Self-Training with Structured Data for Efficient Insect Pest Detection

Bachelor's Thesis

Christoph Wald
Matrikelnummer 17515905
christoph.wald@stud.uni-goettingen.de

Supervisors:
Prof. Alexander Ecker (Georg-August-Universität Göttingen)
Dr. Elias Böckmann (Julius-Kühn-Institut Braunschweig)

September 4, 2025

Contents

Question@Elias: Ich habe am Ende einen Glossar angefangen. Hilft dir das? Wenn ja, soll das ausführlicher oder fehlt was? Wenn nein, kann ich irgendwas anderes tun oder ist das alles so ok für dich mit den ganzen Fachbegriffen?

1 Introduction

Insect pests are responsible for significant production losses in agriculture [1]. However, pesticides threaten the environment by reducing biodiversity [2], particularly among insects, including pollinators that support agricultural production [3]. Overuse of pesticides causes resistance in pests. Additionally, global warming leads to increasing pest populations, which aggravates the problem further [1]. To address these challenges, the concept of integrated pest management (IPM) was developed. As defined by the Food and Agriculture Organization of the United Nations (FAO), it sets the goal of minimizing the use of pesticides by applying alternative management strategies wherever possible [4,5]. IPM is also the legal basis for the use of pesticides in the EU [6], which, among other things, prescribes the monitoring of insect pests when using pesticides. This enables the early and targeted use of pesticides for only small infested patches, the use of less harmful pesticides, or alternative ways of control like beneficial insects. Monitoring is usually based on setting up a grid of sticky traps, which are regularly checked for trapped insects. Because the manual counting requires expertise and takes a lot of time, it is not feasible for larger applications [7]. Hence, computer vision-based systems for automated monitoring of insect pests are considered to be part of the solution. Ongoing efforts to automatize insect pest monitoring have more recently adopted the methods of deep learning (see the surveys Teixeira et al. 2023 [8] and Mittal et al. 2024 [9]). My project aims to facilitate the implementation of deep learning-based, automated insect pest monitoring in the controlled environment of the greenhouse.

The dominant approach in pest monitoring in general involves the supervised training of YOLO models. YOLO ("You Only Look Once") is a deep learning architecture that unifies the tasks of object localization and classification into a single step. It was introduced by Redmon et al. in 2016 [10] and has been improved on since then in several iterations by different contributors. For greenhouses, specifically designed neural network models have been employed successfully in 2021 [11,12], but the recent studies from 2023 onwards use YOLO models on individual datasets [13,14,15,16]. These six papers all report high performance, with mean average precision (mAP) or F1 scores ranging from 0.89 to 0.94.

Main challenges in the implementation are the small size of pests—which, even with high-resolution cameras, occupy only a very tiny fraction of an image—and the high degree of morphological variation among individuals. In less controlled environments additional complications arise from high morphological similarity between different species. These challenges create a demand for large annotated datasets; however, the collection and labeling of biological data require considerable effort and expertise. As a result, datasets are often small and exhibit other limitations, such as class imbalance. For an overview of

these specific challenges, see Teixeira et al. (2023) [8]. As a remedy, the authors recommend semi-supervised learning (SSL), a family of methods for training with only partially data. Despite this, no survey reports applications of SSL in the field of pest monitoring in general and only one study in the context of the greenhouse makes use of SSL methods. Rustia et al. [17] proposed a self-training pipeline as a supplement to a fully supervised model as early as 2021. While the results were promising, the implementation was highly intricate, involving multi-stage setups with hand-tailored solutions. This complexity may explain why the approach has not been further explored. Outside of the greenhouse, in open-field settings, some efforts were made from 2024 onwards, either focusing on a single insect species with more diverse backgrounds or on a set of larger insects [18,19,20,21,22]. Another notable paper in this context is Dang et al. (2024) [23], who employ a field dataset of ten larger insect species and perform segmentation without manual annotation by leveraging a grounding language–image model. While extending this approach with classification and adapting it to the smaller insect pests typically found in greenhouses might be feasible, grounding models will not be considered in this study due to their limited speed and efficiency compared to YOLO-based methods.

Table 1 below shows an overview over the relevant literature. Green indicates a correspondence with the demands of my project, red the opposite, and orange overlappings. Because of the differing datasets and metrics, the results are not directly comparable and left out. Also not seen in the table are the pest classes included in the datasets: Only Rustia et. al [11] covered the same pest classes featured in this project. As of now no YOLO model has been trained to identify common greenhouse pests without relying on supervised training. Although working solutions for automatic insect pest monitoring in the greenhouse have been successfully tested, practical deployment stagnates and is hindered by the need for large, manually annotated datasets for new target domains. To my knowledge, there is no adequate method to reduce the amount of labels needed. The goal of the thesis is therefore an improvement in implementation efficiency by adapting the labeling and training method.

Authors (Year)	Greenhouse	YOLO	Training
Li et al. 2021 [12]	yes	no	supervised
Rustia et al. 2021 [11]	yes	no	supervised
Rustia et al. 2021 [17]	yes	no	SSL (predicted pseudo-labels by separate pipeline)
Costa et al. 2021 [13]	yes	yes	supervised
Zhang et al. 2023 [14]	yes	yes	supervised
Li et al. 2024 [18]	no	no	SSL (teacher-student)
Majewski et al. 2024 [19]	no	yes	SSL (pseudo-labels)
Wang et al. 2024 [15]	yes	yes	supervised
Zhou et al. 2024 [20]	no	no	SSL(teacher-student)
Gomez-Zamanillo et al. 2025 [22]	no	no	supervised, labels enhanced by image processing
Shi et al. 2025 [16]	yes	yes	supervised
Zhou and Shi 2025 [21]	yes	no	SSL(teacher-student)

Table 1: Overview over relevant papers.

I propose that given a specifically structured training dataset, a model can be trained without the need for manual labeling. This is achieved by a combination of image processing and a semi-supervised learning method (SSL) called self-training. The results are expected to contribute to a software foundation for a low-cost, open-source insect pest monitoring solution. It should be trainable on a single GPU and also provide efficient inference. Such a system could support practical deployment in greenhouse production settings and serve as a research tool for further studies.

The key requirement for the proposed solution is the specific structure of the dataset, which should be partitioned into subsets. All images in one subset should only contain one specific pest class as objects. With already existing labels a supervised YOLO model is trained as reference point, which should reach achieve at least 0.9 precision and 0.8 recall per class. Precision is prioritized in this target values over recall for two reasons. First, recall can be improved later through denser placement of sticky traps, whereas precision depends primarily on model quality. Second, low precision risks erroneous pesticide use, which conflicts with the principles of IPM.

The envisioned adapted automated labeling and SSL is done in two stages. First, with image processing segmentation algorithms bounding boxes for the pest classes are created. Since the images are presorted to contain only one pest class, the detected bounding boxes can be directly associated with the corresponding label, yielding a partially labeled dataset. In the second stage, the partially labeled dataset as given by stage one is used in a self-training SSL framework with an adapted YOLO to produce a model.

