YOLO-based detection of Halyomorpha halys in orchards using RGB cameras and drones

YOLO-based Detection of *Halyomorpha halys* in Orchards Using RGB Cameras and Drones [⋆]

Francesco Betti Sorbelli[a], Lorenzo Palazzetti[b,a], Cristina M. Pinotti[a]

[a] *Department of Computer Science and Mathematics, University of Perugia, Italy*
[b] *Department of Computer Science and Mathematics, University of Florence, Italy*

Abstract

This paper explores the utilization of innovative technologies, such as drones and artificial intelligence algorithms, for monitoring pests in orchards, with a specific focus on detecting the *Halyomorpha halys* (HH), commonly known as the *Brown marmorated stink bug.* The integration of autonomous drones and suitable vision chips into integrated pest management shows promising potential for effectively combating HH infestations. However, challenges arise from relying solely on deep learning models trained using high-quality images from public datasets. In this work, we significantly improve the quality of preliminary results obtained on artificial datasets by constructing an enhanced dataset of images mainly captured in the field. We then conduct an in-depth analysis of the captured images, considering factors such as blurriness and brightness, to assess the use of our hardware and to improve the performance of the machine learning (ML) algorithms. Subsequently, we proceed to train and evaluate various ML models based on the YOLO framework, employing different metrics to compare their performance. Through the optimization of ML models and the correction of image imperfections, this paper contributes to advancing automated decision-making processes in pest insect monitoring and management, specifically in HH monitoring.

Keywords: Unmanned aerial vehicles, Insect detection, *Halyomorpha halys*, Brown marmorated stink bug, Computer vision algorithms, YOLO

[⋆] This work was supported in part by the "GNCS – INdAM", by "HALY.ID" project funded by the European Union's Horizon 2020 under grant agreement ICT-AGRI-FOOD no. 862665, no. 862671, by MIPAAF, by RESIDUAL, and by RB_DMI_2019.

Email addresses: `francesco.bettisorbelli@unipg.it` (Francesco Betti Sorbelli), `lorenzo.palazzetti@unifi.it, lorenzo.palazzetti@collaboratori.unipg.it` (Lorenzo Palazzetti), `cristina.pinotti@unipg.it` (Cristina M. Pinotti)

1. Introduction

Agriculture is the basis of human sustenance. Yet work in the fields is strenuous that fewer and fewer workers will be found willing to toil under the sun. Furthermore, due to the expanding global population, there is an escalating demand for food. So, it is crucial to enhance both production capacity and quality food standards. Due to these reasons, numerous institutions are actively striving by leveraging new technologies. Until recently, in agriculture there has been an underutilization of state-of-the-art technologies and automation that other sectors of the economy have already benefited from [1]. In light of this, this paper focuses on the transition towards informed automated decision-making processes in agriculture, leveraging innovative technologies such as drones, vision chips, and machine learning algorithms. Our aim is to explore their application in the monitoring of orchards to effectively detect the presence of harmful insects [2, 3, 4].

Orchard monitoring is a complex agricultural activity that necessitates a multidisciplinary approach encompassing agronomy, botany, biology, as well as computer science and engineering expertise [5]. Monitoring pests in orchards is a vital aspect of integrated pest management (IPM) but is recognized as a labor-intensive and time-consuming task. In this paper, we address the IPM of the *Halyomorpha halys* (HH), commonly known as the *Brown marmorated stink bug* (BMSB), as our case study. HH is an invasive stink bug originally native to East Asian regions such as China and Japan [6]. It poses a significant threat as a harmful and polyphagous pest, known to feed on a wide range of host plants. In particular, fruits such as pears, peaches, and nuts are among its preferred targets, leading to crop damage and loss. The spread of HH across the globe can be attributed to human activities, such as global trade, as well as climate change. Its presence was first documented in the United States of America in the 1990s and subsequently in Europe in the 2010s. In Italy, the detrimental impact of HH was first observed in 2012 in the Emilia Romagna region, which is home to some of Europe's most valuable orchards [7]. Only in Italy, HH caused several million euros in damages to the main fruit productions in 2019. Currently, HH monitoring is performed with traps, that are unreliable and increase the overall damage, or using time-consuming active monitoring techniques. Attempts to counteract the HH outbreaks led to a massive increase in broad-spectrum pesticide use, disrupting all previous IPMs, increasing producer costs, and causing negative consequences to the environment and consumers.

In February 2021, the Horizon 2020 HALY.ID project [8] was granted with the objective of automating the monitoring activities by growers and plant health operators by July 2024. The main idea is to minimize or eliminate the reliance on traditional monitoring devices and activities, such as traps, baits, visual sampling, sweep netting, and *frappage* (i.e., tree beating). Instead, an automated monitoring process for scouting the HH is proposed. Ongoing preliminary results concerning the whole automated system for scouting HH are presented by Almstedt et al. [9]. The current system incorporates advanced technological innovations, including a drone equipped with an RGB vision chip, a smart camera integrated into a specially designed sticky trap, and micro-climate stations for collecting temperature, pressure, and other pertinent data. Additionally, another ongoing research involves the analysis of ripe fruits using near-infrared hyperspectral imaging (NIR-HSI) to detect punctures that are not visible to the naked eye.

In this paper, we primarily concentrate on one of the initial objectives of the HALY.ID project, which involves the identification of the HH from in-field images, possibly captured by drones or other devices, utilizing machine learning (ML) models. We decided to rely on the YOLO framework as the ML model since it was the best model in preparatory papers [10, 9], and which is acknowledged for meeting real-time performance requirements which indeed are the ultimate goal of the HALY.ID project. The initial attempts to detect HH have been conducted by using datasets not built in the field. Consequently, when these models have been tested on in-field images, their performance significantly deteriorated. Especially, the results in [9] suggest that ML algorithms trained on datasets formed of optimal and high-quality images sourced from public repositories and artificially augmented, do not produce satisfactory results if then applied to a real scenario like ours. Accordingly, it became imperative for us to construct a suitable dataset comprising predominantly field-captured images obtained from drones or other devices to train, verify, and test the ML model. Since the authors in [9] suggested that factors such as blurriness and brightness of the images captured by the drone contribute to unsatisfactory results, in this study, we thoroughly investigate these factors on the image quality. We conclude that by appropriately configuring the vision chip and optimizing the drone's positioning, issues of blurriness and brightness can be effectively mitigated and are not limiting factors for the use of the drone. Additionally, we observed that a certain level of blurriness enhances the robustness of the model. However, it is important to avoid the misconception that increasing the blur level improves object recognition. We have demonstrated that the localization and detection of the bug deteriorate when all images are excessively blurred. The key lesson to take away is the significance of training the computer vision algorithm on the specific context in which it will be deployed for testing, or in a real-world application.

Contributions and Paper Organization

Our results are summarized as follows:

- We create the first dataset[1] of HH by using in-field images *mainly* taken by a drone, and improve its quality through a preliminary screening process aimed at eliminating unsuitable images prior to the ML training;

- We train and evaluate multiple ML models using the YOLO framework on our built dataset obtaining very good results by evaluating different metrics;

- We also prove the influence of blurriness and brightness on image quality in the HH detection experiments.

The paper is structured as follows: We present the relevant related work in Section 2. The built dataset used for the project is presented in Section 3, while Section 4 covers the conducted ML experiments. Finally, conclusions are drawn in Section 5.

2. Related Works

The utilization of drones in the field of agriculture can be advantageous in various ways. It can be used in conjunction with satellites to create vegetation indicators [11, 12, 13], or for monitoring wildlife and cows [14, 15], just to mention a few. The use of drones can result in the capture of a vast amount of imagery, and when combined with ML algorithms, it can make the system faster and more accurate than human observers in monitoring and estimating animal populations.

Nowadays, there has been a growing emphasis on employing ML techniques for monitoring insect species. Traditional methods such as support vector machine, adaptive boosting, artificial neural network [16, 17, 18, 19], and deep learning techniques based on convolutional neural networks (CNNs) [20, 21, 22] have demonstrated optimal results in insect monitoring. For instance, a

[1] Currently, the dataset is set as private as required by the HALY.ID consortium agreement. At the end of the project, it will be released to the scientific community for research purposes.

novel approach for early detection and continuous monitoring of adult-stage whitefly and thrip in greenhouses has been proposed in [17]. The approach is based on an image-processing algorithm and artificial neural networks. The developed whitefly and thrip identification algorithm achieved very satisfactory results. This proposed approach has the potential to improve IPM strategies and reduce the use of harmful chemicals in greenhouse agriculture.

To the best of our knowledge, apart from the HALY.ID project, only a limited number of studies have showcased the direct detection of insects through aerial surveys conducted with drones in open fields. This is despite the fact that drones can be equipped with specialized cameras capable of capturing high-resolution images of small objects and GPS technology for efficient positioning. One of these works aims to determine the effectiveness of drones in detecting the immobile stage of the *Monema flavescens* [23]. The results indicate that an aerial survey performed with a drone at a height of 3 m above the tree canopy is more efficient and successful in identifying butterfly cocoons than a ground survey. Additionally, the captured images demonstrate the ability to differentiate between open and closed cocoons. So, the authors highlight the potential of drones for detecting insects directly in agriculture.

In the initial phase of the HALY.ID project, there have been several attempts to scouting HH using different imaging technologies. Ferrari et al. [24] evaluate the use of NIR-HSI as a potential technology for detecting HH specimens on various vegetal backgrounds that can mimic field conditions. From a set of hyperspectral images comprising HH, two chemometric approaches have been used to develop classification models. The first one focuses on spectral information, and selects relevant spectral regions for discrimination, while the second one uses CNN to model spatial and spectral features in the hyperspectral images. The authors then merge the two strategies by considering only the spectral regions selected by the first approach for CNN modeling. The results demonstrate the potential of NIR-HSI combined with chemometric analysis and CNNs to detect HH accurately, even when mimicking different background conditions. Although this technology holds the potential to become an effective tool for IPM in the agricultural sector, offering timely and accurate information to prevent substantial economic losses, it is currently not ready for field deployment due to its prohibitive cost.

