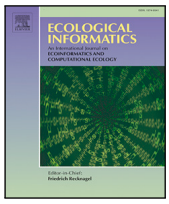
Ecological Informatics 91 (2025) 103383

Contents lists available at ScienceDirect

Ecological Informatics

journal homepage: www.elsevier.com/locate/ecolinf



AntPi: A Raspberry Pi based edge–cloud system for real-time ant species detection using YOLO☆

Lorenzo Palazzetti [a,*,1], Daniele Giannetti [b,1], Antonio Verolino [b], Donato A. Grasso [b,2], Cristina M. Pinotti [a,2], Francesco Betti Sorbelli [a,2]

[a] Department of Computer Science and Mathematics, University of Perugia, Italy
[b] Department of Chemistry, Life Sciences and Environmental Sustainability, University of Parma, Italy

ARTICLE INFO

Dataset link: https://github.com/bettisfr/ant-cloud, https://github.com/TheAnswer96/ant-exp-ei, https://doi.org/10.5281/zenodo.16740459

Keywords:
IoT
Ant detection
Computer vision
Technological transfer
Edge computing

ABSTRACT

Ant detection is essential for ecological research, offering insights into biodiversity, habitat health, and environmental change. Traditional detection techniques rely on manual sampling methods, which are labor-intensive and time-consuming. Recent advances in autonomous, vision based systems show promise for insect monitoring, yet no dedicated, field ready solution exists for ant identification. In this work, we present AntPi, a deep learning based system for real-time detection and classification on a Linux development board. To the best of our knowledge, the system is trained on the first dedicated dataset for arboricolous ants, comprising five species and one morphotype, sourced from citizen science contributions and direct field captures. Our approach employs the "You Only Look Once" (YOLO) framework for efficient object detection, augmented with environmental sensors to enable correlation between climatic variables and ant activity. To evaluate performance and robustness, we compare AntPi with an alternative configuration, including controlled experiments using background-only images with artificial ant-like noise, and introduce a novel robustness indicator to assess reliability under realistic conditions. Experimental results demonstrate strong detection performance and confirm the feasibility of automated, in-field ant monitoring.

1. Introduction

Ants (Hymenoptera, Formicidae) are the largest group of eusocial organisms, with over 14,000 species worldwide (Wang et al., 2024b). Their ubiquity and ecological interactions make them keystone organisms, shaping arthropod, plant, and soil communities in both natural and anthropogenic ecosystems (Parker and Kronauer, 2021). Their interactions with plants range from mutualism and protection from herbivores, to antagonism. Moreover, ants significantly influence soil structure, aeration, and chemistry, acting as true ecosystem engineers (Lach et al., 2009; Rico-Gray and Oliveira, 2009; Grasso et al., 2015). In agroecosystems, ants provide both ecological services and disservices, shaping multitrophic networks through complex (co)evolutionary pathways (Schifani et al., 2024). Few studies have addressed their role in Italian agroecosystems, but native Mediterranean ant species could be effective against pests such as the brown marmorated stink bug (*Halyomorpha halys*), fungi, and other herbivores (Castracani et al., 2017; Giannetti et al., 2019; Schifani et al., 2020).

The detection of insect populations is essential for habitat management in natural and agroecosystems (McCravy, 2018; Gao et al., 2024). Traditional methods, such as pitfall traps, baits, and visual sampling, are reliable but labor-intensive (Montgomery et al., 2021). To reduce human effort, autonomous detection systems using imaging technology have been explored, particularly for pest and disease detection (Palazzetti et al., 2024; Crupi et al., 2025; Ahmad et al., 2023; Ali et al., 2024). However, in-field deployment of computer vision algorithms remains limited due to challenges in algorithmic robustness,

☆ This work was supported by the PRIN 2022 PNRR M4.C2.1.3 "BREADCRUMBS" project, funded by the European Union – Next Generation EU, grant number P2022K7ERB, CUP J53D23014990001. F. Betti Sorbelli, L. Palazzetti, and M.C. Pinotti are member of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM). This work has benefited from the equipment and framework of the COMP-HUB and COMP-R Initiatives, funded by the "Departments of Excellence" program of the Italian Ministry for University and Research (MIUR, 2018-2022 and MUR, 2023-2027).

* Corresponding author.
E-mail addresses: lorenzo.palazzetti@unipg.it (L. Palazzetti), daniele.giannetti@unipr.it (D. Giannetti), antonio.verolino@unipr.it (A. Verolino), donatoantonio.grasso@unipr.it (D.A. Grasso), cristina.pinotti@unipg.it (C.M. Pinotti), francesco.bettisorbelli@unipg.it (F. Betti Sorbelli).
[1] Co-first authors.
[2] Co-last authors.

https://doi.org/10.1016/j.ecoinf.2025.103383
Received 16 April 2025; Received in revised form 6 August 2025; Accepted 6 August 2025
Available online 18 August 2025

hardware constraints, and environmental adaptability (Tabani et al., 2021; Keasar et al., 2024).

For ants, their small size and complex backgrounds make automated detection particularly difficult, confining research largely to high-level detection (Das et al., 2024). Initial studies have used deep learning (DL) to estimate ant abundance via nest detection in aerial imagery (Alhawsawi et al., 2024; Monsimet et al., 2024; Dos Santos et al., 2022), but these methods lack species classification, affecting biological indicators. Vision based approaches in controlled environments have tracked ant behavior using machine learning (ML) (Hu et al., 2024; Cordonnier et al., 2020; Ramalingam et al., 2020), yet their real-world applicability remains limited. In contrast, automated detection of other insect species has progressed, leveraging convolutional neural networks (Palazzetti et al., 2024; Crupi et al., 2025; Betti Sorbelli et al., 2023; Kargar et al., 2024; Bjerge et al., 2024). However, existing solutions face challenges such as high costs, limited power autonomy, and narrow focus on single specimens, limiting broader ecological applications (Saradopoulos et al., 2022). Developing scalable, multi-species detection methodologies remains an open research challenge, requiring innovations in vision algorithms and edge computing (Bairi and Dulhare, 2024; Gardiner et al., 2024; Kumar et al., 2024).

This paper explores the feasibility of ant identification using DL models on edge devices. We adopt the "You Only Look Once" (YOLO) methodology, known for its real-time accuracy (Betti Sorbelli et al., 2023; Almstedt et al., 2023, 2025b,a), and introduce the first dedicated dataset for ant identification, covering five arboricolous species and one morphotype. Sourced from field captures and supplemented by citizen science platforms (iNaturalist, 2025), this dataset ensures diverse environmental representation for robust model training and evaluation. We deploy the models on a Raspberry Pi 5, AntPi, benchmarking efficiency metrics such as energy consumption and inference time across different YOLO versions and model sizes. Then, to provide insights into computational feasibility as well as energy consumption, we set up an alternative version equipped with a Raspberry Pi 3B (RPi3B). AntPi is further equipped with sensors to correlate climatic data with collected imagery, enabling autonomous, scalable detection. Our long-term goal is to generate a continuous time series of ant and climate data for population tracking. Additionally, AntPi includes a cloud based service with APIs for data offloading and a gallery website, providing non-expert users with intuitive access to collected data. To assess model robustness before field deployment, we introduce a novel evaluation methodology using synthetic noise testing sets and propose the first robustness indicator to quantify resistance to false positives across confidence thresholds. Experimental results demonstrate strong detection performance and confirm the feasibility of automated, in-field ant monitoring.

Contributions and paper organization

This work presents the following key contributions:

- We developed the first dedicated dataset for ant detection, encompassing five arboricolous ant species and one morphotype.
- Leveraging this dataset, we trained and evaluated multiple YOLO variants (i.e., v9, v10, and v11) to assess their performance in terms of precision, recall, and accuracy.
- We deployed the trained models on an RPi5, analyzing key performance metrics such as RAM usage and CPU load. The device is further integrated with environmental sensors to associate climatic conditions (i.e., temperature and humidity) with collected imagery, enabling long-term ecological detection. For the sake of comparison, we conducted a comparison between AntPi and an alternative configuration made up with an RPi3B to extract insight into the energy consumption and the computational feasibility.
- To assess system robustness, we conducted a series of controlled experiments using background-only images augmented with artificial ant-like noise, and introduced a novel robustness indicator to evaluate model reliability under real-world conditions.

The paper is structured as follows: Relevant related work is surveyed in Section 2. Section 3 introduces the custom dataset developed for this study, while Section 4 describes the hardware and software architecture of AntPi. Then, Section 5 presents experimental results on detection accuracy and robustness to visual noise. Section 6 focuses on the evaluation of energy consumption. Finally, Section 7 discusses the system's limitations and future directions, and Section 8 concludes the paper.

2. Related work

In this section, we provide an overview of existing methods for detecting ants. Section 2.1 reviews automated approaches based on computer vision and DL, including works focused on other insect species to provide broader context. Then, Section 2.2 examines traditional field techniques based on manual observation and baiting. While automated systems offer scalability and reduced human effort, traditional methods remain prevalent due to their simplicity and standardization. By reviewing both approaches, we contextualize our contribution and highlight the current limitations in the field.

2.1. Smart detection systems

Nowadays, autonomous detection systems leverage advanced sensing and DL techniques to facilitate insect detection and behavioral analysis while minimizing human intervention (Gouse and Dulhare, 2022; Alhawsawi et al., 2024; Khan et al., 2024; Almstedt et al., 2025a,b). Although the effectiveness of such systems has been widely demonstrated across several species, ants have received relatively little attention (Das et al., 2024). In the following, we review relevant studies on ants detection.

Dos Santos et al. (2022) employed DL for remote detection of leaf-cutting ant nests using drone imagery. A comparative analysis of traditional ML methods against YOLO demonstrates the superior performance of convolutional neural network based approaches. The authors highlighted the advantages of aerial detection for large-scale habitat assessment, enabling non-intrusive and efficient nest localization, which can aid in ecological management and pest control strategies.

Similarly, Monsimet et al. (2024) explored the integration of Unmanned Aerial Vehicles (UAVs) imagery and DL techniques to detect and map ant mounds in sub-Arctic ecosystems. By utilizing RGB, thermal, and multispectral imaging, the study evaluated detection performance alongside ecological indicators such as Normalized Difference Vegetation Index (NDVI). Their benchmarks highlighted YOLO as an effective model, particularly with RGB and thermal imagery, while also revealing the quantifiable impact of ant mounds on vegetation productivity, influencing up to 4% of the surveyed area. The research underscored the role of ants as ecosystem engineers and the potential of remote sensing in studying their ecological influence.

Hu et al. (2024) proposed a system integrating consumer-grade camera equipment with specialized software to enable continuous video based monitoring of ant foraging behavior. This approach mitigated human-centered limitations by supporting video post-processing tasks such as annotation and counting analysis through YOLO based algorithms. The system is designed to facilitate long-term ecological studies, allowing researchers to analyze colony interactions, food transport patterns, and activity cycles without the need for constant human presence.

A DL based detection and tracking methodology for ants is introduced by Wu et al. (2020), applicable to both indoor and outdoor environments. The system employed a two-stage object detection methodology with a ResNet backbone and a Kalman filter for instance

tracking. Experimental results on high-resolution videos indicated robust performance across diverse settings. The study also addressed the challenge of tracking multiple ants in cluttered backgrounds, a common issue in behavioral studies, by optimizing object association and trajectory prediction.

Ramalingam et al. (2020) presented a detection system based on DL techniques, specifically utilizing Faster Region based Convolutional Neural Networks with ResNet50 for feature extraction. Their model outperformed other object detection methods such as Single Shot Detector (SSD) and YOLO, demonstrating higher accuracy in detecting ants along with other small organisms like cockroaches, whiteflies, and lizards. The study emphasized the potential of cross-species detection systems in integrated pest management and biodiversity assessments.

While these studies provided valuable insights, they differ significantly from our contribution. Unlike our AntPi system that constituted a complete hardware–software solution for in-field, species-level ant detection, previous works primarily focused on nest-level detection, generic ant presence without taxonomic resolution, or relied on high-resolution imaging setups unsuitable for low-power edge devices. Our contribution lay in both the development of a lightweight, field-deployable platform and the implementation of an efficient, species-aware detection pipeline optimized for embedded execution. In addition, we conducted dedicated robustness experiments by introducing controlled visual noise into ant-free background images to simulate real-world environmental conditions. To the best of our knowledge, none of the reviewed studies explicitly assessed model robustness under such conditions, which we considered critical for reliable deployment in natural habitats. Although limited research exists on ant detection systems, several studies have proposed automated insect detection approaches. Unlike our AntPi, these works differ significantly in terms of design, complexity, and energy-awareness.

Freitas et al. (2022) proposed a detection system for fruit flies built around an RPi3. The device included a white sticky panel to trap flies and a camera aimed at the panel for image capture. Detection was performed using a simple blob segmentation algorithm that exploited the high contrast between the flies and the white background. The authors evaluated the performance of four different classification architectures, focusing on accuracy and inference latency. However, their study did not include any analysis of energy consumption or operational longevity. Additionally, the visual detection task in their setup was significantly simplified by the uniform white background, which reduces noise and false detections. Despite this simplification, the performance metrics of their classifiers were comparable to those achieved by AntPi, which operates under more complex and variable environmental conditions.

Bjerge et al. (2023) introduced an extensive annotated dataset comprising almost 30,000 insect occurrences from nine taxonomic groups, collected through time-lapse photography in natural floral environments. Leveraging this dataset, they implemented a detection pipeline using YOLOv5 and deployed cameras in locations with direct access to electrical power. The study also included a counting mechanism to estimate insect abundance over time. Although the authors emphasized the energy efficiency of their system, no quantitative evaluation of energy consumption or lifetime analysis was provided. In contrast, AntPi is designed to function in energy-constrained field environments, with detailed evaluations of power usage and operational sustainability.

In a subsequent study, Bjerge et al. (2025) explored a more sophisticated DL pipeline for the real-time monitoring of nocturnal insects and arachnids. Their system was split across multiple devices: image capture was performed using an RPi4, while detection and tracking were offloaded to a more powerful NVIDIA Jetson Nano. The three-stage pipeline — comprising detection, classification, and tracking — demonstrated strong performance but suffered from reduced frame rates (down to 0.5 FPS) due to the Jetson Nano's high energy consumption. This energy demand significantly constrained the system's deployment viability in long-term or remote field conditions. In contrast, AntPi supports both detection and classification locally on a single low-power device, ensuring extended operational lifetime without compromising performance.