ToDo: Actualize this in the end, especially the final solution found for the thrips problem

2 Methods

2.1 Data

2.1.1 Dataset description

ToDo: Include mixed images, also in the other sections

The data is provided by the Julius-Kühn-Institut (JKI) and was collected as part of the Smart-Checkpots project [24]. It consists of four subsets of images each associated with one of four insect pest classes (see Table 2). Every image in a subset is supposed to depict several instances of only one pest class trapped on a yellow sticky trap (YST)[1] (see Figure 1).

Common Name	**Scientific Name**	**EPPO Code**
Fungus gnats	Bradysia impatiens, JOHANNSEN 1912 (Diptera: Sciaroidea)	BRAIIM
Tomato leaf miner	Liriomyza bryoniae, KALTENBACH 1858 (Diptera: Agromyzidae)	LIRIBO
Western flower thrips	Frankliniella occidentalis, PERGANDE 1895 (Thysanoptera: Thripidae)	FRANOC
Greenhouse whitefly	Trialeurodes vaporariorum, WESTWOOD 1856 (Hemiptera: Aleyrodidae)	TRIAVA

Table 2: Overview of insect pest classes.

[1] In general, there are different types of traps e.g. pheromone traps and there are also sticky traps with different colours, but YSTs are the most commonly used traps in greenhouses. Yellow is a compromise which most attract most insect pests.

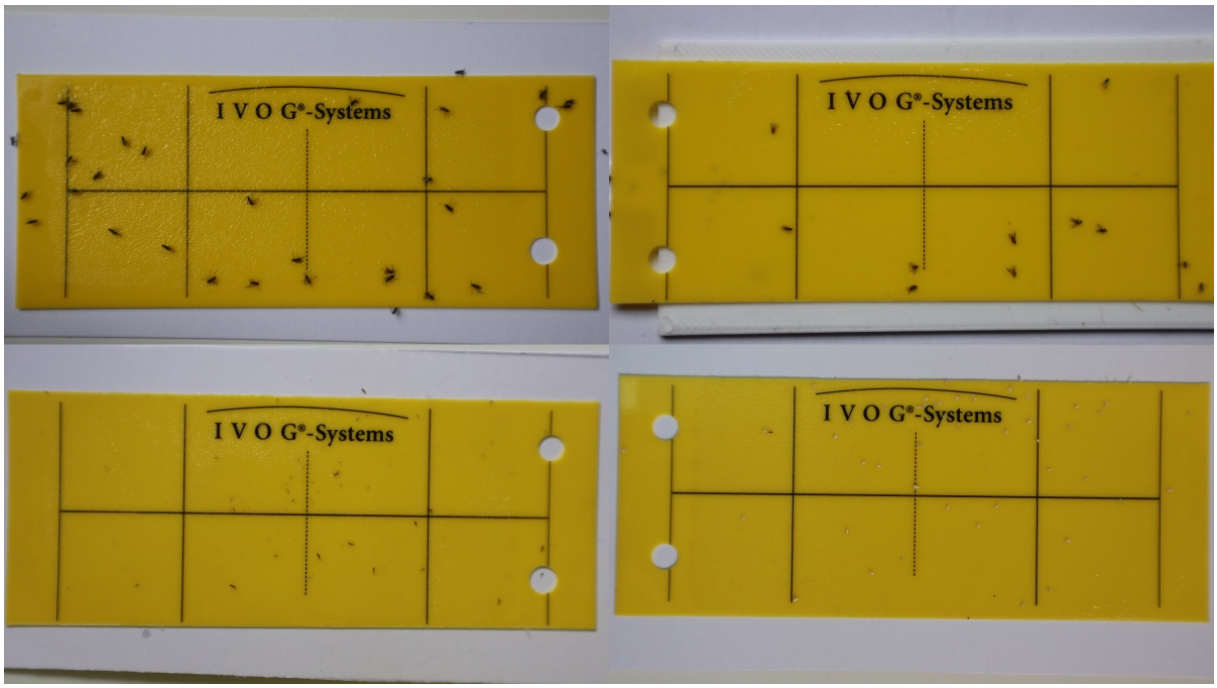


Figure 1: Example images from the four subsets with images from fungus gnats, leaf miner flies, thrips, and whiteflies (left to right, top to bottom). Reduced resolution.

The chosen insect pests are common in greenhouses, particularly on crops such as tomatoes and cucumbers, where they can cause severe damage, up to complete crop loss [11]. As Figure 2 shows, the size of the pest classes and thereby the number of pixels they cover differs considerable. In thrips, also male and female differ in appearance. The female individuals are larger and have good contrast against the yellow background, whereas the smaller and more transparent males are harder to detect (Row 3, box 3,7, and 9).
Question@JKI: Are there wrong insects in the figure? Are male/female thrips assigned correctly?

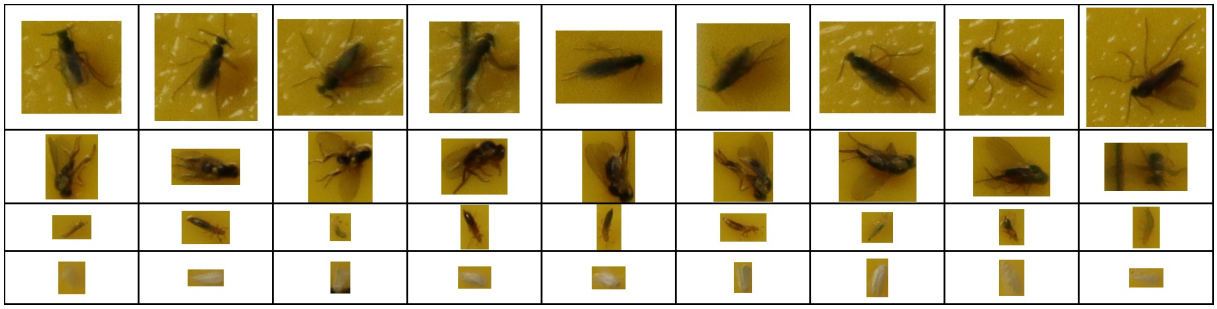


Figure 2: Cropped boxes with fungus gnats, leaf miner flies, thrips, and whiteflies (top to bottom). Original resolution.

The populated YSTs in this dataset were created at the JKI by breeding the insects to the adult stage in a closed environment (see Figure 3a) and exposing the YST in the

environment for some time until it became populated. The traps were then immediately attached onto a piece of white paper, and pictures were taken with a Canon EOS M50 Mark II on a tripod at a distance of approximately 13 cm (see Figure 3b) and a resolution of 6000 × 3368 pixels, resulting in files of about 5–7 MB. No artificial light was used to avoid reflections. The pictures have slightly different view angles and different lighting conditions, some still affected from reflections and some having areas with low sharpness. Some of these aspects may be also be part of the real data to be processed after training.