Additionally, in the context of HALY.ID project, several preliminary studies using RGB cameras have also been conducted to detect HH. Trufelea et al. [25] propose the use of deep learning models to classify pests belonging to the Pentatomidae family. Specifically, they propose to train a CNN to recognize four different kinds of Pentatomidae insects, including HH adult, HH nympha, *Pyrrhocoris apterus*, and *Nezara viridula.* A total of 760 images have been used for training (600) and validation (160). Among them, 520 images are from the Maryland Biodiversity database [26], and 240 from a custom dataset collected by professional cameras. A modified Single Shot Detector (SSD) model having the Intersection over Union (IoU) value of 70.2 as performance indicator has been proposed by the same authors. Subsequently, Ichim et al. [27] investigate the identification of HH insect with four CNNs, namely, GoogLeNet, ResNet101, DenseNet201, and VGG19. The dataset is built on two public datasets with many different insects, and a custom dataset collected with a DJI Mini 2 drone containing images with adults and nymphae of HH. All the images from the dataset are resized to 227 × 227 pixels. Transfer learning and data augmentation are utilized to reduce computational effort during the learning phase, and statistical indicators (such as precision and recall) derived from confusion matrices are employed to evaluate the performance of each CNN. The performances are good when the networks are tested on data similar to the ones in the public dataset, i.e., images of the insects taken in the laboratory with professional cameras by expert photographers. The learning time is also considered in this paper.

Very recently, the “You Only Look Once” (YOLO) [28] model has been used to detect harmful insects in ecological orchards by Sava et al. [10]. The authors evaluate YOLO with region-based CNNs (R-CNN), and several YOLO models have been trained, validated, and tested on the aforementioned Maryland dataset [26], which contains professional macro images of HH in different poses, from different short distances, and at different stages of evolution. As a conclusion, the best results have been obtained using the YOLO-m model which obtains for all metrics (precision, recall, and mean average precision) results above 95%. However, the authors did not evaluate their models on datasets built from in-field images.

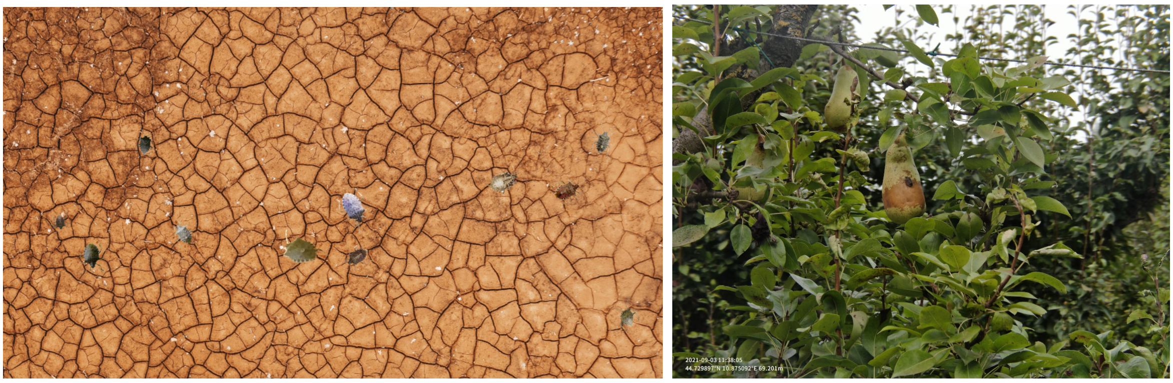
2.1. Motivations for a New Experiment

After the preliminary results in [25, 10], the primary objective became the training of ML models on new in-field images. To accomplish this, in our previous work in [9] we adopted a lightweight deep neural network (DNN) called *CenterNet* [29]. CenterNet is an SSD model capable of processing each image in a single step and is compatible with various backbones, including *RegNet* [30], which we utilized. Initially, CenterNet was trained solely on images captured by the HALY.ID project's drone, specifically the DJI Matrice 300, in a first-person view mode, while flying within the aisles between the rows of the pear trees. Unfortunately, the performance results were unsatisfactory, primarily due to the limited size of the in-field dataset. To address this, an additional semi-artificial dataset was generated, distinct from the Maryland dataset [26] previously utilized in [10, 27]. This semi-artificial dataset includes the silhouette and appearance of the HH extracted randomly from images with varying orientations sourced from public datasets. Additionally, selected random backgrounds were combined with the HH images, and various image transformations (such as scale, rotation, translation, and photometric distortions) were applied, similar to those described in [24].

Table 1: Experimental results (*precision* among all the classes) on different testing datasets, as reported in [9].

Dataset	P
hh_unimore_rgb_lab-images_july2021_scaled	0.97
2021-09-01-by-smartphone_scaled	0.79
farmer_scaled	0.82
drone-camera_2021-08-30_scaled	0.28
drone-camera_2021-08-31_scaled	0.36
drone-camera_2021-09-01_scaled	0.30
drone-camera_2021-09-03_scaled	0.32

At the end of the process, the number of artificial images so created was $8,880$ generated starting from 221 backgrounds, and 105 silhouettes of the HH. By folklore, this size is considered suitable for detection of medium objects. After training CenterNet on the semi-artificial dataset, it was subsequently tested on various other datasets whose results are reported in Table 1 [9]. The results indicate that the precision (P) is high for images captured with smartphones and even better for images obtained from professional cameras in the laboratory. However, the performance is considerably lower when using images taken with the drone. Although one could be tempted to brutally ascribe the poor results to the low resolution of the drone images, we know that a careful analysis of the optics' properties of the DJI Zenmuse H20 camera (the one attached to the DJI Matrice 300 in the HALY.ID project) has been conducted before taking the photos, and the parameters (such as focal length, distance from the subject) were selected so as the resolution is suitable to detect the HH features [9]. Namely, when observed with the naked eye, the images taken by the drone generally appear good, and even excellent in the portion of the photo where the bugs reside. Hence, we suspect that the decline in performance is primarily due to the disparity between the training and testing datasets. In fact, the training dataset comprises silhouettes on artificial backgrounds (see Figure 1a), whereas the testing dataset consists of actual bugs moving on leaves, branches, and pears (see Figure 1b).



(a) Silhouette on a background.

(b) Drone's taken image.

Figure 1: The training dataset.

In conclusion, considering the disparity in results between [9] and [25, 10], we attribute the variations in performance to the neural network employed and the fact that [25, 10] utilize the same type of images for both training and testing. Therefore, in this paper, we propose the utilization of in-field images taken by the DJI Matrice 300 drone or other digital devices not only for testing, but also for training.

To ensure confidence in the image quality and evaluate it, we assess the blurriness and brightness using metrics proposed in the literature. Specifically, we employ the no-reference blurring metric proposed in [31] and the brightness metric proposed by the International Commission on Illumination Laboratory (CIELAB) [32]. Furthermore, we decide to employ YOLO as the object detector due to its excellent performance as demonstrated in [10] for HH detection in an artificial dataset. It is worth noting that YOLO is renowned for its suitability in real-time detection, which aligns with the future extension requirements of our project [33].

In the following section, we will present and explain the process we undertook to create the dataset utilized for the recognition of HH in the field.

3. Created Dataset

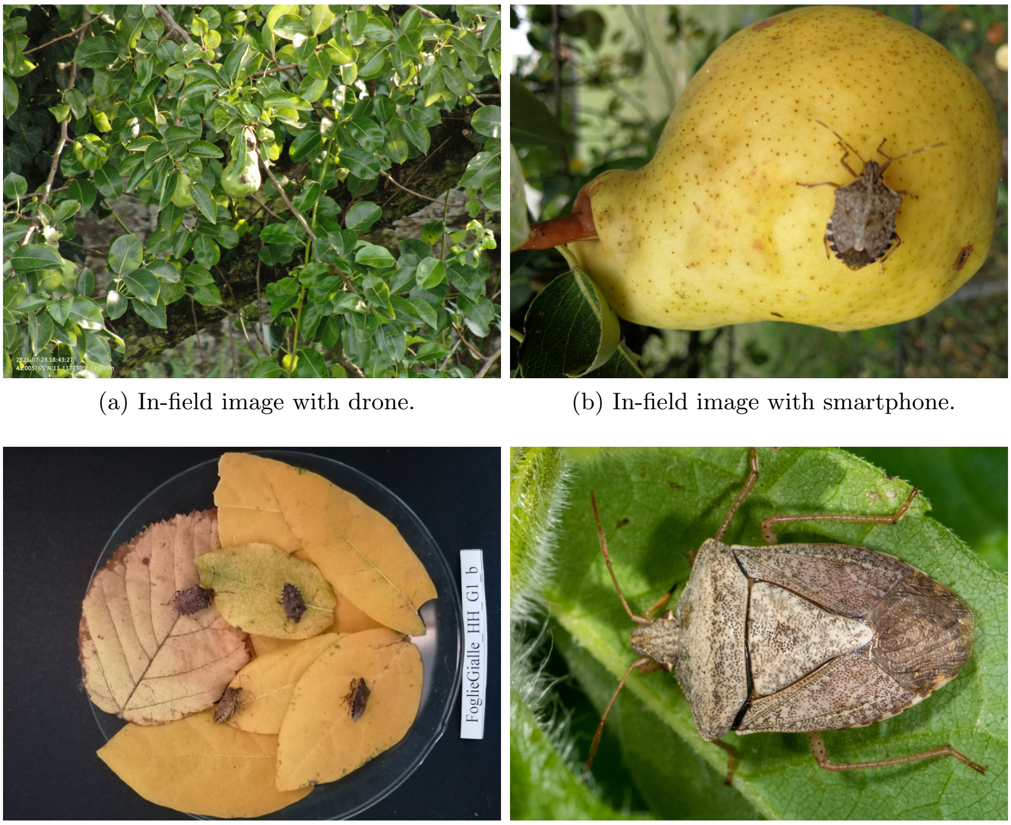
In this section, we present the dataset that we created for our study. We begin by explaining the composition of the dataset in Section 3.1. The dataset primarily consists of in-field and real-world images captured in the orchard using various devices, including the DJI Matrice 300 drone, smartphones, and professional cameras. Notice that the collected images not only contain specimens of HH, but also specimens of the other native stink bug called *Nezara viridula* (briefly, NV). Furthermore, in Section 3.2 we investigate the blurriness and brightness of the selected images to objectively evaluate their quality. Since the dataset was curated by a human operator from the overall acquired images, this assessment provides valuable insights. Additionally, in Section 3.3 we examine the size of the bugs in the images, as this factor plays a crucial role in the labeling procedure. A suitable labeling is essential for the subsequent development of computer vision models, which we discuss in Section 4.

3.1. Dataset Composition

One of the primary goals of the HALY.ID project was to develop a comprehensive and collaborative dataset of in-field images containing stink bugs. This dataset would serve as a valuable resource for computer vision algorithms to effectively identify the presence of the bugs in orchards.