Sittinger et al. (2024) developed the insect detect Do It Yourself (DIY) camera trap, which was assembled using relative low-cost (≈€700), off-the-shelf components, including an RPi Zero 2, an AI camera module, and solar power integration. Their setup was specifically tailored for monitoring flower-visiting insects and incorporated visual attractants in the form of printed colored flower patterns. While this approach effectively increased insect visibility against the background, it based its effectiveness to a less representative natural environmental complexity. Although they achieved detection accuracy comparable to AntPi, their system lacked support for cloud based data management or offloading, which limits its scalability and automation potential. In contrast, AntPi integrates cloud services for data storage, analysis, and remote monitoring, offering a more comprehensive and scalable solution.

Gebauer et al. (2024) introduced a stereo vision based insect monitoring approach focused on detecting and tracking small, fast-moving insects in cluttered, natural environments. Their method employed an asynchronous algorithm that relied on contrast thresholds and specific shape recognition to identify targets. While the system marked a significant advancement in using dynamic vision sensors, its evaluation was confined to controlled laboratory settings. Moreover, the study did not address practical deployment concerns such as power efficiency, device autonomy, or environmental robustness. By contrast, AntPi has been validated under real-world conditions and includes extensive energy profiling, making it more applicable for long-term ecological monitoring.

Lastly, Biswas and Tiwari (2024) presented a handheld camera trap designed to attract and detect moths using specialized lighting. The hardware configuration included an RPi and a camera module, with image data processed offline via a custom neural network. While their system demonstrated promising detection capabilities, the lack of onboard inference and absence of energy performance assessments limited its applicability for real-time, autonomous operation. AntPi, in contrast, performs all processing directly on the device in real time, ensuring low-latency responses and reduced communications needs, while also incorporating thorough energy analysis to support continuous field deployment.

In summary, while prior work has advanced insect detection technologies in various directions, AntPi distinguishes itself by combining real-time, onboard processing with energy-efficient design and cloud integration. This makes it particularly well-suited for autonomous, long-term monitoring of ants and other small insects in uncontrolled environments.

2.2. Traditional detection systems

Manual detection approaches, frequently based on baiting and direct observation, are the most widely used for assessing ant abundance and behavior, though their massive reliance on human operators. The majority of research in this field is related to evaluation of baits under diverse environmental conditions. For example, Rahardjo et al. (2023) evaluated the effectiveness of three bait types (i.e., tuna, eggs, and sugar) for measuring ant abundance and diversity in Indonesian sugarcane plantations. From a total of 6,650 collected individuals spanning 20 morphospecies, the study found that species richness remains consistent across bait types, while abundance varies significantly. The findings provided important implications for pest control strategies and biodiversity assessments in agricultural landscapes, where ant populations can serve as bioindicators of ecosystem health.

Similarly, Nyamukondiwa and Addison (2014) investigated ant foraging preferences using different bait types in vineyard environments. Their findings indicated a general preference for wet baits over dry ones, with further analyses extending to species such as *Crematogaster*

peringueyi and *Linepithema humile* under varying bait conditions. This research contributed to a better understanding of resource selection in ants and highlights the importance of bait composition in ecological detection and invasive species control.

Gaigher et al. (2012) examined the role of bait quantity in limiting exposure to non-target organisms. The proposed bait station method scaled according to target species while maintaining cost efficiency, demonstrating effectiveness against invasive species like the big-headed ant, *Pheidole megacephala*. Their findings emphasized the need for targeted baiting strategies to minimize ecological disruption while effectively managing invasive ant populations.

García-Martínez et al. (2018) assessed the potential of the commercial fruit fly food attractant CeraTrap as an attractant for foraging ants in agroecosystem canopies. Five trap models were tested, yielding insights into inventory completeness, ant activity levels, and species occurrence per trap. The study showed the importance of lure based traps in detection ant community in complex habitats such as orchards and plantations.

Warren et al. (2023) analyzed the effects of warming and habitat fragmentation on native ant foraging behavior. By employing tuna baits at controlled intervals, the research manually evaluated species behavior and abundance at bait stations. The findings contributed to the growing body of evidence on climate change impacts on insect communities and emphasized the role of standardized baiting protocols in long-term ecological studies.

Lastly, Song et al. (2024) investigated the impact of chemical pesticides on red imported fire ants and surrounding ant communities. Their manual sampling approach revealed that control strategies not only manage red fire ant populations but also aid in the ecological recovery of local species. The study underscored the need for sustainable pest management practices that balance agricultural productivity with biodiversity conservation.

While these methods provided valuable ecological insights, they heavily relied on human observations, leading to potential biases in estimation analysis. Moreover, the effectiveness of manual sampling was influenced by user preparation and attention. As a consequence, poor methodological decisions can introduce additional biases, potentially implying negative effects on in-field practices. Differently, our AntPi can be deployed in multiple locations guaranteeing a constant attention level regardless of the work time. Moreover, DL models allow us to extend the range of species with reasonable effort and in a short time or more in general objects to be monitored by a fine-tuning of a couple of hours.

3. The dataset

In this section, we introduce the dataset built in Section 3.1, which describes six different ant classes, and then analyze it in Section 3.2.

3.1. Collected images

The constructed dataset to develop an autonomous system for detecting ants consists of 1,253 images collected in the field using smartphones and cameras, supplemented by materials gathered from and georeferenced photographs uploaded on citizen science platforms (iNaturalist, 2025). We selected five species of arboricolous ants, *Crematogaster scutellaris* (CS) (Olivier, 1792), *Dolichoderus quadripunctatus* (DQ) (Linneus, 1771), *Camponotus vagus* (CV) (Scopoli, 1763), *Colobopsis truncata* (CT) (Spinola, 1808), *Plagiolepis pygmaea* (PP) (Latreille, 1798) and morphotype of *Temnothorax spp.* (TS), as shown in Fig. 1. To the best of our knowledge, the constructed dataset represents the first dataset for ants detection.

It is worth pointing out that the dataset annotation was manually conducted by an expert operator relying on the cloud service MakeSense.ai (Skalski, 2019). The annotation consists of the minimal rectangle that includes the captured ants. Finally, annotations were encoded utilizing the YOLO format. In this format, each image in the dataset should have a corresponding text file with the same name as the image, containing the bounding box annotations for that image.

Although they may appear quite similar at first glance, each species is distinguished by several unique traits. CS ants (Fig. 1(a)) exhibit a heart-shaped gaster raised above the mesosoma, with a smooth cuticle and vivid coloration, often combining red and black hues. DQ ants (Fig. 1(b)) are identified by their shiny dark bodies, slightly flattened mesosoma, and two pairs of dorsal abdominal spots. The slightly arched propodeum adds to their distinctiveness. CV (Fig. 1(c)) stands out as one of the largest ants, characterized by a black, robust body with a matte to slightly glossy cuticle. Their massive heads, long mandibles, and strong mesosoma emphasize their powerful build. CT (Fig. 1(d)) is medium-sized and notable for their truncated, flattened heads. The elongated, smooth, and shiny body surface provides a distinct separation from related species. PP ants (Fig. 1(e)) are small, with slender bodies and a smooth, shiny cuticle. Their large heads, prominent eyes, and narrow petioles contribute to their delicate appearance. TS ants (Fig. 1(f)) are compact, displaying a robust mesosoma and an oval head. Their segmented antennae, featuring a distinct three-segmented club, and their rounded propodeum with a uniform body texture distinguish them from other species.

The selection of the six ant species used in this study is justified by a combination of ecological importance, morphological distinctiveness, and practical detectability. Due to their ubiquity in most terrestrial habitats and the countless ecological interactions they establish, ants are considered keystone organisms. They have profound effects on the arthropods, plants, and soil of the ecological communities they inhabit, in both natural and anthropogenic ecosystems. Most of them are generalist predators of arthropods, but they also establish mutualistic relationships with several arthropod species. Likewise, their complex relationships with plants may range from mutualism—for instance, when ants kill or displace organisms that would damage the plants (either insects or pathogens). In particular, the selected species, which occur in various environments including agroecosystems and urban areas, serve as study models capable of providing diverse ecosystem services and playing a significant role in plant defense.

Fig. 2 illustrates the distribution of ants within the images of the dataset according to their respective classes. Specifically, Fig. 2 (left) shows the number of images containing at least one occurrence from any of the dataset classes, while Fig. 2 (right) shows the number of occurrences actually labeled for each class.

Looking at Fig. 2 (left), we observe that the 1,253 collected images are fairly evenly distributed across the classes, with approximately 208 images per class. However, when examining the number of occurrences in Fig. 2 (right), the distribution appears more imbalanced. With a total of 2,873 ants labeled across all classes, the PP and CS classes account for approximately 68% of all occurrences, with 757 and 922 occurrences, respectively. This uneven distribution is primarily due to the methods used to collect the images. Images captured in the field using mobile devices tend to feature smaller occurrences with higher density, whereas images gathered from online sources are predominantly macro shots of the insects, typically containing only one or two ants per image. As a result, the two classes with the most occurrences reflect the higher number of images manually captured in the wild.

3.2. Analysis

Given that computer vision techniques for object detection are data-driven, it is essential to study and, therefore, visualize the dataset we have created. Moreover, this analysis is critical due to the heterogeneous nature of the collected data, as previously emphasized. Specifically, as we observed, while the number of images per class is fairly uniform, the number of occurrences per class significantly varies. Therefore, we decided to further characterize the data distribution behavior behind our dataset.

Fig. 1. Comparison between different species of ants: (a) *Crematogaster scutellaris* (CS), (b) *Dolichoderus quadripunctatus* (DQ), (c) *Camponotus vagus* (CV), (d) *Colobopsis truncata* (CT), (e) *Plagiolepis pygmaea* (PP), and (f) *Temnothorax spp.* (TS).

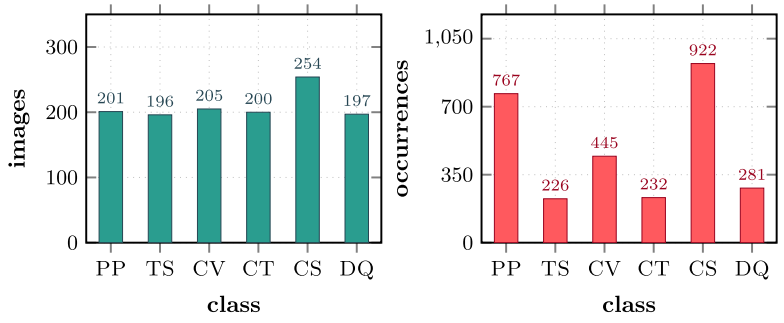


Fig. 2. Ant classes distribution.

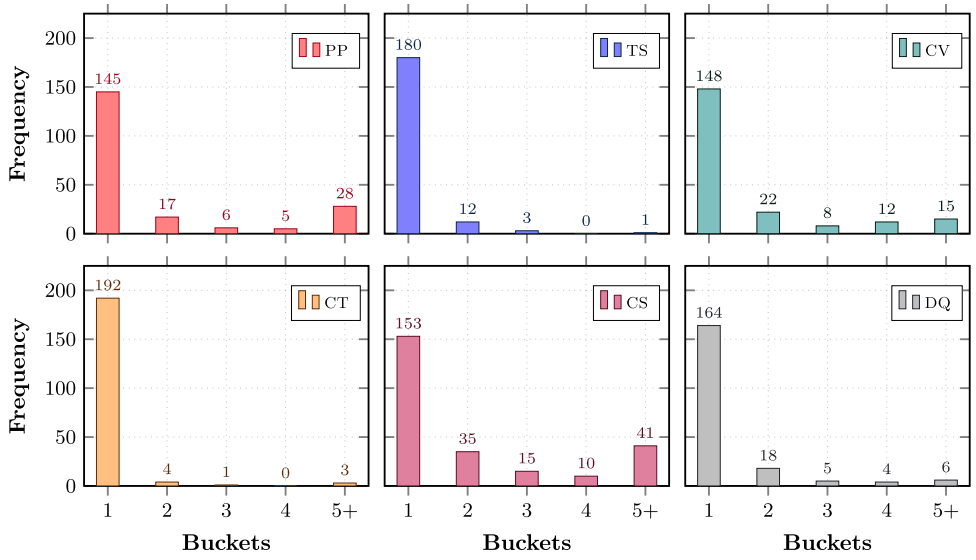


Fig. 3. Number of occurrences per image distribution.

Fig. 3 illustrates, for each class, the number of occurrences contained within each image. Specifically, the images have been grouped into five distinct buckets: the four most frequent occurrence ranges, plus an additional class for the outlier cases (those with five occurrences or more). As observed in Fig. 3, in each class, the majority of images overwhelmingly contain a single ant, with a non-increasing trend for the other occurrences. However, in ant species with the highest number of occurrences (PP and CS), the "5+" category stands out, showing an increase compared to the previous categories, although it significantly remains lower than the "1" category.

While slightly less pronounced than in the previous two ant species, the same trend is also observed for CV ants. This behavior is not surprising, given that the CV class ranks third in terms of the number of individual occurrences, or in other words, images captured in the wild.

Since the task we are about to tackle is object detection, important features include the position and size of the bounding boxes drawn around the objects. Indeed, the distribution underlying each class could provide valuable information for the predictions made by our models. Therefore, in Fig. 4, we plot the positions of the bounding boxes, specifically the x-axis and y-axis coordinates of the center of each bounding box, expressed as percentages relative to the size of the image containing them. Similarly, in Fig. 5, the sizes of the bounding boxes are shown, with the x-axis and y-axis representing the horizontal and vertical side dimension of the boxes, respectively.

Starting with the position data (Fig. 4), it is evident that, in general, the bounding boxes surrounding the ants predominantly fall near the center of their respective images, with an increasing dispersion of occurrences as they move away from the center. This phenomenon is particularly pronounced in the classes with a higher prevalence of macro images, where there is typically one or, at most, two ants captured with very high resolution detail.

