv5 has potential as a detector for automatic collection purposes of field images. However, the amount of event splits would have to lessen with more training images, otherwise finding a more stable type of lightweight model for videos would be recommended. Here, event split is when the model stops detecting a present insect for a brief moment.

6.3.3 Field detection future use

TensorFlow Light [40] was considered for deploying YOLOv5 or another light-weight model, such as MobileNet [19], to a mobile device for detection tests in field or for taking images of traps each time a new detection is made. This is because a trap could be suddenly filled with flies in a short amount of time, making it hard to read detections. Especially for sticky traps in windy conditions, where the insects would stick in place in contrast to cup traps where the insects could be moved around to better positions. Also, rape beetles would often come visit the cup on the edges, alone or two at a time, when it was standing empty on the side or put upside down. Therefore images of when insects visits empty cups could perhaps be collected.

6.3.4 Work in a wider context

When placing cameras in fields the integrity of others must be protected. The camera should be placed with this in mind. For cup images this is easier since the camera often face the trap from above. A possible environmental impact of using camera systems instead of physical visits is reduced travelling and more accurate measures against pests. In the future, if pests could be automatically detected on plants along with the damage they do, and perhaps even the beneficial insects interaction with them, it could aid in understanding the relationships between them better. In that way, if the model implemented can handle multiple purposes, the yearly inventory of pests in fields could aid in data collections for environmental and agricultural research.

7

Conclusion

In this chapter, the conclusions made from the discussions of the results are presented as well as the prospects of future work.

7.1 Conclusion

The sliding window approach with HOG based SVM, combined with check size and trap detection, shows promise to be useful for developing an automatic warning system for pests caught in traps. Compromises were made, regarding what state of the pests to prioritize for detection. The best combination found in this thesis was to find trap and crop to detection using YOLOv5, and then based on size of the cropped image set new thresholds for check size.

A model trained using a RBF kernel with only positive rape beetles and negative patches was the most inclusive for rape beetles in high resolution camera images and therefore deemed to have most potential if more positive and negative patches are added, see Figure 6.4. Similar for models trained with a LinearSVC kernel, for detection of rape beetles with step size 7 on Hunter 3G images in Figure 6.6 and for detection of fungus gnat in smartphone images in Figure 6.7. They were deemed to have potential with further development.

However, more testing is needed for refinement of check size to improve performance. The HOG based SVM models need to be reinforced with more training images to improve accuracy, especially more rape beetles and fungus gnat as positive patches and more flea beetles, light coloured flies and background as negative patches.

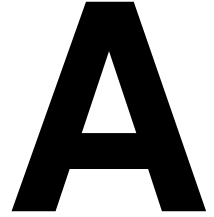
Mask R-CNN and YOLOv5 both proved their potential as rape beetle detectors in field images. YOLOv5 showed multi-class detector potential with distances customized with regards to size of insect. No matter what use, it is deemed beneficial to add more training images to all Mask R-CNN and YOLOv5 models.

7.2 Future work

Testing the sliding window with check size approach with other classifiers and/or adding more data to datasets could be of interest to see if the performance can increase. Building on this concept, carrot root fly pests and leaf hoppers are deemed worthwhile to test for detection. To overcome the difficulties of detecting insects closely clustered, a machine learning model that sees larger parts of or the whole image instead could be pursued and compared.

It is deemed worthwhile to continue to pursue more field images, 64MP for larger insects and Macro 5MP for smaller pests, and build a large scale multi-class detector using a model similar to YOLOv5. If accuracy can be improved and event splits lessened with more training, a YOLOv5 similar model could have potential use on video stream on a field camera to help automate the collection of images through extraction of frames. Using YOLOv5 or another CNN model for detection of insects in trap, empty or not, could be worthwhile to pursue.

Appendix



Additional material

A.1 Extended pest chart

An extended pest chart with translations of pest names, see Table A.1.

A.2 Additional images dataset

An overview of possible negative and positive images included into the datasets, see Figure A.1.

A.3 Additional Results

In Figure A.2 some result of Mask R-CNN trap detection background removal can be seen. In Figure A.3 the models performance for SWSH on test images are shown without size check. Results on additional field images can be seen in Figure A.4.