During the summer campaign of the project in 2021[2], we acquired a total of $1,234$ images. Figure 2 shows a few examples of the collected images. Specifically, we captured a total of 855 images using a drone-based automated protocol (see Figure 2a) in a pear orchard located in Carpi, Italy [34, 35]. Additionally, we took 299 images in the same orchards at different times of the day using various digital devices (see Figure 2b) such as smartphones and digital/professional cameras. These images were taken under different lighting conditions, including morning and afternoon. Furthermore, we obtained 80 images of live bugs in a laboratory setting (see Figure 2c). Overall, these $1,234$ images were collected for the HALY.ID project. Finally, to ensure completeness, we downloaded additional 34 high-quality images from the Internet (see Figure 2d), sourcing them from other available datasets, preexisting the HALY.ID project. As a result, our initial dataset consisted of a total of $1,268$ images. Among the 855 pictures taken by the drone, we discovered the presence of bugs in only 653 of them. Furthermore, out of the 299 images taken in the field by other devices, we identified bugs in 274 of them. So, the actual number of images that contain at least one specimen of bugs is $(653 + 274)$ (i.e., 89%) from the orchard and $80 + 34 = 114$ (i.e., 11%) from the laboratory or external datasets.

[2] This study is limited to the project's campaign 2021.



(a) In-field image with drone.

(b) In-field image with smartphone.

(c) Laboratory image.

(d) Maryland dataset image.

Figure 2: An example of image for each category.

We anticipate that several images were excluded from the dataset due to low quality issues. So, the final dataset has 677 images, out of which 83% where captured in the orchard, 12% in the laboratory, and 5% from other dataset in Internet.

The drone's images have been captured by leveraging the DJI Zenmuse H20 camera mounted as a payload on the DJI Matrice 300 drone, hovering at the height of 1.5-2 m above the ground. We took the images of the stink bugs from distances ranging between 3-4 m, resulting in diagonal angles of view ranging from 24-18°. We chose these parameters to ensure a resolution of 0.2 mm which guarantees to identify the features that characterize the HH, in particular the characteristic white-and-black antennas and the connexivum (see Figure 3). To meet the aforementioned conditions, considering the diagonal length of the DJI Zenmuse H20 chip as 9.60 mm and 6,483 pixels [9], the images captured by the DJI Matrice 300 must have a diagonal field of view no greater than 1.3 m. As a result, the above distances and focal lengths have been determined.

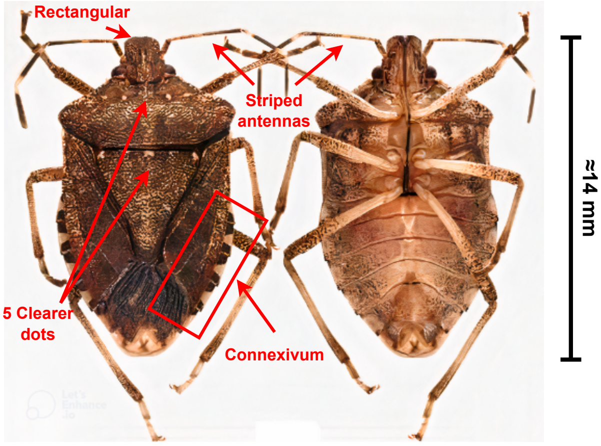


Figure 3: HH and its distinctive features.

As aforementioned, the created dataset contains two classes of stink bugs: the invasive species, *Halyomorpha halys* (HH), and the most common stink bug in Italian orchards, *Nezara viridula* (NV), as shown in Figure 4. The HH and NV are quite different. From our dataset consisting of 677 images, there are 1,803 actual distinct instances of bugs, divided in 1,502 HH and 301 NV specimens. So, the scope of our algorithm is not only to detect a “stink bug”, but also to classify it as either a HH or a NV. Although the classification is not a trivial task, the localization and detection of the bugs is definitely an even more challenging task.

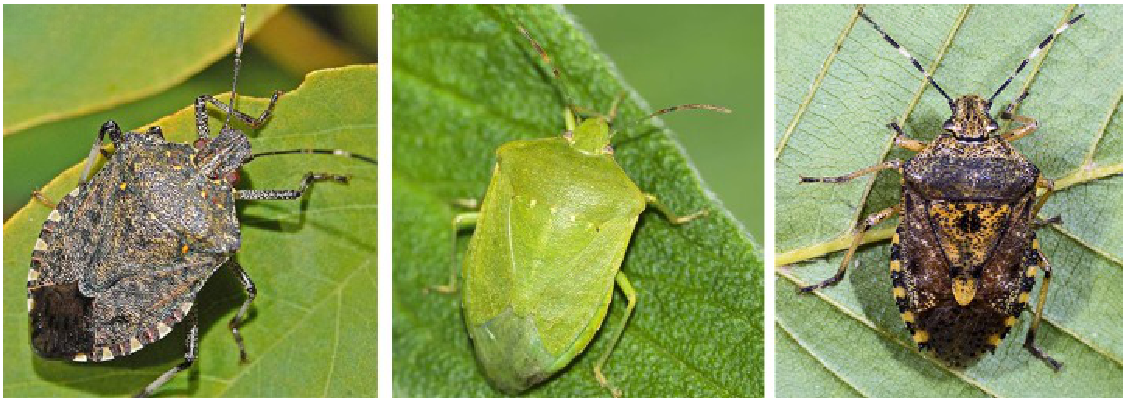


Figure 4: Three Pentatomidae stink bugs: *Halyomorpha halys* (HH, left), *Nezara viridula* (NV, center), and *Rhaphigaster nebulosa* (RN, right, never observed in the monitored orchard).

3.2. Evaluation of Blurriness and Brightness

As previously mentioned, the constructed dataset is heterogeneous as it has been assembled from diverse sources. Consequently, there is a significant amount of variability in the image quality, including variations in resolutions, aspect ratios, and other factors that could represent a bias for the computer vision algorithms, such as non-optimal blurriness and brightness [9].

Regarding the *blurriness*, some of the images taken during the drone's flight are out of focus due to the camera's auto-focusing mechanism. It is worthy to point out that the DJI Zenmuse H20 camera attached to our drone, although it is equipped with the zoom ability, is a compact camera born to work with a wide focal length to capture large landscapes. When capturing images at large distances, unexpected changes in distance have relatively less impact on the auto-focus performance. However, when shooting at small distances, there is a possibility that the camera may fail to accurately focus at the intended distance, leading to out-of-focus images. For example, the irregular shape of the trees, including unpruned branches, can mislead the auto-focus mechanism, resulting in a sharp image focus on an insignificant branch, while the desired subject results blurred.

Regarding the *brightness*, we have observed the presence of certain pictures with excessive exposure in certain areas or even throughout the entire image. This issue is likely attributable to the fact that the majority of the images have been captured during the peak hours of the day (late morning and early afternoon) in summer, when the sun is at its brightest. Additionally, it is not uncommon for the shadows created by the trees to cause an imbalanced distribution of light, resulting in excessive contrast along the edges of the image.

With the purpose of avoiding biased training, a preliminary operator screening procedure has been applied. Specifically, an expert human operator has inspected one image at a time estimating different quality indicators before the labeling phase, e.g., the focus around the target, the brightness balancing, the amount of pixels that characterize the stink bug, and so on. As previously said, 677 images have been selected. However, because the selection process used during the screening phase is operator-dependent, we have chosen to validate our selection phase by utilizing established no-reference estimators from the literature for both blurriness [31] and perceived brightness [32].

Figure 5 illustrates an analysis of the images in the dataset in terms of blurriness and perceived brightness according to the acquisition source. In detail, on the x-axis we list the blurriness scores, while the brightness scores are represented on the y-axis. The blurriness-metric is a score in the range $[0, 1]$: 0 represents blurred image, whereas 1 very sharp image. In other words, a sharpen image is represented by a high blurriness-metric score, whereas a low blurriness-metric score implies image blurred. Similarly, the brightness-metric is a score in the range $[0, 1]$: 0 represents perception of light absents, whereas 1 means dazzling light. Hence, the images with the best metrics (blurriness tends to 1, brightness tends to 0.5) will stay in the central right column of the plots.

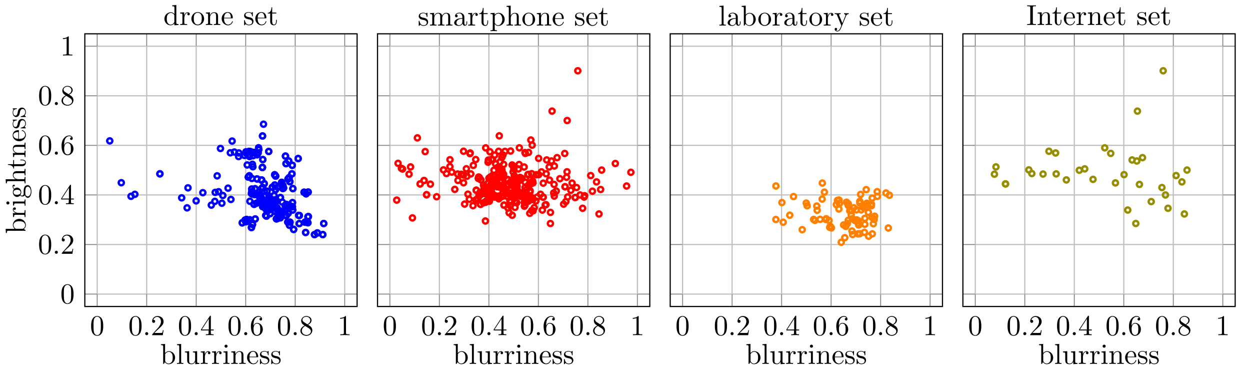


Figure 5: Dataset evaluation on blurriness, and perceived brightness for each acquisition source.

In principle, the laboratory image set (Figure 2c) is the most uniform according the two metrics. This is due to the fact that the images were taken in a light controlled environment, and maintaining a fixed view of the target.

The drone image set (Figure 2a) reports higher values of brightness than the laboratory image set because the images have been taken outside. Nonetheless, we notice that overall the dots (images) are clustered together in a limited area with a few outliers due to the blurriness score. The clustered blurriness behavior can be attributed to the fact that the images have been shot under similar distance and focal length conditions. On the other hand, the presence of a few present outliers can be attributed to the fact that when the drone takes a photo, as explained before, the auto-focus can be misled. However, as previously discussed, the target specimens are still clearly visible in the selected images.

A separate analysis is required for both the smartphone and Internet sets. Regarding to the smartphone set (Figure 2b), it exhibits the largest variance among blurriness scores. This is because the set comprises images taken by different operators, mainly farmers that helped in collecting images, using different devices, each equipped with diverse image sensors. Moreover, differently from the images collected with the DJI Matrice 300, no specific programmatic policy governed the image capture process. As for the Internet set (Figure 2d), it exclusively contains macro photos. In this case, the stink bug specimen occupies a significant portion of the image and is the only element in focus. The low blurriness scores in this set occur when the stink bug occupies a limited area within the image. This is the only set that has quite high values for brightness, which seem to depend on the prevalence of white inside the picture. For example, the image with the highest brightness 0.91 score is a macro photo where a HH lays on a white wall. No image has brightness-score below 0.15 (i.e., an estimated score which establishes an "underexposed" image).