We observe this trend in which the majority of ants are centered in the images for TS, CT, and DQ. Conversely, in the remaining ant classes (PP, CV, and CS), the distribution of bounding boxes is much more

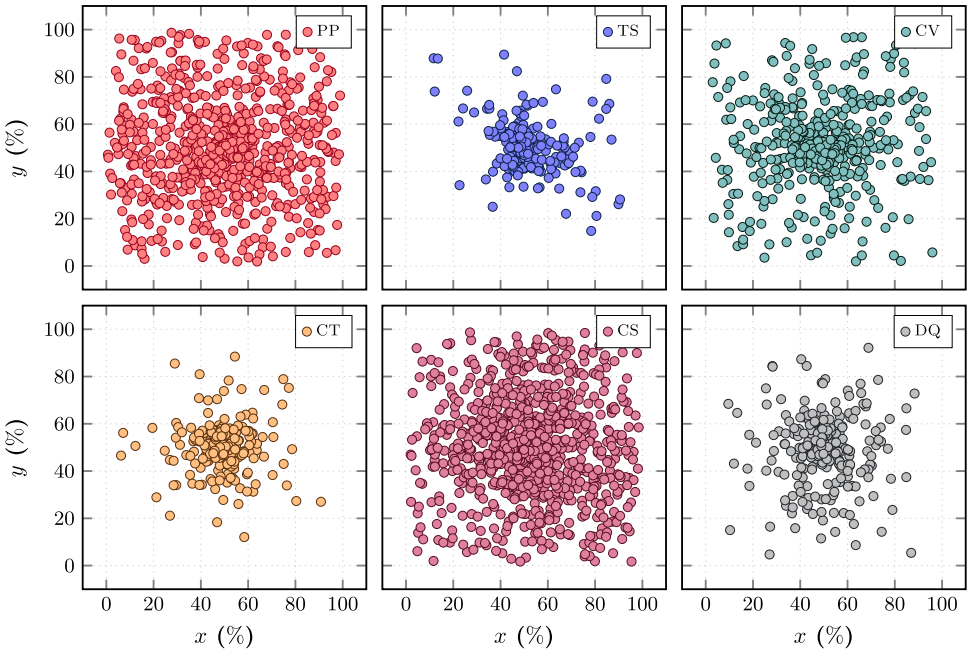


Fig. 4. Bounding box positions distribution.

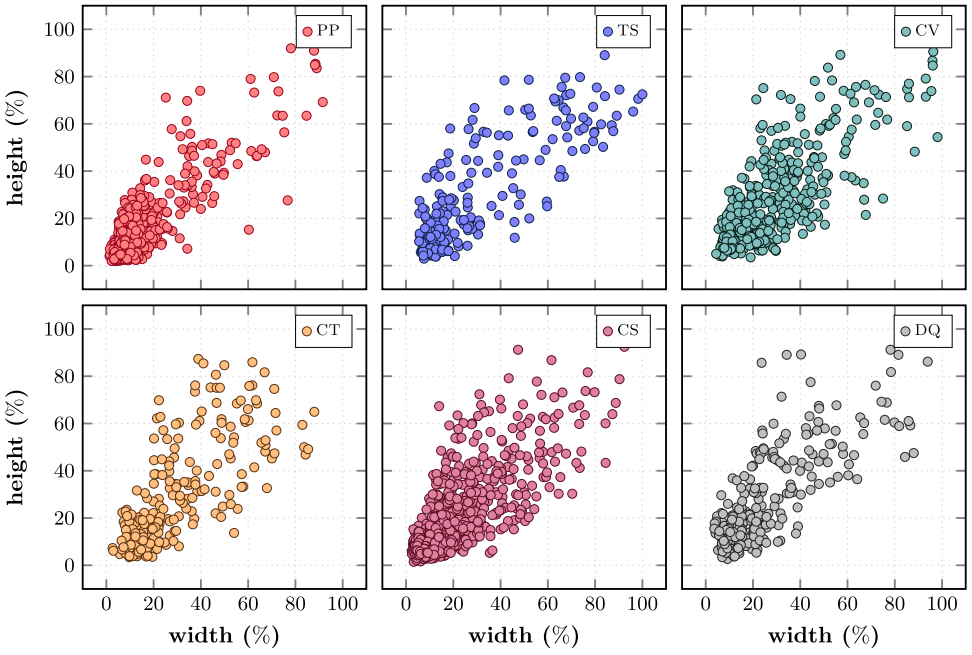


Fig. 5. Bounding box sizes distribution.

scattered, covering almost the entire surface of the plot. This occurs because, in these classes, there is a larger presence of field-captured images, where the position of the ant is not controlled. Therefore, while the photographer attempted to center the ants in the frame, some occurrences still ended up in off-center positions within the image.

Regarding the size of the bounding boxes (Fig. 5), a common trend can be observed across all the plots, with a noticeable difference in the classes dominated by field images. In principle, it can be seen that the majority of ants photographed occupy less than 20% of the image containing them, with a tendency for the bounding boxes to be square-shaped. This latter detail is evident from the tendency of the points to align along the diagonal of the quadrant, which, in other words, suggests that the two sides of the bounding boxes are of approximately equal length.

In more detail, similarly to the position analysis (Fig. 4), the classes with a prevalence of macro images tend to have fewer points near the origin of the axes, resulting in greater dispersion. This suggests that images from the TS, CT, and DQ classes tend to have larger bounding boxes on average, with a lower frequency of square-shaped boxes. In contrast, the remaining classes, characterized by a significant presence of field images, tend to have smaller bounding boxes on average, with shapes that are more frequently square.

Therefore, it can be observed that the dataset we constructed presents a fairly standardized set of data related to six different ant

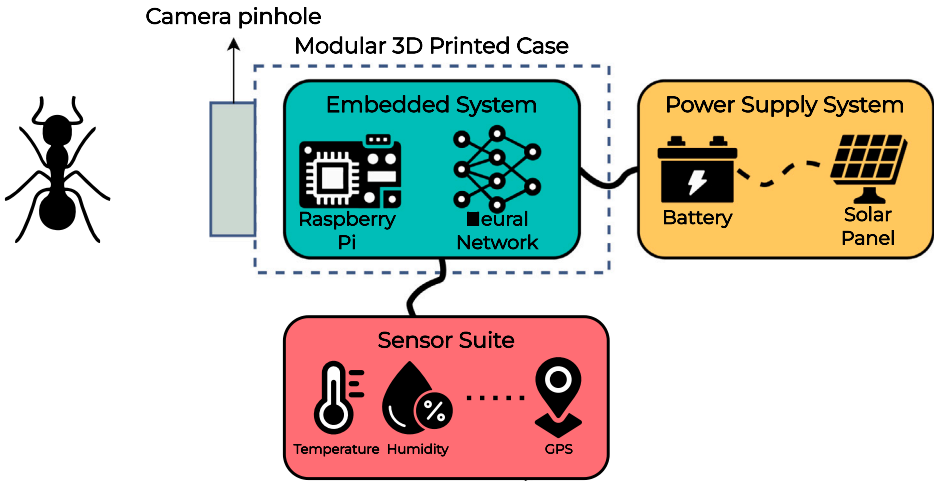


Fig. 6. Hardware system scheme.

classes. Although the presence of occurrences per species is imbalanced due to varying percentages of images captured in the field, this behavior is consistently reflected in the shape, size, and position of the objects to be detected. This suggests that our computer vision models will need to operate on a relatively homogeneous dataset compared to the diverse range of devices and sources used for data collection, as well as the broad diversity of ant species considered.

4. System implementation

In this section, we describe the hardware and software architecture of the proposed intelligent detection system AntPi that is a low-cost, modular, and field-deployable platform for ant detection, built around RPi single-board computers. The AntPi system operates on the RPi5 board, which offers sufficient computing power for on-device inference using state-of-the-art object detection models (e.g., YOLO). In addition, we set up a second variant, based on the less performant RPi3B. The latter configuration is intended exclusively as a laboratory setting for benchmarking, energy profiling, and evaluating lightweight tasks.

Both configurations share the same peripheral components—camera, environmental sensors, power supply, and protective enclosure. However, only the RPi5 based system is intended for deployment in the field, while the RPi3B version is used solely for experimental comparisons under controlled conditions.

The remainder of this section is organized as follows: Section 4.1 describes the hardware platform, Section 4.2 outlines the software and cloud architecture, and Section 4.3 analyzes the theoretical energy consumption of both configurations.

4.1. Platform

In this section, we describe the low-cost detection system we developed for deployment directly in the field to monitor the aforementioned ant species. Let us now present the hardware components of AntPi, summarized in Fig. 6.

In principle, AntPi is designed to capture images, process them (i.e., execute DL model on captured images), annotate them with relevant metadata, such as temperature, humidity, and GPS location, perform on-the-edge computations by leveraging lightweight ML models, and ultimately transmit results and data to an external server (cloud based) in real time. AntPi comprises three main hardware components: the **embedded system board**, the **power supply system**, and the **sensor suite** used to annotate the collected images, as highlighted in Fig. 6.

Since accurate computer vision algorithms for multiclass object recognition requires substantial computational resources, we have chosen to equip our system with an RPi, which represents an optimal trade-off between performance, cost, and energy consumption. As aforementioned, in AntPi we decided to rely on the RPi5, the latest release in the RPi series. However, for the sake of comparison, we also configured a less recent RPi3B, a significantly less performant earlier model, as a baseline for our experiments. In detail, although the RPi3B and RPi5 are both popular choices for embedded systems, they significantly vary in performance, connectivity, and expansion capabilities, as reported in Table 1. The RPi3B, released in 2016, features a 1.2 GHz quad-core CPU and 1 GB of RAM. Due to its lower power consumption and cost, it remains a viable choice for energy-efficient applications. On the other hand, RPi5 launched in 2023, offers a 2.4 GHz quad-core CPU and 8 GB RAM. These upgrades make the RPi5 ideal for AI, ML, and high-performance embedded applications. Consequently, RPi3B is best suited for low-power IoT and automation, while RPi5 provides the processing power and connectivity needed for advanced real-time computing tasks.

The second component of AntPi concerns the power supply. Since the detection process will take place in the field, the entire system must be capable of operating without access to a continuous power source. Therefore, we chose to equip the system with a power bank that is in turn connected to a solar panel. The decision to rely on a power bank stems from the fact that commercially available models offer a compact form factor, substantial energy capacity, and affordability. Specifically, we opted for a standard and common power bank, typically used for charging smartphones, with capacity of 20 Ah (Aukey Powerbank). However, to make AntPi fully modular and interchangeable, the power supply component is designed to be externally to the 3D-printed case that hosts the RPi, to facilitate maintenance operations and upgrades. In other words, the powerbank will be enclosed in another water proof box along with the USB cables to/from the solar panel and the RPi.

As the final third component of AntPi, the sensor suit, we installed the main camera sensor as well as many environmental sensors to monitor parameters around the detection system. For visual detection, we included the official RPi Camera Board v3, featuring a 12 MP sensor capable of capturing 1,080 p video (i.e., $1,920 \times 1,080$ px), allowing for image based analysis and recording. Fig. 7 showcases sample ant images collected using the AntPi system in various natural conditions. Concerning the other deployed environmental sensors, we included a DHT11 (DHT-sensor-library, 2025) for temperature and humidity measurements, which provides digital output with basic accuracy and operates using a single GPIO pin for communication. Additionally, we integrated a GT-U7 GPS (Open-smart-GT-7U-GPS, 2025) module, based on the Ublox NEO-6M chip, which provides real-time location tracking

Table 1
Comparison of RPi3B and RPi5.

	RPi3B	RPi5
Processor	Quad-core ARM Cortex-A53 (1.2 GHz)	Quad-core ARM Cortex-A76 (2.4 GHz)
RAM	1GB LPDDR2	8GB LPDDR4X
GPU	VideoCore IV	VideoCore VII
Storage	microSD (standard speed)	microSD (high-speed), PCIe for SSD
USB ports	4× USB 2.0	2× USB 3.0, 2× USB 2.0
Networking	Wi-Fi 4 n, Bluetooth 4.2, 100 Mbps Ethernet	Wi-Fi 5 ac, Bluetooth 5.0, Gigabit Ethernet
GPIO pins	40-pin header	40-pin header (same layout, faster I/O)
Display output	1× HDMI (Full HD)	2× micro-HDMI (4k)
PCIe support	No	Yes (PCIe 2.0 for high-speed peripherals)
Power input	Micro-USB (5 V, 2.5 A)	USB-C (5 V, 5 A)
Idle power	1.4 W	3.2 W
Stress power	9 W	15 W
Use cases	IoT, basic automation, low-power apps	AI, ML, high-speed computing

Fig. 7. Sample ant images collected using the AntPi system in various natural conditions.

through NMEA data. The GPS module requires a UART interface for communication, ensuring reliable data transmission for positioning and navigation. Overall, the goal is to annotate the captured and processed images with the number of ant occurrences, along with environmental data that enriches the immediate analysis of the surroundings. Over time, this stored data could form a valuable database for understanding ant behavior.

4.2. Architecture

In the RPi, we connected all the aforementioned sensors through different available ports, namely the serial port, the GPIO pins, and the UART port. To handle all these sensors, we mounted a Proto HAT Shield and soldered all the required wires and resistors to provide a 5 V supply to all the sensors. Inside the RPi, we developed a Python script that runs as soon as the device is turned on and continuously performs tasks in a loop. The Python script uses various libraries to interact with the sensors, such as `gpiozero`, `picamera2`, `serial`, `adafruit_dht`, and `board`. Let us now detail the architecture of AntPi, illustrated in Fig. 8.

The GPS module retrieves latitude and longitude coordinates in NMEA format, which require conversion to decimal degrees. In contrast, the temperature and humidity values are retrieved directly in the appropriate format, namely degrees Celsius and percentage, respectively. The script periodically captures an image and records all relevant information, including GPS coordinates and environmental data (i.e., temperature and humidity). Once the photo is taken, we embed the GPS location and weather data into the image by modifying its metadata using the `piexif` library. This allows each picture to be geotagged, which is essential for creating a diverse dataset of images collected from different locations and climates.

The captured images are then analyzed to detect the presence of ants using various YOLO models of different versions and sizes. This process will be thoroughly described from Section 5. The ant recognition phase of AntPi is performed directly on the RPi board to evaluate edge computing performance.

Whenever an image is captured, enriched with metadata, and analyzed by the YOLO models, it is sent to the server, as shown in Fig. 8. The server hosts the back-end component for data analysis and storage, and provides an HTTP based interface (APIs) for clients to visualize data and statistics via a front-end. The back-end, implemented using the `Flask` library, listens for HTTP requests from the RPi. When the RPi sends an image, the server verifies the request format and stores the image in a local folder. The front-end is a dynamic HTML page that does not rely on continuous polling. Instead, it listens for server-triggered events using the `eventlet` library. When an image is received, an event is triggered, allowing the front-end to update the gallery immediately, improving efficiency by avoiding unnecessary polling.

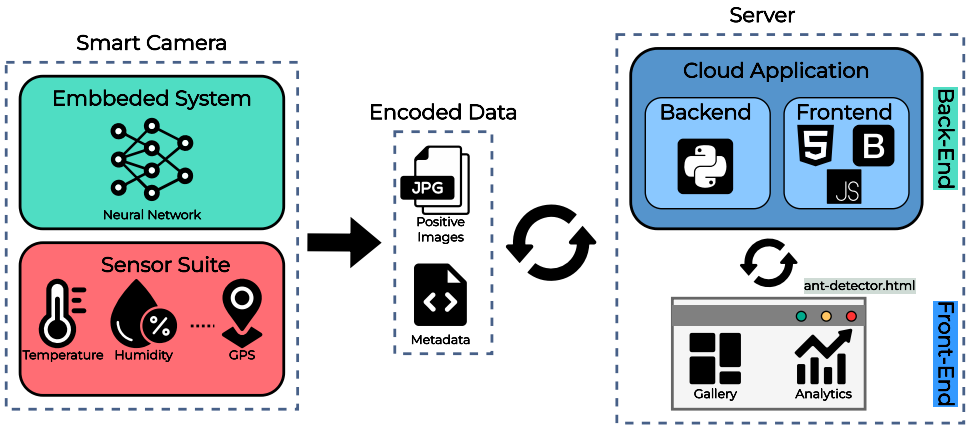


Fig. 8. System architecture scheme.