(a) Breeding station

(b) Camera setup

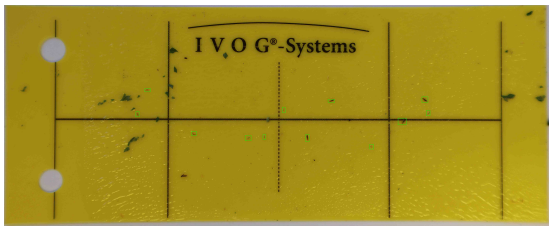
Figure 3: Data creation, pictures with kind permission M. Pollmann, JKI

Question@JKI: How was this done exactly and why?

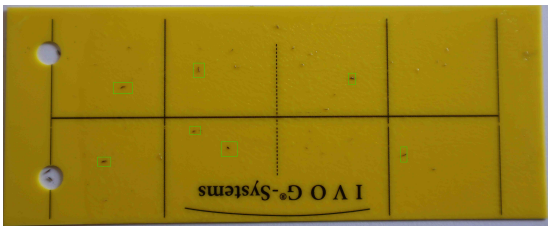
Question@JKI: Is Figure 4b an example for not labeled male thrips or just some larvae present?

ToDo: Describe thrips image creation and the problems it presents and rewrite the following paragraph

On the images of the thrips subset sometimes larger amounts of foliage (see Figure 4a) are present. Also there are thrips in larval stages, which are hard to distinguish from the male individuals. This posed difficulties for the manual labeling, but is also a critical problem for the automated labeling process.



(a) Labeled thrips with foliage on the image



(b) Labeled thrips with large bounding boxes

Figure 4: Examples for thrips images and labels

For each pest class a proportion of images were manually labeled. Other than for the thrips, visual inspection showed the label quality to be consistent, although with a noticeable proportion unlabeled individuals for the fungus gnats. Also the size of the bounding boxes may differ for the batches that different persons labeled (see Figure 4), which will be taken into account.

2.1.2 Preprocessing

As preparation for the project, I sorted the image files by their creation time given by the metadata. The first 100 images for each class were visually inspected and images with identical YSTs and different camera settings were removed manually. These were leftovers of early experiments with the camera setup. I removed further duplicates (identical images) by comparing file checksums. Images with labels that indicated no insects or unexpected insects not belonging to the class of the subset were also removed.

For the thrips, an additional visual inspection on this subset was carried out. This led to the exclusion of 200 images with wrong labels or too much other objects. To improve the quality of the thrips labels further, for the remaining pictures the bounding boxes given by the labels were cut out and inspected by an expert from the JKI. 1276 labels were identified as incorrect and removed from the label files.

Table 3 shows the number of images and labels after all cleaning reported in this section. The dataset is imbalanced, both in terms of the number of labeled images per class and the mean number of instances per image.

Because of the summed difficulties the thrips images and labels posed, the class had to be discarded for further experiments after the supervised training, which showed that no satisfying results are to be expected in the proposed self-training. There are three major reasons for this decision: low sample size, presence of non-target objects, and low label quality. Because of size and morphological differences between adult males and

Class	Images	Labeled	Fraction labeled	Instances labeled	Mean instances per image
FungusGnats	1328	1018	76.66 %	33373	32.78
LeafMinerFlies	1422	659	46.34 %	6518	9.89
Thrips	1134	392	34.57 %	5453	13.91
WhiteFlies	1105	433	39.19 %	25735	59.43
Total	**4989**	**2502**	**50.15 %**	**71527**	**28.59**

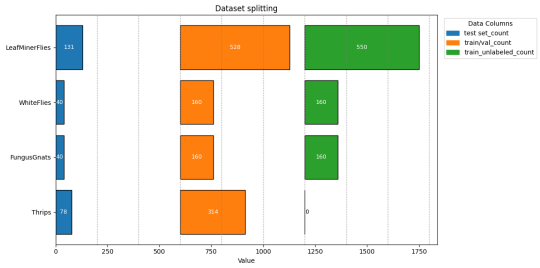
Table 3: Annotation statistics after cleaning and preprocesing.

females the thrips is the hardest pest class to detect and has to be represented with an according number of images and instances in the dataset. Unfortunately, after cleaning the thrips subset has the lowest number of images and labels in the dataset. Furthermore the thrips are easiest to be mixed up with other objects like foliage and therefore require clean images. But the thrips images are the only pest class that has lots of non-target objects, including thrips larvae on their images. The 1276 false labels clearly show that the automated labeling cannot work under the assumption that all objects on an image are instances of the respective pest classes. Finally, the inconsistent labeling quality prohibits proper testing.

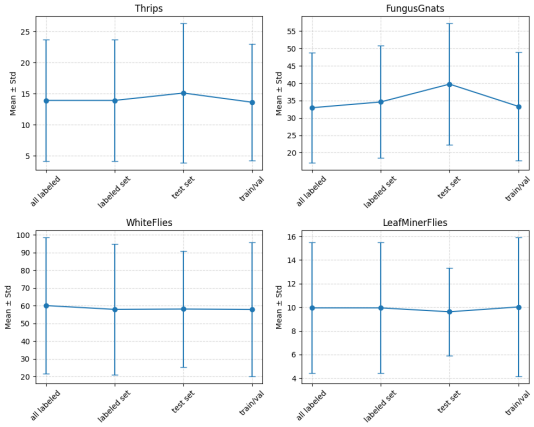
Some labels were found to contain negative coordinates. After visual inspection of samples, these values were interpreted as their absolute counterparts, since this matched the actual object positions. The underlying cause of the negative coordinates could not be identified.

2.1.3 Splitting

From the dataset, three sets are built (see Figure 5a).



(a) Subsets



(b) Mean instances per image for the labeled (sub)sets

Figure 5: Data splitting

To balance the labeled sets, only 200 randomly chosen labeled images from the fungus gnat and the white flies are used, because these pest classes have considerably more instances on a single image than thrips and leaf miner flies. The other labels are discarded and the images retained as unlabeled. The labeled set is then further divided randomly into 20% test set and 80% training set. As Figure 5b shows, the mean instances per image are comparable for the full labeled set, the reduced labeled set, the test set, and the training set. From the unlabeled images excluding the thrips a set is drawn with comparable sizes as the training set. Finally, both training sets are further split into training and validation subsets, using the same 80/20 ratio.

2.2 Supervised training

Question@Alex: Do I have to include more info on YOLOv8 here?
Question@Alex: I did not do this by the book. Do I have to redo it?
Question@Alex maybe just train on full images, as escarda did?
Question@Alex Isn't it strange to train on the tiles but predict on the images, because validation test metrics could diverge?

As a reference point for the automated labeling process, a YOLOv8 model is trained using the manual labels provided. The aim is to demonstrate that the data is sufficient to reach the target metrics of a minimum per-class precision of 0.9 and recall of 0.8. Since supervised training is not the main focus of this project, experiments are not designed to fully optimize results. A small YOLOv8s model pretrained on COCO is chosen because initial tests showed that larger models or later versions did not improve performance. With 4-5 GB of weights, the small model also meets the requirement of single-GPU training and

the more modular code of version 8 is easier to adapt for self-training. No task-specific pretrained models are available to my knowledge.