Table 2: Dataset analysis and composition. The average (avg) is among all the pictures.

Set	num.	blurriness			brightness			perc.
		avg	min	max	avg	min	max	
Drone	289	0.69	0.05	0.92	0.40	0.24	0.69	43%
Smartphone	274	0.48	0.03	0.84	0.44	0.28	0.90	40%
Laboratory	80	0.66	0.38	0.84	0.33	0.21	0.45	12%
Maryland Dataset	34	0.49	0.10	0.86	0.52	0.28	0.91	5%
Total	677							100%

In conclusion, as summarized in Table 2, the drone image set offers on average good quality, also in comparison with the other image sets. Its performance validates the parameters that we set using the DJI Matrice 300. Furthermore, the operator selection process has resulted in a satisfactory dataset of images, considering the available quantity.

Overall, the dataset demonstrates minimal issues related to brightness and blurriness. Nevertheless, we conducted a further investigation that involves the YOLO neural network to assess with more certainty the impact of blurriness on our dataset. We construct, as described in Section 4, three distinct training sets, called Rand, Best, and Worst, based on the blur scores. We train the YOLO models using each of these sets to gauge the impact of blurriness on the learning capability of the YOLO models. Subsequently, we evaluate the three trained models on the same test set. It appears that no one of the three training sets dominates the others. As demonstrated in the experiment discussed in Section 4.6, blurriness can be a limiting factor. However, based on this assessment, we can conclude that the level of blurriness in the created dataset is not critical.

3.3. Bug Size Analysis and Labeling Phase

The consequent labeling procedure, performed to allow the training of computer vision models, has been done through the well-known open-source software called `make-sense` [36]. We decided to draw bounding boxes including the antennae, the paws of the bug, as well as the body so as not to lose any HH feature. Indeed, the acquired experience concerning the resolution with the employed devices (DJI Zenmuse H20 and other smartphones) permitted us to recognize all the features previously detailed in Figure 3.

In Figure 6 (first row), we provide insights into the size distribution of the different bounding boxes for the two stink bug classes. The x-axis represents the width in pixels, while the y-axis represents the height in pixels of the drawn bounding boxes for each set of pictures. In Figure 6 (second row), we illustrate the distribution of the bounding box positions relative to the examined pictures. Essentially, these plots display the center position of each bounding box as a percentage with respect to the width and height of the picture. So, on each x-axis and y-axis, we depict the x coordinate and y coordinate of the bounding box center, respectively, as a percentage relative to the maximum x and y coordinates of the image. Furthermore, the analysis categorizes the information from the bounding boxes based on the adopted source of acquisition. The sources include drone, smartphone/professional camera, laboratory, and the Maryland (Internet) image set.

Concerning the size distribution in Figure 6 (first row), as said before, the built dataset is heterogeneous since we have used pictures taken by the drone, by the farmers, by the entomologists, and images obtained from the Internet. Not surprisingly, the sizes of the pictures we have (in pixels) directly impact the detection of stink bugs, as well as the accuracy of determining their size and position, along with other distinctive features. As a consequence, the dimension of the HH varies from, approximately, $30 \times 30\,\text{px}$ to $1000 \times 1000\,\text{px}$. Differently, the NV samples have a limited bug size variance. Notice that the majority of images, i.e., 97% of instances, which contain NV have been taken using the drone, and the remaining 3% using smartphone cameras. However,

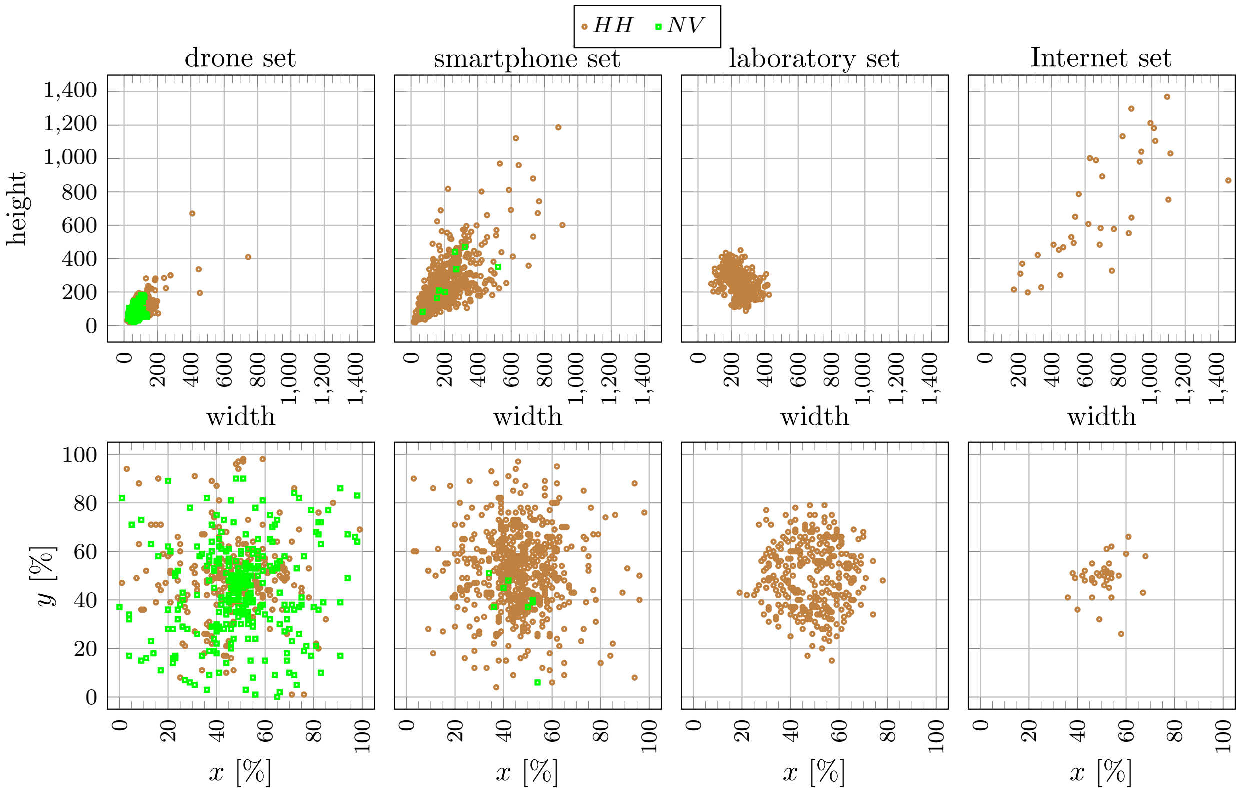


Figure 6: Bounding boxes analysis.

although there is a large variance in the bug size, the vast majority of the bug instances, i.e., more than 50% of instances, are in the range of 200 × 200 px. This is because 289 of 677 (total) images have been taken by the drone with a strictly configured setting. When examining images within a single category, it is evident that the drone set exhibits the smallest bounding boxes and the least variability in size. This observation is particularly true when considering only the NV class. Considering the smartphone set of images, we note a larger variability in the size, with bounding box sizes up to 1200 × 900 px. We also find that the largest NV instances are in this particular set. However, we can note that the more is the bounding box size, the more rarefied are the instances in the plot. The huge variability in the acquisition devices' image sensor characterizes the way how bounding boxes components spread inside the plots. Differently, the laboratory set contains bounding boxes which are extremely similar with respect to their side sizes. As we can evaluate in the example of Figure 2c, the bugs are placed on a disc with some underneath leaves, and the image is completely focused on them. Therefore, the few differences in the size depend on the poses of the bug, and the amount of occlusion duty to the handcrafted background. Finally, taking under consideration the images downloaded from the Internet, we observe that this set contains the largest bounding boxes of the built dataset regarding the HH. Indeed, this tiny set is characterized by high resolution macro images, where the bug represents the target of the shot.

Concerning the position distribution in Figure 6 (second row), differently from what we stated for the bounding box sizes, the drone set highlights a strong variability for bug positions for both the classes. In fact, even if the majority of the bugs approximately reside in the center of the images, there are some instances whose bounding boxes are located at the borders of the image. This behavior can be attributed to the strict acquisition strategy employed by the drone. Since the position of the bugs in the orchard, especially on the trees, is unpredictable, the drone images capture a wider range of bug sizes and positions. A similar behavior is exhibited for the smartphone set, where the bounding box positions span over the entire surface of the images. We affirm that it is a predictable behavior because, as mentioned before, this set is built using the most diverse image sensors of the dataset. To conclude, both the laboratory and the Internet sets are characterized by a very stable positioning of the bugs. Indeed, in both the sets the bugs approximately cover the center of the image. The differences observed in the laboratory set can be attributed to the circular placement of bugs on a disc by hand. On the other hand, in the Internet set, macro photos typically center the target (stink bug) in the middle of the picture (see Figure 2d).

In the next section, we evaluate our computer vision algorithms from the created dataset in order to detect the stink bugs.

4. Evaluation of the Localization and Detection of the Bug

In this section, we detail and discuss the setup and results of our conduced experiments from the created dataset. We start by giving an overview of the used computer vision algorithms and tools (Section 4.1) as well as the adopted metrics for the comparison (Section 4.2). We describe the training configuration phase in Section 4.3. We thoroughly compare and analyze first the training and validation results in Section 4.4, and then the testing results in Section 4.5, obtained from different models. Finally, we evaluate the impact of the blurriness on the previous results (Section 4.6).

4.1. Algorithms and Tools

Here, we briefly recap the YOLO algorithm giving an insight of its working process, and the well-known strategy of the *Transfer learning* (TL) that we used to identify the stink bugs.

The YOLO algorithm [28], the fifth version of the well-know YOLO (You Only Look Once) firstly introduced in [37], is a deep learning-based architecture based on the PyTorch framework that is used to conduct this experiment. The main innovation of this family of algorithms is framing the object detection problem as a regression problem instead of a classification task by spatially separating bounding boxes and associating probabilities to each of the detected images using a single CNN. YOLO is lightweight and fast, and also needs much less computational capabilities than the other current state-of-the-art architecture models while keeping the performance near to them [28]. In particular, it can process images at 45 frames per second, so making YOLO suitable for performing the detection of the bug directly on an edge-computing device possibly embedded in the drone itself.

Furthermore, since we have a limited amount of images and limited computational power, we decided to perform the training phase either from scratch or by using the TL paradigm [38]. When training a computer vision model, the TL gives the possibility to reuse aspects of a computer vision algorithm already trained in depth on a huge amount of images for a new model for which far less images are available. The process takes relevant parts of the existing ML model and transfer them to solve a new, possibly similar, problem. Hence, a key part of TL is the generalization. This means that only the knowledge that can be used by another model in different scenarios or conditions, is transferred. Instead of models being rigidly tied to a training dataset from scratch, models trained using TL will be more generalized. Models developed in this way can be utilized in changing conditions and with different datasets.