Table A.1: Pest information chart [5, 6, 21]. TL = Threat level. Since some pests are a larger threat to certain crops depending on the region, the region is denoted (M) for the middle part of Sweden and (S) for the south part of Sweden following crop names. SP = Sightings probable from. The * denotes those crops that could suffer the most damage.

Pest information chart					
Pest	Latin name	Swedish name	TL	Plants infested	SP
Rape beetle	Meligethes aeneus	Rapsbagge	3	Oilseed plants (spring and autumn rape*)	April
Cabbage-stem flea beetle	Psylliodes chrysocephala	Rapsjordloppa	2	Oilseed plants, cruciferous vegetables	June, August
Bean (seed) beetle	Bruchus rufimanus	Bönsmyg	5	Legumes (field beans*)	June
Carrot root flies	Chamaepsila rosae / Psila rosae	Morotsfluga	3	Carrots (S), parsnip (M)	May
Leafhopper	Eupteryx atropunctata	Potatisstrit	5	Potatoes	June
Bird cherry-oat aphid	Rhopalosiphum padi	Havrebladlus	3	Oats, spring wheat, spring grain	May
Grain aphid	Sitobion avenae	Sädesbladlus	3	Spring wheat, spring grain	May
Pea aphid	Acyrtosiphon pisum	Ärtbladlus	2	Legumes	May
Black bean aphid	Aphis fabae	Bönbladlus	2	Legumes, sugar beets, potatoes	June
Frit fly	Oscinella frit	Fritfluga	3	Oats, spring wheat, sweet corn	May
Fungus gnat	Sciaridae	Sorgmygga	2	Young plants	-
Blue stem weevil	Ceutorhynchus sulcicollis	Blygrå rapsvivel	1	Oilseed plants	May
Weevil		Fyrtandad rapsvivel	1	Oilseed plants	May
Pea leaf weevil	Sitona lineatus	Randig ärtvivel	1	Legumes	June

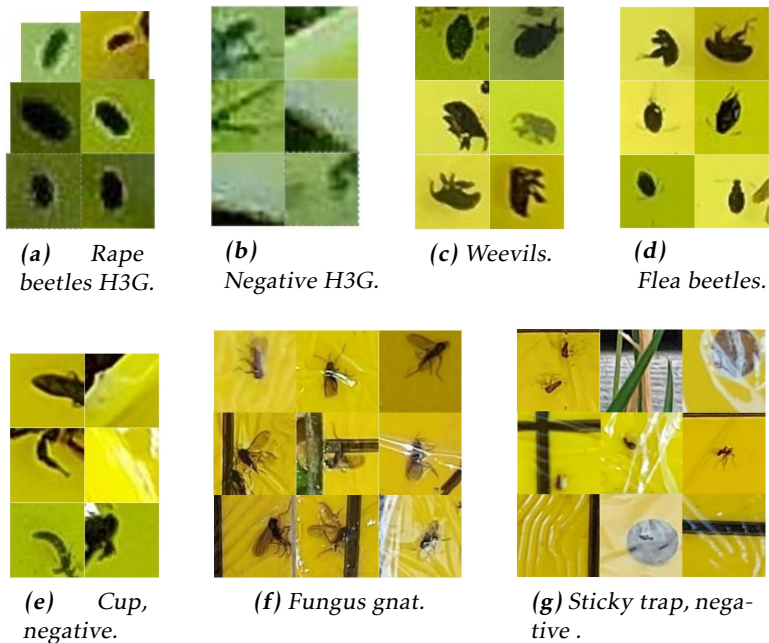


Figure A.1: Example of positive and negative images in the datasets. Some of the crops in c)-e) are from original images provided by Jordbruksverket.



Figure A.2: Mask R-CNN background removal on cup traps. Two of the images were cropped to detection. The a) are examples of passing results. The H3G image in b) needed a rerun.



(a) LinearSVC H3G, stepsize 3.



(b) LinearSVC.



(c) RBF.