The images are cropped to the size of the YST with color threshold based segmentation (compare YSTs in Figure 1 and Figure 4). This decreases the size of the dataset by 17,1 GB but also introduces different image sizes. Labels are therefore recalculated after cropping. Since YOLO labels are relative to image width and height, the new image dimensions must be used when transforming labels back into absolute coordinates.

YOLO expects images of size 640x640 and resizes images with larger resolution. Because the insect pest classes are of small size, this resizing might may cause difficulties. Therefore images are tiled. After padding each image such that its dimensions are divisible by 640, the images are tiled into 640×640 patches with a stride of 440. The overlap of 200 pixels corresponds to the size of the largest pest class, the fungus gnats, such that an instance is at least contained in one tile. The labels are recalculated according to the tiles and labels and are only kept if they are contained to at least 80% inside the respective tile. The resulting tiles contain a large number of empty background tiles, which account for over 68% of the tiles in the labeled training set.

Question@Alex: Do I have to document all this augmenation techniques?
Starting from the default settings, short training runs with selected parameter changes were conducted to identify optimization potential, where augmentation settings and the ratio of background to foreground images are identified as the main factors influencing performance. Changing the default optimizer (stochastic gradient descent), learning rate schedule, and loss weights did not yield improvements. Two augmentation parameter sets (see Table 4) are tested in longer training runs. Because the dataset already contains considerable variance (see Section 2.1.1), augmentations are applied only subtly. In parameter set 1, the scaling factor is reduced to avoid unrealistic size differences between objects, and the probability of mosaic augmentation is lowered to prevent the model from learning lighting differences between images. A small amount of mixup is introduced with a 5% probability. All other parameters are kept at the default values. Parameter set 2 is designed to counteract the reduced variance caused by lowering the number of background images in the training set. It retains the changes of Set 1 but additionally reduces the probability of horizontal flipping, to avoid generating mirrored text in the backgrounds. To compensate for the reduced background variance, cropping with a maximum fraction of 10% is allowed, and multiscale training (rescaling of complete batches) is enabled.

For the training runs the proportion of background tiles (tiles without objects) is varied (seeTable 5). The initial training contains 68% background tiles, which is reduced to 10% and 30% in a number of training runs. To test the effect of the thrips images on

Parameter	Default	Parameter Set 1	Parameter Set 2
Scale	0.5	0.3	0.3
Mosaic	1.0	0.25	0.25
Mixup	0.0	0.05	0.05
Crop fraction	0.0	0.0	0.1
Multiscale	no	no	yes
FlipLR	0.5	0.5	0.3

Table 4: Adapted parameter sets

the training of the other classes, the set with only 30% background images is stripped from all images from the thrips subset which changed the background to foreground ratio to 47%. All but the last three trainings are run 100 epochs and a batch size of 16. Though not necessary for regularization, early stopping with patience 10 is used, to prohibit unneccessary training. Two additional trainings are run with 200 epochs and early stopping and a last one also at 200 epochs without early stopping.

Train	Parameterset	Background	Epochs	Patience
train1	1	68	100	10
train2	1	10	98	10
train3	2	10	100	10
train6	2	30	100	10
train7	1	30	100	10
train8	1	30	148	10
train9	1	47*	111	10
train10	1	47*	200	-

* 30% background → thrips deleted

Table 5: Training parameters for different runs.

2.3 Testing the model

The prediction on the test set images is done by sliding over full images with a given stride, predicting on the tiles and afterwards filtering the detected bounding boxes. For stride the value used in tiling the training set is compared to a slightly reduced one. Bounding boxes detected are then first filtered by a confidence threshold. As starting point, the confidence threshold chosen during YOLO training (based on the best F1-score across classes) is used. This value gives the optimal balance between precision and recall. Since precision is favored over recall in this study, prediction with confidence values increased by 0.1 and 0.2 are used for testing. Table 6) shows different combination of stride and confidence threshold that are used in testing.

	confidence threshold	stride
eval1	F1max	440
eval2	F1max+.1	440
eval3	F1max+.1	420
eval4	F1max+.2	420
eval5	F1max+.2	440

Table 6: Evaluation settings for F1 and stride values.

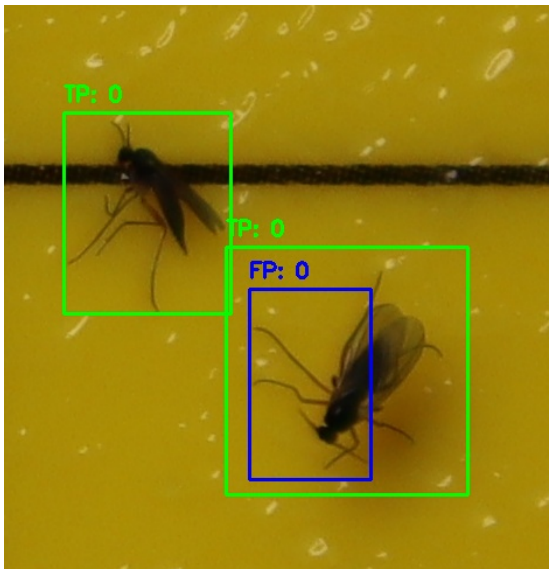
Question: Is the following correctly part of the methods or does it belong to results already?

Non-maximum suppression (NMS) is used to filter overlapping bounding boxes for the same object. It relies on the intersection of union (IoU) metric, which is the ratio between the area where two boxes overlap and the area that the boxes cover when combined. When two boxes have an IoU above a threshold, only the box with the higher confidence is kept. However, if a small box is completely or partially inside a larger box, the IoU will be low, because the intersection is only as small as the small box and the union is as big as the large box. As visual inspection shows, the small objects in the dataset make it necessary to include an extra filtering for this case. IoU for NMS was set to 0.4 and the threshold to filter out the other case was set to 0.5. See Figure 6 a and b for false positives that occour without the extra filter and see Table 7 that the extra filter reduces the number of false positives.

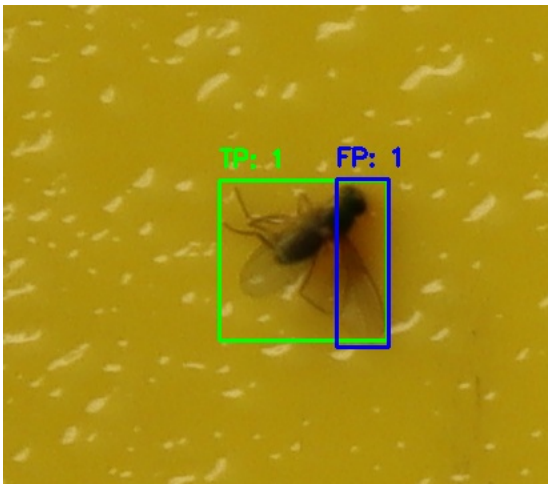
Comparison to the ground truth also relies on IoU and as usual, predicted bounding boxes were counted as true positives with an IoU equal or larger than 0.5. Again, visual inspection shows small correct predictions inside larger ground truth boxes are not covered by this procedure. In an additional routine, also predicted boxes contained at least with 0.9 of their area in ground truth boxes are counted as true positive. See Figure 6 c and d for false positives that occur without the extra comparision and see Table 7 that the extra comparision reduces the number of false negatives.