To conclude, Fig. 9 presents the physical prototype of AntPi, highlighting its main components: the custom-designed 3D-printed enclosure shown from two different angles in (a) and (b), and the complete system assembly in (c), which includes a RPi5, solar panel, and power bank. The enclosure is waterproof and robust, ensuring reliable outdoor operation under diverse environmental conditions.

4.3. Energy consumption analysis

Although the detection system is powered by a power bank, it cannot sustain operation indefinitely. In the case of intensive computations, this will require frequent intervention by a human operator. To address this issue, we equip the system with a solar panel capable of mitigating the risk of depletion during periods of high energy consumption. The system will primarily operate during daylight hours to take advantage of solar energy, which will enhance its ability to monitor ant activity. During the night, the system will remain in a constrained idle state to minimize power depletion. Therefore, since the peak energy output of the solar panel corresponds to the system's maximum workload, selecting an efficient panel could significantly reduce the need for manual intervention to recharge the power bank.

In detail, we opted for the IDOR STORE solar panel (IDOR STORE, 2025) able to deliver 10–12 W in optimal conditions. It is also worth noting that by switching off peripheral, limiting both the CPU and RAM, we can reduce by 1 W the idle state power consumption reported in Table 1 for both boards, obtaining so ≈ 0.4 W and ≈ 2.2 W for RPi3B and RPi5, respectively.

Given these specifications, the system's time-to-live (TTL) can be estimated by first calculating the energy of the power bank based on its capacity and nominal voltage, using the following formula:

$$E = \frac{C \cdot V}{1{,}000}, \tag{1}$$

where E is the energy (Wh), C the capacity (mAh), and V the nominal voltage (V). The TTL of the system depends on the power required to operate it, which can be derived using the formula:

$$\text{TTL} = \frac{E \cdot \eta}{P_d}, \tag{2}$$

where η represents the efficiency factor of the system, assumed to be 0.9, and P_d is the power (W).

By equipping the system with a panel, some of the energy lost can be replenished. If the solar panel's power output (P_s) exceeds the RPi's power requirement, such that $P_s \geq P_d$, the board can theoretically remain active indefinitely, accounting for minor approximation errors. In general, the system's net power drain (P_m) can be expressed as the difference between the board's power consumption and the solar panel's power output:

$$P_m = \max(0, P_d - P_s). \tag{3}$$

Substituting the relevant values, the energy of the power bank is calculated as:

$$\frac{20{,}000\,\text{mAh} \cdot 5\,\text{V}}{1{,}000} = 100\,\text{Wh}. \tag{4}$$

Thus, based on the power requirements specified by the manufacturer (as reported in Table 1), the system can remain in an idle state for approximately 28 h with the RPi5, and 64 h with the RPi3B. Moreover, under average stress-mode consumption, the expected runtime is approximately 6 h for the RPi5 and 10 h for the RPi3B, according to the manufacturer's specifications. Since the solar panel delivers more power than the power of both RPi, under sunlight there is no loss of energy theoretically speaking. It is worth to point out that we experienced extreme stress conditions (see Table 1) for a few milliseconds during the execution of the prediction. However, since these peaks are short, it is reasonable to consider RPi5 fully powered by the selected solar panel on typical usage. Moreover, we observe that in a forced reduced-energy idle state, the two boards can remain operational for approximately 41 h (RPi5) and 225 h (RPi3B), respectively.

Table 2 summarizes the estimated RPi's life times under the stress conditions considered above.

5. Assessment results

In this section, we provide a comprehensive evaluation of the computer vision models. We begin by introducing the training settings and key hyperparameters, along with their validation process (Section 5.1). We then analyze the models' detection performance at both average and class-specific levels (Section 5.2), followed by an assessment on a preliminary in-field image collection (Section 5.3). To further examine robustness, we introduce the notion of visual noise and report detection results under various noise conditions (Section 5.4).

5.1. Models training

For our experiments, we selected models from the YOLO family due to their efficiency and real-time capabilities compared to two-stage detector counterparts. Specifically, we employed YOLOv9 (Wang et al., 2024c), YOLOv10 (Wang et al., 2024a), and YOLOv11 (Khanam and Hussain, 2024), which offer a trade-offs between accuracy and efficiency.

Given our focus on deploying an in-field detection system for ants with limited energy requirements, we selected the three lightest versions of each model. As shown in Table 3, the medium variants contain significant numbers of parameters, impacting memory footprint. While the smallest variants are most suitable for edge computing, we extended our experiments to larger models to evaluate adaptability to resource-constrained environments and assess the relationship between model size and energy consumption.

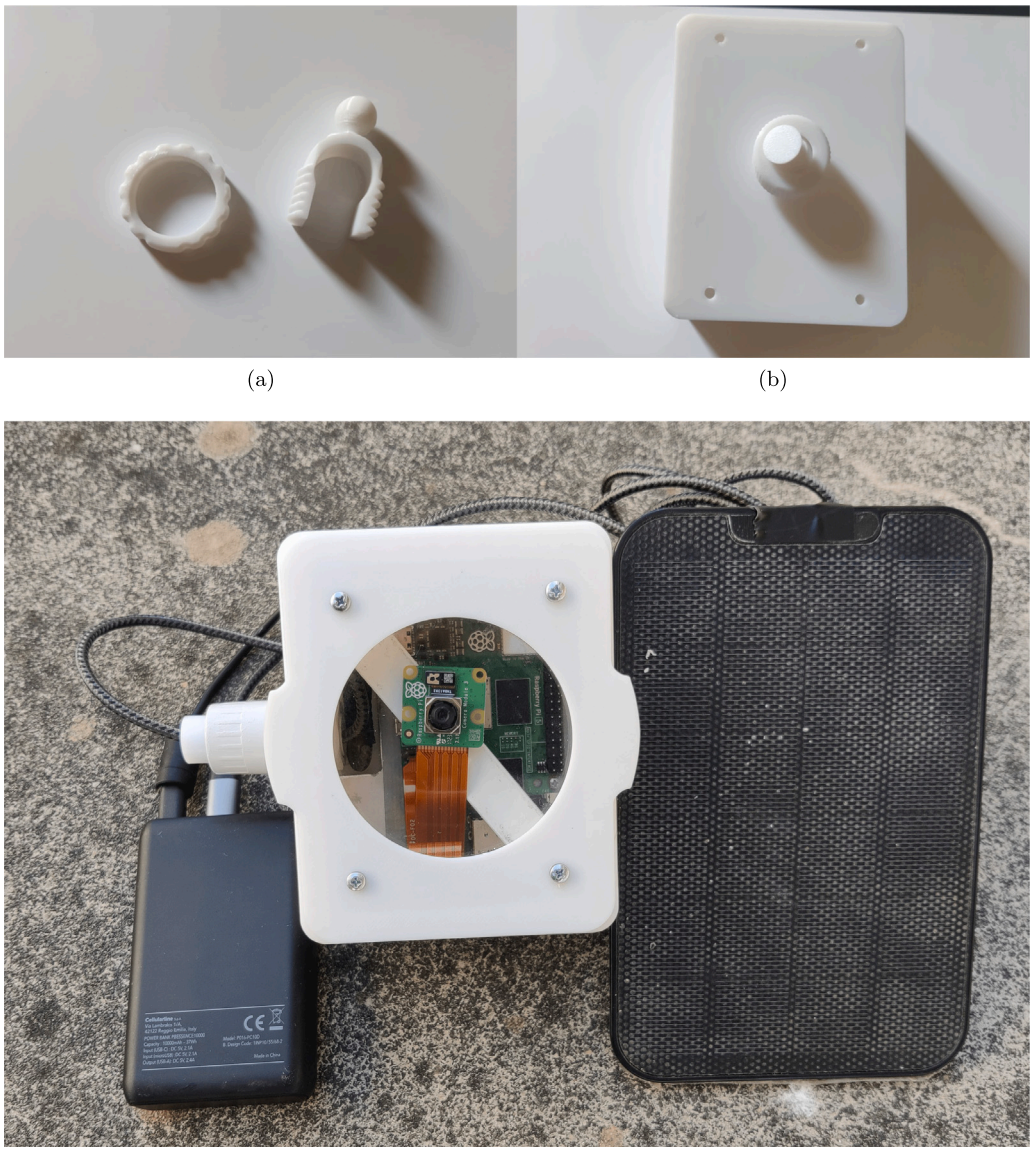


Fig. 9. Pictures of the AntPi components: (a), and (b) show the 3D-printed case sub-components from different perspectives; and (c) shows the whole hardware system along with the solar panel and the powerbank.

Table 2
Estimated lifetime of RPi models under different stress states.

Model	Idle (h)	Stress (h)	Deep Idle (h)	Optimal Solar (h)
Raspberry RPi3B	64.29	10.00	225.00	∞
Raspberry RPi5	28.13	6.00	40.91	∞

Table 3
Computational footprint of different YOLO model variants.

Version	Variant	GFLOPs	Parameters (M)	Size (MB)
YOLOv11	m	68.0	20.1	76.60
	s	21.5	9.4	36.10
	n	6.5	2.6	10.00
YOLOv10	m	59.1	15.4	58.70
	s	21.6	7.2	27.70
	n	6.7	2.3	8.88
YOLOv9	m	76.8	20.1	76.60
	s	26.7	7.2	27.60
	t	7.7	2.0	7.81

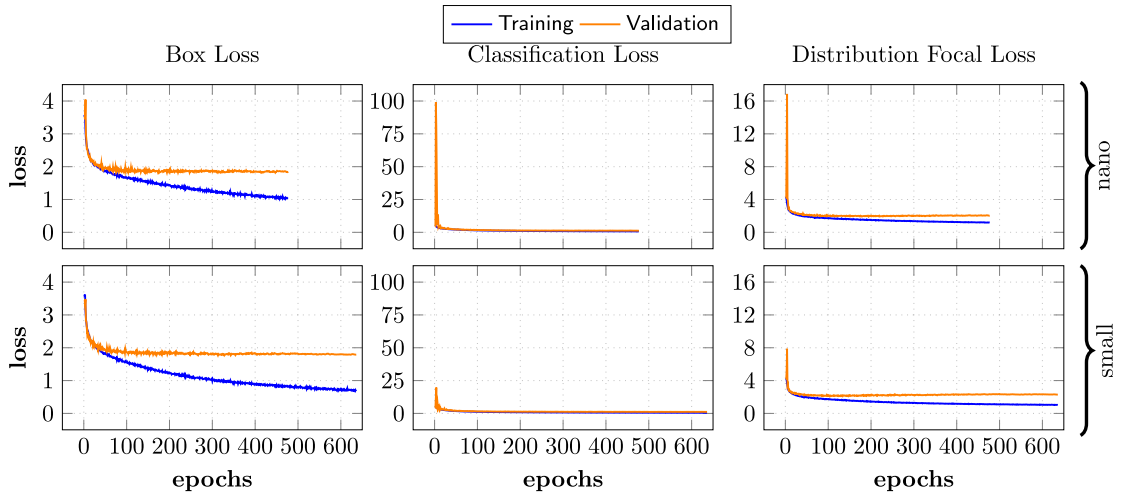


Fig. 10. v11n and v11s training performance loss functions.

For training our models we relied on 1,235 images split 70%, 20%, and 10% into training, validation and test set, respectively. For each model, we adopted the Transfer Learning (TL) paradigm: the pre-trained weights are obtained from intense training over the MS-COCO dataset (Lin et al., 2014). Moreover, we opted for augmentation strategies in order to limit overfitting phenomena and enriching training set examples. Specifically, we performed horizontal flips (0.5 probability), scaling (0.5), translation (0.1), and random erasing (0.4), with mosaic augmentation enabled throughout and closed during the final 10 epochs. The number of classes has been overridden to match the number of actual classes, i.e., 6. Therefore, for each training sample we created three new additional samples generated by the combination of the listed effects. As a result, the training set size increases from 877 to $877 + 2{,}631 = 3{,}508$ samples. We configured 1,000 epochs, a batch size of 16, and a squared image size of 640 pixels per side. Learning rate started at 0.01 and decayed to 0.01 of its initial value (`lrf` = 0.01) with a cosine schedule, using Stochastic Gradient Descent optimizer (Bottou, 2010) with 0.937 momentum and 0.0005 weight decay. We relied on an NVIDIA RTX 3060 with 12 GB of VRAM for the training procedure. Notice that, we configured the early stopping for the training process.

We assess hyperparameter selection by examining the behavior of v11n and v11s, which serve as representative models. It is worth pointing out that the hyperparameter selection was driven by trial and error following commonly shared best practices (e.g., reuse already optimized hyperparameters) and keeping the configuration that returned the best empirical results. Their behavior can be attributed to the trends observed in other YOLO releases. Note that we discarded the v11m variant, as every version showed stronger convergence. Fig. 10 illustrates the training and validation loss curves for v11n (first row) and v11s (second row) models, respectively. We relied on three loss functions as objective metrics to evaluate model performance during the training:

- **Box loss:** Measures the discrepancy between predicted and ground truth bounding boxes, using an Intersection over Union (IoU) based metric.
- **Classification loss:** Determines the accuracy of predicted class labels by employing a cross-entropy loss function.
- **Distribution focal loss:** A refinement of focal loss that enhances the prediction of class distributions, particularly beneficial for handling class imbalance.

Notice that although we set 1,000 epochs, due to early-stopping both the training models ended up before the last epoch. Specifically, v11n and v11s stopped at epoch 477 and 636, respectively.

Considering the box loss curves, they exhibit a decreasing trend for both models, indicating effective learning of object localization. However, v11s demonstrates a steeper convergence and a lower final loss value compared to the v11n variant. This suggests a superior bounding box prediction accuracy in the v11s model that can be attributed to its increased number of parameters. Both the validation curves showcased a stabilization after a few epochs ($\approx$ 100) with minor improvements up to the end. Conversely, training curves continue to decrease with a stable trend suggesting a potential overfitting behavior for prolonged training phases.

Classification loss follows a similar downward trend in both the models, confirming improved class prediction over training epochs. The v11n model exhibits larger starting fluctuation in validation loss, whereas the v11s model maintains a more stable validation loss for the entire training phase. In principle, both the models converge fast close to 0, namely the optimal value. The focal loss is roughly speaking a classification loss weighted according to infrequent classes, it showcased for both the models a behavior extremely similar to classification loss. Although the two loss differ in their formula, their behavior coincides due to a balance distribution among the six classes.