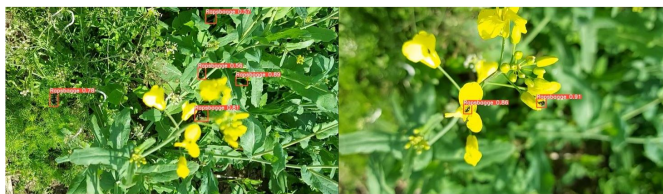
Figure A.3: Examples of the different trap results when no area or density checks are performed. However, the SWSH has a computational stop implemented as it reaches the white information bar at the bottom in H3G image a). Otherwise it would also detect the letters within that area. The detections within each image are 1163 in a), 14 in b) and 179 in c).



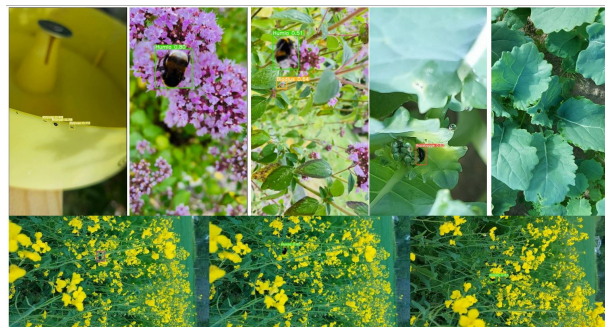
(a) BB1_Rape beetle test on the same video as BB3, 6 seconds. No false positives. The only rape beetle was detected whenever video in focus, however 13 event splits occurred.



(b) BB3 test on the same video as BB1_Rape beetle, 6 seconds. 10 FP (often caused by same spot on plant/ground), and 4 event splits. The only rape beetle was briefly detected.



(c) BB1_Rape beetle. Example of how wrong detections could occur when video out of focus for flower, and how correct detections were more probable when video focused.



(d) BB3. Example of handling occlusions for bumblebees or different distances. Detection of thrips is possible, although wrongly classified since no class. Also a TN of a background, a TP for a rape beetle, and some FP of an aphid and weevils.

Figure A.4: Example of detection in video stream with YOLOv5 using one class rape beetles, BB1_Rape beetle, or 6 classes, BB3. Rape beetles, bumblebee, weevil (RVirvel), pea leaf weevil (AVirvel) and aphids, see A.1 for translations.

Bibliography

- [1] Hunter 3G, . URL <https://www.kjell.com/se/varumarken/hunter/natverk/natverkskameror/hunter-premium-3g-overvakningskamera-p51383>. Accessed: 2022-02-11.
- [2] M5stack ov3660, . URL <https://www.elfa.se/en/ov3660-esp32-psram-timer-camera-module-m5stack-u082/p/30185742>. Accessed: 2022-02-11.
- [3] Makesense, . URL <https://www.makesense.ai/>. Accessed: 2022-03-01.
- [4] Waleed Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow, 2017. URL https://github.com/matterport/Mask_RCNN. Accessed: 2022-05-01.
- [5] Louise Aldén, Anders Arvidsson, Gunilla Berg, Therese Christerson, Alf Djurberg, Lovisa Eriksson, Anna Gerdtsen, Christer Johansson, Lars Johansson, Anders Lindgren, Eva Mellqvist, Lina Norrlund, Annika Sohlman, and Rebecka Östlund. Bekämpningsrekommendationer: Svampar och insekter 2022. 2022. URL <https://www2.jordbruksverket.se/download/18.2e41a0a017fbf0833c41c14a/1648217495565/be17v32.pdf>. Accessed: 2022-05-01.
- [6] Gunnel Andersson, Anna Gerdtsen, Göran Gustafsson, Lars Johansson, Lina Norrlund, and Anders Lindgren. Skadegörare i jordbruksgrödor. 2021. URL <https://www2.jordbruksverket.se/download/18.3aa284aa17ebbd39bc6260c1/1643880042249/be26v6.pdf>. Accessed: 2022-02-07.
- [7] Pernilla Borgström, Karin Ahrné, and Niklas Johansson. *Pollinatörer och pollinering i Sverige Värden, förutsättningar och påverkansfaktorer*, volume RAPPORT 6841. 2018. ISBN 978-91-620-6841-7. URL <https://www.naturvardsverket.se/om-oss/publikationer/6800/pollinatorer-och-pollinering-i-sverige/>. Accessed: 2022-05-10.