4.2. Adopted Metrics

Let us review a few definitions before going in depth with our results. As explained before, YOLO detects the stink bug by returning a *prediction box* B_p, and the consequent labeling associates to each bug a *ground truth box* B_{gt}. For each B_p and B_{gt}, the IoU is defined as the ratio between the intersection area between B_p and B_{gt}, and the union area between B_p and B_{gt}, i.e., $\text{IoU} = \frac{\mathcal{A}(B_p \cap B_{gt})}{\mathcal{A}(B_p \cup B_{gt})}$, where $\mathcal{A}$ represents the area function. A *detection box* is considered a prediction box if the IoU is above a threshold τ. If not differently stated, we assume $\tau = 0.5$. We define:

1. **True Positive** (T_P): A correct detection, i.e., a detection with IoU $\geq \tau$ and object class identified.

2. **False Positive** (F_P): A wrong detection (i.e., $B_p \neq 0$) and detection with IoU $< \tau$. Note that this includes the case of no-overlap with a B_{gt} (Region of Interest equals to B_{gt}). Namely, if $B_{gt} = 0$, we have IoU $= 0$.

3. **False Negative** (F_N): A ground truth not detected, i.e., missed detection $B_p = 0$. Since $B_{gt} > 0$, it holds that $B_p \cup B_{gt} > 0$.

Note that, since $B_{gt} \cup B_p \neq 0$, the IoU is always correctly defined.

Once again, our experiments consider two stink bug classes: HH and NV. We measure the performance on the test set for each class, as well as for the combined performance of the two classes, by considering four metrics. Concerning the first two, we have precision (P) and recall (R), computed as follows:

$$P = \frac{T_P}{T_P + F_P}, \qquad R = \frac{T_P}{T_P + F_N}, \tag{1}$$

where T_P, F_P, and F_N are computed by fixing $\tau = 0.5$. Note that $T_P + F_N$ at the denominator of R is the number of stink bugs, i.e., the number of ground truth boxes computed during the labeling process. Precisely, when R refers to HH (respectively, NV), $T_P + F_N$ is the number of HH (respectively, NV) found in the training set. When R refers to all classes, $T_P + F_N$ is the number of HH $\cup$ NV found in the training set.

Furthermore, we compute the mean AP $m_{0.5}$ (Pascal VOC challenge [39]) also known as $mAP[0.5]$, and $m_{0.95}$ (MS COCO challenge [40]) metrics, also known as $mAP[0.5 : 0.05 : 0.95]$). These metrics refine the T_P and F_P definitions by using the *confidence* parameter γ, i.e., a likelihood value returned by the network. Specifically:

1. **True Positive** (T_P): A correct detection, i.e., a detection with IoU $\geq \tau$, given that $\gamma \geq \tau_\gamma$.

2. **False Positive** (F_P): A wrong detection, i.e., a detection with IoU $< \tau$, given that $\gamma \geq \tau_\gamma$.

To compute the m_τ metric with $\tau = 0.5$, denoted as $m_{0.5}$, the precision and recall values are computed with the IoU threshold $\tau \geq 0.5$ and varying the confidence threshold τ_γ. For each considered confidence threshold τ_γ, only the prediction boxes that satisfy $\gamma \geq \tau_\gamma$ are considered. Then, among such images, the values T_P and F_P are determined based on the IoU condition.

Consequently, the relative precision and recall values are recomputed. Then, a curve is built by plotting for each value of the recall (on the x-axis) the corresponding precision value on the y-axis and the approximated area of such a curve is returned as the mean average precision value $m_{0.5}$. The term "average" comes to the fact that the area of the curve refers to several values of confidence values. Note that when τ_γ decreases, the recall score cannot decrease because the T_P cannot decrease and the denominator (ground truth) remains the same; while the behavior of precision is not predictable because both T_P and F_P cannot decrease, no claims can be made about the recall ratio.

Later on, in our discussion, we say that an object detector is *confidence-robust* if the precision is little affected by the variations of the confidence level. If that is the case, it means that all the prediction boxes have a high confidence level, and the precision level remains almost constant. In other words, an object detector is robust if the m_τ score with $\tau = 0.5$, i.e., $m_{0.5}$ and the P value, are close. The $m_{0.95}$ metric repeats the computation of m_τ by changing τ between $0.5 \leq \tau \leq 0.95$, with steps of 0.05; and returns the average of all the computed values m_τ. From now on, we say that the detector is *IoU-robust* if the average precision is little affected by the IoU variations. If that is the case, the actual IoU value of the prediction boxes is high (close to 1) for all the images. Hence, if the object detector is IoU-robust, the $m_{0.95}$ and $m_{0.5}$ scores are close.

In the following, we explain how to train our neural networks based on the YOLO models. We work with three different training sets to demonstrate the satisfactory quality of the images, i.e., the minimum impact of blurriness issues on our dataset, as anticipated in Section 3.2.

4.3. Configuration of the Object Detectors

In this section, we explain how we built the neural networks based on the YOLO models. Before testing the computer vision algorithms, a suitable training phase of the networks is required utilizing the created dataset of pictures. To accomplish this, the entire dataset has been divided into two parts: a *testing* set of 135, and the remaining 542 images, corresponding, respectively, to the 20% and 80% of the dataset.

For three times, the 542 images have been then partitioned into two sets: a training set consisting of the 407 (i.e., 60%) the images, and a validation set consisting of the remaining 135 (i.e., 20%) images. In fact, in order to investigate the claim that "the less blur an image has, the more accurate the prediction will be" [9], the 542 images have been organized as three distinct training and validation sets based on a blurriness score [31]. Specifically, the least 407 blurred images are selected and denoted as the Best training group. Similarly, the Worst training group consists of the most 407 blurred images. Finally, a Rand training group of 407 images randomly selected is simply assembled. Then, for each Rand, Best, and Worst group, it is created a corresponding validation set using the remaining 135 unselected images. These three groups will be used to train different object detectors, as explained below.

In order to extend the number of samples for the models, we performed an *augmentation phase*. Specifically, for each image of the group training set, three new images obtained by random transformations are also included. The extended set contains different transformations such as image hue, saturation, and value (HSV) augmentation, translation, scale, flipping, and mosaic. By performing this augmentation, the total number of training samples increased from 407 to $1,628$. In order to implement this, we employed the augmentation techniques recently proposed in the YOLO's library [28] which applies random on-the-fly transformations during the training epochs. Figure 7 shows an example of three different transformations, where a combination of mosaic enhancement and HSV-saturation (left), scale-in and transformation (middle), and combination of mosaic enhancement and scale-out (right), are shown. Since different images differ for their size, and since that TL uses weights obtained from the COCO dataset [41] which is composed by pictures with a size of 640×640 px, we decided to scale down all the pictures to this particular value. This process is divided into two sub-phases: initially, the images are cropped in a square shaped eventually padded with white color, and subsequently scaled down to the fixed side length of 640 px without losing the original ratios.

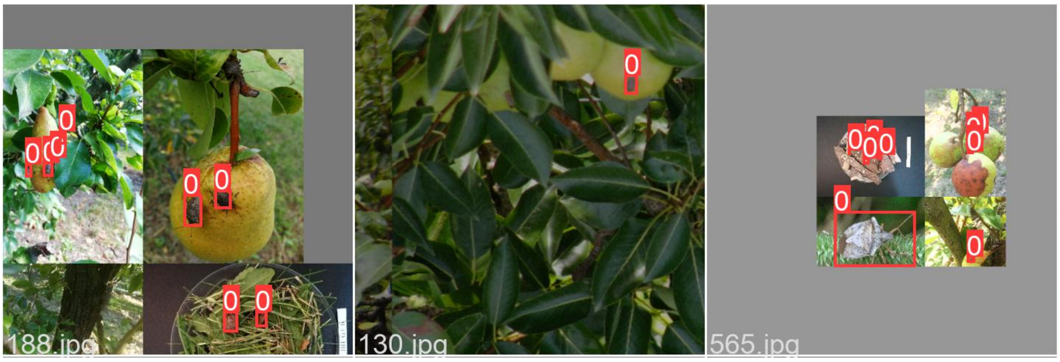


Figure 7: Example of image augmentation.

Each augmented training group will be used to validate and train an object detector based on a YOLO model. Specifically, we trained the following versions of YOLO, namely, small ($\mathcal{S}$), medium ($\mathcal{M}$), and extra large ($\mathcal{X}$). Each YOLO model has been trained and evaluated using an NVIDIA Tesla V100 with 16 GB of VRAM, provided by Google Colab. Furthermore, the models are trained by exploiting the pre-trained weights of COCO dataset, thus implementing the TL. We trained each model by setting a few parameters[3] such as batches with 32 pictures, 200 epochs, learning rate equals to 10^{-2}, SGD optimizer [42], and image size of 640 × 640 pixels. Section 4.4 displays the training results. Finally, nine object detectors are obtained because the $\mathcal{S}$, $\mathcal{M}$, and $\mathcal{X}$ YOLO models are trained with the three RAND, BEST, and WORST groups with TL mechanism.