In the next, we present the performance of our models during training on validation set. Before describing the results, we briefly recap the key performance metrics used to evaluate the parameters selection. Specifically, for our assessment, we employed four key metrics: Precision (P), Recall (R), mean average precision at IoU 0.5 (mAP50), and mean average precision across multiple IoU thresholds (mAP50-95). Precision measures the proportion of correctly identified objects among all the predictions returned by the model. It is defined as $\frac{TP}{TP+FP}$, where TP (True Positives) represents correctly detected objects, and FP (False Positives) denotes wrong detections. Recall score quantifies the ability to identify all relevant objects and is computed as $\frac{TP}{TP+FN}$, where FN (False Negatives) are missed detections. The mAP50 metric represents the average precision (AP) computed at an IoU threshold of 0.5, while mAP50-95 averages mAP across multiple IoU thresholds from 0.5 to 0.95 in increments of 0.05, providing a more stringent evaluation of localization accuracy.

Once again, v11n and v11s are used as representative benchmark. The x-axis of each plot represents the number of epochs, while the y-axis quantifies the respective performance metric. Results are shown in Fig. 11.

Starting from the Precision score, both the models depict an initial fast rise, indicative of improved prediction confidence in training progresses. The Precision curve of v11s model is higher than v11n one, suggesting that the v11s variant achieves better accuracy in positive predictions. After $\approx$ 100 epochs, the curves tend to stabilize, indicating

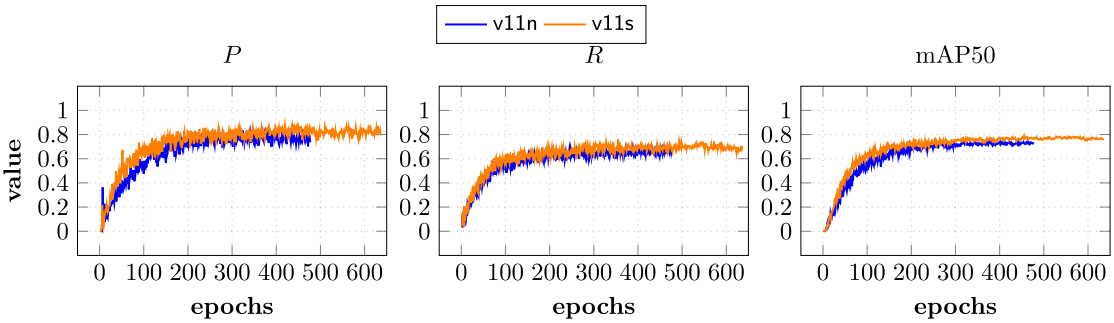


Fig. 11. v11n and v11s performance metrics on validation set.

that the models reach a saturation point in learning precision-related features. Similarly, the Recall curves show a steady increase during early epochs, reflecting enhanced model sensitivity in detecting ants. The v11s model again maintains a higher metric value. Overall, we observe complemental fluctuations in Precision and Recall curves, suggesting that the models are attempting to balance the two indicators, as optimizing one often leads to trade-offs with the other. The same behavior is showcased for mAP50 where both the curves reveal a rapid improvement during initial training phase, after which both models approach a plateau. The fast saturation of the curves suggests that further gains in performance are possible but they may require architectural adjustments or alternative training strategies.

The conducted analysis highlights convergence trends for both loss functions and metrics, suggesting that the hyperparameter selection was successful. Additionally, the training dynamics reveal superior performance indicators for larger models, as anticipated before the training.

5.2. Object detection performance

As earlier anticipated, we evaluated the model performance using a test set comprising 10% of the entire dataset. To ensure a fair assessment, we stratified the sampling to include 10% of images from each of the six classes within the dataset. Performance evaluation incorporated standard metrics computed both as class-averaged values and as class-specific indicators. Table 4 depicts a comprehensive summary of these results, with columns representing the evaluated models and rows indicating metrics. The tabular data is organized hierarchically, with the first group of rows displaying class-averaged performance followed by class-specific metrics for each ant category.

The evaluated models demonstrate robust performance, with Precision and Recall metrics consistently achieving approximately 80% and 75%, respectively. All models attained substantial mAP50 values exceeding 80%, indicating high confidence in the returned predictions. Additionally, the stability of mAP50-95 metrics around 45% reflects favorable IoU levels across the predictions.

Focusing on YOLO models across different versions reveals distinct performance trade-offs. v9 generally outperforms its successors in terms of mAP50 and mAP50-95, indicating improved localization capabilities. Specifically, v9t achieves the largest mAP50 value, i.e., 88.5%, while v9m obtains the largest mAP50-95, i.e., 47.3%. This behavior depicts an increasing detection IoU and confidence trend with larger architecture of ninth series. v11 exhibits a strong balance between precision and recall, with its smallest model (v11n) achieving the largest precision value, i.e., 87.3% but at the cost of a lower though acceptable recall of 74.9%. In contrast, v10 struggles to maintain consistency, with its models demonstrating relatively lower recall and mAP scores.

On a per-class basis, v11s and v11n excel in precision for the PP class (90.5% and 95.6%, respectively), as previously observed, while v9m dominates recall with a score of 80.6%. For the TS class, both v9m and v11n achieve optimal precision, whereas v9t attains the highest recall at 100%. The CV class is also best handled by v9t, which achieves a strong 94.0% in mAP50 and 50.4% in mAP50-95. Detection of the CT class favors v11s in terms of mAP50-95, while v9m stands out with a recall of 90.5%. For the CS class, detection remains consistently strong across all models, with the v11 and v9 series outperforming v10 overall. Finally, the DQ class remains the most challenging to detect, as indicated by relatively lower performance across models, though v9t provides solid mAP50 and mAP50-95 scores. This performance drop may be due to an insufficiently diverse data distribution, leading the model to develop a strong understanding of frequent occurrences but a weaker ability to detect outliers.

Summarizing, v9 emerges as the most reliable series for high-accuracy detection, particularly in terms of recall and overall mAP, whereas v11 showcased high precision scores along with variability in the rest of the indicators. Despite v10 is majorly outperformed, it demonstrates stable and solid detection performance not to far with respect to the other series.

5.3. Preliminary Results from AntPi

We now present an evaluation of the trained models on a preliminary dataset composed of images collected in real-world field conditions. Specifically, we gathered 58 images using AntPi, each containing at least one of the six target ant species. To complement this limited sample, we acquired additional 71 images using a smartphone camera with resolution comparable to that of the AntPi. These supplementary images were captured under similar environmental conditions and with consistent camera orientation relative to the ants, to ensure visual similarity with the AntPi's content.

As a result, the final evaluation dataset comprises 129 images, closely matching the size of the original test set (126 images). We preserved almost the same 10% split ratio used previously to enable a fair comparison between the idealized testing conditions described in Section 5.2 and this in-field evaluation. Due to the early-stage nature of the current AntPi release, not all target species were captured. This experiment is therefore intended as an initial demonstration of the methodology's viability under real deployment conditions.

To mitigate class imbalance, which could skew interpretation, we report in Table 5 only the average performance across all classes.

At a high level, model performance under field conditions remains largely consistent with the results from Table 4, despite the increased variability introduced by real-world image acquisition. Notably, precision improved by approximately 8%, while recall dropped by nearly 45%. This divergence is largely attributable to the presence of challenging visual conditions in the field, such as motion blur and background noise, which were largely absent in the controlled test set. Consequently, while the models continued to predict in-focus occurrences accurately (leading to few false positives), they struggled with noisy occurrences, resulting in a marked increase in false negatives.

The mAP50 and mAP50-95 metrics also experienced a deterioration, mirroring the drop in recall. This indicates reduced prediction

Table 4
Performance Metrics.

Metric	v11m	v11s	v11n	v10m	v10s	v10n	v9m	v9s	v9t
All Classes									
Precision	81.7	85.9	**87.3**	81.5	79.7	76.9	83.1	81.5	82.1
Recall	83.3	75.4	74.9	73.9	75.7	71.8	**86.4**	79.7	85.3
mAP50	85.4	85.2	84.1	81.7	82.8	80.6	87.5	85.4	**88.5**
mAP50-95	44.4	46.8	43.9	43.6	44.3	42.5	**47.3**	45.3	47.0
***Crematogaster scutellaris* (CS)**									
Precision	91.9	86.3	**94.9**	88.6	91.0	80.5	86.0	85.9	89.3
Recall	90.7	84.9	85.9	84.9	84.9	88.4	**91.9**	88.4	89.5
mAP50	92.1	90.5	93.1	91.1	91.4	91.7	**93.2**	91.0	92.0
mAP50-95	**54.4**	53.8	53.7	50.2	53.9	50.7	53.7	52.6	52.6
***Dolichoderus quadripunctatus* (DQ)**									
Precision	70.1	75.8	**77.7**	63.0	75.5	67.2	76.3	75.2	71.3
Recall	60.9	65.2	52.2	52.2	67.0	65.2	**73.9**	65.2	69.6
mAP50	73.7	72.4	60.9	65.3	77.1	68.8	74.3	76.5	**80.5**
mAP50-95	37.6	37.9	26.4	33.6	38.4	33.5	40.1	34.9	**41.7**
***Camponotus vagus* (CV)**									
Precision	90.4	**98.2**	90.5	93.9	86.5	93.9	89.0	91.9	86.9
Recall	85.2	68.3	70.6	75.8	79.3	57.0	**90.1**	84.0	89.8
mAP50	90.6	88.8	85.4	89.5	89.9	81.8	92.3	89.9	**94.0**
mAP50-95	44.3	44.6	43.9	47.3	45.3	40.9	44.4	46.6	**50.4**
***Colobopsis truncata* (CT)**									
Precision	70.3	76.6	65.1	61.6	64.6	62.7	65.1	62.5	77.0
Recall	**90.5**	78.0	**90.5**	61.9	78.2	71.4	**90.5**	71.5	85.7
mAP50	78.0	84.2	84.6	66.8	79.5	81.3	**85.2**	72.6	80.4
mAP50-95	45.3	**57.3**	51.7	39.4	46.6	48.9	53.3	45.3	45.1
***Plagiolepis pygmaea* (PP)**									
Precision	81.9	90.5	**95.6**	89.4	83.3	78.8	82.6	87.4	81.7
Recall	76.3	71.4	69.4	72.5	75.3	71.9	**80.6**	73.1	77.4
mAP50	81.7	83.4	85.0	81.0	83.9	78.6	83.7	84.3	**86.7**
mAP50-95	41.0	41.8	42.5	43.3	45.1	41.0	45.2	44.2	**46.7**
***Temnothorax spp.* (TS)**									
Precision	85.9	87.7	**100.0**	92.6	77.0	78.5	**100.0**	86.0	86.3
Recall	96.2	84.6	80.7	96.2	69.2	76.9	91.6	96.2	**100.0**
mAP50	96.6	92.2	95.7	96.8	75.0	81.5	96.4	**97.9**	97.3
mAP50-95	43.8	45.2	45.1	**48.0**	36.4	40.3	47.3	**48.0**	45.6

Table 5
Preliminary Performance Metrics on In-Field Data.

Metric	v11m	v11s	v11n	v10m	v10s	v10n	v9m	v9s	v9t
All Classes									
Precision	88.1	85.7	88.9	94.9	**96.3**	90.8	94.0	92.2	89.0
Recall	43.5	42.4	42.4	43.5	**45.9**	34.7	**45.9**	41.8	42.9
mAP50	42.8	42.4	44.6	44.4	44.2	34.9	**45.3**	43.6	43.7
mAP50-95	18.4	19.3	21.3	21.6	20.5	16.8	**22.1**	17.8	17.1

confidence and lower IoU scores, further highlighting the importance of high-quality image capture. However, fine-tuning the models on this type of degraded imagery could help recover performance levels observed in the original testing setup.

From a model-specific perspective, v9m demonstrated the best detection performance, identifying the largest number of true positives. The v10 series, across all sizes, appeared more resilient to real-world image conditions than both v9 and v11, with the latter series showing the largest degradation in recall.

5.4. Object detection robustness assessment

In this subsection, we investigate how robust is our proposed model to images with noises.

While the performance metrics in Section 5.2 demonstrate the models' suitability for ant detection, a comprehensive evaluation must consider real-world deployment challenges. Field implementations will frequently encounter background-only frames which are subject to variable lighting and environmental conditions that may introduce significant noise, potentially compromising detection accuracy and population estimates. Indeed, a critical factor in practical applications is the robustness against false positives. To assess such robustness, we generated a dataset which comprises different degrees of standardized noise (Fig. 12). We captured 400 high-resolution images of natural ant habitats (leaves, flowers, tree trunks, and ground surfaces) using a Canon EOS6D with a 100 mm Macro lens. These images were then digitally modified with standardized shapes of increasing ant-like morphology, organized into four distinct complexity categories/levels:

- level **(a)** Background-only images (Fig. 12(a));
- level **(b)** Images with randomly placed black dots simulating shadow or leaf burns (Fig. 12(b));
- level **(c)** Images containing tapered, bean-shaped black forms (Fig. 12(c));
- level **(d)** Images featuring complete ant-like silhouettes (Fig. 12(d)).

The selection of noise levels has been guided by practical challenges in insect detection. First, background-only (level **a**) frames are introduced because the majority of captured images will not contain any target objects. Next, the black dot (level **b**) and the black tapered shape (level **c**) are included to evaluate robustness against vegetation defects (e.g., burns, diseases) and lighting effects, such as shadows, respectively. Finally, although the ant silhouette (level **d**) closely resembles an actual ant, this noise level is introduced to assess the generalization capability of the models. While the previous noise levels are expected to produce only a limited number of false positives, level **d** is likely to generate significantly more due to its strong resemblance to an ant.

Each noise category consists of 100 frames. Since none of these frames contain real ants, we use the number of false positive predictions generated by each model as a performance indicator. Clearly, the fewer the false positives, the more robust the model.

(a) Background-only frame (b) Black dots simulating shadow or leaf burns

(c) Tapered, bean-shaped black forms (d) Complete ant-like silhouette

Fig. 12. Examples of noisy frames across four complexity levels.