- [8] Andriy Burkov. *The hundred-page machine learning book*. 2019. URL <http://themlbook.com/wiki/doku.php>.
- [9] Sergio Canu. Train mask r-cnn for image segmentation (online free gpu), tutorial, 2021. URL <https://pysource.com/2021/08/10/train-mask-r-cnn-for-image-segmentation-online-free-gpu/>. Accessed: 2022-03-04.
- [10] B. Cederberg, G. Holmström, K. Hall, and A. Berg. Slu artdata-banken, 2022. URL <https://artfakta.se/artbestamning/taxon/bombus-1005547>. Accessed: 2022-07-28.
- [11] Hisham Cholakkal, Guolei Sun, Fahad Shahbaz Khan, and Ling Shao. Object counting and instance segmentation with image-level supervision. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12389–12397, 2019. doi: 10.1109/CVPR.2019.01268. Accessed: 2022-02-20.
- [12] Chunfang Deng, Mengmeng Wang, Liang Liu, and Yong Liu. Extended feature pyramid network for small object detection. arXiv, 2020. doi: 10.48550/ARXIV.2003.07021. URL <https://arxiv.org/abs/2003.07021>.
- [13] Weiguang Ding and Graham Taylor. Automatic moth detection from trap images for pest management. In *Computers and Electronics in Agriculture*, volume 123, pages 17–28, 2016. doi: <https://doi.org/10.1016/j.compag.2016.02.003>.
- [14] YINI GAO. A one-stage detector for extremely-small objects based on feature pyramid network. 2020. URL <http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A1417070&dswid=-3054>. Examen-sarb. KTH, EECS.
- [15] Ross Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015. doi: 10.1109/ICCV.2015.169.
- [16] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Li Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. Recent advances in convolutional neural networks. arXiv, 2015. doi: 10.48550/ARXIV.1512.07108. URL <https://arxiv.org/abs/1512.07108>.
- [17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.
- [18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):386–397, 2020. doi: 10.1109/TPAMI.2018.2844175. Accessed: 2022-03-20.

- [19] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. URL <https://arxiv.org/abs/1704.04861>.
- [20] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, TaoXie, Jiacong Fang, imyhxy, Kalen Michael, Lorna, Abhiram V, Diego Montes, Jebastin Nadar, Laughing, tkianai, yxNONG, Piotr Skalski, Zhiqiang Wang, Adam Hogan, Cristi Fati, Lorenzo Mammana, AlexWang1900, Deep Patel, Ding Yiwei, Felix You, Jan Hajek, Laurentiu Diaconu, and Mai Thanh Minh. ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference. Zenodo, February 2022. doi: 10.5281/zenodo.6222936. URL <https://doi.org/10.5281/zenodo.6222936>.
- [21] Jordbruksverket. Växtskyddsinfo. URL <https://fou.jordbruksverket.se/vxinfo/mobil/index.php>. Accessed: 2022-02-23.
- [22] Thenmozhi Kasinathan, Dakshayani Singaraju, and Srinivasulu Reddy Uyyala. Insect classification and detection in field crops using modern machine learning techniques. In *Information Processing in Agriculture*, volume 8, Issue 3, pages 446–457, 2021. doi: <https://doi.org/10.1016/j.inpa.2020.09.006>. URL <https://www.sciencedirect.com/science/article/pii/S2214317320302067>.
- [23] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8):5455–5516, 2020. doi: 10.1007/s10462-020-09825-6.
- [24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2014. doi: [arXiv:1412.6980](https://doi.org/10.48550/arXiv.1412.6980). URL <https://doi.org/10.48550/arXiv.1412.6980>.
- [25] Sevcan Aytaç Korkmaz, Ayşegül Akçiçek, Hamidullah Bınoğlu, and Mehmet Fatih Korkmaz. Recognition of the stomach cancer images with probabilistic hog feature vector histograms by using hog features. In *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 000339–000342, 2017. doi: 10.1109/SISY.2017.8080578.
- [26] Jinsu Lee, Junseong Bang, and Seong-Il Yang. Object detection with sliding window in images including multiple similar objects. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 803–806, 2017. doi: 10.1109/ICTC.2017.8190786.
- [27] Wenyong Li, Tengfei Zheng, Zhankui Yang, Ming Li, Chuanheng Sun, and Xinting Yang. Classification and detection of insects from field images using