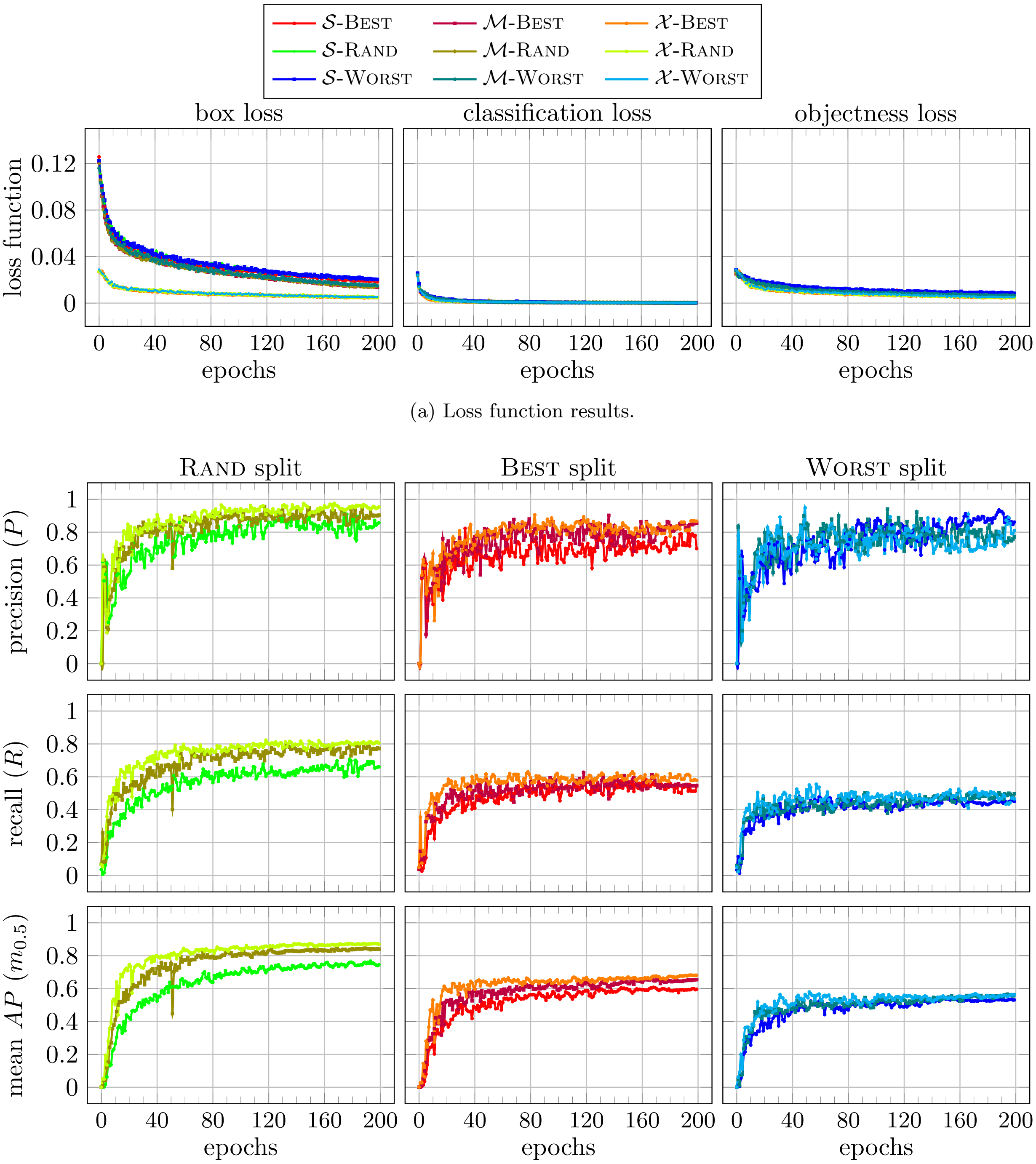
4.4. Training and Validation Results

In this section, we evaluate the training the validation performance among the three aforementioned groups, i.e., RAND (random selection), and BEST and WORST (informed selection). Specifically, in Figure 8 we report the training and validation performance of the resulting YOLO models. For all the models and for all the training splits, Figure 8a reports the training performance in terms of loss functions, while Figure 8b reports the specific metrics used for the validation performance (i.e., P, R, and $m_{0.5}$). These are reported on the y-axis. Moreover, each plot shows the epochs on the x-axis.

Concerning Figure 8a, in each of the three plots nine curves are shown: one curve for each combination of training split (RAND, BEST, WORST) and YOLO model ($\mathcal{S}$, $\mathcal{M}$, $\mathcal{X}$). The first plot reports the *box loss*, namely the function which establishes how well the model guesses bounding boxes coordinates. We observe that the models converge to a loss close to 0 fairly quickly. All the three training groups converge faster when the $\mathcal{X}$ model is used. The *classification loss*, which measures how well the model distinguishes between different classes, and the *objectness loss*, which is roughly speaking the confidence that some object exists in a given box, rapidly converge for all the models and all the groups. One can notice that the objectness loss, although it reaches a plateau, remains slightly higher than 0.

Focusing on Figure 8b, the three rows report the *precision*, *recall*, and *mean AP* achieved by the YOLO models during the validation phase (see Eq. (1)). The first column depicts the results of the three models trained on the RAND split, while the central and the last columns show the results for the BEST and the WORST split, respectively. Notice that the used metrics are values

[3] For the training parameters, we refer to the YOLO's yaml file named *hyp.scratch-low* [28].



(a) Loss function results.

(b) Precision, recall, and mean average precision metric results.

Figure 8: Training and validation results using blur score as splitting parameter.

averaged on the two classes HH and NV to be recognized. In all the rows, the models trained on the Rand split reach better performance, faster than the models trained on the Best and Worst splits, where the $\mathcal{X}$ is always the best. While the three groups behaves almost the same for the P metric, the R and $m_{0.5}$ performance of Worst and Best are worse than those of Rand. The P curves of Worst and Rand fluctuate much more than those of R and $m_{0.5}$.

Summarizing, we note that all the models obtain a satisfactory performance with both the loss functions and metrics. Also, they stabilize their trends reaching a plateau in 200 epochs. So, we can state that 200 epochs represent a reasonable trade-off between quality of the solution, and training time. Furthermore, we observe that the models trained on randomly selected images (Rand) reach the best results on all the metrics with a stable and smooth trend. The models trained on the Worst split exhibit the worst performance, while the Best split allows the models to place in between the Rand and the Worst splits. Finally, focusing on the size of the models, we can notice that the larger is the model, the higher is the score, and this remark is valid for each metrics, and for each split considered.

4.5. Testing Results

In this section, we finally validate all the YOLO models on the testing set, without applying any transformations.

Table 3 reports the results on the testing set achieved by the three different neural networks obtained by training the YOLO models with the three different training splits, i.e., Rand, Best, and Worst. In general, looking at all the three groups, we can observe that the $\mathcal{M}$ and $\mathcal{X}$ models obtain a performance above 89% for all classes. The R metric scores for the same models varies between 73% and 85%. Both HH and NV classes are recognized with a good balance between P and R. For the HH class and Rand split, P is always above 92%, and hence approximately a tenth of the HH bugs is misclassified. However, R is only above 75%, which indicates that the models tend to miss the HH achieving more F_N. The NV class has higher P than HH, although obtains a worse performance on R. This means that the NV individuals are rarely misclassified, although they are often unrecognized.

One can observe that the job is hard. Namely, during our labeling phase we have observed that also expert phytosanitary operators or entomologists sometimes miss the stink bugs on the pictures. Generally, the $m_{0.5}$ of the HH class is higher than that of the NV class, and the difference

Rand split

Mod.	HH Class				NV Class				All Classes			
	P	R	$m_{0.5}$	$m_{0.95}$	P	R	$m_{0.5}$	$m_{0.95}$	P	R	$m_{0.5}$	$m_{0.95}$
$\mathcal{S}$	0.84	0.74	0.80	0.52	0.77	0.66	0.70	0.38	0.80	0.70	0.75	0.45
$\mathcal{M}$	0.92	0.75	0.85	0.58	0.91	0.77	0.82	0.51	0.92	0.76	0.83	0.55
$\mathcal{X}$	0.92	0.78	0.85	0.60	0.92	0.79	0.83	0.59	0.92	0.79	0.84	0.60

Best split

Mod.	HH Class				NV Class				All Classes			
	P	R	$m_{0.5}$	$m_{0.95}$	P	R	$m_{0.5}$	$m_{0.95}$	P	R	$m_{0.5}$	$m_{0.95}$
$\mathcal{S}$	0.77	0.72	0.77	0.48	0.86	0.67	0.72	0.36	0.81	0.69	0.75	0.42
$\mathcal{M}$	0.90	0.73	0.81	0.56	0.89	0.67	0.72	0.36	0.89	0.73	0.81	0.54
$\mathcal{X}$	0.98	0.71	0.84	0.59	0.99	0.83	0.86	0.62	0.97	0.77	0.85	0.61

Worst split

Mod.	HH Class				NV Class				All Classes			
	P	R	$m_{0.5}$	$m_{0.95}$	P	R	$m_{0.5}$	$m_{0.95}$	P	R	$m_{0.5}$	$m_{0.95}$
$\mathcal{S}$	0.89	0.74	0.81	0.54	0.80	0.70	0.71	0.38	0.84	0.72	0.76	0.46
$\mathcal{M}$	0.95	0.73	0.84	0.58	0.94	0.77	0.81	0.52	0.95	0.75	0.82	0.55
$\mathcal{X}$	0.95	0.77	0.86	0.61	0.95	0.85	0.87	0.60	0.95	0.81	0.86	0.60

Table 3: Results of YOLO models trained on different images.

between $m_{0.5}$ and P is smaller for the HH class than for the NV class. This means that the models for the HH class are more confidence-robust than those for the NV class. Instead, the difference between the $m_{0.95}$ from $m_{0.5}$ is high for both the classes. This means that there is a high variations in the IoU size of the prediction boxes.

The results on the "ALL Classes" confirm the above results. The models are not IoU-robust because the $m_{0.95}$ and $m_{0.5}$ scores significantly differ. Instead, the models are relatively confidence-robust since the distance between P and $m_{0.5}$ is moderate.

Looking at the metrics, there is a dominance of the $\mathcal{X}$ (extra large) model, which confirms the behavior of the training. Namely, concerning the Best split, it reaches the highest values for P on both the classes HH and NV (98% and 99%, respectively) with the $\mathcal{X}$ model. However, with the same model, the best split reports the worst R scores, except for the NV class. One possible interpretation is that when presented with very sharp images, the model can establish a strong understanding of the distinctive features of stink bugs, and it almost never misclassifies. However, it lacks of flexibility in the decision, and it misses several bugs.

The Rand is the set of images that guarantees the most balanced P and R values. This is likely due to the random selection, which enables the networks to train on a more diverse set of samples. As a result, they acquire a higher level of generality that aids in the training process. As regard the Worst set, it surprisingly reaches really good results. In particular, we can observe that results of the $\mathcal{S}$ model dominate those of the $\mathcal{S}$ model with the Best and Rand sets, but in general there is no dominance. We believe that these results simply confirm that there is not a marked gap between the best and worse samples in our dataset, as proved in the previous Section 3.2). In other words, the quality of our dataset is on average good.

In conclusion, we obtained a significant improvement in the results with respect to those reported in Table 1 (data from [9]). We attribute this achievement to the quality of our dataset and the fact that both the testing and training sets comprise images captured under similar conditions. In contrast to [9], our dataset does not include images captured in completely different contexts, which contributes to the improved performance.

Finally, to further support our assertion that blurring is not a significant issue in our dataset, we propose conducting an examination in the next section where we intentionally introduce a strong level of blurring and evaluate the results on our dataset.

4.6. Impact of Blurriness

The previous experiments, where the three splits were arranged based on their blurriness scores, seem to suggest that the presence of out-of-focus regions within an image does not pose an issue for the detection. This is evident as the results of the Worst split are not significantly different from the Best split, and overall, the Rand split achieves the best performance. In this section, our objective is to systematically explore the impact of blur by deliberately blurring a certain percentage of photos from a specific split. We begin by introducing a blur effect on 50% of the training and validation samples, and subsequently expand it to 100%. Next, the networks are subsequently retrained using the same parameters, but this time on the transformed set of images. By following this approach, we aim to disprove that increasing levels of blur would paradoxically enhance the detection performance.