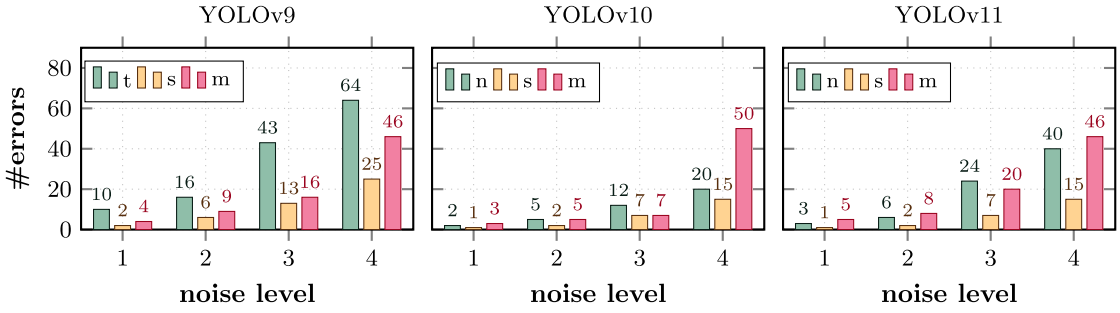


Fig. 13. Models' robustness summary.

Notice that, the Non-Maximum Suppression (NMS) mechanism is applied as in the general experimental evaluation (Section 5.2), and the confidence threshold is fixed at 0.5.

Fig. 13 reports the number of false positive detections for all the YOLO series considered so far across the previously mentioned noise conditions. The x-axis represents the anomaly level, while the y-axis quantifies the number of errors per 100 images. Each model is plotted separately to better appreciate the progressive increase in false positives as the noise complexity rises. Fig. 13 depicts notable trends in model robustness.

Among all models, the number of false positives increases with the noise level, indicating a direct correlation between noise complexity and model susceptibility to potential noisy shades. Examining individual models, v10 exhibits the lowest overall false positive rate with a relatively moderate rise from background-only images to the most complex noise shape. This is due to the fact that v10 appears less prone to recommend detection than the other series as demonstrated by its low Recall score. v11 follows a similar trend but demonstrate slightly higher sensitivity in correspondence to increasing levels of noise. v9, in contrast, shows the largest error rates. This may be caused by a marked orientation of ninth series to optimistic in detecting objects. Concerning model's sizes, the most robust appears Small variant, whereas the worst Medium one. The Nano variant delivers an intermediate robustness between the previous, though some fluctuations. This is suggesting us that smaller models tend to be more conservative although at the bottom of the hierarchy the lack of generalization inverts slightly the trend.

We can observe that across noise conditions, background-only images yields the lowest error rates for all the models, demonstrating a solid robustness against empty frames with at most a 0.1 error rate. Dot noise introduces mild degradation, while tapered shapes and ant silhouettes significantly increase false detections. Despite the raise of errors, the rate is always strictly under 1 (at most 0.64 per image) error per image, revealing a valuable resistance to noise.

To further quantify the robustness of the models, we introduce a novel metric, the **Robustness Score** (RS). This metric is based on the **False Positive Rate** (FPR), which is computed as the ratio between the erroneous detections (false positives) to the total number of empty frames, evaluated across various confidence thresholds. The RS is derived from the area under the curve of FPR and confidence threshold curve, offering a standardized metric for evaluating model robustness. Specifically, the FPR at a given confidence threshold τ is defined as:

$$\mathrm{FPR}(\tau) = \frac{\text{FP at } \tau}{\text{Total Noisy Frames}},$$

where τ represents the confidence threshold. The RS is then calculated as:

$$\mathrm{RS} = \frac{1}{1 + \left(\int_0^1 \mathrm{FPR}(\tau)\, d\tau\right)}$$

This score provides a more comprehensive and standardized method for evaluating the model's robustness, where a higher RS indicates

Table 6
Robustness Scores for different models at various noise levels.

Noise level	v9			v10			v11		
	t	s	m	n	s	m	n	s	m
level **a**	0.882	0.944	0.953	0.970	**0.976**	0.968	0.954	0.951	0.952
level **b**	0.907	0.945	0.936	0.961	**0.973**	0.967	0.942	0.953	0.956
level **c**	0.731	0.904	0.920	0.909	0.944	**0.953**	0.719	0.896	0.834
level **d**	0.730	0.862	0.778	0.879	**0.907**	0.747	0.810	0.750	0.784

a superior ability to minimize false positives across different confidence thresholds, thus ensuring more reliable detection in a real-world setting. Table 6 reports the RS value for all the trained models.

The results presented in Table 6 confirm the trends previously observed. For all noise levels, v10 depicts the highest RS across the different model variants. This is evident at level **c** and level **d**, where v10's scores remain significantly higher compared to the other models. However, the RS is above 0.5 for every model suggesting a limited number of false positive regardless the value of confidence. In other words, since τ varies from 0.1 up to 1 with a step of 0.36, it suggests that number of false positive per image, i.e., FPR value, is low also in correspondence of particularly low value of confidence, becoming 0 increasing the value of τ.

In the next section, since this work is interested in developing a working prototype that performs reasonably well in semi-controlled conditions and lays the groundwork for future improvements, we put our attention to energy efficiency and hardware feasibility. In particular, we ponder on energy efficiency and hardware feasibility by comparing the performance and consumption trade-offs between RPi3B and RPi5 boards.

6. Embedded system performance

In this section, we evaluate the computational performance of the deployed models in terms of CPU usage, RAM consumption, and inference time (Section 6.1), followed by an attempt to improve efficiency through model quantization (Section 6.2).

6.1. Benchmarking

We now evaluate performance based on additional metrics that are highly dependent on the used hardware and the format into which the original YOLO model, initially developed using the `pytorch` library, was converted. Specifically, we compare four key metrics: average RAM usage, average CPU utilization, average power consumption, and average inference time. The power consumption has been detected by using the Fnirsi FNB58 USB-tester. These evaluations are conducted on two RPi devices: the RPi5 and RPi3B. We first assess all YOLO models and sizes in their original `PyTorch` format and then analyze the same metrics after converting the models into alternative formats, including `openvino`, `mnn`, `tflite`, and `ncnn`. In this test, we perform 250 image recognitions with some parallelization depending on the number of cores, namely, 4 on both the RPi3B and RPi5.

In Fig. 14 we show the system performances on RPi5 for different sizes on v11 and fixing `pytorch` format. So, we compare the most recent YOLO version we used in AntPi.

As expected, the RAM consumption is more in the larger models, where v11m requires on average more than 750 MB of memory, whereas v11n needs less than 650 MB of memory. Interestingly, the CPU usage on the larger models is the least, in which v11m exploits around 40% of the total CPU (i.e., in average less than 2 cores), while the smallest model, in our case the v11n, runs at the 50% of the total CPU. This counterintuitive behavior is actually correct, as already observed in other similar evaluations, such as in Feng et al. (2022). However, from a power perspective, all the v11 models perform at around 6 W of power, balancing the RAM and CPU utilization among the models. Finally, as expected, the inference time in v11m takes approximately 2.5 s, much more than those in v11n, which take around 0.35 s, whose speed is 7× faster. Fig. 14 also provides an indication on how much energy is needed to perform, in average, a single detection. In fact, v11m requires 15.42 J of energy, whereas v11n needs only 2.14 J.

In the next Fig. 15, we instead compare the same metrics on different YOLO versions and fixing the lightest size in the `pytorch` format, again on the RPi5. In other words, we compare v9t, v10n, and v11n. It is interesting to see that, overall, the RAM occupation, as well as the other key compared parameters are more or less the same, with some negligible fluctuation. Notably, one can experimentally observe that v11n is slightly faster than the others, but overall, all of them are able to perform a detection in less than half a second, which indicates a very good index of time-performance in an embedded system.

In both Figs. 14 and 15 we only evaluated the aforementioned key performance indicators on the most performing embedded device RPi5. In Fig. 16, we aim to show how the RPi3B behaves with respect to the RPi5. First of all, RAM occupation and CPU utilization are more or less comparable, but as expected RPi3B is less demanding than RPi5. In fact, we observe that RPi3B requires around 10–15% of resources less than those required by RPi5. Notice that we are comparing two very different devices in terms of hardware, though they belong on the same family of Raspberry's. The most interesting aspects will be now described. As expected, since RPi3B is very constrained in terms of onboard hardware, the power is much less than that of RPi5. Experimentally, RPi5 requires around 2.5× more power, and this impacts the energy consumption as well. However, the more power we have in RPi5 implies a tremendous difference in the inference time. Indeed, if RPi5 takes approximately 1 s, RPi3B is in general 8-9× slower. So, this implies that from one hand RPi3B can have a longer lifetime in terms of battery due to its smaller energy consumption, but at a cost of much slower inferences time. Obviously, we are comparing two devices with different "age", but in principle if one aims more on availability rather than performance, an RPi3B can be a viable solution.

Lastly, in Table 7 we assess the key performance by fixing `v11n` on both RPi5 and RPi3B, and also varying the format other than `pytorch`, by also including `openvino`, `mnn`, `tflite`, and `ncnn`.

First of all, the used format heavily impacts these key system performances. We experimentally observe that the YOLO internal scheduler tends to increase the parallel detection in formats like `openvino`, `tflite`, and `ncnn`. In fact, the average CPU utilization on these formats is approximately 80%, with peaks even 100% (in the case of `openvino` and `tflite`). This in turn significantly impacts on the power and energy consumption. Although the average power on these formats is around 7 W, we experiences peaks even up to 12.7 W on the RPi5. This aspect should not be overlooked. In fact, when using the official RPi5 wall charger, the system operated smoothly. However, with third-party chargers, such as smartphone adapters or power banks, we encountered voltage instability, causing the RPi5 to shut down abruptly during computation. Experimentally, these issues only arose during intensive parallel detections, suggesting that, under typical workloads, the RPi5 remains stable. Nonetheless, this highlights a potential deployment challenge: while the RPi5 offers the best overall performance, inadequate power delivery can lead to system disruptions. Therefore, Table 7 can be used as a **benchmark** for developers, farmers, or in general stakeholders interested in detecting ants on the edge. From the inference time point of view, `openvino` format is around 4× and 2× faster than the original `pytoch` in RPi5 and RPi3B, respectively.

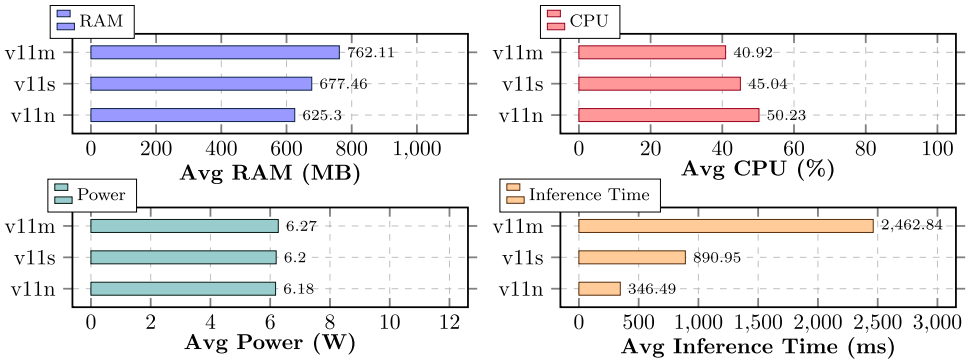


Fig. 14. System performance for different sizes on v11 and fixing `pytorch` format (RPi5).

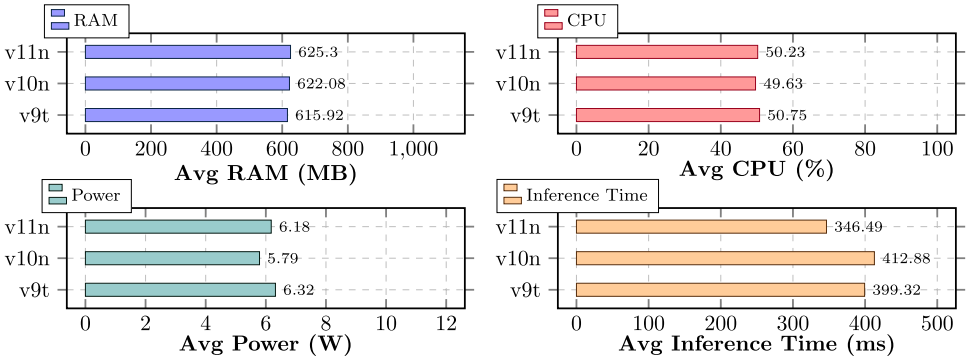


Fig. 15. System performance for different YOLO versions and fixing the smallest size and `pytorch` format (RPi5).

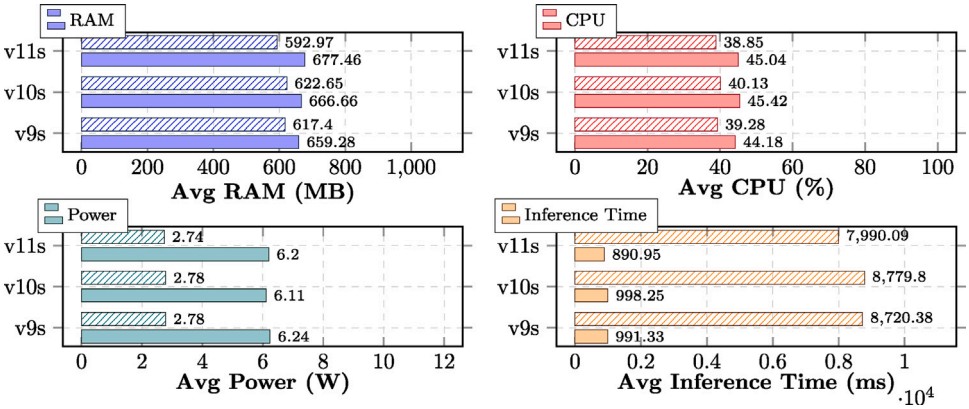


Fig. 16. System performance for different YOLO versions and fixing `small` size `pytorch` format (RPi5 solid bars, RPi3B bars with patterns).

Table 7
System performance fixing v11n (RPi5 and RPi3B).

Description	RPi 5					RPi 3B				
	openvino	mnn	tflite	ncnn	pytorch	openvino	mnn	tflite	ncnn	pytorch
Max RAM (GB)	866.6	**749.0**	1,389.6	767.2	798.0	**672.5**	694.3	719.8	693.6	713.4
Avg RAM (GB)	690.1	**589.8**	1,163.1	611.4	625.3	612.9	**583.1**	638.0	618.2	602.1
Max CPU usage (%)	92.5	**60.4**	100.0	85.7	62.2	100.0	**65.8**	100.0	94.4	97.4
Avg CPU usage (%)	80.5	**46.2**	80.5	77.0	50.2	78.8	**35.3**	70.1	48.4	44.9
Max power (W)	12.0	**8.7**	12.3	12.7	9.4	8.5	5.0	5.9	8.9	**4.8**
Avg power (W)	7.1	**6.2**	7.0	6.8	**6.2**	**2.8**	2.9	**2.8**	2.9	**2.8**
Max voltage (V)	5.2	5.2	5.2	5.2	5.2	5.2	5.2	5.2	5.2	5.2
Avg voltage (V)	5.1	5.1	5.1	5.1	5.1	5.1	5.1	5.1	5.1	5.1
Max current (A)	2.3	1.7	2.4	2.5	1.8	1.6	1.0	1.1	1.8	0.9
Avg current (A)	1.4	1.2	1.4	1.3	1.2	0.6	0.6	0.5	0.6	0.6
Avg Preproc. time (ms)	8.5	7.9	7.9	8.4	**6.9**	54.9	42.1	52.7	60.4	**34.2**
Avg Inference time (ms)	**79.4**	111.7	298.9	94.6	346.5	1,293.3	1,263.6	1,882.0	**1,146.6**	2,715.3
Avg Postproc. time (ms)	1.4	1.2	1.2	1.2	**1.0**	9.9	7.5	11.3	12.1	**5.2**
Frames per second (FPS)	**11.2**	8.3	3.3	9.6	2.8	0.7	**0.8**	0.5	0.8	0.4

Table 8
FPS performance of various models under different deployment formats.