- deep learning for smart pest management: A systematic review. In *Ecological Informatics*, volume 66, pages 264–269, 2021. doi: <https://doi.org/10.1016/j.econinf.2021.101460>. URL <https://www.sciencedirect.com/science/article/pii/S157495412100251X>. Accessed: 2022-02-20.
- [28] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. *arXiv*, 2014. doi: 10.48550/ARXIV.1405.0312. URL <https://arxiv.org/abs/1405.0312>.
- [29] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. *arXiv*, 2016. doi: 10.48550/ARXIV.1612.03144. URL <https://arxiv.org/abs/1612.03144>.
- [30] Tao Liu, Wen Chen, Wei Wu, Chengming Sun, Wenshan Guo, and Xinkai Zhu. Detection of aphids in wheat fields using a computer vision technique. In *Biosystems Engineering*, volume 141, pages 82–93, 2016. doi: <https://doi.org/10.1016/j.biosystemseng.2015.11.005>.
- [31] E. Olsson. Dataset collected for an agtech 2030 pilot project, 2022.
- [32] Johanna Orsholm. Detektering och kontroll av rapsbaggar, 2021.
- [33] Johanna Orsholm. Rapsbaggar: Detektering och bekämpning. In *Nummer 5 i rapportserien Agtech innovation*. LiU-Tryck, Linköping, 2021. doi: <https://doi.org/10.3384/9789179290801>.
- [34] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979. doi: 10.1109/TSMC.1979.4310076.
- [35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. In *Journal of Machine Learning Research*, volume 12, pages 2825–2830, 2011. Accessed: 2022-05-01.
- [36] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. doi: 10.1109/CVPR.2016.91.
- [37] Adrian Rosebrock. Sliding windows for object detection with python and opencv, 2015. URL <https://pyimagesearch.com/2015/03/23/sliding-windows-for-object-detection-with-python-and-opencv/>. Accessed: 2022-03-04.

- [38] Dan Jeric Arcega Rustia, Chen-Yi Lu, Jun-Jee Chao, Ya-Fang Wu, Jui-Yung Chung, Ju-Chun Hsu, and Ta-Te Lin. Online semi-supervised learning applied to an automated insect pest monitoring system. In *Biosystems Engineering*, volume 208, pages 28–44, 2021. doi: <https://doi.org/10.1016/j.biosystemseng.2021.05.006>. URL <https://www.sciencedirect.com/science/article/pii/S1537511021001069>.
- [39] Yongbin Song, Guanghua Chen, and Jingjing Liu. An improved mask rcnn with global context modeling for instance segmentation. In *2019 Chinese Automation Congress (CAC)*, pages 5644–5648, 2019. doi: 10.1109/CAC48633.2019.8997063. Accessed: 2022-02-20.
- [40] TensorFlow. Tensorflow lite. URL <https://www.tensorflow.org/lite>. Accessed: 2022-04-07.
- [41] Tao Wang, Kunming Zhang, Wu Zhang, Ruiqing Wang, Shengmin Wan, Yuan Rao, Zhaohui Jiang, and Lichuan Gu. Tea picking point detection and location based on mask-rcnn. *Information Processing in Agriculture*, 2021. ISSN 2214-3173. doi: <https://doi.org/10.1016/j.inpa.2021.12.004>. URL <https://www.sciencedirect.com/science/article/pii/S2214317321000962>.
- [42] Lilian Weng. Object detection for dummies part 3: R-cnn family, 2017. URL <http://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-dummies-part-3.html>. Accessed: 2022-03-20.