To achieve appropriately blurred images, we introduce blurriness by convolving each original image with a suitable matrix of size 5×5 [43]. This convolution applies a smoothing effect to the interior ridges using the `cv.blur` function. Figure 9 depicts the impact of this kernel. Specifically, we can observe that the convolution operation produces a resulting image with slightly smoothed contours, effectively simulating an image captured with subtle flickering.

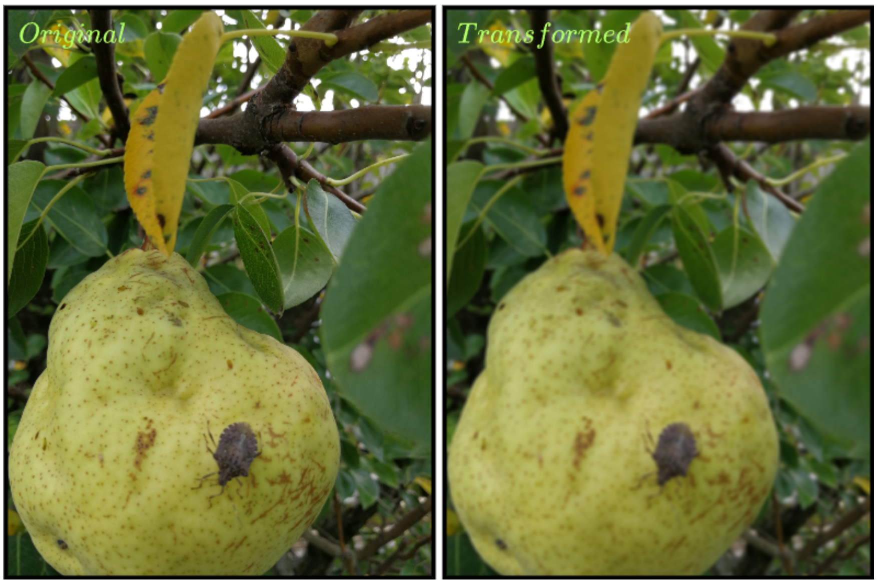


Figure 9: Example of blur kernel effects.

As previous mentioned, we repeat the training of the YOLO networks, i.e., $\mathcal{S}$, $\mathcal{M}$, and $\mathcal{X}$, by using transformed training and validation set with increasing percentage of blurred images. For each previously created split, we intentionally introduce blur to 50% and subsequently 100% of the samples. Figure 10 displays the training performance of the Rand split with 50% of the images transformed. Specifically, the first row illustrates the loss functions, namely bounding box loss, classification loss, and objectiveness loss. The second row presents the validation metrics, including precision (P), recall (R), and mean average precision ($m_{0.5}$).

Overall, the training behavior of the models on transformed images is similar to what has already been observed in Figure 8. Indeed, as regard to the loss functions, we can basically observe that all of them converge very close to 0 after a few epochs. Furthermore, the $\mathcal{X}$ model demonstrates a tendency to converge faster compared to the other two networks. Conversely, the $\mathcal{S}$ model is the slowest in terms of convergence. When considering the validation performance, we observe a rapid stabilization of all metrics, although each line exhibits some noise in its trend. Consequently, we can conclude that, from a training perspective, the introduction of 50% induced blur does not significantly affect the learning performance of the networks. The behavior remains the same even when more blurred images are introduced. Also the training results of the models trained on 100% blurred images do not display any significant discrepancies. So, we have opted not to plot them.

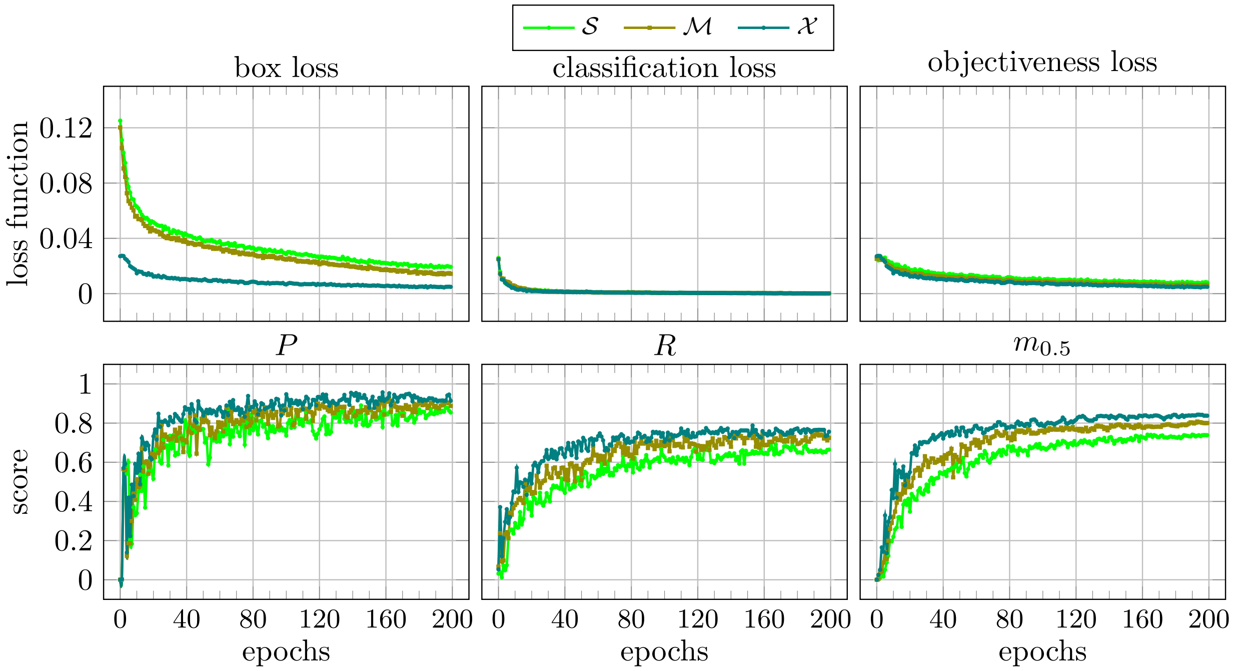


Figure 10: Training results using RAND split with 50% of images convolved with blur kernel.

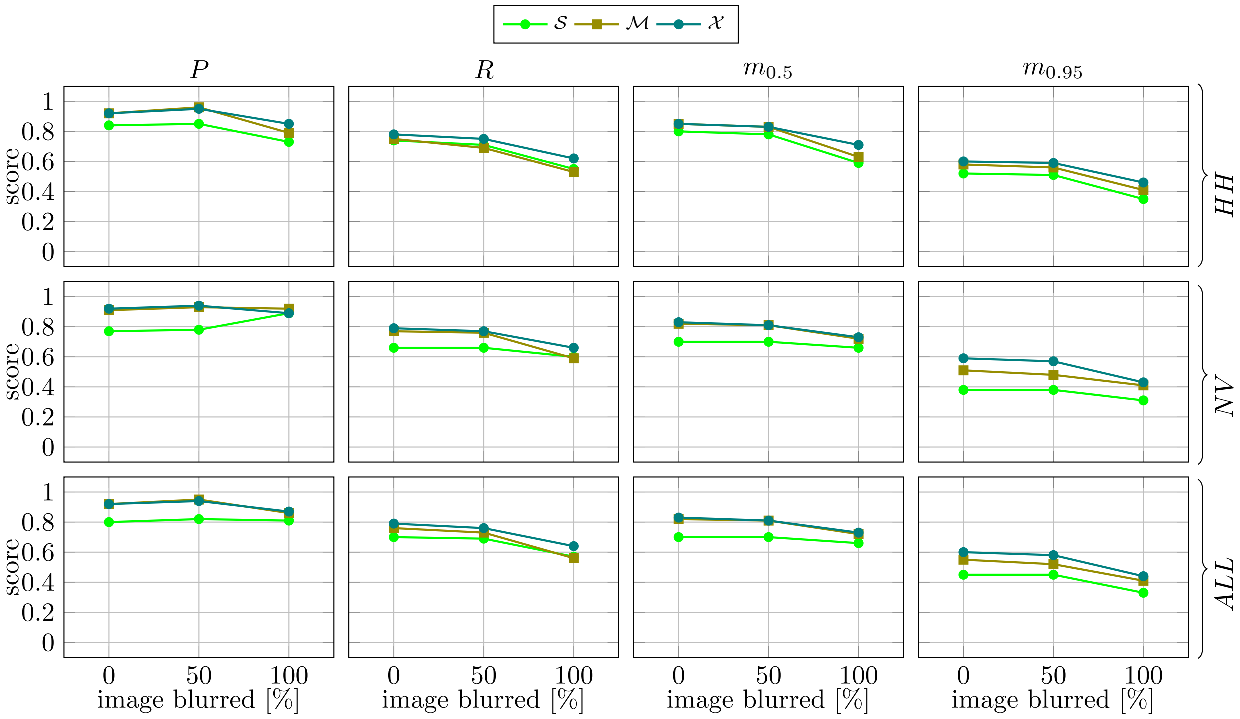


Figure 11: Evaluation of blur impact on YOLO models using RAND split.

Figure 11 presents the performance evaluation metrics of the three YOLO networks, considering different percentages of forced blur in the training set with 0%, 50%, and 100% of blurred images. Each row of plots in Figure 11 corresponds to a different group of results, namely HH, NV, and all the classes, respectively. On the other hand, each column represents the evaluation metrics. In each plot, the amount of blur imposed on the training set is fixed on the x-axis, while the y-axis represents the scores obtained by the models.

In principle, we can observe that as the percentage of blur increases, the performance of the networks tends to decrease for each model. When analyzing the behavior of the networks based on their size, we find that $\mathcal{X}$ achieves the highest results, while $\mathcal{S}$ demonstrates the lowest performance, as previously observed. Regarding the precision, all three lines consistently decline as the percentage of blurred images increases, specifically for the HH class. This pattern is also observed in the other rows of the plot. However, we observe the opposite trend for the $\mathcal{S}$ network and the NV class.

We observe a significant increase of 1% in precision when using a training set with 100% blurring. Similarly, when the models are trained on a set with 50% blurred images, we observe a smoother, but still noticeable, increase in precision for $\mathcal{M}$ and $\mathcal{X}$ models. This can be attributed to a degradation in recall, causing the models to become more selective in detecting stink bugs. Consequently, they only predict the clearest targets, resulting in a reduction in false positives and an improvement in precision. The recall parameter is the evaluation metric most impacted by blur. For all network sizes, we observe a drop of 2% in recall when using a training set with 100% blurring. This behavior is expected since, similar to the human eye, the ability to recognize unclear objects diminishes. When examining the two mean average precision metrics ($m_{0.5}$ and $m_{0.95}$), we observe a consistently decreasing trend, confirming that blurring progressively reduces the recognition capacity of the networks.