Model	OpenVINO	MNN	TFLite
v9m	2.19	1.50	0.60
v9s	4.34	3.64	1.32
v9t	9.46	7.89	2.97
v10m	2.29	1.70	0.73
v10s	5.15	3.73	1.51
v10n	10.90	8.31	3.26
v11m	2.07	1.37	0.63
v11s	5.37	3.55	1.52
v11n	11.60	8.32	3.25

Even in the most constrained RPi3B, the v11n model in `openvino` format performs quite well, taking approximately 1 s for each detection, which can be considered more than enough. In RPi5, the fastest model to infer is instead `openvino`, with less than 0.1 s, while in RPi3B the fastest is `mnn`, with a little bit more than 1 s.

To conclude our analysis, we discuss the results reported in Table 8, which presents the frames-per-second (FPS) performance of various DL models deployed across three formats: OpenVINO, MNN, and TFLite. Each row corresponds to a specific model variant (e.g., v9m, v10n), while the columns report the FPS achieved under each deployment format.

As previously observed, OpenVINO consistently delivers the highest FPS across all models, with v11n achieving the peak performance at 11.60 FPS. MNN follows closely, often achieving between 70%–80% of OpenVINO's performance. TFLite exhibits the lowest FPS values across all configurations, typically around 30% of OpenVINO's performance. Within each group, the smallest models (i.e., v9t, v10n, v11n) run the fastest due to their reduced size. Conversely, the medium size models (i.e., the largest) perform with lowest FPS values.

These results suggest that for high-performance inference, OpenVINO is the preferred backend, especially when paired with the smallest model variants. Therefore, those models are suitable for video processing directly on the edge. Although MNN offers a reasonable trade-off between speed and platform flexibility, it is mostly suitable for image detection. This latter comment holds also for TFLite which appears the slowest in terms of processing time.

6.2. Quantization attempt

As we observed, the lifespan of RPi devices is influenced by data movement, specifically the amount of memory utilized for computation and the type of commands executed. Consequently, higher computational loads lead to a faster rate of energy depletion. To extend the device's lifespan, it is crucial to reduce resource usage and minimize prolonged periods of high computational stress. For this purpose, we investigate quantization (Wu et al., 2016) to reduce the RAM usage as well as CPU demand and inference time. Specifically, we evaluated the impact of quantization in two different precisions, **half-precision** (FP16) and **quantized precision** (INT8). The quantization is applied as post-processing, after the training with respect to **full-precision** (FP32). In other words, original weights are represented in FP32. To effectively apply it, we calibrated the output of each YOLO head leveraging a set of 252 representative images (42 occurrences per class). Notice that the YOLO series considered in this paper do not completely support quantization for RPi, and so we limit our analysis only to stable indicators.

Fig. 17 shows the mAP50 deterioration as the precision of neural network weights representation decreases. The x-axis reports the precision used to represents neural network weights (FP32, FP16, and INT8), whereas the y-axis denotes the corresponding mAP50 score. We labeled FP32 with the word "`org`" to remark that such a precision represents original weights without quantization.

In principle, the results highlight a progressive decline in mAP50 as the quantization level increases, with INT8 exhibiting the largest performance drop. From a model perspective, this behavior aligns with expectations, as lower-precision representations introduce quantization errors that accumulate across layers, ultimately affecting the final prediction.

Regarding precision, FP16 results in a moderate deterioration in mAP50 while offering a potential 2× reduction in memory usage. In contrast, INT8 experiences the most significant drop in performance. However, once full hardware support is ensured for the RPi, INT8 will provide a 4× reduction in memory usage. This behavior is consistent with the known limitations of aggressive bit-width reduction, particularly in high-dimensional feature networks such as YOLO.

Within the YOLO series, the impact of quantization appears to be less severe in newer architectures (v11) compared to earlier ones (v9). Additionally, model size plays a role in mitigating quantization effects. Larger models tend to experience less deterioration, suggesting that they may exhibit greater sparsity or that a larger proportion of neurons contribute less to the final prediction. As a result, weight precision compression has a more moderate impact on performance.

Although quantized models can offer reduced energy consumption, hardware incompatibility prevented us from conducting such experiments at the time of this research.

7. Discussion and limitations

Our results in Sections 5 and 6 demonstrate that AntPi is a viable edge-based solution for ant species detection using Raspberry Pi hardware and lightweight YOLO models. The integration of environmental sensors and GPS metadata improves the ecological value of collected data, while the cloud-based platform supports accessibility and collaboration among researchers and citizen scientists. The proposed robustness indicator introduces a novel way to assess model reliability under noisy inputs, which is essential for real-world deployments.

However, some limitations remain. First, the models were mainly trained on high-quality images, and their generalization to field-acquired data still needs to be validated. Second, although our benchmarks indicate feasible energy consumption under inference workloads, long-term deployments in harsh or isolated environments will require further hardware optimization and power management.

To address these limitations, we plan to deploy the AntPi system in real-world scenarios to collect new image data from varied habitats and lighting conditions. This data will be used to assess and, if necessary, retrain our models for improved robustness and generalization. We also aim to investigate DL model optimization techniques suited for Raspberry Pi, including quantization, pruning, knowledge distillation, and neural architecture search. In addition, we plan to explore converting our models to the ONNX format to improve inference efficiency and enhance portability across different edge platforms. This conversion may enable better runtime performance and help assess compatibility with constrained devices. Porting to more constrained platforms such as Arduino Nicla Vision is a further direction, which would broaden the range of viable deployments.

Finally, we envision extending AntPi to support a larger set of ant species and possibly other ecologically relevant insects (e.g., bees). A longer-term goal is the creation of a distributed wireless sensor network

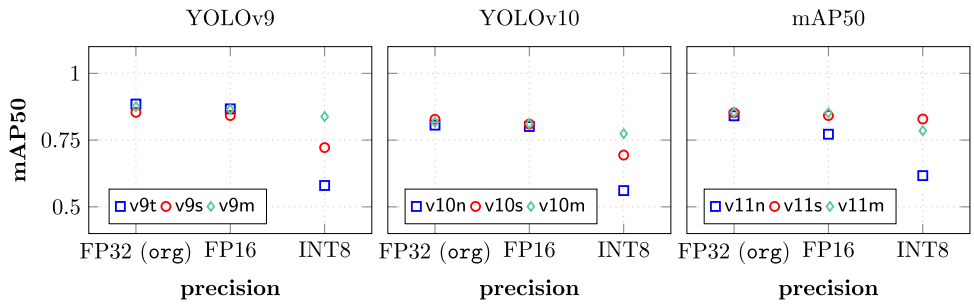


Fig. 17. Quantization performance reshaping based on mAP50.

of AntPi devices. Such a network would enable scalable deployment in areas lacking Wi-Fi by using multi-hop communication based on LPWAN technologies (e.g., LoRa), thereby increasing coverage and data resilience in fragmented or remote ecosystems.

Beyond these high-level considerations, we also analyzed specific limitations emerging from real-world experiments with images acquired using the Raspberry Pi camera in field conditions.

Despite only moderate variation across models, consistent patterns of misclassification were observed. Most missed detections were associated with high levels of blur that affected either the target ants or surrounding pixels. For example, motion-induced distortion rendered several ant instances unrecognizable, despite their distinctive shape and coloration. This issue reflects a bias in the training dataset toward high-clarity images. In future iterations, this will be mitigated by fine-tuning models using lower-quality and motion-blurred examples.

Another source of blur was an insufficient shutter speed relative to ant movement. This type of distortion is correctable through better camera calibration, and we expect improved shutter speed settings to substantially reduce this issue.

We also observed a spatial bias in model performance: ants located near the center of the image were more reliably detected, while those near the edges were often missed. This likely results from uneven spatial coverage in the training data and can be addressed through appropriate data augmentation strategies.

Additionally, clusters of ants — defined as more than three individuals in close proximity — proved challenging. Although rare, these dense scenarios were underrepresented in the training set and led to increased false negatives. Expanding the dataset to include such examples would improve model generalization to biologically realistic scenes.

Despite these limitations, the models remained robust against other common sources of failure, such as changes in object scale, background complexity, and color variation. The absence of systematic degradation in these dimensions suggests that the pretrained weights generalize well. Overall, the remaining weaknesses — namely blur sensitivity, spatial bias, and crowding effects — are actionable and likely to diminish through additional tuning and modest hardware improvements.

8. Conclusions

In this paper, we presented AntPi, a Raspberry Pi-based edge system for detecting multiple ant species using lightweight YOLO models. Our contributions are twofold: (i) the creation of the first dedicated dataset for ant species detection, and (ii) the development of a deployable, low-power system for real-time, automated monitoring in the field. We demonstrated that AntPi can serve as a practical and accessible alternative to manual sampling, requiring only minor maintenance and offering significant potential for long-term ecological monitoring. The integration of environmental sensors enhances the biological value of the captured data, enabling richer analysis of species behavior and habitat conditions. The inclusion of GPS metadata and a cloud-based platform supports collaborative research and citizen science initiatives. In addition, we introduced a novel robustness indicator to quantify the resilience of detection models under noisy conditions, addressing a critical challenge in real-world deployments.

CRediT authorship contribution statement

Lorenzo Palazzetti: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Daniele Giannetti:** Writing – original draft, Resources, Project administration, Investigation, Data curation, Conceptualization. **Antonio Verolino:** Resources, Data curation. **Donato A. Grasso:** Supervision, Conceptualization, Funding, Review & editing. **Cristina M. Pinotti:** Writing – review & editing, Supervision, Project administration, Conceptualization. **Francesco Betti Sorbelli:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to thank Mr. Lorenzo Fabbri for fabricating the 3D-printed enclosure used in this study. His contribution was essential to the development of a robust and waterproof prototype suitable for outdoor deployment.

Data availability

The data and code supporting the findings of this study are available through the following resources:

• **AntPi Client-Server Pipeline:** The complete Raspberry Pi-based data acquisition and gallery visualization system is available at:
https://github.com/bettisfr/ant-cloud
• **AntPi Quantization and Evaluation Toolkit:** This repository contains the full experimental pipeline for model evaluation, quantization, robustness analysis, and dataset processing:
https://github.com/TheAnswer96/ant-exp-ei
• **Ant Dataset Archive:** The annotated datasets used in this study are available at Zenodo:
https://doi.org/10.5281/zenodo.16740459
This archive includes:
– `dataset.zip` – The main dataset with 1,253 manually annotated images of six ant species (Crematogaster, Dolichoderus, Camponotus, Colobopsis, Plagiolepis, and Temnothorax), labeled in YOLO format using MakeSense.ai.
– `in-field.zip` – Additional real-world ant images captured using a Raspberry Pi 5 device during in-field deployments.
– `exp_false_positives.zip` – A curated set of synthetic images created to assess the robustness of ant detection models against out-of-distribution and adversarial scenarios.
To replicate the experiments or deploy the system:

• Extract images and YOLO labels into the corresponding folders under `src/images/` and `src/labels/`.
• Use the data preprocessing and experiment pipeline provided in the second repository to generate training/validation/test splits and execute quantization or evaluation routines.

References

Ahmad, A., Saraswat, D., El Gamal, A., 2023. A survey on using deep learning techniques for plant disease diagnosis and recommendations for development of appropriate tools. Smart Agric. Technol. 3, 100083. http://dx.doi.org/10.1016/j.atech.2022.100083.

Alhawsawi, A.N., Khan, S.D., Rehman, F.U., 2024. Enhanced YOLOv8-based model with context enrichment module for crowd counting in complex drone imagery. Remote. Sens. 16 (22), http://dx.doi.org/10.3390/rs16224175.

Ali, A.H., Youssef, A., Abdelal, M., Raja, M.A., 2024. An ensemble of deep learning architectures for accurate plant disease classification. Ecol. Informatics 81, 102618. http://dx.doi.org/10.1016/j.ecoinf.2024.102618.

Almstedt, L., Baltieri, D., Betti Sorbelli, F., Cattozzi, D., Giannetti, D., Kargar, A., Maistrello, L., Navarra, A., Niederprüm, D., O'Flynn, B., Palazzetti, L., Patelli, N., Piccinini, L., Pinotti, C.M., Wolf, L., Zorbas, D., 2023. Technological innovations in agriculture for scouting halyomorpha halys in orchards. In: 2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things. DCOSS-IoT, pp. 702–709. http://dx.doi.org/10.1109/DCOSS-IoT58021.2023.00110.

Almstedt, L., Betti Sorbelli, F., Boom, B., Calvini, R., Costi, E., Dinca, A., Ferrari, V., Giannetti, D., Ichim, L., Kargar, A., Lazar, C., Maistrello, L., Navarra, A., Niederprüm, D., Offermans, P., O'Flynn, B., Palazzetti, L., Patelli, N., Pinotti, C.M., Popescu, D., Rangarajan, A.K., Serghei, L., Ulrici, A., Wolf, L., Zorbas, D., Zurek, L., 2025a. Beyond the naked eye: Computer vision for detecting brown marmorated stink bug and its punctures. IEEE Trans. AgriFood Electron. 3 (1), 6–17. http://dx.doi.org/10.1109/TAFE.2024.3429537.

Almstedt, L., Betti Sorbelli, F., Boom, B., Calvini, R., Costi, E., Dinca, A., Ferrari, V., Giannetti, D., Ichim, L., Kargar, A., Lazar, C., Maistrello, L., Navarra, A., Niederprüm, D., Offermans, P., O'Flynn, B., Palazzetti, L., Patelli, N., Pinotti, C.M., Popescu, D., Rangarajan, A.K., Serghei, L., Ulrici, A., Wolf, L., Zorbas, D., Zurek, L., 2025b. A comprehensive pest monitoring system for brown marmorated stink bug. IEEE Trans. AgriFood Electron. 3 (1), 110–120. http://dx.doi.org/10.1109/TAFE.2024.3469538.