	TP	FP	FN
No extras	5646	1157	993
Additional containment filtering for NMS	5644	983	994
Additional containment comparison of ground truth	5788	1175	850
Both	5788	839	850

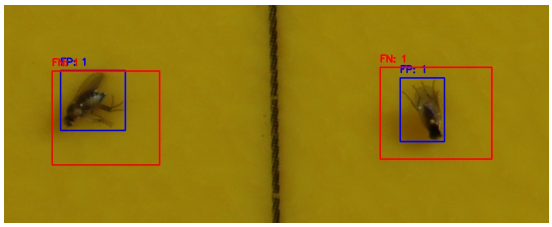
Table 7: Comparison of detection results under different settings of extra thresholds. Model is taken from training run 7, evaluation setting are set 3.



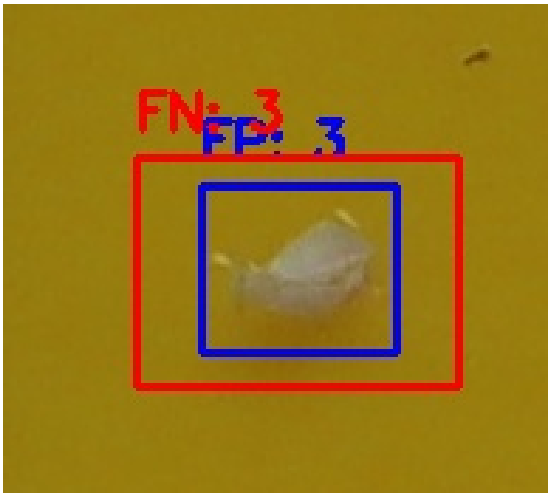
(a) Overlapping Bounding boxes not detected by NMS (fungus gnats)



(b) Overlapping Bounding boxes not detected by NMS (leaf miner fly)



(c) Bounding boxes counted as FP because of low IoU (leaf miner flies)



(d) Bounding boxes counted as FP because of low IoU (whitefly)

Figure 6: Detection errors because of low IoU with small boxes. Green boxes are TP, blue boxes FP, red boxes FN

Visual inspection further revealed, that a proportion of the false positives are due to missing labels (see Figure 7). Therefore for the best models the false positives in the test set are inspected visually and corrected precision metrics are calculated were necessary.

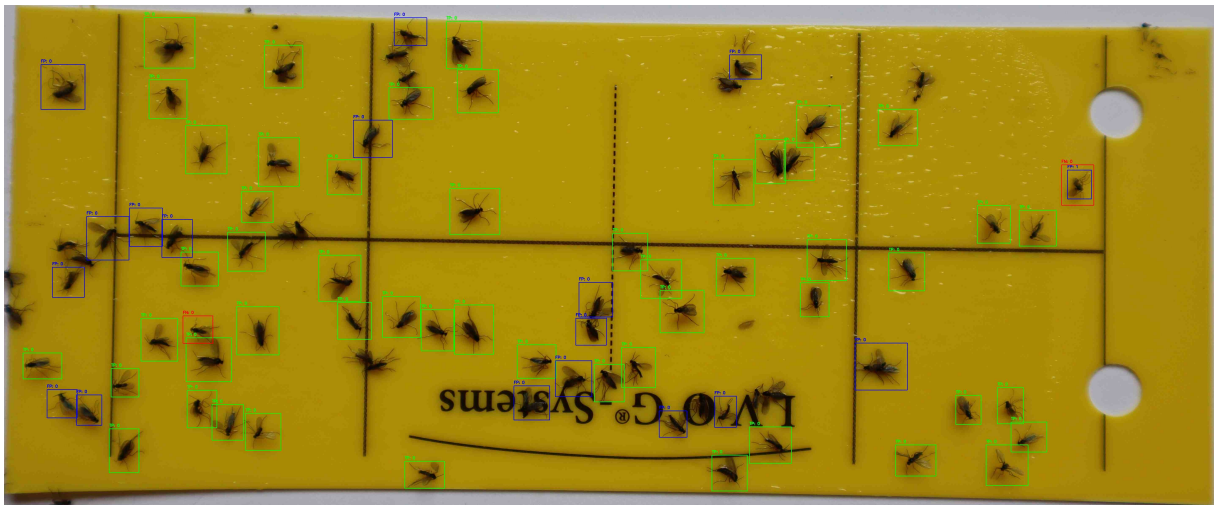


Figure 7: False positive fungus gnats (blue boxes) are actually correct and only false due to missing labels. Green boxes are true positives, red false negatives.

Metrics taken are per class as well as summarized precision, recall and F1 scores.

2.4 Image processing

2.5 Self-Training

The SSL approach implemented in this work follows the self-training paradigm, originally introduced as "pseudo-labeling" by Lee (2013) [25]. In its simplest form, it entails training a model on a small amount of labeled data, use the model to predict on unlabeled data and then use the prediction as pseudo-labels to retrain the model. According to the comprehensive survey on the topic by Amini et al. (2025) [26], self-training remains one of the most prevalent and effective strategies in semi-supervised learning, particularly in domains with partially labeled data.

3 Results

3.1 Supervised training

In the training runs with all four insect pest classes included, the best balance of precision and recall was achieved in training run 7, which used 30% background images and parameter set 1 with slight augmentations over 100 epochs (see Table 8, and compare Table 4 and Table 5 for settings). For testing, always the weights which gave the best

results are taken, which are reported by YOLO and detected by the mAP@50:95 metric. Testing was performed with evaluation set 3, i.e., predictions with a confidence threshold of maximum F1 plus 0.1 and a stride of 420 for the sliding 640 × 640 window. Longer training, as in training run 8, led to the best overall recall, but at the cost of reduced precision. The highest precision values were associated with the lowest recall values and were mostly obtained when applying stricter evaluation thresholds, as in evaluation sets 4 and 5.

train	eval	precision	recall	F1
7	3	0.87	0.87	**0.87**
8	3	0.85	0.88	0.87
2	3	0.85	**0.88**	0.87
3	3	0.86	0.87	0.87
1	2	0.90	0.84	0.87
1	1	0.88	0.84	0.86
7	4	0.90	0.81	0.85
6	3	0.89	0.81	0.85
1	3	0.90	0.79	0.84
3	4	0.90	0.79	0.84
6	4	0.91	0.77	0.83
7	5	0.91	0.76	0.83
6	5	**0.91**	0.73	0.81

Table 8: Metrics with thrips included in training.