In summary, the results suggest that blurring has a detrimental effect on the recognition capacity of the networks, as indicated by the decreasing trend in all precision metrics. The detrimental effect was not evident in the previous experiments because the Worst set was not enough deteriorated with respect to the Best set.

5. Conclusion

In this paper, we have undertaken the study of a system with the ultimate goal of automating the HH pest scouting in orchards by leveraging drones and computer vision algorithms, particularly ML. Our study primarily focused on constructing a suitable dataset of images featuring the HH and enhancing its quality through a preliminary screening process. To capture the images in the field, we carefully selected appropriate hardware, including a vision chip and drone. Subsequently, we proceeded to train and evaluate various ML models based on the YOLO framework, employing different metrics to assess their performance. Additionally, we conducted an in-depth analysis of the captured images, considering factors such as blurriness and brightness, to improve the performance of the ML algorithms. Our results are highly satisfactory and underscore the critical significance of meticulous dataset construction, model training, and image analysis in the successful implementation of ML for HH recognition.

Further research and developments are required to complete a fully autonomous orchard monitoring, which can be extended to other invasive and emergent pests. To progress towards this goal, several key areas need attention. For instance, the development of a client-server application that leverages the bug-detectors described in this work for real-time detection is crucial. This application would enhance the practicality and accessibility of the monitoring system. Additionally, integrating bug-detectors with microclimate weather observations to build a HH prediction model holds immense potential. This integration can provide valuable insights into the pest population dynamics, enabling proactive decision-making and pest management strategies. Continued research efforts are essential to identify novel approaches that can complement the existing methods and contribute to more effective IPMs.

References

[1] K. Jha, A. Doshi, P. Patel, M. Shah, A comprehensive review on automation in agriculture using artificial intelligence, Artificial Intelligence in Agriculture 2 (2019) 1–12.

[2] H. Nagar, R. Sharma, Pest detection on leaf using image processing, in: 2021 International Conference on Computer Communication and Informatics (ICCCI), IEEE, Virtual, 2021, pp. 1–5.

[3] M. Lippi, N. Bonucci, et al., A yolo-based pest detection system for precision agriculture, in: 29th Mediterranean Conference on Control and Automation (MED), IEEE, Virtual, 2021, pp. 342–347.

[4] J. Martinazzo, S. C. Ballen, J. Steffens, C. Steffens, Sensing of pheromones from euschistus heros (f.) stink bugs by nanosensors, Sensors and Actuators Reports 4 (2022) 100071.

[5] A. Kamilaris, F. X. Prenafeta-Boldú, Deep learning in agriculture: A survey, Computers and electronics in agriculture 147 (2018) 70–90.

[6] A. L. Nielsen, G. C. Hamilton, Life history of the invasive species halyomorpha halys (hemiptera: Pentatomidae) in northeastern united states, Annals of the Entomological Society of America 102 (4) (2009) 608–616.

[7] M. Bariselli, R. Bugiani, L. Maistrello, Distribution and damage caused by halyomorpha halys in italy, Eppo Bulletin 46 (2) (2016) 332–334.

[8] HALY.ID, Project, `https://www.haly-id.eu` (2022).

[9] L. Almstedt, et al., Technological innovations in agriculture for scouting halyomorpha halys in orchards, in: 2023 19th International Conference on Distributed Computing in Sensor Systems (DCOSS), 2023, pp. 1–8.

[10] A. Sava, L. Ichim, D. Popescu, Detection of halyomorpha halys using neural networks, in: 2022 8th International Conference on Control, Decision and Information Technologies (CoDIT), Vol. 1, IEEE, 2022, pp. 437–442.

[11] Y. Guo, X. Jia, D. Paull, J. Zhang, A. Farooq, X. Chen, M. N. Islam, A drone-based sensing system to support satellite image analysis for rice farm mapping, in: IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium, 2019, pp. 9376–9379.

[12] D. Murugan, A. Garg, T. Ahmed, D. Singh, Fusion of drone and satellite data for precision agriculture monitoring, in: 2016 11th International Conference on Industrial and Information Systems (ICIIS), 2016, pp. 910–914.

[13] L. Moreno, V. Ramos, M. Pohl, F. Huguet, Comparative study of multispectral satellite images and rgb images taken from drones for vegetation cover estimation, in: 2018 IEEE 38th Central America and Panama Convention (CONCAPAN XXXVIII), 2018, pp. 1–8.

[14] S. Tansuriyavong, H. Koja, M. Kyan, T. Anezaki, The development of wildlife tracking system using mobile phone communication network and drone, in: 2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), Vol. 3, 2018, pp. 351–354.

[15] A. Mitra, B. Bera, A. K. Das, Design and testbed experiments of public blockchain-based security framework for iot-enabled drone-assisted wildlife monitoring, in: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2021, pp. 1–6.

[16] C. Xie, J. Zhang, R. Li, J. Li, P. Hong, J. Xia, P. Chen, Automatic classification for field crop insects via multiple-task sparse representation and multiple-kernel learning, Computers and Electronics in Agriculture 119 (2015) 123–132.

[17] K. Espinoza, D. L. Valera, J. A. Torres, A. López, F. D. Molina-Aiz, Combination of image processing and artificial neural networks as a novel approach for the identification of *Bemisia tabaci* and *Frankliniella occidentalis* on sticky traps in greenhouse agriculture, Computers and Electronics in Agriculture 127 (2016) 495–505.

[18] M. Valan, K. Makonyi, A. Maki, D. Vondráček, F. Ronquist, Automated taxonomic identification of insects with expert-level accuracy using effective feature transfer from convolutional networks, Systematic Biology 68 (6) (2019) 876–895.

[19] C. Wen, D. Guyer, Image-based orchard insect automated identification and classification method, Computers and electronics in agriculture 89 (2012) 110–115.

[20] C.-J. Chen, Y.-Y. Huang, Y.-S. Li, C.-Y. Chang, Y.-M. Huang, An aiot based smart agricultural system for pests detection, IEEE Access 8 (2020) 180750–180761.

[21] Y. He, Z. Zhou, L. Tian, Y. Liu, X. Luo, Brown rice planthopper (*Nilaparvata lugens* (Stål)) detection based on deep learning, Precision Agriculture 21 (6) (2020) 1385–1402.

[22] W. Li, D. Wang, M. Li, Y. Gao, J. Wu, X. Yang, Field detection of tiny pests from sticky trap images using deep learning in agricultural greenhouse, Computers and Electronics in Agriculture 183 (2021) 106048.

[23] Y.-L. Park, J. R. Cho, G.-S. Lee, B. Y. Seo, Detection of *Monema flavescens* (Lepidoptera: Limacodidae) cocoons using small unmanned aircraft system, Journal of Economic Entomology 114 (5) (2021) 1927–1933.

[24] V. Ferrari, R. Calvini, B. Boom, C. Menozzi, A. K. Rangarajan, L. Maistrello, P. Offermans, A. Ulrici, Evaluation of the potential of near infrared hyperspectral imaging for monitoring the invasive brown marmorated stink bug, Chemometrics and Intelligent Laboratory Systems (2023) 104751.

[25] R. Trufelea, M. Dimoiu, L. Ichim, D. Popescu, Detection of harmful insects for orchard using convolutional neural networks, UPB Sci. Bull. Ser. C 83 (4) (2021) 85–96.

[26] DJI, Maryland Biodiversity Dataset, `https://www.marylandbiodiversity.com/` (2022).

[27] L. Ichim, R. Ciciu, D. Popescu, Using drones and deep neural networks to detect halyomorpha halys in ecological orchards, in: IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2022, pp. 437–440.

[28] G. Jocher, et al., ultralytics/yolov5: v6. 1-tensorrt, tensorflow edge tpu and openvino export and inference (2022).

[29] X. Zhou, D. Wang, P. Krähenbühl, Objects as points, arXiv preprint arXiv:1904.07850 (2019).

[30] J. Xu, Y. Pan, X. Pan, S. Hoi, Z. Yi, Z. Xu, Regnet: self-regulated network for image classification, IEEE Transactions on Neural Networks and Learning Systems (2022).

[31] N. D. Narvekar, L. J. Karam, A no-reference image blur metric based on the cumulative probability of blur detection (cpbd), IEEE Transactions on Image Processing 20 (9) (2011) 2678–2683.

[32] C. Ware, Chapter three - lightness, brightness, contrast, and constancy, in: C. Ware (Ed.), Information Visualization (Fourth Edition), fourth edition Edition, Interactive Technologies, Morgan Kaufmann, 2021, pp. 69–94.

[33] G. Jocher, A. Stoken, J. Borovec, L. Changyu, A. Hogan, L. Diaconu, J. Poznanski, L. Yu, P. Rai, R. Ferriday, et al., ultralytics/yolov5: v3. 0, Zenodo (2020).

[34] F. Betti Sorbelli, F. Corò, S. K. Das, E. Di Bella, L. Maistrello, L. Palazzetti, C. M. Pinotti, A drone-based application for scouting halyomorpha halys bugs in orchards with multifunctional nets, in: 2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), IEEE, 2022, pp. 127–129.

[35] F. Betti Sorbelli, F. Corò, S. K. Das, L. Palazzetti, C. M. Pinotti, Drone-based optimal and heuristic orienteering algorithms towards bug detection in orchards, in: 2022 18th International Conference on Distributed Computing in Sensor Systems (DCOSS), 2022, pp. 117–124.

[36] P. Skalski, Make Sense, `https://github.com/SkalskiP/make-sense/` (2019).

[37] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection (2016). `arXiv:1506.02640`.

[38] F. Zhuang, et al., A comprehensive survey on transfer learning, Proceedings of the IEEE 109 (1) (2021) 43–76.

[39] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, International journal of computer vision 88 (2010) 303–338.

[40] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft COCO: common objects in context, CoRR abs/1405.0312 (2014). arXiv:1405.0312.

[41] T.-Y. Lin, et al., Microsoft coco: Common objects in context, in: European conference on computer vision, Springer, Zurich, Switzerland, 2014, pp. 740–755.

[42] L. Bottou, Large-scale machine learning with stochastic gradient descent, in: 19th Intl. Conference on Computational Statistics, COMPSTAT, Physica-Verlag, Paris, 2010, pp. 177–186.

[43] Opencv: Smoothing images, https://docs.opencv.org/4.x/d4/d13/tutorial_py_filtering.html, (Accessed on 05/13/2023) (2023).