Bairi, A., Dulhare, U.N., 2024. Advanced cotton boll segmentation, detection, and counting using multi-level thresholding optimized with an anchor-free compact central attention network model. Eng 5 (4), 2839–2861. http://dx.doi.org/10.3390/eng5040148.

Betti Sorbelli, F., Palazzetti, L., Pinotti, C.M., 2023. YOLO-based detection of halyomorpha halys in orchards using RGB cameras and drones. Comput. Electron. Agric. 213, 108228. http://dx.doi.org/10.1016/j.compag.2023.108228.

Biswas, D., Tiwari, A., 2024. Utilizing computer vision and deep learning to detect and monitor insects in real time by analyzing camera trap images. Nat. Eng. Sci. 9 (2), 280–292. http://dx.doi.org/10.28978/nesciences.1575480.

Bjerge, K., Alison, J., Dyrmann, M., Frigaard, C.E., Mann, H.M.R., Høye, T.T., 2023. Accurate detection and identification of insects from camera trap images with deep learning. In: Fang, W.-T. (Ed.), PLOS Sustain. Transform. 2 (3), e0000051. http://dx.doi.org/10.1371/journal.pstr.0000051.

Bjerge, K., Karstoft, H., Høye, T.T., 2025. Towards edge processing of images from insect camera traps. Remote. Sens. Ecol. Conserv. http://dx.doi.org/10.1002/rse2.70007.

Bjerge, K., Karstoft, H., Mann, H.M., Høye, T.T., 2024. A deep learning pipeline for time-lapse camera monitoring of insects and their floral environments. Ecol. Informatics 84, 102861. http://dx.doi.org/10.1016/j.ecoinf.2024.102861.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent. In: Lechevallier, Y., Saporta, G. (Eds.), Proceedings of COMPSTAT'2010. Physica-Verlag HD, Heidelberg, pp. 177–186.

Castracani, C., Bulgarini, G., Giannetti, D., Spotti, F.A., Maistrello, L., Mori, A., Grasso, D.A., 2017. Predatory ability of the ant Crematogaster scutellaris on the brown marmorated stink bug Halyomorpha halys. J. Pest Sci. 90 (4), 1181–1190. http://dx.doi.org/10.1007/s10340-017-0889-1.

Cordonnier, M., Blight, O., Angulo, E., Courchamp, F., 2020. The native ant lasius niger can limit the access to resources of the invasive argentine ant. Animals 10 (12), 2451. http://dx.doi.org/10.3390/ani10122451.

Crupi, L., Butera, L., Ferrante, A., Giusti, A., Palossi, D., 2025. An efficient ground-aerial transportation system for pest control enabled by AI-based autonomous nano-UAVs. ACM J. Auton. Transp. Syst. http://dx.doi.org/10.1145/3719210.

Das, M., Liu, F.L.C., Hartle, C., Yang, C.-C.S., Chen, C., 2024. Ant detective: An automated approach for counting ants in densely populated images and gaining insight into ant foraging behavior. http://dx.doi.org/10.2139/ssrn.5123383, arXiv preprint arXiv:2410.20638.

DHT-sensor-library, 2025. Arduino library for DHT11, DHT22, etc. temperature & humidity sensors. https://github.com/adafruit/DHT-sensor-library. (Accessed 3 August 2025).

Dos Santos, A., Biesseck, B.J.G., Latte, N., de Lima Santos, I.C., dos Santos, W.P., Zanetti, R., Zanuncio, J.C., 2022. Remote detection and measurement of leaf-cutting ant nests using deep learning and an unmanned aerial vehicle. Comput. Electron. Agric. 198, 107071. http://dx.doi.org/10.1016/j.compag.2022.107071.

Feng, H., Mu, G., Zhong, S., Zhang, P., Yuan, T., 2022. Benchmark analysis of yolo performance on edge intelligence devices. Cryptography 6 (2), 16. http://dx.doi.org/10.3390/cryptography6020016.

Freitas, L., Martins, V., de Aguiar, M., de Brisolara, L., Ferreira, P., 2022. Deep learning embedded into smart traps for fruit insect pests detection. ACM Trans. Intell. Syst. Technol. 14 (1), 1–24. http://dx.doi.org/10.1145/3552435.

Gaigher, R., Samways, M.J., Jolliffe, K.G., Jolliffe, S., 2012. Precision control of an invasive ant on an ecologically sensitive tropical island: a principle with wide applicability. Ecol. Appl. 22 (5), 1405–1412. http://dx.doi.org/10.1890/11-2002.1.

Gao, Y., Xue, X., Qin, G., Li, K., Liu, J., Zhang, Y., Li, X., 2024. Application of machine learning in automatic image identification of insects - a review. Ecol. Informatics 80, 102539. http://dx.doi.org/10.1016/j.ecoinf.2024.102539, URL https://www.sciencedirect.com/science/article/pii/S1574954124000815.

García-Martínez, M.A., Presa-Parra, E., Valenzuela-González, J.E., Lasa, R., 2018. The fruit fly lure CeraTrap: An effective tool for the study of the arboreal ant fauna (hymenoptera: Formicidae). J. Insect Sci. 18 (4), http://dx.doi.org/10.1093/jisesa/iey078.

Gardiner, R., Rowands, S., Simmons, B.I., 2024. Towards scalable insect monitoring: Ultra-lightweight CNNs as on-device triggers for insect camera traps. arXiv preprint arXiv:2411.14467.

Gebauer, E., Thiele, S., Ouvrard, P., Sicard, A., Risse, B., 2024. Towards a dynamic vision sensor-based insect camera trap. In: 2024 IEEE/CVF Winter Conference on Applications of Computer Vision. WACV, pp. 7142–7151. http://dx.doi.org/10.1109/WACV57701.2024.00700.

Giannetti, D., Castracani, C., Spotti, F.A., Mori, A., Grasso, D.A., 2019. Gall-colonizing ants and their role as plant defenders: From 'bad job' to 'useful service'. Insects 10 (11), 392. http://dx.doi.org/10.3390/insects10110392.

Gouse, S., Dulhare, U.N., 2022. Automation of rice leaf diseases prediction using deep learning hybrid model VVIR. In: Rajagopal, S., Faruki, P., Popat, K. (Eds.), Advancements in Smart Computing and Information Security. Springer Nature Switzerland, Cham, pp. 133–143. http://dx.doi.org/10.1007/978-3-031-23092-9_11.

Grasso, D.A., Pandolfi, C., Bazihizina, N., Nocentini, D., Nepi, M., Mancuso, S., 2015. Extrafloral-nectar-based partner manipulation in plant–ant relationships. AoB PLANTS 7, http://dx.doi.org/10.1093/aobpla/plv002.

Hu, J., Bogar, T.A., Gu, Y.F., Guénard, B., 2024. Ant-observer: A new approach for automatic acquisition and autonomous analyses of individual species abundance and interactions. Ecol. Informatics 82, 102752. http://dx.doi.org/10.1016/j.ecoinf.2024.102752.

IDOR STORE, 2025. IDOR STORE solar panel. https://www.amazon.it/Caricabatteria-Portatile-Cellulare-Telecamera-Fotovoltaico/dp/B0D3RS5F6P. (Accessed 24 February 2025).

iNaturalist, 2025. iNaturalist — inaturalist.org. https://www.inaturalist.org. (Accessed 5 March 2025).

Kargar, A., Zorbas, D., Tedesco, S., Gaffney, M., O'Flynn, B., 2024. Detecting halyomorpha halys using a low-power edge-based monitoring system. Comput. Electron. Agric. 221, 108935. http://dx.doi.org/10.1016/j.compag.2024.108935.

Keasar, T., Yair, M., Gottlieb, D., Cabra-Leykin, L., Keasar, C., 2024. STARdbi: A pipeline and database for insect monitoring based on automated image analysis. Ecol. Informatics 80, 102521. http://dx.doi.org/10.1016/j.ecoinf.2024.102521, URL https://www.sciencedirect.com/science/article/pii/S1574954124000633.

Khan, A.J., Bakr, M.A., Khan, S.D., 2024. Vegetation detection using UAV imagery with deep learning segmentation models. In: 2024 19th International Conference on Emerging Technologies. ICET, pp. 1–6. http://dx.doi.org/10.1109/ICET63392.2024.10935177.

Khanam, R., Hussain, M., 2024. YOLOv11: An overview of the key architectural enhancements. arXiv:2410.17725.

Kumar, A.V.S., M., A., Dutta, A., Ray, S., Rahman, H., Masadeh, S.R., Musirin, I.B., L., M.R., V., S.R., Malladi, R., Dulhare, U.N., 2024. Advanced Computational Methods for Agri-Business Sustainability. IGI Global, p. 16. http://dx.doi.org/10.4018/979-8-3693-3583-3.ch004,

Lach, L., Parr, C., Abbott, K., 2009. Ant Ecology. Oxford University Press, http://dx.doi.org/10.1093/acprof:oso/9780199544639.001.0001.

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

McCravy, K.W., 2018. A review of sampling and monitoring methods for beneficial arthropods in agroecosystems. Insects 9 (4), http://dx.doi.org/10.3390/insects9040170.

Monsimet, J., Sjögersten, S., Sanders, N.J., Jonsson, M., Olofsson, J., Siewert, M., 2024. UAV data and deep learning: efficient tools to map ant mounds and their ecological impact. Remote. Sens. Ecol. Conserv. n/a (n/a), http://dx.doi.org/10.1002/rse2.400.

Montgomery, G.A., Belitz, M.W., Guralnick, R.P., Tingley, M.W., 2021. Standards and best practices for monitoring and benchmarking insects. Front. Ecol. Evol. Volume 8 - 2020, http://dx.doi.org/10.3389/fevo.2020.579193.

Nyamukondiwa, C., Addison, P., 2014. Food preference and foraging activity of ants: Recommendations for field applications of low-toxicity baits. J. Insect Sci. 14 (48), 1–13. http://dx.doi.org/10.1673/031.014.48.

Open-smart-GT-7U-GPS, 2025. How to use open-smart GPS GT-7U GPS module with arduino. https://github.com/jumejume1/Open-smart-GT-7U-GPS. (Accessed 3 August 2025).

Palazzetti, L., Rangarajan, A.K., Dinca, A., Boom, B., Popescu, D., Offermans, P., Pinotti, C.M., 2024. The hawk eye scan: Halyomorpha halys detection relying on aerial tele photos and neural networks. Comput. Electron. Agric. 226, 109365. http://dx.doi.org/10.1016/j.compag.2024.109365.

Parker, J., Kronauer, D.J., 2021. How ants shape biodiversity. Curr. Biology 31 (19), R1208–R1214. http://dx.doi.org/10.1016/j.cub.2021.08.015.

Rahardjo, B.T., Muhammad, F.N., Setiawan, Y., Febryadi, A., Ihsan, M., Wibowo, D., Fernando, I., 2023. Ant preference for different types of bait at sugarcane plantations in East Java, Indonesia. Biodiversitas J. Biological Divers. 24 (4), http://dx.doi.org/10.13057/biodiv/d240420.

Ramalingam, B., Mohan, R.E., Pookkuttath, S., Gómez, B.F., Sairam Borusu, C.S.C., Wee Teng, T., Tamilselvam, Y.K., 2020. Remote insects trap monitoring system using deep learning framework and IoT. Sensors 20 (18), http://dx.doi.org/10.3390/s20185280.

Rico-Gray, V., Oliveira, P., 2009. Ant Ecology. Oxford University Press, http://dx.doi.org/10.1093/acprof:oso/9780199544639.001.0001.

Saradopoulos, I., Potamitis, I., Ntalampiras, S., Konstantaras, A.I., Antonidakis, E.N., 2022. Edge computing for vision-based, urban-insects traps in the context of smart cities. Sensors 22 (5), http://dx.doi.org/10.3390/s22052006.

Schifani, E., Castracani, C., Giannetti, D., Spotti, F.A., Reggiani, R., Leonardi, S., Mori, A., Grasso, D.A., 2020. New tools for conservation biological control: Testing ant-attracting artificial nectaries to employ ants as plant defenders. Insects 11 (2), 129. http://dx.doi.org/10.3390/insects11020129.

Schifani, E., Giannetti, D., Grasso, D.A., 2024. Toward sustainable management of ant-hemipteran mutualism in agricultural settings: a comparison of different approaches. Crop. Prot. 175, 106468. http://dx.doi.org/10.1016/j.cropro.2023.106468.

Sittinger, M., Uhler, J., Pink, M., Herz, A., 2024. Insect detect: An open-source DIY camera trap for automated insect monitoring. In: Mansour, R. (Ed.), PLOS ONE 19 (4), e0295474. http://dx.doi.org/10.1371/journal.pone.0295474.

Skalski, P., 2019. Make sense. https://github.com/SkalskiP/make-sense/.

Song, Y., Chen, M., Wu, J., Hong, J., Ouyang, T., Liang, Y., Liang, M., Lu, Y., 2024. Impact on ant communities by chemical pesticides applied in controlling the red imported fire ant (Solenopsis invicta Buren) in the field. Insects 15 (11), 876. http://dx.doi.org/10.3390/insects15110876.

Tabani, H., Balasubramaniam, A., Arani, E., Zonooz, B., 2021. Challenges and obstacles towards deploying deep learning models on mobile devices. arXiv preprint arXiv:2105.02613.

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., Ding, G., 2024a. YOLOv10: Real-time end-to-end object detection. arXiv preprint arXiv:2405.14458.

Wang, R., Kass, J.M., Chaudhary, C., Economo, E.P., Guénard, B., 2024b. Global biogeographic regions for ants have complex relationships with those for plants and tetrapods. Nat. Commun. 15 (1), 5641. http://dx.doi.org/10.1038/s41467-024-49918-2.

Wang, C.Y., Yeh, I.H., Mark Liao, H.Y., 2024c. Yolov9: Learning what you want to learn using programmable gradient information. In: European Conference on Computer Vision. Springer, pp. 1–21. http://dx.doi.org/10.1007/978-3-031-72751-1_1.

Warren, R.J., Frankson, P.T., Mohan, J.E., 2023. Global change drivers synergize with the negative impacts of non-native invasive ants on native seed-dispersing ants. Biol. Invasions 25 (3), 773–786.

Wu, M., Cao, X., Guo, S., 2020. Accurate detection and tracking of ants in indoor and outdoor environments. http://dx.doi.org/10.1101/2020.11.30.403816, BioRxiv, Cold Spring Harbor Laboratory.

Wu, J., Leng, C., Wang, Y., Hu, Q., Cheng, J., 2016. Quantized convolutional neural networks for mobile devices. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 4820–4828. http://dx.doi.org/10.1109/CVPR.2016.521.