Table 9 summarizes the best results per class. For the F1 scores, leaf miner flies and whiteflies benefited slightly from training without thrips (runs 9 and 10), whereas thrips achieved their best value when empty images were included. Precision across all classes was maximized in the shorter training runs (100 epochs) combined with stricter confidence thresholds, with whiteflies in particular profiting from the stronger augmentations of parameter set 2 in run 3. Recall, in contrast, was highest in the longer training runs (8 and 10) with more lenient confidence thresholds; in this case, leaf miner flies profited from increased augmentations.

train	eval	Fun_F1	Leaf_F1	Whi_F1	Tr_F1	Fun_P	Leaf_P	Whi_P	Tr_P	Fun_R	Leaf_R	Whi_R	Tr_R
1	1	0.85	0.92	0.88	**0.75**	0.87	0.93	0.92	0.75	0.83	0.90	0.85	0.75
1	2	0.88	0.92	0.89	0.72	0.89	0.93	0.93	0.79	0.87	0.91	0.86	0.67
1	3	0.84	0.91	0.87	0.71	0.89	0.94	0.94	0.79	0.80	0.89	0.81	0.65
2	3	0.88	0.93	0.90	0.73	0.85	0.92	0.92	0.68	0.90	**0.94**	0.89	0.79
3	3	0.87	0.93	0.90	0.73	0.85	0.92	0.93	0.69	0.90	0.94	0.87	0.77
3	4	0.87	0.93	0.85	0.67	0.88	0.93	**0.95**	0.77	0.86	0.92	0.78	0.59
6	3	0.84	0.92	0.87	0.73	0.88	0.94	0.94	0.76	0.82	0.90	0.82	0.71
6	4	0.87	0.93	0.85	0.63	0.89	0.94	0.95	**0.82**	0.84	0.92	0.76	0.52
6	5	0.83	0.91	0.82	0.62	**0.90**	0.94	0.95	0.82	0.77	0.89	0.72	0.51
7	3	**0.88**	0.93	0.90	0.74	0.86	0.93	0.93	0.72	0.89	0.93	0.88	0.76
7	4	0.87	0.92	0.88	0.70	0.88	0.93	0.94	0.79	0.86	0.91	0.82	0.62
7	5	0.83	0.91	0.85	0.69	0.89	**0.94**	0.95	0.80	0.78	0.87	0.77	0.60
8	3	0.88	0.93	0.90	0.73	0.85	0.92	0.92	0.68	**0.90**	0.94	0.89	**0.79**
9	3	0.88	**0.94**	0.89		0.88	0.93	0.93		0.88	0.94	0.85	
10	3	0.88	0.93	**0.91**		0.86	0.92	0.91		0.89	0.93	**0.92**	

Table 9: Performance metrics for each train/eval combination, ordered by F1, precision, and recall values.

The experiments were structured to first identify evaluation parameters that maximize precision on the dataset with all backgrounds included (see train 1, eval 1–3 in Table 9). This showed that evaluation set 3 yielded the best per-class precision values. Reducing the proportion of background images in the training set shortens training time considerably. On a single GPU core Question@Martin: Which GPU?, an epoch on the full training set takes ca. 18 minutes (100 epochs are approx. 30 hours) and on the training set with only 10% images it takes ca. 7 min. (100 epochs are approx. 12 hours).

Therefore the subsequent runs aimed to reproduce or improve the results with reduced background-to-foreground ratios. Runs 2 and 3 used only 10% background images with parameter sets 1 and 2, respectively. Runs 6 and 7 used 30% background images and compared parameter set 2 (run 6) with set 1 (run 7), as well as the three evaluation settings 3–5, which led to the best overall results presented above. Run 8 tested the effect of longer training. Runs 9 and 10 explored shorter and longer training while excluding thrips images from the training set. Compared with the best results from run 7 (eval 3), the shorter run without thrips maintained precision and slightly improved recall for leaf miner flies. The longer run 10 again reduced per-class precision but achieved the best recall for whiteflies. Across all classes, run 10 slightly outperformed run 7.

train	eval	precision	recall	F1	thrips included in training
10	3	0.90	**0.91**	**0.91**	no
7	3	**0.91**	0.90	0.90	yes

Table 10: Metrics with thrips excluded from training.

Visual inspection of th false positives showed that for the fungus gnats a significant number of unlabeled images caused to false positives (see Figure 7 which in turns leads to a low precision. Therefore the common false positives predicted by the best model/evaluation pair train7/eval3 and train10/eval3 were selected, their confidence scores averaged and the predictions filtered for confidences higher than 0.6. This resulted in 156 predictions for the fungus gnats, 66 for the leaf miner flies and 91 for the whiteflies. As Figure 8 shows, the false positives for the last two pest classes are indeed of mixed quality. However, a visual inspection of the fungus gnat detections returned 116 correctly detected instances. Question@JKI: Is this correct? See images in Dropbox. Recalculating the precision now returns precision 0.94 and 0.93 for the two best training results. With this correction, all classes but the thrips fulfill the required minimum precision and recall values (see Figure 11 below).

Figure 8: False positive with maximum confidences for fungus gnats (BRAIIM), leaf miner flies (LIRIBO) and whiteflies (TRIAVA) (from top to bottom)

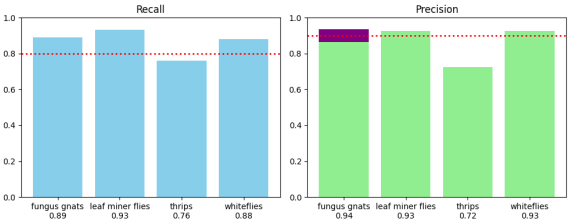


Figure 9: Results for training 7, evaluation set 3. Purple are the corrected labels.

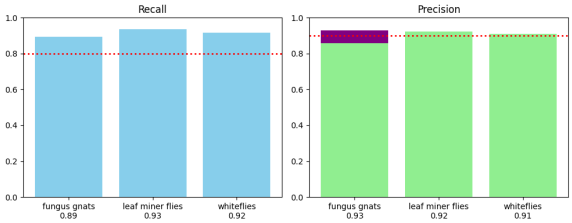


Figure 10: Results for training 10, evaluation set 3. Purple are the corrected labels.

Figure 11: Results with corrected labels for fungus gnats false positives.

Comparing the validation metrics (Figure 12) shows the improvements by reducing the background (upper to middle) and by omitting the thrips images (lower). Because the validation is done on tiles, metrics are not comparable to the metrics reported above, which were taken on full images. The loss curves in the two upper images still show underfitting, however, in the longer training (bottom) the balance between precision and recall degrades: Best mAP@50:05 is already in epoch 135 (which is chosen for above metrics) (ToDo: include the right mAP-metric in the graphs), best recall is reported before, in epoch 124 but best precision is in epoch 162.

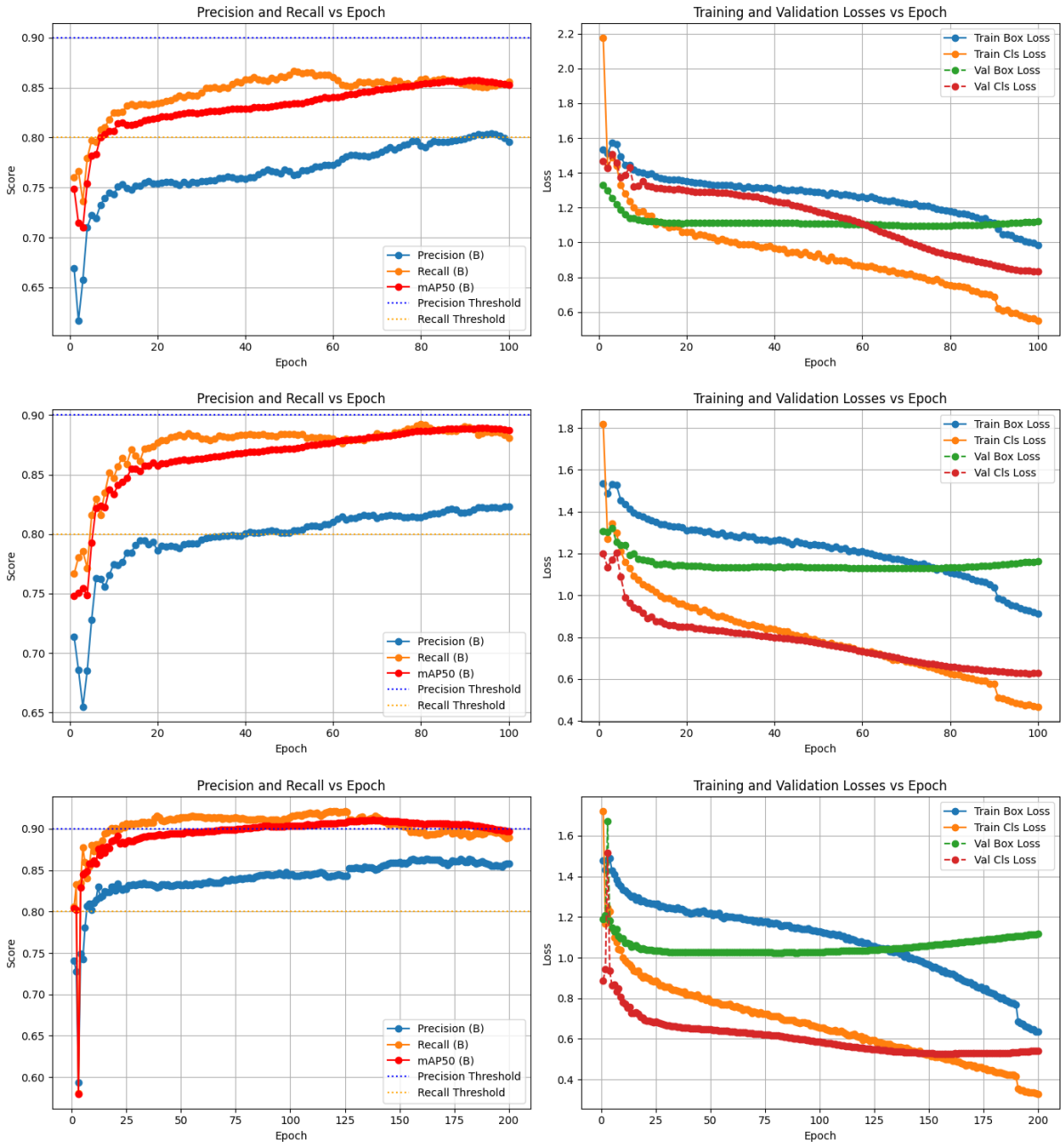


Figure 12: Validation metrics and loss curves for training run 1 (complete training set), 7 (background tiles reduced to 30%) and 10 (without thrips, background images 47%) (top to bottom)

4 Discussion

results supervised: why the thrips are hard to detect? because male adults and larvae are so similar and the labels are ambiguous

Data in a later implementation will differ in at least two aspects. Most notably a domain

shift has to be expected, as images taken in real use cases will likely be of much lower resolution. add picture of the checkpot? mention resolution of thrips, might be solvable by SSL, too Also, in the final usage, several pictures of the same YST are taken over the day. Because the individuals can interact with the glue and become partially hidden or change their appearance over time, this poses an additional problem to be addressed. If the difference between the individuals in the training data and those in the final data becomes too great, individuals might no longer be detected, and the population might be underestimated.

5 Conclusion

6 References

[1] Curtis A. Deutsch et al. "Increase in crop losses to insect pests in a warming climate". In: *Science* 361.6405 (2018), pp. 916–919. DOI: 10.1126/science.aat3466.

[2] Laure Mamy et al. *Impacts des produits phytopharmaceutiques sur la biodiversité et les services écosystémiques. Rapport de l'expertise scientifique collective*. Research Report. INRAE ; IFREMER, 2022, 1408 p. DOI: 10.17180/0gp2-cd65.

[3] Matthew R. Smith et al. "Pollinator Deficits, Food Consumption, and Consequences for Human Health: A Modeling Study". In: *Environmental Health Perspectives* 130.12 (2022), p. 127003. DOI: 10.1289/EHP10947.

[4] URL: https://www.fao.org/pest-and-pesticide-management/ipm/integrated-pest-management/en/ (visited on 04/04/2025).

[5] G.J. Messelink and H.M. Kruidhof. "Advances in pest and disease management in greenhouse cultivation". In: *Achieving sustainable greenhouse cultivation*. Ed. by L. Marcelis and E. Heuvelink. United Kingdom: Burleigh Dodds Science Publishing, Sept. 2019, pp. 311–356. DOI: 10.19103/as.2019.0052.17.

[6] URL: https://food.ec.europa.eu/plants/pesticides/sustainable-use-pesticides/integrated-pest-management-ipm_en (visited on 04/02/2025).

[7] Tiago Domingues, Tomás Brandão, and João C. Ferreira. "Machine Learning for Detection and Prediction of Crop Diseases and Pests: A Comprehensive Survey". In: *Agriculture* 12.9 (2022). DOI: 10.3390/agriculture12091350.

[8] Ana Cláudia Teixeira et al. "A Systematic Review on Automatic Insect Detection Using Deep Learning". In: *Agriculture* 13.3 (2023). DOI: 10.3390/agriculture13030713.

[9] Mamta Mittal et al. “Machine learning for pest detection and infestation prediction: A comprehensive review”. In: *WIREs Data Mining and Knowledge Discovery* 14.5 (2024). DOI: https://doi.org/10.1002/widm.1551.

[10] Joseph Redmon et al. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788. DOI: 10.48550/arXiv.1506.02640.

[11] Dan Jeric Arcega Rustia et al. “Automatic greenhouse insect pest detection and recognition based on a cascaded deep learning classification method”. In: *Journal of Applied Entomology* 145.3 (Apr. 2021), pp. 206–222. DOI: 10.1111/jen.12834.

[12] Wenyong Li et al. “Field detection of tiny pests from sticky trap images using deep learning in agricultural greenhouse”. In: *Computers and Electronics in Agriculture* 183 (Apr. 2021), p. 106048. DOI: 10.1016/j.compag.2021.106048.

[13] Dinis Costa et al. “Intelligent System for Automatic Whitefly Detection in Tomato Greenhouses”. In: *2023 6th Experiment@ International Conference (exp.at'23)*. Évora, Portugal: IEEE, June 2023, pp. 29–30. DOI: 10.1109/exp.at2358782.2023.10546195.

[14] Xiaolei Zhang et al. “Automatic pest identification system in the greenhouse based on deep learning and machine vision”. In: *Frontiers in Plant Science* 14 (Sept. 2023), p. 1255719. DOI: 10.3389/fpls.2023.1255719.

[15] Song Wang et al. “A Deep-Learning-Based Detection Method for Small Target Tomato Pests in Insect Traps”. In: *Agronomy* 14.12 (Dec. 2024). Publisher: MDPI AG, p. 2887. DOI: 10.3390/agronomy14122887.

[16] Jianyu Shi et al. “Small Target Insect Detection Based on Improved YOLOv8n”. In: *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Hyderabad, India: IEEE, Apr. 2025, pp. 1–5. DOI: 10.1109/icassp49660.2025.10890801.

[17] Dan Jeric Arcega Rustia et al. “Online semi-supervised learning applied to an automated insect pest monitoring system”. In: *Biosystems Engineering* 208 (Aug. 2021), pp. 28–44. DOI: 10.1016/j.biosystemseng.2021.05.006.

[18] Xianwei Li et al. “MATeacher: Multi-scale Views Learning and Adaptive Unsupervised Weight for Semi-supervised Pest Region Detection”. In: *2024 6th International Conference on Robotics, Intelligent Control and Artificial Intelligence (RICAI)*. Nanjing, China: IEEE, Dec. 2024, pp. 698–703. DOI: 10.1109/RICAI64321.2024.10911445.

[19] Paweł Majewski et al. “Improved Pest Detection in Insect Larvae Rearing with Pseudo-Labelling and Spatio-Temporal Masking:” in: *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. Rome, Italy: SCITEPRESS - Science and Technology Publications, 2024, pp. 349–356. DOI: 10.5220/0012311300003660.

[20] Jiale Zhou et al. “Mutual learning with memory for semi-supervised pest detection”. In: *Frontiers in Plant Science* 15 (June 2024), p. 1369696. DOI: 10.3389/fpls.2024.1369696.

[21] Cheng Zhou and Fanhuai Shi. “Perceptive Teacher: Semi-Supervised small object detection of Brassica Chinensis seedlings and pest infestations”. In: *Computers and Electronics in Agriculture* 229 (Feb. 2025), p. 109956. DOI: 10.1016/j.compag.2025.109956.

[22] Laura Gomez-Zamanillo et al. “Semi-Supervised Approach for Automatic Counting of Whiteflies With Small Annotated Dataset”. In: *IEEE Access* 13 (2025), pp. 55586–55598. DOI: 10.1109/ACCESS.2025.3552195.

[23] L. Minh Dang et al. “An efficient zero-labeling segmentation approach for pest monitoring on smartphone-based images”. In: *European Journal of Agronomy* 160 (Oct. 2024), p. 127331. DOI: 10.1016/j.eja.2024.127331.

[24] Waldemar Raaz et al. “Smart Checkpots–Proof of concept for a mobile pest and climate monitoring system in greenhouses”. In: *DGG-Proceedings* 11.8 (2023). DOI: 10.5288/DGG-PR-11-08-WR-2023.

[25] Dong-Hyun Lee et al. “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks”. In: *Workshop on challenges in representation learning, ICML* 3.2 (2013), p. 896.

[26] Massih-Reza Amini et al. “Self-Training: A Survey”. In: *Neurocomputing* 616 (Feb. 2025), p. 128904. DOI: 10.1016/j.neucom.2024.128904.

7 Glossary

- augmentation – Techniques that artificially increase dataset diversity by combining or altering images to improve model generalization.
- batches – Subsets of the dataset processed together during one iteration of training.
- class imbalance – When some classes appear much more frequently than others in the dataset, potentially biasing the model.

- COCO – A large-scale labeled image dataset commonly used for object detection and segmentation.
- color threshold based segmentation – A method to separate objects from the background based on color ranges.
- confidence – The probability assigned by a model to indicate how certain it is about a prediction.
- deep learning – Neural networks with multiple layers that learn features from data.
- domain shift – Differences between training and deployment data distributions that can reduce model performance.
- early stopping – A training strategy that stops training when the model performance on a validation set stops improving.
- epoch – Complete passes through the entire training dataset.
- F1-score – The harmonic mean of precision and recall, balancing false positives and false negatives.
- file checksums – Unique hash values used to verify file integrity; they are unambiguous identifiers for files.
- GPU – Graphics Processing Unit; used to accelerate large-scale computations for deep learning.
- grounding language-image model – Models that connect textual descriptions with images to detect or label objects.
- image processing – Processing images using algorithmic methods with manually designed features, as opposed to features learned automatically by deep learning models.
- IoU – Intersection over Union; a metric to measure the overlap between predicted and ground-truth bounding boxes.
- learning rate schedule – A strategy to adjust the optimizer's learning rate during training to improve convergence.
- loss function – A function that quantifies the difference between predicted and true values, guiding the optimizer to adjust weights.
- loss weights – Weights assigned to different components of a loss function to prioritize certain tasks or classes.

- mean average precision – A standard object detection metric averaging precision across classes and confidence thresholds.
- mixup (included in augmentation) – A specific data augmentation technique that blends two images and their labels.
- NMS – Non-Maximum Suppression; a method to remove redundant overlapping bounding boxes in detection.
- mosaic – A data augmentation technique that combines multiple images into one, helping the model learn context and improve generalization.
- neural network – A computational model inspired by the brain, composed of interconnected nodes (neurons) that process data.
- optimizer – An algorithm that updates model parameters to minimize the loss function during training.
- padding – Adding extra pixels around images to make it divisible without remainder.
- patience – Number of epochs to wait without improvement before applying early stopping.
- precision – The proportion of positive predictions that are correct.
- precision-recall curve – A graph showing the trade-off between precision and recall at different confidence thresholds.
- pretrained – Models previously trained on large datasets, used as a starting point for new tasks.
- pseudo-labels – Labels generated by a model for unlabeled data; used in self-training methods to improve learning.
- recall – The proportion of true positives correctly detected by the model.
- self-training – A semi-supervised method where a model trains on unlabeled data using its own predictions as labels.
- semi-supervised learning – Training with a combination of labeled and unlabeled data to reduce labeling effort.
- stride – The step size used when sliding over an image in a window or tiling an image. If the stride is equal to the size of the window, no overlaps are created.
- stochastic gradient descent – An optimizer that updates model parameters using gradients computed on batches of data.

- supervised training / learning – Training a model with input-output pairs where the correct output (label) is known.
- teacher-student – A semi-supervised training setup where a "teacher" model generates labels for a "student" model to learn from.
- validation set – A dataset used to monitor model performance during training; different from the test set, which is only used for final evaluation.
- weights – The trainable parameters of a neural network that determine the influence of each input feature on the output.
- YOLO model – A real-time object detection neural network architecture designed for speed and accuracy.